Evolvable Hardware Platform for Fault-Tolerant Reconfigurable Sensor Electronics

Evangelos F. Stefatos

A thesis submitted for the degree of Doctor of Philosophy
The University of Edinburgh
May 2007

Declaration of originality

I hereby declare that the research recoded in this thesis and the thesis itself was composed and originated entirely by myself in the School of Engineering and Electronics at the University of Edinburgh.

Evangelos Fotiou Stefatos 2 May 2007

Meaningful Outcome

Ιθάκη

Σα βγεις στον πηγαιμό για την Ιθάκη,
να εύχεσαι νάναι μακρύς ο δρόμος,
γεμάτος περιπέτειες, γεμάτος γνώσεις.
Τους Λαιστρυγόνας και τους Κύκλωπας,
τον θυμωμένο Ποσειδώνα μη φοβάσαι,
τέτοια στον δρόμο σου ποτέ σου δεν θα βρείς,
αν μέν' η σκέψις σου υψηλή, αν εκλεκτή
συγκίνησις το πνεύμα και το σώμα σου αγγίζει.
Τους Λαιστρυγόνας και τους Κύκλωπας,
τον άγριο Ποσειδώνα δεν θα συναντήσεις,
αν δεν τους κουβανείς μες στην ψυχή σου,
αν η ψυχή σου δεν τους στήνει εμπρός σου.

Να εύχεσαι νάναι μακρύς ο δρόμος.
Πολλά τα καλοκαιρινά πρωϊά να είναι
που με τι ευχαρίστησι, με τι χαρά
θα μπαίνεις σε λιμένας πρωτοειδωμένους·
να σταματήσεις σ' εμπορεία Φοινικικά,
και τες καλές πραγμάτειες ν' αποκτήσεις,
σεντέφια και κοράλλια, κεχριμπάρια κ' έβενους,
και ηδονικά μυρωδικά κάθε λογής,
όσο μπορείς πιο άφθονα ηδονικά μυρωδικά·
σε πόλεις Αιγυπτιακές πολλές να πας,
να μάθεις και να μάθεις απ' τους σπουδασμένους.

Πάντα στον νου σου νάχεις την Ιθάκη.
Το φθάσιμον εκεί είν' ο προορισμός σου.
Αλλά μη βιάζεις το ταξίδι διόλου.
Καλλίτερα χρόνια πολλά να διαρκέσει·
και γέρος πια ν' αράξεις στο νησί,
πλούσιος με όσα κέρδισες στον δρόμο,
μη προσδοκώντας πλούτη να σε δώσει η Ιθάκη.

Η Ιθάκη σ' έδωσε το ωραίο ταξίδι.
Χωρίς αυτήν δεν θάβγαινες στον δρόμο.
Αλλο δεν έχει να σε δώσει πια.

Κι αν πτωχική την βρεις, η Ιθάκη δεν σε γέλασε.
Ετσι σοφός που έγινες, με τόση πείρα,
ήδη θα το κατάλαβες η Ιθάκες τι σημαίνουν.

Κωνσταντίνος Π. Καβάφης (1911)

Ithaca

When you set out on your journey to Ithaca,
pray that the road is long,
full of adventure, full of knowledge.
The Lestrygonians and the Cyclops,
the angry Poseidon -- do not fear them:
You will never find such as these on your path,
if your thoughts remain lofty, if a fine
emotion touches your spirit and your body.
The Lestrygonians and the Cyclops,
the fierce Poseidon you will never encounter,
if you do not carry them within your soul,
if your soul does not set them up before you.

Pray that the road is long.
That the summer mornings are many, when,
with such pleasure, with such joy
you will enter ports seen for the first time;
stop at Phoenician markets,
and purchase fine merchandise,
mother-of-pearl and coral, amber and ebony,
and sensual perfumes of all kinds,
as many sensual perfumes as you can;
visit many Egyptian cities,
to learn and learn from scholars.

Always keep Ithaca in your mind.
To arrive there is your ultimate goal.
But do not hurry the voyage at all.
It is better to let it last for many years;
and to anchor at the island when you are old,
rich with all you have gained on the way,
not expecting that Ithaca will offer you riches.

Ithaca has given you the beautiful voyage.
Without her you would have never set out on the road.
She has nothing more to give you.

And if you find her poor, Ithaca has not deceived you.
Wise as you have become, with so much experience,
you must already have understood what Ithacas mean.

Constantine P. Cavafy (1911)

Acknowledgements

For the completion of my PhD thesis I would like to heartily thank my beloved *Elina Karamichou*. Besides myself, there is no other person that has endeavoured more for the successful completion of this work. Every day, for more than 3 years, she made my life easier and more beautiful. She was the one who patiently endured my frustration and bad-temper during the difficult moments of my research, and I always had someone to love me and help eliminate any problems apart from those of my research. For all the above reasons, I feel that there are no words enough to thank *Elina*. I cannot dedicate my thesis to her, simply because a significant part of it belongs to her.

Most importantly, I would like to thank my parents (*Foti and Christina*). My strong will to make them proud was the original motivation to start my PhD. The outcome of this research is exclusively dedicated to them, as a small payback for everything they have done for me during my 29 years of life, and as forgiveness for every time I have depressed them. Without the values I have adopted from them, I would have never achieved anything in my life.

In addition, I would like to thank my supervisor, *Prof. Tughrul Arslan*, who trusted me as a person and scientist and gave me the opportunity to implement a PhD in a very interesting and challenging field. We spent three very interesting years together and I have to say that our collaboration was very close and, above all, productive. I want to thank him for his excellent supervision and for the chance he gave me to travel a lot and present my work at outstanding high-quality scientific conferences, to meet people of important scientific background and gain a lot of invaluable experience. For all these reasons, I feel that it would be a great pleasure for me to collaborate with *Prof. Tughrul* sometime in the future.

Moreover, I would like to acknowledge my second supervisor, *Dr. Alister Hamilton*, because he never refused to give me his prompt assistance and share with me his widespread knowledge, every time I asked for these.

I would also like to acknowledge *Dr. Robert (Bob) Thomson* for his invaluable guidance during the early stages of my PhD and *Mrs. Katherine Cameron* for volunteering to proof-read my thesis.

I would also like to thank *all my colleagues* in our group because their technical expertise and professionalism was the best motivation to me and pushed me into working harder every day.

Last but not least, I would like to thank *Mrs. Liz Glass* for her kind smile every time I asked her assistance. This is a gift of life, it is not a profession.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

A/S Unit	Addition/Subtraction Unit
ADC	Analogue-to-Digital Converter
AGC	Automatic Gain Control
ASIC	Application Specific Integrated Circuit
CALB	Configurable-Arithmetic-Logic-Block
CALU	Configurable-Arithmetic-Logic-Unit
CPL	Computational Layer
CSD	Canonical-Signed-Digit
CSE	Common Sub-expression Elimination
CTL	Control Layer
DAC	Digital-to-Analogue Converter
DFG	Data Flow Graph
DSP	Digital Signal Processing
EA	Evolutionary Algorithm
EHW	Evolvable Hardware
FFT	Fast-Fourier-Transform
FIR	Finite Impulse Response
FOG	Fiber-Optic-Gyroscope
FPGA	Field Programmable Gate Array
FSM	Finite-State-Machine
GA	Genetic Algorithm
GPS	Global Positioning System
IIR	Infinite Impulse Response

JPL	Jet Propulsion Laboratory
MAG	Minimum-Adder-Graph
MCM	Multiple Constant Multiplication
MEMS	Micro-Electro-Mechanical Systems
MO	Multi-Objective
MOEA	Multi-Objective Evolutionary Algorithm
PALU	Programmable-Arithmetic-Logic-Unit
PE	Processing Element
PGA	Parallel Genetic Algorithm
PID	Proportional Integral Derivative
PLD	Programmable Logic Device
POF	Primitive Operation Filtering
R/L Shifter	Right/Left Shifter
R4SDC	Radix-4 Single Delay Commutator
RAG-n	Reduced-Adder-Graph
RLG	Ring-Laser-Gyroscope
SCM	Single Constant Multiplication
SDF	Standard-Delay-Format
SEU	Single Event Upset
SHE	Single Hard Error
SoC	System-on-Chip
TI	Texas Instruments
TMR	Triple Modular Redundancy
UCLA	University of California, Los Angeles
VLSI	Very Large Scale Integration

Nomenclature

$\|F(jw)\|^2$	Amplitude Spectral Density
$c(t)$	Actual process variable measurement
Deg/hr	Degrees per hour
$e(t)$	Error signal
$e_D[j]$	Digitalized input
f_p	Pass-band edge frequency
F_s	Sampling frequency
f_s	Stop-band edge frequency
$h(k)$	Impulse response
$H(z)$	Transfer Function
$Imag_{in}$	Imaginary part of the input data samples
$Imag_{out}$	Imaginary part of the system's output
K_d	Derivative coefficient
K_i	Integral coefficient
K_p	Proportional coefficient
$r(t)$	Reference signal of set-point
$Real_{in}$	Real part of the input data samples
$Real_{out}$	Real part of the system's output
T_c	Sampling period
T_d	Derivative time
T_i	Integral time
$u(n)$	Impulse unit

$x\ (n)$	Present values of the input
$X\ (t)$	Input data in time domain
$X\ (z)$	Input data in frequency domain
$y\ (n)$	Current output sample
$Y\ (t)$	System's output in frequency domain
$Y\ (z)$	System's output in frequency domain
β	Elevation Angle
δ_p	Peak pass-band deviation
δ_s	Stop-band deviation
φ	Azimuth Angle

Abstract

The advent of System-on-Chip technology and the continuous shrinkage in silicon device feature size are dictating the need for realization of miniaturized integrated robust and autonomous systems. This need is especially evident in systems operating within hostile environments, such as aerospace.

Evolvable Hardware (EHW) is a technology, which shows promise in meeting the needs of systems facing malfunctions due to harsh electronics environments. Key features of EHW are a reconfigurable fabric and an evolutionary strategy. The design of a complete and efficient EHW framework must consider both these features concurrently. This comprehensive approach to the design of EHW based systems is an issue considered by only a few researchers in the literature.

This thesis presents a novel holistic EHW framework that accomplishes all the electronics associated with the JPL/Boeing gyroscope sensor. It includes an efficient fault-tolerant reconfigurable fabric and an integrated on-chip multi-objective evolutionary strategy. The conception and implementation of both parts also consider real-time adaptation and low-power consumption for enabling ultra-long life aerospace missions.

A number of key objectives have been achieved: a) The Verilog implementation of an autonomous reconfigurable fabric that is capable of accomplishing the sensor's electronics with substantial accuracy (>99.7%), b) The implementation of numerous evolutionary strategies that are able to primarily guide the hardware evolution even in the presence of 30% faults injected in the user and configuration memory of the system, c) This in addition to a reduction from 8.6 to 9.8 times in the number of generations that are needed for the evolution of a 31-tap FIR filter, compared with previous research in this field, d) Furthermore, the circuits evolved consume 3.3 times less power than similar implementations within industrial reconfigurable devices.

Chapter 1 Introduction

1.1 Introduction

Evolvable Hardware (EHW) is a very promising field at the confluence of *reconfigurable hardware*, *automatic design*, *artificial intelligence* and *autonomous systems*. In a narrow sense, EHW is reconfigurable hardware whose on-chip configuration is under the control of an Evolutionary Algorithm (EA) [32], [42]. EAs provide genetic search/optimization algorithms, which work on a population (multiple chromosomes usually expressed with binary strings) of prospective solutions and apply operations motivated by natural selection to provide better solutions to multi-objective problems.

The main objective of EHW is to automate the design of mixed-analog electronic circuits that are capable to adapt/reconfigure their hardware structure [39], [50] over time, so as to achieve an optimal configuration that meets better their functional specifications. Since the emergence of EHW, evolutionary reconfigurable systems have successfully been employed for numerous applications that demand adaptive reconfiguration, such as in fault-recovery systems [92], [100], automated circuit synthesis and design [12], [56], [57], [123], dynamical control of MEMS circuits, evolutionary Global-Positioning-System (GPS) attitude determination of vehicles [119], automated antenna design [66], etc.

The underlying theme of this thesis has been to investigate a number of custom reconfigurable architectures for the autonomous design and reconfiguration of digital high-order FIR filters using the principle of EHW. Each reconfigurable architecture is therefore configured using a Genetic Algorithm (GA) that derives from a class of non-heuristic search and optimization techniques termed EAs, which are inspired by the process of biological evolution.

The outcomes of this research work are evaluated using a real-life application such as the vibratory gyroscope, which has been fabricated by Jet-Propulsion-Laboratory (JPL) and Boeing. Due to the nature of the applications that this sensor targets, the overall design policy of this architecture must be guided by the fact that nowadays ultra-long life space missions require hardware to remain operational within specified performance parameters for decades. Therefore, power consumption is a factor that has a key role in the design of the reconfigurable platforms employed to accommodate the sensor's electronic circuits. Moreover, the reconfigurable hardware fabrics must be able to cope with anticipated faults that occur mainly due to radiation. The goal of this research is very challenging, considering that micro-machined gyroscopes for measuring rotational rates have attracted a lot of attention during the past few years for several applications involving space conquest, military missions, biomedical activities, etc.

This thesis focuses on investigating and implementing a *low-power autonomously reconfigurable EHW architecture* that aims at controlling the core electronics of the JPL/Boeing gyroscope against several unstable factors like aging of electronic devices, fault occurrence and temperature variations.

The novelty in this research lies on the implementation of an efficient Very-Large-Scale-Integration (VLSI) EHW framework including a *low-power* and *fault-tolerant reconfigurable hardware* substrate and a customized *reconfiguration mechanism* able to efficiently guide the implementation and/or reconfiguration process of the JPL/Boeing gyroscope's electronics, in terms of *accuracy* and *speed of adaptation*. The *reconfigurable hardware* must be able to accomplish and maintain the gyroscope's functionality in the presence of endogenous and exogenous factors, such as anticipated faults and environmental variations. The recovery method should not be based on hardware redundancy because this method is impractical, due to restrictive size and weight requirements imposed on spacecraft. On the contrary, the fault-tolerant method is based on the reconfiguration of fine-grained configurable components, which comprise the *reconfigurable hardware*. This approach differentiates from commercial architectures and presents better performance in terms of silicon occupation, power dissipation and efficiency in adaptation.

1.2 Thesis Achievements

As has been mentioned in the introduction of this thesis the main scope of this research is to invent and implement an EHW framework that is able to accomplish the electronics associated with the JPL/Boeing gyroscope's electronic circuits and satisfy the requirements imposed by the nature of the targeted application. The final achievements of this research work can be summarized below:

- The development of a low-power and fault-tolerant EHW framework that is able to accommodate the JPL/Boeing gyroscope's electronic circuits and provide to these real-time adaptation/reconfiguration in the presence of endogenous and/or exogenous factors.

- The development of four novel reconfigurable hardware substrates, which target the implementation of the sensor's electronics. The temporal order in the demonstration of these designs within the thesis reveals the problems, which have been emerged, during the implementation of the targeted circuits and the actions, which have been taken in order to work these problems out.

- The development of a fault-tolerant reconfiguration mechanism. It is a novel implementation of a parallel fine-grained GA that aims at providing a fault-tolerant solution to applications that require autonomous dynamic reconfiguration mechanisms.

- The development of three novel reconfiguration mechanisms, which in collaboration with a novel reconfigurable hardware substrate target to further improve the implementation of the sensor's electronics in terms of accuracy, power-consumption and real-time adaptation.

1.3 Thesis Outline

This thesis therefore focuses on the implementation of an efficient EHW framework including a low-power and fault-tolerant reconfigurable hardware substrate and a customized reconfiguration mechanism able to efficiently guide the implementation and/or reconfiguration process of the JPL/Boeing gyroscope's electronics, in terms of accuracy and speed for adaptation. This thesis is organized as follows:

- Chapter 2 introduces the JPL/Boeing gyroscope and provides an overview of its structure and operation principles. Moreover, it analyses the electronic circuits that implement the control-loops of the gyroscope and presents previous work that has been done concerning the gyroscope itself and possible architectures that could be used to efficiently accomplish the sensor's electronics.

- Chapter 3 introduces the concept of GAs and describes the methodology that these are applied on EHW in general and particularly in this thesis in order to evolve the electronics associated with the JPL/Boeing gyroscope's control-loops.

- Chapter 4 introduces the concept of a stand-alone architecture that consists of a reconfigurable hardware and a reconfiguration mechanism. It mainly presents the implementation of a novel fault-tolerant reconfiguration mechanism. The architecture is based on a parallel fine-grained GA. The overall idea is a System-on-Chip fault-tolerant implementation that would be able to compensate for faults occurring either in the reconfigurable hardware that accommodate the sensor's electronics or the reconfiguration mechanism that guide the evolution of the digital circuits.

- Chapter 5 composes the main part of this thesis and presents the continuous development of several reconfigurable hardware substrates in order to reach the main objective of this thesis that is to evolve the sensor's electronics under certain requirements.

- Chapter 6 is mainly concentrated on comparing conventional reconfiguration mechanisms with three proposed evolutionary strategies that have not been published before. In this Chapter the best found reconfigurable hardware substrate is employed in order to evaluate the evolutionary strategies.
- Chapter 7 employs the most successful combination of reconfigurable hardware substrate and reconfiguration mechanism to evolve the electronic circuits related with the control-loops of the sensor. Total power calculation is also provided for the majority of the associated electronics.
- Chapter 8 summarizes the conclusions obtained from each of the previous chapters. Furthermore, improvements are suggested concerning the research work that is presented in this thesis in order to further extent the gained knowledge.

1.4 Summary

This research work aims at investigating several architectural solutions for identifying the most appropriate physical mean for the autonomous design of high-order FIR filters. Moreover, these filters must incorporate several techniques such as primitive operations, multi-objective optimization and effective approaches for minimizing the search space of the employed EAs in order to meet the criteria imposed by the electronics associated with the JPL/Boeing vibratory sensor. The following chapters analyze the nature of the targeted application, the specification of electronics that have to be implemented and presents several EHW architectures that target to successfully achieve the ideal functionality and simultaneously meet the targeted performance in terms of power consumption, speed for adaptation, throughput and fault-robustness when stuck-at faults are injected in the *user* and *configuration* memory of the system.

Chapter 2
JPL/Boeing Gyroscope

2.1 Introduction

A gyroscope is a commonly used sensor for measuring angular velocity. Although conventional rotating-wheel, fiber-optic and ring-laser gyroscopes are dominant in many applications, their size, power inefficient mechanism and cost limit their utilization in a wider range of industries such as automobile, satellite, video game and handheld positioning systems. However, with the advent of MEMS machining technologies, micro-machined gyroscopes for measuring rate or angle of rotation have attracted a lot of attention during the past few years, for several applications. They can be used either as a low-cost miniature companion with micro-machined accelerometers to provide heading information for inertial navigation purposes or in other areas including automotive applications for ride stabilization and rollover detection, biomedical applications, in consumer applications (video-camera stabilization, virtual reality, etc.), in robotics and in a wide range of military applications.

JPL/Boeing MEMS gyroscopes aim at providing future Space missions with a low-cost, reduced mass and low-power consumption inertial measurement unit [11]. Typical applications that would benefit from this technology include: inertial spacecraft navigation to complement a star tracker or sun sensor, integration of the device in a planetary rover and detection of angular rotation in all axes of a robotic arm. This sensor presents superiority over conventional designs (spinning-mass) because it is typically orders of magnitude smaller and consumes less power. Moreover, optical gyroscopes such as Fiber-Optic-Gyros (FOGs) [122] and Ring-Laser-Gyros (RLGs) [8], [107] are also large, power consuming and expensive. In addition to this, FOGs suffer from performance degradation when they are left exposed in radiation due to optical fiber darkening [31], while RLGs suffer from laser life issues [63]. Moreover, while there have been several MEMS gyroscopes [13], [14], [54], [121] reported in the literature, the performance of these devices for spaceflight has been inadequate due to large bias stabilities (>1 deg/hr) resulting in non zero-rate drift. On the contrary, the JPL/Boeing gyroscope presents a vastly improved bias stability of 0.1 deg/hr [30], [75].

2.2 Structure and Operation Overview

The JPL/Boeing gyroscope is a vibratory rate sensor whose operation depends on the Coriolis coupling [80] of one degree of freedom to another degree of freedom within the sensor. Figure 2.1 depicts an electron microscope photograph of the JPL/Boeing gyroscope [20]. There are four paddles labeled D_1, D_2, S_1 and S_2 that compose the *drive* and *sense* rocking modes, respectively. These paddles are suspended above four electrodes by the thin silicon springs, which are evident in Figure 2.1, between the paddles. Paddles D_1 and D_2 compose the drive axis inputs. Applying a potential between the electrodes beneath D_1 and D_2 causes the paddles to pull closer to the electrodes due to electrostatic forces. Consequently, these forces rock the assembly about the y-axis, shown in Figure 2.1. The large post, in the center, adds inertia to the rocking modes and aids in coupling the degrees of freedom. Excitation at the drive axis natural frequency is desirable, since a large response is obtained, which boosts the sensitivity of the sensor. Furthermore, angular rotation of the frame about the z-axis induces post-rocking, due to Coriolis Effect, about the x-axis. The x-axis is called the *sense axis* and the rocking velocity about this is measured by capacitive sensors at paddles S_1 and S_2. These measurements are related to the angular rate of rotation of the frame. In an ideal device, both the *sense* and *drive* rocking modes have equal frequencies and the nodal axes coincide with the x-axis and y-axis in the sensor frame. However, fabrication irregularities may cause a split between the rocking mode frequencies as well as a change in orientation of the nodal axes with respect to the electrodes. Moreover, the *drive* and *sense* axis frequencies are sensitive to temperature variations. Therefore, the JPL/Boeing gyroscope, as any other vibratory sensor, needs the implementation of numerous control-loops to improve its bandwidth, linearity, dynamic range and to maintain performance in the presence of perturbations to the sensor dynamics.

The electronics that compose the control-loops of the gyroscope are presented in the next section, while further details on the design, identification of the sensor's dynamics and operation principles of the sensor can be found in [71], [97], [98], [99], since these are beyond the scope of this research.

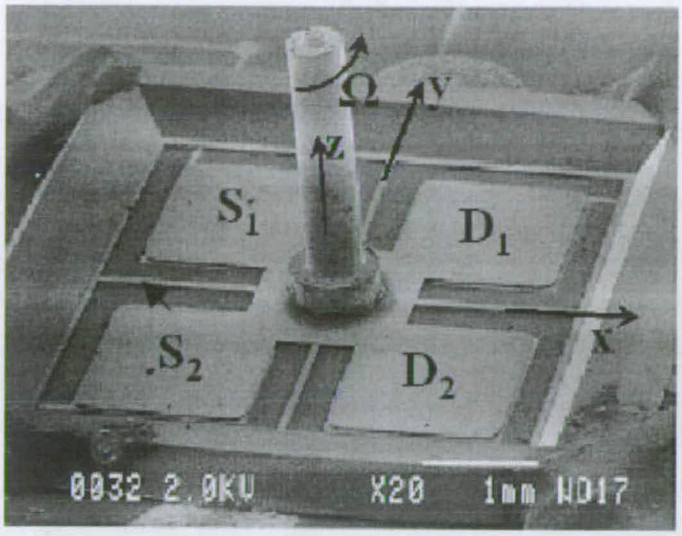


Figure 2.1: Electron microscope photograph of the JPL/Boeing gyroscope [20]

2.3 Circuits Analysis

As it was mentioned in Section 2.2, JPL/Boeing gyroscope relies on the coupling of an excited vibration mode into a secondary mode due to Coriolis acceleration. Figure 2.2 depicts a high-level schematic, which presents the functionality of the JPL/Boeing gyroscope [20]. It employs feedback compensation to achieve *harmonic excitation* of selected modes, *disturbance rejection* and *tuning* of the sensor's dynamics. These tasks are employed by two control loops named *drive* and *sense-rebalance* loop. There is also a *demodulation stage* that estimates the angular rotation rate of the gyroscope by demodulating the sense rebalance signal with respect to a measurement of the drive loop response.

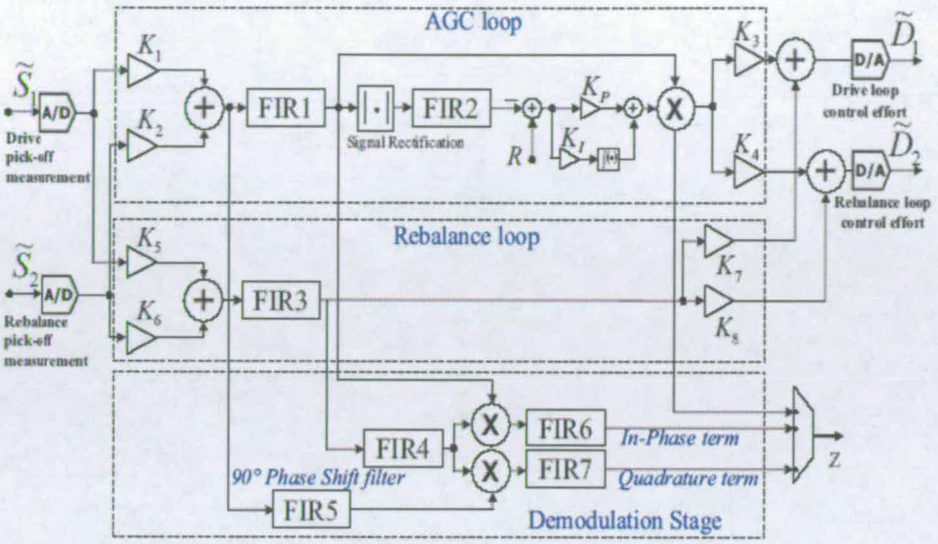


Figure 2.2: High-level schematic electronics of the gyroscope's control-loops [20]

Since the Coriolis acceleration is proportional to the velocity of the driven mode, it is desirable to keep the amplitude and the frequency of the *drive* oscillation as large as possible and maintain it constant because small variations can swamp the Coriolis acceleration. This task is accomplished by the *drive-control loop*, which employs for amplitude control, an Automatic-Gain-Control (AGC) loop [72]. According to Figure 2.2, the AGC consists of an all-pass filter (FIR 1), an amplitude detection scheme, a comparator for identifying the amplitude error and a Proportional-Integral (PI) controller that is in charge of providing feedback compensation. According to [20] FIR 1 is an 85-tap filter that achieves attenuation at 2700 Hz. Furthermore, due to the lag that is introduced by the analog-to-digital (ADC) and digital-to-analog (DAC) converters and the anti-aliasing filters, the phase of FIR 1 must be shaped to achieve the target loop phase of 0 degrees at 4428 Hz. Subsequently, the amplitude detection scheme consists of a 2's compliment module and a 51-order low-pass filter with 50 Hz cut-off frequency and stop-band attenuation of 40 db. The scope of this module is to produce an estimate of the drive mode response amplitude. Then this amplitude is compared to a programmable level and the error that is generated by the comparator is fed to the PI controller. Finally, the output of the controller modulates the drive mode signal before the signal is fed back to the actuators. The second control loop consists of a linear rebalance loop controller, implemented by FIR3. The primary objective of this loop is to reject the disturbance injected by the Coriolis Effect into the second degree of freedom. A secondary objective is to achieve disturbance rejection in the neighborhood of 2700Hz, which corresponds to damping an elastic mode, whose excitation can lead to degraded sensor performance. Finally, the *demodulation* stage consists of the FIR4 and FIR5 filters, which are typically used to shift signal phases and FIR6 and FIR7, which are low-pass filters, located after the multipliers.

2.4 Previous work and new challenges

Significant research work has been done by both JPL and University of California Los Angeles (UCLA) associated with the design, packaging, identification and analysis of the JPL/Boeing gyroscope's dynamics and the implementation of a reprogrammable ASIC solution that incorporates the gyroscope's tasks, introduced in Section 2.3. In [99], JPL has presented up-to-date work on the design, fabrication and packaging of a silicon MEMS gyro design for Space applications. Furthermore, UCLA in [71] discusses the identification of multi input/output models of the JPL/Boeing gyroscope. Subsequently, the authors in [73] show the analysis of a non-linear control system that is used to excite and maintain a pre-defined amplitude of a lightly damped degree of freedom in the JPL/Boeing gyroscope. After the completion of the dynamics analysis, further work from UCLA [33] presents on-going achievements on the implementation of the control-loops on a reprogrammable ASIC solution. In addition to this JPL reports in [11] on-going research into a MEMS gyroscope and presents a new fabrication method that improves the gyro's performance. Work analyzing the gyroscope's dependence on temperature variations, is presented in [30], where JPL determines the effect of hysteresis over the range of 35°C to 65°C.

From a hardware perspective, the platform, which accommodates the electronics associated with the gyroscope's control-loops, needs to meet some criteria, such as low-power consumption, adaptability and fault-tolerance that compose significant prerequisites for designs targeting Space applications. UCLA has implemented a low-power reprogrammable ASIC environment, which nevertheless does not provide more flexibility than *custom reconfigurable designs*, *industrial FPGAs* such as Virtex-II and Virtex-4 devices [118], 2nd generation reprogrammable flash devices (ProASIC^{PLUS}), provided by Actel [1], AT6000, AT40K and AT40KAL series provided by Atmel [5], *programmable logic devices (PLDs)* [3], [61] and digital signal processing (DSP) platforms, such as the TMS320 provided by Texas Instruments (TI) [105]. On the contrary, these coarse-grained reconfigurable structures are not optimized for a certain application and they target general purpose applications. Therefore, they are quite inefficient in terms of power, while performing evolution on these devices is quite challenging, since their hardware structure is not transparent to the user and the their configuration string is very long. On the other hand, ASIC designs are not able to compensate for faults that occur in Space, due to the high density of radiation and hence it is very difficult for an organization to invest a huge budget in a Space mission, without securing first high reliability. Therefore, there is a need for new custom reconfigurable architectures that would be able to efficiently accommodate the gyroscope's electronics that mainly consists of seven high-order (up to 128 taps) FIR filters.

FIR filters are employed in the majority of DSP based electronic systems (image, audio/video processing and coding, sensor filtering etc.) and therefore their efficient implementation within embedded System-on-Chip (SoC) systems is a crucial issue. Specifically, the arithmetic unit and especially the multiplier block in conventional FIR filters, produces several drawbacks that negatively affect the power consumption, the throughput and the fault-tolerance of these filters. Thus, it is imperative for reconfigurable SoC systems that primarily target high robustness, to use a fine-grain design model. According to this model, redundancy can be employed more efficiently because the electronic components that compose the overall architecture are primitive, not complicated operations (such as addition, subtraction and shifting) that do not introduce extensive area overhead, like conventional multipliers do. Hence, the primitive operation filtering technique (POF) [17] is suitable for reconfigurable fabrics that target fault-tolerant applications. Systems that operate in Space must be robust to high density radiation in order to recover from faults without the need of human intervention, which is usually impossible. Particularly, in highly radiated environments, the existence of alpha-particles and cosmic-ray radiation creates Single-Hard-Errors (SHEs) [47]. These are permanent stuck-at fault errors that mainly affect the latch state and the memory cells of electronic devices. Conventional fault-tolerant VLSI systems employ techniques such as *redundancy, check-pointing* [96] and *concurrent error detection* [78]. Their purpose is to maintain system operation or prevent further successive faults by minimizing the damaged sustained [65]. However, these approaches are costly as they reduce operational speed and increase physical area. Alternatively, the approach of using evolutionary based techniques in the expanding field of EHW [67], [76], [92], [100] has proved to be suitable for providing to reconfigurable structures the best possible configuration string under faulty operational conditions. These approaches provide novel techniques for fault recovery without the need for additional redundancy, fault detection or diagnosis [52]. Hence, they can detect the functional deviation of a system and provide alternative configuration schemes to the hardware substrate in order for the system to maintain its original functionality without any procurance.

Moreover, it has been proven that the POF technique can also be used in order to substitute the inefficient (in terms of silicon area and power consumption) multipliers, which are contained in FIR filters, with primitive operators. This problem is known as multiple constant multiplication (MCM) and it is considered a fundamental problem in computer arithmetic and is particularly relevant for the multiplier-less implementation of FIR filters. The multiplication of a variable by a known integer can be decomposed into additions, subtractions and binary shifts. The problem of finding the decomposition with the least number of operations is known as single constant multiplication (SCM) problem and it is NP-complete, as shown in [19]. Initially, using the canonical-signed-digit (CSD) method [10], the decomposition of a multiplication can be done more efficiently, than using conventional two's complement (2's) coding. However, the optimal decomposition is not obtained with CSD. The existing MCM algorithms can be divided into four general classes [113]:

- Digit-based recoding
- Common sub-expression elimination (CSE) algorithms
- Graph-based algorithms
- Hybrid algorithms

Digit-based recoding employs simple methods like CSD [83], [110]. It is the fastest method in terms of computational effort but still the worst performing in terms of adder cost.

Common sub-expression elimination employs an algorithm [35], [64], [77] that mainly attempts to find common sub-patterns in the representations of the constants after the constants have been converted to a convenient number system, such as CSD. Moreover, recent research work [25] has proposed alternative number representations to retrieve considerably better solutions using a CSE algorithm. However, this algorithm does not provide the optimal MCM solution and it has the disadvantage that it depends on the number representation.

Graph-based algorithms incorporate the bottom-up methodology that iteratively constructs a graph that consists of vertices (additions and/or subtractions) and binary shifts to represent the multiplier block. These algorithms are guided by a heuristic that determines the next graph vertex to add to the graph. Moreover, they produce solutions with the lowest number of operations. There are several representative examples such as [17], which proposes four MCM algorithms and [23] that present an enhancement over the previous algorithm. Subsequently, the authors in [114] presented a POF-based methodology for the synthesis of FIR filters, while in [7] the authors investigated a number of configurable arithmetic macro structures designed to perform coefficient multiplication using the POF design technique. Furthermore, there is the Reduced-Adder-Graph (RAG-n) algorithm [23] that has been derived by an exhaustive search using the Minimum-Adder-Graph (MAG) algorithm [22]. The RAG-n algorithm is assumed to provide optimal solutions only for word-lengths up to 12 bits. This work has been extended for constants up to 19 bits [34]. However, both algorithms share the same important disadvantage, that is, the dependence on a pre-computed table of optimal single constant decompositions, whose size is exponential to the number of bits. Subsequently, the authors in [82] presented a GA based on RAG-n algorithm that can be used for the ASIC implementation of FIR filters. Finally, the authors in [113] presented a new MCM algorithm that achieves a 20% reduction in additions and subtractions for word-lengths up to 32 bits, compared with the best previously known algorithms and furthermore it considers additional optimization metrics, such as the critical path [24].

Hybrid algorithms employ a mixture of different algorithms in order to obtain the optimal result. A representative example is the RAG-n algorithm, which is a combination of the MAG and a heuristic algorithm. Hence, the hybrid algorithm can switch between different algorithms at a given iteration, in order to obtain better result.

All the previous research work is composed of software implementations that mostly incorporate multiplier-less solutions, which suit the implementation of ASIC low-power FIR filters. Furthermore, the authors in [115] presented a synthesis technique for the implementation of medium-order filters, which are superior to equiripple method in terms of chip area and clock rate. The drawback of this technique is that it becomes computationally impractical when implementing higher-order filters. Finally, from a software perspective, work has been reported in [89] that uses a GA for the implementation of a 2-dimensional FIR filter (up to 13 taps).

From the hardware perspective, previous research work on FIR filters' evolution can be found in [74], where the author performs evolution at gate level to achieve the functionality of up to 6-tap filters. However, the disadvantage is that the evolved filters are nearly linear and that this methodology is not suitable for the realization of high-order filters. The work in [44] presents an evolutionary driven reconfigurable architecture that is suitable for fault-tolerant and considerably low-power applications. However, the achieved results have been limited to small or medium-order filters and the whole design incorporates significant area overhead. Consequently, this overhead increases the power consumption of the design and enlarges the search area which higher-order filters require for their encoding.

2.5 Summary

Considering the previous work that has been done in the realization of FIR filters and the limitations implied by the nature of the targeted application, it is apparent that the ideal architecture for the gyroscope's electronics must be an amalgam of characteristics adopted from different methodologies. Therefore, the POF technique is indispensable for the implementation of low-power and simultaneously fault-tolerant reconfigurable structures. However, the MCM problem cannot be solved using large memories that store the single constant decompositions because a global memory is very prone to SHEs and moreover the architecture must be autonomous, which is a characteristic provided by EAs. Therefore, a specially tailored EA has to be implemented for the efficient guidance in terms of *accuracy, power* and *adaptation-speed* of the filters' evolution.

Chapter 3
Evolvable Hardware for the Design of Digital Circuits

3.1 Introduction

Evolvable hardware lies at the intersection of evolutionary computation and physical design. Through the use of evolutionary computation methods, the field seeks to develop a variety of technologies that enable automatic design, adaptation and reconfiguration of electrical and mechanical hardware systems in ways that outperform conventional techniques [68]. EHW employs reconfigurable hardware whose configuration is under the control of an EA. There are different types of EAs, such as *Genetic Algorithms*, *Evolutionary Programming*, *Evolution Strategy*, *Genetic Programming* and *Learning Classifier System*.

It is well known that MEMS sensors are devices whose functionality is highly affected by manufacturing irregularities, temperature and pressure variations and that their associated electronics operating in Space environments are prone to faults, due to high densities of radiation. Due to this need of adaptivity, EHW seems to be the most suitable framework for the implementation of the JPL/Boeing gyroscope's electronics. It inherently comprises unique features, compared with other reconfiguration mechanisms (static, compiler driven reconfiguration, etc.) that facilitate dynamic reconfiguration, autonomous learning for adaptation to environmental variations and anticipated faults and continuous evolution towards better solutions.

3.2 Overview of Genetic Algorithms

This section presents the main operation and features of a conventional GA. The concept was first proposed by John Holland in 1975 [42], and was the first EA to encode possible solutions using bit-strings.

To apply evolution to the design of circuits, one or more candidate designs are described by a string of bits within a memory pool. This memory keeps the so called *phenotype* or *population,* which initially consists of randomly selected binary-strings named *chromosomes*, *individuals* or *genotype* and may encode the configuration of reprogrammable arithmetic and logic units and/or reprogrammable interconnections between hardware components. Each one of these *chromosomes* is evaluated in terms of how accurately the given configuration (chromosome) approaches the targeted functionality. The evaluation mechanism is called the *fitness-function* and incorporates one or more objectives that characterize the ideal behaviour of a circuit. Therefore, in the evolution of FIR filters the primary objective of the fitness-function is the coefficient-set that determines the filter's functionality. *Selection* mechanisms are then used to identify the most successful chromosomes within the current *population.* Moreover, GAs employ simple operators during reproduction, such as *crossover* and *mutation*. At each reproduction two new chromosomes are produced *(offsprings)* from the *parents*, which are selected based on their *fitness-score* from the current *population.*

The aim of the GA is to potentially produce fitter *chromosomes* compared with those comprising the old *population*. This iterative process is repeated until an acceptable solution is found or a specified number of iterations, called *generations,* have been completed. Figure 3.1 depicts the design flow of a generic GA. According to this, the first population is initialized with random candidate designs.

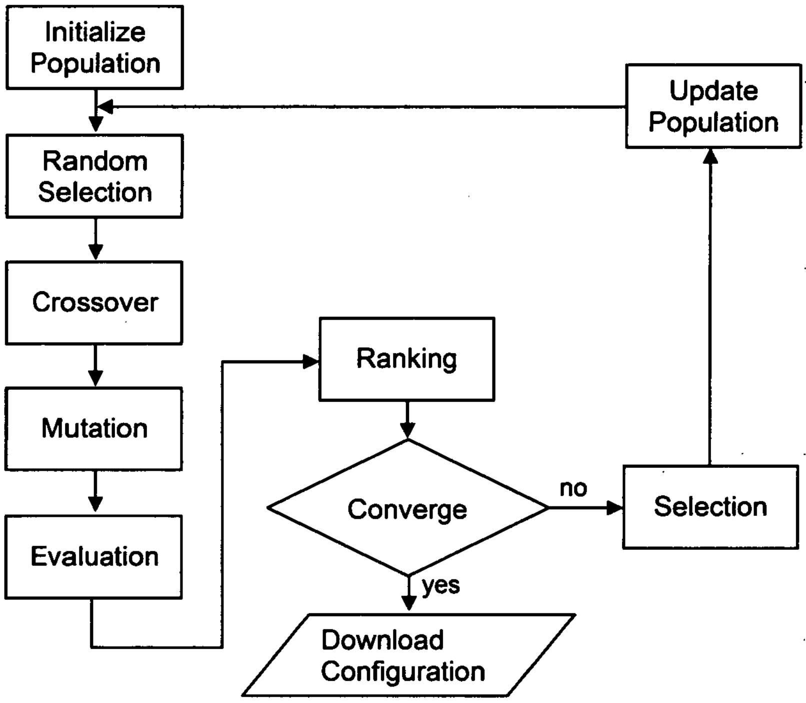


Figure 3.1: Design flow of a generic GA

The initialization of the population is quite important process. During the implementation phase of different electronic circuits the initialization must be random. This happens in order for the individuals to present diversity and maximize the chance of the GA efficiently exploring the total search space. However, during adaptation/reconfiguration phase, the initial population must contain a priori knowledge concerning the initial implementation. This approach is very beneficial because the GA has to perform minor modifications on the system's behaviour and not to evolve from scratch the functionality of the targeted electronics. Subsequently, the population is extended by selecting pairs of chromosomes to perform reproduction.

Selection is a very critical parameter in EAs and significantly affects the convergence time and the accuracy of the solution. Therefore, if *selection* is too low then the rate of convergence towards the optimum is likely to be slow. Alternatively, the GA may become stuck in a local optimum due to the loss of diversity in the population. There are different kinds of *selection* approaches such as *random selection*, *tournament selection*, *truncation* or *deterministic selection* and *proportional selection.*

In *random selection* the parents that are going to be used in the next reproductions are randomly selected by just generating two random numbers at a time that correspond to a position within the memory that stores the population. This strategy does not secure that the most successful parents are selected but it manages to balance *exploration* and *exploitation* [2]. An efficient search methodology must be time-limited and therefore intelligent search must combine exploration of new search regions with evaluation of potential solutions already identified. Too much stress on exploration results in a pure random search, whereas too much exploitation results in a purely local search that does not give the optimum solution.

In *tournament selection,* the selection of the each parent is done by randomly selecting two chromosomes and keeping the dominant one based on the fitness-score. In the contrast to *random selection*, *tournament selection* considers the fitness-score of the randomly selected candidates that are to become parents. However, the candidates that participate in the tournament are randomly selected and therefore it is possible for not very successful parents to participate in new reproductions. Due to its simplistic functionality, tournament selection is very popular for hardware implementations of GAs

In *truncation* or *deterministic selection*, the most successful parents are selected to form the new population. In this thesis the employed GAs extend the initial population from size 50 to 100 by performing 25 reproductions. According to *truncation* or *deterministic selection* the 50 higher ranked individuals are always selected to form the new population after each generation.

In *proportional selection*, the probability of an individual becoming a parent is proportional to its percentage of population total fitness. This process corresponds to a weighted roulette-wheel that is divided in slices. The size of the slices is proportional to each individual's percentage of the population's total fitness. Therefore, each time an offspring is required, a simple spin of the weighted roulette wheel yields the reproduction candidate [32]. Compared with the other selection techniques, *proportional selection* is the most fair, since the chance criterion is minimized and it is always proportional to the individual's fitness-score. However, this approach requires more computational effort compared with the previous selection schemes and therefore *proportional selection* is not preferred in hardware implementation of GAs.

After *selection*, the following step is to generate the next generation population of solutions by using genetic operators, such as *crossover* and *mutation*. The offsprings, which are produced using these genetic operators, typically share many of the characteristic of their parents. Generally, the average fitness of the new population increases through this reproduction process, since only the most successful parents from the previous generation are selected for breeding, along with a small proportion of less successful solutions.

Crossover has always been regarded as the primary search operator in GAs. During this genetic operation the two parents exchange information to form the two offsprings. Generally, *crossover* can be categorized in three different types based on the manner that information is exchanged between the parents:

- One-point crossover
- Multi-point crossover
- Uniform crossover

Figure 3.2 depicts a paradigm in which one-point *crossover* is used to generate the offsprings. The cross-point is arbitrarily selected using a random number generator. It is apparent that after *crossover*, child A adopts from parent A the information that corresponds on the left part of the cross-point and from parent B the information on right hand of the cross-point. Similarly, child B adopts from parent B the part of the chromosome that corresponds on the left part of the cross-point and the right one from parent A.

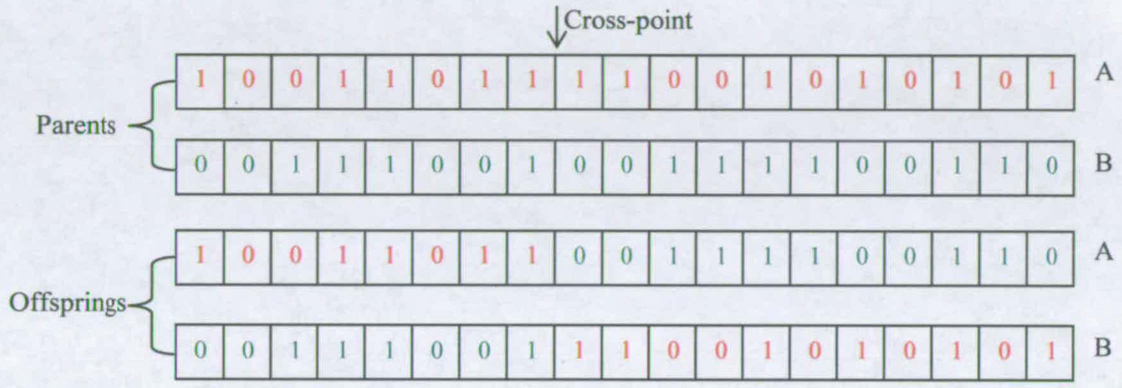


Figure 3.2: Paradigm of one-point crossover

Figure 3.3 depicts a paradigm of a two-point *crossover* and Figure 3.4 shows an example of a *uniform crossover*. In *two-point crossover*, there are two cross-points that are randomly selected. The parents exchange the genetic information that corresponds to the part of the chromosome between these two points, in order to produce the offsprings. Finally, in *uniform crossover*, the parents exchange information by swapping genes (bits) with a predefined probability that, in the paradigm in Figure 3.4, is equal to 25% per gene. The arrows indicate which genes of the chromosome have been randomly selected to be swapped.

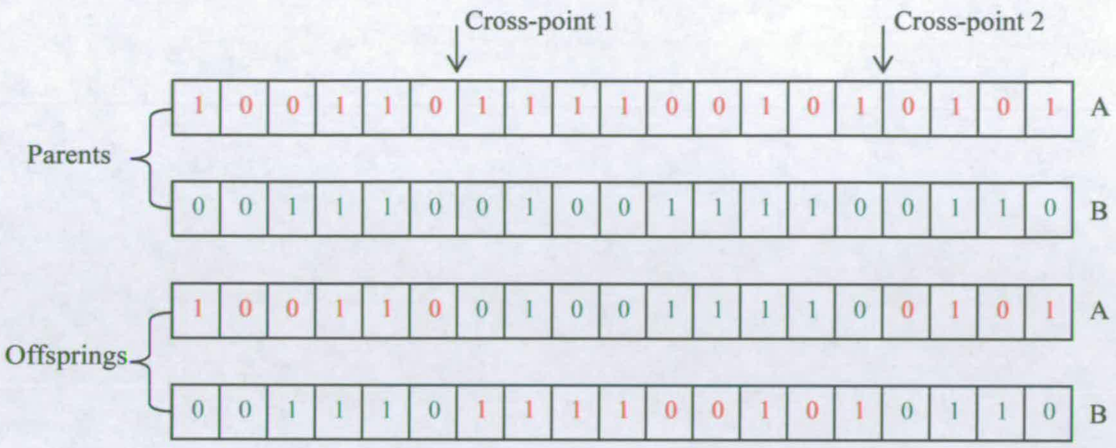


Figure 3.3: Paradigm of two-point crossover

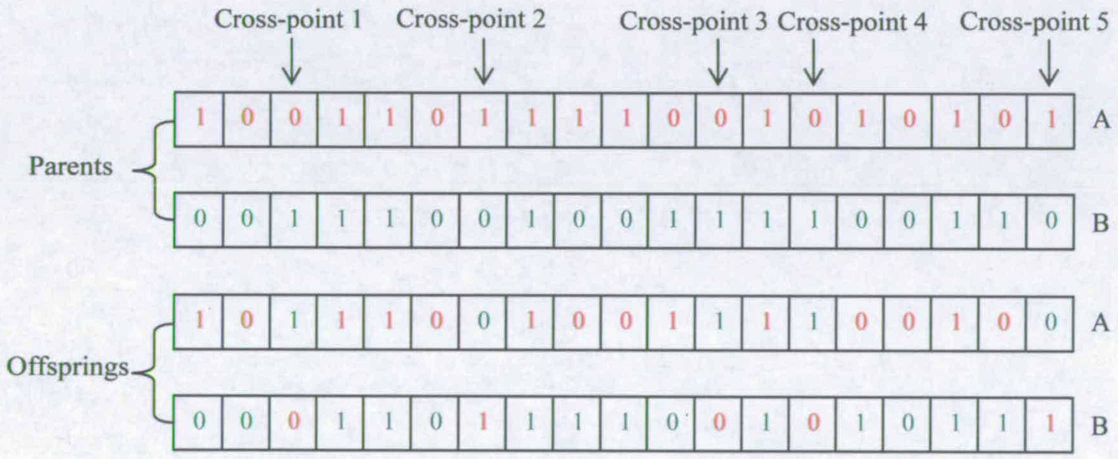
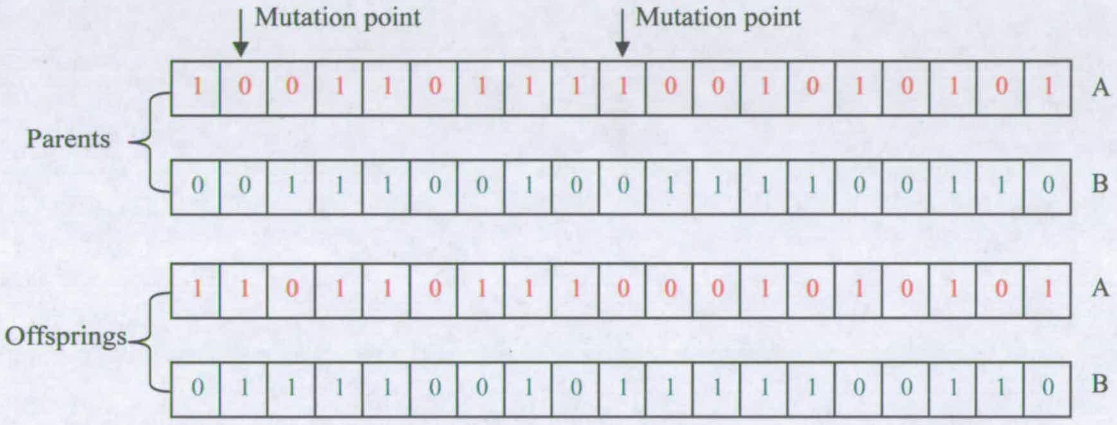


Figure 3.4: Uniform crossover with probability 25% per gene

The role of *mutation* is to maintain genetic diversity from one generation of a population to the next. Thus, it allows evolution to avoid local optimal solutions by maintaining a non uniform population. Similarly with *selection*, high rates of *mutation* support the exploration of new search spaces, whereas low rates of mutation support exploitation since the offsprings are only generated by using *crossover*. The operation of mutation is very simple and associates the flipping of randomly selected bits. Figure 3.5 depicts a paradigm according to which mutation probability is equal to 10% per gene.



Mutation point
Mutation point
Parents
1 0 0 1 1 0 1 1 1 1 0 0 1 0 1 0 1 0 1 A
0 0 1 1 1 0 0 1 0 0 1 1 1 1 0 0 1 1 0 B
Offsprings
1 1 0 1 1 0 1 1 1 0 0 0 1 0 1 0 1 0 1 A
0 1 1 1 1 0 0 1 0 1 1 1 1 1 0 0 1 1 0 B

Figure 3.5: Mutation with probability 10% per gene

After the reproduction phase has finished, the total population, including the parents and offsprings is evaluated based on the fitness-function, which composes a mathematical formula that is application specific and incorporates fundamental objectives of the circuit's behaviour. Therefore, after *evaluation*, the GA assigns a fitness-score to each chromosome that indicates how close the ideal circuit behaviour is to the evolved one, as it is described by the specific chromosome. Subsequently, the chromosomes are ranked based on their fitness-score from the most successful to the least one. Finally, after the *ranking* phase, the GA considers the best found solution and checks whether this solution meets the criteria of the targeted application. If it does, the configuration string is downloaded to the hardware, otherwise the GA proceeds for a new generation.

3.3 Different Levels of Evolution

A major problem in the evolutionary design of digital circuits is the *level of granularity*. Initial attempts to apply EHW to complicated circuits suffered from insuperable problems that primarily originate from the large and non-linear search spaces of such circuits. This is mainly because hardware evolution has initially been based on transistor [60], [91] or gate (AND and OR gates) level [111]. Therefore, as the level of abstraction in circuit design decreases, there is a need for more bits to encode the targeted electronics including the enormous number of transistors and/or primitive gates that have to be used and the associated interconnections. Another obstacle of performing evolution at transistor or gate level is the time needed to calculate the fitness-score of a circuit. Hence, the evolution time increases as the size of the truth table of the evolved circuit becomes larger. Many approaches have been used such as *a priori knowledge* [106], *scalability* (re-use of previously evolved circuits to evolve more complex ones) [111] and *generalized disjunction decomposition* [95] in order to improve the efficiency of transistor and gate level evolution. However, based on up-to-date achievements, these two types of evolution have only been successfully employed for the implementation of small circuits such as a 6-tap FIR filter [74], a 3-bit multiplier [111], a 4-bit multiplier [106], a 6-bit adder [95], an analogue comparator [108] and a 3-bit [124] and 6-bit [60] digital-to-analogue (DAC) converter.

To overcome this obstacle the authors in [38] introduced the evolution at functional level, which can use adders, subtractors, shifters and sine generators to build more complicated designs. This type of evolution reduces significantly the search space because the encoding of the evolved circuits incorporates functional blocks and not transistors or primitive gates. Moreover, the number of interconnections that compose the interface between functional blocks is substantially smaller than in the first two types of evolution. Therefore, by keeping the same search area, more complicated circuits can be implemented in functional level, than in transistor and gate level, respectively. However, one of the most intriguing discoveries in the field of EHW is presented in [102] and shows that it is possible for artificial evolution to create electronic circuits in FPGAs using the physical characteristics of the silicon substrate. Thus, the author discovered that unconstrained evolution, unlike evolution at functional level, can explore unusual ways to solve problems because it is able to exploit the physical characteristics of the hardware substrate. However, this intriguing discovery has a stability problem because the physical characteristics of a hardware substrate differ with temperature variation. Hence, the same design may not operate properly in different environmental conditions or when the same configuration is applied on a different reconfigurable fabric of the same type. Therefore, the selection of the most appropriate type of evolution in terms of abstraction is a dilemma that has to be considered by the designer, based on the technical specification of the targeted application.

3.4 Different Modes of Evolution

The previous section concentrated on the types of evolution based on the level of granularity that this takes place. However, since the emergence of the field of EHW, there have been introduced several methods that aim at designing more complicate circuits with more accurate transfer functions and within a shorter time. Therefore, evolution can be categorized in three different modes [62]: extrinsic, intrinsic and mixtrinsic, based not on the degree of evolvability (granularity level) but on the manner that evaluation is performed. In the first mode the search for the optimal solution is made by using software simulations of hardware models, whereas in *intrinsic* mode the evolution occurs directly in hardware. *Mixtrinsic* evolution mode combines both *extrinsic* and *intrinsic* elements in order to summarize the advantages of the previous two modes and eliminate the mismatches between models and physical hardware.

3.4.1 Extrinsic

Extrinsic evolution is more versatile compared to *intrinsic* and it is easier to implement. It is carried out using a simulation model and only the final configuration is downloaded to the hardware [37]. Specifically the population consists of netlists that describe different configurations of electronic circuits. Therefore, it affords the research more freedom since there is not limitation on the type of basic elements and their interconnection

However, extrinsic evolution presents several disadvantages. Firstly, it introduces a computational overhead in modeling and evaluating the evolved circuits. Secondly, the results of the extrinsic evolution are not always mapped successfully into hardware, unless very well defined constraints are imposed during evolution. Finally, exploiting continuous-time dynamics and subtle physical effects, *intrinsic* evolution can produce circuits beyond the scope of the human designer.

3.4.2 Intrinsic

In *intrinsic* evolution each individual genotype, which may specify component types, values and interconnections of a candidate electronic circuit, is instantiated in physical hardware using a reconfigurable device. Hence, the fitness evaluation is done in hardware [49], [125].

Intrinsic evolution can reduce the evolution time by orders of magnitude because the evolution process takes place on hardware. Therefore, it seems to be the most promising solution for high-speed, adaptive control applications. However, it is constrained by the availability and product support for the up-to-date industrial reconfigurable devices. Moreover, circuits that have been evolved on a FPGA may be extremely difficult to be analyzed, since it is not possible to access individual circuit elements with test equipment.

3.4.3 Mixtrinsic

This mode of evolution emerged from the need to solve the portability problem between the extrinsic and intrinsic evolution. This problem is mainly associated with the mismatches between the software models of the hardware and the physical hardware substrate itself [101]. According to this evolution technique, a part of the population is probabilistically selected to perform extrinsic evolution, while the rest is evaluated in hardware. A further enhanced technique derived from mixtrinsic evolution is called *combined mixtrinsic evolution* [93]. According to this, each individual is evaluated both extrinsically and intrinsically and subsequently, the overall fitness-score of the each evaluation is calculated from the average value.

3.5 Background Theory of the Gyroscope's Electronics

This section presents background theory for the digital electronic circuits that are associated with the JPL/Boeing gyroscope. The perfect understanding of these electronics is very important for the conception of the EHW platform that will accommodate them. As was mentioned in Section 2.3, the most complicated electronic circuits that compose the control-loops of the gyroscope include 7 high-order FIR filters and a PI controller.

3.5.1 Finite Impulse Response filters

A filter is essentially a system that selectively changes the wave shape, the amplitude-frequency and/or phase-frequency characteristics of a signal in a desired manner [45]. Common filtering objectives are to improve the quality of a signal (remove or reduce noise, etc.), to extract information from signals or to separate two or more entwined signals, to make, for example, efficient use of an available communication channel. Digital filters play a very important role in DSP and compared with analogue filters, they are preferred in numerous applications such as data compression, biomedical signal processing, speech/image processing, data transmission, digital audio, and telephone echo cancellation. Their advantages are summarized below:

- Digital filters can have fully linear phase response.
- The performance of digital filters does not vary with environmental changes, as in analogue filters.
- The frequency response of digital filters can be automatically adjusted, using adaptive hardware of software filters.
- Several input signals can be filtered by one digital filter without the need to replicate the hardware.
- Both filtered and unfiltered data can be saved for the further processing.
- Tremendous advancement in VLSI technology facilitates the implementation of digital filters with tiny size and very low-power consumption.
- Digital filters can be employed at very low frequency, where the use of analogue filters is impractical.

On the other hand, digital filters face problems such as speed limitation and finite word-length effects. The former disadvantage implies that the highest frequency of a digital filter is limited by the conversion time of the ADC and the settling time of the DAC.

The latter emerges from the quantization of a continuous signal and the round-off noise that is incurred during computation.

Digital filters are broadly divided into two classes, namely finite-impulse-response (FIR) and infinite-impulse-response (IIR) filters. This thesis is only concerned with FIR filters since these are the main electronic components of the sensor's electronics. The basic FIR filter is characterized by the following two equations:

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \quad (3.1)$$

$$H(z) = \sum_{k=0}^{N-1} h(k)z^{-k} \quad (3.2)$$

Where *h (k)*, *k=0, 1, ..., N-1*, are the impulse response coefficients of the filter, *H(z)* is the transfer function of the filter and *N* defines the filter's order (number of filter coefficients). Equation (3.1) is the FIR difference equation. It is a time domain equation and describes the FIR filter in its non-recursive form. According to this, the current output sample, *y (n)*, is a function only of the past and present values of the input, *x(n)*.

One of the most important advantages of FIR over IIR filter is that the former is always stable, suffers less from round-off and coefficient quantization errors and achieves exactly linear phase response.

Linear phase response is one of the most important properties of FIR filters. When a signal passes through a filter, it is modified in amplitude and/or phase. The nature and extent of the modification of the signal is dependent on the amplitude and phase characteristics of the filter. The phase or group delay of the filter provides a useful measure of how the filter modifies the phase characteristics of the signal. Hence, if a signal consists of several frequency components, the phase delay of the filter dictates the amount of time delay each frequency component of the signal suffers in going through the filter. On the other hand, the group delay defines the average time delay the composite signal suffers at each frequency. Further analysis and mathematical details can be found in [45].

The frequency response of a FIR filter is often specified in the form of a tolerance scheme. Figure 3.6 shows such a scheme for a low-pass filter. Similar schemes can be drawn for other frequency selective filters. Referring to the figure, the following parameters are of interest:

- δ_p, peak pass-band deviation (ripple)
- δ_s, stop-band deviation
- f_p, pass-band edge frequency
- f_s, stop-band edge frequency
- F_s, sampling frequency

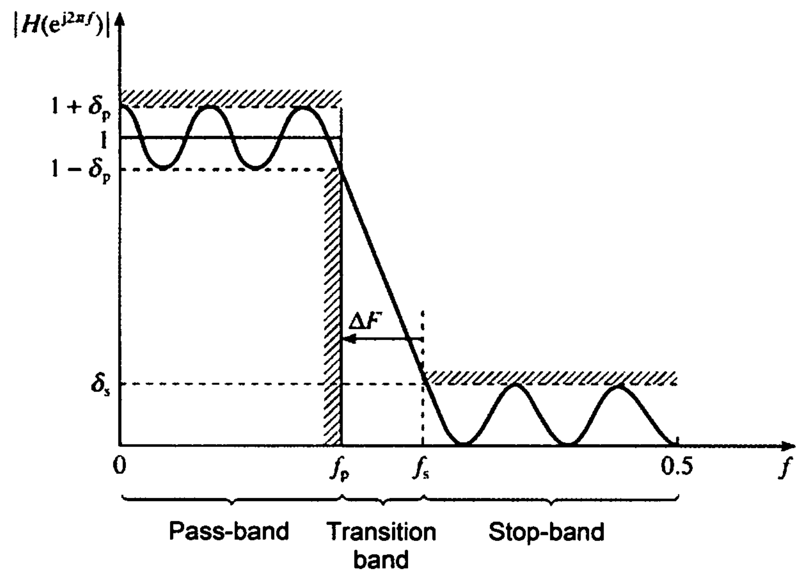


Figure 3.6: Frequency response specification of low-pass filter

The difference between f_s and f_p gives the transition width of the filter. Moreover, the pass-band (20*Log ($1+\delta_p$)) dB and stop-band (-20*Log (δ_s)) dB deviations are expressed in decibels.

3.5.2 Proportional Integral Derivative controller

A feedback controller is designed to generate an output that causes corrective effort to be applied to a process so as to drive a measurable process variable, y(t), towards a desired value, known as the set-point or reference, noted r(t). As it is shown in Figure 3.7, the concept is the system output to follow the reference r(t). Therefore, feedback controllers determine their output by observing the difference, called error e(t), between the reference r(t) and the actual process variable measurement.

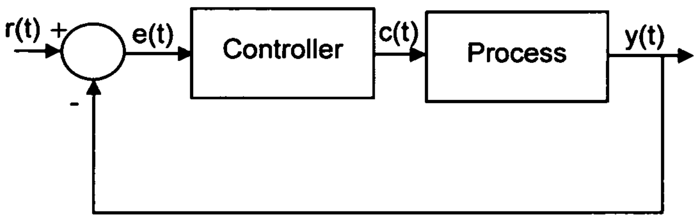


Figure 3.7: Feedback control system

For the class of linear systems, a very widely used controller is the PID (Proportional Integral Differential). This controller considers different aspects of the system, such as the current value of the error, the integral of the error over a recent time interval and the current derivative of the error signal to determine not only how much of a correction to apply but for how long, as well. Each of these quantities is multiplied by a tuning constant and added together [4]. Thus, the output of the PID controller, c(t), is a weighted sum, as shown in Figure 3.8.

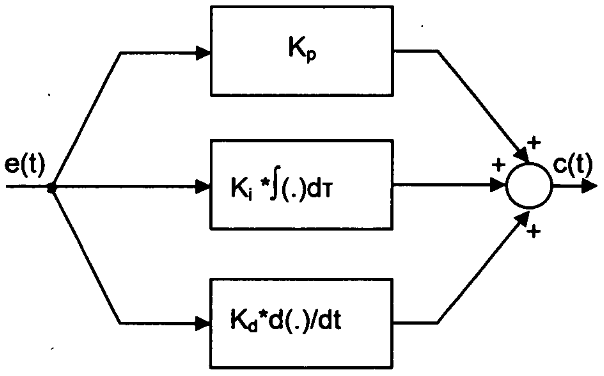


Figure 3.8: PID controller

Nowadays, there is a trend for digital controllers to substitute analogue ones due to the availability of low-cost digital computers and the flexibility provided by digital designs.

A digital version of a PID controller is shown in Figure 3.9. In the digital version, the integral part becomes a sum and the differential a difference. The continuous time signal e(t) is sampled in fixed time intervals, which equal a determined sample period, T_C. A/D and D/A converters are employed to establish the interface with the input and output, respectively. The digitalized input, $e_D[j]$, is valid only in time instants $t = kT_c$ for all $k >= 0 \in Z$. It is desirable for the controller to have sample period as small as possible in order to have more levels of quantization.

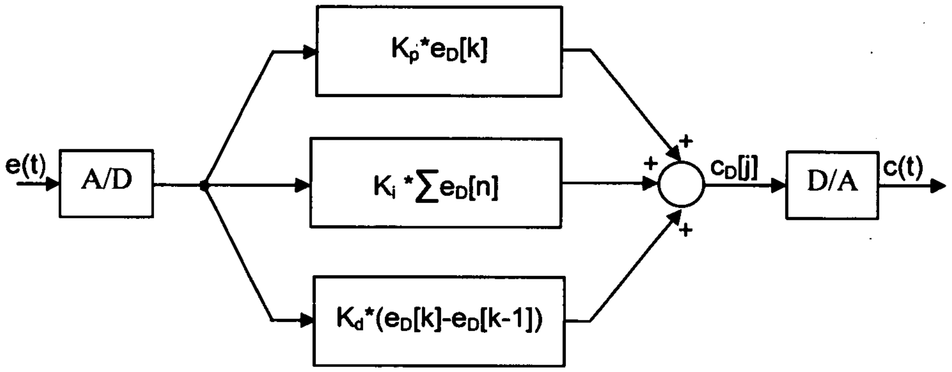


Figure 3.9: Discrete-time PID controller

The PID controller receives as input the error, given by e(t) = r(t) – y(t) and computes the control variable c(t), which is its output. The PID has three terms: a) the proportional term P, corresponding to proportional control, b) the integral term I, giving a control action that is proportional to the time integral of the error and c) the derivative term D, proportional to the time derivative of the error. The control signal c(t) is calculated as follows [4]:

$$c(t) = k_p \cdot e(t) + k_i * \int_0^t e(\tau)d\tau + k_d \cdot \frac{de(t)}{dt} \quad (3.3)$$

It is common to define $K_p = K$, $K_i = K/T_i$ and $K_d = K*T_d$. Where T_i and T_d correspond to integral and derivative time, respectively. Hence, equation (3.3) becomes:

$$c(t) = k * [e(t) + \frac{1}{T_c} * \int_0^t e(\tau)d\tau + T_d \cdot \frac{de(t)}{dt}] \quad (3.4)$$

In the digital version, it is necessary to discretize the controller by approximating the integral and derivative terms [4].

$$\frac{de(t)}{d(t)} \approx \frac{e(t) - e(t - T_c)}{T_c} \quad (3.5)$$

$$\int_0^t e(\tau)d\tau \approx T_c \cdot \sum_{n=0}^{k} e(n * T_c) \quad (3.6)$$

In equation (3.4), it is assumed that $K = t/T_c$. Moreover, considering the approximations described by (3.5) and (3.6), equation (3.4) can be approximated by the following equation:

$$c(k * T_c) = k * [e(k * T_c) + \frac{T_c}{T_i}\sum_{n=0}^{k} e(n * T_c) + T_d * \frac{e(k * T_c) - e((k-1) * T_c)}{T_c}] \quad (3.7)$$

3.6 Conclusions

This chapter initially presents an overview of GAs and criticizes on different features such as initialization, genetic operations, selection, exploration and exploitation that determine the functionality of a GA. Moreover, it analyses the different levels and modes of evolution and emphasizes how these may have an important role in the successive evolution of an electronic circuit and which are the factors that have to be considered by the designer in order to achieve the best possible results. The chapter concludes with background knowledge related with the electronic circuits (FIR filter and PID controller) that have been evolved in this research work.

Chapter 4
Designing a Stand-Alone Fault-Tolerant Architecture

4.1 Introduction

Apart from a fault-robust hardware substrate that accommodates the electronics of the JPL/Boeing gyroscope, the implementation of a stand-alone fault-tolerant architecture must provide a fault-tolerant environment for the EA, as well. The concept of a stand-alone board-level evolvable system was first introduced in [94]. It consists of two main components, including the *reconfigurable hardware* and the *reconfiguration mechanism*.

This chapter focuses on a single chip solution that accommodates a fault-robust parallel GA (PGA) capable of resolving in real-time the attitude determination of a GPS based system [119], [120]. However, this is preliminary work and the obtained achievements can be used for the implementation of the reconfiguration mechanism related with the JPL/Boeing gyroscope. In previous research work a variety of platforms have been employed to act as *reconfiguration mechanisms* including a supercomputer [53], [55], a single PC [101], a DSP [29], an FPGA [86] and an ASIC [40], [116]. Moreover, a number of programmable logic devices have been used to implement EAs in hardware [21], [81], [87], [109]. These are capable of running considerably faster than those implemented on general-purpose micro-processors and are therefore suitable for applications, which require on-line adaptation. However, these are implementations of conventional GAs, which are generally able to find good solutions in a reasonable amount of time, but as they are applied to harder and more complicated problems there is an increase in the time required to find adequate solutions. As a consequence, there have been efforts to make GAs faster and more accurate. One of the most promising choices is to use PGA implementations. Moreover, the targeted *reconfiguration mechanism* needs to be robust to the presence of faults and therefore PGAs compose an ideal solution. On the contrary to the previous research work, the architecture presented in this chapter exploits: a) massive parallelism of fine-grained processing elements (PEs) and b) a two layers approach, in order to cope with faults.

In general, PGAs consist of multiple populations that evolve separately most of the time and exchange individuals occasionally. The basic idea is to divide a task into chunks and to solve these simultaneously using multi-processors [18]. This divide-and-conquer approach can be applied to many different PGA approaches. Different methodologies can exploit massively parallel computer architectures or take advantage of multi-computers with fewer and more powerful PEs [79], [85].

4.2 System Description

The overall system consists of two distinct layers. The first performs the intended functionality and is called the *Computational Layer (CPL)*. Its functionality can be differentiated based on the targeted application. The second controls the execution of the CPL and therefore is termed the *Control Layer (CTL)*.

The ideal fine-grained PGA requires a large number of PEs because the population is partitioned into a large number of sub-populations that ideally must consist of one individual. Furthermore, each PE has its own chromosome that provides a different optimized solution to the problem. Hence, each one of the two layers consists of N PEs. The exact number can differ according to the trade-off between the degree of robustness and the area occupation that a specific application requires. For the purpose of this work both CPL and CTL consists of 64 PEs each. Their structure is similar, however their functionality is different. The taxonomy of the PEs in both layers is arranged in an 8x8 array that forms the fine-grained PGA that act on each member of the population in parallel. Consequently, each member of the population performs crossover with its immediate (north, east, south, west) neighbours. Figure 4.1 illustrates the topology of the PGA and the interface between PEs belonging in the same layer, while Figure 4.2 illustrates the interface between a unique controller in the CTL and CPL. Each one of the controllers in the CTL takes an input, which corresponds to the fitness-score (5-bit) of each computational PE in the CPL. Hence, controllers are able to monitor the performance of the PEs. In addition to this, each controller has 64 output signals (PE_Enable – 1-bit) and each one corresponds to a PE defining which of these should operate. The fitness-score of the GPS PEs is presented by decimal numbers with fractional accuracy and its range may vary from 0 to 0.96875.

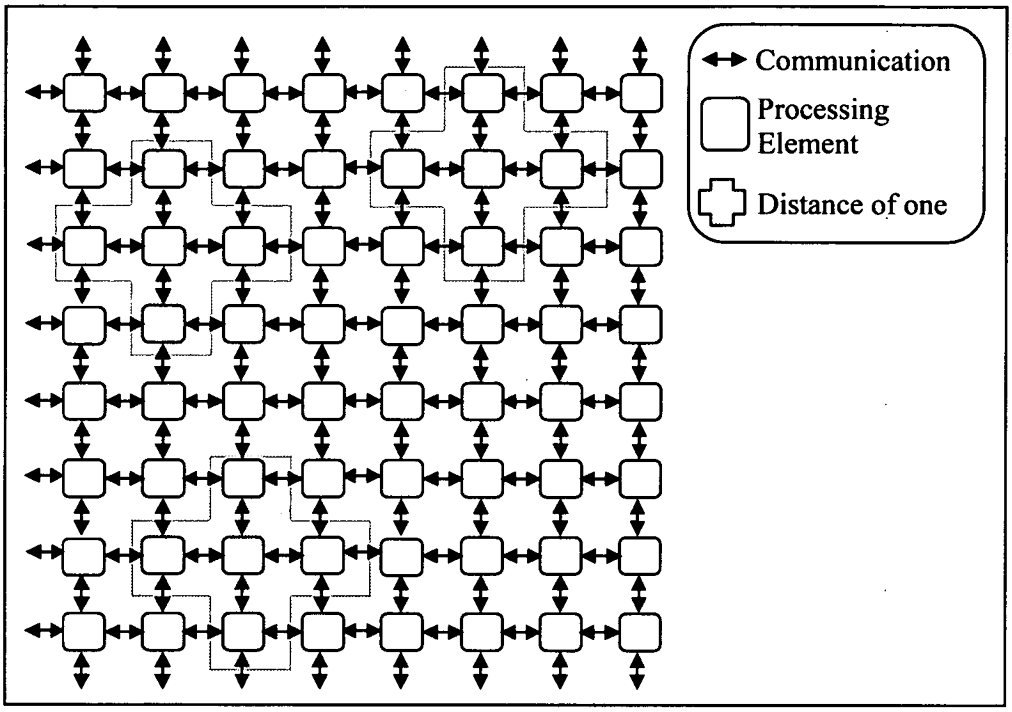


Figure 4.1: Topology of the PGA and interface between PEs belonging in the same layer

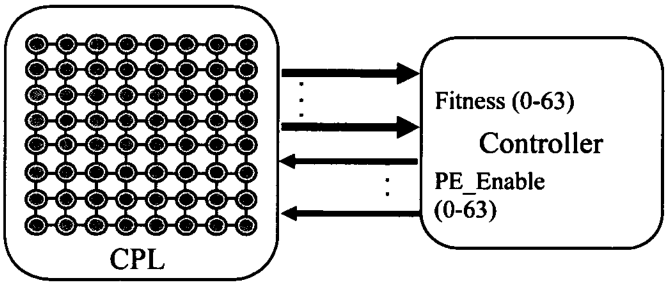


Figure 4.2: Interface between a unique controller in CTL and CPL

4.2.1 Computational Layer

The 64 PEs forming the CPL run a fine-grained PGA, which performs a parallel search to identify the following parameters:

- **Azimuth Angle φ**, of the baseline
- **Elevation Angle β**, of the baseline
- **Length b**, of the baseline

The first two are used to determine the attitude of an object. The length of the baseline is defined as the distance between the two receiver antennas that are attached on the vehicle. The Finite-State-Machine (FSM) that controls the operation of each PE is identical. Figure 4.3 depicts a brief description of the states that compose the FSM. The population of each PE consists of one individual, which is initially evaluated. Then, in the third state (*Exchange Data)* of the FSM, each PE exchanges information with the four PEs existing in the north, east, south and west direction. Thus, it forwards to them its own best solution (chromosome) and the equivalent fitness-score and similarly receives the same information from its four neighbours. In addition to this, in the *Selection* state the best delivered chromosome is selected, according to its fitness-score, to participate with the unique individual that forms the initial population in a new reproduction. After, the two offspring are generated the PE evaluates the two children and the best received chromosome, as well, for verification. Finally, it selects the best out of the three and updates its population.

- Initialize Population X_1
- Evaluate X_1
- Exchange Data with adjacent PEs (X^N, X^E, X^S, X^W) and (F^N, F^E, F^S, F^W)
- Selection of best chromosome X_M = max (X^N, X^E, X^S, X^W)
- Crossover $X_1 \infty X_M$ (probability 90%)
- Mutation (probability 0.1%)
- Evaluation of two children chromosomes X', X'' and the best chromosome X_M
- Ranking Chromosomes X_1 = max (X_M, X', X'')
- GA Convergence

Figure 4.3: Description of the states comprising the FSM of each PE

4.2.2 Control Layer

The structure of this layer is identical to the CPL. It is organized as an 8x8 array and the 64 PEs, termed *Controllers* due to the nature of their functionality, run a fine-grained PGA. The scope of the PGA is to optimize the problem that indicates the optimal and most efficient selection of the "alive" PEs in the CPL. This means that when a PE is surrounded by faulty ones in all the possible (north, east, south, west) directions, there is no point for it to operate. This is because it cannot disseminate its solution to the rest of the PEs and contribute to the solution. Therefore, the PGA continuously tries to resolve 8 combinations of five PEs, which form a cross, as it is shown in Figure 4.4. This information is given by each one of the 64 chromosomes that compose the whole population of the CTL. The faulty PEs are depicted in a black color. It can be deduced that crosses, which contain even one faulty PE are not valid because the information that the faulty PE transfers to its neighbours is invalid and hence the maximum search area that can be reached at an ideal case is decreased. The "alive" PEs, which are not used, will remain in *stand by* mode until the next generation. In addition to this, they exchange information with the "alive" ones but it is not imperative for them to converge.

Concerning the implementation details of the PGA in the CTL, each chromosome encodes the coordinates (x, y) of 8 PEs in the CPL. These PEs exist in the centre of the crosses selected by the chromosome. The coordinates are 8-bit long each and therefore the total length of each individual is 64 bits. The fitness function is defined as the summation of the fitness-scores corresponding to the 8 selected PEs, which form a cross in the CPL. Equation 4.1 below expresses the mathematical formula.

$$\text{Fitness}_{(CTL)} = \sum_{i=0}^{7} \text{fitness\_i}_{(CPL)} \qquad (4.1)$$

Whenever the 8 selected crosses consist of "alive" PEs the fitness value should not be less than 38.75. This number comes out from the multiplication of the following parameters: 1) the number 8, which defines the crosses selected by a chromosome, 2) the minimum, acceptable fitness score (0.96875) for each PE in CPL and 3) the number 5, which defines the PEs that form each cross.

Similarly to the CPL, faults can also occur in the CTL. In the contrast to the CPL, the robustness of the *control layer* is only based on the inherent parallelism of the PGA. Therefore there is no mechanism to evaluate the quality of the solutions that are given by the CTL in order to isolate the faulty PEs.

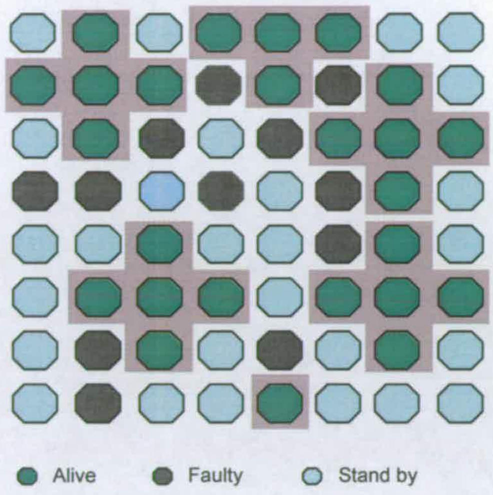


Figure 4.4: Potential combinations of valid (groups of 5 "alive" PEs forming a cross) solutions

4.3 Introduction of Faults

It is a primary concern for a designer who tests a system for its ability to tolerate against faults induced into its hardware resources, to accurately specify the possible nature of faults that may occur and how these can be efficiently modeled into the system under test. This work presents a fault-tolerant platform that targets Space applications, where Single-Hard-Errors (SHEs) and multiple SHEs dominate. These errors occur when charged particles, which usually originate from radiation belts or cosmic-rays, lose energy by ionizing the medium through which they pass. Consequently, SHEs appear as a substantial deviation from the expected level values. In digital systems this is translated to single or multiple bit flips in memory elements.

Both the CPL and CTL have been subjected to different levels of SHEs in order to explore and demonstrate the capability of the overall architecture to compensate for failures. In order for the system to meet all the constraints imposed by fault-tolerant applications, it is imperative for all the selected (alive) PEs in the CPL to converge. One of the most destructive scenarios for the functionality of the system is to cope with stuck-at zero and/or one faults that occur on the input/output registers of the PEs in both layers (CPL and CTL). In the case of stuck-at zero faults, it will be an unattainable objective for the "alive" PEs to converge because one or more bits of the register representing the fitness-score may be stuck-at zero. As a conclusion, there will be a deadlock in the operation of the PGA preventing the "alive" PEs converging.

On the contrary, whenever a stuck-at one fault occurs on the memory cells that keep the fitness-score, the controller may erroneously believe that a PE in CPL has converged. However, even this worst-case scenario is not destructive for the system because this fault will not be absorbed by the adjacent PEs. This can be deduced from Figure 4.3 that describes the FSM of each PE. Specifically, after the *Exchange* state at which each PE exchanges both the best chromosome and its respective fitness-score, it subsequently performs another reproduction. Thus, it is impossible for an erroneous value affected by a stuck-at one fault, to propagate to the whole array. Therefore, the simulation patterns of this promising approach have been concentrated to the study of stuck-at zero faults. According to this approach each fault was individually injected into the system by pulling to logic zero the input/output registers of the faulty PEs.

4.4 Results Analysis

Initially, the average number of generations, which the PGA in the CPL needs to converge, was investigated. The obtained measurements correspond at different failure levels (0%, 15%, 30% and 40%) that have been injected in the CPL. These results were obtained assuming that the CTL is out of operation. The ultimate scope of this simulation approach is to present the crucial role of the CTL, as the amount of faults gradually increases. Figure 4.5 depicts the results of the first simulation scenario (CTL out of operation).

The conclusion of the first simulation is that the mean number of generations is relative not only to the amount of faults but also to the way these emerging inside the array. Figures 4.6, 4.7 and 4.8 depict the three distinctive configurations of the faults that were applied on the CPL. Based on the different scheme of faults that were simulated, it is apparent that the number of generations increases significantly when “alive” PEs are surrounded by faulty ones. This is more obvious in the third scenario that is depicted in Figure 4.8, where five PEs cannot take advantage of the fine-grained parallelism, as they operate individually.

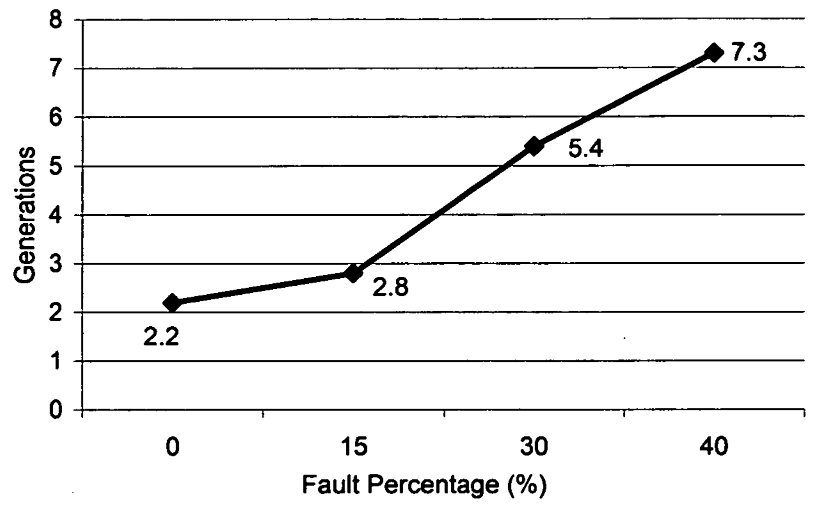


Figure 4.5: Mean number of generations that CPL needs to converge for different percentages of faults without the utilization of CTL

The majority of the simulations showed that in these worst-case scenarios the number of generations can substantially increase and thus the performance of the system may be inadequate for real-time applications. Subsequently, the same simulations were repeated, incorporating this time the functionality of the CTL. During these simulations, it is assumed that the CTL is immune to faults because initially the main scope is to present its beneficial effect over the CPL. Figure 4.9 depicts the results of the two different simulation approaches. The two graphs present how the performance of CPL is affected by the CTL in the presence of 4 different fault-levels. It is apparent that the existence of the CTL significantly improves the performance of the CPL. This enhancement becomes more evident whenever the number of faults increases. Apparently, this is the maximum boost that the CTL can provide because it has been ideally assumed that there are not any faults in the CTL.

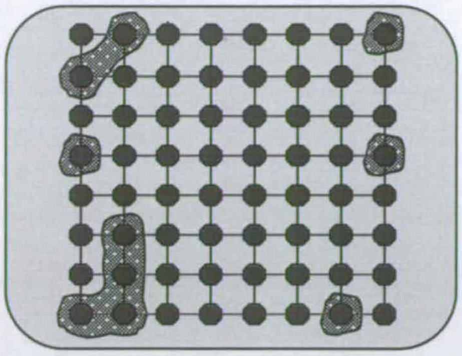


Figure 4.6: Simulation scenario for percentage of faults 15% (shaded PEs are faulty)

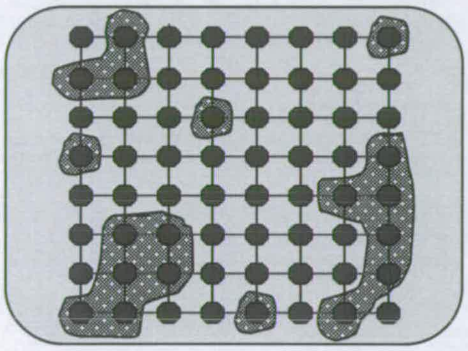


Figure 4.7: Simulation scenario for percentage of faults 30% (shaded PEs are faulty)

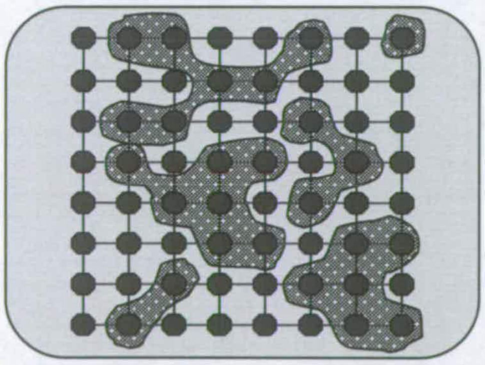


Figure 4.8: Simulation scenario for percentage of faults 40% (shaded PEs are faulty)

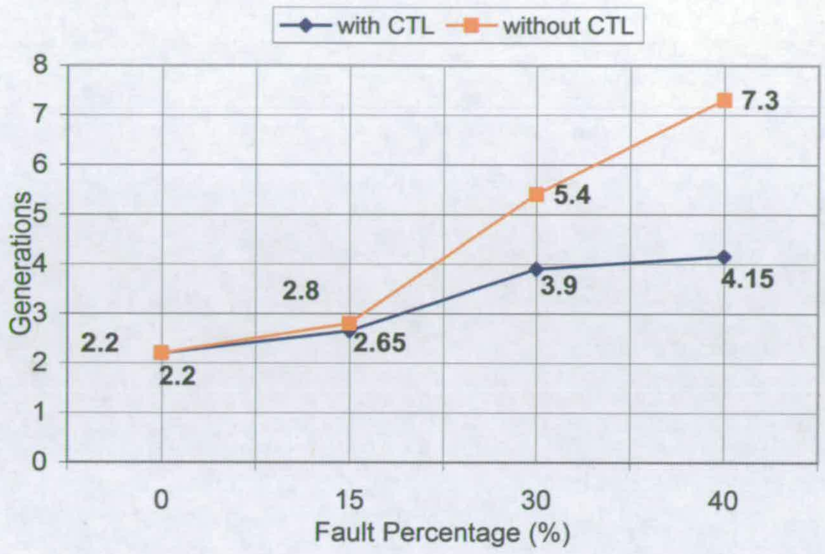


Figure 4.9: Mean number of generations that CPL needs to converge with and without the CTL usage

Moreover, it is also very important to investigate how the performance of the CTL is affected whenever a growing number of failures occur on the *controllers*. Figure 4.10 demonstrates that the mean number of generations that the controllers need to converge increases with a higher rate, for a CPL fault percentage larger than 20%. Therefore, after numerous simulations, it can be inferred that the CTL is able to cope with fewer faults than the CPL does. This is reasonable because CTL does not have a self-mechanism to isolate the PEs, which are surrounded by "faulty" ones. However, the simulation results proved that the CTL can compensate very efficiently for percentage of faults between 0 and 35%.

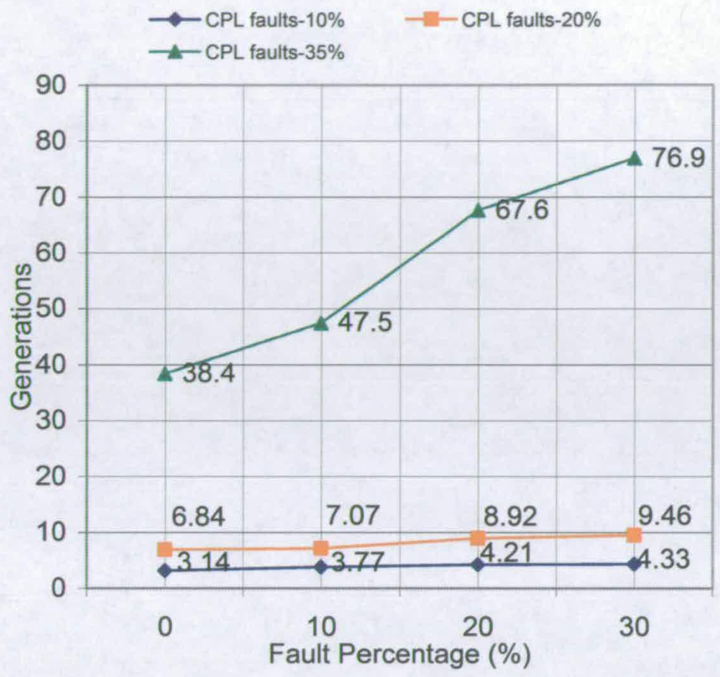


Figure 4.10: Mean number of generations that CTL needs for different percentage of faults in both layers

4.5 Conclusions

This section presented the VLSI implementation of a PGA that exploits massive parallelism and a two-dimensional array topology. Due to the massive parallelism that is employed, the overall architecture is able to maintain its functionality despite the presence of faults. According to the obtained results, the percentage of the occurring faults can securely reach 35% and 30% for the CPL and CTL, respectively. The obtained results showed that this architecture outperforms other conventional methods that deal with fault-tolerance, such as *Triple Modular Redundancy (TMR)*, which has traditionally been used for protecting digital logic from Single Event Upsets (SEUs) and/or SHEs in Space born applications. This approach uses three identical logic circuits performing the same task in parallel with corresponding outputs being compared through a majority voter circuit. In the simulation scenario depicted in Figure 4.8, there are 26 faulty PEs in the CPL, 27 "alive" PEs that efficiently contribute to the final solution and 11 PEs that are in a stand-by mode. On the contrary, TMR approach would require 27 (number of alive PEs that form the active part of the PGA) multiplied by the redundancy factor of 3. This implies that TMR methodology would require 81 PEs instead of 64 that are used for the implementation of the fine-grained PGA. Thus, it can be deduced that there is a reduction of 21% in the number of PEs that are required for providing a fault-robust reconfiguration mechanism.

Future work can be done in this architecture, in order to embed within the CPL the actual functionality of the electronics associated with the JPL/Boeing gyroscope. However, due to massive parallelism the whole implementation and integration would be very challenging and therefore, a smaller number of PEs must be considered.

Chapter 5
Custom-Reconfigurable FIR Structures

5.1 Introduction

This chapter presents five novel reconfigurable hardware substrates that are specially tailored for the implementation of the sensor's FIR filters, shown in Figure 2.2. Nowadays, FIR filters constitute the back-bone of most DSP systems. Based on the nature of the targeted applications, there are several issues that must be considered like *performance, physical area, power consumption* and the system's *robustness*. General purpose DSPs, such as TMS320 series from Texas Instruments do not provide adequate throughput and lack of reliability since they do not provide any redundancy. On the other hand, general-purpose FPGAs present high flexibility in terms of design re-usability but are not suitable for low-power applications. Therefore, there is a need in the industrial, electronic market for hybrid reconfigurable fabrics that constitute a trade-off solution between general-purpose DSPs and FPGAs [36]. This chapter ends with the latest achievement that is later used for the realization of the sensor's FIR filters. This reconfigurable hardware substrate exploits the majority of the beneficial outcomes from the previous four architectures and manages to meet all the requirements imposed by the JPL/Boeing gyroscope.

5.2 Zig-zag Interconnection Scheme between CALBs

5.2.1 Introduction

This section presents a custom reconfigurable VLSI architecture that targets the implementation of low-power medium/high order digital FIR filters. These are realized within a reconfigurable array that consists of *heterogeneous* programmable arithmetic-logic units, which have been optimized for *area, speed* and *power*. The design is a multiplier-less architecture, which is based on the primitive operation filtering (POF) [17] technique that presents superiority over canonic signed digit code (CSD). According to this technique the arithmetic and accumulation unit of the FIR filters is realized using only primitive operations such as shifts, additions and subtractions. Furthermore, a hybrid arithmetic approach is followed that makes the reconfigurable architecture able to cope with overflow events. Finally, an evolutionary strategy is introduced, which utilizes a radix-4 256-point single delay commutator (R4SDC) pipelined Fast-Fourier-Transform (FFT) module to calculate the frequency response of the evolved filters and provide the best possible configuration string to the reconfigurable hardware substrate.

5.2.2 System Architecture

From the hardware prospective, the architecture consists of 13 by 13 configurable arithmetic/logic units (CALUs) array. Unlike more macro-based approaches [7], this architecture is more fine-grained. It constitutes a heterogeneous array since it consists of three different kinds of CALUs. The aim behind this is to reduce the complexity and the configuration bits needed per CALU in order to increase the performance and decrease the physical-area and power-consumption of the array. Almost all of these are configurable and able to implement *right/left shifting* or *addition/subtraction* operations. The output of each CALU is fed into a synchronous register. This hardware approach composes a pipelined architecture, which increases data throughput. Moreover, there are a few blocks (switch boxes) that provide data routing in the horizontal/vertical level and store the output in a register. Considering the size of the reconfigurable hardware array, it can be deduced that the total number of registers is 169 and the maximum filter's order that can be achieved is:

$$\text{Taps} = 169 - (\text{width} + \text{height} - 1) \quad (5.1)$$

Width denotes the number of CALUs in the horizontal axis, while *height* denotes the number of CALUs that are attached on the vertical axis of the array. Thus, it can be implied that a 13 by 13 array structure may theoretically achieve a 144-order filter.

Figure 5.1 depicts the topology of the reconfigurable array and how the CALUs are connected to each other. It can be seen that a zig-zag interconnection scheme has been selected that can theoretically increase the efficiency of the architecture to utilize the majority of the synchronous registers. Figures 5.4 and 5.5 show the functional hardware description of the A/S and R/L shifter CALU, respectively. It can be deduced that by using the proper bit-configuration, both CALUs have the ability to output the input signal without performing any manipulation. Therefore, for different configuration schemes applied to the reconfigurable hardware in Figure 5.1, CALUs that belong in the same row but are not adjacent can communicate within a single clock cycle. Moreover, it is apparent that numerous data paths with different delay times can be created in different physical areas of the reconfigurable hardware.

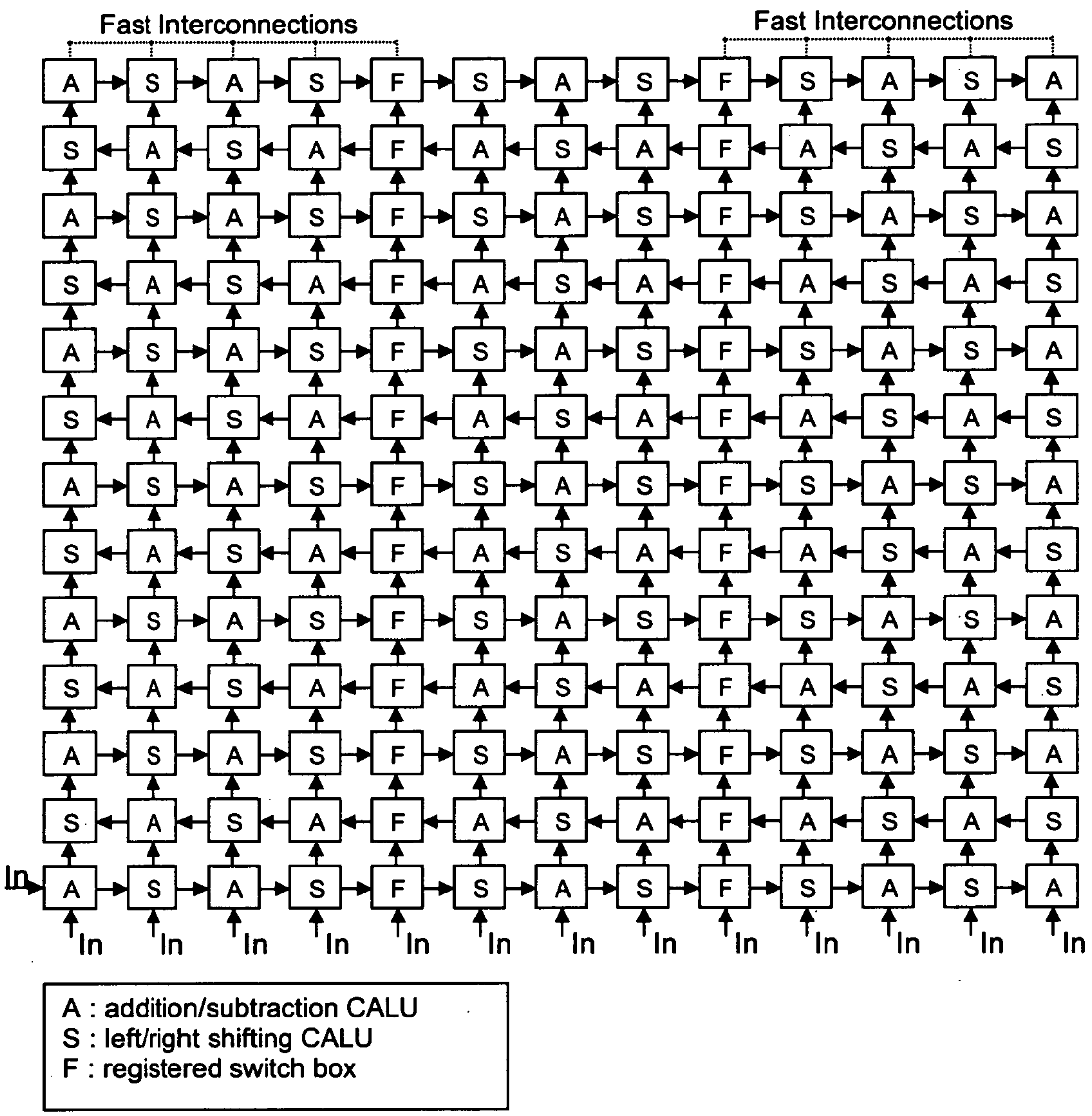

Fast Interconnections
Fast Interconnections
A
S
F
In
A : addition/subtraction CALU
S : left/right shifting CALU
F : registered switch box

Figure 5.1: Topology of the reconfigurable array

Each data path may compose a distinctive tap of the filter or alternatively two or more data paths can collaborate to produce one tap. In addition to this, the switch-boxes denoted with "F" define the critical path in the design and fix the timing violations that may occur when invalid configuration schemes are applied on the hardware substrate. Hence, the output of a switch-box is always stored in a register in both horizontal and vertical direction. Moreover, a 4 to 1 multiplexer, attached before each switch-box, is employed in order to establish the *fast interconnection*. This approach enables a switch-box to select each input from each one of the previous 4 CALUs. The terminology *fast* derives from the fact that the communication occurs in one clock cycle. Concerning the configuration process, the hardware needs 464 bits in total to be configured. The load of the configuration string occurs in parallel and hence only one clock cycle is needed for the hardware to change its configuration. This may be a drawback from physical-area point of view compared with a bit-serial loaded architecture but on the other hand, this approach decreases tremendously the power-consumption during configuration time. The achieved reduction factor is very important because the implementation of the filters is evolutionary based and hence many configuration schemes will be applied and evaluated until the GA's convergence. Figure 5.2 depicts the overall architecture which consists of a reconfigurable hardware substrate and a custom GA that is responsible for the evolution of medium/high order FIR filters. The proposed GA uses a 256-point R4SDC FFT module to calculate the frequency response of the evolved filter and compare it against the ideal one. Figure 5.3 depicts the three stages of the FFT architecture [15]. Each stage includes a commutator, a complex multiplier and a simpler butterfly element, which generates 256 outputs consecutively in 256 clock cycles. The function of the commutators, which comprise memory blocks, is ordering sequential input data into parallel output data for the data temporal separation requirement of FFT algorithms. From the first stage to the third one, the size of the commutators is 384, 96 and 24 words, respectively. The R4SDC architecture can be directly interfaced to a sequential word input without the requirement for input buffers.

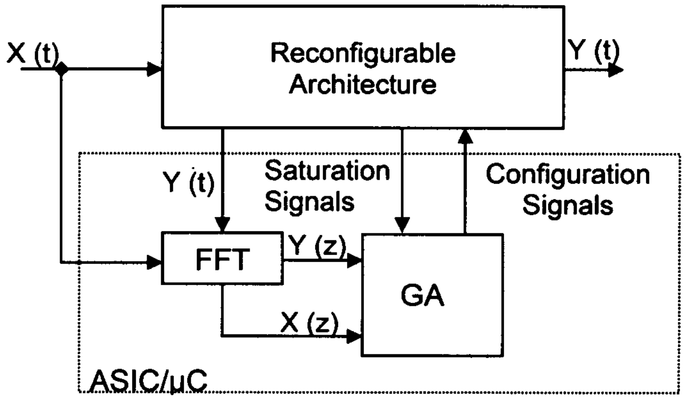


Figure 5.2: Overall system architecture comprising a reconfigurable hardware substrate and a GA performing evolution in frequency domain

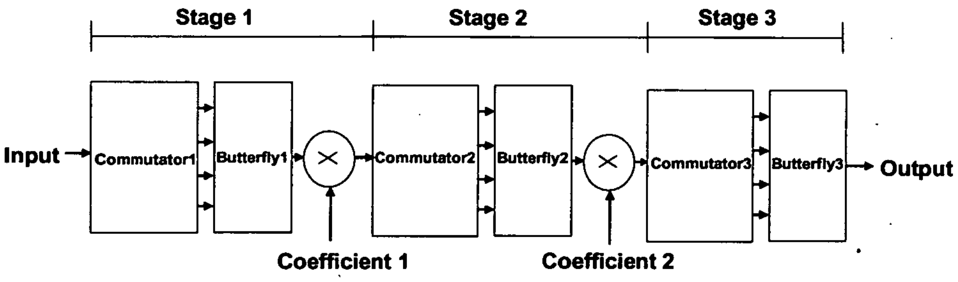


Figure 5.3: Three stages Fast Fourier Transform module

Finally, the fitness-function of the GA besides the frequency response, it also considers the number of saturation events that appear for a specific hardware configuration scheme. Based on this technique, the GA penalizes configuration schemes that produce extensive saturation events and the evolution of noisy filters is prevented.

5.2.2.1 Addition/Subtraction CALU

The addition/subtraction CALU (A/S-CALU) consists of both combinational and sequential logic. The combinational part includes a unit that performs either 20-bit addition or subtraction. As it can be seen in Figure 5.4, the inputs of A/S-CALU are determined by two multiplexers. The vertical output is fed into a synchronous register that stores the output of the A/S unit. Moreover, the horizontal output can be either the output of the register of the one of the two CALU's inputs. Thus, each A/S-CALU is able to obtain a different output value at each of the two output-ports. Furthermore, A/S-CALU is associated with some additional logic that copes with overflow phenomena by using a new arithmetic scheme that is a combination of fixed and floating point arithmetic, truncation, saturation and scaling. The detailed mechanism will be fully described in Section 5.2.3.

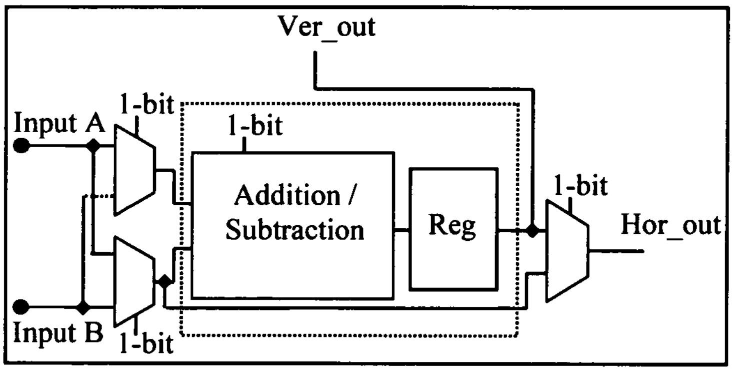


Figure 5.4: Addition/subtraction CALU

5.2.2.2 Left/Right Shifting CALU

Similarly to the A/S-CALU, the left/right shifting CALU (L/R-S CALU) has identical combinational logic but obviously different arithmetic unit. As Figure 5.5 depicts, the binary shifter itself is configured by 1-bit and based on its value, the arithmetic unit shifts on the right or left by 1 position the input value. Furthermore, the L/R-S CALU has two multiplexers that select the input and output of the CALU. The enable attached to the horizontal output can either select to forward the value stored in the register or automatically pass the selected input to the output, without performing any shifting operation.

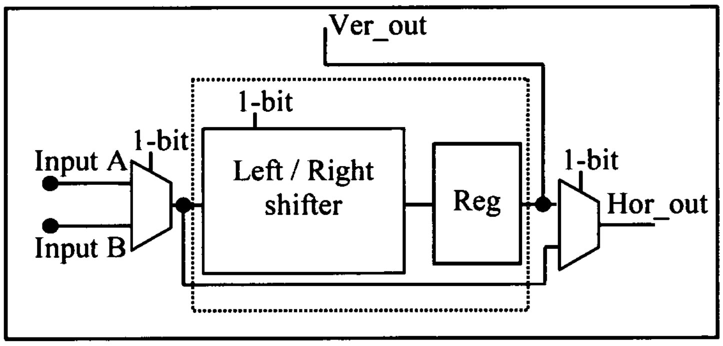


Figure 5.5: Binary L/R shifter CALU

5.2.3 Overflow Protection Mechanism

As the design consists of numerous arithmetic units that perform primitive operations, overflow is very likely to happen. Therefore, reconfigurable fabrics must be very carefully designed and incorporate additional logic in order to secure the precise behavior of the targeted applications. This section presents an evolutionary based mechanism that has been employed in this custom reconfigurable structure for the prevention of overflow phenomena. According to this mechanism, a protection-bit has been added in the 20-bit data bus and set whenever an overflow occurs. In each CALU, there is an overflow detection mechanism embedded inside the arithmetic unit and whenever overflow is to occur, the data is shifted on the right by 3 bits. After the shift, the sign bit is extended and the 3 less significant bits are truncated. This can happen only once for a specific data sample and the protection bit is latched high in order to indicate that there is a binary exponent of 3. Using this protection bit, the architecture can safely add and subtract data samples, which have different binary exponents. Hence, it can be deduced that the employed arithmetic representation is a kind of hybrid one that mainly acts like fixed-point but utilizes the simplest form of floating-point, as well. In addition to this, each CALU utilizes a 1-bit register that indicates whether saturation has occurred. This composes an alternative approach to handle the overflow. Based on this, if the result is out of the range that can be represented, it takes the closest value within the available range. Saturation takes place whenever the protection bit has already been latched to logic high and an overflow is going to occur again. Consequently, the GA that controls the configuration of the system considers the "saturation" signals and adds the equivalent penalty in the fitness-function in order that such configurations are avoided. Moreover, the role of the right shifts can also be other than performing a conventional division by 2 on a single number. The division of the whole coefficient-set by a constant scale factor inserts attenuation into the frequency response but it does not affect its shape. This is an alternative way to prevent overflow that is known as scaling. However, it is very important to determine how much scaling is necessary to avoid overflow because it results in a loss of the effective number of digits and therefore increases the quantization error. The process of scaling is controlled by the GA, which is responsible for finding the configuration scheme with the most acceptable frequency response.

5.2.4 Synthesis Results

Considering the order of the targeted filters, the reconfigurable substrate consists of 13x13 CALUs. The hardware has been synthesized using the Synopsys synthesis tool and UMC 0.18 μm technology cell library. The results depicted in Table 5.1, are very promising in terms of physical area comparing with industrial purpose FPGAs. Moreover, the timing analysis shows that the clock frequency of this architecture can be up to 100 MHz which is quite competitive even compared with ASIC applications.

TABLE 5-1: SYNTHESIS RESULTS OF THE RECONFIGURABLE HARDWARE (13X13 ARRAY)

Area (sq.mm)	A/S CALU	L/R-S CALU	SBox	Array (13x13)
Total cell	0.0074	0.0042	0.0025	1.139
Net	0.0015	0.0008	0.0004	0.326
Total	0.0089	0.005	0.0029	1.465

5.2.5 Power Analysis

The power consumption of this architecture has been calculated using Prime-Power tool (Synopsys), considering the standard-delay-format (SDF) information, which has been extracted after synthesis. Figure 5.6 depicts the results from the top module of the reconfigurable hardware that consists of 169 CALUs, from which 63 perform addition/subtraction operations, another 63 perform left/right binary shifting and the rest 41, are used to switch data samples.

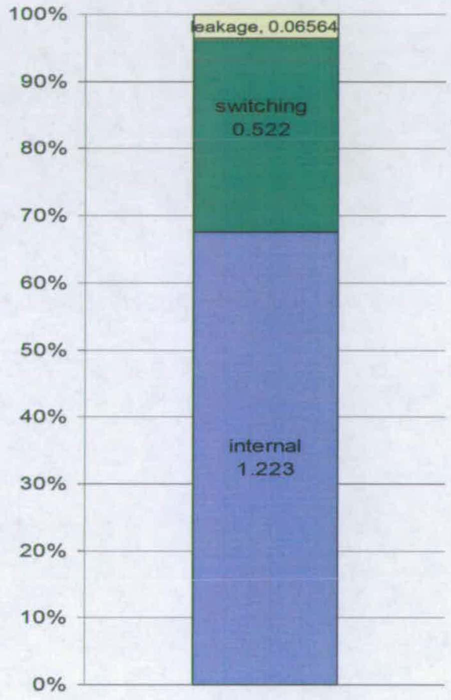


Figure 5.6: Power analysis of the reconfigurable hardware

The figure illustrates the power in mW, which is consumed per MHz of operation for the evolution of a 12-tap low-pass filter. It is important to mention here that the operational voltage of the design is 1.8 Volt, which implies that the design pulls current equal to 1mA/MHz. In Figure 5.7 is depicted the maximum power consumed by each CALU separately. The obtained figures are very significant because since each CALU is clocked through an AND gate, it can be switched-off in the case it does not on a certain configuration. Hence, the total consumption depicted in Figure 5.6 can be further decreased under certain circumstances.

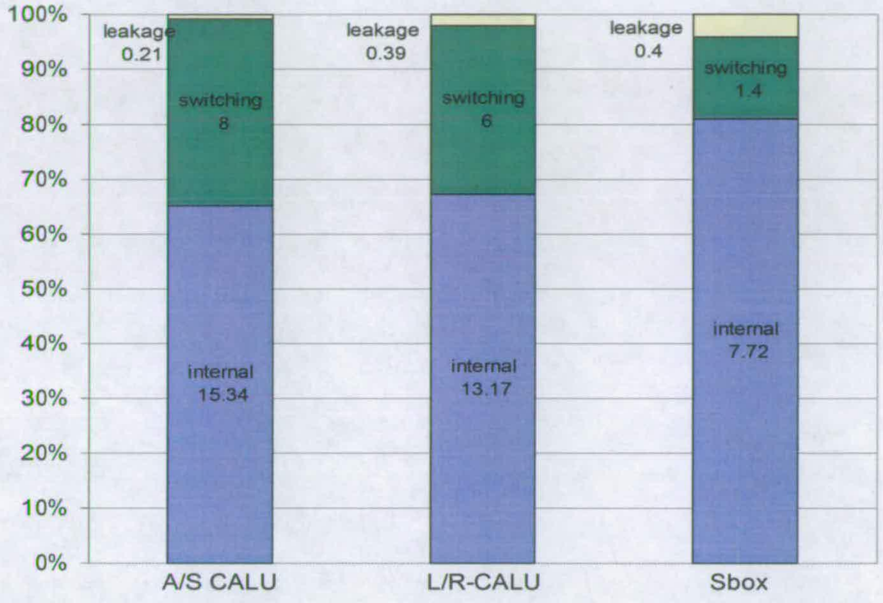


Figure 5.7: Power analysis of the 3 different CALUs

5.2.6 Conclusions

This section presented a novel low-power custom reconfigurable VLSI architecture that is tailored for FIR filters realization. The novelty of this architecture lies in several fields that are associated with the interconnection scheme of the 169 CALUs within a fine-grained architecture and the way the coefficients are retrieved from the same output, using paths with different delay times based on a certain configuration. Additionally, by using the POF technique and designing heterogeneous configurable blocks, the power calculation of the reconfigurable hardware substrate has been proved

proved through the paradigms of a low-pass and high-pass filter, using evolution in time and frequency domain. The obtained results demonstrate the physical characteristics of the reconfigurable substrate and the performance of the GA in successfully designing low order FIR filters. Finally, the power results of the reconfigurable architecture are compared with these of the AT6000 series FPGAs [9] and an algorithmically power-optimized reprogrammable FIR core [27].

5.3.2 System Architecture

The architecture of the overall system consists of a reconfigurable hardware substrate, which is based on the POF design principle and a GA module, which is responsible for providing the best solution for the realization and the autonomous adaptation of the FIR filters. The substrate comprises an array structure that consists of 4x12 CALUs. There are two kinds of CALUs. The first performs addition/subtraction (A/S-CALU) of 20-bit numbers and the second left/right shifting (L/R-CALU) operations. Each CALU contains a programmable delay chain, which can generate delays that vary from 0 to 3 clock cycles. The interconnection scheme between two adjacent columns is controlled by six 4:1 multiplexers. Moreover, a column of six registers is inserted, in order to prevent timing violations in the case of configurations that do not insert the proper number of delays. Figure 5.8 depicts the structure of the reconfigurable hardware in which the evolution of the filters takes place. At the right end, there is a single A/S-CALU that gathers all the outputs. This CALU can only create a delay of one clock cycle. Therefore, it can be deduced that the minimum path for a data sample to go inside the reconfigurable architecture can be 12 clock cycles, while the maximum path can be 3+12*3=39 clock cycles. Hence, theoretically this specific array is capable of realizing a FIR filter with maximum of 36 taps. Figure 5.9 shows the architecture of the two CALUs. It is apparent that the A/S CALU needs 3 bits to be configured, while the L/R needs four. Moreover, in the A/S CALU there is 1 bit that determines whether the block will perform addition or subtraction and 2 bits that define the delay. Respectively, in the L/R CALU there is 2-bit programming of the delay chain and 2 bits that define the kind of shifting (shift left by 1, shift right by 1 or 2, no shift). Hence, it is apparent from Figures 5.8 and 5.9 that it is a pipelined architecture capable of convolving the different samples by creating paths with different delay times.

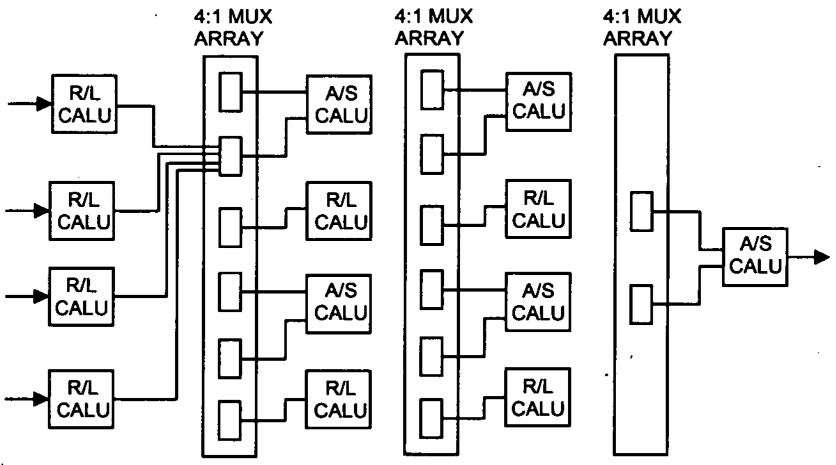


Figure 5.8: Custom reconfigurable hardware tailored for the implementation of FIR filters

The overall system architecture is the same as that depicted in Figure 5.2. The reconfigurable architecture informs the GA of the number of saturations which occur after each applied configuration. Moreover, the overflow prevention mechanism that is used in this architecture is identical to the one in Section 5.2.

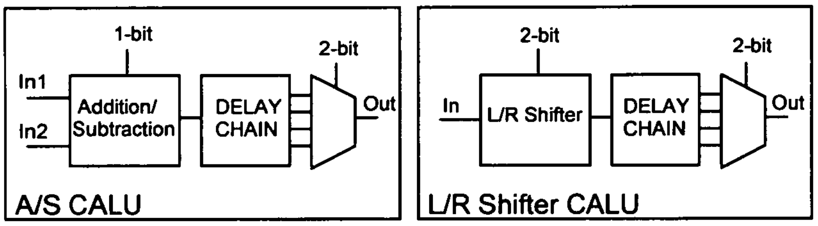


Figure 5.9: Hardware description of the two employed programmable CALUs

5.3.3 Genetic Algorithm

The two GAs that are used for the evolution of the coefficients and the frequency response, utilize the (μ+λ) methodology. According to this, the initial population consists of 50 chromosomes (parents) and is then extended to 100 by the addition of another 50 chromosomes (children). The children chromosomes are generated by crossover and mutation operations, which are applied on randomly selected chromosomes taken from the initial population. The rate of crossover is 80%, while the rate of mutation is 0.014% per gene. After evaluation and ranking the best chromosome is selected through elitism and in conjunction with another 49 survivors, which are selected through tournament selection, they compose the new parent chromosomes (population). Each chromosome consists of a 309-bit binary string, which encodes the operation of 25 A/S–CALUs, 24 R/L-CALUs and 70 4:1-multiplexers.

5.3.3.1 Evolution of Coefficients (time-domain)

The fitness function of the first GA is expressed as the sum of the square of the difference between the ideal and the evolved coefficients. Moreover, there is a penalty applied on configurations that realize filters with different numbers of taps. Finally, there is an additional penalty associated with configurations that cause extensive saturation. The mathematical equation that expresses the final fitness-score is given below:

$$Fitness = \sum_{i=0}^{taps-1} (coef_{ideal} - coef_{evolved})^2 - penalty_{tap} - penalty_{sat} \quad (5.2)$$

Because the reconfigurable architecture is capable of dealing with fractional coefficients as well, both coefficients in equation (5.2) are multiplied first by a constant number (x10^3), in order to achieve the appropriate accuracy.

5.3.3.2 Evolution of Frequency Response (frequency-domain)

The fitness function of this GA is expressed by the sum of the difference of the evolved and ideal squared magnitudes. The ideal squared magnitude is pre-computed according to the applied specification, while the evolved one (amplitude spectral density $|F(jw)|^2$) is calculated by dividing the square of the output $Y^2(z)$ with the square of the input $X^2(z)$. Hence:

$$|F(jw)|^2 = (\mathrm{Re}^2_{out} + \mathrm{Im}^2_{out}) / (\mathrm{Re}^2_{in} + \mathrm{Im}^2_{in}) \quad (5.3)$$

Because the FFT module is 256-point one, only the half frequencies have to be compared. Similarly with the previous methodology, configurations that produce extensive saturation are penalized. Therefore, the fitness-score is derived by the following equation:

$$Fitness = \sum_{i=0}^{127} (| F(jw) |^2_{ideal} - | F(jw) |^2_{evolved}) - penalty_{sat} \quad (5.4)$$

5.3.4 Simulation Results

The simulation results that were obtained, concern the realization of two FIR filters (a low-pass and a high-pass) by using evolution in the time and frequency domain. The simulation process consists of two components: the reconfigurable architecture, which is synthesizable and the GA unit (including the FFT module) that runs in a Verilog testbench environment that is not fully synthesizable. The characteristics of the filters, which were designed by the evolution of coefficients, are shown in Tables 5-2 and 5-3.

TABLE 5-2: LOW-PASS FILTER SPECIFICATION

Characteristics	Fixed point Equiripple Low-pass filter (13-taps)
Passband edge	0.1 rad/s
Passband Attenuation	3 db
Stopband Edge	0.15 rad/s
Stopband Attenuation	20 db

TABLE 5-3: HIGH-PASS FILTER SPECIFICATION

Characteristics	Fixed point Equiripple High-pass filter (9-taps)
Stopband edge frequency	0.01 rad/s
Stopband attenuation	20 db
Passband edge frequency	0.1 rad/s
Passband attenuation	3 db

5.3.4.1 Results in time-domain

In order to obtain the transfer function of the evolved filter, the impulse response is applied to the architecture at the beginning of each evaluation followed by 39 (maximum path delay) samples which are zero. The word length of the achieved coefficients can be up to 20-bit. Moreover, because a number of negative coefficients are used, a 2's compliment encoding is required. Figure 5.10 depicts the transfer function of the ideal and the evolved low-pass filter. It is apparent that the evolved filter has similar pass-band characteristics and that the stop-band of the evolved filter presents the same attenuation with the ideal one. However, the GA evolved a symmetrical 9-tap filter, instead of a symmetrical 13-tap one (ideal). The GA retrieves the best solution after 3500 generations. The process for the realization of the high-pass filter is exactly the same as before. Figure 5.11 illustrates the comparison in the transfer function between the evolved filter and the specification. The GA achieves the best solution after 6300 generations and corresponds exactly to a 9-tap filter. However, as it can be seen, there is a small deviation in the stop-band and pass-band edge frequency.

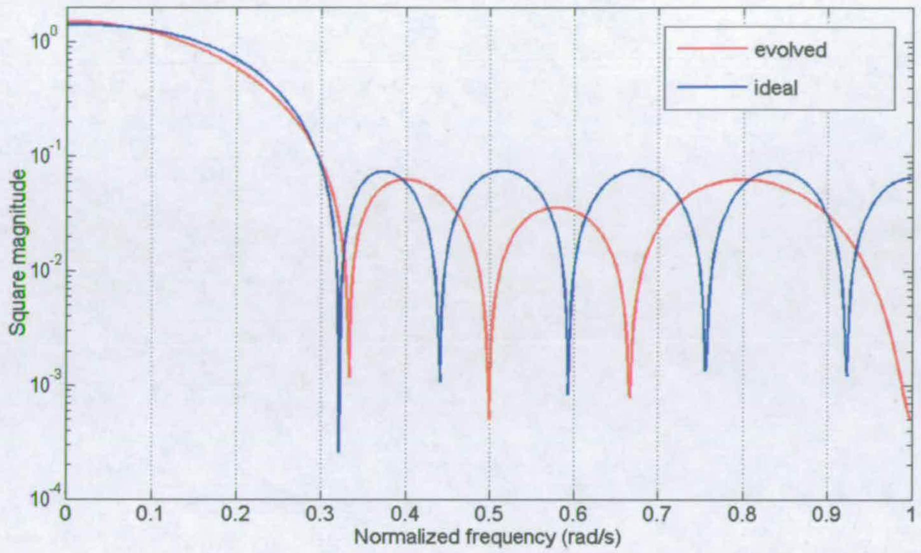


Figure 5.10: Transfer function of the 13-tap low-pass filter

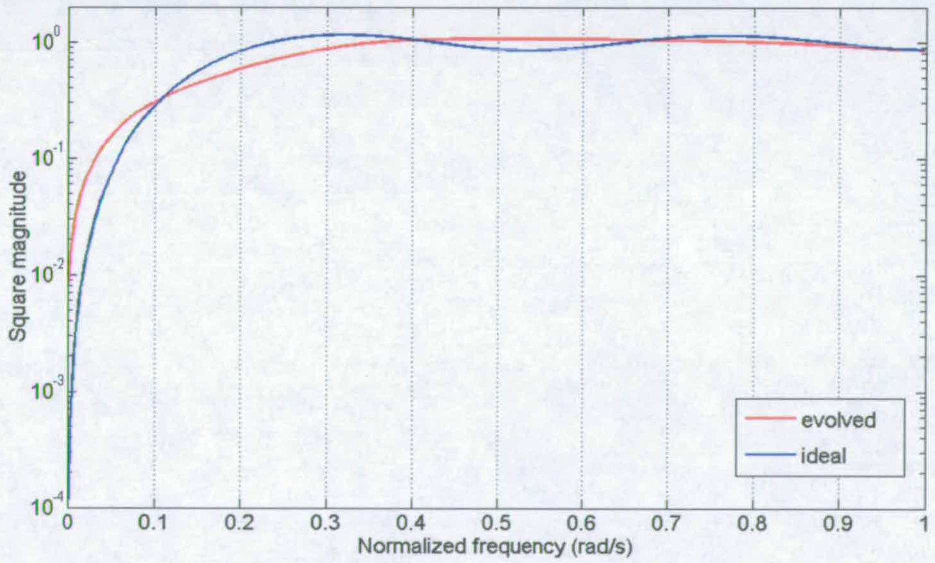


Figure 5.11: Transfer function of the 9-tap high-pass filter

5.3.4.2 Results in frequency-domain

This methodology presents superiority over the previous one because it calculates the magnitude from the frequency response of the input and output, when real data is fed in to the reconfigurable architecture. Therefore, a more accurate estimation can be achieved in comparison with previous work reported in [109], [112] concerning overflow phenomena that generate noise in digital filters. Moreover, results prove that the GA requires much fewer generations to converge with respect to the previous methodology. However, it has the drawback that a single generation needs more computational time due to the FFT calculations.

It has been mentioned in section 5.3.2 that the maximum number of taps that can be achieved by this architecture is 36. Therefore, considering the 256-point FFT, 220 randomly selected data samples are fed in to the reconfigurable architecture. The remainder of the 256 is padded with zeros. The frequency response of the input is calculated only once, while the frequency response of the output changes in each generation.

Figure 5.12 illustrates the comparison between the evolved low-pass filter and the applied specification. The actual number of generations needed is only 72. This reduction in the number of generations in comparison with the previous low-pass filter realization is quite sensible, considering the fact that the GA is not attempting to precisely match the value of each coefficient, but instead it tries to evolve a filter that has the same spectral density with the one applied in the specification. After the design of the filter, the coefficients of the filter were obtained by applying the impulse unit to the reconfigurable architecture and verifying the achieved magnitude in Matlab. Table 5-4 summarizes the results. It is apparent from the evolved coefficients that the filter is not linear phase.

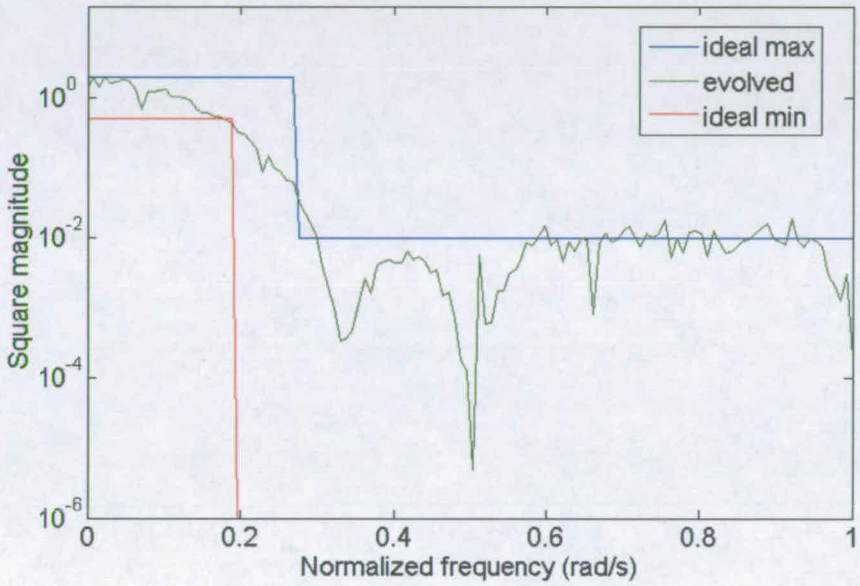


Figure 5.12: Magnitude of the evolved low-pass filter

For the design of the high-pass filter the GA needs 204 generations in order to find the optimal solution. In Figure 5.13 is the magnitude of the evolved filter is depicted in comparison with the ideal filter. Similarly to the previous example, Table 5-5 shows the coefficients of the evolved filter.

TABLE 5-4: COEFFICIENTS OF THE 8-TAP LOW-PASS FILTER

Coefficients	Value
C_0	-0.125
C_1	-0.129
C_2	-0.203
C_3	-0.254
C_4	-0.207
C_5	-0.203
C_6	0.129
C_7	-0.078

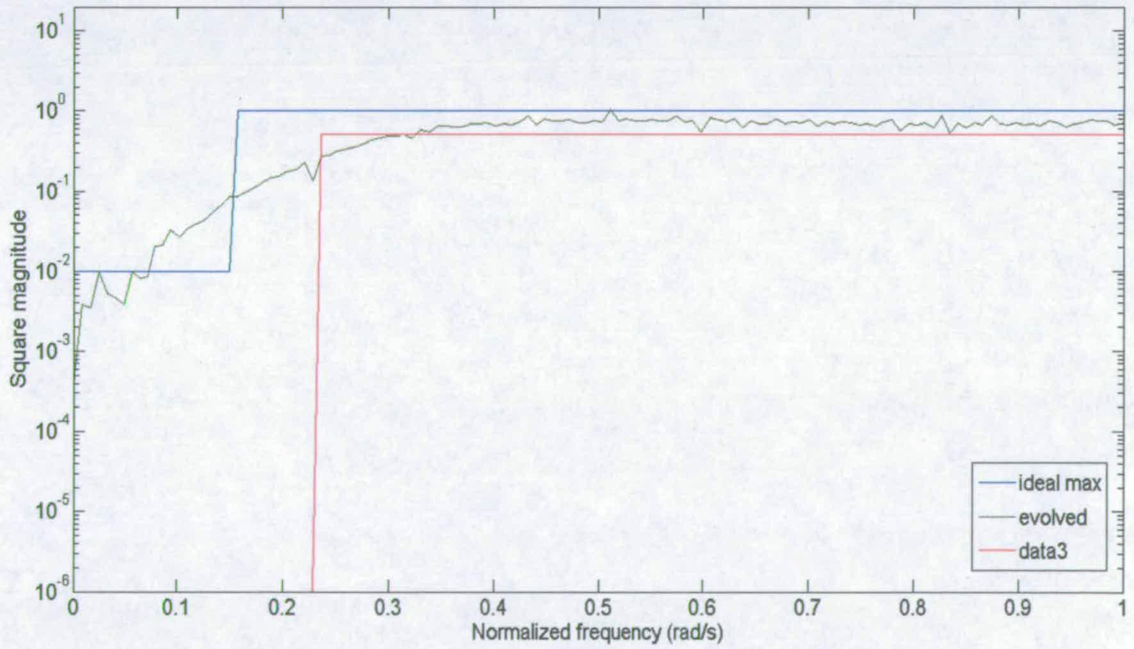


Figure 5.13: Magnitude of the evolved high-pass filter

TABLE 5-5: COEFFICIENTS OF THE 12-TAP HIGH-PASS FILTER

Coefficients	Value
C_0	0.0098
C_1	-0.0117
C_2	-0.0215
C_3	-0.0019
C_4	-0.0058
C_5	-0.1152
C_6	0.6328
C_7	-0.2520
C_8	-0.2340
C_9	-0.1289
C_{10}	0.0313
C_{11}	0.0313

5.3.5 Synthesis and Power Analysis

A reconfigurable 4x12 array has been synthesized using the Synopsys synthesis tool and utilizing the UMC 0.13μm technology cell library. The post-layout netlist was then employed in order to calculate the power consumption of the reconfigurable architecture by running a simulation with 1000 data samples both for the low-pass and high-pass filter. Table 5-6 summarizes the area results of the synthesized design.

TABLE 5-6: SILICON AREA OF THE RECONFIGURABLE HARDWARE

Area (sq. mm)	A/S CALU	L/R CALU	Reconfigurable Array (4x12)
Combinational	$4.73* 10^{-3}$	$3.6* 10^{-3}$	0.2613
Sequential	$2.08* 10^{-3}$	$2* 10^{-3}$	0.1087
Interconnections	$1.2* 10^{-3}$	$1*10^{-3}$	0.1028
Total chip	$8.01* 10^{-3}$	$6.6* 10^{-3}$	0.4728

Figure 5.14 depicts the power consumed by the evolved low-pass and high-pass filters, which have 8 and 12 taps, respectively. The power is measured in mW and corresponds to the clock frequency equal to 1MHz. It is important to mention that the operational voltage of the design is 1.08 Volts, which implies that the reconfigurable architecture pulls current equal to 0.32 mA and 0.34 mA for the low-pass and high-pass filter, respectively.

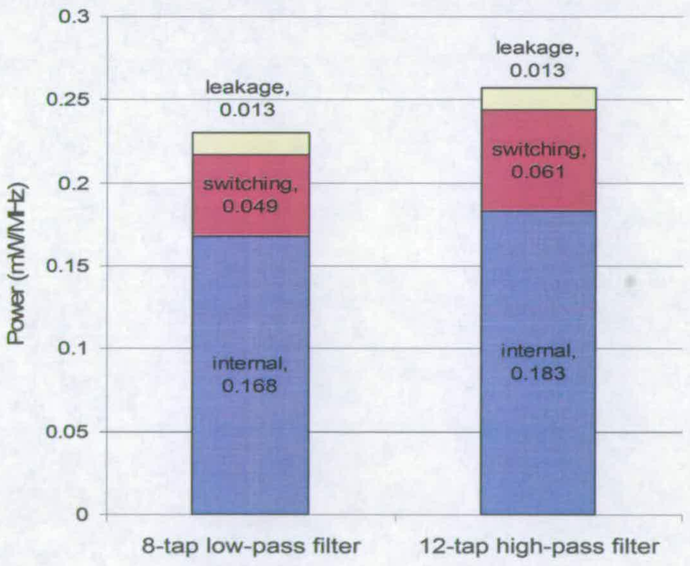


Figure 5.14: Power analysis of the evolved filters

It is interesting to examine the power that is consumed by each CALU because they are clocked through AND gates and therefore some CALUs can be switched-off, when they do not participate in a certain configuration and further reduce the total power consumption. In the example of the high-pass filter realization, the utilization of the CALUs is 81%. This implies that there are 9 CALUs (5 adders/subtractors and 4 left/right shifters), which can be switched-off without affecting the filters behaviour. Figure 5.15 depicts the power results concerning the two kinds of CALUs and the 4 to 1multiplexer, respectively. Consequently, Table 5-7 summarizes the power consumed by the reconfigurable hardware. This is compared to the power consumed by AT6000 series FPGAs [9], specifically for the realization of FIR filters. Based on the power that the 12-tap filter consumes and on the utilization of the CALUs, the power consumption for the 16, 24 and 32-tap filter has been estimated.

Moreover, the evolved filters are also compared against an algorithmically power-optimized reprogrammable 128-tap FIR filter [27] implemented in the System Level Integration group that belongs in the University of Edinburgh. It comprises an ASIC implementation that exploits efficient algorithms in order to reduce power. Specifically, the first algorithm decomposes individual coefficients into two sub-components, such that a part can be produced using a single shift operation leaving another one with a reduced word-length to be applied on the multiplier.

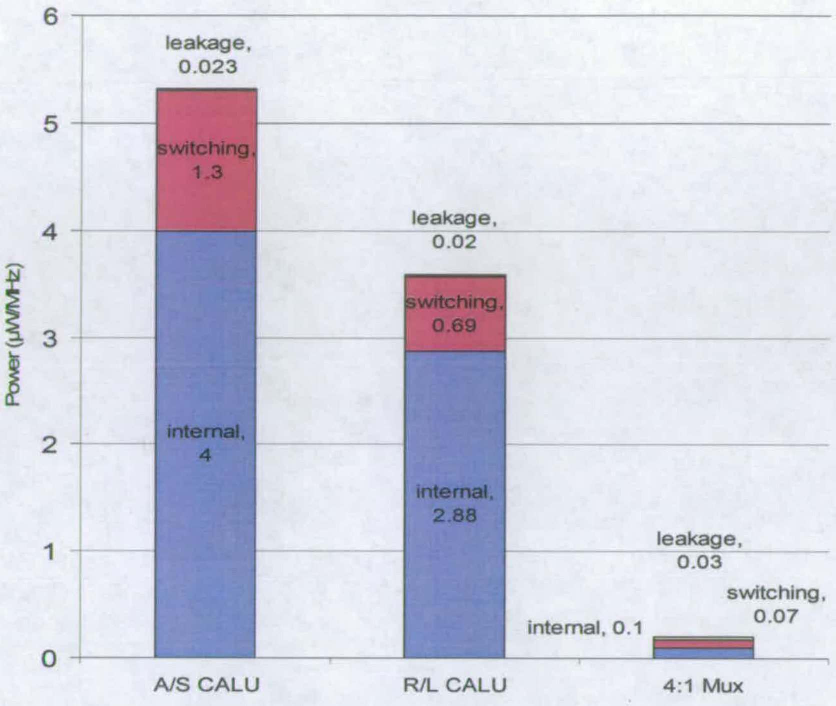


Figure 5.15: Power analysis of the CALUs

As a result, the effective switched capacitance decreases on the input of the multiplier and coefficient buses. The second algorithm reduces the switching activity not only at the multiplier inputs but also on the address and data buses, by processing multiple data samples at the same time, rather than one at a time. The filter has been synthesized using the same technology cell library (UMC 0.13μm), with the proposed reconfigurable architecture. The power results of this comparison are also depicted in Table 5-7. Finally, analysis of the work in [72] reveals that the ASIC core that has been designed to implement the filtering tasks of the JPL/Boeing gyroscope consumes

4.5mW for a 128-tap filter (ASIC operating frequency is equal to 37 MHz), by using 0.25 μm technology cell library and 2.5 core voltage supply.

TABLE 5-7: POWER COMPARISON BETWEEN THE RECONFIGURABLE ARCHITECTURE AND SIMILAR FIR IMPLEMENTATIONS WITH ASIC AND FPGA TECHNOLOGIES

FIR Filter (mA/MHz)	AT6000	Reconfigurable Architecture	ASIC
8-tap	1.048	0.21	$3.3*10^{-3}$
12-tap	-	0.24	$5*10^{-3}$
16-tap	1.29	0.35	$6.6*10^{-3}$
24-tap	1.94	0.47	$10.1*10^{-3}$
32-tap	2.62	0.58	$13.3*10^{-3}$

5.3.6 Conclusions

This section reveals an EHW architecture that targets a very power-optimized reconfigurable substrate to accomplish the control and filtering tasks of the JPL/Boeing gyroscope. The obtained simulation results emphasize the capability of this platform to realize FIR filters by using GAs in both time and frequency domain. Moreover, it has been proven that the evolution of the amplitude spectral density of the filter is more efficient than evolving the coefficients themselves directly, considering the number of generations that the GA needs to converge. This improvement is translated to a reduction of approximately 97% in the number of generations for both the realization of the low-pass and high-pass filter. Finally, the obtained power results classify the presented reconfigurable hardware substrate in an intermediate category between FPGA and ASIC technology. Specifically, it presents a reduction in power that is about 68% over the AT6000 series FPGAs and an increase that comes up to 98% over an ASIC implementation of a programmable FIR filter.

5.4 FIR Architecture with Chain Interconnection Scheme between CALUs

5.4.1 Introduction

The EHW architectures, which have been presented so far in Sections 5.2 and 5.3, suffer from low-accuracy and the EAs, which have been employed to guide the filters' evolution, require a substantial amount of time to converge.

The performance of the employed EAs is constrained by deceptive problems, such as *pleiotropy* (a single gene may simultaneously affect several phenotypic traits) and *epistasis* (one gene masks the phenotypic affects of another) [48], which prevent the algorithm reaching a global maximum within a time period that satisfies the time limitations of the targeted application. A typical example can be found in [43] where the evolution of the targeted filter can either take a significant number of generations or the evolved filters may not present very high accuracy. The phenomenon of *pleiotropy* occurs when the same hardware resource affect the realization of two or more coefficients, while *epistasis* occurs when a change in the control-bit of a multiplexer negates a part of hardware that was responsible for a part of the coefficient-set. In the paradigm of filters' evolution the phenomenon of *pleiotropy* is more destructive because a single change in control-bits of a piece of hardware may affect two or more coefficients in a contradictive manner and therefore it is infeasible to resolve a configuration that accurately satisfies all the involved coefficients. Another problem that conventional evolutionary techniques encounter in the evolution of filters is their inability of drastically changing the hardware configuration after a significant number of coefficients, within the coefficient-set, have been partially satisfied.

5.4.2 System Architecture

This section reveals an EHW architecture that mainly targets the reduction of the deceptive problems (*epistasis* and *pleiotropy*) that the previous architectures (Sections 5.2 and 5.3) encounter. Hence, the first aim was to reduce the multiplexers within the design that produce an alternation in the behaviour of the circuit when their control-bit changes. Therefore, after investigating the manner of filter evolution in the previously presented EHW architectures, a new topology (chain interconnection scheme) has been introduced that mainly targets to minimize the deceptive effect of *epistasis* and *pleiotropy* and evolve more accurate filters.

Similarly with the previous architectures, it combines both the adaptability of reconfigurable devices and the powerful search methods of GAs. This amalgam provides distinctive characteristics such as autonomous reconfiguration and recovery from faults. Compared with previous research achievements [43] in this field, this design presents several advantages. Firstly it can evolve each part of the FIR filter (delays, arithmetic and accumulation unit). Secondly, the interconnection scheme of the CALUs enables the power cut-off of the hardware components that do not participate in the filter's evolution. Moreover, the proposed architecture, compared with previous research [109], [112], composes a fine-grained architecture that can be efficiently used for low-power and fault-tolerant applications. Figure 5.16 presents the overall architecture, which consists of the custom reconfigurable device that is adequate to realize higher-order FIR filters and the GA that guides the reconfiguration process and provides the best possible configuration. Similarly to the overall architecture, presented in Section 5.2, this one utilizes a radix-4 256-point FFT to perform filter evolution in the frequency domain.

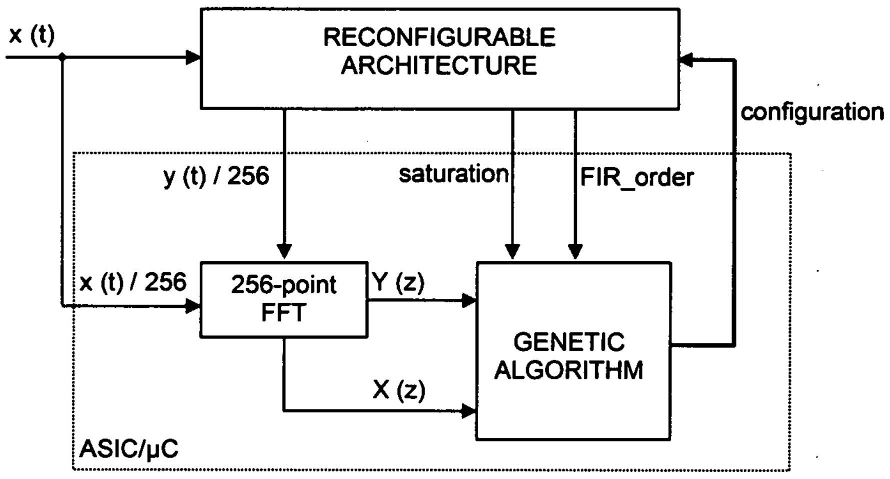


Figure 5.16: Overall EHW architecture comprising a custom reconfigurable hardware and an evolutionary driven reconfiguration mechanism

Moreover, random data samples X(t) are used because one of the purposes is to control and avoid overflow and saturation. Consequently, the output Y(t) of the reconfigurable architecture is obtained and then the frequency response of the input X(z) and output Y(z) is calculated in order to obtain the square magnitude of the system. The only difference compared to that in Section 5.2 is that the GA also considers the order of the filter that is evolved. Hence, whenever there is a substantial deviation between the ideal and the evolved order, the GA applies a penalty in the fitness-score of these hardware configurations.

5.4.2.1 Reconfigurable Architecture

The main target of this reconfigurable architecture is to employ a more customized physical mean for the evolution of the digital filters. Figure 5.17 depicts the interconnection scheme of programmable elements. Similarly to the previous designs, this reconfigurable architecture is based on the POF technique, which is suitable for area and power optimization. Hence, the arithmetic unit of the system consists of the same A/S and R/L-S CALUs that have been introduced in Section 5.2. According to this hardware topology, the circuitry between two subsequent A/S CALUs is called Configurable Arithmetic/Logic Block (CALB) and is illustrated in Figure 5.17 within the dashed frame.

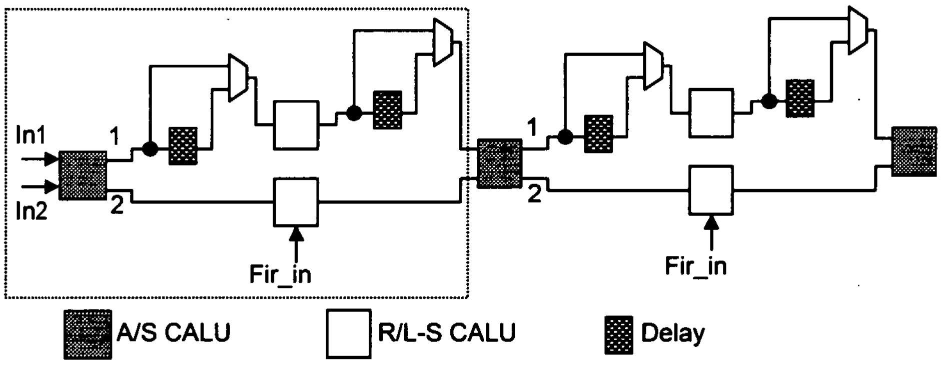


Figure 5.17: Functional description of the custom reconfigurable hardware

Moreover, non-adjacent CALBs are connected to each other via 2:1 multiplexers, according to Figure 5.18. This additional interconnection scheme is beneficial because it enables the system to control the order of the filter and/or avoid faulty components by providing alternative routing solutions. Furthermore, Figure 5.18 shows that CALB5 is connected directly with CALB1 and that the intermediate CALBs are over-passed. It can also be seen that the employed power control mechanism utilizes the control-bit of the two multiplexers to control the enable signal of CALBs 2, 3 and 4. Hence, according to this configuration, the architecture switches-off these CALBs to reduce the power-consumption of the evolved circuit.

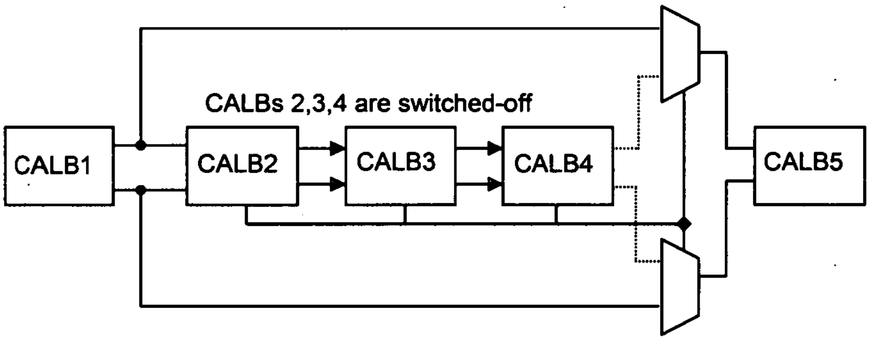


Figure 5.18: Power control mechanism – CALBs 2, 3 and 4 are switched-off because CALB 1 is virtually connected to CALB 5. The control-bit of the two multiplexers control the operation of CALBs 2, 3 and 4

5.4.2.2 Genetic Algorithm

The problem of identifying the binary configuration for the reconfigurable hardware, based on a certain specification and the random appearance of faults, is uniquely suited to EAs. Each individual that composes the population of the GA, encodes the control bits of the different reprogrammable elements (A/S CALU, R/L-S CALU and multiplexers). The length of each individual is 528 bits. The selection of applying evolution in frequency domain has the advantage that the converging process speeds-up. Especially for online reconfiguration the GA does not have to accurately match the same set of the specified filter coefficients but just to evolve a filter with equivalent amplitude spectral density. In Figure 5.16 it can be seen that the GA also considers the order of the evolved filter after each generation. The GA has a priori knowledge of the order of the targeted filter. Moreover, based on this order, it is given a freedom to design filters that have similar order within a range that varies according to a pre-defined number. Hence, when the GA evolves a filter that has more taps than the number provided by the specification, a penalty is introduced in the fitness-function. Moreover, based on the required frequency response the GA is also provided with a rough estimation of the filter's order. This approach proved to facilitate significantly the convergence of the GA since it reduces the search space of the algorithm. Moreover, the GA, which was employed for the evolution of the low-pass and pass-band filters, utilizes the (μ+λ) methodology. According to this, the initial population consists of 50 individuals, which after the genetic operations (crossover and mutation) is extended to 100. The fittest chromosome is selected through elitism and in conjunction with another 49 chromosomes, which are selected through tournament selection compose the new population. The rate of elitism (1%) and the kind of selection have been intentionally chosen in order for the GA to find a creative balance between exploration (searching of new solutions) and exploitation (taking advantage of previously found solutions). The fitness-function is given in equation 5.5 and slightly differs from the equation 5.2 in

Section 5.3. It calculates the ratio between the evolved and the ideal squared magnitude but also introduces a penalty for configuration schemes that produce substantially different filter orders.

$$Fitness = \sum_{i=0}^{127} (|F(j\omega)_{ideal\_i}|^2 / |F(j\omega)_{evolved\_i}|^2) - penalty_{sat\_i} - penalty_{fir\_order\_i} \quad (5.5)$$

The ideal squared magnitude is pre-computed according to the targeted filter specification, while the evolved one (amplitude spectral density $|F(j\omega)|^2$) is calculated by dividing the square of the output $Y^2(z)$ with the square of the input $X^2(z)$. The evolved amplitude spectral density is given by equation (5.6) below:

$$|F(jw)|^2 = (\mathrm{Re}^2_{out} + \mathrm{Im}^2_{out}) / (\mathrm{Re}^2_{in} + \mathrm{Im}^2_{in}) \quad (5.6)$$

Where Re_{in}, Im_{in} represent the real and imaginary part of the data samples applied on the input of the reconfigurable architecture and similarly Re_{out} and Im_{out} represent the complex output. Finally, equation 5.5 includes another two objectives, which are associated with the occurrence of saturation events and the order of the evolved filter, respectively. In both cases the GA applies a penalty on the fitness score in order to control saturation and prevent the implementation of filters with a substantially different specification.

5.4.3 Injection of Anticipated Faults

As was mentioned in Chapter 2, the JPL/Boeing gyroscope has been designed for airspace application where harsh environmental conditions prevail, such as radiation and temperature variations. Single-Hard-Errors (SHEs) and multiple-bit SHEs [26], [59] dominate in electronic devices, which are exposed to highly radiated environment. The interaction of radiation particles with VLSI circuits can produce charge decomposition in critical regions of the circuit. This can in turn lead to bit-reversal errors. JPL/Boeing gyro is meant to operate in such environmental conditions and therefore the design of its electronics must be tolerable to behavioral deviations caused by such conditions. For the purpose of this simulation, the anticipated SHEs are modeled as permanent stuck-at faults, which occur on the memory elements of the system. Moreover, the reconfiguration can be distinguished in two different types (off-line and on-line reconfiguration), depending on the initial conditions of the evolution and the timing limits of the system for recovering its performance in the presence of faults.

5.4.3.1 Off-line Reconfiguration

The computation time of this approach usually takes longer than the online one because it does not employ any *a priori* knowledge for the realization of the targeted electronics. Thus, the content of the population is randomly selected. In the simulation results that are presented in Section 5.4.4, it can be seen that offline reconfiguration is capable of coping with fabrication imperfections and faults that occur at the very early state of reconfiguration.

5.4.3.2 On-line Reconfiguration

Alike the offline reconfiguration, the online one incorporates knowledge of the design implementation. Hence, the initial population comprises the achieved population after the first convergence of the GA. Its scope is to make the reconfigurable device adapt in real-time with operational perturbations, which lie in environmental changes (temperature, pressure, etc.) or occurrence of faults after the initial realization of the electronic circuit.

5.4.4 Simulation Results

This section presents the simulation results associated with the realization of the low-pass (73-tap) and pass-band (60-tap) filters, shown in Tables 5-8 and 5-9. These filters have been designed in MatLab using the equiripple algorithm [70]. The three simulated scenarios correspond to 0%, 5% and 10% of faults. It must be mentioned here that the initial conditions (population content and seeds of the random number generators) for these three simulation cases were exactly the same. Figure 5.19 and 5.20 show the filters, which have been designed with the help of the GA for the two given specifications. From the results it is obvious that using the proposed overall architecture it is feasible to precisely achieve the desired specification.

TABLE 5-8: LOW-PASS FILTER SPECIFICATION

Sampling frequency	1 KHz
Pass-band edge	0.1 rad/s
Pass-band tolerance	1 db
Stop-band edge	0.1156 rad/s
Stop-band attenuation	35 db

TABLE 5-9: PASS-BAND FILTER SPECIFICATION

1st stop-band edge	0.1 rad/s
2nd stop-band edge	0.4 rad/s
Stop-band attenuation	20 db
1st pass-band edge	0.112 rad/s
2nd pass-band edge	0.388 rad/s
Pass-band tolerance	1 db

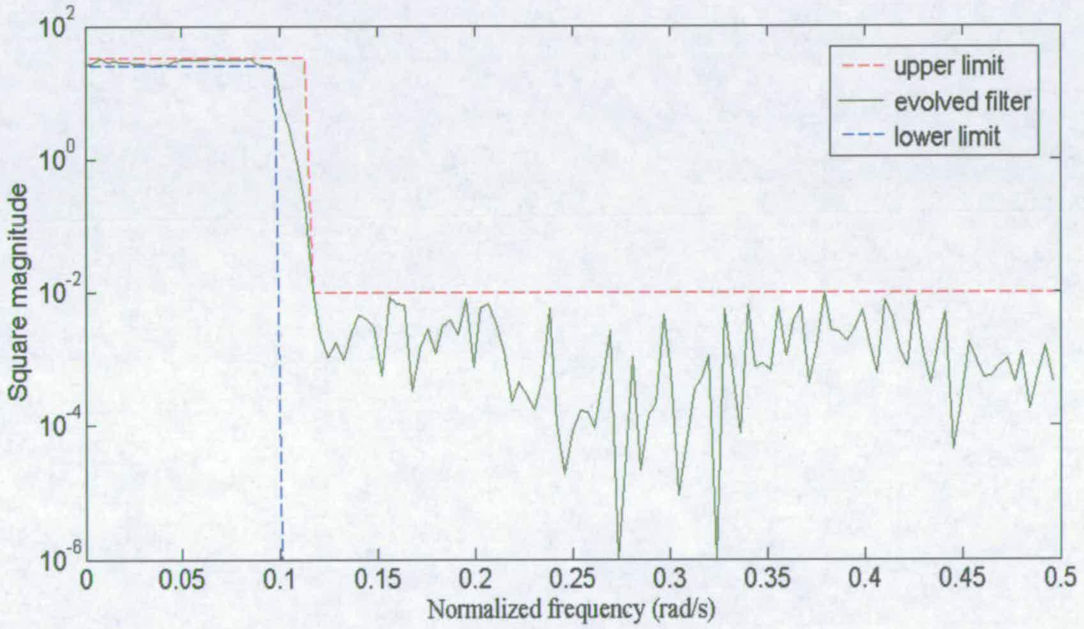


Figure 5.19: Low-pass filter evolution with 0% faults

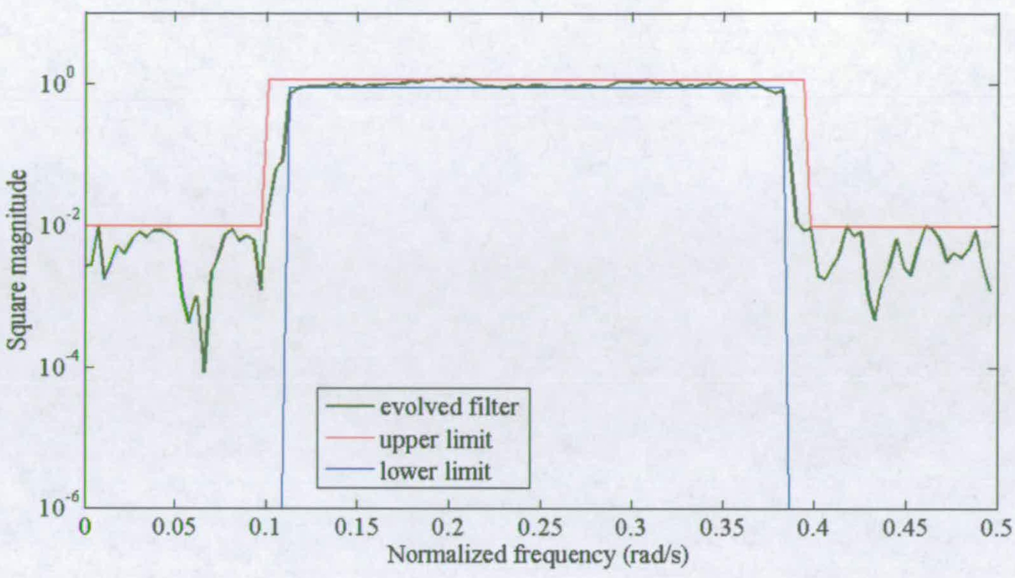


Figure 5.20: Pass-band filter evolution with 0% faults

5.4.4.1 Off-line Reconfiguration for Faults 5% and 10%

In this simulation scenario, the performance of the GA is monitored, when the pass-band filter specified in Table 5-9 is implemented within a faulty environment. Thus, faults equal to 5% and 10% were injected into the *user* and *configuration* memory of the reconfigurable architecture [90]. The effect of an error in the *configuration* memory forces for example, an AS-CALU to perform only addition or subtraction, or always select the same input, while an error in the *user* memory produces a deviation in the content of a register. Figure 5.21 demonstrates how the system manages to approach the desired functionality in the presence of faults. A comparison with a fault-free simulation scenario (Figure 5.20) is also depicted. In order to obtain an objective comparison the injected faults in the 5% scenario compose a subset of the 10% one. The result of the three graphs represents the best run out of the three that have been performed. This run achieves higher accuracy within fewer number of generations compared to the other two successful runs. Finally, the practical outcome of this figure is that the reconfigurable architecture is able to implement FIR filters even if numerous faults emerge, after the fabrication process and/or when the system starts to implement a new circuitry from scratch.

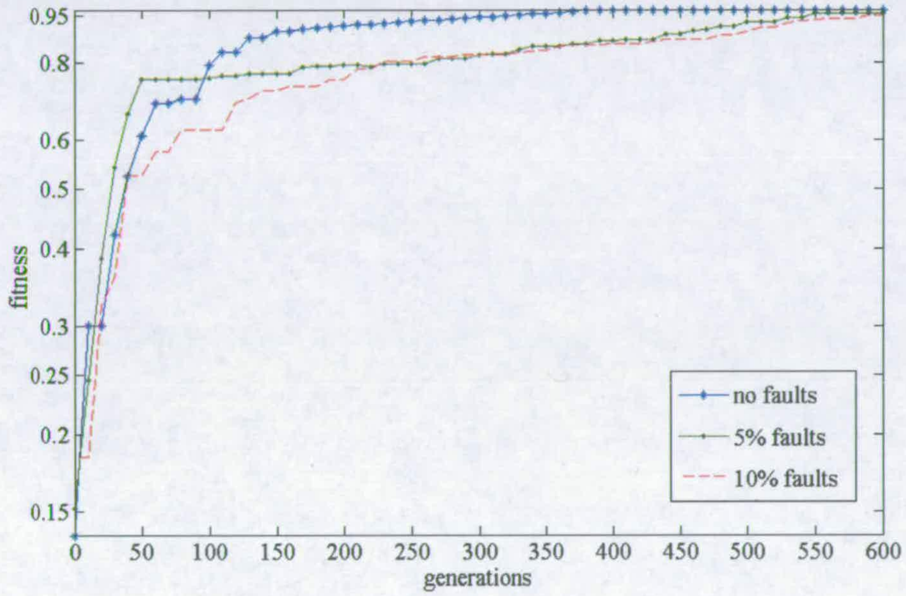


Figure 5.21: Evolution process of the pass-band filter for faults 0%, 5% and 10%

5.4.4.2 On-line Reconfiguration for Faults 5% and 10%

The purpose of the on-line reconfiguration is to demonstrate whether the system is able to recover its functionality in real-time. Similarly to the off-line reconfiguration, the same number of faults was injected. Figures 5.22 and 5.23 demonstrate two different simulations, which illustrate how efficiently the system can recover the functionality of the pass-band and low-pass filter, respectively.

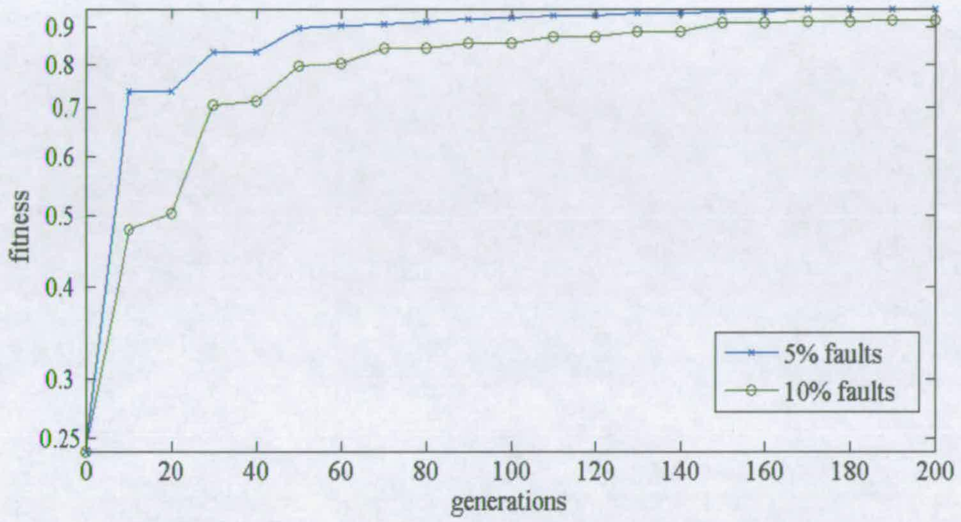


Figure 5.22: On-line pass-band filter recovery for faults 5% and 10%

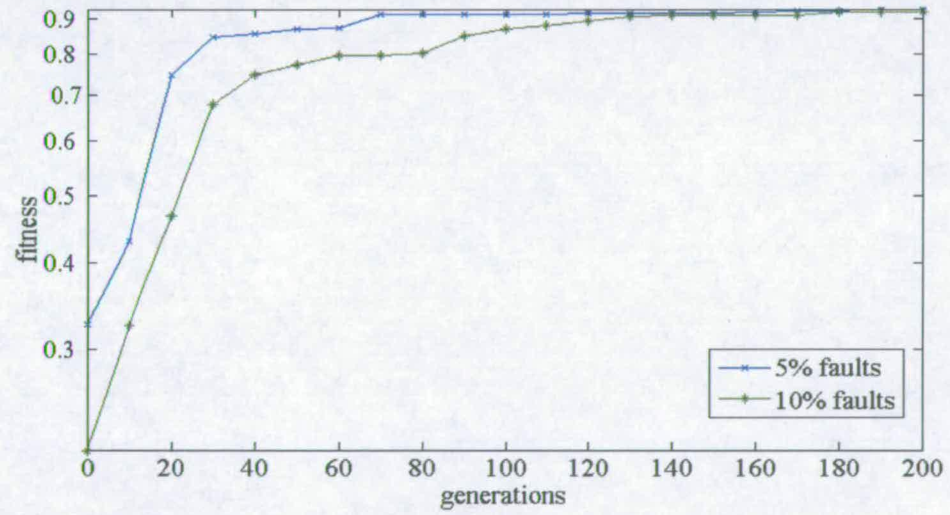


Figure 5.23: On-line low-pass filter recovery for faults 5% and 10%

The two simulations run a limited number of generations (200) because the time required for the on-line reconfiguration must be shorter than the update rate of the JPL/Boeing gyroscope. Again these results represent the best run out of three successful simulations that use different initialization parameters for the GA.

5.4.5 Synthesis and Power Analysis

The reconfigurable hardware substrate consists of 40 CALBs. The design was synthesized using the Synopsys synthesis tool and employing the UMC 0.13μm technology cell library. Based on these criteria, the die area of the design occupies 0.67mm^2. Subsequently, power analysis was carried out using the Synopsys Prime-Power tools. Interconnect parasitics were taken into account in order to obtain a more realistic result. From Figure 5.17, it can be deduced that power savings can be achieved even in active CALBs. This is feasible because the control-bit of the 2 multiplexers inside each CALB can control the enable-bit of the matching registers. Initially, both filters consume approximately the same power 0.3767 mW/MHz. However, Figure 5.24 shows the power that can be saved by the low-pass and pass-band filter, respectively. This power corresponds to the dynamic power since the leakage power still will remain after disabling the unused components. Thus, considering the power analysis for both filters, the low-pass filter consumes 0.2767mW per 1MHz of operation, while the pass-band consumes 0.256mW/MHz. Finally, the achieved power results are very competitive compared with the reconfigurable architecture presented in Section 5.3 and the ATMEL 6000 series FPGAs (Table 5-7).

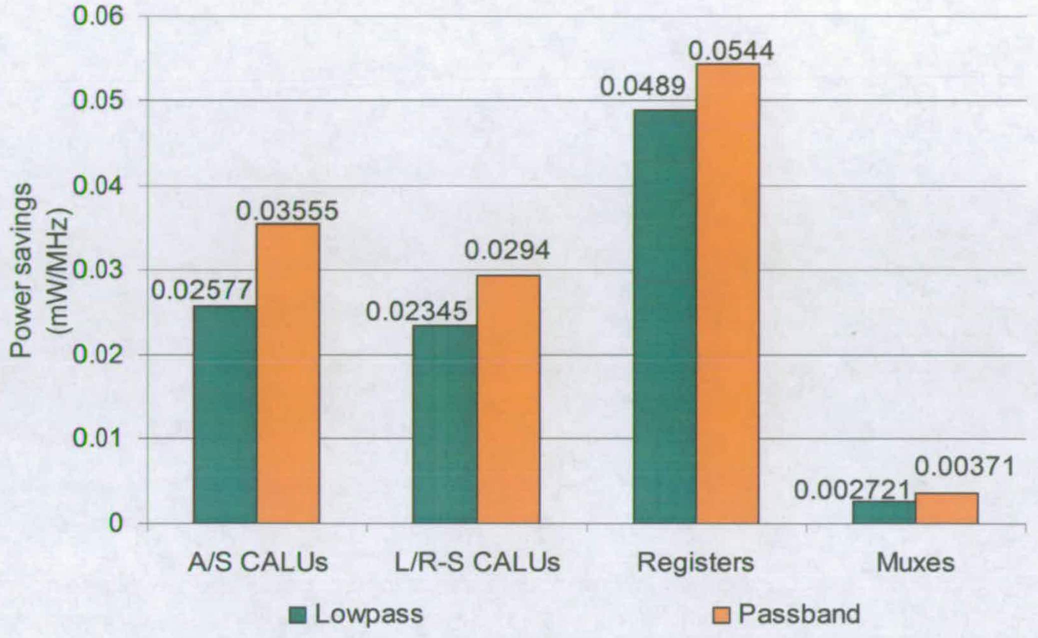


Figure 5.24: Power savings for the low-pass and pass-band filters

5.4.6 Conclusions

This section presents an EHW architecture that is capable of evolving higher-order filters than the first two architectures in Sections 5.2 and 5.3. Moreover, it has been proved that it is adequate for the implementation of electronics that require continuous adaptation due to environmental perturbations (temperature, radiation, pressure, etc). The obtained results show that the system provides enough adaptability and robustness to cope with an occurrence of faults of up to 10%. Moreover, power analyses of the simulated netlists prove that the customized design secures significantly lower power consumption compared to industrial general-purpose FPGAs.

5.5 Development of a Folded-Transposed form FIR Filter

5.5.1 Introduction

The reconfigurable architecture that has been presented in Section 5.4 comprises the first successful attempt to evolve more accurate and higher-order FIR filters that would be suitable for the JPL/Boeing electronics. Moreover, it has been proven that these filters are very low-power and that the reconfigurable architecture is capable of recovering their functionality in the presence of SHEs. However, there are a few drawbacks with the previous EHW architecture. The first limitation is that evolution in frequency-domain requires high computational effort. In [115] authors present the evolution of medium order filters in the frequency-domain and suggest that the evolution of higher order filters becomes impractical due to the complexity of their architecture and the computational time needed for evolution in the frequency-domain. The second drawback is the achieved moderate accuracy due to the round-off error produced by the FFT module [117] and the filter [41] in order both to fit the data to the prescribed word-length. Finally, the evolved filters are not always linear-phase, which is a prerequisite for the FIR filters that compose the gyroscope's electronics. Therefore, this section presents an EHW architecture that performs evolution in the time-domain and adopts alternative techniques, such as the folded-transposed form FIR filter implementation and a custom non-stochastic search algorithm, in order to shorten the time that the GA needs to converge. This architecture differentiates from previous research work on reconfigurable FIR filter architecture [16], [58]. Although these architectures are able to change on demand the specification of a filter, they are vulnerable to the presence of hardware faults because of their granularity in critical components (multiplier and accumulation block) is not adequate for exploiting redundancy. Finally, compared with [43], this architecture is capable of coping with faults that emerge even in the accumulative unit of the filter.

5.5.2 Overall Architecture

The overall architecture consists of a reconfigurable hardware topology that aims at accomplishing the filtering tasks of the JPL/Boeing gyroscope and a custom search algorithm, which provides the binary configuration on the hardware. Both the hardware topology and the GA are combined in order to implement filters with minimum hardware resources. Figure 5.25 depicts the block diagram of the overall architecture. It can be seen that the GA during the evolution of the filters, considers the impulse response H(z), which corresponds to the response of a discrete-time system to a unit-impulse u(n). The impulse response is represented by the coefficients of the evolved filter and is compared with the ideal impulse response (set of coefficients). Finally, after the implementation of the targeted filter, the GA provides the representative binary configuration and some additional control signals, which concern the accumulation unit of the filter.

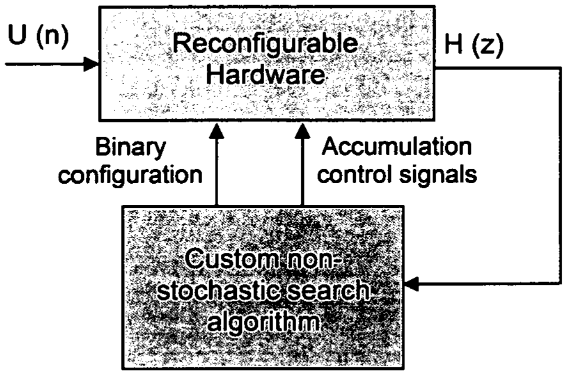

U (n)
Reconfigurable Hardware
H (z)
Binary configuration
Accumulation control signals
Custom non-stochastic search algorithm

Figure 5.25: Block diagram of overall architecture

5.5.2.1 Reconfigurable Hardware Topology

The topology of the hardware has been specifically selected for the implementation of the transposed form of FIR filters. This technique has the advantage that only half the coefficients have to be evolved and therefore the search algorithm needs considerably less time in order to obtain the binary configuration. The design consists of two parts named multiplier and accumulation block. Figure 5.26 shows the interface between these two parts. It must be mentioned here that the multiplier block does not employ a dedicated multiplier but instead it uses the POF technique for realizing the targeted coefficients. This approach increases the granularity of the multiplier block and hence redundancy can be exploited more efficiently. In the proposed architecture, each coefficient can be implemented with different combinations of addition/subtraction (A/S) and left/right (L/R) shift operations. Moreover, it may need different number of vertices in order to be implemented.

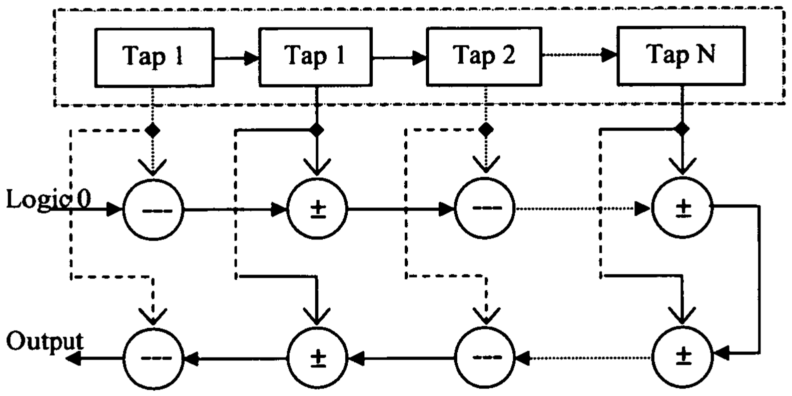


Figure 5.26: Reconfigurable hardware topology

The concept of a vertex is shown in Figure 5.27 (circuitry included within the shaded box). The main target of this architecture is to minimize the number of the employed A/S units, as they consume considerably more power, compared with shifters and multiplexers. Hence, each vertex is able to connect to either one of the three previous adjacent vertices in order to re-use former partial sums that facilitate the realization of subsequent coefficients.

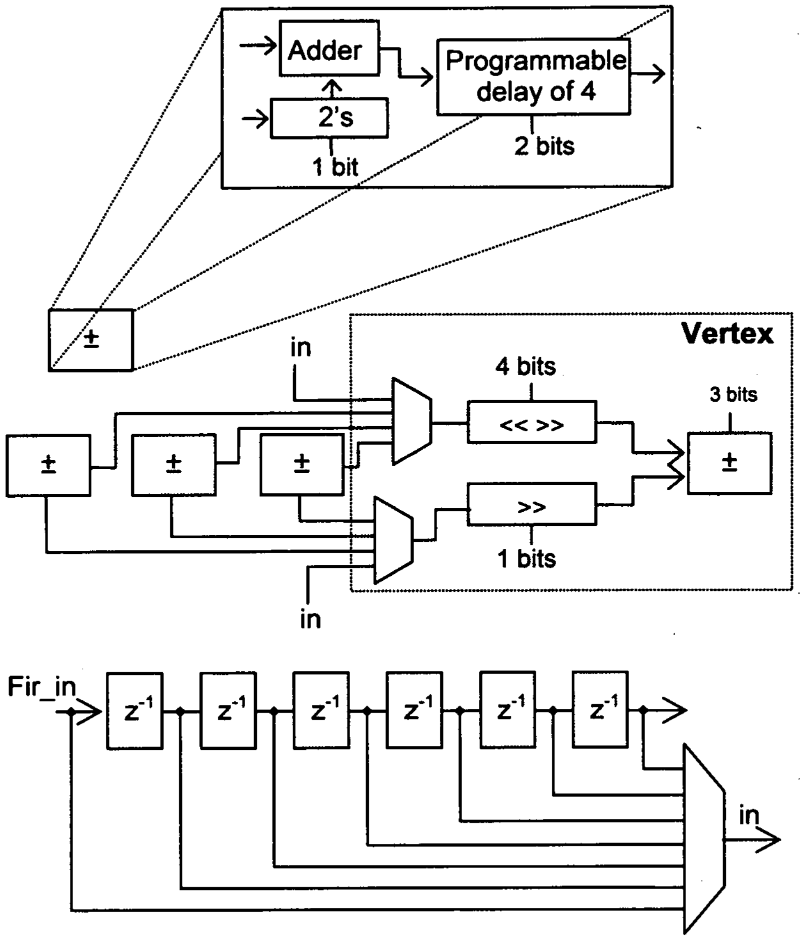


Figure 5.27: Hardware architecture and interconnection scheme

Figure 5.26 depicts that the first coefficient needs the hardware resources of two vertices and therefore only the second vertex is connected with the accumulation block. This mechanism is controlled by the GA and enables the architecture to be very flexible when system adaptation is required or faults occur on hardware components either in the multiplier or accumulation block. Based on the folded transposed form of FIR filters, the multiplication of each data sample with each coefficient must be performed at the same time. The design is a pipelined architecture and the multiplier block has been implemented so that each tap needs 6 clock cycles. Therefore, since each coefficient may need a different number of vertices, each vertex must incorporate a variable clock cycle delay in order to cope with synchronization problems. Figure 5.27 shows that each vertex includes a programmable delay of 4 clock cycles. Hence, if for example a coefficient employs two vertices, the delay on these two must be adjusted to give a sum of 6. In addition to this, the A/S units include a two's compliment module that determines whether addition or subtraction will be performed. Moreover, the architecture has been designed to realize either positive/negative and odd/even integer coefficients. Therefore, both L/R shift operations are needed. In Figure 5.27, the upper shifter is capable to either perform binary left or right shift by 8. On the other hand, the lower shifter performs binary left shift by one or it just forwards its input. This technique secures that any odd or even integer number can be realized. It must be mentioned here that decimal coefficient sets have first to be transformed into integer ones. On the contrary to the A/S units, the shifters are asynchronous circuits and their outputs feed the two inputs of each A/S unit, as it is depicted in Figure 5.27. From the last picture in the same figure (bottom end), it can be deduced that when the input of the FIR filter is selected to feed the A/S units, this must be consistent in time with the A/S unit. Therefore, there is a need for a programmable delay chain in order to evolve the correct timing. The output of this programmable delay chain is the input of the reconfigurable hardware.

Finally, Figure 5.28 illustrates the internal design of the A/S units, which are attached to the accumulation block. These are different to the A/S units of the multiplier block. Specifically, the control signal of the multiplexer is controlled by an adaptive mechanism, which collaborates with the search algorithm and determines whether the respective vertex on the multiplier block corresponds to a tap or it just constitutes an intermediate vertex. In the later case the horizontal input of the adder instantaneously feeds the output and neither addition or subtraction operation is performed.

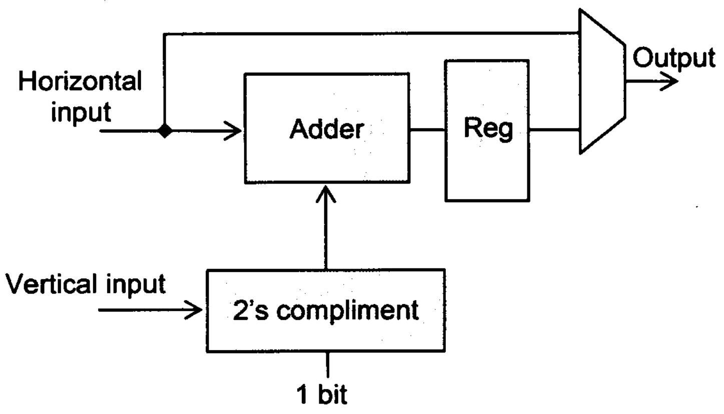


Figure 5.28: A/S unit in the accumulation block

5.5.2.2 Architecture of Custom Search Algorithm

The GA, which has been employed to provide the binary configuration on the reconfigurable hardware substrate, composes a custom design that aims at realizing FIR filters with the minimum possible adder-cost and speeding-up the time that is needed for the filter implementation. This aim has been achieved by following several techniques in both the reconfigurable hardware (each vertex re-uses former partial sums) and GA.

From the algorithmic perspective, genetic operations take place only on the parts of the chromosome that correspond to the currently realized coefficient and the three previous vertices. According to this approach the search algorithm gradually realizes each coefficient individually and not the entire coefficient set at the same time. Hence, at any instance the search area of the GA is minimized and consequently the realization time of each coefficient is reduced. Figure 5.29 depicts the encoding approach of each chromosome and shows how the GA assigns the coefficients to vertices.

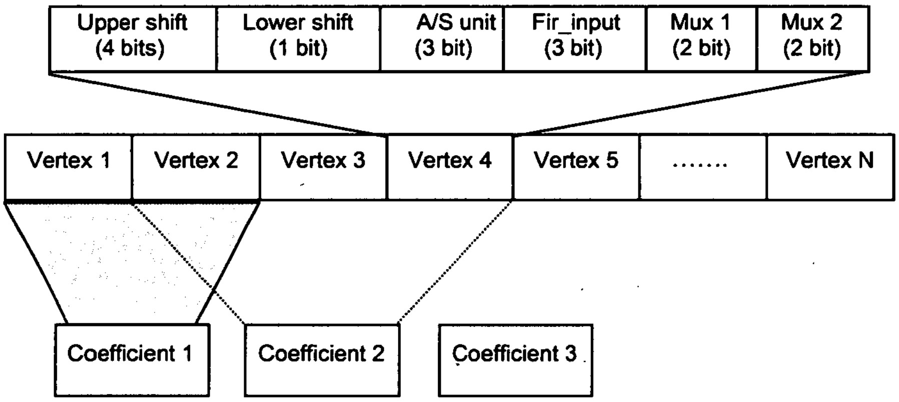


Figure 5.29: Encoding scheme for search algorithm

From the picture it can be deduced that the first coefficient utilizes two vertices, while the second realized coefficient employs the output of the second vertex plus two additional ones. During the realization of the second vertex, the GA performs genetic operations on vertices 1 to 4, since vertex 4 can utilize the output of each one of the three previous vertices. When evolving the third coefficient, the GA initially assigns to it one additional vertex (vertex 5) and it waits to see whether the coefficient 5 will be realized by utilizing vertex 5 and the partial sums of the vertices 2, 3 and 4. If the GA is not able to realize the third coefficient within a certain number of generations, then it assigns an additional vertex (vertex 6). Based on this approach, the GA ensures that the binary configuration does not employ redundant vertices for realizing a given coefficient set. Consequently, this has a significant impact on the power, which is consumed by the implemented FIR filter. Finally, the objective function (equation 5.7) of the GA is given as the summation of the ratios between the ideal and the evolved coefficients (C_{ideal} and $C_{evolved}$) as it is illustrated in the formula below. It can also be deduced that the maximum fitness-score for each coefficient is 1.

$$\text{Fitness} = \begin{cases} C_{ideal} / C_{evolved} & \text{if } C_{evolved} >= C_{ideal} \\ C_{evolved} / C_{ideal} & \text{if } C_{evolved} < C_{ideal} \end{cases} \qquad (5.7)$$

Moreover, only when the fitness-score is equal to 1 (first coefficient has been evolved), the GA considers the fitness-score of the second coefficient. This approach evolves the targeted coefficient-set in a gradual method and does not consider all the coefficients at the same time. This is another point that the proposed GA differentiates from previous research works [109], [112], in which all the sub fitness-scores of the different coefficients are considered from the start of the evolution, even if none of them have been precisely realized. The conventional methodology has been proved to create hardware dependencies between coefficients, which are not adjacent within the coefficient set and thus, a small change in the binary configuration that improves a certain coefficient may negatively affect other non-adjacent coefficients. Therefore, this effect introduces significant delay in the convergence process of the GA or even worse forces the GA to get stuck on sub-optimal solutions. Finally, the GA collaborates with an adaptive mechanism, which controls the control-bits of the A/S units in the accumulation block. More specifically, this controller initially configures the A/S unit to operate as accumulation node, considering that the realization of a specific coefficient needs one vertex. Subsequently, it considers the fitness-score, which can be obtained with this specific configuration and if for a certain number of generations the fitness-score does not achieve an acceptable threshold, then the controller assigns an additional vertex in the realization of the coefficient by turning-off the previous A/S unit and making accumulation node the next A/S unit.

5.5.3 Simulation Results

The capability of this architecture to accomplish the electronics of the JPL/Boeing gyroscope is demonstrated through a 41-tap low-pass filter. Based on the targeted specification the pass-band stops at 0.2 rad/s, the stop-band starts at 0.284 rad/s and there is 50db attenuation in the stop-band. Initially, the realization of the filter is performed in an error free environment. Subsequently, different scenarios of SHEs are injected either in the multiplier or accumulation unit of the realized filter.

5.5.3.1 Error-free Simulation of FIR Filter

Due to the folded transposed form that the proposed architecture supports, the GA has to realize only 21 coefficients. According to the simulation results the filter needs 45 A/S units in order to accomplish this specific coefficient-set. This implies that on average the realization of each coefficient takes 2.14 vertices, which is quite impressive, considering that the coefficient-set consists of 16 bit integer numbers. Finally, Figure 5.30 depicts how precisely the evolved filter matches the specification. The green graph represents the frequency response of the evolved filter and it has been obtained in MatLab by applying the freqz command [69] to the evolved coefficient-set of the reconfigurable architecture. The blue one represents the ideal one, as it was generated in MatLab. The only weak point of this methodology is that the evolution time of the filter needs a significant number of generations in order the evolved circuit to achieve an acceptable level of accuracy. Hence, in this simulation the GA took almost 2100 generations to evolve the filter depicted in Figure 5.30 (green line). The long reconfiguration time is justified considering that the GA must evolve the correct timing (6 clock cycles) for each tap.

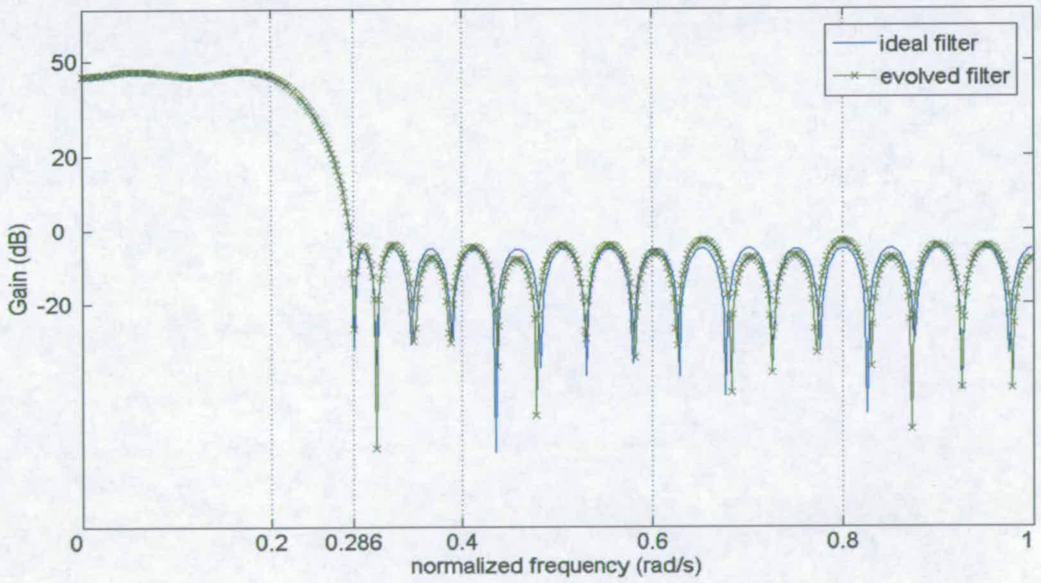


Figure 5.30: Comparison of the evolved and ideal filter

5.5.3.2 Adaptive Mechanism for Fault-Robustness

The proposed architecture employs an adaptive mechanism in order to recover from the occurrence of faults. Preliminary results show that the proposed reconfigurable design is capable of coping with SHEs, which occur either in user (registers) or configuration memory. According to this mechanism, there is a control unit embedded in the search algorithm that whenever the fitness-score of the GA cannot reach a certain threshold, it changes the control bits of the multiplexers that exist inside the A/S units in the accumulation block after a number of generations. In the first scenario of faults injection, SHEs occur in the *user* memory of the multiplier or accumulation block.

Figure 5.31 depicts how the controller modifies (lower graph) the configuration in the accumulation block and assigns an extra vertex-block in the currently realized filter. It can also be seen that a vertex-block consists of a vertex (multiplier block) and two A/S units (accumulation block). According to this specific scenario, the filter initially employs three vertices in order to realize a coefficient. However, because there are faults (modeled as stuck-at faults) in the third vertex-block, the fitness-score cannot overcome a satisfactory threshold and thus the controller changes the configuration and assigns an additional vertex block for the coefficient realization. Thus, after the reconfiguration only the A/S units in the fourth vertex-block accumulate and vertex 3 is isolated because vertex 4 selects only vertices 1 and 2 in order to realize the coefficient. Moreover, the architecture is able to recover even when faults occur in the *configuration* memory. A representative example is given in Figure 5.32, where the control bit of the multiplexer in the third A/S unit in the accumulation block has stuck-at logic 1, while the controller has assigned it to be logic 0 because the currently realized coefficient needs two vertices. However, due to the injection of a fault in the A/S unit of the third vertex-block, there are two A/S units that accumulate for the same tap and furthermore the additional A/S unit introduces an extra delay of one clock cycle that affects the behaviour of the filter. Therefore, the architecture has to take two steps in order to recover the functionality of the filter.

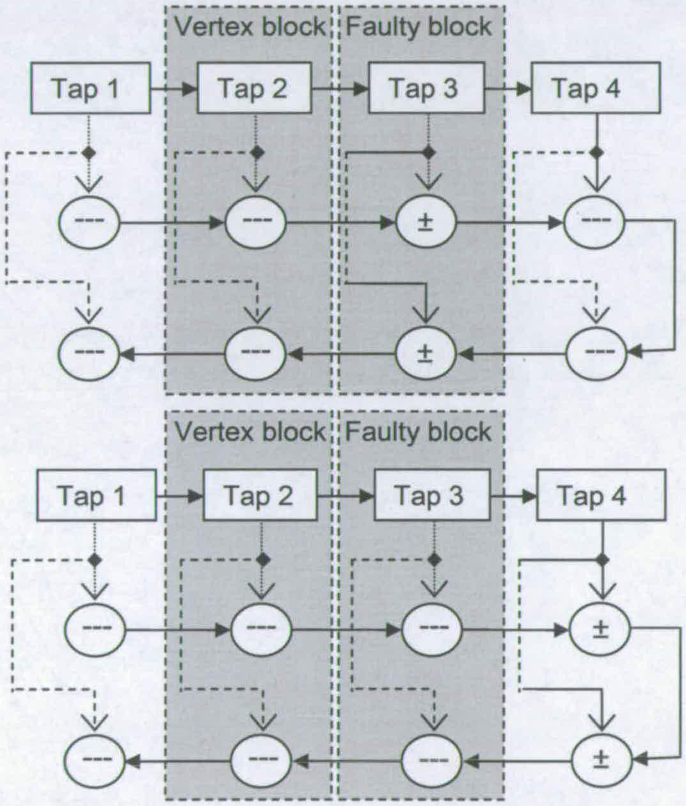


Figure 5.31: Recovery process from SHE in user memory

Firstly, the three vertices (multiplier block) have to realize the coefficient within 5 clock cycles instead of six in order the architecture to compensate for the introduction of the additional delay in the accumulation block. Secondly, the second vertex has to be appropriately configured by the GA in order to output logic 0. Thus, the second vertex will be isolated from the realization process of the coefficient and the accumulation will be performed in the third vertex-block.

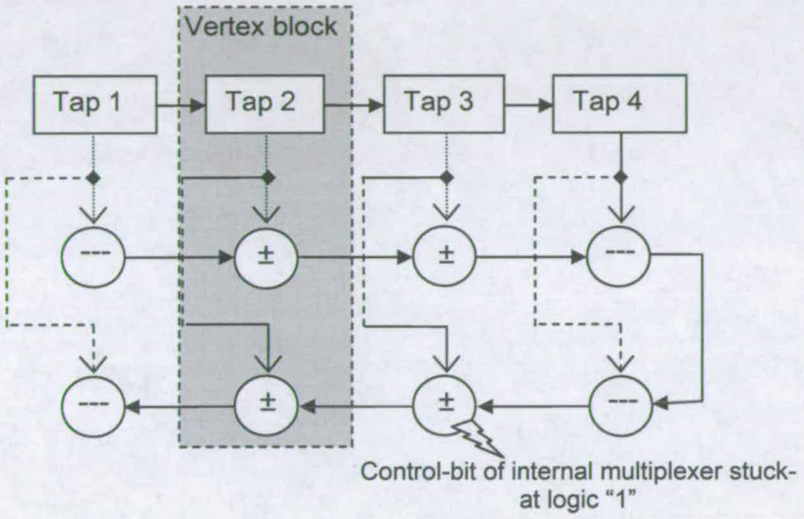


Figure 5.32: Recovery process from SHE in configuration memory

5.5.4 Synthesis and Power Analysis

For simulation purposes, the synthesized design consists of 50 vertices. For the synthesis, Synopsys synthesis tools and UMC 0.13μm technology cell library have been used. Moreover, since in this example the width of the data samples and the coefficient-sets is 16 bit, the reconfigurable hardware has been designed with 32 bit bus, in order to avoid overflow. Table 5-10 shows the synthesis results, which concern the die area that the multiplier and accumulation block of the filter occupies, respectively. Subsequently, the power analysis of the 41-tap low-pass filter has been carried out using the Synopsys

Prime-Power tool. Moreover, interconnect parasitics were taken into account for achieving a more realistic result. Figure 5.33 depicts the power, which is consumed by the simulated filter (0.693mW/MHz).

TABLE 5-10: AREA RESULTS OF RECONFIGURABLE DESIGN (50 VERTICES)

32-bit Bus Design	Area (mm^2)
Multiplier block	0.964
Accumulation block	0.401
Total Design	1.365

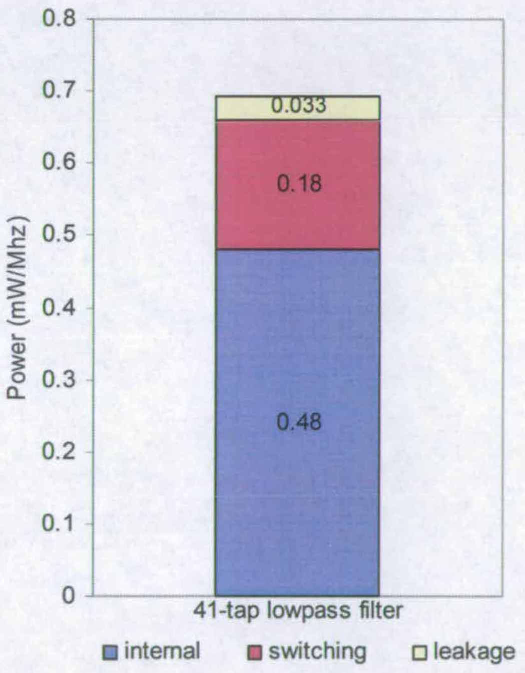


Figure 5.33: Power analysis of the 41-tap filter

Furthermore, the operational voltage of the design is 1.2 Volt. This in-turn implies that the evolved filter pulls current equal to 0.577mA/MHz. This value is outstanding compared with ATMEL series FPGAs [9], which consume 2.62mA/MHz in order to realize a symmetrical 32-tap filter. On the other hand, this architecture consumes twice as much power as the one in Section 5.4 for the realization of a 60-tap filter. However, as it can be seen in Figure 5.30, the achieved accuracy is outstanding and superior to that achieved by the EHW architecture in Section 5.4. Finally, the power analysis of the simulated netlist showed that the 69% of the total power is consumed by the multiplier block and the rest 31% by the accumulation block.

5.5.5 Conclusions

This section has presented an adaptive architecture, which is specifically tailored for the implementation of linear-phase FIR filters that would suit the JPL/Boeing gyroscope's control and interface circuitry. In conjunction with a custom non-stochastic search algorithm that has been specially designed for the reconfigurable hardware substrate, this architecture forms an autonomous fault-tolerant SoC solution. Moreover, power analysis has shown that the evolved filter outperforms industrial FPGAs that target programmable filter implementation. Several techniques have been employed both at the physical and algorithmic level to reduce the power consumption of the realized filters, since battery autonomy is very crucial issue in space applications. Finally, preliminary simulation results show that the architecture is capable of compensating for SHEs that occur either in the *user* or *configuration* memory of the reconfigurable fabric.

5.6 Reconfigurable Architecture for Real-Time Realization of High-Quality and Fault-Robust FIR Filters

5.6.1 Introduction

Each one of the EHW architectures that have been presented so far in Chapter 5 is a unique design that targets efficiently accomplishing the electronics associated with the JPL/Boeing gyroscope. However, simulation results have shown that architectures may suffer from poor accuracy and/or long implementation/reconfiguration time. The EHW architecture that has been presented in Section 5.5 proved to be adequate for the implementation of high-order FIR filters. However, it presents long reconfiguration time because there are difficulties in evolving the right timing for each tap (6 clock cycles). Hence, inspired from the characteristics of the architecture, a new EHW architecture emerged that eliminates the problem of timing evolution. Moreover, it also copes efficiently with *epistasis* and *pleiotropy* because it employs a novel hardware topology that does not create hardware dependencies between different coefficients within the targeted coefficient-set. Hence, it can be deduced that this architecture does not re-use previous partial sums in order to evolve other coefficients. Still it achieves efficient hardware utilization due to its hardware topology that employs the same CALUs for both the arithmetic and accumulation unit.

5.6.2 Reconfigurable Architecture

This approach presents an evolutionary driven custom-reconfigurable VLSI platform, whose hardware topology has been specially tailored for the implementation of digital

FIR filters. From a hardware perspective, the design presents a multiplier-less architecture that employs the POF technique in order to incorporate an efficient arithmetic unit in terms of physical-area and power-consumption. The reconfigurable hardware substrate that has been employed for the automatic design and/or adaptation of the targeted FIR filters consists of 200 CALUs. The interconnection scheme of these CALUs suits the implementation of direct-form filters. Moreover, it presents a novel architecture that enables the same addition/subtraction units to be employed in both the POF-based arithmetic and accumulation units of the filter. This has a great impact on the power consumption of the implemented filters since fewer addition/subtraction units are required.

The reconfigurable hardware can operate in two different operational versions (RT, RT+route). From a hardware perspective, the only difference between these two is that the RT+route version provides some additional programmable routing capabilities between four adjacent CALUs. This scheme is quite beneficial especially when faults occur on wires and not only in memory elements including *user* and *configuration* memory. Section 5.6.3 presents an evolutionary based mechanism that enables the interchangeable utilization of these two versions according to the evolution environment. Therefore, if evolution takes place in an error-free environment then the RT version is preferred because it provides prompt system adaptation and less power consumption compared with the RT+route version. On the other hand, the RT+route version is able to cope with higher density of faults including faults that occur on wirings.

Figure 5.34 illustrates the hardware topology of the RT version, while Figure 5.35 depicts the enhanced one (RT+route). In the enhanced version, due to the chain-style interconnection scheme that is incorporated, the CALUs that do not participate in the implementation of a filter can be easily disabled by the control-bit of the corresponding multiplexer. Hence, the RT+route version does not introduce any substantial overhead in the power consumption compared with the RT version, due to operating CALUs that do not participate in the design of a filter.

Subsequently, Figure 5.36 shows the distinctive arithmetic/logic components that build a single CALU. For better understanding of the proposed interconnection scheme, a consistent name coding has been kept in Figures 5.34, 5.35 and 5.36 for the input and output signals of the two hardware versions. It is apparent in Figure 5.36 that each CALU consists of an adder, a 2's compliment module, a programmable left shifter, a synchronous register and a 2:1 multiplexer. The adder is associated with a 2's compliment module to form an efficient (in terms of physical area and power consumption) programmable addition/subtraction (A/S) unit. The control bit in the 2's compliment module defines whether an addition or subtraction operation is to be performed. In addition to this, the programmable shifter incorporates 4 control bits that encode 16 different shifting operations. According to the shifter implementation, the incoming data to the shifter can be shifted left from 0 to 14 positions that correspond to a binary multiplication from 1 to 16384. Furthermore, in Figure 5.34 it can be seen that each CALU has two inputs. The former (input A) is always fed by the output of the previous CALU, while the latter (input B) is the original input of the FIR filter.

According to the hardware architecture, the same data sample is fed at the same time to all CALUs through *input B*. Finally, the 2:1 multiplexer, which provides the output of the CALU, determines the creation of a new tap. Therefore, if it selects to feed the output with the content of the synchronous register then a new tap within the filter is created. Otherwise, the CALU does not create a new tap and it just participates in the realization of a specific coefficient. Hence, it can be deduced that more than one CALU can participate in the creation of one tap. Furthermore, the importance of the multiplexer (inside the CALU) is very significant especially in circumstances where faults have occurred on the *user* memory (register) of a CALU. According to this scenario, faulty CALUs cannot be used to create new taps within the filter and the multiplexer must be appropriately controlled in order for the CALU to cope with the occurred failure. The number of the CALUs that are employed in the creation of a coefficient is basically defined by the value of the coefficient itself. Figure 5.37 depicts a paradigm, which is based on the RT hardware version and it analytically shows the relation between the evolved coefficient and the number of the employed CALUs. According to this, three CALUs are employed to realize coefficient with value 284. Specifically, the first CALU performs left-shift by 8 and addition (control-bit in 2's compliment module is equal to 0) to create the number 256. At the same clock cycle the second CALU performs left-shift by 5 and addition (control-bit equals to 0) to create number 32, which is added-up with the previously created number 256. Subsequently, the third CALU produces number 4 by performing left-shift by 2 and subtraction (control-bit equals to 1). This result is subtracted from number 288 that has been created from the first two CALUs. This process occurs on the same clock cycle with the former operations of CALUs 1 and 2. Moreover, it can be seen that only the third CALU stores the output of the A/S unit in the synchronous register because all the three described operations, which correspond to the realization of a single coefficient, must be accomplished within one clock cycle.

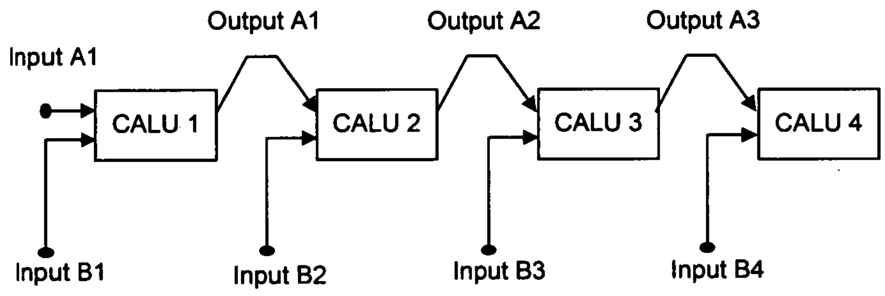


Figure 5.34: Hardware topology of RT version

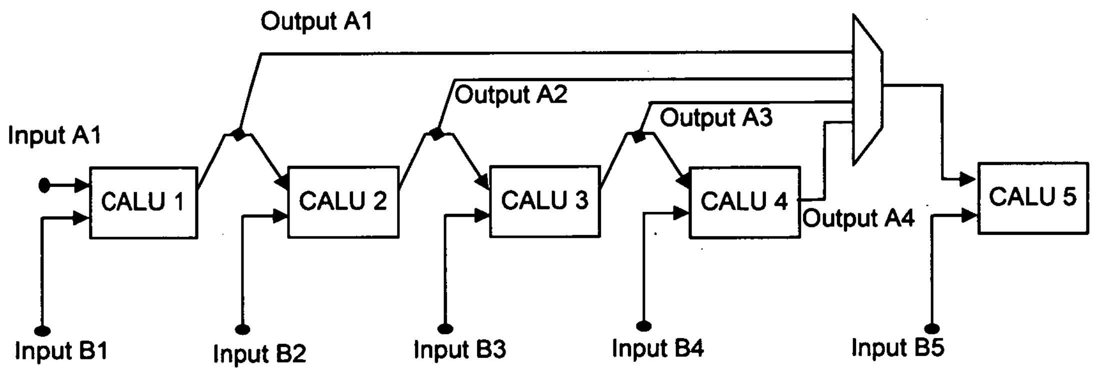


Figure 5.35: Hardware topology of RT+route version

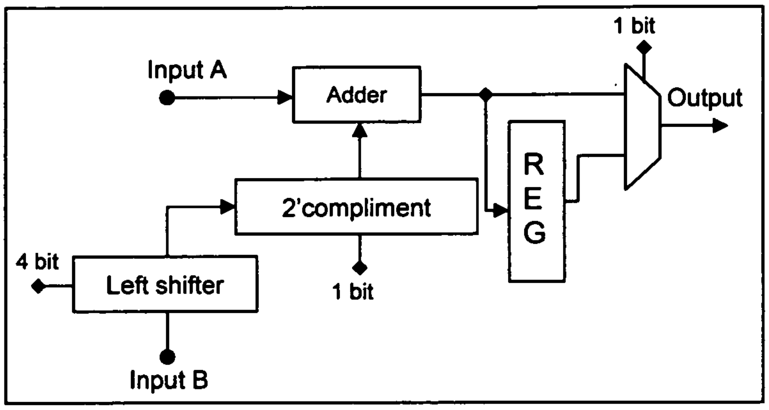


Figure 5.36: CALU hardware description

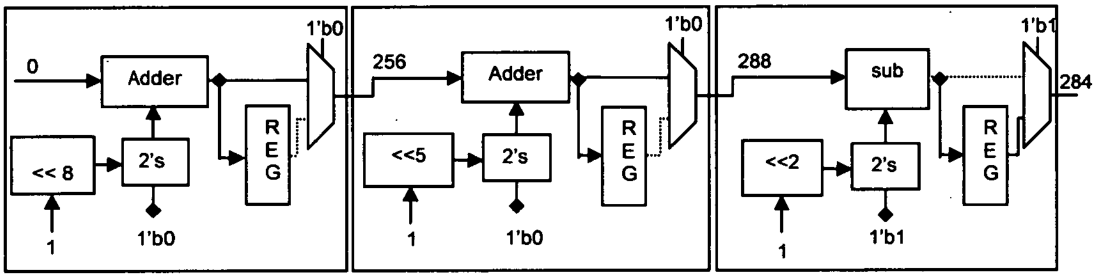


Figure 5.37: Relation between realized coefficients and number of employed CALUs

5.6.3 Hybrid Evolutionary Technique

As was mentioned in the introduction, this evolutionary technique targets problems, such as insufficient accuracy in the evolved circuit behaviour and long convergence time of the EA that implies long time for implementation and/or reconfiguration of the FIR filters. The proposed EA has a correlative conjunction with the proposed reconfigurable hardware topology and therefore it cannot be widely used for any reconfigurable structure. Its novelty is primarily based on the instantaneous minimization of the search space of the algorithm by constraining the genetic reproduction (mutation) only to the effective part of the chromosome that participates on the currently evolved coefficient. This is mainly feasible due to the interconnection scheme of the CALUs. According to this chain-style interconnection scheme, it is easy to dynamically map the realization of each coefficient on specific hardware resources. To accomplish this task there are three parameters (*current fitness-score* of each chromosome, the *current number of CALUs* that each configuration utilizes and the *current number of taps* that have been created) that the EA needs to consider. Moreover, unlike conventional fitness-functions, this EA evolves filters in an incremental manner (this approach was first introduced in Section 5.5). The EA evolves one coefficient at a time and does not consider the total coefficient set simultaneously. Therefore, a fitness-score of 213,000 means that the first two coefficients have been successfully evolved (each contributes 100,000 to the total fitness-score) and that the third coefficient, which is currently being evolved, deviates 1-0.13=0.87% from the ideal coefficient. Figure 5.38 explains the mechanism that the proposed EA employs in order to calculate at each phase of the evolution the number of CALUs that are utilized. It is apparent that the first coefficient (C1=96) occupies two CALUs, while the second one utilizes 3 CALUs. Moreover, the shaded box inside each CALU indicates that the output of the A/S unit is stored in the synchronous register (creation of a new tap) while the blank box indicates that the CALU is just participating in the realization of the same coefficient. Therefore, in order to calculate the total number of CALUs that are currently employed for the scenario in Figure 5.38 the following pseudo-code can be used.

Pseudo-code 1:

```
$number\_of\_taps = 0;$
$number\_of\_CALUs = 0;$
$While(number\_of\_taps < (\text{int})(fitness\_score/100000)\{$
$if(tap\_bit[0] == 1)\{$
$number\_of\_taps++;$
$\}$
$number\_of\_CALUs++;$
$tap\_bit = tap\_bit >> 1;$
$\}$
$number\_of\_CALUs = number\_of\_CALUs - 1;$
```

In this example the (int)(fitness_score/100000) gives the integer part of the division 213000/100000=2. In addition to this, all the control-bits that define the creation or not of new taps are stored in a register (tap_bit[199:0]) in order to calculate how many out of the total number of the employed CALUs form a new tap. The output of this pseudo-code gives the total number of employed CALUs (number_of_CALUs) equal to 5. Therefore, for the evolution of the third coefficient (C3=2080) mutation will be performed in the part of the chromosome that begins in the 6th CALU and ends in the 11th CALU. Thus, it is obvious that mutation is restricted to a chromosome area that encodes only 6 CALUs. The size of this area (6 CALUs) has been empirically chosen after the completion of several simulations. However, this size can be further decreased to 5 CALUs for instance in a faulty-free environment, since 5 CALUs are considered to be adequate to realize the majority of targeted coefficients. Using this approach the EA is not allowed to utilize redundant hardware resources since there is a restriction on the realization of each coefficient. Consequently, this technique is also beneficial from the power consumption perspective.

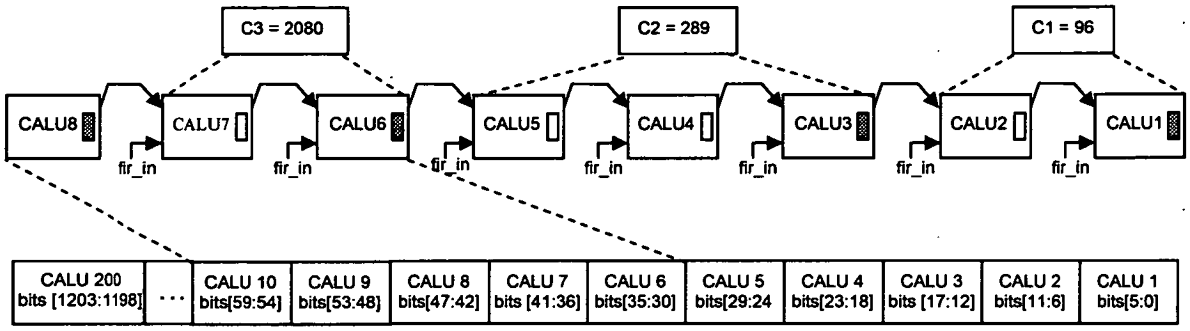


Figure 5.38: Description of search-area minimization mechanism

Another significant factor that contributes to the more efficient realization (in terms of quality and speed) of FIR filters emerges from the fact that the proposed reconfigurable hardware structure is not prone to creating hardware dependencies between adjacent coefficients within the coefficient-set. This can be easily understood since the architecture does not use the value of previously realized coefficients in order to create the latter ones. Hence, in order for the architecture to compensate for the fact that it cannot employ previous coefficient values, shifters have been designed to provide relatively large binary integer numbers up to 16384. From an evolution perspective, due to the EA architecture used, the two hardware versions can be merged into a unique reconfigurable hardware substrate that has two operational modes, since RT is a subgroup of RT+route. With this approach, the EA can occasionally take advantage of the additional flexibility that comes with the RT+route version when there is a burst of faults on the hardware resources or it can use the RT version, which appears to be more efficient in terms of speed of adaptation, particularly when very few faults appear on the hardware components. This switch between the two versions can be achieved if the EA appropriately imposes under evolution the proper parts of the chromosomes. Figure 5.39 depicts how the encoding within each chromosome is structured and that the chromosome that encodes the RT+route is just an extension of the RT one. Hence, depending on which version the EA intends to use, genetic operations are performed on the corresponding chromosome size. Finally, when the RT version is used, the control-bits of the multiplexers at the extension of the chromosome are programmable so that each CALU is connected to its adjacent one.

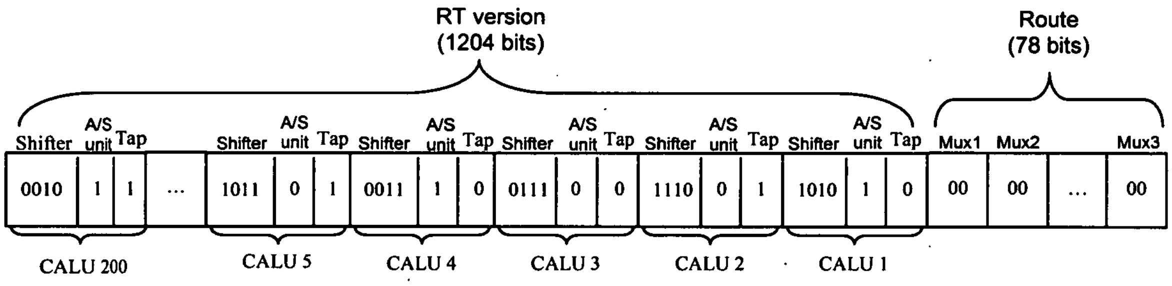


Figure 5.39: Differentiation in the encoding between the RT and RT+route chromosome

5.6.4 Simulation Results

To prove the ability of the architecture to implement any order FIR filters efficiently in terms of quality, power consumption and speed, two digital filters were evolved, whose specification is defined in Tables 5-11 and 5-12.

TABLE 5-11: SPECIFICATION OF SYMMETRICAL 17-TAP LOW-PASS FIR FILTER

Coefficient set	Value
C_{-8}, C_8	-256
C_{-7}, C_7	-1877
C_{-6}, C_6	-3718
C_{-5}, C_5	-4485
C_{-4}, C_4	-1855
C_{-3}, C_3	5168
C_{-2}, C_2	15021
C_{-1}, C_1	23783
C_0	27296

For the evolution of these filters only the RT version of the reconfigurable platform is used. Subsequently, the two versions (RT and RT+route) of the reconfigurable fabric were compared with the reconfigurable platform in [44] that has been proven to be adequate when realizing FIR filters even in the presence of faults. Thus, the specification of the same FIR filter is employed for the purpose of this comparison. This filter is a symmetrical 31 tap low-pass filter, which was taken from the industrial design of low-power filter cores for hearing aids and it has been developed in joint collaboration with Bernafon LTD and the University of Edinburgh [126].

TABLE 5-12: SPECIFICATION OF SYMMETRICAL 49-TAP PASS-BAND FIR FILTER

Coefficient set	Value
C_{-24}, C_{24}	-59
C_{-23}, C_{23}	-207
C_{-22}, C_{22}	-73
C_{-21}, C_{21}	474
C_{-20}, C_{20}	498
C_{-19}, C_{19}	-606
C_{-18}, C_{18}	-1150
C_{-17}, C_{17}	350
C_{-16}, C_{16}	1609
C_{-15}, C_{15}	164
C_{-14}, C_{14}	-1398
C_{-13}, C_{13}	-243
C_{-12}, C_{12}	562
C_{-11}, C_{11}	-992
C_{-10}, C_{10}	-23
C_{-9}, C_{9}	3779
C_{-8}, C_{8}	1189
C_{-7}, C_{7}	-7091
C_{-6}, C_{6}	-4876
C_{-5}, C_{5}	8932

C_{-4}, C_4	10336
C_{-3}, C_3	-7589
C_{-2}, C_2	-15317
C_{-1}, C_1	2988
C_0	17336

The corresponding coefficient set shown in Table 5-13 is highly challenging as it exhibits a large dynamic range with coefficient multiplicands scaling the filter input from 2^7 to 2^{14}, using word-lengths of 16-bits. Initially, evolution is performed within an error-free environment in which it is assumed that there are no faults occurring in the reconfigurable hardware. Subsequently, different fault scenarios are simulated in order to demonstrate the capability of the architecture to adapt its functionality within faulty environments. The faults are modeled either as stuck-at "0" or stuck-at "1" faults and they are injected in both the *user* and *configuration* memory of the system.

TABLE 5-13: SPECIFICATION OF SYMMETRICAL 31-TAP LOW-PASS FIR FILTER

Coefficient set	Value
C_{-15}, C_{15}	-59
C_{-14}, C_{14}	-207
C_{-13}, C_{13}	-73
C_{-12}, C_{12}	474
C_{-11}, C_{11}	498

C_{-10}, C_{10}	-606
C_{-9}, C_{9}	-1150
C_{-8}, C_{8}	350
C_{-7}, C_{7}	1609
C_{-6}, C_{6}	164
C_{-5}, C_{5}	-1398
C_{-4}, C_{4}	-243
C_{-3}, C_{3}	562
C_{-2}, C_{2}	-992
C_{-1}, C_{1}	-23
C_{0}	3779

5.6.4.1 Evolution of the 17-tap low-pass and 49-tap pass-band filter

The filter in Table 5-11 is a 17-tap symmetrical low-pass filter that presents a stop-band attenuation of 50 db. Although the order of the filter is small, the evolution of this filter is challenging because the coefficient-set is spread in a range from 2^9 to 2^{15}. Subsequently, the filter defined in Table 5-12 is a 49-tap symmetrical pass-band filter, which presents an asymmetrical left and right stop-band attenuation of 60 db and 80 db, respectively. The coefficient-set of this filter is even more challenging since its range lies from 2^5 to 2^{15}. Figure 5.40 depicts the comparison between the frequency response of the ideal and the evolved 17-tap low-pass filter. From the two traces on the graph it can be deduced that the evolved filter matches the ideal one very accurately. Moreover, the evolution process lasts only 448 generations and the filter utilizes 76 CALUs in total. Figure 5.41 compares the frequency response between the ideal and evolved 49-tap pass-band filter. Again the evolutionary strategy managed to accurately evolve the pass-band filter within 1715 generations by using 189 CALUs.

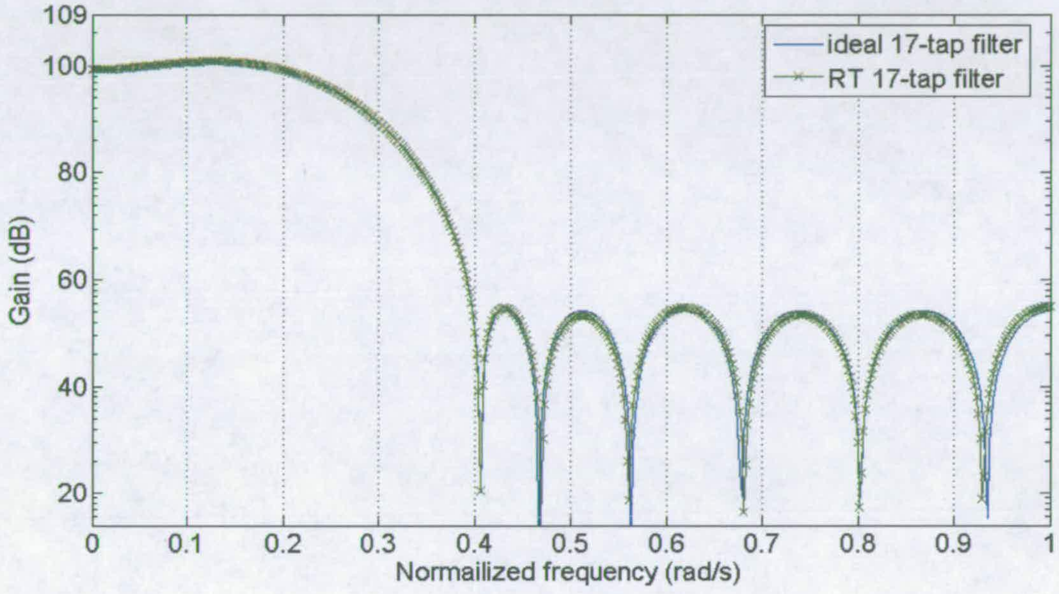


Figure 5.40: Comparison of the frequency response between the ideal and the evolved 17-tap low-pass filter, using the RT version

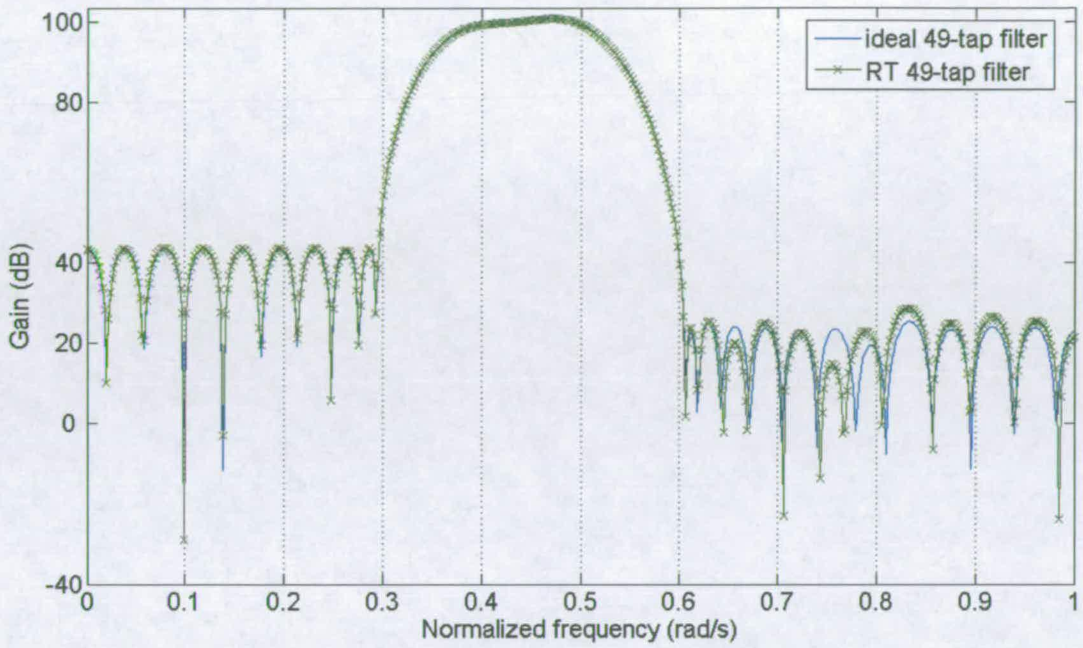


Figure 5.41: Comparison of the frequency response between the ideal and the evolved RT 49-tap pass-band filter, using the RT version

5.6.4.2 Evolution of the 31-tap filter within an error-free hardware mean

In the first simulation scenario it is assumed that evolution occurs upon a fault-free reconfigurable hardware mean. Figure 5.42 depicts the comparison between the frequency response of the ideal filter and the filter that has been evolved by the RT version. Whereas, Figure 5.43 depicts the corresponding comparison between the ideal filter and the evolved filter of the RT+route version. In both comparisons it is apparent that the evolved filters accurately match the ideal ones. Moreover, it can be seen that the RT+route evolved a filter that presents better attenuation in the stop-band for a specific range of frequencies than the ideal one.

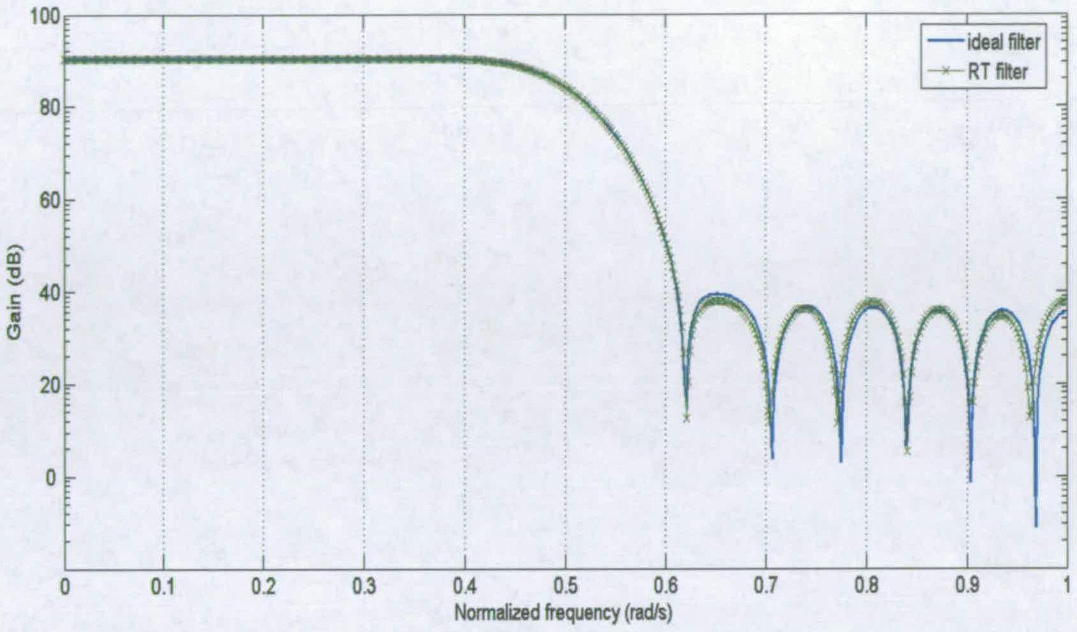


Figure 5.42: Comparison of the frequency response between the ideal and the evolved 31-tap low-pass FIR filter, using the RT version

It is also equally important to investigate both the duration that each evolution lasts and the number of hardware resources that are used for the realization of the filters. In order to obtain a more realistic result, the same filter was realized by using the RT version with 9 different reproduction rates.

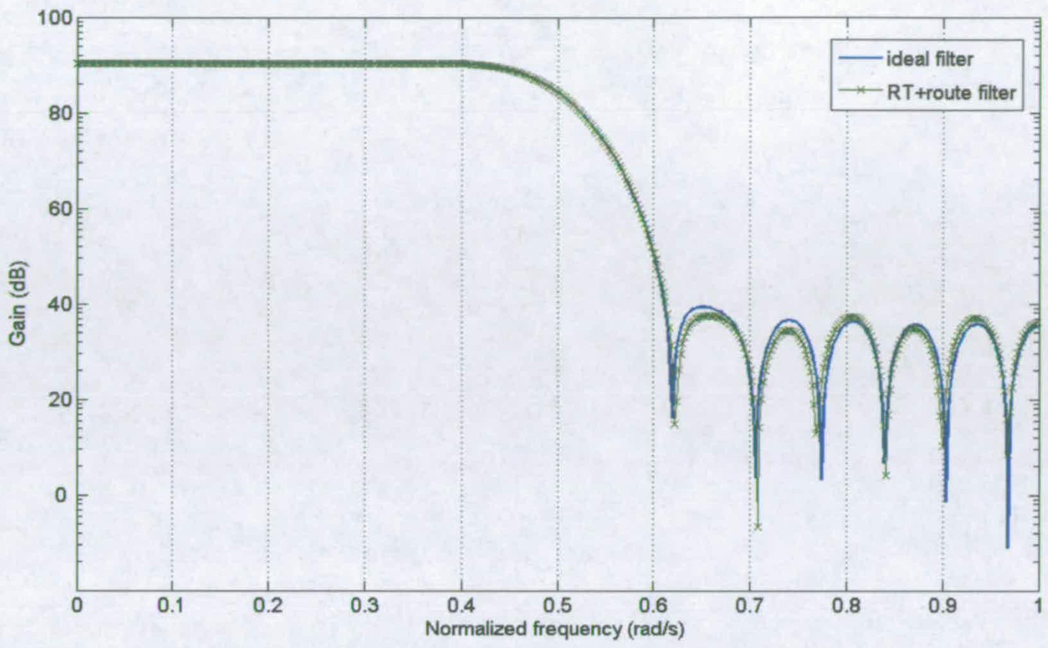


Figure 5.43: Comparison of the frequency response between the ideal and the RT + route filter

In Figure 5.44 the crossover rate is kept constant and equal to 10%, while the mutation percentage rate dynamically varies from 0 to 5, 0 to 10 and 0 to 15 for the three simulated scenarios, respectively. It can be deduced that the second scenario takes less time (440 generations), while the third evolution lasts longer than the other two (580 generations). From a hardware utilization perspective, the first two evolutions employ 73 and 79 CALUs, respectively. The third one has found a less effective configuration scheme and needs 88 CALUs. It must be mentioned here that the depicted results in Figures 5.44, 5.45, 5.46 and 5.47 have been obtained by the mean number of generations and CALUs of 10 different simulation runs. In Figure 5.45 the crossover rate is increased to 30% and the evolution of the filter with three different mutation rates has been performed. In this scenario the third evolution (mutation 15%) is the most efficient in terms of number of generations (430) and utilizes 81 CALUs. Subsequently, in the three simulations depicted in Figure 5.46, the crossover rate is equal to 50%. In this scenario the second evolution (mutation 8%) is the most efficient from any perspective, since it takes only 430 generations and it employs 72 CALUs. According to this result, the architecture needs on average 13.8 generations and 2.32 CALUs per coefficient. This is a very impressive figure considering the word-length of the evolved coefficients.

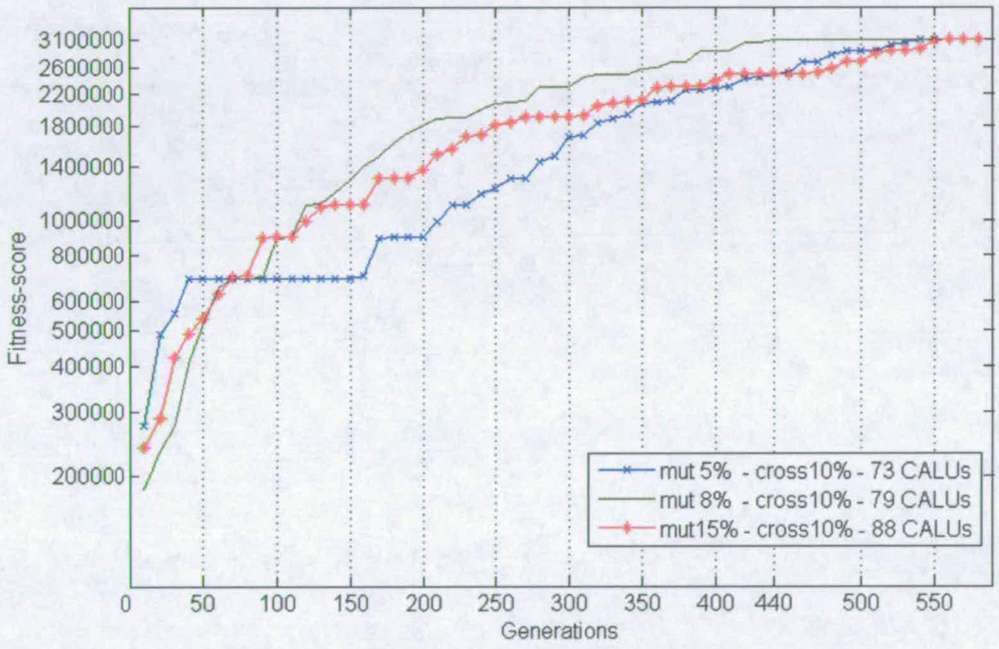


Figure 5.44: RT evolution process for crossover rate 10% and dynamic mutation rates 5%, 10% and 15%

Moreover, these results are superior to previous research work [44]. The authors in [44] achieved a fitness-score that varies from 91.7% to 97.3% within 6500 generations. This figure is 11 times bigger than the longest evolution time and 15 times bigger than the shorter evolution time, achieved in this section. Moreover, in these evolutions the achieved fitness-score never drops below 99% and it is most often above this threshold. Moreover, in [44] the 31-tap filter is realized within a reconfigurable 8x8 array structure and the authors claim that the hardware utilization is about 78%, which implies that 50 Programmable Arithmetic Logic Units (PALUs) are employed in total. However, there are another 31 adders that compose the accumulation unit and have to be counted. Thus, it can be deduced the design employs about 81 PALUs. On the other hand, the proposed architecture has successfully realized the same filter in all 9 different evolutions by using between 72 and 89 CALUs.

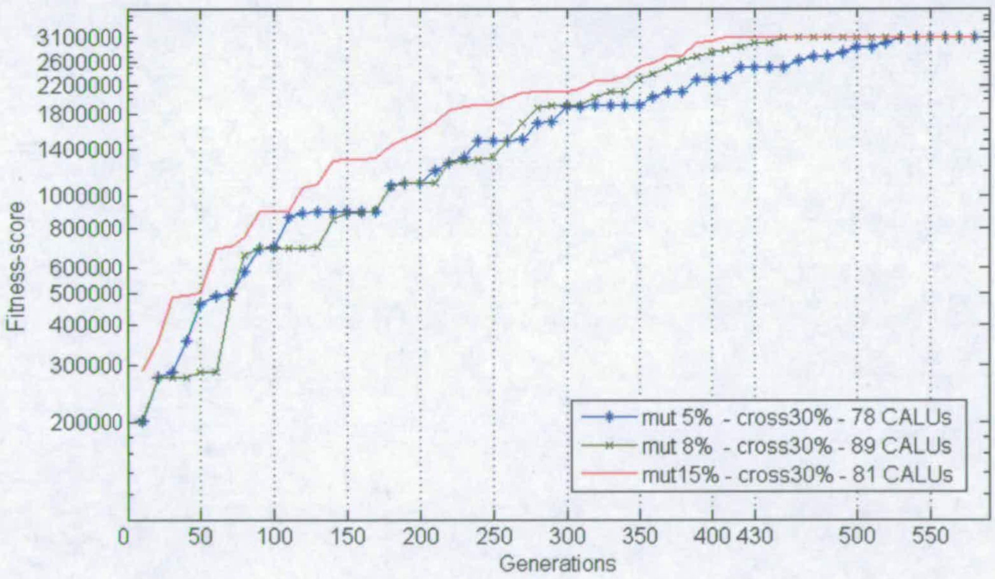


Figure 5.45: RT evolution process for crossover rate 30% and dynamic mutation rates 5%, 10% and 15%

Finally, the RT+route version has been tested for its capability to realize the 31-tap filter. Three different simulations with three different reproduction rates were performed, which can be seen in Figure 5.47. From the results it is apparent that the RT version is more efficient than the RT+route one when evolution is performed in the absence of faults. This is quite sensible considering that the additional routing capabilities in the RT+route version increase the maximum search area of the algorithm.

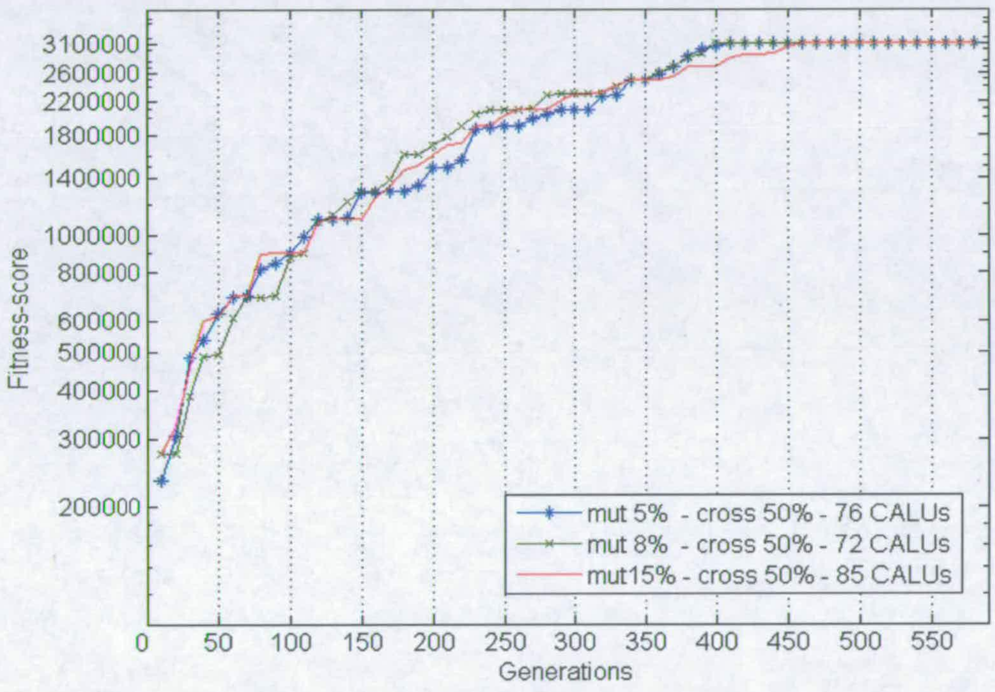


Figure 5.46: RT evolution process for crossover rate 50% and dynamic mutation rates 5%, 10% and 15%

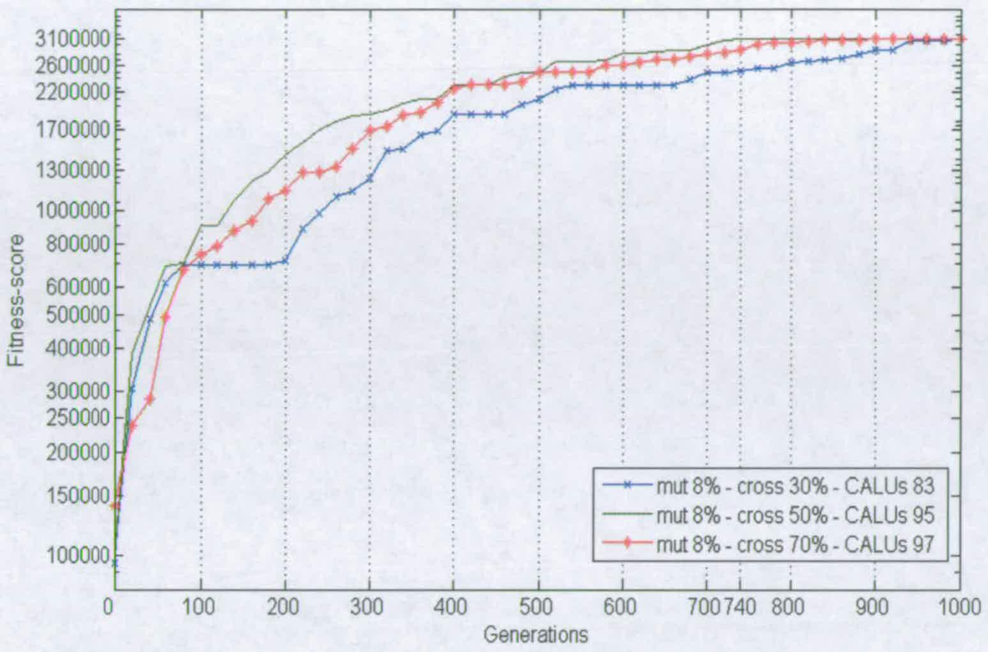


Figure 5.47: RT + route evolution process for dynamic mutation rate 8% and crossover rates 30%, 50% and 70%

5.6.4.3 Filter Adaptation for 15% Faults Occurring in the User Memory

It is very important for the proposed architecture to evolve electronic circuits that are able to adapt during their operation to behavioral deviations that appear due to the presence of faults. To model this scenario the configuration string of the 31-tap filter given in Table 5-13 was taken and SHEs were injected in the *user* memory of the system. These faults are stuck-at faults that make the content of the synchronous register inside a CALU permanently erroneous. Therefore, in order for the system to recover from a fault, it must resolve a configuration that does not utilize the faulty register. According to the architecture of the reconfigurable hardware, the RT version can cope with this only by adjusting the faulty CALU so that it does not create a tap. Based on Figure 5.34 and 5.36, the EA has to switch the control-bit of the 2 to 1 multiplexer to logic "1" and hence the output of the adder is passed directly to the output of the CALU, without being saved in the register. On the other hand, the RT+route version has an additional mechanism to eliminate this fault that is to overpass the faulty CALU by utilizing a different routing solution between the CALUs. Therefore, in this scenario faults were injected in the 15% of the CALUs that compose the reconfigurable hardware. It must be mentioned here that the faults were intentionally placed in CALUs that actively contribute in the realization of the filter. Figure 5.48 depicts how promptly the RT and RT+route versions adapt to the same scenario of faults. However, because their initial configuration strings differ, the same faults affect the two filters differently. Therefore, it can be seen in Figure 5.48 that in the RT filter the first coefficient that is affected is the 14th one (fitness-function starts from 1400000), while in the RT+route filter the first affected coefficient is the 5th one (fitness-score starts approximately from 500000). The gap between the two graphs remains almost constant throughout the evolution process. Hence, it can be deduced that for low density of faults both hardware versions adapt in a similar way. However, there is a period of time (approximately 110 generations) in the evolution process of the RT version where there is no improvement in the fitness-score of the EA. This is reasonable considering the fact that the RT version has fewer mechanisms to cope with faults.

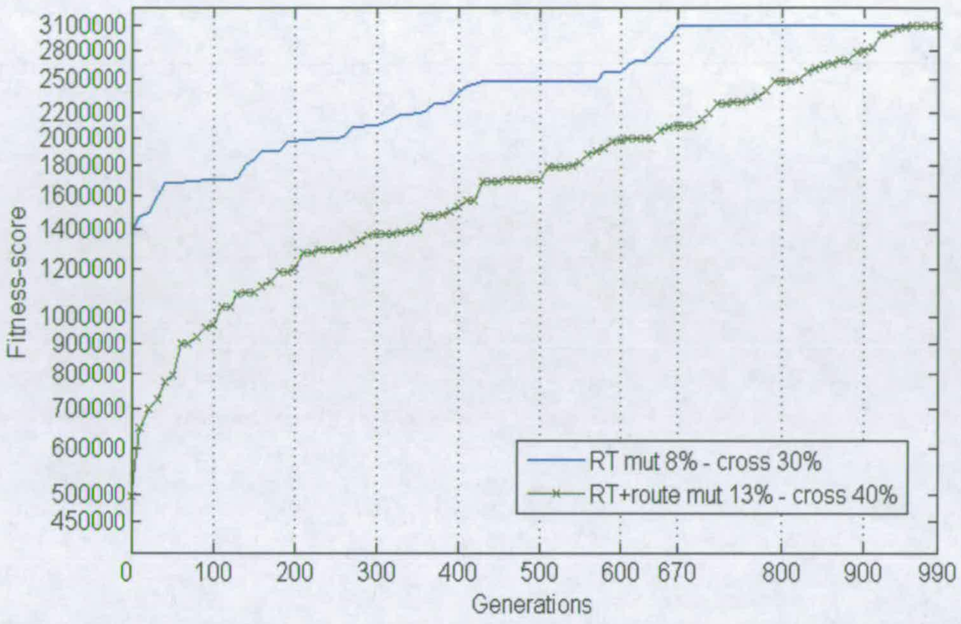


Figure 5.48: Adaptation process of RT and RT+route versions for injection of 15% faults in the user memory

5.6.4.4 Filter Adaptation for 30% Faults Occurring in the User Memory

In this simulated paradigm the density of the injected faults was increased. The scope of this scenario is to investigate whether the reconfigurable hardware can also cope efficiently with a burst of faults and not only with isolated ones. Thus, it was intentionally selected to inject faults into CALUs close to each other. Figure 5.49 depicts the adaptation process of the two filters that employ the RT and RT+route version. In this scenario it is obvious that the RT+route version adapts more efficiently in the presence of faults. Apart from the fact that the adaptation of the filter with RT+route uses 130 fewer generations, the equivalent adaptation with the RT version presents two evolution regions, which last 230 and 400 generations respectively, and provides insubstantial improvement to the filter's functionality.

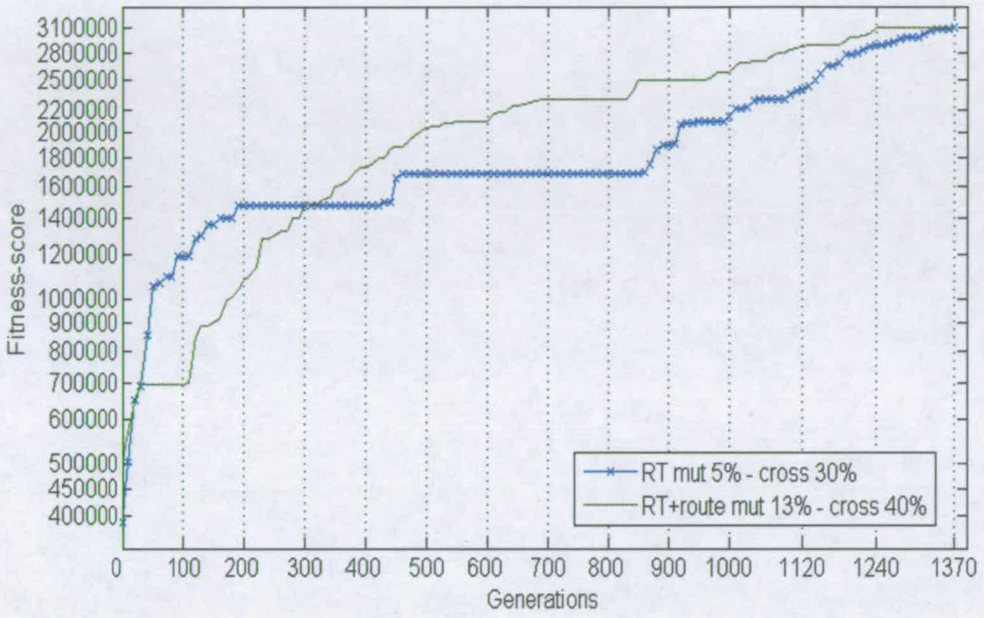


Figure 5.49: Adaptation process of RT and RT+route versions for injection of 30% faults in the user memory

5.6.4.5 Filter Adaptation for 15% Faults Occurring in the Configuration Memory

In this scenario the faults are injected exclusively in the *configuration* memory of the system. On the contrary with the *user* memory, the *configuration* memory defines the control-bits of the shifter, the adder and the 2 to 1 multiplexer of each CALU. Hence, a faulty *configuration* memory element can affect each one of these three components. For example due to a fault a CALU could perform only subtraction or it could shift only the input data by 2 positions or it would always act as a new tap. In the simulated scenario faults were injected in the *configuration* memory of 15% of the CALUs that compose the 31-tap filter. These faults have been arranged to affect the faulty CALUs in all the possible manners that were mentioned above. Figure 5.50 presents the adaptation process of the 31-tap filter with the two hardware versions. It is apparent that both hardware versions are able to successfully adapt to the filter's degradation. The RT one adapts within 840 generations, while the adaptation with the RT+route takes 810 generations. Similarly with the previous scenarios where faults are injected in the hardware, the adaptation of the filter with RT presents two inefficient time regions during which there is not any improvement in the functionality of the filter.

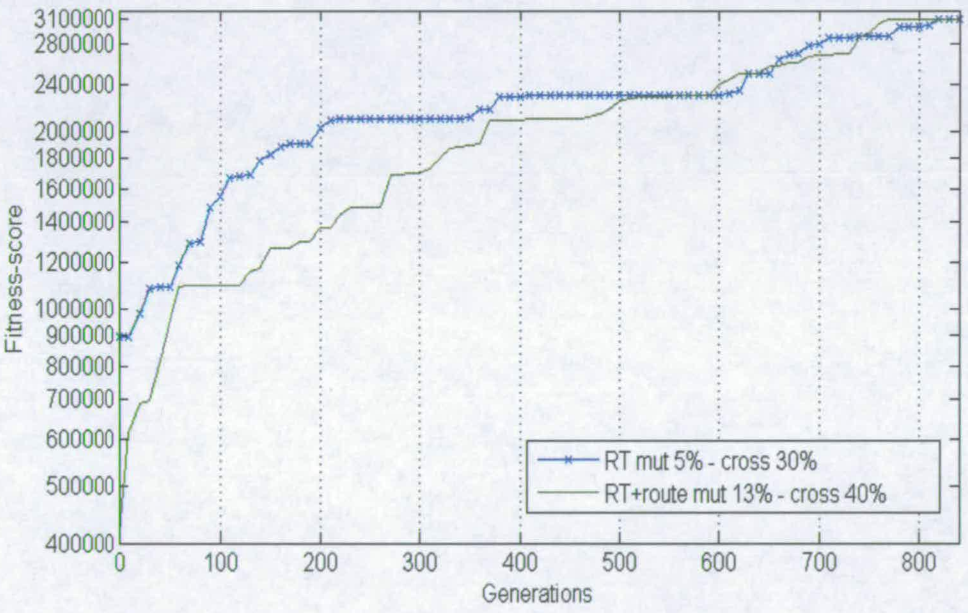


Figure 5.50: Adaptation process of RT and RT+route versions for injection of 15% faults in the configuration memory

5.6.4.6 Filter Adaptation for a Mixture of 30% Faults Occurring in both Kinds of Memory

In this scenario faults have been injected in both the *user* and *configuration* memory of the reconfigurable hardware. In this simulation paradigm only the RT+route version has been tested because the RT one is not able to cope with all combination of faults that occur in the two kinds of memory. Specifically, when there is a fault in the *user* memory of a CALU and at the same time the control-bit of the multiplexer inside the CALU is stuck-at "1" then the RT version is unable to cope with this situation and the only way for the filter to recover is for the EA to provide an alternative routing scheme between the CALUs and bypass the faulty one. Figure 5.51 depicts the adaptation process of the filter.

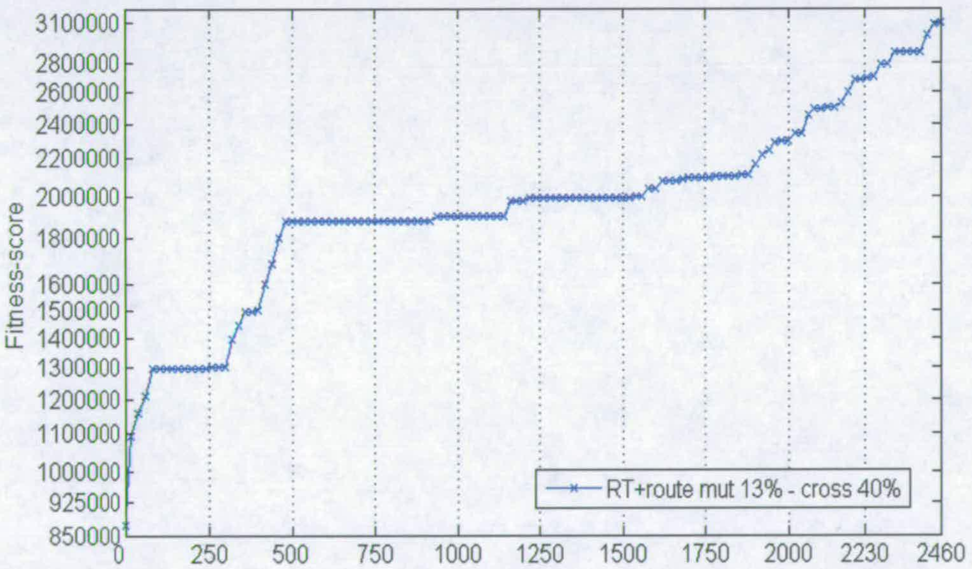


Figure 5.51: Adaptation process of RT+route version for injection of 30% faults in total in the user and configuration memory

It can be deduced that the adaptation of the 13th, 19th, 20th and 21st coefficient takes considerably longer time. However, this is reasonable because several faults have been injected in the hardware resources that correspond to these specific coefficients.

5.6.5 Synthesis

To obtain the results in Section 5.6, a reconfigurable structure that consists of 200 CALUs has been synthesized. The synthesis was performed with the Synopsys synthesis tools using UMC 0.13 μm technology cell library. Moreover, since the width of the targeted coefficients and incorporated data samples is 16 bits, the reconfigurable hardware has been designed with a 32 bit data path in order to avoid overflow. The obtained results show that the RT+route version that occasionally can also operate as RT version occupies 1.48 mm^2 of silicon area and that the maximum clock frequency can be set to 13MHz. This frequency is defined by the maximum number of CALUs that compose the critical path of the design. Simulation results have shown that 4 CALUs are adequate to realize each one of the coefficients of the 31-tap low-pass filter. Finally, this clock frequency is adequate for the implementation of the gyroscope's electronics since the gyroscope has update rate of 48 KHz.

5.6.6 Power Analysis

The power analysis of the evolved FIR filters has been carried out using the Synopsys Prime-Power tools. Accurate interconnect analysis has been considered by setting appropriately appropriate wire load models during the synthesis of the proposed reconfigurable structure. Apart from the utilization of the primitive operators that make the proposed reconfigurable platform suitable for low-power applications, the chain-style interconnection scheme that is incorporated in the RT and RT+route versions enables further reduction of power. This is feasible because the CALUs that do not participate in the implementation of a filter can be easily switched-off. Specifically, in the RT+route version the control-bits of the 4 to 1 multiplexers, which provide the alternative routing schemes between adjacent CALUs, can also control the enable signals of the CALUs and switch-off the unused ones. Therefore, during the operation of the reconfigurable platform the disabled CALUs consume only leakage power. Figure 5.52 depicts the power analysis of the 17-tap low-pass filter defined in Table 5-11, the 31-tap filter defined in Table 5-13 (evolution with both the RT and RT+route version) and the 49-tap pass-band filter defined in Table 5-12. It can be seen that the 17-tap filter consumes 0.364 mW/MHz, the 48-tap filter consumes 0.75mW/MHz and the RT and RT+route 31-tap filters consume 0.3 and 0.377 mW/MHz, respectively. Moreover, it can be seen that the RT version consumes 0.077 mW/MHz less power than the RT+route for the realization of the same filter. Hence, it is obvious that the reconfigurable platform has to operate in the RT version when the density of faults is not high and it must switch to the RT+route version when the RT one is not able to cope with the quantity of the occurred faults. Another interesting point in Figure 5.52 is that the 17-tap low-pass filter consumes almost the same amount of power as the 31-tap low-pass filter. However, this is reasonable because the coefficient-set of the 17-tap filter is more challenging than the 31-tap one and hence more CALUs are needed for the realization of each coefficient. Furthermore, based on the technology cell library (UMC 0.13 μm) that has been employed in the synthesis of reconfigurable design, the operational voltage is 1.2 Volt. This implies that the evolved filters (17-tap, 49-tap, 31-tap RT and 31-tap RT+route) pull current equal to 0.303, 0.625, 0.25, and 0.314 mA/MHz. These values are outstanding compared with ATMEL series FPGAs [9], which consume 1.29 mA/MHz in order to implement a symmetrical 16-tap FIR filter and 2.62 mA/MHz to implement a 32-tap FIR filter. Finally, the 49-tap pass-band filter consumes almost the same power as the 41-tap filter that has been evolved by the EHW architecture in Section 5.5. This implies that this architecture is even less power demanding and still achieves the same accuracy.

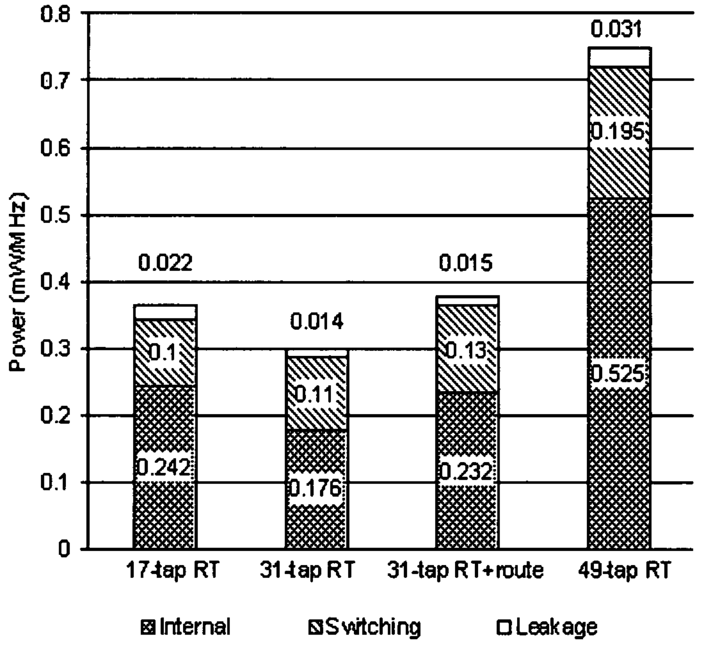


Figure 5.52: Power analysis for the evolved filters

5.6.7 Conclusions

This section presents an autonomous evolutionary driven custom reconfigurable VLSI platform that attempts to overcome the problems (decrease the design and reconfiguration time of any order digital FIR filter), which the previous architectures encounter, and composes the ideal framework to accomplish the gyroscope's electronics. The evolutionary strategy in collaboration with a novel custom reconfigurable hardware structure have been specially tailored for the realization of low-power filters, which can also adapt their functionality in the presence of faults occurring in both the *user* and *configuration* memory of the system. The simulation results associated with the two operational modes of the reconfigurable hardware, presented in Section 5.6, have proven that the RT version can implement FIR filters within an error-free environment in 11 to 15 times less generations than in previous work and that the RT+route can efficiently adapt the functionality of these filters in the presence of faults that occupy the 15% of the total number of employed CALUs. Moreover, it consumes 4 to 10 times less power than the ATMEL series FPGAs that have been employed for the implementation of similar FIR filters.

Finally, Table 5-14 summarizes some important features that characterize the reconfigurable hardware (RH) substrates presented in Chapter 5. In this comparison the reconfigurable hardware presented in Section 5.2 was excluded because the quality of the achieved simulation results was not comparable to that of the following four reconfigurable hardware substrates. The quantity of faults that have been introduced in order to evaluate the fault-robust capability of the presented reconfigurable hardware substrates correspond to the user and configuration memory of the system. Moreover, the results in power consumption correspond to the average power that is dissipated per coefficient and it is measured in mW/MHz. The four reconfigurable hardware substrates do not have the same ability to evolve filters of similar order. Therefore, based on the filters that have been evolved by each, the average power per coefficient has been calculated. From Table 5-14 it can be deduced that RH4 presents superiority over the rest reconfigurable hardware substrates in most of the features that compose the following benchmark. RH2 consumes less power than RH4. However, this benefit introduces a penalty in the all the other features, comparing RH2 with RH4.

TABLE 5-14: COMPARISON OF RECONFIGURABLE HARDWARE SUBSTRATES

Features	RH1 (section 5.3)	RH2 (section 5.4)	RH3 (section 5.5)	RH4 (section 5.6)
Accuracy	Low	High	Very high (>99%)	Very high (>99.9%)
Fault-Robust	Small number of faults	Up to 10% faults	Small number of faults	Up to 30% faults
Linearity	Not always linear phase	Not always linear phase	Linear phase	Linear phase
Capability of evolving filters	Low order filters	Medium/High order filters	Medium/High order filters	High order filters
Power consumption	0.018 – 0.026	0.0037 – 0.0042	0.017	0.009 – 0.021

Chapter 6
Evolutionary Mechanisms

6.1 Introduction

A major problem in the evolution of electronic circuits is the inefficiency of the incorporated evolutionary mechanisms to accurately guide the implementation of the targeted electronics. Moreover, evolution may take a considerably long time and this drawback makes these techniques unattractive to the electronic industry, since the quality and the worst time for adaptation is not always guaranteed. On the other hand, evolutionary design presents unique characteristics such as automatic design, autonomous reconfiguration, fault-robustness and continuous exploration towards novel and better designs. Therefore, there is a need for enhanced evolutionary techniques able to overcome the design weaknesses of the current genetic based approaches.

This section focuses mainly on the evolutionary part of an EHW architecture and presents four different evolutionary methodologies, which are compared with a conventional GA that has been widely employed for the realization of FIR filters. Moreover, specially tailored crossover and mutation operators are employed in order to further increase the efficiency of the applied evolutionary techniques. Three FIR filters (19-tap high-pass, 26-tap low-pass and 31-tap low-pass) are used as a benchmark for the evaluation of the 5 evolutionary techniques. These techniques employ the same reconfigurable substrate for realizing the three filters. The custom VLSI reconfigurable structure that has been presented and evaluated in Section 5.6 composes the physical evolution environment. Hence, the hardware topology employs the POF technique to design the arithmetic unit of the filters and multiplication is performed by combining additions, subtractions and binary shifts. Resolving the best possible combination of primitive operations for the realization of a coefficient is a complicated task and requires a powerful and specially tailored EA. Moreover, as mentioned in Section 5.6.2, the topology of the hardware supports the temporal minimization of the search space and the gradual evolution of the coefficient-set. These two approaches are crucial because they contribute to the elimination of the hardware dependencies, which are created between different coefficients. Previous research work has been published in the field of discrete-time filter evolution. In [84], the authors concern themselves with evolving the transfer functions of high-order filters using multi-objective criterion. However, this design framework does not appear competitive in terms of the time needed for design or adaptation, since it takes up to 50,000 generations for the realization of the targeted filter. Computationally, this corresponds to 1.5 days effort of a 500 MHz Pentium computer. Another notable work concerns the implementation of POF FIR filters [43]. It utilizes a simulated reconfigurable hardware platform that has been designed for fault-tolerance via redundancy. The authors achieved implementation of a symmetrical 31 order low-pass FIR filter within 6700 generations. From these figures, it can be deduced that the employed EA needs 6700 generations to evolve the half coefficient-set (16 coefficients). Moreover, the average achieved fitness-score is reported to be bigger than 98.5%. Further work [103] has been done using POF technique to implement FIR filters. This work takes a Data-Flow Graph (DFG) of a filter as input and performs evolution in the frequency domain in order to implement optimized filters in terms of silicon area and delay. However, the designed filters are of low order and they can not be applied on the majority of electronic applications. Moreover, their realization may take up to 20000 generations, which is an unacceptable performance for real-time applications. In [109], the authors demonstrate the capability to adapt on-line the coefficients of FIR filters. It composes an on-chip solution where the GA and the evolving filter design itself are implemented on the same chip. Results have shown that this framework is very effective in terms of time needed for design and/or adaptation. However, on the contrary with the two previous POF architectures, during evolution the structure is fixed and only the value of the coefficient-set is evolved. Furthermore, conventional multipliers are employed for the realization of the arithmetic unit of the filters that produce significant overhead in terms of silicon area and power dissipation. In [112], a similar work is demonstrated, where both the EA and the evolved filter exist in the same chip. The evolution of a 8-tap is presented. Nevertheless, this approach has the advantages and disadvantages as the work presented in [109].

6.2 Limitations of Existing Evolutionary Techniques

There are several paradigms that demonstrate the successful use of evolutionary computation to electronic VLSI applications [40], [67], [92], [100]. However, the performance of the employed EAs is constrained by deceptive problems, such as *pleiotropy* and *epistasis*, mentioned in Section 5.4.1, which prevent the algorithm from reaching a global maximum within a time period that satisfies the time limitations of the targeted application. Typical examples can be found in [43] where the evolution of the targeted filters either takes a significant number of generations or the evolved filters are not very accurate. Another problem that conventional evolutionary techniques encounter in the evolution of filters is their incapability of drastically changing the hardware configuration after a significant number of coefficients, within the coefficient-set, have been partially satisfied.

6.3 Modified Crossover and Mutation Operations

Conventional reproduction operations, such as crossover and mutation, lack capability of performing beneficial changes in the hardware configuration whenever evolution seems to have stopped. The *modified crossover* operation has been applied on the evolutionary techniques that simultaneously evolve the whole coefficient-set. Consequently, the *modified mutation* has been applied on the evolutionary techniques that perform gradual filter evolution. Figure 6.1 presents a typical example that describes how the *modified crossover* operation improves the efficiency of the employed EA. According to this paradigm the targeted coefficients are the integer numbers 18, 208 and 64. It can be seen that before crossover only the two coefficients (1st and 3rd) have been successfully evolved, while the intermediate one (208) has been partially realized. For the intermediate coefficient the fitness-score as it is given by the percentage ratio between the ideal and the evolved one is 92.8. Therefore, by using conventional reproduction operations is very difficult to improve the fitness-score of the second coefficient without negatively affecting the fitness-score of the 1st and 3rd coefficient.

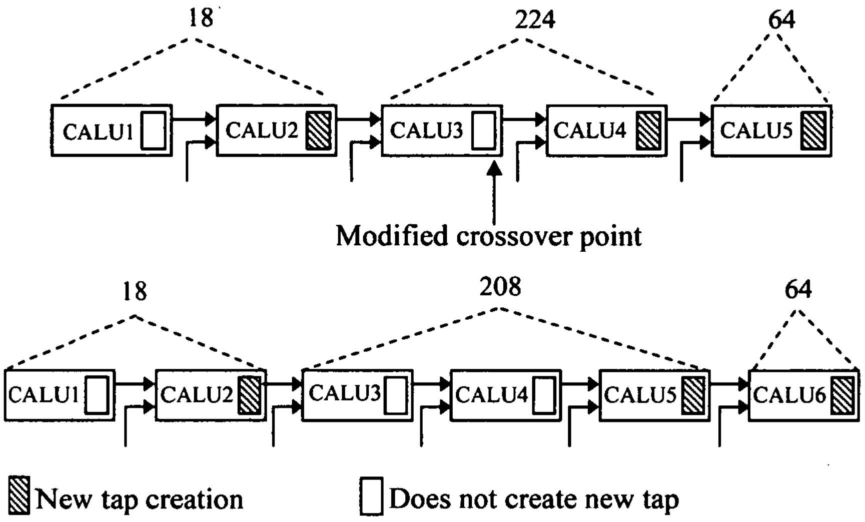


Figure 6.1: Description of modified-crossover operation, which assigns an additional CALU in the realization of the 2nd coefficient without affecting the configuration of the rest coefficient-set

Based on the *modified crossover*, a cross-point is randomly selected and then a new CALU, which does not create a new tap (white box), is inserted to contribute in the accurate realization of the coefficient. After the crossover operation two offsprings are generated. In the former, the inserted CALU performs addition, while in the latter the CALU performs subtraction. In this manner, it is guaranteed that one of the two offsprings will achieve better a fitness-score than the parents. However, the rate of the *modified crossover* operation must be kept low because otherwise the EA may waste significant hardware resources. Therefore, it is more efficient when multi-objective optimization (filter accuracy and efficient hardware utilization) is employed.

The *modified mutation* facilitates, during the gradual evolution of a filter, the insertion of a new CALU for the realization of the currently evolved coefficient. It has the same target with that of *modified crossover*. Contrary to the *modified crossover*, the point where mutation is performed is known because during gradual evolution the EA keeps monitoring how many CALUs are utilized per coefficient and the total number of the utilized coefficients.

6.4 Evolutionary Techniques

Five evolutionary techniques are presented and their ability to evolve three digital FIR filters is evaluated. The first two evolutionary techniques are conventional and do not incorporate any novelty. The rest perform gradual filter evolution. In addition to this, they also introduce a combination of *incremental* and *multi-objective* mechanisms to achieve better results in terms of hardware utilization.

6.4.1 Conventional EA

The conventional EA employs the frame of a generic GA that utilizes the $\mu+\lambda$ methodology [32]. According to this the initial population consists of 50 chromosomes (parents) and is then extended to 100 by adding another 50 chromosomes (children). The children are generated by crossover and mutation operations, which are applied on randomly selected chromosomes, taken from the initial population bank. In the evolution of the 19-tap high-pass filter, two runs were performed. In the first the conventional EA incorporates only conventional reproduction operators, while the second takes also advantage of the *modified-crossover* that has been introduced in

Section 6.3. The rate of *modified-crossover* is 20%, while the rate of conventional crossover is 80%. During conventional crossover, the two parents exchange information that corresponds to the configuration of one or more CALUs. Furthermore, the mutation rate varies dynamically between 0 and 20%. Considering that the chromosome size is 1204 bits (encodes 200 CALUs), it can be deduced that the average mutation rate is approximately 0.008% per gene. Consequently, after evaluation and ranking, 50 survivors are selected through deterministic selection to compose the new parent population. The employed fitness-function is described in equation (6.1) and has been widely used in previous attempts of digital FIR filters evolution [43]. Similarly in [109] and [112], the fitness-function is considered as the absolute difference between the ideal and the evolved coefficient-set. The fitness-function considers the impulse response of the evolved filter and compares it with the ideal one. Hence, it calculates the ratio between the ideal and the matched evolved coefficient depending which one is bigger (the ratio cannot be bigger than 1). Then, it adds up the sub fitness-scores to calculate the total one.

$$\text{Fitness}=\sum_{I=0}^{\text{taps}-1}\begin{cases}\text{coeff}_{i\_ideal}/\text{coeff}_{i\_evolved},\ \textit{when}\ \text{coeff}_{i\_ideal}<=\text{coeff}_{i\_evolved}\\ \text{coeff}_{i\_ideal}/\text{coeff}_{i\_evolved}+0.001,\ \textit{when}\ \text{coeff}_{i\_evolved}=0\ ,\\ \text{coeff}_{i\_evolved}/\text{coeff}_{i\_ideal},\ \textit{when}\ \text{coeff}_{i\_ideal}>\text{coeff}_{i\_evolved}\end{cases}\quad(6.1)$$

6.4.2 Conventional MOEA

This section focuses on identifying evolutionary techniques that secure accurate implementation of filters with the minimum possible hardware resources. Hence, there are two conflicting objectives for which a trade-off must be found. This evolutionary technique employs a multi-objective evolutionary algorithm (MOEA), which uses Pareto domination [51] to guide the search. Ideally, a MOEA returns the Pareto optimal-set, the solutions not dominated by any other solution in the search space [46]. Unless the system is provided with additional information about the properties of the desired circuit, there is no reason to prefer a Pareto point over another. The Pareto optimal set of individuals can be extracted from the final population and the designer can then choose the most appropriate design for a particular application [104].

Similarly to the previous technique, this method employs the *modified-crossover* operation and it uses exactly the same rates for genetic reproduction. Specifically, the MOEA employs an initial population, which after evaluation, is ranked twice according to the two targeted objectives. Then 25 survivors are selected from the first population bank (ranking based on accuracy) and 25 survivors from the second bank (ranking based on minimum hardware utilization). However, according to the followed strategy, the 25 selected chromosomes from the first bank should not be worse in the second objective than the 80th chromosome of the second bank. Hence, the chromosomes, which are on the top 25 in one objective and worse than the 80th chromosome in the other objective, are discarded. Similarly, the best 25 chromosomes are selected from the second population bank. The new 50 chromosomes will compose the new parent population. Hence, after each generation, the population contains solutions that adequately satisfy both objectives and the evolution of a FIR filter is not guided towards one objective, resulting to an impractical implementation.

6.4.3 Generic Gradual Evolution

This evolutionary technique has been specially tailored to overcome weaknesses of previously employed EAs for the implementation of digital filters. The proposed EA has a correlative conjunction with the proposed reconfigurable hardware topology and therefore it cannot be widely used for any reconfigurable structure. The novelty of this EA is firstly based on the instantaneous minimization of the search space of the algorithm by constraining the genetic reproduction (mutation) only to the effective part of the chromosome that participates on the currently evolved coefficient. This evolutionary technique has been presented in details in Section 5.6.3.

6.4.4 Incremental Evolutionary Technique

This technique is mainly based on the concept of the previous one (*generic gradual evolution*) but it specifically focuses on the implementation of filters from the power consumption perspective. On the contrary to the *generic gradual evolution*, it does not arbitrarily assign a maximum of 6 CALUs for the realization of a single coefficient.

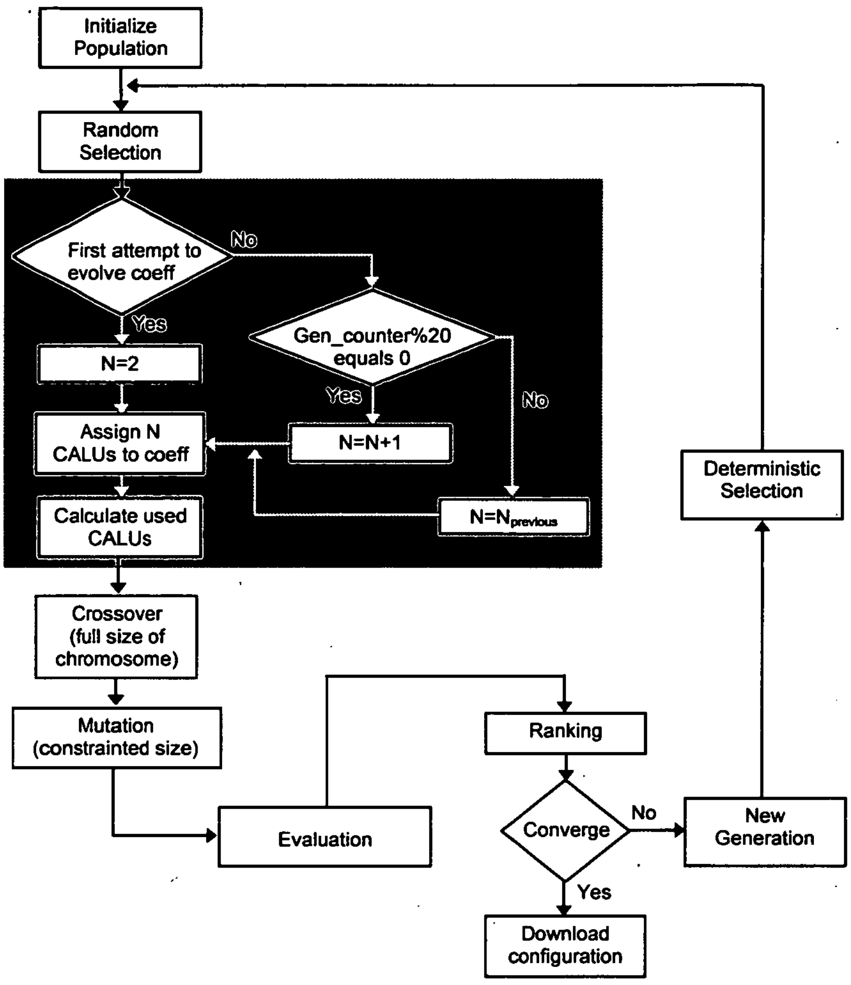


Figure 6.2: Flow-diagram of the *Incremental* evolutionary technique

It employs a mechanism that initially assigns two CALUs to each coefficient under evolution and consequently considers the number of generations that have passed. Whenever, a predefined number of generations pass, the mechanism increases by 1 the number of the assigned CALUs. Figure 6.2 depicts a detailed flow diagram that describes the employed evolutionary technique.

The blocks within the shaded box compose the part of the algorithm that is in charge of assigning the minimum possible number of CALUs for the realization of a coefficient. Hence, whenever the evolution of a new coefficient starts, the mechanism always assigns two CALUs. However, if the evolved coefficient cannot approach the ideal one at a satisfactory percentage within the next 20 generations, then the algorithm increases by one the number of the assigned CALUs every 20 generations. The most significant action that is hidden behind this methodology is the capability to further reduce the search area of the EA compared with the *generic gradual evolution* technique.

Unlike mutation that is restricted to the parts of the chromosome imposed by the mechanism shown in Figure 6.2, crossover always occurs on the full chromosome. The rate of crossover dynamically varies per reproduction. Thus, the two parents may exchange from 0 up to 5% of their information to generate the offsprings. The rate of mutation also varies but its maximum rate is decreased every time the evolutionary mechanism increases the assigned number of CALUs. This approach is quite beneficial because as the evolution of a coefficient proceeds less reproduction is required that corresponds to finer configuration medications. Table 6-1 depicts how the mutation rates are moderated based on the assigned number of CALUs. In all scenarios shown in this table, the minimum rate of mutation is 0%.

TABLE 6-1: MUTATION RATES PER GENE

Number of CALUs	Maximum (%) rate
2	50
3	33
4	25
5	23
6	22
7	19

6.4.5 Combination of Incremental and MOEA

This evolutionary technique is an amalgam of the *incremental evolutionary technique*, introduced in the previous section and a MO algorithm. Its target is to further decrease the utilized piece of hardware by keeping the convergence time considerably the same. This EA employs the same pseudo-code (pseudo-code_1) as the *generic gradual evolution*, described in Section 5.6.3, in order to calculate how many CALUs are utilized for a given hardware configuration.

The employed MOEA considers whether the different number of CALUs can justify a small improvement in the accuracy of the filter under evolution. Therefore, if between two individuals there is a negligible improvement in the filter's accuracy and there is also a differentiation of two or more CALUs in their hardware configuration, the individual that uses the redundant hardware is discarded in the *deterministic selection* phase of the algorithm, depicted in Figure 6.2. Specifically, the pseudo-code 2 that describes the functionality of the MOEA is given below. The MOEA is applied only on the 10 first chromosomes that have the highest fitness-score in terms of accuracy. Consequently, each one of these is compared with the rest in terms of the number of utilized CALUs. Pseudo-code_2 presents the criteria according to which the MOEA evaluates the 10 selected chromosomes. For example the first criterion says that if the second best scored chromosome (chromosome_2) has evolved fewer taps (*currently_evolved_tap_2*) than the best chromosome (*currently_evolved_tap_1*) then the best chromosome (chromosome_1) is not discarded only if it has utilized less than 6 CALUs, compared with the number of CALUs of the second chromosome. On the other hand, if the two compared chromosomes have evolved the same number of taps then the comparison lies on how accurately they have evolved the current one. Hence, if there is a smaller percentage difference than 90, the fitter chromosome (chromosome_1) is discarded, only if its configuration uses more than 4 CALUs compared with the configuration of the second chromosome (chromosome_2). Similarly, there are two more criteria for cases where the fitness-scores of the two examined chromosomes differ less than 40% and 10%, respectively. Finally, the thresholds in pseudo-code_2 have been identified in a practical manner after many runs of the MOEA for the evolution of different filters. The selected ones present better results in terms of power consumption compared with the *incremental evolutionary technique* and still the algorithm does converge in within similar time.

Pseudo-code 2:

```
if (currently_evolved_tap_1 > currently_evolved_tap_2){
    if (number_of_CALUs_1 - number_of_CALUs_2 > 5){
        selected_chromosome = chromosome_2;
    }
    else {
        selected_chromosome = chromosome_1;
    }
}
else if (currently_evolved_taps_1 == currently_evolved_taps_2){
    if (fitness_score_1 - fitness_score_2 < 90%) &&
    (number_of_CALUs_1 - number_of_CALUs_2 > 4){
        selected_chromosome = chromosome_2;
    }
    else if(fitness_score_1 - fitness_score_2 < 40%) &&
    (number_of_CALUs_1 - number_of_CALUs_2 > 2) {
        selected_chromosome = chromosome_2;
    }
    else if(fitness_score_1 - fitness_score_2 < 10%) &&
    (number_of_CALUs_1 - number_of_CALUs_2 > 1) {
        selected_chromosome = chromosome_2;
    }
    else{
        selected_chromosome = chromosome_1;
    }
}
```

6.5 Simulation Environment

For the evaluation of the five different evolutionary techniques the following FIR filters were employed: a) a 19-tap high-pass, b) a 26-tap low-pass filter and c) the 31-tap low-pass filter introduced in Section 5.6.4. Table 6-2 depicts the ideal coefficient-set of the three filters.

TABLE 6-2: SPECIFICATION OF IDEAL FILTERS

19-tap high-pass filter		26-tap low-pass filter		31-tap low-pass filter	
$C_{0,18}$	395	$C_{0,25}$	92	$C_{0,30}$	-59
$C_{1,17}$	-518	$C_{1,24}$	336	$C_{1,29}$	0
$C_{2,16}$	-1131	$C_{2,23}$	668	$C_{2,28}$	96
$C_{3,15}$	3524	$C_{3,22}$	759	$C_{3,27}$	0
$C_{4,14}$	-1868	$C_{4,21}$	109	$C_{4,26}$	-220
$C_{5,13}$	-4862	$C_{5,20}$	-1541	$C_{5,25}$	0
$C_{6,12}$	7005	$C_{6,19}$	-3691	$C_{6,24}$	461
$C_{7,11}$	6724	$C_{7,18}$	-4899	$C_{7,23}$	0
$C_{8,10}$	-30536	$C_{8,17}$	-3330	$C_{8,22}$	-876
C_9	42678	$C_{9,16}$	2085	$C_{9,21}$	0
		$C_{10,15}$	10574	$C_{10,20}$	1606
		$C_{11,14}$	19410	$C_{11,19}$	0
		$C_{12,13}$	25060	$C_{12,18}$	-3171
				$C_{13,17}$	0
				$C_{14,16}$	10326
				$C_{15,15}$	16384

It can be seen that the evolution of these filters is a very challenging task because of the wide dynamic range of the associated coefficient-sets. The range of the high-pass filter lies from 2^9 to 2^{16}, while the 26-tap and 31-tap low-pass have a range from 2^7 to 2^{15} and 2^7 to 2^{14}, respectively. All the three filters are symmetrical. The first have been designed with Matlab tools, while the third one was taken from the industrial design of low-power filter cores for hearing aids and it has been developed in joint collaboration with Bernafon LTD and the University of Edinburgh [126].

In the simulations three parameters (*accuracy*, *number of CALUs*, *number of generations*) are considered to evaluate the efficiency of each evolutionary technique to evolve FIR filters. Obviously, the evolution of very accurate filters is the first priority of this work and therefore throughout the evaluation process evolutionary techniques that provide higher accuracy are preferred to these that present poorer frequency response and better results in terms of hardware utilization and convergence time. It must be mentioned here that for the two conventional evolutionary techniques, evolution stops after a number of generations beyond which the fitness-score does not provide any substantial improvement. On the other hand, the three proposed evolutionary techniques (*generic gradual evolution*, *incremental evolution* and *incremental with MO evolution*) secure accuracy that varies from 99.7% to 99.98% according to the bit-length of the targeted coefficient.

6.6 Comparisons

6.6.1 19-tap High-pass Filter

For the evolution of the 19-tap high-pass filter, the performance of all the 6 EAs is compared. However, because the three conventional evolutionary techniques do not consider the gradual evolution of the targeted coefficient-set, the simulation results have been grouped according to this criterion. Nevertheless, the performance, in terms of accuracy, power-consumption and convergence-time, for the 6 evolutionary techniques can be clearly evaluated.

Figure 6.3 depicts the evolution process of the three conventional evolutionary techniques. It must be mentioned here that two of them use the *modified-crossover* that has been introduced in Section 6.3.

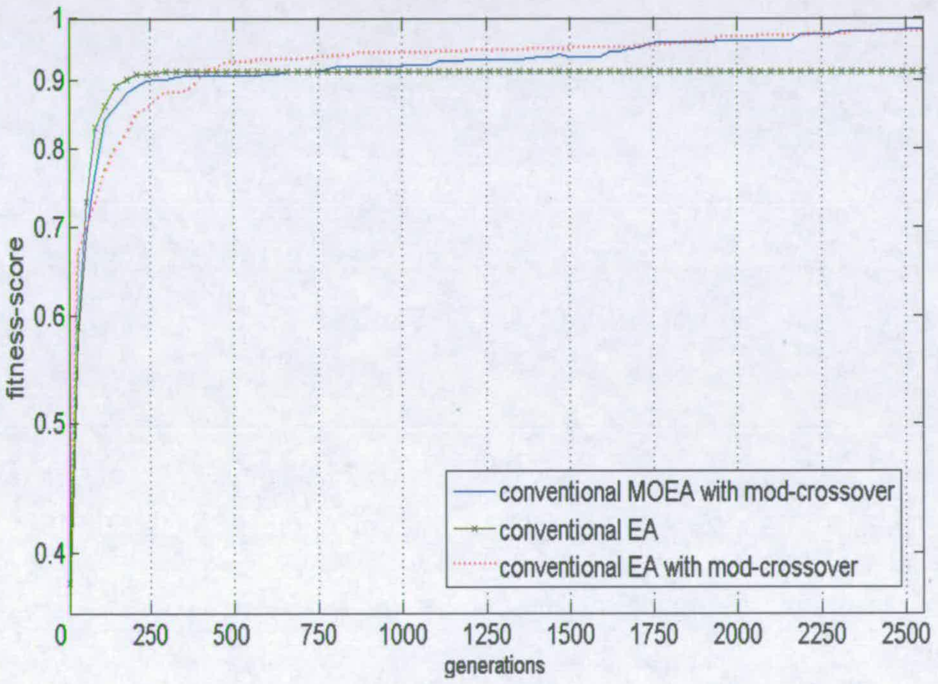


Figure 6.3: Evolution process of the three conventional EAs

Figure 6.4 shows how accurately the achieved solution matches the ideal frequency response after 2550 generations. Considering both figures it can be deduced that the conventional MOEA using *modified-crossover* achieves the most accurate result (fitness-score equals to 0.98). However, even this solution does not fully satisfy the targeted filter. Its frequency response, graph marked with squares, presents similar pass-band characteristics but the achieved attenuation in the stop-band is poorer than the ideal one. In addition to this, the hardware configuration that the MOEA resolved utilizes 72 CALUs.

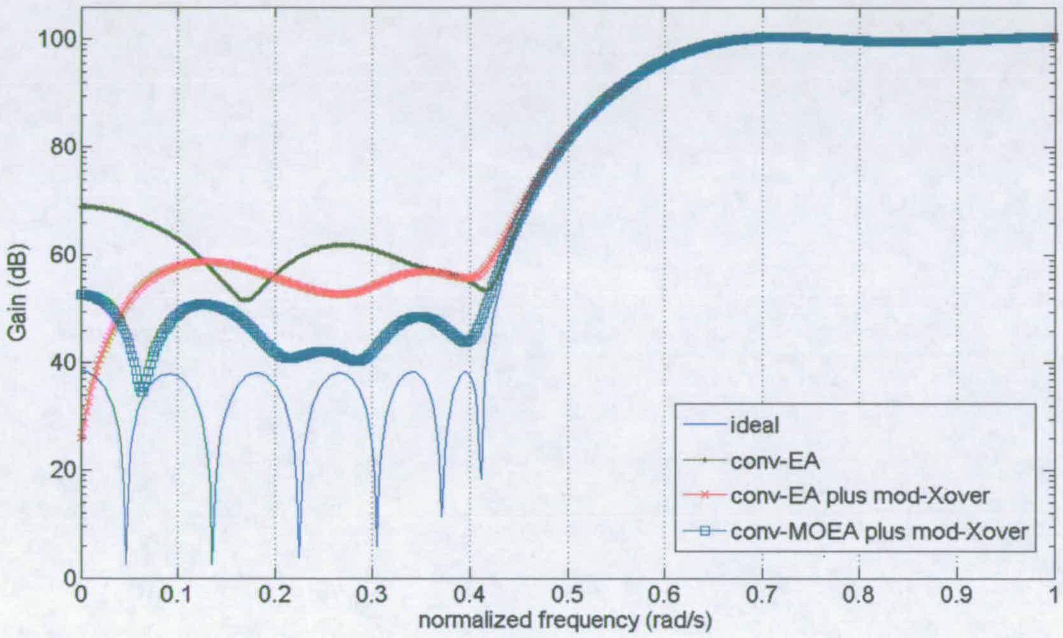


Figure 6.4: Evolved frequency response of the three conventional EAs after 2550 generations

The conventional EA using *modified-crossover* and the *conventional EA* achieve fitness-scores 0.976 and 0.913 and utilize 196 and 34 CALUs, respectively. These results verify the weaknesses (mentioned in Section 6.2) that conventional EAs encounter. Specifically, the poor frequency response evolved by the conventional EA utilizes only 34 CALUs, which cannot accurately evolve the specified filter. This happens because the employed EA is incapable of drastically changing the hardware configuration towards fitter solutions after an average solution has been resolved. On the other hand, the conventional EA with *modified-crossover* wastes a significant number of hardware resources (196 CALUs) because it utilizes new CALUs to improve the fitness-score without gaining the maximum from each CALU.

Figure 6.5 depicts the frequency responses that have been evolved by the guidance of the *generic gradual*, *incremental* and *incremental with MO* evolutionary techniques. It is apparent that all the three evolutionary techniques resolved hardware configurations that satisfy the filter's specification.

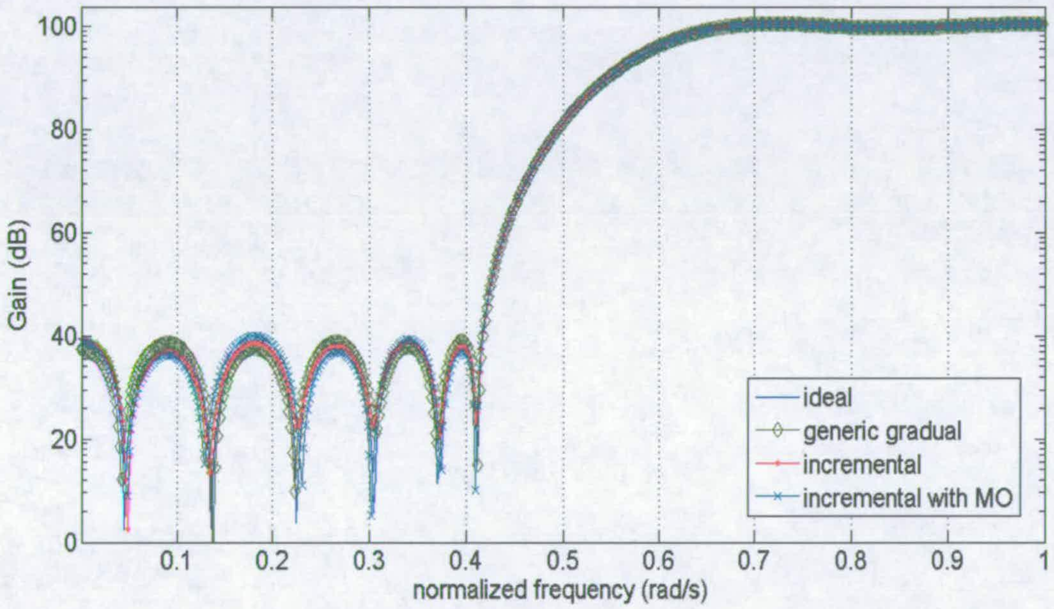


Figure 6.5: Evolved frequency response of the 19-tap high-pass FIR filter using the three EAs

Since all the three novel evolutionary techniques evolve an accurate 19-tap filter, the other two parameters (*number of generations to converge* and *number of utilized*

CALUs) are examined to find out which technique is more efficient. Figure 6.6 depicts that the *gradual* evolutionary technique requires 703 generations in order to evolve the 19 coefficients.

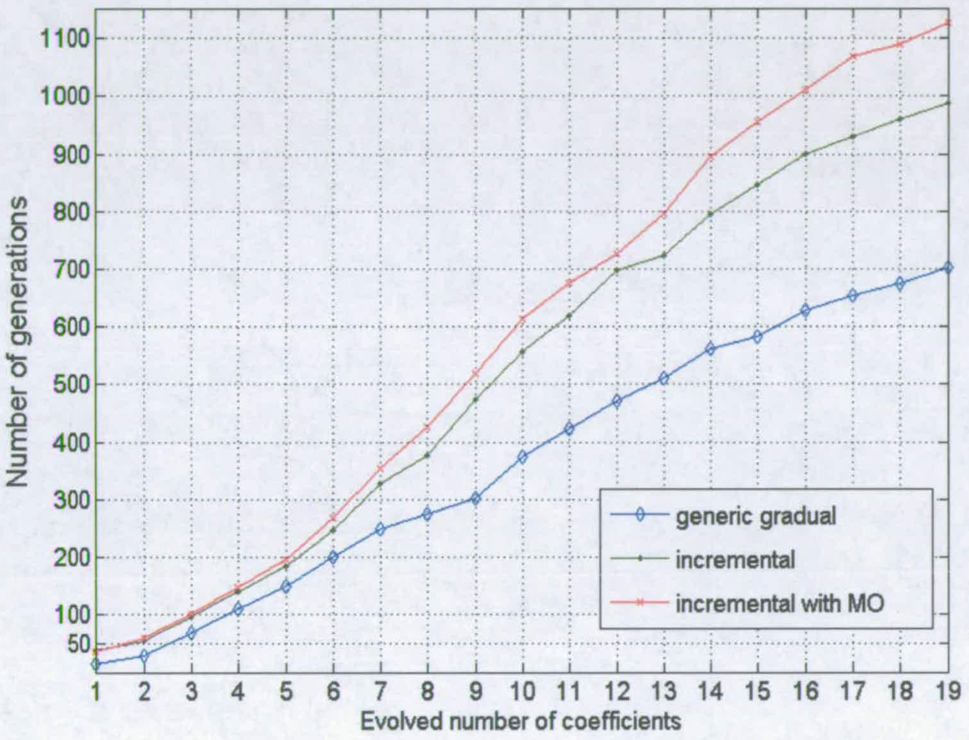


Figure 6.6: Evolution process of the three EAs for the evolution of the 19-tap high-pass FIR filter

This implies that the *gradual* evolutionary technique needs in average 37 generations per coefficient. This is an outstanding performance considering the wide dynamic range of the numbers that compose the coefficient-set and the word-length, as well. Moreover, the *gradual* evolutionary technique presents 40% and 60% reduction in the number of generations compared with the *incremental* (987 generations) and *incremental with MO*

(1123 generations) techniques, respectively. However, even the MO solution is superior to the introduced conventional evolutionary techniques and these presented in [43].

Figure 6.7 shows how efficiently the three novel evolutionary techniques utilize the hardware resources to evolve the 19-tap filter.

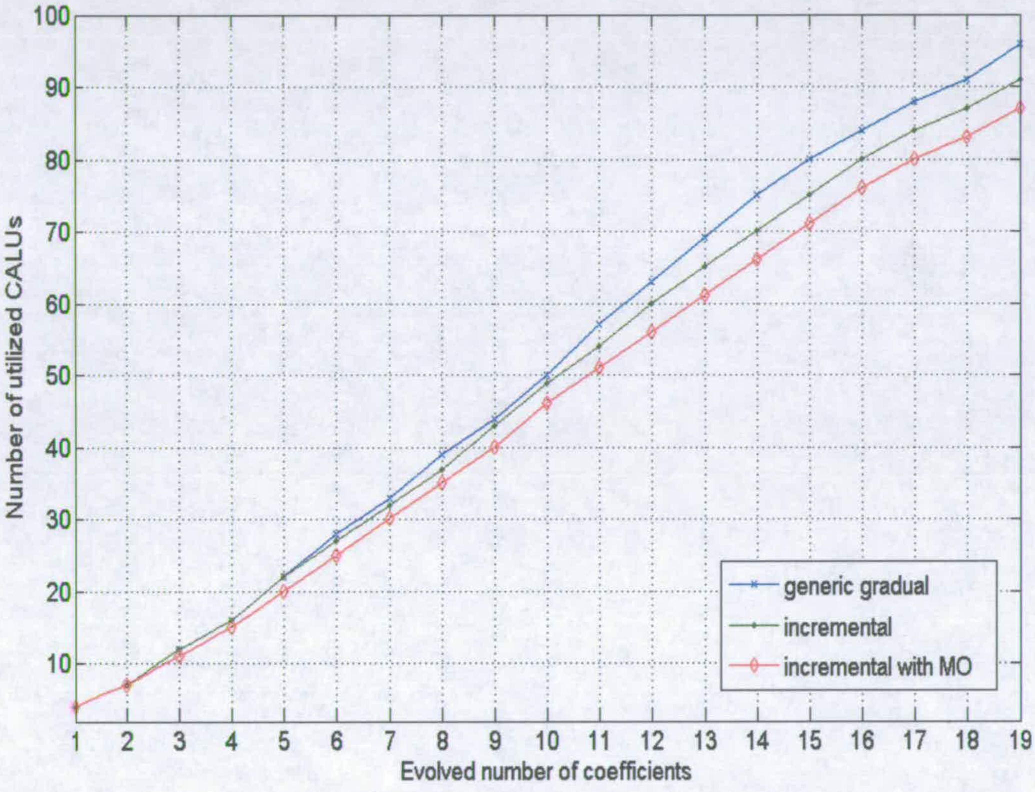


Figure 6.7: Hardware utilization of the three EAs for the evolution of the 19-tap high-pass FIR filter

The x-axis shows which coefficient is currently under evolution, while the y-axis presents the total number of CALUs that have been employed so far. The achieved results contradict those in Figure 6.6, as the *incremental with MO* evolutionary technique provides the most challenging solution in terms of power-consumption (87 CALUs).

Consequently, the *incremental* technique employs 91 CALUs and the *gradual* 96 CALUs. Thus, it can be deduced that there is a trade-off between the convergence-speed and the hardware resources that the EA employs for the filter implementation. Hence, the choice of the best evolutionary technique actually depends on the application that is targeted. In all three evolutions the accuracy is guaranteed so it is up to the designer to select one between the lowest power-consumption solution (*incremental with MO*), the one with the shorter convergence-time (*gradual*) or the intermediate solution that its results are between the highest and lowest achievements.

6.6.2 26-tap Low-pass Filter

For the evolution of this filter, only the 3 novel evolutionary techniques using the gradual approach of evolution were employed. The conventional methodologies have been excluded from this paradigm since from the first one (19-tap high-pass filter), it has been shown that they are inferior to the proposed ones. Figure 6.8 depicts the three evolved frequency responses that correspond to the 26-tap low-pass filter. In comparison with the previous paradigm, there is a small deviation in the stop-band region of the evolved filters.

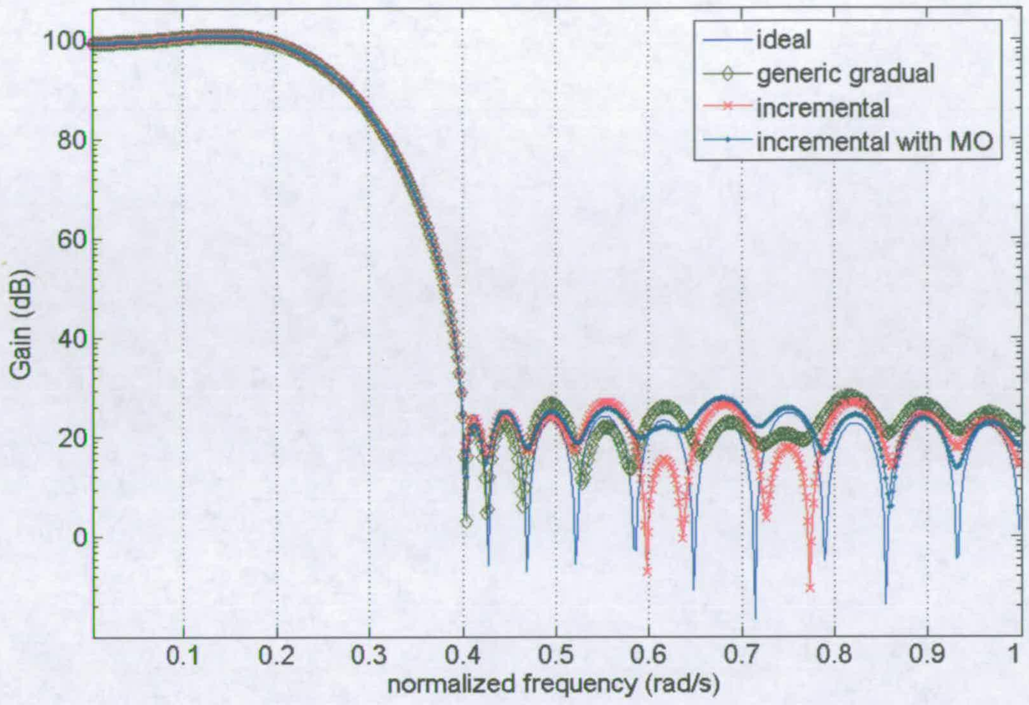


Figure 6.8: Evolved frequency response of the 26-tap low-pass FIR filter using the three EAs

However, this deviation is negligible because it does not exceed 1 db of attenuation. Figure 6.9 shows the evolution process of the three implementations. Similar to the previous example, the *gradual* evolutionary technique requires considerably fewer generations (880 generations) than the *incremental* (1053 generations) and the *incremental with MO* (1186 generations). However, if Figures 6.9 and 6.10 are considered together, it can be deduced that the *incremental with MO* evolutionary technique requires more generations compared with the *gradual* at specific points (after the evolution of the 7th and 19th coefficient) where the *gradual* wastes hardware resources for the filter's implementation. From a hardware perspective, the *gradual* evolution needs 119 CALUs in total for the implementation of the filter, while the *incremental* and the *incremental with MO* need 118 and 114, respectively. This implies that the *incremental with MO* evolution introduces a reduction in the utilized hardware resources of 4.2% and 3.4% over the *gradual* and *incremental*, respectively.

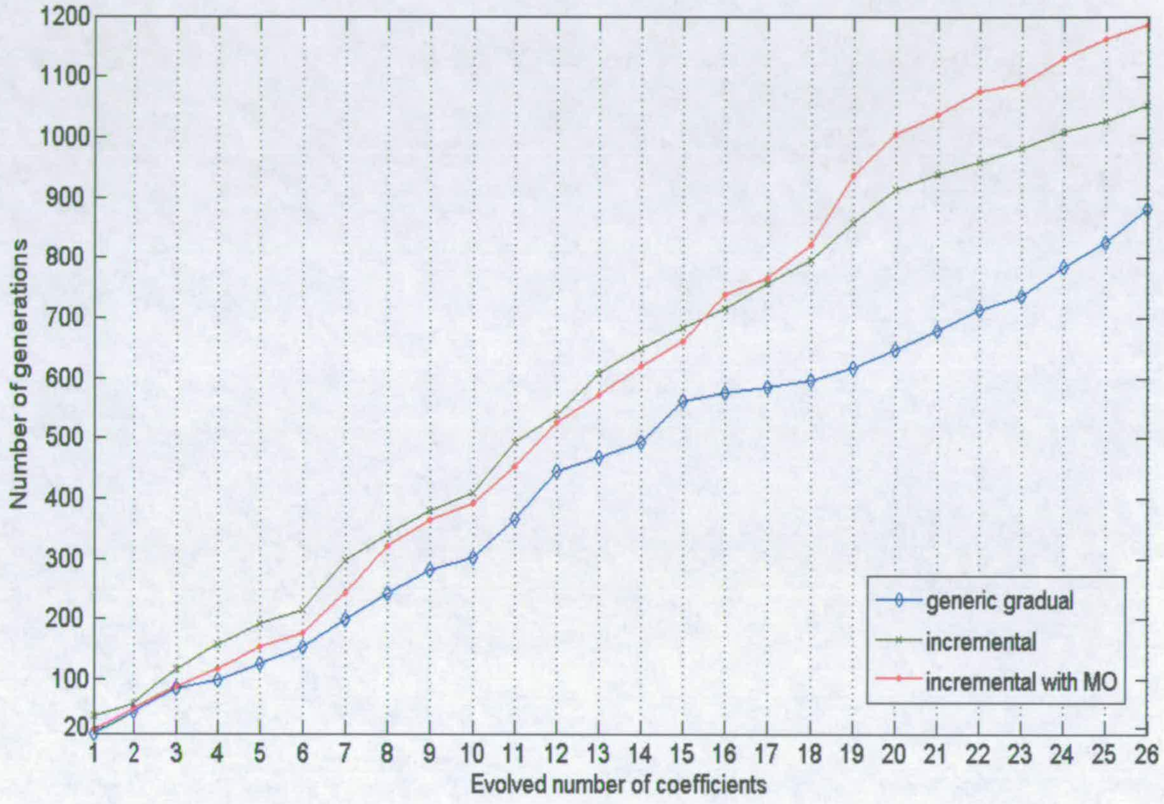


Figure 6.9: Evolution process of the three EAs for the evolution of the 26-tap low-pass FIR filter

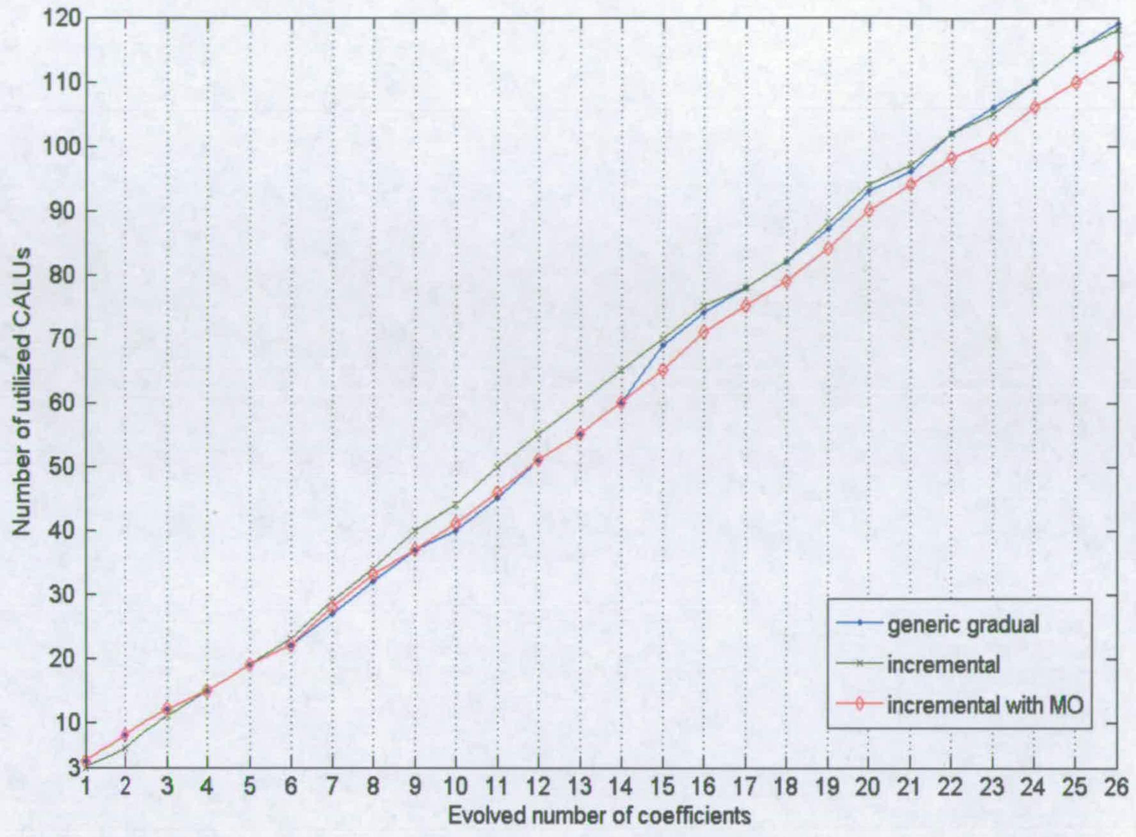


Figure 6.10: Hardware utilization of the three EAs for the evolution of the 26-tap low-pass FIR filter

6.6.3 31-tap Low-pass Filter

This paradigm compares the three novel evolutionary techniques against the evolvable architecture presented in [43]. Therefore, the evaluation has been performed for the implementation of the filter (31-tap low-pass), shown in Table 6-2. Figure 6.11 depicts the high accuracy (from 99.7% to 99.98%) that is achieved by all the evaluated evolutionary techniques. Specifically, based on the GA that has been employed for the realization of the filters, the targeted accuracy of each coefficient depends on each value. Therefore, accuracy of 99.7% is sufficient for a coefficient that is smaller than 100, while it is very poor for a coefficient with value 10000. Hence, the employed GA utilizes different scales of accuracy for different coefficients based on their values. On the other hand, the authors in [44] suggest that the average achieved accuracy is 97.3%. This figure may be sufficient for small coefficient bit-lengths but for higher ones it produces such deviations between the evolved and ideal coefficients that are adequate for the filter to malfunction. Therefore, the quality of the implemented filter is not always secured.

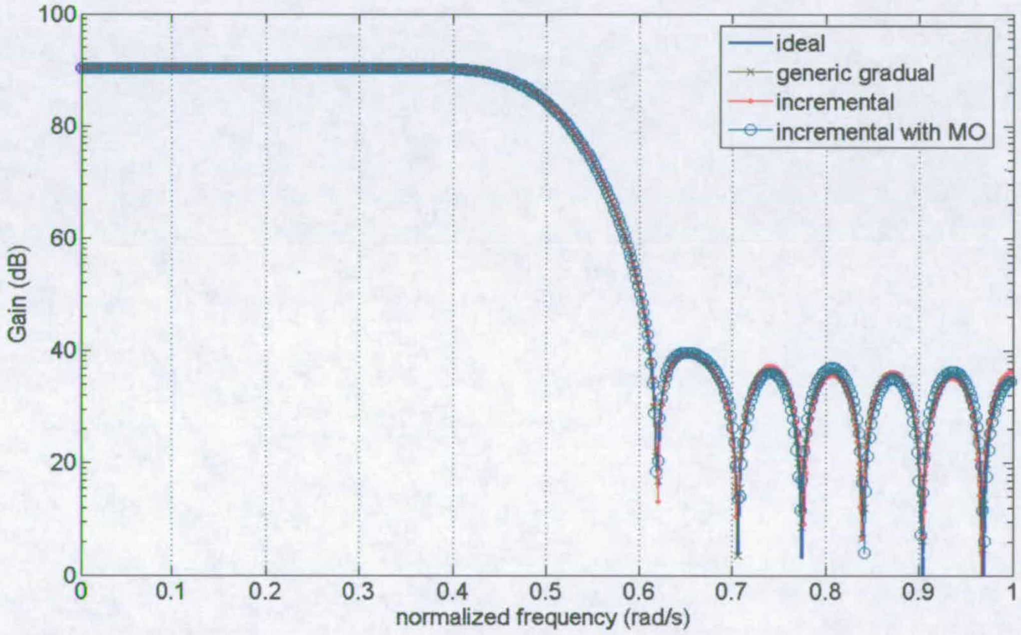


Figure 6.11: Evolved frequency response of the 31-tap low-pass FIR filter using the three EAs

Moreover, the evolvable framework in [44] takes 6700 generations to resolve the best possible hardware configuration. Figure 6.12 shows that the novel evolutionary techniques, *gradual*, *incremental* and *incremental with MO* need 683, 672 and 778 generations, respectively. These results imply that the proposed evolvable framework (custom-reconfigurable hardware and evolutionary techniques) provides more accuracy and a reduction in the number of generations by 8.6 to 9.8 times.

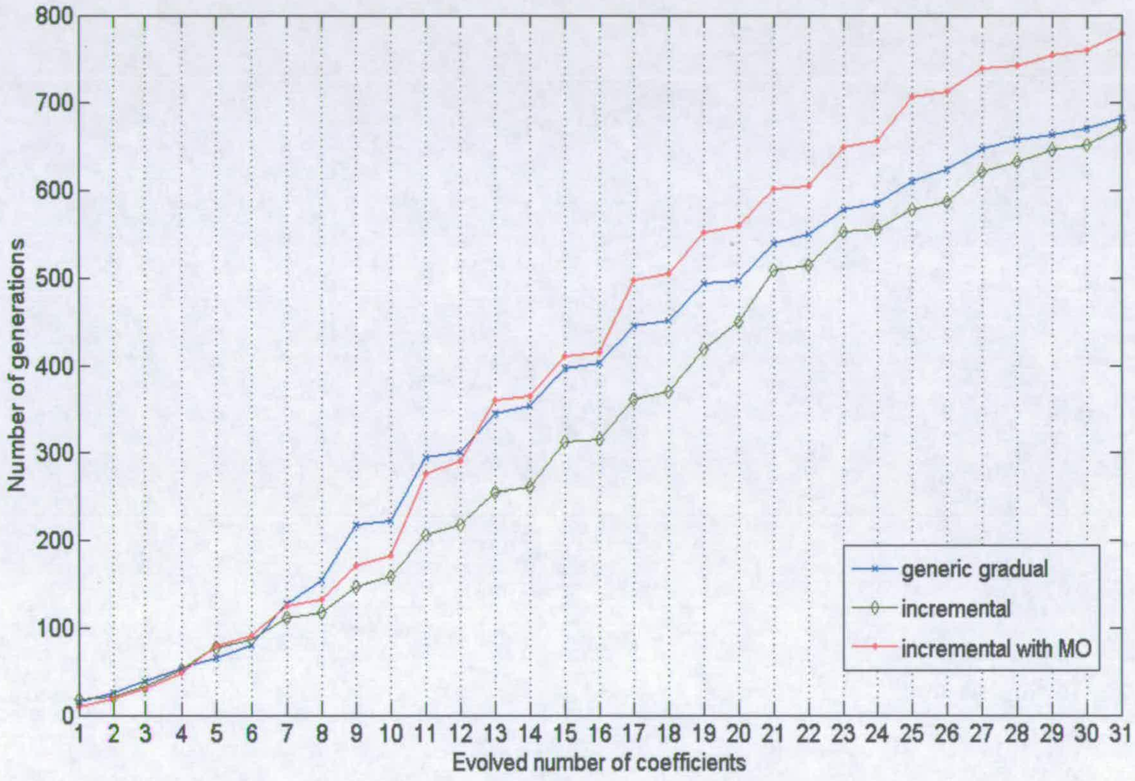


Figure 6.12: Evolution process of the three EAs for the evolution of the 31-tap low-pass FIR filter

The hardware utilization of the *incremental with MO* evolutionary technique achieves the most impressive result in terms of power-consumption (Figure 6.13). It utilizes 84 CALUs to realize both the arithmetic and accumulation unit of the 31-tap filter, shown in Table 6-2. The authors in [44] suggest that their architecture needs 81 PALUs on average (calculated in Section 5.6.4.2) for the realization of the arithmetic unit of the filter. Hence, it employs slightly fewer hardware resources but not without cost in both accuracy and convergence-time. This architecture manages a reduction in hardware utilization by re-using partial products from previous hardware components. The proposed architecture does not take advantage of hardware re-usability because this creates hardware dependencies between different coefficients within the same coefficient-set and prevents the accurate implementation of filters. Instead hardware economy is achieved through the specially tailored reconfigurable hardware and the three proposed evolutionary techniques.

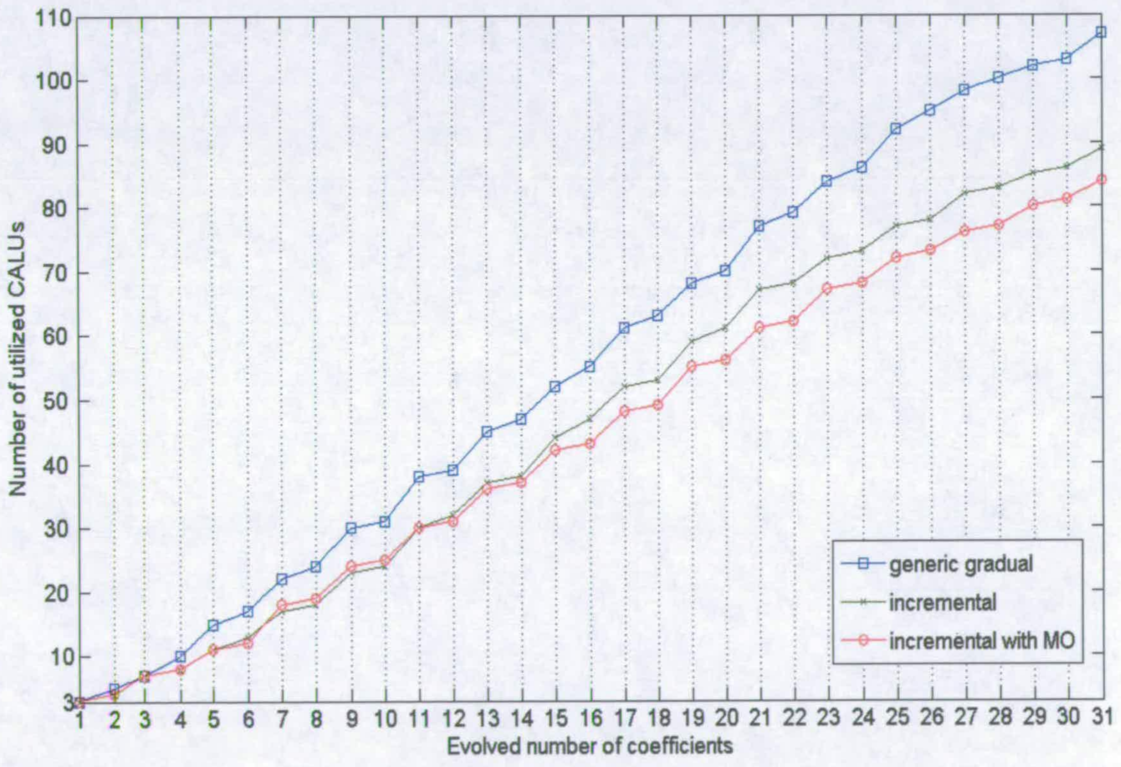


Figure 6.13: Hardware utilization of the three EAs for the evolution of the 31-tap low-pass FIR filter

6.7 Conclusions

This chapter has presented three novel evolutionary techniques that prove their superiority to conventional single and multi-objective evolutionary methodologies. Table 6-3 summarizes the results obtained by the three proposed evolutionary techniques for the realization of the 19, 26 and 31 order filters. Simulation results have proved that the evolution of any order FIR filter can be guaranteed with accuracy (not less than 99.7% for small coefficients and 0.998 for bigger word-lengths) that improves on previously employed evolutionary techniques. Moreover, real-time implementation and adaptation of FIR filters is achieved. The best and worst cases filter implementation takes between 8.6 and 9.8 times fewer generations than a previously employed EA. The novel evolutionary techniques guarantee a low average number of generations per coefficient independent of the filter's size and the associated complexity. Moreover, high accuracy, efficient hardware utilization and short speed-convergence per coefficient can be achieved for any order of FIR filters, due to the novelty introduced in the custom reconfigurable hardware substrate and the employed evolutionary techniques that are used to guide evolution.

Hence, the reconfigurable hardware substrate presented in Section 5.6 in collaboration with the proposed evolutionary techniques can enable the accurate and efficient, in terms of power consumption and time for adaptation, implementation of the JPL/Boeing gyroscope's electronics. In the next chapter the implementation of the gyroscope's electronics is presented, based on this EHW architecture.

TABLE 6-3: EVALUATION OF PROPOSED EVOLUTIONARY TECHNIQUES

	Generic Gradual		Incremental		Incremental with MO	
Filter Order	Generations	CALUs	Generations	CALUs	Generations	CALUs
17	703	96	987	91	1123	87
26	880	119	1053	118	1186	114
32	683	107	672	89	778	84

Chapter 7
Implementation of the JPL/Boeing Gyroscope's Electronics

7.1 Introduction

Chapter 5 introduced the reconfigurable architectures that have been implemented during this research work. Based on the simulation results that have been achieved with each one of them independently, the most efficient reconfigurable hardware has been selected to implement the gyroscope's electronics. Therefore, the reconfigurable hardware initially introduced in Section 5.6 was selected and subsequently employed in Chapter 6 for evaluating the best reconfiguration mechanism. This is a novel reconfigurable fabric that meets all the criteria imposed by the targeted application. Briefly summarizing its characteristics, it can be said that its topology enables the efficient utilization of the hardware resources, resulting in significant reduction in power consumption, compared with industrial FPGAs.

Moreover, its structure minimizes deceptive problems, such as *pleiotropy* and *epistasis* that prevent the EA from reaching a global optimum within a short period of time. Finally, as shown in Section 5.6, it incorporates appropriate flexibility to cope with a significant amount of faults that occur either in the initial or later stage of the filters' implementation.

7.2 Evolution of the FIR Composing the Control Loops

This section presents the evolution of the seven FIR filters depicted in Figure 2.2 and presents the reconfigurable fabric that has been employed for the implementation of the PI controller. The first two filters form the drive control loop, whereas the third implements the sense-rebalance loop. The other 4 filters are part of the demodulation stage of the gyroscope's circuitry.

The EA that has been proposed in Section 6.4.5 was employed for the evolution of the filters. It utilizes a combination of incremental and MO evolution and it has been selected in these simulations because it provides the best results in terms of power consumption. The specification of the evolved filters has been taken from previous research work done by JPL and UCLA. The coefficient-sets of these filters cannot be shown in this thesis since this is forbidden by both JPL and UCLA. Therefore, the comparison of the ideal and the evolved filter will be performed by using the frequency response.

7.2.1 Evolution of Drive Control Loop

The drive loop consists of FIR1, which is used to correct the lag introduced by the signal conversion and the anti-aliasing filters and AGC. Consequently, the AGC consists of FIR2 (amplitude detector) and a PI controller.

FIR1 corresponds to a symmetrical 85-tap all-pass filter. Its purpose is to achieve attenuation at 2.7 KHz and shape the phase of the incoming signal to be 0 degrees at

4.428 KHz. According to the ideal coefficient-set the dynamic range of the coefficients varies from 2^7 to 2^{12}. Figure 7.1 shows the frequency response of the ideal (blue plot) and the evolved (green plot) filter. It is apparent that the two frequency responses are identical. Based on the simulation results, the evolution of FIR1 takes 3518 generations and occupies 316 CALUs.

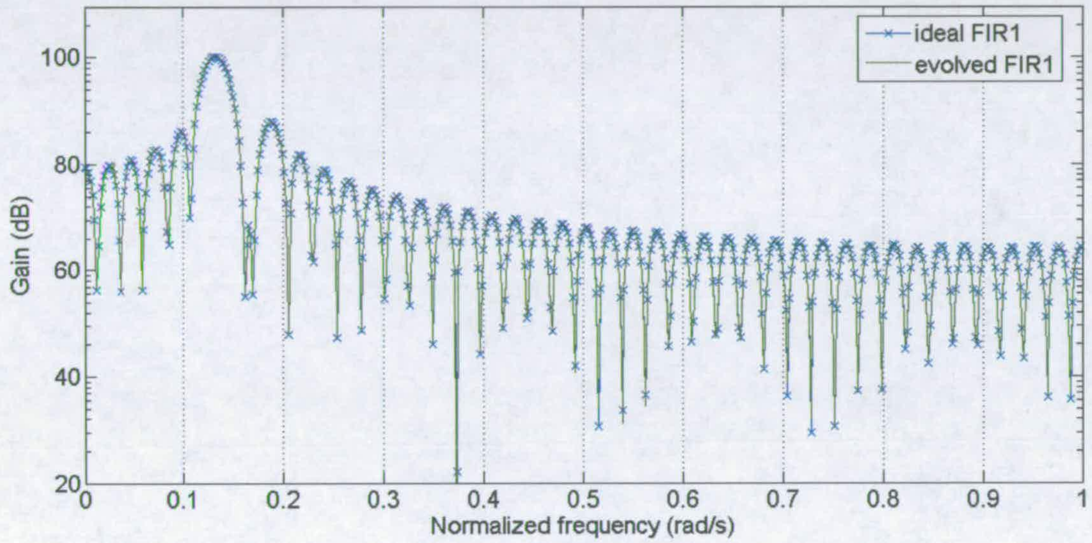


Figure 7.1: Comparison of the frequency response between the ideal and the evolved filter (FIR1)

FIR2 is a symmetrical 51-tap filter that composes part of the amplitude detector of the AGC. The ideal filter presents a cut-off frequency at 50 Hz and a stop-band attenuation of 40 dB. The dynamic range of the coefficient set varies from 2^9 to 2^{12}. Similarly with the previous comparison, Figure 7.2 depicts the accuracy of the evolution of the FIR2. Based on the obtained results the implementation of the filter requires 208 CALUs and needs 2324 generations.

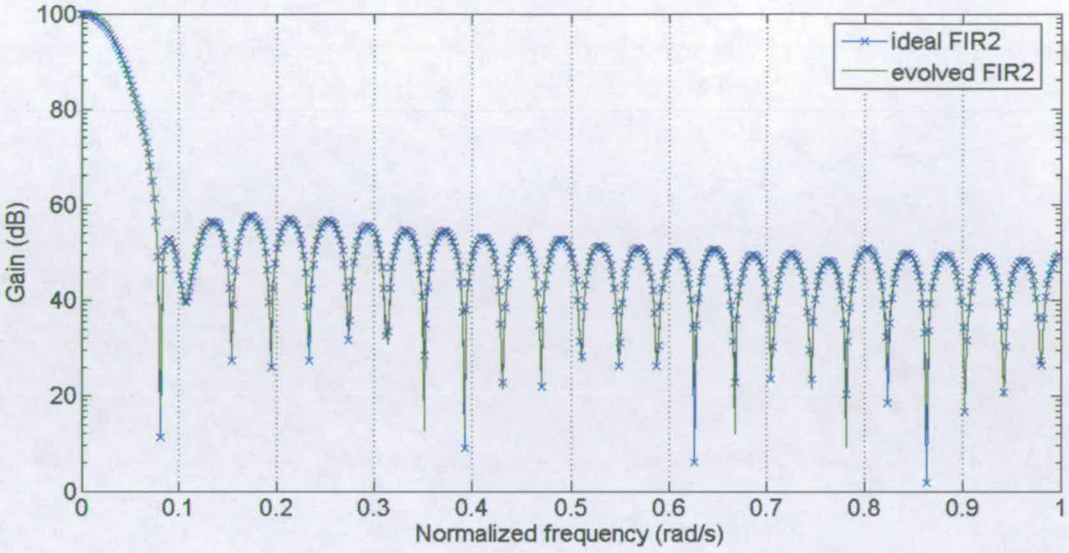


Figure 7.2: Comparison of the frequency response between the ideal and the evolved filter (FIR2)

The PI controller is part of the AGC that is used to excite the drive mode of the sensor to constant amplitude. Figure 7.3 depicts the overall block diagram of the drive control loop and the part within the shaded box contains the components of the PI controller. In addition to this, K_p and K_i modules compose the reconfigurable components of the controller and correspond to the proportional and integral coefficients, respectively. Each module presents an identical architecture, which is based on the POF technique as shown in Figure 7.4. The PI controller deals with decimal numbers with fractional accuracy. According to Figure 7.4 each module consists of 4 A/S units, which can select their inputs either from their previous A/S units or form the primary input of the controller. The first input (upper) of each A/S unit passes through a R/L shifter, which has 4 configuration bits. Hence, it can perform a binary shift from $3.9*10^{-3}$ to 128. The second input (lower) of each A/S unit passes through a shifter that includes 1 configuration bit and can perform left shift by 0 or 1 positions.

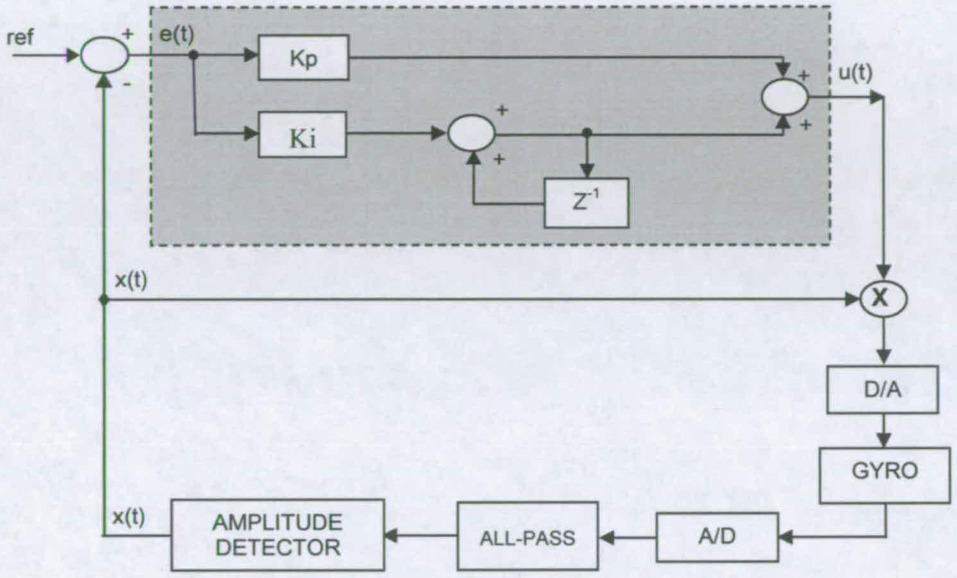


Figure 7.3: Block diagram of the drive control loop

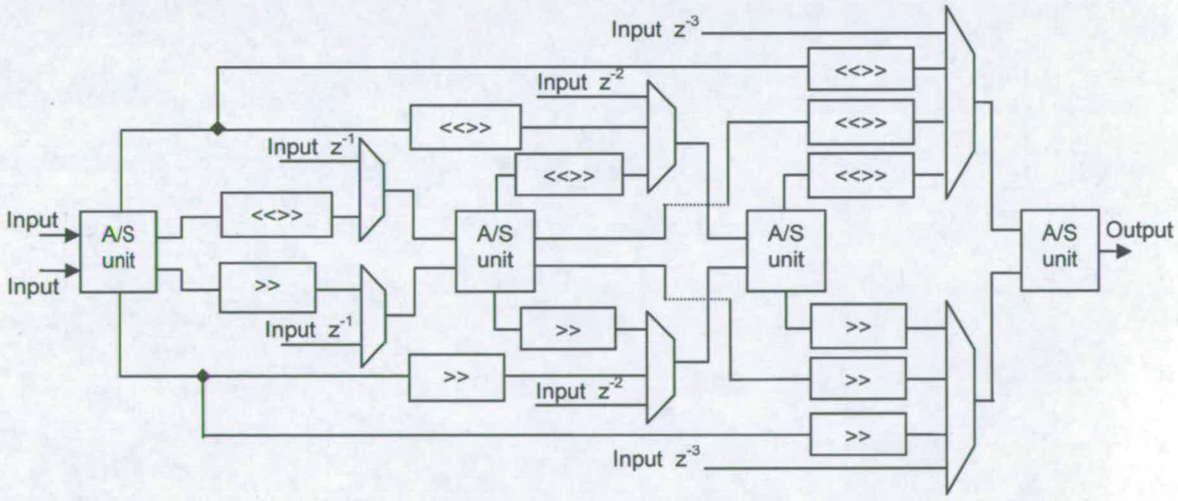


Figure 7.4: Reconfigurable hardware for Kp and Ki modules

The K_p and K_i coefficients of the controller must be adjusted appropriately in order that the PI controller provides the correct feedback compensation. Hence, the K_p and K_i modules must be able to evolve decimal numbers with fractional accuracy of up to 5 digits. To demonstrate the capability of the proposed reconfigurable hardware platform depicted in Figure 7.4, several decimal numbers (0.7629, 2.4414, 6.1035 and 12.20703) have been evolved, which can be potential K_p and K_i coefficients. For each one of these numbers, different simulation scenarios have been obtained, where the crossover rate is kept constant (80%) and the mutation rate dynamically varies from 0-10%, 0-15%, 0-20% and 0-25%, respectively. Simulation results depicted in Figure 7.5 reveal that the proposed PI controller is capable of evolving the assigned potential coefficients with great accuracy.

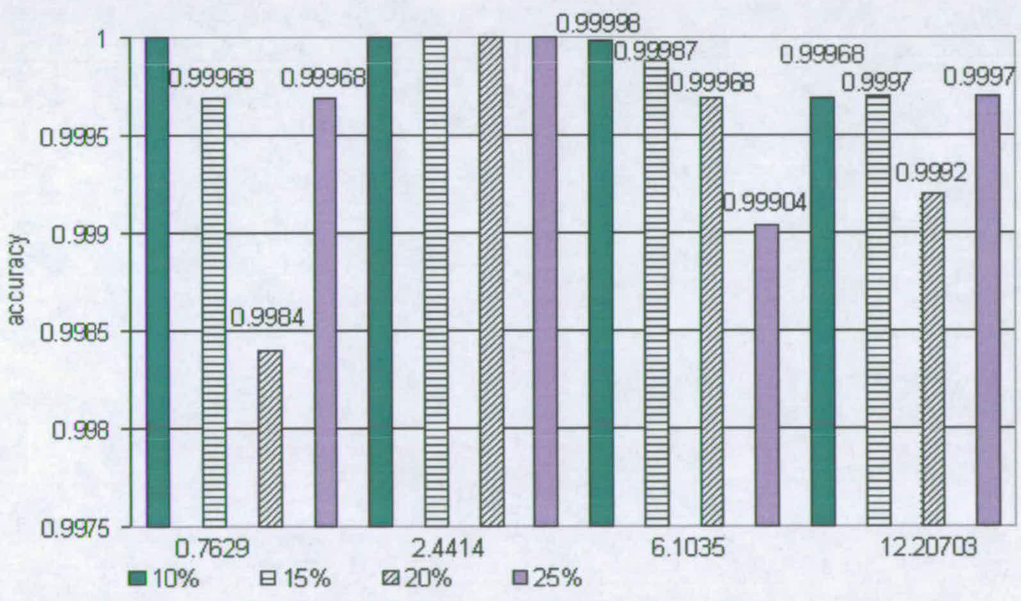


Figure 7.5: Simulation results of PI controller for 4 dynamically varying mutation rates (10%, 15%, 20% and 25%)

Specifically, there is an average deviation of $3*10^{-4}$ % between the assigned numbers and the evolved ones. Finally, it can be deduced that for dynamic mutation rate of 10% the controller provides the most accurate solutions (average deviation $8*10^{-5}$).

7.2.2 Evolution of the Sense-Rebalance Loop

The sense-rebalance loop is implemented in FIR3. It composes a symmetrical linear-phase 87-tap filter. The aim of this filter is to reject the disturbance that is injected into the sense drive due to the Coriolis Effect and reject the disturbance at the region of 2700 Hz, which can produce degradation of the sensor's performance. In this filter, the dynamic range of the ideal coefficient-set varies from 2^8 to 2^{12}. Figure 7.6 clearly shows that there is not any visible difference between the ideal and the evolved frequency response. For the evolution of this filter the employed EA takes 2165 generations and occupies 181 CALUs.

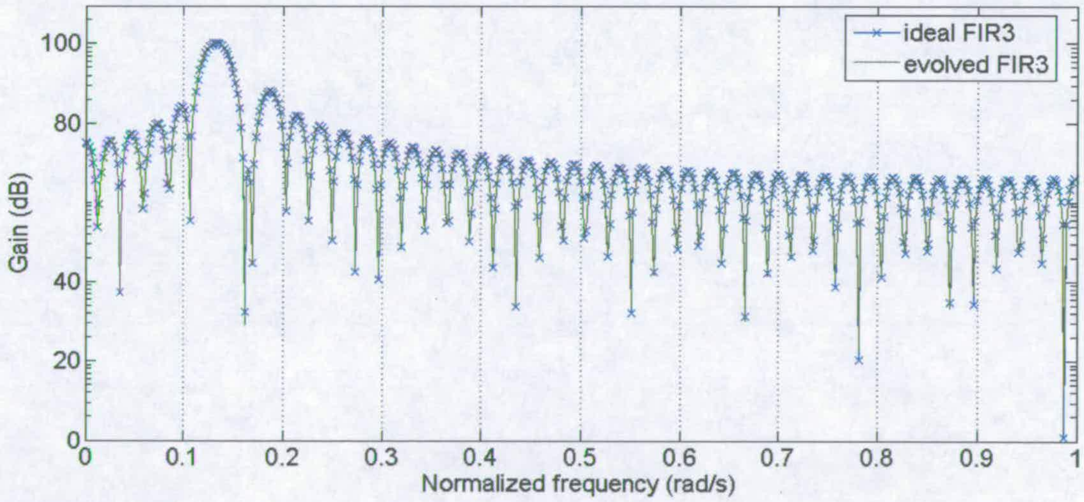


Figure 7.6: Comparison of the frequency response between the ideal and the evolved filter (FIR3)

7.2.3 Evolution of the Demodulation stage

The demodulation stage estimates the angular rotation rate of the gyroscope by demodulating the sense-rebalance signal with respect to a measurement of the drive loop control. This electronic circuit consists of FIR4 and FIR5, which are used to shift signal phases and FIR6 and FIR7, which are low-pass filters.

FIR4 is a symmetrical 20-tap filter and its ideal coefficient-set presents a dynamic range that varies from 2^8 to 2^{16}. The evolution of this filter can be seen in Figure 7.7. The evolved frequency response is depicted with the red dashed plot and it is obvious that its trace is exactly upon the trace formed by the ideal frequency response (blue line). Furthermore, the evolution of FIR4 occupies 87 CALUs and takes 1290 generations.

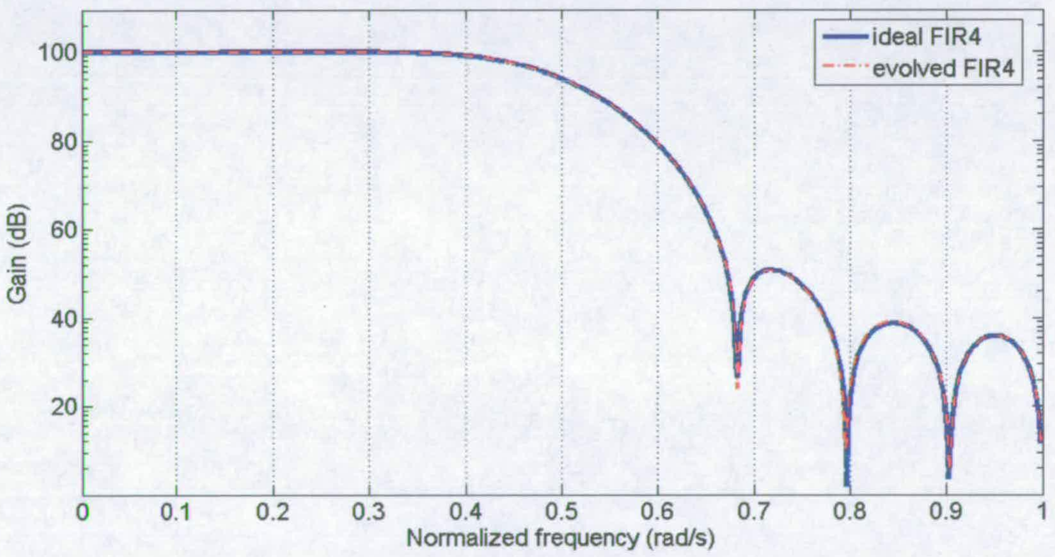


Figure 7.7: Comparison of the frequency response between the ideal and the evolved filter (FIR4)

Table 7-1 shows the average number of CALUs and generations that is needed for the implementation of each one of the seven filters. For the realization of FIR4, it can be seen that there is a small increase in both figures (number of CALUs and generations) compared with the implementation of the previous three filters. This can be justified considering the dynamic range of the targeted coefficient-set. Hence, the wider dynamic range that the coefficient–set of FIR4 presents, compared with FIR1, FIR2 and FIR3, introduces a substantial increment in the average number of CALUs and generations.

FIR5 is a symmetrical 29-tap filter and similarly with FIR4 is employed to shift signal phases. Its coefficient-set has a dynamic range from 2^1 to 2^{17}. Although its wide dynamic range, this coefficient-set consists of some coefficients that are equal to zero and this fact facilitates the evolution of the filter.

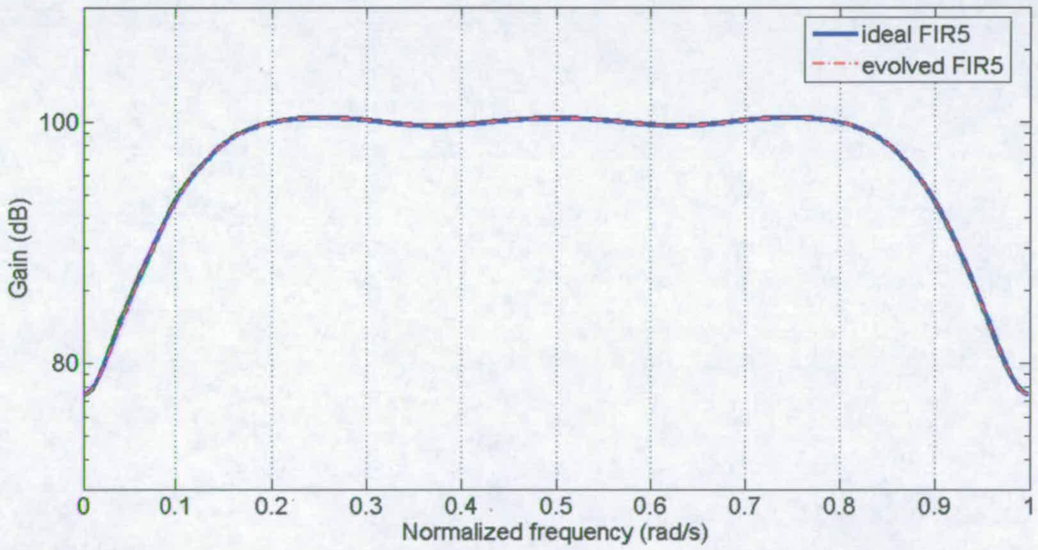


Figure 7.8: Comparison of the frequency response between the ideal and the evolved filter (FIR5)

Hence, this fact has a beneficial impact on the number of utilized CALUs (87) and especially on the number of generations (868). Figure 7.8 shows the comparison in the frequency response between the ideal and the evolved filter.

FIR6 is a symmetrical 15-tap filter and its coefficients introduce a dynamic range that varies from 2^1 to 2^{17}. Figure 7.9 depicts the frequency response of the ideal and the evolved low-pass filter and proves the high accuracy that the proposed evolutionary driven reconfigurable platform can secure. Concerning the hardware utilization, FIR6 occupies 61 CALUs and its implementation takes 600 generations.

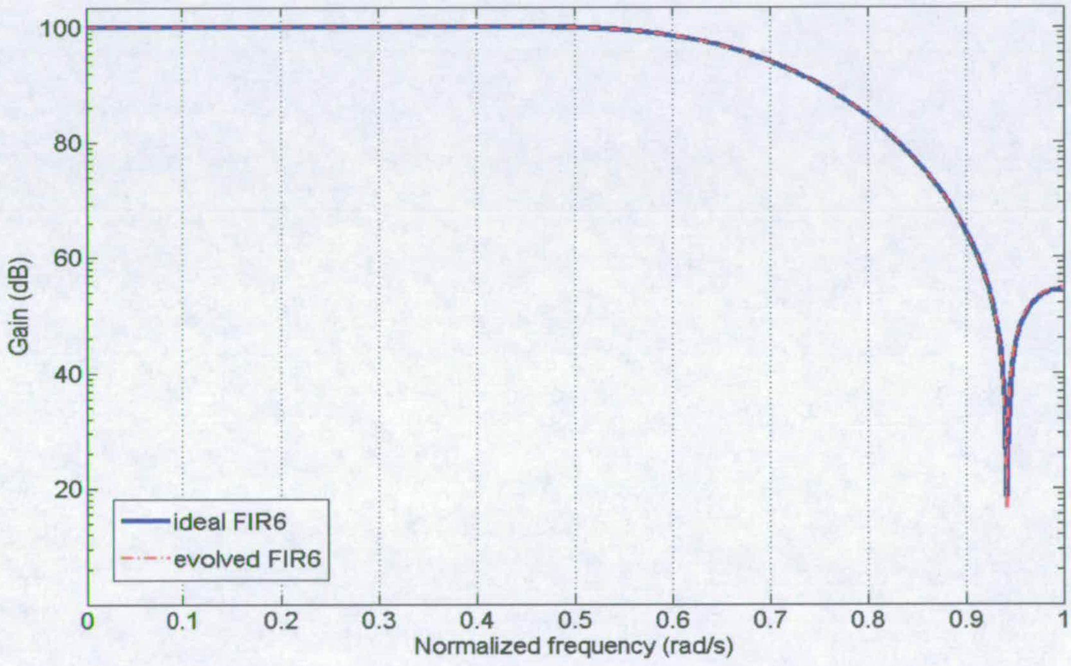


Figure 7.9: Comparison of the frequency response between the ideal and the evolved filter (FIR6)

Finally, FIR7 is a symmetrical 32-tap low-pass filter. The dynamic range of the filter's coefficients varies from 2^8 to 2^{14}. The filter needs 138 CALUs to be implemented and its evolution lasts 664 generations. Figure 7.10 proves even in this implementation the high accuracy that can be achieved.

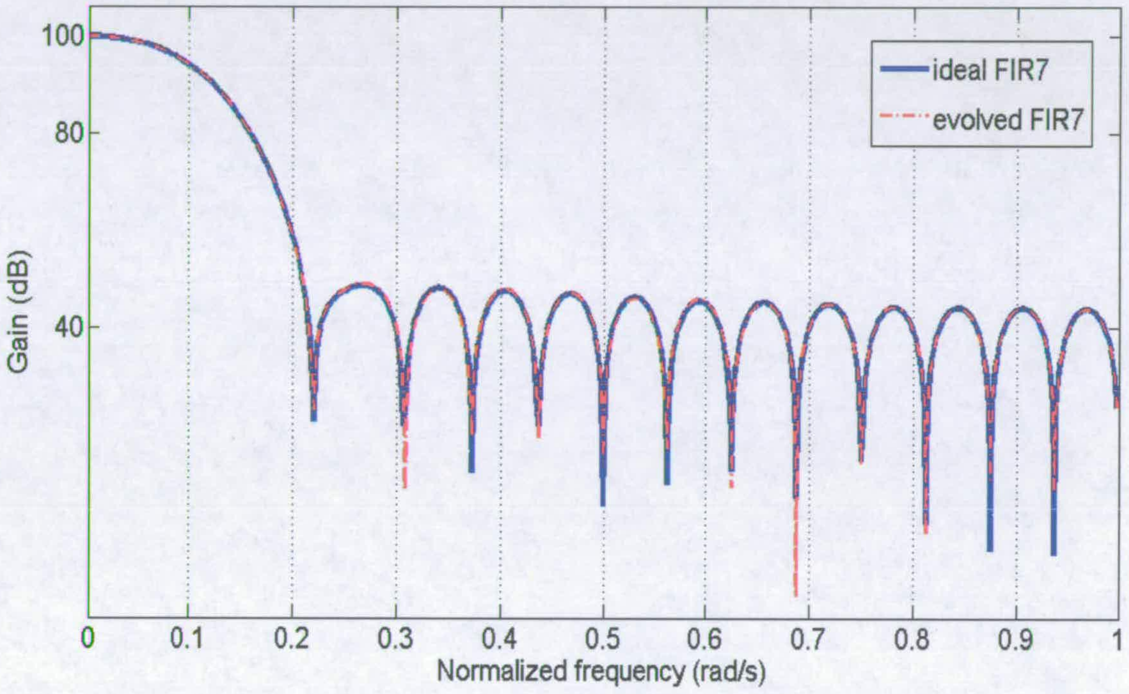


Figure 7.10: Comparison of the frequency response between the ideal and the evolved filter (FIR7)

TABLE 7-1: CHARACTERISTICS OF FILTERS' SPECIFICATION AND SIMULATION RESULTS CONCERNING HARDWARE UTILIZATION AND CONVERGENCE TIME

Sensor's Electronics	Order	Dynamic range of coefficient-set	Average number of CALUs per coefficient	Average number of generations per coefficient
FIR 1	85	2^7 - 2^{12}	3.71	41.38
FIR 2	51	2^9 - 2^{12}	4.07	45.56
FIR 3	46	2^8 - 2^{12}	3.93	47.06
FIR 4	20	2^8 - 2^{16}	4.35	64.5
FIR 5	29	2^1 - 2^{17}	3	29.93
FIR 6	15	2^7 - 2^{17}	4.06	40
FIR 7	32	2^8 - 2^{14}	4.31	20.75

7.3 Power Analysis

After the evolution of the circuits that compose the control-loops of the gyroscope, it is also important to measure the power consumption and see whether the achieved result suits an aerospace application.

The power analysis of the evolved FIR filters has been carried out using Synopsys Prime-Power tools. Accurate interconnect analysis has been considered by setting appropriately appropriate wire load models during the synthesis of the two reconfigurable structures that have been used for the realization of the filters and the PI controller.

Figure 7.11 shows the power that is consumed by each FIR filter separately and the PI controller, respectively. Moreover, it can be seen which part of the total power corresponds to the dynamic (internal plus switching) and which to the leakage. Further details for the exact figures can be found in Table 7-2. It must be mentioned here that the obtained figures correspond to mW per MHz of operation. It is known that the update rate of the gyroscope is 48 KHz, therefore the frequency, at which the power measurement has been done, meets the throughput criteria set by the targeted application.

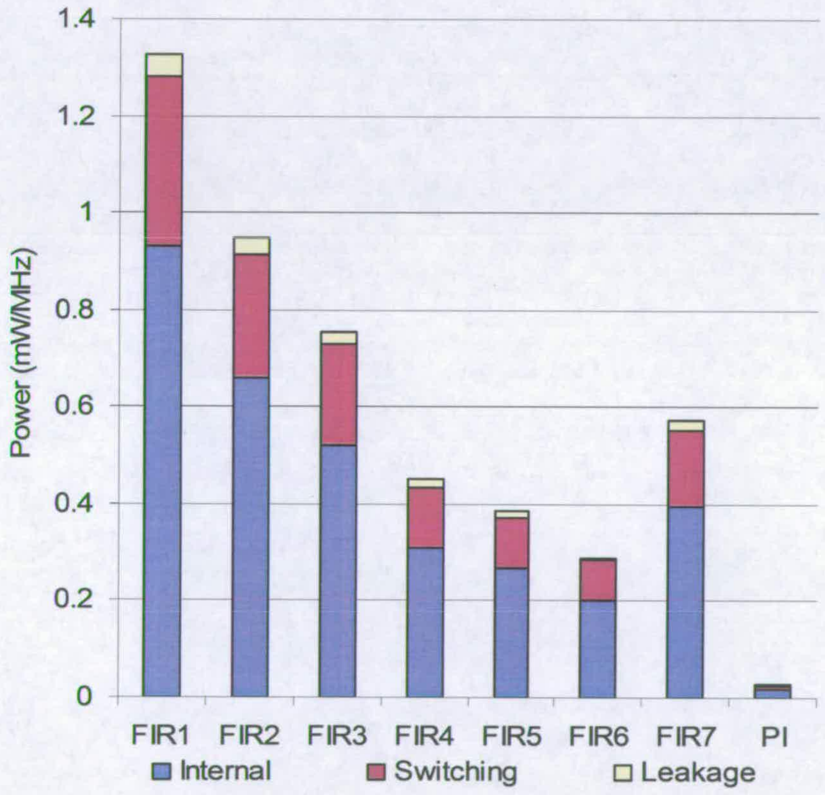


Figure 7.11: Power analysis of the 7 FIR filters and the PI controller that compose the drive and sense-rebalance loop and the demodulation stage of the gyroscope

Finally, Figure 7.12 illustrates the power that is consumed by each control-loop. The results show that the proposed architecture consumes in total 4.745 mW/MHz and that the drive control loop consumes the 48.47% (2.3 mW/MHz) of the total power, while the sense-rebalance loop and the demodulation stage consume the 15.91% (0.755 mW/MHz) and 35.61% (1.69 mW/MHz), respectively.

TABLE 7-2: POWER RESULTS FOR THE FIR FILTERS AND THE PI CONTROLLER

Power (mw/MHz)	FIR 1	FIR 2	FIR 3	FIR 4	FIR 5	FIR 6	FIR 7	PI
Internal	0.932	0.66	0.52	0.31	0.266	0.2	0.396	1.77E-2
Switching	0.349	0.255	0.21	0.125	0.106	0.086	0.156	7.43E-3
Leakage	0.045	0.032	0.025	0.015	0.013	0.001	0.02	1.86E-3
Total	1.326	0.947	0.755	0.45	0.385	0.287	0.572	2.70E-2

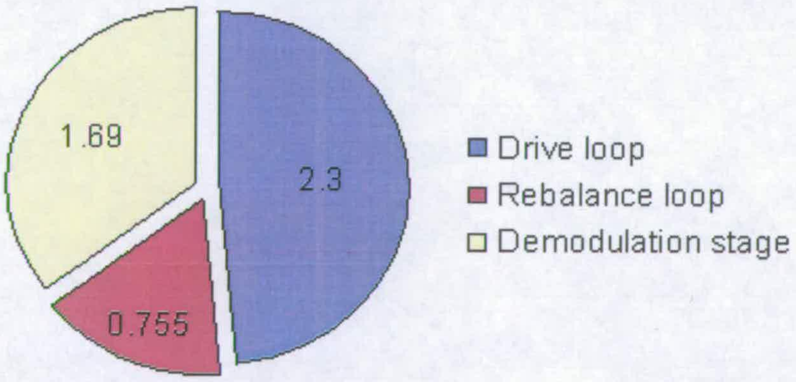


Figure 7.12: Power analysis (mW/MHz) of the different control loops of the gyroscope

7.4 Conclusions

This section analyzed the implementation of the JPL/Boeing gyroscope electronics, which comprise the seven FIR filters and a PI controller. The simulation results showed that the employed EHW platform is able to accurately implement the electronics. Moreover, the power that is consumed by the evolved electronics (4.745 mW/MHz) shows that the presented architecture is very promising for Space application where long operational life of electronics is required.

Future work has to be done to complete the on-chip implementation of the holistic system that comprises a reconfigurable hardware substrate and a reconfiguration mechanism that is in charge of guiding the implementation/reconfiguration of the targeted electronics. Figure 7.13 shows a preliminary floorplan of the holistic system. The reconfigurable hardware is a custom reconfigurable fabric that based on the simulation results must consist of 1400 CALUs. This implies that this fabric would occupy approximately 10.36 mm^2 of silicon area, since in Section 5.6.5 it was calculated that 200 CALUs occupy area of 1.48mm^2. Moreover, the selected reconfiguration mechanism (GA) can run on a micro-processor that meets the time requirements of the sensor. In addition to this, based on the implementation of the sensor's electronics, the requirements of memory size have been calculated. It is assumed that there is a 30% overhead in the number of CALUs needed for the implementation of the electronics and that the size of population that is used in the incorporated GA is 100. The additional number of CALUs has been practically selected considering the spare resources that are needed for the reconfiguration of a single coefficient when a fault occurs. Moreover, it provides more flexibility in case of a change in the filters' specification. Therefore, considering the previous assumptions, the size of the memory that is required for the holistic system is 106Kbytes. This is a demonstrative figure showing that the employed GA does not introduce significant overhead in terms of memory requirements.

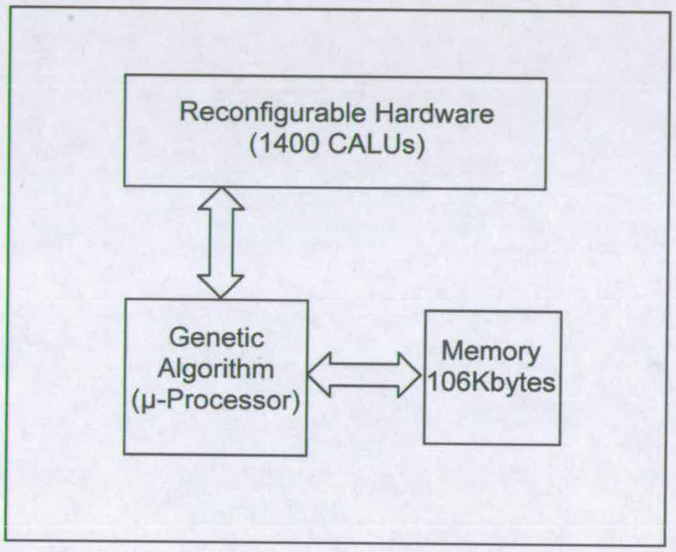


Figure 7.13: On-chip solution of the holistic system

Chapter 8
Summary and Conclusions

8.1 Introduction

The focus of this thesis has been to investigate which is the most suitable autonomously reconfigurable platform that would be able to conformably accommodate the electronics associated with the JPL/Boeing gyroscope. Particular emphasis has been given on low-power, fault-robust, self-reprogrammable/reconfigurable architectures. Therefore, all the architectures investigated in this thesis compose a conjunction of custom reconfigurable hardware and specially tailored GAs, which derive from a class of non-heuristic search and optimization techniques termed EAs and are inspired by the process of biological evolution.

This chapter will be organized as follows: Section 8.2 gives a summary of the material presented in each chapter of this thesis. Subsequently, Section 8.3 provides conclusions drawn from the collected results and the ascertainments that are implied. Finally, Section 8.4 outlines future work, which would add to the knowledge already gained from research undertaken in this thesis and discusses what might be done to improve the performance of the obtained EHW platforms.

8.2 Summary

The underlying theme of this thesis has been to investigate various autonomously reconfigurable architectures for the automated design and/or adaptation of the electronics associated with the control-loops of the JPL/Boeing gyroscope. The principle of EHW, which lies at the intersection of reconfigurable physical design and evolutionary computation, has been employed because it has been shown to provide unique characteristics such as automated design, autonomous adaptation to endogenous and/or endogenous variations (alternation of specification standards, anticipated faults, exposure to radiation, temperature variation, etc.) that affect the system's behavior, exploring and continuous seeking better solutions to complicated problems. Therefore, specially tailored evolutionary techniques have been implemented, which are in charge of successfully searching for and manipulating the encoding used to configure the investigated reconfigurable hardware substrates in order to generate the desired system's behavior.

Chapter 2 introduces the JPL/Boeing gyroscope and provides an overview of its structure and operation principles. Moreover, it analyses the electronic circuits that implement the control-loops of the gyroscope and presents previous work that has been done concerning the gyroscope itself and possible architectures that could be used to efficiently accomplish the sensor's electronics.

Chapter 3 justifies the selection of EHW to be the most suitable solution for the targeted application. Furthermore, it introduces the concepts behind EHW and provides a theoretical background for the electronic circuits that compose the gyroscope's control-loops. It also demonstrates the benefits and limitations behind different levels (transistor, gate and functional) and modes (extrinsic and intrinsic) of evolution for circuit design, through literature review.

Chapter 4 introduces the design of a stand-alone fault-tolerant architecture. This concept demonstrates that both the reconfigurable hardware and the reconfiguration mechanism are able to compensate for faults. This chapter presents the concept and hardware implementation of a parallel fine-grained GA that employs the inherent parallelism in order to cope with faults. The capability of this architecture is demonstrated through a practical application of an attitude determination algorithm but it can be extend to further applications, including the reconfiguration mechanism of the JPL/Boeing gyroscope.

Chapter 5 presents the implementation of numerous reconfigurable architectures and evaluates their ability to implement high-order FIR filters, which compose the major electronic components of the gyroscope's electronics. The demonstration of these architectures has been arranged according to temporal order. Therefore, descendant architectures attempt to satisfy the system's requirements such as accuracy in high-order filters, low power consumption and real-time adaptation, and address drawbacks that ancestor architectures encounter.

Chapter 6 selects the most powerful reconfigurable architecture presented in Chapter 5 and employs it in order to evaluate several evolutionary techniques by evolving numerous FIR filters. The proposed evolutionary techniques present superiority over conventional ones in terms of accuracy and ability to evolve higher-order filters. Moreover, two out of the three proposed techniques are focused on different objectives, such as speed for adaptation and efficient hardware utilization. Finally, the third one presents a trade-off between the previous mentioned objectives and provides intermediate solutions in both objectives.

Chapter 7 presents the implementation of the gyroscope's electronics as these have been designed by Jet Propulsion Laboratory and UCLA University. For the evolution of the seven FIR filters and the PI controller, the most powerful reconfigurable hardware substrate is employed in conjunction with the proposed evolutionary technique presented in Chapter 6 that targets minimum power consumption. Finally, the total power consumption is calculated for the control-loops of the gyroscope using synthesizable hardware netlists and accurate technology libraries.

8.3 Conclusions

This thesis suggests that an EHW architecture is the most suitable environment to accommodate the electronics of the JPL/Boeing gyroscope and meet all the characteristics that the targeted application implies.

From the literature review in Chapters 2 and 3 it can be concluded that functional level evolution can be effectively utilized for the realization of complicated electronic circuits, such as high-order FIR filters. Specifically, primitive operators, such as additions, subtractions and binary shifts can be the fundamental functional blocks, which in conjunction with synchronous delays can be appropriately combined and form high-order FIR filters. This approach induces beneficial results both in terms of power consumption and fault-tolerance. The substitute of conventional multipliers with efficient combinations of primitive operators reduces significantly the electronic components needed and this has a significant effect in the reduction of power consumption. Moreover, this approach reduces the granularity of the reconfigurable hardware and hence in the presence of faults the fine-grained granularity can take advantage of hardware redundancy.

From the system analysis in Chapter 4, it can be concluded that fine-grained parallel GAs are suited for the implementation of fault-robust evolutionary mechanisms within a single chip. Simulation results have shown that due to massive parallelism and the incorporation of efficient communication between the adjacent processing elements, faults that occur on the hardware resources do not disseminate through the hardware framework.

Chapter 5 presents five different reconfigurable hardware substrates that employ different configurable arithmetic/logic units and in particular alternative interconnections between these units. From the unsuccessful experiments to evolve FIR filters with the reconfigurable hardware presented in Section 5.2, it can be concluded that the conception of the physical design must be effectively done considering two deceptive problems, such as pleiotropy and epistasis that prevent the evolutionary mechanism from reaching a global optimum within a valid period of time. Initial simulation results in Section 5.3 support the conclusion that evolution of FIR filters in the frequency domain may result in alternative hardware designs that require fewer generations than evolution of digital filters in the time domain. However, evolution in the frequency domain does not guarantee the implementation of linear phase filters that is one of the prerequisites. Moreover, results in Section 5.4 enhance the aspect that evolution of medium-order FIR filters in the frequency domain can be successful for offline and/or online adaptation even in the presence of anticipated faults occurring in the memory elements of the reconfigurable hardware. Subsequently, Section 5.5 presents a reconfigurable hardware architecture that is specially tailored for the implementation of folded-transposed form FIR filters. From the simulation results, important conclusions can be made concerning the re-usability of previously obtained partial products and the evolution of both the timing and arithmetic part of the filter. The approach, that takes advantage of previously evolved partial products in order to evolve new coefficients, may be beneficial in terms of hardware utilization with power consumption implications. However, during the evolution of medium or high order FIR filters, this approach produces hardware dependencies between different coefficients within the under evolution coefficient-set. In case these dependencies are contradictive, then the accurate evolution of the evolved coefficients is not feasible or the evolution may take substantially longer than the application's throughput requirements. Moreover, although this approach provides a higher degree of fault-robustness, it is very likely that evolution may stop in sub-optimal solutions before the search space becomes very complicated. Section 5.6 presents the most successful reconfigurable hardware substrate based on simulation results. It meets all the requirements applied by the targeted application, such as accurate evolution of high-order FIR filters, low-power consumption, appropriate throughput that concurs with the speed specification of the gyroscope, high degree of fault-robustness in the presence of anticipated faults occurring in the user and configuration memory and real-time adaptation and recovery of the original system's behavior due to exogenous and/or endogenous factors. Its great success is based on a couple of architectural approaches relative to the interconnection scheme between the CALUs. According to this, the hardware dependencies between different coefficients have been eliminated resulting in a very fast and more accurate filter evolution. Moreover, the same hardware resources (CALUs) are employed both in the arithmetic and accumulative unit of the FIR filter.

The architectural approaches incorporated in Section 5.6 result in a significant reduction in power consumption compared with conventional approaches. Summarizing the obtained results, this architecture in conjunction with the three proposed evolutionary strategies presented in Chapter 5 manages significant achievements, which can be summarized below:

- Evolution of FIR filters that exceeds 99.7% of accuracy.
- Implementation and/or adaptation of FIR filters' even in the presence of 30% faults, occurring in the user and configuration memory of the reconfigurable hardware.

- Reduction of between 8.6 to 9.8 times in the number of generations that are needed for the evolution of a 31-tap FIR filter, compared with previous research in this field.

- Evolution of FIR filters that consume 3.3 times less power than similar implementations within industrial reconfigurable devices.

It can be concluded from Chapter 6 that novel evolutionary reconfiguration mechanisms can be employed in order to accurately evolve FIR filters of any order that introduce a substantially large search space. This is feasible by utilizing a uniform reconfigurable hardware substrate and performing gradual evolution of the coefficient-set. This approach instantaneously minimizes the total search space to smaller parts that effectively affecting the realization of the coefficient under evolution. Finally, the synergy of these evolutionary mechanisms with multi-objective optimization can be very beneficial for the implementation of very low-power FIR filters.

8.4 Future Work

This thesis has endeavored to provide a rigorous investigation of the focus of research outlined in Section 8.1. However, a number of additional potentially interesting areas remain, which might further add to the knowledge already gained from the research presented.

A specially tailored parallel GA can be implemented to identify the configuration string of the reconfigurable hardware, which accommodates the JPL/Boeing gyroscope's electronics. It would be very interesting to have a comparison between a fine-grained parallel GA, like the one in Chapter 4, and the customized evolutionary algorithms that have been implemented and evaluated in this thesis.

The implementation of a reconfigurable hardware substrate that originates from the one presented in Section 5.5 and overcomes the limitations of the prior architecture. The new implementation has to incorporate two architectural approaches. Firstly, the timing will be fixed and hence, evolution will target only the realization of the correct values of the ideal coefficient-set. Secondly, re-usability of previously evolved coefficients will be allowed only in the distance of one. This means that for the realization of a new coefficient, the evolutionary mechanism can use partial products that have been obtained only during the realization of the previous coefficient.

Another interesting modification that can be done in future relates the reconfigurable hardware in Section 5.6. Assuming that the targeted application requires very fast clock frequency and is not interested in fault-robustness, then in each CALU an additional synchronous register can be added to the output and after the binary shifter. This modification will increase the speed performance substantially without affecting the time needed for the filter implementation and/or adaptation.

The fabrication of a chip can also be scheduled in the future. This chip may include only the reconfigurable hardware substrate that accommodates the gyroscope's electronics or it can be a holistic System-on-Chip solution that includes both the reconfigurable hardware and the reconfiguration mechanism.

This thesis has investigated numerous autonomously reconfigurable platforms for the implementation of mainly high-order FIR filters. Low-power, high performance and reliable/robust digital adaptive FIR filters are in great demand throughout the communication, automotive, audio, biomedical industry that requires data control and manipulation. Industrial requirements might set priority in different characteristics, such as device re-usability, fast operational speed, low-power consumption, fault-robustness, etc. Therefore, several minor modifications can be done in the EHW platforms presented in this thesis, in order to accurately satisfy requirements of several other industrial applications besides the JPL/Boeing gyroscope.

Bibliography

[1]. Actel's 2nd Generation Reprogrammable Flash FPGAs "ProASIC^{PLUS}", http://www.actel.com/products/proasicplus/. Copyright© 1985-2006 by Actel Corporation, Mountain View, CA. 94043-4655, USA. All rights reserved.

[2]. Alba, E., Dorronsoro, B., "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms", *Evolutionary Computation, IEEE Transactions on*, vol.9, no.2pp. 126- 142, April 2005.

[3]. Quartus II Software Design Features, http://www.altera.com/products/software/products/quartus2/design/qts-design_flow_pld.html#advantage. Copyright© 1995-2006, Altera Corporation, 101Innovation Drive, San Jose, California, 95134, USA.

[4]. Astrom, K., Hagglund, T., "PID Controllers: Theory, Design, and Tuning", *2nd Edition. The Instrumentation, Systems, and Automation Society,* 1998.

[5]. Atmel's AT40K and AT40KAL Series co-processor FPGAs, http://www.atmel.com/products/FPGA/. Atmel Corporation © 2006.

[6]. Aporntewan, C., Chongstitvatana, P., "A hardware implementation of the compact genetic algorithm", *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on,* vol.1, no.pp.624-629 vol. 1, 2001.

[7]. Arslan, T., Eskikurt, H.I., Horrocks, D.H., "Configurable structures for a primitive operator digital filter FPGA", *Signal Processing Systems, 1997. SIPS 97 - Design and Implementation., 1997 IEEE Workshop on,* vol., no.pp.532-540, 3-5 Nov 1997.

[8]. Atherton, B., Diddams, S., Diels, J.-C., "Stabilization of a mode-locked ring laser gyroscope", *Lasers and Electro-Optics, 1996. CLEO '96., Summaries of papers presented at the Conference on* , vol., no.pp. 201- 202, 2-7 June 1996.

[9]. AT6005 – Application Notes, www.atmel.com/dyn/products/app_notes.asp?family_id=623&part_id=2073. Atmel Corporation © 2006.

[10]. Avizienis, A., "Signed-digit number representation for fast parallel arithmetic", *IRE Transactions on Electronic Computers EC-10,* 389-400.

[11]. Bae, S.Y., Hayworth, K.J., Yee, K.Y., Shcheglov, K., Wiberg, D.V., "High performance MEMS micro-gyroscope", *SPIE-Int. Soc. Opt. Eng. Proceedings of SPIE – the International Society for Optical Engineering*, vol. 4755, pp. 316-324, USA, 2002.

[12]. Bennett III F. H., Koza, J. R., Andre, D., Keane, M. A., "Evolution of a 60 decibel Op Amp using genetic programming", Proceedings of the First International Conference on Evolvable Systems (ICES96), LNCS 1259, pp. 455-469, Tsukuba, Japan, October, 1996.

[13]. Bernstein, J., Cho, S., King, A.T., Kourepenis, A., Maciel, P., Weinberg, M., "A micromachined comb-drive tuning fork rate gyroscope", *Micro Electro Mechanical Systems, 1993, MEMS '93, Proceedings 'An Investigation of Micro Structures, Sensors, Actuators, Machines and Systems'. IEEE.*, vol., no.pp.143-148, 7-10 Feb 1993.

[14]. Bernstein, J., "An overview of MEMS inertial sensing technology", *Sensors (Peterborough, Nh), Publisher: Advanstar Communications, USA,* vol. 20, no. 2, pp. 14-21, Feb. 2003.

[15]. Bi, G., Jones, E.V., "A pipelined FFT processor for word-sequential data", *Acoustics, Speech, and Signal Processing, IEEE Transactions on*, vol.37, no.12pp.1982-1985, Dec 1989.

[16]. Bruce, H., Veljanovski, R., Owall, V.; Singh, J., "Power optimization of a reconfigurable FIR-filter", *Signal Processing Systems, 2004. SIPS 2004. IEEE Workshop on*, vol., no.pp. 321- 324, 13-15 Oct. 2004.

[17]. Bull, D.R., Horrocks, D.H., "Primitive operator digital filters", *Circuits, Devices and Systems, IEE Proceedings G*, vol.138, no.3pp.401-412, Jun 1991.

[18]. Cantu-Paz, E., "A survey of parallel genetic algorithms", Calculateurs Paralleles, Reseaux et Systems Repartis. vol. 10. no. 2, pp. 141-171, Paris: Hermes, 1998.

[19]. Cappello, P., Steiglitz, K., "Some complexity issues in digital signal processing", *Acoustics, Speech, and Signal Processing, IEEE Transactions on*, vol.32, no.5pp. 1037-1041, Oct 1984.

[20]. Yen-Cheng Chen, M'Closkey, R.T.; Tran, T.A.; Blaes, B., "A control and signal processing integrated circuit for the JPL-boeing micromachined gyroscopes", *Control Systems Technology, IEEE Transactions on*, vol.13, no.2pp. 286- 300, March 2005.

[21].Yun-Ho Choi, Duck jin Chung, "VLSI processor of parallel genetic algorithm", *ASICs, 2000. AP-ASIC 2000. Proceedings of the Second IEEE Asia Pacific Conference on*, vol., no.pp.143-146, 2000.

[22]. Dempster, A.G., Macleod, M.D., "Constant integer multiplication using minimum adders", *Circuits, Devices and Systems, IEE Proceedings,* vol. 141, no. 5, pp. 407-413, Oct 1994.

[23]. Dempster, A.G., Macleod, M.D., "Use of minimum-adder multiplier blocks in FIR digital filters", *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on,* vol. 42, no. 9, pp. 569-577, Sep 1995.

[24]. Dempster, A.G., Dimirsoy, S.S., Kale, I., "Designing multiplier blocks with low logic depth", *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol.5, no.pp. V-773- V-776 vol.5, 2002.

[25]. Dempster, A.G., Macleod, M.D., "Using all signed-digit representations to design single integer multipliers using sub-expression elimination", *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, vol.3, no.pp. III- 165-8 Vol.3, 23-26 May 2004.

[26]. Dufour, C., Garnier, P., Carriere, T., Beaucour, J., Ecoffet, R., Labrunee, M., "Heavy ion induced single hard errors on submicronic memories [for space application]", *Nuclear Science, IEEE Transactions on*, vol.39, no.6pp.1693-1697, Dec 1992.

[27]. Erdogan, A.T., Hasan, M., Arslan, T., "Algorithmic low power FIR cores", *Circuits, Devices and Systems, IEE Proceedings [see also IEE Proceedings G- Circuits, Devices and Systems]*, vol.150, no.3pp. 155- 160, 6 June 2003.

[28]. Evans, J.B., "An efficient FIR filter architecture", *Circuits and Systems, 1993., ISCAS '93, 1993 IEEE International Symposium on* , vol., no.pp.627-630, 3-6 May 1993.

[29]. Ferguson, M.I., Zebulum, R., Keymeulen, D., Stoica, A., "An evolvable hardware platform based on DSP and FPTA", Late breaking papers at the *Genetic and Evolutionary Computation Conference* (GECCO-2002).

[30]. Ferguson, M.I., Keymeulen, D., Peay, C., Yee, K., Li, D.L., "Effect of temperature on MEMS vibratory rate gyroscope", *Aerospace, 2005 IEEE Conference*, vol., no.pp. 1- 6, 5-12 March 2005.

[31]. Frigo, N.J., "A comparison of the radiation vulnerabilities of ring resonator and interferometric fiber optic gyroscopes", *Lightwave Technology, Journal of*, vol.7, no.12pp.2009-2012, Dec 1989.

[32]. Goldberg, D.E., "Genetic algorithms in search optimization & machine learning", Addison-Wesley, 1989.

[33]. Grayver, E., M'Closkey, R.T., "Automatic gain control ASIC for MEMS gyro applications", *American Control Conference, 2001. Proceedings of the 2001*, vol.2, no.pp.1219-1222 vol.2, 2001.

[34]. Gustafsson, O., Dempster, A.G., Wanhammar, L., "Extended results for minimum-adder constant integer multipliers", *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol.1, no.pp. I-73- I-76 vol.1, 2002.

[35]. Hartley, R.I., "Sub-expression sharing in filters using canonic signed digit multipliers", *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on [see also Circuits and Systems II: Express Briefs, IEEE Transactions on]*, vol.43, no.10pp.677-688, Oct 1996.

[36]. Hawley, R.A., Wong, B.C., Thu-Ji Lin, Laskowski, J., Samueli, H., "Design techniques for silicon compiler implementations of high-speed FIR digital filters", *Solid-State Circuits, IEEE Journal of*, vol.31, no.5pp.656-667, May 1996.

[37]. Higuchi, T., Niwa, T., Tanaka, T., Iba, H., De Garis, H., Furuya, T., "Evolving hardware with genetic learning: A first step towards building a Darwin machine", In Meyer, J.A., Poitblat, H.L., Stewart, W., editors, *From Animals to Animats II: Proceedings of the 2nd International Conference on Simulation of Adaptive Behaviour*, pp. 417-424, Cambridge, MA1993. MIT Press.

[38]. Higuchi, T., Murakawa, M., Iwata, M., Kajitani, I., Weixin Liu, Salami, M., "Evolvable hardware at function level", *Evolutionary Computation, 1997., IEEE International Conference on* , vol., no.pp.187-192, 13-16 Apr 1997.

[39]. Higuchi, T., Kajihara, N., "Evolvable hardware chips for industrial applications", Communications of the ACM, vol. 42, no. 4, pp. 60-66, ACM, 1999.

[40]. Higuchi, T., Iwata, M., Keymeulen, D., Sakanashi, H., Murakawa, M., Kajitani, I., Takahashi, E., Toda, K., Salami, N., Kajihara, N., Otsu, N., "Real-world applications of analog and digital evolvable hardware", *Evolutionary Computation, IEEE Transactions on* , vol.3, no.3pp.220-235, Sep 1999.

[41]. Hinamoto, T., Ohnishi, H., Wu-Sheng Lu, "Roundoff noise minimization of state-space digital filters using separate and joint error feedback/coordinate transformation optimization", *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol.50, no.1pp. 23- 33, Jan. 2003.

[42]. Holland, J. H., "Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology", Control and Artificial Intelligence. Ann Arbor, MI: University of Michigan Press 1975. Second Edition. Cambridge, MA: The MIT Press 1992.

[43]. Hounsell, B.L., Arslan, T., "Evolutionary design and adaptation of digital filters within an embedded fault tolerant hardware platform", *Evolvable Hardware, 2001. Proceedings. The Third NASA/DoD Workshop on,* vol., no.pp.127-135, 2001.

[44]. Hounsell B.I., Arslan, T., Thomson, R., "Evolutionary design and adaptation of high performance digital filters within an embedded reconfigurable fault tolerant hardware platform", Soft. Comp. – A Fusion of Found., Methoh. and Appl., vol 8, issue 5, pp. 307-317, 2004.

[45]. Ifeachor, C.E., Jervis, W.B., "Digital Signal Processing: A practical approach", *2nd edition*, Addison-Wesley, 2002.

[46]. Jensen, M.T., "Reducing the run-time complexity of multi-objective EAs: The NSGA-II and other algorithms", *Evolutionary Computation, IEEE Transactions on,* vol.7, no.5pp. 503- 515, Oct. 2003.

[47]. Johnston, A.H., Swift, G.M., Shaw, D.C., "Impact of CMOS scaling on single-event hard errors in space systems", *Low Power Electronics, 1995., IEEE Symposium on* , vol., no.pp.88-89, 9-11 Oct 1995.

[48]. Evolvable Hardware for Autonomous Systems, CEC – 2004 Tutorial, Stoica, A., Keymeulen, D., Zebulum, R., Ferguson, M.I., Daud, T., Arslan, T., Jet-Propulsion-Laboratory, Portland, Oregon.
http://ehw.jpl.nasa.gov/Content/Public/TutorialCEC2004/TutorialCEC2004.pdf.

[49]. Kajitani, I., Hushino, T., Nishikawa, D., Yokoi, H., Nakaya, S., Yamauchi, T., Inuo, T., Kajihara, N., Iwata, M., Keymeulen, D., "A gate-level EHW chip: Implementing GA operations and reconfigurable hardware on a single LSI", In Sipper, M., Mange, D., and Perez-Uribe, editors, *Proceedings of the 2nd International Conferenceon Evolvable Systems: From Biology to Hardware*, vol. 1478 of *Lecture Notes in Computer Sciences*, pp. 1-12, Heidelberg, 1998. Springer-Verlag.

[50]. Kajitani, I., Murakawa, M., Nishikawa, D., Yokoi, H., Kajihara, N., Iwata, M., Keymeulen, D., Sakanashi, H., Higuchi, T., "An evolvable hardware chip for prosthetic hand controller", Microelectronics for Neural, Fuzzy and Bio-Inspired Systems, 1999. MicroNeuro 1999. Proceedings of the Seventh International Conference on, 7-9 April 1999.

[51]. Kalyanmoy, D., "Multi-objective optimization using evolutionary algorithms", Addison-Wesley, 2002.

[52]. Keymeulen, D., Zebulum, R.S., Jin, Y., Stoica, A., "Fault-tolerant evolvable hardware using field-programmable transistor arrays", *Reliability, IEEE Transactions on*, vol.49, no.3pp.305-316, Sep 2000.

[53]. Keymeulen, D., Klimeck, G., Zebulum, R., Stoica, A., Lazaro, C., "EHWPack: A parallel software/hardware environment for evolvable hardware", In Whitley Darrell (eds.), Proceedings of the Genetics and Evolutionary Computation Conference (GECCO-2000), July 8-12, 2000, pp. 538-539. Las Vegas, Nevada, USA.

[54]. Kourepenis, A., Borenstein, J., Connelly, J., Elliott, R., Ward, P., Weinberg, M., "Performance of MEMS inertial sensors", *Position Location and Navigation Symposium, IEEE 1998* , vol., no.pp.1-8, 20-23 Apr 1998.

[55]. Koza, J.R., Bennett, F.H., Andre, D., Keane, M. A., "Genetic programming III – Darwinian invention and problem solving", Morgan Kaufman, San Francisco, 1999.

[56]. Koza, J.R., Bennett, F.H., Andre, D., Keane, M.A., Dunlap, F., "Automated synthesis of analog electrical circuits by means of genetic programming," *IEEE Transactions on Evolutionary Computation,* vol. 1, no. 2, pp. 109-128, 1997.

[57]. Koza J. R., Andre, D., Bennett, F.H., Keane, M.A., "Design of a high-gain operational amplifier and other circuits by means of genetic programming", *Evolutionary Programming VI. 6th International Conference, EP97,* Indianapolis, Indiana, USA, April 1997 Proceedings. Lecture Notes in Computer Science, Volume 1213. Berlin: Springer-Verlag. Pages 125-136.

[58]. Kuan-Hung Chen, Tzi-Dar Chiueh, "Design and implementation of a reconfigurable FIR filter", *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on,* vol.4, no.pp. IV-205- IV-208 vol.4, 25-28 May 2003.

[59]. Kurosawa, H., Iwai, H., Ishihara, M., "Alpha-particle-induced hard error mechanism in DRAMs", *Electron Devices, IEEE Transactions on* , vol.35, no.12pp.2432-2433, Dec 1988.

[60]. Langeheine, J., Meier, K., Schemmel, J., Trefzer, M., "Intrinsic evolution of digital-to-analog converters using a CMOS FPTA chip", *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on,* vol., no.pp. 18- 25, 24-26 June 2004.

[61]. Lattice CPLD and SPLD Product Family, http://www.latticesemi.com/products/index.cfm. Lattice Semiconductor Corporation 2006.

[62]. Layzell, P., "Reducing hardware evolution's dependency on FPGAs", *Microelectronics for Neural, Fuzzy and Bio-Inspired Systems, 1999. MicroNeuro '99. Proceedings of the Seventh International Conference on,* vol., no.pp.171-178, 1999.

[63]. Lefevre, H.C., Bourbin, Y., Graindorge, P., Arditty, H.J., "Review of fiber optic gyroscopes", *Proceedings of SPIE the international society for optical engineering*, vol. 369, pp. 386, 1983.

[64]. Lefevre, V., "Multiplication by an integer constant", *Research report RR1990-08, Laboratoire de l' informatique du parallelisme,* Lyon, France, 1999.

[65]. S., Levi, A. K., Agrawala, "Fault Tolerant System Design", McGraw-Hill, New York, 1994.

[66]. Linden, D.S., Altshuler, E.E., "Evolving wire antennas using genetic algorithms: a review", *Evolvable Hardware, 1999. Proceedings of the First NASA/DoD Workshop on,* vol., no.pp.225-232, 1999.

[67]. Lohn, J.D., Haith, G.L., Colombano, S.P., Stassinopoulos, D., "Towards evolving electronic circuits for autonomous space applications", *Aerospace Conference Proceedings, 2000 IEEE*, vol.5, no.pp.473-486 vol.5, 2000.

[68]. Lohn, J.D., Hornby, G.S., "Evolvable hardware: using evolutionary computation to design and optimize hardware systems", *Computational Intelligence Magazine, IEEE*, vol.1, no.1pp. 19- 27, Feb. 2006.

[69]. The Mathworks, www.mathworks.com/access/helpdesk/help/toolbox/signal/freqz.html. 1994-2006, The Mathworks Inc.

[70]. McClellan, J.H., Parks, T.H., "A unified approach to the design of optimum FIR linear-phase digital filters", *IEEE Trans. Circuit Theory*, vol. CT-20, pp. 506-526, 1973.

[71]. M'Closkey, R.T., Gibson, S., Hui, J., "Input-output dynamics of the JPL microgyroscope", *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, vol.4, no.pp.4328-4333 vol.4, 16-18 Dec 1998.

[72]. M'Closkey, R., Challoner, A.D., "Modeling, identification, and control of micro-sensor prototypes", *American Control Conference, 2004. Proceedings of the 2004*, vol.1, no.pp. 9- 24 vol.1, 30 June-2 July 2004.

[73]. M'Closkey, R.T., Vakakis, A., "Analysis of a microsensor automatic gain control loop", *American Control Conference, 1999. Proceedings of the 1999*, vol.5, no.pp.3307-3311 vol.5, 1999.

[74]. Miller, J.F., "On the filtering properties of evolved gate arrays", *Evolvable Hardware, 1999. Proceedings of the First NASA/DoD Workshop on*, vol., no.pp.2-11, 1999.

[75]. Continuous Tuning and Calibration of Vibratory Gyroscopes, NASA's Jet-Propulsion-Laboratory, PASADENA, California, http://www.nasatech.com/Briefs/Oct03/NPO30449.html.

[76]. Park, T.S., Lee, C.H., Chung, D.J., "Intrinsic evolution for synthesis of fault recoverable circuit", *IEICE Trans Fundamentals* E83-A(12): 2488-2497, 2000.

[77]. Pasko, R., Schaumont, P., Derudder, V., Vernalde, S., Durackova, D., "A new algorithm for elimination of common subexpressions", *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.18, no.1pp.58-68, Jan 1999.

[78]. Patel, J.H., Fung, L.Y., "Concurrent error detection in ALU's by recomputing with shifted operands", *Computers, IEEE Transactions on*, vol.C-31, no.7, pp. 589- 595, Jul 1982.

[79]. Patrick, D., Green, P., York, T., "A distributed genetic algorithm environment for UNIX workstation clusters", *Genetic Algorithms In Engineering Systems:Innovations And Applications, 1997. GALESIA 97. Second International Conference On (Conf. Publ. No. 446)*, vol., no.pp.69-74, 2-4 Sep 1997.

[80]. Crossbow, Rate Gyro Application Note, www.xbow.com/Support/Support_pdf_files/RateSensorAppNote.pdf

[81]. Porter, R., McCabe, K., Bergmann, N., "An applications approach to evolvable hardware", *Evolvable Hardware, 1999. Proceedings of the First NASA/DoD Workshop on*, vol., no.pp.170-174, 1999.

[82]. Redmill, D.W., Bull, D.R., "Design of low complexity FIR filters using genetic algorithms and directed graphs", *Genetic Algorithms In Engineering Systems:Innovations And Applications, 1997. GALESIA 97. Second International Conference on (Conf. Publ. No. 446)*, vol., no.pp.168-173, 2-4 Sep 1997.

[83]. Samueli, H., "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients", *Circuits and Systems, IEEE Transactions on*, vol. 36, no. 7, pp. 1044-1047, Jul 1989.

[84]. Schnier, T., Yao, X., Liu, P., "Digital filter design using multiple Pareto fronts", *Evolvable Hardware, 2001. Proceedings. The Third NASA/DoD Workshop on*, vol., no.pp.136-145, 2001.

[85]. Scott, S.D., Samal, A., Seth, S., "HGA: A Hardware-based Genetic Algorithm", *Field-Programmable Gate Arrays, 1995. FPGA '95. Proceedings of the Third International ACM Symposium on*, vol., no.pp. 53- 59, 1995.

[86]. Shackleford, B., Okushi, E., Yasuda, M., Koizumi, H., Seo, K., Iwamoto, T., Yasuura, H., "An FPGA-based genetic algorithm machine", *Proceedings of the 2000 ACM/SIGDA 8th International Symposium on FPGAs,* Monterey, California, US, pp. 218, ACM press, 2000.

[87]. Sipper, M., Goeke, M., Mange, D., Stauffer, A., Sanchez, E., Tomassini, M., "The firefly machine: online evolware", *Evolutionary Computation, 1997., IEEE International Conference on* , vol., no.pp.181-186, 13-16 Apr 1997.

[88]. Sitkoff, N., Wazlowski, M., Smith, A., Silverman, H., "Implementing a genetic algorithm on a parallel custom computing machine", *FPGAs for Custom Computing Machines, 1995. Proceedings. IEEE Symposium on,* vol., no.pp.180-187, 19-21 Apr 1995.

[89]. Sriranganathan, S., Bull, D.R., Redmill, D.W., "Design of 2-D multiplierless FIR filters using genetic algorithms", *Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995. GALESIA. First International Conference on (Conf. Publ. No. 414),* vol., no.pp.282-286, 12-14 Sep 1995.

[90]. Sterpone, L., Violante, M., Rezgui, S., "An analysis based on fault injection of hardening techniques for SRAM-based FPGAs", *Nuclear Science, IEEE Transactions on,* vol.53, no.4pp. 2054- 2059, Aug. 2006.

[91]. Stoica, A., Keymeulen, D., Tawel, R., Salazar-Lazaro, C., Wei-Te Li, "Evolutionary experiments with a fine-grained reconfigurable architecture for analog and digital CMOS circuits", *Evolvable Hardware, 1999. Proceedings of the First NASA/DoD Workshop on,* vol., no.pp.76-84, 1999.

[92]. Stocia, A., Keymeulen, D., Duong, V., Salazar-Lazaro, C., "Automatic synthesis and fault-tolerant experiments on an evolvable hardware platform", *Aerospace Conference Proceedings, 2000 IEEE,* vol.5, no.pp.465-471 vol.5, 2000.

[93]. Stoica, A., Zebulum, R.S., Keymeulen, D., "Mixtrinsic evolution", *Proc. 3rd International Conference on Evolvable Systems: From Biology to Hardware (ICES2000),* J. Miller et al. Eds. Pp. 208-217, Springer-Verlag, 2000.

[94]. Stoica, A., Zebulum, R.S., Ferguson, M.I., Keymeulen, D., Vu Dong, "Evolving circuits in seconds: experiments with a stand-alone board-level evolvable system", *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on,* vol., no.pp. 67-74, 2002.

[95]. Stomeo, E., Kalganova, T., Lambert, C., "Generalized disjunction decomposition for the evolution of programmable logic array structures", *Adaptive Hardware and*

Systems, 2006. AHS 2006. First NASA/ESA Conference on, vol., no.pp. 179- 185, 15-18 June 2006.

[96]. Tamir; Y., Tremblay, M., "High-performance fault-tolerant VLSI systems using micro rollback", *Computers, IEEE Transactions on,* vol. 39, no. 4, pp. 548-554, Apr 1990.

[97]. Tang, T.K., Gutierrez, R.C., Wilcox, J.Z., Stell, C., Vorperian, V., Calvet, R., Li, W.J., Charkaborty, I., Bartman, R., Kaiser, W.J., "Silicon bulk micromachined vibratory gyroscope", *Solid-state sensor and actuator workshop,* Hilton Head, Sc, pp. 288-293, 1996.

[98]. Tang, T.K., Gutierrez, R.C., Wilcox, J.Z., Stell, C., Vorperian, V., Dickerson, M., Goldstein, B., Savino, J.L., Li, W.J., Calvet, R., Charkaborty, I., Bartman, R., Kaiser, W.J., "Silicon bulk micromachined vibratory gyroscope for microspacecraft", *Proc. of the SPIE – The International Society for Optical Engineering,* Denver, CO, vol. 2810, pp. 101-115, 1996.

[99]. Tang, T.K., Gutierrez, R.C., Stell, C., Vorperian, V., Arakaki, G.A., Rice, J.T., Li, W.J., Charkaborty, I., Shcheglov, K., Wilcox, J.Z., Kaiser, W.J., "A packaged silicon MEMS vibratory gyroscope for microspacecraft", *Proc. IEEE, The 10th annual international workshop on Micro Electro Mechanical Systems,* Nagoya, Japan, pp. 500-505, 1997.

[100]. Thompson, A., "Evolving fault tolerant systems", Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995. GALESIA. First International Conference on (Conf. Publ. No. 414), vol., no.pp.524-529, 12-14 Sep 1995.

[101]. Thompson, A., "Silicon evolution", Proceedings of Genetic Programming 1996 (GP96), J.R., Koza et al. (eds), pp. 444-452, MIT press, 1996.

[102]. Thompson, A., "An evolved circuit, intrinsic in silicon, entwined with physics", *Proceedings of the 1st International Conference on Evolvable Systems: From Biology to Hardware (ICES96),* LNCS, vol. 1259, Springer-Verlag, Heidelberg, pp. 390-405, 1997.

[103]. Thomson, R., Arslan, T., "Evolvable hardware for the generation of sequential filter circuits", *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on,* vol., no.pp. 17- 25, 2002.

[104]. Thomson, R., Arslan, T., "An evolutionary algorithm for the multi-objective optimisation of VLSI primitive operator filters", *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on,* vol.1, no.pp.37-42, 12-17 May 2002.

[105]. High Performance: TM 5320 C6000 DSP Platform, http://focus.ti.com/paramsearch/docs/parametricsearch.tsp?family=dsp&sectionId=2&tabId=25&familyId=44. Copyright 1995-2005, Texas Instruments Incorporated. All rights reserved.

[106]. Torresen, J., "Exploring knowledge schemes for efficient evolution of hardware", *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on*, vol., no.pp. 209- 216, 24-26 June 2004.

[107]. Tranfield, R., "INS/GPS navigation systems for land applications", *Position Location and Navigation Symposium, 1996., IEEE 1996*, vol., no.pp.391-398, 22-26 Apr 1996.

[108]. Trefzer, M., Langeheine, J., Meier, K., Schemmel, J., "A Modular framework for the evolution of circuits on configurable transistor array architectures", *Adaptive Hardware and Systems, 2006. AHS 2006. First NASA/ESA Conference on*, vol., no.pp. 32- 42, 15-18 June 2006.

[109]. Tufte, G., Haddow, P.C., "Evolving an adaptive digital filter", *Evolvable Hardware, 2000. Proceedings. The Second NASA/DoD Workshop on*, vol., no.pp.143-150, 2000.

[110]. Yong Lim, S., Parker, "FIR filter design over a discrete powers-of-two coefficient space", Acoustics, Speech, and Signal Processing, IEEE Transactions on, vol. 31, no. 3, pp. 583-591, Jun 1983.

[111]. Vassilev, V.K., Miller, J.E., "Scalability problems of digital circuit evolution evolvability and efficient designs", *Evolvable Hardware, 2000. Proceedings. The Second NASA/DoD Workshop on*, vol., no.pp.55-64, 2000.

[112]. Vinger, K.A., Torresen, J., "Implementing evolution of FIR-filters efficiently in an FPGA", *Evolvable Hardware, 2003. Proceedings. NASA/DoD Conference on*, vol., no.pp. 26- 29, 9-11 July 2003.

[113]. Voronenko, Y., Puschel, M., "Multiplierless multiple constant multiplication", (to appear in *ACM Journal*).

[114]. Wacey, G., Bull, D.R., "Architectural synthesis of digital filters for ASIC implementation", *Digital and Analogue Filter and Filtering Systems, IEE Colloquium on,* vol., no., pp. 6/1-6/5, 13 Dec 1991.

[115]. Wade, G., Roberts, A., Williams, G., "Multiplier-less FIR filter design using a genetic algorithm", *Vision, Image and Signal Processing, IEE Proceedings-* , vol.141, no.3pp.175-180, Jun 1994.

[116]. Wakabayashi, S., Koide, T., Toshine, N., Goto, M., Nakayama, Y., Hayya, K., "An LSI implementation of an adaptive genetic algorithm with on-the-fly crossover operator selection", *Design Automation Conference, 1999. Proceedings of the ASP-DAC '99. Asia and South Pacific* , vol., no.pp.37-40 vol.1, 18-21 Jan 1999.

[117]. Weinstein, C., "Roundoff noise in floating point fast Fourier transform computation", *Audio and Electroacoustics, IEEE Transactions on*, vol.17, no.3pp. 209-215, Sep 1969.

[118]. Xilinx data book, 2000, www.xilinx.com.

[119]. Jiangning Xu, Arslan, T., Dejun Wan, Qing Wang, "GPS attitude determination using a genetic algorithm", *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol.1, no.pp.998-1002, 12-17 May 2002.

[120]. Jiangning Xu, Arslan, T., Qing Wang, Dejun Wan, "An EHW architecture for real-time GPS attitude determination based on parallel genetic algorithm", *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on*, vol., no.pp. 133- 141, 2002.

[121]. Yazdi, N., Ayazi, F., Najafi, K., "Micromachined inertial sensors", *Proceedings of the IEEE*, vol.86, no.8pp.1640-1659, Aug 1998.

[122]. Yu, A.; Siddiqui, A.S., "Novel fibre optic gyroscope with a configuration combining Sagnac interferometer with fibre ring resonator", *Electronics Letters*, vol.28, no.19pp.1778-1780, 10 Sep 1992.

[123]. Zebulum, R.S., Pacheco, M.A., Vellasco, M., "Artificial Evolution of Active Filters: A case study", *Proc. of the First NASA DoD Workshop on Evolvable Hardware*, pp. 66-75, IEEE Computer press., July, 1999.

[124]. Zebulum, R., Stoica, A., Keymeulen, D., "Experiments on the evolution of digital to analog converters", *Aerospace Conference, 2001, IEEE Proceedings.*, vol.5, no.pp.2321-2331 vol.5, 2001.

[125]. Yang Zhang, Smith, S.L., Tyrrell, A.M., "Digital circuit design using intrinsic evolvable hardware", *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on*, vol., no.pp. 55- 62, 24-26 June 2004.

[126]. Zwyssig, E., "Low power digital filter design for hearing aid applications", Master's Thesis. The University of Edinburgh, UK, 2000.

Appendix A
List of Publications

1. **Stefatos, E.F.**, Arslan, T., "An efficient fault-tolerant VLSI architecture using parallel evolvable hardware technology", *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on*, vol., no., pp. 97- 103, 24-26 June 2004.

2. **Stefatos, E.F.**, Arslan, T., "High-performance adaptive GPS attitude determination VLSI architecture", *Signal Processing Systems, 2004. SIPS 2004. IEEE Workshop on*, vol., no., pp. 233- 238, 13-15 Oct. 2004.

3. **Stefatos, E.F.**, Wei, H., Arslan, T., Thomson, R., "Low-power reconfigurable VLSI architecture for the implementation of FIR filters", *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, vol., no., pp. 4 pp.-, 4-8 April 2005.

4. **Stefatos, E.F.**, Arslan, T., Keymeulen, D., Ferguson, I., "An EHW architecture for the design of unconstrained low-power FIR filters for sensor control using custom-reconfigurable technology", *Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on*, vol., no., pp. 147- 153, 29 June-1 July 2005.

5. **Stefatos, E.F.**, Arslan, T., Keymeulen, D., Ferguson, I., "Custom reconfigurable architecture for autonomous fault-recovery of MEMS vibratory sensor electronics", *VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design., 19th International Conference on*, vol., no., pp. 4 pp.-, 3-7 Jan. 2006.

6. **Stefatos, E.F.**, Arslan, T., Keymeulen, D., Ferguson, I., "Autonomous realization of Boeing/JPL sensor electronics based on reconfigurable system-on-chip technology", *Emerging VLSI Technologies and Architectures, 2006. IEEE Computer Society Annual Symposium on*, vol.00, no., pp. 6 pp.-, 2-3 March 2006.

7. **Stefatos, E.F.**, Bravos, I., Arslan, T., "Low-power implementation of FIR filters within an adaptive reconfigurable architecture", *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, vol., no.pp. 4 pp.-, 21-24 May 2006.

8. **Stefatos, E.F.**, Arslan, T., Keymeulen, D., Ferguson, I., "Towards the Integration of Drive Control Loop Electronics of the JPL/Boeing Gyroscope within an Autonomous Robust Custom-Reconfigurable Platform", *Adaptive Hardware and*

Systems, 2006. AHS 2006. First NASA/ESA Conference on, vol., no.pp. 207- 214, 15-18 June 2006.

9. **Stefatos, E.F.**, Arslan, T., "An Incremental Evolutionary Strategy for the Design of FIR Filters Targeting Real-Time Applications", *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, vol., no., pp. 2787- 2792, 16-21 July 2006.

10. **Stefatos, E.F.**, Arslan, T., Keymeulen, D., Ferguson, I., "Integrating the Electronics of the Control-Loops of the JPL/Boeing Gyroscope within an Evolvable Hardware Architecture", *Field Programmable Logic and Applications (FPL '06)*, International Conference on, Vol., Iss., Aug. 2006, Pages: 1-4.

11. **Stefatos, E.F.**, Arslan, T., Hamilton, A., Keymeulen, D., "An Autonomous Evolvable Hardware Platform for the Efficient Implementation of Low-Power and Fault-Tolerant FIR Filters", *submitted on IEEE Transactions on Computers*.

12. **Stefatos, E.F.**, Arslan, T., Hamilton, A., "Enhanced Evolutionary Techniques for Precise and Real-Time Implementation of Low-Power FIR Filters", *submitted on IEEE Transactions on Evolutionary Computation*.