Tolerant GPS receiver circuit for electronics errors

Mohamed Mourad Hafidhi

UNIVERSITE BRETAGNE LOIRE

THÈSE / UNIVERSITÉ DE BRETAGNE SUD

UFR Sciences et Sciences de l'Ingénieur

sous le sceau de l'Université Européenne de Bretagne

Pour obtenir le grade de :

DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE SUD

Mention : STICC

École Doctorale SICMA

présentée par

Mohamed Mourad HAFIDHI

Lab-STICC: Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la connaissance

Un circuit de réception GPS tolérant aux erreurs de l'électronique

Thèse soutenue le 16-11-2017,
devant la commission d'examen composée de :

M. Jean-Didier Legat
Professeur, Université Catholique de Louvain / Président

M. Patrick Girard
Directeur de Recherche CNRS, LIRMM Montpellier / Rapporteur

M. Emmanuel Simeu
Maître de conférences, HDR, TIMA Grenoble / Rapporteur

M. Arnaud Dion
Ingénieur de recherche, ISAE Toulouse / Examinateur

Mme. Fabienne Uzel-Nouvel
Maître de conférences, HDR, INSA de Rennes / Examinatrice

M. Emmanuel Boutillon
Professeur, Université de Bretagne-Sud / Directeur de thèse

Résumé — Avec l'avancée technologique, la minimisation des transistors et la réduction des tensions d'alimentation ont permis de concevoir des applications complexes à faible consommation. Cependant, en abordant les limites de l'intégration des transistors de faible tension d'alimentation, la fiabilité des circuits devient critique. Les circuits sont sujet des erreurs dues aux perturbations environnementales, défauts de fabrication et interférences. L'apparition de ces erreurs peut affecter le comportement du circuit et générer en sortie un dysfonctionnement du système. Par conséquent, il est de plus en plus important de considérer les effets de ces erreurs dans la conception des futurs circuits.

L'objectif de la thèse est de traiter la fiabilité des systèmes numériques et d'introduire de nouvelles techniques de tolérance aux pannes permettant de construire des applications de traitement de signal fiables sur un électronique peu fiable. Un exemple d'application a été considéré durant la thèse: les modules de poursuite dans un récepteur GPS. Ces modules contiennent un ensemble d'applications de traitement de signal avec des exigences de fiabilité différentes : fonction de corrélation, boucles fermées, machine d'état, générateurs de codes et de porteuses. À partir d'une version standard d'un récepteur GPS, des mécanismes de redondance ont été proposés et ajoutés pour concevoir un récepteur GPS plus tolérant aux erreurs. Un circuit intégré (ASIC) sera conçu en utilisant la technologie 28 nm pour valider les performances de ces techniques et faire les tests de mesures de consommation d'énergie. Au cours de la thèse, une plate-forme d'émulation a été conçue pour préparer l'environnement expérimental à utiliser une fois l'ASIC fondu.

Abstract — There is continual motivation to scale down transistors and reduce the supply voltage of circuits. However, by approaching the limits of transistor scaling and operating at a minimal supply voltage, circuit reliability has emerged as a critical concern. Circuits become more and more susceptible to errors due to Process, voltage and temperature (PVT) variations. Occurrence of errors can affect the behaviour of circuits and generate a permanent system failure. Therefore, it is increasingly important to deal with errors effects in order to keep future devices working properly.

The objective of the thesis is to address the reliability in digital systems and introduce new fault tolerant techniques to perform reliable signal processing applications on unreliable hardware. An example of application has been considered in the thesis: the tracking process of GPS receivers. It contains a very interesting set of different signal processing problem with different requirements of reliability: Correlation process, tracking loops (recursive operations), state machine, Gold and carrier generators. Starting from a noiseless GPS receiver, redundant mechanisms have been proposed and added to design a more resilient GPS receiver tolerant to errors. An Application-Specific Integrated Circuit (ASIC) will be designed, based on thesis results, using the 28 nm technology to validate the performances of the proposed techniques performances. During the thesis, an emulation platform was designed to prepare the experimental environment for the ASIC.

N° ordre : 458
Université de Bretagne Sud, Lab-STICC
Centre de Recherche Christiaan Huygens - BP 92116, 56321 Lorient CEDEX
Tél : + 33(0)2 97 87 45 62 Fax : + 33(0)2 97 87 45 27

Acknowledgements

I would like to express my deepest gratitude to my thesis director, Prof. Emmanuel Boutillon. His encouraging attitude, profound guidance and invaluable advises helped keeping my research well directed. Also for his carefully reviewing this report which helped to improve the quality of the manuscript.

My sincere thanks also goes to Arnaud Dion, Guillaume Beaugendre and Benoit Priot, members of the Navigation team in the Institut Supérieur de l'Aéronautique et de l'Espace. Their support and valuable advices helped me resolve many problems during my research.

My grateful thanks are also extended to Prof. Chris Winstead for his generous welcome and his valuable and constructive suggestions during my stay in Utah State University for a collaborative research visit.

I would also like to acknowledge all my master students who provided me an opportunity to enrich my understanding through sharing my knowledge about the subject and also for their valuable contribution to my work.

Finally, none of this would have been possible without the love and patience of my family and my friends. Their constant support and strength has aided and encouraged me throughout this endeavour.

Contents

List of Figures

List of Tables

Glossary

AMBA Advanced Micro-controller Bus Architecture. 104

ANT Algorithmic Noise Tolerance. 1, 19, 20, 26

ARQ Automatic Repeat reQuest. 23, 25

ASET Algorithmic Soft Error Tolerance. 19

ASIC Application-specific integrated circuit. 2, 3

AXI Advanced eXtensible Interface. 84, 104, 105

BRRNS Bi-Directional Redundant Residue Number System. 17, 18, 23, 25

CMOS Complementary Metal Oxide Semiconductor. 2

CRT Chinese Remainder Theorem. 17, 18

DSC Duplication with Syndrome based Corrector technique. 2

FFT Fast Fourier transform. 23

FIR Finite Impulse Response. 25

FT-ECC Fault Tolerant Error Code Correcting based technique. 2

GPS Global Positioning System. vi, vii, 2, 3, 53, 59–63, 65, 66, 73–81

ISAE Institut Supérieur de l'Aéronautique et de l'Espace. 3, 81, 83, 84

LFSR Linear Feedback Shift Register. 66–71

MRC Mixed Radix Conversion. 18

MTTF Mean Time To Failure. 2

MWSCAS 2015 Midwest Symposium on Circuits and Systems. 74

PVT Process, Voltage and Temperatures. 2, 6

RE-Criteria Reliability Efficiency Criteria. 22, 23

RFB Restorative FeedBack. 14, 16, 26

RNS Residue Number System. 16, 17

RPR Reduced Precision Redundancy. 19

S-TMR Spatial TMR. 13, 23, 26

SDK Software Development Kit. 93

SEU Single Event Upsets. 7

SNR Signal to Noise Ratio. 86

SOI Silicon On Insulator. 7

T-TMR Temporal TMR. 13, 14, 23, 24

TMR Triple Modular Redundancy. 2, 13, 14, 16

Introduction

During the past decades, transistor dimensions scaling has greatly lead to enhance computational capabilities of devices and build more and more complex system in one circuit, billions of transistors can now be integrated per square of centimetres in single chip [2]. As well as the huge performances improvements, supply voltage of circuits have also been reduced which has driven to a power savings of circuits. However, approaching the manufacturing nanoscale imperfections limits and by operating at a minimal supply voltage, technology scaling faces new challenges. The increase in integration density makes transistors performances varying under high temperatures and low voltage conditions. Therefore, the reliability of circuit is no longer guaranteed. Faults can appear in transistor outputs which can change the state of a gate, affect the performance of circuits and generate a permanent system failure. Hence, it is increasingly important to consider the reliability issue for the development of high performance and low power devices.

There are three general classes of faults than can affect the performances of a circuit: permanent fault, transient fault and timing fault. Permanent faults are an irreversible physical change that last on time: a stuck at a value of 0 or 1 in a gate output. Transient fault are a temporary change of the binary value of a signal while timing faults are momentary delay in the execution time of an operation. In the literature, a considerable amount of fault tolerant architectures have been proposed to improve reliability of systems in the presence of different type faults. It is John von Neumann that pioneered the idea of using redundancy, in the 1950's, to improve the reliability of systems in [3]. He proved the existence of an arbitrarily reliable architecture under constraint of a minimum reliability of the NAND gate and showed that the objective of making reliable computation on an unreliable architecture is realistic. The well-known Triple Modular Redundancy (TMR) appears as a similar approach with less complexity [4]. In a TMR system, the original module is replicated three times, and error correction is achieved by a majority vote operation. Predictions techniques, such as Algorithmic Noise Tolerance (ANT) technique [5], propose to increase the reliability of circuit by adding a reduced precision replica to the original function. This reduced precision replica consumes much less power than the original function. The final output is chosen between the original function output and the output of the replica. Error Correction Codes has also been proposed to protect memories, [6] and then applied for interconnect networks [7]. All these methods are based on the hardware redundancy to detect and eliminate faults. Another approach to detect faults is to repeat the execution of one operation multiple times and compares the outputs; it refers as the temporal redundancy based approach. The TMR-temporal is an example of temporal based techniques. Finally, Razor systems have been also proposed to deal with timing and transient faults. These systems are based on sampling the output of functions at different instants and detect occurrence of faults by observing the incoherence between the outputs at the two instants [8]. Razor systems don't deal with permanent faults.

The thesis, part of the RELIASIC project, is funded by French government sponsors COMIN Labs[1], the National Research Agency in the "Investing for the Future" program under reference ANR-10-LABX-07-01 and the Brittany Region. It addresses the reliability in digital systems and introduces new fault tolerant techniques to perform reliable signal processing applications on unreliable hardware. The robustness capacity in term of Mean Time To Failure (MTTF) and the hardware efficiency of the proposed fault tolerant techniques were solved first to study techniques improvements in general. Then, an example of digital application was considered to evaluate the quality of service of the protected application against faults. The application was the tracking process inside GPS receivers. It is a very interesting application since it contains a several signal processing problems with different requirements of reliability: Correlation process, tracking loops (recursive operations), state machine, Gold and carrier generators.... Starting from the standard version of the GPS application, redundant mechanisms have been proposed and added to design a more resilient GPS receiver tolerant to faults due to Process, Voltage and Temperatures (PVT) variations coupled with the Complementary Metal Oxide Semiconductor (CMOS) technology evolution. An Application-specific integrated circuit (ASIC) will be designed using the 28 nm technology to test the fault tolerant techniques in a real design case and validate by experiments and measurements the performances in terms of robustness and power consumption.

The following of the manuscript is organised as follow:

- Chapter 1 introduces the context and the motivation of the thesis research. It starts by discussing the reliability threat with the trend of technology scaling in integrated circuits. Then, it summarises the different approach of reliability improvement: Fault avoidance, fault removal and fault tolerance. The last section of the chapter provides a survey of the fault tolerant techniques proposed in the literature to improve reliability of circuits and compares the characteristics of their architectures.

- Chapter 2 presents two new fault tolerant techniques based on error correcting codes and duplication with syndromes computation. The two techniques are named the Fault Tolerant Error Code Correcting based technique (FT-ECC) and the Duplication with Syndrome based Corrector technique (DSC). The performances of the two techniques in term of resilience capacity and hardware efficiency are performed and compared to the performances of the classical Triple Modular Redundancy (TMR). This part of work was published in [10].

- Chapter 3 is a brief introduction of the GPS system. It describes first the different segments of the GPS navigation system. Then, it discusses how the GPS position is computed based the GPS satellites signal. Finally, we provide an overview of the GPS user equipment and we describe the structure of the GPS tracking modules.

- Chapter 4 addresses the fault tolerance of the tracking process of GPS receivers. It starts by defining elements of the tracking module that requires the high reliability concern. Then, we detail the fault tolerant techniques proposed to improve reliability

[1] Acronym of COMmunication and INFormation sciences Laboratories [9]

of each block in the GPS tracking process. This work is the summary of publications in [11], [12], [13] and [14].

- Chapter 5 describes a GPS hardware emulation platform developed by the Institut Supérieur de l'Aéronautique et de l'Espace (ISAE) to study GPS system and improve navigation algorithms. Several modifications are made on this GPS platform to be exploited for ASIC experiences and measurements: we improved a key module of the tracking loop (i.e. carrier and code estimators) and we modified the whole architecture to isolate the part of the design that will be replace by the ASIC. This chapter details these modifications and presents the resilient version that will be designed in the ASIC.

Finally, the conclusion summarises contributions during the thesis and presents future perspectives.

Chapter 1

Context and Motivation

Contents

1.1 Technology Scaling Challenges

The past five decades were marked by a huge evolution of semiconductor manufacturing technology where transistors were scaled down in order to improve their speed and decrease their cost and power dissipation. This growth helped to increase the number of transistors per area unit and to build more and more complex system in one chip. One of the first single chips was introduced by Intel in November 1971 and was named the Intel 4004. The chip had 2,300 transistors that ran at a clock speed of up to 740 KHz and dissipated in total 0.5 watts [15]. Today chips employ billions of transistors, include multiple processor cores on a single silicon die, run at clock speeds measured in gigahertz and deliver more than

4 million times the performance of the original 4004 [15]. Fig. 1.1 illustrates the trend of technology advancements during the last four decades. Moreover, since power consumption is proportional to the square of the supply voltage V_{dd}, voltage scaling has been started in the late 80s in order to reduce the consumption of circuits. During the last decades, V_{dd} was scaled from 5V to 3.3V then to 2.5V and it is predicted to be reduced to 0.64V in 2028 [16].

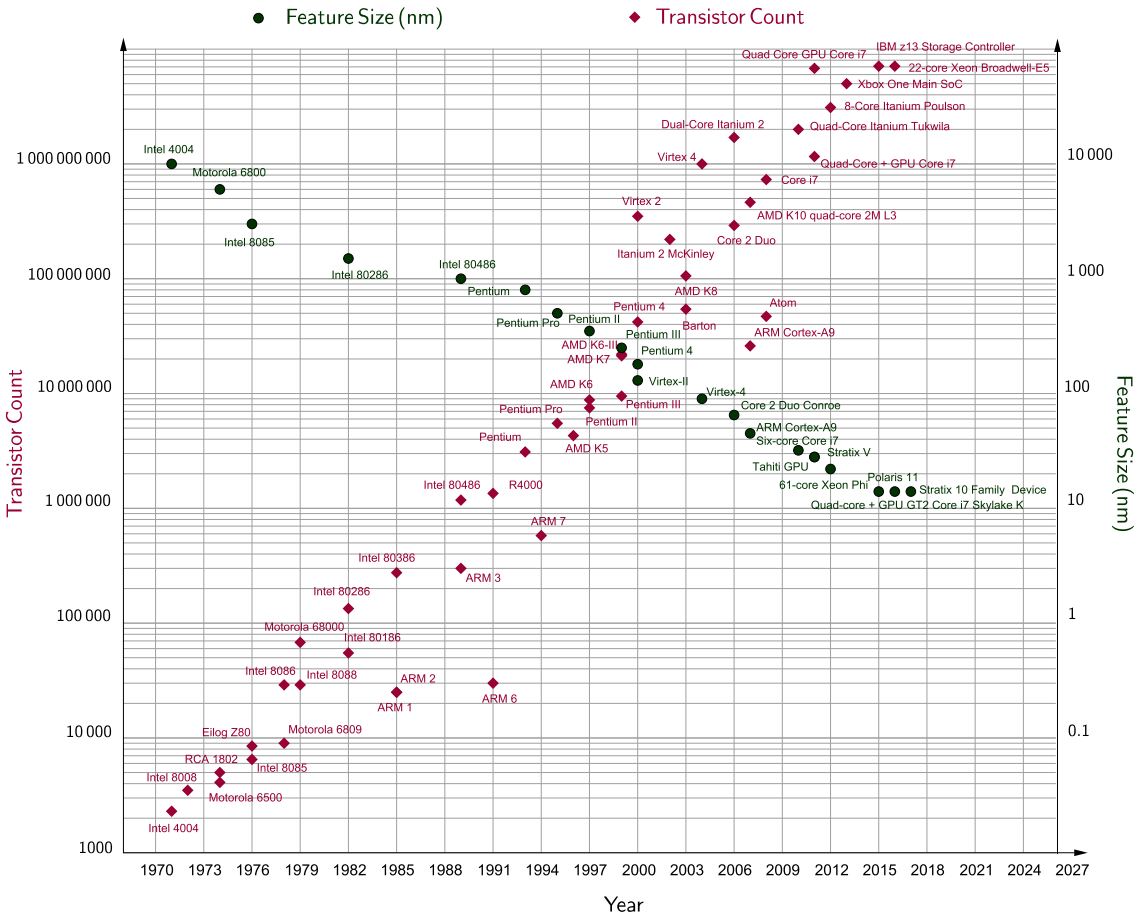


Figure 1.1: Transistor count and clock Feature size scaling trend over years.

Clearly, the increase of integration density with technology scaling has offered the possibility for designers to build very complex system on a single chip. However, approaching the limits of integration, circuit reliability[1] has emerged as a critical concern. Two main sources can affect the reliability of circuits: 1/ PVT variations and 2/ interference.

➤ **PVT variations:** Process variations are defined as the imperfections and the lack of precise control of equipment during the fabrication processes [18]. These imperfections appear as a variability of transistor characteristics such as random dopant fluctuation

[1]Reliability is defined as the "ability for a system to continue delivering correct service, i.e., perform failure-free operation, for a specified period of time" [17].

and line edge roughness. Voltage variations can increase delays inside circuit when V_{dd} is lower than the transistor threshold voltage V_{th}. Chip power densities due to scaling can also lead to global temperature variations as well as local fluctuations in regions of high-activity, so called hot-spots. The temperature fluctuations affect the timing characteristics of circuits and can provoke the wear-out failure that limit the useful lifetime and performance of circuits.

- **Interferences:** Integrated circuits are more and more sensitive to interference such as electromagnetic influences, alpha particle radiation or cosmic radiation. A local creation of charges in the bulk can flip the output of a gate of a memory flip-flop.

Occurrence of faults leads to errors that can change the state of a gate, affect the behaviour of circuits and change the output of a system. Errors or faults can be classified into three main groups: permanent, transient and timing errors. Permanent errors are defined as irreversible physical changes: a stuck at the value 0 or 1 of a bit for a gate or a memory. They can be caused by manufacturing defects or device wear-out [19]. Transient errors are temporal malfunctions that manifest as a temporary change of the binary value of a bit. The other term used for transient errors is Single Event Upsets (SEU). Environmental changes, such as higher or lower temperature and voltage variation, are the essential sources of the occurrence of these errors [20]. Timing errors result when a input signal arrives too late and misses the reference arrival of the clock due to the size down-scaling process [21]. Appearance of faults can cause a system failure if they are not eliminated. Therefore, reliability improvement practises must be considered to guarantee correct system behaviour with the actual technological evolution of the semiconductor industry.

1.2 Reliability Improvement Approaches

In order to improve reliability of devices, reliability practices might be taken at different time in the overall development cycle of a system: design time, fabrication time and device lifetime. Reliability practises differ from one level of abstraction to another. In the following sections we describe the different reliability practices according to their chronology in the development cycle as defined in [22].

1.2.1 Fault Avoidance

Fault avoidance includes techniques and methods that aim to limit the introduction of faults during specification, design and fabrication phases. A typical example of fault avoidance at design phase is employing transistor resizing within critical gate to decrease the vulnerability to errors [23]. Fault avoidance may also include technology mitigation techniques that implicate modifications of conventional manufacturing processes, for example modern chips designs use Silicon On Insulator (SOI) technology that provides many advantages including

significantly reduced susceptibility to soft errors due to the smaller volume for charge collection [24]. Besides these, using radiation hardened components is another example of fault avoidance practise at design phase.

1.2.2 Fault Removal

Despite all best effort done during the manufacturing process to avoid faults, systems may still contain faults. Fault removal techniques are applied so to find and remove causes of faults that can exist in the fabricated device. It includes designs reviews, unit verification, and system testing... Improper fabricated devices can be suppressed.

1.2.3 Fault Tolerance

Neither fault avoidance nor Fault removal ensures that fabricated circuits are free of faults. Despite all effort, faults can appear during the lifetime of a system. Besides to that, the moment of the occurrence of faults is unpredictable. Fault-tolerance is the set of measures and techniques that aim to enable continuity of correct service delivered by a system even in presence of errors during its lifetime. They involve generally two essential and sequential steps: error detection and error recovery. Error detection during the normal life of a system is referred as On-Line detection or concurrent error detection. Error recovery mechanisms follow the error detection to eliminate the effect of the errors. This section discusses some well-known techniques for error detection and the main approaches for error recovery.

1.2.3.1 On line Error Detection

There exist various strategies for error detection but what they have in common is that they all use redundancies to detect different types of errors. Redundancy takes two forms: spatial and temporal. The spatial redundancy (also known as the hardware redundancy) refers as an addition of components, functions or data in a system. This type of redundancy is very costly in terms of area and power since it is involves extra hardware resources and power dissipation for redundant components. In the temporal redundancy, a computation or a transmission of information is repeated in time. It comes with a throughput penalties and a reduction of system performance. Each approach (temporal or spatial) has advantages and disadvantages in term of cost, performance and error coverage. The decision of which approach to choose depends on the application. For applications where the system cost is less important than its reliability, spatial redundancy is suitable while temporal redundancy is used for applications that tolerate spending extra time to recompute operations. Some known error detection mechanisms are presented in the following sections.

- **Error detecting codes:** The main concept of this technique is to add some hardware redundancy to the main logic to detect possible anomalies after operation. Fig.1.2

illustrates how error detecting codes can be used for fault tolerance. The result y is the output of a circuit logic that performs a function Ψ on the input x. The redundant logic processes the input x to output $z = \Psi_R(x)$. $F(y,z)$ is a function that checks the characteristics that y and z must cater. Let consider the following two examples to illustrate this method. $1/\Psi(x_1,x_2) = x_1 \times x_2$, $\Psi_R(x_1,x_2) = (x_1 \mod 3) \times (x_2 \mod 3)$, where mod is the modulo function. $F(y,z)$ checks that $y \mod 3 = z \mod 3$. Let us take $x_1 = 7$, $x_2 = 8$. If for example the main logic is faulty and $y = 31$ instead of 56 while the redundant module is fault free, z=(7 mod 3) $\times$ (8 mod 3)= $2 \times 1 = 2$. (31 mod 3) $= 1 \neq (2 \mod 3)$. An error is flagged in this case. 2/ $y = \Psi(x) = \text{FFT(x)}$, $z = \Psi_R(x) = \|x\|^2$. $F(y,z)$ verifies if $\|y\|^2 = \|x\|^2$.
The next section will present another example of error detecting codes based technique where the redundant logic is the replica of the original logic.

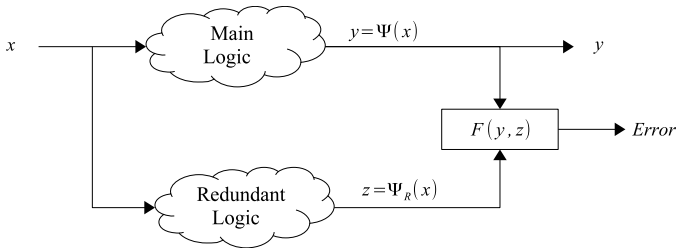
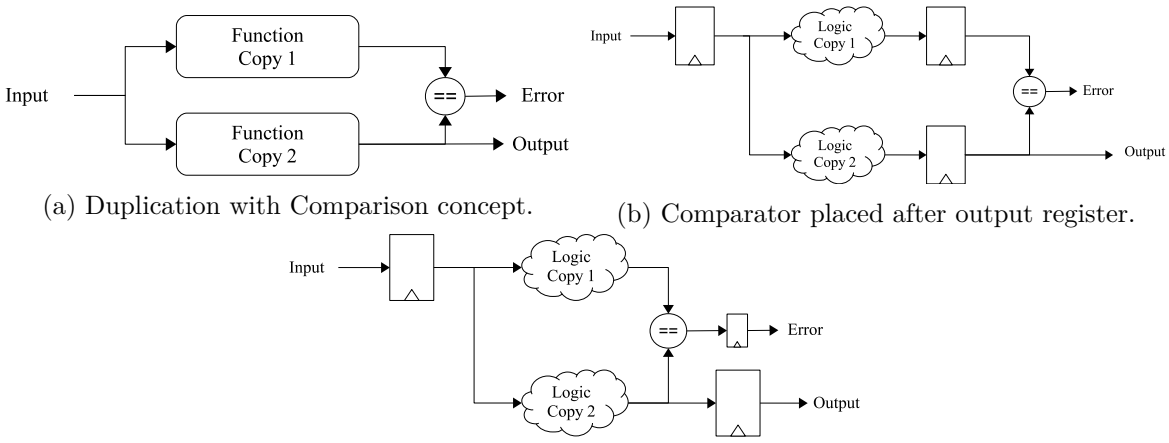


Figure 1.2: General scheme of error detection with error detecting codes.

➤ **Duplication with comparison:** Duplication with comparison is a very popular error detection technique based on hardware redundancy. The original module is duplicated, outputs are compared and an error is flagged when there is an inequality in the result computed by the two copies as shown in Fig.1.3a. The block labelled as '==' represents the static comparator. Apart its simplicity to implement, the Duplication with comparison can detect all type of faults which include permanent, transient and timing faults. An important design decision for schemes that use duplication with comparison is the placement of the comparator.

The simplest way of placing the comparator is to insert it after the output register as shown in Fig.1.3b. It is true that the scheme reduces the power dissipation in the comparator since one transition per cycle is experienced for comparator inputs, but, this requires duplication in the same time of the logic and the output register. The comparator can also be inserted before the output register as shown in Fig.1.3c. The duplication of the output register is, thus, no more required with this comparator placement. However, since the comparator compares during the entire cycle, the comparator experiences an increased switching activity (thus higher power consumption) during the time where the logic outputs are unstable due to the difference in circuit path lengths. Moreover, the comparator output is an unstable signal and needs a latching mechanism to have a stable error signal. The additional path for signals to traverse the comparator has to be considered when choosing when to store the error signal because if the error signal is captured with the clock edge, i.e. at the same time as data is saved in the register, a timing fault may get latched in the output register but can possibly escape

getting latched in error FF. The work in [25] proposes a circuit-level implementation of a special comparator that offers about 30% reductions in power compared with a static comparator with a negligible area overhead.



(a) Duplication with Comparison concept.

(b) Comparator placed after output register.

(c) Comparator placed before output register.

Figure 1.3: Duplication with Comparison scheme.

➤ **Double sampling:** The main idea of this method is to observe the output signals of a given circuit at two instants. Incoherence is flagged as error. Compared to the duplication with comparison technique, the double sampling adds a register to the original design which is less costly than adding hardware redundancy in general. Nevertheless, double sampling requires carefully timing constraints related to the critical path of combinatorial logic (short path shouldn't be detected as errors) and is effective only to detect occurrence of transient and timing errors [26].

1.2.3.2 Error recovery

Error recovery is the action to transform the system faulty state into an error-free state. There are two main approaches for error recovery: Rollback Error Recovery and Forward Error Recovery.

➤ **Rollback Error Recovery:** In rollback recovery scheme, errors correction is achieved using the temporal redundancy: 1) system sate is saved periodically or occasionally, the saved state is called a checking point and refers as an error free state for the system, 2/ once an error is detected, the system is taken back to the error free state and operation(s) is (are) repeated. These rollbacks can be just for one cycle deep [27] or up to several thousands of cycles [28]. Although it is simple and independent of the nature fault, the roll-back error recovery has a few inherent drawbacks: First, a degradation of the area due to the overhead to store the error free system sates. Second, the re-execution affect the processing throughput of the design. Finally, there is no guarantee that error will not persist when the operation is repeated (for example with permanent faults).

➤ **Forward Error Recovery:** The Forward Error Recovery mechanism uses the spatial redundancy to compensate the error state by acting on the damaged part. Error masking is a typical example of forward error recovery scheme. Since no re-computation is needed, the forward error recovery is efficient in application with timing constraints. However, this recovery mechanism needs an accurate assessment of the damage in the system, so, the use of this technique is limited a pre-defined set of applications.

1.3 Fault tolerant architectures

The challenge in fabricating small size transistors operating in low supply voltage have resulted in a possible unexpected occurrence of faults in circuits and a loss of reliability of systems. To improve the reliability against faults that appear during the lifetime of a system, a considerable amount of fault tolerant architectures have been proposed until now. In this section, we discuss a set of the relevant fault tolerant architectures that represent most categories of architectures available in literature.

1.3.1 Von Neumann Multiplexing Architectures

In 1952, Von Neumann addressed the problem of performing reliable computation on unreliable devices and introduced the first redundancy technique called multiplexing [29]. The main concept of the multiplexing technique is to replace the single processing module Ψ by the so-called "the multiplexing unit" ($N \gg 1$). The design of the multiplexing unit involves two stages: the executive stage and the restorative stage. In the executive stage, the single processing module Ψ is replicated N times. Each replica processes the same inputs in parallel which gives N independent outputs. If the inputs and devices are reliable, the N outputs should be identical. However, if there are errors, incoherence between outputs is observed. To tackle these errors, outputs of the executive stage are duplicated (or triplicated) and used as inputs of the restorative stage. The restorative stage contains k blocks of N copies of a NAND or a majority gate. The multiplexing unit is called the NAND Mux (respectively Maj Mux) when the corresponding restorative stage is formed by NAND gates (respectively majority gates). Fig. 1.4 illustrates the two multiplexing units.

In the following, we demonstrate how increasing the number of stages (k) improves fault tolerance. To simplify our study, we chose a processing module Ψ that outputs a binary signal and we determinate $\eta^{(j)}_{NAND}$ (and respectively $\eta^{(j)}_{Maj}$) the probability that the NAND gate (respectively the majority gate) in the j^{th} block U_j of the restorative stage in the NAND Mux (respectively Maj Mux) as illustrate in Appendix. A.3. Let us consider $\eta^{(1)}_{Maj}$ as defined in (A.25), p the failure probability of the module Ψ and ϵ the failure probability of the NAND and the majority gate. The equation ($\eta^{(1)}_{Maj}$ - $p = 0$) has three solutions: p_1=0.5, $p_2 = \frac{1}{2}(1+\sqrt{\frac{1-6\epsilon}{1-2\epsilon}})$, $p_3 = \frac{1}{2}(1+\sqrt{\frac{1-6\epsilon}{1-2\epsilon}})$. p_2 and p_3 exist only if $\epsilon < 1/6 = 0.16$. With that, Von Neumann demonstrated that it is possible to build a fault tolerant multiplexing architecture

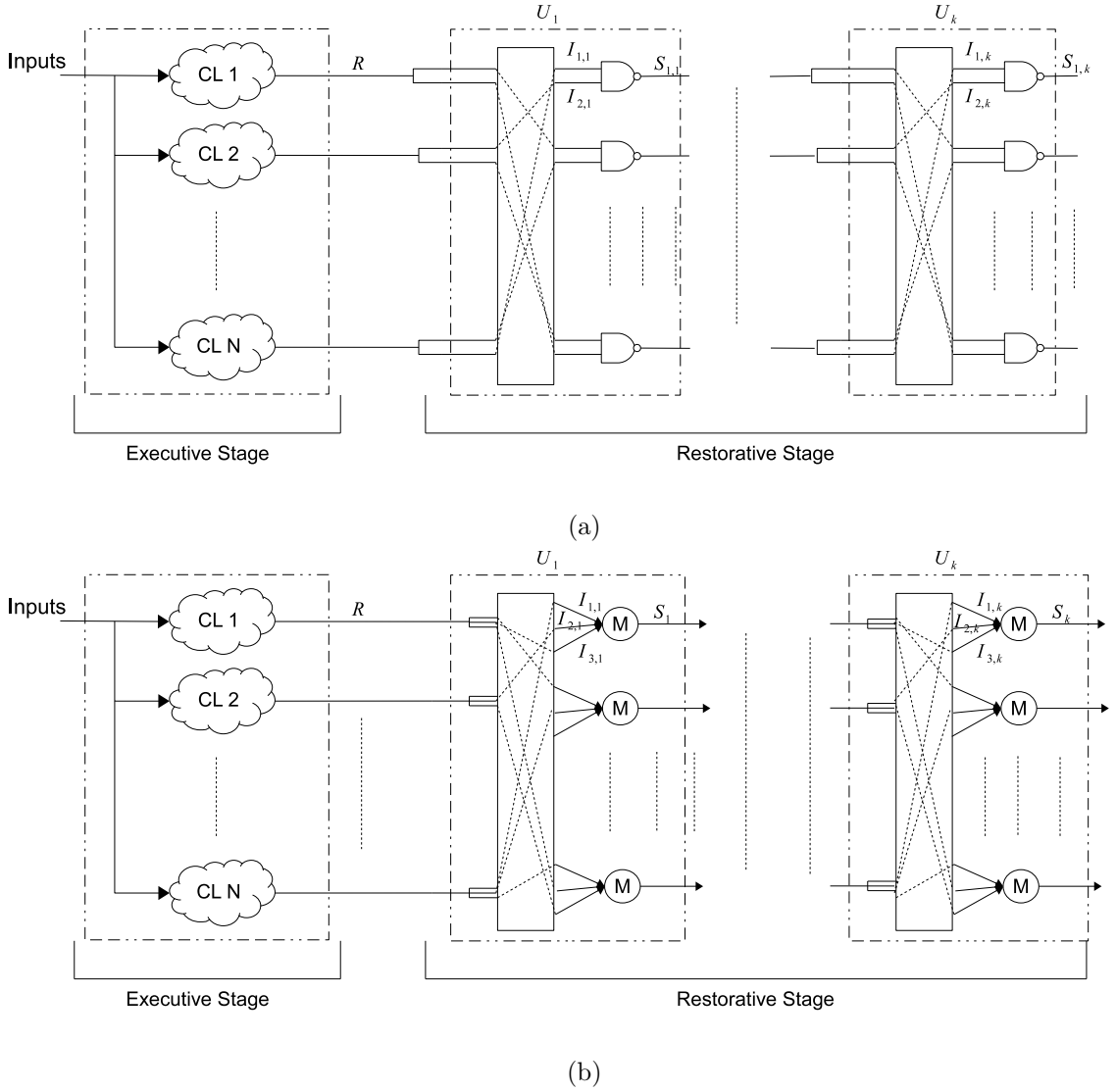


Figure 1.4: Two Von Neumann architectures: (a) NAND Mux (b) Maj Mux.

only if the failure probability of the basic element of the restorative stage is less than 0.16. Moreover, by increasing the number of blocks inside the restorative stage, the robustness of the multiplexing unit can be improved.

Fig. 1.5 shows the curves of $\eta_{NAND}^{(k)}$ and $\eta_{Maj}^{(k)}$ for $k = 1$, 2 and 4 when $\epsilon = 10^{-3}$. As we can see in this figure, $\eta_{NAND}^{(1)}$ is an anti-monotone[2] function since NAND transforms an information to its opposite (the NAND gate brings two inputs equal to '0' to an output equal to '1' and vice versa). When p is close to 0, in most case the two inputs are the same, the output will take the opposite value which brings $\eta_{Maj}^{(1)}$ near to 1. For this reason, the restorative operation needs to be iterated with a second block U_2 in the NAND Mux. The figure demonstrates also that the error probability decreases as the number of blocks inside

[2]An anti-monotone function f satisfies the property x $\leq$ y implies f(x) $\geq$ f(y),

the restorative stage increases for both the Maj and the NAND Mux (when $p \leq 0.5$). For example, when $p = 0.05$, $\eta^{(1)}_{Maj} = 0.0165$, $\eta^{(2)}_{Maj} = 0.0024$, $\eta^{(2)}_{NAND} = 0.0186$ and $\eta^{(4)}_{NAND} = 0.0046$.

This result of Von Neumann is a theorem that prove the existence of an arbitrarily reliable architecture under constraint of a minimum reliability of the NAND gate. It shows that the objective of the thesis of making reliable computation on an unreliable architecture is realistic. However, this theorem gives little indication on the hardware because the complexity of the Von Neumann architectures grows very quickly with the required level of reliability and its efficiency (number of operations per time unit and area unit) trends rapidly toward zero. This theorem is similar to the Shannon's second theorem that proves the basic existences of error-correcting codes without giving a practical method for constructing them.

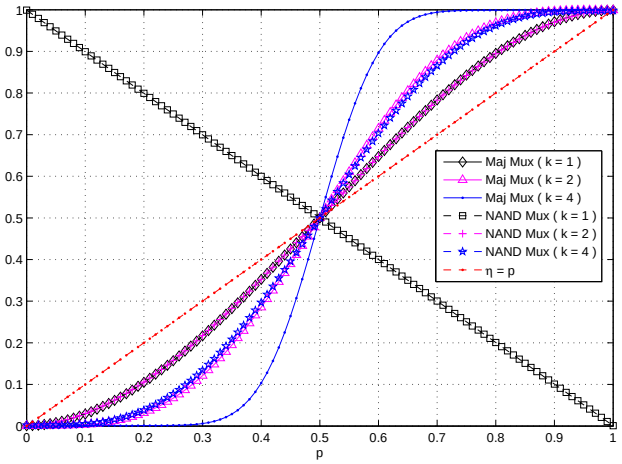


Figure 1.5: Maj Mux and NAND Mux performances.

1.3.2 Triple Modular redundancy

TMR is a classical solution developed in the 60's for fault tolerance in electronic systems [4]. Two versions of TMR exist: the Spatial TMR (S-TMR) and the Temporal TMR (T-TMR). In a standard S-TMR, the original logic module is replicate three times and error correction is achieved by a majority vote operation as shown in Fig. 1.6a. If a fault appears in any one of the three replicas module, the other two replica can correct and mask the fault. But once two of the three replicas fail, an uncorrectable failure results. Faults in the voter can cause the whole system to fail. To deal with these drawbacks, others alternative configurations were proposed, such as the triple-voter method or the restorative-feedback voter [30]. Fig. 1.6b illustrates the triple-voter configuration. All these configurations are costly in term of area and power consumption. As a consequence, the S-TMR is used in general for the robustness of extremely critical applications, such as space, avionics and healthy applications, where the system cost is less important than its reliability. In the T-TMR, the original logic module processes the same input three times during three successive clock cycles. Error correction

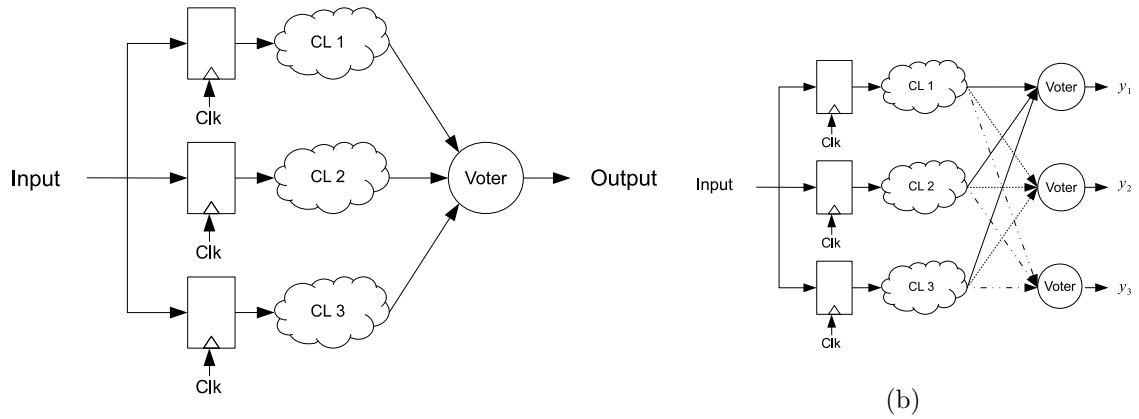


Figure 1.6: Two configurations for the TMR: (a) The simple TMR and (b) TMR with three voters.

is achieved by the majority vote between the outputs at the three clock cycles. The T-TMR doesn't tolerate permanent faults.

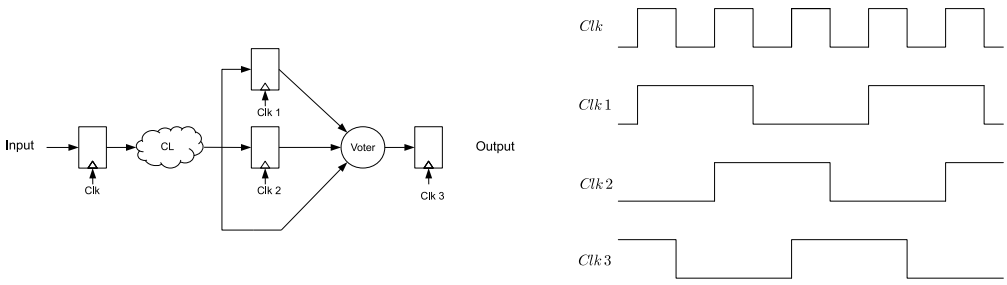


Figure 1.7: TMR temporel configuration.

1.3.3 Restorative Feedback method

The Restorative FeedBack (RFB) method is logically identical to the TMR method but it relies on the use of C-Muller elements instead of majority gate. The C-element is a well-known gate that has long been used in asynchronous circuit design [31] and was more recently recognised for its inherent fault-compensating abilities. A standard binary C-element circuit is shown in Fig. 1.8. The C-element is logically defined as a two-input latch, a, b and output c. It is composed of C-not gate and S gate. The C-not gate output at the time step t, Q^t, is defined as,

$$Q^t = \begin{cases} \overline{a}, & \text{if } a = b \\ Q^{t-1}, & \text{otherwise} \end{cases} \tag{1.1}$$

When $a \neq b$, the state of S latch is maintained via weak feedback.

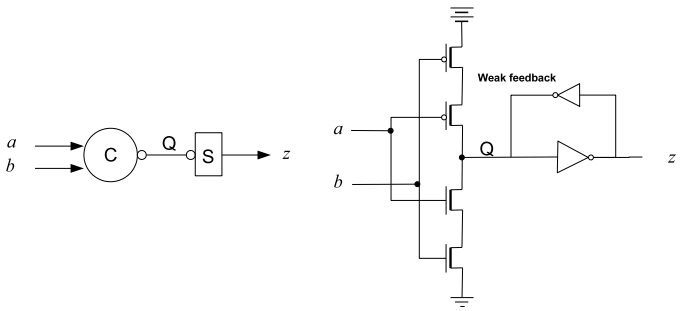


Figure 1.8: A standard C-element circuit; the C-not gate detects if the inputs are equal, the S gate acts as an inverting state memory [32].

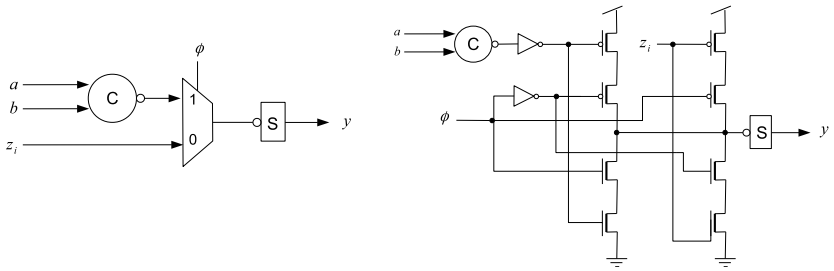


Figure 1.9: The modified C-element design; phase $\phi = 0$ active the setup process while $\phi = 1$ enable the Restoration process [32].

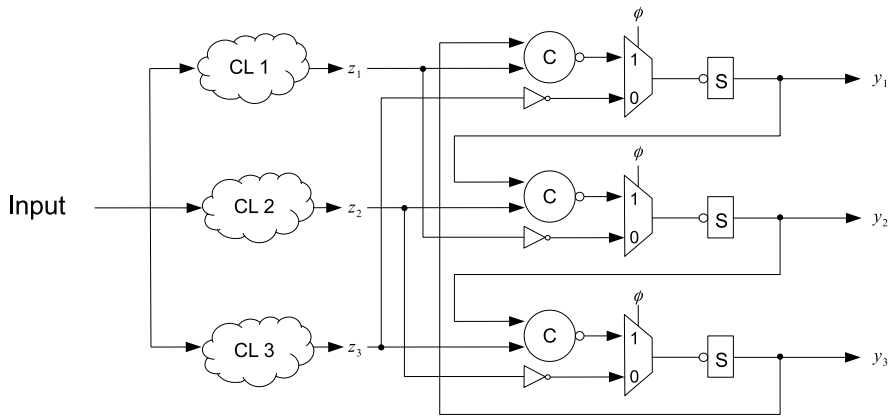


Figure 1.10: Implementation of the Retroactive FeedBack method.

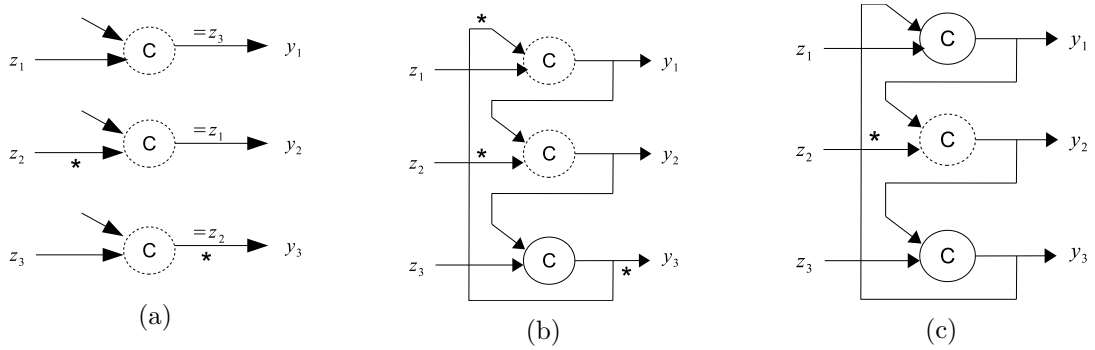


Figure 1.11: Two phases of the RFB method: the setup and the restoration phase. The dotted circles indicate C-elements which are not activated (feed-backs are not activated) while the star indicates the error location. (a) During the setup phase the outputs y_1, y_2 and y_3 are initialised with replica signals z_1, z_2 and z_3. (b) In the restoration phase, feed-backs are activating. (c) After a propagation delay, errors are eliminated once C-elements are activated [30].

A modified C-Muller element is given in Fig. 1.9. The modified C-element operates in two phases, called initialisation (set-up) and restoration. During the initialisation phase, the C-not gate is disconnected and the multiplexer force the state of S to be set to a known initial value, z_i. During the restoration phase, the C-element is activated.

In the RFB scheme, the original module is replicated three times; the three independent output signals, z_1, z_2 and z_3 are connected to three modified C-element as illustrate in fig. 1.10. Similar to the TMR, the RFB scheme is able to correct any single error that appears during the initialisation phase. For example, if z_2 is erroneous, the initialized value of the last C-element is incorrect. During the restoration phase, feed-backs are activated. Errors are corrected once the output signal y_2 and the input data z_3 remain correct as shown in fig. 1.11. The RFB method yields an error probability similar to TMR when errors occur during the set-up phase. However, it suppresses most errors that occur during the restoration phase.

1.3.4 Redundant Residue Number System

1.3.4.1 Introduction and fundamentals

The Residue Number System (RNS) was introduced by Garner in [33]. A standard RNS is defined by a set of k pairwise relative prime positive integers, m_1, m_2,..., m_{k-1}, m_k, i.e, greatest common divisor $(m_i,m_j)=1$ with $i \neq j$. m_1, m_2,..., m_{k-1}, m_k are called moduli. The order of moduli is in increasing, i.e, $m_1 < m_2 < ... < m_{k-1} < m_k$. Their product represents the interval $[0,M)$ called the legitimate range that defines the useful computational range of the number system, that is

$$M = \prod_{i=1}^{k} m_i. \tag{1.2}$$

Every natural integer X in the legitimate range can be represented by a set of residues, r_1, r_2, ..., r_{k-1}, r_k, where

$$X \equiv r_i \pmod{m_i}, \quad i.e.\ \exists\ \alpha\ \in\ \mathbb{N}\ / X = \alpha\ m_i + r_i;\ r_i < m_i. \tag{1.3}$$

It is frequently desirable to determine the natural number, X, associated with a particular residue representation m_i. The conversion technique is named the Chinese Remainder Theorem (CRT). According to the the CRT, X can be computed by

$$r_1 A_1 \frac{M}{m_1} + ... + r_k A_k \frac{M}{m_k} \equiv X \pmod{M}, \tag{1.4}$$

where

$$A_i \frac{M}{m_i} \equiv 1 \pmod{m_i}. \tag{1.5}$$

Example 1: Consider a residue number system with bases (m_1=2, m_2=3, m_3=5, m_4=7). The legitimate range is equal to 210. The input data 173 is represented by the residue vector (1,2,3,5). The conversion formula for the residue system with base (2,3,5,7) is obtained by

$$\begin{aligned}
&105\ A_1\ \equiv\ 1 \pmod 2 \quad so\ A_1 = 1,\\
&70\ A_2\ \equiv\ 1 \pmod 3 \quad so\ A_2 = 1,\\
&42\ A_3\ \equiv\ 1 \pmod 5\ \Rightarrow\ 2\ A_3\ \equiv\ 1 \pmod 5\ so\ A_3 = 3,\\
&30\ A_4\ \equiv\ 1 \pmod 7\ \Rightarrow\ 2\ A_4\ \equiv\ 1 \pmod 7\ so\ A_4 = 4.
\end{aligned}$$

The conversion formula can be used now to determine the natural number corresponding the residue vector (1,2,3,5) as follow

$$\begin{aligned}
&1\ \times\ 1\ \times\ 105\ +2\ \times\ 1\ \times\ 70\ +3\ \times\ 3\ \times\ 42\ +5\ \times\ 4\ \times\ 30\ =\ X \pmod{M,}\\
&1223\ \equiv\ X \pmod{210}\ \Rightarrow X =\ 173.
\end{aligned}$$

The great utility of RNS in the context of fault tolerance comes from the following theorem

$$(X\ *\ Y) \pmod{M}\ \equiv\ (\ (X(\text{mod } M))\ *\ (Y(\text{mod } M))\ )(\text{mod } M), \tag{1.6}$$

$$(x_1, x_2, ..., x_k)\ *\ (y_1, y_2, ..., y_k)\ =\ (z_1, z_2, ..., z_k)\ ,\ z_i \equiv x_i * y_i(\text{mod } m_i). \tag{1.7}$$

where * denotes the three operations: addition, subtraction, multiplication. By checking this relation, errors in the (X*Y) operation can be detected. The following section presents a RNS based system that permits the detection and correction of errors.

1.3.4.2 Bi-Directional Redundant Residue Number System (BRRNS)

The Bi-Directional Redundant Residue Number System (BRRNS) is an RNS based system. It is characterised by n pairwise relative prime positive moduli, m_1, m_2,..., m_{n-1},m_n, that are formed in increasing, i.e., $m_1 < m_2 < ... < m_{n-1} < m_n$ [32]. The first k moduli are called

the information moduli while the remaining $n-k$ moduli are redundancies. Similar to the definition of moduli, the residue vector is composed of the information residues r_1, r_2, ..., r_{k-1}, r_k and the redundant residues r_{k+1}, r_{k+2}, ..., r_{n-1}, r_n. BRRNS codes satisfies the following constraints

$$M = \prod_{i=1}^{k} m_i \approx M_R = \prod_{i=k+1}^{n} m_i, \tag{1.8}$$

$$m_n < m_1 \times m_2. \tag{1.9}$$

The legitimate range of the BRRNS is $Min(M, M_R)$ and any redundant modulus is smaller than the product of the two minimum redundant values as (1.9). Since the product of the redundant moduli is approximately equivalent to the product of the information moduli, any integer X belonging to the legitimate range can be restored through the information residue vector or from the redundant residue vector (r_{k+1}, r_{k+2}, ..., r_{n-1}, r_n). BRRNS apply the Mixed Radix Conversion (MRC) for the conversion of residue vector to integer in place of the CRT. The MRC is expressed by the following equation

$$X = \sum_{i=1}^{n} \alpha_i \prod_{j=1}^{i-1} m_j, \tag{1.10}$$

where $0 \leq \alpha_i < m_i$ and $\prod_{j=1}^{0} m_j = 1$. The digits α_i are called the mixed radix radix digit. They can be obtained from a computation iteration, that is

$$\begin{aligned}
\alpha_1 &= r_1, \\
\alpha_2 &= (r_2 - \alpha_1)\, m_{1,2}^{-1} \pmod{m_2}, \\
\alpha_3 &= ((r_3 - \alpha_1)\, m_{1,3}^{-1} - \alpha_2)\, m_{2.3}^{-1} \pmod{m_3}, \\
\alpha_4 &= (((r_4 - \alpha_1)\, m_{1,4}^{-1} - \alpha_2)\, m_{2,4}^{-1} - \alpha_3)\, m_{3,4}^{-1} \pmod{m_4}, \\
\alpha_5 &= ((((r_5 - \alpha_1) m_{1,5}^{-1} - \alpha_2)\, m_{2,5}^{-1} - \alpha_3)\, m_{3,5}^{-1} - \alpha_4)\, m_{4,5}^{-1} \pmod{m_4}...
\end{aligned} \tag{1.11}$$

These computations are performed in a serial way, thus, a pipe-lined architecture for the implementation of the mixed radix converter is mostly employed in digital filters [32].The BRRNS codes with $n-k$ redundant modulus detect errors if the condition $\alpha_j \neq 0$, $j \in [k+1,\mathbf{n}]$, is satisfied. The integer belonging to the legitimate range is restored through the redundant residue vector (if digits of the information residue vector are faulty) or from the information residue vector (if digits of the redundant residue vector are faulty).

Example 2: Let us consider the BRRNS composed of the moduli set (m_1=4, m_2=5,m_3=7, m_4=11, m_5=13). The legitimate range is [0,4 × 5 × 7)=[0,140), according to $Min(M,M_R)$. If the input data is X=125 is considered, then the corresponding residue values are X=(1, 0, 6, 4, 8). Assuming an error occurs in r_3, then the received BRRNS code word becomes (1, 0, 1, 4, 8). The first consistent checking involves r_1, r_2, r_3 and r_4, α_4=5≠0. The second consistent checking involves r_1, r_2, r_3 and r_5, α_5=7≠0. Hence, the corrupt digits are in the information residue vector. The integer is restored by r_4 and r_5, the redundant residue set.

1.3.5 Algorithmic Noise Tolerance

In the context of signal processing applications, the ANT technique proposes to add only one additional module to the original computation module. The additional replica provides a reduced precision estimation for the output of F and consumes less than the original one. By applying some criteria to make the decision, the final output is chosen between the output of the original module and the output of the replica. An ANT based system, the Reduced Precision Redundancy (RPR) is introduced in [5] and illustrated in Fig. 1.12a. It is composed of an error control block that calculates the Euclidean distance, d_e, between the original module output, z_m, and the RPR replica output, z_r, then decides which signals to feed. The final output can be described by

$$y = \begin{cases} z_m, & \text{if } d_e \leq z_{th} \\ z_r, & \text{if } otherwise \end{cases} \tag{1.12}$$

where the z_{th} is a specified threshold. Since the reduced precision replica consumes much less power than the original function, the ANT technique achieves reliable low-power digital signal processing. However, the occurrence of errors in the estimator leads the output to be faulty. To tackle this drawback, another ANT based approach, called Algorithmic Soft Error Tolerance (ASET) has been proposed in [34]. Instead of using a single RPR estimator,

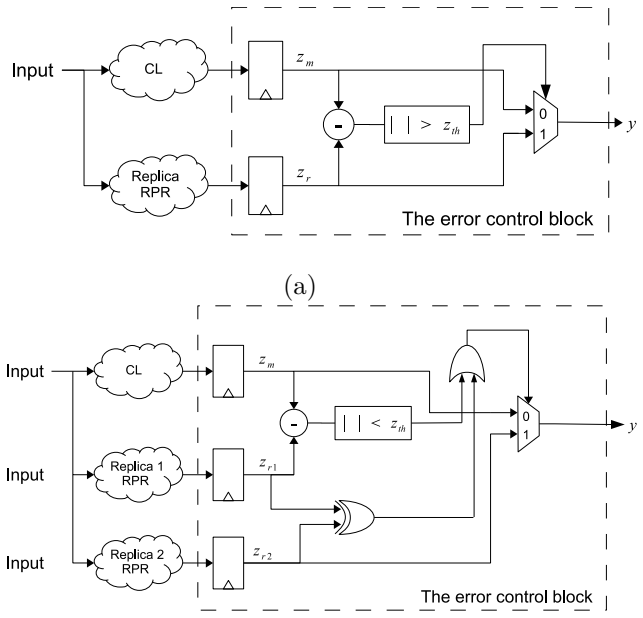


Figure 1.12: Two ANT based schemes: (a) RPR technique, (b) ASET technique.

ASET employs two RPR estimators to produce more reliable redundancy which help to make a more reliable decision after (see fig. 1.12b). Two differences are calculated: $d_e(z_m,z_{r1})$ and $d_h(z_{r1}, z_{r2})$. $d_e(z_m,z_{r1})$ is the Euclidean metric between z_m and z_{r1}, and $d_h(z_{r1}, z_{r2})$ is the

Hamming distance between z_{r1} and z_{r2}. The error correction block is summarised as follow:

$$y = \begin{cases} z_m, & \text{if } d_e(z_m, z_{r1}) \leq z_{th} \\ z_m, & \text{if } d_e(z_m, z_{r1}) > z_{th} \text{ and } d_h(z_{r1}, z_{r2}) \geq 1 \\ z_{r2}, & \text{if } d_e(z_m, z_{r1}) > z_{th} \text{ and } d_h(z_{r1}, z_{r2}) = 0 \end{cases} \tag{1.13}$$

Two drawbacks limit the based ANT method to be widely used. First, the reduced precision block should be constructed to provide the ability to detect and correct errors which it is feasible for signal processing application but not control application for example. Then, the ANT method requires that the decision block is reliable to perform a correct error-control.

1.3.6 Razor systems

The main concept of Razor systems is to take advantage of the temporal nature of timing and transient faults and achieve faults detection by observing the output of a given circuit at two instants. Three versions of Razor system exist: Razor I, Razor II and bubble Razor. The architecture of Razor I system is illustrate in Fig.1.13; each flip-flop in a design is augmented with a shadow latch controlled with a delayed clock. An error is flagged when both outputs disagree. A timing diagram of the Razor concept is given in Fig.1.14. By the rising edge of the clock in cycle 1, both the main flip-flop and the shadow latch will latch the correct data. The error signal at the output of the comparator remains low. Let suppose that a timing fault occur and the operation in cycle 2 exceeds the intended delay. Data is not latched correctly by the main flip-flop, in cycle 3, it is the shadow latch that will successfully latch the correct data. An error signal is flagged so and the recovery phase starts.

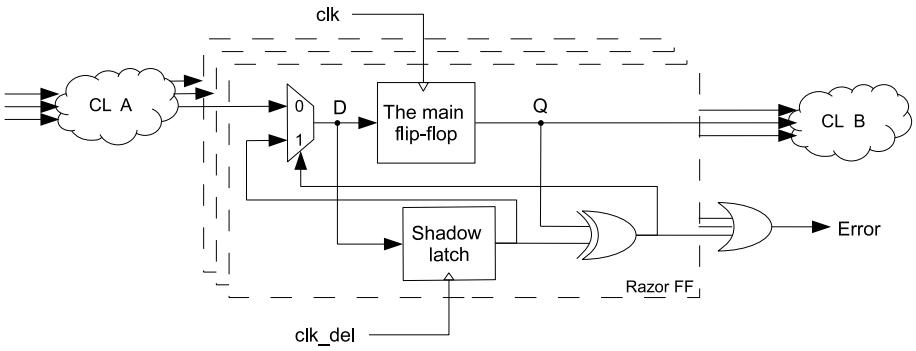


Figure 1.13: Razor I system architecture to protect logics against timing faults [35].

One of the proposed mechanisms for the recovering phase is the global clock gating whose architecture is shown in fig.1.15. It involves a stalling of the entire circuit for one clock cycle and reloading of main Razors with correct values stored in shadow latches. Any previously erroneous data value in main flip-flops is replaced with the correct value from the shadow latches. Thus any number of timing faults can be tolerated in a single cycle. Since this technique is based on communicating the erroneous state to the entire circuit in one cycle, it cannot be used for circuits with aggressive clock periods and for large performance designs.

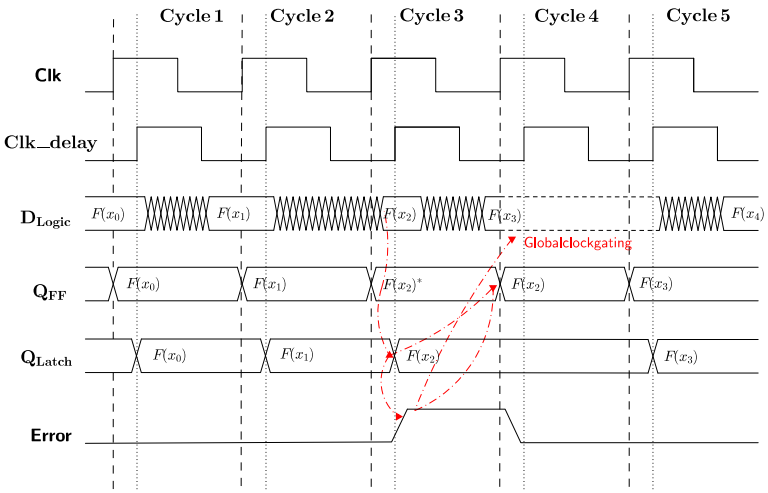


Figure 1.14: Example timing diagram of the Razor I system: A timing error occur in cycle 2 when the computation of $F(x_2)$ exceeds the intended delay, An error is flagged in cycle when the FF and latch outputs disagree.

Another approach was proposed to deal with this drawback, named the counter-flow pipelining. More details are given in [8].

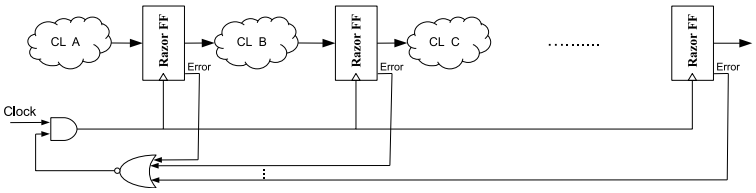


Figure 1.15: Recovery using the global clock gating.

Energy savings and tolerance against transient faults were considered when designing the Razor II. This new version performs a timing and transient fault detection by a so called "transition detector", and achieve recovery by a replay mechanism from a Razor validated check pointed state [36]. Using the negative edge of the clock to samples data in the shadow latch reduces also overhead of using a second delayed clock [8]. However, short path can be flagged as an error in this version of Razor. In fact, when data arrives after the positive edge, it must guarantee to be from a long path launched from the previous clock edge. Fig .1.16 presents the case where a short path is flagged as an error. To deal with this drawback, the authors of [37] have proposed a new version of Razors referred as Bubble Razors.

The bubble Razors use a two phase latch based data path instead of a flip-flop based data path by breaking the flip-flops into their master and slaves latches. Since two neighbouring latches are not opening at the same cycle, there is no possibility that two successive input data are latched at the same time. This enables large speculation window up to one half a clock cycle [37]. Once errors are detected, bubble signals are propagated to neighbouring latches, causing a latch to skip its next clock phase. To prevent bubbles from propagating indefinitely in loops, a latch that receives a bubble from one or more of its neighbours, stalls

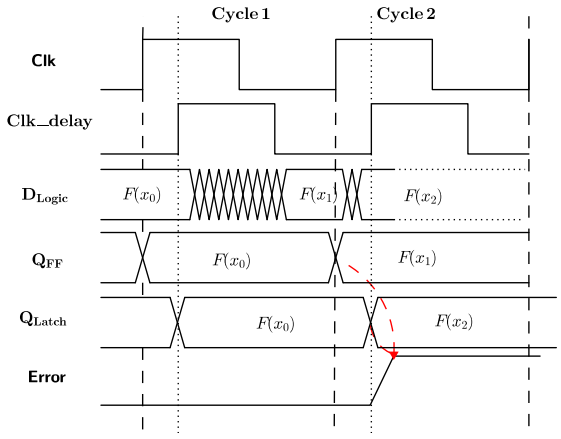


Figure 1.16: A timing diagram where a short path is flagged as an error in cycle 3.

and sends a bubble to the other neighbours (input and output) one half cycle later; A latch that receives a bubble from all of its neighbours doesn't send out any bubbles. This algorithm remains the same in the presence of multiples errors during the same cycle and for high error rates. Bubble Razor reduces the throughput penalty to 1 cycle. Fig.1.17 describes the bubble Razor algorithm when the occurrence of an error was detected in second latch.

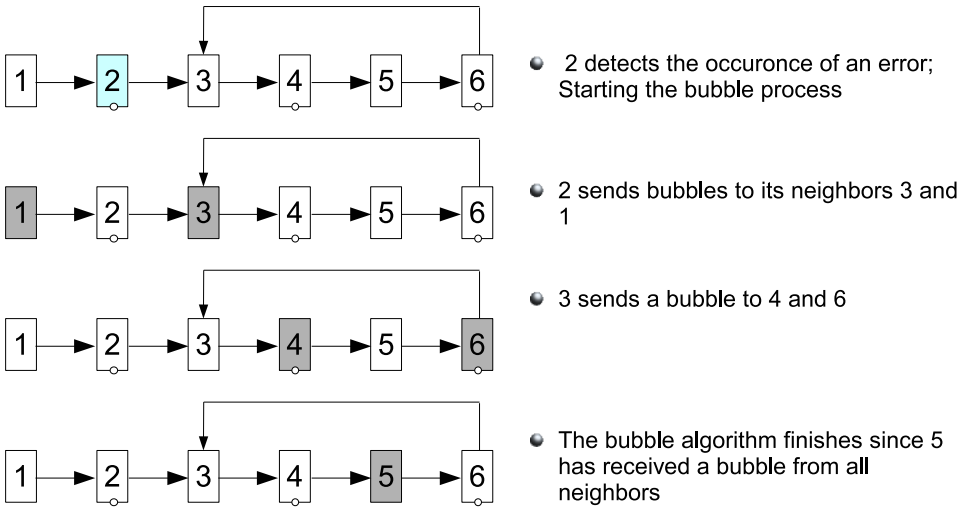


Figure 1.17: Description of the bubble Razor algorithm.

1.4 Hardware efficiency versus error probability

All state of the art techniques have in common that they requires extra hardware or/and throughput penalties (extra clock cycles needed for the recovery process). To compare architectures in unreliable hardware, the authors of [32, 38] define a novel metric called the Reliability Efficiency Criteria (RE-Criteria). The RE-Criteria consists on a couple (P_{Error},

γ) where P_{Error} is the error probability in the output of the architecture and γ represents the hardware efficiency of an architecture (defined as the normalised number of operation per unit area and time unit). To compute the hardware efficiency, the nature of the computation inside the area unit is not specified, it can be a simple multiplier or more complex operation like an Fast Fourier transform (FFT)... If we consider an operation Ψ that takes n_Ψ area units and m_Ψ area clocks to be executed, the efficiency is expressed as

$$\gamma_\Psi \triangleq \frac{1}{n_\Psi \times m_\Psi} \text{ operations/(area unit} \times \text{time unit)}. \tag{1.14}$$

The RE-Criteria has already been used in [32, 38] to compare the performances of BRRNS, S-TMR and Automatic Repeat reQuest (ARQ)[3] architectures against faults. In this section, we first review the RE-Criteria of S-TMR and T-TMR (for the rest of architectures, the reader is invited to [32] for more details). Then we will investigate the RE-Criteria of the Razor II Architecture to be added to the comparison.

1.4.1 The Reliability Efficiency Criteria of the S-TMR

Let P_Ψ the error probability at the output of the function Ψ during one clock cycle, P_v the error probability of a voter, and n_v the area cost of the voter (m_v is the number of clock cycle to perform the voter is assummed to be equal to 0). We remind that one of the hypotheses made in [32] is that two successive errors cannot lead to a correct result. It is a simplification hypothesis that gives an upper band of the output error probability. The output of the S-TMR is wrong when at most one replica is faulty and the voter is faulty or at least two of the replicas are faulty. Let P_1 the probability that at least two of the three replicas is correct. The resulting error probability of the S-TMR is expressed as

$$P_{S-TMR} = (1 - P_1) + P_1 P_v. \tag{1.15}$$

P_1 is expressed as,

$$P_1 = (1 - P_\Psi)^3 + 3\ P_\Psi\ (1 - P_\Psi)^2. \tag{1.16}$$

The efficiency of the S-TMR is determined by,

$$\gamma_{S-TMR} = \frac{1}{(3.n_\Psi + n_v).m_\Psi} \tag{1.17}$$

The normalised hardware efficiency is defined as,

$$\Gamma_{S-TMR} = \frac{\gamma_{TMR}}{\gamma_\Psi} = \frac{n_\Psi}{(3.n_\Psi + n_v)} \tag{1.18}$$

[3]The ARQ technique performs the error detection by adding hardware redundancy (modulo 4, modulo 8...). When an error occurs, the detector asks to restart the computation [39].

1.4.2 The Reliability Efficiency Criteria of the T-TMR

In the T-TMR, the computing function executes the same operation three clock cycles. The majority vote is performed between the outputs of the computing function at the three clock cycles. Registers are used to save outputs of the first and the second computations. Faults occur in these registers. Let P_f the probability that an error appear at the output of a register, R_1, R_2 and R_3 the probability of a correct input of the voter. $R_1 = (1 - P_R).(1 - P_f)^2$, $R_2 = (1 - P_R).(1 - P_f)$ and $R_3 = (1 - P_R)$. P_1 in this case is expressed as

$$P_1 = 1 - (R_1\ R_2\ R_3 +\ R_1\ R_2\ (1 - R_3) + R_2\ R_3\ (1 - R_1) + R_3\ R_1\ (1 - R_2)). \tag{1.19}$$

The resulting error probability of the T-TMR is,

$$P_{T-TMR} = (1 - P_1)(1 - P_v) + P_1 P_v. \tag{1.20}$$

The efficiency of the T-TMR is determined by,

$$\gamma_{S-TMR} = \frac{1}{(3.n_\Psi + n_v + 2.n_R).m_\Psi} \tag{1.21}$$

where n_R is the area cost of a register. Since the register area cost is low compared to the area of the function Ψ ($n_R \ll n_\Psi$), its error probability is much smaller than P_R. Hence, $PT - TMR$ is approximate to P_{S-TMR} as well as its hardware efficiency [32].

1.4.3 The Reliability Efficiency Criteria of the Razor II

The Razor II technique performs complete transient and timing error detection by the double sampling approach (see Sec. 1.3.6). When an error occurs, the computation is restarted. The average number of time unit (or clock cycles) for the Razor II to output a correct data is given by,

$$M_{RazorII} = \sum_{k=0}^{\infty}\ (k+1)\ (m_\Psi + 1)\ P_\Psi^k\ (1 - P_\Psi)\ = \frac{m_\Psi + 1}{1 - P_\Psi}. \tag{1.22}$$

The efficiency of the Razor is determined by,

$$\gamma_{Razor} = \frac{1 - P_\Psi}{n_\Psi \cdot (m_\Psi + 1)} \tag{1.23}$$

The error probability of the Razor II is expressed as,

$$P_{Razor} = \sum_{i=2}^{\infty} P_\Psi^i \cdots = \ \frac{P_\Psi^2}{1 - P_\Psi} \tag{1.24}$$

1.4.4 Comparison results

To compare different architectures, four pipeline Finite Impulse Response (FIR) filters are considered as in [32, 38]: FIR-I, FIR-II, FIR-III and FIR-IV. Each FIR filter is characterised by an order N and a input data bit length l. Table 1.1 summarises the number of area unit for each filter.

In the following comparison, we study the robustness of different architectures against only transient and timing faults since Razor and ANT don't deal with permanent faults. Let $\mathbf{p_e}$ be the probability that a transient or a timing fault occur in a unity area during one clock cycle. We assume that $\mathbf{p_e}$ is constant for all the design and independent of the underneath logic. As a result, the probability $P_R = 1-(1-p_e)^{n.m}$, $P_v = 1-(1-p_e)^{n_v}$. Fig. 1.18 shows the RE-Criteria of different architectures applied to the four FIR filters. From the figure, we can see that the BRRNS solution doesn't bring any interest since it degrades the reliability and add overhead in the design because of digit to residue and residue to digit process. Two architectures present a high level of reliability: the ARQ and Razor II. The Razor II is more reliable because of its complete capacity of detection of timing and transient faults while the ARQ is not able to correct some errors.

FIR Filters size (N,l)	FIR I (16,5)	FIR II (32,6)	FIR III (64,7)	FIR IV (128,8)
Simplex	94	128	171	232

Table 1.1: Logic synthesis results from XILINX Virtex V in term of Slices. (N, l) are (order of filter, bit length of input) [32, 38].

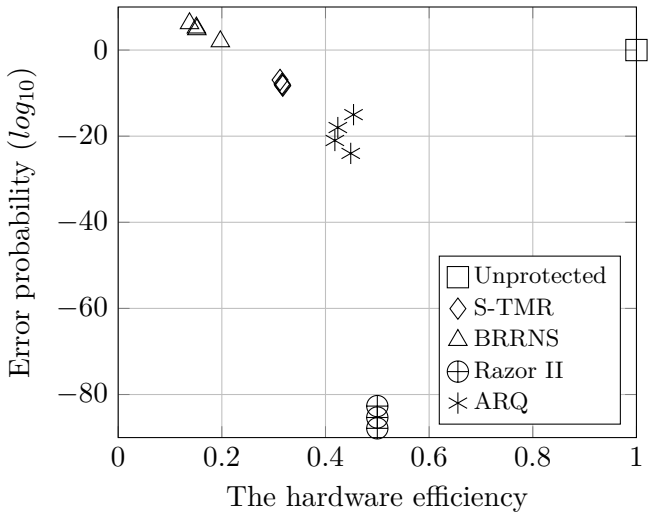


Figure 1.18: The RE-Criteria for S-TMR, BRRNS, Razor and ARQ architectures

1.5 Summary

Table 1.2 provides a comparison between the fault tolerant architectures discussed in previous sections and listed in the first row. Rows 2 and 3 details techniques used to respectively detect and correct errors in each case. The capability of each solution to detect different type of faults is presented in row 4. Finally, rows 5 and 6 gives briefly the components that the architecture involves (area overhead) and the number of clock cycles of delay need to mask errors (throughput penalties). If p and ϵ are the failure probability of the original module and the majority gate (or the NAND gate) respectively. Von Neumann demonstrated that it is possible to build a multiplexing unit that gives an error probability of the multiplexing unit lower than p only if ϵ is less than 0.16. By increasing the number of blocks inside the restorative stage the reliability of the complete unit can be improved.

The S-TMR and the RFB architectures tolerate both one faulty replica with more than 200% area overhead regardless the type of fault, permanent, timing or transient. The ANT architecture is an approximate logic circuit. It can detect occurrence of all types of faults, but, the recovered data can be erroneous once the reduced precision replica is faulty. The Razor offer a complete timing and transient error detection without adding much hardware to the design, but it reduces the throughput penalties since it uses extra clock cycles for the recovery process to eliminate errors.

To conclude this chapter, various fault tolerant architectures exist in the literature to deal with occurrence of faults. However, a few of them can tolerate all type of faults. Only Von-Neumann and S-TMR architecture can tolerate permanent, timing and transient faults. The drawback of theses architecture is that they are very costly in term of area. The following chapter introduce two new fault tolerant architectures: the Fault Tolerant Error Code Correcting based architecture (FT-ECC) and the Duplication with Syndrome based Corrector architecture (DSC). Performances results show that these architectures can reduce the complexity of the TMR to 30 % without a throughput penalty, when limited degradation of the robustness capacity can be tolerated.

		Von Neumann Architectures	TMR Spatial	RFB	ANT	Razor Systems	TMR Temporel	FT-ECC	DSC
Fault Detection		Hardware redundancy	Hardware redundancy	Hardware redundancy	approximation logic prediction	Double Sampling	Temporal Redundancy	Error code correcting (ECC)	Duplication
Fault Recovery		NAND or majority gates	Majority Vote	C-element Activation	use RPR output	Re computation	Majority Vote	ECC	Roll back
Type	Permanent	✓	✓	✓				✓	✓
	Transient	✓	✓	✓	✓	✓	✓	✓	✓
	Timing	✓	✓	✓	✓	✓	✓	✓	✓
Area cost		CL×N, k blocks of N NAND (or majority)	CL×3, FF×3, Voter	CL×3, C-element×3	CL, RPR, Comparator	CL, FF×2, Comparator	CL, FF×3, Voter	CL×1.75, Decoder	CL×2, Syndrome corrector
Throughput penalties		0 clock cycle	0 clock cycle	0 clock cycle	0 clock cycle	1 or multiples clock cycles	3 clock cycles	0 clock cycle	0 clock cycle

Table 1.2: Comparison of different fault tolerant architectures;Architectures in grey are described in the following chapter

Chapter 2

FT-ECC and DSC: New fault tolerant schemes

Contents

In the previous chapter, several fault tolerant architectures have been introduced and compared. Between these various architectures, only the Von Neumann multiplexing and the spatial TMR scheme can tolerate all type of faults. However, these architectures are very costly in term of area and power consumption. The objective of the thesis research is to address the reliability in signal processing systems and to develop new fault tolerant techniques to perform signal processing computation on unreliable hardware where all type faults can occur. This chapter presents two new resilient schemes to improve the reliability of signal processing applications: the Fault Tolerant Error Code Correcting based scheme (FT-ECC) and the Duplication with Syndrome based Corrector scheme (DSC).

We start the chapter by defining minimum hypothesis on targeted signal processing applications required to apply FT-ECC and DSC in Sec. 2.1. Then, FT-ECC will be described in Sec. 2.2 while Sec. 2.3 introduces the DSC scheme. Finally, Sec. 2.4 evaluates the performance and the complexity of each scheme and compares them to the classical TMR.

2.1 Definitions

Definition 1: Let $\mathbb{P}$ a set with an operation (+) that combines any two elements of $\mathbb{P}$, x_1 and x_2, to form another element of $\mathbb{P}$, denoted x_1+x_2. ($\mathbb{P}$,+) is a group if the four following properties are satisfied:

- **Closure**: For all x_1, x_2 in $\mathbb{P}$, the result of the operation, $x_1 + x_2$, is also in $\mathbb{P}$,
- **Associativity**: For all x_1, x_2 and x_3 in $\mathbb{P}$, $(x_1 + x_2) + x_3 = x_1 + (x_2 + x_3)$,
- **Identity element**: There exists an element $0_{\mathbb{P}}$ in $\mathbb{P}$ such that, for every element x_1 in $\mathbb{P}$, $0_{\mathbb{P}} + x_1 = x_1 + 0_{\mathbb{P}} = x_1$. Such an element is unique
- **Inverse element**: For each x_1 in $\mathbb{P}$, there exists an element x_2 in $\mathbb{P}$, commonly denoted -x_1, such that $x_1 + x_2 = x_2 + x_1 = 0_{\mathbb{P}}$, where $0_{\mathbb{P}}$ is the identity element.

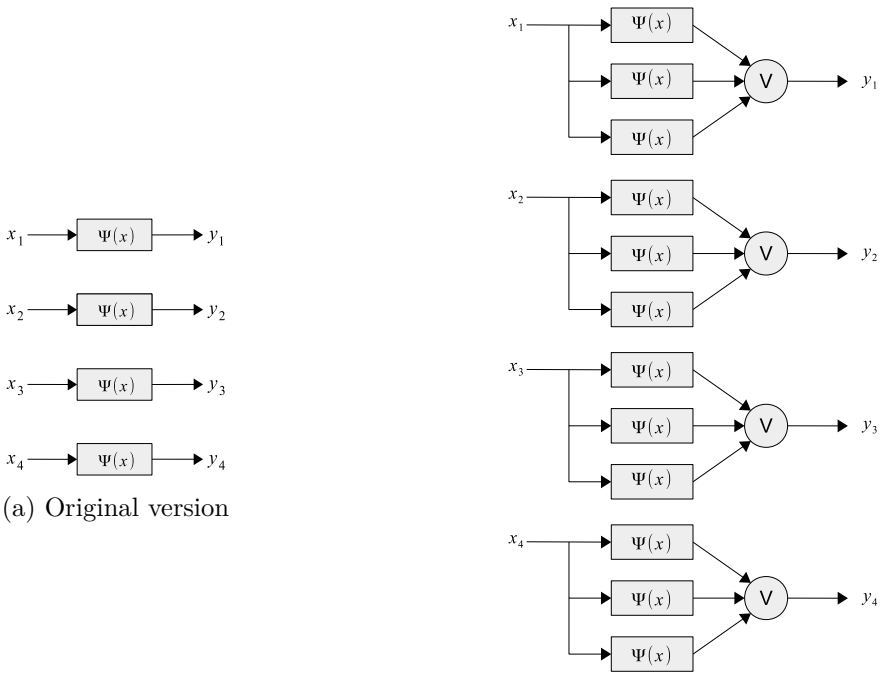
Definition 2: The norm in a group ($\mathbb{P}$,+), often denoted $\| \cdot \|_{\mathbb{P}}$, is a distance metric: $\mathbb{P} \rightarrow \mathbb{R}$ with the following conditions:

- $\forall x \in \mathbb{P}$, $\|x\|_{\mathbb{P}} \geq 0_{\mathbb{R}}$,
- $\|x\|_{\mathbb{P}} = 0_{\mathbb{R}} \Leftrightarrow$ x $= 0_{\mathbb{P}}$
- $\|x + y\|_{\mathbb{P}} \leq \|x\|_{\mathbb{P}} + \|y\|_{\mathbb{P}}$

Definition 3: Let consider two groups ($\mathbb{P}$,+) and ($\mathbb{Q}$,+). We define a group homomorphism Ψ: ($\mathbb{P}$,+) $\longrightarrow$ ($\mathbb{Q}$,+) where for all x_1 and x_2 in $\mathbb{P}$, Ψ $(x_1 + x_2)$= Ψ (x_1) + Ψ (x_2). There is many examples of group homomorphisms in the context of signal processing application, for example multiplication with a fix value (Ψ(x) = $\alpha \cdot$ x, x $\in \mathbb{Z}$, $\mathbb{R}$, $\mathbb{C}$), a multiplication of a polynomial (Ψ(P[x]) = P[x] $\cdot$ Q[x]), filtering and convolution ($\Psi(x) = (h \circledast s)(x), s(t) \in \mathbb{R}$), matrix product, Fast Fourier transform (FFT)... Those operations can be performed in any set (real, integer modulo $\mathbb{P}$, Galois Field...).

Hypothesise 1: We assume in this work that inside a same design, many operations and functions exists in replica; we can find three or more multiplications processing on different inputs in a given design, three or more FFT...

Let us consider four identical group homomorphisms, Ψ, processing in parallel on four independent inputs, denoted x_1, x_2, x_3 and x_4 to generate $y_1 = \Psi(x_1)$, $y_2 = \Psi(x_2)$, $y_3 = \Psi(x_3)$ and $y_4 = \Psi(x_4)$ (see Fig.2.1a). Variables $x_1 \ldots x_4$ and $y_1 \ldots y_4$ are elements of $\mathbb{P}$ and $\mathbb{Q}$ respectively. Researchers previously proposed to triplicate each operation independently and add a vote majority to mask errors, as shown in Fig 2.1b. With this method, the design that contains four group homomorphisms at the beginning will contain twelve functions Ψ and four voters. In these following sections, two new resilient schemes less complex in terms of area will be introduced: the FT-ECC and the DSC schemes.



(a) Original version

(b) Protection using TMR

Figure 2.1: Reliability improvement using TMR

2.2 Fault Tolerant based Error Code Correcting scheme (FT-ECC)

Researchers previously considered linear error correcting codes to protect dynamic systems against faults such as finite state controllers, RAMs, wireless communications... For example, Hadjicostis and Verghese considered the linearity properties of the Linear finite-state machines (LFSM) to jointly encoded them using LDPC codes, and protect against transient faults [40]. This section introduces a similar based scheme but adapted to signal processing operations. The new scheme refers as the Fault Tolerant based Error Code Correcting scheme (FT-ECC). To introduce the concept of the FT-ECC, we take an example of an error code correcting which is the (4,7) Hamming code. The corresponding architecture is named the FT-Hamming.

2.2.1 The FT-Hamming scheme

Three redundant group homomorphisms Ψ are added to the four original group homomorphisms. Fig. 2.2 illustrates the FT-Hamming scheme. Inputs of the seven homomorphisms, $c_1 \ldots c_7$, are generated using a Hamming encoding operation

$$(c_1, c_2, c_3, c_4, c_5, c_6, c_7) = (x_1, x_2, x_3, x_4) \times G, \tag{2.1}$$

where G is the generator matrix G of the (7,4) Hamming code defined as

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Note that, since all coefficients of the matrix are equal to 1, the vector matrix multiplication implies only additions in $\mathbb{P}$. Thus, $c_1 \ldots c_7$ are elements of $\mathbb{P}$. Let H be the parity check matrix corresponding to G. $H \times G^T = 0_{\mathbb{P}}$ and is defined as,

$$H = \begin{pmatrix} -1 & 0 & -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & -1 & -1 & 0 & 0 & 1 \end{pmatrix}.$$

The seven group homomorphisms process their different inputs, $c_1, \ldots, c_7$ independently, to

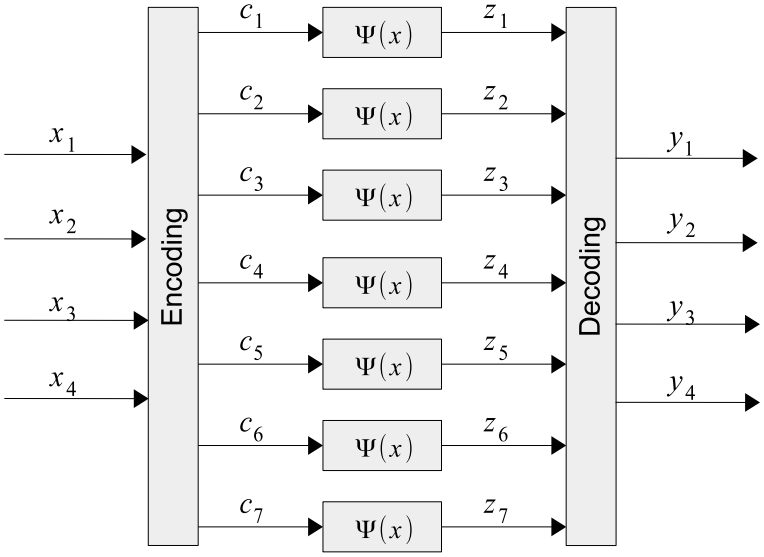


Figure 2.2: Introduction to the FT-Hamming solution

generate seven different outputs $(z_1, z_2, \ldots, z_7) \in \mathbb{Q}$, as

$$z_1 = \Psi(c_1) = \Psi(x_1), \tag{2.2}$$
$$z_2 = \Psi(c_2) = \Psi(x_2), \tag{2.3}$$
$$z_3 = \Psi(c_3) = \Psi(x_3), \tag{2.4}$$
$$z_4 = \Psi(c_4) = \Psi(x_4), \tag{2.5}$$
$$z_5 = \Psi(c_5) = \Psi(x_1 + x_3 + x_4), \tag{2.6}$$
$$z_6 = \Psi(c_6) = \Psi(x_1 + x_2 + x_3), \tag{2.7}$$
$$z_7 = \Psi(c_7) = \Psi(x_2 + x_3 + x_4). \tag{2.8}$$

According to the group homomorphism structure of the function Ψ, in case of error free computation, the three following equations are all verified

$$\begin{cases} z_5 = \Psi(x_1 + x_3 + x_4) = \Psi(x_1) + \Psi(x_3) + \Psi(x_4) = z_1 + z_3 + z_4, \\ z_6 = \Psi(x_1 + x_2 + x_3) = \Psi(x_1) + \Psi(x_2) + \Psi(x_3) = z_1 + z_2 + z_3, \\ z_7 = \Psi(x_2 + x_3 + x_4) = \Psi(x_2) + \Psi(x_3) + \Psi(x_4) = z_2 + z_3 + z_4. \end{cases} \tag{2.9}$$

$$\Leftrightarrow \begin{cases} z_5 - (z_1 + z_3 + z_4) = 0_{\mathbb{Q}}, \\ z_6 - (z_1 + z_2 + z_3) = 0_{\mathbb{Q}}, \\ z_7 - (z_2 + z_3 + z_4) = 0_{\mathbb{Q}}. \end{cases} \quad \Leftrightarrow \quad D \triangleq Z \times H^T = (0_{\mathbb{Q}}, 0_{\mathbb{Q}}, 0_{\mathbb{Q}}). \tag{2.10}$$

Note that, in case of computation with rounding and/or saturation noise, the strict equality can be relaxed to a distance compatible with the noise computation (z_{Th}). z_{Th} will depend on the maximum magnitude of error that an application can tolerate to continue delivering a given quality of service.

$$\begin{cases} \| z_5 - (z_1 + z_3 + z_4) \|_{\mathbb{Q}} \leq z_{Th}, \\ \| z_6 - (z_1 + z_2 + z_3) \|_{\mathbb{Q}} \leq z_{Th}, \\ \| z_7 - (z_2 + z_3 + z_4) \|_{\mathbb{Q}} \leq z_{Th}. \end{cases} \tag{2.11}$$

Those three equality/inequality can be represented by a triplet of Boolean values, or syndromes, $S = (s_1 s_2 s_3)$, with s_1 (respectively s_2 and s_3) equals to 0 if the first element of D (respectively second and third element) is equal to 0_Q, 1 otherwise.

Case 1: No faults occurs in the scheme: In case of no error, $D = (0_{\mathbb{Q}},0_{\mathbb{Q}},0_{\mathbb{Q}})$, so, the syndromes are all equal to zero. The values of z_1, z_2, z_3 and z_4 are simply copied in the outputs y_1, y_2, y_3 and y_4.

Case 2: One fault occur: If a single fault occurs in one of the seven modules then the value of the syndromes allows to detect the error and correct it. Let us consider, for example, that an error occurs in the computation of $\Psi(x_1)$ and that the first operator outputs $\tilde{z_1} = z_1 + e$, with $e \neq 0$. According to (2.10), replacing z_1 by the erroneous value $\tilde{z_1}$, it will generate the violation of the equality of the first and second equation. D = (-e,-e,0) $\neq (0_Q,0_Q,0_Q)$. This leads the 3 syndromes $(s_1 s_2 s_3)$ to be equal to $(s_1 s_2 s_3)$ = (110). In this case, it is still possible to recover the exact value y_1 by processing $y_1 = z_5 - (z_3 + z_4) = \Psi(x_1 + x_3 + x_4) - \Psi(x_3) - \Psi(x_4) = \Psi(x_1)$. This example can be generalised for any single error, as shown in Table 2.1. Fig. 2.3 provides an overview of the hardware required for the FT-Hamming scheme.

Case 3: More than one fault occurs Finally, if more than one error occurs, in most of the case, either the 3 equations of (2.10) won't be fulfilled (s=(111)) and the system will behave as if z_3 was faulty (for example z_1 and z_2 are faulty), or, the syndrome computed corresponds to one the case in Table 2.1 and the system behaves as if a

Faulty variable	Syndrome	y_1	y_2	y_3	y_4
$\tilde{z}_1$	(110)	$z_5-(z_3+z_4)$	z_2	z_3	z_4
$\tilde{z}_2$	(011)	z_1	$z_7-(z_3+z_4)$	z_3	z_4
$\tilde{z}_3$	(111)	z_1	z_2	$z_6-(z_1+z_2)$	z_4
$\tilde{z}_4$	(101)	z_1	z_2	z_3	$z_7-(z_3+z_2)$
$\tilde{z}_5$	(100)	z_1	z_2	z_3	z_4
$\tilde{z}_6$	(010)	z_1	z_2	z_3	z_4
$\tilde{z}_7$	(001)	z_1	z_2	z_3	z_4

Table 2.1: The syndrome corresponding to each faulty module for the FT-Hamming method

single fault occurs which mismatch the reality (for example if z_5 and z_6 are faulty, the syndrome will be equal to S=(110) which corresponds to the first row of Table 2.1).

To conclude this section, any single error in the FT-Hamming scheme is detected and corrected. However, if more than one fault occurs, the system can misinterpret the situation and add new faults at the outputs.

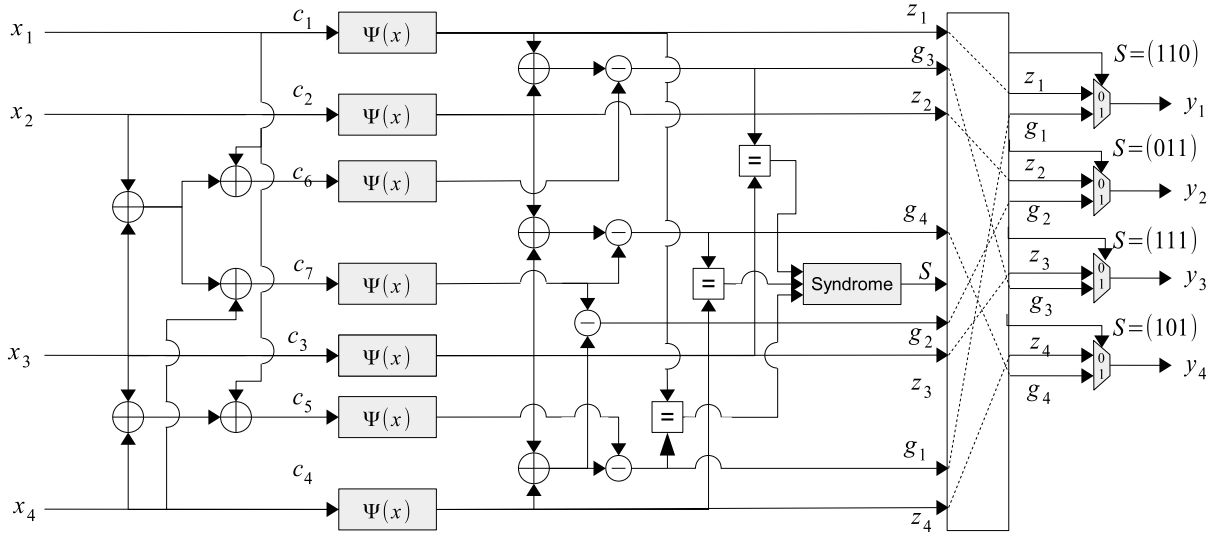


Figure 2.3: Architecture of the FT-Hamming scheme

2.2.2 Application of the ANT in the FT-Hamming scheme

The ANT technique presented in Section. 1.3.5 can be adapted to the FT-Hamming. Instead of doing computation of z_5, z_6 and z_7 in full precision, it is possible to do them in a reduced precision to save area and power dissipation as shown in Fig. 2.4. The drawback is that, in case of error occurring in z_1 for example, the reconstructed value $y_1 = z_5-(z_3+z_4)$ will have a degraded precision due to the reduced precision of z_5. For some digital signal processing applications, there is no need to have outputs with full precision, protection of most significant bits of the outputs of some operations is more important.

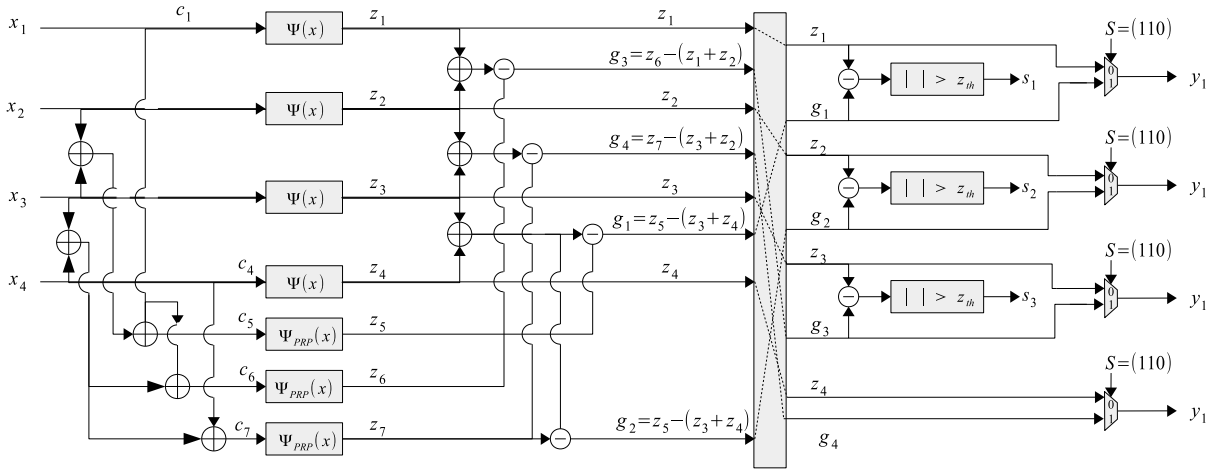


Figure 2.4: ANT applied in the FT-Hamming scheme

2.2.3 Generalisation: FT-ECC scheme

In the general case, M redundant homomorphisms Ψ are added to a design that contains N group homomorphisms Ψ as shown in Fig. 2.5. Inputs are generated using the encoding operation,

$$(c_1, c_2, \ldots, c_{N+M}) = (x_1, x_2, \ldots, x_N) \times G, \tag{2.12}$$

where G is the N by M generator matrix of a linear error correcting code. The corresponding parity matrix H (where $H \times G^T = 0$) serves to decode outputs of the $(N + M)$ functions, detect errors and correct them.

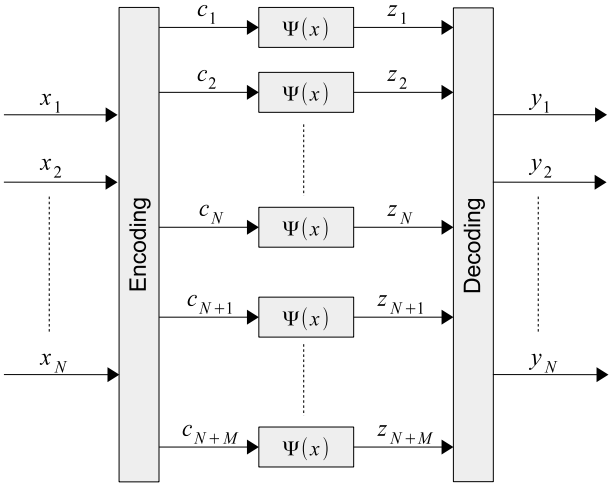
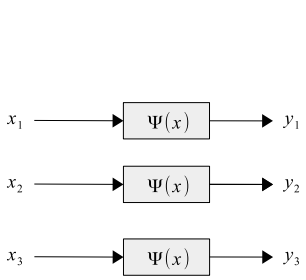


Figure 2.5: Illustration of the FT-ECC solution

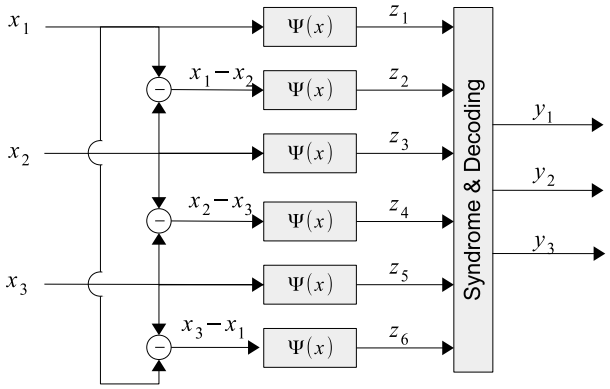
2.3 Principle of the Duplication with Syndrome based Correction (DSC)

2.3.1 The Three Duplication with Syndrome based Correction scheme (3-DSC)

Let's start from an original design that contains only 3 group homomorphisms Ψ processing in parallel on three independent inputs, x_1, x_2, x_3, to generate $y_1 = \Psi\ (x_1)$, $y_2 = \Psi(x_2)$ and $y_3 = \Psi(x_3)$ as shown in Fig. 2.6a. The 3-DSC scheme adds three redundant functions Ψ and a syndrome based corrector that performs the task of detection and recovery of errors (see Fig. 2.6b).



(a) Original version



(b) Protection 3-DSC

Figure 2.6: Introduction to the 3-DSC concept

The three redundant group homomorphisms process three different inputs to generate three independent outputs z_4, z_5 and z_6 as follow,

$$\begin{cases} z_4 = \Psi(x_1 - x_2), \\ z_5 = \Psi(x_2 - x_3), \\ z_5 = \Psi(x_2 - x_3). \end{cases} \tag{2.13}$$

According to the group homomorphism structure of the function Ψ, in case of error free computation, the three following equations are all verified,

$$\begin{cases} z_4 = \Psi(x_1 - x_2) = \Psi(x_1) - \Psi(x_2) = z_1 - z_2, \\ z_5 = \Psi(x_2 - x_3) = \Psi(x_2) - \Psi(x_3) = z_2 - z_3, \\ z_6 = \Psi(x_3 - x_1) = \Psi(x_3) - \Psi(x_1) = z_3 - z_1. \end{cases} \tag{2.14}$$

Note that, in case of computation with rounding and/or saturation noise, the strict equality

Faulty variable	Syndrome	y_1	y_2	y_3
$\tilde{z}_1$	(101)	$z_4 + z_2$	z_2	z_3
$\tilde{z}_2$	(110)	z_1	$z_5 + z_3$	z_3
$\tilde{z}_3$	(011)	z_1	z_2	$z_6 + z_1$
$\tilde{z}_4$	(100)	z_1	z_2	z_3
$\tilde{z}_5$	(010)	z_1	z_2	z_3
$\tilde{z}_6$	(001)	z_1	z_2	z_3

Table 2.2: The syndrome corresponding to each faulty module

can be relaxed to a distance compatible with the noise computation (z_{Th}).

$$\begin{cases} \| z_4 - (z_1 - z_2) \|_{\mathbb{Q}} \leq z_{Th}, \\ \| z_5 - (z_2 - z_3) \|_{\mathbb{Q}} \leq z_{Th}, \\ \| z_6 - (z_3 - z_1) \|_{\mathbb{Q}} \leq z_{Th}. \end{cases} \tag{2.15}$$

Those three equality/inequality can be represented by a triplet of Boolean values, or syndromes, ($s_1 s_2 s_3$), with s_1 (respectively s_2 and s_3) equals to 1 if the first equation (respectively second and third equation) is not fulfilled. In a more robust version, it is also possible to further consider z_7, the parity check sum, defined as

$$z_7 \triangleq z_4 + z_5 + z_6 = (z_1 - z_2) + (z_2 - z_3) + (z_3 - z_1) = 0. \tag{2.16}$$

Case 1: No faults occurs in the scheme: In case of no error, the syndromes and z_7 are all equal to zero. The values of z_1, z_2 and z_3 are simply copied in the outputs y_1, y_2 and y_3.

Case 2: One fault occur: If a single fault occurs in one of the six modules then the value of the syndromes allows to detect the error and correct it. Let us consider, for example, that an error occurs in the computation of $\Psi(x_1)$ and that the first operator outputs $\tilde{z_1}$ $= z_1 + e$, with $e \neq 0$. According to (2.14), replacing z_1 by the erroneous value $\tilde{z_1}$, it will generate the violation of the equality of the first and last equation of (2.14). This leads the 3 syndromes ($s_1 s_2 s_3$) to be equal to ($s_1 s_2 s_3$) = (101). In this case, it is still possible to recover the exact value y_1 by processing $y_1 = z_4 + z_2 = \Psi(x_1 - x_2) + \Psi(x_2) = \Psi(x_1)$. This example can be generalised for any single error, as shown in Table 2.2. The hardware required for the 3-DSC is illustrated in Fig. 2.7.

Case 3: More than one fault occurs: Finally, if more than one error occurs, two situations exist: 1/ the 3 equations of (2.14) won't be fulfilled. So, ($s_1 s_2 s_3$) = (111). 2/ the computed syndromes correspond to one of the first three rows in Table 2.2. This second situation happens when outputs of one of the combinations (z_1,z_4), (z_1,z_6), (z_2,z_4), (z_2,z_5), (z_3,z_5),(z_3,z_6), (z_1,z_4,z_6), (z_2,z_4,z_4) and (z_3,z_5,z_6) are faulty; In this case, z_7 = 0 won't be fulfilled (see (2.16)). In both situations, an error is flagged and a roll-back error recovery mechanism or other higher level mechanism may be activated. Nevertheless, it should be note that two errors event of same amplitude may not be detected. For example, if the function Ψ is applied on integer and both z_1 and z_2 are affected by

an additive noise of same amplitude e, i.e. $\tilde{z}_1 = z_1 + e$ and $\tilde{z}_2 = z_2 + e$, then equation $z_4 = \tilde{z}_1 - \tilde{z}_2$ will remain correct. The syndrome is thus (011), which implies the erroneous correction of z_3. Note that, in case of this double event of same amplitude, TMR system will also output a wrong value.

To conclude this section, as for FT-Hamming, we should mention that the ANT technique presented in Section. 1.3.5 can be adapted to 3-DSC. Instead of doing computation of z_4, z_5 and z_6 in full precision, it is possible to do them in a reduced precision to save area and power dissipation. The drawback of this technique is that, in case of error occurring in z_1 for example, the reconstructed value $y_1 = z_4 + z_2$ will have a degraded precision due to the reduced precision of z_4.

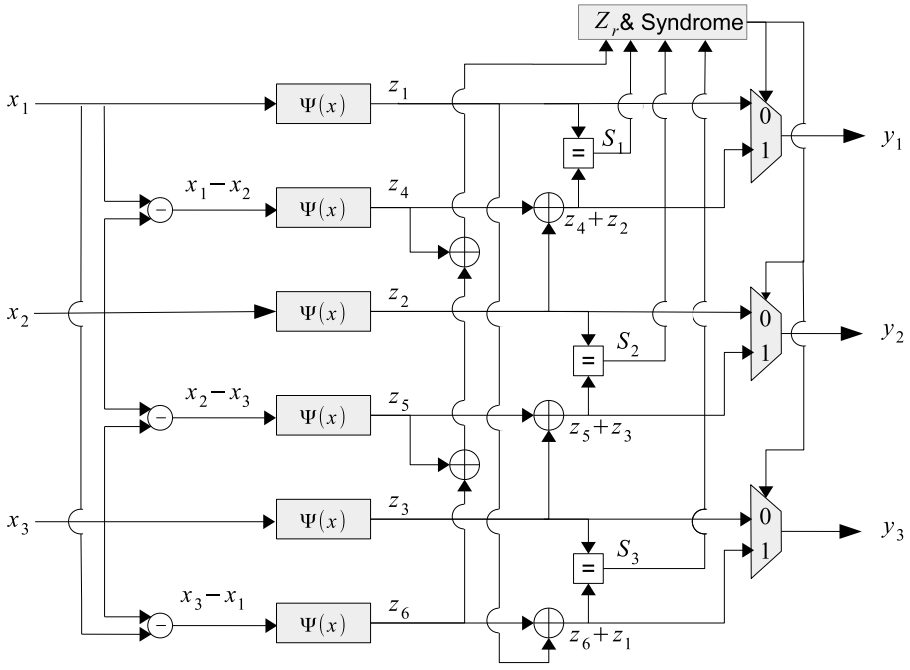


Figure 2.7: Protection using 3-DSC

2.3.2 Generalisation: N-DSC scheme

In this section, a resilient scheme for a design with N group homomorphisms is proposed as an extension of the 3-DSC ($N > 3$). The extension is straightforward: to the N functions $z_q = \Psi(x_q)$, $q = 1 \dots N$, N additional function are added to generate,

$$\begin{cases} z_{N+q} = \Psi(x_q - x_{q+1}), \ \forall \ q = 1 \dots N-1 \\ z_{2N=\Psi(x_N)-\Psi(x_1)} \end{cases} \tag{2.17}$$

Similarly to the 3-DSC case, in case of no error, the N following equation are fulfilled,

$$\begin{cases} z_{N+q} = z_q - z_{q+1}, \ \forall \ q = 1 \dots N-1 \\ z_{2N} = z_N - z_1 \end{cases} \tag{2.18}$$

N syndrome s_q, $q = 1 \dots N$ can be defined, with $s_q = 0$ if the q^{th} equation is fulfilled, 1 otherwise.

From the local syndrome s_q and s_{q+1}, it is possible to evaluate if the computation of z_q is correct. In fact, if due to an error, $\tilde{z}_q$ is output instead of z_q, both syndromes s_q and s_{q+1} are equals to 1. In that case, the correct value of y_q can be estimated as $y_q = z_{N+q} + z_{q-1}$. The hardware required to perform those operation is shown in Fig. 2.8.

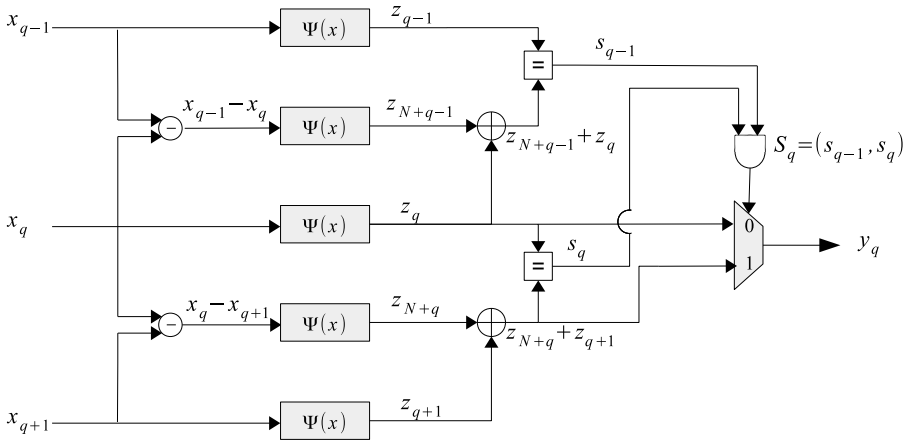


Figure 2.8: Correction using Local syndromes for N-DSC scheme

To summarise, any single error in the N-DSC module is detected and corrected. Two errors that appear in the $N-DSC$ module can be corrected if and only if their corresponding local syndromes don't interact. However, two errors that appear in two adjacent redundant modules z_{N+q} and z_{N+q-1} lead to a wrong estimation of y_q. In fact, if z_{q-1}, z_q and z_{q+1} are correct while $\tilde{z}_{N+q-1}$ and $\tilde{z}_{N+q}$ are faulty the corresponding syndrome $s_{q-1} = 1$ and $s_q = 1$ and thus, according to the correction mechanism, y_q will be estimated as $y_q = \tilde{z}_{N+q} + z_{q-1}$. This type of error can result to non correctable error, which may be problematic if the result of function Ψ is used to feed a feedback loop. In this case we can propose to split the original design in blocks of 3/4/5 functions Ψ and use the 3-DSC, 4-DSC and 5-DSC to protect each blocks thanks to the parity check variable as defined in (2.16).

2.4 Evaluation of the FT-Hamming and 4-DSC schemes

Two criteria were considered to evaluate FT-Hamming, 4-DSC and the TMR: the reliability (probability that no error appears in the output of the architecture) and the hardware efficiency of the architecture. As mentioned in Sec. 1.4, the hardware efficiency of an architecture

is defined as the normalised number of operation per unit area and time unit. To compute the hardware efficiency, the nature of the computation inside the area unit is not specified. If we consider an operation Ψ that takes n_Ψ area units and m_Ψ area clocks to be executed, the efficiency is expressed as,

$$\gamma \triangleq \frac{1}{n_\Psi \times m_\Psi} \text{ operations/(area unit} \times \text{time unit)} \tag{2.19}$$

2.4.1 The normalised hardware efficiency

- The 4-DSC solution: The efficiency of the 4-DSC-scheme is given by,

$$\gamma_{4-DSC} = \frac{4}{(8 \cdot n_\Psi + n_{DSC}) \cdot (m_\Psi + m_{DSC})} \tag{2.20}$$

  where n_{DSC} and m_{DSC} represents the number of area unit and time unit respectively of the syndrome computation and correction operations. The normalised hardware efficiency is,

$$\Gamma_{DSC} = \frac{4 \cdot n_\Psi \cdot m_\Psi}{(8 \cdot n_\Psi + n_{DSC}) \cdot (m_\Psi + m_{DSC})} \tag{2.21}$$

- The FT-Hamming solution: The efficiency of the FT-Hamming scheme is given by,

$$\gamma_{FT-Hamming} = \frac{4}{(7 \cdot n_\Psi + n_{FT_c}) \cdot (m_\Psi + m_{FT_c})} \tag{2.22}$$

  where n_{FT_c} and m_{FT_c} represents respectively the number of area unit and time unit required for the detection and correction of errors in the FT-Hamming. The normalised hardware efficiency is,

$$\Gamma_{FT-Hamming} = \frac{4 \cdot n_\Psi \cdot m_\Psi}{(7 \cdot n_\Psi + n_c) \cdot (m_\Psi + m_c)} \tag{2.23}$$

- The TMR solution solution: The efficiency of the TMR based fault tolerant scheme is given by,

$$\gamma_{TMR} = \frac{4}{(12 \cdot n_\Psi + 4 \cdot n_v) \cdot (m_\Psi + m_v)} \tag{2.24}$$

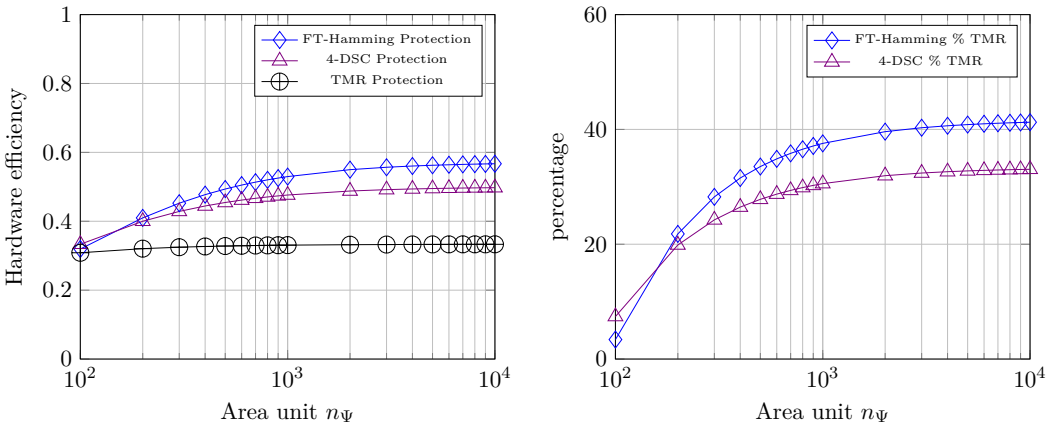
  where n_v and m_v represents respectively the number of area unit and time unit required for majority (voter) operation. The normalised hardware efficiency is,

$$\Gamma_{FT-Hamming} = \frac{4 \cdot n_\Psi \cdot m_\Psi}{(7 \cdot n_\Psi + n_c) \cdot (m_\Psi + m_c)} \tag{2.25}$$

Table 2.3 details the logic synthesis results in term of number of NAND of an homomorphism Ψ with a 12-bits output obtained using Synopsis Design compiler in the 28 nm technology. Fig. 2.9a shows the normalised hardware efficiency of the FT-Hamming, 4-DSC and the TMR as a function of the number of area unit of the homomorphism n_Ψ. Fig. 2.9b presents the

Component	n_v	n_{FT_c}	n_{DSC}
Number of NAND	24	552	400

Table 2.3: Logic synthesis results in term of number of NAND obtained by synthesising the designs with Synopsys Design compiler in the 28 nm technology



(a) (b)

Figure 2.9: Complexity comparison of the FT-Haming, 4-DSC and TMR scheme

area gain of the FT-Hamming compared to the TMR, the 4-DSC compared to the TMR and the FT-Hamming compared to the 4-DSC. For example for a 4 by 8 multiplier ($n_\Psi = 629$ NAND), the FT-Hamming and the 4-DSC reduce the complexity of the TMR by respectively 35 % and 29 %. It is important to note that for a high complex application (we can think about an FFT or convolution function where $n_\Psi > 10^4$) the FT-Hamming can reduce the complexity of the 4-DSC up to 12%. The next section will study the performances of each scheme in term of robustness.

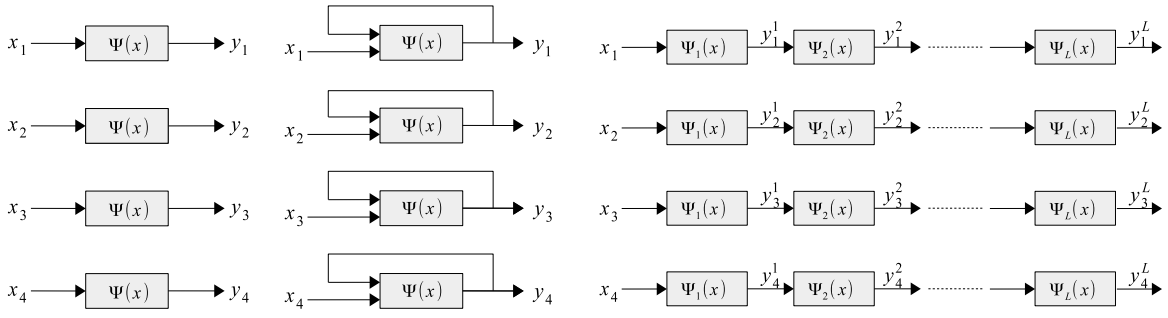
2.4.2 Robustness Performances

Let's consider the three following types of architectures: the standard, the recursive and the pipeline architectures as illustrated in Fig. 2.10. FT-Hamming and 4-DSC are then applied to each type of design in Fig. 2.12. To note here, for the pipeline case, it is possible to perform the voting, the decoding and the syndrome correction operation of respectively the TMR, FT-Hamming and the 4-DSC either at the output of each function $\Psi_1, \ldots, \Psi_L$ (Full protection) or at the end of the pipeline application i.e. only at the output of the function Ψ_L (one column protection) as shown in Fig. 2.12.b/c/e/f. The one column protection permits to reduce the complexity but suffers from a degradation of the robustness performance. This section will evaluates and compares the performance of the FT-Hamming, 4-DSC and TMR by computing the Mean Time To Failure (MTTF): larger MTTF implies a more reliable system.

Definition 4: We define the failure as the event where at least one of the four original

modules output is faulty after correction.

Hypothesise 2: We assume that the encoding and decoding process in the FT-Hamming and the syndrome computation and the correction inside the 4-DSC are fault free.



(a) Standard (b) Recursive (c) Pipeline (or a serial)

Figure 2.10: Examples of application to explore fault tolerance

Let P_Ψ be the probability that the original function Ψ is faulty. To compute the Mean Time To Failure (MTTF) it is first necessary to determine, $P_e(t)$, the error probability of failure of the system at an instant t, after (t -1) clock cycles of correct behaviour.

$$P_e(t) = (1 - P_{\text{Fail}})^{(t-1)} P_{\text{Fail}}. \tag{2.26}$$

where P_{Fail} is the probability of an instantaneous failure in the complete system and depends on the proposed scheme: The MTTF is given by

$$\text{MTTF} \triangleq \sum_{t=1}^{\infty} t P_e(t) = \frac{1}{P_{Fail}}. \tag{2.27}$$

- **The standard and the recursive design**:
  - The TMR solution: The outputs are correct only if the output of the three voters are correct. Thus,

$$P_{\text{Fail}} = 1 - (\ [\ (1 - P_\Psi)^3 + 3\ P_\Psi\ (1 - P_\Psi)^2\ ]^4\ ). \tag{2.28}$$

  - The FT-Hamming solution: The FT-Hamming scheme is able to correct only one faulty processing module among 7. Thus,

$$P_{\text{Fail}} = 1 - [\ (1 - P_\Psi)^7 + 7\ P_\Psi\ (1 - P_\Psi)^6]. \tag{2.29}$$

  - The 4-DSC solution: The 4-DSC corrects one faulty output among eight. It is able also to elliminated some cases where two faults appears in the scheme. The cases are when errors appears in (z_1,z_8), (z_2,z_5), (z_3,z_6), (z_4,z_7), (z_5,z_7) and (z_6,z_8). Thus,

$$P_{\text{Fail}} = 1 - [\ (1 - P_\Psi)^8 + 8\ P_\Psi\ (1 - P_\Psi)^7 + 6\ P_\Psi^2\ (1 - P_\Psi)^6]. \tag{2.30}$$

Now let consider the 4-DSC with a parity check bit z_9 (as defined in (2.16) for the 3-DSC). This scheme is called 4-RDSC (R: Robust). We assume that an error is flagged and a roll-back error recovery mechanism is activated when $z_9 \neq 0$ and more than one fault occur in the original computing modules. The probability that the 4-RDSC schema doesn't detect the faulty behaviour and outputs a wrong value is given as,

$$P_{\text{Fail}}^{4-RDSC} = 1-[\,(1-P_\Psi)^8+8\,P_\Psi\,(1-P_\Psi)^7+(1-(1-P_\Psi)^4)(1-(1-P_\Psi)^4)]. \quad (2.31)$$

- **The pipeline architecture**: To simplify our studies, we suppose that the failure probability of Ψ_1, Ψ_2, ... Ψ_L, is the same and equal to P_Ψ.
  - The Full TMR protection:

$$P_{\text{Fail}} = 1 - (\ [\ (1-P_\Psi)^3 + 3\ P_\Psi\ (1-P_\Psi)^2\ ]^4\ )^L. \quad (2.32)$$

  - The Full FT-Hamming protection:

$$P_{\text{Fail}} = 1 - [\ (1-P_\Psi)^7 + 7\ P_\Psi\,(1-P_\Psi)^6]^L. \quad (2.33)$$

  - The full 4-DSC protection:

$$P_{\text{Fail}} = 1 - [\ (1-P_\Psi)^8 + 8\ P_\Psi\ (1-P_\Psi)^7 + 6\ P_\Psi^2\ (1-P_\Psi)^6\ ]^L. \quad (2.34)$$

Let $P_i(t)$ the error probability at the output of the i^{th} function of the pipeline architecture.

$$P_i(t) = (1-P_{i-1}(t-1)) \times P_\Psi + (1-P_\Psi) \times P_{i-1}(t-1) \quad (2.35)$$
$$= (1-2\ P_{i-1}(t-1)) + P_\Psi. \quad (2.36)$$

Iterating the calculation, the error probability at the output of the n^{th} function P_n is determinate.

  - One Column TMR protection:

$$P_{\text{Fail}} = 1 - (\ [\ (1-P_n)^3 + 3\ P_n\ (1-P_n)^2\ ]^4\ ). \quad (2.37)$$

  - One Column FT-Hamming protection:

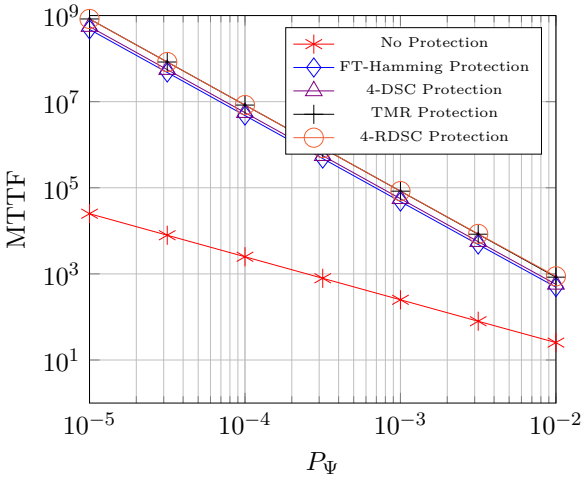
$$P_{\text{Fail}} = 1 - [\ (1-P_n)^7 + 7\ P_n\,(1-P_n)^6]. \quad (2.38)$$

  - One Column 4-DSC protection:

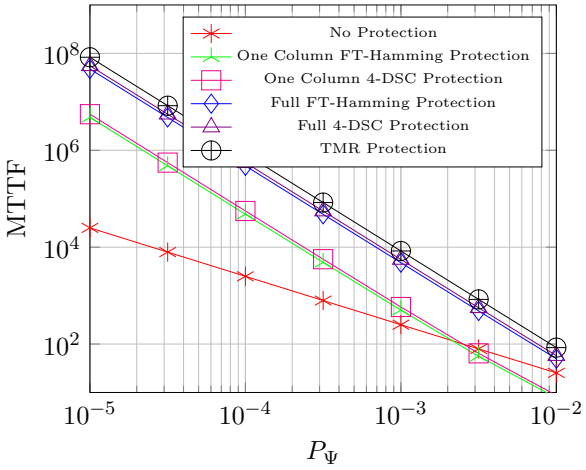
$$P_{\text{Fail}} = 1 - [\ (1-P_n)^8 + 8\ P_n\ (1-P_n)^7 + 6\ P_n^2\ (1-P_n)^6\ ]. \quad (2.39)$$

Fig. 2.11 summarises the performances results in term of MTTF. From this figure, we can see that the 4-DSC scheme provides robustness better (larger MTTF) than the FT-Hamming

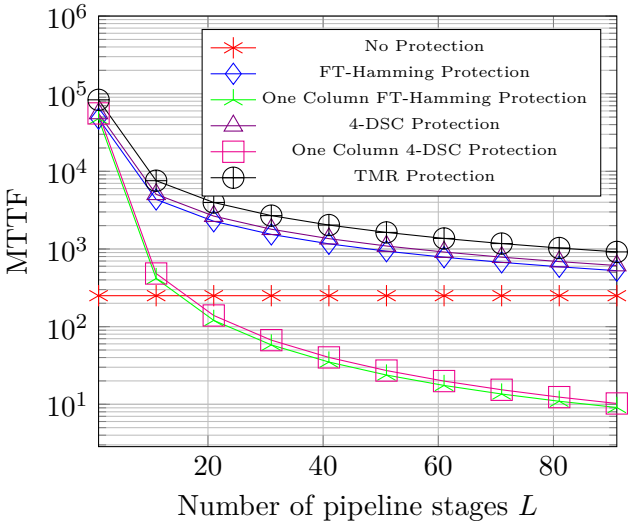
and closed to the TMR method. For example, with the recursive application in Fig. 2.11a, for a unit error probability $P_\Psi = 10^{-3}$, MTTF goes from 1000 clock cycles if the system is not protected, to 47 778 with the FT-Hamming protection and 55 716 with the 4-DSC while the TMR achieves a MTTF = 83 389. The 4-RDSC demonstrates its interest in term of robustness since it the probability of a failure (non detection of faulty computation in the scheme) shows a performances similar to the TMR. It is furthermore important to mention the same approach can be applied to improve the detection behaviour of the TMR for two errors by comparators at the output of each two replicas. The high robustness' performance of



(a) The recursive application: MTTF= f (P_Ψ)



(b) The pipeline architecture: $L = 10$



(c) The pipeline architecture: $P_\Psi = 10^{-3}$

Figure 2.11: Performance Results

the 4-DSC and the FT-Hamming is valid with the pipeline application for a number of stages $L = 10$. As we can see in Fig. 2.11b, the one column configurations of the FT-Hamming and

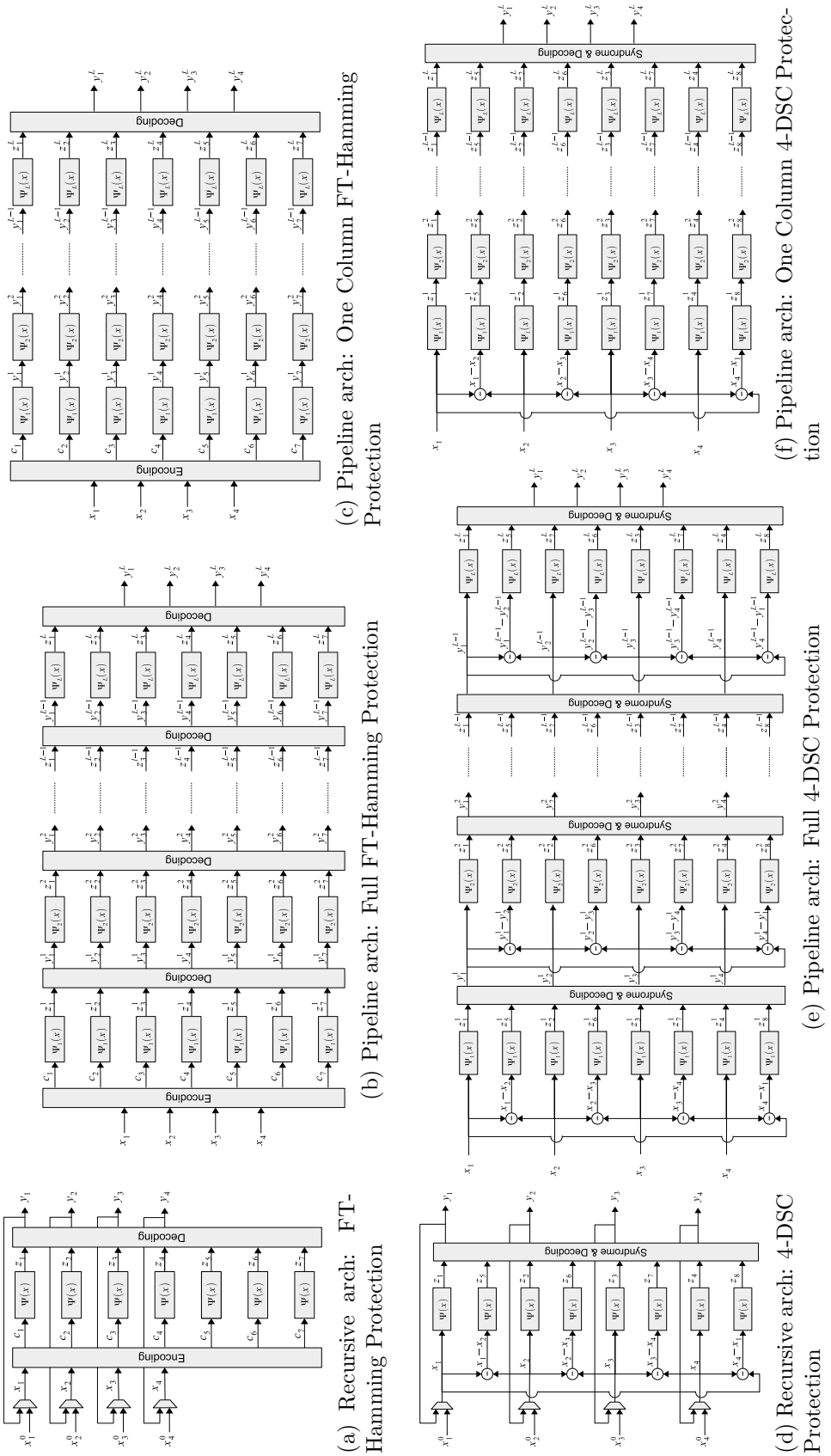
4-DSC lose their efficiency in term of robustness for high unit error probability ($P_\Psi > 0.003$) and have similar or even less robustness against errors than the no protected solution. The performances of FT-Hamming, 4-DSC, 4-RDSC and the TMR are also evaluated for different number of stages L in the pipeline structure in Fig. 2.11c. From this figure, we can see that the FT-Hamming, the 4-DSC and the TMR continue to perform well for a high number of stage while the One column configurations scheme fails to offer a high level of reliability for Ł > 40.

2.5 Summary

In this chapter, we presented two new resilient schemes based respectively on error correcting codes and duplication: FT-ECC and DSC. In this work, FT-Hamming and 4-DSC, two instances of the FT-ECC and DSC, were introduced, evaluated and compared to the classical TMR at two levels: the hardware efficiency and robustness performances in term of MTTF. Analysis results show that for a 4 by 8 multiplier, the 4-DSC scheme reduces the complexity by 29%, compared to the classical TMR scheme, while maintaining a level of reliability closed to the TMR. The FT-Hamming demonstrates also an interesting level of reliability and reduces by 12 % the hardware compared to the 4-DSC. In recursive applications where injecting errors can cause a system failure, it is less harmefull to inject nothing in loops (or using previous output value) than injecting wrong value. The more resilient version of the 4-DSC, 4-RDSC, can offer the possibility to mitigate the event where more than one fault occur in the scheme. The probability to inject errors in loops is similar to the failure probability of the TMR. An example of such recursive application will be given in the next chapter. This work is published in [10] as part of the SIPS 2017 conference.

During our study we assumed that the detection and the correction in the FT-Hamming, 4-DSC and TMR are fault free. In future works, we can extend our analyse of the reliability of different schemes by taking into consideration the possible faulty detection and correction operations and by exploring the idea of using the ANT technique to reduce the area of each scheme for signal processing applications. Finally, it is important to note that it is also possible to use the temporal redundancy to detect and correct errors: for example with the 3-DSC concept, $z_4 = \Psi(x_1 - x_2)$, $z_5 = \Psi(x_2 - x_3)$ and $z_6 = \Psi(x_3 - x_1)$ can be performed at the clock cycle following the computation of $z_1 = \Psi\ (x_1)$, $z_2 = \Psi(x_2)$ and $z_3 = \Psi(x_3)$ instead of duplicating the hardware.

The rest of the report will address the reliability of an example of signal processing application which is the tracking module of GPS receivers. Fault tolerant techniques including the FT-Hamming and 3-DSC and the state of art techniques are added to make the standard version of the GPS tracking modules tolerant to errors due PVT variations. Next chapter provides a brief introduction to the GPS receivers and its tracking modules.



(a) Recursive arch: FT-Hamming Protection
(b) Pipeline arch: Full FT-Hamming Protection
(c) Pipeline arch: One Column FT-Hamming Protection
(d) Recursive arch: 4-DSC Protection
(e) Pipeline arch: Full 4-DSC Protection
(f) Pipeline arch: One Column 4-DSC Protection

Figure 2.12: Application of the FT-Hamming and 4-DSC for recursive and pipeline applications

Chapter 3

Introduction to the GPS navigation system

Contents

The GPS is a satellite-based system that allows determining the physical position and the absolute time of a user. It was funded and operated by the U.S. Department of Defense (DOD) in the early 1970's. The concept was originally intended for military applications [41]. Later, the U.S. government made the system available for civilian use. Two separate services exist actually: the Standard Positioning Service (SPS) and the Precise Positioning Service (PPS). The SPS is designated for the civil community, whereas the PPS is intended for U.S. authorised military and selected government agency users. This chapter is a brief introduction to the SPS service of the GPS system, known as Coarse/Acquisition (C/A) service.

3.1 GPS system segments

The C/A service' system is composed of three segments: satellite constellation, ground-control/ monitoring segment, and user receiving equipment [41]. 1/ The satellite constellation is the set of satellites in orbit that provide the ranging signals and data messages to the user equipment. Fig. 3.1 shows the standard GPS satellites constellation. Each GPS satellite is characterised by a Space Vehicle Pseudo Random Noise Number (SV PRN Number). The actual constellation consists of more than 24 GPS satellites at the option of commissioning and decommissioning. Constellation on the first of June 2017 is composed of 31 satellites of the 2^{nd} generation (Block II) [42]. All the constellation provides continuous (24 hours/day),

real-time, 3-dimensional positioning, navigation and timing worldwide. Any person with a civil GPS receiver can access the system and uses it for any application that requires location coordinates or time. 2/ The control segment monitors satellite health and signal integrity and maintains the orbital configuration of the satellites. Finally, 3/ the user receiver equipment performs the navigation, timing, or other related functions [43].

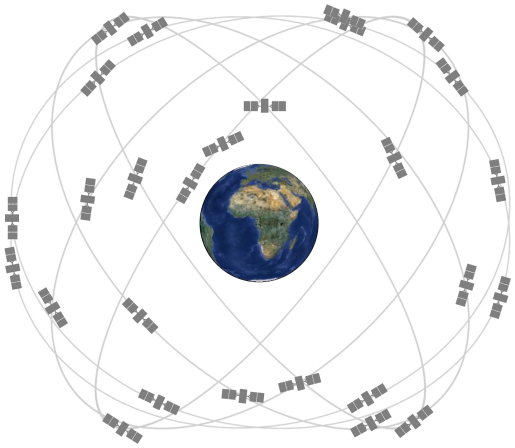


Figure 3.1: 24-GPS satellite constellation, as defined in the SPS Performance Standard [41]

3.2 GPS position measurements

In literature, it is possible to determinate the physical position of a user if the locations of three GPS satellites and the distances to them are known (see Fig.3.2). To compute d_i, the distance to the i^{th} GPS satellite, the user measures the time it takes for a signal transmitted by the satellite at the known location to reach the user.

$$d_i = c \times (t_i - t_0), \tag{3.1}$$

where t_0 is the time when the signal left the GPS satellite and t_i is the time when the signal transmitted by the i^{th} GPS satellite reach the user. All GPS satellites are synchronised to a reference clock. However, the clock of the receiver may not be synchronised to this clock. For this reason, the GPS receiver calculates the time shift between signals sent on the same time and defines its position as the intersection of the hyperboles[1] H_{13}, H_{12} and H_{23}. (P_1,P_2), (P_1,P_2) and (P_1,P_2) are respectively the foci of H_1, H_2 and H_3.

$$(H_{12}) : d_2 - d_1 = c \times (t_2 - t_1) \tag{3.2}$$
$$(H_{13}) : d_3 - d_1 = c \times (t_3 - t_1) \tag{3.3}$$
$$(H_{23}) : d_3 - d_2 = c \times (t_3 - t_2) \tag{3.4}$$

[1] A hyperbola H of foci F_1 and F_2 is defined as the set of points P where the absolute difference of the distances $|PF_1|$ and $|PF_2|$is constant.

The time is an unknown parameter for the GPS receiver. So to compute the position in space and time, the GPS receiver have to resolve an equation with four unknown parameters. Thus four GPS satellites are required at least to uniquely determine the user position in space and time. The time t_0 is required for a GPS receiver to be able to determine the position of a user. For this reason, each GPS satellite broadcast over time this parameter (t_0) as a part of a message named the navigation. Since GPS satellites are in continuous motion, continual clock correction compared to the reference clock is also transmitted in the navigation message. More details about the navigation calcul of the position can be found [**?**].

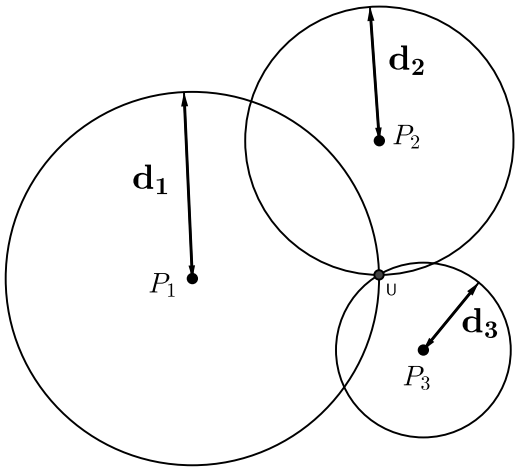


Figure 3.2: The GPS user position.

3.3 GPS Satellite signals

For the C/A Service, each GPS satellite transmits on the L1 band, centred at 1575.42 MHz, using the Code Division Multiple Access (CDMA) technology, two types of signals: the navigation message and the C/A code (PRN sequence). The analytical expression of the transmitted signal of a satellite k is:

$$e_k(t) = c_k(t)\ d_k(t)\ e^{2\pi j f_{L1} t}, \tag{3.5}$$

where:

- $d_k(t)$: navigation message of the k^{th} satellite. It is sent at the frequency of 50 Hz ($d_k(t) \in \{+1, -1\}$). The navigation message consists of a frame of 1500 bits divided into five sub frames of 300 bits. The sub frames number 1, 2 and 3 contain data for calculating the position of the satellite transmitter (ephemeris) and other parameters such as clock corrections and ionosphere. The duration of a complete message is 25 frames or 37500 bits issued in 750 s (12 min). Sub frames 1 to 3 are identical for the 25 frames while sub frames 4 and 5 changes each frame [43].

- $c_k(t)$: k^{th} Coarse/Acquisition (C/A) satellite spreading code with a Binary Phase Shift Keying (BPSK) modulation (i.e. $c_k \in \{-1, 1\}$). The length of the C/A code is 1023 chips and is transmitted at the frequency 1.023 MHz. C/A codes belong to the family of Gold pseudo random noise (PRN) sequences. They have a good correlation properties and are almost orthogonal one to the other. It can therefore be used to distinguish different GPS satellites.

- f_{L1}: the carrier frequency in the L1 GPS Band ($f_{L1} = 1575.42$ MHz).

3.4 GPS User Equipment

At the receiver level, the GPS incoming signal is first shifted to base band thanks to a local oscillator of frequency $f_{L1} - f_I$ (f_I represent the intermediate frequency) and after a pass band filter, sampled at a frequency of 4 MHz. Fig. 3.3 describes the generic diagram of the GPS receiver.

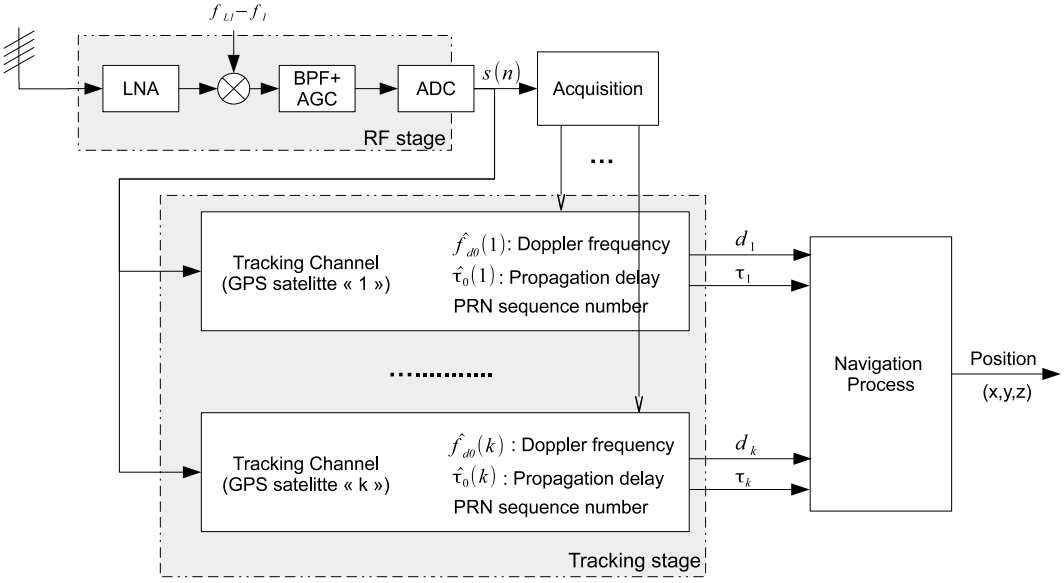


Figure 3.3: Generic GPS receiver block diagram

The n^{th} received sample $s(n)$ is given by

$$s(n) = \sum_{k\in\{1..K\}} A_k(nT)\; d_k(nT - \hat{\tau}_k(nT))\; c_k(nT - \hat{\tau}_k(nT))\; e^{j(\hat{\theta}_k n + \hat{\varphi}_k)} + w(n), \qquad (3.6)$$

where:

- $\hat{\theta}_k = 2\pi T \left(\hat{f}_{d(k)}(nT) + f_I\right)$,
- $A_k(nT)$: Attenuation parameter of k^{th} satellite signal at time nT,

- $\hat{\tau}_k(nT)$: Propagation delay between satellite k and receiver at time nT,
- $\hat{f}_{d(k)}(nT)$: Estimated Doppler frequency of satellite k at time nT,
- $\{1..K\}$: Set of satellites visible at time n,
- T: Sampling period ($T = 0.25\ \mu s$),
- $w(n)$: Additive White Gaussian Noise at time n.

The digital part of GPS receiver has to extract and demodulate the navigation message of all GPS satellites in view to make the distance measurement and determine the position of a user as introduced in Sec. 3.2. Extraction of navigation messages involves two essential and sequential process: the acquisition process and tracking process. 1) The acquisition process is the process by which the receiver identifies which satellites are in view. It is a a three-dimensional search to determine the GPS satellite identifier (SV PRN number), the code phase (represented by τ) and the carrier frequency offset due to Doppler effect (represented by f_d). 2) Since satellites are in continuous motion, the distance between any satellite and the receiver is dynamic. Besides to that, the carrier of the received signal is also constantly changing in time due to Doppler shifts. Therefore, the acquisition process is followed by a tracking process where signal received from each GPS satellites are tracked in permanent. Multiple tracking channel modules exits in a GPS receiver design; each module tracks a unique GPS satellite. To note here, any GPS receiver design contain at least four channel tracking module. A simplified representation of the channel tracking module is given in Fig. 3.4.

Each tracking channel module is composed of a correlation computation module and two tracking loops (the carrier tracking loop and the code tracking loop). The carrier tracking loop performs the task of aligning the local generated carrier with the carrier of the incoming signal (i.e, suppressing the Doppler effect) while the code tracking loop ensures the time alignment of the local generated codes. The correlation process compares the locally generated signals with the incoming signals. Based on the correlation output, the locally generated carrier and codes are updates over time to track parameters of the incoming signal. A maximum correlation is achieved when the two signals are aligned which leads to a precise demodulation of the navigation message.

3.4.1 The carrier tracking loop

Two parameters are characterising the carrier: the frequency and the phase. Thus, the carrier tracking loop is formed in reality of two nested loops: a phase loop or Phase Locked Loop (PLL) and a frequency loop also called Frequency Locked Loop (FLL). The FLL and PLL are each one made of a discriminator and use a common loop filter: the carrier filter. The FLL produce an estimate of the frequency error between the incoming signal and the local generated carriers (the role of the frequency discriminator in Fig. 3.4). A well designed frequency locked loop will generally yield a frequency estimate that converges toward the true carrier frequency of the incoming signal. With this frequency uncertainty, a receiver try to estimate the second carrier parameter: its phase. Homologous to the FLL, the PLL is a recursive estimator of the carrier phase offset.

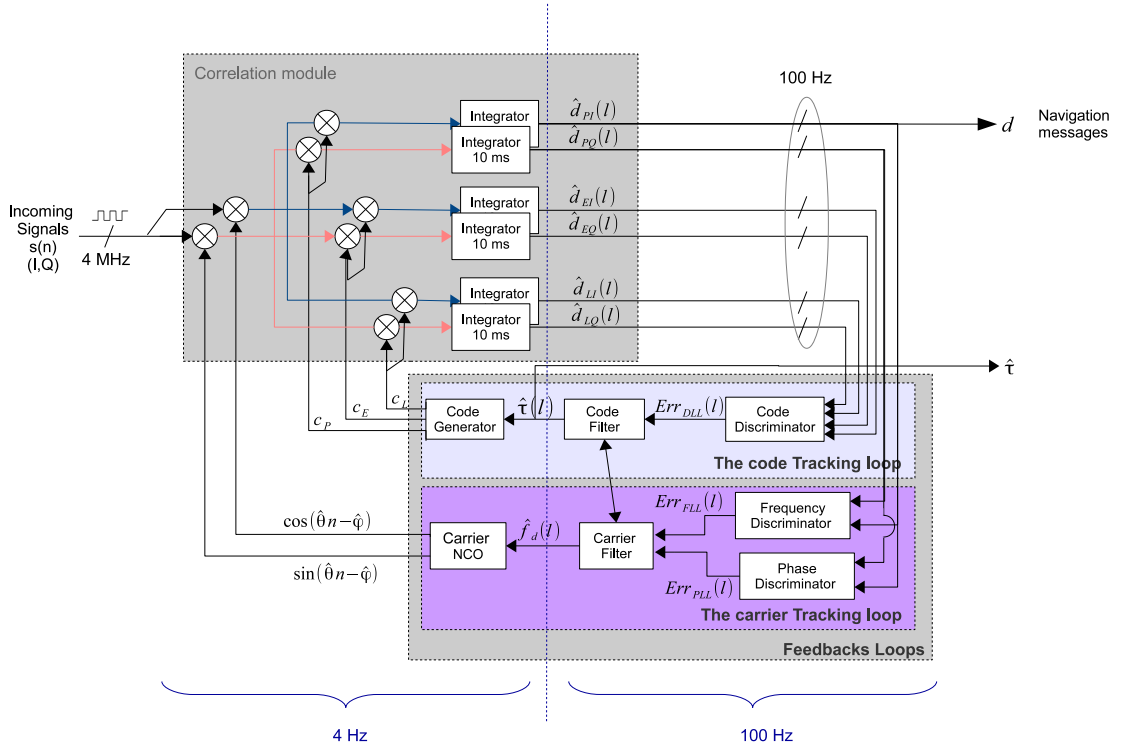


Figure 3.4: Generic digital receiver channel block diagram. $\hat{d}_P^a(\ell) \triangleq \sum_{10\,\mathrm{ms}} s(n)\, c_a(nT - \hat{\tau}_a(nT))\, e^{-j(\hat{\theta}_a n + \hat{\varphi}_a)}$. C/A codes belongs to the family of Gold Pseudo Random Noise (PRN) sequences. Hence codes of all GPS satellites are orthogonal, a perfect time alignment ($\hat{\tau}_a = \tau_a$) and a perfect frequency and phase alignment ($\hat{\theta}_a = \theta_a$ and $\hat{\varphi}_a = \varphi_a$) implies that $\hat{d}_P^a(\ell) = d_a(\ell)$. Thus a navigation message is extracted. c_E and c_L are delayed versions of c_P; E: Eraly ,L: Late

There are several discriminator for each PLL and FLL. The choice depends on the application and the implementation's complexity [43]. Arc Tangent (ATAN) based discriminators are the most used because of their performances [43]. They are defined as

$$\mathrm{Err}_{FLL}(l) = \frac{\tan^{-1}(\frac{\mathrm{cross}(l)}{\mathrm{dot}(l)})}{2\pi T_{int}}; |\,\mathrm{Err}_{FLL}(l)\,| < 25\ \mathrm{Hz}, \tag{3.7}$$

and

$$\mathrm{Err}_{PLL}(l) = \frac{\tan^{-1}(\frac{\hat{d}_{PQ}(l)}{\hat{d}_{PI}(l)})}{2\pi};\ |\,\mathrm{Err}_{PLL}(l)\,| < 0.25\ \mathrm{rad}, \tag{3.8}$$

where

- $T_{int} = 10\ ms$,
- cross(l)= $\hat{d}_{PI}(l-1) \cdot \hat{d}_{PQ}(l) - \hat{d}_{PI}(l)\hat{d}_{PQ}(l-1)$,
- dot(l)=$\hat{d}_{PI}(l-1)\hat{d}_{PI}(l) + \hat{d}_{PQ}(l)\hat{d}_{PQ}(l)$,

- $\hat{d}_{PI}(l)$: l^{th} In-phase copy of the correlation function result during 10 ms,
- $\hat{d}_{PQ}(l)$: l^{th} Quadrature copy of the correlation function result during 10 ms,
- $\tan^{-1}$: the inverse tangent function

Frequency and phase offsets are then applied to the Carrier Filter. The loop filter order and bandwidth are the parameters characterising the carrier filter. There are many design approaches to construct digital loop filters. Details about loop filters characteristics can be found in [43]. Ground GPS receivers use the first order FLL assisted second order PLL as shown in Fig. 3.5. If the phase error input is zeroed in this filter, the filter becomes a pure FLL. Similarly, if the frequency error input is zeroed, the filter becomes a pure PLL. The filtered carrier error estimate is applied to the NCO generator, to adjust the local carrier for the generation of the next local carrier signal. Fig. 3.6 shows a block diagram of the carrier loop NCO and its sine and cosine mapping functions.

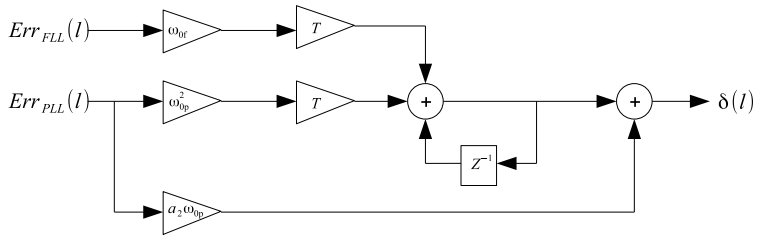


Figure 3.5: First order FLL assisted second order PLL where $\omega_{0f} = \frac{B_{FLL}}{0.25}$ and $\omega_{0p} = \frac{B_{PLL}}{0.53}$. B_{PLL} and B_{FLL} are the phase and the frequency filters bandwidths. a_2=1.414

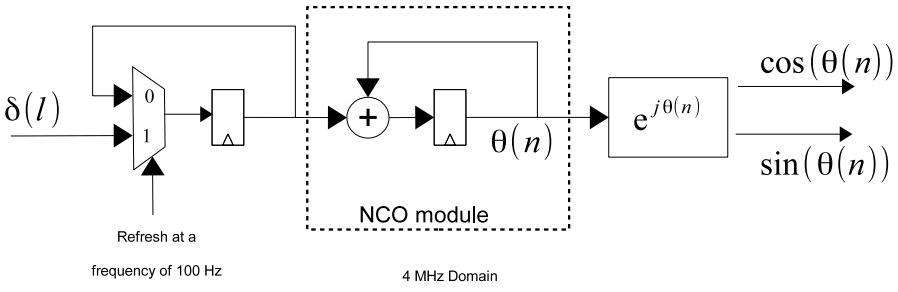


Figure 3.6: NCO generator block diagram. $\hat{\theta}$ represents the output of the NCO carrier. It is generated each 0.25 µs ($\frac{1}{4\,\text{MHz}}$). $\delta(l)$ is the estimated Doppler offset. It remains constant each during 10 ms

3.4.2 The code tracking loop

The code tracking loop is a recursive estimator of the code delay. It is called an early-late tracking loop and is formed by a discriminator, a filter and the code generator as seen in Fig. 3.4. The idea behind the DLL is to correlate the input GPS signal with three replicas

of the code: Early, Prompt and Late. These three replicas are generated with a spacing of $\pm\frac{1}{2}$ chip. The six correlation outputs, are then compared. Fig. 3.7 illustrates how the early, prompt and late normalised correlation output change with respect to the incoming signal. In Fig. 3.7a, the late code has the highest correlation output, the code phase must be decreased while in Fig. 3.7a, the early code has the highest correlation output, the code phase must be increased in this case. Fig. 3.7a presents the case when the incoming signal and the prompt replica signal are perfectly aligned.

Four code discriminators are listed in [43]. The Normalized early minus late power code discriminator produces the most precise code phase tracking error. It is defined as

$$\mathrm{Err}_{DLL}(l) = \frac{1}{2}\frac{E(l) - L(l)}{E(l) + L(l)}, \tag{3.9}$$

where $E(l) = \sqrt{\hat{d}_{EI}^2(l) + \hat{d}_{EQ}^2(l)}$ and $L(l) = \sqrt{\hat{d}_{LI}^2(l) + \hat{d}_{LQ}^2(l)}$. It important to note that the carrier loop jitter is considered to be much less noisy than the code loop jitter. For this reason, the estimation of the carrier frequency can be used to aid the estimation of the code phase and a first-order code loop filter is enough at the output of the code discriminator [43]. The filtered code error estimate is applied to the code generator to produce another three replica codes. The process of the generation of GPS codes is explained in the next section.

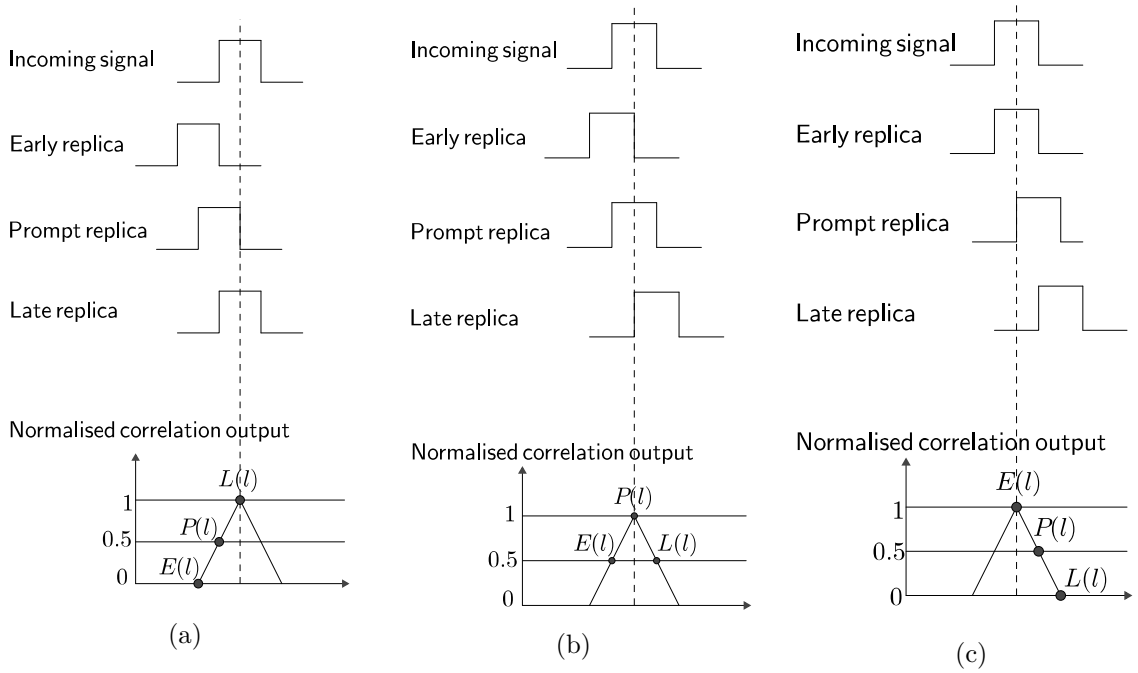


Figure 3.7: Two cases for the code correlation case. (a) Prompt replica is 1/2 early than the incoming signal, so the code phase must be decreased. (b) the prompt replica code is perfectly aligned with the incoming signal. (c) Prompt replica is 1/2 late than the incoming signal, so the code phase must be increased.

3.4.3 GPS Code Generation

The GPS C/A code is a Gold code of 1,023 chips transmitted at the frequency 1,023 MHz. The repetition period of the GPS C/A code is 1ms ($\frac{1023}{1.023\times10^6}$). Fig. 3.8 shows the design architecture of the GPS C/A code generator. Two Linear Feedback Shift Registers (LFSR) are used to produce the C/A code: the G1 shift register and the G2 shift register. The feedback taps of the G1 shift register is connected to stages 3 and 10 while the feedback taps of the G2 shift register is connected to stages 2, 3, 6, 8, 9 and 10. For each GPS satellite, the C/A code is unique, it is the result of the exclusive-or of the G1 output sequence (stage 10) and the exclusive-or output of the GPS satellite corresponding taps combinations of the G2 register stages. Table 3.1 illustrates the tap combinations corresponding to each GPS satellite identified by the SV PRN number. Note that Initial state of G1 and G2 registers are both equal to "1111111111".

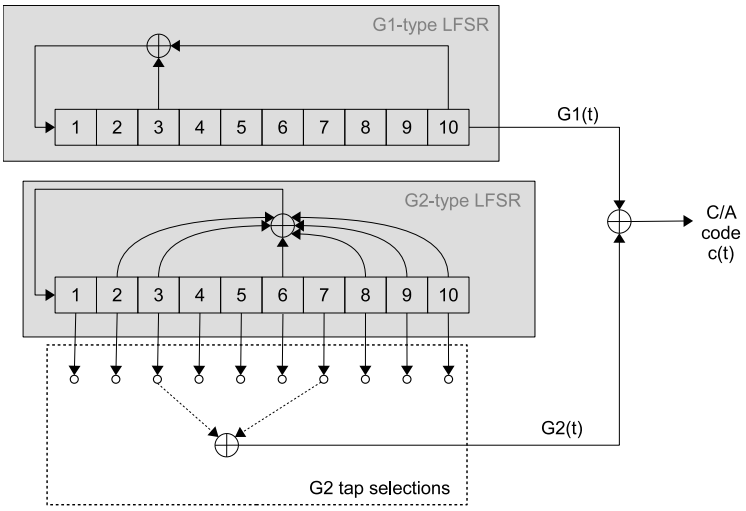


Figure 3.8: C/A Code Generator

3.5 Summary

This chapter discussed the GPS navigation system. It first gave a brief description of the overall GPS system. Then, we presented the digital part inside GPS receivers and we detailed the GPS tracking loops: the code tracking loop and the carrier tracking loop. The next chapter will study the robustness of the GPS tracking modules against faults and propose fault tolerant techniques to improve the quality of the GPS position.

SV PRN Number	G2 register Tap selection	First 10 C/A chips (octal)	SV PRN Number	G2 register Tap selection	First 10 C/A chips (octal)
1	$2 \oplus 6$	1440	2	$3 \oplus 7$	1620
3	$4 \oplus 8$	1710	4	$5 \oplus 9$	1744
5	$1 \oplus 9$	1133	6	$2 \oplus 10$	1455
7	$1 \oplus 8$	1131	8	$2 \oplus 9$	1454
9	$3 \oplus 10$	1626	10	$2 \oplus 3$	1504
11	$3 \oplus 4$	1642	12	$5 \oplus 6$	1750
13	$6 \oplus 7$	1764	14	$7 \oplus 8$	1772
15	$8 \oplus 9$	1775	16	$9 \oplus 10$	1776
17	$1 \oplus 4$	1156	18	$2 \oplus 5$	1467
19	$3 \oplus 6$	1633	20	$4 \oplus 7$	1715
21	$5 \oplus 8$	1746	22	$6 \oplus 9$	1763
23	$1 \oplus 3$	1063	24	$4 \oplus 6$	1706
25	$5 \oplus 7$	1743	26	$6 \oplus 8$	1761
27	$7 \oplus 9$	1770	28	$8 \oplus 10$	1774
29	$1 \oplus 6$	1127	30	$2 \oplus 7$	1453
31	$3 \oplus 8$	1625	32	$4 \oplus 9$	1712

Table 3.1: The syndrome corresponding to each faulty module for the FT-Hamming method

Chapter 4

Case Study: Fault tolerance of the GPS tracking channel module

Contents

This chapter addresses the fault tolerance topic in the GPS context. Starting from the standard GPS receiver architecture, redundant mechanisms are added to design a more resilient GPS receiver tolerant to hardware errors. The redundant mechanisms include some of the state of the art fault tolerant techniques introduced in chapter 1 and the new resilient scheme FT-ECC or DSC described in chapter 2. The choice depends on the required level of reliability of each part of the GPS receiver architecture. This chapter is organised as follows: First we discuss the sensibility of each element of the GPS tracking channel module to faults and define the error reliability level of each component of the tracking module. Then, we separately analyse the fault resilience of each element of the tracking channel module and propose fault tolerant techniques to improve their reliability against faults.

4.1 Resilience of the tracking loop channel against faults

The GPS tracking module contains a very interesting set of different signal processing problems with different requirements of reliability: Correlation process, tracking loops (recursive operations), Gold and carrier generators... So it is important first to identify which element of the architecture, errors at its output have the most impact on the computed GPS position.

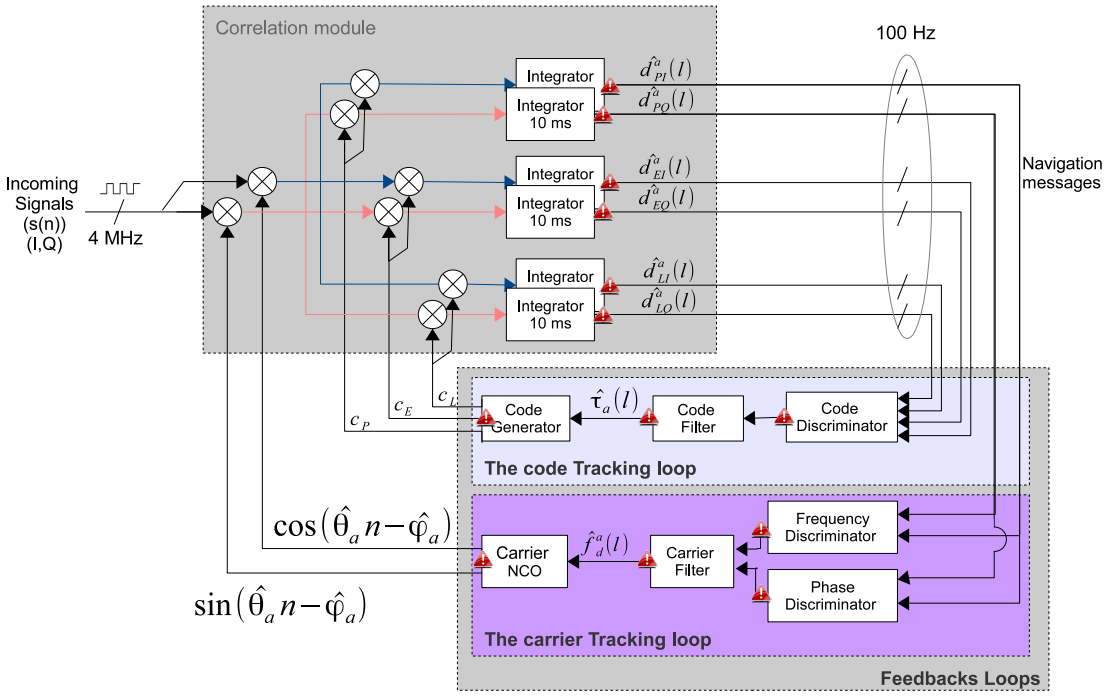


Figure 4.1: Top Level of the tracking channel module; Error tolerance has been studied in components with warning marks at the output.

We considered S_{track}, the set of elements in the tracking channel module to analyse their required level of error reliability. It includes the carrier generator, the code filter, the carrier filter, the code discriminator, the frequency discriminator, the phase discriminator and the correlation function module. We assume that the code generator requires a high level of error reliability, otherwise the generated codes don't correlate with the codes of the tracked GPS satellite and thus the tracking channel is no more able to extract a correct navigation message which in return perturbs the computation of the position. Fig. 4.1 illustrates fault elements of the architecture to analyse. Seven case studies exist; at each case we assume that one of element of S_{track} is faulty while the rest of elements are non faulty and then, we use the following methodology to determine the error reliability of the faulty element:

- Use the front end of a GPS receiver to store incoming GPS signal $s(n)$ over a significant period of time (few minutes).
- Process the stored input GPS signal with a reference noiseless GPS receiver [44] to generate the set of successive estimated positions $X(j)_{j=1...Q}$.

- Replace the noiseless GPS receiver with a noisy GPS receiver where only one element is fixed to be faulty (or noisy) while the others elements are assumed to be non faulty and process N times the stored incoming GPS signal, by Monte-Carlo simulations to generate sets of noisy position $\tilde{X}^i(j)_{j=1..Q}$, $i = 1..N$.
- Compute $\sigma_X(p)$, the standard deviation between positions given by the faulty (noisy) and non-faulty (the noiseless) GPS receivers. $\sigma_X(p)$ gives us an idea about the distance, on average, between positions given by the noiseless and the noisy GPS receivers. It is defined as

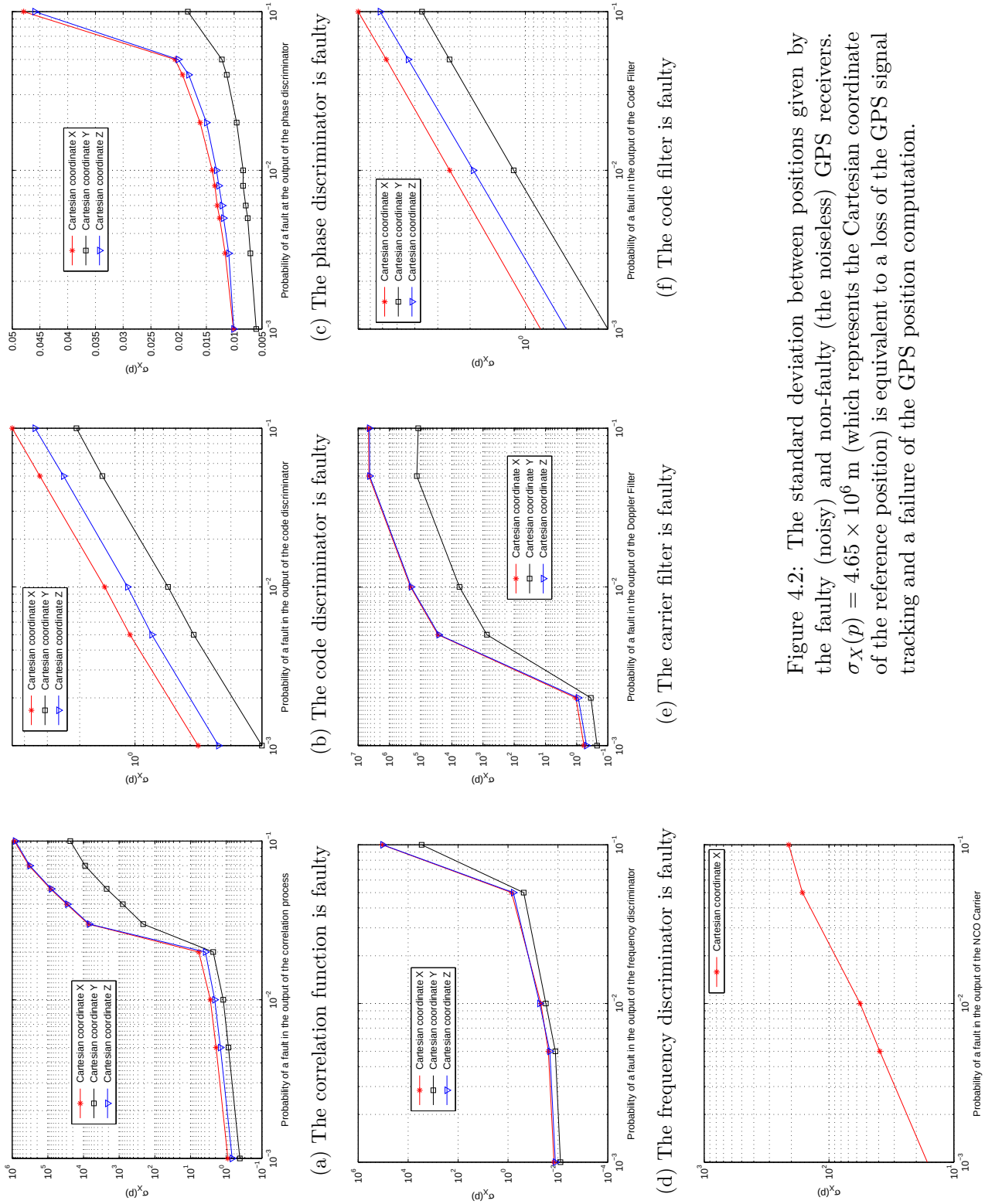
$$\sigma_X(p) \;=\; \sqrt{E(\mid X - \tilde{X} \mid^2)}. \tag{4.1}$$

The model of noise is very simple. For each active tracking module, the output of the faulty element is assumed to be exact with a probability $(1-p)$ or to be faulty with a probability p. In case of faulty result, a random value is uniformly drawn between the smallest and highest value of the faulty element's output to replace the exact value.

Results of the analyse of $\sigma_X(p)$ for different cases study are summarised in Fig. 4.2; In this figure, $\sigma_X(p) = 4.65 \times 10^6$ m (which represent the Cartesian coordinate of the reference position) is equivalent to a loss of the GPS signal tracking: the GPS receiver is not able to determine the position of the user. Let us consider the two following cases: when the correlation process is faulty (Fig. 4.2a) and when the phase discriminator is faulty (Fig. 4.2c).

- The correlation process output is faulty: When $p < 0.02$ the error in the position $\sigma_X(p)$ doesn't exceed 7 m while for a probability of error equal $p = 0.1$, $\sigma_X(p) = 3 \times 10^4$ m, i.e, the computed position is 30 km far from the reference position. So, we can conclude that the GPS receiver tolerate until 2 % of incorrect correlation outputs. However, for high error probability, $p > 0.02$ the correlation process need to be protected to be able to maintain the tracking of the GPS satellites.
- The phase discriminator is faulty: even for high error probability, $p = 0.1$, $\sigma_X(p)$ doesn't exceed 1 m. The phase discriminator will be considered as high reliable unit and doesn't require a high reliability effort. This result is explained by the fact that the PLL is less sensitive to dynamic stress.

Similar as the correlation process module, the frequency discriminator, the carrier filter and the NCO carrier need an high error reliability effort while the code discriminator and filter are more tolerant to high error probabilities if we tolerate a loss of 10 m in the GPS position compared to the reference position.



(a) The correlation function is faulty (b) The code discriminator is faulty (c) The phase discriminator is faulty

(d) The frequency discriminator is faulty (e) The carrier filter is faulty (f) The code filter is faulty

(g) The NCO Carrier is faulty

Figure 4.2: The standard deviation between positions given by the faulty (noisy) and non-faulty (the noiseless) GPS receivers. $\sigma x(p) = 4.65 \times 10^6$ m (which represents the Cartesian coordinate of the reference position) is equivalent to a loss of the GPS signal tracking and a failure of the GPS position computation.

4.2 Robustness of the Correlation Function

Faults, when computing the correlation function, can produce errors at its output. Because of feedback loops these errors will propagate over time and will corrupt severly the time and the frequency estimations. Loss in the signal tracking process can be reported (see Fig. 4.2a when $p > 0.3$), forcing the receiver to restart the initial signal acquisition procedure. So it is increasingly important to deal with the impact of the faults when they appear in the correlation process. As mentioned earlier, any GPS receiver design contains at least four channel tracking modules. Each tracking channel module is composed of one complex product multiplication, 3 multiplications and 3 accumulations. To protect the correlation function against faults, we propose to apply the 4-RDSC for the complex product computation of the four channel tracking modules, then for each tracking module we apply the 3-RDSC to protect the Code Correlation process as illustrated in fig. 4.3. An error signal, S_{Ek} is generated for each tracking channel. $S_{Ek} = 1$ if more than one fault occurs in the 4-RDSC scheme or in the 3-RDSC scheme of the tracking channel number k. An error mitigation mechanism is activated in the channel a when $S_{Ek} = 1$. We propose two forms of error recovery techniques.

Feedback freezing loop (FFL) method The first method we propose is to freeze the feedback loops once we detect occurrence of faults. The idea is that it is less harmful to inject nothing in the feedback loop than a wrong value. Fig. 4.3 illustrates the configuration used in this method. When $S_{Ea} = 1$, feedback loops of the channel a are frozen (or blocked) until a new computation of the correlation function is achieved (i.e until the 10 next millisecond T_2). The corresponding Code and carrier generators will use the same codes and carrier that have been generated during the previous 10 ms period (T_0). With this approach we can guarantee that errors in the correlation process don't propagate over time.

Last Correct Value (LCV) method The second proposed method uses the last correct value of the integrator output to feed the feedback loops instead of the faulty value. Fig. 4.3 describes the configuration used for this method. If $\tilde{d}_P^a(l)$ is the result of the correlation function at time step l, if $S_{Ea} = 1$ at time step $l+1$ (i.e. $\tilde{d}_P^a(l+1)$ is faulty), the system uses the value $\tilde{d}_P^a(l)$ to be sent to the feedback loops instead of $\tilde{d}_P^a(l+1)$.

Comparison results: The effectiveness of the two mitigation techniques is proven by comparing the performance a faulty GPS receiver with a non-faulty (noisy-free) GPS receiver at two levels: the standard deviation between positions given by the faulty (noisy) and non-faulty (the noiseless) GPS receivers, $\sigma_X(p)$, as defined in (4.1), and the distribution of the error probability of the position, $f(d,p)$. $f(d,p)$ is defined as:

$$f(d,p) = Pr\left(\mid X - \tilde{X} \mid > d\right). \tag{4.2}$$

Given the probability of error injection, p, $f(d,p)$ reflect the probability that the noisy GPS receiver computes a position at a distance greater than d from the reference position. The methodology and the model of noise required to compute $\sigma_X(p)$ and $f(d,p)$ is the same described in Section 4.1. Fig. 4.4a compares $\sigma_X(p)$ of LCV, FFL and TMR.

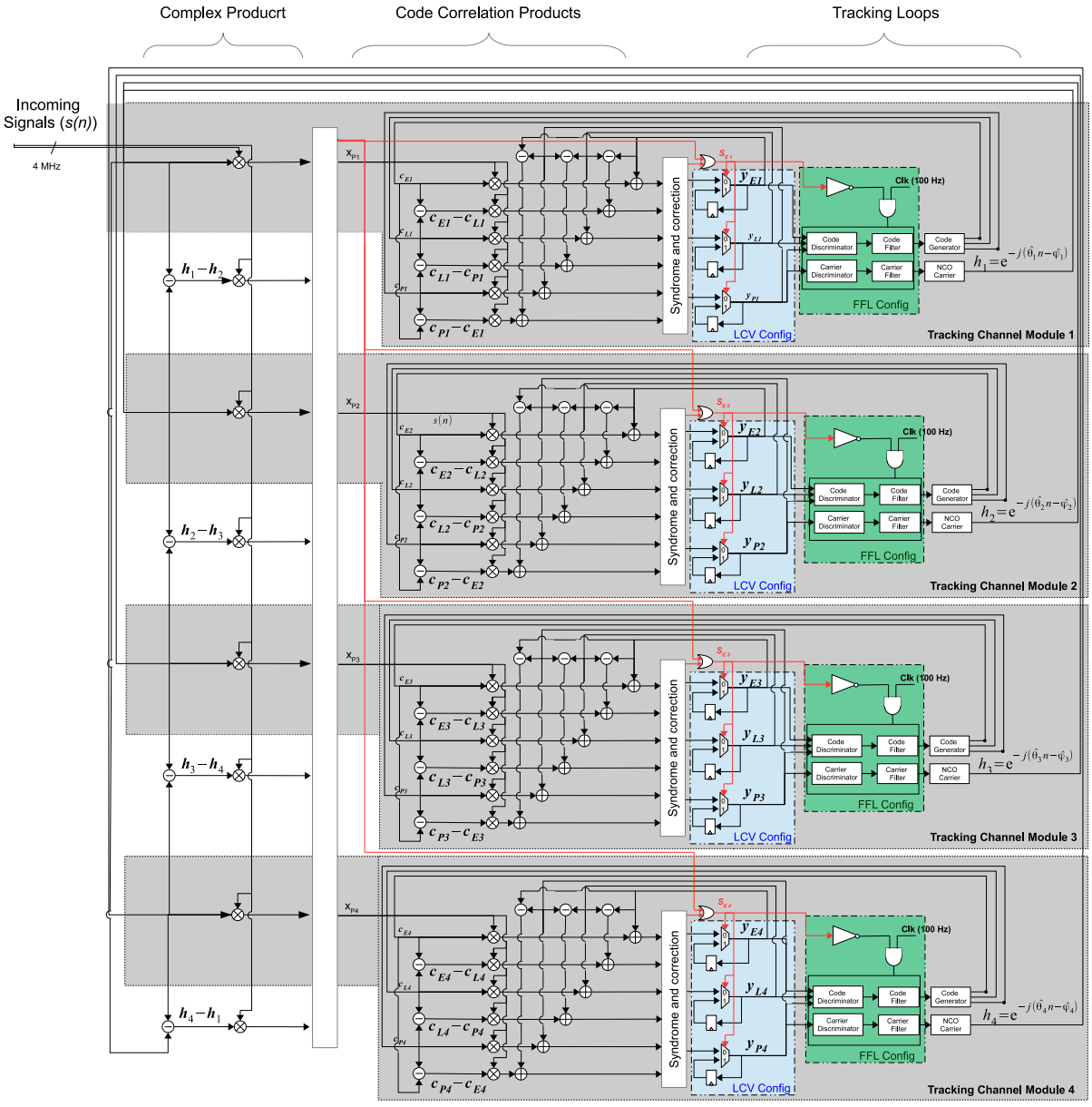
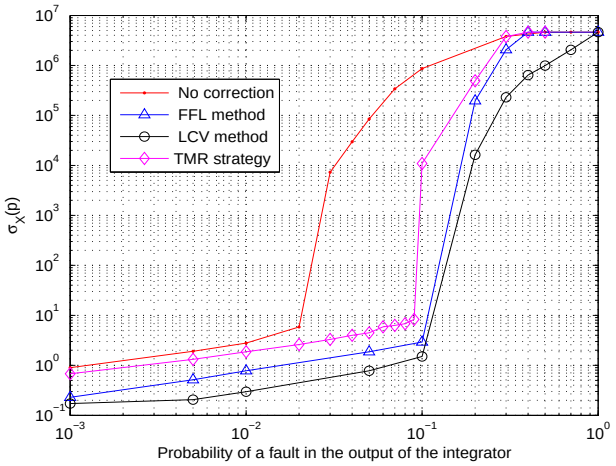
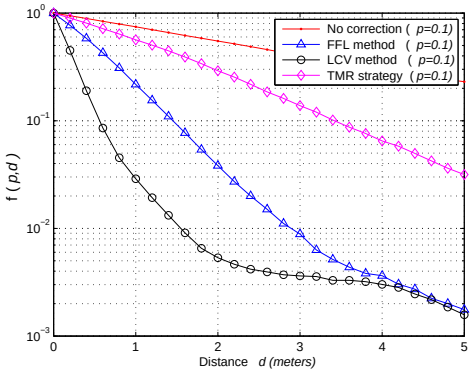


Figure 4.3: Configuration of proposed method. We add to the original tracking channel either the FFL or the LCV configurations

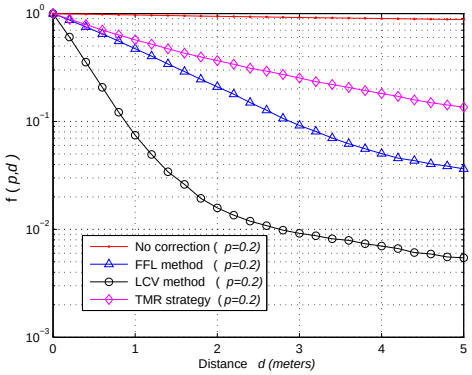
In this figure, $\sigma_X(p) = 4.65 \times 10^6$ m is equivalent to a loss of the GPS signal tracking. As we can see in Fig. 4.4a, for a probability of upsets up to $p = 0.4$, LCV method gives an error in the position that does not exceed on average 2 m. Thus the LCV methods permit to achieve an acceptable reliability even with high upset probability. Fig. 4.4b and Fig. 4.4c show the variation of this probability as a function of the distance between the noisy and the reference position when $p = 0.1$ and $p = 0.2$ respectively. For instance when $p = 0.1$, the probability of been 1 m far from the reference position goes from 0.6 to 0.009 using the LCV method. This part of the work has been published in [12] as a part of the DASIP 2015 conference.



(a) $\sigma_X = f(p)$



(b) $f(d, p)$ where $p = 0.1$



(c) $f(d, p)$ where $p = 0.1$

Figure 4.4: FFL and LCV Performances in term of $\sigma_X(p)$ and $f(d, p)$

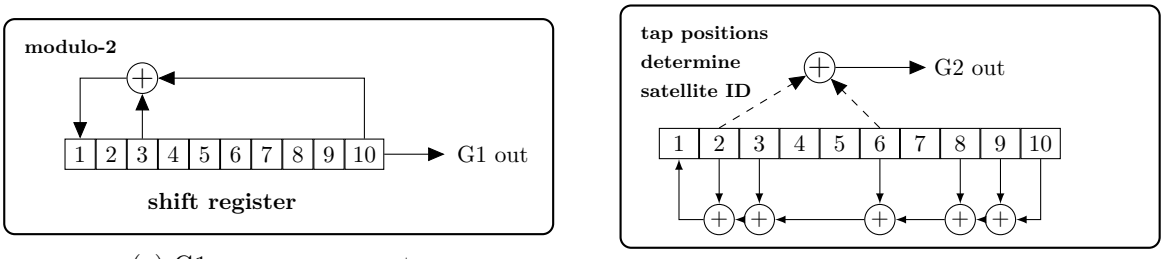
4.3 Reliable Code Generator

This section presents the first contribution of the thesis. It has been published in [11]. Since Chapter 2 generalises the first two methods presented here, the reader may find some

redundancies in Sections 4.3.2 and 4.3.3 with the analysis presented in Chapter 2.

4.3.1 The principle of GPS codes

GPS systems use a family of ranging signals called Coarse/Acquisition(C/A) codes belonging to the family of Gold pseudo random noise (PRN) sequences. The C/A codes are generated from the product of two 1,023-bit PRN sequences called G1 and G2 (see Sect. 3.4.3). Both G1 and G2 are generated by 10-stage Linear Feedback Shift Register (LFSR) modules [43]. Fig.4.5 shows the standard G1 and G2 LFSR generators. If the LFSR register states are disturbed, then the corresponding C/A code state becomes corrupted, resulting in loss of the GPS signal tracking. This problem can be mitigated if the LFSR state registers are protected. Five solutions have been proposed to make a system formed of four G1 registers more robust to errors (four is the minimum number of satellites needed to make distance measurements, so the minimum number of LFSR that are already exist in any GPS receiver design is 4). Performance of all the proposed solutions are evaluated by computing the mean time to failure (MTTF): larger MTTF implies a more reliable system. The hardware complexity has also been evaluated by calculating the number of equivalent NAND gates for each solution. Values stored in all registers, at the clock cycle t, will be represented by the matrix $R^{(t)} = (r_{i,j}^{(t)})$ with $1 \leq i \leq 4$ and $1 \leq j \leq 10$, where j is the j^{th} position of the i^{th} LFSR.



(a) G1 sequence generator

(b) G2 sequence generator

Figure 4.5: GPS codes generators

4.3.2 Protection with TMR

All registers, taps and sum operations are triplicated. A majority voter is placed between each set of registers and operators (see Fig.4.6). It is assumed that upset events in the replicated flip-flops are independent. Let p be the flip-flop upset probability per time step (i.e clock cycle) and $\mathbf{X_{k,t}}$ a random variable describing the upset state of the k^{th} bit in an LFSR at time step t. The stored value is either correct ($\mathbf{X_{k,t}} = 0$) or erroneous ($\mathbf{X_{k,t}} = 1$). Given no overall failure prior to time step $t-1$, $\Pr(\mathbf{X_{k,t-1}} = 1) = p$ for all k. After applying the majority vote operation, the probability of a correct state at the voter's output, q_v, is given by:

$$q_v = (1-p)^3 + 3p\,(1-p)^2\,. \tag{4.3}$$

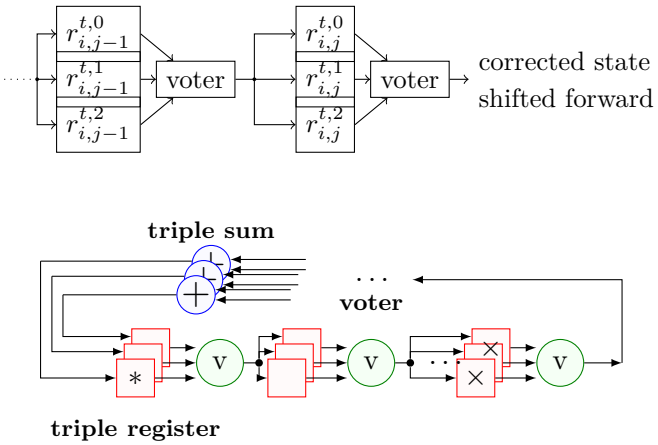


Figure 4.6: Shift register with TMR protection in each flip-flop, where $R^{t,x}_{i,j}$ denotes the x^{th} replica of the j^{th} flip-flop in the i^{th} LFSR of a G1 sequence generator group at the step t. Correctable error patterns are indicated by $*$, and uncorrected patterns by $\times$.

Now let p_c be the probability that the entire LFSR state is correct after applying majority vote operations on all flip-flops:

$$p_c = \prod_{k=1}^{10} q_v$$

Let P_{Fail} be the probability of an instantaneous failure in the complete system. This probability is expressed as

$$P_{\text{Fail}} = 1 - {p_c}^4. \tag{4.4}$$

Finally, we calculate $P_e(t)$, the probability that the failure of the complete system occurs at time step t:

$$P_e(t) = (1 - P_{\text{Fail}})^{(t-1)} P_{\text{Fail}}. \tag{4.5}$$

Since $P_e(t)$ has a geometric distribution, the mean of this distribution is well known and yields the MTTF:

$$\text{MTTF} \triangleq \sum_{t=1}^{\infty} t P_e(t) = \frac{1}{P_{\text{Fail}}}. \tag{4.6}$$

As an example, given a flip-flop upset probability $p = 10^{-3}$, we find $P_{\text{Fail}} = 1.2 \times 10^{-4}$ and the mean time to failure (MTTF) is equal to 8339 cycles. In the TMR solution the vote operation is done for every bit of every LFSR. To reduce the complexity, it is possible to perform voting in a single column as described in Fig. 4.7. Since the data is shifted in a circular pattern around the LFSR, any single error will eventually pass through the voter where it can be corrected. The LFSR has length ten, so a single error may persist in the system for at most ten clock cycles. One-column voting tends to decrease (i.e. worsen) the MTTF of TMR. In order to model the propagation of errors through the LFSR, we use the trellis model shown in Fig. 4.8, which assumes the voter is placed in column three. By choosing this position of the voter, we guarantee that every error in this system will pass by the voter and will be corrected in at most 10 clock cycles. Given that no system failure has occurred prior to time step t,

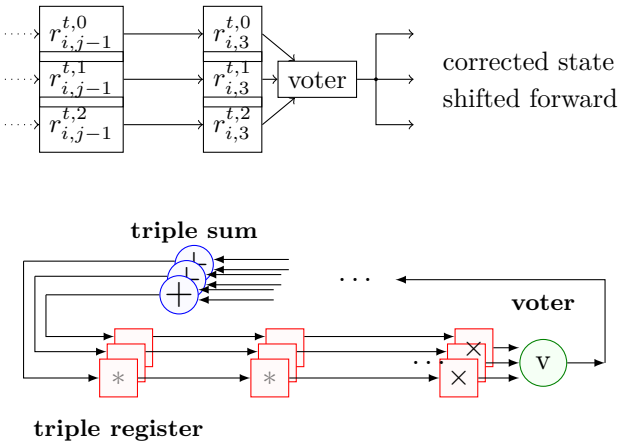


Figure 4.7: Shift register with TMR protection in the 3^{thd} flip-flop. Correctable error patterns are indicated by $*$ and uncorrected patterns by $\times$.

we only need to consider a trellis depth of ten stages, since a successful correction would have eliminated any errors in the voter's column at time $t-10$. Let $p_f(t-\tau)$ be the error probability in a flip-flop τ time-steps before it's data reaches the voter. Clearly $p\,(t-10)=0$ and for subsequent stages we have

$$p_f\,(t-\tau)=p\,(1-p_f\,(t-\tau-1))+(1-p)p_f\,(t-\tau-1)\,.$$

Iterating this calculation in the trellis model, we determine p_v, the probability of flip-flop error in the voting column, as the second element in the state vector P_v given by

$$P_v=\begin{bmatrix}1-p & p\\ p & 1-p\end{bmatrix}^{10}\times\begin{bmatrix}1\\0\end{bmatrix} \tag{4.7}$$

Using this model, we find the probability of a correct state at the one-column voter's output, q_v, by substituting p_v in place of p in (4.3). Then the instantaneous failure probability for the four-LFSR system is

$$P_{\text{Fail}}=1-q_v^4. \tag{4.8}$$

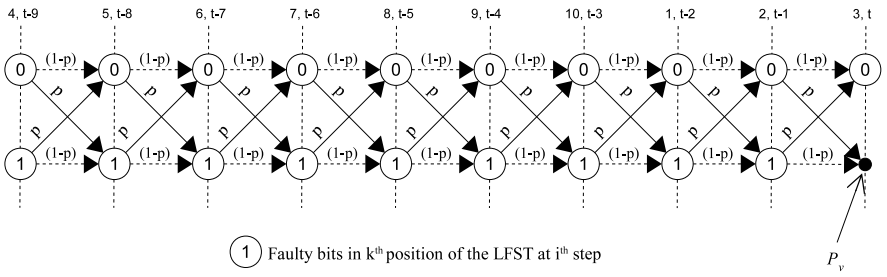


Figure 4.8: Trellis graph describing error propagation in the one-column error correction solution.

The MTTF, in this case and for $p = 10^{-3}$, is equal to 854 clock cycles, nearly ten times lower than TMR. The one-column approach trades reliability for complexity gains.

4.3.3 Protection with FT-Hamming

In this section, we applied the FT-Hamming method as introduced in Section. 2.2.1. The system contains in this case 4 original LFSRs and 3 redundant (or parity) LFSRs. The initial states of the parity LFSRs, represented by the matrix $C^{(0)}$, are obtained using a Hamming encoding operation:

$$\begin{pmatrix} R^{(0)} \\ C^{(0)} \end{pmatrix} \triangleq G^T \times R^{(0)},$$

where G^T is the transpose of the generator matrix G of the (7,4) Hamming code and $R^{(0)}$ represents initial states of the original four LFSRs. For this encoding we use the standard Hamming generator matrix given by

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

. Since the LFSR states are updated by a linear transformation at each time step, if no errors are generated in the system, then the parity LFSRs will preserve the Hamming code structure [40], i.e. at any time step t,

$$\mathrm{Q}^{(\mathrm{t})} \triangleq \begin{pmatrix} R^{(t)} \\ C^{(t)} \end{pmatrix} = \mathrm{G}^{\mathrm{T}} \times \mathrm{R}^{(\mathrm{t})}.$$

The system, defined by $Q^{(t)}$, is now a matrix with 7 rows and 10 columns $\mathrm{Q}^{(\mathrm{t})} = (Q_1^{(t)}, \cdots, R_{10}^{(t)})$, as shown in Fig. 4.9. By decoding all columns of $Q^{(t)}$, we are able to detect and correct errors present in each column decoded at a step t. We assume a syndrome decoding method, which is able to correct a single error. The system in this case fails once two errors or more are generated in the same column. The probability that a failure is not declared in one column is:

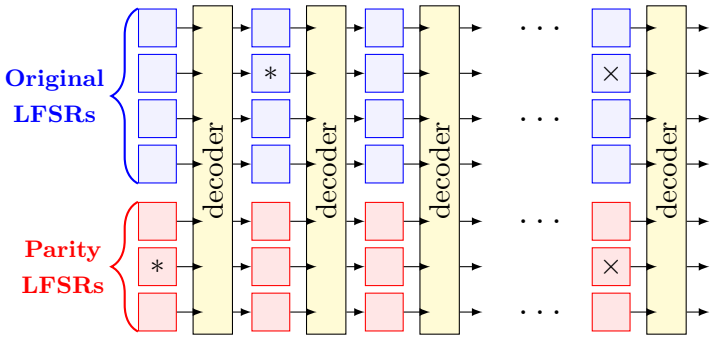
$$q_h = (1-p)^7 + 7p\,(1-p)^6\,. \tag{4.9}$$

The probability of an instantaneous failure in the system is:

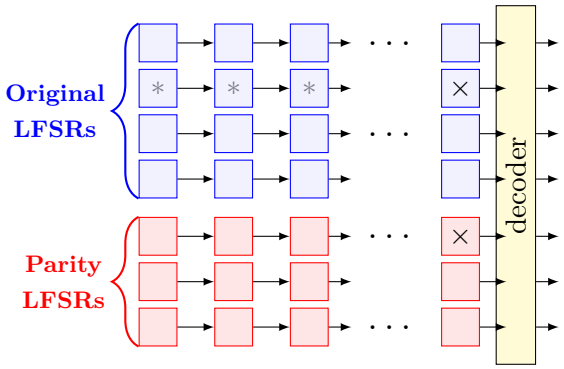
$$P_{\text{Fail}} = 1 - \prod_{k=1}^{10} q_h \tag{4.10}$$

Using this method, with $p = 10^{-3}$, the MTTF is equal to 4713 clock cycles.To reduce the complexity, decoding can be performed only in one column. To determine the MTTF for this method, we modify the method used for the one-column TMR method. Since Hamming decoding is performed for a single column only, the flip-flop error propagation is modelled by the trellis process in Fig. 4.8 and the probability of flip-flop error in the decoder's column is

p_v as given by (4.8). Since a system failure is produced by two or more errors among the



(a) The full Hamming solution



(b) One column Hamming solution

Figure 4.9: Shift registers with FT-Hamming protection. Correctable error patterns are indicated by $*$ and uncorrected patterns by $\times$.

flip-flops in the decoded column, we find that

$$P_{\text{Fail}} = 1 - \left[(1-p_v)^7 + 7p_v\,(1-p_v)^6\right]. \tag{4.11}$$

We find, with $p = 10^{-3}$, MTTF = 501. Once again the MTTF is made worse, with a possible gain in complexity.

4.3.4 Protection with parity-check error detection

In this section, we will modify the architecture of each LFSR by adding a parity flip-flop to each row. We define the row parity $v_i^{(t)}$ as the modulo-2 sum over all ten values in a single LFSR i at time step t, hence $v_i^{(t)} = \bigoplus_{j=1}^{10} r_{i,j}^{(t)}$. Given $r_{i,1}^{(t+1)} = r_{i,3}^{(t)} \oplus r_{i,10}^{(t)}$, the expression for

$v_i^{(t+1)}$ can be simplified as

$$v_i^{(t+1)} = \bigoplus_{j=1}^{10} r_{i,j}^{(t+1)} = (r_{i,3}^{(t)} \oplus r_{i,10}^{(t)}) \oplus (\bigoplus_{j=1}^{9} r_{i,j}^{(t)})$$
$$= r_{i,3}^{(t)} \oplus (\bigoplus_{j=1}^{10} r_{i,j}^{(t)}) = v_i^{(t)} \oplus r_{i,3}^{(t)}. \tag{4.12}$$

Based on this equation, we add the new parity flip-flop to every LFSR in the system, according to the architecture shown in Fig. 4.10. In order to detect errors in LFSR i, we independently compute the error as $e_h^{(t)}(i) = (\bigoplus_{j=1}^{10} r_{i,j}^{(t)}) \oplus v_i^{(t)}$. If $e_h^{(t)}(i) \neq 0$, an odd number of errors has occurred.

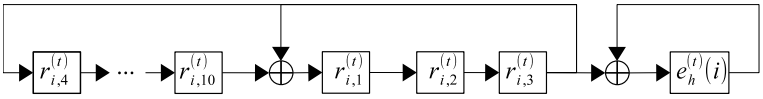


Figure 4.10: New architecture of G1 LFSR with row parity.

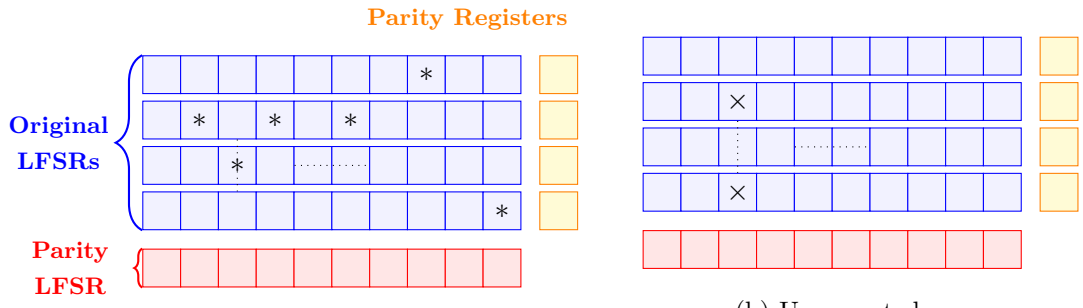
The row-parity solution based on Fig. 4.10 is sufficient to detect an odd number of errors but cannot identify their positions. In order to detect the location of errors, a new LFSR is added to the system comprised of the column-parity, as shown in Fig. 4.11, so that

$$r_{5,j}^{(t)} = \bigoplus_{i=1}^{4} r_{i,j}^{(t)}. \tag{4.13}$$

Since the parity LFSR is identical to the others, it only needs to be initialized with (4.13) at a single time t_0. Under error-free operation, the condition described by (4.13) is preserved in the LFSR state evolution. The new LFSR system consists of five LFSRs: The fifth row is the parity LFSR and the eleventh column is the set of parity-check bits corresponding to Fig. 4.10. The column and row errors are detected as $e_v^{(t)}(j) = (\bigoplus_{i=1}^{4} r_{i,j}^{(t)}) \oplus r_{5,j}^{(t)}$ and $e_h^{(t)}(i)$, respectively. If there is a row l where $e_h^{(t)}(l) \neq 0$, then we should find all positions q for which $e_v^{(t)}(q) \neq 0$. Errors are corrected by flipping the bit in these positions q using the XOR function as shown in the architecture in Fig. 4.12. Corrected register values are denoted as $\bar{r}$ in this figure. With this method, errors can be corrected if there is no even number of errors that appear in one row or one column. The probability of a system failure in this case is approximately,

$$P_{\text{Fail}} = 1 - \Big((1-p)^{55} + 55p(1-p)^{54} \Big). \tag{4.14}$$

Continuing the example from the previous sections with $p = 10^{-3}$, we find that $P_{\text{Fail}} = 0.0014$ and the MTTF is 697. This is below the performance of TMR. Due to its simplicity, the parity method provides good complexity gain.



(a) Correctable errors

(b) Uncorrected errors

Figure 4.11: The Parity checking method configuration; the G1 LFSRs are arranged in a matrix where each row is an LFSR, taps, feed-backs and modulo-2-sums are not shown. Any single error causes parity violation in both the row and column, allowing it to be corrected. Two errors cancel if they occur in the same row or column, causing failure.

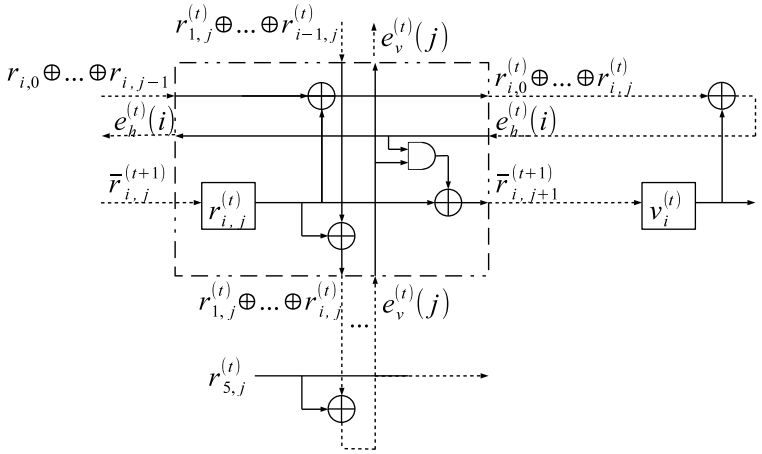


Figure 4.12: Schematic for parity-based correction of the j^{th} flip-flop in the i^{th} LFSR.

Method	FF	XOR2	AND2	AND3	Maj	NAND eq.
TMR	120	12	0	0	40	242.5
TMR (one col.)	120	12	0	0	4	220
Hamming	70	167	0	70	0	235.375
Hamming (one col)	70	23	0	7	0	139.5
Parity-Check	55	159	55	0	0	209.375

Table 4.1: Synthesis results for each method, showing total module counts and the equivalent number of NAND gates per LFSR

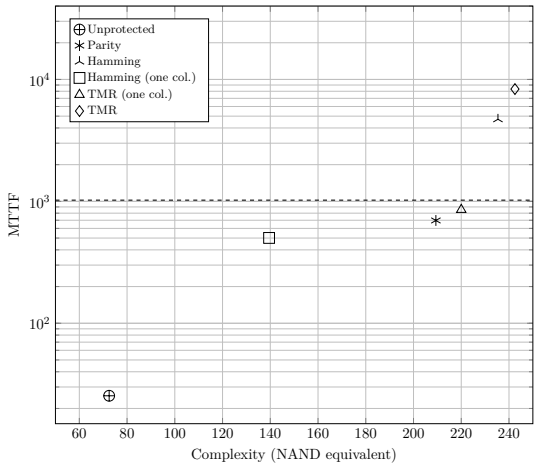


Figure 4.13: Complexity and performance of the proposed methods with $p = 10^{-3}$. The dashed line indicates the PRN length.

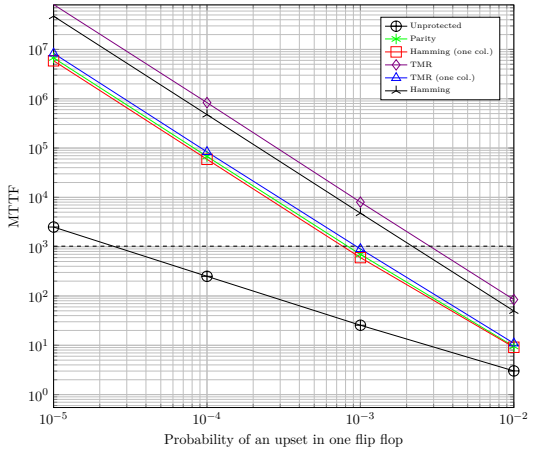


Figure 4.14: Performance as a function of the upset probability p. The dashed line indicates the PRN length.

4.3.5 Evaluation

MTTF was solved for each method using the analytically techniques and Monte-Carlo simulations. The Logic synthesis of each solution was also done to express complexity in terms of equivalent NAND gates. Fig. 4.14 compares the MTTF of all the solutions for different values of the upset probability p. GPS Gold sequences have a period of 1023, so LFSRs can be reset to a known state each 1023 chips. For an upset probability, $p = 10^{-3}$, the full Hamming method's MTTF of 4713 cycles is enough for the GPS application since all registers are reinitialised to ones every 1023 cycles. When $p = 10^{-4}$, the Hamming one column satisfy the condition of MTTF $>$ 1023 cycles.

Table 4.1 list the associated equivalent NAND complexity for each solution. Given a probability of an upset in one flip flop, $p = 10^{-3}$, results from the analyse of the complexity

of different method are summarised in Fig. 4.13. The best choice of protection depends on the technology's upset probability. For a high upset probability, the full Hamming or TMR method may be necessary to achieve an acceptable MTTF. For a small upset probability, the the Hamming (one col.) method provides the required MTTF, but reduces the gate complexity by 42% compared to full TMR. To conclude this section, it is necessary to mention all the proposed solutions can be extended and generalised for any type of LFSR register. This work has been published in [11] as a part of the 58^{th} IEEE International Midwest Symposium on Circuits and Systems (MWSCAS 2015) conference. It is important to note that these methods will not be integrated in the robust version of the GPS tracking modules designed in the RELIASIC project. The existing version of the GPS application contains ROM that saves the whole PRN sequences.

4.4 Reliability of the carrier discriminator

4.4.1 The linearised representation of the carrier tracking loop

The carrier tracking loop, described in Sec. 3.4.1, is linear and can be represented by a system of z-domain transfer functions where the system is updated each 10 ms. A linearised representation of the loop is quite useful as it facilitates the estimation of loop stability and tracking performance. The linearised model of the carrier tracking module is represented in Fig. 4.15. The signal carrier phase and frequency are represented by respectively θ and ω. The functions NCO(z) and D(z) represent the z-transform of the numerically controlled oscillator and the frequency discriminator respectively. In fact, the numerically controlled oscillator is a digital integrator and its transfer function is represented by

$$\mathrm{NCO}(z) = \frac{T_{int}}{z-1}. \tag{4.15}$$

The frequency estimate in the presence of errors can be approximated by a gain plus an

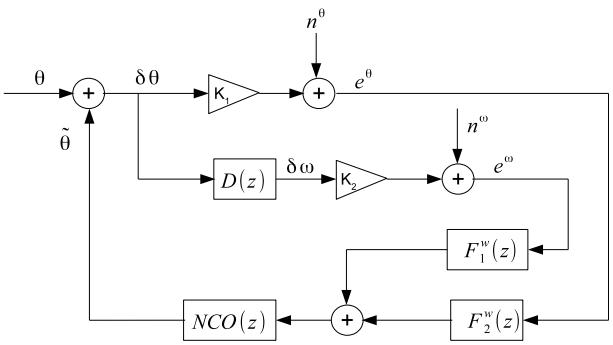


Figure 4.15: Structure of the GPS carrier tracking loop

independent corrupting noise

$$e^{\omega} = K_2\ \delta\omega + n^{\omega}. \tag{4.16}$$

The phase estimate can also be defined as

$$e^{\theta} = K_1 \ \delta\theta + n^{\theta}. \tag{4.17}$$

The z transform of the frequency discriminator function is defined by

$$D(z) = \frac{z-1}{z \ T_{int}} \tag{4.18}$$

Scalars K_1 and K_2 represent the gain of the phase and the frequency discriminators. The authors of [45] demonstrate that values of these parameters depend on the discriminator function and the signal noise ratio (SNR) of incoming signals. In this work, ATAN discriminators are considered where $K_1 = K_2 = 1$ for $SNR > 20dB$.
The carrier filter can be linearised and defined as follow

$$F(z) = \frac{A_0 \ z}{z-1} \ \ E^{\omega}(z) + \left(B_0 + \frac{B_1 \ z}{z-1}\right) E^{\theta}(z). \tag{4.19}$$

where $E^{\omega}(z)$ and $E^{\theta}(z)$ represents the z transform of the frequency error, e^{ω} and the phase error and e^{θ} respectively, $A_0 = \omega_{0f} T_{int}$, $B_0 = a_2 \ \omega_{0p}$ and $B_1 = \omega_{0p}^2 T_{int}$.
The transfer function between the frequency noise and the tracking phase error, $H_n^{\omega}(z)$ and the transfer function between the phase noise and the tracking phase error, $H_n^{\theta}(z)$, are defined by

$$H_n^{\omega}(z) = \frac{\Delta\Omega}{N^{\omega}} = \frac{\mathrm{NCO}(z) \ F_1^{\omega}(z)}{1 + \mathrm{NCO}(z) F_1^{\omega}(z) D(z) + \mathrm{NCO}(z) F_2^{\theta}(z)} \tag{4.20}$$

$$H_n^{\theta}(z) = \frac{\Delta\Theta}{N^{\theta}} = \frac{\mathrm{NCO}(z) \ F_2^{\omega}(z)}{1 + \mathrm{NCO}(z) F_1^{\omega}(z) D(z) + \mathrm{NCO}(z) F_2^{\theta}(z)} \tag{4.21}$$

where upper case symbols represents the z-transform of the corresponding lower case time series, $F_1^{\omega}(z) = \frac{A_0 \ z}{z-1}$ and $F_2^{\theta}(z) = B_0 + \frac{B_1 \ z}{z-1}$.

4.4.2 The proposed method

When designing the carrier filter there is a trade-off between noise resistance and response to dynamics. Narrow bandwidth filters are more resistant to noise, that makes them suitable for moderate jamming (low SNRs) environments [46]. Wide bandwidth tracking loops are more responsive to dynamics. Thus, carrier filter bandwidth requirements for GPS receivers are conflicting. For a ground GPS receiver, values of the phase and frequency filters bandwidths are both equal to 4 Hz [44]. When the Carrier discriminator output is corrupted due to PVT variations, distance measurements in the GPS receiver are disturbed and this leads to a faulty localization of the receiver. To deal with errors produced in the carrier discriminator we propose to modify the optimal filter bandwidth values.

4.4.3 The standard deviation $\sigma_X(p)$

The methodology used to compute the $\sigma_X(p)$ metric is similar to one introduced in Section. 4.1. The model of noise (n^ω and/or n^θ) used in this case is simple. For each active tracking module, every output of the frequency and/or the phase discriminators is assumed to be exact with a probability $(1-p)$ or to be faulty with a probability p. In case of faulty result, a random value in $[-b, b]$ is uniformly added to the output of a discriminator ($b = 25$ Hz and $b = 0.25$ rad when the frequency and/or the phase discriminators are faulty. The value of b is chosen based on (3.7) and (3.8)).

4.4.4 The error induced Tracking Error Variance

To evaluate the error induced frequency tracking error, the Power Spectral Density (PSD) of the noise N^ω and N^θ must be found. N^ω is assume to be a uniform random variable in $[-b, b]$, thus, its PSD is defined as:

$$S_n^\omega = (2b+1)(b+1)\frac{p}{6}. \tag{4.22}$$

This power spectral density is used in conjunction with (4.20) to find the tracking error variance when the frequency discriminator is faulty. The variance of $\delta\omega$, denoted $\delta^2_{\delta\omega}$, is given by

$$\sigma^2_{\delta\omega}(p) = \frac{1}{2\pi T}\int_{-\pi}^{\pi} S_n^\omega(\omega)|H_n^\omega(\omega)|^2\,\mathrm{d}\omega. \tag{4.23}$$

This paper will focus on the difference in the tracking error variance between the noiseless and the noisy GPS receivers, σ_p^2.

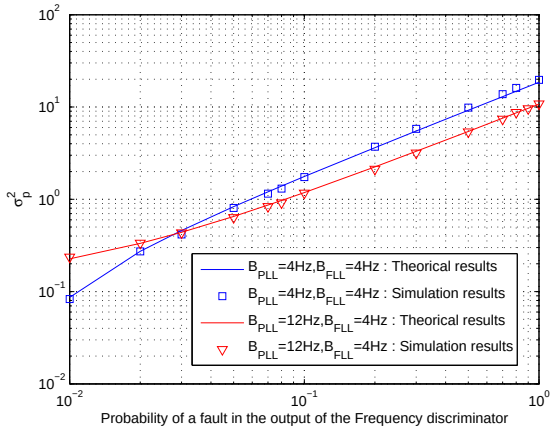
$$\sigma_p^2 = \sigma^2_{\delta\omega}(p=0) - \sigma^2_{\delta\omega}(p). \tag{4.24}$$

4.4.5 Performances and results

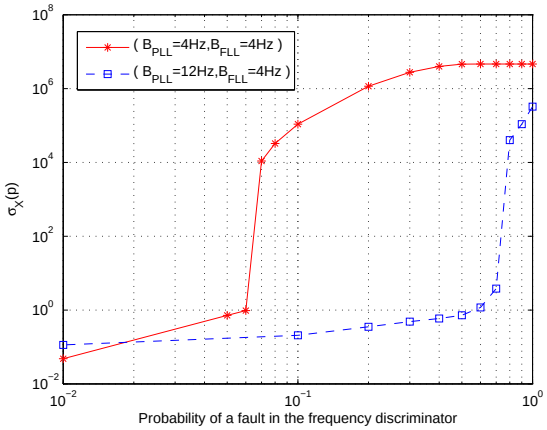
Two metrics have been considered to evaluate the proposed method: the standard deviation between positions given by the faulty (noisy) and non-faulty (the noiseless) GPS receivers, $\sigma_X(p)$, as defined in (4.1), and the error induced Tracking Error Variance, σ_p^2, defined in (4.24).

Using expressions (4.20) and (4.24), the tracking error variance of the carrier loop, when errors appear in the frequency discriminator (i.e $N^\theta = 0$), can be estimated for different values of carrier filter bandwidths. Both simulated results (markers) and theoretical estimates (solid lines) are shown in Fig. 4.16a. The relationship between the tracking error variance and the probability of a fault is linear ($\alpha p + \beta$). For a high upset probability, the tracking error variance is decreasing for high value of PLL bandwidth filters ($B_{PLL} = 12$ Hz). Fig. 4.16b presents estimated $\sigma_X(p)$ from Monte-Carlo simulations when only the frequency discriminator is faulty. For instance, for a probability of upsets up to $p = 0.6$, the error in position does not

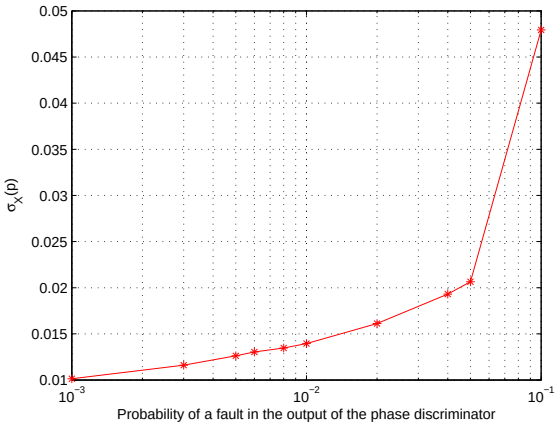
exceed 1 meter by using 12 Hz as the bandwidth in the PLL filter. On the other hand, when $N^{\omega} = 0$ (only the phase discriminator is faulty), $\sigma_X(p)$ does not exceed 0.1 meters for high probability of upsets as shown in Fig. 4.16c. This result is explained by the fact that the FLL is more sensitive to dynamic stress than the PLL.



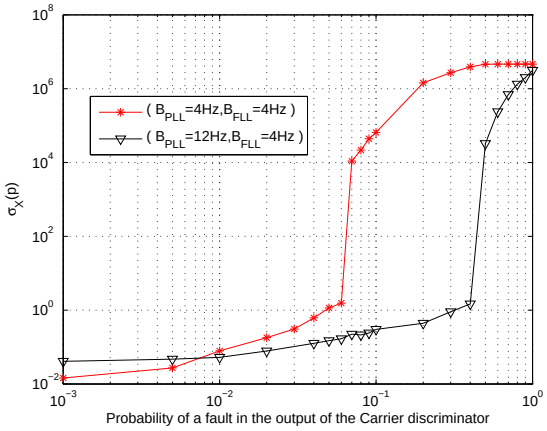
(a) Theoretical and simulated tracking error variance for a range of the probability of a fault



(b) $\sigma_X = f(p)$ when only the frequency discriminator is faulty



(c) $\sigma_X = f(p)$ when only the phase discriminator is faulty



(d) $\sigma_X(p)$ when the carrier discriminator is faulty

Figure 4.16: Performance results of tuning filters bandwidths; $\sigma_X(p) = 4.65 \times 10^6$ m (which represents the Cartesian coordinate of the reference position) is equivalent to a loss of the GPS signal tracking.

Finally we propose to evaluate the proposed method when $N^{\omega} \neq 0$ and $N^{\theta} \neq 0$. As shown in Fig. 4.16d, when the carrier discriminator is faulty, the GPS signal tracking is lost for a probability of upset $p > 0.06$ in the case of the reference GPS receiver (the reference GPS receiver does not support so more than 6 % of errors). However, $\sigma_X(p)$ goes from 6.5×10^4 m to 1.4 m for a probability of upset $p = 0.4$ (so with 40 % of upset errors, $\sigma_X(p)$ does not exceed 2 m) by increasing the PLL bandwidth from 4 to 12 Hz. To illustrate all these results, positions given by the noiseless and the faulty receivers are compared in Fig. 4.17. In this

figure, the noiseless position $X(t)$ is shown with color orange. We represent, then with the red color, the faulty position of the noisy GPS receiver when the fault tolerant technique was not used. Finally, the position given by the faulty GPS receiver, after adding the fault tolerant mechanism to the design, is appeared with the blue color. This work was published in [12] as a part of the SIPS 2016 Workshop.

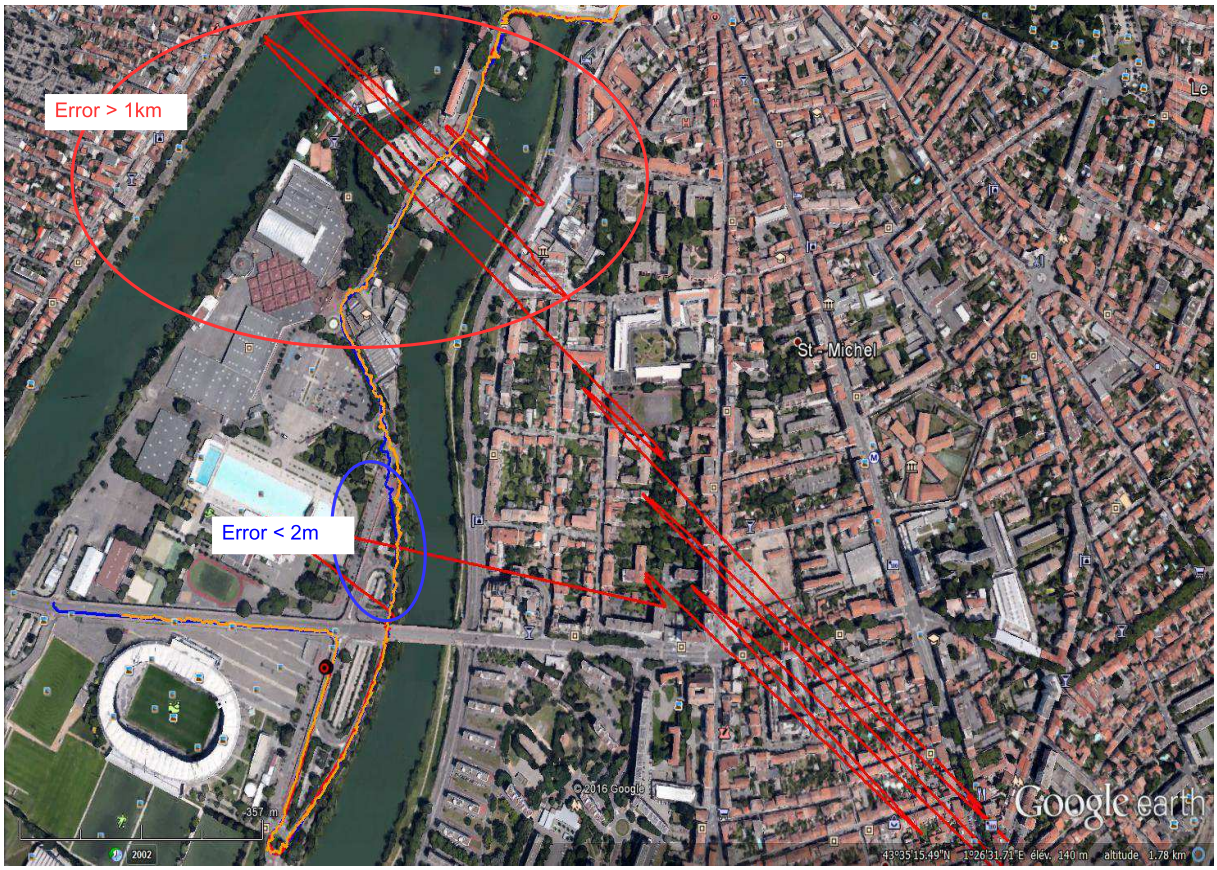


Figure 4.17: Position given by a noiseless GPS receiver (color orange), by a faulty GPS receiver when $B_{PLL} = 4Hz, B_{FLL} = 4Hz, p = 0.1$ (color red), a faulty GPS receiver when $B_{PLL} = 12Hz, B_{FLL} = 4Hz, p = 0.3$ (color blue) displayed using Google earth.

4.5 Reliable NCO carrier Generator

GPS receivers use Numerically controlled oscillators (NCO) to produce a local copy of the carrier offset (due to Doppler offset) of incoming satellite signals. Fig. 3.6 shows a block diagram of the carrier loop NCO and its sine and cosine mapping functions. The NCO carrier receives at its input the estimated Doppler offset value δ. The value of δ remains constant during each 10 ms (tracking loops are operating at 100 Hz). To improve the robustness of the NCO carrier, we proposed in [13] two fault tolerant techniques: the Rendez-vous Checking Method (RvC) and Time Freezing Method (TF).

4.5.1 The Rendez-vous Checking method (RvC)

If we consider $\hat{\theta}$ the output of the NCO carrier, $\hat{\theta}$ is produced each 0.25 µs (i.e at 4 MHz) and is defined as

$$\forall n \qquad \hat{\theta}(n) = \hat{\theta}\,(n-1) + \delta\,(\lfloor n/N_s \rfloor), \tag{4.25}$$

where $\lfloor x \rfloor$ is the floor function of x and N_s is total number of samples generated by the NCO in 10 ms ($N_s = 40000$ in the GPS context). Thus,

$$\forall \ell \in \mathbb{N} \quad \theta((\ell+1)\,N_s) = \theta(\ell\,N_s) + N_s\,\delta(\ell) \tag{4.26}$$

Based on (4.26), we add to the original architecture (i.e the adder and the register) a correction module composed of a multiplier and an adder (see Fig. 4.18. At the end of each 10 ms, the correction module produces a new output (ψ_c) by substituting θ_c in place of θ in (4.26). The Rendez-vous Checking method will consist on a meeting (Rendez-vous) organised at the end of each 10 ms to check the coherence of values between θ_c and θ_o (where θ_o is the output of the original module given by substituting θ_o in place of θ in (4.25)). Once an incoherence ($\theta_o \neq \theta_c$) is declared, it is θ_c that will be used later. Note that 40 000 cycles are suitable to compute θ_c. There is more than enough time redundancy to guarantee that the result θ_c can be correct. In case where θ_o and θ_c are not consistent, it means that the integration process that generates correlation output is probably faulty. In that case, FFL and LCV techniques defined in section 4.2 can be applied. Fig. 4.19 illustrates how we are able to correct errors

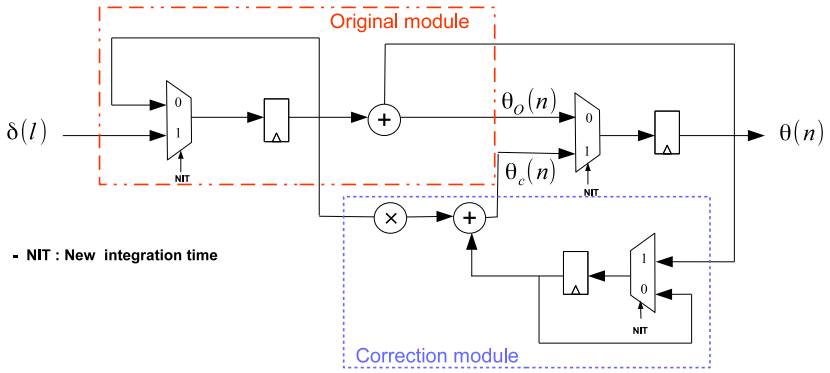


Figure 4.18: Rendez-vous checking method's configuration

occurring when the NCO is operating. However, when an error occurs when the correction module is operating, an uncorrectable failure results in the system.

4.5.2 Time Freezing Method (TF)

In this method we use the Double with comparison technique introduced in Sec. 1.2.3.1 to detect occurrence of faults. Once an incoherence between the output of two replicas is detected, the correct previous output of the NCO (i.e $\theta(n-1)$) is used to feedback adders as shown in Fig. 4.20.

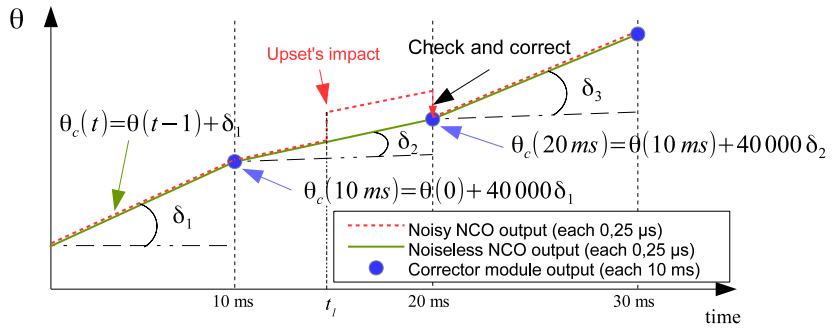


Figure 4.19: Correction of errors in RvC method; when an error occurs at time step t1, it can be corrected at time at 20 ms thanks to the correct output of the correction module.

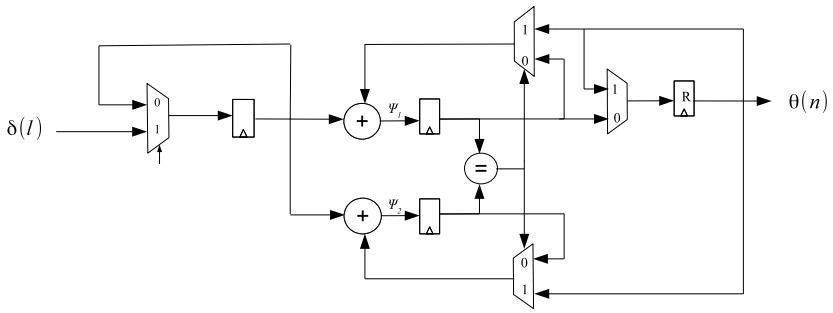


Figure 4.20: Time Freezing Method's Configuration

4.5.3 Comparison results

The effectiveness of the two mitigation techniques is proven by comparing the performance a faulty GPS receiver with a non-faulty (noisy-free) GPS receiver. The standard deviation between positions given by the faulty (noisy) and non-faulty (the noiseless) GPS receivers, $\sigma_X(p)$, as defined in (4.1) is evaluated for each solution and compared to the TMR 1 voter solution. Both the original modules and the corrections modules are assumed to be faulty in this analysis. As in the previous sections, the model of noise used in this comparison is very simple. For each active tracking module, every output of the NCO module is assumed to be exact with a probability $(1-p)$ or to be faulty with a probability p. In case of faulty result, a random value is uniformly drawn between 0 and 2π to replace the exact value. Fig. 4.21 compares $\sigma_X(p)$ of different methods. As we can see in this figure, the RvC method is effective for low or medium error probability ($\sigma_X(p) < 10\,\text{m}$ for $p < 10^{-3}$) but it loses its efficiency for higher error probability. For high error rates, the Time Freezing Method demonstrates it interest: it is more efficient than the TMR one voter since the impact of two faulty modules is limited by the injecting the previous NCO output instead of the wrong computed value. For example, for a probability of upsets up to $p = 10^{-2}$, the TF method gives an error of the position of about 1 meters. More details about this work can be found in [13].

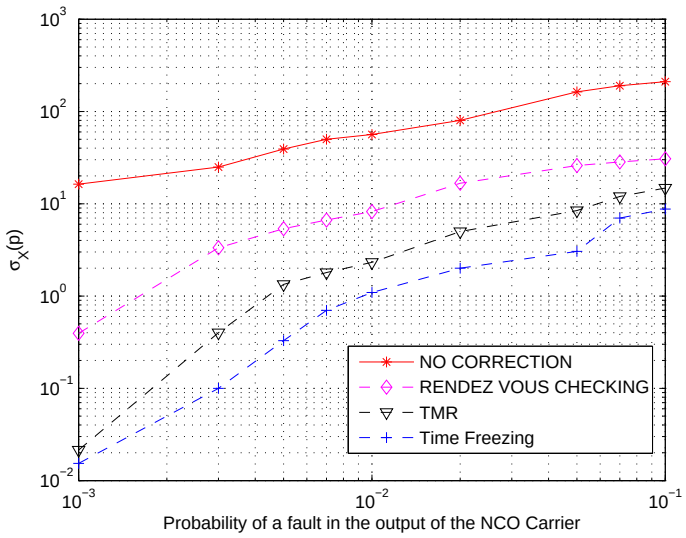


Figure 4.21: The performances of the Rvc, TF and the TMR solutions in term of $\sigma_X(p)$.

4.6 Summary

In this chapter, we first briefly gave informations about the GPS system and the role of tracking channel modules in the GPS navigation process. Then, we presented an analyse of the robustness of the GPS tracking channel modules and selected the elements of the architecture that require a high level reliability effort. After that, we proposed techniques that can improve the reliability each element of the tracking channel module in order to build a robust version of the GPS tracking process tolerant to faults to PVT variation coupled with the CMOS technology evolution.

We present in the following chapter the GPS Emulation platform: an emulation hardware environment designed by the ISAE to analyse the GPS algorithm and will be used, then, in the RELIASIC project and the thesis to 1/ integrate fault tolerant techniques described in this section in the standard version, and, 2/ prepare the simulation environment to explore the ASIC (containing the standard and the resilient versions of the GPS tracking process) once it be designed.

Chapter 5

The GPS Emulation platform

Contents

In the last chapter, several fault tolerant techniques were proposed to improve the reliability of the GPS tracking channels against faults. As a part of the RELIASIC project, an ASIC will be designed using the 28 nm technology to test the proposed techniques in a real design and validate by experiments and measurements the performances in term of robustness and power consumption. The ASIC will contain two versions of the GPS tracking channels: the standard version and the robust (fault tolerant) version.

In order to prepare the design of the ASIC and the experimental environment later, we used an existing GPS emulation platform that was developed by the ISAE [44]. The platform contains the standard version of the whole GPS digital system. It involves a user interface and a board that contain the processor and the hardware description of the GPS algorithm. Sec. 5.1 will detail the architecture of the platform. During the thesis, engineering works were elaborated to adapt the platform for the context of the RELIASIC project. Related contributions are presented in Sec. 5.2. They include:

1. The modification of some part of the tracking algorithm (the discriminators and filters functions). Then, the sensitive clock of the tracking algorithm was changed from 100 MHz to 4 MHz which has driven to some transformations on the design of the GPS tracking channels. Sec. 5.2.1 will detail this contribution.

2. The separation of the GPS tracking channels in a different FPGA board that will be replaced by the ASIC later. More information will be given in Sec. 5.2.2.

3. The improvement of the GPS platform observability: More signals are triggered for the analysis task after. Sec. 5.2.3 summarises this contribution.

4. Finally, the inclusion of the robust version of the GPS tracking channels illustrated in Sec. 5.2.4.

5.1 The GPS Emulation platform

5.1.1 General description

The GPS Emulation platform was developed by the ISAE to study GPS system and improve navigation algorithms. The platform environment is presented in Fig. 5.1a. It can be split into two main parts: the user software interface and the GPS receiver board. The user software interface represents the space from where the board is commanded and controlled. It is written by the Matlab software and offer various functionalities: program the board, launch multiples simulations, display GPS tracking measures evolution over time, launch Google Earth software to display the trajectory of the user.

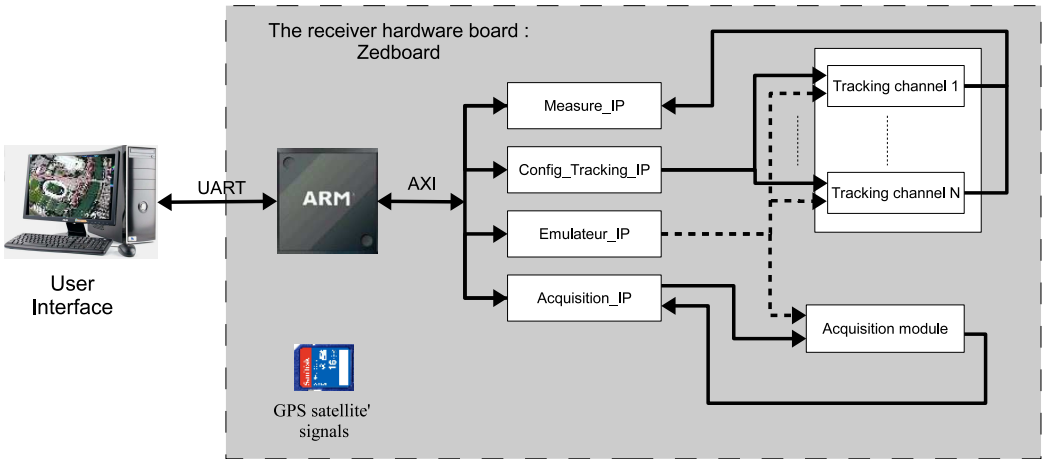
The GPS receiver board is the Xilinx Zynq-7000 Zedboard[1]. It contains the hardware description of the GPS digital algorithm and the processor (ARM Cortex^{TM}-A9) that will manage the communication between the user software interface and the GPS digital algorithm. Commands and data sent by the user interface are routed to the ZedBoard via the USB-UART Zedboard feature. The communication between each part of the Zedboard is governed by the Advanced eXtensible Interface (AXI) Interconnect protocol. More details about the Xilinx AXI protocol are given in Appendix A.2. Signal received from more than 4 GPS satellites, over a significant period of time, is stored in a file in the SD card in the ZedBoard. It is transferred to the RAM before lunching a simulation for the first time.

5.1.2 Architecture of the platform

Fig. 5.1b describes the complete GPS Emulation platform diagram. Four main functionalities can be ordered by the user software interface to the ZedBoard: Start Emulation (START_EMUL), Launch the acquisition process (ACQUISITION), configure a tracking channel and initialise it (INIT_CHANNEL) and finally trigger the measure process to collect channel tracking measures (TRIG_MEASURE). One Matlab command and one board acknowledgement messages are considered for each functionality (see Table. 5.1); Via the USB-UART feature, the user software interface sent the command message and input signals

[1]The ZedBoard is a low-cost development board for the all programmable SoC (AP SoC). It takes advantage of the Zynq-7000 AP SoCs tightly coupled ARM processing system and 7-series programmable logic to create powerful designs. Target applications include video processing, motor control, software acceleration, Linux/Android/RTOS development, embedded ARM processing and general Zynq-7000 AP SoC prototyping. The ZedBoard kit is supported by the www.zedboard.org community website where users can collaborate with other engineers also working on Zynq designs [47].

(a) The platform environment



(b) The platform diagram

Figure 5.1: A representation of the GPS emulation platform

to order the processing of a given functionality, the processor answers by sending back the corresponding acknowledgement message and results when the processing of the functionality is finished. In the ZedBoard, four user IP cores exists to make the interface between the processor and the GPS digital algorithm modules: Emulator_IP, Acquistion_IP, Configure_IP and Measure_IP. All IP cores communicate with each other using the AXI4 protocol and are sensitive to the clock of the processor (= 100 MHz). The reader is invited to Appendix A.3 for more details about the user IP cores. The GPS digital algorithm modules (Acquisition module and tracking channel modules) are also sensitive to the 100 MHz clock.

Symbol	Matlab Commands	Symbol	Board Acknowledgement
C0	ID_CMD_BOARD_CONNECT	A0	ID _ACK_BOARD_CONNECT
C1	ID_CMD_BOARD_INIT	A1	ID_ACK_BOARD_READY
C2	ID_CMD_START_EMUL	A2	ID_ACK_START_EMUL
C3	ID_CMD_STOP_RX	A3	ID_ACK_STOP_RX
C4	ID_CMD_ACQUISITION	A4	ID_ACK_ACQUISITION
C5	ID_CMD_INIT_CHANNEL	A5	ID_ACK_INIT_CHANNEL
C6	ID_CMD_TRIG_MEASURE	A6	ID_ACK_TRIG_MEASURE

Table 5.1: Principle Matlab Commands and Board Acknowledgements for the GPS platform

5.1.3 Simulation flow inside the platform

(0) Run the Matlab user interface: The Matlab sends the command "C0" to test the connexion with the board and wait until the processor in the board sends back the acknowledgement "A0" to start the GPS simulation, otherwise the message "Communication with Xilinx board not established" is displayed on the screen.

(1) If the connection with the board is established, Matlab sends the command "C1" to order the relocation of GPS satellite's signals from SD card to the RAM of the board. The processor informs Matlab when the operation is finished by sending back the acknowledgement "A1" and turns in the Idle mode to wait another command from Matlab.

(2) Receiving "A1", the Matlab commands the board to begin the emulation via "C2". The processor start the emulation task: GPS satellite's data are transferred from the RAM to the digital GPS algorithm circuit (the acquisition module and GPS tracking channel modules) thanks to the Emulator_IP.

(3) The board sends back the message "A2". The Matlab order then the beginning of the acquisition process of all the 32 GPS satellites in serial via the Command "C4": one satellite by one. The acquisition results after each acquisition are returned to Matlab in serial by blocks of 32 bits following the message "A4".

(4) When Matlab receives the acquisition results of the last GPS satellite, the eight GPS satellites with the best high Signal to Noise Ratio (SNR) are chosen by Matlab to be tracked. Let consider that $sat_1, \ldots sat_8$ are the GPS satellites to be tracked.

(5) Matlab orders another time an acquisition for sat_1 by sending "C4" and parameters required by the acquisition function (the SV PRN number...). This acquisition serves to refine the previous acquisition results. The processor informs Matlab when the acquisition is finished and sends back acquisition results following the message "A4". Then, Matlab order the ARM to configure the first tracking channel module with tracking parameters signals sent following the command "C5". The first tracking channel configured and the tracking process is initiated by the processor thanks to the Configure_IP module.

(6) (5) is repeated until the rest of the tracking channel modules in the design are configured to track $sat_2 \ldots sat_8$.

(7) During the tracking process, measures from tracking channels are sent periodically to Matlab to calculate the GPS position; Matlab sends the trigger signal "C6" each time to request measures of tracking channels from the board, the process uses the Measure_IP to send back measures following the message "A5".

(8) The simulation continues until Matlab commands to stop it by sending the interrupting message "C3".

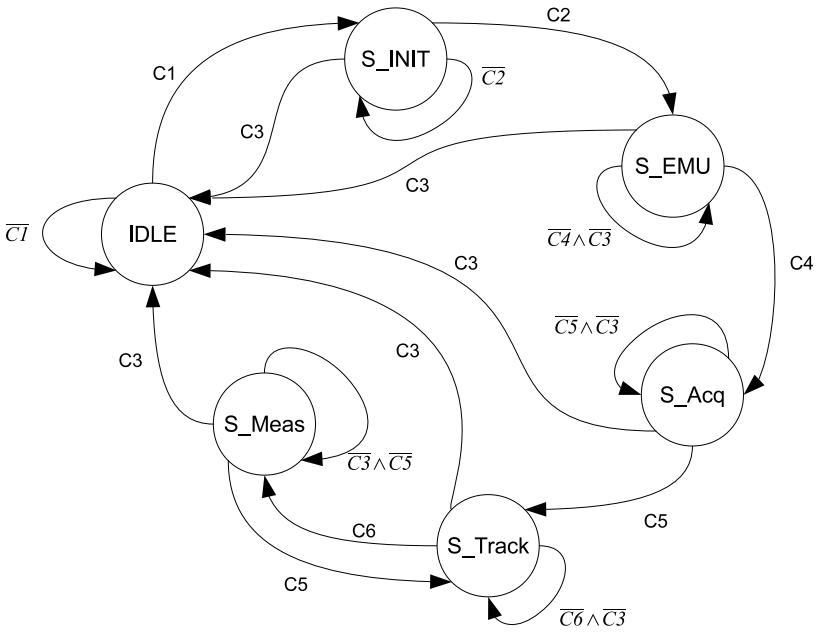


Figure 5.2: An illustration of the state machine of the processor; S_INIT, S_EMU, S_Acq, S_Track and S_Meas are respectively states where the processor connects to Matlab, launch the START_EMUL functionality , starts the acquisition process, configures channels and collects measures. $\wedge$ is the and logic operation

5.2 Contributions and modifications

5.2.1 Adapt the design of the GPS tracking channels for 4 MHz clock

The whole GPS receiver board is sensitive to the ARM clock which is 100 MHz. The ASIC to produce will be clocked at the frequency of 4 MHz i.e a pure clock flow circuit that works at the same clock as the frequency of the input GPS signal. For this reason, the channel tracking modules of the GPS hardware platform have been modified and adapted to this purpose. Moreover, the discriminators and filters used in the GPS platform are the classic and the simplest ones. However, in our study in Chapter. 4, we considered Arc Tangent (ATAN) based discriminators. For this reason, we added CORDICs[2] in the carrier and code discriminator to perform the subtraction and tangent functions. Fig. 5.3 illustrates the new GPS hardware platform diagram.

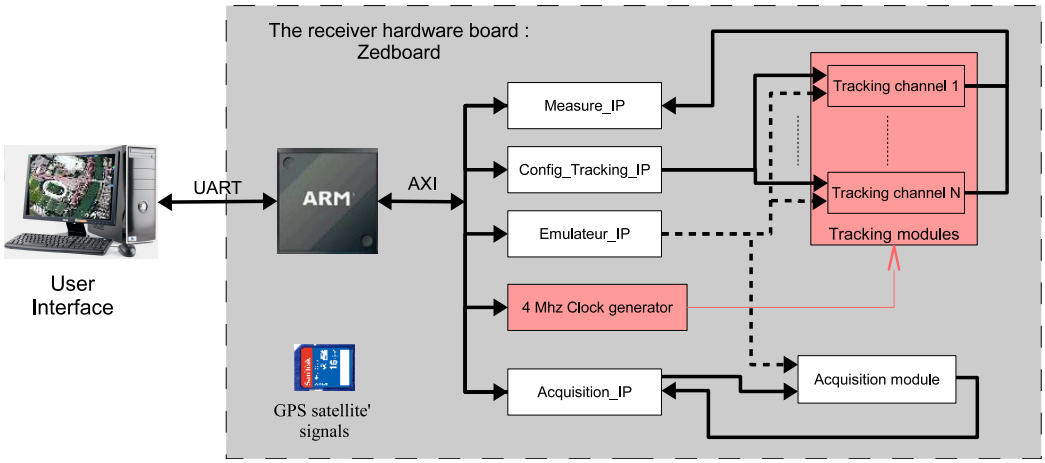


Figure 5.3: An illustration of the first modification on the GPS emulation platform; the modified elements of the platform are represented by the color red.

5.2.2 Isolate the tracking channels modules in a separate board

The main objective by producing an ASIC in the RELIASIC project is to validate by experiences and measurements the performances (in term of power dissipation and quality of service) of fault tolerant techniques proposed in Chapter.4. To save ASIC area (and then the design process cost), it is more judicious to not include the GPS acquisition module, the processor and its IP cores in the ASIC. For this reason and to prepare the environment to be used for experimental work when the ASIC is produced, we decided to isolate the GPS tracking channels in a separate board which is the ML605 board[3] and keep the rest of the

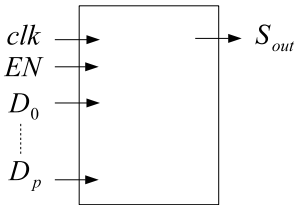
[2]CORDIC is an acronym for COrdinate Rotation DIgital Computeris. It is a simple and efficient hardware algorithm to calculate hyperbolic and trigonometric functions [48].

[3]The ML605 board is a Vertex-6 Xilinx base platform for developing high-performance applications for markets such as wired telecommunications, wireless infrastructure, broadcast... It provides a flexible environ-

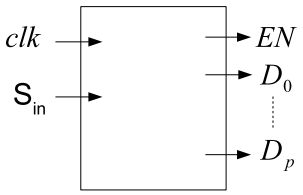
previous platform in the Zedboard. At this step, the ML605 will replace the ASIC to be designed. It will contain the tracking channel modules and only the necessary elements to configure channels and sent back measurements results.

To configure channels and collect measurements results, 6 input signals (with 168 bits in total) and 15 outputs signal per channel (318 bits per channel) are communicated to tracking channels, but the ASIC is pad limited, we are not allowed to have this number of Input/Output. To solve this problem, we decided to sent configuration parameters to the ML605 board and gets measurements from it based on a serial communication. Two type of entities are added to the ASIC design to produce (in the ML605 board) and the ZedBoard in order to route configuration parameters and get measurements: parallel to serial modules (l input bits $\times$ 1 input bits, in Fig. 5.4a) and serial to parallel modules (l input bits $\times$ 1 input bits in Fig. 5.4b).

- When EN ='1', the parallel to serial entity saves the block of data $D_0 \dots D_p$ in a frame, then sends the frame in a serial communication; it sends first '1', then the frame bit per bit at the 4 Mhz clock cycle, at the end sends '0'.
- When S_{in}='1', the serial to parallel entity became active a period l clock cycles to reconstitute the frame sent by the parallel to serial entity, re-generate the data blocks $D_0, \dots, D_p$ and put EN to 1 for one clock cycle.



(a) the parallel to serial entity



(b) the serial to parallel entity

Figure 5.4: Supplementary entities for the communication between boards

Fig. 5.5 illustrates the new version of the GPS hardware platform (GPS_hard_v2). To note here, for each GPS tracking channel module a serial to parallel and a parallel to serial measure entities are used to avoid interference between channels since each channel is processing independently and results measurement are collect for each channel at the end of the correlation function. Delay_Data_I_Q module serves to delay arriving GPS satellites signals the time to regenerate configuration parameters in the ML605.

To connect the ZedBoard to the ML605, we used two XM105 FMC Debug cards (one connected to the FMC-LPC pin of the zedboard and the other connected to FMC-LPC pin of the the ML605) and 12 pin PMOD connectors to make the link between the two XM105 FMC Debug cards (see Fig. 5.6). FMC LPC has 68 singles ended I/O which can also be configured

ment for higher-level system design including applications which need to implement features such as DDR3, Gigabit Ethernet, industry-standard FMC (FPGA Mezzanine Card) connectors for scaling and customization to specific applications and markets... It supports embedded processing with MicroBlaze, soft 32bit RISC [49]

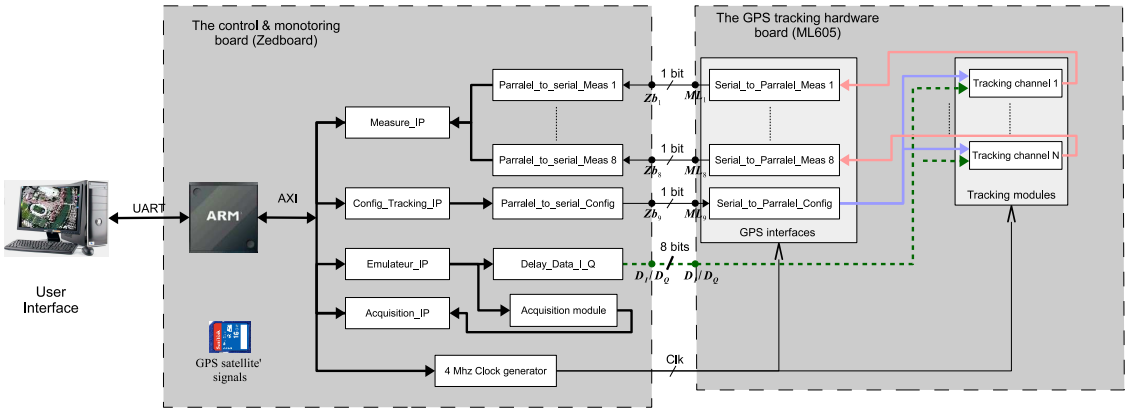


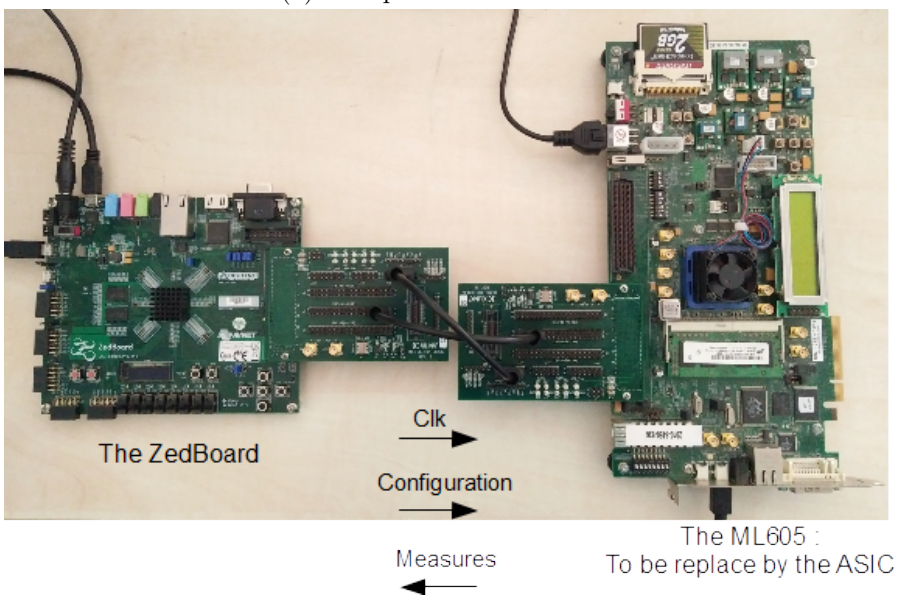
Figure 5.5: Schematic of the GPS hardware platform modified

as 34 differential pairs. For our application, we have configured all the connections as single-ended I/O. Input/Output signals of the GPS tracking channel can be attached to any pins of the FMC-Connector but the 4 Mhz clock pin transmitted to the ML605 board must be a clock capable pin. There are three types of clock pins available: GCLK (Global Clock Pin), SRCC (Single Region Clock Capable Pin) and MRCC (Multiple Region ClocK Capable Pin). It is generally preferred to use the Global Clock Pins since they have specifically designed paths for clock transfer with least possible delay and less clock skew. We give in Appendix A.4, the Zedboard and the ML605 net and pin names inside each board and their corresponding FMC pin names for the two different FMC Debug Card. Appendix A.5 presents the two constraints files used to make the placement configuration for the ML605 board and the ZedBoard, It shows also the schematic for the FMC Debug cards. Finally, The FMC slot in zedboard has an adjustable voltage setting to drive the pins of the FMC which can be adjusted to 1.8, 2.5 or 3.3 V by changing the jumper position on J18. It is necessary to define the correct Vadj for the FMC to avoid damage to the pins. Unlike Zedboard, the ML605 board allows only a fixed operating voltage of 2.5 V on the FMC connections hence for Zedboard we have to set the jumper to 2.5 V on the J18 of the Zedboard.

5.2.3 Improving the GPS tracking observability

The two previous modified versions of the GPS hardware platform offer only the possibility to collect navigation messages from GPS tracking channels to compute the GPS position. In the fault tolerance context, this helps to analyse only the faults impact on the computed position and obtain a high level view of the robustness of the complete system with all the GPS tracking channels. However, it is also very interesting to be able to analyse the fault tolerance of some specific blocks inside the tracking channels to determine precisely the performances of the corresponding fault tolerant techniques. The robustness of the following blocks is investigated: the carrier generator, the code filter, the carrier filter, the code discriminator, the frequency discriminator, the phase discriminator and the correlation function module of all the tracking channels. For this reason, outputs of these blocks have been connected to the parallel to serial and serial to parallel measures entities in respectively the Vertex-

(a) New platform environment



(b) Zoom to the link between the Zedboard and the ML605

Figure 5.6: Presentation of the new GPS hardware platform

6 ML605 and the ZedBoard. Then, using a user IP module named FT_Sotre_Sign, we store them in the RAM. Fig. 5.7 illustrates the new GPS platform design. The Matlab command "ID_CMD_TRIG_MEASURE" is used as a first trigger signal that initiate the storage process of data. It arrives at the frequency of 0.5 s. Since the correlation outputs are updated each 10 ms, we decided to create a second trigger signal that produce triggers at regular intervals of 10 ms. The interval can be controlled by the processor. This is achieved via the Xilinx AXI Timer IP[4].

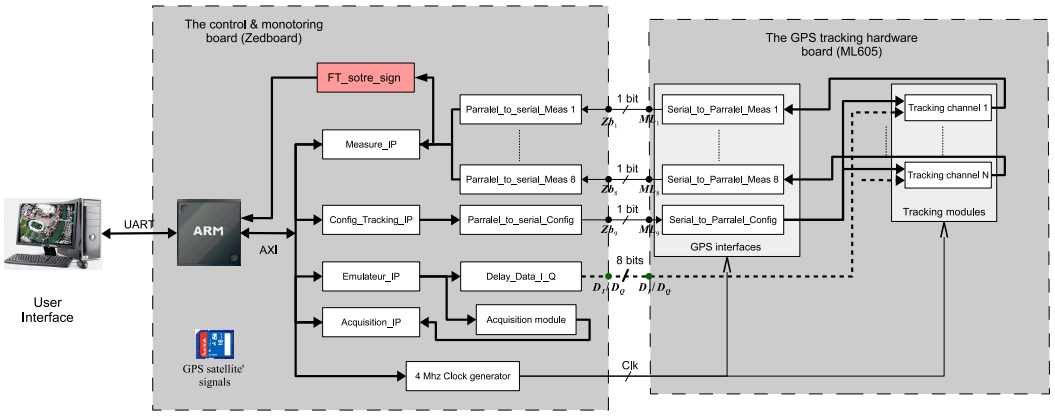


Figure 5.7: An illustration of the GPS emulation platform with the observability option; the modified elements of the platform are represented by the color red.

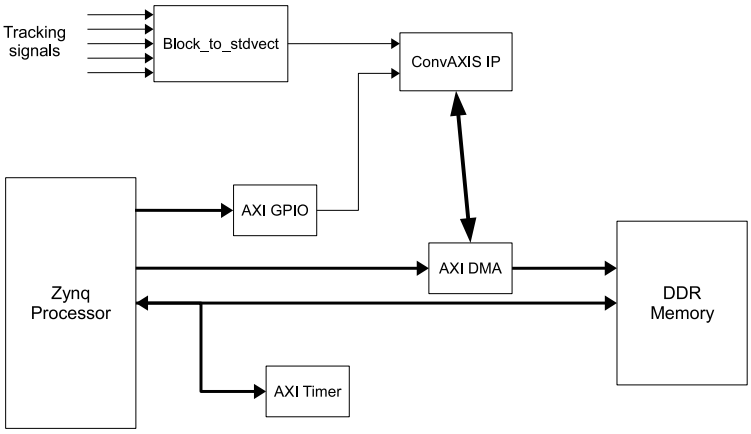


Figure 5.8: The diagram of the FT_sotre_sign module

The hardware architecture of the FT_Sotre_Sign module is given in Fig. 5.8. Measurements signals are raw signals which cannot be transmitted through AXI bus. They have to be converted into AXI-stream packets. For this reason, they are transformed first to the Block_to_Stdvect entity that converts the block of measurements signals to single standard logic vector signal (std_verctor signal). Then, the output std_verctor signal is proceed by the convAxis module and converted into packets of 32 bits (32 bits is the data with for the AXI channel) once a trigger signal arrives to convAxis via the AXI GPIO Xilinx IP[5]. Fig. 5.9

[4]AXI Timer is a Xilinx IP library defined used to implement programmable timer in the FGPA fabric
[5]AXI-GPIO: AXI General Purpose Input Output interface is a Xilinx IP used to transfer data from the

describes the convAxis' state machine. Three states exist: Idle, Read_Trig, Write_Stream. Receiving a trigger signal, the ConvAXIS IP enter the Read_Trig state. The std_vector signal is putted in a buffer and a Valid Signal becomes high. The system enter then the Write_Stream state where packets are transferred one packet per clock cycle. When the last data in the buffer is transmitted, the Last Signal is set to high and the system re-enter the Idle state. After that, the AXI stream packets are then transferred to an AXIDMA[6] entity to be stored in the DDR RAM.

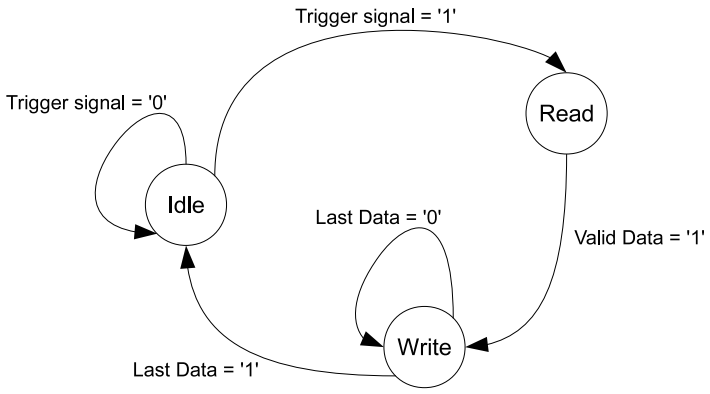


Figure 5.9: ConvAxis state machine

The memory address location can be changed and modified from the processor. At the end of simulation, data stored are recovered using a function defined in gps_save.c file. gps_save.c file contains also functions defined to initiate the DMA module (initialize_dma_system()) and generate triggers signals to be transferred via the AXI GPIO. By running gps_save.c on the Xilinx Software Development Kit (SDK) tool[7], values of stored signal are displayed in the screen as shown in Fig. 5.10. It is important that this part of work was realised by Sushant Samuel[8] under my supervision. India -247667

5.2.4 Include the robust version of the GPS tracking channels

In the thesis several techniques were proposed to improve the robustness of each element of GPS tracking module. Now we have to make a choice for the design of the ASIC: what should be integrated and what not. Since it will be the first 28 nm ASIC of the Lab-STICC, only standard digital logic will be used which excluded the use of Razor systems and C-Muller gates in our design. Our intention is to implement 16 tracking modules: 4 with the reference

Zynq processor to FPGA. It can be configured to send data with at maximum 1024 bits and allows only single data transfer at a time

[6]AXIDMA: The AXI Direct Memory Access is a Xilinx IP to manage communication between memory and AXI4-Stream-type target peripherals using the Axi protocol

[7]The Software Development Kit is a Xilinx integrated tool that provides an environment for creating software platforms and applications targeted for Xilinx embedded processors. SDK works with hardware designs created by either ISE or Vivado tools

[8]Sushant Samuel is a master student in the Indian Institute of Technology Roorkee. He spent three months of internship in Lab-STICC from May to July 2017.

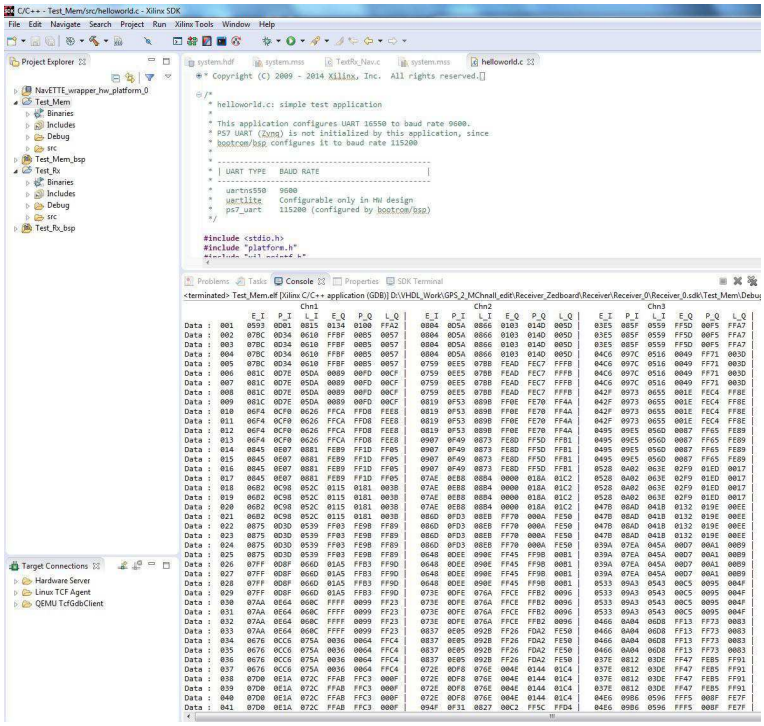


Figure 5.10: Correlation output signals stored and displayed on the console of the SDK tool

version and 12 that include redundant mechanisms (TMR with one voter, TMR with three voters, our contributions...). Note that each module can be configured individually. It is also possible to launch all the tracking modules with to track the same GPS satellite which gives the possibility to compare the robustness and the efficiency of each solution. Finally, an idle state will be programmable for each tracking module to be able determine the power consumption of each solution.

5.3 Summary

The main objective of the work introduced in this chapter is to present the GPS hardware platform that will be used for experiments and measures in the RELIASIC project when the ASIC is produced. The experimental environment can be divided into three parts: the user software interface, the controller and manager board and the ASIC circuit. The user software interface represents the space from where the user order experiments, the ASIC will contain only the GPS tracking channels and necessary elements to configure channels and send back measures and finally the manager board is an intermediate organ between the user interface and the ASIC, it will ensure the communication between them in both directions: write configure parameters and read tracking measures. The GPS hardware platform will offer the possibility to analyse the performances of the implemented fault tolerant techniques at two level; by observing the faults impact on the computed position (high level analysis) and the faults impact in each block inside each tracking channels (low level which offer the possibility

to diagnostic fault tolerance of proposed techniques of each block and validate theoretical results announced in the previous chapter). In the preparation phase of the experimental environment and to test its functionality, we have used the Xilinx Vertex-6 ML605 board to replace the ASIC circuit to be produced.

General Conclusion

The increase in integration density and the requirement of low power supplies to design complex system with a reduced energy consumption can make circuit more and more sensitive to different type of faults: permanent, transient and timing faults. The loss of robustness increases with environmental changes (PVT variations) and interferences (electromagnetic influences, alpha particle radiation ...). Therefore, it is increasingly important to deal with errors effects in order to keep future devices working properly.

The objective of this work was to address the reliability concern in digital systems and introduce new fault tolerant techniques to perform reliable signal processing applications on unreliable hardware. We summarise the major contributions of the thesis as follow:

- Two fault tolerant schemes that can deal with transient, timing and permanent faults [10]. The two schema are called the Fault Tolerant Error Code Correcting based scheme (FT-ECC) and the Duplication with Syndrome based Corrector scheme (DSC). The FT-ECC uses the error code correcting to detect and eliminate faults impact. The DSC duplicates the original functions, combines the inputs of the replica and computes syndromes. The values of syndromes help to detect occurrence of faults and eliminate their impact by either correcting the values or activating a roll-back error recovery mechanism. Two instances of the FT-ECC and DSC were introduced, evaluated and compared to the classical TMR at two levels: the hardware efficiency and the robustness performances in term of MTTF. Results showed that the 4-DSC can reduces the complexity of the TMR to about 30 % while maintaining a level of reliability closed to the TMR. The FT-Hamming presents an interesting level of reliability with a reduction of complexity of up to 12 % compared to the 4-DSC.
- A study of the robustness of an example of a signal processing application which is the GPS tracking channels of GPS receivers. An analyse of the robustness of each block of the tracking channel was made first to identify which element of the architecture, errors at it outputs have the most significant impact on the computed GPS position. Then, fault tolerant techniques were proposed to improve the robustness of each block [11, 12, 13, 14]. They include the FT-Hamming, 3-DSC, some techniques of the state of the art and other system level reliability approaches. A robust version of the GPS tracking channels is ready to be implemented in a real ASIC design.
- A GPS emulation platform was designed to prepare the experimental environment for the ASIC. Starting from an existing GPS emulation platform developed by the ISAE, engineering works were elaborated to adapt the platform to the context of the thesis project. It includes: 1/ some transformation of the design of the tracking channels, 2/ the separation of the GPS tracking channels in a different board that will be replaced by the ASIC later, 3/ the improvement of the GPS platform observability (more signals inside the tracking channels are triggered) and finally, 4/ the inclusion of the robust version of the GPS tracking channels in the design.

The thesis offers several possible future works. The redaction of this report is a new step in the RELIASIC project. The next one is to aggregate all the methods we have proposed in a single project in order to finalise the ASIC design (tapeout theoretically in novembre 2017). Then, when the ASIC will come back, an exciting campaign of measurement will be performed in order to obtain models and informations about how errors appear which will permit to refine the model of noise used in the thesis. This step will also help also to evaluate the fault tolerant techniques proposed in this work in a real design case and will show if they are really effective or not. In our analysis, we evaluated the proposed techniques based on the hardware efficiency (defined as the normalised number of operation per unit area and time unit) and the MTTF. Somme other issues like power, frequency and delay will be investigated once the ASIC is produced. Several methods will be used to stress the circuit and generate faults: low power supply, high temperature, clock skew, electromagnetic shock and eventually, fault insertion by laser or by bombarding the chip with particles. For each experiment, we will compare the resilience and the power dissipation of all the implement methods. We expect the acquisition of a lot of knowledge from these measurements that will trigger new scientific questions, new projects and new collaborations.

In the theoretical study in chapter 2, we don't exhaust the subject, all the contrary. We assumed that the detection and correction process for the FT-Hamming and 4-DSC, and the majority vote in the TMR are fault free. In future works, we propose to take into account the possible fault behaviour of these functions in the comparison. An other two area of future works can be proposed: The application of the ANT and the use of temporal redundancy. In the context of signal processing application where a small precision can be tolerated in the output, the ANT can be adapted to the FT-Hamming and 3-DSC to reduce the area overhead of each solution as mentioned in sections 2.2.2 and 2.3.1. On the other hand, it is possible to use the temporal redundancy instead of adding hardware to execute $\Psi(x_1 \text{ - } x_2)$, $\Psi(x_2 \text{ - } x_3)$, $\Psi(x_3 \text{ - } x_1)$ for the 3-DSC schema for example. Moreover it is possible to process the detection and correction for each solution in a separate clock cycle.

PhD Summary

Publications

1. M. M. Hafidhi, E. Boutillon and C. Winstead "Reliable gold code generators for gps receivers," in 2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1–4, Aug 2015.
2. M. M. Hafidhi, E. Boutillon and C. Winstead, "Reducing the impact of internal upsets inside the correlation process in gps receivers," in 2015 Conference on Design and Architectures for Signal and Image Processing (DASIP), pp. 1–5, Sept 2015.
3. M. M. Hafidhi, E. Boutillon,"Reliable GPS position on an unreliable hardware," in 11^{th} Colloque GDR SoC-SiP, Nantes, France, Juin. 2016.
4. M. M. Hafidhi and E. Boutillon, "Improving the performance of the carrier tracking loop for gps receivers in presence of transient errors due to pvt variations," in 2016 IEEE International Workshop on Signal Processing Systems (SiPS), pp. 80–85, Oct 2016.
5. M. M. Hafidhi and E. Boutillon, "Hardware error correction using local syndromes," in 2017 IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2017

Demo Night

1. M. M. Hafidhi, E. Boutillon, A. Dion "Demo : Localisation in a faulty digital GPS receiver", in Design and Architectures for Signal and Image Processing conference (DASIP), France, Oct. 2016
2. M. M. Hafidhi, S. Samuel, I. Wali, A. Dion and E. Boutillon "Demo : Tracking loop of a GPS receiver under noisy hardware", in 2017 IEEE International Workshop on Signal Processing Systems (SiPS), Oct 2017

Mobility

1. Utah State University in March 2015 to continue the analysis work of the reliability of the Gold sequence generator modules.
2. The Institut Supérieur de l'Aéronautique et de l'Espace (ISAE) (February 2016, January 2017) to acquire the competences on the software used for the existing GPS hardware platform and discuss the platform architecture and functionalities.

A.1 The analysis of the the robustness of Von Neumann architectures

In this section, we detail how $\eta^{(j)}_{NAND}$ and $\eta^{(j)}_{NAND}$ are determinate. Let us consider the following notations:

- p: the failure probability of Ψ,
- ϵ: the failure probability of the NAND and the majority gate,
- b: the correct binary value of the processing module output Ψ,
- R: a random variable describing the output value of the first replica in the executive stage,
- S_j: a random variable describing the output value of the first basic element (the NAND gate or the majority gate) in the block U_j of the restorative stage, $j \in [1..k]$,
- e: the number of inputs of the basic elements of the restorative stage; e= 3 (respectively 2) when the basic element of the restorative stage is the majority gate (respectively the NAND gate),
- $I_{i,j}$: the input number i of the first basic element of the block U_j in the restorative stage, $j \in [1..k]$, $i \in [1..e]$,
- $P^{(j)}_{x \to y}$; the conditional probability of having a value (S_j= y) given ($b = x$). $P^{(j)}_{x \to y} \triangleq \Pr(\ S_j = y \setminus b = x\ )$, $(x,y) \in (\{0,1\})^2$,
- $\eta^{(j)}_{Maj}$: the probability that the majority gate in the bloc U_j is faulty,
- $\eta^{(j)}_{NAND}$: the probability that the NAND gate in the bloc U_j is faulty.

Given no errors in the inputs of all the processing modules Ψ,

$$\begin{cases} \Pr(\ R = 1 \setminus b = 0\ ) = \Pr(\ R = 0 \setminus b = 1\ ) = p \\ \Pr(\ R = 1 \setminus b = 1\ ) = \Pr(\ R = 0 \setminus b = 0\ ) = 1 - p \end{cases} \tag{A.1}$$

We note also that, $\forall\ i \in [1..e]$, we have,

$$\Pr(\ I_{i,1} = 1 \setminus b = 0\ ) = \Pr(\ I_{i,1} = 0 \setminus (b = 1)\ ) = p \tag{A.2}$$

$$\Pr(\ I_{i,1} = 1 \setminus b = 1\ ) = \Pr(\ I_{i,1} = 0 \setminus b = 0\ ) = 1 - p \tag{A.3}$$

$$\forall j \geq 2,\ \ \Pr(\ I_{i,j} = y \setminus b = x\ ) = \Pr(\ S_{j-1} = y \setminus b = x\ ) = P_{x \to y}^{(j-1)} \tag{A.4}$$

$\forall\ j \in [1,k]$, $(x,\ y) \in (\{0,1\})^3$, $(i_0,i_1) \in ([1..e])^2$, $i_0 \neq i_1$, the two events $(\ I_{i_0,j} = y \setminus b = x\ )$ and $(\ I_{i_1,j} = y \setminus b = x\ )$ are independents. So,

$$\Pr(\ I_{i_0,j} = y \cap I_{i_1,j} = y\ \setminus \mathbf{b} = x\ ) = \Pr(\ I_{i_0,j} = y \setminus b = x\ ) \times \Pr(\ I_{i_1,j} = y \setminus b = x\ ) \tag{A.5}$$

$$\Pr(\ I_{i_0,j} = y\ \setminus b = x\ ) = \Pr(\ I_{i_1,j} = y_1\ \setminus b = x\ ) \tag{A.6}$$

A.1.1 Reliability of the NAND Mux architecture

The probability that of the NAND gate in the bloc U_j is faulty is defined by $\eta_{NAND}^{(j)}$ as,

$$\eta_{NAND}^{(j)} \triangleq \frac{1}{2}(\ P_{1\to0}^{(j)} + P_{0\to1}^{(j)}\ ) \tag{A.7}$$

To compute $\eta_{NAND}^{(j)}$, it is important to determinate $P_{x\to y}^{(j)}\ \forall\ (x,y) \in (\{0,1\})^2$. Let us start by $P_{1\to0}^{(j)}$. The NAND gate outputs 0 only if it receives two input values equal to 1 (see Fig. A.1).

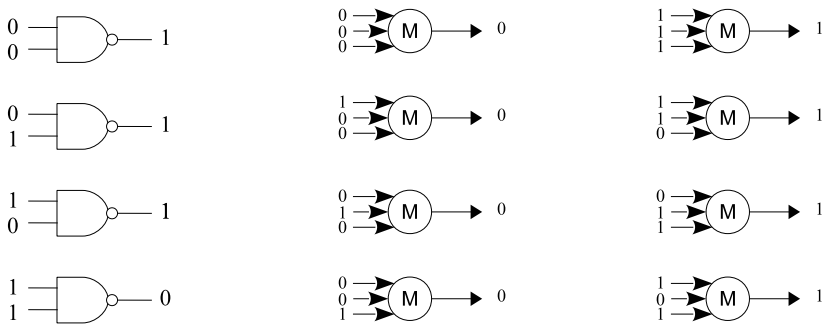


Figure A.1: Majority and NAND gates configurations.

$$P_{1\to0}^{(1)} \triangleq \Pr(\ S_1 = 0 \setminus b = 1\ ) = (1-\epsilon) \cdot \Pr(\ I_{1,1} = 1 \cap I_{2,1} = 1 \setminus b = 1\ ) +$$
$$\epsilon \cdot \Pr(\ \overline{I_{i,1}^{(1)} = 1 \cap I_{2,1}^{(1)} = 1}\ \setminus b = 1\ ) \tag{A.8}$$

Applying ((A.5)) and ((A.3)) to ((A.8)), we obtain,

$$\boxed{P_{1-0}^{(1)} = (1-\epsilon) \cdot (1-p)^2 + \epsilon \cdot (\ 1 - (1-p)^2\ )} \tag{A.9}$$

$$P^{(2)}_{1\to 0} \triangleq \Pr(\ S_2 = 0 \setminus b = 1\ ) = (1-\epsilon)\cdot \Pr(\ I_{1,2} = 1 \cap I_{2,2} = 1 \setminus b = 1\ ) + \epsilon \cdot \Pr(\ \overline{I_{1,2} = 1 \cap (I_{2,2} = 1}\ \setminus b = 1\ ) \tag{A.10}$$

Applying ((A.5)) and ((A.4)) to ((A.10)), we obtain,

$$\boxed{P^{(2)}_{1\to 0} = (1-\epsilon)\ (P^{(1)}_{1\to 1})^2 + \epsilon\ (\ 1 -\ (P^{(1)}_{1\to 1})^2)} \tag{A.11}$$

To generalise, $\forall\ j \geq 2$,

$$\boxed{P^{(j)}_{1\to 0} = (1-\epsilon)\ (P^{(j-1)}_{1\to 1})^2 + \epsilon\ (\ 1 -\ (P^{(j-1)}_{1\to 1})^2)} \tag{A.12}$$

Now let us determinate $P^{(j)}_{0\to 0}$:

$$P^{(1)}_{0\to 0} \triangleq \Pr(\ S_1 = 0 \setminus b = 0\ ) = (1-\epsilon)\cdot \Pr(\ I_{1,1} = 1 \cap I_{2,1} = 1 \setminus b = 0\ ) + \epsilon \cdot \Pr(\ \overline{I_{1,1} = 1 \cap I_{2,1} = 1}\ \setminus b = 0\ ) \tag{A.13}$$

By applying ((A.5)) and ((A.2)) to ((A.13)), we obtain,

$$\Rightarrow \quad \boxed{P^{(1)}_{0\to 0} = (1-\epsilon)\cdot p^2 + \epsilon \cdot (\ 1 - p^2\ )} \tag{A.14}$$

$$P^{(2)}_{0\to 0} \triangleq \Pr(\ S_2 = 0 \setminus b = 0\ ) = (1-\epsilon)\cdot \Pr(\ I_{1,2} = 1 \cap I_{2,2} = 1 \setminus b = 0\ ) + \epsilon \cdot \Pr(\ \overline{I_{1,2} = 1 \cap I_{2,2} = 1}\ \setminus b = 0\ ) \tag{A.15}$$

$$\Rightarrow \quad \boxed{P^{(2)}_{0\to 0} = (1-\epsilon)\ (P^{1}_{0\to 1})^2 + \epsilon\ (\ 1 -\ (P^{1}_{0\to 1})^2)} \tag{A.16}$$

To generalise, $\forall\ j \geq 2$,

$$\boxed{P^{(j)}_{0\to 0} = (1-\epsilon)\ (P^{(j-1)}_{0\to 1})^2 + \epsilon\ (\ 1 -\ (P^{(j-1)}_{0\to 1})^2)} \tag{A.17}$$

$P^{(j)}_{1\to 1}$ and $P^{(j)}_{0\to 1}$ can be deducted respectively from $P^{(j)}_{1\to 0}$ and $P^{(j)}_{0\to 0}$ as,

$$P^{(j)}_{1\to 1} = 1 - P^{(j)}_{1\to 0} \quad \Rightarrow \quad \boxed{P^{(j)}_{1\to 1} = (1-\epsilon)\ (\ 1 -\ (P^{(j-1)}_{1\to 1})^2) + \epsilon\ (P^{(j-1)}_{1\to 1})^2} \tag{A.18}$$

$$P^{(j)}_{0\to 1} = 1 - P^{(j)}_{0\to 0} \quad \Rightarrow \quad \boxed{P^{(j)}_{0\to 1} = (1-\epsilon)\ (\ 1 -\ (P^{(j-1)}_{0\to 1})^2) + \epsilon\ (P^{(j-1)}_{0\to 1})^2} \tag{A.19}$$

A.1.2 Reliability of the Maj Mux architecture

The majority gate outputs 1 once two or more of the input values are equal to 1 (see Fig. A.1). Let consider $P^{(j)}_{>1,x}$ the probability that two or more of the inputs of the majority gates inside

the bloc U_j are 1 given $b = x$. $P^{(1)}_{>1,0}$ can be seen as the probability that more than two from three of the original processing modules Ψ are faulty. It is expressed as,

$$P^{(1)}_{>1,0} = 1 - \Pr(\ (I_{1,1}, I_{2,1}, I_{3,1}) \in \{(000), (100), (010), (001)\}\ \setminus b = 0\ ) \tag{A.20}$$

$$\Rightarrow \quad \boxed{P^{(1)}_{>1,0} = 1 - (\ (1-p)^3 + 3 \cdot p \cdot (1-p)^2\ )} \tag{A.21}$$

$$\forall\ j \geq 2,\ P^{(j)}_{>1,x} = 1 - \Pr(\ (I_{1,j}, I_{2,j}, I_{i,j}) \in \{(000), (100), (010), (001)\}\ \setminus b = x\ ) \tag{A.22}$$

According to (A.6), we obtain,

$$\begin{aligned} P^{(j)}_{>1,x} = 1 - [\ (1 - \Pr(\ I_{1,j} = 1 \setminus b = x\ )\ )^3 + 3 \cdot \Pr(\ I_{1,j} = 1 \setminus b = x\ ) \\ \times (1 - \Pr(\ I_{1,j} = 1 \setminus b = x\ )\ )^2\ ] \end{aligned} \tag{A.23}$$

$$\Rightarrow \quad \boxed{P^{(j)}_{>1,x} = 1 - (\ (1 - P^{(j-1)}_{x\to 1})^3 + 3 \cdot P^{(j-1)}_{0\to 1} \cdot (1 - P^{(j-1)}_{x\to 1})^2\ )} \tag{A.24}$$

The error probability at the output of the majority gate of the block U_j is defined as,

$$\eta^{(j)}_{Maj} \triangleq \frac{1}{2}(\ P^{(j)}_{1\to 0} + P^{(j)}_{0\to 1}\ ) \tag{A.25}$$

Now let us determinate $P^{(j)}_{1\to 0}$ and $P^{(j)}_{0\to 1}$. The output of the majority gate of the bloc U_j is equal to 1 when more than two of the its inputs are 1 and the majority gate is non faulty, or, when at most one input value is equal 1 and the majority gate is faulty. Thus,

$$P^{(1)}_{1\to 0} = P^{(1)}_{0\to 1} = (1-\epsilon) \cdot p^{(1)}_{>1,0} + \epsilon(1 - p^{(1)}_{>1,0}) \tag{A.26}$$

$$P^{(1)}_{1\to 1} = P^{(1)}_{0\to 0} = (1-\epsilon) \cdot (1 - p^{(1)}_{>1,0}) + \epsilon p^{(1)}_{>1,0} \tag{A.27}$$

$\forall\ j \geq 2$,

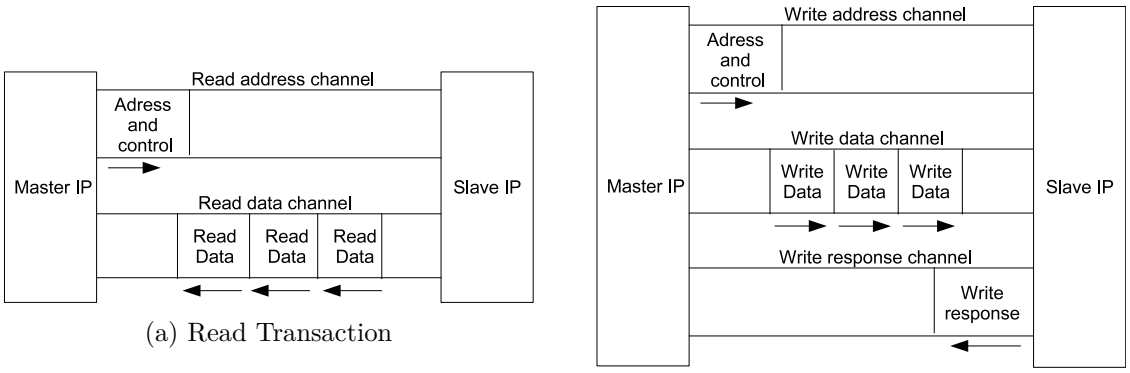
$$P^{(j)}_{1\to 0} = P^{(j)}_{0\to 1} = (1-\epsilon) \cdot p^{(j)}_{>1,0} + \epsilon(1 - p^{(j)}_{>1,0}) \tag{A.28}$$

$$P^{(j)}_{1\to 1} = P^{(j)}_{0\to 0} = (1-\epsilon) \cdot (1 - p^{(j)}_{>1,0}) + \epsilon p^{(j)}_{>1,0} \tag{A.29}$$

A.2 The Xilinx AXI protocol

AXI interconnect protocol is part of the ARM Advanced Micro-controller Bus Architecture (AMBA) specification. It is especially prevalent in Xilinx's Zynq devices, providing the inter-

face between the processing system and programmable logic sections of the chip. Released in 2010, AXI4 is the version of AXI that exists in the ZedBoard. It is a burst based transaction of informations between a master and a slave IP cores. The AXI4 interface consists of five different channels where each channel transfer data in one direction: Read Address Channel, Write Address Channel, Read Data Channel, Write Data Channel, Write Response Channel. The separate channels of AXI4 allows simultaneous bidirectional data transfer (Read and write transactions). Fig. A.2 illustrates the channel architectures for the Read and Write data transfer. The limit of the AXI4 is a burst of up to 256 data transfers.



(a) Read Transaction

(b) Write Transaction

Figure A.2: The architecture of the AXI4 interface between slave and master IP cores. The address channel for every transaction contains the address and control information that describes the nature of the data to be transferred. In write transactions, in which all the data flows from the master to the slave, the AXI4 protocol has an additional write response channel to allow the slave to signal to the master the completion of the write transaction.

A.3 The platform functionality

A.3.1 Initiate the communication with the board

Role: Test the communication with the ZedBoard, and, send GPS satellites signals from the SD Card to the RAM.

User Software interface			The ZedBoard	
Matlab ("TestRx_Zedboard.m")			ARM Processor ("TextRx_Nav.c")	
Function Name	initBoard.m	USB	Function Name	taskInitBoard()
Variables		URT	Pointeurs	
Name	Size	⇔	Name	Size
ID_CMD_BOARD_CONNECT	uint8	⇒	DataR	int8
ID_ACK	uint8	⇐	ID_ACK_BOARD_CONNECT	int8
ID_CMD_RAM_STATUS	uint8	⇒	DataR	int8
ID_ACK	uint8	⇐	ID_ACK_BOARD_READY	int8

A.3.2 Run the Emulator

Role: Transfer I/Q GPS satellite' samples from the RAM block memory to GPS algorithm modules at the frequency of 4 Mhz. It uses the Xilinx DMA CONTROLER IP to manage memory.

Parameters:

- DATA_I: Real part of GPS satellite signals
- DATA_Q: Imaginary part of GPS satellite signals
- ValidData: valid sample signal (equal to '1' one clock each 250 ns)
- SimuOn: 1 bit: indicate if emulator is on

User Software interface			The ZedBoard								
Matlab ("TestRx_Zedboard.m")			ARM Processor ("TextRx_Nav.c")		AXI	FPGA IP					
Function Name	com_init_channel.m	USB	Function Name	taskInitChannel()	Interconnect	Configure_IP					
Variables		URT	Pointeurs		AXI	Ports & User logic slave registers					
Name	Size	⇔	Name	Size	Channel	Register Name	Direction	Pin Name	I/O	Size	
ID_CMD_START_EMUL	uint8	⇒	DataR	int8							
TotalTransfertSize	uint32	⇒	DMA_Ctrl_Reg1	int	⇒ WDCHN	slv_reg1	⇒	SIGNAL_SIZE	in	32 bits	
			DMA_Ctrl_Reg0	int	⇒ WDCHN	slv_reg0	⇒	SIGNAL_BASE_ADDRESS	in	32 bits	
			DMA_Ctrl_Reg2	int	⇒ WDCHN	slv_reg2		DMA_IP_BASE_ADDRESS	in	32 bits	
								ValidData	out	1 bit	
								SimuOn	out	1 bit	
								DATA_I	out	4 bits	
								DATA_Q	out	4 bits	
ID_ACK	uint8	⇐	ID_ACK_START_EMUL	int8							

A.3.3 Configure and initiate tracking channels

Role: Configure GPS tracking channels

Parameters:

- ADDR_CHANNEL: the number of the tracking channel to configure.
- PRN_ID: GPS satellite PRN number
- DOPPLER_INIT: Doppler parameter to initiate the tracking
- SAMPLE_INIT: Number of GPS satellite signal Samples to start the tracking from.

User Software interface			The ZedBoard							
Matlab ("TestRx_Zedboard.m")			ARM Processor ("TextRx_Nav.c")		AXI	FPGA IP				
Function Name	com_init_channel.m	USB	Function Name	taskInitChannel()	Interconnect	Configure_IP				
Variables		URT	Pointeurs		AXI	Ports & User logic slave registers				
Name	Size	⇔	Name	Size	Channel	Register Name	Direction	Pin Name	I/O	Size
ID_CMD_INIT_CHANNEL	uint8	⇒	DataR	int8						
DOPPLER_FREQ	int16	⇒	Track_Parameters	int 32 bits	Write Data Channel ⇒	slv_reg0(15 downto 0)	⇒	DOPPLER_INIT	out	16 bits
PRN_TRACK	uint8	⇒			Write Data Channel ⇒	slv_reg0(23 downto 16)	⇒	PRN_ID	out	8 bits
CHN_NO	uint8	⇒			Write Data Channel ⇒	slv_reg0(30 downto 24)	⇒	ADDR_CHANNEL	out	8 bits
INIT_SAMPLE_TRACK	2×uint32	⇒	Init_Sample_Track	int 64 bits	Write Data Channel ⇒	slv_reg1 & slv_reg2	⇒	SAMPLE_INIT	out	64 bits
								INIT	Out	1 bit
ID_ACK	uint8	⇐	ID_ACK_INIT_CHANNEL	int8						

A.3.4 Get tracking measurements

Role: Collect tracking measurements of different GPS tracking channels

Parameters:

- MEASURED_SAMPLE: GPS satellite Sample number
- CODE_ACC_CHN0: Propagation delay related the satellite tracked by the first GPS channel tracking (channel number 0)
- F1_CHN0: Doppler offset related the satellite tracked by the channel number 0
- BIT_CNT_0, CODE_CNT_0, BIT_ARRAY_0: Navigation messages corresponding to the satellite tracked by the channel number 0
- CODE_ACC_CHN1: Propagation delay related the satellite tracked by the second GPS channel tracking (channel number 1)
- F1_CHN1: Doppler offset related the satellite tracked by the channel number 1
- Navigation messages corresponding to the satellite tracked by the channel number 1
- ...
- CODE_ACC_CHN7: Propagation delay related the satellite tracked by the 8^{th} GPS channel tracking (channel number 7)
- F1_CHN1: Doppler offset related the satellite tracked by the channel number 7
- BIT_CNT_7, CODE_CNT_7, BIT_ARRAY_7: Navigation messages corresponding to the satellite tracked by the channel number 7

User Software interface			The ZedBoard																
Matlab ("TestRx_Zedboard.m")			ARM Processor ("TextRx_Nav.c")							AXI	FPGA IP								
Function_Name	com_measure	USB	Function_Name	taskTrigMeasures()						Interconnect	Measure_IP								
Variables		URT	Pointeurs							AXI	Ports & User logic slave registers								
Name	Size	⇔	Name	Size						Channel	Register Name	Direction	Pin Name	I/O	Size				
ID_CMD_TRIG_MEASURE	uint8	⇒	DataR	int8															
meas_sample	2×uint32	⇒	MeasuredSample	int 32 bits						WDCH	⇒	slv_reg1 & slv_reg0							
SAMPLE	2×uint32	⇐	Sample_cnt	2×int32						RDCH	⇐	slv_reg1 & slv_reg0	⇐	SAMPLE_CNT	in	64 bits			
ID_ACK	uint8	⇐	ID_NAK_TRIG_MEASURE	int8															
track_obs(1).code_accumulator	uint32	⇐	RegMeasures	bit32						Read Data Channel	⇐	slv_reg2	⇐	CODE_ACC_0	in	32 bits			
track_obs(1).doppler_estimate	uint32	⇐									⇐	slv_reg3	⇐	F1_0	in	32 bits			
track_obs(1).cnt_bit	uint32	⇐									⇐	slv_reg4(13 downto 5)	⇐	BIT_CNT_0	in	10 bits			
track_obs(1).cnt_code	uint32	⇐									⇐	slv_reg4(4 downto 0)	⇐	CODE_CNT_0	in	5 bits			
track_obs(1).databit	uint32	⇐									⇐	slv_reg5 & slv_reg6	⇐	BIT_ARRAY_0	in	64 bits			
											⋮	⋮	⋮	⋮	⋮				
track_obs(4).code_accumulator	uint32	⇐									⇐	slv_reg37	⇐	CODE_ACC_7	in	32 bits			
track_obs(8).doppler_estimate	uint32	⇐									⇐	slv_reg38	⇐	F1_7	in	32 bits			
track_obs(8).cnt_bit	uint32	⇐									⇐	slv_reg39(13 downto 5)	⇐	BIT_CNT_3	in	10 bits			
track_obs(8).cnt_code	uint32	⇐									⇐	slv_reg39(4 downto 0)	⇐	CODE_CNT_3	in	5 bits			
track_obs(8).databit	uint32	⇐									⇐	slv_reg40 & slv_reg41	⇐	BIT_ARRAY_3	in	64 bits			

A.4 Pin Connection for the Zedboard, the ML605 and the FMC Cards

Zedboard			Virtex-6 ML605		
Net Name	Board pin Name	FMC pin Name	FMC pin Name	Board pin Name	Net Name
Clk_4Mhz	J21	LA-08-P	LA-18CC-P	L29	Clk_4Mhz
Zb_1	C20	LA-18CC-N	LA-08-N	K29	ML_1
Zb_2	D20	LA-18CC-P	LA-08-P	J30	ML_2
Zb_3	B20	LA-17CC-N	LA-07-N	H32	ML_3
Zb_4	B19	LA-17CC-P	LA-07-P	G32	ML_4
Zb_5	R21	LA-08-N	LA-19-N	N30	ML_5
Zb_6	R20	LA-09-P	LA-09-P	L26	ML_6
Zb_7	J22	LA-08-N	LA-19-P	M30	ML_7
Zb_8	G16	LA-19-N	LA-18CC-N	L30	ML_8
Zb_9	G15	LA-19-P	LA-09-N	L25	ML_9
$D_I(3)$	B17	LA-31-N	LA-29-N	P34	$D_I(3)$
$D_I(2)$	B16	LA-31-P	LA-29-P	N34	$D_I(2)$
$D_I(1)$	B15	LA-30-N	LA-28-N	M33	$D_I(1)$
$D_I(0)$	C15	LA-30-P	LA-28-P	N33	$D_I(0)$
$D_Q(3)$	C18	LA-29-N	LA-31-N	L31	$D_Q(3)$
$D_Q(2)$	C17	LA-29-P	LA-31-P	M31	$D_Q(2)$
$D_Q(1)$	A17	LA-28-N	LA-30-N	M27	$D_Q(1)$
$D_Q(0)$	A16	LA-28-P	LA-30-P	M26	$D_Q(0)$

Table A.1: Corresponding Pin Connections of the Xilinx Vertex-6 ML605 board, the Zedboard and the FMC debug cards.

A.5 Placement constraint files

A.5.1 The Vertex-6 ML605 board

```
NET Digital_Signal_I<3>    LOC=P34;
NET Digital_Signal_I<2>    LOC=N34;
NET Digital_Signal_I<1>    LOC=M33;
NET Digital_Signal_I<0>    LOC=N33;

NET Digital_Signal_Q<3>    LOC=L31;
NET Digital_Signal_Q<2>    LOC=M31;
NET Digital_Signal_Q<1>    LOC=M27;
NET Digital_Signal_Q<0>    LOC=M26;

NET Data_in_asic_1Bit      LOC=L25;
NET Clk_4Mhz CLOCK_DEDICATED_ROUTE = FALSE;
NET Clk_4Mhz               LOC=L29;

NET Data_out_asic_1Bit     LOC=K29;
NET Data_out_asic_1Bit_1 LOC=J30;
NET Data_out_asic_1Bit_2 LOC=H32;
NET Data_out_asic_1Bit_3 LOC=G32;
NET Data_out_asic_1Bit_4 LOC=N30;
NET Data_out_asic_1Bit_5 LOC=L26; #L29;#M30;
NET Data_out_asic_1Bit_6 LOC=M30;#L30;
NET Data_out_asic_1Bit_7 LOC=L30;#L29;
```

```
NET "Clk_4Mhz" TNM_NET = Clk_4Mhz;
TIMESPEC TS_Clk_4Mhz = PERIOD "Clk_4Mhz" 250 ns HIGH 60% INPUT_JITTER 50 ps;

OFFSET = IN 0.5 ns VALID 250 ns AFTER "Clk_4Mhz";
OFFSET = OUT 2 ns BEFORE "Clk_4Mhz";
```

A.5.2 The ZedBoard

```
set_property PACKAGE_PIN J21 [ get_ports Clk_4Mhz ]
set_property PACKAGE_PIN C20 [ get_ports Data_in_FPGA_1Bit]
set_property PACKAGE_PIN D20 [ get_ports Data_in_FPGA_1Bit_1]
set_property PACKAGE_PIN B20 [ get_ports Data_in_FPGA_1Bit_2]
set_property PACKAGE_PIN B19 [ get_ports Data_in_FPGA_1Bit_3]
set_property PACKAGE_PIN R21 [ get_ports Data_in_FPGA_1Bit_4]
set_property PACKAGE_PIN R20 [ get_ports Data_in_FPGA_1Bit_5]
set_property PACKAGE_PIN J22 [ get_ports Data_in_FPGA_1Bit_6]
set_property PACKAGE_PIN G16 [ get_ports Data_in_FPGA_1Bit_7]

set_property PACKAGE_PIN G15 [ get_ports Data_out_FPGA_1Bit ]
set_property PACKAGE_PIN B17 [ get_ports Digital_Signal_I_OUT [3] ]
set_property PACKAGE_PIN B16 [ get_ports Digital_Signal_I_OUT [2] ]
set_property PACKAGE_PIN B15 [ get_ports Digital_Signal_I_OUT [1] ]
set_property PACKAGE_PIN C15 [ get_ports Digital_Signal_I_OUT [0] ]
set_property PACKAGE_PIN C18 [ get_ports Digital_Signal_Q_OUT [3] ]
set_property PACKAGE_PIN C17 [ get_ports Digital_Signal_Q_OUT [2] ]
set_property PACKAGE_PIN A17 [ get_ports Digital_Signal_Q_OUT [1] ]
set_property PACKAGE_PIN A16 [ get_ports Digital_Signal_Q_OUT [0] ]

set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_I_OUT [3] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_I_OUT [2] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_I_OUT [1] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_I_OUT [0] ]

set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_Q_OUT [3] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_Q_OUT [2] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_Q_OUT [1] ]
set_property IOSTANDARD LVCMOS25 [ get_ports Digital_Signal_Q_OUT [0] ]

set_property IOSTANDARD LVCMOS25 [ get_ports Clk_4Mhz]

set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_1 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_2 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_3 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_4 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_5 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_6 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_in_FPGA_1Bit_7 ]
set_property IOSTANDARD LVCMOS25 [ get_ports Data_out_FPGA_1Bit ]
```

A.5.3 FMC Debug Cards

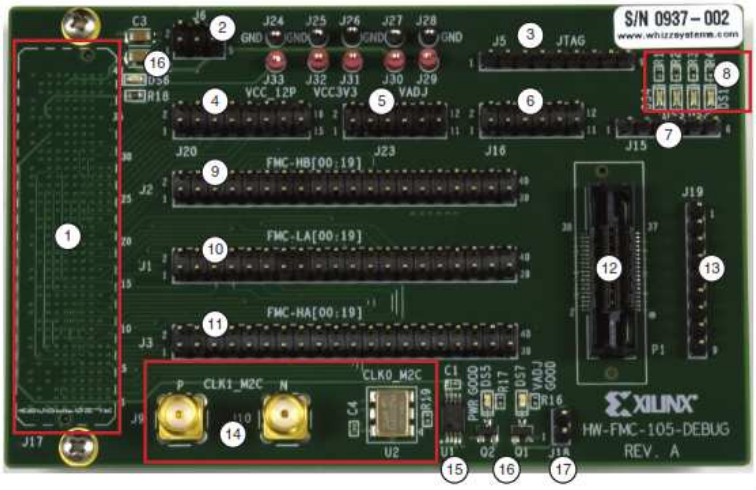


Figure A.3: The FMC Card referred as the XM 105 board [1]

This section describes the technical description of FPGA Mezzanine Card (FMC) as introduced in [1]. Fig. A.3 shows the XM105 features.

(1) Connector J17: The Vita 54.1 FMC HPC connector provides the interface to boards with Xilinx FPGA. It mates with a FMC LPC connector or an FMC HPC connector on Xilinx board. This connector is on the bottom side of the board.

(2) Connector J6: 6 pin header providing a means to manually configure XM105 voltage connections.

(3) Connector J5 FMC JTAG: This connector with 9 positions provides an interface from the FMC to the board's FMC LPC or FMC HPC JTAG signals. It can not offer the possibility to program the FPGA on the board.

(4) Connector J20: 16-pin header that provides an interface from the XM105 to 16 single ended signals on the board's FMC LPC or FMC HPC interface.

(5) Connector J23: 12-pin header that provides an interface from the XM105 to 12 single ended signals only on the board's FMC HPC interface. Some of these pins are not connected on the Xilinx ML605 FMC HPC board interface.

(6) Connector J16: 12-pin header that provides an interface from the XM105 to 12 single ended signals on the board's FMC LPC or FMC HPC interface. Power can be either 3.3V or 2.5

(7) Connector J15: 6-pin connector that provides an interface from the XM105 to four single-ended signals on the board's FMC LPC or FMC HPC interface. These four FMC LPC signals are connected to green LEDS on the XM105.

(8) Four green user LEDS: illuminated with an active-High signal from the FPGA on the board

(9) Connector J1/2/3: 40-pin connector tha provides an interface from the XM105 to 40 single-ended signals on the board's FMC HPC interface. J2 and J3 can only be utilized when installed in a board supporting the FMC HPC interface

(10) Mictor connector P1 with 38 female pin

(11) Connector J19: does not have an interface to the FMC LPC or FMC HPC interface of the board. It provides a connection to Mictor connector P1 and includes 3.3V and GROUND connections.

(12) Clocks Two clocks sources to to provide a high-precision differential clock

(13) 2 Kb EEPROM: electrically erasable programmable memory (EEPROM) with 2 Kb (256 bytes) of non-volatile storage.

(14) Power Good LEDS Power good LEDS for + 12V, board to mezzanine card (PG_C2M) and V adjust /3.3V

(15) Connector J18 2 pins for GND connection to PG_M2C LPC Connector

Bibliography

[1] Xilinx, "FMC XM105 Debug Card User Guide." https://www.xilinx.com/support/documentation/boards_and_kits/ug537.pdf.

[2] Karl Rupp, "40 Years of Microprocessor Trend Data." https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/.

[3] D. Bhaduri, S. Shukla, P. Graham, and M. Gokhale, "Comparing reliability-redundancy tradeoffs for two von neumann multiplexing architectures," *IEEE Transactions on Nanotechnology*, vol. 6, pp. 265–279, May 2007.

[4] R. E. Lyons and W. Vanderkulk, "The use of triple-modular redundancy to improve computer reliability," *IBM J. Res. Dev.*, vol. 6, pp. 200–209, 1962.

[5] B. Shim, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 497–510, May 2004.

[6] V. Gherman, S. Evain, M. Cartron, N. Seymour, and Y. Bonhomme, "System-level hardware-based protection of memories against soft-errors," in *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, pp. 1222–1225, April 2009.

[7] A. Ejlali, B. M. Al-Hashimi, P. Rosinger, and S. G. Miremadi, "Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks," in *Design, Automation Test in Europe Conference Exhibition, 2007. DATE '07*, pp. 1–6, April 2007.

[8] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: a low-power pipeline based on circuit-level timing speculation," in *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, pp. 7–18, Dec 2003.

[9] COMIN Labs, "COMIN Labs Official site." http://www.ueb.eu/Theme/recherche/investissementsAvenir/comingLabs2/.

[10] M. M. Hafidhi and E. Boutillon, "Hardware error correction using local syndromes," in *2017 IEEE International Workshop on Signal Processing Systems (SiPS)*, Oct 2017.

[11] M. M. Hafidhi, E. Boutillon, and C. Winstead, "Reliable gold code generators for gps receivers," in *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1–4, Aug 2015.

[12] M. M. Hafidhi, E. Boutillon, and C. Winstead, "Reducing the impact of internal upsets inside the correlation process in gps receivers," in *2015 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, pp. 1–5, Sept 2015.

[13] M. Dridi, M. M. Hafidhi, C. Winstead, and E. Boutillon, "Reliable nco carrier generators for gps receivers," in *2015 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, pp. 1–5, Sept 2015.

[14] M. M. Hafidhi and E. Boutillon, "Improving the performance of the carrier tracking loop for gps receivers in presence of transient errors due to pvt variations," in *2016 IEEE International Workshop on Signal Processing Systems (SiPS)*, pp. 80–85, Oct 2016.

[15] A. Danowitz, K. Kelley, J. Mao, J. P. Stevenson, and M. Horowitz, "Cpu db: Recording microprocessor history," *Queue*, vol. 10, pp. 10:10–10:27, Apr. 2012.

[16] International Technology Roadmap for Semiconductors, "Summary 2013 ORTC Technology Trend Targets." https://www.dropbox.com/sh/z80p2ne051g770t/AADvOmLAEcVwCpxtexqSvE7ra/2013ORTC_SummaryTable.pdf?dl=0. Online; accessed 18 April 2017.

[17] H. Aysan, "New strategies for ensuring time and value correctness in dependable real-time systems," May 2009.

[18] S. Bhunia, S. Mukhopadhyay, and K. Roy, "Process variations and process-tolerant design," in *20th International Conference on VLSI Design held jointly with 6th International Conference on Embedded Systems (VLSID'07)*, pp. 699–704, Jan 2007.

[19] S. Borkar, "Tackling variability and reliability challenges," *IEEE Des. Test*, vol. 23, pp. 520–520, Nov. 2006.

[20] S. Kamal and C. V. Page, "Intermittent faults: A model and a detection procedure," *IEEE Transactions on Computers*, vol. C-23, pp. 713–719, July 1974.

[21] E. H. Neto, F. L. Kastensmidt, and G. Wirth, "Tbulk-bics: A built-in current sensor robust to process and temperature variations for soft error detection," *IEEE Transactions on Nuclear Science*, vol. 55, pp. 2281–2288, Aug 2008.

[22] W. L. Heimerdinger and C. B. Weinstock, "A conceptual framework for system fault tolerance," *Tech. rep. DTIC Document*, 1992.

[23] Q. Zhou and K. Mohanram, "Gate sizing to radiation harden combinational logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 155–166, Jan 2006.

[24] S. Hareland, J. Maiz, M. Alavi, K. Mistry, S. Walsta, and C. Dai, "Impact of cmos process scaling and soi on the soft error rates of logic processes," in *2001 Symposium on VLSI Technology. Digest of Technical Papers (IEEE Cat. No.01 CH37184)*, pp. 73–74, June 2001.

[25] D. A. Tran, A. Virazel, A. Bosio, L. Dilillo, P. Girard, A. Todri, M. E. Imhof, and H. J. Wunderlich, "A pseudo-dynamic comparator for error detection in fault tolerant architectures," in *2012 IEEE 30th VLSI Test Symposium (VTS)*, pp. 50–55, April 2012.

[26] L. Anghel and M. Nicolaidis, "Cost reduction and evaluation of a temporary faults detecting technique," in *Proceedings Design, Automation and Test in Europe Conference and Exhibition 2000 (Cat. No. PR00537)*, pp. 591–598, 2000.

[27] D. A. Tran, A. Virazel, A. Bosio, L. Dilillo, P. Girard, S. Pravossoudovitch, and H. J. Wunderlich, "A hybrid fault tolerant architecture for robustness improvement of digital circuits," in *2011 Asian Test Symposium*, pp. 136–141, Nov 2011.

[28] M. Mehrara, M. Attariyan, S. Shyam, K. Constantinides, V. Bertacco, and T. Austin, "Low-cost protection for ser upsets and silicon defects," in *2007 Design, Automation Test in Europe Conference Exhibition*, pp. 1–6, April 2007.

[29] J. v. Neumann, "Probabilistic logics and synthesis of reliable organisms from unreliable components," in *Automata Studies*, pp. 43–98, 1956.

[30] C. Winstead, Y. Luo, E. Monzon, and A. Tejeda, "An error correction method for binary and multiple-valued logic," in *Multiple-Valued Logic (ISMVL), 2011 41st IEEE International Symposium on*, pp. 105–110, May 2011.

[31] W. B. D.E. Muller, "A theory of asynchronous circuits," in *Proc. International Symposium on the Theory of Switching Part 1*, pp. 204–243, 1959.

[32] Y. Tang, *Computation on unreliable architecture.* PhD thesis, Universite Bretagne Sud, 2013.

[33] B. L. Garner, "The residue number system," *Managing Requirements Knowledge, International Workshop on*, vol. 0, p. 146, 1959.

[34] B. Shim and N. R. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, pp. 336–348, April 2006.

[35] S. Das, *Razor: A variability-tolerant design methodology for low power and robust computing.* PhD thesis, The University of Michigan, 2009.

[36] S. Das, C. Tokunaga, S. Pant, W. H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw, "Razorii: In situ error detection and correction for pvt and ser tolerance," *IEEE Journal of Solid-State Circuits*, vol. 44, pp. 32–48, Jan 2009.

[37] M. Fojtik, D. Fick, Y. Kim, N. Pinckney, D. M. Harris, D. Blaauw, and D. Sylvester, "Bubble razor: Eliminating timing margins in an arm cortex-m3 processor in 45 nm cmos using architecturally independent error detection and correction," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 66–81, Jan 2013.

[38] Y. Tang, E. Boutillon, C. Jégo, and M. Jézéquel, "Hardware efficiency versus error probability in unreliable computation," in *2011 IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 168–173, Oct 2011.

[39] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-repeat-request error-control schemes," *IEEE Communications Magazine*, vol. 22, pp. 5–17, December 1984.

[40] C. Hadjicostis and G. C. Verghese, "Coding approaches to fault tolerance in linear dynamic systems," *Information Theory, IEEE Transactions on*, vol. 51, no. 1, pp. 210–228, 2005.

[41] U.S. government, "Official U.S. government information about the Global Positioning System (GPS) and related topics." http://www.gps.gov/.

[42] United States Naval Observatory, " Block II Satellite Information." ftp://tycho.usno.navy.mil/pub/gps/gpsb2.txt.

[43] J. B.-Y. Tsui, *Fundamentals of Global Positioning System Receivers*. Wiley-Interscience, 2000.

[44] ISAE, "NAVETTE: NAVigation En Tout Type d'Environnement." https://websites.isae-supaero.fr/navette/about/introduction.

[45] J. T. Curran, G. Lachapelle, and C. C. Murphy, "Improving the design of frequency lock loops for gnss receivers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, pp. 850–868, Jan 2012.

[46] A. M. Kamel, "Design and testing of an intelligent gps tracking loop for noise reduction and high dynamics applications," *ION GNSS*, 2010.

[47] Xilinx, "ZedBoard Zynq-7000 ARM/FPGA SoC Development Board." https://www.xilinx.com/products/boards-and-kits/1-elhabt.html.

[48] Steve Arar, "An Introduction to the CORDIC Algorithm." https://www.allaboutcircuits.com/technical-articles/an-introduction-to-the-cordic-algorithm/.

[49] Xilinx, "VIRTEX-6 FPGA ML605 EVALUATION KIT." https://www.xilinx.com/publications/prod_mktg/ml605_product_brief.pdf.