ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE ÉLECTRIQUE
M.Ing.

PAR
Rami LOUKIL

CONCEPTION ET MISE EN ŒUVRE DE MODULE DE CONTRÔLE DE PUISSANCE ET DE GÉNÉRATION DE BRUIT POUR CONDITIONNER LES SIGNAUX RF D'UN SIMULATEUR DE CONSTELLATION GPS ET GALILEO

MONTRÉAL, LE 4 MAI 2010

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Ammar Kouki, directeur de mémoire
Département de Génie électrique à l'École de technologie supérieure

M. René Jr. Landry, président du jury
Département de Génie électrique à l'École de technologie supérieure

Jocelyn Dore, examinateur externe
AGENCE SPATIALE CANADIENNE

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 9 AVRIL 2010

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

CONCEPTION ET MISE EN ŒUVRE DE MODULE DE CONTRÔLE DE PUISSANCE ET DE GÉNÉRATION DE BRUIT POUR CONDITIONNER LES SIGNAUX RF D'UN SIMULATEUR DE CONSTELLATION GPS ET GALILEO

Rami LOUKIL

RÉSUMÉ

La radionavigation par satellite a reçu beaucoup d'intérêt pendant le 21^{e} siècle. Les avènements les plus importants dans ce secteur sont le nouveau système de positionnement par satellite de l'Union européenne baptisé Galileo et la modernisation du système américain GPS (*Global Positionning System*). Le regroupement de ces différents systèmes sous l'acronyme GNSS (*Global Navigation Satellite System*) permettrait une fiabilité et un positionnement d'une très grande précision. Cette tendance pousse les industries à concevoir des récepteurs capables de gérer ce nombre croissant de signaux. Ces nouveaux produits nécessitent des procédures de validation avant même que le système européen Galileo ne soit opérationnel.

Dans ce contexte, le laboratoire LACIME développe un simulateur de signaux GPS et Galileo. Le présent mémoire traitera les différentes parties de ce simulateur ainsi qu'une étude détaillée la conception d'un système de contrôle de puissance des signaux piloté par logiciel ainsi qu'un système de contrôle du ratio C/N_0 (*Carrier to Noise Ratio*).

Le système de contrôle de puissance a été développé afin d'offrir une précision de l'ordre de 0.1 dB sur une plage de 40 dB pour les signaux « GPS L1, Galileo E1 » à la fréquence de 1575.42 MHz, « GPS L5, Galileo E5a » à la fréquence 1176.45 MHz et le signal Galileo E5b à la fréquence 1207.14 MHz. Le système de contrôle de puissance a été mis en œuvre par l'élaboration d'une carte à circuit imprimé et d'une communication bidirectionnelle avec le logiciel qui permet une gérance totale du système. L'architecture implémentée a été judicieusement étudiée afin de permettre une flexibilité et une extensibilité du système. Elle permet d'exécuter plusieurs modes de contrôle à savoir « contrôle de gain », « contrôle automatique de puissance », « calibration de puissance » et « variation de la puissance dans le temps ». Le système de contrôle de puissance a été testé et validé pour donner une précision inférieure à 0.1 dB sur 40 dB pour les deux modes de fonctionnement de base (contrôle de gain et contrôle automatique de puissance). Les deux autres modes découlent de ces derniers pour donner plus d'options à l'utilisateur du simulateur GPS et Galileo.

Afin de donner à l'utilisateur une maîtrise totale de la qualité des signaux, un générateur de bruit blanc gaussien a été mis en œuvre. Il permet, en combinaison avec le système de contrôle de puissance, de contrôler le ratio C/N_0. Une étude sur les méthodes de génération d'un tel bruit blanc a été faite. Le choix s'est établit sur une implémentation numérique au sein du FPGA qui présente une flexibilité et une complexité moindre par rapport à une approche analogique. Le générateur de bruit blanc a été développé par la méthode Box-Muller qui présente beaucoup d'avantages dans le cas d'une implémentation numérique. Son architecture a été optimisée pour rejoindre les contraintes des ressources matérielles du

FPGA. Le bruit blanc gaussien est combiné aux signaux en analogique et sa puissance est contrôlable par le biais de la partie logicielle. Il a été testé et validé par un récepteur commercial et un récepteur développé au sein de l'école. L'augmentation linéaire du niveau du bruit dégrade la qualité des signaux et le rapport C/N_0 diminue linéairement dans les deux récepteurs de test.

Mots clés : GNSS, contrôle de puissance, bruit blanc gaussien, FPGA.

DESIGN AND IMPLEMENTATION OF A POWER CONTROL MODULE AND NOISE GENERATOION FOR RF SIGNALS CONDITIONNING IN A GPS AND GALILEO CONSTELLATION SIMULATOR

LOUKIL, Rami

ABSTRACT

Satellite radio navigation has received considerable interest during the 21st century. The most important advents in this sector are the new satellite positioning system of the European Union called Galileo and the modernization of the Unites States' Global Positionning System GPS. The combination of these different systems under the acronym GNSS (Global Navigation Satellite System) would allow reliable positioning and a high accuracy. This trend pushes the industry to design receivers able to handle the growing number of signals. These new products require validation procedures before the European system, Galileo becomes operational.

In this context, the laboratory LACIME is developing a GPS and Galileo signals simulator. This master's thesis presents the different parts of the simulator and a detailed design study of a power control system driven by software for GPS and Galileo signals and a C/N_0 ratio control system.

The power control system has been developed to provide an accuracy of about 0.1 dB over a range of 40 dB for signals "L1 GPS, Galileo E1" at a frequency of 1575.42 MHz, "GPS L5, Galileo E5a" at 1176.45 MHz and the Galileo signal E5b at 1207.14 MHz. The power control system has been implemented by developing a printed circuit board and a two-way communication with the software that allows complete system management. The architecture has been carefully studied and implemented to allow flexibility and scalability. It allows the user to run multiple control modes namely "gain control", "automatic power control, "power calibration" and "time varying power control". The power control system has been tested and validated to give an accuracy of less than 0.1 dB over 40 dB for both basic modes of operation (gain control and automatic power control). The other two modes resulting from the latter give more features to the user of the simulator.

To give the user a total control of signal quality, a white Gaussian noise generator has been implemented. It allows, in addition to the power control system, the ratio C/N_0 control. A study on methods for generating such a white noise was made. The choice was a digital implementation in the FPGA which provides a flexibility and lower complexity compared to an analog approach. The white noise generator was developed with the Box-Muller method which has many advantages in the case of digital implementation. Its architecture has been optimized to reach the hardware resources constraints of FPGA. The white Gaussian noise is combined with the signals in the RF part and its power can be controlled by software. It has been tested and validated by a commercial receiver and receiver built within the school. The linear increase in the noise level degrades the signal quality and the C/N_0 ratio decreases linearly in both test receivers.

Key words: GNSS, power control, white Gaussian noise, FPGA.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

ADC	Analog to Digital Converter
ADS	Advanced Design System
ALC	Automatic Level Control
BER	Bit Error Rate
BM	Box Muller
C/N_0	Carrier to Noise Density Ratio
CAN	Convertisseur Analogique Numérique
CDMA	Code Division Multiple Access
CAN	Convertisseur Numérique Analogique
CPLD	Complex Programmable Logic Device
DAC	Digital to Analog Converter
DS-CDMA	Direct Sequence-CDMA
DSP	Digital Signal Processing
ÉTS	École de Technologie Supérieure
FPGA	Field Programmable Grid Array
FSB	Fast Synchronous Bus
GNSS	Global Navigation Satellite System
GPIO	General Purpose Input Output
GPS	Global Positioning System
IF	Intermediary Frequency
JTAG	Joint Test Action Group

LACIME	Laboratoire de Communication et d'Intégration de la Micro Électronique
LFSR	Linear Feedback Shift Register
LP-LFSR	Leap forward Linear Feedback Shift Register
LUT	Look Up Table
OCXO	Oven Controlled Crystal Oscillator
PC	Personal Computer
PCB	Printed Circuit Board
PCI	Peripheral Component Interconnect
PLL	Phase Locked Loop
RF	Radio Frequency
SNR	Signal to Noise Ratio
SSBM	Single SideBand Mixer
TEB	taux d'Erreurs Binaire
v.a	variable aléatoire
VCO	Voltage Controlled Oscillator
VGA	Variable Gain Amplifier/Attenuator
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit

INTRODUCTION

Depuis une vingtaine d'années, les États-Unis étaient le seul pays à fournir un système pleinement opérationnel de radionavigation avec le GPS (*Global Positionning System*). Le 21^{e} siècle s'annonce prometteur en matière de positionnement par satellites. Le système le plus attendu est celui de l'Union européenne « Galileo » qui fournira une multitude de nouveaux services aux utilisateurs ainsi qu'une meilleure précision de positionnement.

La tendance industrielle du moment tourne autour du développement des récepteurs hybrides et spécialement GPS et Galileo. Ces industries doivent tester leurs équipements avant la mise sur le marché. Une première solution consiste à placer une antenne sur le toit d'un bâtiment et la relier au laboratoire de test. Les longueurs de câbles peuvent atteindre des centaines de mètres et donc engendrer de grandes pertes en puissance ce qui est très contraignant vu que les signaux reçus des satellites sont déjà très faibles. En plus, on ne peut simuler qu'une seule position (celle du bâtiment) avec GPS. Le système Galileo est encore en cours de déploiement, il n'y a pas donc assez de satellites en orbite pour le calcul des positions. La deuxième solution consiste à embarquer tout l'équipement de test (récepteur, ordinateur, logiciel d'analyse) dans un véhicule et de circuler dans les rues afin de tester les capacités des récepteurs. Cette solution est encombrante et est couteuse en temps et en effort. Un simulateur de constellations GPS et Galileo permet, donc, de s'affranchir de tous ces problèmes, car on peut créer tous les scénarios de déplacement et tester les récepteurs en laboratoire.

Dans ce cadre, le LACIME (Laboratoire de Communication et d'Intégration de la Micro Électronique) développe un simulateur de signaux GPS/Galileo qui lui permettra de tester des récepteurs hybrides. Ce simulateur doit reproduire les signaux des constellations de satellites de chacun des deux systèmes. Il doit être capable de générer des signaux identiques (codage, modulation, bandes de fréquences…) pour recréer des conditions similaires à celles des deux systèmes.

Le simulateur est composé de trois parties. Une partie logicielle qui fait tous les calculs nécessaires à la génération des signaux des constellations. Elle permet aussi de piloter la deuxième partie du simulateur, la partie du traitement numérique des signaux (partie IF) qui a pour tâche de mettre en forme les signaux en se basant sur les données reçues de la partie logicielle. Trois signaux analogiques sont générés par la partie IF (à 70 MHz) et sont injecté dans une carte RF, le troisième et dernier maillon du simulateur. La partie RF permet de transposer les signaux IF à leurs fréquences réelles (celles des signaux GPS et Galileo). Elle a aussi le rôle de mettre ces signaux à la puissance adéquate et de les combiner pour les injecter dans le récepteur de tests.

Ce projet de maitrise est composé de deux parties. La première consiste à développer un système de contrôle de puissance pour les trois signaux GPS et Galileo. Le contrôle doit être établit sur une plage de 40 dB avec un pas de 0.1 dB. La deuxième partie du travail comprend la mise en œuvre d'un système de contrôle du ratio C/N_0 et plus précisément, développer un générateur de bruit blanc qui sera injecté aux signaux GPS et Galileo pour dégrader leurs qualités et par conséquent évaluer la robustesse des récepteurs.

Le présent mémoire est divisé en trois chapitres. Le premier présente un survol des systèmes de positionnements GPS et Galileo ainsi qu'une présentation du simulateur GPS et Galileo. Il se focalisera sur la partie RF du simulateur. Le deuxième chapitre traitera en détail la conception et la réalisation du système de contrôle de puissance en commençant par l'architecture du module pour passer ensuite aux composants et les modes d'ajustements ainsi que la communication entre la partie logicielle et la partie RF et enfin, être conclu par la validation du système. Le troisième et dernier chapitre aborde le deuxième volet de ce projet à savoir le système de contrôle du ratio C/N_0. Il traitera, les différents types d'implémentation d'un générateur de bruit blanc et les étapes de conception d'un tel générateur. Ce chapitre est conclu par la validation du générateur de bruit par le biais d'un récepteur commercial et d'un récepteur développé au sein de l'ÉTS.

Equation Chapter 1 Section 1

CHAPITRE 1

LE SIMULATEUR DE SIGNAUX GNSS

Le positionnement par satellites a reçu beaucoup d'intérêt depuis leurs créations dans la fin des années cinquante. Plusieurs systèmes ont été développés pour satisfaire ce besoin, uniquement militaire à l'époque. Actuellement, on compte quelques systèmes pleinement et partiellement opérationnels à savoir :

- GPS (*Global Positionning System*) pour les États-Unis (opérationnel) ;
- Galileo pour l'Europe (en cours de déploiement) ;
- GLONASS (*GLObal'naya NAvigatsionnaya Sputnikovaya Sistema*) pour la Russie (n'est pas pleinement opérationnel) (NovAtel, 2007);
- Compass/Beidou pour la Chine (opérationnel uniquement en asie) (UNOOSA, 2008).

Ce chapitre présente en premier lieu le principe du positionnement par satellites et décrit les systèmes GPS et Galileo. Ensuite, il introduit le projet GNSS (*Global Navigation Satellite System*) en détaillant la partie RF sur laquelle nous allons travailler.

1.1 Principe du positionnement par satellite

Le principe de positionnement est simple. Si les distances séparant un récepteur de quatre satellites sont connues, on peut alors, déterminer la position de ce récepteur.

1.1.1 Principe de la trilatération

La trilatération est un concept géométrique qui permet de calculer les coordonnées d'un point dans un plan à l'aide de trois points de références (Sauriol B, 2008). Elle se base uniquement sur les coordonnées de ces points ainsi que les distances les séparant du point dont on veut calculer la position. En connaissant la distance entre un point de référence et le point désiré (le point M dans la Figure 1.1), on affirme que se dernier ce trouve sur un cercle de centre le

point de référence et de rayon la distance les séparant. De cette manière, et en obtenant les deux distances restantes, le point M n'est que l'intersection des trois cercles. La Figure 1.1 illustre le principe de la trilatération dans un espace à deux dimensions.

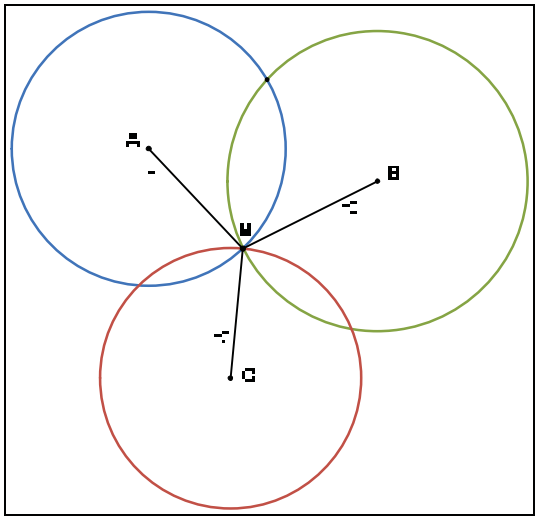

Figure 1.1 Principe de la trilatération en 2D.

La trilatération peut s'appliquer aussi pour un espace tridimensionnel. Les cercles de la Figure 1.1 se transforment, alors, en sphères centrées en les points de références et de rayon les distances d1, d2 et d3. La Figure 1.2 illustre le résultat de la trilatération en trois dimensions. En effet, l'intersection de deux sphères donne naissance à un cercle qui intercepte la troisième sphère en deux points. La position ne peut encore être déterminée avec précision d'où le besoin d'un quatrième point de référence qui permettra de lever l'ambigüité sur la décision.

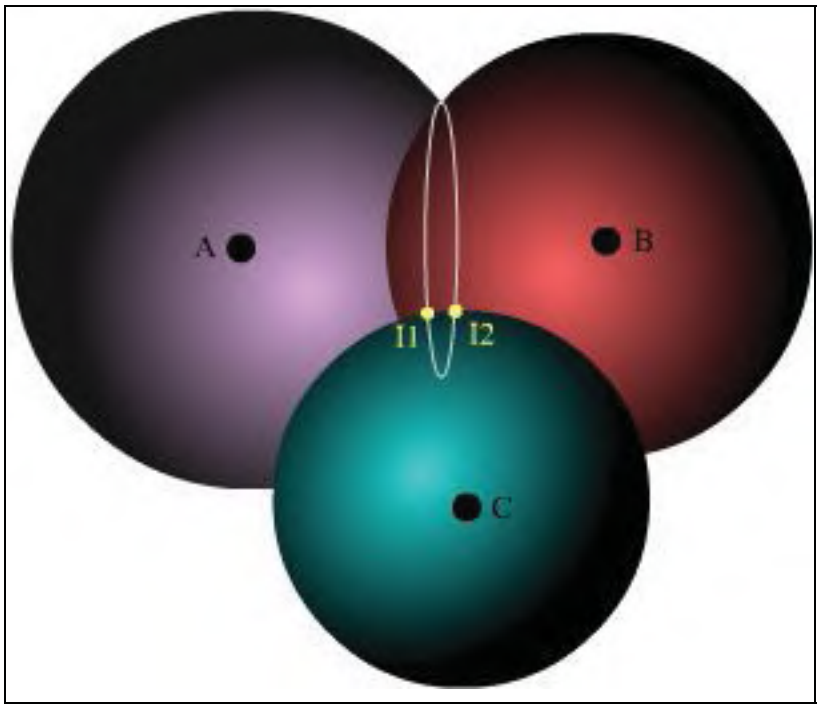

Figure 1.2 Principe de la trilatération en 3D.

Si on connait les coordonnées (x_i,y_i,z_i) des quatre points de références et les distances d_i, on peut déterminer, en résolvant le système d'équations (1.1), les coordonnées (x_M,y_M,z_M) du point désiré.

$$\begin{cases} d_1 = \sqrt{(x_1 - x_M)^2 + (y_1 - y_M)^2 + (z_1 - z_M)^2} \\ d_2 = \sqrt{(x_2 - x_M)^2 + (y_2 - y_M)^2 + (z_2 - z_M)^2} \\ d_3 = \sqrt{(x_3 - x_M)^2 + (y_3 - y_M)^2 + (z_3 - z_M)^2} \\ d_4 = \sqrt{(x_4 - x_M)^2 + (y_4 - y_M)^2 + (z_4 - z_M)^2} \end{cases} \quad (1.1)$$

Dans un cas idéal, la résolution de ce système donne une solution unique. Toutefois, dans la pratique, des erreurs surviennent au niveau des positions des points de référence et des distances. L'intersection des sphères n'étant plus un point dans l'espace, on obtient une région de probabilité qui est représentée par la Figure 1.3 pour un espace à deux dimensions.

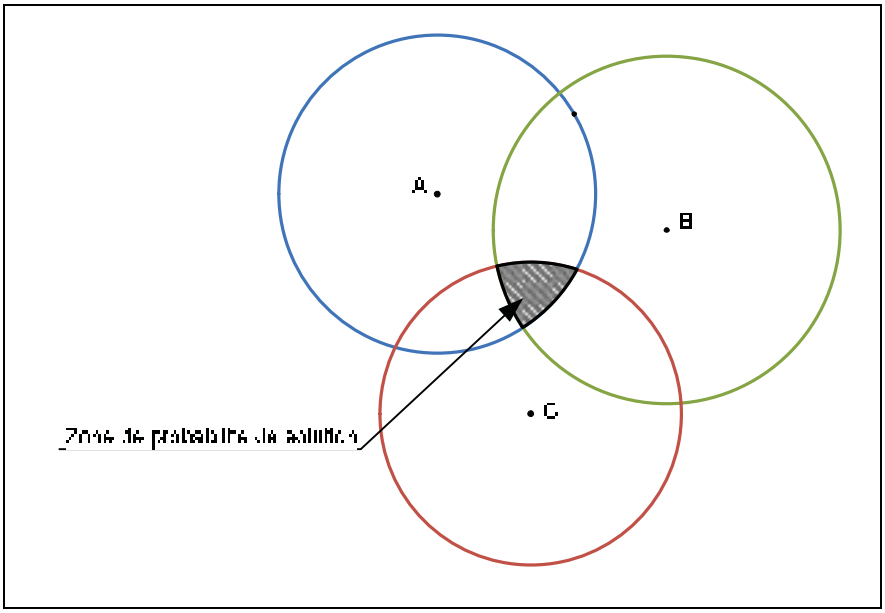

Figure 1.3 L'effet des erreurs sur la trilatération.

1.1.2 Le positionnement par satellites

Les systèmes de positionnement par satellites utilisent la méthode de la trilatération afin de calculer la position d'un récepteur. Les points de références dans ces systèmes constituent la constellation des satellites gravitant autour de la terre. Il faudrait donc connaitre leurs positions de façon précise. Ensuite, il suffit de mesurer la distance séparant le satellite du récepteur avec une précision compatible avec la précision de navigation qu'on veut obtenir.

Les satellites émettent continuellement des signaux radio qui contiennent leurs positions et l'instant d'envoi t_e. Le récepteur a la tâche de calculer le temps de réception de ces signaux t_r. Connaissant leur vitesse de propagation (c : vitesse de la lumière), on en déduit la distance séparant le récepteur du satellite :

$$d = c(t_r - t_e) \tag{1.2}$$

Le calcul du temps de propagation doit être extrêmement précis, car une erreur d'une microseconde dans le temps provoque une erreur de 300 m en position. Le temps d'envoi des

signaux est déterminé par les horloges atomiques embarquées à bord des satellites. Ce sont des horloges atomiques extrêmement coûteuses, mais aussi extrêmement fiables et stables. Le temps d'arrivée est donné par le récepteur qui possède une horloge de moindre qualité. Au niveau du temps d'arrivée, il y a donc une erreur provoquée par le récepteur (Sauriol B, 2008). La différence entre le temps d'envoi et le temps de réception est calculé par le récepteur, ensuite il en déduit la distance qui le sépare du satellite. Cette distance est biaisée principalement à cause des imperfections de l'horloge du récepteur, elle est appelée pseudo-distance (Sauriol B, 2008).

Dans la section précédente, nous avons conclu qu'il faudrait quatre points de références pour arriver à calculer une position. Dans le fait, il est facile de lever l'ambigüité sur la position, car une des solutions se trouve sur la surface de la terre alors que la deuxième se situe dans l'espace. Par conséquent, trois satellites suffiraient pour déterminer la position d'un récepteur. Cependant, ce n'est pas le cas en pratique. En effet, en plus des erreurs d'horloges des satellites dont la correction est envoyée dans le message de navigation, les erreurs causées par l'horloge du récepteur engendrent une imprécision au niveau du calcul de la pseudo-distance. On suppose, donc, qu'il y a un biais entre le véritable temps de réception des signaux et le temps déterminé par le récepteur. Pour éliminer ce biais, un quatrième satellite est requis et ainsi, le récepteur doit résoudre le système d'équations (1.3) pour déterminer sa position (Sauriol B, 2008).

$$\begin{cases} \Delta t_1 = \dfrac{1}{c}\sqrt{\left[\left(x_M - x_1\right)^2 + \left(y_M - y_1\right)^2 + \left(z_M - z_1\right)^2\right]} + \Delta H_m \\ \Delta t_2 = \dfrac{1}{c}\sqrt{\left[\left(x_M - x_2\right)^2 + \left(y_M - y_2\right)^2 + \left(z_M - z_2\right)^2\right]} + \Delta H_m \\ \Delta t_3 = \dfrac{1}{c}\sqrt{\left[\left(x_M - x_3\right)^2 + \left(y_M - y_3\right)^2 + \left(z_M - z_3\right)^2\right]} + \Delta H_m \\ \Delta t_4 = \dfrac{1}{c}\sqrt{\left[\left(x_M - x_4\right)^2 + \left(y_M - y_4\right)^2 + \left(z_M - z_4\right)^2\right]} + \Delta H_m \end{cases} \tag{1.3}$$

Où Δt_i représente le temps de propagation calculé par le récepteur, (x_m, y_m, z_m) sont les coordonnées du mobile, (x_i, y_i, z_i) sont les coordonnées du satellite *i* et ΔH_m est le biais inconnu de l'horloge du récepteur.

1.2 Spécifications des systèmes GPS et Galileo

1.2.1 Spécification du système GPS

Le GPS est un système de positionnement par satellite qui a été développé vers la fin des années soixante-dix par le département de la défense américain (Ublox, 2007). Il a été conçu, initialement, pour un usage militaire. Ce n'est que plus tard qu'il a été accessible pour les civils. Le GPS offre une capacité de positionnement n'importe où sur la planète et dans toutes les conditions météorologiques. Il peut desservir un nombre illimité d'usagers vu que le système est unidirectionnel (les utilisateurs ne font que recevoir les signaux et calculer leurs positions).

Structure du système GPS

Le système GPS est composé de trois segments : spatial, contrôle et utilisateur.

- Segment spatial : il est constitué de 30 à 32 satellites opérationnels qui sont placés sur six orbites quasi circulaires inclinées de 55°. La Figure 1.4 illustre l'organisation des satellites sur les orbites;
- Segment de contrôle : comme son nom l'indique, le segment de contrôle permet de piloter le système GPS. Il est composé d'un réseau mondial de stations de contrôle. Sa première tâche est de suivre les satellites de la constellation, déterminer leurs positions, vérifier l'intégrité du système, les horloges atomiques, l'état des satellites et les erreurs de positionnement (Ublox, 2007). Ces informations sont ensuite envoyées vers les satellites sur la bande S (2 à 4 GHz);

- Segment utilisateur : il inclut tous les utilisateurs civils et militaires. Avec un récepteur GPS, ils peuvent capter les signaux et déterminer leurs positions. Le GPS est accessible partout dans le monde sauf où la puissance est affaiblie (ex : environnement *Indoor*, environnement urbain, etc) sans frais (Ublox, 2007);

Figure 1.4 Constellation des satellites du système GPS.
Tiré de (Cole-Parmer, 2009)

Spécifications des signaux GPS

Les satellites GPS émettent actuellement trois signaux L1, L2 et L5. Ces signaux transportent les messages de navigation permettant aux récepteurs de calculer leurs positions (ARINC research corporation, 2000).

- Signal L1 : ce signal transporte les messages de navigation avec l'utilisation de deux codes d'étalement de spectre. L'un pour l'usage public et l'autre pour l'usage militaire;
- Signal L2C : ce signal est une modernisation du système GPS. Il a été ajouté pour accroitre la précision de navigation. Il y a actuellement huit satellites en orbite qui transmettent le signal L2C. Les récepteurs auront le choix entre les différents signaux selon leurs niveaux de puissances, la nature de l'application utilisée, etc;

- Signal L5 : ce signal sera destiné pour le grand public (SoL : ***S**afety **o**f **L**ife*). Il est une amélioration du système GPS et il servira à améliorer les applications en aviation. Il y a maintenant un seul satellite en orbite à titre de démonstration.

Les signaux GPS sont basés sur la technique DS-CDMA (*Direct Sequence – Code Division Multiple Access*). Cette solution se révèle la plus adéquate à une telle application vu qu'il n'y a pas de communication ascendante entre le segment utilisateur et le segment spatial (du récepteur au satellite). De cette façon, les satellites émettent sur les mêmes fréquences L1, L2 et L5 et ils sont différenciés selon des codes. En plus, cette technique assure le cryptage des données pour les applications militaires ou commerciales et permet la réception des signaux de très faibles puissances (noyés dans le bruit).

Les caractéristiques des signaux GPS sont listées dans le Tableau 1.1. La largeur de bande des signaux est grande, car le système utilise l'étalement de spectre comme technique.

Tableau 1.1 Caractéristiques physiques des signaux GPS
Tiré de (ARINC research corporation, 2000) (ARINC Incorporated, 2001)

Signal	**Fréquence porteuse**	**Bande**	**Polarisation**
L1	1575.42 MHz	20.46 MHz	Circulaire droite
L2	1227.6 MHz	20.46 MHz	Circulaire droite
L5	1176.45 MHz	24 MHz	Circulaire droite

La Figure 1.5 illustre le plan de fréquence des signaux GPS. Les fréquences L1, L2 et L5 doivent dériver d'une même source de fréquence, f_0=10.23 MHz, commune sur les satellites (ARINC research corporation, 2000). De cette façon, on obtient f_{L1}=154·f_0, f_{L2}=120·f_0 et f_{L5}=115·f_0.

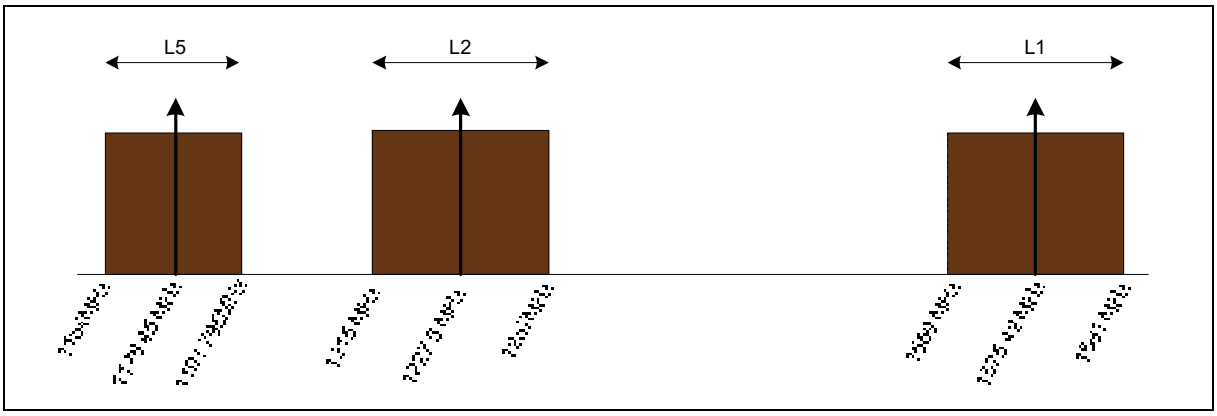

Figure 1.5 Le plan de fréquence des signaux GPS.

1.2.2 Spécifications du système Galileo

Le système de positionnement Galileo a été conçu principalement pour enlever la dépendance de l'Europe vis-à-vis des Etats-Unis.

Structure du système Galileo

Le système Galileo est composé de quatre segments indépendants (European Space Agency, 2006) : spatial, sol de contrôle, sol de mission et test des utilisateurs :

- Segment spatial : il est constitué de 30 satellites placés sur trois orbites circulaires à une altitude de 23,616 km. Chaque satellite comprend plusieurs horloges atomiques pour calculer le temps d'émission des signaux;
- Segment sol de contrôle : ce segment a pour tâche de surveiller et de contrôler les satellites, il est composé de deux centres localisés en Europe et de cinq autres stations qui ont la charge de maintenir les liaisons de télécommande et de télémesures avec les satellites;
- Segment sol de mission : ce segment est chargé de créer les messages de navigation diffusés par le satellite. Il permet aussi de localiser les éventuelles anomalies du système et prévenir les récepteurs via des messages envoyés par les satellites;

- Segment de test des utilisateurs : comme c'est le cas pour le GPS, cette partie est constituée par les récepteurs qui ne font que recevoir les signaux provenant des satellites.

Spécifications des signaux Galileo

Chaque satellite Galileo transmet des messages dans les signaux E1, E6, E5a et E5b (European Space Agency, 2006) :

- Signal E1 : ce signal est transmis sur la bande L1 comprenant deux canaux (E1-B et E1-C). Il inclut des données de navigation accessibles au grand public. Le canal E1-B transmet des données et le canal E1-C transmet des signaux pilotes. Les signaux pilotes permettent, comme indiquent leurs noms, de guider le récepteur pour l'acquisition du signal E1-B;
- Signal E6 : ce signal transporte des informations pour un but commercial. Il comprend deux canaux (E6-B et E6-C). Le premier étant un canal de données et le deuxième transmet des signaux pilotes;
- Signal E5a : ce signal est transmis sur la bande E5 et comprend deux canaux (données, pilote). Il transporte des données de navigation accessible pour le grand public;
- Signal E5b : ce signal est aussi transmis sur la bande E5 et comprend aussi deux canaux, un pour les données de navigation et un pour les signaux pilotes. Ce signal comporte des données non cryptées accessibles par tous les utilisateurs et des données cryptées réservées pour un usage commercial.

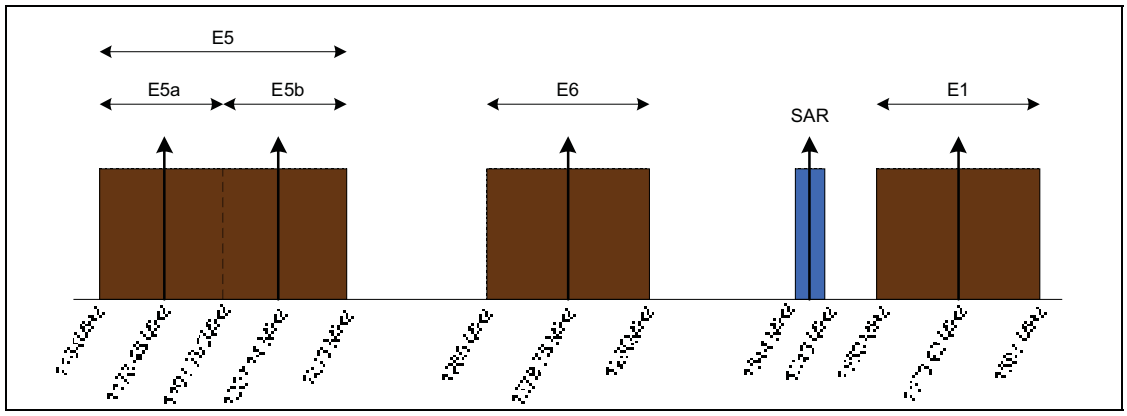

Figure 1.6 Plan de fréquence Galileo.

Comme c'est le cas pour les signaux GPS, le système Galileo utilise la technique DS-CDMA et par conséquent, les satellites émettent chaque type de signal dans une même bande de fréquence. Chaque satellite possède son propre code pour être différencié lors de la réception. La Figure 1.6 illustre les fréquences centrales et les bandes de fréquences des différents signaux Galileo. Ces informations sont résumées dans le Tableau 1.2 et 1.3.

Tableau 1.2 Fréquences porteuses des signaux Galileo
Tiré de (Galileo Open Service, 2006)

Signal	**Fréquence porteuse**
E5a	1176.45 MHz
E5b	1207.14 MHz
E5 (E5a+E5b)	1191.795 MHz
E6	1278.75 MHz
E1	1575.42 MHz

Tableau 1.3 Bande de réception et polarisation des signaux Galileo

Tiré de (Galileo Open Service, 2006)

Signal	Bande de réception	Polarisation
E5	51.150 MHz	Circulaire droite
E6	40.92 MHz	Circulaire droite
E1	24.552 MHz	Circulaire droite

La composition des deux signaux E5a et E5b est considérée en un seul signal E5 de fréquence centrale $\frac{f_{E5a} + f_{E5b}}{2}$. Les deux fréquences étant proches l'une de l'autre, ils peuvent être traités comme une seule bande large avec une implémentation spécifique au niveau des récepteurs. E5 n'est pas un signal à part entière, il est juste la composition des deux signaux E5a et E5b (European Space Agency, 2006).

1.3 Présentation globale du simulateur de signaux GNSS

Au lancement du projet, il n'existait pas de simulateur de signaux Galileo sur le marché. C'était une bonne initiative de démarrer le projet d'un simulateur hybride GPS/Galileo. Dernièrement, un tel simulateur a été mis sur le marché, mais il est encore trop coûteux (autour de 800,000 $) ce qui est stimulant pour notre projet en cours, présentant un coût nettement inférieur.

Le projet est très riche en matière de développement. Il inclut plusieurs sections qui opèrent ensemble. L'architecture est centrée autour d'une plateforme de développement à base de FPGA (*Field Programmable Gate Array*) qui offre une très grande flexibilité au niveau du développement vu que le système de positionnement Galileo était encore en phase de développement et était susceptible à des changements majeurs. Cette plateforme est reliée à un logiciel qui gère la nature de la simulation et effectue tous les calculs nécessaires au bon

déroulement du scénario. Enfin, une partie analogique RF (Radio Fréquence) complète le tout en générant les signaux désirés.

1.4 Architecture du simulateur

Comme décrit dans l'introduction, le simulateur comporte trois parties fondamentales : la partie logicielle, la partie de traitement des signaux (partie IF) et la partie RF. La Figure 1.7 illustre globalement l'architecture du simulateur GNSS.

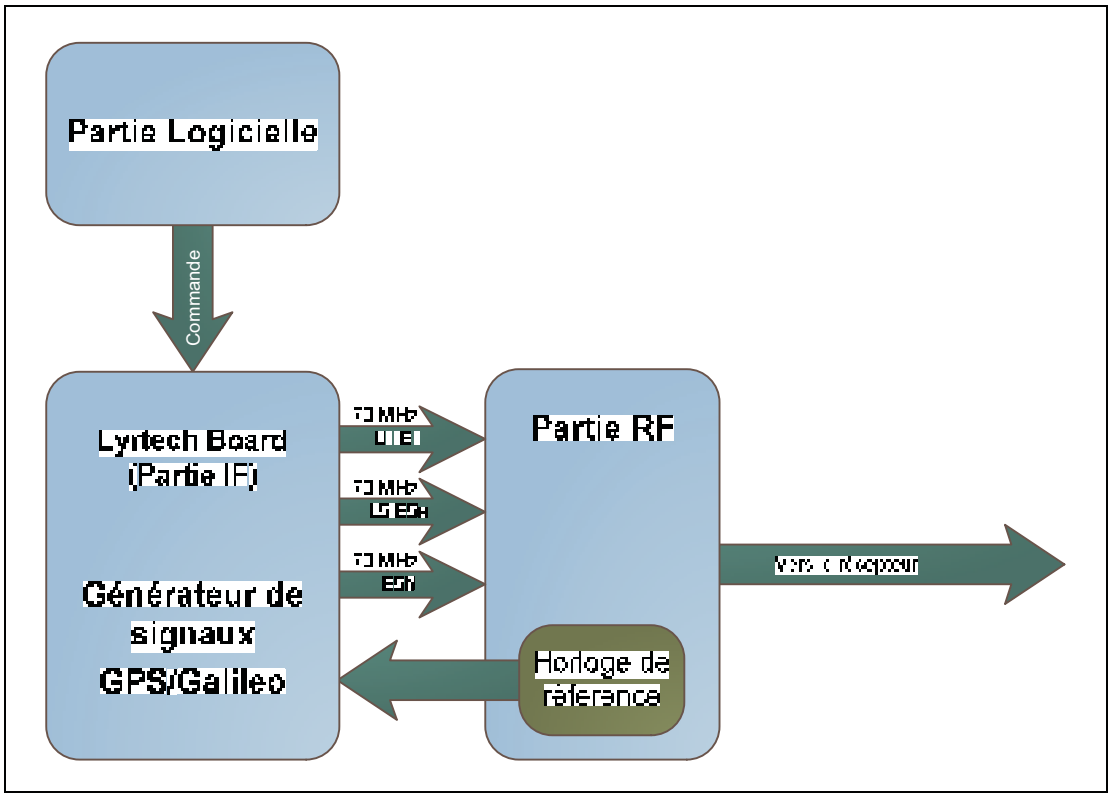

Figure 1.7 Structure du simulateur GNSS.

La totalité du simulateur est intégrée dans un boitier compact PCI (*Peripheral Component Interconnect*), qui comporte une carte mère compacte d'un PC embarqué, une carte à base de FPGA de la compagnie Lyrtech pour le traitement en temps réel des signaux ainsi qu'une carte RF développée au sein du LACIME. Les signaux E1, L5, E5a et E5b ne sont pas

implémentés dans la partie logicielle du simulateur. La Figure 1.8 présente une photo du simulateur actuel.

Figure 1.8 Le simulateur GNSS du LACIME.

1.4.1 Partie logicielle

La partie logicielle est responsable de la configuration du scénario de navigation et du calcul des paramètres nécessaires pour la simulation. Elle est composée d'un cœur de base en MATLAB et d'une partie C++. La partie MATLAB sert à exécuter tous les calculs nécessaires ainsi qu'à configurer la simulation. Elle intègre des interfaces d'utilisation (voir Figure 1.9 et Figure 1.10) permettant de définir le scénario de navigation et les paramètres primaires nécessaires à l'élaboration de la simulation. La section C++ du simulateur a pour rôle de mettre en forme les données calculées par MATLAB et les envoyer à la carte de traitement des signaux en temps réel via le port *PCI Express*. Une mise à jour de la partie logicielle est en cours de développement par un collaborateur afin de migrer tout le code en C++.

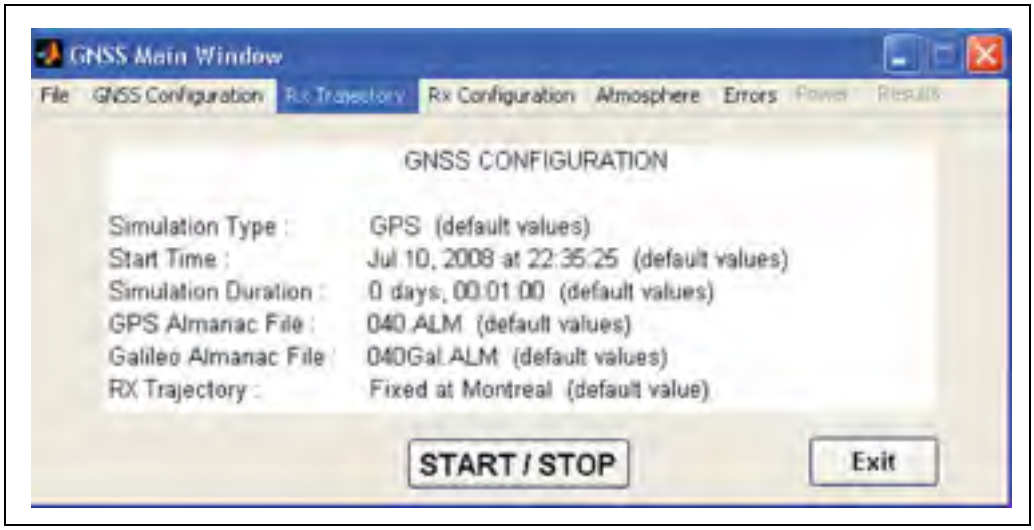

Figure 1.9 Interface principale du simulateur.

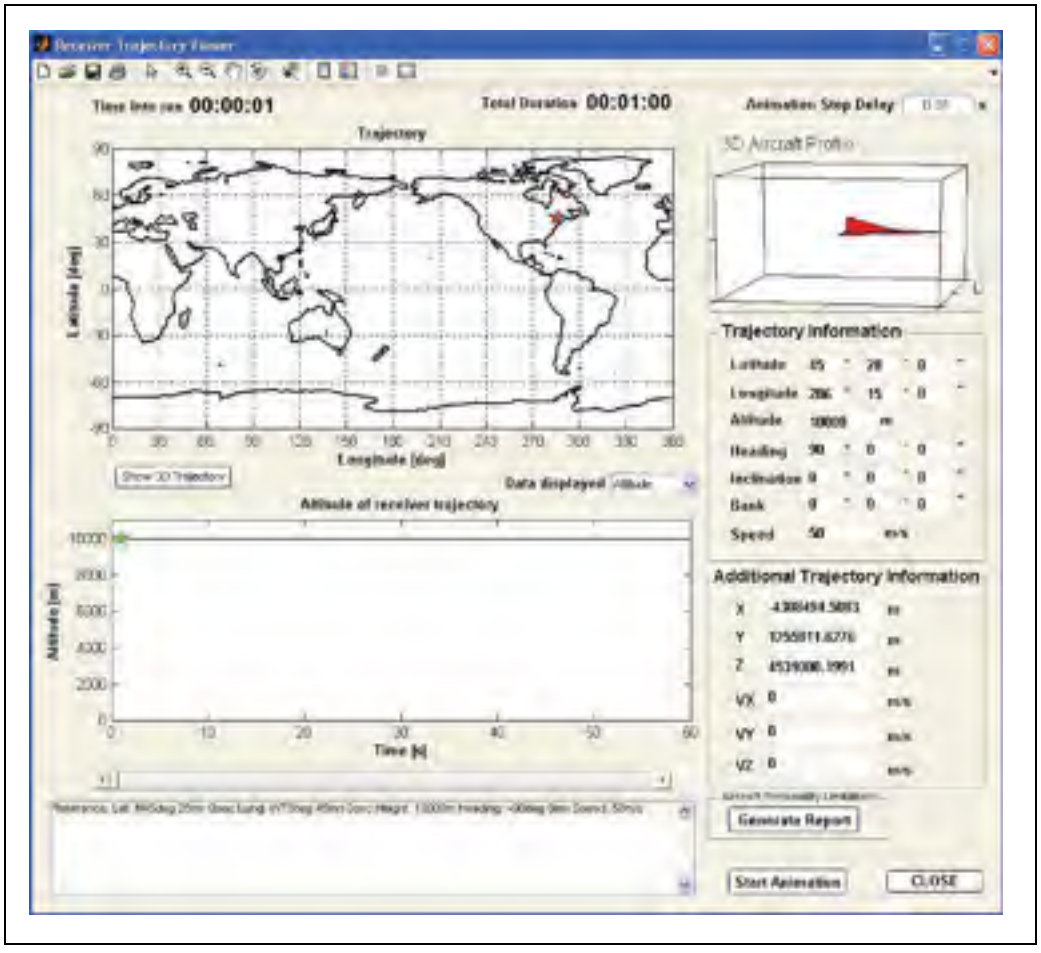

Figure 1.10 Interface du simulateur de trajectoire.

La partie logicielle permet aussi de piloter la partie RF du simulateur. En effet, l'utilisateur peut définir plusieurs paramètres se rapportant au fonctionnement de la partie analogique du simulateur à savoir, choisir l'horloge de référence du système (interne ou externe) et fixer la puissance des signaux GPS et Galileo. Le contrôle de puissance n'est pas implémentée dans l'ancienne version de la partie logicielle. Elle l'est dans la nouvelle version entièrement développée en C++ et sera traité dans le chapitre suivant.

1.4.2 Partie IF

La partie IF (*Intermediate Frequency*) est celle responsable du traitement numérique des signaux en temps réel. C'est une carte électronique à base de FPGA. Elle sert à mettre en forme les signaux GPS et Galileo (codage, modulation…) à partir des informations reçues de la partie logicielle. Les signaux sont donc générés à une fréquence intermédiaire de 70 MHz pour être, ensuite, convertit en analogique par un CNA (Convertisseur Analogique Numérique) ou DAC en anglais (*Digital to Analog Converter*). Ces signaux sont envoyés vers la carte RF, dernier maillon du simulateur. La partie IF sert aussi de relais de communication entre la partie logicielle et la partie RF du simulateur pour des fins de contrôle (ex : puissance des signaux).

1.4.3 Partie RF

La partie RF du simulateur a pour rôle de translater la fréquence des signaux (70 MHz) aux fréquences désirées GPS et Galileo. Cette partie est au cœur de nos travaux. Elle est étudiée plus en détail dans la section suivante de ce chapitre.

1.5 Description de la partie RF du simulateur

La partie RF consiste en trois transmetteurs RF qui permettent de translater les signaux reçus de la partie IF aux fréquences des signaux transmis par les satellites GPS et Galileo à savoir L1, L5, E1, E5a et E5b. Elle permet aussi de contrôler la puissance de ces signaux à sa sortie de façon à ce qu'ils arrivent au récepteur de test comme s'ils étaient reçus par leurs antennes

(c'est-à-dire très affaiblis). Cette section du simulateur a été en grande partie développée par un collaborateur qui a travaillé longuement sur le projet. J'ai principalement travaillé sur la partie de contrôle et d'ajustement de la puissance des signaux ainsi que sur le débogage de la carte RF finale.

La partie RF du simulateur est composée de trois sous chaines qui partagent des ressources communes. Chacune de ces chaines est assignée à un des signaux issus de la partie IF et les transpose aux fréquences suivantes :

- f=1575.42 MHz pour le signal GPS L1 et Galileo E1;
- f=1176.45 MHz pour les signaux GPS L5 et Galileo E5a;
- f=1207.14 MHz pour le signal E5b.

L'architecture détaillée de la partie RF est illustrée par la Figure 1.11.

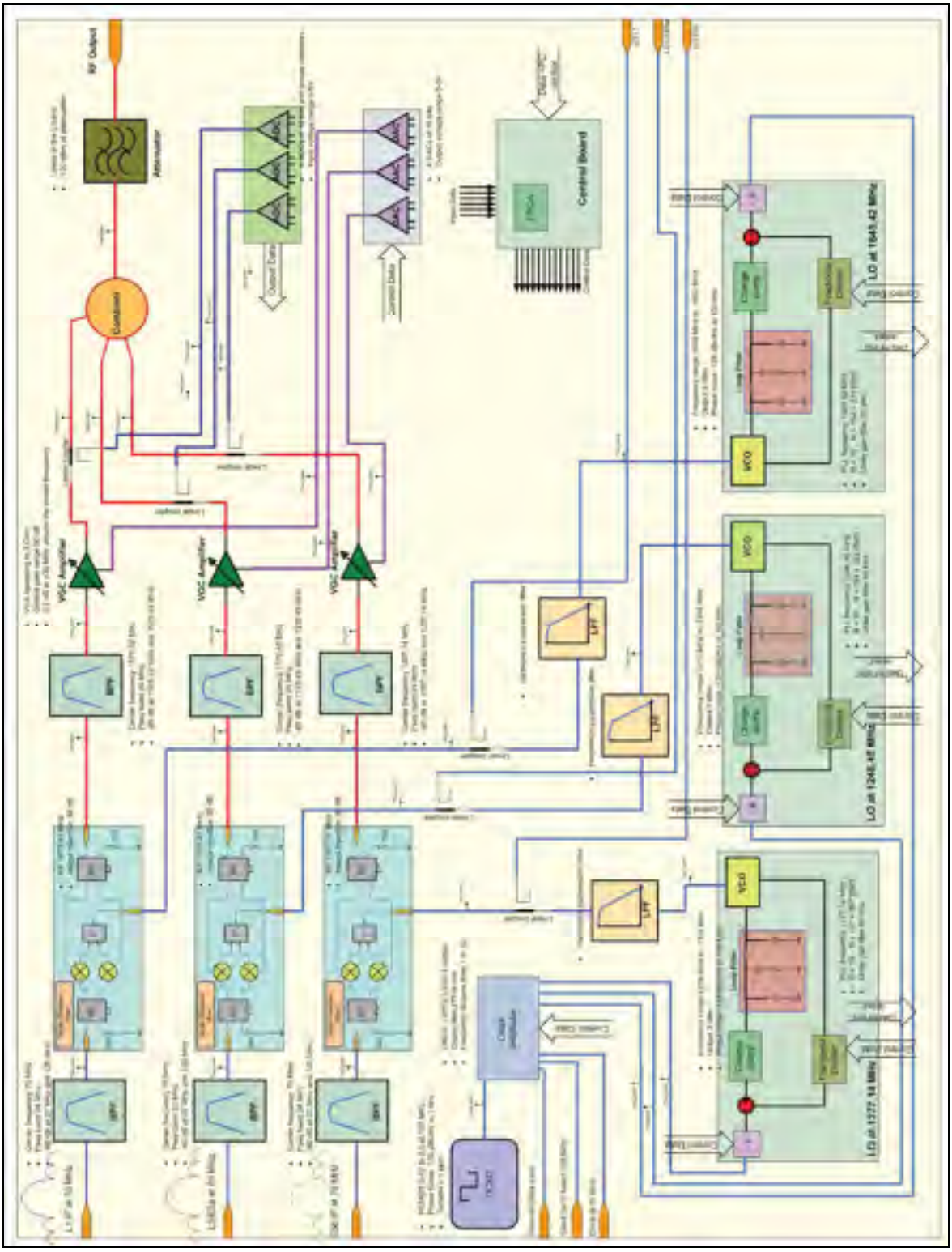

Figure 1.11 Architecture de la partie RF du simulateur.

Les composants communs aux chaines d'émissions sont les suivants :

- Horloge de référence : c'est une horloge OCXO (*Oven Controlled X-tal (Crystal) Oscillator*) de 100 MHz de haute qualité avec un faible bruit de phase et une très faible variation au cours du temps ce qui permet de s'approcher un peu de la qualité des horloges embarquées dans les satellites. Une tension de raffinement de la précision de la fréquence centrale de l'OCXO est disponible dans le composant et fixée expérimentalement, au niveau matériel, en fonction du design;
- Distributeur d'horloge : il permet de distribuer l'horloge vers différents points du simulateur en préservant sa qualité et son niveau de puissance. Ce composant est configurable et peut, donc, soit distribuer l'horloge interne de la carte, soit, pour plus de précision, une horloge externe fournie par l'utilisateur. L'horloge est transmise vers la partie IF du simulateur et elle est utilisée comme horloge de référence. Elle est transmise aussi vers les boucles à verrouillage de phase ou PLL en anglais (*Phase Locked Loop*) pour chacune de chaines d'émissions;
- Calibration et ajustement de la puissance : ce système comporte plusieurs composants qui permettent de calibrer le niveau de puissance des signaux RF ainsi que leurs niveaux à la sortie de la partie RF. Ce système présente une grande partie de nos travaux et sera détaillé dans le chapitre 2;
- Combineur : il permet de combiner les signaux issus des trois chaines d'émissions sur une seule ligne de transmission;
- Atténuateurs fixes : ce sont deux atténuateurs de 20 dB chacun. Ils sont placés après le combineur afin d'atténuer le signal final pour que sa puissance soit conforme à la réalité;
- CPLD (*Complex Programmable Logic Device*): il en existe deux sur la carte. Le premier a pour tâche de commander la tension de précision de l'horloge de référence, de configurer le distributeur d'horloge (horloge interne ou externe) et de programmer les trois PLL pour qu'ils verrouillent aux bonnes fréquences. Le deuxième CPLD est utilisé pour piloter le système de contrôle des puissances des signaux.

Les trois chaines d'émissions sont identiques sauf qu'ils sont adaptées à chacun des signaux GPS et Galileo. Les composants de ces chaines sont les suivants :

- Filtre IF : c'est un filtre passe-bande de fréquence centrale 70 MHz et d'une largeur de bande de 24 MHz. Il a une haute rejection hors bande qui filtre les harmoniques générés par le CNA;
- Boucle à verrouillage de phase : c'est un synthétiseur de fréquence qui génère l'Oscillateur local ou, en anglais, LO (*Local Oscillator*) requis pour chaque chaine d'émission. Les fréquences générées sont 1645,42 MHz pour le signal L1, 1246.45 MHz pour les signaux L5 et E5a et enfin 1277.14 MHz pour le signal E5b. Chaque PLL comporte (voir annexe III) :
  - Un synthétiseur de fréquence fractionnel qui, à partir d'un signal de référence (100 MHz dans notre cas), génère la fréquence désirée,
  - Un filtre de boucle conçu pour une minimisation du bruit de phase, et
  - Un VCO (*Voltage Controlled Oscillator*) : chaque chaine a son propre VCO à bande étroite qui permet d'obtenir de bonnes performances au niveau de la précision et du bruit de phase;
- Mélangeur à bande latérale unique ou en anglais SSBM (*Single Side Band Mixer*) : ce composant permet de transposer les signaux IF à la fréquence désirée. Son constituant de base est un modulateur quadratique (Modulateur IQ) qui, combiné à un coupleur 3 dB hybride 90°, permet d'atténuer un des deux signaux générés par le mélangeur (NorthWood Labs LL, 2003);
- Filtre RF : les signaux GPS et Galileo étant centrés sur des fréquences différentes, les filtres RF sont par conséquent différents. Ces filtres sont à base de quatre résonateurs (*Stepped impedance hairpin resonator*) qui sont couplés entre eux. Une multitude de paramètres sont contrôlés pour obtenir la bonne fréquence centrale et la largeur de bande requise. Ils ont été conçus et fabriqués au sein du laboratoire en technologie micro-ruban ou *micro-strip* en anglais (Yacine Adane, 2007); et
- Amplificateur/Atténuateur contrôlé par tension : ou en anglais VGA (*Voltage Controlled Amplifier/Attenuator*). Il assure l'ajustement du niveau de la puissance à la sortie de la

chaine RF. Ce composant est contrôlé par une tension continue (0 à 1.3 V) qui fixe le niveau du gain de l'amplificateur soit en atténuant le signal, soit en l'amplifiant.

Afin de réaliser une telle architecture, la partie RF a été développée en deux étapes. La première consistait à développer chaque sous-système à part et ensuite connecter le tout ensemble. Cette étape s'est avérée très importante, car elle a permis de tester chaque sous-système et de corriger les problèmes sans pour autant nuire à la totalité du système (réglages, adaptation d'impédance, déverminage des signaux numériques). Le prototype modulaire de la partie RF est présenté dans la Figure 1.12. La deuxième étape de développement consiste en l'intégration de tous les modules sur une seule et unique carte compacte et adaptée au boitier compact PCI du simulateur. La carte RF complète ainsi réalisée est illustrée par la Figure 1.13.

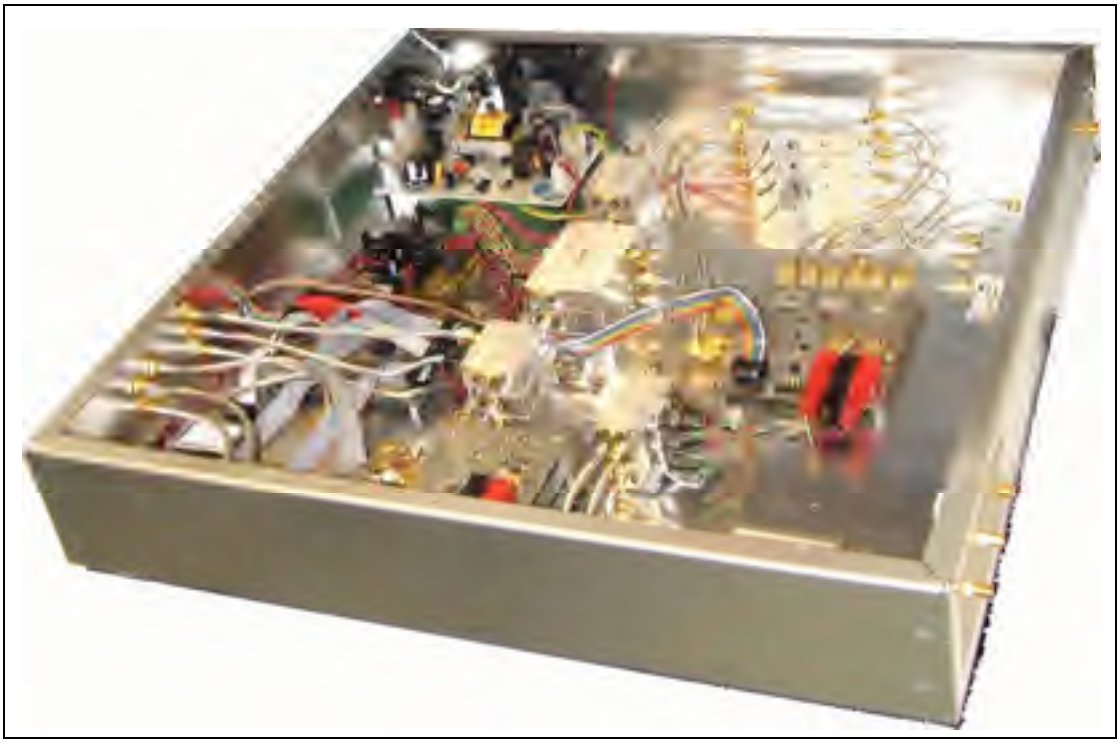

Figure 1.12 Plateforme RF modulaire.

Figure 1.13 Carte RF finale.

1.6 Défauts reliés au design RF final

Après la conception de la carte RF finale du simulateur, nous nous sommes rendu compte de quelques défauts au niveau du dessin du circuit imprimé et du choix de quelques composants.

1.6.1 Défauts reliés au dessin du circuit imprimé

La taille de certains composants n'a pas été prise totalement en compte lors du dessin du circuit imprimé de la carte RF ce qui a conduit à une difficulté lors de l'assemblage et de la soudure des composants sur la carte. Les problèmes que nous avons remarqués sont les suivants :

- L'horloge de référence touche le transformateur RF (*Balun*) sur la couche supérieure du circuit (encadré en blanc sur la Figure 1.14). Nous avons remédié à ce problème en enlevant une petite partie métallique de la coquille de l'horloge de référence;

- Un connecteur JTAG (*Joint Test Action Group)*, dédié pour un des deux CPLD de la carte, a été placé près du câble coaxial reliant la sortie de la PLL du signal L1 au modulateur. Lors de l'assemblage de la carte, nous nous sommes aperçus que le connecteur JTAG touchait le câble coaxial (Conducteur externe à la masse) et nous nous sommes rendu compte, lors du débogage de la carte, qu'il causait un court-circuit vu qu'une patte du connecteur JTAG est connectée à V_{DD}. Nous avons, alors, résolu ce problème en plaçant une gaine en caoutchouc sur le câble coaxial et nous avons incliné les pattes du connecteur JTAG pour qu'il soit accessible pour la reprogrammation du CPLD sans recourir au détachement du câble coaxial. La partie encadrée de la Figure 1.14 illustre le défaut et la correction réalisée;
- L'empreinte, ou en anglais *footprint*, du connecteur PCI express avec lequel la carte est connectée à la plate-forme a été mal dessiné. Lors de la soudure de ce connecteur, il s'est avéré que les trous, dans la carte RF, étaient plus petits que les pattes du connecteur. Nous avons, alors, aplati ces pattes par une pince pour qu'ils puissent se placer correctement sur la carte.

Ces problèmes ne causent pas une dégradation des performances de la carte RF du simulateur, elles sont juste des défauts mineurs à éviter lors d'une deuxième itération. Les solutions sont simples à réaliser. Pour le premier problème, il suffit de décaler un peu le transformateur RF et ajuster les connexions. De même pour le deuxième problème, il suffit de décaler un peu le connecteur JTAG ou de modifier son orientation. En ce qui concerne le dernier problème, il suffit d'augmenter le diamètre des trous dans l'empreinte du connecteur.

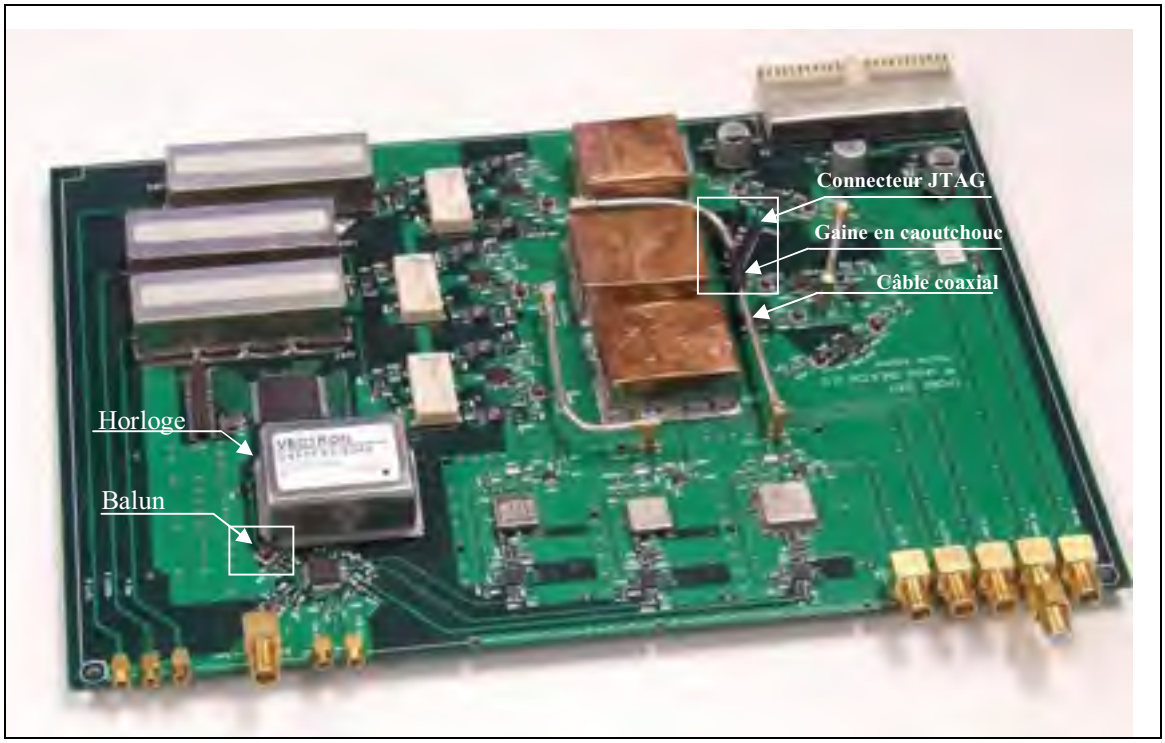

Figure 1.14 Défauts de dessin du circuit imprimé.

1.6.2 Défauts reliés aux choix des composants

Certains aspects de la carte ont été mal dimensionnés ce qui a causé quelques problèmes au niveau de la validation. Ces erreurs ont été contournées pour pouvoir accéder à des tests en RF. Elles sont au nombre de deux et sont les suivantes :

- Le régulateur de tension qui permet d'alimenter l'horloge de référence de la carte RF ne fournit pas assez de courant. Les spécifications données par le constructeur de l'horloge confirment que ce composant ne consomme pas plus que 500 mA, ce qui s'est révélé faux. Lors de la conception de cette carte, un régulateur de tension qui fournit un maximum de 500 mA a été utilisé ce qui est juste le nécessaire pour faire fonctionner ce composant. Avec la constatation qu'il consommait plus que 500 mA, nous avons, en premier lieu, décidé de mettre deux régulateurs de tension en parallèle (l'un au-dessus de l'autre). Cette solution permettait un fonctionnement normal de la carte, mais c'était juste une solution de dépannage. Après, nous avons cherché d'autres régulateurs de tension qui

pourraient remplacer le précédant et nous avons trouvé un seul et unique composant qui correspondait à tous les critères (package, *pin out*) et qui fournissait un courant de 700 mA au maximum, ce qui a permis de résoudre totalement le problème;

- Lors de sa réception, le signal GPS est très affaibli et se situe au-dessous du plancher de bruit. Dans le cas du simulateur, les signaux doivent sortir à un niveau très bas pour les injecter directement dans le récepteur de test. Pour arriver à cette fin, deux atténuateurs RF, d'une valeur de 20 dB chacun, ont été placé juste avant le connecteur de sortie de la carte RF. Avec ces conditions, les niveaux de puissance des signaux sont encore hauts et n'atteignent pas les niveaux nominaux des signaux GPS et Galileo qui se situent autour de -130 dBm. En effet, le niveau de puissance à la sortie est autour de -70 dBm. En utilisant le VGA qui permet de fournir une atténuation supplémentaire de 30 dB, nous arrivons à un seuil de -100 dBm ce qui est loin des spécifications. Le choix de ces composants n'est pas une erreur en tant que telle, car augmenter les valeurs d'atténuations conduirait à la difficulté de visualiser les signaux sur un analyseur de spectre. Le problème ici, est qu'un espace pour ajouter plus d'atténuateurs n'a pas été prévu. Pour pallier ce problème, nous avons utilisé un atténuateur externe contrôlable qui permet d'ajouter les 60 ou 70 dB d'atténuations. Cette solution est seulement temporaire, car le produit final doit s'affranchir d'un tel atténuateur. Nous avons effectué une recherche de composants chez le même fournisseur d'atténuateur et nous en avons trouvé un, d'une valeur de 30 dB d'atténuation, susceptible de remplacer l'ancien. Nous obtenons alors, sans utiliser l'atténuation supplémentaire du VGA, un niveau de puissance de -90 dBm. Il faut donc, ajouter encore 40 dB d'atténuation sans recourir au VGA, car sa fonction principale est de contrôler la puissance des signaux sur une plage aux alentours de 40 dB. La solution était donc de rechercher d'autres atténuateurs qu'on pourrait insérer dans la carte. Le même fournisseur de composants offre une gamme d'atténuateurs qui peuvent se connecter à une trace linéaire. Il faudrait alors, gratter le cuivre sur la carte, couper la trace et insérer un dernier atténuateur pour atteindre les spécifications. Cette approche permet de régler le problème tout en conservant la carte actuelle. Ces modifications n'ont pas été effectuées sur la carte pour des fins de tests. Néanmoins, les composants ont été choisis et il ne reste qu'à les placer.

Au moment de la rédaction de ce mémoire, la carte est fonctionnelle et permet d'effectuer les tests requis, mais un design final s'impose pour éliminer, d'une façon définitive, les problèmes cités ci-dessus. Il faut aussi penser à changer le CPLD responsable du contrôle de puissance car le design actuel est inséré de justesse. Dans cette partie, ma tâche était d'aider à déboguer la carte RF, aider à trouver des solutions à court terme et trouver les solutions à long terme pour faire un design final sans erreurs.

1.7 Conclusion

Ce chapitre a donné un aperçu sur les systèmes GPS et Galileo du point de vue de la structure globale et spécifications des signaux. Nous avons aussi décrit le projet GNSS en cours de développement. Nous avons mis l'accent sur la partie RF. Elle a été développée en deux étapes. Une modulaire qui consiste en la mise en œuvre de chaque sous-système à part et de combiner le tout à la fin. La deuxième s'est effectuée après le débogage du premier prototype et se présente comme l'intégration de tous les sous-systèmes sur une seule carte s'insérant dans un boitier compact PCI. Le simulateur doit permettre à l'utilisateur de créer le scénario adéquat afin de réaliser ses tests. Il doit être en mesure de reproduire le niveau des puissances des satellites ainsi que le niveau du bruit à la réception. Ceci donnera un control sur la qualité des signaux. Le chapitre suivant traitera le système de contrôle des puissances des signaux issues du simulateur GNSS.

Equation Chapter 2 Section 1

CHAPITRE 2

SYSTÈME DE CONTRÔLE DE PUISSANCE RF

Le but de construire un simulateur de signaux GPS et Galileo est de permettre aux industriels et aux chercheurs de tester leurs équipements (récepteurs). Pour un fonctionnement de base, il faut reproduire toutes les conditions qui peuvent affecter les signaux lors de leur propagation directe, en vue directe, des satellites vers les récepteurs. Mais cela ne suffit pas, car il faut prévoir la possibilité de tester les récepteurs sous certaines conditions, à savoir, un milieu urbain (multi-trajet) ou encore dans des positions où le signal peut être fort (C/N_0 supérieur à 50 dBHz) ou très faible (C/N_0 inférieur à 35 dBHz). Un des critères d'un récepteur est son seuil de détection. En vue directe, le niveau d'un signal GPS ou Galileo ne varie pas trop au cours du temps (quelques dB en fonction des positions des satellites et du récepteur). Mais dans un milieu où il y a des obstacles, le signal peut être très affaibli. L'utilisateur doit alors déterminer la sensibilité de son récepteur, c'est-à-dire la puissance minimale avec laquelle il peut générer une position. Pour cela, un système de contrôle des niveaux des signaux est requis. Ce chapitre traite la conception et la réalisation d'un tel système.

2.1 Description du système de contrôle de puissance

Les signaux GPS et Galileo ont des niveaux de puissances minimales et maximales spécifiés par les normes (voir
Tableau 2.1). Deux facteurs essentiels influent sur le niveau du signal à la réception. Le premier facteur est connu, il dépend de la position des satellites (ils ne sont pas à la même distance du récepteur) ainsi que l'antenne du récepteur. Les niveaux de puissances des satellites, par rapport à un niveau de référence fixé au préalable, sont contrôlés par la partie logicielle du simulateur et sont exécutés par la partie IF. Le deuxième facteur est aléatoire. Il dépend du milieu où se trouve le récepteur (atténuation due aux obstacles). L'atténuation des signaux peut, dans ce cas, varier de quelques dB (dépendant de la position des satellites et du récepteur) à quelques dizaines de dB (atténuation causé par les obstacles). Le simulateur de

constellation GPS et Galileo doit s'adapter à ces conditions et doit offrir à l'utilisateur un contrôle sur ce type d'information. Nous avons alors, développé un système électronique de contrôle de puissance piloté par logiciel qui permettrait à l'utilisateur de fixer le niveau de puissance désiré au cours de la simulation. Ce système est implémenté au niveau de la partie RF du simulateur. Il est inséré juste avant le combineur des signaux à la sortie. Ainsi, nous pouvons contrôler chacun des signaux séparément (L1/E1, L5/E5a et E5b). Avec son mécanisme de rétroaction, le système de contrôle de puissance est une boucle fermée reconfigurable qui permet d'effectuer plusieurs opérations tel que détaillé dans la section suivante.

Tableau 2.1 Niveaux de puissance des signaux GPS et Galileo à la réception
Tiré de (ARINC research corporation, 2000) (ARINC Incorporated, 2001) (European Space Agency, 2006)

Signal		**Niveau maximal (dBW)**	**Niveau minimal (dBm)**
GPS L1	C/A	-153	-130
	P (Y)	-156	-133
GPS L5		-150	-127
Galileo E1		-150	-127
Galileo E5		-148	-125

2.2 Architecture et fonctionnement du système de contrôle de puissance

Le système de contrôle de puissance des signaux GPS et Galileo du simulateur est une sorte d'appareil de mesure automatique qui permet d'ajuster les trois puissances de la chaine RF aux niveaux défini par l'utilisateur à travers l'interface logicielle. Il possède l'architecture d'un contrôleur automatique de niveau ou en anglais ALC (*Automatic Level Control*) numérique piloté par un CPLD (*Complex Programmable Logic Device*). La Figure 2.1 illustre l'architecture du système de contrôle de puissance. L'entrée du système est un amplificateur/atténuateur contrôlable par tension (VGA) qui permet d'ajuster le niveau du signal. Afin de détecter la puissance, des coupleurs 20 dB ont été placés aux sorties des

VGA. Ces coupleurs prélèvent des échantillons des signaux qui sont injectés dans un multiplexeur dont le rôle est de basculer entre les trois signaux au besoin. Le détecteur de puissance placé en aval du multiplexeur permet, quant à lui, de mesurer la puissance du signal à son entrée. Il génère une tension continue proportionnelle au niveau du signal. Cette tension est ensuite convertie en numérique via un CAN (Convertisseur Analogique Numérique) ou en anglais ADC (*Digital to Analog Converter*) et envoyée au CPLD. Ce dernier analyse l'information et envoie une commande au convertisseur numérique analogique ou en anglais DAC (*Digital to Analog Converter*) qui délivre la tension de commande adéquate au VGA d'intérêt pour réguler son gain.

Cette architecture est très flexible car elle peut fonctionner en boucle fermée comme en boucle ouverte. Le système peut alors être configuré de plusieurs manières afin de fonctionner à différents modes selon les besoins de l'utilisateur. Dans ce qui suit, nous décrivons sommairement les quatre modes de fonctionnement qui ont été développés et testés.

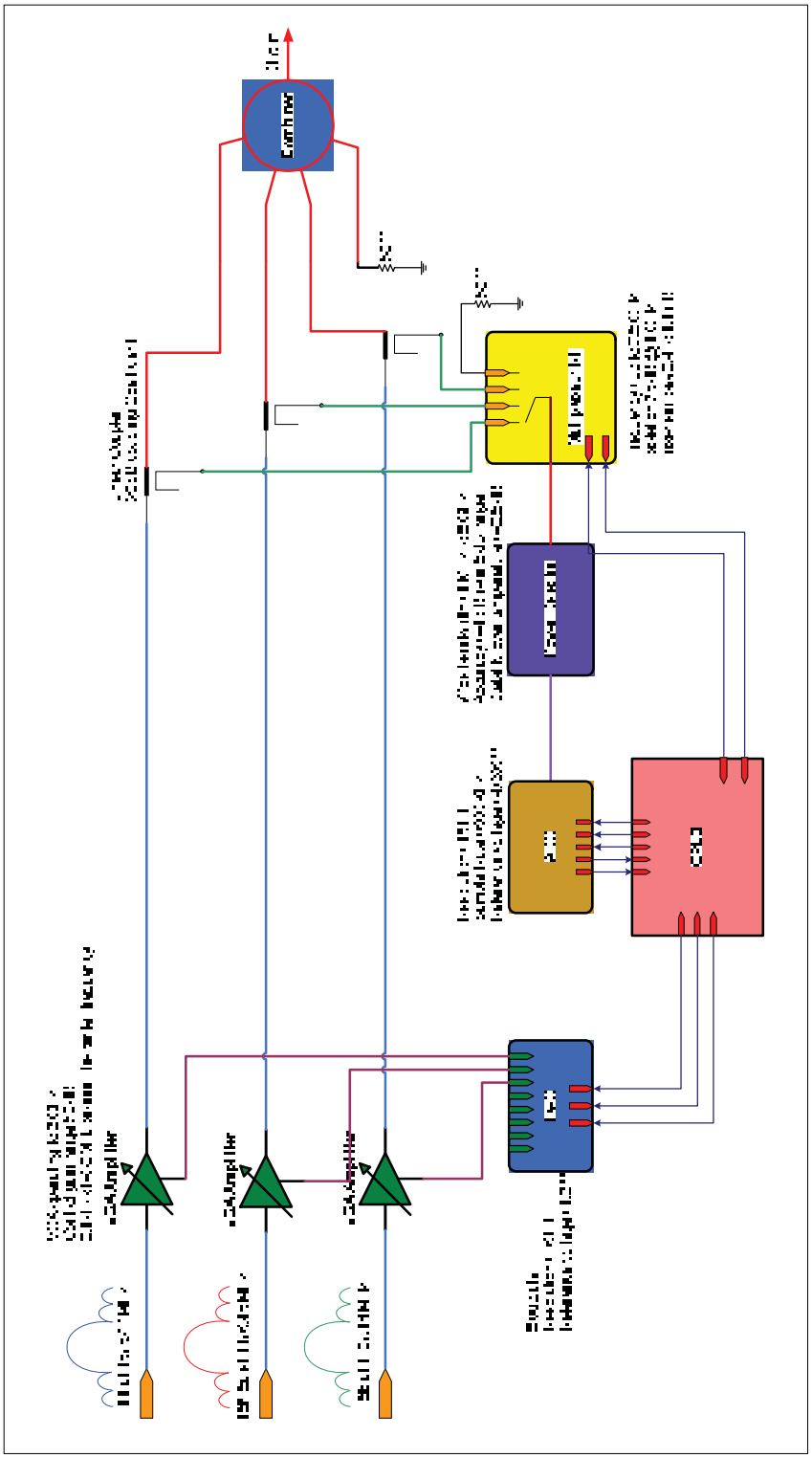

Figure 2.1 Architecture du système de contrôle de gain.

2.2.1 Calibration de la puissance

La précision du niveau des signaux GPS et Galileo est importante à fin de permettre à l'utilisateur pour de déterminer la sensibilité du récepteur et de sa robustesse. Les composants RF possèdent des caractéristiques bien définies dans les conditions normales et ces performances peuvent se dégrader au cours du temps. Le système de contrôle de puissance permet d'ajouter une option de calibration de la puissance des signaux pour une plus grande précision. Cette option est activée par l'utilisateur au besoin et son fonctionnement se déroule de la façon suivante. L'activation de cette option ordonne à la partie logicielle de générer les informations nécessaires pour la construction du signal d'un seul satellite par la partie IF. La partie logicielle envoie aussi, une commande au système de contrôle de puissance pour qu'il s'apprête à cette opération. La calibration est effectuée pour un seul signal à la fois (soit L1/E1, L5/E5a ou E5b). Le signal est donc capturé par le système et la boucle de contrôle s'active pour aligner le signal avec le niveau de référence spécifié (voir la Figure 2.2). Quand le niveau du signal est ajusté, le système renvoie une commande à la partie logicielle l'informant du succès ou de l'échec de l'opération. Dans le cas d'un succès la nouvelle valeur de la tension de référence du VGA est stockée. La partie logicielle dispose ainsi, des informations nécessaires pour un fonctionnement en mode « contrôle de gain ».

Une fonction de calibration automatique est intégrée dans la partie logicielle du simulateur et peut être activée ou non par l'utilisateur. Cette fonction permet de lancer la calibration de la chaine RF automatiquement à l'ouverture du logiciel.

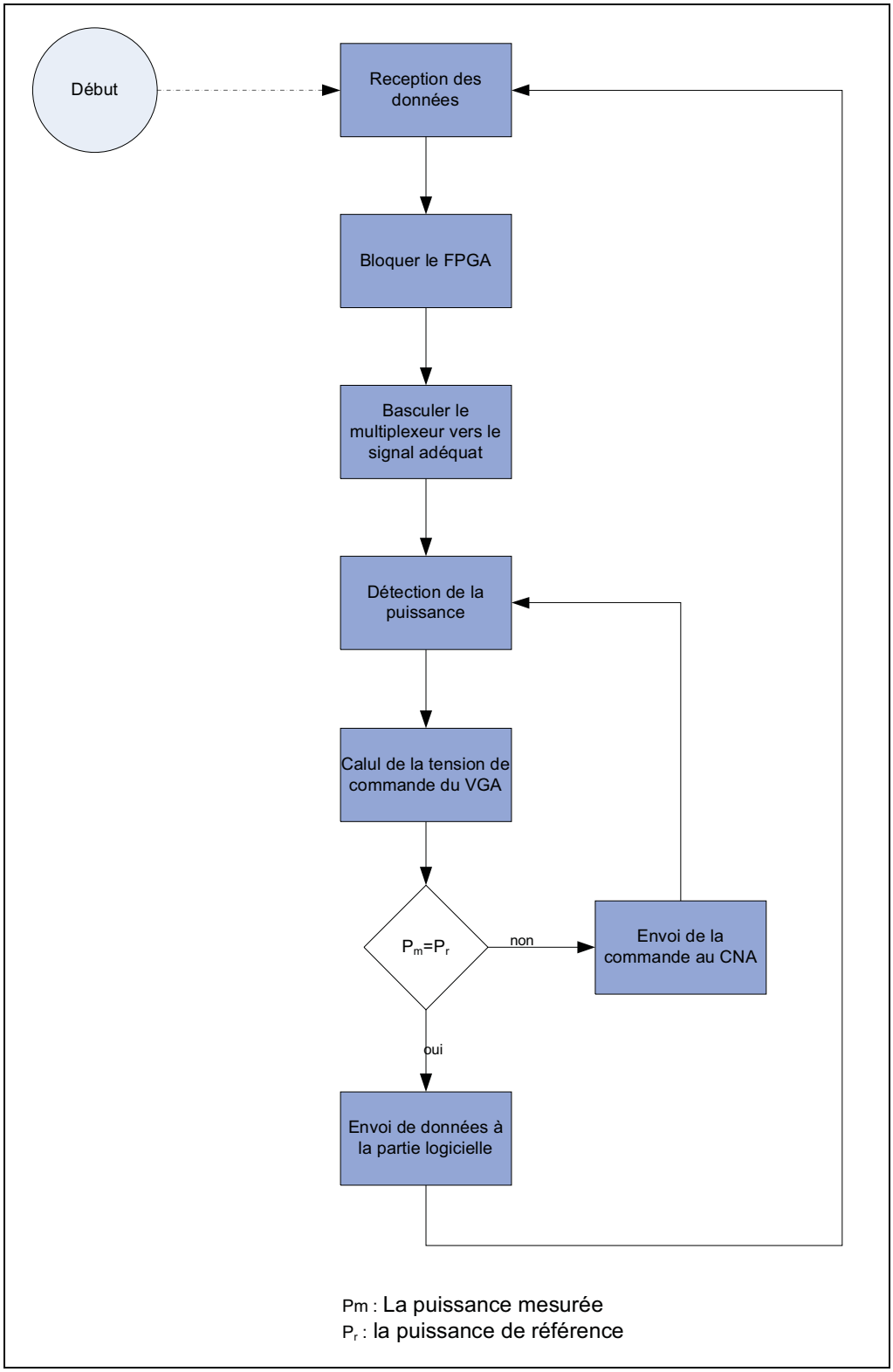

Figure 2.2 Fonctionnement du mode « Calibration ».

2.2.2 Contrôle de gain

Le contrôle de gain est une opération qui permet d'ajuster les gains des trois VGA par rapport aux puissances de références pour les signaux L1/E1, L5/E5a et E5b. Il suffit donc, de spécifier des références de gains qui permettent de générer les niveaux nominaux des puissances des signaux. L'utilisateur a la possibilité de changer cette référence afin de tester son récepteur sur plusieurs puissances. L'information est traitée par la partie logicielle qui transfère la valeur de la tension de commande du VGA, d'après la calibration, vers le système de contrôle de puissance via la partie IF. La Figure 2.3 illustre les différentes étapes de changement du niveau de référence.

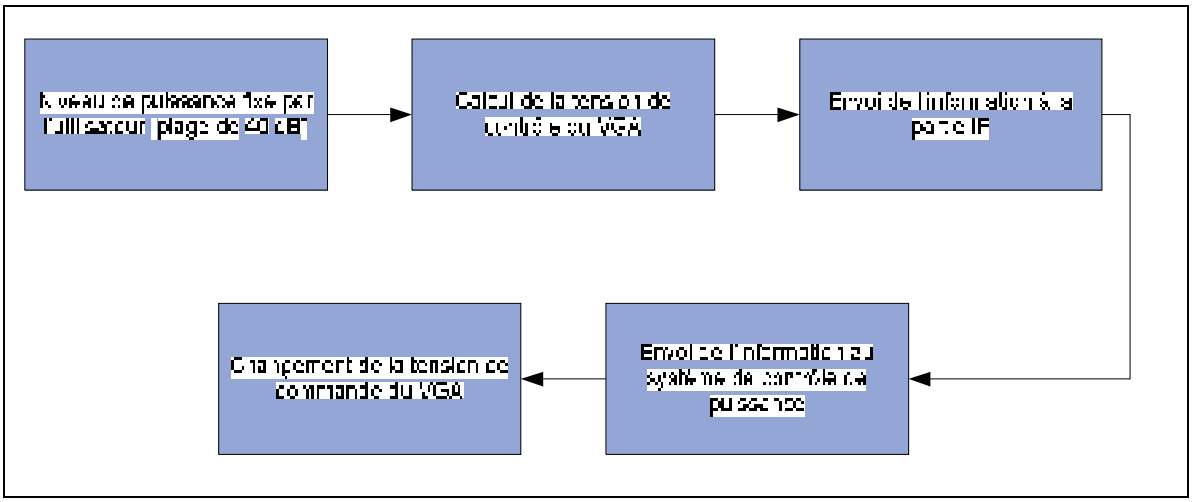

Figure 2.3 Fonctionnement du mode « contrôle de gain ».

2.2.3 Contrôle automatique de puissance

Comme le système est conçu pour fonctionner en boucle, il peut donc agir comme un contrôleur automatique de puissance durant la simulation. En effet, dans ce mode, le CPLD effectue un ajustement du gain en fonction de la puissance de référence fixée au préalable par l'utilisateur.

L'ajustement de la puissance des trois signaux L1/E1, L5/E5a et E5b s'effectue en boucle et suit les étapes illustrées par la Figure 2.4. Le nombre de signaux à ajuster dépend du scénario de simulation. Il peut y avoir un seul signal, comme il peut en avoir plusieurs en même temps (un maximum de trois pour la version actuelle de la carte RF). Comme dans toute boucle d'asservissement, le système doit avoir l'état précédant et l'état final comme entrée pour pouvoir fonctionner. Dans le cas de plusieurs signaux, le CPLD doit être en possession des tensions appliquées aux VGA (l'état précédant) et les puissances de références (l'état final). Le CPLD n'ayant pas une telle capacité de mémoire, nous avons alors opté pour un contrôle des boucles par le biais du FPGA. La boucle de contrôle n'est donc pas fermée au niveau du CPLD. Au lieu d'enregistrer les valeurs des puissances de références, le CPLD les demande aux FPGA à itération (voir Figure 2.4).De cette manière il n'a pas à les enregistrer et mémorise juste les tensions appliqués aux VGA.

Dans le cas de trois signaux, le système fonctionne comme suit. Après avoir reçu les trois trames de commandes, le FPGA bascule en mode de contrôle du CPLD. Il va prendre en charge de fermer la boucle d'asservissement. Le FPGA envoie la première commande qui correspond au mode d'ajustement automatique de la puissance du premier signal. Cette trame inclut le niveau de référence du signal en question. Le CPLD reçoit cette trame et bascule en mode exécution tout en bloquant le FPGA d'émettre quoi que ce soit. Le CPLD ordonne le multiplexeur de basculer sur le signal en question pour qu'il puisse détecter sa puissance. Après cette étape, le CPLD calcule la nouvelle valeur de tension de commande du VGA et la lui envoie. Ceci termine une itération de la boucle pour un seul signal. Il faudrait donc maintenant passer au signal suivant. Le CPLD bascule alors en mode réception et enlève le blocus qu'il à mis sur le FPGA. Ce dernier comprend alors que la commande du premier signal à été exécuté et envoie la deuxième trame de commande qui correspond au second signal. La même opération est effectuée pour le deuxième signal. Le FPGA envoie la troisième trame du dernier signal et le CPLD effectue toujours la même opération. L'ajustement de la valeur de la tension de commande du dernier VGA clôture une itération de l'opération de contrôle automatique de puissance et le cycle recommence. Ce cycle n'est arrêté que si le FPGA reçoit une nouvelle commande à exécuter.

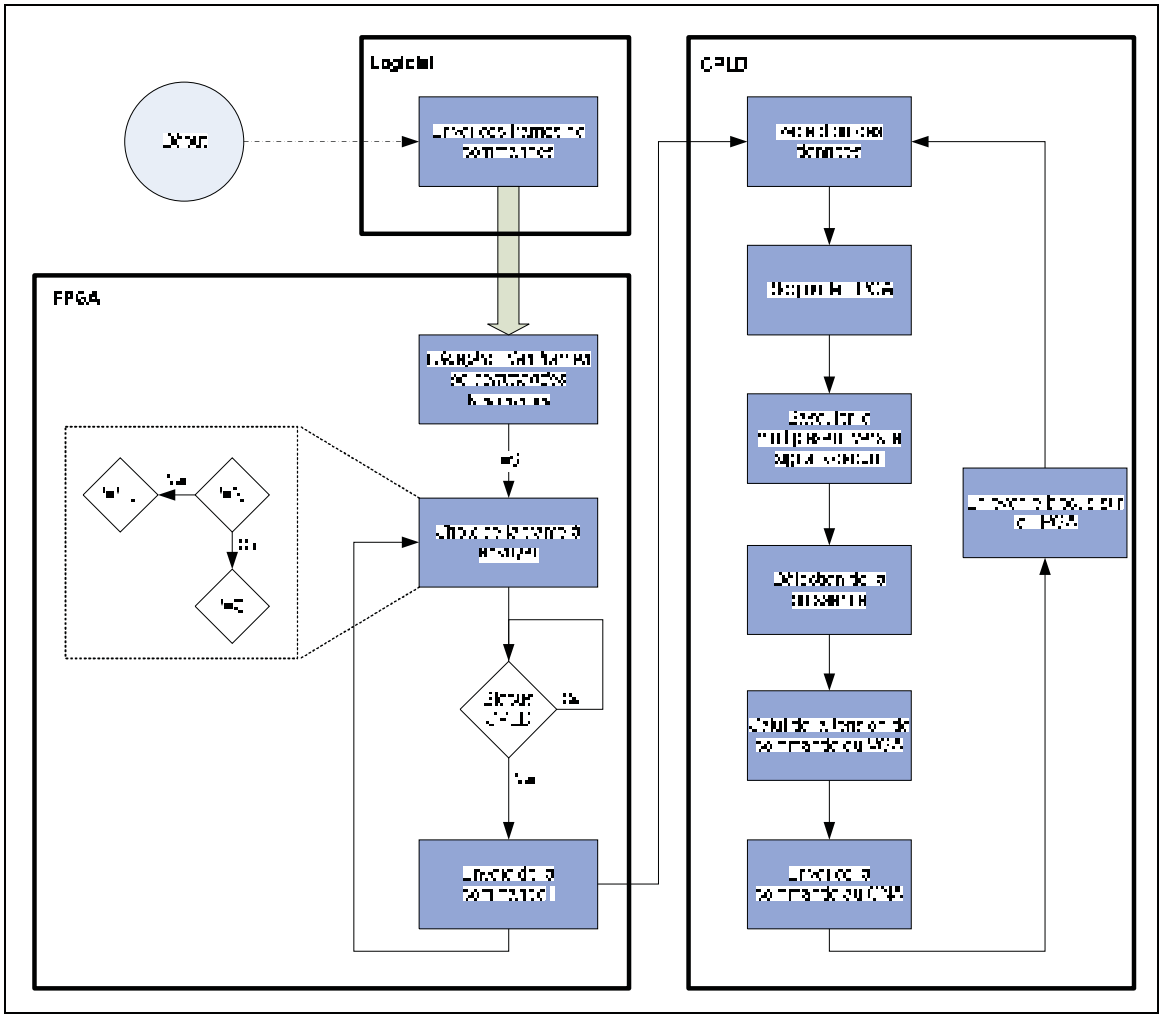

Figure 2.4 Fonctionnement du mode « contrôle automatique de puissance ».

2.2.4 Contrôle de puissance dans le temps

Le système de contrôle de puissance permet d'émuler un profil d'évanouissement d'un canal, soit des atténuations ou encore des augmentations inattendues du niveau de puissance, et ceci d'une façon variable dans le temps. Cela permet de tester le comportement du récepteur face à des variations brusques du canal. L'utilisateur crée alors un scénario dans lequel il spécifie plusieurs événements successifs. Chaque événement contient le gain désiré par rapport au niveau de puissance de fonctionnement (niveau de fonctionnement fixé pour le mode

« contrôle de gain » ou « contrôle automatique de puissance »), l'heure du début de l'événement et l'heure de sa fin.

La partie logicielle crée les trames de commande, selon le protocole de communication décrit dans la section 2.5.1, du scénario en question. Ensuite, elle exécute le premier événement en envoyant au système de contrôle de puissance la commande adéquate. Après l'écoulement de la durée de l'événement, la partie logicielle envoie la commande suivante et ainsi de suite.

2.3 Choix des composants pour la réalisation du système de contrôle de puissance

Le système de contrôle de puissance est une carte électronique comportant plusieurs composants illustrés dans l'architecture de la Figure 2.1. Ces composants sont choisis sur le marché des pièces électroniques et sont issus de plusieurs constructeurs :

- Les coupleurs : XC0900A-20S et XC1900A-20S d'Anaren ;
- Le multiplexeur 4:1 : ADG904 d'Analog Devices ;
- Le détecteur de puissance : AD8318 d'Analog Devices ;
- Le convertisseur analogique numérique : ADS8380 de Texas Instrument ; et
- Le convertisseur numérique analogique : AD5668 d'Analog Devices.

Tous ces composants, excepté les coupleurs, requièrent une alimentation. Pour cela, des régulateurs de tension ont été utilisés pour stabiliser les tensions. Quatre niveaux de tensions sont requis :

- 5V : LT1761-5V (Linear Technology) pour le détecteur de puissance et pour l'alimentation analogique pour le CAN et pour un AOP (amplificateur opérationnel);
- 3.3V : LT1761-3.3V (Linear Technology) pour l'alimentation numérique du CAN et pour le CNA;
- 1.8 V : LT1761-1.8V (Linear Technology) pour le multiplexeur; et
- -5 V : LT1964 (Linear Technology) pour un AOP

Les composants fondamentaux possèdent plusieurs propriétés qu'il faudra connaître et plusieurs paramètres dont il faut tenir compte pour le bon fonctionnement du système. Dans les sections suivantes, nous détaillerons tous ces composants et leurs caractéristiques.

2.3.1 Les coupleurs

Le système de contrôle de puissance intègre trois coupleurs (voir Figure 2.1). Nous avons utilisé deux coupleurs différents pour pouvoir couvrir toute la plage de fréquence. Le XC0900A-20S est utilisé pour les signaux aux fréquences 1176.45 MHz et 1207.14 MHz tandis que le XC1900A-20S est utilisé pour le signal de fréquence 1575.42 MHz.

Le XC0900A-20S est conçu pour fonctionner sur des fréquences allant jusqu'à 1 GHz et XC1900A-20S est conçu pour fonctionner sur une bande de fréquence allant de 1.7 GHz à 2 GHz (Anaren Inc, 2006b) (Anaren Inc, 2006a). Ces performances sont assurées par le constructeur. Mais en réalité, ces composants peuvent aller au-delà des bandes de fréquences citées, mais avec une légère dégradation des performances. La Figure 2.5 illustre le PCB sur lequel les tests des deux coupleurs ont été effectués. Ce PCB (*Printed Circuit Board*) a été réalisé avec l'outil Protel dont la fonction principale est le design des cartes électroniques.

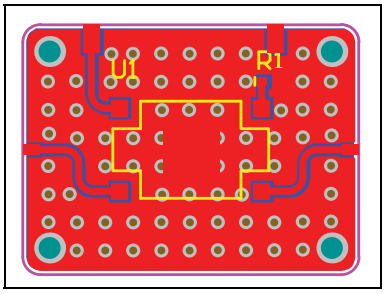

Figure 2.5 PCB de test des coupleurs.

Le substrat utilisé dans ce PCB est le RO3006 (Rogers Corp) qui possède les caractéristiques résumées dans le Tableau 2.2.

Tableau 2.2 Caractéristiques du substrat RO3006

Constante diélectrique ε_r	6.15 ±0.15
Facteur de dissipation	0.002
Épaisseur	25 mil

La largeur des lignes coplanaires 50 Ω a été calculée à l'aide de l'outil LineCalc du simulateur ADS (**A**dvanced **D**esign **S**ystem). En prenant une séparation de 10 mil entre la ligne et la masse, la largeur de la ligne est alors 25mil. Ces coupleurs ont été testés à l'aide d'un analyseur de réseau. Les paramètres S11, S21 et S31 relatifs aux ports 1, 2 et 3 des deux coupleurs (Figure 2.6) sont représentés dans les Figure 2.7 et Figure 2.8.

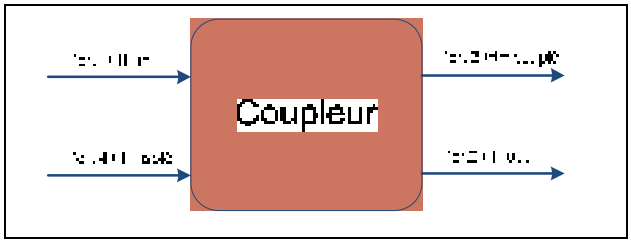

Figure 2.6 Schéma bloc des coupleurs.

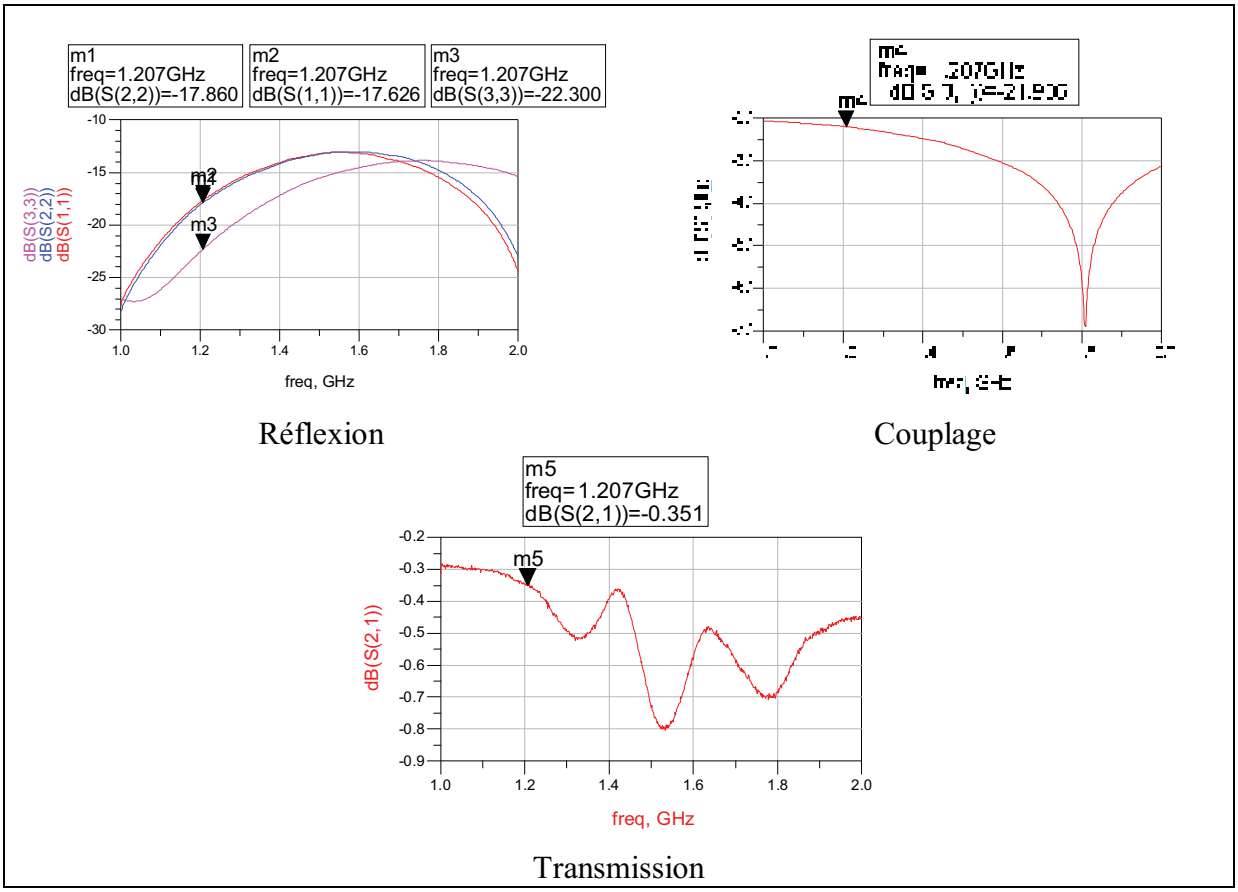

Figure 2.7 Paramètres « S » du coupleur XC0900A-20S.

Les paramètres « S » relatifs aux réflexions et aux couplages nous donnent une information sur les performances des deux coupleurs vu qu'ils sont utilisés sur des fréquences non certifiées par le constructeur. Ils servent à déterminer le niveau de puissance à l'entrée du système. Les paramètres de transmissions quantifient les pertes dans les coupleurs qui seront comptabilisées lors de l'ajustement des niveaux de puissances. Les coupleurs possèdent une bonne isolation. Elle est de l'ordre de -35 dB dans les plages de fréquence des signaux GPS et Galileo.

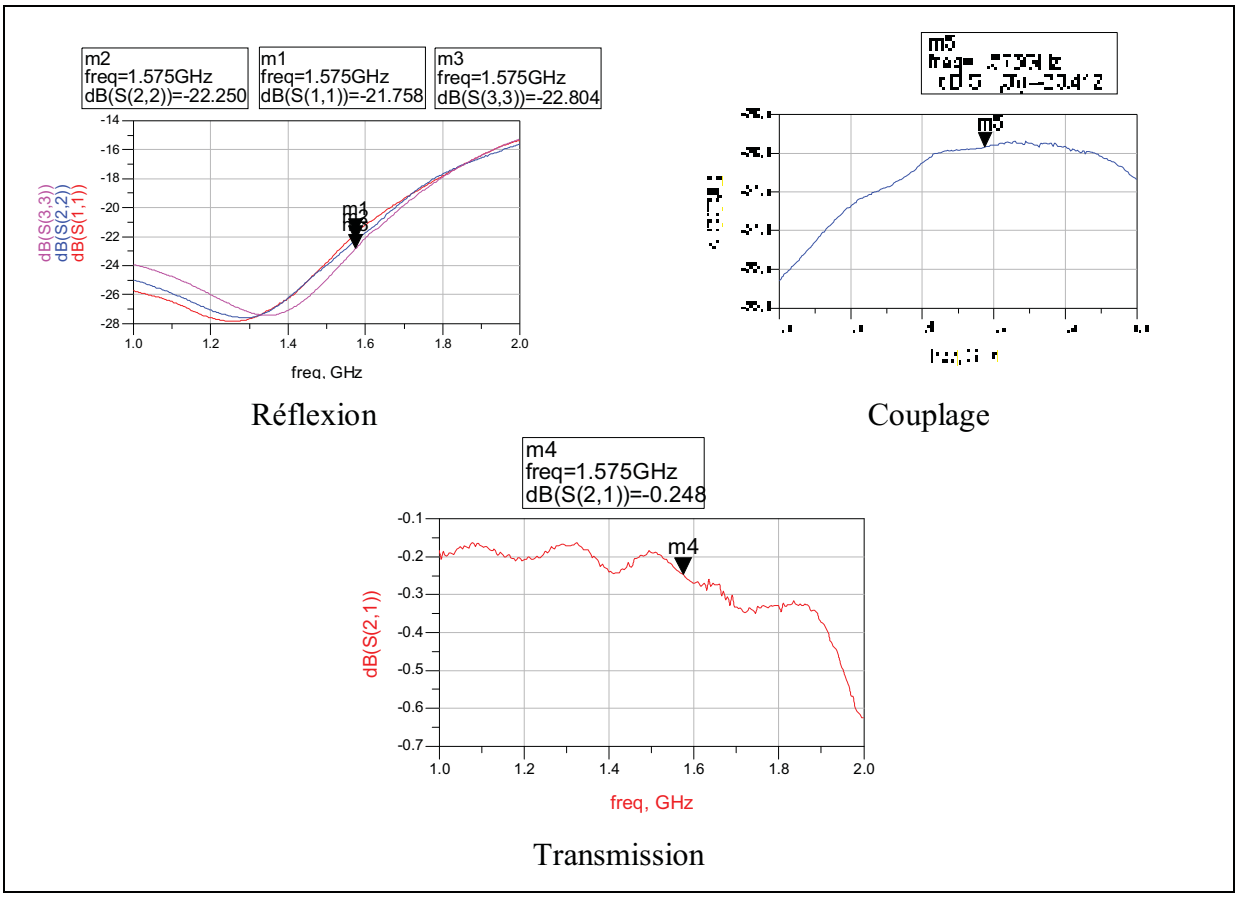

Figure 2.8 Paramètres « S » du coupleur XC1900A-20S.

2.3.2 Le détecteur de puissance

Avec la totalité de la chaine RF située dans un boitier, il est prévisible qu'une augmentation de la température dans l'enceinte pourra dégrader les résultats que le détecteur de puissance délivre. Nous avons alors choisi le composant AD8318 qui offre un détecteur de puissance avec une option de compensation de l'effet de la température.

Ce type de détecteur convertit le signal RF en une tension continue proportionnelle au niveau de puissance du signal. Il présente ainsi une courbe linéaire définie par la pente et le point d'interception avec l'axe des abscisses. Le datasheet du circuit donne la pente de la réponse du détecteur en fonction des fréquences (voir annexe I.2), mais ce sont des résultats expérimentaux qui dépendent fortement des conditions de tests. Dans notre cas, nous avons

préféré effectuer ces tests pour avoir une conformité par rapport aux conditions de travail et aussi parce que nous ne pouvions pas extraire des informations précises des datasheets. La Figure 2.9 illustre la réponse du détecteur de puissance mesurée au laboratoire. Les valeurs des pentes et des points d'intersections avec l'axe des abscisses de la zone linéaire du détecteur de puissance (-60 à -10 dBm) sont résumés dans le Tableau 2.3.

Tableau 2.3 Les pentes et les points d'intersectioins avec l'axe des abscisses de la réponse du détecteur de puissance pour les trois signaux

	Signal L1	**Signal L5**	**Signal E5b**
Pente	-0.0245	-0.0247	-0.0246
Point d'intersection	1.0088	0.8246	0.8178

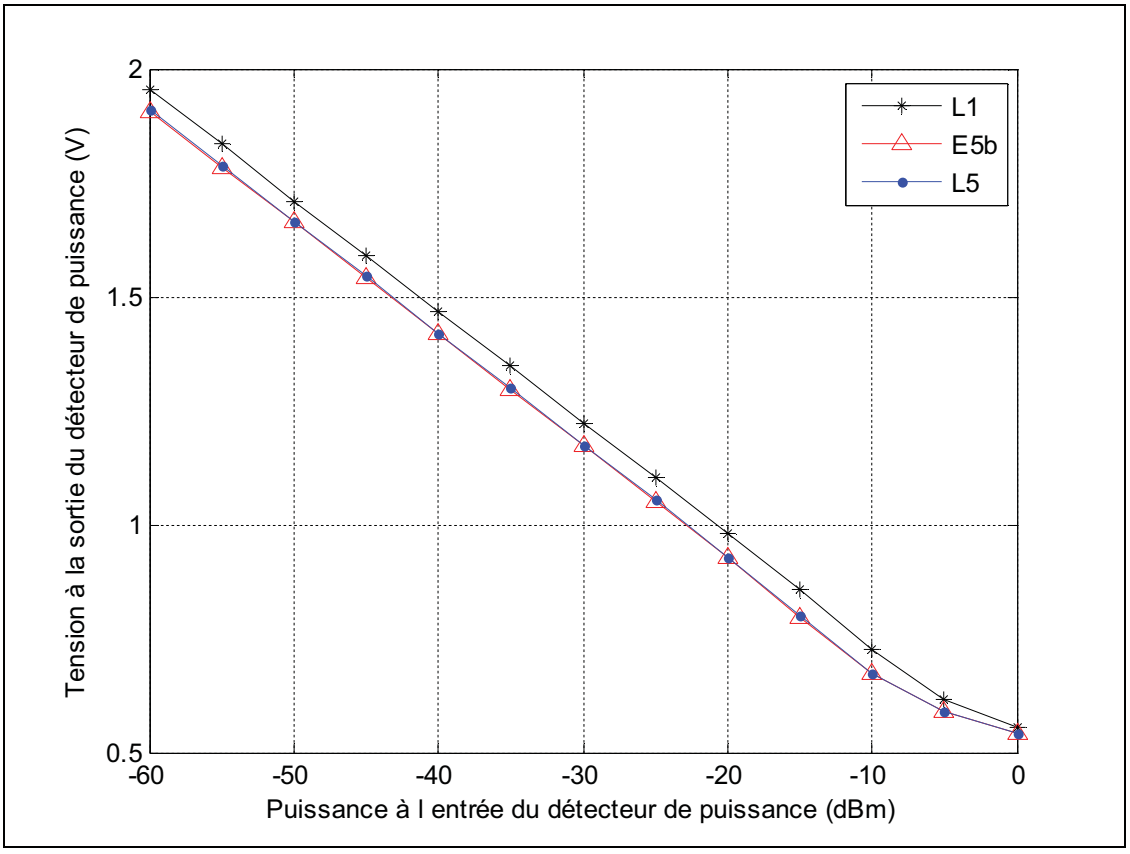

Figure 2.9 Réponse du détecteur de puissance AD8318.

2.3.3 Le convertisseur analogique numérique

Le convertisseur analogique numérique est utilisé pour convertir la tension de sortie du détecteur de puissance en une valeur numérique. Il a été testé et évalué à l'aide du CPLD et d'un programme qui contrôle le port parallèle d'un PC (détails dans la section 2.7.1). Le signal à l'entrée du CAN étant une tension continue, il n'y a donc aucune contrainte sur le choix du convertisseur. Il existe plusieurs types de CAN sur le marché (SAR, Sigma delta, flash…). Dans le système de contrôle de puissance, nous avons choisi un CAN de type SAR (voir annexe II pour le fonctionnement des CAN de type SAR) dont les caractéristiques sont les suivantes (Texas Instrument Inc, 2004) :

- Résolution : 18 bits;
- Référence interne : 4.096V;
- Débit symbole : jusqu'à 600 Ksymbole/s ; et
- Interface série fonctionnant jusqu'à 40 MHz.

Comme tout ADC, l'ADS8380 doit être contrôlé par des commandes externes lui ordonnant les étapes qu'il doit respecter. Ceci doit être conforme à un synchronisme (voir annexe I.3) que présente le constructeur. Une dégradation des performances peut apparaître si l'on ne respecte pas ce synchronisme.

2.3.4 Le convertisseur numérique analogique

Le convertisseur numérique analogique est le dernier composant du système de contrôle de puissance. Il permet de convertir les données entrantes en signaux analogiques qui commanderont les trois VGA de la chaine RF. L'AD5668 est un CNA de 16 bits qui comprend 8 sorties analogiques. Cette caractéristique est idéale pour notre application vu que nous avons besoin de trois tensions pour contrôler les VGA. Les autres sorties sont inutilisées, mais pourraient servir ultérieurement. L'AD5668 possède une interface série qui fonctionne à une horloge allant jusqu'à 50 MHz (Analog Devices Inc, 2005a). Elle reçoit les

données selon une procédure décrite dans le « datasheet » (voir annexe I.4). Ce composant a été testé et évalué à l'aide du CPLD qui lui envoie les commandes et en mesurant ses sorties à l'aide d'un multimètre d'une grande précision (détails dans la section 2.7.1).

2.3.5 Le CPLD

Le CPLD (CoolRunner XC2C256 de Xilinx) est le cœur du système de contrôle de puissance. Il permet d'exécuter les algorithmes et les procédures décrites dans la section 2.2. Il est à la base de la flexibilité du système car il permet de fermer la boucle de contrôle pour exécuter les modes « contrôle automatique de puissance » et « calibration de puissance » comme il peut la laisser ouverte pour fonctionner sur les autres modes. Il est interfacé avec le CAN pour récupérer le niveau de puissance des signaux, avec le CNA pour piloter les tensions de commande des VGA et avec le FPGA pour pouvoir communiquer avec la partie logicielle. L'interfaçage avec le FPGA n'est pas direct car les signaux passent par le deuxième CPLD de la carte et sont transférés vers le connecteur GPIO (*Genral Purpose Input Output*).

2.3.6 Le VGA

Le VGA (ADL5330) est un amplificateur variable contrôlé par tension. Il est large bande (jusqu'à 3000 MHz) et peut alors être utilisé pour les trois signaux. Il présente une réponse linéaire en fonction de la tension de commande, mais cette réponse n'est pas stable en fonction de la fréquence (Analog Devices Inc, 2005b). Pour cela, il nous a fallu caractériser ce composant pour les fréquences des signaux GPS et Galileo. Nous avons alors réalisé un montage, illustré par la Figure 2.10, composé d'un générateur de tension et d'un analyseur de réseau. Le tout étant commandé par ordinateur. Nous avons pris les mesures et nous avons par la suite transposé les résultats sur le logiciel ADS et nous avons obtenu les courbes illustrées dans la Figure 2.11.

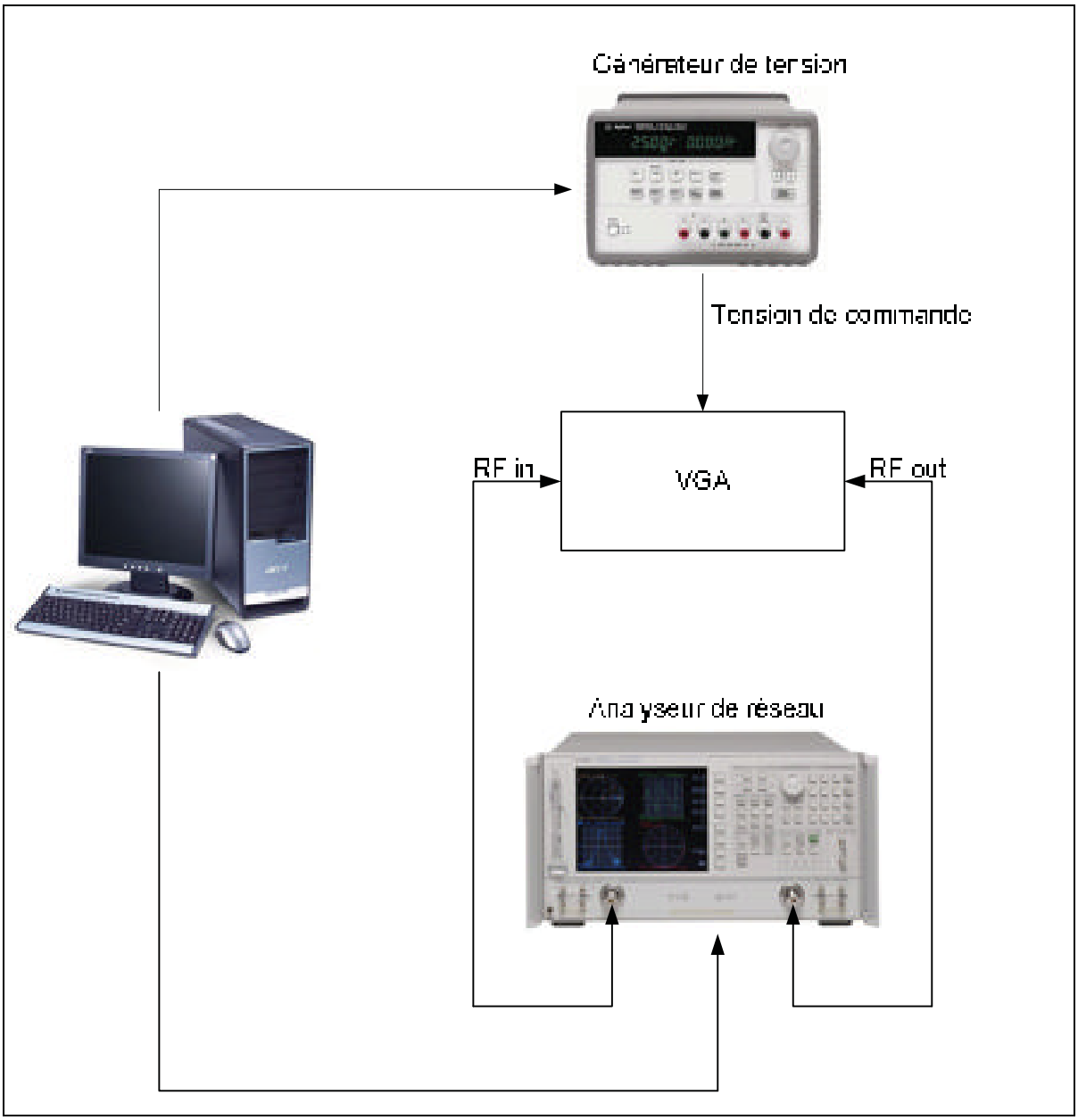

Figure 2.10 Montage de test du VGA.

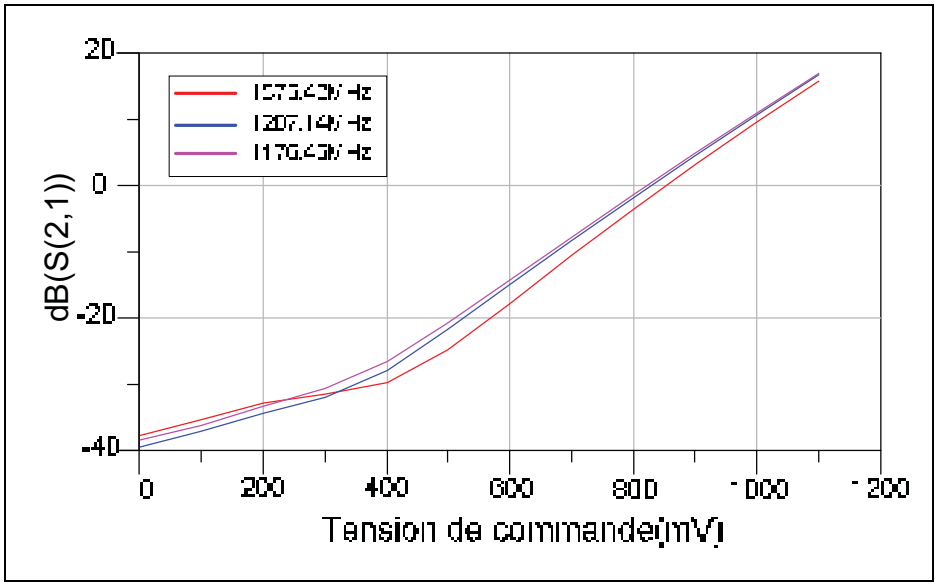

Figure 2.11 Réponse du VGA pour les fréquences GPS et Galileo.

Les courbes suivent une allure linéaire, dans la portion allant de -25 dB d'atténuation jusqu'à 17 dB de gain, et sont exprimées par l'équation (2.1) où Y_i représente le gain et X_i la tension de commande.

$$Y_i = a_i X_i + b_i \tag{2.1}$$

La réponse du VGA pour les différentes fréquences est alors :

- f=1575.42 MHz : $Y_{L1} = a_{L1} X_{L1} + b_{L1} \quad où \begin{Bmatrix} a_{L1} = 54.35 \\ b_{L1} = -53.069 \end{Bmatrix}$;
- f=1176.45 MHz : $Y_{L5} = a_{L5} X_{L5} + b_{L5} \quad où \begin{Bmatrix} a_{L5} = 51.348 \\ b_{L5} = -48.88 \end{Bmatrix}$;
- f=1207.14 MHz : $Y_{E5b} = a_{E5b} X_{E5b} + b_{E5b} \quad où \begin{Bmatrix} a_{E5b} = 51.79 \\ b_{E5b} = -49.5956 \end{Bmatrix}$.

Les valeurs a_i et b_i sont calculés à partir de la courbe de la Figure 2.11.

Les VGA peuvent fournir une atténuation maximale de l'ordre de 40 dB, mais la réponse entre -25 et -40 dB n'est pas parfaitement linéaire. On peut alors l'approximer et l'utiliser pour atténuer plus le signal (en informant l'utilisateur de la résolution dans cet intervalle).
La réponse du VGA nous permettra par après de déterminer la valeur de la tension de commande nécessaire pour obtenir un gain quelconque. Le mode de « contrôle de gain » utilisera ces fonctions pour ajuster la puissance des signaux.

2.4 Algorithmes de contrôle de puissance

Dans cette section, nous détaillerons les algorithmes développés pour le mode de « contrôle automatique de puissance ». Le premier consiste à effectuer des calculs, en fonction du niveau de puissance détecté, pour trouver la nouvelle tension de commande du VGA. Cet algorithme nécessite une puissance de calcul et le CPLD n'est pas adéquat pour ce genre de traitement. C'est pour cette raison que nous avons développé un deuxième algorithme. Ce dernier est nettement plus simple, il n'effectue aucun calcul et s'appuie sur une comparaison entre le niveau de référence et le niveau de puissance détectée.

2.4.1 Algorithme N° 1

Cet algorithme se base sur le calcul de la tension de commande adéquate pour le VGA en se basant sur le niveau du signal de référence et le niveau de puissance détecté. En effet, connaissant les fonctions de transfert du détecteur de puissance ainsi que du VGA, on peut déduire la tension nécessaire pour avoir un certain niveau de signal.

Supposons que le niveau de puissance désiré est égal à P_{ref}. Les détecteurs de puissance ont une fonction de transfert affine qui a pour expression l'équation (2.2) où P_i représente la puissance détectée et X_i^{DP} la tension issue du détecteur de puissance, a_i^{DP} représente la pente de la droite et b_i^{DP} représente l'ordonnée à l'origine.

$$P_i = a_i^{DP} X_i^{DP} + b_i^{DP} \tag{2.2}$$

Considérons le signal GPS L1 pour illustrer la démarche de calcul de la puissance de référence à envoyer au système. La puissance détectée est une fonction de la tension du détecteur de puissance X_{L1}^{DP} et est illustré par l'équation (2.3) :

$$P_{L1} = a_{L1}^{DP} X_{L1}^{DP} + b_{L1}^{DP} \tag{2.3}$$

Pour obtenir une calibration du signal GPS L1, on a :

$$P_{ref} + A_{C_1} = P_{L1} + Y_{L1} \tag{2.4}$$

Où A_{C_1} est l'atténuation provoquée par le coupleur (paramètre de transmission S_{21} du coupleur) et Y_{L1} représente le gain du VGA (voir équation (2.1)) L'inconnu ici est Y_{L1} mais le but du calcul est de parvenir à X_{L1}. En combinant les équations (2.3) et (2.4) nous obtenons l'expression de X_{L1} :

$$X_{L1} = \frac{P_{ref} + A_{C_1}}{a_{L1}} - \frac{\left(b_{L1}^{DP} + b_{L1}\right)}{a_{L1}} - \frac{a_{L1}^{DP}}{a_{L1}} X_{L1}^{DP} - \frac{\left(b_{L1}^{DP} + b_{L1}\right)}{a_{L1}} \tag{2.5}$$

X_{L1}^{DP} est une donnée qu'on reçoit directement de l'ADC. Il faudra donc conserver sa valeur décimale afin de minimiser les calculs dans le CPLD. On obtient donc la valeur de la tension réelle de la tension qui est exprimée par l'équation (2.6) où $X_{L1}^{DP_{num}}$ représente la valeur décimale de X_{L1}^{DP} représentée sur 18 bits et V_{ref}^{ADC} représente la tension de référence du CAN égale à 4.096 V

$$X_{L1} = \frac{P_{ref} + A_{C_1}}{a_{L1}} - \frac{\left(b_{L1}^{DP} + b_{L1}\right)}{a_{L1}} - \frac{a_{L1}^{DP}}{a_{L1}} \frac{V_{ref}^{ADC}}{2^{18}} X_{L1}^{DP_{num}} \tag{2.6}$$

De cette façon, on retrouve l'expression finale de la valeur décimale de la tension de commande du VGA X_{L1}^{num}, codée sur 16 bits (équation (2.7)) et en nommant les constantes de cette équation C_{L1}^{1} et C_{L1}^{2}, on obtient l'expression simplifiée de l'équation (2.8) où $\lfloor x \rfloor$ représente la partie entière de x et V_{ref}^{DAC} est la tension de référence du CNA égale à 1.3 V.

$$X_{L1}^{num} = \left\lfloor \underbrace{\frac{2^{16}}{V_{ref}^{DAC}}\left[\frac{P_{ref}+A_{C_1}}{a_{L1}} - \frac{\left(b_{L1}^{DP}+b_{L1}\right)}{a_{L1}}\right]}_{C_{L1}^{1}} + \underbrace{\left(-\frac{a_{L1}^{DP}\cdot V_{ref}^{ADC}}{4a_{L1}\cdot V_{ref}^{DAC}}\right)}_{C_{L1}^{2}} X_{L1}^{DP_{num}} \right\rfloor \tag{2.7}$$

$$X_{L1}^{num} = \left\lfloor C_{L1}^{1} + C_{L1}^{2} \times X_{L1}^{DP_{num}} \right\rfloor \tag{2.8}$$

Le CPLD effectue juste une opération de multiplication et une opération de somme. La nouvelle valeur de la tension de commande est prête à être envoyée vers le DAC. L'opération est la même pour les deux autres signaux; il suffit de commander le multiplexeur RF.

En calculant la nouvelle tension de commande directement, cet algorithme permet un ajustement immédiat de la puissance. Mais, le CPLD est un composant qui n'est pas adapté à effectuer des calculs. Le contrôle de l'ADC, du DAC, du multiplexeur, des différents modes de fonctionnement ainsi que la communication entre la partie IF et la partie RF prend déjà beaucoup d'espace dans le CPLD. L'algorithme a été écrit en VHDL et le compilateur n'a pas réussi de l'intégrer. Pour cette raison, nous avons développé un autre algorithme non gourmant en termes de calculs. Il est décrit dans la section suivante.

2.4.2 Algorithme Nº 2

Ce deuxième algorithme est nettement plus simple que le premier. Il consiste à comparer la puissance détectée avec la puissance de référence. Selon le résultat, le programme va hausser ou abaisser le niveau de gain du VGA jusqu'à obtenir une puissance égale à la puissance de

référence. La Figure 2.12 illustre le fonctionnement de l'algorithme. Après la détection de la puissance, le programme exécute la comparaison. Si la puissance détectée est supérieure à la puissance de référence, la tension de commande du VGA baisse pour diminuer le gain du VGA. Dans le cas contraire, la tension hausse et augmente le gain afin de stabiliser le système. Si les deux valeurs sont égales, la puissance du signal est ajustée et l'opération est terminée.

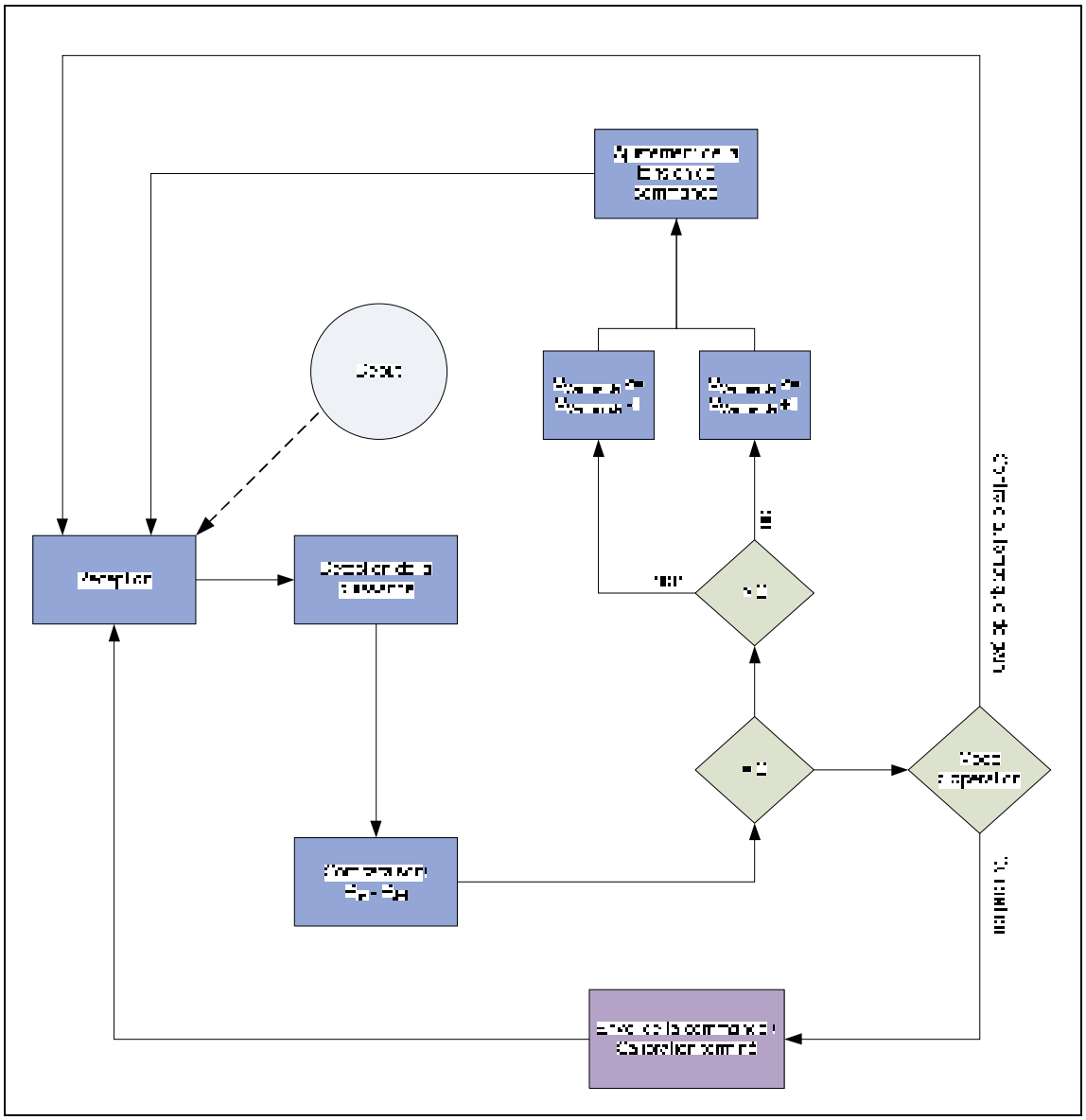

Figure 2.12 Mode de fonctionnement du deuxième algorithme.

La simplicité de cet algorithme permet son intégration avec le reste des commandes. Par contre, le temps de réponse du système est lent car, pour atteindre le niveau de référence quand la puissance est assez loin de la référence, le programme nécessite plusieurs itérations. Le système de contrôle de puissance n'est pas conçu pour fonctionner en réception, il a pour principale tâche de réguler le niveau des signaux à la sortie de la chaine RF. Par conséquent, le temps de réponse n'est pas critique. En effet, en réception, un système de contrôle de gain doit avoir un temps de réponse rapide pour garder un niveau constant du signal, mais pas très rapide pour ne pas altérer le signal (*gain pumping*). En comparant les deux algorithmes et en tenant compte des contraintes matérielles, nous choisissons le deuxième algorithme.

2.5 Communication entre la partie logicielle et le système de contrôle de puissance

Le système de contrôle de puissance peut opérer selon plusieurs modes décrits dans la section 2.2. Il faut alors lui transmettre toutes les données nécessaires pour assurer son fonctionnement. Nous avons alors créé un protocole de communication entre la partie numérique du système de contrôle de puissance (CPLD) et la partie logicielle. Étant en communication permanente avec la partie IF du simulateur, la partie logicielle aura la tâche de construire le message de commande et l'envoyer au FPGA. Ce dernier s'occupera de transmettre les données au CPLD.

2.5.1 Protocole de communication du système de contrôle de puissance

La communication entre la partie RF et le reste du simulateur doit être bidirectionnelle et le design a été effectué pour permettre une telle communication. Avant l'introduction du système de contrôle de puissance, cette communication était absente. Nous avons donc développé un protocole de communication pour permettre une gérance totale du système et pour offrir une flexibilité dans le cas d'ajout de nouvelles fonctionnalités ou options.

Trame de communication du FPGA vers la partie RF

La trame de communication est composée de 32 bits et possède la structure illustrée dans la Figure 2.13. Les trois premiers bits (31 à 29) sont dédiés pour le choix du mode de fonctionnement. Les trois bits qui les succèdent sont consacrés aux choix du signal. Les bits 24 à 18 sont réservés et les derniers bits (17 à 0) transportent les données nécessaires pour chacun des modes,

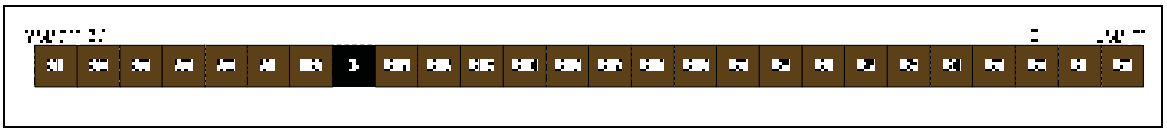

Figure 2.13 Trame de communication générique entre le FPGA et la partie RF.

Le CPLD reçoit les 32 bits envoyés par le FPGA et les analyse pour déterminer le mode de fonctionnement. Les modes sont au nombre de 4. Il suffirait donc, 2 bits pour les représenter mais 3 bits ont été alloués pour permettre une éventuelle extension et ils sont symbolisés par les 3 bits de la trame de communication (voir Figure 2.13) Les combinaisons des bits de commande sont résumées dans le Tableau 2.4. Le mode « contrôle de puissance dans le temps » s'appuie sur le mode « contrôle de gain ». En effet, puisque les durées des événements du scénario créé par l'utilisateur sont contrôlées par la partie logicielle, nous allons juste envoyer la tension de commande du VGA ce qui revient à une trame associée au mode « contrôle de gain ». Par conséquent, il n'aura pas une séquence de bit associée.

Tableau 2.4 Table de vérité des bits de commande

C3	C2	C1	Mode de fonctionnement
0	0	0	Contrôle automatique de puissance
0	0	1	Contrôle de gain
0	1	0	Calibration de puissance
Autres			Applications ultérieures

Pour chacun des modes de fonctionnement, il faut spécifier le signal sur lequel nous allons agir. Le Tableau 2.5 contient les différentes combinaisons nécessaires. Le simulateur contient actuellement trois signaux distincts et 2 bits dans la trame peuvent nous satisfaire. Le simulateur est en phase d'évolution et pourrait intégrer plus de signaux au fur et à mesure que les systèmes GPS et Galileo évoluent et par conséquent nous avons réservé un bit supplémentaire afin d'adapter le système à d'éventuelles évolutions.

Tableau 2.5 Table de vérité des bits de choix des signaux

S3	**S2**	**S1**	**Signal**
0	0	0	GPS L1 / Galileo E1
0	1	0	GPS L5 / Galileo E5a
0	1	1	Galileo E5b
1	1	1	Tous les signaux
Autre combinaison			Pour des signaux ultérieurs (L2C par exemple)

La Figure 2.14 illustre la trame de communication qui se transmet du FPGA vers le CPLD pour les différents modes de fonctionnement du système de contrôle de puissance. Pour chacun de ces modes, l'interprétation des bits qui suivent les trois bits de commande et les trois bits de choix du signal change. Pour le mode (a) et (c) de la Figure 2.14, il suffit d'envoyer la puissance de référence sur laquelle l'algorithme de contrôle se basera pour la boucle automatique. Le mode (b) est aussi très simple, on envoie juste la valeur de la tension de commande du VGA correspondant pour fixer le gain du signal choisi.

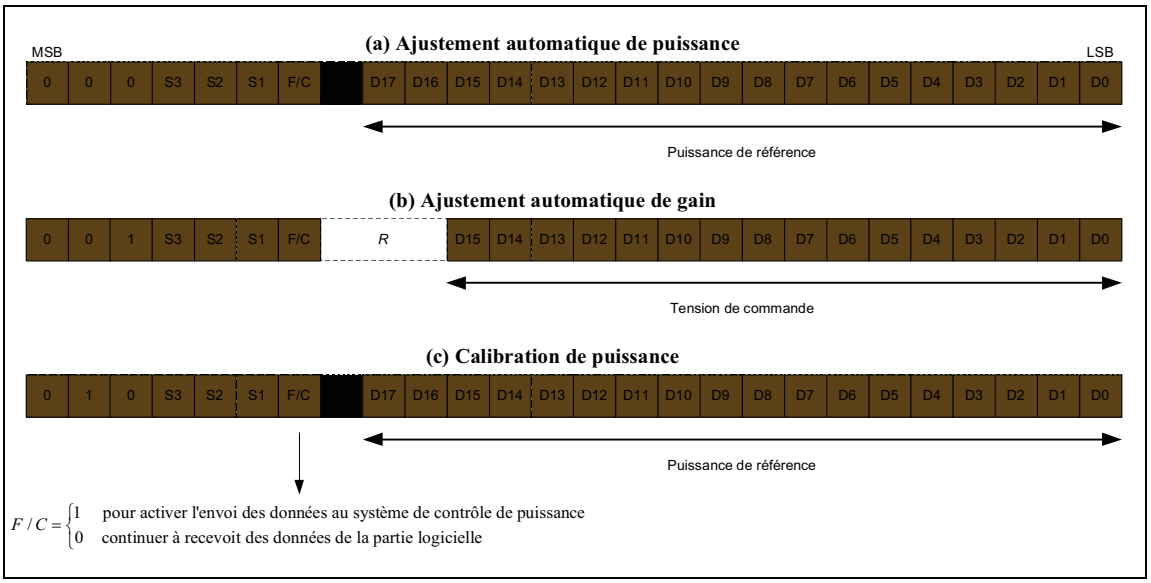

Figure 2.14 Trame de communication du FPGA vers le CPLD.

Trame de communication de la partie RF vers le FPGA

La communication entre la partie RF du simulateur et le FPGA de la partie IF est nécessaire si on veut transmettre des données vers l'utilisateur. De cette façon, on peut renvoyer des informations concernant la carte RF à savoir la température de l'enceinte, l'état des synthétiseurs de fréquence, les valeurs de tensions des VGA, etc. À la fin de l'exécution du mode « calibration de puissance », le système envoie une commande au FPGA lui indiquant le succès ou l'échec de l'opération et la valeur de la tension de commande du VGA. L'échec de l'opération est annoncé quand le système atteint un certain nombre d'itérations (compteur 16 bits) sans stabiliser la puissance (voir Figure 2.15). Pour garder le même format de données que la communication inverse (du FPGA vers la partie RF), nous allons fixer la taille de la trame à 32 bits. Les six premiers(MSB) bits auront la même signification que ceux de la trame allant du FPGA à la partie RF pour que la partie logicielle puisse calibrer la bonne fréquence.

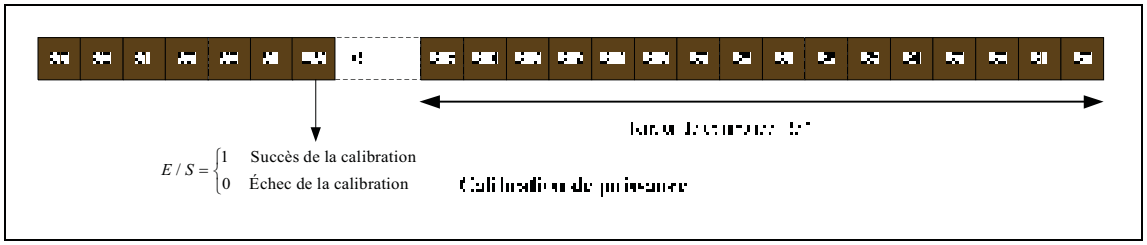

Figure 2.15 Trame de communication du FPGA vers le CPLD.

2.5.2 Communication entre la partie logicielle et la partie IF

La partie logicielle communique de deux manières avec la partie IF du simulateur. Une consiste à envoyer les données dans une mémoire externe au FPGA et ce dernier s'occupe de les récolter. Ce type de communication est dédié pour le contrôle des canaux dans le FPGA. La deuxième manière consiste en l'utilisation d'un bus rapide synchronisé (*fast synchronous bus, FSB*). Ce bus est utilisé pour transférer les données de contrôle de puissance à la partie IF.

Le bus FSB est contrôlé par une horloge de 33 MHz. Il occupe 2 Kbit de mémoire qui s'étendent de l'adresse 0x1800 à 0x1FFF (Sauriol B, 2007). Cette mémoire n'est pas suffisante pour le fonctionnement du simulateur. Un registre placé à l'adresse 0x011C, sous le nom de « *FSB Page* » est utilisé pour étendre cette mémoire à 512 Kbits subdivisée en 256 pages de 2 Kbit. Le bus FSB est composé d'un bus d'adresse de 11 bits (19 avec l'utilisation du registre « *FSB Page* ») et deux bus de données de 32 bits pour écriture et lecture.

Pour le fonctionnement du simulateur, 128 pages sont utilisé pour les éventuel 128 canaux que doit générer la partie IF (actuellement, la partie IF est capable de supporter 8 canaux). Les autres 128 pages sont libres et peuvent être utilisées pour d'autres types de contrôle. Nous allons alors utiliser ce bus pour transmettre les données du contrôle de puissance à la partie IF du simulateur.

Pour envoyer les données à la partie IF via le bus FSB, le logiciel doit les écrire dans des registres et par la suite, ces données sont récupérées au niveau du FPGA. Le problème est

que le logiciel ne sait pas si le FPGA a récupéré les données ou pas et le FPGA ne sait pas si l'information que contient le registre est nouvelle ou pas. Pour remédier à cette situation, nous allons utiliser un registre accessible en lecture et écriture par le FPGA et le logiciel (voir annexe V pour les adresses des registres utilisés). Ce registre est mis à '1' quand le logiciel veut envoyer une nouvelle donnée et il est mis à '0' quand le FPGA récupère cette donnée.

Le logiciel envoie des données au début de la simulation pour initialiser le système de contrôle de puissance et pendant la simulation si le mode de « contrôle de puissance dans le temps » est activé. Le logiciel peut alors envoyer une ou plusieurs trames. Le FPGA collecte ces trames et les classe par mode d'exécution. En ce qui concerne le mode « calibration », une seule trame de commande est envoyée à la fois par le logiciel. Elle est envoyée immédiatement au système de contrôle de puissance. Le FPGA attend ensuite la trame de retour et la renvoie à la partie logicielle. Dans le cas des modes de contrôle (« contrôle de gain » et « contrôle automatique de puissance »), le FPGA commence à envoyer les trames reliées au mode de « contrôle de gain » car il ne nécessite pas un contrôle extérieur. Il passe ensuite au mode de « contrôle automatique de puissance » qui nécessite l'intervention du FPGA (voir section 2.2.3). Ce dernier reste cependant en écoute de la partie logicielle dans le cas où il y aurait de nouvelles trames de commandes (activation du mode « contrôle de puissance dans le temps »).

2.5.3 Communication entre la partie IF et la partie RF

La carte RF possède deux CPLD. Un pour le système de contrôle de puissance et le deuxième pour tous les autres types de contrôles. Ce CPLD est relié à la carte IF par le biais d'un connecteur à 34 pattes (GPIO) dont juste 16 sont connectés. Les deux CPLD de la carte RF (voir Figure 2.16) sont reliés entre eux avec 8 signaux. Il faut alors implémenter de part et d'autre une interface qui permet la communication bidirectionnelle entre la partie IF et la partie RF. Nous avons implémenté au début une communication série (*full duplex*) avec l'utilisation de trois signaux pour l'émission et trois signaux pour la réception (interface SPI sans le signal *Chip Select*). Les ressources du CPLD étant très limitées (Le nombre de

macrocell ne suffit pas pour l'intégration du design), la totalité du design ne pouvait pas s'y intégrer. Nous avons alors opté pour une communication parallèle en *half duplex* avec l'utilisation d'un bus de 5 bits dont 4 pour les données et un bit pour l'acquisition. Le bus bidirectionnel est donc à trois états (*tri-state bus*) et est implémenté au niveau du FPGA et du CPLD. Le deuxième CPLD de la carte RF intègre aussi cette fonctionnalité pour pouvoir passer les données du port GPIO au système de contrôle de puissance et inversement. Le bus doit avoir un contrôle de lecture et écriture. Un signal est dédié à cette opération qui est effectuée au niveau du CPLD du système de contrôle de puissance. La Figure 2.16 illustre les signaux de communications entre le FPGA et la carte RF.

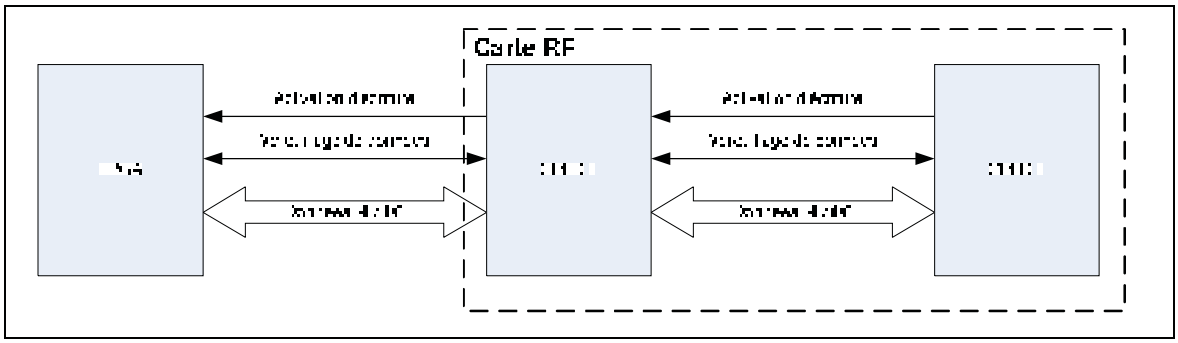

Figure 2.16 Signaux de communication entre la partie IF et la partie RF.

2.6 Interface d'utilisation du système de contrôle de puissance

L'interface d'utilisation du système de contrôle de puissance a été réalisée par un membre de l'équipe de recherche chargé de la partie logicielle du simulateur. Elle comporte deux volets. Le premier est relié aux modes fondamentaux de contrôle de puissance à savoir « contrôle de gain » et « contrôle automatique de puissance ». La Figure 2.17 illustre cette interface. L'utilisateur doit spécifier la puissance du signal dans la plage (-110 à -150 dBm). La case « *Automatic level Control* » sert à activer et désactiver le mode de « contrôle automatique de puissance. Le champ « *Noise Density* » est destiné au contrôle du niveau de bruit traité au chapitre 3.

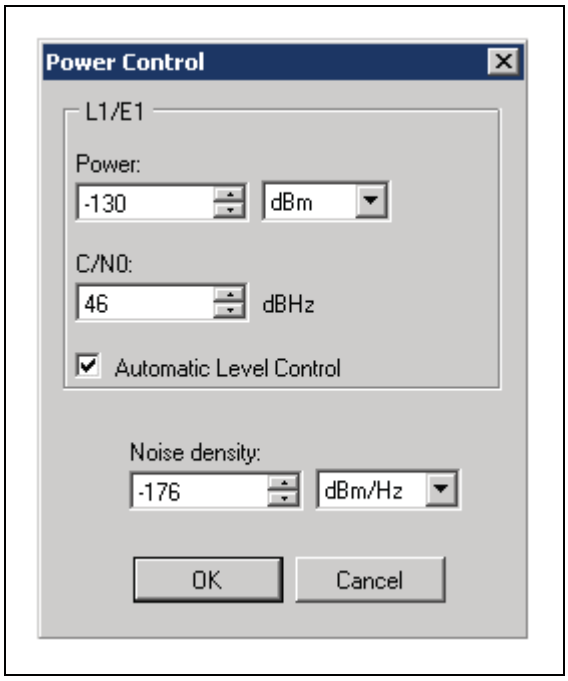

Figure 2.17 Interface de contrôle de puissance.

Le deuxième volet contient deux interfaces et est en relation avec le mode « contrôle de puissance dans le temps ». Comme il a été expliqué ultérieurement, l'utilisateur doit créer un modèle de variation de puissance. Cette fonctionnalité est illustrée par la Figure 2.18. Les modèles créés sont assignés aux signaux en choisissant le modèle désiré pour la fréquence spécifiée (voir Figure 2.19).

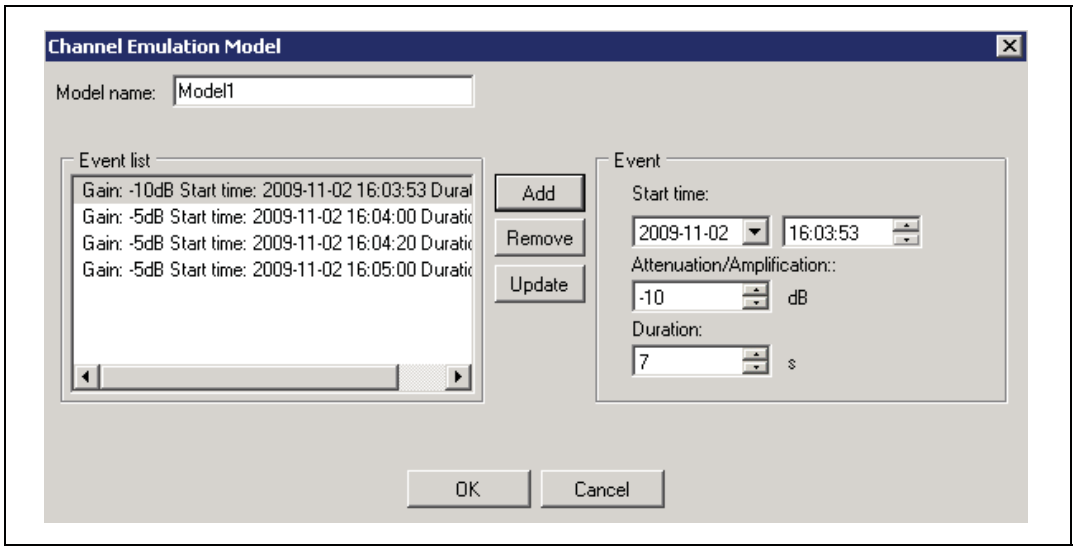

Figure 2.18 Interface de création d'un mode de contrôle de puissance dans le temps.

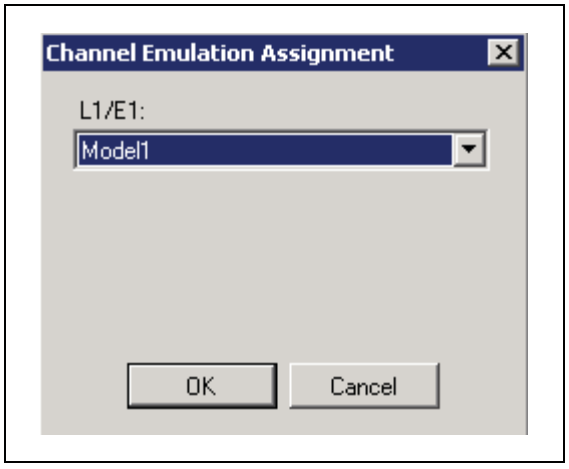

Figure 2.19 Interface d'assignation d'un modèle à un signal.

2.7 Validation du système de contrôle de puissance

2.7.1 Validation des composants du système de contrôle de puissance

Avant d'implémenter le système sur la carte finale, deux cartes ont été développées pour permettre le test et la validation des composants (voir annexe VI pour les schémas électriques des cartes).

Le CNA a pour rôle de piloter les VGA (voir l'architecture du système à la Figure 2.1), en produisant la tension adéquate à chaque sortie. L'AD5668 peut fonctionner avec une tension de référence interne ou externe. Dans le cas du système de contrôle de puissance, nous avons utilisé une référence externe pour limiter la tension de sortie du CNA afin de protéger les VGA et aussi pour gagner en résolution (la diminution de la tension de référence implique la diminution du pas). La référence a été réalisée avec un diviseur de tension à partir du régulateur 3.3 V. Le diviseur donne une tension de référence au CNA de 1.355 V. Le CNA a été testé indépendamment des autres composants du système de contrôle de puissance pour vérifier les commandes envoyés et vérifier le fonctionnement de la référence externe. La commande du CNA est envoyée à partir du CPLD selon une procédure décrite par le constructeur (voir annexe I.4). La sortie est alors vérifiée par un multimètre d'une grande précision.

Le CAN « ADS8380 » requiert des signaux de contrôle qui permettent de le guider pour l'acquisition des données. Comme cité dans le chapitre précédant, l'acquisition des données est faite par CPLD en utilisant une horloge de 16 MHz. Le programme de commande intègre une machine à état (voir annexe I.3 pour les spécifications du constructeur et annexe VII pour le programme de contrôle du CAN). Il permet aussi l'acquisition des données issues du CAN et les renvoyer vers le PC via le port parallèle. Ce port possède 8 broches pour les données, par conséquent, les 18 bits sortant du CAN sont partagées sur trois parties chacune de 6 bits. Les deux autres bits sont remplis avec le numéro de la trame pour l'identification au niveau du PC. La Figure 2.20 illustre la procédure d'envoi de données du CPLD vers le PC.

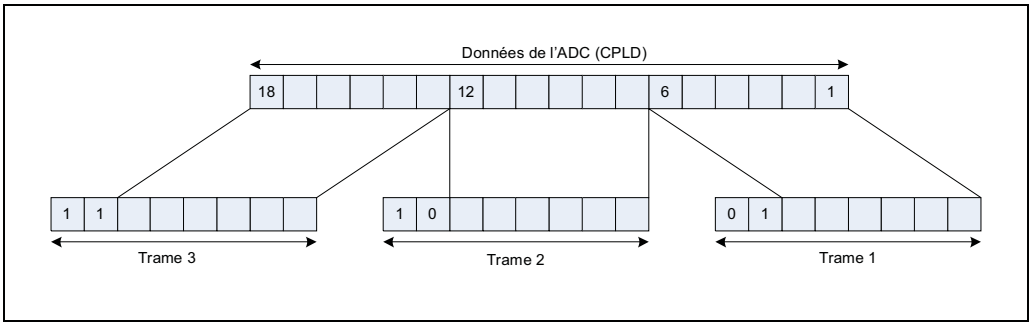

Figure 2.20 Transfert des données du CPLD au PC.

La connexion avec le PC sert à vérifier le fonctionnement du CAN en comparant la valeur analogique à son entrée et la valeur numérique de sortie. Nous avons développé un programme en langage C qui permet l'acquisition des données du port parallèle du PC et calculer la valeur analogique correspondante. Ensuite le programme enregistre les données sur un fichier qui est transféré sous MATLAB pour le traçage des courbes et le calcul de l'expression des fonctions de transfert du détecteur de puissance illustrées par la Figure 2.9 de la section 2.3.2.

De cette manière, nous avons validé toutes les autres composants du système de contrôle de puissance à savoir le multiplexeur, le CAN et le détecteur de puissance.

2.7.2 Validation de la communication entre la partie logicielle et la partie RF

La validation de la communication entre la partie logicielle et le système de contrôle de puissance s'est déroulée sur deux étapes. Nous avons commencé par valider la communication entre le logiciel et le FPGA ainsi que les signaux à la sortie de ce dernier. Nous avons alors effectué une comparaison entre la forme des signaux observés avec un analyseur logique de la compagnie « Agilent Technologies » et la simulation effectuée sous le logiciel « ModelSim ». La deuxième étape consistait en la validation de la réception des données du coté du système de contrôle de puissance. Nous avons alors dirigé le mode reçu et le signal sélectionné sur un connecteur de la carte et nos observations ont permit de valider l'exactitude des informations reçues. Pour finaliser la validation de la communication, nous

avons observé la tension à la sortie du convertisseur numérique analogique et nous l'avons comparé à la valeur envoyée.

2.7.3 Validation du fonctionnement du système de contrôle de puissance

Le système de contrôle de puissance doit atteindre une résolution de l'ordre de 0.1 dB sur une plage dynamique de 40 dB. Nous avons alors effectués des tests pour observer le comportement du système avec le mode de « contrôle de gain » et « contrôle automatique de puissance ». Nous allons traiter uniquement le cas du signal L1/E1 car la même procédure a été appliquée pour les autres signaux.

Test et validation du mode « contrôle de gain »

Pour vérifier la résolution du système de contrôle de puissance fonctionnant en mode « contrôle de gain », nous avons utilisé la partie logicielle du simulateur afin de varier la tension de commande du VGA et mesurer le niveau de puissance à la sortie de la carte RF avec un analyseur de signaux (voir Figure 2.21). Nous avons alors obtenu la courbe de la Figure 2.22. Nous pouvons observer que la courbe n'est pas parfaitement linéaire et par conséquent elle peut engendrer une grande erreur de précision qui dépasse les 1.5 dB (voir Figure 2.23).

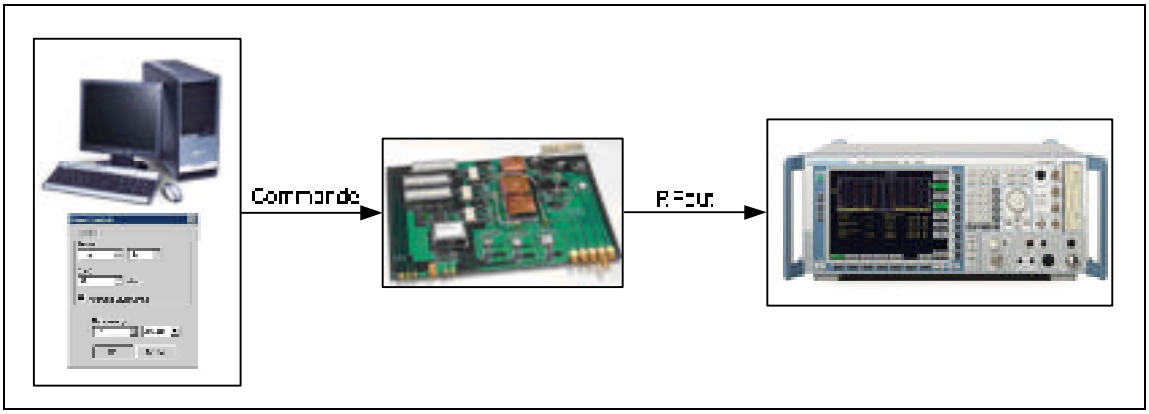

Figure 2.21 Montage de validation du système de contrôle de puissance.

Pour remédier à ce problème, nous avons effectué une approximation polynomiale d'ordre 10 de la courbe de la Figure 2.22 à l'aide du logiciel MATLAB qui consiste en la recherche d'un polynôme de degré *n* qui s'approche de la courbe associée. Cette solution permet de donner une erreur maximale de 0.089 dB (voir Figure 2.24).

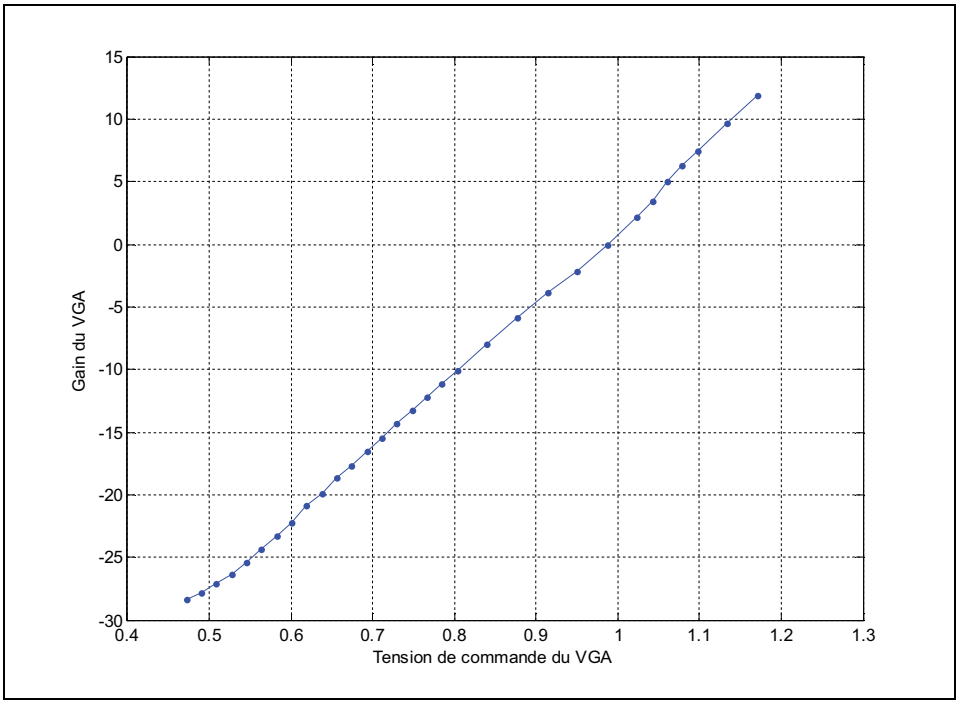

Figure 2.22 Réponse du VGA pour le signal L1/E1.

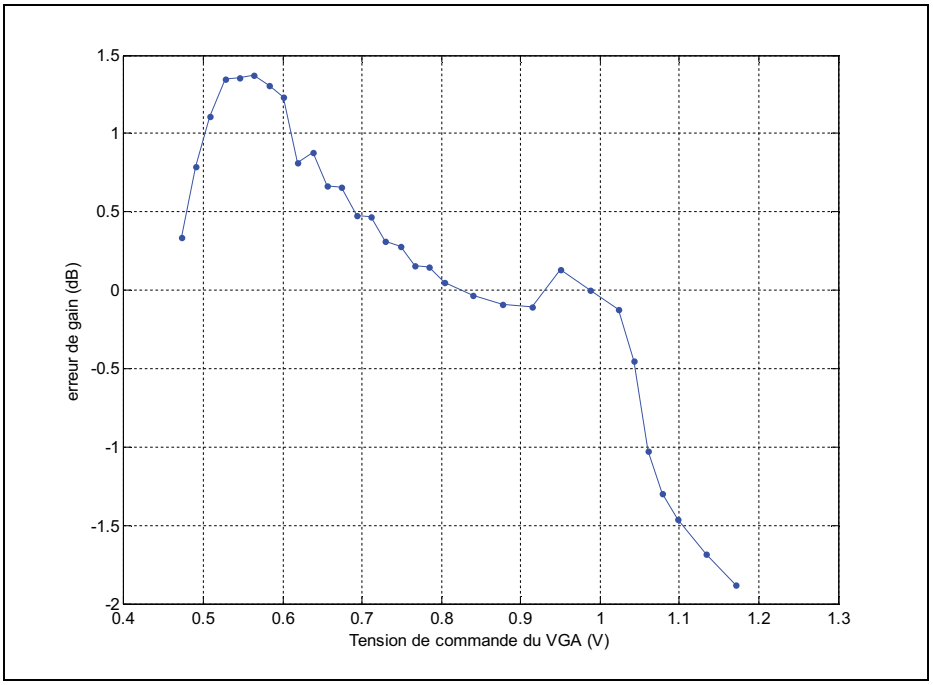

Figure 2.23 Erreur de gain du VGA avant correction.

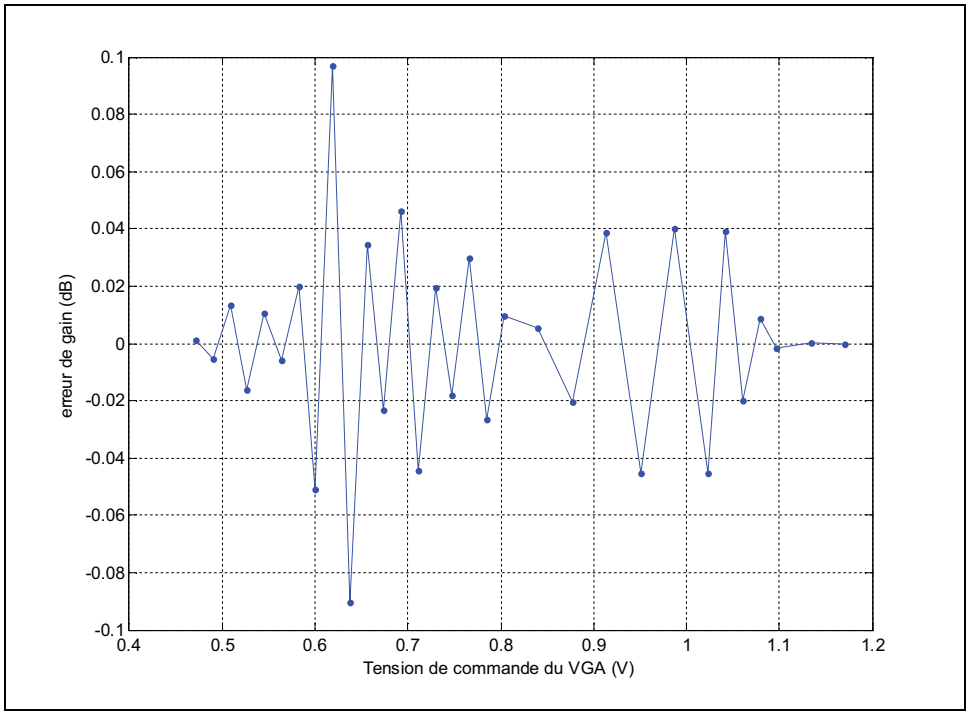

Figure 2.24 Erreur de gain du VGA après correction.

Test et validation du mode « contrôle automatique de puissance »

La même approche a été suivie pour la validation du système de contrôle de puissance en mode « contrôle automatique de puissance ». Nous avons mesuré le niveau de puissance à la sortie de la carte RF, à l'aide d'un analyseur de signaux, pour différentes valeurs de tensions du détecteur de puissance programmée dans la partie logicielle du simulateur. Nous avons alors obtenu la courbe de la Figure 2.25. La courbe parait linéaire sauf sur la partie des faibles puissances. Pour quantifier la précision du mode « contrôle automatique de puissance » nous avons tracé la courbe de l'erreur de puissance entre les valeurs mesurées et les valeurs théoriques (voir Figure 2.26).

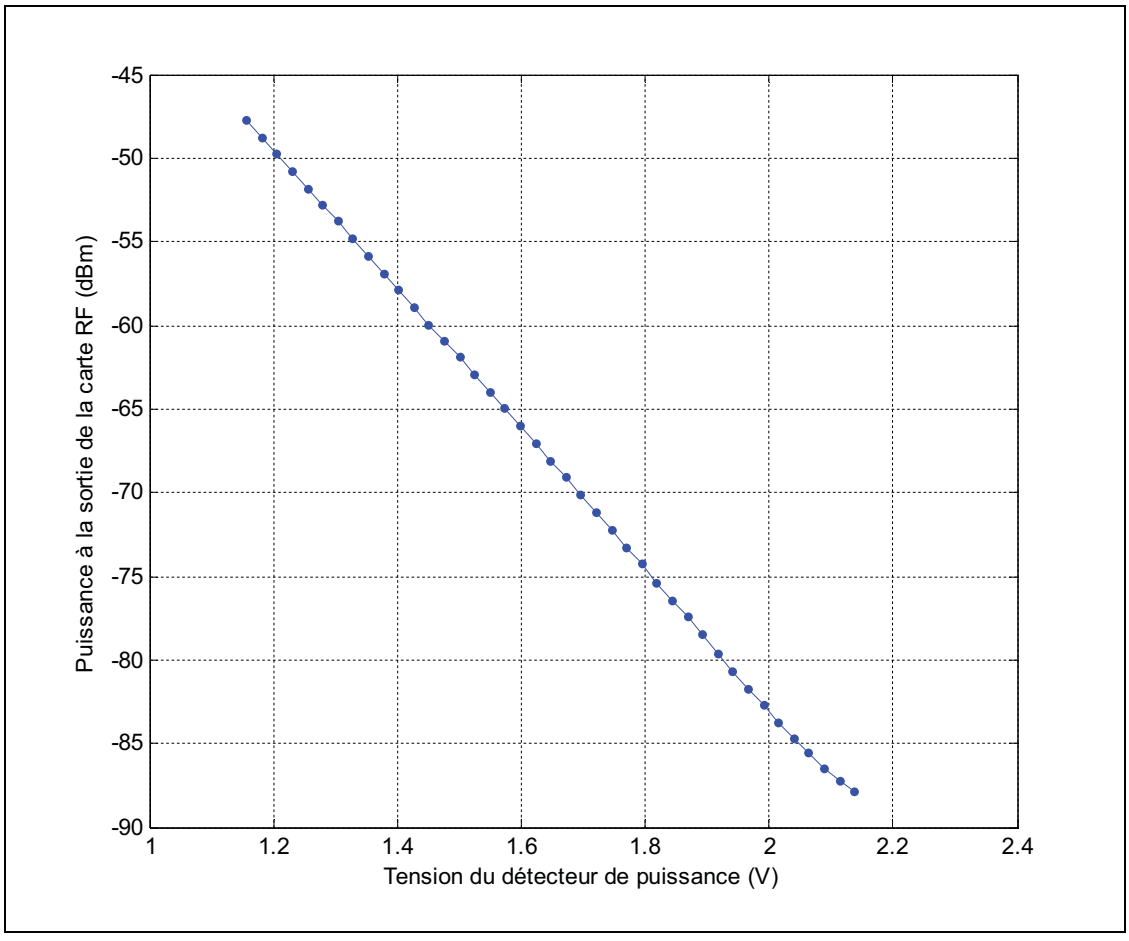

Figure 2.25 Réponse du système de contrôle de puissance en mode « contrôle automatique de puissance ».

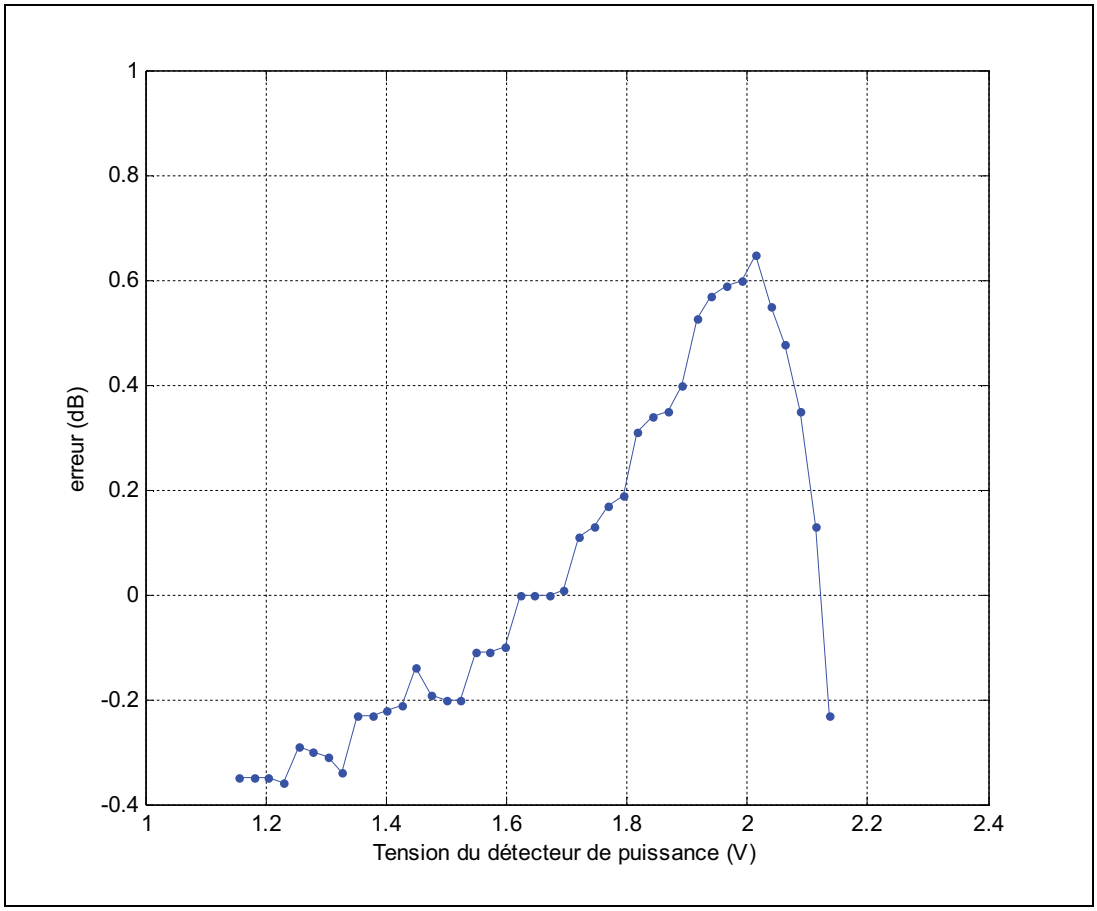

Figure 2.26 Erreur entre la puissance mesurée et la puissance théorique.

Nous remarquons que l'erreur à une valeur maximale de 0.65 dB. Nous avons alors effectué une approximation polynomiale d'ordre 6 qui nous a permit d'avoir une erreur maximale de l'ordre de 0.086 dB (voir Figure 2.27).

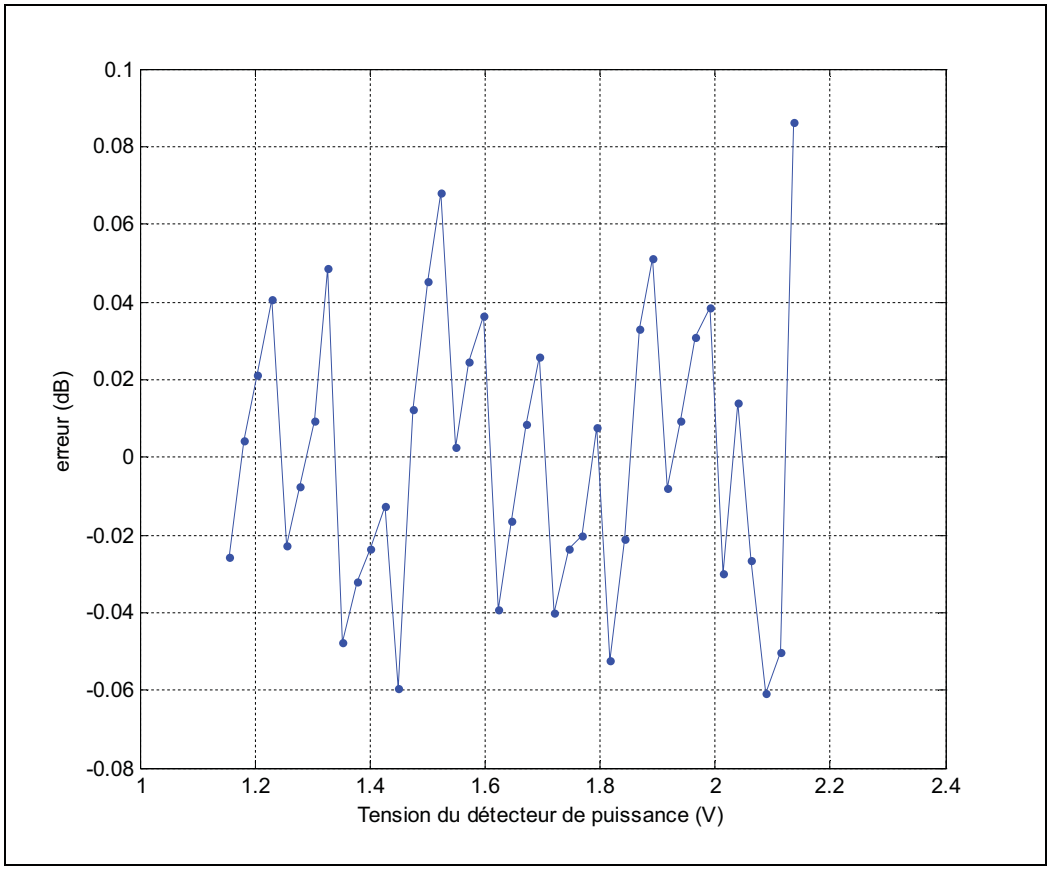

Figure 2.27 Erreur de puissance entre les valeurs corrigées et théoriques.

Test et validation du système de contrôle de puissance avec un récepteur commercial

Pour valider le système de contrôle de puissance dans le cadre du projet global du simulateur de constellations, nous avons effectué des tests avec un récepteur GPS commercial. Nous avons varié la valeur de la puissance au niveau du récepteur et nous avons observé la valeur du ratio C/N_0 qu'affiche le récepteur. Pour le calcul du ratio C/N_0, les récepteurs s'appuient sur des algorithmes. Ils ne peuvent pas mesurer cette valeur sans faire de modification sur le signal. En effet, la puissance d'un signal GPS est très faible et par conséquent il faut l'amplifier et le traiter pour pouvoir le détecter. Ceci peut alors jouer un rôle important dans la précision du calcul du ratio C/N_0.

La Figure 2.28 illustre la réponse du récepteur commercial face aux changements de la puissance. Nous remarquons que le récepteur ne suit pas les variations effectuées au niveau du logiciel. La Figure 2.29 représente l'erreur mesurée de la valeur du C/N_0 affichée au récepteur par rapport à une pente idéale.

Dans les puissances hautes (au dessus de 50 dBHz), le ratio C/N_0 tend à saturer. Ceci peut s'expliquer par la saturation de l'étage RF du récepteur. Dans les faibles puissances, le C/N_0 est presque stable (autour de -35 dBHz). Cette mesure est très inattendue car le système de contrôle de puissance possède de bonnes performances lors des mesures effectuées avec l'analyseur de spectre. Nous avons remarqué que même en utilisant un atténuateur variable externe, nous obtenons le même résultat. Quelques tests effectués nous ont permis de conclure que la carte RF rayonne. Vu que le récepteur GPS est très sensible (il est capable de détecter des puissances de l'ordre de -130 dBm) et à cause de la proximité entre les deux équipements (récepteur et carte RF), le récepteur capte ces petits rayonnement et par conséquent, la diminution de la puissance à son entrée est comblée par la puissance rayonnée ce qui explique le planché de la valeur du ratio C/N_0 dans les basses puissances. Il faudrait alors éliminer l'effet du rayonnement de la carte RF. Comme recommandation, il faudrait connecter la carte RF et le récepteur par un long câble afin d'augmenter la distance entre les deux équipements et faire un test pour retracer la réponse du récepteur et observer son comportement.

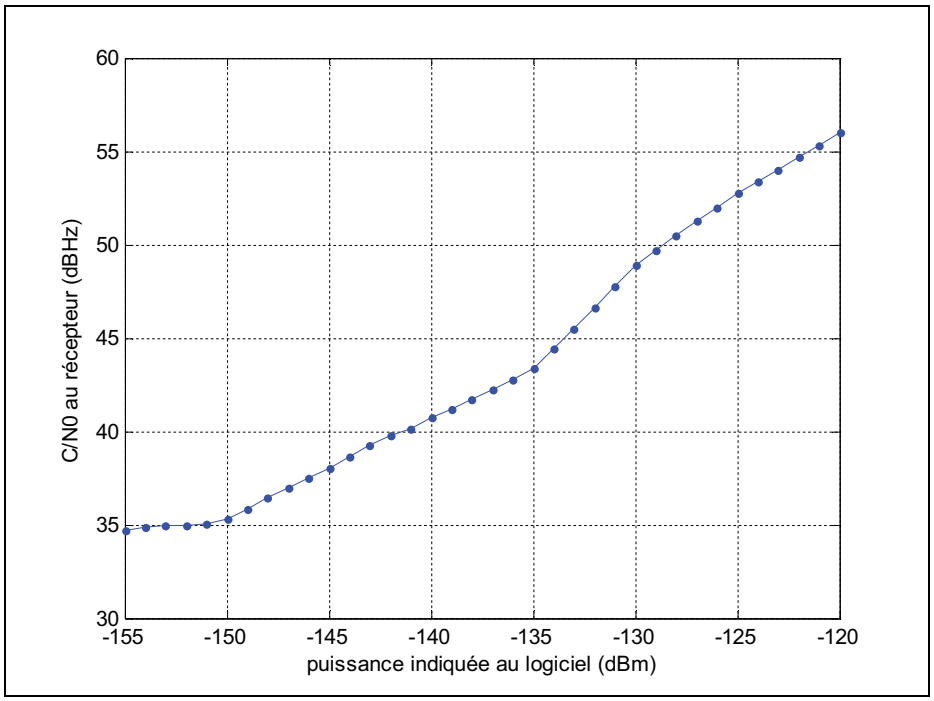

Figure 2.28 Réponse du système de contrôle de puissance avec un récepteur GPS commercial.

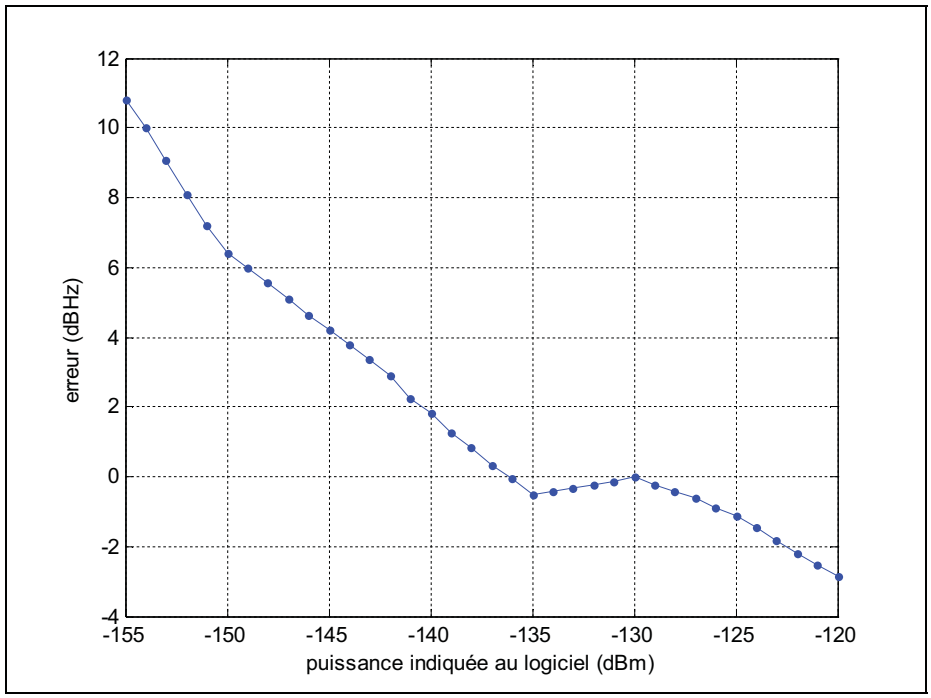

Figure 2.29 Erreur entre les valeurs du ratio C/N_0 observées et les valeurs théoriques.

2.8 Conclusion

Dans ce chapitre, nous avons décrit les étapes de conception du système de contrôle de puissance, ainsi que ses différents modes de fonctionnement. Le simulateur est maintenant capable de contrôler le niveau de puissance des signaux GPS et Galileo à travers une interface logicielle. Il peut par conséquent agir sur le ratio C/N_0 perçu par le récepteur. Un contrôle sur le bruit est nécessaire pour que l'utilisateur puisse tester le récepteur sous d'autres conditions (avoir un certain niveau du signal avec un ratio C/N_0 dégradé). Dans ce contexte, un contrôle du bruit s'impose et un système de contrôle du ratio C/N_0 reposant sur le système de contrôle de puissance et du bruit est développé. Ce système fait l'objet du chapitre suivant.

Equation Chapter 3 Section 1

CHAPITRE 3

SYSTÈME DE CONTRÔLE DU RAPPORT C/N_0

Dans les systèmes de positionnement par satellites, les signaux arrivent aux récepteurs avec des niveaux de puissances très faibles (de l'ordre de -130 dBm). Ils ne sont pas visibles même par un analyseur de spectre, car ils sont noyés dans le bruit thermique. Le choix de la technique CDMA et donc l'étalement de spectre permet de détecter ces signaux et de les ressortir du plancher de bruit afin d'extraire les informations de navigation. Le bruit est essentiellement un bruit thermique et peut changer selon la température du milieu où se trouve le récepteur. Il peut aussi provenir d'un autre signal interférent dans la bande du signal. Pour que les utilisateurs du simulateur de constellations GNSS accèdent à un contrôle de ce niveau de bruit, nous avons entrepris le développement d'un système de contrôle du rapport C/N_0. Ce chapitre traitera les étapes de conception d'un tel système.

3.1 Principe du contrôle du rapport C/N_0

3.1.1 Définition du rapport C/N_0

Il existe plusieurs quantités qui permettent de définir la qualité d'un signal à la réception. Elles permettent de déterminer la capacité d'un récepteur à détecter un signal et à extraire les informations nécessaires avec un certain TEB (Taux d'erreurs binaire) ou en anglais BER (*Bit Error Rate*). On peut distinguer plusieurs types de quantités, essentiellement des rapports de puissances, qui permettent de caractériser la qualité d'un signal à savoir le SNR (*Signal to Noise Ratio*), le C/N (*Carrier to Noise Ratio*) ou encore le C/N_0 (*Carrier to Noise density Ratio*). Dans le cas de la communication par satellite, la quantité la plus utilisée est le C/N_0 qui représente le rapport, souvent exprimé en dBHz, entre la puissance du signal et la densité du bruit. Sa définition mathématique est la suivante (WJ Communication Inc, 2004):

$$\left(\frac{C}{N_0}\right)_{dBHz} = 10\log\left(\frac{P_{signal}}{N_0}\right) \tag{3.1}$$

N_0 désigne généralement la densité spectrale de puissance du bruit thermique N_T. Comme son nom l'indique, il est une fonction de la température. Il est indépendant de la fréquence ou de la bande du signal (WJ Communication Inc, 2004) et son expression est la suivante :

$$N_T = kT \tag{3.2}$$

où k désigne la constante de Boltzmann, soit $1.38 \cdot 10^{-23}$ J/°K. La température est exprimée en °K (Kelvin) qui est relié à l'échelle Celsius par l'équation :

$$T_{^\circ\mathrm{K}} = 273.15 + T_{^\circ\mathrm{C}} \tag{3.3}$$

La valeur de la densité du bruit thermique de l'équation 3.2 est exprimée en J (Joules). Vu que $1J = 1W \times 1s$, on peut alors l'exprimer en W/Hz et sa valeur typique à la température ambiante (autour de 290°K ou 17 °C) est de -174 dBm/Hz.

3.1.2 Fonctionnement du système de contrôle du rapport C/N_0

Comme décrit au début de ce chapitre, le bruit thermique change avec la température. Le récepteur peut être à la surface de la Terre, où la température peut varier en fonction de la saison ainsi qu'en fonction de l'emplacement. Il peut être à bord d'un avion, où la température change avec l'altitude. La variation de la densité du bruit thermique est illustrée dans la Figure 3.1.

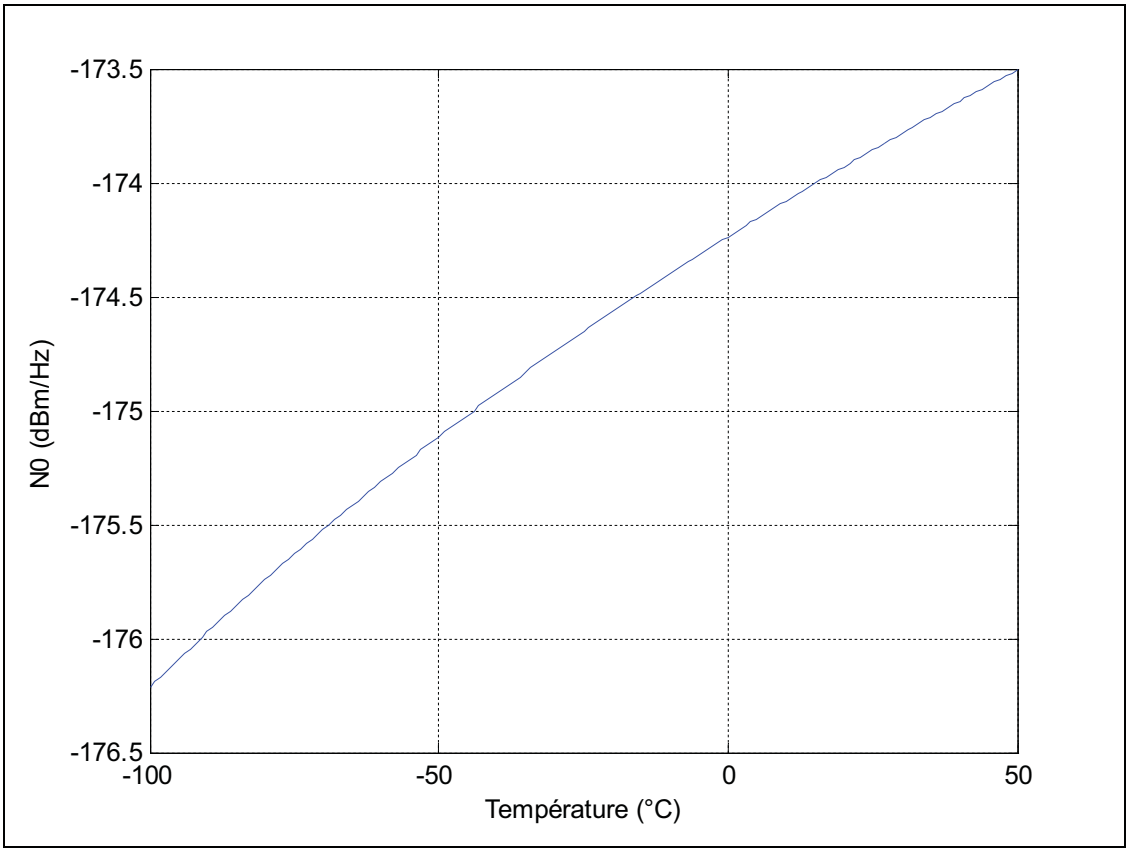

Figure 3.1 Effet de la température sur la densité du bruit thermique.

Sur la surface de la terre, la température peut varier de -40 °C à 40 °C voire 50 °C. Avec l'altitude, la température peut descendre jusqu'à -60°C. Avec la Figure 3.1, nous remarquons qu'il peut y avoir, seulement, une variation de l'ordre de 1.5 dB sur cette large plage de température. Néanmoins, l'affaiblissement du signal ou la présence d'interférents réduisent le rapport C/N_0. Un contrôle du rapport C/N_0 est alors nécessaire pour permettre à l'utilisateur de tester la robustesse de son récepteur.

Le contrôle du rapport C/N_0 peut se faire de deux manières, soit par la variation de la puissance du signal, soit par la variation de la densité du bruit ou les deux en même temps. Ces deux types de contrôle sont nécessaires pour le simulateur de constellation GNSS, car ils permettent à l'utilisateur une flexibilité dans le choix des paramètres et par conséquent une caractérisation plus rigoureuse des récepteurs de tests. Le contrôle de la puissance du signal

est effectué sur deux étapes. La première se produit dans la partie IF du simulateur et elle définit le gain des signaux provenant de chaque satellite par rapport à un niveau de référence. Ce niveau correspond à la puissance de sortie de la carte RF et il est fixé par la deuxième partie du contrôle. Il est définit pour chaque fréquence vu que les signaux des satellites d'une constellation sont combinés ensemble dans la partie IF. Cette opération est pilotée par le système de contrôle de puissance traité dans le chapitre précédent. Le contrôle de la densité du bruit est fait par le biais d'un générateur de bruit dont la puissance est variable et qui se combine avec les signaux. De cette façon, l'utilisateur aura un contrôle total sur la structure du signal.

3.1.3 Interface graphique de l'usager

L'interface graphique est la partie visible à l'utilisateur. Elle lui permet de commander la puissance des signaux et la puissance du bruit. Le contrôle du rapport C/N_0 étant en relation directe avec la puissance du signal, nous avons décidé d'intégrer la commande du système de contrôle de puissance avec le système de contrôle du rapport C/N_0 (Le changement du C/N_0 implique un changement de la puissance dans le simulateur). Il suffit à l'utilisateur de changer la valeur de la densité de bruit (voir Figure 2.17) et la partie logicielle se charge de l'envoi de la commande adéquate au FPGA via le bus FSB.

3.2 Survol de notions sur le bruit

Le bruit est un signal aléatoire qui possède plusieurs propriétés statistiques qu'il faut respecter lors de l'implémentation d'un générateur. Il y a plusieurs types de bruit à savoir le bruit thermique, le bruit généré par les composants électroniques, etc. Mais ils ont des caractéristiques communes (Pozar M, 2005). Dans le domaine des télécommunications, le bruit blanc est une approximation proche de la réalité et pour cette raison, nous allons nous intéresser à la théorie de ce bruit et les possibilités d'implémentation.

3.2.1 Fonction d'autocorrélation

La fonction d'autocorrélation permet de détecter les régularités dans un signal quelconque comme une périodicité cachée par un bruit qu'on ne peut pas observer facilement (Proakis John G, 2000). Elle représente une sorte de comparaison du même signal à deux instants différents. En statistique, pour un processus stochastique X et pour deux instants t_1 et t_2, l'expression de la fonction d'autocorrélation $R(t_1,t_2)$ est donnée par l'équation (3.4) où X_{t1} et X_{t_2} représentent, respectivement, les variables aléatoires du processus X aux instants t_1 et t_2. Aussi, m_{t_1} et m_{t_2} représentent, respectivement, leurs moyennes et. σ_{t_1} , σ_{t_2} leurs écart-types.

$$R(t_1,t_2)=\frac{E\left[(X_{t_1}-m_{t_1})(X_{t_2}-m_{t_2})\right]}{\sigma_{t_1}\sigma_{t_2}} \tag{3.4}$$

En traitement de signal, on n'effectue pas la normalisation par le produit des écarts-types. Pour un processus aléatoire f stationnaire au sens large (voir section 3.2.2), la fonction d'autocorrélation $R(\tau)$ est exprimée, dans le domaine continu, par l'équation (3.5) où f^* représente le conjugué complexe de la fonction f.

$$R(\tau)=\int_{-\infty}^{+\infty} f(t+\tau)f^*(t)dt=\int_{-\infty}^{+\infty} f(t)f^*(t-\tau)dt \tag{3.5}$$

Dans le domaine discret, pour un signal x_n stationnaire au sens large, la fonction d'autocorrélation est définie par l'équation (3.6) où m représente la moyenne du signal.

$$R(j)=\sum_n (x_n-m)(x_{n-j}-m) \tag{3.6}$$

3.2.2 Processus stationnaire au sens large

Un processus aléatoire est dit stationnaire au sens large si ses moments de premier et de second degré ne dépendent pas du temps (Proakis John G, 2000). En d'autres termes, son espérance mathématique et sa fonction d'autocorrélation sont invariants au cours du temps (voir les équations (3.7) et (3.8)).

$$E\left[x(t)\right] = m_x(t) = m_x(t+\tau) \quad \forall \tau \in \text{IR} \tag{3.7}$$

$$R_x(t_1, t_2) = R_x(t_1 - t_2) = R_x(\tau) \tag{3.8}$$

3.2.3 Densité spectrale de puissance

La DSP (Densité Spectrale de Puissance) ou en anglais PSD (*Power Spectral Density*) est un aspect très important du traitement du signal. Elle est définie comme le carré du module de la transformée de Fourrier divisé par le temps d'intégration (Barkat M, 2005). Cependant, en télécommunication, les signaux traités sont souvent de nature aléatoire et par conséquent, nous ne pouvons pas calculer leurs transformées de Fourrier. Le signal n'étant pas connu, on peut utiliser ses propriétés statistiques. Le théorème de Wiener-Khintchine stipule que la densité spectrale de puissance d'un signal aléatoire stationnaire au sens large $S_X(f)$ correspond à la transformée de Fourrier de sa fonction d'autocorrélation (équation (3.9)).

$$S_X(f) = \int_{-\infty}^{+\infty} R_X(\tau).e^{-j2\pi f\tau} d\tau \tag{3.9}$$

3.2.4 Le bruit blanc

Le bruit blanc est un signal aléatoire qui n'est pas corrélé, c'est-à-dire, que la valeur du bruit à un instant donné, n'est pas corrélée (n'a pas de relation) avec ses valeurs à d'autres instants. La fonction d'autocorrélation d'un bruit blanc est alors, la fonction Dirac. En d'autres termes, elle est maximale en zéro et nulle en tout autre instant (Proakis Jhon G, 2000). Par

conséquent, la densité spectrale de puissance d'un bruit blanc est une constante sur toutes les fréquences. Il est une construction purement théorique, car une DSP constante sur toutes les fréquences implique une puissance infinie. Cependant, un bruit peut avoir une DSP constante sur une bande de fréquence limitée (bande du signal utile par exemple) et par abus de langage, on l'appelle bruit blanc aussi. Un bruit blanc est alors un signal aléatoire de moyenne nulle ayant la fonction Dirac comme fonction d'autocorrélation. Le Tableau 3.1 résume les propriétés d'un bruit blanc.

Tableau 3.1 Propriétés d'un bruit blanc

Propriété	**Valeur**
Moyenne	0
Fonction d'autocorrélation	$R(\tau) = \frac{N_0}{2}\delta(\tau)$
DSP	$\frac{N_0}{2}$

Les valeurs, dans le temps, d'un bruit blanc n'ont pas de restriction pourvu qu'elles ne soient pas corrélées. Ainsi, une séquence binaire peut représenter un bruit blanc. Il peut suivre une distribution de probabilité à savoir poisson, Cauchy ou gaussienne (wikipedia, 2009a).

3.3 Solutions d'implémentation d'un bruit banc

Pour implémenter un bruit blanc, nous avons besoin de créer un processus aléatoire qui satisfait ses propriétés statistiques. Générer un processus purement aléatoire ne peut se faire que d'une façon naturelle (mouvement des électrons suivant les lois de la mécanique quantique probabiliste, par exemple). Le caractère aléatoire d'un processus ne peut être généré numériquement. En effet, le développement d'un algorithme pour générer un tel type de variable est impossible à réaliser vu que le caractère intrinsèque de l'algorithme est déterministe. Cependant, on peut générer des variables, dites pseudoaléatoires qui peuvent, sous certaines conditions, satisfaire nos besoins.

Dans le contexte du simulateur GNSS, le but est de combiner le bruit que nous allons générer avec les signaux. L'implémentation d'un tel bruit peut alors, se faire de façon numérique dans la partie logicielle ou dans la partie IF du simulateur, ou de façon purement analogique qu'on intègre dans la partie RF.

3.3.1 Implémentation analogique

La caractéristique la plus importante d'un bruit blanc est son caractère aléatoire. La première idée est alors d'utiliser un phénomène naturel. Une implémentation analogique est alors la seule qui peut satisfaire cette condition.

On peut générer un bruit blanc par le biais de dispositifs à chauffage qui permettent de contrôler le bruit thermique et celui des composants (croissant avec l'augmentation de la température) mais un tel système requiert un grand espace pour donner des résultats satisfaisants (Dusausay S, 2002).

Les composants électroniques génèrent un bruit blanc, mais il est faible (quelques μV). Par exemple, une diode Zener branchée de la manière illustrée dans Figure 3.2 permet, théoriquement, de générer un bruit blanc (Dusausay S, 2002). Polariser la diode dans son coude Zener permet de maximiser le bruit qu'elle génère. Le condensateur a pour rôle de bloquer la composante continue. Ce montage permet de générer un bruit blanc sur une bande étroite à basse fréquence.

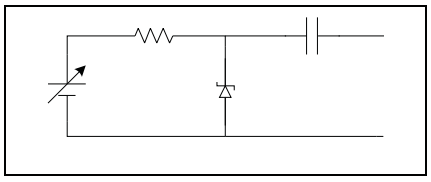

Figure 3.2 Générateur d'un bruit analogique simple.

Afin d'intégrer un générateur analogique dans le simulateur GNSS, les montages deviennent plus compliqués afin de couvrir la totalité de la bande passante et nécessitent plusieurs étages d'amplifications ainsi qu'un contrôle absolu. Le bruit peut être combiné avec les signaux dans trois points du module RF.

- Avant les filtres IF : le but est alors de générer un bruit blanc qui couvre la bande des signaux à la sortie de la partie IF du simulateur. Il devra alors, couvrir au moins la bande des 60 à 80 MHz et il devra être combiné à chacun des trois signaux;
- Après le combineur RF : il faudrait intégrer le bruit avec les signaux après les avoir combinés. Il faudrait donc générer un bruit d'une largeur de bande de 400 MHz pour couvrir toute la plage de fréquences des signaux. Comme le bruit généré par les composants électroniques couvre une bande de fréquence limitée, il faudrait tout un réseau de mixeurs et de combineurs pour couvrir la totalité de la bande de fréquences désirées;
- Après les atténuateurs : Cette solution est identique à la deuxième, mais elle comporte un avantage. La puissance du bruit n'a pas à être élevée vu que les signaux sont déjà atténués par les atténuateurs de la carte RF.

Une implémentation analogique du générateur de bruit blanc est complexe vu les différents obstacles rencontrés, mais elle présente l'avantage d'avoir un signal purement aléatoire. L'implémentation numérique possède plusieurs avantages à savoir la simplicité de l'implémentation (algorithme de génération de variables aléatoires). Nous avons alors, décidé d'adopter cette solution pour l'implémentation du générateur de bruit blanc. Dans ce qui suit, nous détaillerons l'implémentation numérique.

3.3.2 Implémentation numérique

Une implémentation numérique d'un bruit blanc est possible à l'aide d'un algorithme de génération de variables pseudoaléatoires pouvant satisfaire les propriétés statistiques adéquates. Les signaux de la constellation simulés sont créés dans la partie IF du simulateur.

Par conséquent, il faut sommer le bruit avec les signaux juste avant leurs transferts au CNA, à une cadence de 100 MHz.

Pour une implémentation dans la partie logicielle du simulateur, il faudrait générer une variable aléatoire avec une cadence de 100 MHz, c'est-à-dire 10^8 valeurs par seconde. La communication avec la partie IF se fait avec une fréquence inférieure à 100 MHz. Cette contrainte rend difficile l'implémentation du bruit dans la partie logicielle. L'implémentation dans la partie IF permet de s'affranchir de tous les problèmes de vitesses et de communications. Il suffit alors de trouver la meilleure solution pour intégrer le générateur de bruit.

Le processus gaussien est représentatif de plusieurs phénomènes naturels, dont le bruit. Nous avons alors choisi d'implémenter le bruit blanc comme étant gaussien.

3.4 Génération d'un bruit blanc gaussien

Le choix de l'implémentation numérique d'un bruit blanc gaussien restreint le système à la génération d'une variable aléatoire gaussienne centrée réduite, dont les valeurs ne sont pas corrélées.

3.4.1 Variable aléatoire gaussienne

Définition

Une variable aléatoire suit une loi normale de moyenne μ et de variance σ^2 si elle admet une densité de probabilité f telle que (Barkat M, 2005):

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{3.10}$$

On dit alors que la variable aléatoire est gaussienne et elle est notée généralement $\mathcal{N}(\mu,\sigma^2)$. La Figure 3.3 illustre quelques tracés de la densité de probabilité de la loi normale pour différentes moyennes et variances.

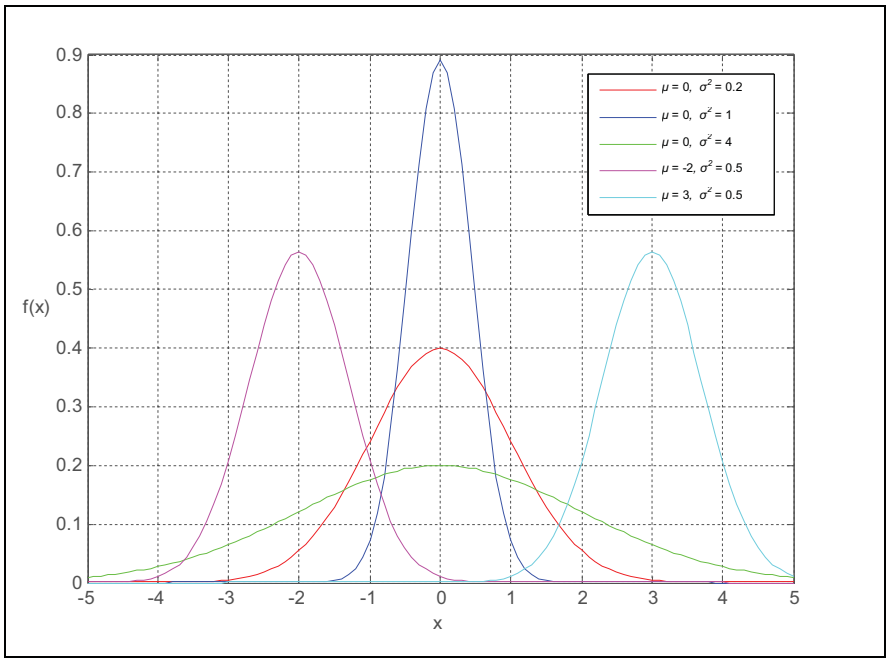

Figure 3.3 Densité de probabilité de la loi normale.

La loi normale centrée réduite admet une moyenne nulle et une variance unitaire et par conséquent sa densité de probabilité est exprimée de la façon suivante :

$$f(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}} \tag{3.11}$$

Propriétés

La loi normale possède plusieurs propriétés (Barkat M, 2005). Nous citerons ici celles qui sont importantes pour la suite de notre travail :

- Si deux variables aléatoires X_1 et *X2* indépendantes suivant, respectivement, les lois $\mathcal{N}(\mu_1, \sigma_1^2)$ et $\mathcal{N}(\mu_2, \sigma_2^2)$, alors la variable $X_1 + X_2$ suit la loi $\mathcal{N}(\mu 1+\mu 2, \sigma_1^2 + \sigma_2^2)$;
- Si X est une variable aléatoire suivant la loi normale $\mathcal{N}(\mu,\sigma^2)$ et a, b sont deux réels, alors $aX + b$ suit la loi $\mathcal{N}(a\mu+b,(a.\sigma)^2)$.

Théorème de la limite centrale

Soient X_1, X_2 ... X_n des variables aléatoires définies dans le même espace de probabilité, de même d'espérance μ et de même écart-type σ, indépendantes et suivant la même loi de probabilité. Alors, la variable aléatoire $S_n = \sum_{i=1}^{n} X_i$ admet $n\mu$ pour espérance et $\sqrt{n}\sigma$ pour écart-type, et elle tend vers la loi normale $\mathcal{N}(n\mu,n\sigma^2)$ quand n tend vers l'infini soit (Barkat M, 2005).

$$\lim_{n\to+\infty} S_n \sim \mathcal{N}(n\mu,n\sigma^2) \tag{3.12}$$

Pour illustrer ce théorème, nous allons prendre un exemple simple. Soient n = 20 variables aléatoires X_i suivant une distribution uniforme dans le segment [0 1]. Elles ont chacune alors, une espérance μ = 0.5 et une variance σ = 1/12. La Figure 3.4 illustre la distribution de la variable aléatoire S exprimée par l'équation (3.13).

$$S = \frac{1}{\sqrt{k}} \sum_{i=1}^{k} X_i \tag{3.13}$$

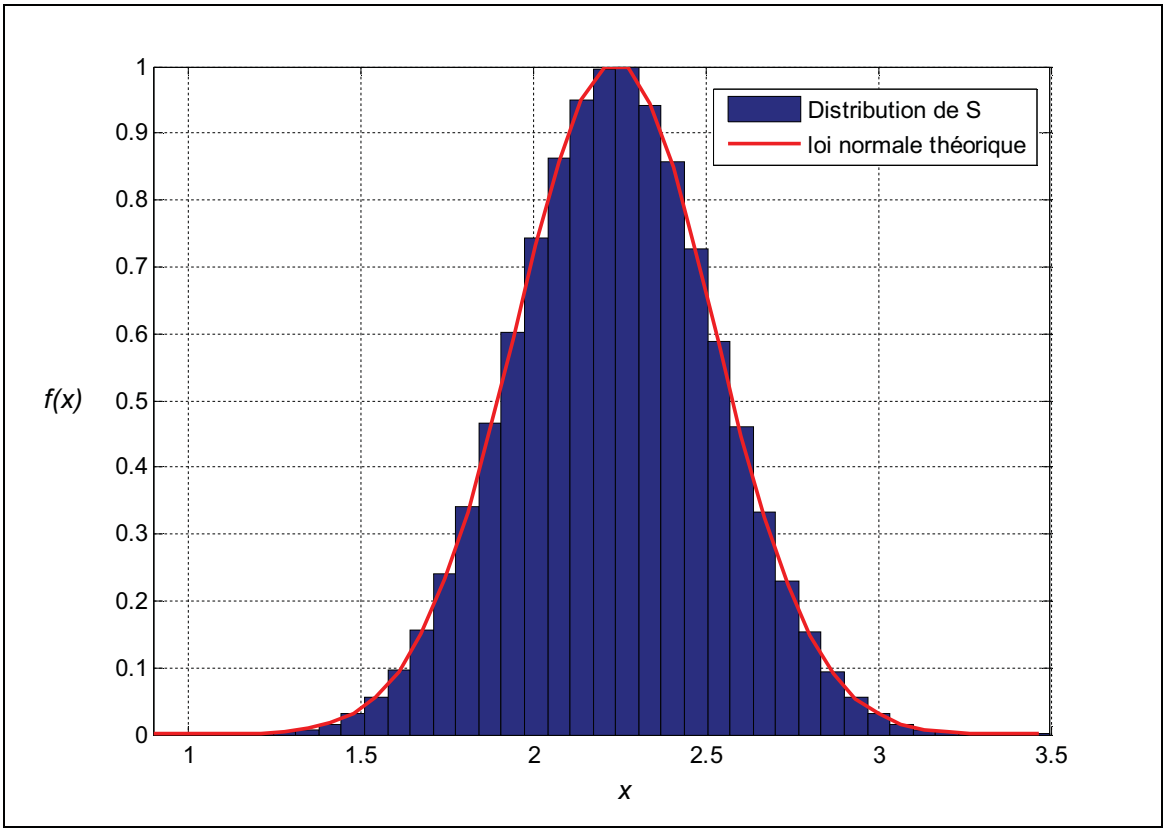

Figure 3.4 Exemple d'illustration du théorème de la limite centrale.

La Figure 3.4 présente une superposition de la distribution de la variable aléatoire S avec la loi normale théorique avec $\mu = \frac{1}{2\sqrt{k}}$ et $\sigma^2 = \frac{1}{12}$. Théoriquement, la variable S devrait avoir ces valeurs comme espérance et variance respectivement. Nous remarquons alors, d'après la figure, et les calculs de l'espérance et de la variance sur MATLAB, que S suit vraiment la loi normale $\mathcal{N}(\frac{1}{2\sqrt{k}}, \frac{1}{12})$.

3.4.2 Algorithmes de génération de variables aléatoires gaussiennes

Le processus gaussien est utilisé dans plusieurs domaines. Quelques algorithmes et méthodes ont été mis en place afin de s'approcher de sa densité de probabilité et nous présenterons les plus performants et les plus utilisés d'entre eux.

Méthode de Box-Muller

La méthode Box-Muller permet de générer des variables aléatoires suivant une loi normale centrée réduite. Elle possède deux formes, la forme de base et la forme polaire.

Pour la forme de base, soit deux variables aléatoires indépendantes U_1 et U_2 uniformément distribuées sur l'intervalle [0 1]. Les variables exprimées dans les équations (3.14) et (3.15) sont alors indépendantes et suivent une loi normale centrée réduite (Ghazel A *et al*, 2001).

$$Z_1 = \sqrt{-2\ln(U_1)}\cos(2\pi U_2) \tag{3.14}$$

$$Z_2 = \sqrt{-2\ln(U_1)}\sin(2\pi U_2) \tag{3.15}$$

La forme de base a été implémentée sous MATLAB pour obtenir sa performance pour la génération d'une variable aléatoire gaussienne (nous avons simulé Z_1). Nous avons alors généré dix millions de points, et nous avons tracé la distribution normalisée de la variable en entier pour la superposer à la fonction de distribution de la loi normale normalisée (voir Figure 3.5) et ainsi, calculer l'erreur relative, $\xi(x)$, exprimée par l'équation (3.16) et illustrée par la Figure 3.6.

$$\xi(x) = \frac{f_X(x) - N(0,1)}{N(0,1)} \tag{3.16}$$

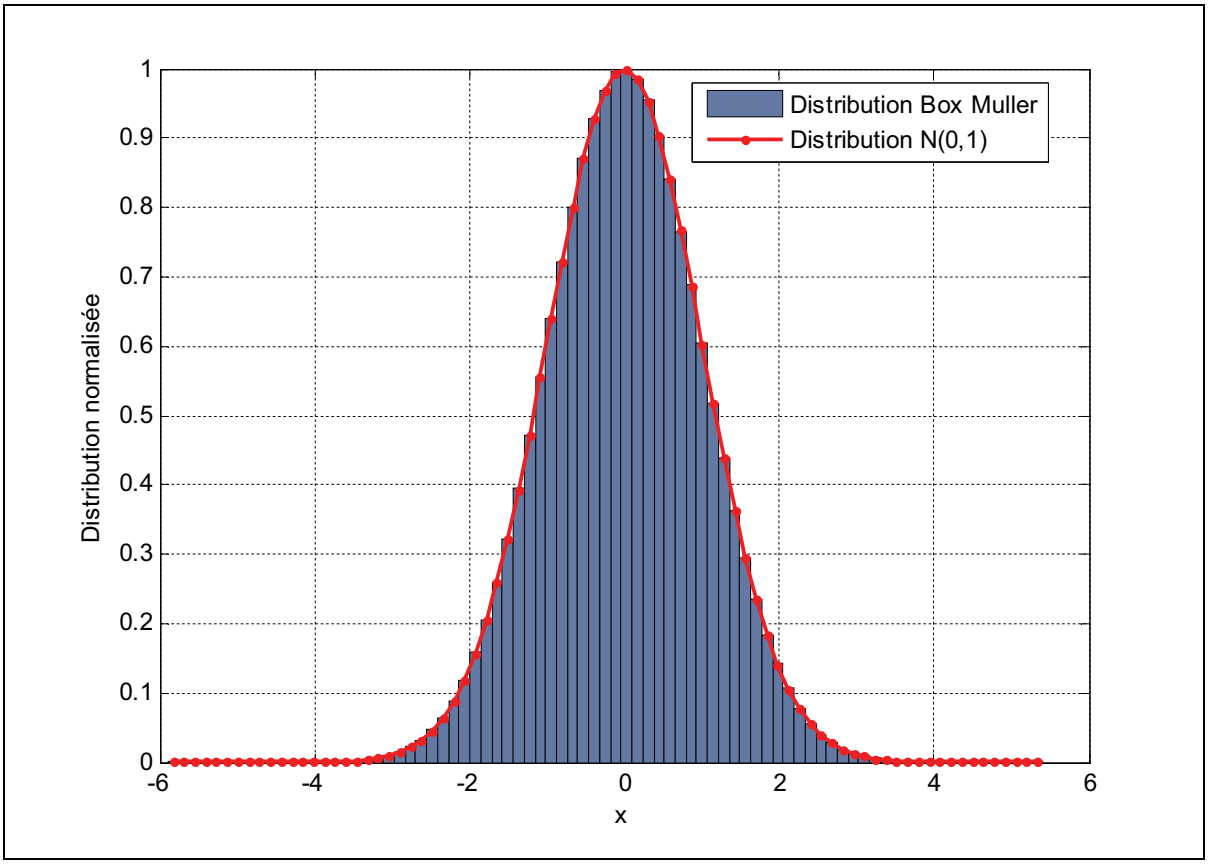

Figure 3.5 Distribution d'une v.a gaussienne générée par la forme de base de la méthode Box-Muller.

L'erreur relative de la distribution de la variable aléatoire gaussienne générée par la méthode Box-Muller pour dix millions de points ne dépasse pas les 6% pour $x < 4\sigma$ ce qui présente une bonne performance. Plus le nombre de points générés augmente, plus la distribution s'améliore.

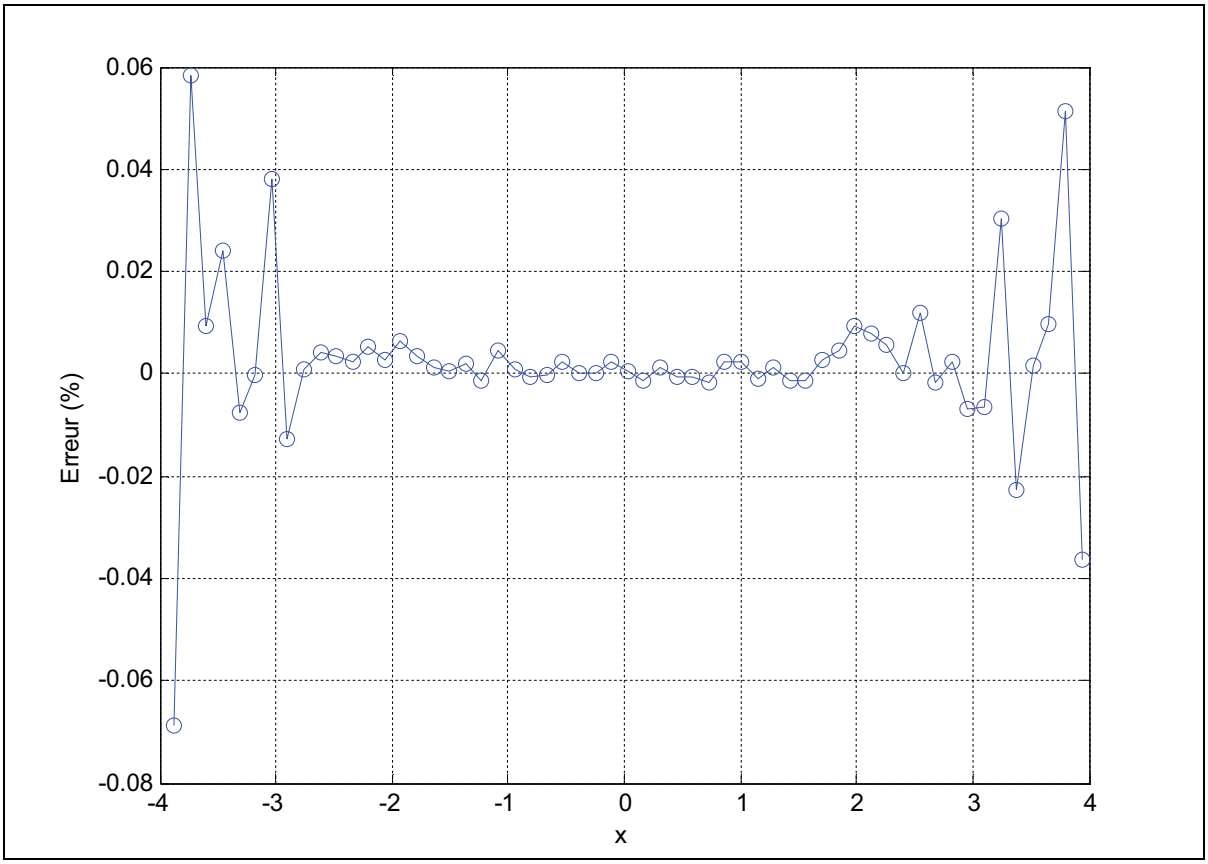

Figure 3.6 Erreur relative entre la distribution de la v.a générée par la forme de base de la méthode Box-Muller et la fonction de distribution de la loi normale N(0,1).

Pour la forme polaire, soit deux variables aléatoires indépendantes u et v uniformément distribuées sur l'intervalle [-1 1]. On calcule $s = u^2 + v^2$. Si $s = 0$ ou $s > 1$ le couple est rejeté (il doit être à l'intérieur du cercle unitaire). Sinon, le couple est accepté et les variables exprimées dans les équations (3.17) et (3.18) sont alors indépendantes et suivent une loi normale centrée réduite (Goodman J, 2005).

$$Z_0 = u\sqrt{\frac{-2\ln(s)}{s}} \tag{3.17}$$

$$Z_1 = v\sqrt{\frac{-2\ln(s)}{s}} \tag{3.18}$$

La forme polaire est plus rapide que la forme de base vu qu'elle n'effectue pas de transformations trigonométriques. Elle est une méthode de réjection qui élimine certaines combinaisons et le taux de réjection est de 21.46 % (Goodman J, 2005). La génération des variables aléatoires uniformes étant simple et très rapide, la forme polaire reste, malgré son grand taux de réjection, plus rapide que la forme de base.

Nous avons aussi implémenté la forme polaire et nous avons comparé la distribution de la loi normale $\mathcal{N}(0,1)$ avec celle de la variable aléatoire générée (voir Figure 3.7) et nous avons tracé son erreur relative illustrée par la Figure 3.8.

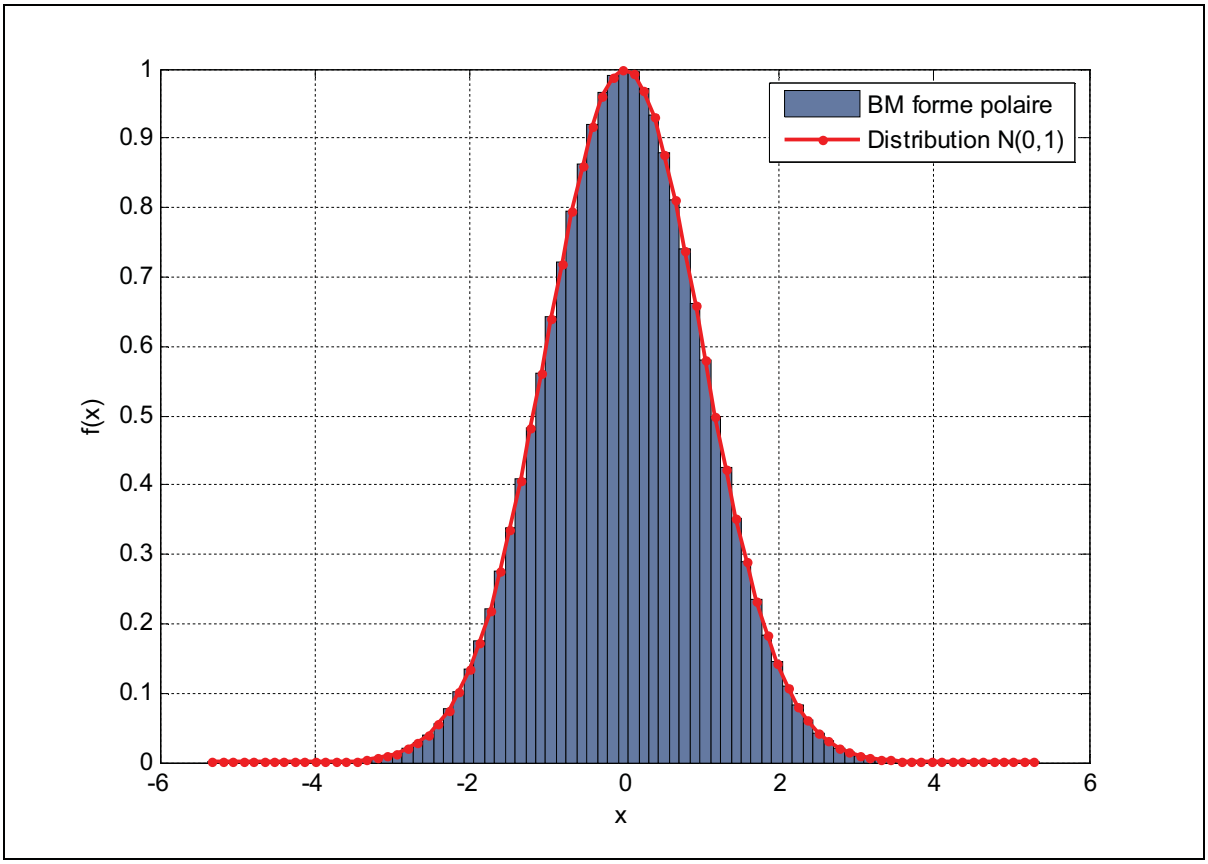

Figure 3.7 Distribution d'une v.a gaussienne générée par forme polaire de la méthode Box-Muller.

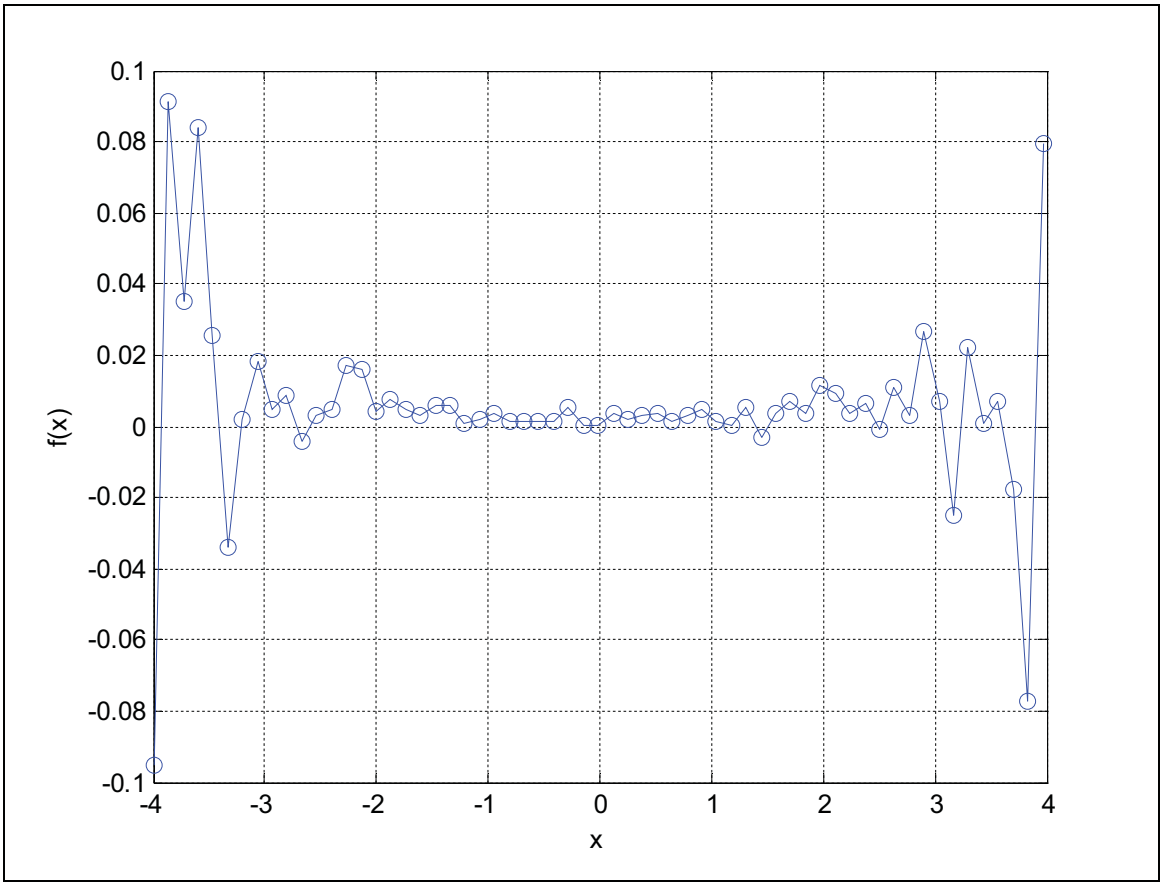

Figure 3.8 Erreur relative entre la distribution de la v.a générée par la forme polaire de la méthode Box-Muller et la fonction de distribution. de la loi normale N(0,1)

Nous remarquons que l'erreur relative est supérieure à celle de la variable aléatoire générée par la forme de base. Ceci n'est pas relié à la structure de la méthode, elle est due au nombre de points simulé. Dans la section précédente, nous avons généré dix millions de points alors qu'avec la forme polaire, près du quart des points ont été rejetés par l'algorithme. De ce fait, la performance de la méthode baisse et ainsi, l'erreur relative augmente.

Algorithme Ziggurat

L'algorithme Ziggurat permet de générer des variables aléatoires suivant une densité de probabilité décroissante strictement monotone comme la loi exponentielle. La méthode peut s'appliquer à des lois symétriques, comme la loi normale dans notre cas en choisissant le côté

décroissant de la fonction. C'est un algorithme basé sur la méthode de rejet, c'est-à-dire que la valeur de la variable choisie subit un test pour savoir si elle correspond aux spécifications de la densité de probabilité en question.

La méthode Ziggurat consiste en la couverture de la courbe de la densité de probabilité par une surface légèrement plus grande. Cette surface est composée de *n* régions ayant la même aire dont *n* -1 d'entre elles sont des rectangles empilés sur une portion non rectangulaire correspondant à la queue de la densité de probabilité. La Figure 3.9 illustre un exemple d'une telle surface pour la moitié de la densité de probabilité de la loi normale.

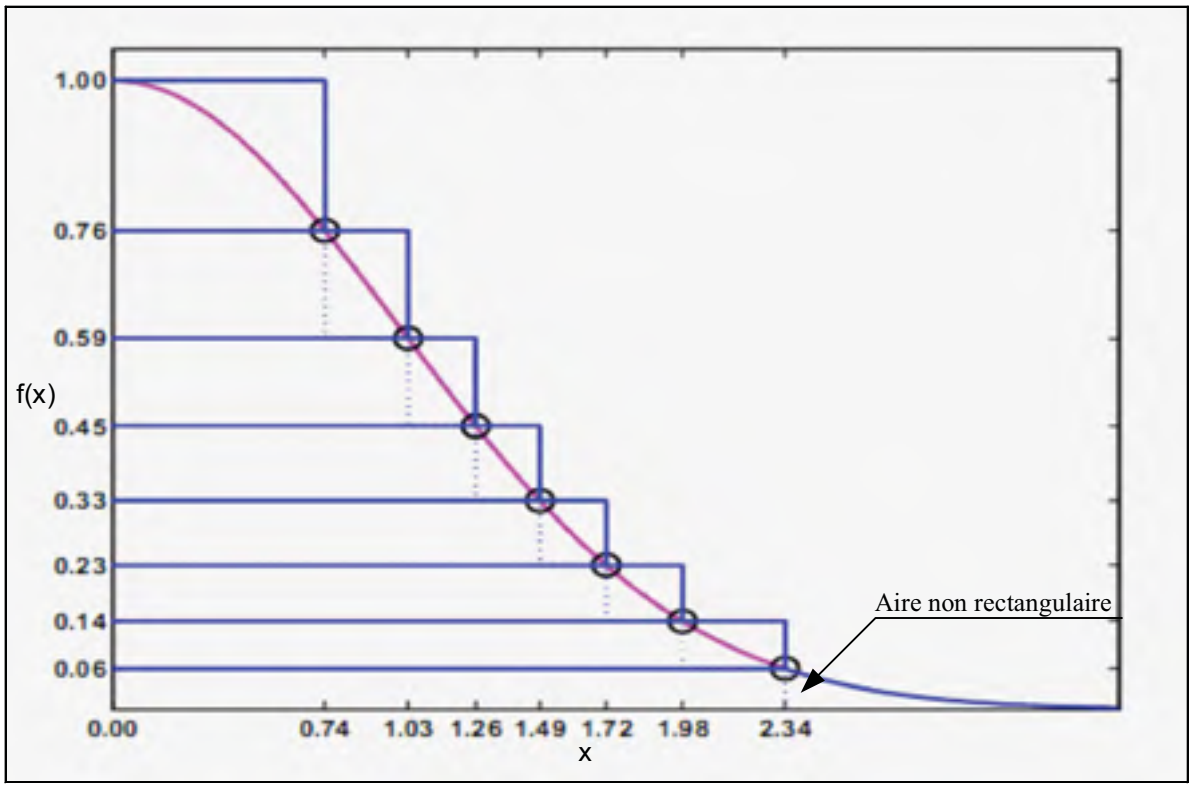

Figure 3.9 Construction d'une ziggurat pour la loi normale.
Tiré de Moler (2009, p 6)

La base de la Ziggurat, d'aire *A*, est formé d'un rectangle dont les coordonnées des points inférieurs gauche et supérieur droit sont respectivement (0,0) et $(x_1,f(x_1))$ et de l'ensemble des points $(x,f(x))$ tel que $x \geq x_1$. Tous les *n*-1 rectangles se situant au-dessus de cette base auront

la même aire. Le premier rectangle aura pour largeur x_1 et pour hauteur A/x_1 pour qu'il ait une aire égale à A. Ce rectangle intercepte la courbe de la distribution dans un autre point (x_2,y_2) tel que $y_2 = f(x_2)$ avec $y_2 = y_1 + A/x_1$. Ce rectangle inclut les points qui se situent à l'intérieur de la distribution entre y_1 et $y2$ mais, au contraire de la base de Ziggurat, elle inclut aussi les points entre x_1 et x_2 qui eux, n'appartiennent pas à la distribution. On continue de construire les rectangles R_i tel qu'ils aient une hauteur de H_i (équation (3.19)), une largeur W_i (équation (3.20)) et intercepte la courbe de la distribution en un point (x_{i+1},y_{i+1}) où y_i est exprimé par l'équation (3.21) et $x_i = f^{-1}(y_i)$.

$$H_i = \frac{A}{x_i} \tag{3.19}$$

$$W_i = x_i \tag{3.20}$$

$$y_i = y_{i-1} + H_{i-1} \tag{3.21}$$

Les rectangles sont empilés l'un sur l'autre et on choisit x_1 tel que $x_n = 0$ (de façon à ce que le dernier rectangle coïncide avec le haut de la fonction de distribution). Le choix de x_1 et de A dépend du nombre de surfaces n et ils sont calculés en résolvant le système d'équations (3.22) par le biais d'un algorithme de recherche.

$$\begin{cases} x_1 \cdot f(x_1) + \int_{x_1}^{+\infty} f(t)dt = A \\ x_{n-1} \cdot (f(0) - f(x_{n-1})) = A \\ x_i = f^{-1}(y_i) \\ y_{i+1} = y_i + \dfrac{A}{x_i} \end{cases} \tag{3.22}$$

L'algorithme Ziggurat nécessite le stockage des deux vecteurs de données x_i et y_i avant son exécution. Ceci permet d'accélérer le processus, car au lieu de les générer à chaque exécution, l'algorithme n'a qu'à les stocker en mémoire. Ayant alors, construit les tables x_i et y_i, et en se donnant deux variables aléatoires uniformément distribuées sur [0 1], l'algorithme est exécuté de la façon suivante (Tsang G.Marsaglia et Wai Wan, 2000):

1. On choisit aléatoirement une couche i avec une probabilité de $1/n$;
2. Soit $x = U_0 \cdot x_i$;
3. Si $x < x_{i+1}$ alors, retourner x;
4. Si $i = 0$, on choisit un point de la queue de la fonction de distribution;
5. Soit $y = y_i + U_1 (y_{i+1} - y_i)$;
6. Calculer $f(x)$, si $y < f(x)$ alors, retourner x;
7. Sinon, on recommence l'algorithme à l'étape 1.

Le but est de vérifier si le point x choisit à l'étape 2 est soit dans la zone (1) soit la zone (2) de la Figure 3.10 pour qu'il puisse être accepté. Après avoir choisi une couche, on effectue le test de l'étape 3. On vérifie si le point x est dans la zone (1). Si c'est le cas, le point est accepté, sinon on passe à une deuxième vérification. On crée l'ordonnée y et on teste avec l'étape 6 si le point est bien à l'intérieur de la courbe de la fonction de distribution. Si c'est le cas, le point est à l'intérieur de la zone (2) et le il est accepté, sinon, il ne peut être que dans la (3) et il est alors rejeté.

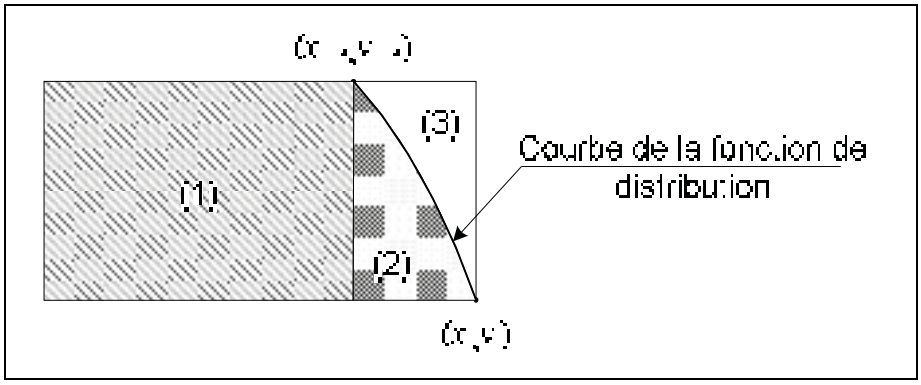

Figure 3.10 Déroulement de l'algorithme Ziggurat.

Il ne reste plus qu'à définir l'étape 4 de l'algorithme dans laquelle, on doit générer un point de la queue de la fonction de distribution. Pour la loi normale, le créateur de cet algorithme, Marsaglia (Tsang G.Marsaglia et Wai Wan, 2000) propose l'algorithme suivant :

1. Soit $x = -\frac{\ln(U_1)}{x_1}$;
2. Soit $y = -\ln(U_2)$;
3. Si $2y > x^2$ alors, retourner $x+x_1$;
4. Sinon, aller à l'étape 1 de l'algorithme global.

Cet algorithme est le plus rapide actuellement pour une implémentation logicielle, car il n'utilise pas beaucoup de calculs. La réjection de l'algorithme Ziggurat est autour de 2.28% avec une décomposition de la fonction de distribution en 128 niveaux (Tsang G.Marsaglia et Wai Wan, 2000), et donc, la majorité des points qu'il génère sont retenus. Les nouvelles versions de MATLAB utilisent une version améliorée de l'algorithme Ziggurat pour une génération très rapide de variable aléatoire gaussienne.

Nous avons implémenté cet algorithme, avec 256 niveaux, sous MATLAB afin de simuler et d'observer son erreur relative par rapport à la fonction de distribution de la loi normale centrée réduite. Nous avons obtenu alors, une bonne distribution gaussienne et une erreur relative illustrée par la Figure 3.11 qui ne dépasse pas les 5% pour dix millions de points générés.

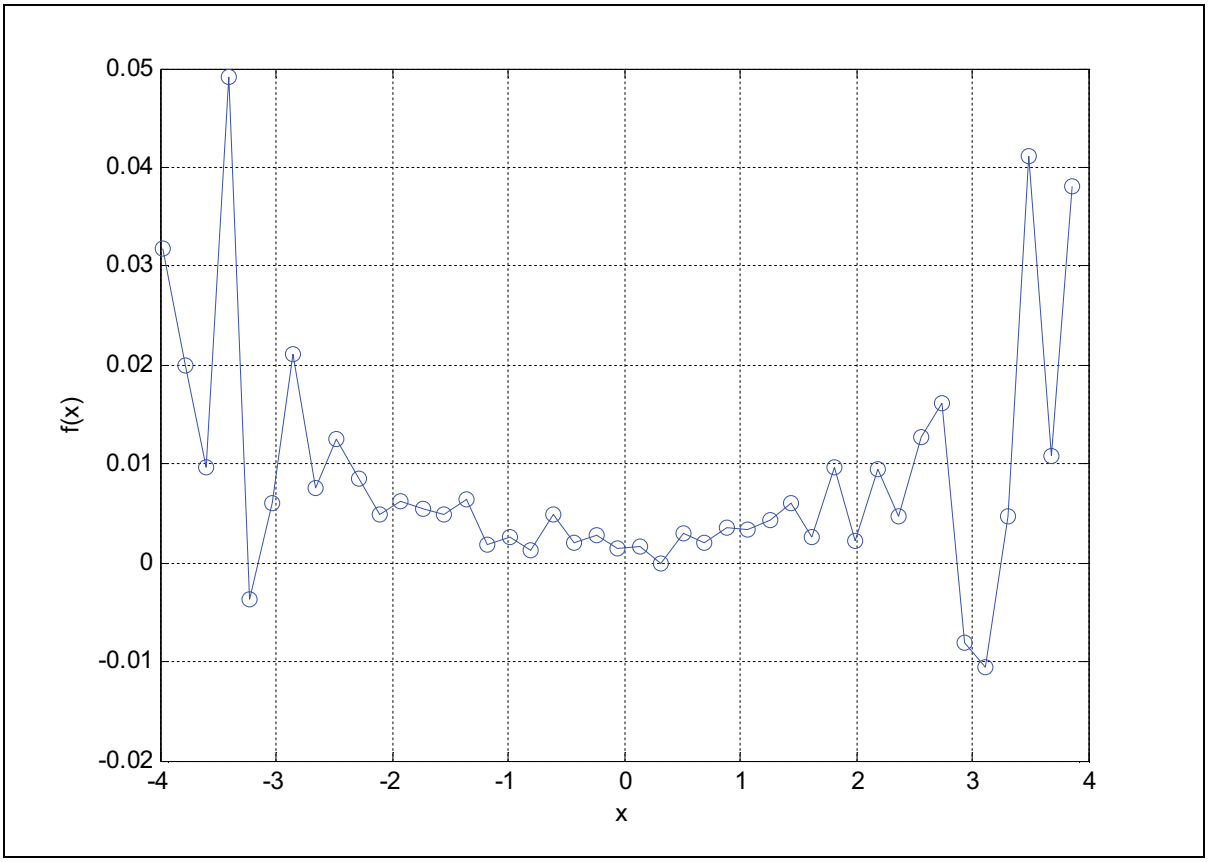

Figure 3.11 Erreur relative entre la distribution de la v.a générée par l'algorithme Ziggurat et la fonction de distribution de la loi normale N(0,1).

Comparaison et choix de l'algorithme

Les algorithmes mentionnés ci-dessus présentent tous de bonnes performances et permettent de générer une variable aléatoire gaussienne avec une erreur ne dépassant pas les 6%. Cependant, une comparaison entre ces algorithmes est nécessaire dans le cas d'une implémentation matérielle (FPGA) car dans ce cas, ce n'est pas que la rapidité de l'algorithme qui compte, vu qu'on est confronté à des ressources matérielles limitées. En effet, la partie IF du simulateur est au point de se saturer. Beaucoup de ressources ont été utilisées pour implémenter le fonctionnement de base du simulateur et le code VHDL doit être optimisé pour libérer des ressources pour d'autres canaux (satellites) et pour l'intégration des fonctions supplémentaire comme le multitrajet et notre générateur de bruit blanc gaussien.

La variable aléatoire gaussienne doit être générée au rythme de 100 MHz. Par conséquent, un algorithme qui présente un haut taux de réjection pourrait causer des problèmes. Pour utiliser un tel algorithme, il faut qu'il tourne à assez haute vitesse afin de combler les valeurs rejetées et il faut implémenter une file d'attente FIFO par exemple. Ceci augmente les contraintes d'implémentation, ce qui implique que la forme polaire de la méthode Box-Muller n'est pas candidate pour une implémentation matérielle. L'algorithme Ziggurat présente un taux de réjection faible et, par conséquent, la vitesse du système est légèrement plus haute que le taux de sorties des valeurs et la file d'attente sera de petite taille.

Une implémentation logicielle est séquentielle et, par conséquent, minimiser le calcul des fonctions complexes (fonctions trigonométriques, logarithme…) est très important. Dans ce sens, l'algorithme Ziggurat est nettement plus rapide que la forme de base de la méthode Box-Muller, car elle ne présente pas à chaque itération un calcul de fonctions complexes. Le Tableau 3.2 présente les fonctions utilisées dans les algorithmes Ziggurat et la forme de base de la méthode Box-Muller.

Tableau 3.2 Fonctions utilisées dans les algorithmes Ziggurat et Box-Muller

Fonction	**Box-Muller**	**Ziggurat**
Cosinus	✓	
Exponentielle		✓
Logarithme	✓	✓
Racine carrée	✓	
Multiplication	✓	✓
Division		✓

La méthode Box-Muller utilise les fonctions listées dans le Tableau 3.2 dans chaque itération alors que l'algorithme Ziggurat n'utilise la fonction exponentielle que lorsqu'il échoue dans le premier test et n'utilise le logarithme et la division que lorsqu'il doit générer une valeur de la queue de la distribution (voir section 3.4.2 – Algorithme Ziggurat) ce qui est

très rare vu que la probabilité d'avoir une valeur de la queue est très petite (taux d'acceptation à l'étape 3 de l'algorithme est de 98.05% (Tsang G.Marsaglia et Wai Wan, 2000)). Cette comparaison place l'algorithme Ziggurat comme meilleur algorithme pour une implémentation logicielle. Cependant, dans le cas d'une implémentation matérielle, nous devons intégrer toute fonction utilisée de l'algorithme même si elle n'est utilisée que quelques fois. Sinon, on doit implémenter une unité arithmétique et logique comme celle d'un processeur séquentiel, mais ce cas de figure nous fait revenir à la question de la génération des variables à une cadence de 100 MHz qui est difficile à réaliser avec une architecture séquentielle. Le pourcentage d'exécution d'une fonction dans l'algorithme n'est pas un critère de sélection dans le cas d'une implémentation matérielle. Nous allons alors nous intéresser au besoin de chaque algorithme en mémoire et en ressources matérielles.

Pour exécuter des fonctions complexes dans un FPGA, nous devons soit utiliser un algorithme approximatif (développement limité), ou créer une table de correspondance ou en anglais LUT (*Look Up Table*) qui permettra de correspondre une valeur de la fonction pour un intervalle de points à l'entrée. Le calcul des fonctions par approximation polynomiale est fait soit en parallèle (calcul de chaque puissance se fait à part), soit en utilisant une architecture de multiplication et accumulation. Dans les deux cas, il y a des inconvénients. Dans le premier, nous avons un gaspillage de ressources matérielles puisqu'on effectue chaque puissance dans un bloc (x, x^2, x^3…) en sachant qu'on a besoin d'au moins un ordre 5 pour une bonne approximation. Dans le deuxième cas, il faut au moins cinq coups d'horloges pour qu'on ait le résultat de la fonction et il faudra alors que ce bloc puisse rouler à au moins 500 MHz. L'utilisation d'une table de correspondance est un bon compromis entre la vitesse d'exécution et la précision des résultats. Ce dernier critère est proportionnel à la taille de la table utilisée (largeur du mot et nombre de valeurs stockées).

Nous allons maintenant nous intéresser aux ressources nécessaires pour la génération d'une variable aléatoire suivant la loi normale centrée réduite qui aura des valeurs allant jusqu'à $\pm 4\sigma$, ce qui est suffisant vu qu'au-delà, la probabilité d'apparition devient très petite (voir Tableau 3.3).

Tableau 3.3 Probabilité d'apparition des valeurs au-delà de 4σ pour la loi normale centrée réduite

Valeur	Probabilité
±4σ	$1.33 \cdot 10^{-4}$
±5σ	$1.48 \cdot 10^{-6}$
±6σ	$6.07 \cdot 10^{-9}$
±7σ	$9.13 \cdot 10^{-12}$
±8σ	$5.05 \cdot 10^{-15}$

L'algorithme Ziggurat utilise intrinsèquement 2 tables de données, soit x_i et y_i (voir section 3.4.2 – Algorithme Ziggurat). Généralement, la longueur de chacun de ces vecteurs est de 256 mots. Vu qu'on va prendre juste des valeurs inférieures à 4σ, la fonction exponentielle doit alors gérer des entrées allant de 0 à 4 pour une loi normale centrée réduite. Sachant que $\lim_{x \to \infty} e^{-x^2} = 0$, la subdivision de l'intervalle doit être adéquate pour avoir une bonne précision pour les valeurs élevées. La fonction logarithmique est utilisée dans l'intervalle $]0\ 1]$. La $\lim_{x \to 0} \ln(x) = -\infty$ et donc plus on s'approche de zéro, plus la valeur augmente et, comme la fonction exponentielle, on doit prendre en compte ce détail pour que la précision des valeurs très faibles soient acceptables. Une subdivision uniforme de l'intervalle de travail de ces deux fonctions dégrade les performances de l'algorithme utilisé (Ghazel A *et al*, 2001).

La méthode Box-Muller utilise la fonction cosinus dans l'intervalle $[0\ 2\pi]$ et on peut la représenter juste par un quart de période $[0\ \pi/2]$ (zone 1 sur la Figure 3.12) (Ghazel A *et al*, 2001). Le quart 4 est identique au premier alors que le 2 et le 3 peuvent être générés par une inversion du signe. Cette astuce réduit la taille de la table de 4 fois. Pour la fonction logarithme, la méthode Box-Muller rejoint l'algorithme Ziggurat. Il reste à traiter la fonction racine carrée. Puisque nous allons utiliser une table de correspondance pour la fonction logarithme, nous pouvons les intégrer ensemble et construire une table pour la fonction $f(x) = \sqrt{-\ln(x)}$ ce qui nous permet d'économiser sur le nombre de tables.

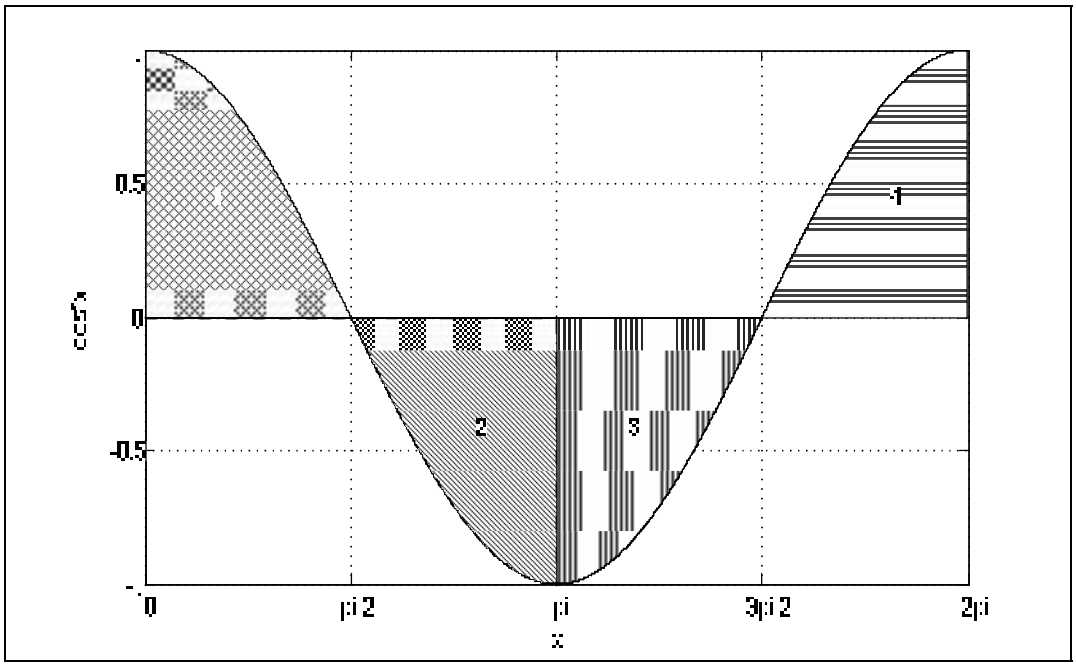

Figure 3.12 Fonction cosinus.

Pour résumer, la fonction logarithme de l'algorithme Ziggurat et la racine carrée combiné avec le logarithme de la méthode Box-Muller sont les mêmes. La fonction exponentielle consomme plus de ressources que la fonction cosinus vu qu'elle est répartie sur un intervalle plus grand ([0 4] contre [0 π/2]) et nécessite une subdivision complexe (non uniforme) afin d'obtenir une bonne précision pour les grands nombres. La méthode Box-Muller nécessite juste une multiplication de deux valeurs, alors que l'algorithme Ziggurat présente plusieurs conditions et plusieurs opérations ce qui implique une sortie non uniforme dans le temps et par conséquent une utilisation plus importante des ressources matérielles. Nous allons alors nous intéresser, dans ce qui suit, à l'implémentation de la forme de base de la méthode Box-Muller sur le FPGA de la partie IF du simulateur GNSS.

3.5 Implémentation du générateur de bruit blanc gaussien

La méthode Box-Muller consiste en la multiplication de deux fonctions *f* et *g* (équations (3.23) et (3.24)). Vu que nous avons choisi d'implémenter ces fonctions en utilisant des

tables de correspondances, nous allons nous intéresser à la manière de quantifier les deux intervalles de définitions ([0 1] et [0 π/2] respectivement) pour les deux fonctions *f* et *g*.

$$f(x) = \sqrt{-\ln(x)} \tag{3.23}$$

$$g(x) = \sqrt{2}\cos(2\pi x) \tag{3.24}$$

3.5.1 Quantification des fonctions *f* et *g*

Afin d'avoir une bonne distribution suivant la loi normale centrée réduite, le générateur de variables aléatoires doit atteindre des valeurs d'au moins 4σ.

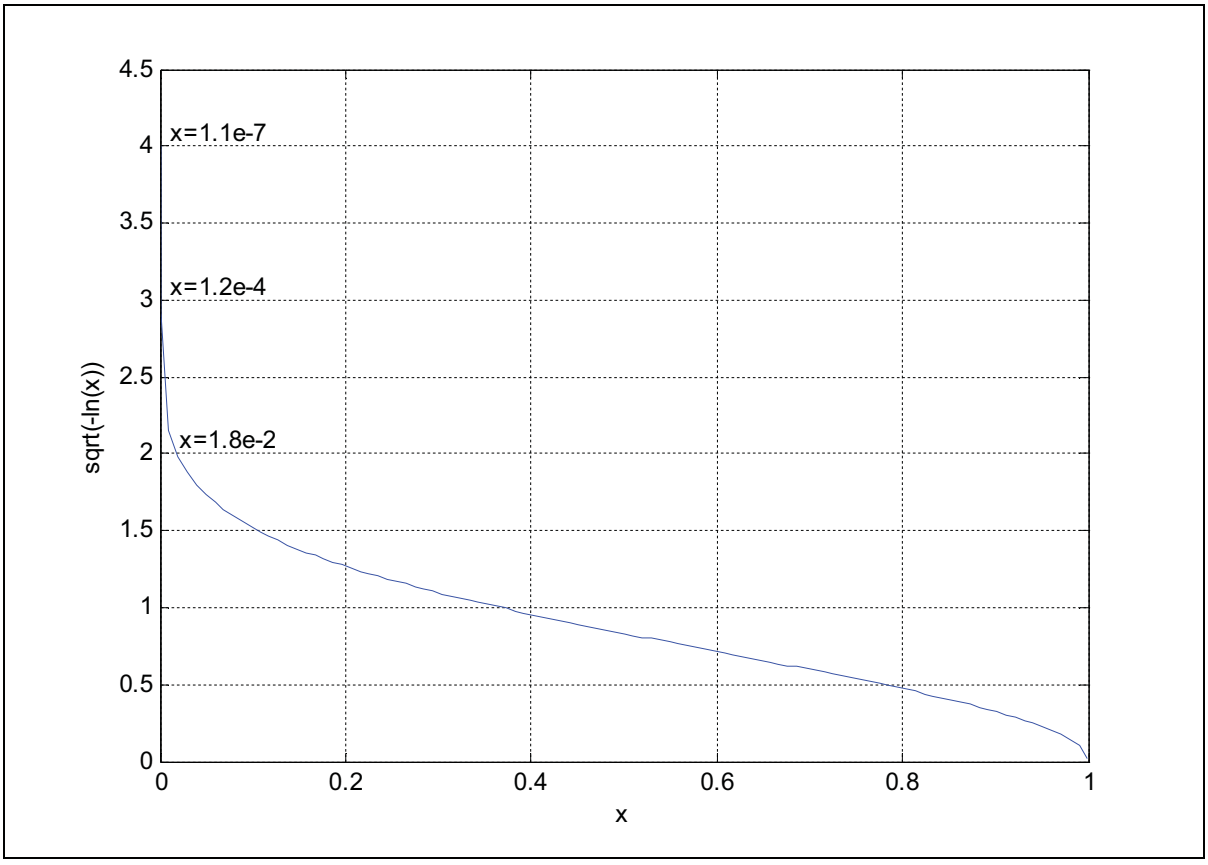

Figure 3.13 Fonction logarithme combinée avec la racine carrée.

En ce qui concerne la méthode Box-Muller, ceci n'est réalisable que si la fonction *f* atteint de valeurs supérieures à 4 vu que la valeur moyenne de la fonction *g* se situe autour de 1 ($2\sqrt{2}/\pi$ sur l'intervalle [0 π/2]). D'après la Figure 3.13, *x* atteint de très petites valeurs. Puisque $f^{-1}(4) < 10^{-7}$, le pas de quantification de la fonction (pas de subdivision de l'intervalle) doit être infiniment petit et, par conséquent, une table de correspondance doit avoir une très grande taille (Ghazel A *et al*, 2001). Une subdivision uniforme de l'intervalle de travail n'est pas alors adéquate pour l'implémentation de cette fonction. Une autre méthode a été présentée par (Ghazel A *et al*, 2001) et permet de s'affranchir de ce problème. Elle consiste en une quantification non uniforme récursive de l'intervalle [0 1] comme illustrée dans la Figure 3.14.

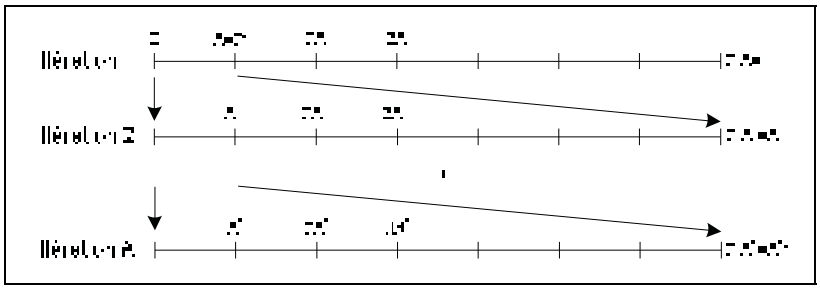

Figure 3.14 Quantification non uniforme récursive de la fonction *f*.

Le principe de la quantification non uniforme récursive est simple. On divise le segment [0 1] en 2^q sous-segments (*q* étant le nombre de bits nécessaires pour sélectionner un segment). Ensuite, on divise le premier sous-segment en 2^q parties. On refait la même opération *K* fois jusqu'à ce qu'on obtient les valeurs requises. Les segments auront alors une taille de $\Delta^r = 2^{-rq}$ où *r* représente le nombre de l'itération (r = 1,2,3…*K*). Nous utiliserons *K* générateurs de variables aléatoires uniformes $s_1, s_2 \ldots s_K$ pour sélectionner un des sous segments. La variable s_1 permet de sélectionner un des 2^q sous-segments de [0 1] de taille $\Delta = 2^{-q}$. Si s_1 est égale à zéro, on passe à la deuxième itération définie par s_2 qui permettra de choisir un sous segment. Si s_2 est égale à zéro, on passe à la troisième itération et ainsi de suite jusqu'à atteindre l'itération *K*. La probabilité de choisir un segment *s* de rang *r* est égale à la taille du segment soit Δ^r. Le processus est alors uniformément distribué sur le

segment [0 1] (Ghazel A *et al*, 2001). La valeur quantifiée, de la fonction *f*, associé au segment choisi est alors exprimé par l'équation (3.25) avec *m* le nombre de bits fractionnel utilisé pour représenter $f_r(s)$, $\lfloor x \rfloor$ représente la partie entière de *x* et δ est un réel compris entre 0 et 1 et donne la position relative dans le sous-segment choisit.

$$f_r(s) = \lfloor 2^m f((s+\delta)\Delta^r) \rfloor \cdot 2^{-m} \tag{3.25}$$

Le problème de quantification est simple pour la fonction *g* car elle ne présente pas les mêmes problèmes que la fonction *f*. Elle subira alors une quantification uniforme sur l'intervalle [0 π/2]. Soit *q*' le nombre de bits nécessaires pour choisir un sous-segment, *s*' la variable aléatoire qui permettra de choisir un des $2^{q'}$ sous-segments et $\Delta'=2^{-q'}$ le pas de quantification. La fonction *g*(*s*') est, alors, exprimée par l'équation (3.26) où *m*' et *δ*' ont la même signification que *m* et *δ*.

$$g(s') = \left\lfloor 2^{m'} \sqrt{2} \cos\left(\frac{\pi \Delta'(s'+\delta')}{2}\right) \right\rfloor \cdot 2^{-m'} \tag{3.26}$$

Il reste maintenant le problème du signe de la variable aléatoire à générer. Jusqu'à maintenant, nous avons quantifié les fonctions *f* et g sans s'occuper du signe. L'utilisation d'une variable auxilière aléatoire binaire uniforme (la loi normale est symétrique par rapport à l'axe des ordonnées) permet de résoudre ce problème. Si elle est égale à 0, la variable est positive, sinon elle est négative. Il suffit maintenant de générer les variables aléatoires uniformes $s_1, s_2 \ldots s_K$, *s*' et *r* et calculer le produit des fonctions *f* et *g* pour obtenir une variable aléatoire qui suit une loi normale centrée réduite.

3.5.2 Génération des variables aléatoires uniformes

Les variables aléatoires uniformes sont les plus simples à générer, car toutes les valeurs possibles possèdent la même probabilité d'apparition. La densité de probabilité de la loi uniforme continue est exprimée par le système d'équations (3.27).

$$f(x) = \begin{cases} \dfrac{1}{b-a} & \text{pour } a \le x \le b \\ 0 & \text{sinon} \end{cases} \tag{3.27}$$

Dans notre cas, nous nous intéresserons à la loi uniforme discrète, c'est-à-dire, que la variable possède un espace fini de possibilités (n valeurs possibles). Chacune des valeurs possibles possède alors une probabilité d'apparition de $1/n$.

Registre à décalages avec rétroaction linéaire

La manière la plus simple de générer une variable aléatoire uniforme est l'utilisation des registres à décalages à rétroaction linéaire ou en anglais LFSR (*Linear Feedback Shift Register*). Le registre à décalage est une série de bascules reliées entre elles de façon qu'à chaque coup d'horloge, tous les bits sont décalés. Le LFSR est un registre à décalage dont le bit d'entrée est une fonction linéaire de l'état précédant (Nguyen M-Q, 2005). Ce bit d'entrée est régit par des opérations XOR (ou exclusif) sur certains bits du mot stocké dans le registre (voir Figure 3.15). Au début, le registre à décalage possède un état initial (obligatoirement différent de 0) appelé en anglais *seed* et à chaque coup d'horloge, un bit est généré et est injecté à l'entrée du LFSR qui passe à un nouvel état. Si un registre est composé de n bascules (n bits), et si la fonction de rétroaction est bien choisie, on aura 2^n-1 états possible. Après avoir parcouru l'ensemble des états, le système revient à son état initial et répète son cycle.

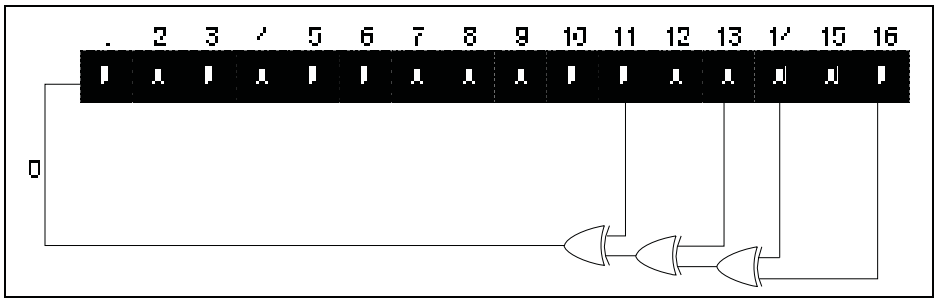

Figure 3.15 Exemple d'un LFSR de type Fibonacci.

Il y a deux types de LFSR, l'architecture Fibonacci (Figure 3.15) et Galois (Figure 3.16). Cette dernière intercale les portes XOR entre les bascules. Les deux architectures peuvent générer exactement les mêmes séquences de bits si les fonctions de rétroactions sont bien choisies. Dans toute la suite, nous allons nous intéresser au LFSR de type Fibonacci.

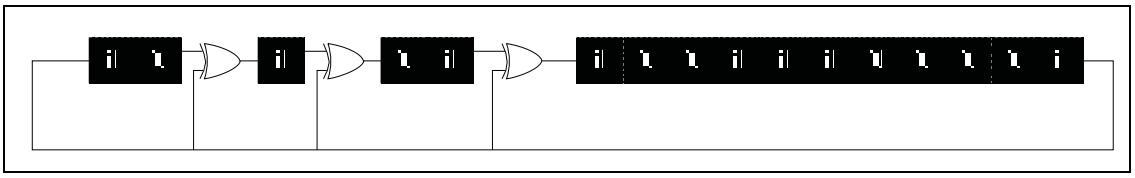

Figure 3.16 Exemple d'un LFSR de type Galois.

La position des bits qui affectent l'état du registre est appelée *taps*. La position la plus à droite représente le bit de sortie. On peut choisir n'importe quelle position des *taps* pour générer des séquences aléatoires, mais il se peut que les positions choisies ne génèrent pas tous les états possibles du registre et par conséquent la périodicité du système diminue. Un LFSR qui permet de générer les 2^n-1 états est appelé LFSR à longueur maximale (*maximum length LFSR*) et la séquence qu'il produit est appelée *m-sequence* (Nguyen M-Q, 2005).

La combinaison des *taps* peut être représentée par un polynôme modulo 2 dont les coefficients sont soit 1 soit 0. Pour l'exemple de la Figure 3.15, le polynôme représentant le LFSR est donné par l'équation (3.28). Les *maximum length LFSR* sont générés à partir de polynômes primitifs.

$$P(x) = x^{16} + x^{14} + x^{13} + x^{11} + 1 \tag{3.28}$$

L'utilisation des LFSR uniformes est la méthode la plus simple pour la génération des variables aléatoire vu la simplicité d'implémentation, mais ils ne possèdent pas de bonnes performances sauf dans le cas d'un générateur binaire. En effet, la sortie d'un LFSR est représentée par un seul bit, mais on peut prendre le mot qui existe dans le registre pour avoir des variables composées de n bits. À chaque coup d'horloge, le registre change d'état avec

l'injection du nouveau bit à son entrée. La différence entre deux états successifs est donc un seul bit et donc la corrélation entre les mots devient grande (Pong P.Chu et Robert E.Jones, 2000). Une solution pour ce problème est de mettre un compteur et à chaque n coups d'horloge, on extrait le mot qui est dans le registre, mais cette méthode s'avère lente vu que la fréquence de sortie des valeurs de la variable aléatoire est divisée par n.

Registre à décalages avec rétroaction linéaire à bond

Ce type de registre est appelé en anglais LP-LFSR (*Leap-Forward Linear Feedback Register*). C'est une méthode de génération de variables aléatoires uniformes à plusieurs bits. Elle est inspirée du fait que le LFSR est une fonction linéaire. Le passage du LFSR d'un état au prochain état peut s'exprimer sous forme d'un produit matriciel représenté par l'équation (3.29) où A est la matrice de transition (Pong P.Chu et Robert E.Jones, 2000) (de taille $n \times n$ avec n la longueur du registre) et les x_i sont les différents états du registre.

$$x_{k+1} = Ax_k \tag{3.29}$$

Pour un LFSR de type Fibonacci, et à partir de son polynôme générateur, on peut écrire le système (3.30) où b_i représentent les bits constituant le mot dans le registre, α_i sont les coefficients du polynôme du LFSR et la somme représente une somme modulo 2.

$$\begin{cases} b_{i+1} = b_i & \text{pour } i = 2...n \\ b_1 = \sum_{i=1}^{n} \alpha_i b_i \end{cases} \tag{3.30}$$

À partir du système (3.30), on peut alors construire la matrice de transition A. Le mot $x_{k+\mathrm{n}}$ est directement généré à partir de x_k par le biais de l'équation (3.31).

$$x_{k+n} = Ax_{k+n-1} = A^n x_k \tag{3.31}$$

Ainsi, on peut s'affranchir du problème des LFSR classiques en construisant simplement la matrice A^n et en l'implémentant avec des portes XOR. Cette solution consomme plus de ressources que les LFSR classiques, car elle utilise plus de portes XOR, mais sa bonne performance l'emporte sur le peu de ressources qu'elle utilise en plus d'un LFSR (Pong P.Chu et Robert E.Jones, 2000).

3.5.3 Architecture du générateur de bruit blanc gaussien

Architecture de base

Le générateur de variables aléatoires gaussiennes possède une architecture simple. Il est composé de générateurs de variables aléatoires uniformes qui, dans le cas d'une implémentation logicielle, subiront des transformations par les fonctions f et g. Dans notre choix d'implémentation, les fonctions f et g seront sous la forme de tables de correspondances et par conséquent, les variables uniformes serviront comme indexation dans ces tables. Enfin, les sorties des LUT des fonctions seront multipliées entre eux pour produire une variable aléatoire suivant une loi normale centrée réduite. Un facteur multiplicatif est placé à la fin pour contrôler la variance de la variable qui est par conséquent la puissance du bruit à la sortie.

Les paramètres q et K sont responsables de la taille de la mémoire utilisée pour la fonction f (longueur de la table). Le paramètre m est responsable quant à lui de la profondeur de la table de correspondance (le nombre de bits de chaque mot stockés). Il reste q' et m et ils sont responsable de la mémoire allouée à la fonction g. Une multitude de simulations ont été effectués par (Ghazel A *et al*, 2001) pour avoir la combinaison de paramètres qui donne la meilleure performance. Ils sont résumés dans le Tableau 3.4.

Tableau 3.4 Paramètres optimaux pour la génération des tables de la méthode Box-Muller

q	K	m	δ	q'	m'	δ'
4	5	7	0.36	8	6	0.5

Avec cette table de paramètres, nous avons construit l'architecture du générateur Box-Muller illustré par la Figure 3.17. La quantification de la fonction *f* se fait sur *K*=5 itérations, nous avons donc crée 5 tables de correspondances de 16 éléments chacune. La table de la fonction *g* comprend sur 128 entrées avec 1 bit de signe qui déterminera le signe de la valeur à la sortie du générateur. Les auteurs de (Ghazel A *et al*, 2001) proposent des LFSR classiques pour générer les variables aléatoires uniformes. Si on utilise un LFSR pour chacun des entrées des 6 tables de correspondances, les performances du système seront mauvaises, car ils ne présentent pas une grande périodicité ($2^4 - 1 = 15$) ce qui est très faible. Nous avons alors décidé de mettre un LP-LFSR de 28 bits (4 bit pour chacune des 5 tables de *f*, 7 bits pour la table de *g* et un bit de signe) qui donnera une meilleure performance vu ses caractéristiques de corrélation. Sa périodicité est alors de 2^{28} -1, et pour une fréquence de fonctionnement de 100 MHz, le système accomplit un cycle entier en 2,6844 secondes.

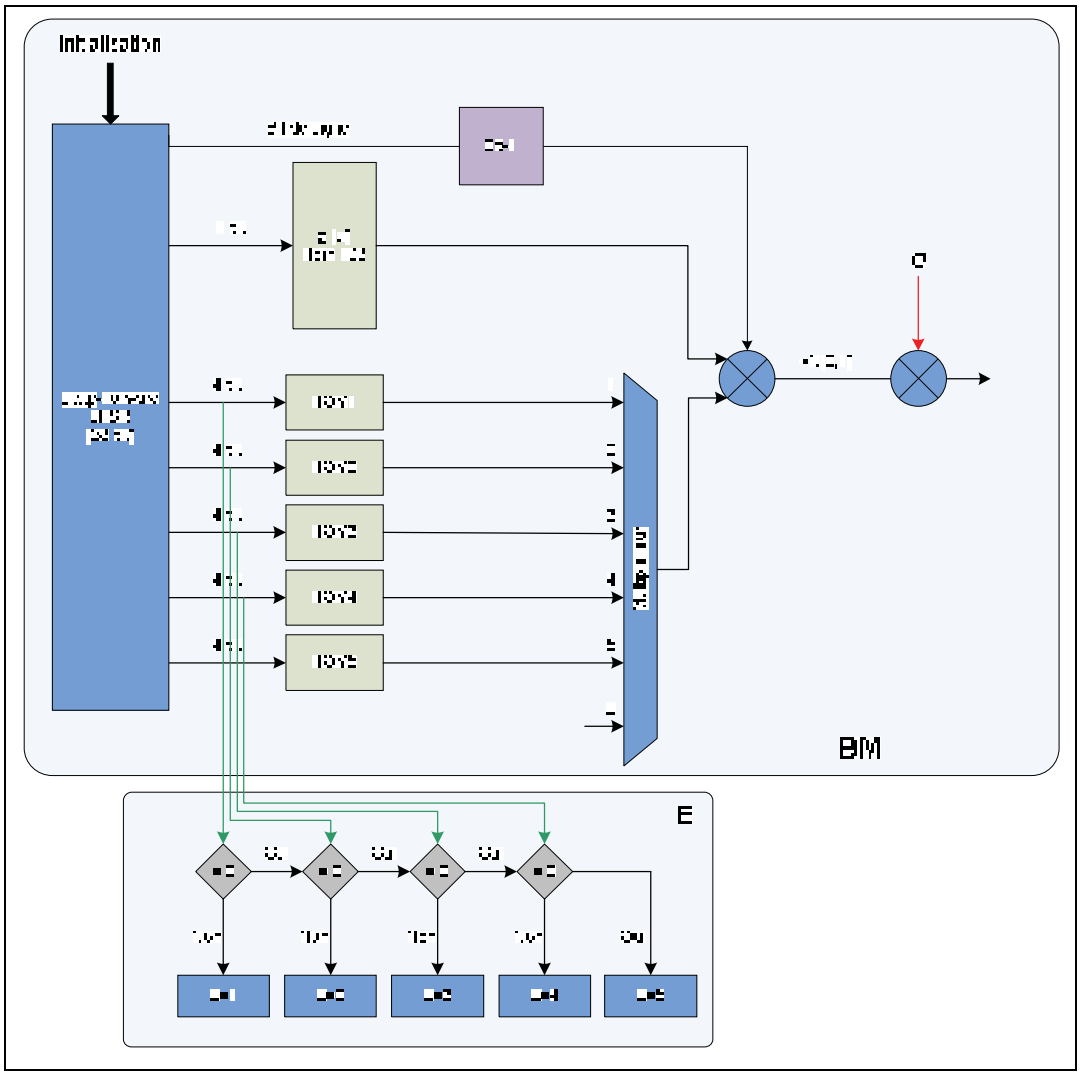

Figure 3.17 Architecture de base du générateur Box-Muller.

Augmenter la taille des tables et le nombre de bits permet d'avoir une meilleure performance, mais augmente aussi les ressources matérielles utilisées par le FPGA. En utilisant un FPGA de type Virtex IV, nous allons changer juste les paramètres *m* et *m*' car les primitives de mémoires implémentés possèdent une largeur de 18 bits. Nous allons alors utiliser *m* = 15 (avec 3 bits pour la partie entière) et *m* = 17 (1 bit pour la partie entière). Nous avons effectué des simulations pour observer l'erreur de la distribution de la variable aléatoire générée par rapport à la fonction de la loi normale centrée réduite. Nous avons remarqué que les performances de la méthode utilisée sont assez basse comparée à la forme de base théorique.

Vu que les performances réelles du système ne se manifestent qu'en effectuant une simulation avec un très grand nombre d'échantillon et faute d'insuffisance de mémoire, nous avons implémenté un modèle sous Simulink qui permet de voir le changement de la distribution au cours du temps. Nous avons alors simulé l'architecture proposée en utilisant un LP-LFSR de 28 bits (voir annexe IV pour la matrice de transition) pour cent millions d'échantillons et nous avons obtenus l'erreur de la distribution par rapport à la fonction de distribution la loi normale centrée réduite illustrée par la Figure 3.18.

Nous remarquons que l'erreur dépasse les 10% pour des valeurs inférieures à 4σ, il y a un décalage d'environ 5 à 6% par rapport à la version théorique de la méthode Box-Muller.

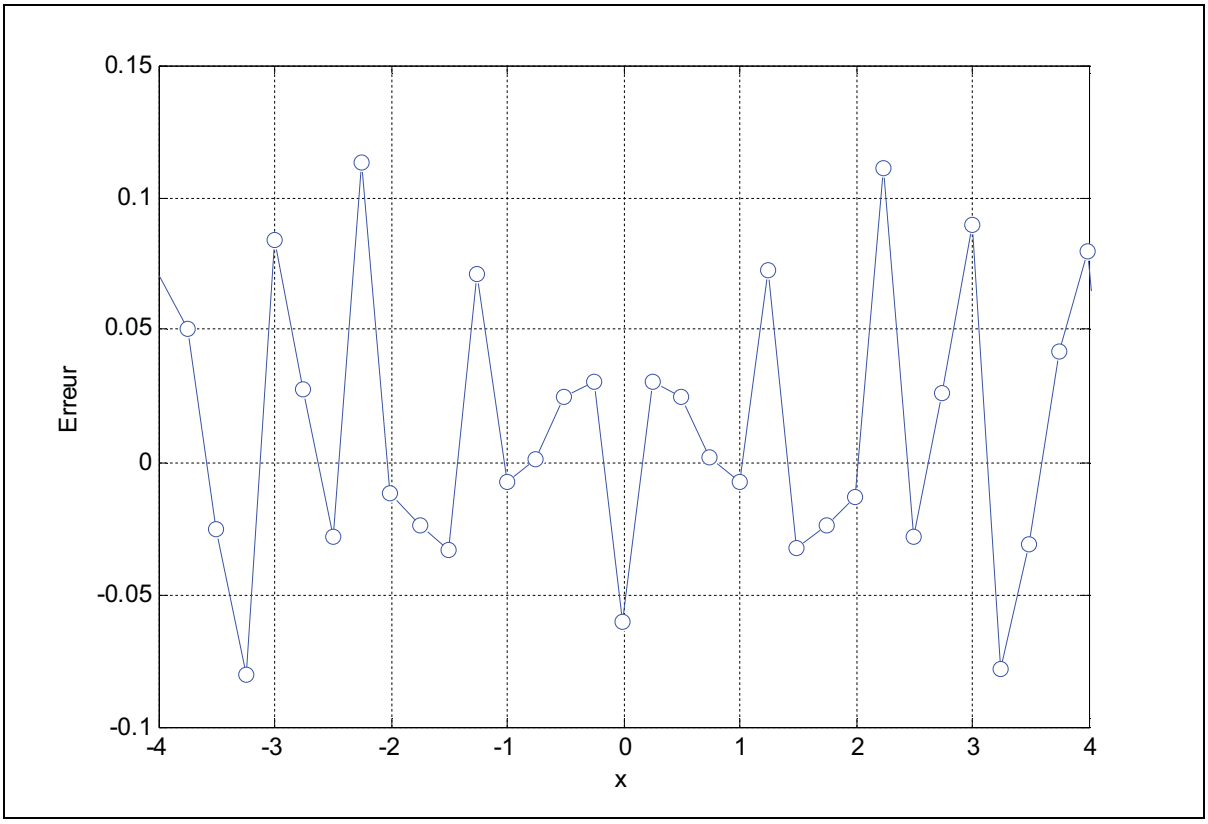

Figure 3.18 Erreur relative de la distribution de la v.a générée par l'architecture de base de la méthode Box-Muller sur FPGA.

Amélioration des performances

Une augmentation des paramètres q et q' n'a pas eue un grand effet sur la performance du système. Il faut les augmenter d'une façon considérable afin d'observer une amélioration (Danger J-L *et al*, 2000). Pour améliorer les performances de l'architecture proposée dans la section précédente, nous pouvons utiliser le théorème de la limite centrale. Si les variables aléatoires utilisées dans le théorème de la limite centrale sont gaussiennes, leur somme converge très rapidement vers une variable ayant une meilleure distribution. Si on duplique la même architecture plusieurs fois, nous obtiendrons plusieurs variables aléatoires gaussiennes indépendantes et par conséquent nous observerons une amélioration des performances. A la fin, pour revenir à la loi normale centrée réduite il faut multiplier la variable par un facteur $1/\sqrt{n}$ avec n étant le nombre de variables sommées. En numérique, une division par un multiple de 2 se résume à un décalage à gauche du mot (on enlève les bits les moins significatifs). Le choix optimal est alors $n = 4$. L'architecture du générateur Box-Muller de la Figure 3.17 est dupliquée 4 fois. On obtient alors l'architecture illustrée par la Figure 3.19 où le bloc BM représente l'architecture de base sans le facteur d'justement de la variance.

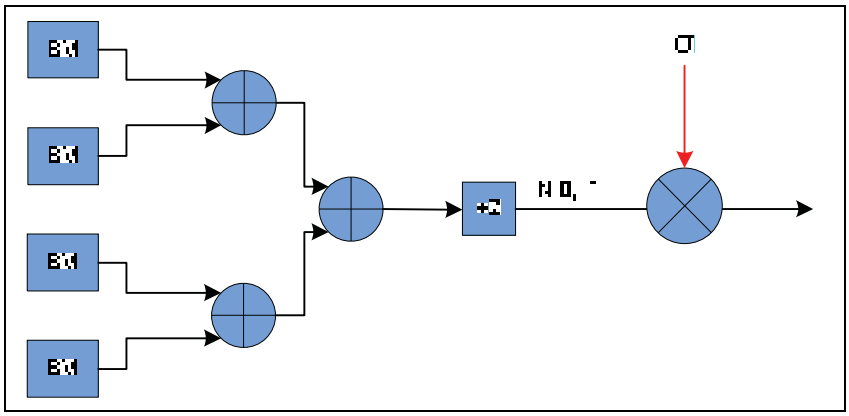

Figure 3.19 Architecture du générateur Box-Muller utilisant le théorème de la limite centrale.

Nous avons effectué une simulation de l'architecture présentée dans la Figure 3.19 pour cent millions de points pour observer l'effet de l'ajout des trois blocs identiques. Nous avons alors

obtenu une erreur très minime (voir Figure 3.20) ne dépassant pas les 1% pour $|x| < 4$. Nous obtenons donc une erreur dix fois moins grande que celle simulée pour un seul bloc du générateur Box-Muller de base.

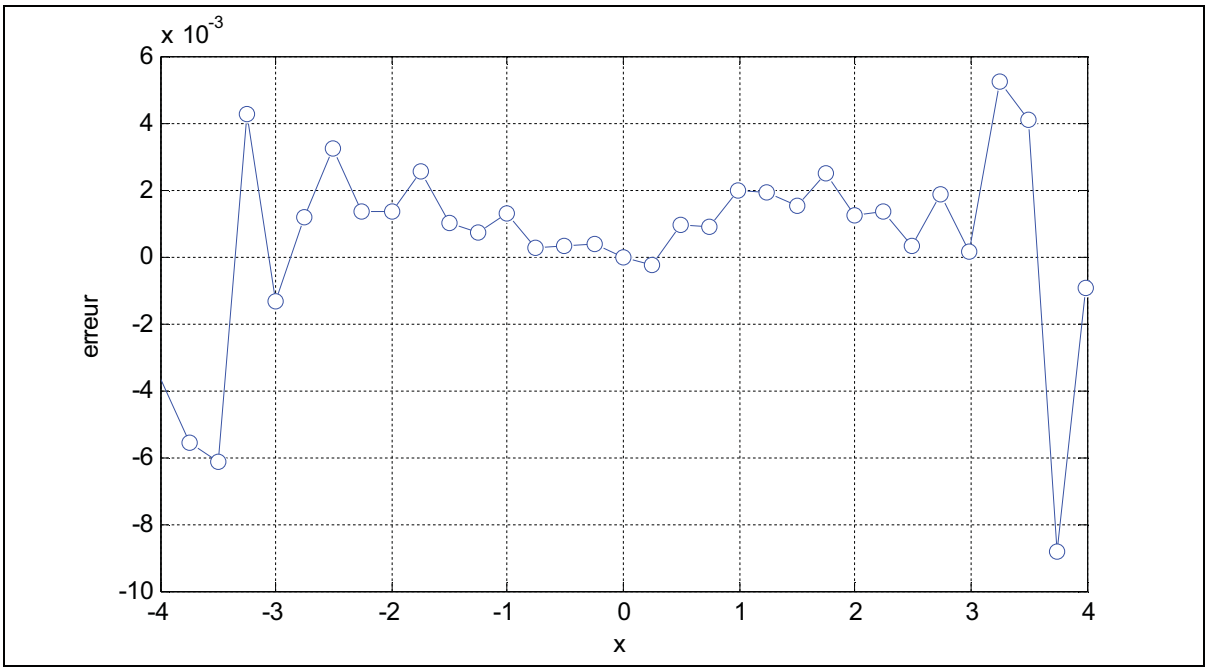

Figure 3.20 Erreur relative entre la distribution de la v.a générée par la combinaison de l'architecture de base de la méthode Box-Muller avec la méthode de la limite centrale et la fonction de distribution de la loi normale N(0,1).

3.5.4 Optimisation de l'architecture du générateur de bruit blanc gaussien

Une duplication de l'architecture de base consomme 4 fois plus de ressources que prévu lors du choix de la méthode Box-Muller. Nous avons alors décidé d'optimiser cette architecture. Le caractère aléatoire des 4 générateurs vient du fait que les variables aléatoires uniformes sont indépendantes, car les LP-LFSR utilisés possèdent chacun sa propre initialisation. Les LP-LFSR étant identiques, ils produisent alors les mêmes séquences, mais décalés dans le temps (initialisation différente). En théorie, si les valeurs d'une variable aléatoire sont complètement décorrélés, on peut alors utiliser un générateur de variable uniforme et faire la somme de 4 valeurs consécutives et par conséquent, on repasse de 4 modules Box-Muller à

un seul. Le module doit, alors, avoir une vitesse de 400 MHz. Dans la pratique, les séquences sont un peu corrélées vu le caractère déterministe de leur génération et sommer 4 valeurs consécutives peut biaiser les résultats. Nous travaillons avec une fréquence de 100 MHz sur le FPGA et via le gestionnaire numérique d'horloge ou en anglais le DCM (*Digital Clock Manager*) nous arrivons à dupliquer la fréquence jusqu'à 200 MHz. Nous avons alors décidé de garder uniquement deux blocs du générateur Box-Muller et faire la somme de deux valeurs successives sur chaque bloc et faire une dernière somme pour les valeurs sortantes de chacun d'eux. Sur le FPGA, nous disposons aussi de primitives de mémoires à double accès qui ne sont pas utilisées par la partie IF du simulateur. Ces blocs mémoires permettent l'accès à la même mémoire via deux entrées différentes et ainsi elle permet de générer deux valeurs distinctes au même coup d'horloge. Ainsi, nous avons changé l'architecture totale du générateur Box-Muller.

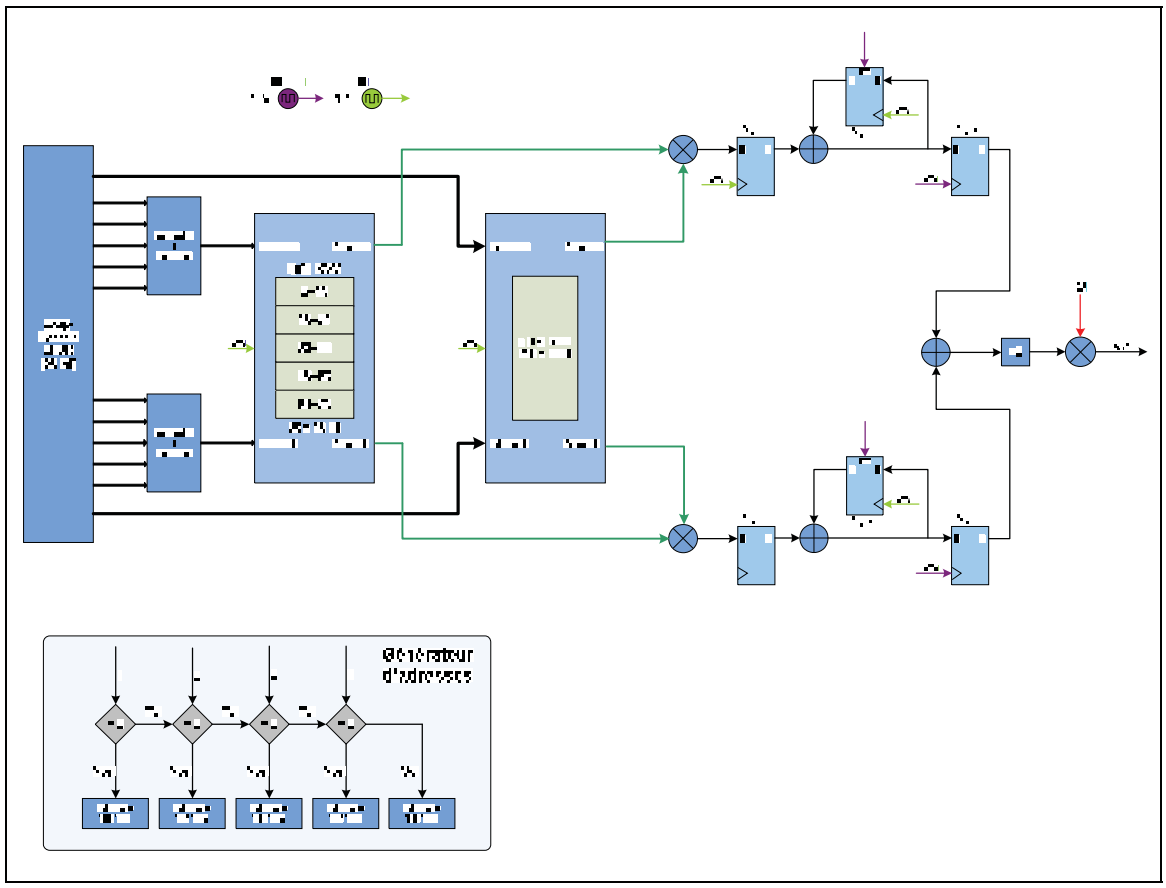

Figure 3.21 Architecture finale du générateur Box-Muller.

L'architecture illustrée par la Figure 3.21 est l'architecture finale du générateur Box-Muller, elle a subi certaines transformations. La table de la fonction *g* a été modifiée de 128 à 256 éléments vu que les primitives sur le FPGA le permettaient. Au lieu d'utiliser un bit de signe et un module de transformation de signe (conversion d'un nombre non signé positif à un nombre signé négatif avec la représentation du complément à 2), nous avons intégré les valeurs négatives du cosinus dans la table de correspondance sans perdre aucune ressource matérielle. L'adressage passe de 7 bits à 8 bits en intégrant le bit de signe dans les 7 premiers. Utiliser 5 tables de 16 éléments chacune s'est avéré une perte de ressources, nous avons alors combiné toutes les tables en une seule et nous avons ajouté un module d'adressage qui permet de convertir les 5 variables aléatoires uniformes en une adresse de la table. En ce qui concerne le LFSR, nous pouvions juste le dupliquer et mettre une initialisation différente pour chacun des blocs. La périodicité d'un LP-LFSR de 28 bits étant petite par rapport au temps des simulations effectuées sur le simulateur, nous avons décidé de passer à 56 bit. Cette solution est très satisfaisante parce que d'un coté elle augmente la périodicité du système à 2^{56} -1 valeurs possibles et d'un autre coté elle diminue la corrélation entre les valeurs successives à la sortie du LP-LFSR (le nombre d'états a augmenté considérablement). La périodicité du système passe de 2,6844 secondes à 11 ans, ce qui est très large et permet d'avoir à chaque simulation un bruit totalement différent du précédent.

Le générateur Box-Muller modifié peut être divisé en deux parties, la première fonctionne à une horloge de 200 MHz (représenté par des flèches CLK2 en vert sur la Figure 3.21) et la deuxième fonctionne à 100 MHz (représenté par des flèches CLK2 en mauve sur la Figure 3.21). La deuxième partie permet de correspondre la fréquence de sortie de la variable aléatoire gaussienne avec la fréquence des échantillons des signaux IF. Ces derniers sont, alors sommés avec le bruit et sont transmis aux CNA pour la conversion numérique analogique.

Afin de vérifier les performances de la nouvelle architecture, nous avons effectué une simulation pour encore cent millions de points et nous avons observé l'erreur relative de la distribution de la variable aléatoire générée. La Figure 3.22 illustre cette erreur et en la

comparant avec les résultats précédents nous remarquons que les performances sont pratiquement les mêmes, c'est-à-dire une erreur inférieure à 1%.

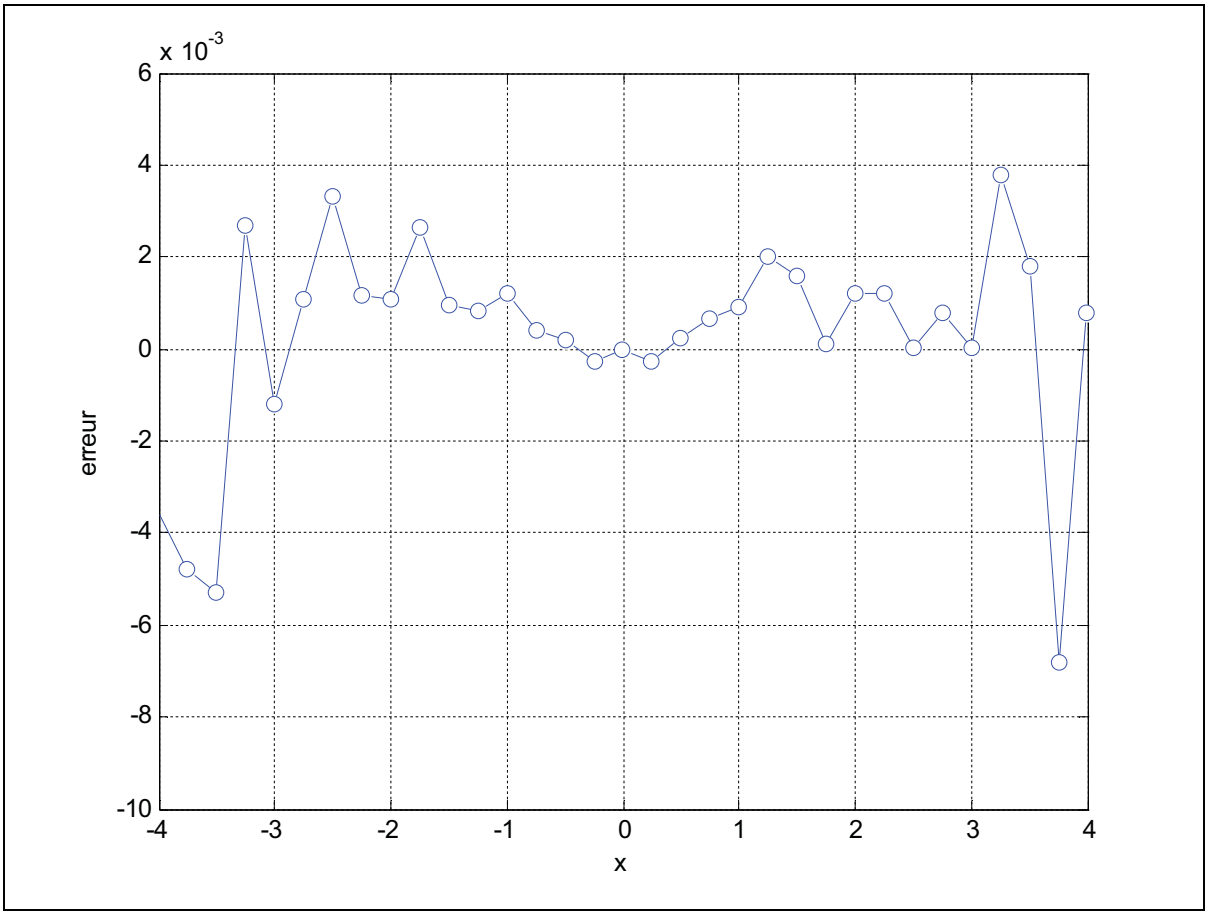

Figure 3.22 Erreur relative entre la distribution de la v.a générée par l'architecture optimisée de la méthode Box-Muller et la fonction de distribution de la loi normale N(0,1).

3.6 Validation du système de contrôle du ratio C/N_0

3.6.1 Le spectre du bruit généré

Afin de valider le bruit généré, nous avons observé son spectre à la sortie du CNA avec l'analyseur de signaux vectoriels FSQ40 de Rhode and Schwarz. La Figure 3.23 illustre le spectre du bruit sur une bande de 100 MHz. Ayant une fréquence d'échantillonnage de

100 MHz, le résultat est normal vu que le CNA introduit une atténuation en sin(x)/x. La bande de fréquence d'observation est de 100 MHz parce que le signal fondamental à la sortie de la partie IF est centré à 30 MHz et nous utilisons le signal de la première harmonique centré à 70 MHz.

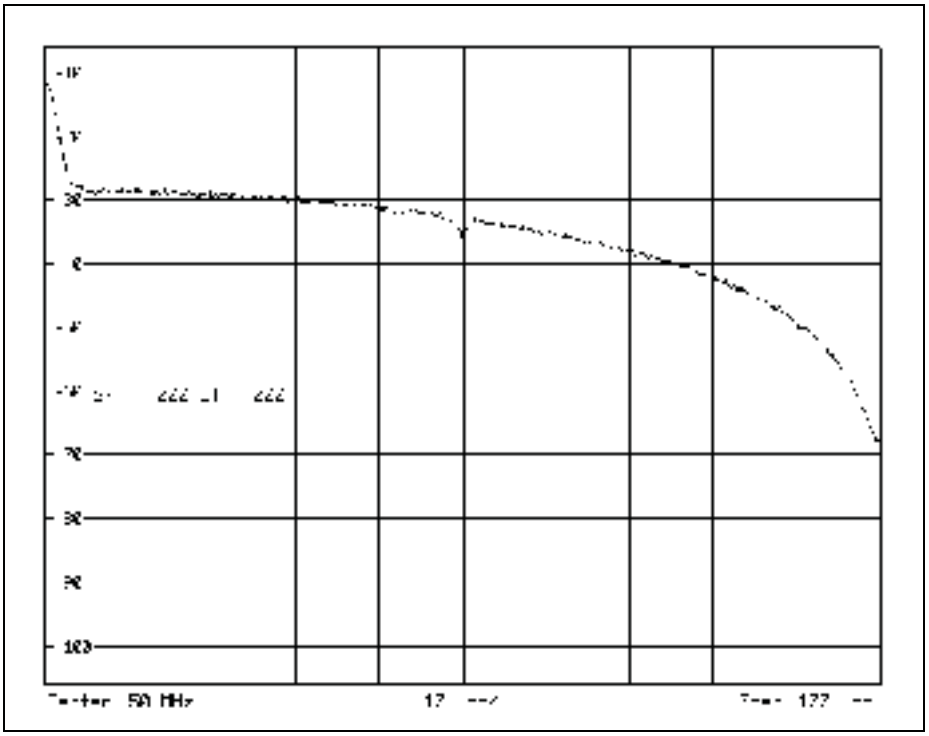

Figure 3.23 Spectre du bruit sur une largeur de 100 MHz.

Le bruit généré est utilisé sur une bande de 20 MHz autour d'une fréquence centrale de 70 MHz. La Figure 3.24 montre le spectre du bruit sur la bande en question. Nous remarquons que le spectre n'est pas constant sur toute la bande de fréquence. Ceci est du à l'utilisation de la deuxième harmonique du signal issu du CNA. En effet, le CNA utilisé n'est pas capable de fournir les signaux à 70 MHz en respectant les spécifications. C'est pour cette raison que les signaux sont à la fréquence fondamentale de 30 MHz et que nous utilisons la deuxième harmonique pour les injecter à la partie RF. La Figure 3.25 illustre le spectre du bruit sur la bande fondamentale. Il est clair qu'il est nettement plus plat que sur la bande des 70 MHz. Le spectre est cependant un peu décroissant (de l'ordre de 0.5 dB) ce qui est aussi expliqué par l'atténuation en sin(x)/x générée par le CNA.

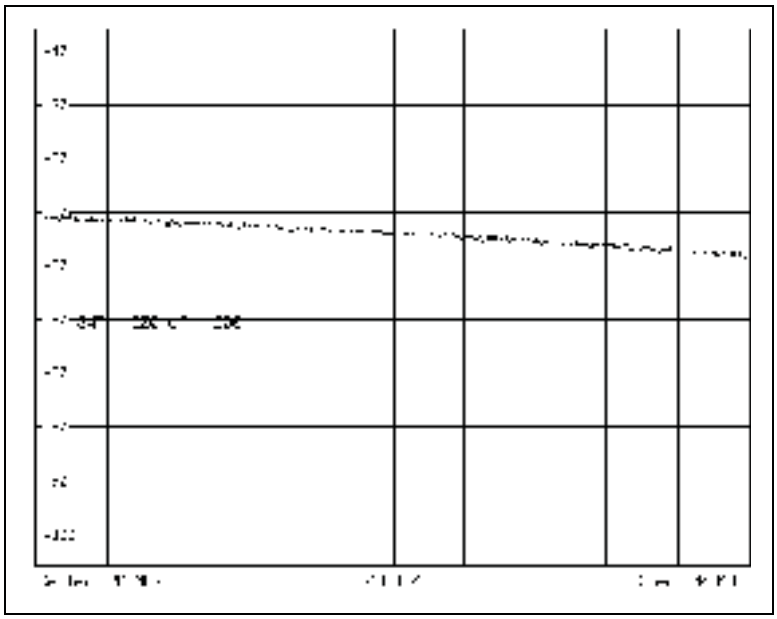

Figure 3.24 Spectre du bruit sur une bande allant de 60 à 80 MHz.

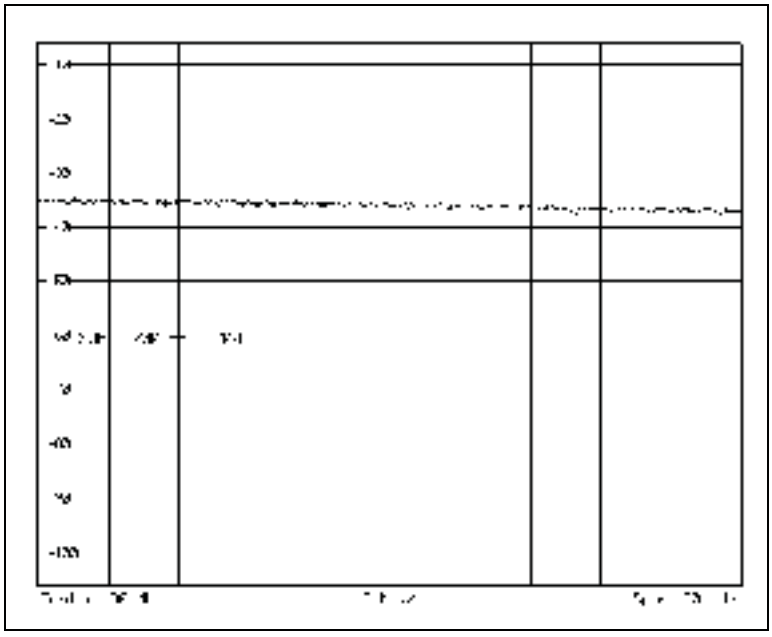

Figure 3.25 Spectre du bruit sur une bande allant de 20 à 30 MHz.

3.6.2 Plage de contrôle de la densité du bruit

Les convertisseurs numérique-analogique utilisés dans le simulateur GPS et Galileo ont une résolution de 14 bits. Pour que le bruit soit suffisamment précis, nous avons besoin d'un minimum de 8 bits. Nous avons alors appliqué un gain sur les valeurs de bruits pour faire varier son niveau de puissance. Nous obtenons, à la sortie du CNA, une densité de puissance minimale de -163 dBm/Hz et une plage de contrôle de 36 dB (l'équivalent de 6 bits). Nous obtenons, à la sortie de la carte RF un niveau maximal de -143 dBm/Hz.

3.6.3 Validation du contrôle de bruit avec les récepteurs

Afin de valider le générateur de bruit blanc avec le simulateur de constellation GPS et Galileo, nous avons effectué deux tests avec deux récepteurs différents. Le premier est un récepteur commercial de la compagnie CMC Electronics et le deuxième est un récepteur développé au sein du laboratoire LACIME.

À la sortie de la carte RF, le signal subit une atténuation de 40 dB pour atteindre le niveau nominal à l'entrée des récepteurs. Les niveaux de la densité du bruit calculé dans la section précédente seront faibles et n'atteindront pas la densité de puissance du bruit thermique à la température ambiante. Nous avons alors décidé d'atténuer le signal GPS en IF de 32 dB. De cette manière, nous aurons juste une atténuation RF de 8 dB et une densité de bruit qui varie entre -190 dBm/Hz et -156 dBm/Hz (en incluant les 3 dB d'atténuation causé par le combineur) et par conséquent une plage de 20 dB de contrôle de la puissance du bruit à partir de la densité de bruit thermique de l'ordre de -176 dBm/Hz.

Le test se déroule en faisant varier la densité du bruit par le biais de la partie logicielle sur une plage de 20 dB (de -176 à -156 dBm/Hz) et d'observer le comportement des deux récepteurs face à ce changement.

La Figure 3.26 et la Figure 3.27 illustrent, respectivement, les réponses du récepteur commercial et celui de l'ÉTS face à l'augmentation de la densité de puissance du bruit

généré. Les deux courbes possèdent la même allure. Elles ne sont pas parfaitement linéaires. La première portion (en dessous de -170 dBm/Hz) a une pente moins raide que le reste des valeurs qui présentent une réponse acceptable. Nous pouvons observer l'erreur de la pente des deux récepteurs, successivement, dans la Figure 3.28 et la Figure 3.29.

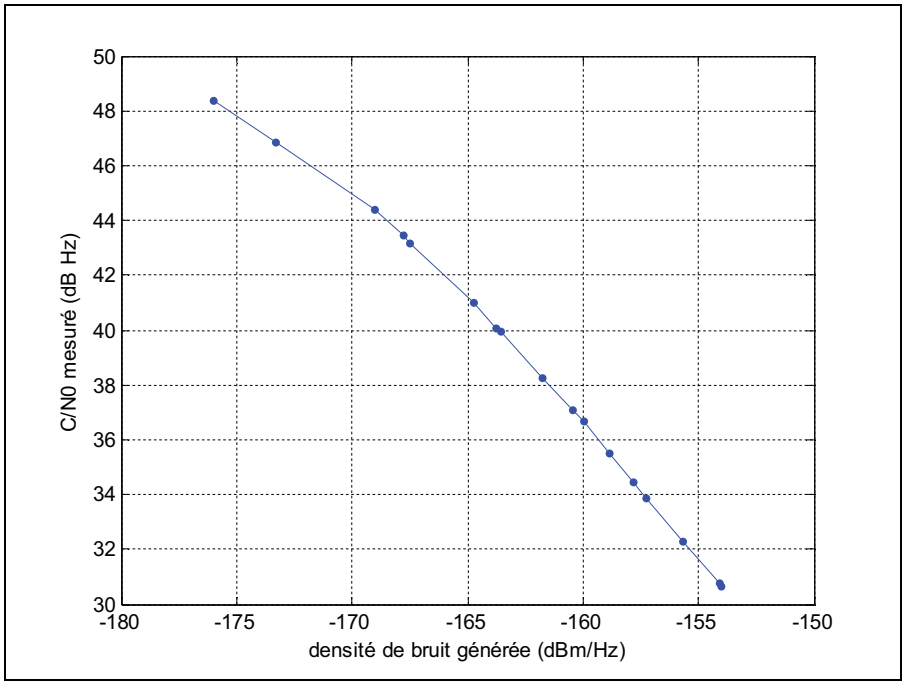

Figure 3.26 C/N_0 mesuré à l'aide du récepteur commercial.

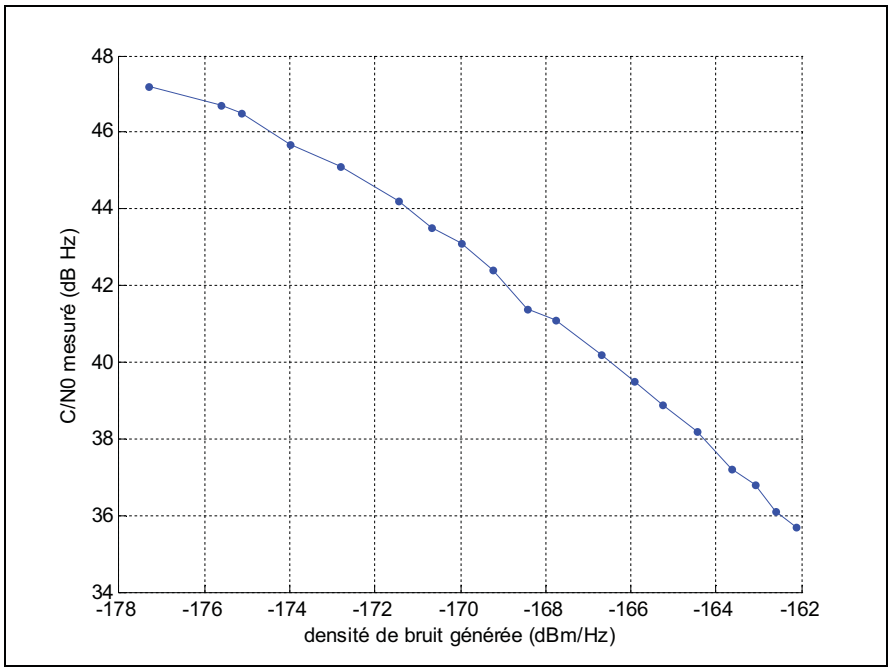

Figure 3.27 C/N_0 mesuré à l'aide du récepteur de l'ÉTS.

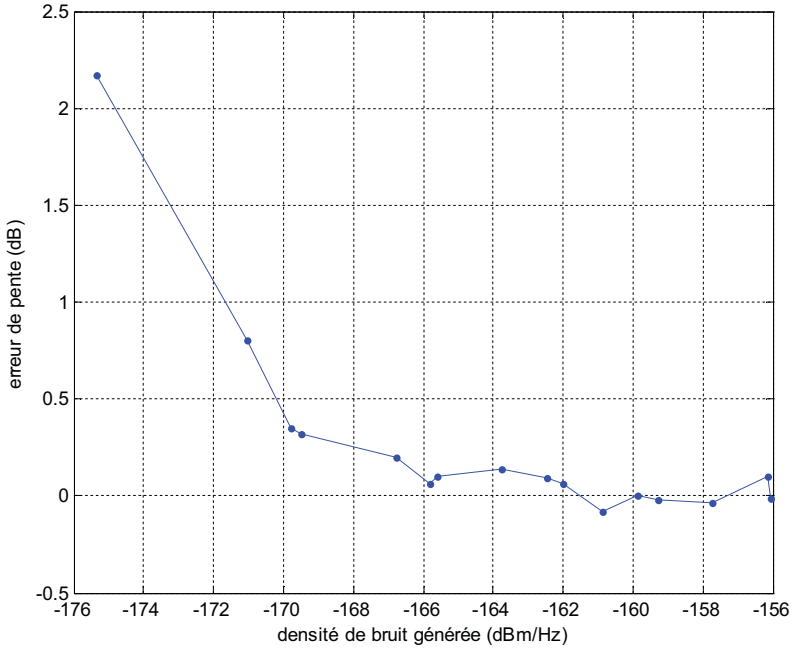

Figure 3.28 Erreur de la pente du récepteur commercial.

Nous remarquons que l'erreur est grande dans la première portion et elle n'est pas la même dans les deux récepteurs. La puissance du signal étant assez forte par rapport au bruit, les récepteurs ont du mal à calculer la valeur exacte du bruit et ceci engendre une grande erreur de mesure. Dans la deuxième portion, l'erreur est inférieure à 0.5 dB ce qui est acceptable.

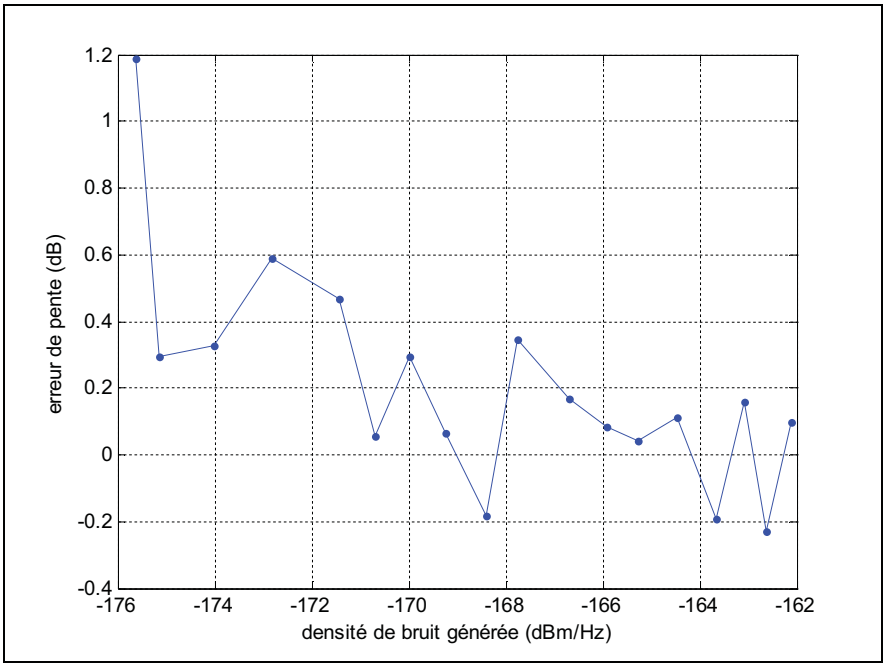

Figure 3.29 Erreur de la pente du récepteur de l'ÉTS.

Nous pouvons remarquer que la variation de la densité du bruit n'est pas la même dans les tests des deux récepteurs (voir l'axe des abscisses de la Figure 3.26 et la Figure 3.27). Nous avons noté que le récepteur de l'ÉTS ne peut plus suivre les satellites quand le niveau de bruit arrive à -162 dBm/Hz (ellipse rouge dans la Figure 3.32 qui indique qu'il n'y a pas de solution) alors que le récepteur commercial continue sa poursuite pour la perdre autour de -156 dBm/Hz (ellipse rouge dans Figure 3.30 qui indique la dégénérescence de la position trouvée).

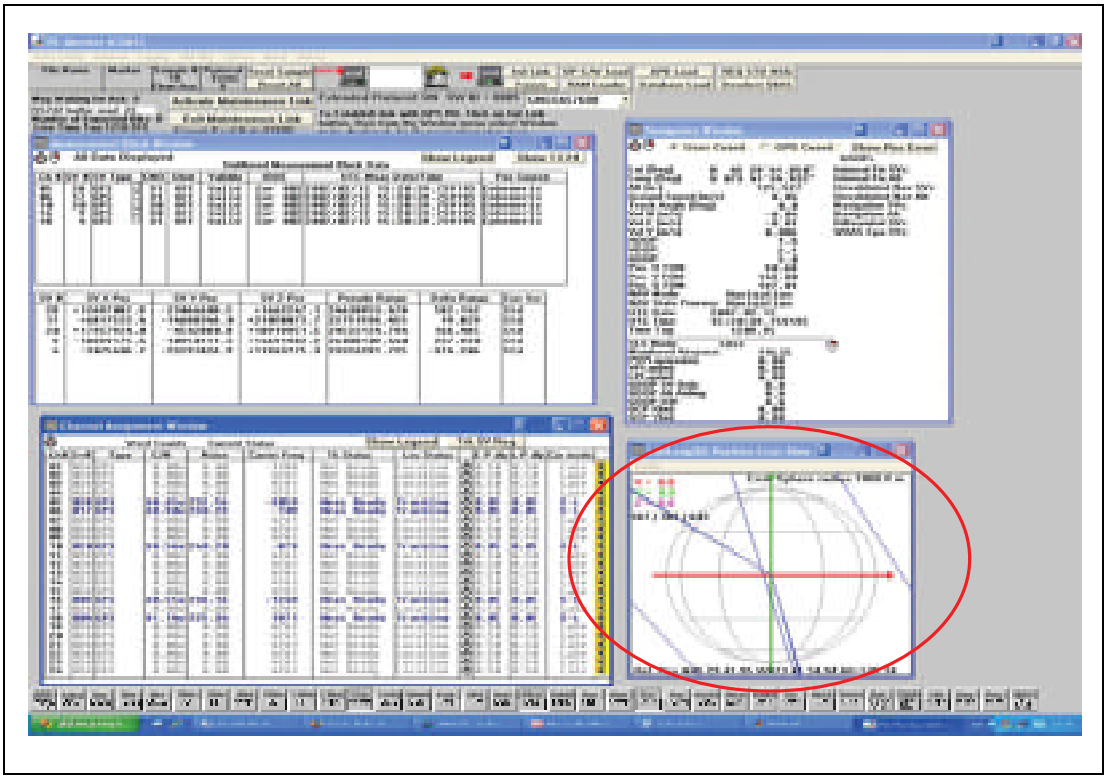

Figure 3.30 Décrochage du récepteur commercial.

Nous pouvons remarquer aussi que le pic de corrélation dans la Figure 3.32 n'est pas net en le comparant avec celui de la Figure 3.31.

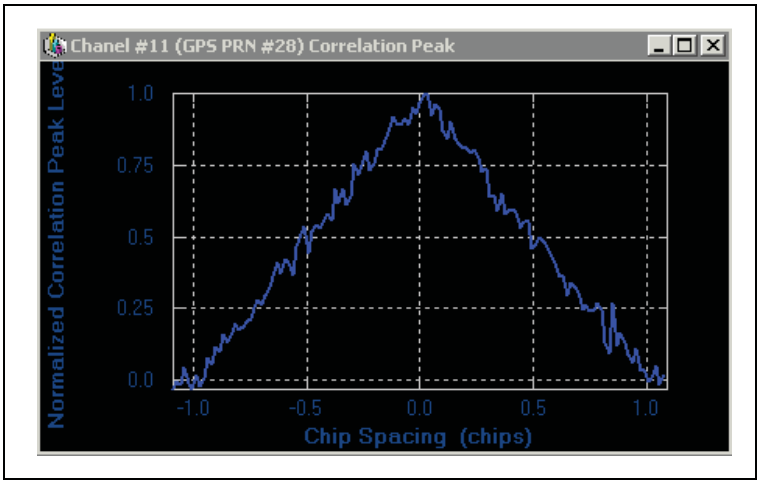

Figure 3.31 Pic de corrélation dans le récepteur de l'ÉTS sans injection de bruit.

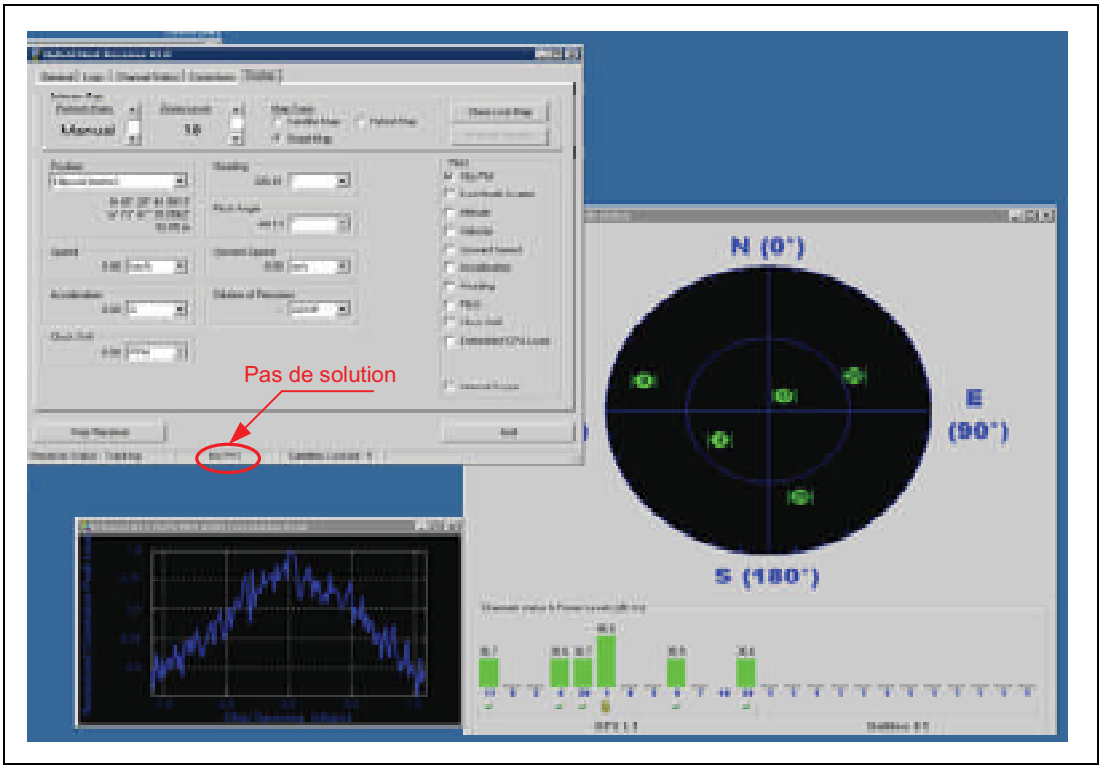

Figure 3.32 Décrochage du récepteur de l'ÉTS en présence de bruit.

3.7 Conclusion

Ce chapitre a traité les étapes de conception du système de contrôle du ratio C/N_0 qui intègre le système de contrôle de puissance. Nous avons commencé par des notions sur le bruit blanc pour passer ensuite aux possibilités de le générer. Nous avons alors, décidé de l'implémenter numériquement dans le FPGA au lieu d'une approche analogique dans la partie RF. Cette dernière possède certains inconvénients qui rendent son implémentation complexe. La méthode choisie offre par contre une flexibilité et une complexité moindre. Une étude sur les différents algorithmes de génération de bruit blanc gaussien a été établie pour élire la méthode Box-Muller qui était la plus adaptée à nos besoins. Le générateur Box-Muller a été alors implémenté et optimisé pour minimiser la consommation des ressources matérielles au sein du FPGA. Ce dernier volet est terminé par une validation du spectre fréquentiel du générateur Box-Muller modifié ainsi que la validation à l'aide d'un récepteur commercial ainsi qu'un récepteur développé au sein de l'ÉTS.

CONCLUSION

Ce mémoire a traité beaucoup d'aspects reliés au simulateur mais reste insuffisant pour décrire les détails d'un tel projet. Le simulateur est encore en phase de développement et il reste maintenant à développer la simulation des signaux L5 pour GPS et tous les signaux de la constellation Galileo.

Le premier chapitre a fait l'objet d'un survol de notions sur les systèmes de positionnement par satellites GPS et Galileo ainsi qu'une description des différentes parties du simulateur de constellation. Nous nous sommes focalisés sur la partie RF du simulateur afin d'étudier son architecture et son fonctionnement.

La deuxième partie de mémoire traite la mise en œuvre du système de contrôle de puissance piloté par la partie logicielle qui permet d'ajuster les niveaux de puissances des signaux GPS et Galileo, à la sortie du simulateur. Nous avons alors commencé par présenter l'architecture du système ainsi que les possibilités de contrôle qu'il peut offrir à l'utilisateur. Quatre modes de fonctionnement ont été, alors, développés à savoir le mode « calibration de puissance», « contrôle automatique de puissance », « contrôle de gain » et le mode « contrôle de puissance dans le temps ». Ce système consiste en un design électronique implémenté au niveau de la carte RF et en communication continuelle avec la partie logicielle qui le commande. Cette communication était absente dans la première version du logiciel du simulateur et a été incluse dans la nouvelle architecture pour permettre non juste une communication avec le système de contrôle de puissance seulement, mais un éventuel contrôle de la carte RF. Ce deuxième chapitre a été clôturé par la validation du système de contrôle de puissance qui fournit une précision inférieure à 0.1 dB sur une plage de 40 dB.

Le troisième et dernier chapitre de ce mémoire a traité la partie complémentaire du système de contrôle de puissance pour permettre un contrôle du ratio C/N_0. En effet, la variation de ce ratio dépend de deux paramètres, le niveau d'un signal et celui de la densité du bruit. Le contrôle du niveau du signal a été traité dans le deuxième volet de ce mémoire. Le contrôle

de la puissance du bruit s'est effectué par la suite. Nous avons alors commencé par étudier les différentes façons d'implémenter un bruit blanc. Entre une implémention analogique complexe dans la partie RF du simulateur et une implémentation numérique flexible dans le FPGA, notre choix s'est concentré sur la deuxième. Nous avons alors, fait une étude comparative entre les algorithmes d'implémentation actuels d'un bruit blanc gaussien en s'appuyant sur la complexité des calculs et de l'implémentation dans le FPGA. La méthode Box-Muller a été retenue pour son efficacité prouvée et sa simplicité d'implémentation par rapport aux autres algorithmes étudiés. Le générateur de Box-Muller a été développé et optimisé pour une consommation minime des ressources du FPGA. Il a été testé et validé à l'aide d'un récepteur commercial et un récepteur développé au sein de l'école.

Le développement du simulateur GPS et Galileo est un projet complexe et d'envergure. Il requiert beaucoup d'expertise dans plusieurs domaines de recherche à savoir les télécommunications et la géomatique. La nouvelle architecture logicielle lui permet d'être très flexible et extensible pour intégrer de nouvelles constellations et de nouvelles options.

ANNEXE I

Détails sur les composants du système de contrôle de puissance

I.1 Multiplexeur : ADG904

L'ADG904 présente des pertes d'insertions qui varient en fonction de la fréquence. La figure 1.1 illustre les pertes d'insertions du multiplexeur.

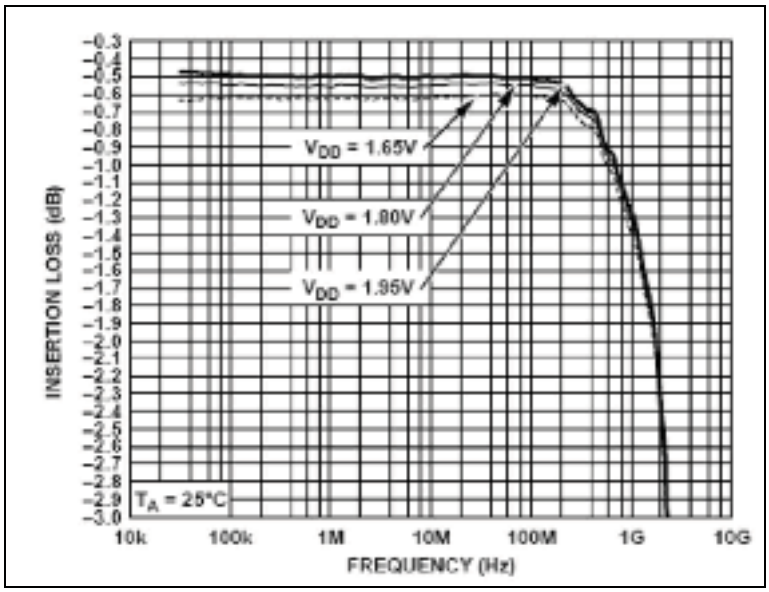

Figure 1.1 Perte d'insertion en fonction de la fréquence pour le multiplexeur ADG904.
Tiré de Analog Devices (2007a, p.7)

Ce multiplexeur présente aussi un chevauchement entre les signaux lors du basculement d'un signal à un autre (figure 1.2).

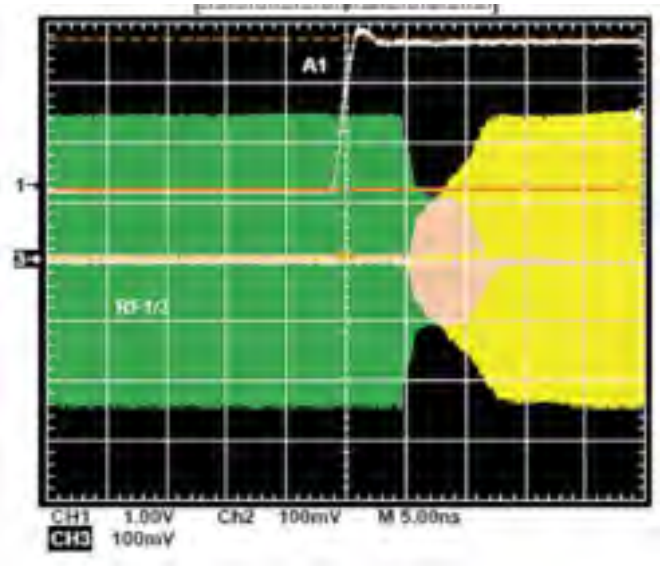

Figure 1.2 Chevauchement des signaux dans le multiplexeur.
Tiré de Analog Devices (2007a, p.8)

Comme chaque multiplexeur, deux bits sont disponibles pour permettre le basculement entre les signaux. Le tableau 1.1 contient les différentes combinaisons de basculement.

Tableau 1.1 Combinaison de basculement en fonction de A0 et A1
Tiré de Analog Devices (2007a ,p.4)

	A0	A1
RF1	0	0
RF2	0	1
RF3	1	0
RF4	1	1

I.2 Le détecteur de puissance AD8318

La figure 1.3 représente le schéma bloc fonctionnel de l'AD8318

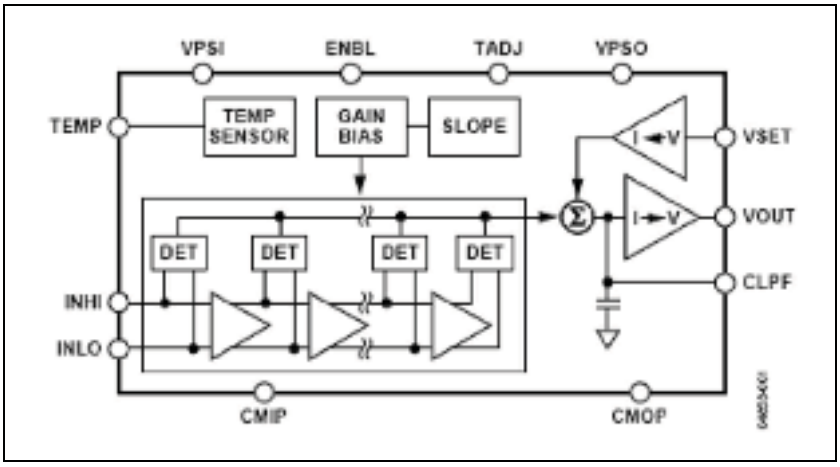

Figure 1.3 Schéma bloc de l'AD8318.
Tiré de Analog Devices (2007b, p.1)

Ce circuit présente aussi une dégradation des performances lors d'un changement de la température (figure 1.4). Mais l'AD8318 contient un dispositif

interne qui lui permet de compenser ce décalage, en effet la broche TADJ, en la branchant avec une résistance de 500Ω permet de compenser l'effet de la température (tableau 1.2).

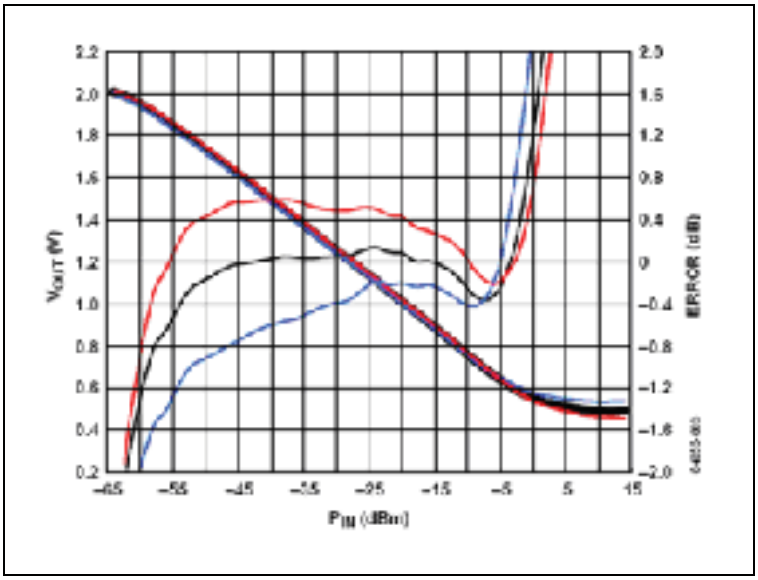

Figure 1.4 Réponse de l'AD8318 et l'erreur en fonction de la température à 1900MHz.
Tiré de Analog Devices (2007b, p.8)

Tableau 1.2 les résistances recommandé par Analog Devices pour la compensation en température
Tiré de Analog Devices (2007b, p.14)

Frequency	Recommended R_{TADJ}
900 MHz	500 Ω
1.9 MHz	500 Ω
2.2 GHz	500 Ω
3.6 GHz	51 Ω
5.8 GHz	1 kΩ
8 GHz	500 Ω

Pour pouvoir utiliser l'AD8318 dans un circuit électronique, le constructeur donne un modèle de connexion illustré par la figure 1.5.

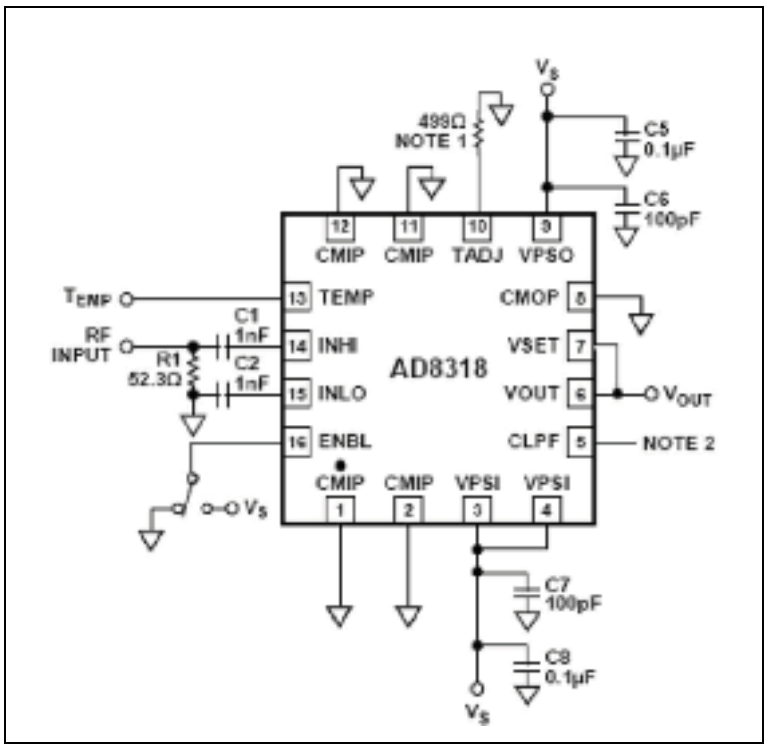

Figure 1.5 Les connexions basiques pour l'AD8318.
Tiré de Analog Devices (2007b, p.12)

I.3 le convertisseur analogique numérique

L'ADS8380 doit être commandé de l'extérieur pour le contrôle des phases d'échantillonnage, de conversion et de lecture des données. Les signaux $\overline{CS}$ et $\overline{CONVST}$ (figure 1.6) sont les signaux qui commandent les étapes d'échantillonnage et de conversion de l'ADC.

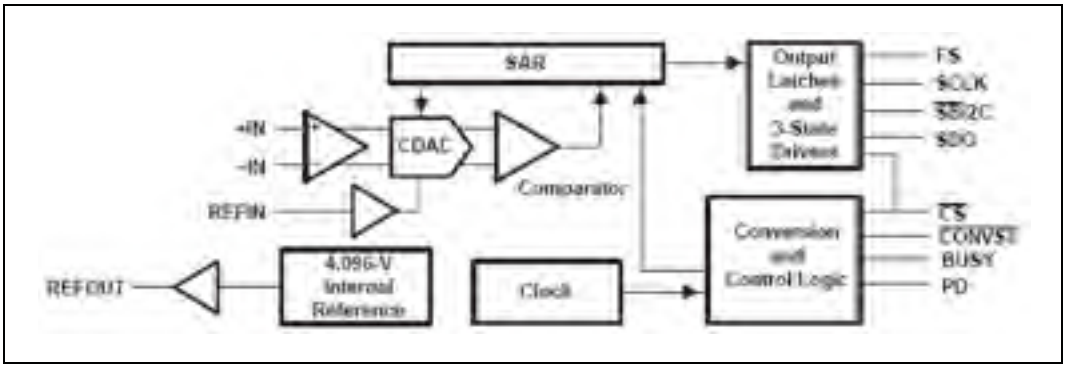

Figure 1.6 Schéma bloc de l'ADS8380.
Tiré de Texas Instrument (2004, p.1)

Texas instrument prévoit plusieurs modes de passages entre l'échantillonnage et la conversion (figure 1.7) :

- Back to back conversion : Passage direct de la conversion à l'échantillonnage ;
- Conversion avec attente : le système attend la commande pour commencer les échantillons ;
- Conversion avec NAP : Cet état est utilisé pour l'économie d'énergie.

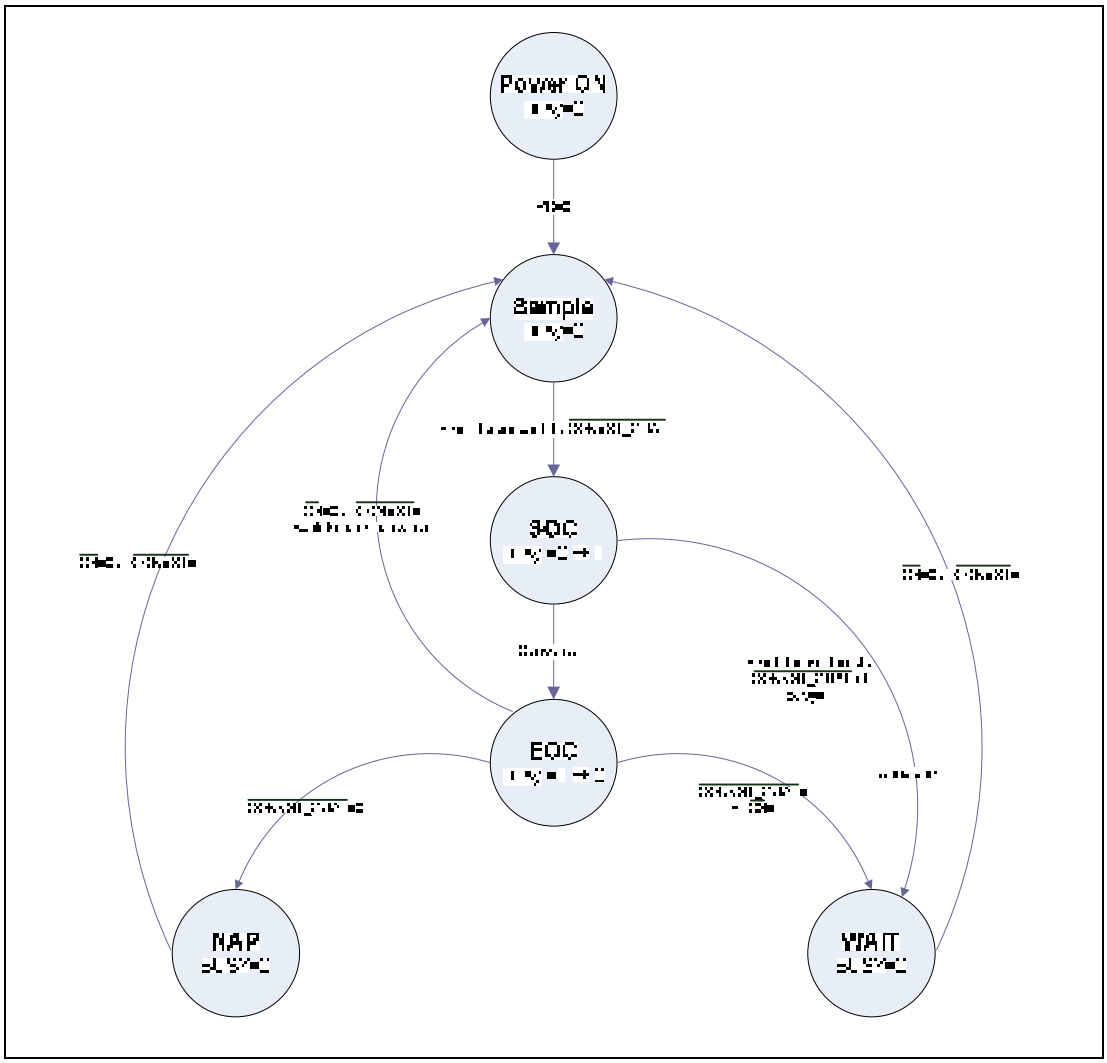

Figure 1.7 Machine à état du fonctionnement de l'ADC.
Tiré de Texas Instrument (2004, p.15)

La figure 1.9 illustre la forme d'onde dans un processus *Back-to-Back conversion*.

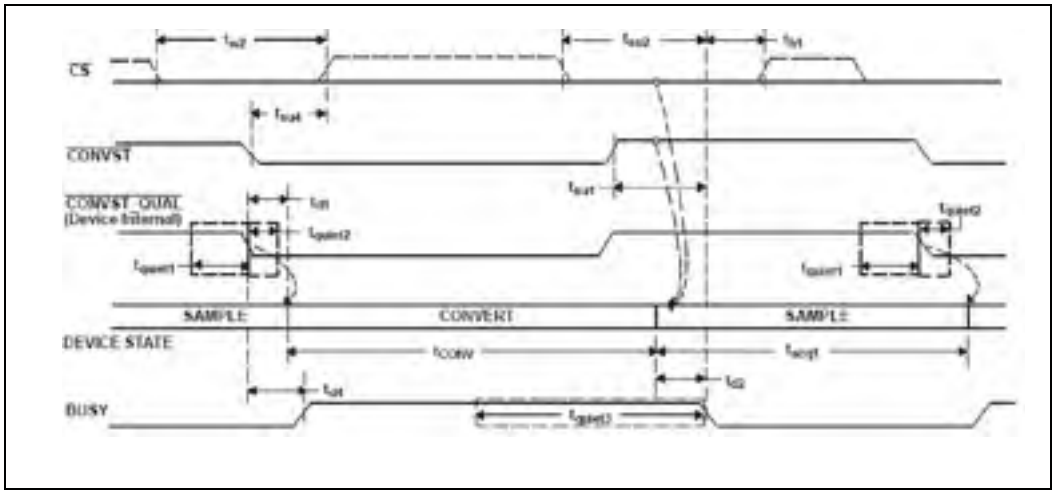

Figure 1.9 Forme des signaux pour le mode Back to back conversion.
Tiré de Texas Instrument (2004, p.16)

La figure 1.10 illustre la forme d'onde dans un processus *Convert And Sample with NAP*.

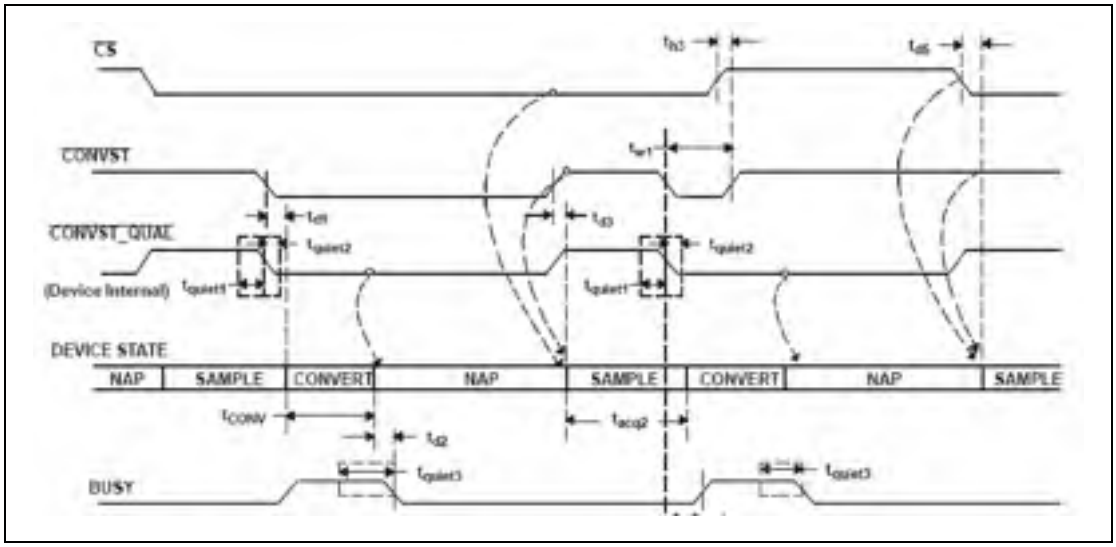

Figure 1.10 processus Convert And Sample with NAP.
Tiré de Tiré de Texas Instrument (2004, p.17)

La figure 1.11 illustre la forme d'onde dans un processus *Convert And Sample with WAIT*.

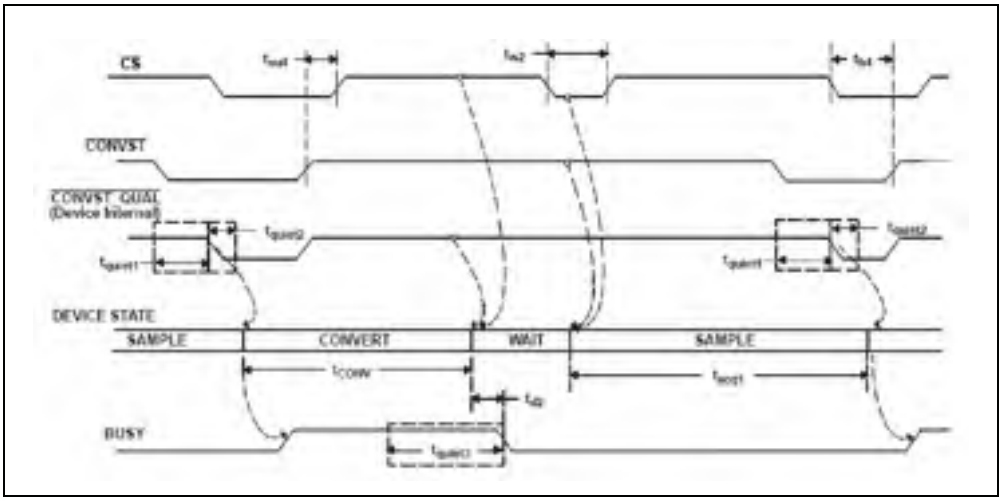

Figure 1.11 processus Convert And Sample with WAIT.
Tiré de Texas Instrument (2004, p.17)

Pour ce qui est de la lecture des données, Texas Instrument propose aussi plusieurs modes de lectures :

- Lecture via $\overline{CS}$ avec FS=1 (figure 1.12) ;
- Lecture via FS (figure 1.13 et 1.14).

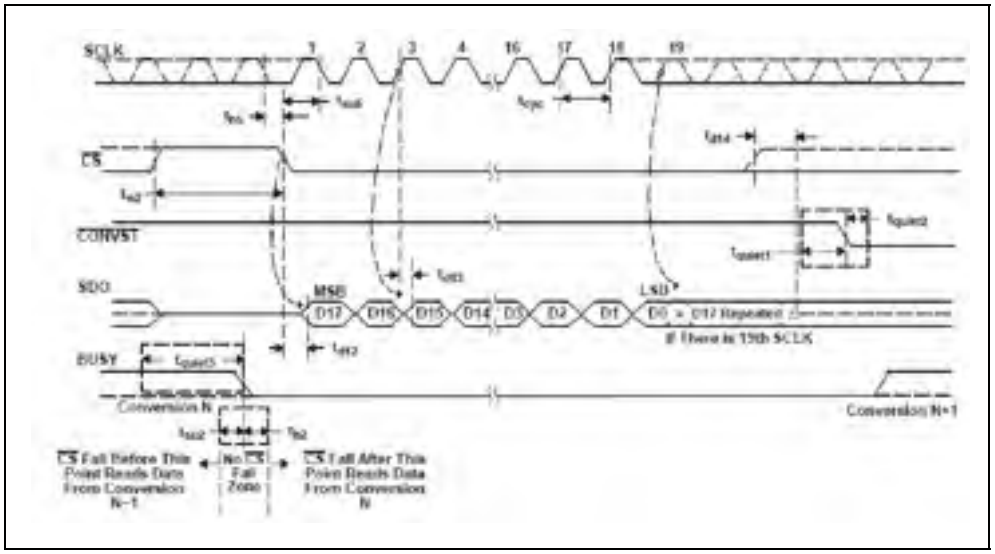

Figure 1.12 Lecture via $\overline{CS}$ avec FS=1.
Tiré de Texas Instrument (2004, p.19)

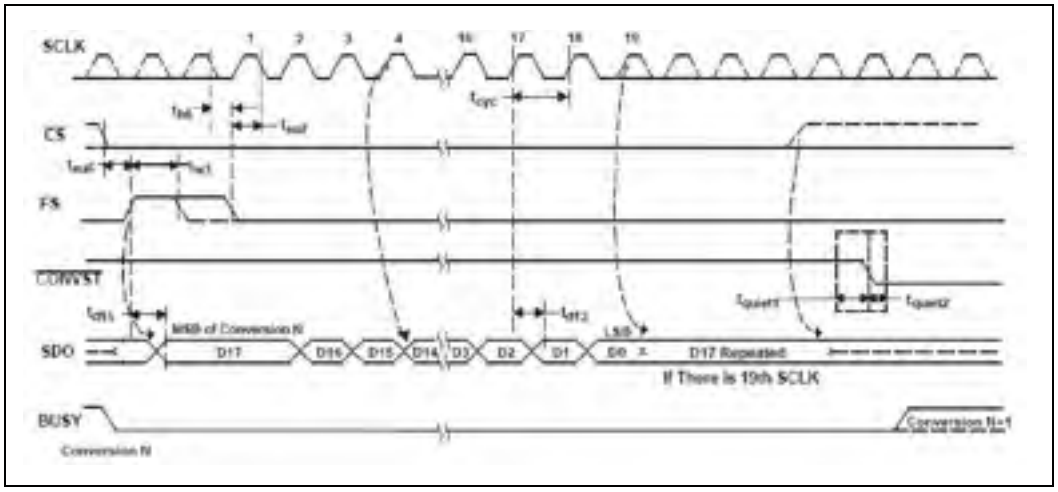

Figure 1.12 Lecture via FS (FS est à l'état bas lors du front descendant de *BUSY*).
Tiré de Texas Instrument (2004, p.19)

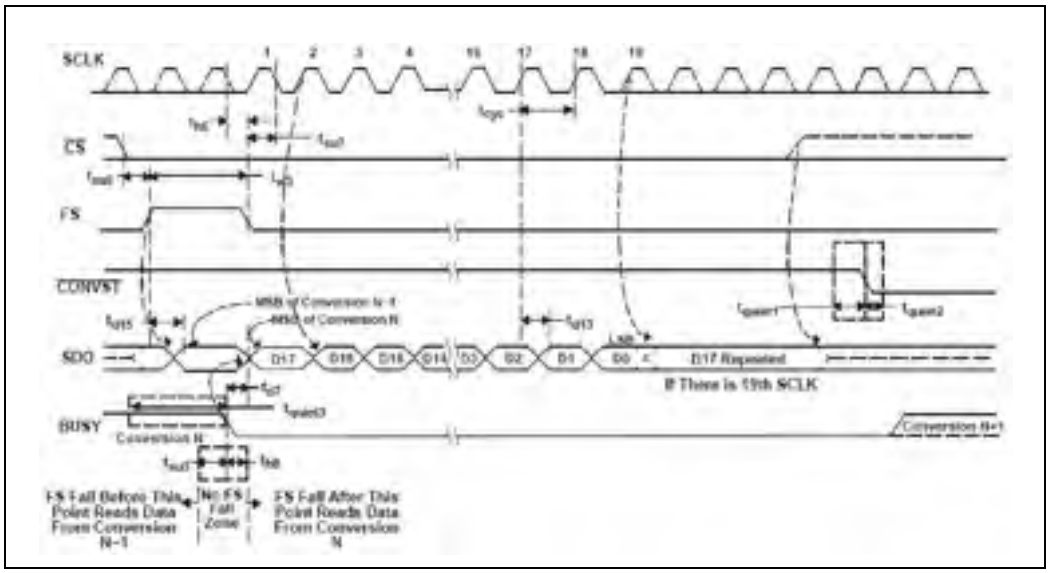

Figure 1.13 Lecture via FS (FS est à l'état haut lors du front descendant de *BUSY*).
Tiré de Texas Instrument (2004, p.20)

I.4 Le convertisseur analogique numérique

L'AD5668 est un DAC qui possède 8 sorties analogiques. Afin de les commander une procédure précisée par le constructeur doit être suivie pour assurer le fonctionnement de ce DAC. L'AD6558 reçoit des informations série sur sa broche DIN. Ces données sont sous la forme de mots de 32 bits qui lui précisent les taches qu'il doit effectuer. La structure de la trame de 32 bits est représentée par la figure 1.14.

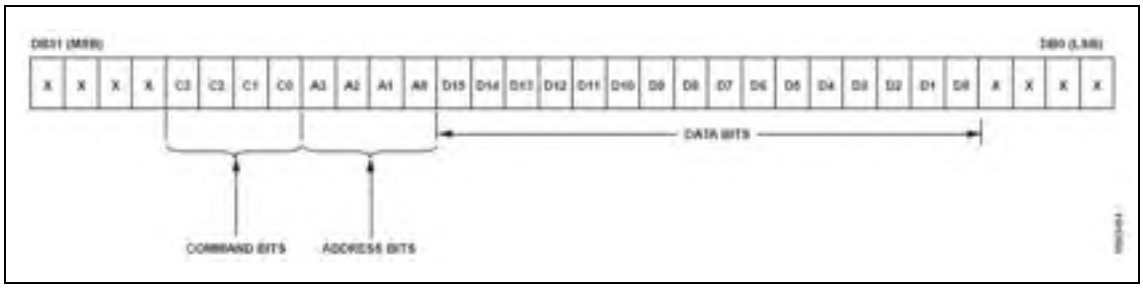

Figure 1.14 Trame de commande du DAC.
Tiré de Analog Devices (2005a, p.23)

Le tableau 1.3 illustre les combinaisons nécessaires pour les 4 bits de commande du DAC.

Tableau 1.3 combinaisons des bits de commandes
Tiré de Analog Devices (2005a ,p.22)

Command				
C3	C2	C1	C0	Description
0	0	0	0	Write to Input Register n
0	0	0	1	Update DAC Register n
0	0	1	0	Write to Input Register n, update all (software LDAC)
0	0	1	1	Write to and update DAC Channel n
0	1	0	0	Power down/power up DAC
0	1	0	1	Load clear code register
0	1	1	0	Load $\overline{\text{LDAC}}$ register
0	1	1	1	Reset (power-on reset)
1	0	0	0	Set up internal REF register
1	0	0	1	Reserved
–	–	–	–	Reserved
1	1	1	1	Reserved

Le tableau 1.4 illustre les combinaisons des 4 bits d'adresses pour sélectionner les sorties du DAC

Tableau 1.4 Combinaison des bits d'adresses
Tiré de Analog Devices (2005a, p.22)

Address (n)				
A3	A2	A1	A0	Selected DAC Channel
0	0	0	0	DAC A
0	0	0	1	DAC B
0	0	1	0	DAC C
0	0	1	1	DAC D
0	1	0	0	DAC E
0	1	0	1	DAC F
0	1	1	0	DAC G
0	1	1	1	DAC H
1	1	1	1	All DACs

La figure 1.15 illustre la forme des signaux pour l'écriture de la trame de commande.

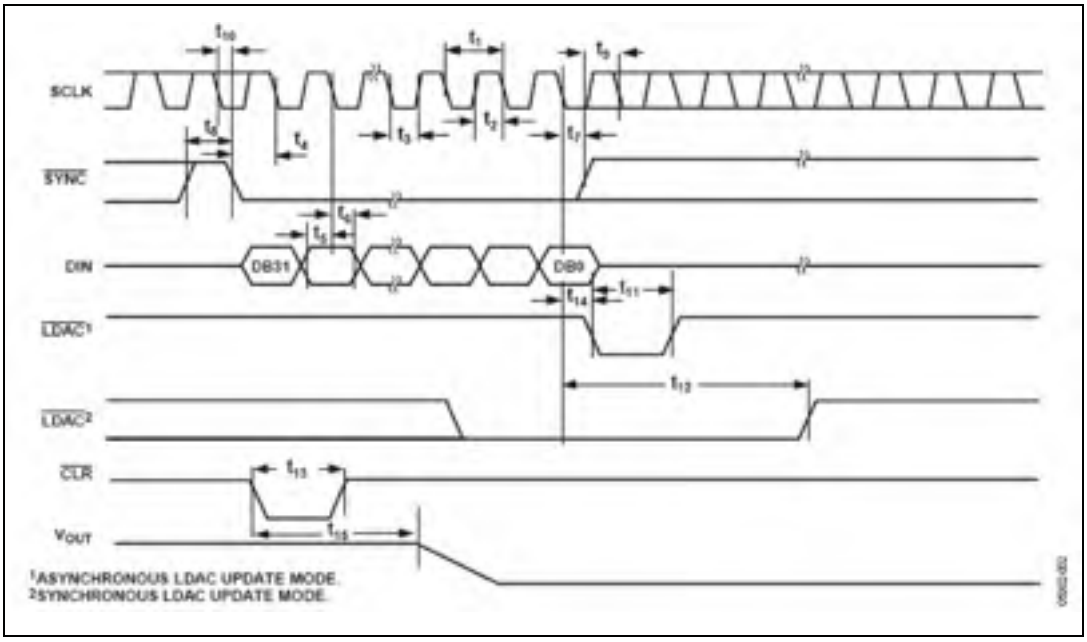

Figure 1.15 Opération d'écriture de la trame de commande pour l'AD5668.
Tiré de Analog Devices (2005a ,p.8)

ANNEXE II

Les ADC de type SAR

Pour les applications qui ne requièrent pas un grand débit symbole, les ADC de type SAR (*Successive Approximation Register*) présentent le choix parfait, ils présentent une faible consommation et sont généralement dans des packages de petites tailles. Ces ADC implémente un algorithme de recherche binaire (binary search algorithm). Même si leurs horloges internes fonctionnent à plusieurs MHz, le débit symbole est restreint. L'architecture d'un ADC de type SAR est représenté par la figure 2.1.

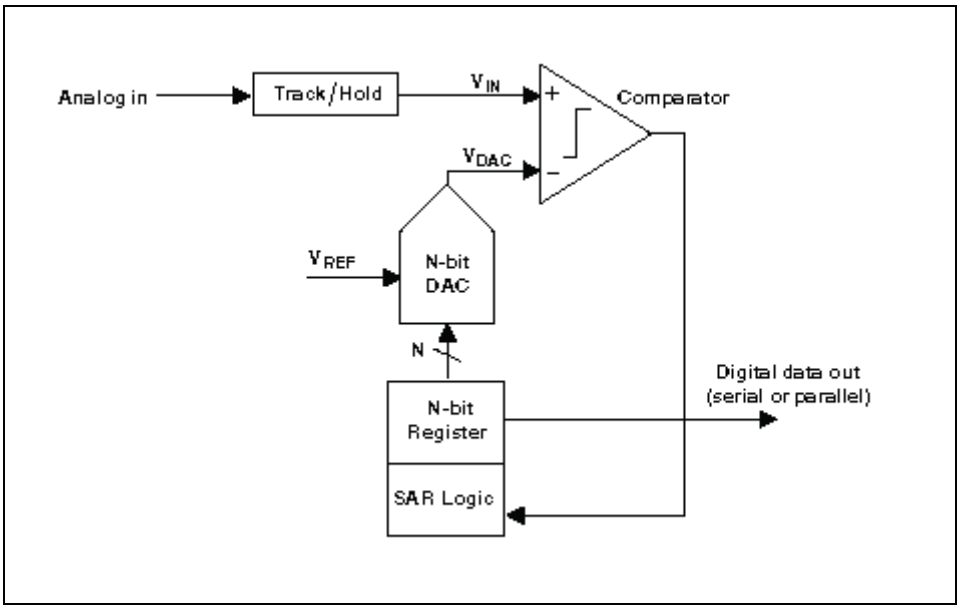

Figure 2.1 Architecture des ADC de type SAR.
Tiré de Maxim IC (2001, p.2)

La tension Vin est bloquée dans le « track/hold ». Pour implémenter l'algorithme de recherche binaire, le registre N bit (où N est la résolution de l'ADC) est initialisé à la moyenne échelle (midscale), c'est-à-dire que le MSB est à 1 et tous les autres bits sont à 0. De cette façon, la sortie du DAC « V_{DAC} » à être Vref/2, où Vref est la tension de référence de l'ADC. Une comparaison est alors effectuée pour déterminer si Vin est supérieur ou inférieur à V_{DAC}. Si Vin est supérieur à V_{DAC}, la sortie du comparateur est égale à 1 poussant

le MSB à 1, et si Vin est inférieur à V_{DAC}, Le MSB est forcé à 0. L'ADC passe alors au deuxième bit en effectuant une deuxième comparaison. Ainsi, le mot binaire est constitué dans le registre et envoyé à la sortie du DAC. La figure 2.2 traite le cas d'une conversion avec un ADC 4 bits de type SAR.

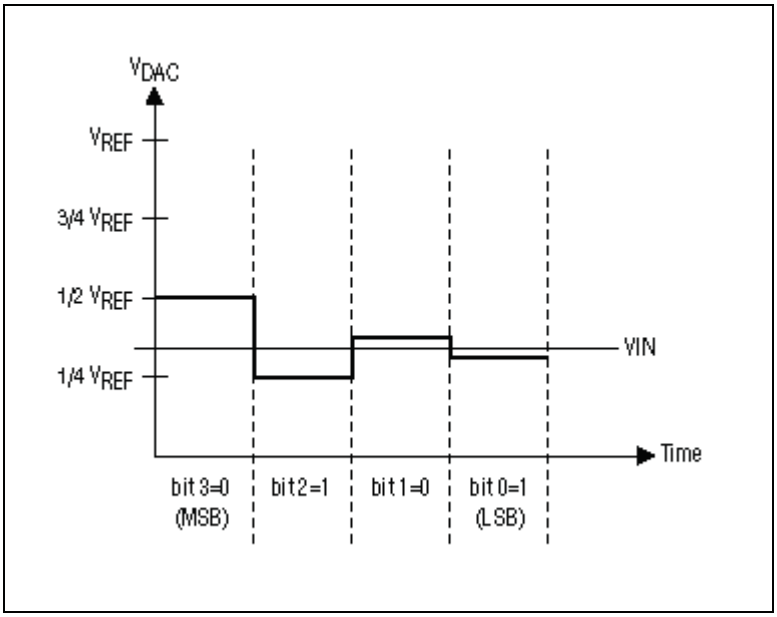

Figure 2.2 : Conversion 4bit avec un ADC de type SAR.
Tiré de Maxim IC (2001, p.2)

ANNEXE III

Boucle à verrouillage de phase

« Une Boucle à verrouillage de phase (PLL : Phase Locked Loop) est un montage électronique permettant d'asservir la phase instantanée de sortie sur la phase instantanée d'entrée, mais elle permet aussi d'asservir une fréquence de sortie sur un multiple de la fréquence d'entrée » (wikipedia, 2009b).

La figure 3.1 présente la structure d'une PLL.

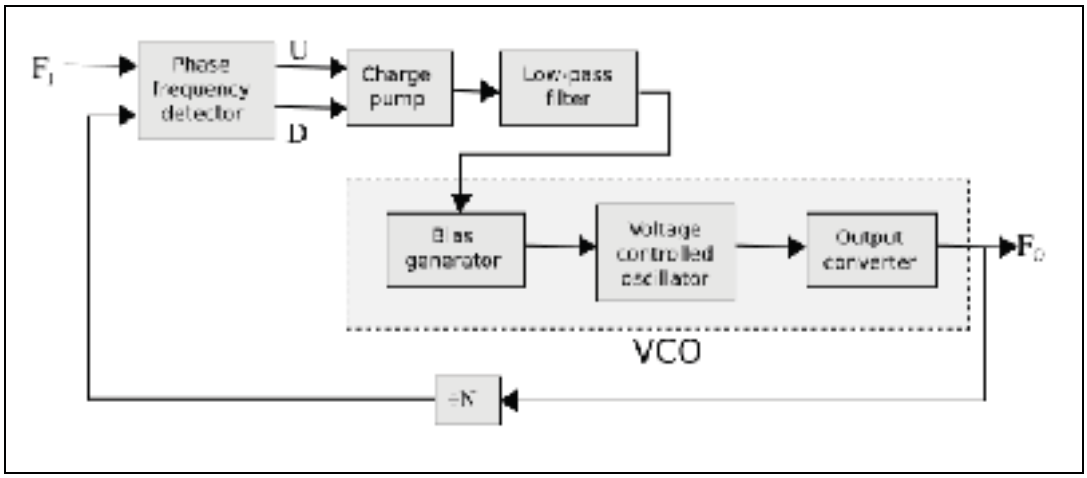

Figure 3.1 Structure de la PLL.
Tiré de Wikipedia (2009b)

Une PLL est généralement construite par un d'un détecteur de phase, un filtre passe bas et un oscillateur contrôlé en tension placé dans une configuration de boucle négative. On peut placer dans la boucle de retour un diviseur de fréquence pour que l'horloge de sortie de la PLL soit un multiple entier de la référence. Un multiple non entier de la référence peut être crée en remplaçant le diviseur par un pulse swallowing counter.

L'oscillateur génère un signal périodique. Admettons que, initialement, l'oscillateur possède une fréquence presque égale au signal référence. Lorsque la phase de l'oscillateur est inférieure à celle de la référence, le détecteur de phase cause le fonctionnement de la pompe afin de changer la tension de contrôle du VCO. Ainsi, l'oscillateur devient plus rapide. Dans le cas contraire, le détecteur de phase cause le changement de l'état de la pompe qui pilote le VCO afin que celui-ci ralentisse. Le filtre passe bas lisse les entrée brusque de la pompe de charge. De cette façon, l'oscillateur peut être loin de la fréquence de la référence.

Applications

Démoduler un signal radiofréquence

> « Dans les récepteurs radiofréquence, un signal électromagnétique modulé suivant un certain protocole (FSK, GMSK, QAM64, ...) génère une force électromotrice dans l'antenne qui est convertie en un signal électrique amplifié et filtré. Ce système utilise notamment une fréquence de référence appelée la LO (Local oscillator) qui est générée par une boucle à verrouillage de phase » (wikipedia, 2009b).

Retrouver la fréquence porteuse d'un signal

> « C'est le cas par exemple dans le cadre de la démodulation radio FM ou AM. Un poste radio contient une PLL dont on fait varier le filtrage du signal d'entrée. Ce signal filtré pilote ensuite la PLL qui se cale sur cette fréquence (phase d'accrochage) et générer ensuite en sortie le signal démodulé. »

Multiplier une fréquence d'horloge

> « Souvent les systèmes digitaux requièrent une fréquence interne de travail importante, de l'ordre de plusieurs centaines de mégahertz. En général un cristal oscillateur piézoélectrique sert de référence car il génère une gigue d'horloge très faible. L'inconvénient du cristal est que souvent sa fréquence est basse, de l'ordre de 20 MHz

à 30 MHz, et qu'il est nécessaire de multiplier sa fréquence par un facteur K afin de générer l'horloge du système digital. C'est le rôle de la PLL. »

ANNEXE IV

Matrice de transition du LP-LFSR

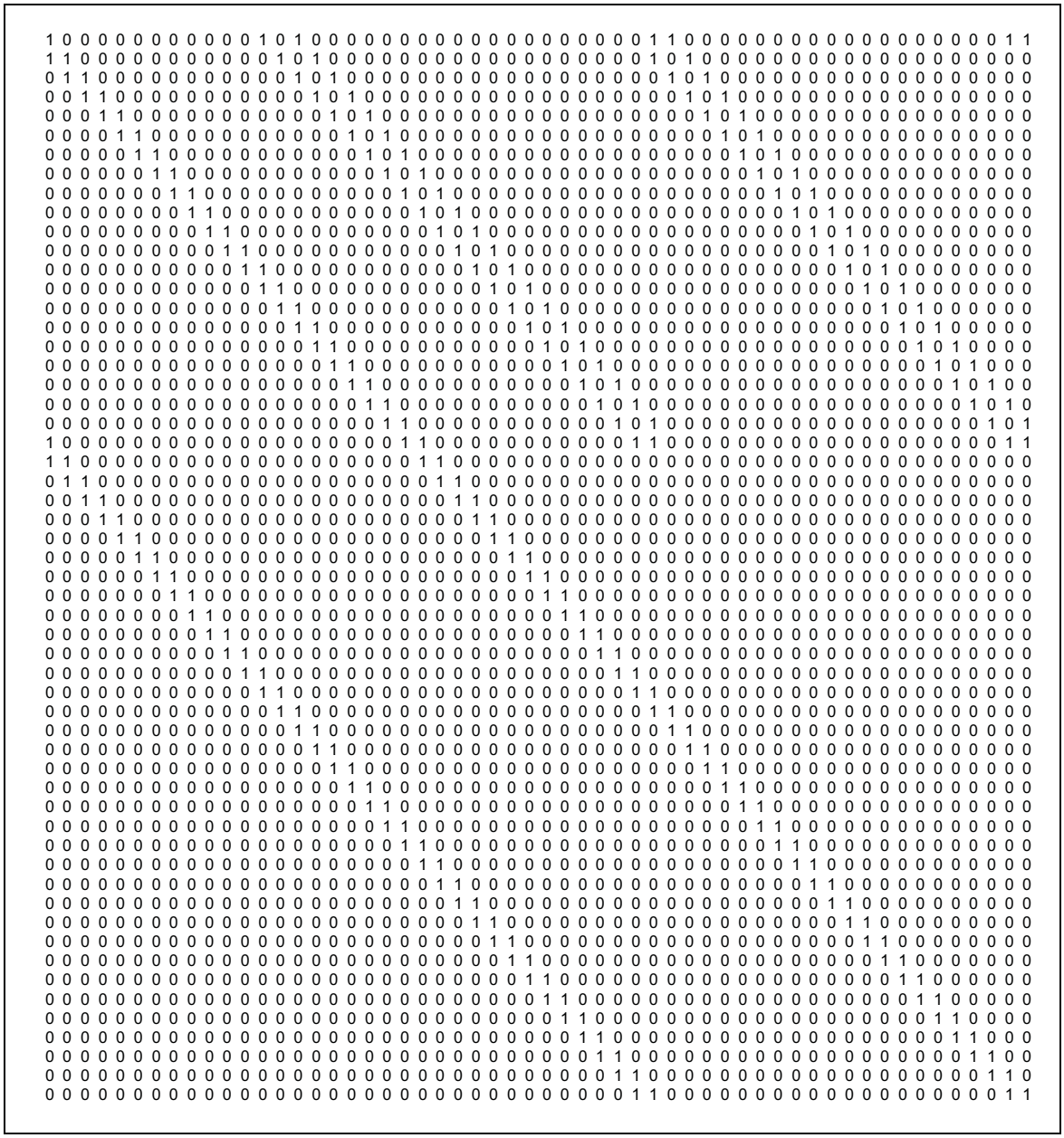

Figure 5.1 Matrice de transition du LP-LFSR.

ANNEXE V

Registres utilisés

Les registres utilisés pour la communication entre la partie logicielle et le système de contrôle de puissance et le systeme de contrôle du ratio C/N_0 sont listés dans le tableau 6.1.

- Le registre « ResetReg » permet d'initialiser le processus de transfert de données entre la partie IF et le système de contrôle de puissance.
- Le registre « NewData » indique au FPGA qu'il y a une nouvelle commande en attente d'être envoyé.
- Le registre « PowerControlReg » contient le mot de commande à envoyer au système de contrôle de puissance.
- Le registre « CalibrationReg » contient la commande de retour du système de contrôle de puissance pour le mode calibration.

Tableau 6.1 Registre utilisé pour la communication
(contrôle de puissance et contrôle de bruit)

Nom	Adresse	Privilège (logiciel)
PowerControlReg	0x1800	Écriture
NoiseControlReg	0x1804	Écriture
CalibrationReg	0x1808	Lecture
NewDataReg	0x180C	Lecture/ Écriture
ResetReg	0x1810	Écriture

ANNEXE VI

Schémas électriques des cartes du système de contrôle de puissance

La première carte de test a été réalisée pour valider l'ADC et pour l'acquisition des fonctions de transferts pour les trois fréquences des signaux GPS/Galileo. Le schéma électronique de cette carte et le PCB correspondant sont présentés par les figures 7.1 et 7.2.

Cette carte a été réalisée en utilisant le substrat FR4 dont les caractéristiques se résument dans le tableau 7.1.

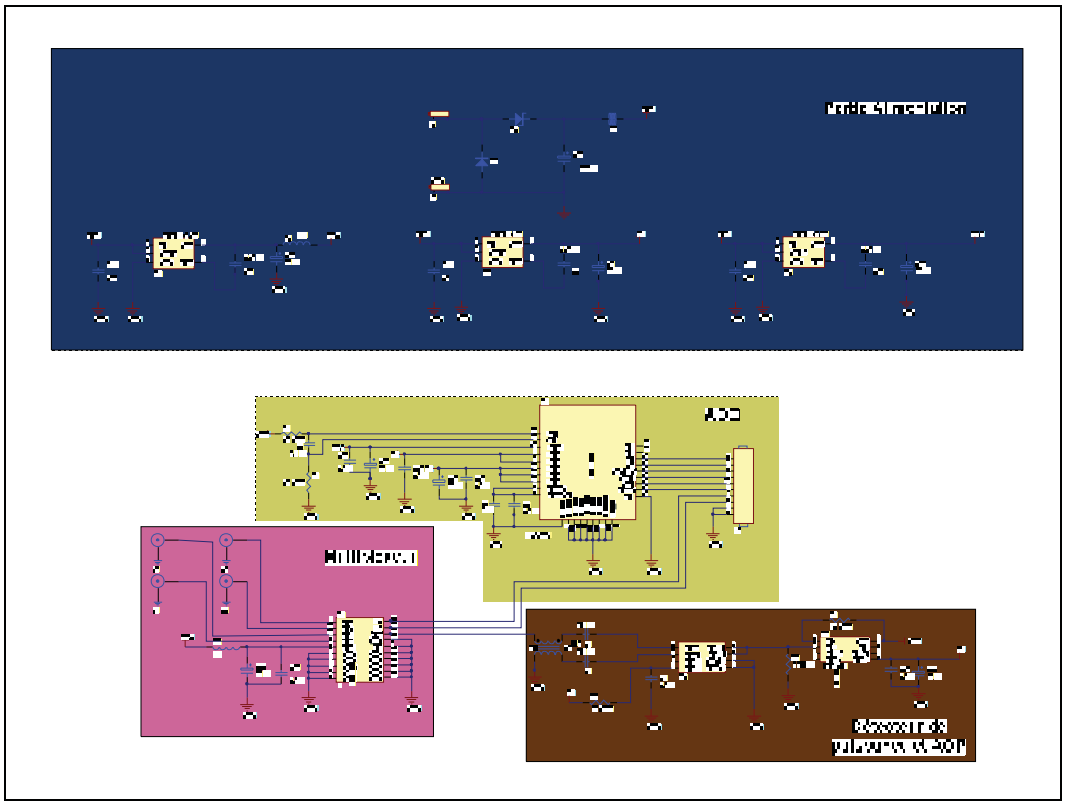

Figure 7.1 Schéma électrique de la première carte de test.

Tableau 7.1 : Caractéristiques du substrat FR4
de la carte n°1

Constante diélectrique ε_r	4.2
Facteur de Dissipation	0.002
Épaisseur	25 mil

La largeur de la ligne coplanaire 50Ω est calculée à l'aide de l'outil LineCalc du simulateur ADS. En prenant une séparation de 10mil entre la ligne et la masse, la largeur de la ligne est alors 35 mil.

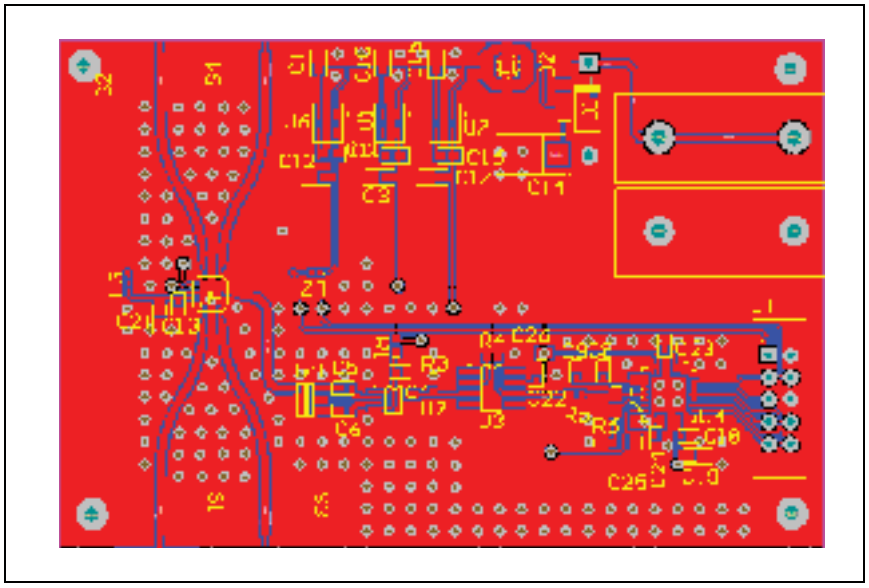

Figure 7.2 PCB du la première carte de test.

La deuxième carte est une extension de la première. Elle intègre les coupleurs 20dB pour le prélèvement des signaux de la chaine RF du simulateur. Elle intègre aussi le DAC qui pilotera les VGA. Dans cette carte, le détecteur de puissance AD8313 a été changé par l'AD8318. La figure 7.3 illustre les changements effectués sur le schéma électrique. La figure 7.4 représente le PCB de la carte finale.

Cette carte aura pour rôle de tester le DAC et enfin le système final en intégrant les VGA. Elle a été réalisée en utilisant le substrat FR4 dont les caractéristiques se résument dans le tableau 7.2.

Tableau 7.2 : Caractéristiques du substrat FR4 de la carte finale

Constante diélectrique ε_r	4.2
Facteur de Dissipation	0.002
Epaisseur	62 mil

La largeur de la ligne coplanaire 50Ω est calculée à l'aide de l'outil LineCalc du simulateur ADS. En prenant une séparation de 7mil entre la ligne et la masse, la largeur de la ligne est alors 43 mil.

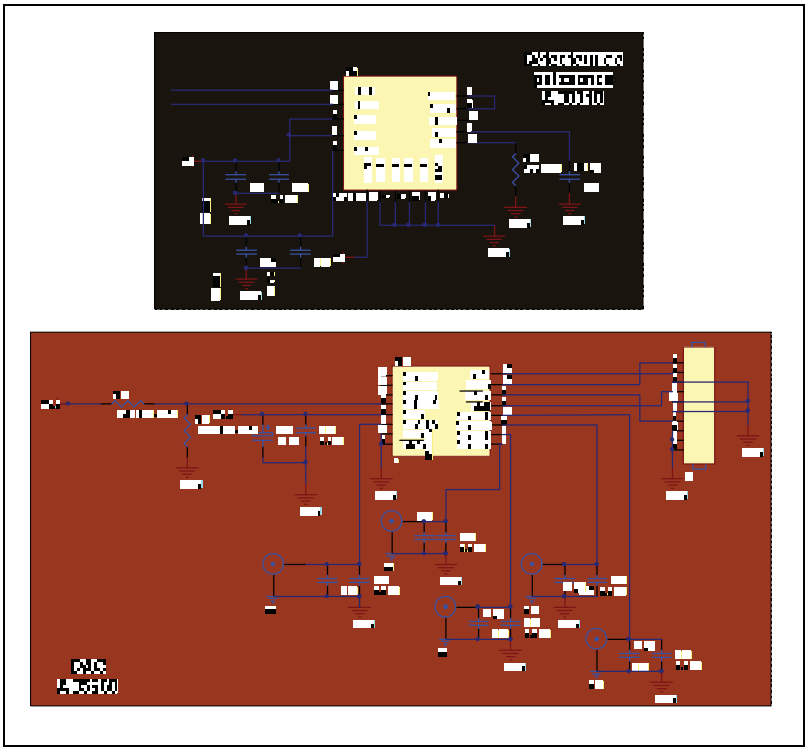

Figure 7.3 Les nouveaux éléments du schéma électrique par rapport à la deuxième carte.

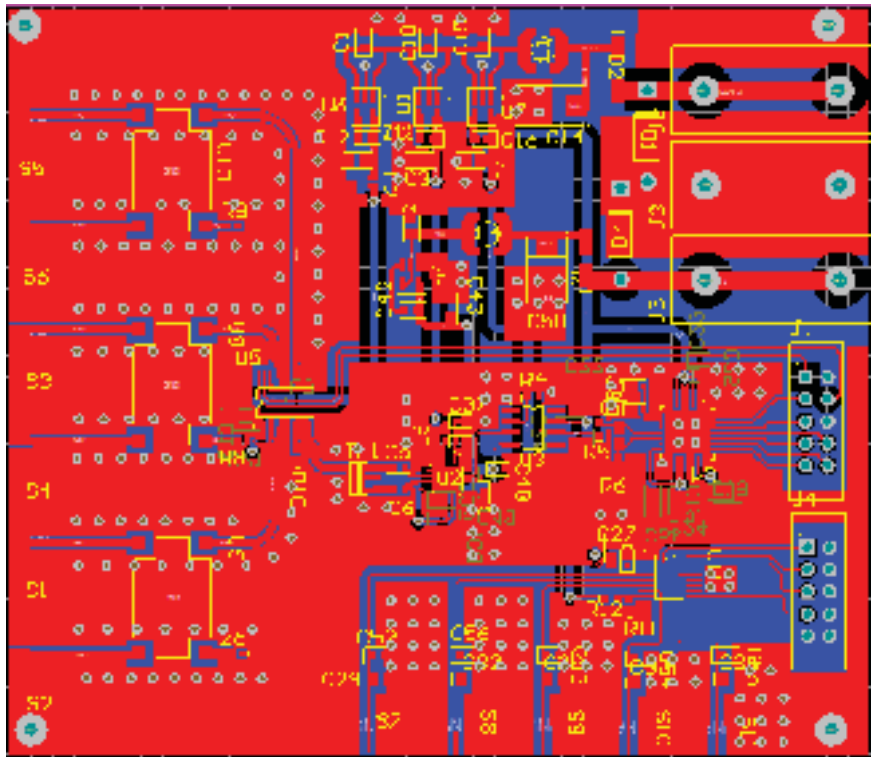

Figure 7.4 PCB de la carte finale du SDAP.

ANNEXE VII

Procédure de test du CAN

La figure 8.1 présente la machine à état implémenté en VHDL pour le contrôle de l'ADC.

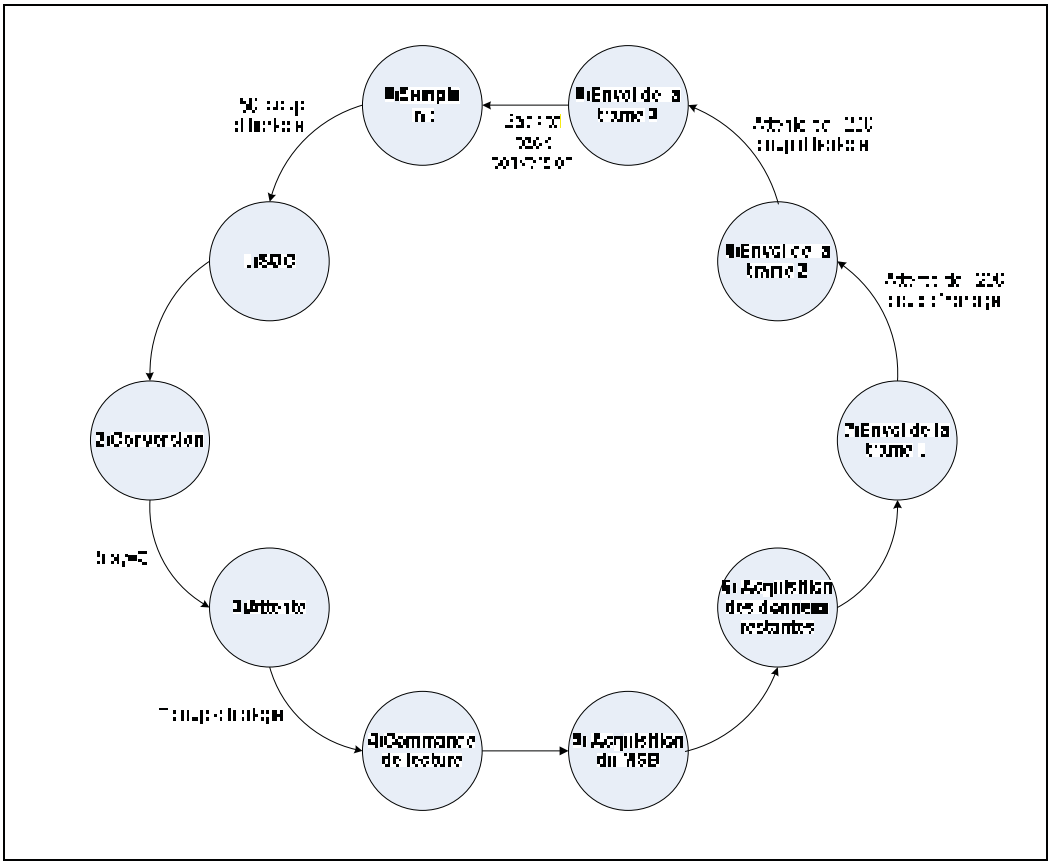

Figure 8.1 Machine à états de contrôle de l'ADC.

Le CPLD fonctionne à une fréquence de 16MHz soit une période de 62.5*ns*. La contrainte sur le multiplexeur (temps de montée à la mise sous tension du système : 10ns) ne présente pas un problème car ce temps est inférieur à la période de l'horloge et le système entame son fonctionnement après un coup d'horloge.

Les contraintes de timing relatives à l'ADC ($t_{quiet1}, t_{quiet2}, t_{quiet3}$) (figure 8.2) sont respectées. Pendants ces périodes l'ADC ne doit recevoir aucune commande :

- $t_{quiet1} = 30ns < T_{horloge}$ et $t_{quiet2} = 10ns < T_{horloge}$: toutes les commandes issues du CPLD sont actives au front montant de l'horloge sauf l'horloge pour l'ADC qui est une copie de l'horloge principale mais celle-ci ne fonctionne que durant la phase d'acquisition des données soit après l'écoulement de t_{quiet1} et t_{quiet2}, et donc aucune opération d'entrée sortie ne peut être effectué durant ces deux périodes ;
- $t_{quiet3} = 600ns \approx 10$ coups d'horloges : Lorsque le signal *Busy*=1, aucune commande n'est envoyée à l'ADC.

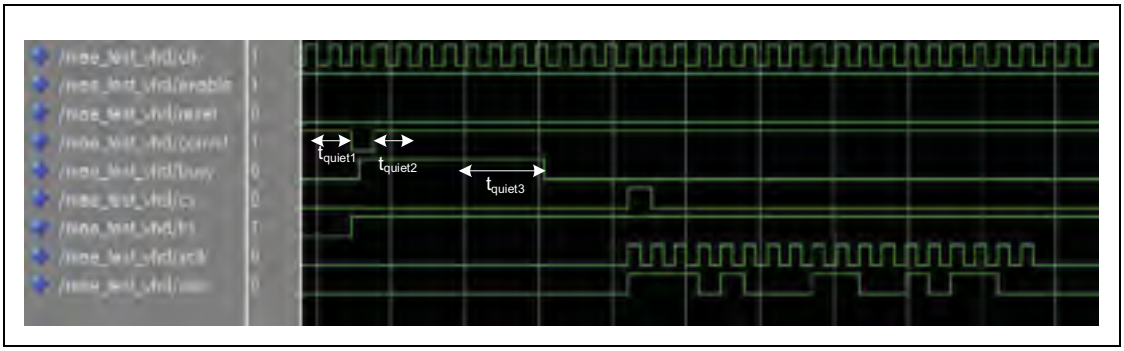

Figure 8.2 forme des signaux pour le contrôle de l'ADC.

Le port parallèle fonctionne à une fréquence de l'ordre des quelques KHz. Afin de ne pas fausser les données, à cause de la non synchronisation entre le port parallèle et le CPLD, un délai de 1200 coups d'horloges soit 75µs est prévu entre l'envoi des trames ce qui nous donne une fréquence de 13.3KHz lors de l'envoie des données (figure 8.1). Ce programme servira pour l'acquisition des données du nouveau détecteur de puissance de la carte finale du système de détection et d'ajustement de puissance.

BIBLIOGRAPHIE

Analog Devices, Inc. 2005a. « Octal, 12-/14-/16-Bit DAC with 5 ppm/°COn-Chip Reference in 14-Lead TSSOP ». En ligne. 28 p. <http://www.analog.com/static/importedfiles/data_sheets/AD5628_5648_5668.pdf>. Consulté le 19 mai 2009.

Analog Devices, Inc. 2005b. « 10 MHz to 3 GHz VGA with 60 dB Gain Control Range ». En ligne. 24 p. <http://www.analog.com/static/imported-iles/data_sheets/ADL5330.pdf> Consulté le 05 juin 2009.

Analog Devices, Inc. 2007a. « Wideband 2.5 GHz, 37 dB Isolation at 1 GHz, CMOS 1.65 V to 2.75 V, 4:1 Mux/SP4T ». En ligne. 16 p. <http://www.analog.com/static/imported-files/data_sheets/ADG904_904R.pdf> Consulté le 05 juin 2009.

Analog Devices, Inc. 2007b.« 1 MHz to 8 GHz, 70 dB Logarithmic Detector/Controller ». En ligne. p24. <http://www.analog.com/static/importedfiles/data_sheets/AD8318.pdf>. Consulté le 30 septembre 2009.

Anaren, inc. 2006a. « Model XC1900A-20 ». En ligne. 22 p. <http://www.datasheetarchive .com/XC1900A-20-datasheet.html>. Consulté le 29 septembre 2009.

Anaren, inc. 2006b. « Model XC0900A-20 ». En ligne. 23 p. <http://www.datasheetarchive .com/XC1900A-20-datasheet.html#>. Consulté le 29 septembre 2009.

ARINC Incorporated. 2001. NAVSTAR GPS Space Segment / User segment L5 Interfaces, Interface Control Document (ICD), ICD-GPS-705. El Segundo, California: ARINC Research Corporation, 70 p.

ARINC Research Corporation. 2000. NAVSTAR GPS Space Segment/Navigation User Interface, Interface Control Document (ICD), ICD-GPS-200, Rev. C. El Segundo, California: ARINC Research Corporation, 138 p.

Barkat, Mourad. 2005. Signal detection and estimation. 2nd ed. Boston : Artech House radar library. 692p.

Cleve Moler, 2009. « Random Numbers ». En ligne. 15 p. <http://www.mathworks.com/moler/random.pdf>. Consulté le 12 octobre 2009.

Cole-Parmer. 2009. « What is GPS? ». En ligne. <http://www.coleparmer.com/techinfo /techinfo.asp?htmlfile=GPS_Garmin.htm&ID=318>. Consulté le 25 novembre 2009.

Danger, J. L., A. Ghazel, E. Boutillon et H. Laamari. 2000. « Efficient FPGA implementation of Gaussian noise generator for communication channel emulation ». In *Electronics, Circuits and Systems, 2000. ICECS 2000. The 7th IEEE International Conference on*. Vol. 1, p. 366-369.

Dusausay, Serge. 2002. « Comprendre l'électronique par la simulation ». En ligne. < http://serge.dusausay.free.fr/ss/complement/art40/art40.html>. Consulté le 06 octobre 2009.

European Space Agency. 2006. Galileo Open Service Signal In Space Interface Control Document (OS SIS ICD), GAL OS SIS ICD/D.0. Bruxelle: Galileo Joint Undertaking, 192 p.

Ghazel, A., E. Boutillon, J. L. Danger, G. Gulak et H. Laamari. 2001. « Design and performance analysis of a high speed AWGN communication channel emulator ». In *Communications, Computers and signal Processing, 2001. PACRIM. 2001 IEEE Pacific Rim Conference on*. Vol. 2, p. 374-377.

Goodman, Jonathan. 2005. En ligne. 10p. <http://www.math.nyu.edu/faculty/goodman/teaching/MonteCarlo2005/notes/GaussianSampling.pdf>. Consulté le 28 novembre 2009.

Maxim IC. 2001. « Understanding SAR ADCs ». En ligne. 6 p.<http://pdfserv.maxim-ic.com/en/an/AN1080.pdf>. Consulté le 20 Octobre 2009.

Nguyen, Minh-Quang. 2005. « FPGA implementation of a digital communication chain ». M.Sc., Canada, Universite Laval (Canada), 111 p.

NorthWood Labs LLC. 2003. « GPS Carrier-to-Noise Density ». En ligne. 3 p. <http://www.northwoodlabs.com/AN101.pdf>. Consulté le 19 juin 2009.

Novatel, 2007. « Glonass Overview ». En ligne. 7p. <http://www.novatel.com/Documents/Papers/GLONASSOverview.pdf>. Consulté le 28 novembre 2009.

Pong P. Chu and Robert E. Jones. 2000. « Design Techniques of FPGA Based Random Number Generator ». En ligne. 6 p. <http://klabs.org/richcontent/MAPLD Con99/Papers/d5_chu_p.pdf>. Consulté le 16 février 2009.

Pozar, David M. 2005. *Microwave Engineering* 3th ed. Hoboken : J.Wiley. 700p.

Proakis, John G. 2000. *Digital communications* 4th ed. Boston: Boston : McGraw-Hill. 1002 p.

Sauriol, Bruno. 2007. « Fast Synchronous Bus ». Technical note : LACIME. 7p.

Sauriol, Bruno. 2008. « Mise en oeuvre en temps réel d'un récepteur hybride GPS-GALILEO ». Montréal, École de technologie supérieure, Montréal, 2008., 308p.

Texas Instrument, inc. 2004. « 18-BIT, 600-kHz, pseudo-differential input, micropower, sampling analog-to-digital converter with serial interface and reference », En ligne. 31 p .<http://focus.ti.com/general/docs/lit/getliterature.tsp?genericPartNumber=ads8380&fileType=pdf>Consulté le 04 octobre 2009.

Tsang, George Marsaglia and Wai Wan. 2000. « The Ziggurat Method for Generating Random Variables ». *Journal of Statistical Software,* vol. 5, (2000-10-02), p. 8.

U-blox. 2007. « GPS Essentials of Satellite Navigation ». En ligne. 175p. <http://www.zogg-jm.ch/Dateien/GPS_Compendium%28GPS-X-02007%29.pdf>. Consulté le 28 novembre 2009.

UNOOSA, 2008. « COMPASS/Beidou Navigation Satellite System Development ». En ligne. 42p. <http://www.oosa.unvienna.org/pdf/icg/2008/icg3/06.pdf>. Consulté le 28 novembre 2009.

Wikipedia. 2009a. « White noise ». En ligne. <http://en.wikipedia.org/wiki/White_noise>. Consulté le 20 Octobre 2009.

Wikipedia. 2009b. « Boucle à phase asservie ». En ligne. <http://fr.wikipedia.org/wiki/Boucle_%C3%A0_phase_asservie#D.C3.A9moduler_un_signal_radio-fr.C3.A9quence>. Consulté le 29 Octobre 2009.

WJ Communications,Inc. 2004.. « Image-reject and single-sideband mixers ». En ligne. 6 p. <http://www.triquint.com/prodserv/tech_info/docs/WJ/ImageRej_n_SSB_mixers.pdf>. Consulté le 21 mai 2009.

Yacine Adane, Aurelian Constantinescu, Jean Belzile, and Ammar Kouki. 2007. « Design of Miniature RF Band-pass Filters for a Hybrid GPS/Galileo Receiver Front-End Insert ». In *National Technical Meeting of the Institute of Navigation*. p. 350 - 356. San Diego, California.