Etude et conception d'une structure BIST pour convertisseur analogique-numérique

Nicolas Mechouk

$N°$ d'ordre : 4079

Thèse

présentée à

L'Université Bordeaux

Ecole Doctorale des Sciences Physiques et de l'Ingénieur

par **Nicolas MECHOUK**

Pour obtenir le grade de

Docteur

Spécialité : électronique

Etude et conception d'une structure BIST pour convertisseur analogique-numérique

---

Soutenue le : 26 Octobre 2010

Après avis de :

M.	**Patrick LOUMEAU**	Professeur	Telecom ParisTech	Rapporteur
M.	**Chiheb REBAI**	Maître de Conférences, HDR	SUP'COM, Tunis	Rapporteur

Devant la commission d'examen formée de :

M.	**Lilian BOSSUET**	Maître de Conférences, HDR	Université St Etienne	Co-Directeur
M.	**Dominique DALLET**	Professeur	IPB	Directeur
M.	**Bertrand LE GAL**	Maître de Conférences	IPB	Examinateur
M.	**Patrick LOUMEAU**	Professeur	Telecom ParisTech	Examinateur
M.	**Guillaume MONNERIE**	Ingénieur de Recherche	NXP Caen	Examinateur
M.	**Claude PELLET**	Professeur	Université Bordeaux	Président
M.	**Chiheb REBAI**	Maître de Conférences, HDR	SUP'COM, Tunis	Examinateur

A la mémoire de Renée....
A mes parents....
A Kate....

"A long time ago in a galaxy far, far away...."
G. Lucas

Les travaux de recherche présentés dans ce mémoire se sont déroulés au sein du laboratoire d'Intégration du Matériau au Système (IMS) de l'Université Bordeaux 1.

Au terme de ce travail, je tiens à remercier Monsieur Claude Pellet, directeur du département COFI de l'IMS, d'avoir accepté de présider le jury.

Je remercie également Messieurs Patrick Loumeau, professeur à Telecom ParisTech, et Chiheb Rebai, Professeur à Sup'com, Tunis, d'avoir accepté la charge de rapporteurs de cette thèse ainsi qu'en participant au jury.

Je souhaite témoigner ma reconnaissance à Monsieur Guillaume Monnerie, ingénieur chez NXP Caen, d'avoir accepté de "jouer" le rôle d'industriel au sein du jury.

Que Monsieur Dominique Dallet, professeur à l'IPB, trouve ici l'expression de mes remerciements pour avoir dirigé mes travaux de recherche.

Je souhaite saluer Messieurs Bertrand Le Gal, maitre de conférence à l'IPB, et Lilian Bossuet, maître de conférence à l'université de St Etienne, pour toute l'aide qu'ils ont pu me fournir en tant qu'encadrants de thèse.

Je tiens également à remercier tout particulièrement Mesdames Simone Dang Van, Mireille Raffin et Christine Chatain pour avoir grandement faciliter le coté administratif de ma thèse.

Un grand merci à mes compagnons de voyages, Mesdemoiselles Marie Lise Bourqui, Charlotte Sury et Laurianne Blanc et Messieurs Brice Grandchamp et Willy Ludurczak, sans qui tout ceci n'aurait pas été possible....

Table des matières

Table des figures

Liste des tableaux

Listes des abréviations

BIST	Built-In Self Test
CAN	Convertisseur Analogique Numérique
CNA	Convertisseur Numérique Analogique
dB	déciBel
$\Delta\Sigma$	Sigma-Delta
ENOB	Effective Number of Bits, Nombre Effectif de bits
FF	Flip-Flop, Bascule
FFT	Fast Fourier Transform, Transformée de Fourier Rapide
FPGA	Field Programmable Gate Array
LDI	Lossless Discret Integrator
LSB	Least Significant Bit
LUT	Look-Up Table
MASH	MultistAge noise SHapping
NLD	Non-Linéarité Différentielle
NLI	Non-Linéarité Intégrale
NTF	Noise Transfer Function
OSR	OverSampling Ratio
SFDR	Spurious-Free Dynamic Range, Plage Dynamique Libre de toute Distortion Harmonique
SINAD	Signal-to-Noise-And-Distortion, Rapport Signal sur Bruit avec Distortion Harmonique
SNR	Signal-to-Noise Ratio, Rapport Signal sur Bruit
STF	Signal Transfer Function
THD	Total Harmonic Distortion, Taux de Distortion Harmonique
TFD	Transformé de Fourier Discrète
VHDL	Very high-speed integrated circuits Hardware Description Language

Fs	Fréquence d'échantillonnage
Ts	Période d'échantillonnage
Ω_r	Pulsation de Résonance
f_r	Fréquence de Résonance
f_0	Fréquence du fondamental d'un signal issu d'un CAN
$H(z)$	Fonction de transfert d'un filtre
$(r,\ \theta)$	Module et Argument des pôles d'une fonction de transfert
(b_i, a_i)	Coefficients de la fonction de transfert d'un filtre

Introduction générale

L'homme d'aujourd'hui vit au centre d'un réseau de communication. La communication correspond à l'envoi d'une information véhiculée par un signal à travers un canal. Le premier type de communication utilisé par l'homme est la parole : *un message* transit à travers *l'air* par *le son*. Il existe une infinité de sons différents, même si l'humain ne peut tous les produire ou même les recevoir. A partir de la fin du XVII^{ème} siècle et suite à l'expérience de Guillaume Amontons (*1663-1705*) entre Meudon et Paris, il est possible d'envoyer un message à distance au moyen de signaux lumineux. Le message est alors transmis lettre par lettre parmi un nombre fini de code. Grâce aux différents progrès technologiques, les communications sont, de nos jours, plus rapides, à plus longue distance et permettent l'envoi d'une énorme quantité d'information.

Aujourd'hui, les communications sont devenus télécommunications et la société est devenue numérique. La radio, la téléphonie, la photographie, la télévision, le cinéma sont aujourd'hui numériques. Ce basculement de l'analogique au numérique a permis l'augmentation du nombre de services disponibles sur un même terminal et des avantages économiques. Cette numérisation a conduit à la réalisation de systèmes travaillant et communiquant avec des informations numériques plutôt qu'analogiques.

Parmi les avantages du traitement numérique du signal par rapport au traitement analogique, on peut citer l'absence de dérives des caractéristiques (les caractéristiques du traitement sont figées d'une manière rigoureuse), une grande reproductibilité (les défauts inhérents aux composants n'ont aucune conséquence), un haut niveau de qualité (il suffit d'augmenter le nombre de bits pour réduire le niveau de distorsion en-deçà d'un niveau donné), la facilité de conservation ou de mémorisation (les mémoires flash ne nécessitent pas d'alimentation pour maintenir les données), la possibilité de fonctions nouvelles (souplesse de programmation),...

Cependant la nature des signaux physiques reste analogique : le son pour la radio et la téléphonie, une image pour la photographie et la vidéo pour la télévision et le cinéma. Il faut donc convertir ces signaux analogiques, à valeurs et à temps continus en signaux numériques,

à valeurs et à temps discrets. C'est ici qu'intervient le Convertisseur Analogique Numérique (CAN). Malheureusement, la conversion d'un signal à valeurs continues en un signal à valeurs discrètes introduit des erreurs lors de la conversion du signal. C'est pourquoi le convertisseur analogique numérique est le maillon critique, mais incontournable, d'une chaîne de traitement numérique du signal : il est donc nécessaire de connaître le comportement, les caractéristiques de fonctionnement et les performances d'un CAN.

La caractérisation classique d'un convertisseur analogique numérique consiste à analyser le signal issu de sa sortie. Le principal inconvénient de cette méthode concerne les moyens qu'elle met en oeuvre. De plus, il faut prendre en compte les erreurs de mesures dues à l'instrumentation, qui doivent être séparées de celles du composant. La caractérisation classique d'un CAN peut se faire selon trois méthodes : l'analyse statique, par application d'un histogramme, l'analyse spectrale, par transformée de Fourier et l'analyse temporelle par l'utilisation d'algorithmes d'interpolation.

Parallèlement aux progrès des unités de traitements numériques de plus en plus rapides, consommant de moins en moins d'énergie et de moins en moins onéreuses, les convertisseurs analogiques numériques et numériques analogiques deviennent de plus en plus précis, rapides et intégrés en nombre croissants dans des systèmes toujours plus complexes. Les outils et moyens de caractérisation doivent également devenir de plus en plus sophistiqués. Malheureusement, il est souvent difficile de trouver une instrumentation adéquate ou il est impossible d'accéder directement aux signaux de sortie d'un convertisseur lorsque celui-ci est intégré dans un système complexe.

Une solution à ce problème est l'intégration des méthodes de caractérisation autour du le CAN, permettant à la fois de s'affranchir des problèmes liés à l'instrumentation et de permettre ainsi le test in-situ de composants intégrés. En effet, plutôt que de caractériser totalement un convertisseur, un utilisateur voudra savoir rapidement si les performances du convertisseur correspondent aux performances recherchées. L'étude menée au cours de cette thèse s'inscrit dans le cadre de cette démarche en intégrant une méthode de caractérisation au composant pour en effectuer un test embarqué (BIST : Built-In Self Test). Cette solution permet donc l'extraction et le test des paramètres d'un CAN dans son environnement mais nécessite l'introduction d'un générateur de stimuli sur la puce.

L'étude du BIST est donc séparée en deux parties : d'un coté, la génération des stimuli

et de l'autre, le traitement du signal de sortie du CAN pour permettre l'estimation de certains paramètres du CAN. La génération de stimuli correspond à la génération d'une sinusoïde. Nous utiliserons un oscillateur numérique associé à un modulateur $\Sigma\Delta$ pour obtenir un signal sinusoïdal sur 1 bit, signal qui sera converti en un signal analogique par un simple filtre analogique passif passe-bas ou passe-bande. Le traitement du signal issu du CAN est constitué de l'extraction des différentes composantes spectrales du signal par filtrage numérique, et du calcul des paramètres spectraux (*SNR*, *SINAD*, *THD*...) par estimation de leur puissance. Chacune de ces parties, du fait de leur intégration au sein du composant, doivent être optimales en terme d'utilisation de surface et de consommation d'énergie, alors que leurs performances doivent être suffisantes pour permettre une bonne estimation des paramètres spectraux, représentatifs des performances d'un CAN. Nous proposons donc une méthodologie pour permettre d'intégrer un système de test à faible coût (surface et consommation) et efficace autour d'un CAN.

Ce mémoire se présente sous la forme de cinq chapitres. Le premier chapitre se concentrera sur l'environnement de la conversion analogique numérique, l'étude des convertisseurs analogique numérique et le test des CAN. Nous commencerons par présenter le principe de la conversion analogique numérique en détaillant les processus d'échantillonnage et de quantification. Puis les caractéristiques des CAN en terme d'erreurs statiques et dynamiques, telles que les erreurs d'offset, de gain, de non linéarité (paramètres de performance statiques) et les erreurs à l'ouverture, le taux de distorsion harmonique ainsi que le rapport signal à bruit (paramètres de performance dynamiques) seront illustrées. La caractérisation dynamique d'un CAN sera présentée dans un troisième temps par la description de trois méthodes d'analyse (analyse temporelle, analyse statistique et analyse spectrale) permettant de déduire différents paramètres spectraux. Enfin, les principales techniques de test embarqués proposées dans la littérature seront présentées.

La génération in-situ d'un signal sinusoïdal analogique à partir d'un oscillateur numérique associé à un modulateur $\Sigma\Delta$ sera présentée au second chapitre. Nous commencerons par détailler la génération d'un signal analogique, à l'aide d'un oscillateur numérique puis nous nous intéresserons à la modulation $\Sigma\Delta$ pour optimiser les performances de l'oscillateur numérique. Dans un second temps, nous présenterons différents générateurs et différents modulateurs $\Sigma\Delta$ proposés dans la littérature. Enfin, dans un troisième temps, la réalisation matérielle du générateur choisi ainsi que ses caractéristiques seront présentées.

Le troisième chapitre se focalisera sur l'unité d'extraction des paramètres spectraux. Dans un premier temps, nous y présenterons une étude théorique du filtre résonateur et du filtre notch. Nous nous intéresserons ensuite aux différentes topologies du banc de filtres (cascadés, parallèles-cascadés et parallèles). Celles-ci seront, pour finir, analysées en fonction de leur capacité à

mesurer avec un biais minimal le *SNR* du CAN sous test.

Dans le quatrième chapitre, l'implémentation du banc de filtres sera abordé. Différentes structures d'implémentation du filtre résonateur (structures en forme directe I et II, en forme transposée et structures *lossless discrete integrator*) seront présentées. Ensuite, l'implémentation d'un premier banc de filtres contraint au niveau de la fréquence du fondamental sera réalisé. L'objectif sera d'estimer le SNR du CAN considéré avec une précision d'un demi LSB. Nous nous concentrerons ensuite sur une méthodologie afin obtenir un banc de filtre optimal en terme de surface tout en conservant des performances adéquates au test. Nos propos seront illustrés par le cas réel du CAN AD9042D d'Analog Devices.

Enfin, dans le dernier chapitre, nous reviendrons aux filtres constituants le banc de filtres pour en présenter une version adaptative. En effet, le calcul des coefficients des filtres se fait par approximation, ce qui entraîne un décalage entre la fréquence de résonance voulue pour un filtre et la fréquence de résonance obtenue lors de la réalisation matérielle de ce même filtre, décalage augmenté par le codage en virgule fixe des coefficients des filtres. De plus, des perturbation externes aux circuits du BIST peuvent introduire une différence entre le signal généré et le signal transmit au CAN. L'utilisation d'une méthode adaptative doit permettre la correction de ces défauts.

Ce mémoire de thèse se terminera par une conclusion générale mettant en évidence les avancés que ce travail peut apporter dans le domaine du BIST appliqué à la conversion analogique numérique et des perspectives quand à l'évolution du système présenté.

Chapitre 1

La conversion analogique numérique : principe, caractéristiques et test

Sommaire

Introduction

Le convertisseur analogique numérique (CAN) constitue l'interface fondamentale entre l'environnement physique, où les signaux sont analogiques, et les circuits numériques de traitement des données, très largement utilisés en raison de leur immunité au bruit, de leur insensibilité au phénomène de dérive, de leur possible reconfigurabilité selon le type de circuit et de la souplesse de leur conception. La grandeur physique (température, pression, lumière, son, image) est convertie par des capteurs en un signal électrique dont les valeurs (tension, courant) dépendent du phénomène physique mesuré. Les CAN sont devenus, à ce titre, un maillon essentiel de l'électronique. Ils sont présents dans la quasi-totalité des circuits mixtes qui contiennent une partie analogique et une partie numérique. L'opération de conversion analogique numérique, appelée aussi numérisation, se fait en deux étapes distinctes : l'échantillonnage et la quantification. Comme la numérisation s'effectue au moyen de composants électroniques non idéaux, cette opération va engendrer des déformations sur le signal à traiter. Celles-ci seront systématiques par rapport au processus de quantification et aléatoires par rapport à la non idéalité des composants. La première partie de ce chapitre est consacrée à l'étude du principe de fonctionnement du processus d'échantillonnage et de l'opération de quantification. Ensuite, les erreurs dues aux imperfections de l'électronique réalisant les circuits de conversion sont détaillées. Dans la troisième partie, nous détaillons les critères de performance du CAN. Enfin, les principales stratégies du BIST appliqué au CAN seront présentées dans la quatrième et dernière partie.

1.1 Principe de la conversion analogique numérique

L'opération de conversion analogique numérique consiste à transformer un signal continu dans le temps et en amplitude en un signal discrétisé en temps et en amplitude qui se propage dans des circuits numériques. Ces signaux numériques sont une suite de mots binaires régulièrement espacés dans le temps, ne prenant qu'un nombre fini de valeurs. Quelque soit l'architecture du CAN, le processus de conversion passe par deux étapes : l'échantillonnage temporel et la quantification des amplitudes [2].

1.1.1 L'échantillonnage

L'opération d'échantillonnage est analysée à travers deux niveaux : le niveau système [3] et le niveau circuit [4].

1.1.1.1 Aspect système

L'échantillonnage consiste à représenter un signal continu dans le temps $s(t)$ par ses valeurs $s(nT_s), n \in \mathbb{Z}$, à des instants multiples de T_s, appelée période d'échantillonnage. L'échantillonnage peut donc être interprété comme la modulation en amplitude d'un signal $s(t)$ par une distribution $u(t)$ nommée peigne de Dirac. La représentation mathématique du peigne de Dirac est :

$$u(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT_s) \tag{1.1}$$

Pour illustrer ce phénomène, nous utilisons un signal sinusoïdal pur $s(t)$, soit :

$$s(t) = A \times \cos(2\pi f_0 t + \phi) \tag{1.2}$$

où A est l'amplitude maximale, f_0 est la fréquence du signal d'entrée et ϕ est la phase. Le signal échantillonné, $s_e(nT_s)$, s'écrit alors sous la forme :

$$s_e(nT_s) = A \times \cos(2\pi f_0 nT_s + \phi) \tag{1.3}$$

D'un point de vue spectrale, l'opération d'échantillonnage affecte le spectre $S_e(f)$ du signal échantillonné. En effet, le spectre du signal échantillonné comprend la fonction $S(f)$, désignée par la bande de base, ainsi que les bandes images qui correspondent à la translation de la bande de base à des multiples entiers de la fréquence d'échantillonnage. Le spectre du signal échantillonné $S_e(f)$ résulte du produit de convolution de $S(f)$ par $U(f)$, $U(f)$ étant le spectre de la fonction peigne de Dirac.

$$S_e(f) = S(f) * U(f) = \sum_{n=-\infty}^{\infty} S\left(f - \frac{n}{T_s}\right) \tag{1.4}$$

Une des caractéristiques fondamentales des signaux échantillonnés est leur périodicité spectrale en raison de la convolution de $S(f)$ par $U(f)$. Restituer le signal d'origine revient donc à supprimer cette périodicité en enlevant les bandes images. Ceci peut être réalisé par un filtre passe bas idéal dont la réponse impulsionnelle est :

$$h(t) = \frac{\sin\left(\frac{\pi t}{T_s}\right)}{\left(\frac{\pi t}{T_s}\right)} \tag{1.5}$$

Pour que le signal $y(t)$ soit identique au signal d'origine, il faut que son spectre soit identique à $S(f)$. Ceci n'est possible que si le spectre d'origine ne contient pas de fréquences supérieures à la moitié de la fréquence d'échantillonnage. Dans le cas contraire, la bande image se replie sur la bande de base, d'où le théorème d'échantillonnage de Shannon, qui est satisfait lorsque $F_s \geq 2f_0$.

1.1.1.2 Aspect circuit

Dans un convertisseur analogique numérique, l'opération d'échantillonnage est directement suivie par la quantification, qui n'est pas instantanée. Il faut donc maintenir la valeur des échantillons pour assurer la quantification. La manière la plus simple est d'associer un interrupteur à une capacité comme illustré par la figure 1.1. La capacité joue le rôle d'élément mémoire alors que l'interrupteur réactualise la valeur mémorisée ou l'isole de l'entrée. Dans le cas où l'interrupteur est fermé, l'entrée est transmise sur la sortie : c'est la phase d'échantillonnage. Dans le cas inverse, la sortie reste constante et égale à la dernière valeur du signal transmis : c'est la phase de maintien ou de blocage.

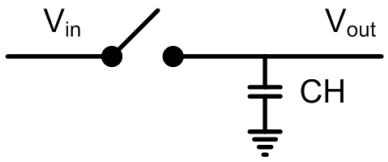


FIGURE 1.1 – Modèle idéal d'un échantillonneur bloqueur

1.1.2 La quantification

1.1.2.1 Principe

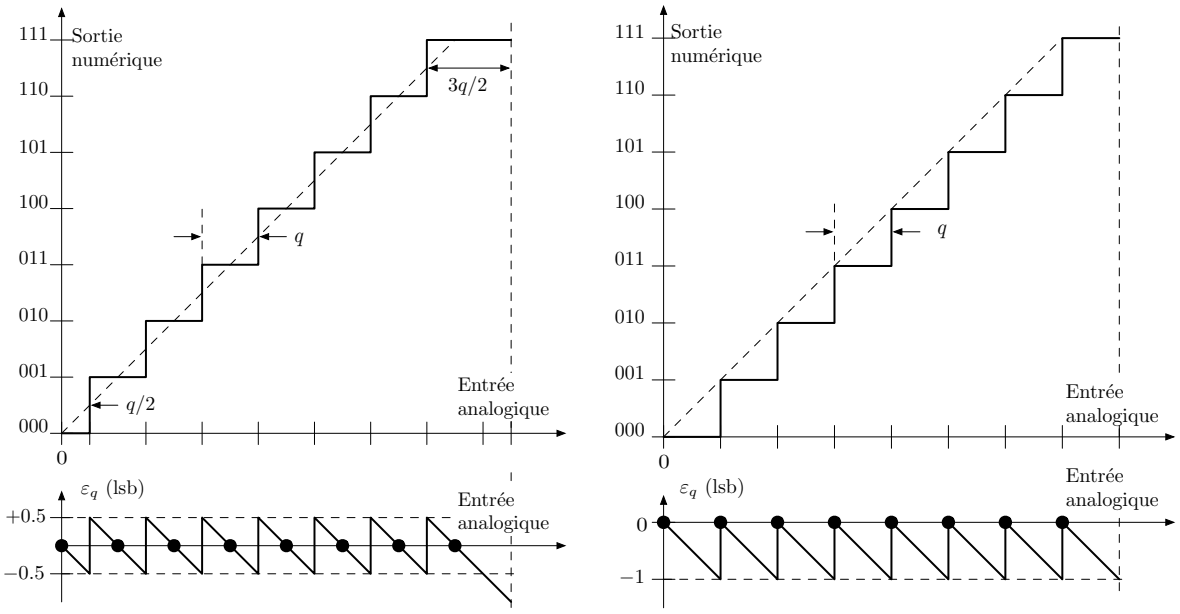
La quantification, qui représente la conversion analogique numérique, consiste en l'approximation de chaque valeur du signal échantillonné $s_e(nT_s)$ par un multiple entier d'une quantité élémentaire q appelée pas de quantification ou LSB (Least Significant Bit). Toutes les valeurs de sortie du quantificateur sont multiples de cette quantité élémentaire. Pour un q constant, quelque soit l'amplitude du signal d'entrée, la quantification est dite uniforme. L'opération de quantification revient alors à appliquer au signal d'entrée une caractéristique de transfert en marche d'escalier, comme le montre la figure 1.2. La position de cette caractéristique de transfert définie le type du quantificateur :

– le quantificateur par arrondi, figure 1.2(a), qui consiste à approcher par nq toutes les valeurs de l'intervalle $[(2n-1)\,q/2, (2n+1)\,q/2[$;

– le quantificateur par troncature, figure 1.2(b), qui consiste à approcher par nq toutes valeurs de l'intervalle $[nq, (n+1)\,q[$ et dont la caractéristique est déplacée de $q/2$ vers la droite sur l'axe des abscisses.

Un des paramètres clés d'un convertisseur analogique numérique est sa résolution res. Le quantum q dépend de la résolution et de la dynamique d'entrée ($V = V_{max} - V_{min}$) par la relation suivante :

$$q = \frac{V}{2^{res}} = \frac{V_{max} - V_{min}}{2^{res}} \tag{1.6}$$



(a) CAN par arrondi

(b) CAN par troncature

FIGURE 1.2 – Caractéristiques de transfert et fonctions d'erreur d'un CAN théorique de résolution égale à 3 bits non signé.

1.1.2.2 Erreur de quantification

Si le quantificateur était idéal avec une résolution $res \longrightarrow \infty$, la caractéristique de transfert serait une droite sur laquelle une équivalence sera faite entre chaque valeur analogique et le code de sortie. Mais, en réalité, res est de valeur finie. Toute une plage de valeur sera convertie en un seul nombre par l'utilisation d'une caractéristique en marche d'escalier. Ceci explique que la quantification, par les approximations utilisées, est un processus irréversible qui provoque une erreur systématique ne dépendant que du pas de quantification utilisé. Ainsi, pour un signal analogique à l'entrée du CAN exprimé par l'équation 1.2, la sortie du quantificateur idéal s'écrit

sous la forme [1] :

$$s(n) = A \times \cos(2\pi f_0 n T_s + \phi) + e_q(n) \tag{1.7}$$

où $e_q(n)$ correspond à l'erreur de quantification appelée aussi bruit de quantification.
L'erreur de quantification dépend non seulement de la fonction de transfert du CAN idéal, mais aussi du signal d'entrée considéré.Dans le cas d'un CAN par arrondi, elle est comprise entre $\frac{-q}{2}$ et $\frac{q}{2}$ autour d'une valeur moyenne nulle pour un signal d'entrée d'amplitude crête à crête égale à $PE - q$, où PE est la valeur quantifiée de la plus grande entrée. Le bruit de quantification généré présente une densité spectrale uniforme dans la bande $\left[\frac{-F_s}{2}, \frac{F_s}{2}\right], F_s = \frac{1}{T_s}$ dont la valeur efficace est donnée [2] par :

$$\sigma_q = \frac{q}{\sqrt{12}} = \frac{1}{\sqrt{12}} \tag{1.8}$$

L'équation 1.8 est vérifiée pour des signaux d'entrée analogiques linéaires (signaux triangulaires par exemple) et pour des signaux sinusoïdaux [4]. Finalement, les équations 1.6 et 1.8 montrent que plus la résolution du CAN est importante, plus la plage analogique définie par le quantum q est réduite, diminuant de fait le bruit de quantification : une plus grande résolution engendre un faible bruit de quantification. À partir de l'expression du bruit de quantification, qui est d'un point de vue théorique la seule perturbation apportée au signal d'entrée, nous pouvons définir deux paramètres théoriques représentatifs des performances du CAN en fonction de sa résolution ou de la largeur de son quantum.

1.1.2.3 La plage dynamique

La plage dynamique est le rapport entre la plus grande entrée (PE) et le plus petit pas du convertisseur, soit :

$$DR_{dB} = 20 \log_{10}\left(\frac{PE}{q}\right) \tag{1.9}$$

Cette expression peut être normalisée par rapport à q et s'exprime alors en fonction de la résolution :

$$DR_{dB} = 20 \times res \times \log_{10} 2 \cong 6.02 \times res \tag{1.10}$$

1. En réalité, dans l'expression de $s(n)$, A n'est pas l'amplitude en volt A du signal d'entrée du CAN mais vaut $A \times q$ exprimé en LSB. Pour simplifier les expressions des différents signaux, nous utiliserons le même nom A

2. Le spectre du bruit de quantification s'étend normalement bien au-delà de la fréquence d'échantillonnage. Mais, puisque l'opération de quantification intervient conjointement avec l'échantillonnage, le repliement spectrale intervient pour borner la bande de ce bruit.

1.1.2.4 Le rapport signal à bruit et le nombre effectif de bits

Ce paramètre est un des plus importants pour analyser le comportement du CAN, il reflète la puissance de bruit introduit dans le processus de conversion. A partir de la relation 1.8, il est possible de déterminer le rapport signal sur bruit théorique d'un CAN parfait de résolution res. Pour un signal d'entrée de type sinusoïdale parcourant la pleine échelle du CAN, sa valeur efficace est donnée par :

$$V_{eff} = \frac{2^{res}}{2\sqrt{2}} \tag{1.11}$$

On peut alors définir le rapport signal sur bruit par [4] :

$$(SNR)_{dB}^{Théorique} = 20\ \log_{10}\left\{\frac{V_{eff}}{\sigma_q}\right\} \approx 6.02\ res + 1.76 \quad \text{(dB)} \tag{1.12}$$

L'inversion de cette formulation permet d'aboutir au nombre effectif de bits, à partir naturellement d'un SNR mesuré :

$$n_{eff} = \frac{SNR_{dB}^{Mesuré} - 1.76}{6.02} \leq res \tag{1.13}$$

Le SNR va diminuer si le bruit engendré par le système de conversion est plus important que le bruit de quantification idéal. Ceci aura pour effet de diminuer la résolution effective n_{eff}.

1.2 Paramètres d'erreur des CAN

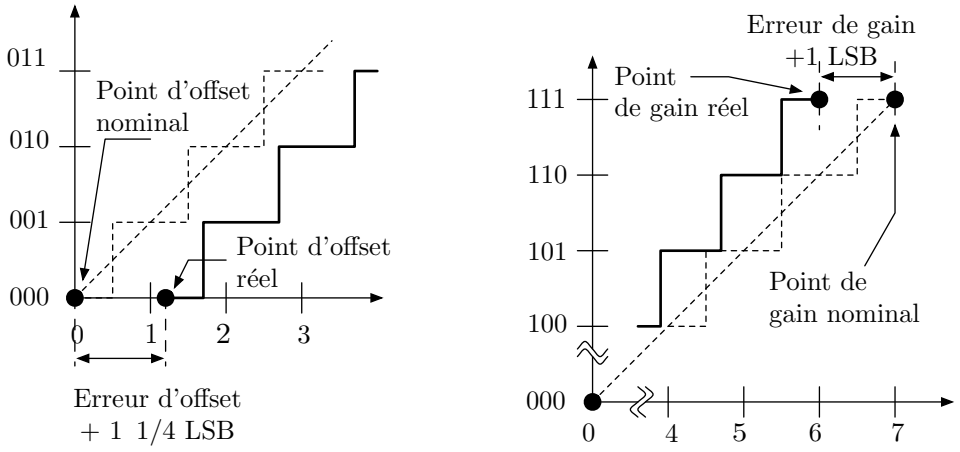
L'opération de quantification induit une erreur systématique, mais un CAN est également caractérisé par ses paramètres d'erreurs. Ces paramètres d'erreurs sont dus aux dispositifs électroniques utilisés pour sa réalisation qui sont, par nature, non idéaux. On va donc constater un décalage entre les performances réelles et idéales d'un CAN. Prenons par exemple le cas d'un comparateur dont la tension de référence est légèrement différente de celle théorique d'un CAN parallèle (flash). On verra alors apparaître une déformation de la caractéristique de transfert qu'il faudra quantifier. C'est la raison pour laquelle il a été défini des termes représentatifs de la variation de la caractéristique de transfert réelle par rapport à celle idéale ou théorique, on parlera alors d'erreurs statiques. Il y a aussi d'autres erreurs à prendre en compte, les erreurs dynamiques qui sont liées au fonctionnement dynamique du CAN. Pour définir ces paramètres de performance, nous avons utilisé la norme JEDEC [5] et la norme IEEE [6].

1.2.1 Paramètres statiques de performance

Les imperfections d'un convertisseur réel influent sur la valeur analogique des seuils de transition de code, modifiant ainsi la fonction de transfert. Les paramètres statiques de performance des CAN permettent de quantifier la déformation de la fonction de transfert du convertisseur par

rapport à celle d'un convertisseur idéal. Trois types d'erreurs statiques sont présentées : l'erreur d'offset, l'erreur de gain et les erreurs de non linéarité différentielle et intégrale.

1.2.1.1 Erreur d'offset



(a) Erreur d'offset

(b) Erreur de gain

FIGURE 1.3 – Erreurs d'offset et de gain sur un CAN unipolaire de résolution égale à 3 bits.

L'erreur d'offset (figure 1.3(a)) est un décalage en tension introduit par le convertisseur sur l'ensemble du signal. Il s'agit d'une constante O additive exprimée en *LSB*. Dans le cas d'un CAN idéal, $O = 0$. Par rapport à l'équation 1.7, la sortie du CAN présentant une erreur d'offset est :

$$s(n) = O + A\cos(2\pi f_0 n T_s + \phi) + e_q(n) \tag{1.14}$$

1.2.1.2 Erreur de gain

L'erreur de gain (figure 1.3(b)) est un changement de la pente de la caractéristique de transfert idéale. Il s'agit d'un constante G multiplicative exprimée en *LSB*. Dans le cas d'un CAN idéal, $G = 1$. Cette erreur ne peut être déterminé qu'après avoir compensé l'erreur d'offset. Par rapport à l'équation 1.7, la sortie du CAN présentant une erreur de gain est :

$$s(n) = G \times A\cos(2\pi f_0 n T_s + \phi) + e_q(n) \tag{1.15}$$

1.2.1.3 Erreur de non linéarité

Les erreurs de non-linéarités reflètent des variations locales, ne pouvant pas s'exprimer de façon linéaire, des seuils analogiques de transition de la caractéristique de transfert. Deux paramètres

de non-linéarité sont définis : la Non Linéarité Différentielle et la Non Linéarité Intégrale. Ces paramètres ne sont déterminés qu'après avoir corrigé les erreurs d'offset et de gain.

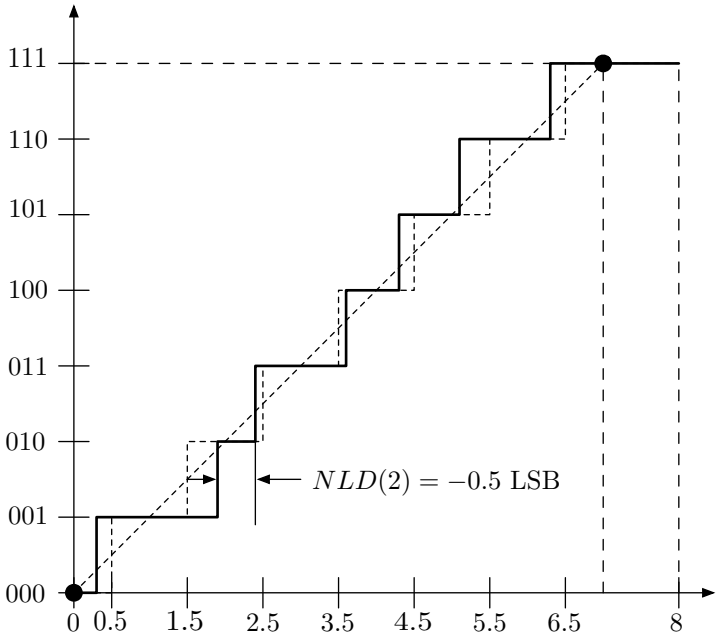


FIGURE 1.4 – Erreur de non linéarité différentielle sur un CAN 3 bits.

Erreurs de non linéarité différentielle C'est un paramètre très important lorsqu'on qualifie des composants dans des applications de mesure. On utilisera le terme NLD ou DNL pour *'Differential Non Linearity'*. Elle correspond à la variation du quantum réel q_i par rapport au quantum théorique q de chaque code (figure 1.4), normalisée par q et exprimée en LSB [7] :

$$NLD(i) = \frac{q_i - q}{q} \tag{1.16}$$

Erreur de non linéarité intégrale C'est également un paramètre très important car il donne une représentation de la non linéarité du composant. Elle mesure la différence entre la caractéristique de transfert réelle et la caractéristique de transfert idéale. On utilisera le terme NLI ou INL pour *'Integral Non Linearity'*. Elle correspond, pour le code i, à la somme cumulée des NLD jusqu'au rang i (figure 1.5) :

$$NLI(i) = \sum_{j \leq i} NLD(j) \tag{1.17}$$

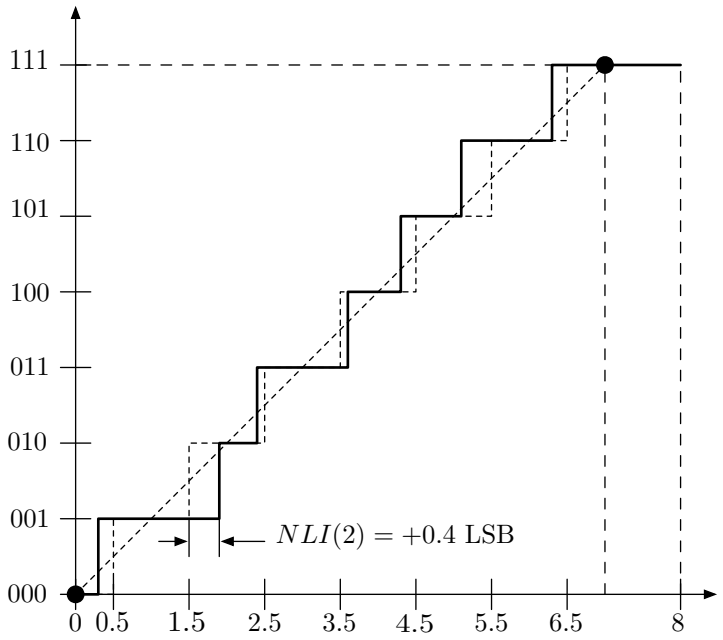


FIGURE 1.5 – Erreur de non linéarité intégrale sur un CAN 3 bits.

1.2.2 Paramètres dynamiques de performance

Les paramètres dynamiques d'un CAN réel représentent les déformations du signal numérique de sortie par rapport au signal appliqué en entrée.

1.2.2.1 Erreurs à l'ouverture

Il existe trois types d'erreurs à l'ouverture dus à l'échantillonnage : l'incertitude à l'ouverture, la gigue à l'ouverture et le retard à l'ouverture [8] (figure 1.6). La figure 1.6(a) représente un échantillonnage idéal. L'incertitude à l'ouverture (figure 1.6(b)) est due à une variation aléatoire du seuil de déclenchement du convertisseur à cause de la pente non infinie de l'horloge. La gigue à l'ouverture (figure 1.6(c)) est due à des variations aléatoire de l'instant d'échantillonnage causé par des sources de bruit (thermique, d'alimentation, d'horloge, etc). Enfin le retard à l'ouverture (figure 1.6(d)) correspond au temps mis par le convertisseur pour répondre à la commande d'échantillonnage dû aux temps de propagation dans les lignes métalliques, au temps de transition des composants et au temps de commutation de l'horloge.

1.2.2.2 Taux de distorsion harmonique

La distorsion harmonique représente l'ensemble des signaux harmoniques générés par la non linéarité du convertisseur sur le signal d'entrée qu'il quantifie. Pour la déterminer, les échantillons

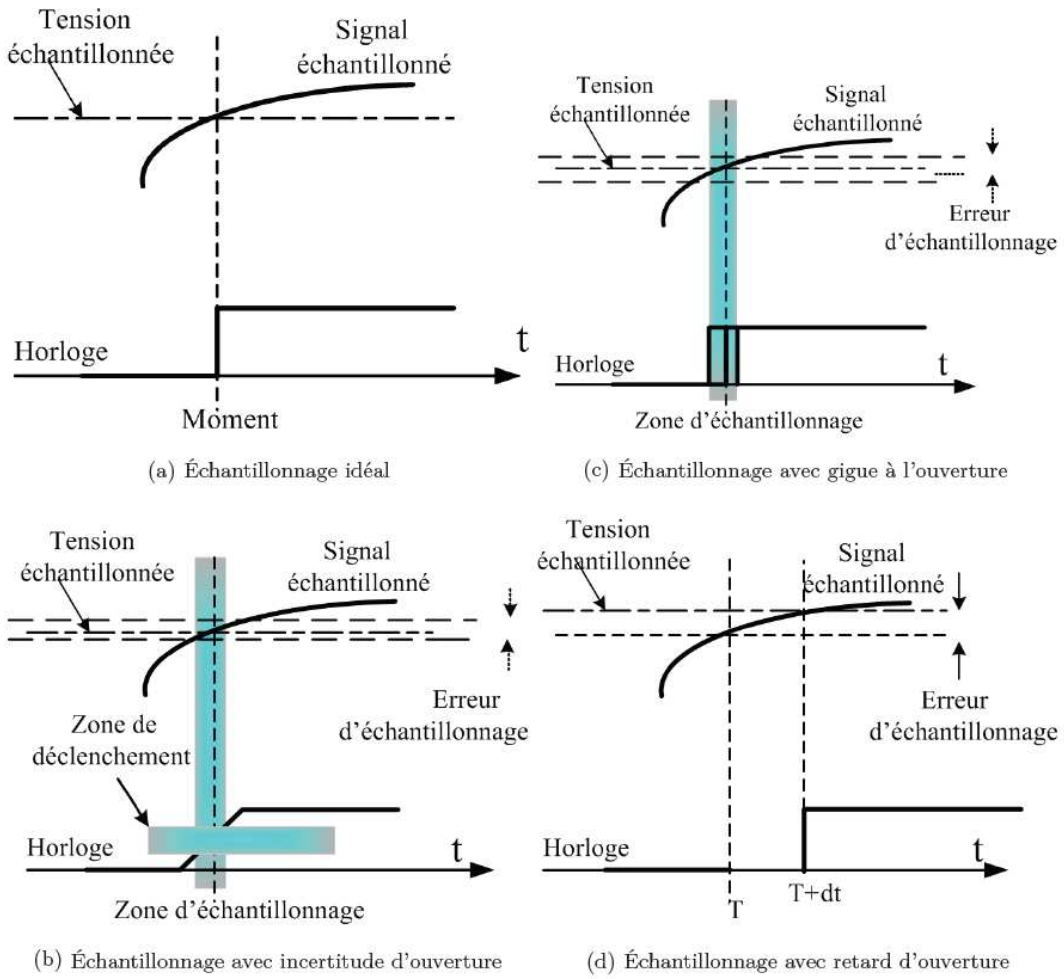


FIGURE 1.6 – Erreurs à l'ouverture [9].

de sortie sont analysés spectralement soit par un analyseur de spectre via un convertisseur numérique analogique (CNA) de résolution suffisante, soit par un traitement numérique des données élaborant une Transformé de Fourier Discrète (TFD). Du spectre de raies obtenu, on peut extraire les raies harmoniques et en déduire le taux de distorsion harmonique (THD) qui est alors la racine carrée de la somme quadratique des amplitudes des raies harmoniques rapportée à l'amplitude du fondamental.

1.2.2.3 Rapport signal à bruit

La quantification d'un signal analogique sur un nombre limité de bits fait que le signal converti présente, par principe, une erreur par rapport au signal d'entrée. Lorsque l'on définit le rapport signal sur bruit (S/B), S correspond à la valeur efficace du signal d'entrée et B à celle du

signal d'erreur global, comprenant l'erreur de quantification plus les erreurs dues au fait que le convertisseur est non idéal. Pour un CAN idéal, la fonction d'erreur est exactement égale à l'erreur de quantification et le rapport signal sur bruit est entièrement déterminé par sa résolution et par la forme du signal d'entrée. Il sert de référence pour estimer les performances du CAN réel. Un CAN réel possède d'autres sources de bruit venant s'ajouter au bruit de quantification, elles sont liées au principe de conversion utilisé ou issues de défauts ponctuels du composant. Le rapport signal sur bruit du CAN réel est comparé avec celui du CAN idéal pour évaluer le bruit dû aux défauts du convertisseur.

1.3 Caractérisation dynamique d'un CAN

Au début des années 1980, Hewlett Packard propose une méthodologie pour la caractérisation dynamique d'un CAN [10]. En effet, auparavant, un signal continu d'amplitude variable était utilisé pour déterminer les niveaux de transition de la caractéristique de transfert, s'apparentant ainsi à un test statique. Cette nouvelle méthodologie fut basée sur l'utilisation d'un signal sinusoïdal balayant la pleine échelle de conversion et dont la fréquence est à l'intérieur du domaine défini par le critère de Nyquist. Le synoptique de base est représenté en figure 1.7

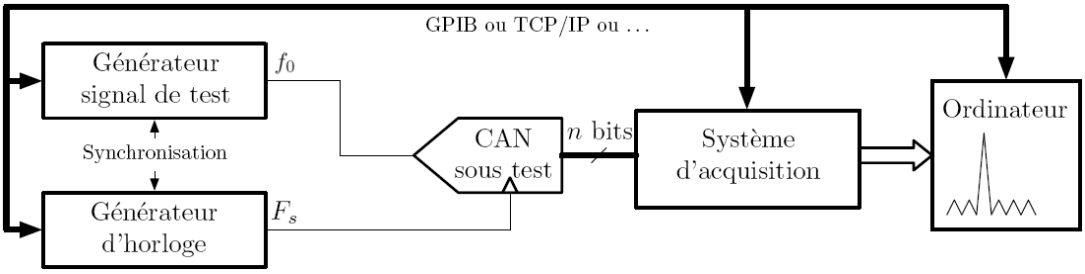


FIGURE 1.7 – Synoptique du banc de test dédié à la caractérisation dynamique d'un CAN.

Lorsque l'on parlera de test dynamique d'un CAN, quelque soit le banc de test utilisé, il faudra essayer de se rapprocher de cette organisation. C'est ainsi le premier élément de comparaison d'un banc de test de CAN. Ensuite, il faudra voir s'il utilise les mêmes traitements que ceux que l'on va énumérer dans ce chapitre et qui sont les algorithmes de base pour l'évaluation des performances d'un CAN [11, 12].
Dans cette partie, nous présentons les méthodes de base à appliquer dans des conditions de test idéales : générateurs de stimuli de qualité spectrale supérieure au CAN sous test, possibilité d'attaquer le convertisseur avec une amplitude égale à la pleine échelle (PE). Nous présentons trois méthodes de test dynamique pouvant être appliquées au test embarqué :
– Le test par analyse temporelle [13][14].

– Le test par analyse statistique [15].
– Le test par analyse spectrale à partir de la DFT (*Discrete Fourier Transform*) ou de la FFT (*Fast Fourier Transform*) [16].

Nous allons dans les paragraphes suivants donner une description plus détaillée de ces méthodes.

1.3.1 Analyse temporelle

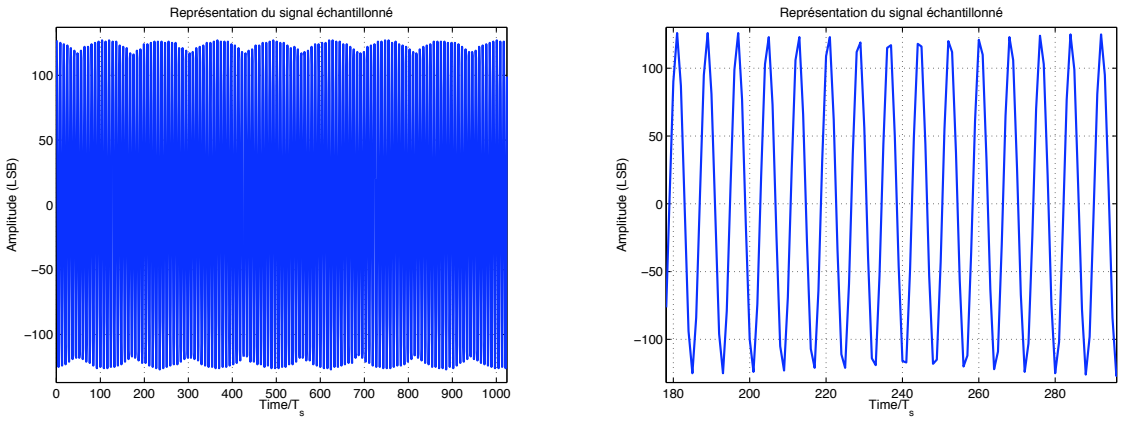
Pour illustrer l'analyse temporelle, nous avons utilisé une acquisition issue d'un CAN simulé de résolution 8 bits. Afin d'assurer un balayage complet des échantillons, l'amplitude du signal de référence est proche de la pleine échelle du CAN. Il a été montré dans [4] que le nombre d'échantillons traités doit vérifier la relation suivante pour assurer la pertinence de l'analyse temporelle :

$$N = 4 \times 2^{res} \tag{1.18}$$

où res est la résolution. De plus, il doit y avoir cohérence entre la fréquence d'échantillonnage et la fréquence du signal d'entrée, soit :

$$N \times T_s = k \times T_{in} \Longleftrightarrow N \times f_0 = k \times F_s,\ N \text{ et } k \text{ étant premiers entre eux.} \tag{1.19}$$

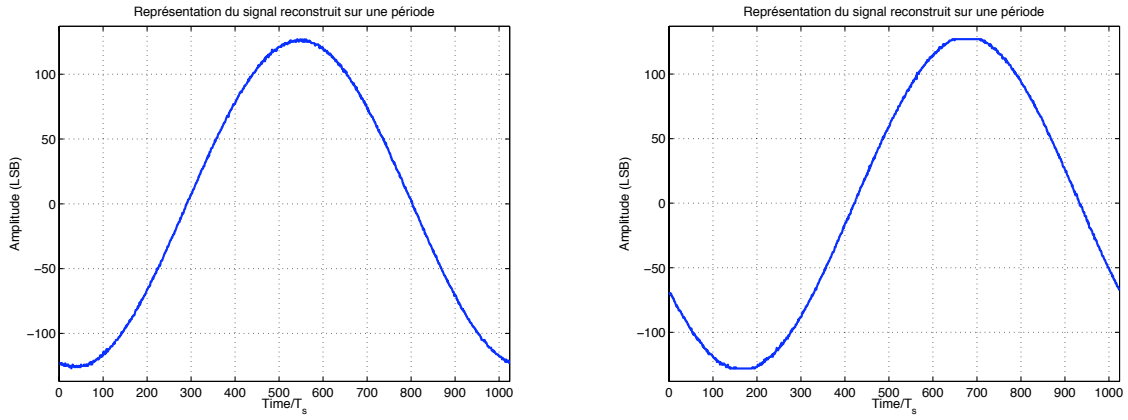
Cela permet d'obtenir un nombre entier de période du signal d'entrée dans l'acquisition, point également utile pour l'analyse spectrale.
La première chose qui apparaît à la sortie du CAN est le **signal échantillonné** que nous noterons $d[i]$. La représentation temporelle de ce signal est simple. On la donne à travers la figure 1.8 :



(a) Signal entier (b) Signal avec zoom

FIGURE 1.8 – Signal échantillonné d'un CAN simulé.

Pour avoir une idée plus précise sur les caractéristiques du banc de test, il est toujours préférable de **reconstruire le signal sur une seule période du signal d'entrée**, comme le montre la figure 1.9(a). En effet, au cas où il y aurait une légère saturation du signal d'entrée, cela apparaîtrait de façon plus évidente sur le signal reconstitué sur une seule période (figure 1.9(b)).



(a) Sans saturation (b) Avec saturation

FIGURE 1.9 – Signal reconstitué sur une période du signal d'entrée.

On peut déterminer, à partir du signal issu du CAN et à l'aide d'une méthode de régression linéaire ou non, la fonction passant par les $d[i]$ avec une erreur quadratique minimale. Le **signal d'erreur** $err[i]$, différence entre le signal de sortie et cette fonction d'interpolation, permet de calculer la valeur efficace du bruit du composant B_{rms}. Une comparaison avec la valeur efficace du bruit de quantification théorique, σ_q, permet l'extraction du nombre de bits effectifs par la relation [6] :

$$n_{eff} = res - \log_2\left(\frac{B_{rms}}{\sigma_q}\right) \tag{1.20}$$

La première étape consiste à déterminer le signal par la méthode des moindres carrés. Le signal d'entrée est de forme sinusoïdale et impose un modèle de même forme. Le critère des moindres carrés s'écrit alors comme la somme des carrés de l'erreur de quantification :

$$J_{err} = \sum_{i=0}^{N-1} \{d[i] - A\sin(2\pi f_0 t[i] + \varphi) - M\}^2 \tag{1.21}$$

où A, M, f_0, et φ sont les paramètres recherchés. $t[i]$ est l'indice temporel correspondant à $i \times T_s$ et N est le nombre de points de l'acquisition. On cherche à minimiser J_{err} de manière à obtenir les meilleures estimations de l'amplitude, de l'offset, de la fréquence et de la phase

$$\left(\hat{A},\ \hat{M},\ \hat{f}_0,\ \hat{\varphi}\right) \tag{1.22}$$

La minimisation de J_{err} passe donc par l'annulation des dérivées partielles et aboutit au système d'équations non linéaires à quatre inconnues suivant $(\frac{\partial J_{err}}{\partial A}, \frac{\partial J_{err}}{\partial M}, \frac{\partial J_{err}}{\partial f_0}, \frac{\partial J_{err}}{\partial \varphi}) = 0$:

$$\left.\begin{array}{l}
\sum_{i=0}^{N-1} d[i]\ \sin(2\pi f_0 t[i]+\varphi) = A\sum_{i=0}^{N-1}\sin^2(2\pi f_0 t[i]+\varphi) + M\sum_{i=0}^{N-1}\sin(2\pi f_0 t[i]+\varphi) \\
\sum_{i=0}^{N-1} d[i] \ = A\sum_{i=0}^{N-1}\sin(2\pi f_0 t[i]+\varphi) + M\ N \\
\sum_{i=0}^{N-1} d[i]\ t[i]\ \sin(2\pi f_0 t[i]+\varphi) = \frac{A}{2}\sum_{i=0}^{N-1} t[i]\sin(2(2\pi f_0 t[i]+\varphi)) + M\sum_{i=0}^{N-1} t[i]\cos(2\pi f_0 t[i]+\varphi) \\
\sum_{i=0}^{N-1} d[i]\ \cos(2\pi f_0 t[i]+\varphi) = \frac{A}{2}\sum_{i=0}^{N-1}\sin(2(2\pi f_0 t[i]+\varphi)) + M\sum_{i=0}^{N-1}\cos(2\pi f_0 t[i]+\varphi)
\end{array}\right\} \tag{1.23}$$

Les travaux effectués sur l'analyse temporelle sont principalement basés sur les méthodes permettant de résoudre ce système [17], [18]. Une méthode parmi les plus couramment utilisées considère la fréquence f_0 issue du générateur connue avec précision, et permet la linéarisation du système (1.23).
Le modèle du signal issu du CAN est écrit sous la forme linéaire suivante :

$$f(t[i]) = A_1\sin(2\pi f_0 t[i]) + A_2\cos(2\pi f_0 t[i]) + M \tag{1.24}$$

avec

$$A_1 = A\cos\varphi \quad \text{et} \quad A_2 = A\sin\varphi \tag{1.25}$$

Le critère des moindres carrés s'écrit alors :

$$J_{err} = \sum_{i=0}^{N-1}\left\{d[i] - A_1\ \sin(2\pi f_0 t[i]) - A_2\ \cos(2\pi f_0 t[i]) - M\right\}^2 \tag{1.26}$$

Il en résulte un système d'équations linéaires dont les inconnues sont A_1, A_2, et M. $\hat{\varphi}$ et $\hat{A}$ sont alors données par :

$$\left.\begin{array}{l}
\hat{\varphi} = \arctan\frac{A_2}{A_1} \\
\hat{A} = \frac{A_1}{\cos\varphi} = \frac{A_2}{\sin\varphi}
\end{array}\right\} \tag{1.27}$$

Une fois que le modèle du signal est déterminé, il convient de déterminer la fonction d'erreur expérimentale, $err[i]$, afin de calculer sa valeur efficace. Ceci est fait à l'aide des relations suivantes :

$$err[i] = d[i] - \left(\hat{A}\ \sin\left(2\pi\hat{f}_{0n}t[i]+\hat{\varphi}\right) - \hat{M}\right) \tag{1.28}$$

et

$$B_{rms} = \sqrt{\frac{\sum_{i=0}^{N-1} err[i]^2}{N}} \tag{1.29}$$

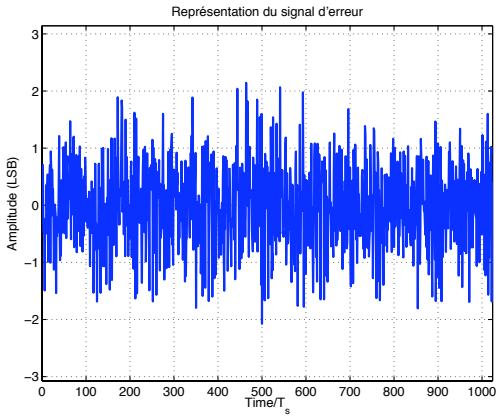


FIGURE 1.10 – Illustration temporelle du signal d'erreur.

Le nombre de bits effectifs peut être calculé à partir du bruit mesuré précédemment [3]. La figure (1.10) donne le signal d'erreur d'un CAN simulé de résolution égale à 8 bits.

1.3.2 Analyse statistique

Il est, avec le test par analyse spectrale, le plus utilisé et le plus étudié [10, 12, 19, 20]. Un signal, dont les propriétés statistiques sont connues, est envoyé à l'entrée du CAN. L'amplitude de ce signal doit couvrir la pleine échelle de manière à exciter tous les codes. Les codes de sortie sont alors répartis selon un histogramme $H(i)$ donnant le nombre d'apparitions pour chaque code. On peut alors exprimer la fréquence d'apparition d'un code i sur une acquisition de N échantillons par :

$$f(i) = \frac{H(i)}{N} \tag{1.30}$$

Si le nombre d'échantillons N de l'acquisition est suffisamment grand ($N \geq 64 \times 2^{res}$) [4], $f(i)$ tend vers la probabilité d'apparition effective $p(i)$ du code i. Pour que cette analyse ait un sens d'un point de vue statistique, il faut que l'acquisition se fasse en fréquences cohérentes [4].

3. Pour que le calcul du nombre de bits effectifs soit valable, il faut que le test soit effectué en pleine échelle et que la fréquence du signal d'entrée ne soit pas liée de façon harmonique à la fréquence d'horloge. Ceci est une précaution à prendre afin d'être sûr de parcourir tous les codes possibles du CAN. C'est pourquoi, il est conseillé de faire le test en fréquences cohérentes. Il faut aussi noter que l'exploitation du signal pour obtenir le bruit B_{rms} annihile intrinsèquement les erreurs de gain et d'offset

4. S'il existe une relation harmonique entre f_0 et F_s, le nombre de codes testés pourrait être inférieur au nombre total de codes du CAN et une périodicité des codes existerait à l'intérieur de l'acquisition. Cela aurait pour effet de faire apparaître des codes de manière privilégiée au détriment de certains autres. De plus, si la périodicité n'était pas entière, l'interprétation de l'histogramme en serait faussée.

En respectant ces conditions et en appliquant une sinusoïde à l'entrée du CAN, nous obtenons un histogramme correspondant à la version discrète de celui d'une sinusoïde [5]. On obtient alors l'histogramme suivant (figure 1.11) :

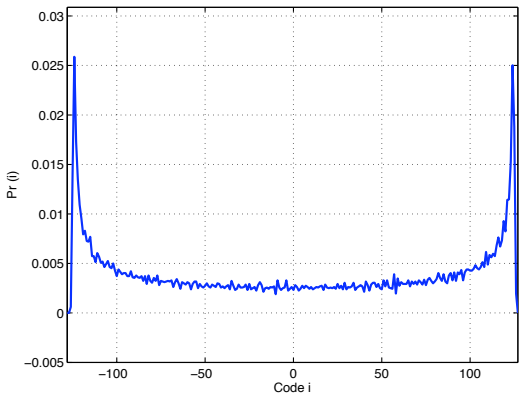


FIGURE 1.11 – Histogramme saturé du signal issu du CAN simulé.

La comparaison avec l'histogramme théorique permet l'extraction de paramètres d'erreurs liés à la caractéristique de transfert. Pour des raisons de précision, le signal d'entrée $V(t)$ de forme sinusoïdale est préféré. Sa densité de probabilité est connue, elle s'exprime par :

$$g(V) = \frac{1}{\pi \sqrt{A^2 - (V - M)^2}} \tag{1.31}$$

où A est l'amplitude et M l'offset du signal.

Il est alors possible de calculer la probabilité théorique d'apparition d'un code i de l'histogramme idéal :

$$p_{th}(i) = \int_{V_t(i)}^{V_t(i+1)} g(V)\, dV = \frac{1}{\pi}\left\{\arcsin\left(\frac{V_t(i+1) - M}{A}\right) - \arcsin\left(\frac{V_t(i) - M}{A}\right)\right\} \tag{1.32}$$

où $V_t(i)$ et $V_t(i+1)$ sont respectivement les tensions de transition des codes i et $i+1$.

Une interprétation qualitative des erreurs liées à la caractéristique de transfert peut être faite. En effet, si un code apparaît plus souvent qu'il ne le doit, le pas de quantification associé est plus large que le quantum théorique, l'inverse exprime le fait qu'il est moins large. Une fréquence d'apparition nulle signifie que ce code est manquant. Cette approche qualitative ne pose pas

5. Pour s'assurer de balayer tous les codes, il est recommandé d'appliquer un signal dont l'amplitude dépasse légèrement la pleine échelle du convertisseur, point que l'on verra ultérieurement

de problème lorsque la résolution du CAN est faible ou lorsque les erreurs sont grossières. Ce n'est plus le cas actuellement et il est impératif de procéder à une étude quantitative des erreurs liées à la caractéristique de transfert par la mesure de la non-linéarité différentielle ($NLD(i)$) et de la non-linéarité intégrale ($NLI(i)$) de chaque code.Deux méthodes sont proposées pour leur mesure : l'une préconise la comparaison entre les probabilités théoriques et expérimentales des codes i (utilisation directe de l'histogramme) [10],[12], [21], l'autre utilise la fonction cumulée d'apparition des codes pour évaluer les tensions de transition expérimentales de la caractéristique de transfert [19], [20].

1.3.2.1 Utilisation directe de l'histogramme

L'histogramme idéal correspondant à l'acquisition est construit à partir des résultats expérimentaux. L'amplitude A et l'offset M en LSB sont alors estimés par :

$$\left.\begin{array}{l} \hat{A} = \frac{i_{max}+1-i_{min}}{2} \\ \hat{M} = \frac{1}{i_{max}+1-i_{min}} \sum_{i=i_{min}}^{i_{max}} i \end{array}\right\} \tag{1.33}$$

où i_{min} et i_{max} sont respectivement les codes testés ayant une valeur de $p(i)$ maximale dans le cas d'un signal d'entrée sinusoïdal. Pour un CAN non signé $i_{min} = 0$ et $i_{max} = 2^{res} - 1$, s'il est signé $i_{min} = -2^{(res-1)}$ et $i_{max} = 2^{(res-1)} - 1$, lorsque le signal d'entrée balaye parfaitement la pleine échelle du CAN sous test.
Il convient donc de donner une autre expression de $p_{th}(i)$ pour des valeurs de $\hat{A}$ et $\hat{M}$ données en LSB. Pour cela, il faut donner les tensions de transition théoriques en LSB afin d'être en accord sur les unités par $V_t(i) = i - 0.5$ LSB ($V_t(0) = 0$ LSB et $V_t(2^{res}) = 2^{res}$ LSB). On obtient alors :

$$\left.\begin{array}{l} p_{th}(i) = \frac{1}{\pi}\left\{\arcsin\left(\frac{i+0.5-\hat{M}}{\hat{A}}\right) - \arcsin\left(\frac{i-0.5-\hat{M}}{\hat{A}}\right)\right\} \quad ; \quad i \in [1, 2^n - 2] \\ p_{th}(0) = \frac{1}{\pi}\left\{\arcsin\left(\frac{0.5-\hat{M}}{\hat{A}}\right) - \arcsin\left(-\frac{\hat{M}}{\hat{A}}\right)\right\} \\ p_{th}(2^n - 1) = \frac{1}{\pi}\left\{\arcsin\left(\frac{2^n-\hat{M}}{\hat{A}}\right) - \arcsin\left(\frac{2^n-1.5-\hat{M}}{\hat{A}}\right)\right\} \end{array}\right\} \tag{1.34}$$

Non-linéarité différentielle (NLD) Il est alors possible de donner une expression mathématique de la $NLD(i)$ (définie par l'équation 1.16) à partir des fonctions densité de probabilité théorique et expérimentale d'un code i [21].

$$NLD(i) = \frac{p(i) - p_{th}(i)}{p_{th}(i)} \tag{1.35}$$

Un code i est considéré manquant lorsque la probabilité $p(i)$ est inférieure à $p_{th}(i)$ d'un pourcentage généralement égal à 90% .

$$NLD\,(i) < -0.9 \tag{1.36}$$

Non-linéarité intégrale (NLI) L'influence des erreurs de gain et d'offset ne doit pas intervenir dans le calcul de la NLI représentative de la distorsion harmonique (norme JEDEC [5]). La méthode la plus utilisée pour minimiser les phénomènes dus aux erreurs de gain et d'offset consiste à définir la meilleure droite passant à travers le vecteur NLI obtenu à l'aide de la relation 1.17. Une régression linéaire, effectuée sur le vecteur NLI, consiste à minimiser l'expression :

$$J_{NLI} = \sum_{i=i_{\min}}^{i_{\max}} \{NLI(i) - (a.i + b)\}^2 \tag{1.37}$$

où a et b définissent la droite recherchée.

La minimisation de J_{NLI} passe par le calcul de ses dérivées partielles par rapport à a et b, et conduit à un système d'équations linéaires dont les solutions sont :

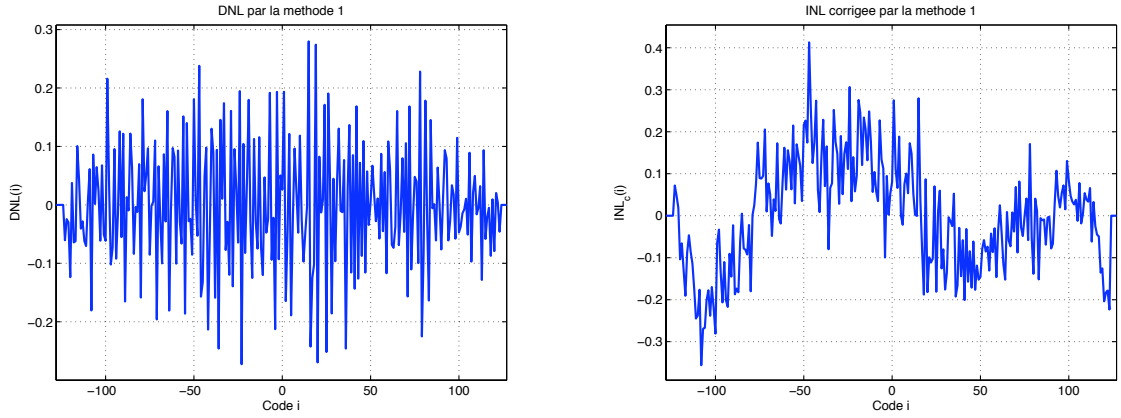
$$\left.\begin{aligned} \hat{a} &= \frac{\sum_{i=i_{\min}}^{i_{\max}} i.NLI(i) - \frac{1}{(i_{\max}-i_{\min}+1)}\left\{\left(\sum_{i=i_{\min}}^{i_{\max}} i\right)\left(\sum_{i=i_{\min}}^{i_{\max}} NLI(i)\right)\right\}}{\sum_{i=l}^{i_{\max}} i^2 - \frac{1}{(i_{\max}-i_{\min}+1)}\left(\sum_{i=i_{\min}}^{i_{\max}} i\right)^2} \\ \hat{b} &= \frac{\frac{1}{(i_{\max}-i_{\min}+1)}\left[\left(\sum_{i=i_{\min}}^{i_{\max}} i^2\right)\left(\sum_{i=i_{\min}}^{i_{\max}} NLI(i)\right) - \left(\sum_{i=i_{\min}}^{i_{\max}} i\right)\left(\sum_{i=i_{\min}}^{i_{\max}} i.NLI(i)\right)\right]}{\sum_{i=i_{\min}}^{i_{\max}} i^2 - \frac{1}{(i_{\max}-i_{\min}+1)}\left(\sum_{i=i_{\min}}^{i_{\max}} i\right)^2} \end{aligned}\right\} \tag{1.38}$$

On obtient alors un vecteur de non-linéarité intégrale corrigé en accord avec la définition donnée dans la norme JEDEC sous l'appellation anglaise Best-Straight-Line [6]. On l'appellera NLI_c(i) :

$$NLI_c(i) = NLI(i) - \left(\hat{a}.i + \hat{b}\right) \tag{1.39}$$

Les paramètres a et b déterminés pourront être considérés comme l'erreur de gain et l'erreur d'offset. Il faut toutefois remarquer que ces paramètres peuvent varier en fonction de l'étendue du domaine testé de la caractéristique de transfert et qu'il serait hasardeux de les considérer comme les valeurs "vraies" d'erreurs de gain et d'offset.

6. Une alternative à cette définition consiste à déterminer les paramètres a et b en traçant la droite passant par les valeurs de la NLI des codes extrêmes (relatif à la définition End-Point de la norme JEDEC).



(a) NLD (b) NLI corrigée

FIGURE 1.12 – NLD et NLI calculées par la méthode de comparaison des histogrammes.

1.3.2.2 Utilisation de l'histogramme cumulé

Cette méthode d'analyse utilise la fonction de répartition (ou fonction de probabilité cumulée) des codes i issus du CAN sous test. Une expression analytique des tensions de transition des codes i à l'aide de cette fonction peut être donnée. En effet, grâce à la fonction $p_{th}(i)$, la fonction de répartition des codes i, $Pr_{th}(i)$, s'écrit :

$$\Pr_{th}(i) = \sum_{j \leq i} p_{th}(j) = \int_{M-A}^{V_t(i+1)} \frac{1}{\pi\sqrt{A^2 - (V-M)^2}} dV \tag{1.40}$$

La résolution de cette intégrale donne l'expression de la probabilité cumulée d'un code i en fonction de sa tension de transition supérieure $V_t(i+1)$.

$$\Pr_{th}(i) = \frac{1}{\pi} \arcsin\left\{\left(\frac{V_t(i+1) - M}{A}\right) + \frac{1}{2}\right\} \tag{1.41}$$

La probabilité cumulée $Pr(i)$ issue d'un histogramme expérimental permet donc la détermination de la caractéristique de transfert réelle du CAN par le calcul des niveaux de transition :

$$V_t(i) = -\hat{A}.\cos\left(\pi.\Pr(i-1)\right) + \hat{M} \quad \text{(LSB)} \quad avec \quad \Pr(i) = \sum_{j \leq i} p(j) \tag{1.42}$$

Rappelons que V_t(i), $\hat{A}$, et $\hat{M}$ sont normalisées par rapport au quantum q et qu'à ce titre, ils peuvent être exprimés en LSB.

De l'histogramme, on tire alors l'histogramme cumulé ainsi que la caractéristique de transfert expérimentale qui en résulte.

Non-linéarité différentielle (NLD) La NLD(i) est calculée à partir des niveaux de transition :

$$NLD(i) = \frac{V_t(i+1) - V_t(i)}{q_{ref}} - 1 = \{V_t(i+1) - V_t(i)\} - 1 \quad \text{(LSB)} \quad ; \quad q_{ref} = 1LSB \quad (1.43)$$

Les valeurs de A et M extraites de l'histogramme sont des valeurs estimées. La non-linéarité différentielle est donc obtenue de façon directe par :

$$NLD(i) = \hat{A}.\{\cos(\pi \Pr(i-1)) - \cos(\pi \Pr(i))\} - 1 \quad (1.44)$$

Non-linéarité intégrale (NLI) On calcule la non-linéarité intégrale en cherchant la meilleure droite passant par la caractéristique de transfert. Cette droite sera donc déterminée là aussi par la méthode des moindres carrés (régression linéaire). L'expression de l'erreur quadratique utilisée pour la détermination des paramètres de cette droite est :

$$J_{V_t(i)} = \sum_{i=i_{\min}}^{i_{\max}} \{V_t(i+1) - (c.i + d)\}^2 \quad (1.45)$$

L'utilisation du critère des moindres carrés consiste à minimiser $J_{V_t(i)}$ par rapport à c et d. Leurs expressions analytiques sont identiques à celles établies précédemment avec le terme $NLI(i)$ remplacé par $V_t(i+1)$.La détermination de la NLI(i) est alors faite par la relation suivante :

$$NLI(i) = \frac{V_t(i+1) - \left(\hat{c}.i + \hat{d}\right)}{q_{ref}} \quad avec \quad q_{ref} = 1\text{LSB} \quad (1.46)$$

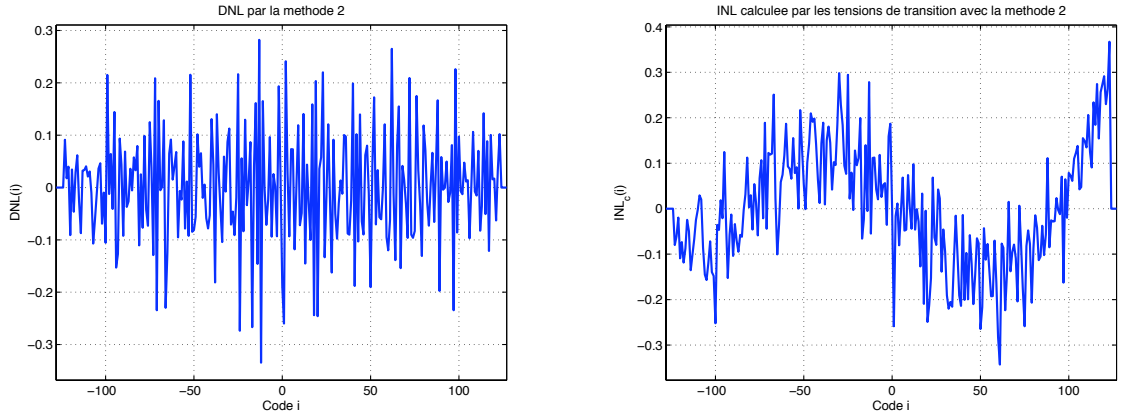
Cette définition correspond à celle donnée par la norme JEDEC sous l'appellation anglaise "Best-Straight-Line" [7]. Avec les mêmes restrictions que celles émises précédemment, c et d peuvent être considérés respectivement comme le gain et l'erreur d'offset du CAN.

1.3.2.3 Utilisation de la caractéristique de transfert normalisée

Une autre méthode d'analyse statistique, développée dans les travaux de Vanden Bossche [22], consiste à utiliser la caractéristique de transfert normalisée. Ce type d'analyse a été mis en oeuvre pour éliminer les problèmes liés à l'estimation de A et M. Par rapport à la méthode précédente, la seule différence réside en un changement de variable modifiant l'expression de $V_t(i)$. On obtient alors une caractéristique de transfert dont les niveaux d'entrée sont normalisés entre $+1$ et -1. Les niveaux de transition sont maintenant normalisés par rapport à l'amplitude estimée $\hat{A}$ et centrés sur la valeur moyenne estimée $\hat{M}$

$$V_{tn}(i) = \frac{V_t(i) - \hat{M}}{\hat{A}} = -\cos(\pi \Pr(i-1)) \quad (1.47)$$

7. De même, une droite passant par les niveaux de transitions des codes extrêmes peut être utilisée pour calculer les paramètres c et d (End-Point Definition selon la norme JEDEC)



(a) NLD (b) NLI corrigée

FIGURE 1.13 – NLD et NLI calculées par la méthode des histogrammes cumulés.

Les calculs de la NLD et de la NLI se font alors selon les mêmes principes décrits auparavant. Le quantum de référence, permettant l'expression de la NLD et la NLI en LSB, est donné à partir des valeurs mesurées des $V_{tn}(i)$, par la relation suivante :

$$q_{ref} = \frac{V_{tn}\left(i_{\max}\right) - V_{tn}\left(i_{\min}+1\right)}{i_{\max} - \left(i_{\min}-1\right)} \tag{1.48}$$

1.3.3 Analyse spectrale

L'analyse spectrale consiste à appliquer au signal issu du CAN un algorithme de Transformation de Fourier Discrète : en général, il s'agit de la FFT développée par Cooley-Tuckey. De même que pour l'analyse temporelle, il doit y avoir cohérence entre la fréquence d'échantillonnage et la fréquence du signal d'entrée (cf. équation 1.19). Comme le signal de test est de forme réelle, il suffira de traiter la partie paire du spectre.

À partir du spectre, il est possible d'identifier la position des raies harmoniques et du fondamental pour déterminer :

- le rapport signal sur bruit sans distorsion harmonique SNR,
- le rapport signal sur bruit avec distorsion harmonique $SINAD$ ou $SNDR$,
- la plage dynamique libre de toute distorsion harmonique $SFDR$,
- le taux de distorsion harmonique THD
- ...

Pour définir ces paramètres, il est important de reprendre la formulation théorique de la trans-

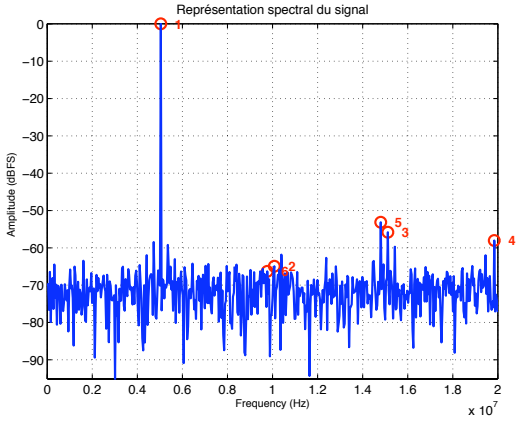


FIGURE 1.14 – Partie paire du module du spectre du signal échantillonné.

formation de Fourier discrète (TFD) qui s'exprime sous la forme :

$$X(n) = \sum_{k=0}^{N-1} x(k) \times \exp\left\{-j\ 2\pi k \frac{n}{N}\right\} \quad ; \quad n \in [\![0, N-1]\!]. \tag{1.49}$$

où les $x(k)$ sont les valeurs des échantillons de l'acquisition. En prenant pour N une puissance de 2, les composantes $X(n)$ de la TFD peuvent s'obtenir par un algorithme de transformation de Fourier rapide (TFR ou FFT) comme celui de Cooley. Ces N composantes définissent un spectre dont la résolution est :

$$\Delta f_{sp} = \frac{F_s}{N} \tag{1.50}$$

L'étude du comportement du CAN se fait alors par l'analyse du spectre d'amplitude (module de $X(n)$) ou du spectre de puissance (module au carré de $X(n)$). Les spectres d'amplitude et de puissance sont pairs, on limite donc l'étude du spectre à l'intervalle $[0, N/2]$. Pour éviter l'utilisation de fenêtre de pondération, les acquisitions doivent être faites en fréquences cohérentes de manière à ce que la séquence $\{x(k)\}$ soit périodique sans discontinuité. Dans le cas contraire (fréquences non cohérentes), l'utilisation d'une fenêtre de pondération (Blackmann-Harris, Hanning, Haming, Kaiser-Bessel, ...) est obligatoire si on veut éviter le phénomène de traînage (leakage) qui rend l'exploitation du spectre impossible [23]. Cependant dans ce cas les mesures sont entachées d'une erreur due au bruit propre de la fenêtre utilisée, difficile à séparer du signal lui-même. Des études sur ce problème ont été faites en considérant l'influence d'une fenêtre sur le rapport signal sur bruit théorique d'un CAN [4],[24], [25].

Les paramètres relatifs au fonctionnement du CAN peuvent être donnés dans les deux cas de figure à condition d'être capable de séparer les différents bruits et harmoniques du signal. Tout ce

qui est énoncé dans les paragraphes suivants concerne une acquisition en fréquences cohérentes qui ne nécessite pas l'utilisation d'une fenêtre.

1.3.3.1 Rapport signal sur bruit théorique

La relation 1.8 définie le rapport signal sur bruit théorique d'un CAN parfait de résolution res pour un signal d'entrée de type sinusoïdal parcourant la pleine échelle du CAN. Mais il est fort probable que l'amplitude crête à crête du signal utilisé lors du test ne couvre pas exactement la pleine échelle PE. Dans ces conditions, il est recommandé de faire la mesure du rapport $2A/PE$ en prenant les valeurs des codes max et min exprimées en LSB du signal de sortie. On trouve alors [4] :

$$SNR_{dB} = 6.02\ res + 1.76 + 20\ \log_{10}\left\{\frac{2A}{PE}\right\} \quad \text{(dB)} \tag{1.51}$$

1.3.3.2 Rapport signal sur bruit avec distorsion harmonique et nombre de bits effectifs

Pour calculer le nombre de bits effectifs par la méthode spectrale, il faut extraire la raie fondamentale et la raie continue, puis considérer les autres comme représentatives de l'énergie du bruit. Toutefois, il est utile de spécifier deux types de rapport signal sur bruit, l'un sans distorsion harmonique que nous dénommerons SNR, l'autre avec, appelé $SINAD$. Ils sont respectivement calculés par :

$$SNR_{dB} = 10\ \log_{10}\left\{\frac{X^2\left(f_0\right) - B_M^2}{\sum\limits_{i\neq k,h.k} X^2\left(i.\Delta f_{sp}\right)}\right\} \tag{1.52}$$

et

$$SINAD_{dB} = 10\ \log_{10}\left\{\frac{X^2\left(f_0\right) - B_M^2}{\sum\limits_{i\neq k} X^2\left(i.\Delta f_{sp}\right)}\right\} \tag{1.53}$$

où B_M est le plancher de bruit, valeur quadratique moyenne du bruit, réparti dans l'ensemble du spectre :

$$B_M^2 = \frac{\sum\limits_{i=1,i\neq k}^{N/2-1} X^2\left(i.\Delta f_{sp}\right)}{\frac{N}{2}-2} \tag{1.54}$$

Lorsque le signal d'entrée est pleine échelle, le nombre de bits effectifs s'obtient à partir du $SINAD_{dB}$ par la relation :

$$n_{eff} = \frac{SINAD_{dB} - 1.76}{6.02} \tag{1.55}$$

Par contre, lorsque l'amplitude est inférieure à la pleine échelle, il faut modifier la valeur du $SINAD_{dB}$ mesurée par la relation suivante [4] :

$$(SINAD_{dB})_{\text{mod}} = (SINAD_{dB}) - 10\ \log_{10}\left\{(SINAD_{dB})^2\ THD^2\left(1-\left(\frac{2A}{PE}\right)^2\right)+\left(\frac{2A}{PE}\right)^2\right\} \tag{1.56}$$

1.3.3.3 Plage dynamique libre de toute distorsion harmonique

La plage dynamique libre de toute distorsion harmonique correspond à l'intervalle de puissance où seul le fondamental est présent. Elle est définie comme le rapport entre l'amplitude du fondamental et l'amplitude maximale des composantes harmoniques ou non observées dans la bande de Nyquist. La valeur du $SFDR$ est mesurée par la relation :

$$SFDR_{dB} = 10\ \log_{10}\left\{\frac{X^2(f_0)}{\max\limits_{f\neq f_0} X^2(f)}\right\} \tag{1.57}$$

1.3.3.4 Taux de distorsion harmonique et plancher de bruit

La moitié de la puissance du signal apparaît dans les $N/2$ composantes du spectre espacées de $\Delta f_{sp} = F_s/N$. Compte tenu de la cohérence des fréquences, la raie fondamentale est située à $k \times \Delta f_{sp}$. Les autres composantes traduisent les défauts du CAN puisque non présentes dans le signal d'entrée. Parmi ces composantes, les raies de distorsion harmonique traduisent la non-linéarité de la caractéristique de transfert. Elles se situent dans le spectre aux abscisses $h \times f_0 = h \times k \times \Delta f_{sp}$ avec $h \geq 2$ entier. Le taux de distorsion harmonique est défini par :

$$THD_{dB} = 10\ \log_{10}\left\{\frac{\sum\limits_h \left(X^2(h.f_0) - B_M^2\right)}{X^2(f_0) - B_M^2}\right\} \tag{1.58}$$

La raie continue ($i = 0$) n'est pas prise en compte dans cette caractérisation purement dynamique, ni la raie fondamentale (i = k) qu'on ne peut dissocier de la composante de bruit à cette fréquence. Il devrait en être de même pour les raies harmoniques qu'on ne peut dissocier, mais l'erreur sur l'estimation de B_M^2 reste faible car ces raies harmoniques sont très peu nombreuses devant N/2.

Dans le cas où seul le bruit de quantification intervient, il se répartit sur tout le spectre à un niveau situé par rapport à la pleine échelle à :

$$(B_M)_{dB} = -6.02 \times res - 10\log_{10} N + 1.25 \tag{1.59}$$

1.4 Tests intégrés

Au début des années 90, Ohletz [26] est l'un des premiers a proposer une méthode de test de circuits mixtes, analogiques et numériques sur puce. En effet la structure de test suivante fut proposée (cf. figure 1.15) : Le circuit sous test est alors déconnecter du système dans lequel il

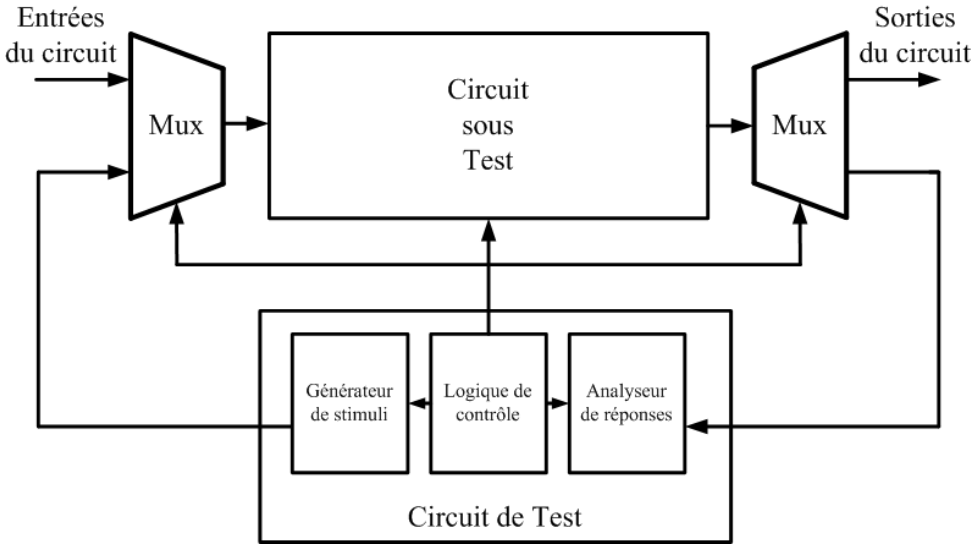


FIGURE 1.15 – Structure de la méthode de test embarqué BIST.

est inclut par des multiplexeurs en entrée et sortie. Un signal de test est alors généré en entrée du circuit sous test puis la réponse de celui-ci est analysée pour déterminer si le circuit est fautif ou non. Le générateur et l'analyseur de réponse constitue le circuit de test.
Nous allons maintenant présenter la structure du circuit de test d'Ohletz et sa stratégie de BIST : le HBIST. Nous détaillerons ensuite différentes stratégies de BIST appliquées aux CAN ayant été proposées dans la littérature. Ainsi, à partir d'une comparaison de ces différentes stratégies, nous pourrons aboutir à une solution de BIST qui nous semble la plus optimale.

1.4.1 HBIST

Une des premières stratégies pour le BIST de circuits analogiques et mixtes fut le BIST Hybride (Hybrid Built-In Self-Test, HBIST) proposée par Ohletz en 1991 [26] et Damm en 1995 [27]. La figure 1.16 décrit la structure du circuit de test utilisée.
Elle consiste à générer un signal pseudo-aléatoire (bruit blanc) grâce un registre à décalage avec rétroaction linéaire (Linear-Feedback Shift Register, LFSR). Ce signal est traité, puis converti en signal analogique et envoyé au circuit sous test. La réponse de celui-ci est ensuite compressée en une signature numérique. Le calcul de cette signature s'effectue grâce à un registre

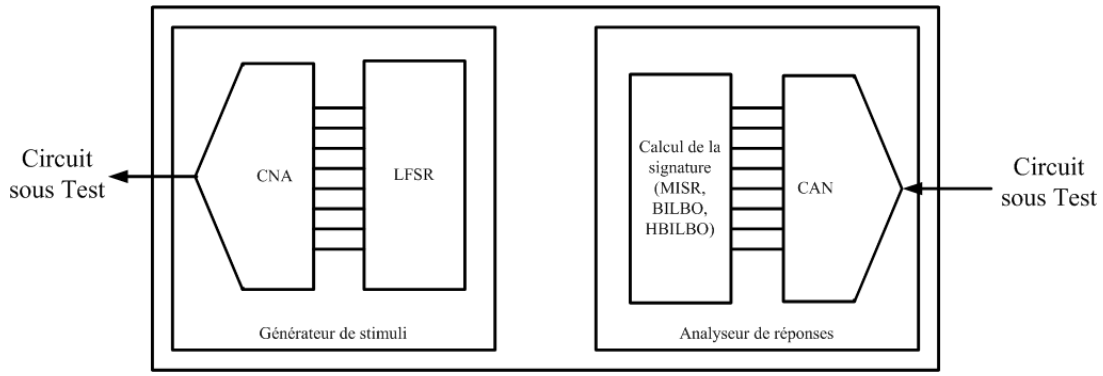


FIGURE 1.16 – Structure du circuit de test de la stratégie HBIST.

particulier (Built-In-Logic-Block-Observer, BILBO [26], Hybrid-Built-In-logik-Block-Observer, HBILBO [27] ou Multiple Input Signature Register, MISR [28], ce qui va permettre de classifier le circuit comme fonctionnel ou défaillant par comparaison à une signature théorique (détection de fautes). Cette méthode nécessite l'utilisation d'un convertisseur numérique analogique (CNA) qu'il faut au préalable tester et ajouter si un CNA n'est pas disponible dans le système. De plus, la signature calculée dépend de nombreux facteurs comme la température environnante lors du test ou les défauts inhérents aux circuits analogiques. Il faut donc définir un intervalle dans lequel la signature du circuit sera considérée comme correcte sans savoir si deux déviations opposées de la réponse de test se seront ou non compensées, entraînant une diminution du taux de fautes détectées.

1.4.2 OBIST

Une autre stratégie, appelée OBIST [29] consiste à forcer le circuit sous test à osciller. La fréquence d'oscillation est alors liée aux paramètres structurels et fonctionnels du circuit. La déviation de la fréquence d'oscillation par rapport à la valeur nominale indique un circuit fautif. Cette méthode a été appliquée avec succès sur un grand nombre de circuits analogiques et mixtes [30, 31, 32] dont les CAN [33, 34]. Ainsi, la stratégie OBIST introduit le CAN dans une boucle et le force à osciller autour de codes prédéterminés (cf. figure 1.17). Une méthode standardisée pour garantir des oscillations entretenues et robustes est proposée dans [35, 36]. Elle utilise une boucle de rétroaction non linéaire composée d'un comparateur (quantificateur 1 bit) et d'un CNA de résolution 1 bit. La particularité de cette stratégie est qu'elle ne nécessite pas de générer des signaux de test. Elle nécessite en revanche des moyens pour mesurer l'amplitude et la fréquence des oscillations avec une grande précision [37, 38, 39]. Vazquez [38] propose l'utilisation d'un modulateur $\Sigma\Delta$ pour convertir la sortie du CAN en un bitstream puis d'utiliser des compteurs pour déterminer la fréquence, l'amplitude et la partie constante du signal oscillant. Une première

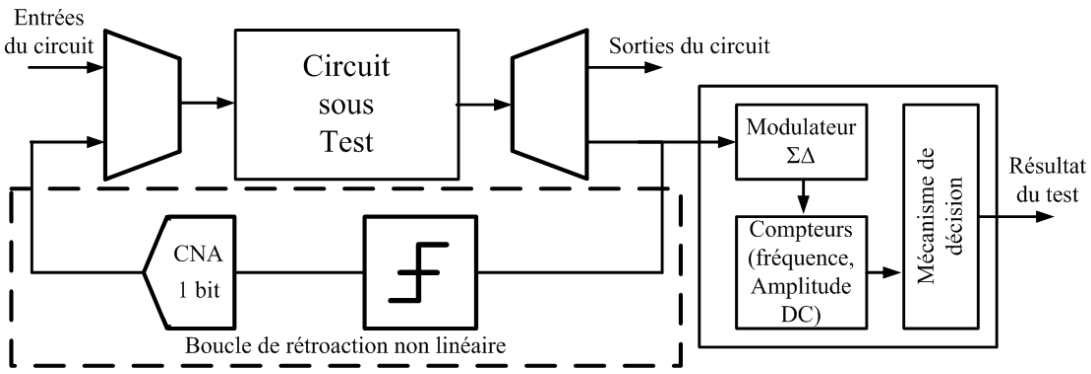


FIGURE 1.17 – Structure de la méthode de test OBIST.

méthode de test vérifie si ces valeurs appartiennent à des domaines d'acceptabilité prédéfinis dépendant du circuit sous test. Ces mesures permettent également de déterminer le temps de conversion et les non linéarités du CAN. Le principal avantage de cette stratégie est l'absence de stimulus et donc il n'y a pas besoin d'un circuit de génération de signal de test. Par contre, le circuit d'analyse est très complexe et nécessite beaucoup de précision.

1.4.3 HABIST

Le BIST par histogramme (Histogram-based Analog Built-in Self Test, HABIST) est l'une des stratégie de BIST la plus étudiée [40, 41, 42, 43, 44, 45]. Cette méthode de test est basée sur l'étude de l'histogramme expérimental construit à partir des codes de sortie du convertisseur sous test. Le stimulus doit être un signal couvrant le pleine échelle. Il peut être une sinusoïde, une rampe ou un signal triangulaire. La figure 1.18 donne le schéma général de l'implémentation de la méthode HABIST :

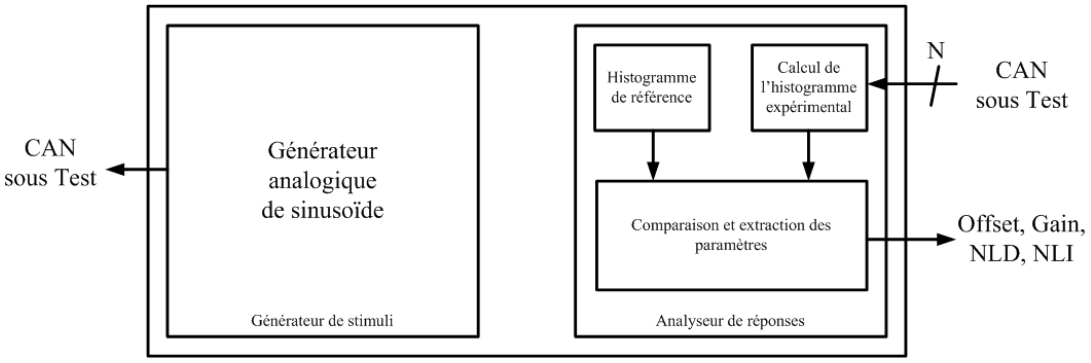


FIGURE 1.18 – Structure du circuit de test de la stratégie HABIST.

Cette stratégie nécessite deux étapes distinctes. La première étape consiste à construire, à par-

tir de nombreuses simulations, un gabarit pour l'histogramme de référence en tenant compte des dispersions permises sur les paramètres fonctionnels à tester. Puis dans un second temps, l'histogramme expérimental est réalisé et est comparé au gabarit cible pré-calculé à la première étape. Enfin, si l'histogramme expérimental ne rentre pas dans le gabarit, alors le convertisseur associé est déclaré défectueux [40]. La stratégie HABIST permet également, grâce à des algorithmes de calcul supplémentaires, la mesure de paramètres statiques de convertisseurs, tels que le gain, l'offset [44] et les non-linéarités [45]. Les inconvénients majeurs de cette stratégies sont la construction du gabarit pour l'histogramme ainsi que la génération d'un signal analogique correspondant aux signaux utilisés lors de la construction du gabarit. Enfin, pour que cette stratégie soit efficace, il est nécessaire de l'appliquer sur un très grand nombre de points de test.

1.4.4 BIST par LSB (LBIST)

La stratégie proposée par De Vries [46, 47] consiste à générer une rampe lente en entrée du convertisseur et à évaluer le temps entre chaque commutation du bit de poids faible du CAN. Comme le signal d'entrée est linéaire, chaque intervalle de temps est directement proportionnel à la largeur du palier correspondant de la fonction de transfert. L'ensemble de ces intervalles de temps permet de reconstruire la fonction de transfert du convertisseur et donc de déterminer le gain, l'offset et les non linéarités du convertisseur. La figure 1.19 décrit la structure du circuit de test de la méthode LBIST : Un premier test fonctionnel est réalisé en comparant les $n-q-1$

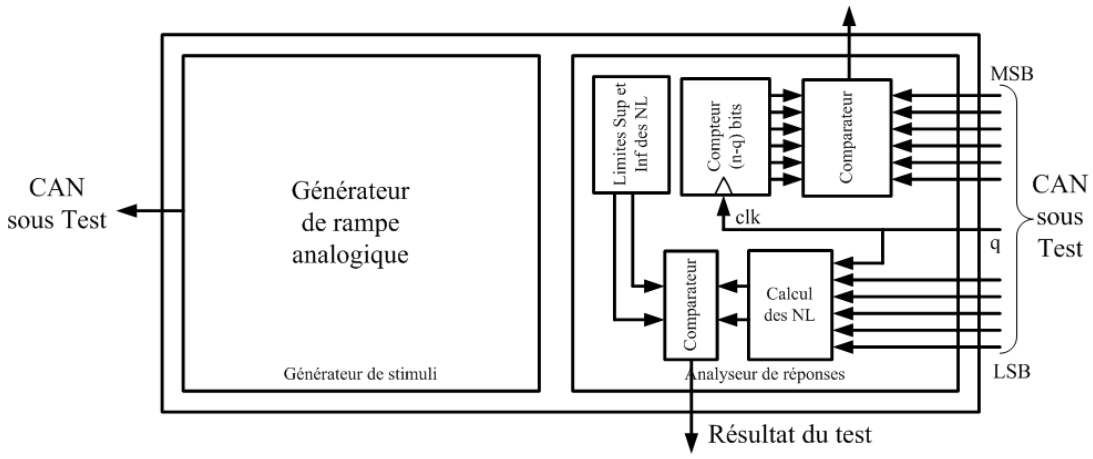


FIGURE 1.19 – Structure du circuit de test de la méthode de test LBIST.

bits de poids fort avec un compteur incrémenté à chaque passage de 1 à 0 du $q^{ième}$ bit issu du CAN. Si ces $n-q-1$ bits sont égaux alors le convertisseur passe le test fonctionnel. Un deuxième niveau de test sur puce peut également être réalisé : à partir de la largeur des différents paliers de la fonction de transfert du CAN (temps de commutation du bit de poids faible du CAN),

les non linéarités (différentielle et intégrale) peuvent être calculées et comparées à des domaines d'acceptabilité. Pour mesurer la durée écoulée entre deux commutations du LSB, un compteur est incrémenté à chaque front du signal d'échantillonnage. Plus la fréquence d'échantillonnage est grande par rapport au signal d'entrée, plus la technique de test est précise. Ces deux tests nécessitent peu de ressources, seulement des compteurs et des comparateurs mais en contrepartie, le signal rampe généré doit être parfaitement linéaire.

1.4.5 BIST par polynôme (PBIST)

Sunter et Nagi [48, 49] propose d'évaluer directement sur la puce la caractéristique de transfert du convertisseur à l'aide d'un polynôme du 3ème ordre en utilisant la méthode des moindres carrés. Le signal de test utilisé est une rampe. La sortie du CAN doit évoluer de sa valeur minimale ($-2^{res-1} = -n/2$) à sa valeur maximale ($2^{res-1} - 1 = n/2 - 1$) en passant par toutes les valeurs intermédiaires. Quatre sommes (S_0, S_1, S_2 et S_3) de sorties sont alors calculées suivant la valeur de l'entrée qui se situe respectivement entre $[-n/2, -n/4[$, entre $[-n/4, 0[$, entre $[0, n/4[$ et entre $[n/4, n/2[$ (figure 1.20).

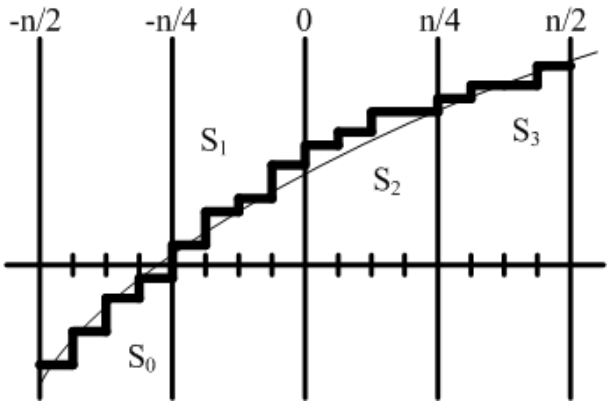


FIGURE 1.20 – Fonction de transfert du CAN montrant les quatre sommes calculées.

A partir de ces quatre sommes, l'offset, le gain et la distortion harmonique du second et troisième harmonique peuvent être facilement calculés par des additions/soustractions ainsi que des décalages, du moins pour l'offset et le gain, le calcul des distortions dépendant du gain calculé. Les opérations nécessaires à cette estimation sont relativement simples et peuvent donc être facilement implantées sur la puce. La connaissance de la courbe de transfert (sortie du CAN en fonction de l'entrée) permet de reconstruire la caractéristique de transfert du CAN et par conséquent de déterminer la valeur de l'erreur de l'offset, de l'erreur de gain et des non-linéarités du convertisseur sous test.

1.4.6 MADBIST

Une autre proposition de stratégie est le MADBIST (*Mixed Analog-Digital BIST*) [50, 51, 52] (figure 1.21). Cette méthode utilise un DSP et requiert l'existence d'un CNA, préalablement testé, dans le circuit. Le DSP analyse la réponse du circuit sous test, au moyen d'une TFD. Ce test, basé sur l'analyse du spectre du signal de sortie, permet le calcul de paramètres dynamiques tels que le taux de distorsion harmonique THD ou le rapport signal sur bruit avec distorsion $SINAD$. Le signal de test est quand à lui généré par un oscillateur numérique dont la sortie est convertie par le CNA.

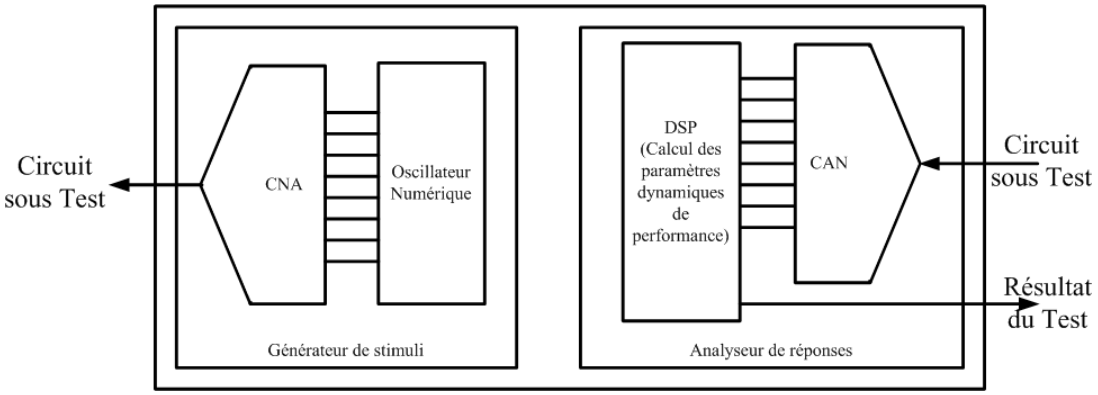


FIGURE 1.21 – Structure de la méthode de test MADBIST.

1.4.7 Comparaison des stratégies de BIST de la littérature

La table 1.1 résume les caractéristiques des différentes stratégies de BIST de la littérature :

	Méthode d'analyse	Génération de stimuli	Spécialisation du circuit de test
HBIST	détection de fautes	× (CNA)	maximale
OBIST	détection de fautes		forte
HABIST	statistique	×	forte
LBIST	statistique	×	faible
PBIST	statistique		faible
MADBIST	spectrale	× (CNA)	faible

TABLE 1.1 – Résumé des caractéristiques des différentes stratégies de BIST de la littérature.

Il existe donc trois méthodes d'analyse utilisées dans ces différentes stratégies de test embarqué : l'analyse statistique (HABIST, LBIST, PBIST), l'analyse spectrale (MADBIST) et la détection de fautes (HBIST, OBIST). La majorité de ces stratégies nécessite l'utilisation d'un signal de test (HBIST, HABIST, LBIST, PBIST) impliquant l'ajout d'un circuit de génération de stimuli. De plus, ces stratégies n'ont pas le même degré de généricité par rapport au circuit à tester. Les stratégies HBIST, HABIST et OBIST sont très spécialisées, que ce soit au niveau des prétraitements comme le calcul d'une signature (HBIST, OBIST) ou d'un gabarit cible (HABIST), uniques pour chaque CAN. Les autres stratégies sont plus souples. En effet, un circuit développé pour tester un CAN à partir des stratégies LBIST ou MADBIST peut être directement utilisé pour un modèle de CAN considéré et ne nécessite pas d'adaptation pour chaque circuit produit. Les contraintes que nous nous imposons dans le cadre du test embarqué d'un CAN sont :

– un surcoût du test (en terme d'occupation et de consommation) minimal,
– une structure de BIST indépendante du CAN à tester,
– un test qui détermine si la qualité spectrale du signal issu du CAN est suffisante pour l'application envisagée.

A partir de ces contraintes, la seule stratégie de BIST de la littérature se rapprochant le plus est la stratégie MADBIST. Malheureusement, celle-ci n'est pas utilisable. En effet, l'utilisation d'un CNA et d'un circuit pour le calcul de la TFD d'un signal ne permettent pas d'avoir un surcoût faible. Nous allons donc proposer une nouvelle structure satisfaisant à ces contraintes.

1.4.8 Structure de BIST proposée

L'utilisation de l'analyse fréquentielle est choisie pour extraire des paramètres spectraux et déterminer si ceux-ci appartiennent au domaine de validité de fonctionnement du CAN. C'est en effet la seule méthode permettant de mesurer la qualité spectrale d'un CAN. Nous nous inspirerons des travaux précédemment réalisés au sein de l'équipe [53]. Pour effectuer une analyse spectrale, un signal de test doit être généré. Ce signal sera généré directement *on-chip*. Pour l'analyse du signal issu du CAN, nous utiliserons un banc de filtres séparant ses différentes harmoniques pour estimer les différents paramètres dynamiques de performance (SNR, $SINAD$, n_{eff}, $SFDR$...). La structure de BIST proposée dans cette étude est illustrée par la figure 1.22 : Les blocs fonctionnels nécessaires sont :

– L'**Oscillateur** $\Sigma\Delta$ qui est un oscillateur numérique utilisant un modulateur $\Sigma\Delta$ passe bas, passe bande ou passe haut pour générer un bitstream (cf. chapitre 2).
– Le **Filtre Analogique** (du même type que le modulateur $\Sigma\Delta$ de l'oscillateur) qui est un filtre passif convertissant le bitstream précédemment généré en un signal analogique qui est ensuite envoyé vers le CAN.

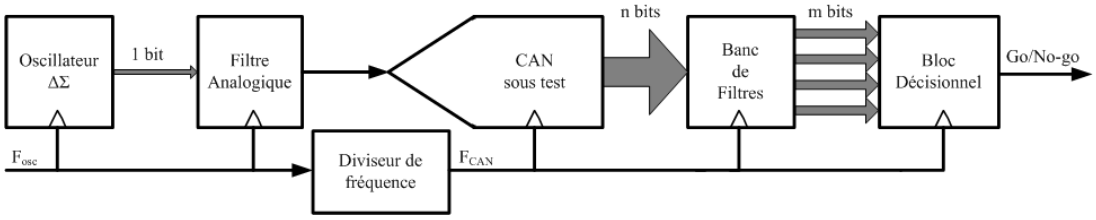


FIGURE 1.22 – Structure de BIST proposée.

- Le **Banc de Filtres** qui est le banc de filtres rejecteur de bande séparant les différentes harmoniques du signal issu du CAN pour l'analyse spectrale du CAN (cf. chapitre 3) .
- Le **Bloc décisionnel** qui calcule le SNR et le compare aux spécifications du CAN pour délivrer la décision de savoir si le CAN fonctionne correctement dans une plage de performances prédéfinie.
- Le **Diviseur de Fréquence** qui permet d'avoir deux fréquences d'horloge multiples l'une de l'autre et synchrones.

1.5 Conclusion

Dans ce chapitre, nous avons rappelé le principe de la conversion analogique numérique et les méthodes de caractérisation en considérant les paramètres d'erreurs des CAN et leurs influences sur leurs performances.

Dans un premier temps, nous avons présenté le principe de la conversion analogique numérique à travers l'échantillonnage et la quantification. Puis, nous avons introduit les différents paramètres d'erreur des CAN : les paramètres de performance statistiques (erreur d'offset, de gain, de non-linéarité différentielle et intégrale) et les paramètres de performance dynamiques (erreurs à l'ouverture, SNR, THD,...).

Les principales méthodes permettant la caractérisation d'un CAN ont ensuite été exposées : l'analyse temporelle, l'analyse statistique et enfin l'analyse spectrale. Ces méthodes d'analyse sont données pour des conditions de test idéales (amplitude crête à crête du signal d'entrée égale à la pleine échelle du convertisseur, signaux parfaits sans bruit et sans distorsion). Chaque type d'analyse permet de déterminer différents paramètres d'erreur du convertisseur.

Enfin, la dernière partie du chapitre est consacrée à la présentation des différentes stratégies de test intégré proposées dans la littérature. Ces stratégies utilisent principalement deux des mé-

thodes d'analyse présentées : l'analyse statistique et l'analyse spectrale. A partir d'une analyse de ces différents stratégies, nous avons proposé une nouvelle stratégie de test.

Durant cette thèse, nous avons cherché à proposer une technique de BIST appliquée aux CAN à bas coût qui soit la plus générique possible dans le sens où elle ne se limite pas à une classe spécifique de convertisseur. Des techniques basées sur l'utilisation d'un oscillateur numérique basé sur un modulateur $\Sigma\Delta$ pour la génération de stimuli et l'utilisation d'un banc de filtres numériques pour extraire les différents paramètres spectraux ont été préférées aux autres méthodes proposées dans la littérature sur le BIST appliqué aux CAN.

Chapitre

2 Génération de signaux In-situ

Sommaire

Introduction

Dans ce deuxième chapitre, nous exposerons la première des deux parties essentielles de la structure BIST proposée : la génération du signal in-situ. Pour cette stratégie de BIST, l'oscillateur

est d'une grande importance puisqu'il doit fournir un signal analogique suffisamment précis pour le test des CAN (qualité spectrale supérieure à la résolution du CAN sous test). Cet oscillateur doit être programmable (fréquence d'oscillation connue) et répétitif (spécifications du signal généré identique d'un circuit à un autre).

Les oscillateurs analogiques sont capables de générer des signaux sinusoïdaux stables avec une grande qualité spectrale, mais ils nécessitent des éléments difficilement intégrables (inductance, quartz). De plus, la fréquence d'oscillation est difficile à ajuster (incertitude sur la valeur des composants constituant l'oscillateur), ce qui rend les oscillateurs analogiques difficilement utilisables pour le BIST.

De plus, la conception des oscillateurs analogiques est vulnérable aux variations dans les processus technologiques ainsi qu'aux fluctuations de la température. Bien que ces facteurs puissent être minimisés par des solutions intelligentes lors de la conception, leurs présences causent des imperfections indésirables dans le cas de la caractérisation des CAN. À l'inverse, les performances des circuits numériques sont beaucoup moins affectées par ces facteurs et plus résistant à leurs variations.

C'est pourquoi nous nous sommes intéressés à la génération numérique du signal de test. Ce chapitre commence par la présentation d'un générateur de sinusoïdes analogiques à partir d'un oscillateur numérique et de son inconvénient majeur : la nécessité d'utiliser un CNA pour obtenir une forme analogique. Puis une solution sera étudiée : la modulation $\Sigma\Delta$. Les générateurs numériques $\Sigma\Delta$ de sinusoïdes analogiques de la littérature seront ensuite présentés dans une seconde partie. Enfin dans une dernière partie, l'étude et l'implémentation du générateur sélectionné sera abordée et ses performances seront détaillées.

2.1 Oscillateur numérique avec modulation $\Sigma\Delta$

2.1.1 Générateur de sinusoïdes analogiques

La première approche consiste à utiliser un oscillateur numérique pour générer un signal numérique, qui pourra ensuite être converti sous un format analogique par un CNA (figure 2.1).

Cet oscillateur consiste en un résonateur numérique du second ordre, comprenant deux intégrateurs à temps discret. Il peut être vu comme deux systèmes rebouclés dont les fonctions de

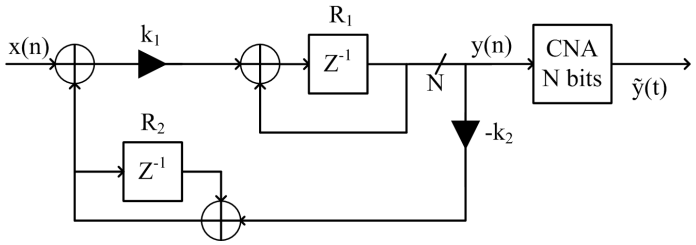


FIGURE 2.1 – Oscillateur numérique simple du second ordre.

transfert sont :

$$H_1(z) = k_1 \frac{z^{-1}}{1 - z^{-1}} \tag{2.1}$$

et

$$H_2(z) = -k_2 \frac{1}{1 - z^{-1}} \tag{2.2}$$

La fonction de transfert entre l'entrée $X(z)$ et la sortie $Y(z)$ de ce système est donc :

$$H(z) = \frac{H_1(z)}{1 - H_1(z)H_2(z)} = \frac{k_1 z^{-1}(1 - z^{-1})}{(1 - z^{-1})^2 + k_1 k_2 z^{-1}} \tag{2.3}$$

On concidère ce système bouclé comme un oscillateur lorsqu'il est possible d'obtenir une fréquence pour laquelle il fonctionne, même avec une entrée nulle. Pour cela il faut satisfaire le critère de Barkhausen, qui impose deux conditions [54] :

1. le gain de la boucle entière doit être égal à 1,
2. la phase de la boucle entière doit être un multiple de 2π.

Pour notre système, la première condition se formule ainsi :

$$|H_1(z)H_2(z)| = \left| -k_1 k_2 \frac{z^{-1}}{(1 - z^{-1})^2} \right| = 1 \tag{2.4}$$

c'est à dire qu'il existe une fréquence pour laquelle le gain de la boucle entière doit être égal à 1. A cette fréquence f_0, le système va être auto-entretenu, même avec une entrée nulle ($x(n) = 0, \forall n$). En résolvant l'équation 2.4, il vient que :

$$\Omega_0 = \frac{2\pi f_0}{F_s} = \arccos\left(1 - \frac{k_1 k_2}{2}\right) \tag{2.5}$$

où Ω_0 est la pulsation normalisée, f_0 la fréquence d'oscillation du système et F_s la fréquence d'échantillonnage.

Le signal généré par ce système est un signal sinusoïdal de pulsation Ω_0, de phase ϕ et d'amplitude A. La phase est l'amplitude de ce signal dépendent des valeurs initiales des registres R_1 et R_2,

respectivement notées $R_1(0)$ et $R_2(0)$. Pour déterminer la valeur de ces caractéristiques, on écrit d'abord la transformée en Z du signal de sortie :

$$Y(z) = \frac{(1-z^{-1})R_1(0) + k_1 z^{-1} R_2(0)}{1-(2-k_1k_2)z^{-1}+z^{-2}} \tag{2.6}$$

Puis le signal $y(n)$, sous forme temporelle, est obtenue par transformée en Z inverse :

$$y(nT_s) = R_1(0)\frac{\sqrt{2(1-\cos\Omega_0)}}{\sin\Omega_0}\sin\left(\Omega_0 nT_s + \frac{\pi+\Omega_0}{2}\right) + R_2(0)\frac{1}{\sin\Omega_0}\sin\left(\Omega_0 nT_s\right) \tag{2.7}$$

$y(nT_s)$ s'écrit donc sous la forme d'une sinusoïde d'amplitude A, de pulsation Ω_0 et de phase ϕ tel que [55] :

$$y(nT_s) = A \times \sin\left(\Omega_0 nT_s + \phi\right) \tag{2.8}$$

où :

$$\Omega_0 = \frac{2\pi f_0}{F_s} = \arccos\left(1-\frac{k_1k_2}{2}\right) \tag{2.9}$$

$$\phi = \arctan\left(\frac{R_1(0)\sin(\Omega)}{(1-k_1k_2-\cos(\Omega))R_1(0)+k_1R_2(0)}\right) \tag{2.10}$$

$$A = \frac{(1-k_1k_2)R_1(0)+k_1R_2(0)}{\sin(\Omega)+\phi} \tag{2.11}$$

Dans le cadre du BIST, cet oscillateur présente deux importants défauts : le coût en surface des multiplications $N \times N$ par k_1 et k_2 et le CNA multibit en sortie. En utilisant une puissance de 2 pour le coefficient k_1, la multiplication par k_1 consiste en un décalage, réduisant ainsi le coût en surface. Il existe cependant une méthode pour réduire encore plus la surface occupée par ce générateur de signal : la modulation $\Sigma\Delta$. En utilisant la propriété du codage sur 1 bit des modulateurs $\Sigma\Delta$ [56], le multiplieur $N \times N$ peut être remplacé par un multiplexeur 2 : 1 commandé par la sortie d'un modulateur $\Sigma\Delta$, comme le montre la figure 2.2. La sortie du multiplieur était un signal sur $2N$ bits qu'il fallait alors troquer alors que le signal de sortie du multiplexeur est un signal sur N bits, ne nécessitant donc aucune modification. Par contre, cela se fera au détriment d'une fréquence de travail beaucoup plus élevée.

Ainsi, un bitstream est obtenu en sortie de l'oscillateur numérique. Celui-ci sera ensuite traité par un filtre analogique pour obtenir le signal de test. Le circuit qui fait usage d'un modulateur $\Sigma\Delta$ dans la boucle de réaction d'un résonateur numérique est appelée désormais oscillateur $\Sigma\Delta$ passe-bas [55] (figure 2.3). Ce système utilise deux intégrateurs numériques, un modulateur $\Sigma\Delta$ passe bas et un multiplexeur. La sortie du modulateur $\Sigma\Delta$ est nommée $Y(z)$. Le modulateur $\Sigma\Delta$ code son entrée en un flux binaire et le multiplexeur effectue la multiplication 1 bit par N bits.

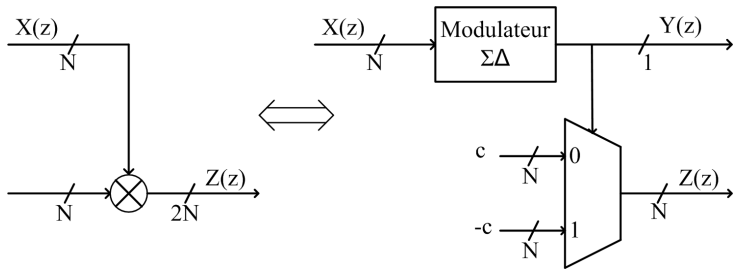


FIGURE 2.2 – Simplification du multiplieur $N \times N$ d'un signal X par une constante c, à l'aide d'un modulateur $\Sigma\Delta$ et d'un multiplexeur $2:1$.

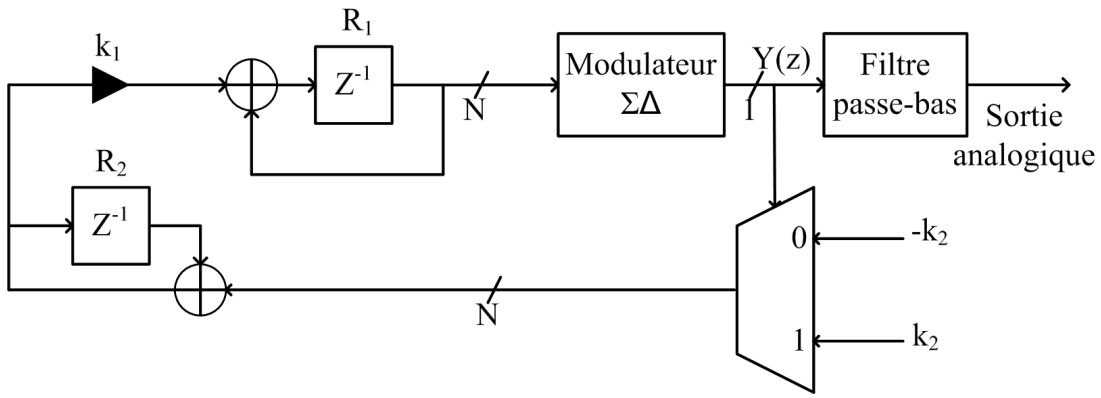


FIGURE 2.3 – Oscillateur $\Sigma\Delta$ passe bas.

La seule restriction à l'utilisation d'une telle simplification est l'obligation d'avoir une fonction de transfert associée au signal (STF) du modulateur $\Sigma\Delta$ égale à l'unité et de ne pas présenter de changement de phase dans la bande du signal. Le circuit peut supporter aussi un modulateur $\Sigma\Delta$ ayant une STF égale à z^{-1} mais le résonateur doit être alors modifié en échangeant l'intégrateur retardé $R1$ (figure 2.1) par un intégrateur non retardé [55].
Il va donc falloir déterminer des structures de modulateur $\Sigma\Delta$ qui garantissent ces exigences tout en ayant un signal généré de qualité spectrale supérieure à celle du CAN. En effet, si par exemple, le SNR du signal d'entrée est inférieur à celui du CAN, le signal issu du CAN aura, non pas un SNR au niveau de celui du CAN, mais au niveau de celui du signal généré. Ainsi, dans notre cas, le SNR du signal généré doit être supérieur à 72 dB, SNR théorique d'un convertisseur ayant une résolution de 12 bits.

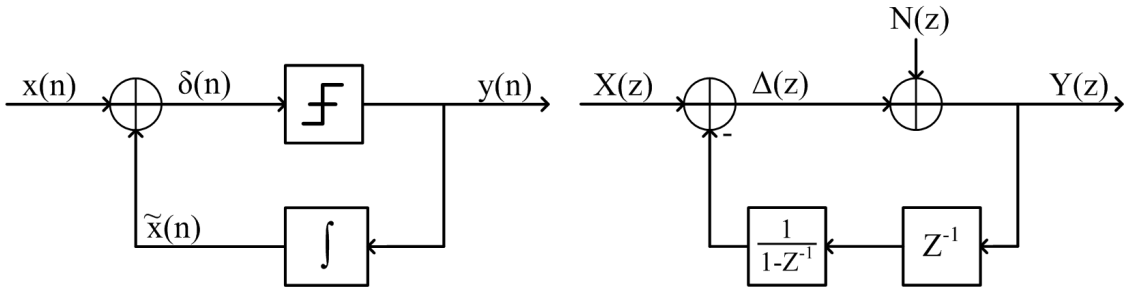
2.2 Modulation $\Sigma\Delta$

La modulation $\Sigma\Delta$ fait partie des modulations numériques différentielles qui quantifient non pas la valeur instantanée du signal d'entrée mais la différence entre cette valeur et une autre valeur

estimée par extrapolation des valeurs du signal aux instants précédents. De plus, et c'est là le principal intérêt de la modulation ΣΔ, elle permet d'affaiblir le bruit de quantification dans la bande utile en le repoussant hors de la bande. C'est ce que l'on appelle la mise en forme du bruit ou noise shaping [57, 58]. Avant de détailler la modulation ΣΔ, nous allons revenir sur la modulation Δ, qui est en quelque sorte, le point de départ de la modulation ΣΔ.

2.2.1 Modulation Δ

Le schéma de principe de la modulation Δ est donné à travers la figure 2.4(a). $\delta(n) = x(n) - \tilde{x}(n)$ est défini comme la différence entre le signal d'entrée $x(n)$ et une estimation de celui-ci $\tilde{x}(n)$. Cette estimation est obtenue en intégrant $y(n)$, quantification sur un bit de la différence $\delta(n)$. $y(n)$ est donc un train binaire indiquant le signe de $\delta(n)$ et son intégration produit un signal $\tilde{x}(n)$ en marche d'escalier qui suit les variations de $x(n)$. La différence $\delta(n)$ évolue alors, dans son fonctionnement normal, dans une plage $[-q, +q]$ où q est défini comme la hauteur d'intégration.



(a) Schéma de principe (b) Modèle linéaire

FIGURE 2.4 – Schéma de principe et modèle linéaire du modulateur Δ.

La figure 2.4(b) décrit le modèle linéaire du modulateur Δ dans l'espace des fréquences. Le bruit de quantification est modélisé par un bruit blanc additif *N(z)*. *X(z)*, *Y(z)* et *Δ(z)* sont respectivement les transformées en Z des signaux temporels *x(n)*, *y(n)* et *δ(n)*, et sont données par :

$$Y(z) = \Delta(z) + N(z) \tag{2.12}$$

$$\Delta(z) = X(z) - \frac{z^{-1}}{1 - z^{-1}} Y(z) \tag{2.13}$$

Si l'on remplace $\Delta(z)$ par son expression dans celle de *Y(z)*, on obtient alors :

$$Y(z) = X(z) - \frac{z^{-1}}{1 - z^{-1}} Y(z) + N(z) \tag{2.14}$$

$$= (1 - z^{-1})X(z) + (1 - z^{-1})N(z) \tag{2.15}$$

En définissant la *STF(z)* et la *NTF(z)* comme étant respectivement la fonction de transfert associée au signal et la fonction de transfert associée au bruit, il vient par identification :

$$STF(z) = 1 - z^{-1} \tag{2.16}$$

$$NTF(z) = 1 - z^{-1} \tag{2.17}$$

La mise en forme du bruit est réalisée par un dérivateur, mais ce même dérivateur déforme également le signal. Or le modulateur ne doit modifier que le bruit de quantification, sans que le signal d'entrée ne soit modifié d'un point de vue fréquentiel. Cela veut dire que la *STF(z)* doit être constante. Pour ce faire, un intégrateur est inséré sur le chemin du signal d'entré avant l'opérateur différence. Ainsi, seul *x(n)* est codé et non pas sa dérivée. De plus, l'opération d'intégration étant linéaire, les deux intégrateurs situé avant l'opérateur différence peuvent être déplacés après cet opérateur. Le modulateur obtenu est alors appelé modulateur ΣΔ du premier ordre.

2.2.2 Modulation ΣΔ du premier ordre

La figure 2.5 décrit le modèle linéaire du modulateur ΣΔ du premier ordre :

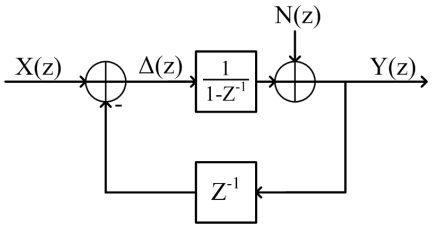


FIGURE 2.5 – Modèle linéaire du modulateur ΣΔ du premier ordre.

En utilisant la même méthode d'analyse que pour le modulateur Δ, les fonctions de transfert associées respectivement au signal et au bruit sont :

$$STF(z) = 1 \tag{2.18}$$

$$NTF(z) = 1 - z^{-1} \tag{2.19}$$

Ainsi, le signal d'entrée n'est pas modifié et seul le bruit est mis en forme par un dérivateur. Le carré du module de la *NTF* s'exprime alors, dans le domaine des fréquences, par :

$$|NTF(f)|^2 = 4\sin^2\left(\pi\frac{f}{F_s}\right) \tag{2.20}$$

La figure 2.6 montre l'effet d'une modulation ΣΔ de type passe-bas. Ce concept de modulation ΣΔ fut breveté par Cutler en 1954 [59]. Nombreux furent ceux qui apportèrent des modifications

et des améliorations dans les années qui suivirent, mais le changement le plus substantiel fut proposé par Ritchie en 1977. Il proposa d'inclure plusieurs intégrateurs en cascade pour augmenter l'ordre du modulateur [60].

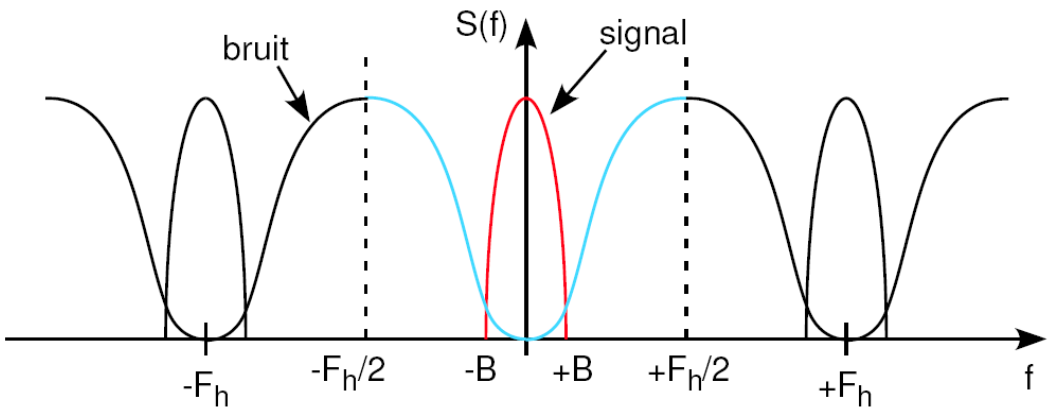


FIGURE 2.6 – Spectre du signal après mise en forme du bruit.

2.2.3 Modulation ΣΔ d'ordre supérieur

La figure 2.7 décrit le modèle linéaire à temps discret du modulateur ΣΔ d'ordre k :

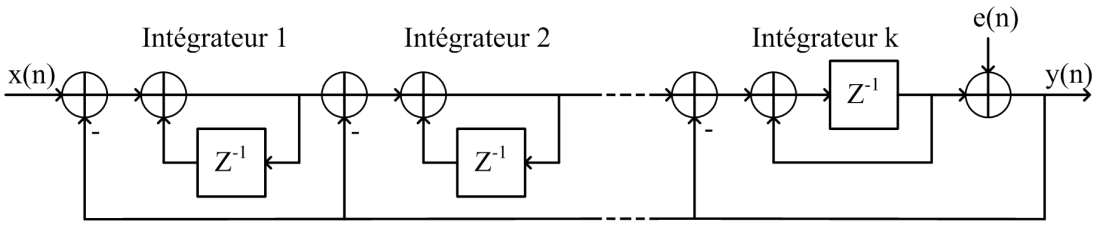


FIGURE 2.7 – Modèle à temps discret du modulateur ΣΔ d'ordre k.

La fonction de transfert de ce modulateur s'écrit :

$$Y(z) = X(z).z^{-1} + E(z).(1 - z^{-1})^k \tag{2.21}$$

La fonction de transfert associé au bruit et le carré de son module sont alors :

$$STF(z) = z^{-1} \tag{2.22}$$

$$NTF(z) = (1 - z^{-1})^k \tag{2.23}$$

$$|NTF(f)|^2 = 2^{2k} \sin^{2k}\left(\pi \frac{f}{F_s}\right) \tag{2.24}$$

La figure 2.8 donne les fonctions de transfert associées au bruit pour des modulateurs ΣΔ d'ordres 1 à 4.

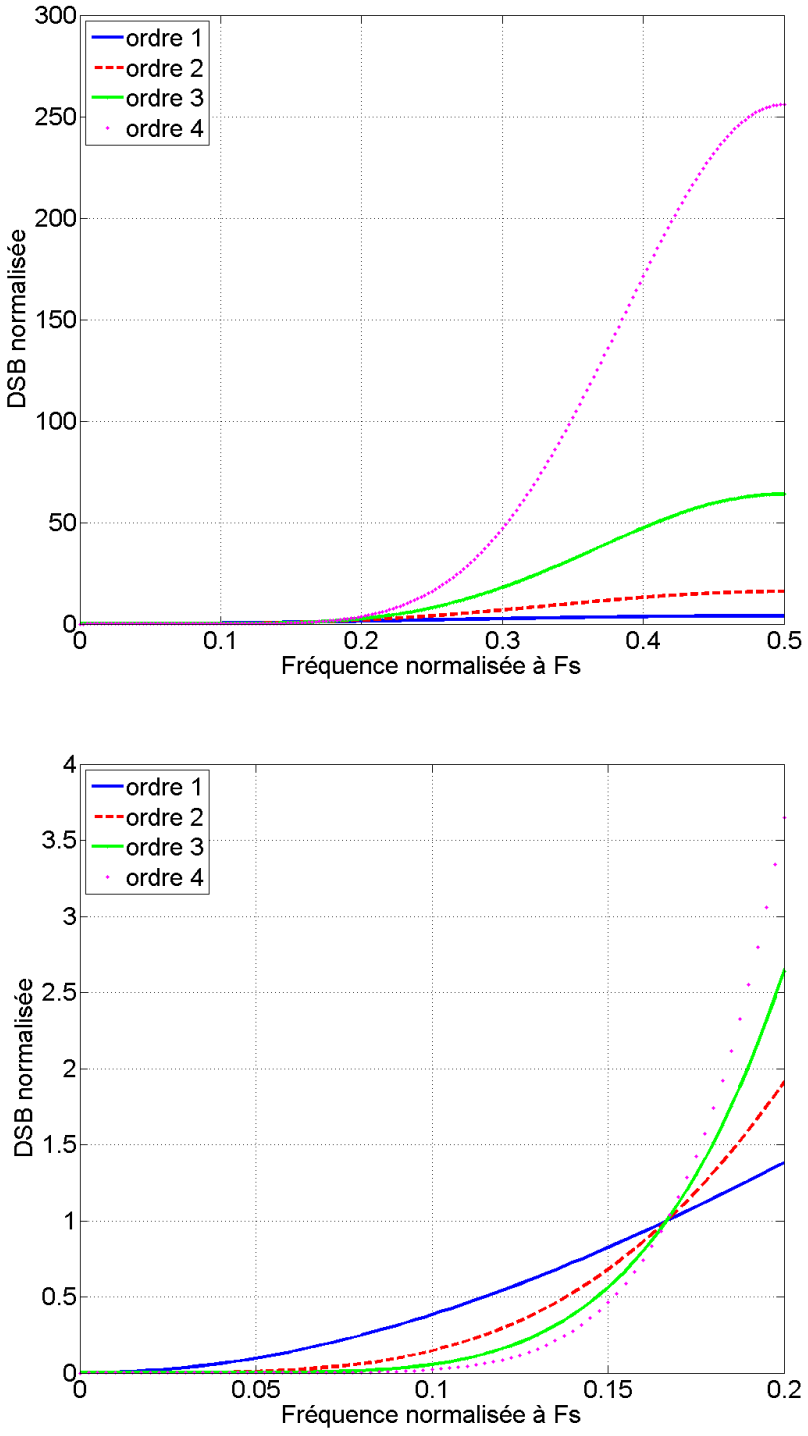


FIGURE 2.8 – Fonctions de transfert du modulateur ΣΔ pour des ordres de 1 à 4.

Une métrique de performance de ce système est donnée par le *SNR* dans la bande utile $[-B, B]$. Pour un signal sinusoïdal d'amplitude A, le *SNR* théorique est donné par [61] :

$$SNR = \frac{A}{\sqrt{2}} \frac{\sqrt{12}\sqrt{2k+1}OSR^{k+0.5}}{q\pi^k} \tag{2.25}$$

où $OSR = F_s/2B$ est le rapport de sur-échantillonnage. Théoriquement, l'amplitude maximale du sinus, A_{max}, que ce type de modulateur peut coder sans devenir instable est $q/2$ (q est le quantum défini par le CNA de retour). Le rapport signal à bruit maximal est alors donné par :

$$SNR_{max} = \frac{\sqrt{3}\sqrt{2k+1}OSR^{k+0.5}}{\sqrt{2}\pi^k} \tag{2.26}$$

Il dépend de deux variables : l'ordre k du modulateur $\Sigma\Delta$ et le rapport du sur-échantillonnage (*OSR*). L'évolution du $SNR_{\max}$ en fonction de l'*OSR* pour des modulateurs des quatre premiers ordres est décrit à la figure 2.9.

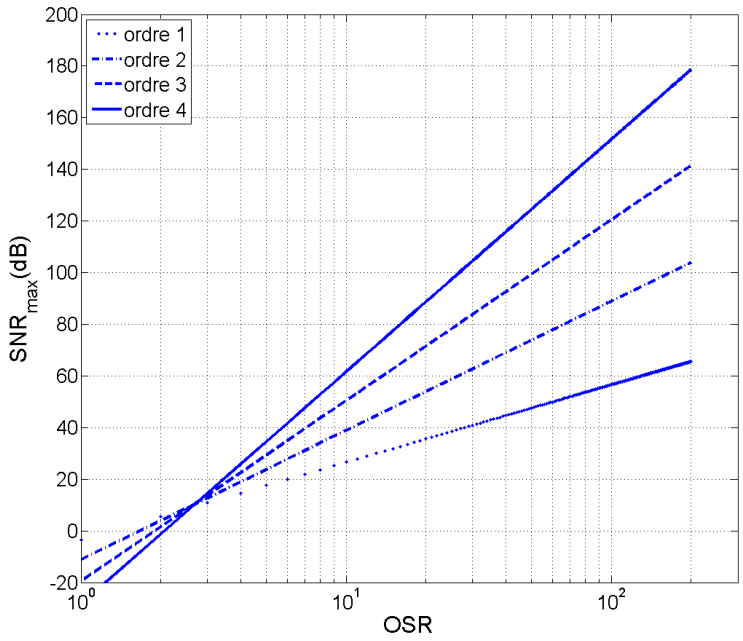


FIGURE 2.9 – *SNR* maximum en fonction de l'*OSR*

Ainsi, pour que le *SNR* du signal généré soit supérieur au *SNR* théorique du CAN (72 dB), un modulateur du second ordre avec un *OSR* de 64 ou un modulateur du troisième ordre avec un *OSR* de 16 sera nécessaire. Dans les prochains paragraphes, nous allons donc présenter deux modulateurs $\Sigma\Delta$, du second et du troisième ordre, proposés dans la littérature.

2.3 Modulateurs ΣΔ

2.3.1 ΣΔ passe bas du second ordre

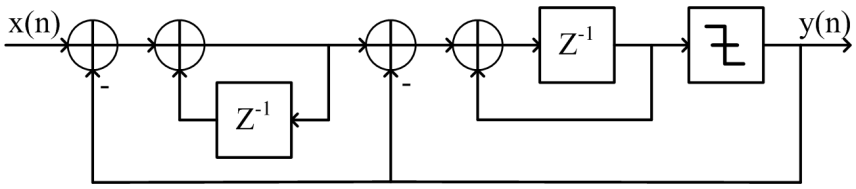


FIGURE 2.10 – Le modulateur ΣΔ de second ordre [55].

Pour obtenir un *SNR* supérieur à 72 dB avec un modulateur ΣΔ de second ordre, il faut au minimum un *OSR* de 64. Lu [55] utilise un tel modulateur ΣΔ de second ordre. Cette méthode est utilisée pour générer des signaux mono-fréquence de basse fréquence par rapport à la fréquence de suréchantillonnage.

2.3.2 ΣΔ d'ordre supérieur

Il est possible améliorer le rapport signal sur bruit en augmentant l'ordre du modulateur, mais des problèmes de stabilité apparaîtront. Matsuya [62] constate que pour un modulateur ΣΔ d'ordre trois classique (utilisant la structure du modèle à temps discret du modulateur décrit figure 2.7), des oscillations dues à un décalage de phase de $3\pi/2$ apparaissent dans la boucle de rétroaction. Une première solution utilisant une structure plus générale qui permet de choisir indépendamment les fonctions de transfert de bruit et du signal est présentée dans [63, 64]. Un signal basse fréquence avec un *SNR* de 100 dB a été obtenu en utilisant un modulateur d'ordre 4 [64]. Il existe d'autres structures permettant de s'affranchir des problèmes de stabilité du modulateur ΣΔ classique [65] : le modulateur MASH (*Multistage Noise Shaping*) [66, 62, 67, 68, 69], le modulateur nFOC (*n First Order Converter*) [70] ou le modulateur MSCL (*Multi Stage Closed Loop*) [65].

La structure du modulateur MASH 1-1-1 du troisième ordre (figure 2.11) utilise trois modulateurs du premier ordre stables cascadés, c'est à dire que l'erreur de quantification du premier modulateur est envoyée en entrée du second et celle du deuxième modulateur est envoyée en entrée du troisième modulateur. Les trois sorties quantifiées des modulateurs sont ensuite ajoutées dans un réseau d'annulation d'erreur composé de dérivateurs.

La structure nFOC découle de la structure MASH. Des modulateurs du premier ordre sont également cascadés mais chaque modulateur traite, non plus l'erreur de quantification de l'étage précédant, mais le signal à quantifier. De plus, cette structure nécessite un étage de prétraitement

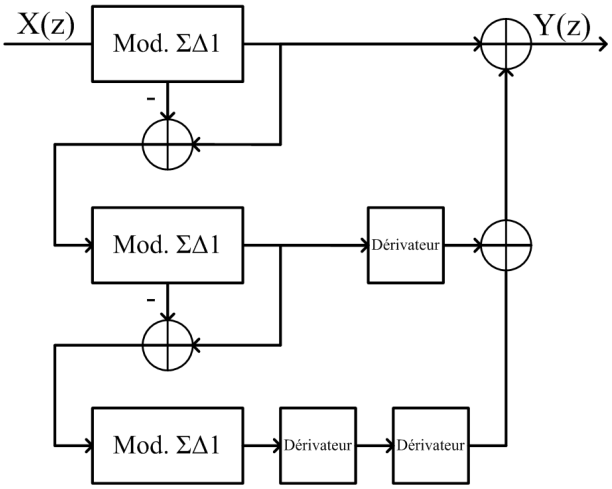


FIGURE 2.11 – Modulateur ΣΔ MASH 111 du troisième ordre.

pour obtenir des performances comparables à la structure MASH.

Enfin, la structure MSCL peut être vue comme une modification de la structure MASH, avec le rebouclage de la sortie du modulateur global en entrée.

Les modulateurs utilisant une structure MASH ou en découlant sont très intéressants grâce à leur stabilité mais présentent deux défauts majeurs pour une utilisation dans un oscillateur numérique. En effet, leurs STF sont différentes de 1 ou de z^{-1}, et leurs sorties sont sous un format multibits. Or dans la cadre de notre stratégie de BIST, une sortie monobit est nécessaire pour permettre la génération d'un signal analogique sans utiliser un CNA. De plus, un modulateur du second ordre est suffisant pour générer un signal de qualité spectrale suffisante pour le test d'un CAN de résolution 12 bits, cadre de cette thèse. Le modulateur du second ordre sous la forme d'intégrateurs cascadés sera donc utilisé dans l'oscillateur ΣΔ numérique et dont l'architecture est présentée dans le paragraphe suivant.

2.4 Etude et implémentation de l'oscillateur ΣΔ Passebas

Nous avons choisi d'implémenter sur FPGA l'oscillateur ΣΔ utilisant le modulateur ΣΔ présenté par Lu [55] (figure 2.10). En effet, c'est le seul proposant un *SNR* supérieur à celui du CAN (72 dB) pour un ordre peu élevé (ordre 2) et possédant une *STF* égale à 1, ce qui répond aux différentes contraintes imposées par son utilisation dans un oscillateur. Nous allons maintenant présenter son implémentation en commençant par l'étude et la définition de son architecture.

2.4.1 Etude de l'architecture de l'oscillateur ΣΔ

L'objectif de ce paragraphe est de définir l'architecture de l'oscillateur ΣΔ pour générer un signal sinusoïdal à la fréquence $f_0 = 0,998$ MHz, correspondant à la fréquence cohérente pour le CAN autour de 1 MHz avec un *SNR* supérieur à celui du CAN (> 72 dB). La structure de l'oscillateur ΣΔ est illustrée par la figure 2.12 sous forme d'un graphe flot de signal.

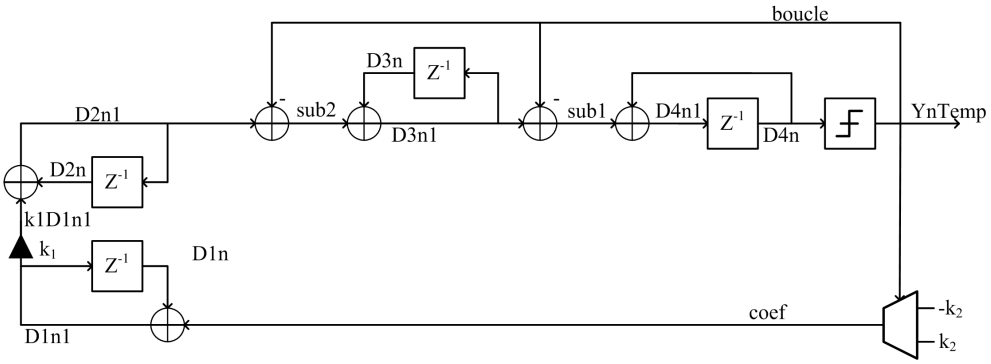


FIGURE 2.12 – Structure de l'oscillateur ΣΔ.

Cette structure comprend quatre additions, deux soustractions, une multiplication (par k_1), quatre éléments de mémorisation, un quantificateur 1 bit et un multiplexeur. A partir de cette structure, l'arbre de dépendance des différents signaux, et donc des données, peut être tracé. En effet, chaque nouvelle valeur de $D1n(n+1)$, $D2n(n+1)$, $D3n(n+1)$ et $D4n(n+1)$ nécessite des calculs préalables. Par exemple, la valeur de $D1n(n+1)$ est celle de $D1n1(n)$ qui dépend de la valeur de $D1n(n)$ et de $Coef(n)$ qui dépend elle même de la valeur de $D4n(n)$. Sur la figure 2.13 représentant cet arbre de dépendance, $(A \rightarrow B)$ signifie que le calcul de B dépend de la valeur de A.

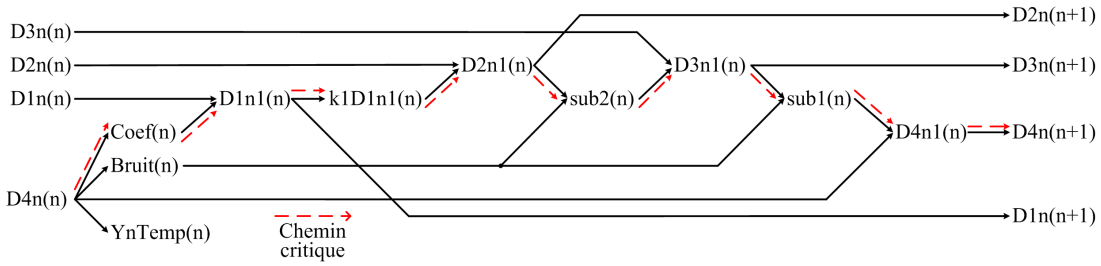


FIGURE 2.13 – Arbre de dépendance des données au sein de l'oscillateur ΣΔ.

A partir du graphe flot de signal et de l'arbre de dépendance des données, différentes métriques peuvent être déduites : le temps de chemin critique qui conduit à la cadence du système ainsi que

la latence du système. Le chemin critique d'un circuit est défini comme l'ensemble des opérations successives qui conditionnent le délai global du circuit pour réaliser sa fonction. En pratique, le chemin critique est le chemin le plus long entre les entrées/sorties et les délais. Le temps du chemin critique donne donc le temps d'exécution minimum des calculs pour une entrée avant de pouvoir accepter l'entrée suivante : c'est la *cadence*. L'inverse de la cadence donne le débit du système. Par exemple, si la cadence est de 20 ns, le débit sera de $1/20.10^{-9}$ Hz soit 50 MHz, c'est à dire que le système est capable de gérer 50 millions d'échantillons par seconde. La *latence* du circuit est le temps de retard entre l'entrée et la sortie d'une même itération. Par exemple, si la latence du système est de 30 ns, il faut attendre 30 ns entre le moment ou l'entrée est prise en compte par le système et le moment où la réponse associée est disponible en sortie du système.

Le chemin critique de l'oscillateur $\Sigma\Delta$ est composé des signaux suivant : *D4n*, *coef*, *D1n1*, *D2n1*, *k1D2n1*, *sub2*, *D3n1*, *sub1* et *D4n1*, soit les six additions/soustractions, la multiplication par k_1, le quantificateur et le multiplexeur. Sur la figure 2.13, il est représenté par des flèches en pointillés.

Avant de pouvoir établir l'architecture de l'oscillateur, il faut définir pour chaque opération un opérateur matériel. Les éléments de mémorisation seront réalisés par des registres et les additions/soustractions par des additionneurs/soustracteurs. Le coefficient k_1 permet de réduire l'amplitude des signaux internes de l'oscillateur. Puisque la fréquence d'oscillation ne dépend que du produit k_1k_2, la valeur de k_1 peut être choisi arbitrairement comme une puissance négative de 2. Cela permet de réaliser cette multiplication par un décalage à droite. Le quantificateur 1 bit revient à isoler le bit de signe du signal qui est soit positif, soit négatif (le nombre zéro est considéré comme étant positif). Aucun opérateur n'est donc nécessaire. Mais il faut ensuite générer un signal multibits (signal *boucle*) dont la valeur correspond à '1' ou '-1'. Ce signal sera généré de la même façon que le signal *coef* avec un multiplexeur commandé par la sortie du quantificateur et dont les entrée sont des signaux multibits valant '1' et '-1'. Seuls les registres sont synchrones, c'est à dire qu'ils sont cadencés par l'horloge du circuit. A chaque coup d'horloge, le signal *D4n(n)* prend la valeur de *D4n1(n-1)* alors que les signaux *coef* et *bruit* ne peuvent être mis à jour qu'une fois *YnTemp(n)*, c'est à dire le MSB du signal *D4n(n)*, a été mis à jour. En synchronisant les multiplexeurs et en utilisant le MSB du signal *D4n1(n-1)* comme signal de commande, la fonctionnalité du circuit sera préservée et les multiplexeurs auront le comportement d'un élément de mémorisation par rapport au chemin critique. Le nouveau chemin critique est maintenant composé des signaux *coef*, *D1n1*, *D2n1*, *sub2*, *D3n1*, *sub1* et *D4n1* soit les six additions/soustractions. La figure 2.14 présente l'architecture ainsi définie :

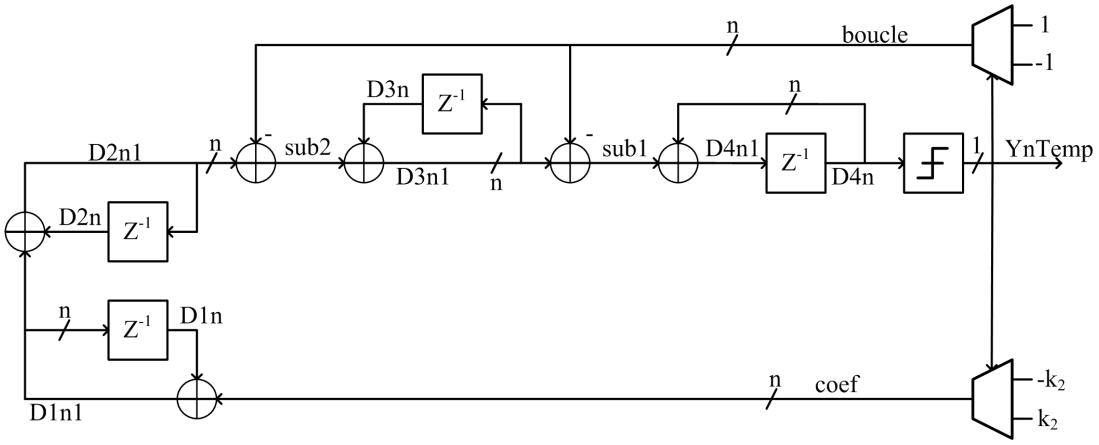


FIGURE 2.14 – Architecture interne de l'oscillateur ΣΔ.

2.4.2 Implémentation de l'oscillateur ΣΔ du second ordre

Pour pouvoir implémenter l'architecture de l'oscillateur, il faut à présent dimensionner les différents signaux du circuit. Pour cela, il faut, dans un premier temps, définir la fréquence de suréchantillonnage du modulateur, puis les valeurs des coefficients multiplicateurs et enfin, les valeurs initiales des registres dont dépendent les caractéristiques du signal généré (paragraphe 2.1.1).

Les équations 2.27 à 2.30 rappellent les expressions de la fréquence f_0, de l'amplitude A, de la phase ϕ et du SNR_{max} du signal généré à partir de la fréquence de suréchantillonnage F_s du modulateur, des deux coefficients multiplicateurs k_1 et k_2 et des valeurs initiales $R_1(0)$ et $R_2(0)$ des registres R_1 et R_2 :

$$f_0 = \frac{F_s}{2\pi} \times \cos^{-1}\left(1 - \frac{k_1 k_2}{2}\right) \tag{2.27}$$

$$\phi = \tan^{-1}\left(\frac{R_1(0)\sin\left(\frac{2\pi f_0}{F_s}\right)}{\left(\frac{1-k_1k_2}{2}\right)R_1(0) + k_1 R_2(0)}\right) \tag{2.28}$$

$$A = \frac{(1-k_1k_2)\,R_1(0) + k_1 R_2(0)}{\sin\left(\frac{2\pi f_0}{F_s}\right) + \phi} \tag{2.29}$$

$$\mathrm{SNR}_{max} = \frac{\sqrt{30\ OSR^5}}{2\pi^2} = \frac{\sqrt{30\,(F_s/3f_0)^5}}{2\pi^2} \tag{2.30}$$

A partir de l'équation 2.30, la fréquence de suréchantillonnage minimale est calculée :

$$F_{s,min} = 3f_0\sqrt[5]{\frac{1}{30}\left(2\pi^2 SNR_{max}\right)^2} \tag{2.31}$$

La fréquence de suréchantillonnage est choisie comme un multiple de la fréquence d'échantillon-

nage du CAN (fixée à 41 MHz pour notre application) :

$$F_s = \lceil F_{s,min}/F_{can} \rceil \times F_{can} \tag{2.32}$$

où l'opérateur $\lceil\ \rceil$ correspond à l'opération d'arrondi supérieur. La valeur du produit k_1k_2 est :

$$k_1k_2 = 2\left(1 - \cos\left(\frac{2\pi f_0}{F_s}\right)\right) \tag{2.33}$$

La valeur de k_1 est ensuite choisie comme une puissance négative de 2 pour deux raisons : minimiser l'amplitude des signaux internes de l'oscillateur et simplifier l'implémentation de la multiplication. En effet, multiplier un signal binaire par une puissance de 2 revient à faire un décalage de la valeur binaire du signal.
Puis, en choisissant une valeur nulle pour $R_1(0)$, la phase du signal générée devient nulle ($\phi = 0$) et l'expression de l'amplitude ne dépend plus que de la valeur de $R_2(0)$ et de k_1 :

$$A = \frac{k_1 R_2(0)}{\sin\left(\frac{2\pi f_0}{F_s}\right)} \tag{2.34}$$

Ainsi, la valeur de $R_2(0)$ vaut :

$$R_2(0) = \frac{A \sin\left(\frac{2\pi f_0}{F_s}\right)}{k_1} \tag{2.35}$$

Pour définir la valeur k_1 et donc la valeur de $R_2(0)$, différentes simulations du modèle du circuit sont réalisées en fonction du $log_2(k_1)$ afin de déterminer la valeur minimisant l'amplitude des signaux internes. Mais, il faut au préalable choisir la valeur de l'amplitude du signal généré A. Si l'on choisit la pleine échelle de l'oscillateur, A vaut 1. Mais dans ce cas là, l'oscillateur est divergeant [71, 72]. Il faut donc diminuer cette amplitude. Le modèle de l'oscillateur est donc simulé en fonction de la valeur de A et le *SNR* du signal généré est calculé afin de déterminer empiriquement sa valeur maximisant le *SNR*. Puisque le comportement de l'oscillateur ne dépend que de la valeur du produit k_1k_2, indépendamment de leurs valeurs respectives, pour cette simulation, on s'affranchit du coefficient k_1 en choisissant de lui attribué 1 comme valeur. La figure 2.15 montre le résultat de cette simulation. On peut alors constater que la valeur de A maximisant le *SNR* (56 dB) est 0.65.
Malheureusement, ce *SNR* maximal que l'on peut obtenir avec cet oscillateur n'est pas suffisant. Cela n'est pas pour autant inattendu. En effet, le *SNR* d'un modulateur $\Sigma\Delta$ dépend linéairement de l'amplitude du signal d'entrée (cf. équation 2.25). En diminuant l'amplitude du signal généré, le *SNR* du signal généré décroît. Pour corriger ce défaut, il existe plusieurs solutions : augmenter la fréquence de fonctionnement du circuit, augmentant alors l'*OSR* et donc le *SNR*. Mais cette solution n'est pas viable. En effet, plus la fréquence de fonctionnement augmente, plus l'oscillateur devient divergeant [72]. Une autre solution consiste à augmenter l'ordre du

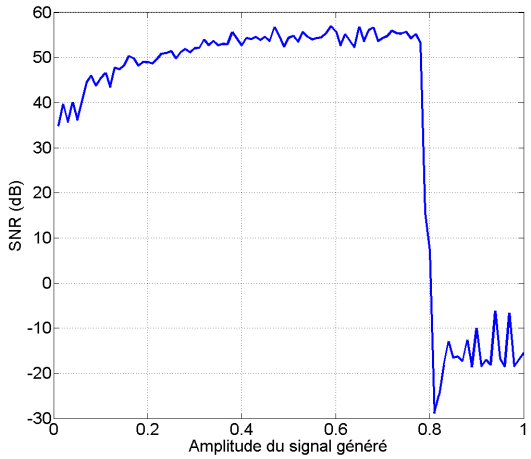


FIGURE 2.15 – SNR du signal généré en fonction de son amplitude.

modulateur ΣΔ. Ainsi avec un modulateur passe bas d'ordre 4, un *SNR* supérieur à 72 dB peut être obtenu.

2.4.3 Implémentation de l'oscillateur ΣΔ du quatrième ordre

Un oscillateur ΣΔ passe bas d'ordre 4 a déjà été réalisé au sein de l'équipe [53]. Le modulateur utilisé a été spécifiquement conçu pour avoir une précision et une stabilité souhaitée, c'est ce que l'on appelle la conception de modulateur ΣΔ *single bit* d'ordre élevé. Il a été conçu pour un *OSR* de 64 avec une fréquence de fonctionnement de 2.56 MHz et une fréquence du signal généré allant jusqu'à 20 *kHz*. La figure 2.16 décrit ce modulateur qui utilise une structure LDI (Loseless Discrete Intégrator) et des coefficients sous forme de puissances de 2.
Ce modulateur n'a pas été conçu pour des fréquences élevées mais en utilisant une fréquence de suréchantillonnage à 246 MHz (soit 5 fois la fréquence d'échantillonnage du convertisseur), la fréquence du signal généré peut atteindre 1.921 MHz. Il n'est donc pas nécessaire de modifier ce modulateur pour l'utiliser dans notre oscillateur. La table 2.1 donne la valeur des coefficients du modulateur [73, 53] :

Ces coefficients n'influent pas sur la fréquence des oscillations, mais uniquement sur la mise en forme du bruit de quantification. De même que pour l'oscillateur d'ordre 2, deux simulations sont réalisées pour définir les valeurs de A et k_1 afin de déterminer celle de $R_2(0)$. La figure 2.17 montre le résultat de la première simulation. L'oscillateur du quatrième ordre présente la même

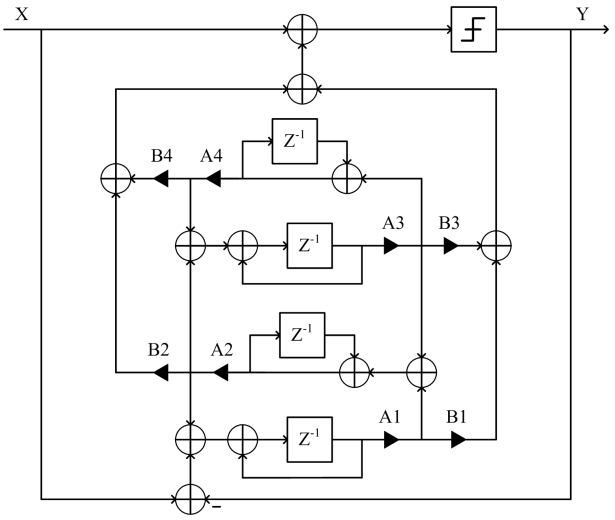


FIGURE 2.16 – Structure du modulateur passe bas d'ordre 4.

A1	2^{-8}	B1	2^{6}
A2	-2^{-4}	B2	-2^{9}
A3	-2^{-4}	B3	2^{10}
A4	2^{-6}	B4	2^{14}

TABLE 2.1 – Coefficients du modulateur passe bas d'ordre 4.

limitation que l'oscillateur du second ordre : lorsque l'amplitude du signal généré dépasse un certain palier, celui-ci diverge. On peut alors constater que la valeur de A maximisant le SNR (75.87 dB) est 0.56.
Maintenant que la valeur de A est fixée, l'oscillateur peut être simulé en fonction de de la valeur de k_1. La table 2.2 donne la valeur des paramètres k_2 et $R_2(0)$ ainsi que l'amplitude maximale des signaux internes de l'oscillateur en fonction de k_1 :

Ainsi, on peut constater que pour k_1 valant 2^{-5}, tous les signaux et les paramètres sont maximaux mais inférieurs à 1. Nous allons donc utiliser cette valeur pour k_1
Les valeurs numériques (à trois chiffres significatifs) des paramètres de l'oscillateur sont résumées dans la table 2.3 pour un OSR de 64 :

Les figures 2.18(a) et 2.18(b) montre le spectre du signal généré par l'oscillateur $\Sigma\Delta$ passe bas d'ordre 4 et filtré par une filtre passe bas passif du quatrième ordre. Il présente une unique raie

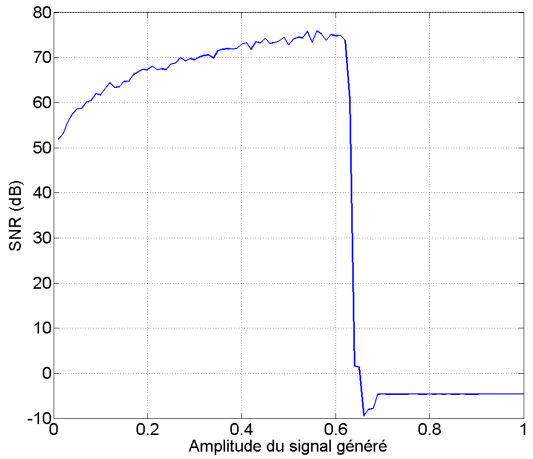


FIGURE 2.17 – SNR du signal généré en fonction de son amplitude.

k1	k2	R2(0)	max(R_1)	max(R_2)
1	0.0175	0.0007	0.0143	0.0175
2^{-1}	0.0351	0.0013	0.0286	0.0351
2^{-2}	0.0701	0.0026	0.0571	0.0701
2^{-3}	0.1402	0.0052	0.1142	0.1403
2^{-4}	0.2805	0.0104	0.2285	0.2805
2^{-5}	0.5610	0.0208	0.4570	0.5610
2^{-6}	1.1220	0.0416	0.9139	1.1220
2^{-7}	2.2440	0.0832	1.8278	2.2440

TABLE 2.2 – Valeurs des paramètres et amplitude maximale des signaux internes de l'oscillateur.

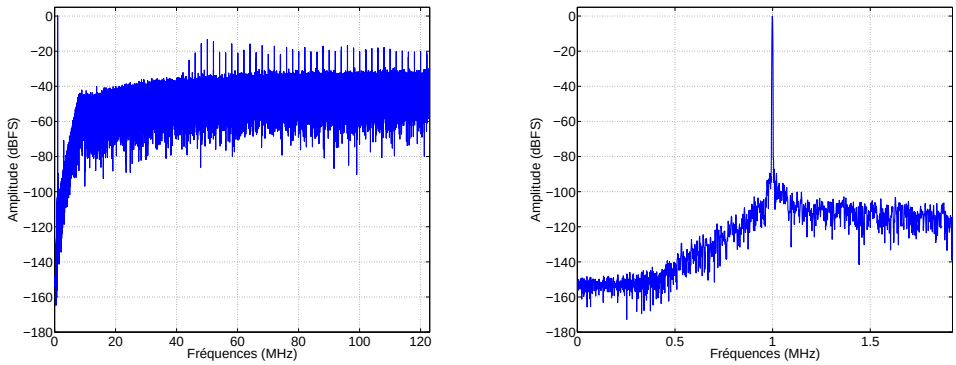
f_0	F_s	k_1	k_2	$R_1(0)$	$R_2(0)$
0,998 MHz	246 MHz	2^{-5}	0.561	0	0.0208

TABLE 2.3 – Valeurs des différents paramètres de l'oscillateur ΣΔ.

à 0.998 MHz, un plancher de bruit inférieur à -100 dB et un *SNR* de 75.87 dB.

2.4.3.1 Dimensionnement des signaux

Il faut maintenant dimensionner les coefficients et les signaux internes. Pour cela, on effectue une comparaison entre les *SNR* du signal généré de la simulation en virgule flottante et des simulations en point fixe en fonction de la taille des radix des coefficients (figure 2.19). Un *SNR*



(a) Spectre entier (b) Spectre dans la bande utile

FIGURE 2.18 – Spectre du signal analogique généré.

de 72 dB est atteint pour des coefficients de 18 bits pour la simulation en point fixe, c'est à dire 17 bits pour le radix. A partir de cette simulation en point fixe, le domaine de définition des signaux internes est mesuré. Ces signaux sont strictement compris entre -4 et 4, correspondant à une partie entière de 2 bit dans leur écriture binaire. Ainsi, le format de tous les signaux de l'oscillateur $\Sigma\Delta$ est (20,2,17).

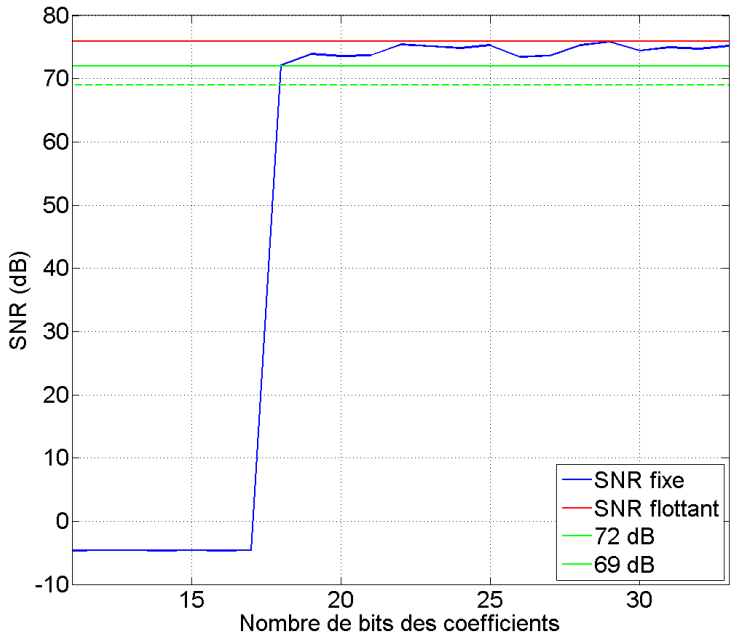


FIGURE 2.19 – Rapport signal à bruit en fonction de la tailles des radix des coefficients de l'oscillateur.

2.4.3.2 Réalisation matérielle de l'oscillteur

Les seules entrées de l'oscillateur ΣΔ sont l'horloge (*clkS*) et un signal de reset asynchrone (*rst*). En sortie, la cellule délivre le signal généré (*Yn*) sur 1 bit. Le circuit est finalement décrit de façon comportementale en VHDL puis synthétisée avec l'outil ISE 10.1 de Xilinx pour un FPGA Xilinx Virtex IV. Ensuite, le fonctionnement de la cellule est vérifié par une simulation comportementale pré-synthèse sous ModelSim. Chaque synthèse est réalisée en n'utilisant ni les blocs DSP, ni les blocs de RAM, ni la RAM distribuée proposés par le FPGA. Chaque slice contient [74] :

– Deux générateurs de fonction pouvant être utilisée comme LUT à 4 entrées.
– Deux éléments de mémorisation
– Des portes logiques
– De larges multiplexeurs
– Une chaîne de propagation rapide de retenue

Une paire de slices dans un CLB (Configurable Logic Bloc) peut être configurée comme un registre à décalage de 16 bits ou comme 16 bits de RAM distribuée. De plus, les deux éléments de mémorisation sont soit des bascules D, soit des latches.

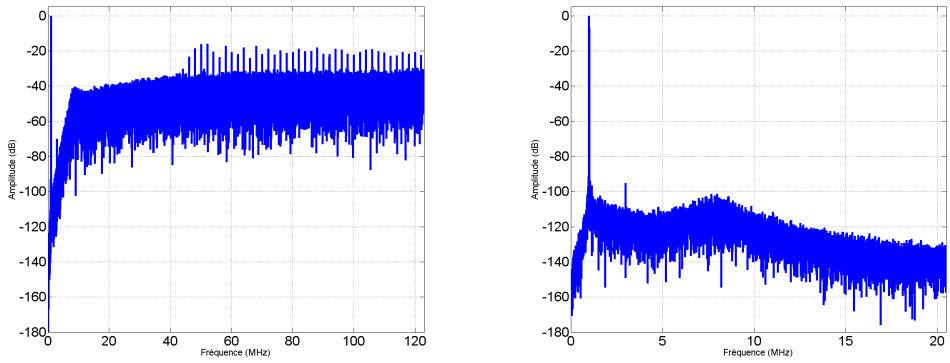
L'oscillateur contient 6 délais, 9 décalages, 10 additionneurs, 4 soustracteurs, 2 multiplexeurs et 4 constantes. La table 2.4 résume les résultats de synthèse de l'oscillateur.

Structure	Nb de slices	Nb de slices FF	Nb de LUT4	Fréq. de fonctionnement (MHz)
Oscillateur ΣΔ	191	114	356	86

TABLE 2.4 – Résultat de la synthèse architecturale de l'oscillateur ΣΔ.

Pour valider notre conception de l'oscillateur ΣΔ, une simulation comportementale du design est réalisée sous ModelSim. Le signal de sortie est récupéré sous MATLAB et ses spectres avant et après filtrage calculés (figure 2.20. Le filtrage est ici réalisé par un filtre passe bas de Butterworth du 4ème ordre.

Le *SNR* du signal généré dans la bande utile est ensuite calculé. Celui-ci est de $72,2$ dB, ce qui est supérieur non seulement au *SNR* du CAN (69 dB) mais aussi à son *SNR* théorique (72 dB). Lors de la conversion, les erreurs qui pourront apparaître seront bien dues aux imperfections du CAN.



(a) Spectre entier (b) Spectre du signal analogique

FIGURE 2.20 – Spectre du signal généré par l'oscillateur ΣΔ.

2.5 Conclusion

Le test des CAN par BIST nécessite la génération in-situ d'un signal de qualité spectrale supérieure à celle supposée du CAN sous test. De plus, il faut imaginer une structure la plus simple possible pour occuper un minimum d'espace sur le silicium. Si l'on veut introduire ce système sur une carte d'évaluation commerciale, il faudra aussi que les spécifications soient constantes. De ces remarques, la voie du numérique semblait inéluctable.

Nous venons de voir différentes structures et stratégies pour générer les stimuli du test. Le premier générateur présenté a été l'oscillateur numérique simple associé à un CNA multibit. Le principal défaut de cet oscillateur est l'utilisation de ce CNA multibit, consommateur d'une très grande surface et source de non linéarité dans le signal analogique. Nous nous sommes donc concentré sur les oscillateurs ΣΔ. La modulation ΣΔ permet d'appliquer une transformation sur le bruit de quantification afin de le repousser hors de la bande utile en utilisant les principes de suréchantillonnage. Après avoir étudier quelques modulateurs extraits de la littérature, il semblait que les oscillateurs à base de modulateurs MASH pouvait être une solution satisfaisante mais les solutions détaillées présentaient chacune un défaut (sortie multi-bits ou $STF(z) \neq 1$). Nous avons donc utilisé un modulateur ΣΔ passe bas du second ordre sous forme d'intégrateurs cascadés. Malheureusement, ce modulateur ne permet pas d'atteindre les performances attendues. Un modulateur ΣΔ du quatrième ordre a été alors utilisé.

L'oscillateur ΣΔ passe bas ainsi réalisé présente un *SNR* de 72,2 *dB* pour une occupation de 191 slices. La fréquence de suréchantillonnage du modulateur ΣΔ est très élevée (de l'ordre de plusieurs centaines de mégahertz). Ces fréquence de fonctionnement ne peuvent être atteinte sur

FPGA pour des calculs sur plusieurs dizaines de bits. Il faut réaliser un circuit intégré spécifique (ASIC) pour cet oscillateur. Le prototypage réalisé sur FPGA montre seulement la faisabilité d'un tel oscillateur.

Chapitre

3 Etude théorique du banc de filtres linéaires

Sommaire

Introduction

Le signal issu d'un CAN, attaqué par une sinusoïde pure, est généralement constitué du fondamental (fréquence du signal de test), de ses harmoniques (générées par la non linéarité du CAN) et d'un bruit (bruit de quantification, bruit de jitter, ...). Grâce à l'analyse spectrale (cf. partie 1.3.3), il est possible d'estimer les paramètres dynamiques tels que le rapport signal sur bruit (SNR : Signal-to-Noise Ratio) ou la distortion dynamique totale (THD : Total Harmonic Distortion) à partir des puissances des différentes composantes harmoniques du signal converti. Pour mesurer ces puissances, nous proposons une solution basée sur le filtrage numérique. Dans ces conditions, le principe du banc de filtres consiste à utiliser des cellules notch (filtre rejecteur de composante harmonique) séparant chacune une des composantes harmoniques (le fondamen-

tal sera, à juste titre, considéré comme une harmonique dans sa terminologie) du reste du signal.

Dans la première partie de ce chapitre, nous présentons une étude théorique du filtre notch en commençant par l'étude de son élément principal : le filtre résonateur du second ordre. Nous détaillerons leurs fonctions de transfert, leurs réponses en fréquence ainsi que leurs bandes passantes ou encore leurs fréquences de résonance.

Puis dans un second temps, nous nous intéresserons aux différentes topologies de banc de filtres utilisées pour l'extraction des paramètres. Nous étudierons les réponses en fréquence de différentes topologies et nous présenterons la topologie la plus adaptée à notre application.

3.1 Etude théorique du filtre notch

Le filtre notch isole d'un coté une composante spectrale du signal X_{bp} grâce à un filtre résonateur et rejette le reste du spectre X_{nt} de l'autre (Figure 3.1).

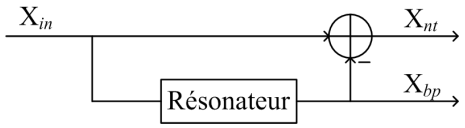


FIGURE 3.1 – Structure du filtre notch.

Avant de présenter le filtre notch, nous commencerons par l'étude de son élément principal, à savoir le filtre résonateur.

3.1.1 Étude théorique du filtre résonateur

Le filtre résonateur est une cellule biquadratique dont l'équation aux différences s'écrit :

$$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a_1y(n-1) - a_2y(n-2) \tag{3.1}$$

et dont la fonction de transfert est :

$$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}} \tag{3.2}$$

Le résonateur numérique est un cas particulier de la cellule du second ordre qui possède deux pôles complexes conjugués localisés prés du cercle unité [75]. Le terme résonateur réfère à la large réponse en amplitude du filtre à proximité des lieux de pôles, c'est à dire qu'il ne laisse passer que les fréquences proches de sa fréquence de résonance et qu'il atténue tout le reste. La position

de la fréquence de résonance étant fixée par le module et l'argument des pôles, leurs coordonnées polaires ($p = re^{\pm j\theta}$ avec $0 < r < 1$) sont utilisées pour écrire la fonction de transfert :

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 - (2r\cos\theta)z^{-1} + r^2 z^{-2}} \tag{3.3}$$

En identifiant les équations 3.2 et 3.3, nous en déduisons que $a_1 = -2r\cos\theta$ et $a_2 = r^2$. Bien qu'il existe plusieurs possibilités pour le choix des zéros, la solution la plus intéressante pour la réalisation du filtre notch correspond à des zéros placés en $z = 1$ et $z = -1$. En effet, cela permet une meilleure réjection et cela assure une phase nulle à la fréquence de résonance, point fondamental lors de son insertion dans une structure notch.
La fonction de transfert du filtre résonateur est alors :

$$H(z) = \frac{b_0(1 - z^{-1})(1 + z^{-1})}{1 - (2r\cos\theta)z^{-1} + r^2 z^{-2}} = \frac{b_0(1 - z^{-2})}{1 - (2r\cos\theta)z^{-1} + r^2 z^{-2}} \tag{3.4}$$

Pour obtenir un gain (maximal) unitaire à la pulsation de résonance Ω_r, le gain b_0 vaut alors $\frac{1-r^2}{2}$ [73]. Le carré du module de H(z) est :

$$|H(z)|^2 = \frac{(1 - r^2)^2}{2} \frac{(1 - \cos 2\Omega)}{(1 + r^4 + 4r^2\cos^2\theta) - 4r\cos\theta(1 + r^2)\cos\omega + 2r^2\cos 2\Omega} \tag{3.5}$$

Avec les zéros placés en $z = 1$ et $z = -1$, la pulsation de résonance du filtre pour laquelle le module est maximum et unitaire vaut approximativement, lorsque r est proche de 1 [76] :

$$\Omega_r \approx \arccos\left(\frac{2r}{1 + r^2}\cos\theta\right) \tag{3.6}$$

La largeur de la bande passante à $-3dB$, pour un module des pôles r proche de 1 est donnée par [77] :

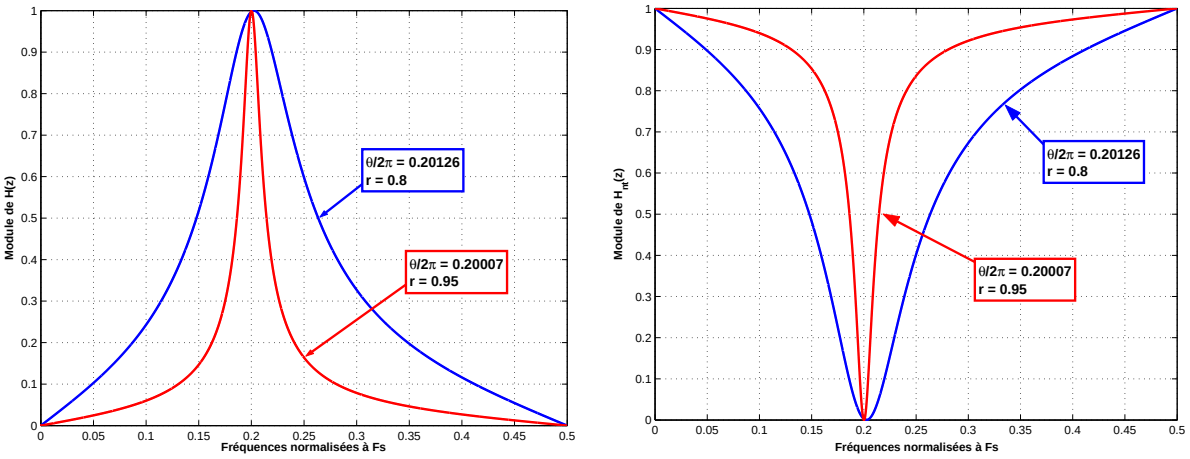
$$B_{-3\,\mathrm{dB}} \approx \frac{1 - r}{\pi} \tag{3.7}$$

La figure 3.2(a) illustre la réponse en fréquence du filtre résonateur ainsi défini, pour une fréquence de résonance normalisée à $0,2$ avec $r = 0,8$ et $r = 0,95$.

3.1.2 Étude théorique du filtre notch

Le filtre notch est composé du filtre résonateur précédent dont la sortie X_{bp} (composante harmonique) est soustraite au signal d'entrée pour obtenir la sortie coupe-bande X_{nt} (cf. Figure 3.1). Ceci est possible uniqement lorsque la phase est nulle à Ω_r, ce qui est le cas pour une structure avec des zéros à ± 1.
La fonction de transfert de la sortie X_{bp} est donc :

$$H_{bp}(z) = \frac{X_{bp}(z)}{X_{in}(z)} = \frac{1 - r^2}{2} \frac{(1 - z^{-2})}{1 - (2r\cos\theta)z^{-1} + r^2 z^{-2}} \tag{3.8}$$



(a) Banc A (b) Banc B

FIGURE 3.2 – Réponse en fréquence des filtres résonateur et coupe-bande avec zéros en z = *-1* et z = *1*

et celle de la sortie coupe-bande X_{nt} est :

$$H_{nt}(z) = \frac{X_{nt}(z)}{X_{in}(z)} = 1 - H_{bp}(z) = \frac{\frac{1-r^2}{2} - (2r\cos\theta)z^{-1} + \frac{r^2-1}{2}z^{-2}}{1 - (2r\cos\theta)z^{-1} + r^2 z^{-2}} \tag{3.9}$$

La figure 3.2(b) illustre la réponse en fréquence du filtre coupe-bande ainsi défini, pour une fréquence de résonance normalisée à $0,2$ avec $r = 0,8$ et $r = 0,95$. De même que pour le filtre résonateur, la position de la fréquence de coupure ainsi que la bande passante sont données par les équations 3.6 et 3.7.

3.2 Etude des topologies de bancs de filtres

Il s'agit pour le banc de filtre de séparer les différentes composantes harmoniques du signal issu du CAN. Nous allons donc voir comment organiser ces cellules pour arriver à nos fins. Nous présenterons ici trois topologies différentes : le banc de filtres cascadés, le banc de filtres cascadés-parallèles et enfin le banc de filtres parallèles.

3.2.1 Banc de filtres cascadés

Une première topologie de banc de filtres effectuant cette séparation est la suivante : N filtres notch sont positionnés en cascade, chacun retirant une composante spectrale du signal comme le montre la figure 3.3.

L'entrée du banc de filtres est notée X_{in}. La sortie $X_{bp,i}$ correspond à la $i^{ème}$ sortie passe-bande, et X_{nt}, la sortie coupe-bande associée au bruit. Chaque cellule réalise deux fonctions de

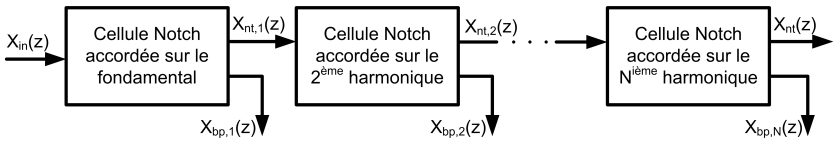


FIGURE 3.3 – Banc de filtres cascadés.

transfert : H_i, correspondant à la sortie passe-bande et $H_{nt,i} = 1 - H_i$ correspondant à la sortie coupe-bande. La fonction de transfert $H_{bp,i}$ de chaque sortie passe-bande $X_{bp,i}$ est donnée par :

$$H_{bp,i}(z) = H_i(z) \cdot \prod_{j=1}^{j<i} H_{nt,j} = H_i(z) \cdot \prod_{j=1}^{j<i} (1 - H_j(z)) \tag{3.10}$$

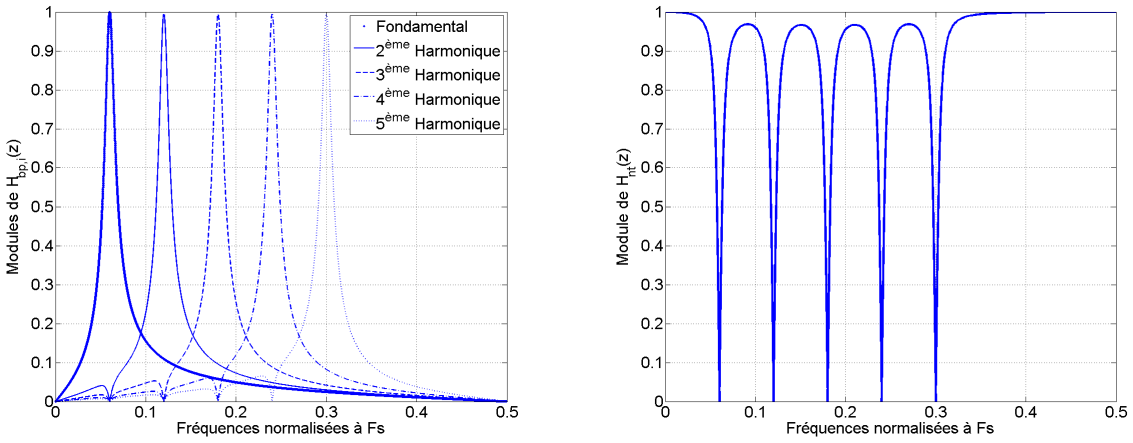
où

$$H_i(z) = \frac{b_{0,i}(1 - z^{-2})}{1 + a_{1,i}z^{-1} + a_{2,i}z^{-2}} \tag{3.11}$$

et $b_{0,i}$, $a_{1,i}$ et $a_{2,i}$ sont les coefficients du filtre résonant à $f_{r,i}$. La fonction de transfert H_{nt} de la sortie coupe-bande X_{nt} est donnée par :

$$H_{nt}(z) = \prod_{i=1}^{N} H_{nt,i} = \prod_{i=1}^{N} (1 - H_i(z)) \tag{3.12}$$

La figure 3.4 montre respectivement la réponse en fréquence de chaque sortie passe-bande du banc de filtre pour $N = 5$ (a) et la réponse en fréquence de la sortie coupe-bande (b).



(a) Sorties passe-bandes

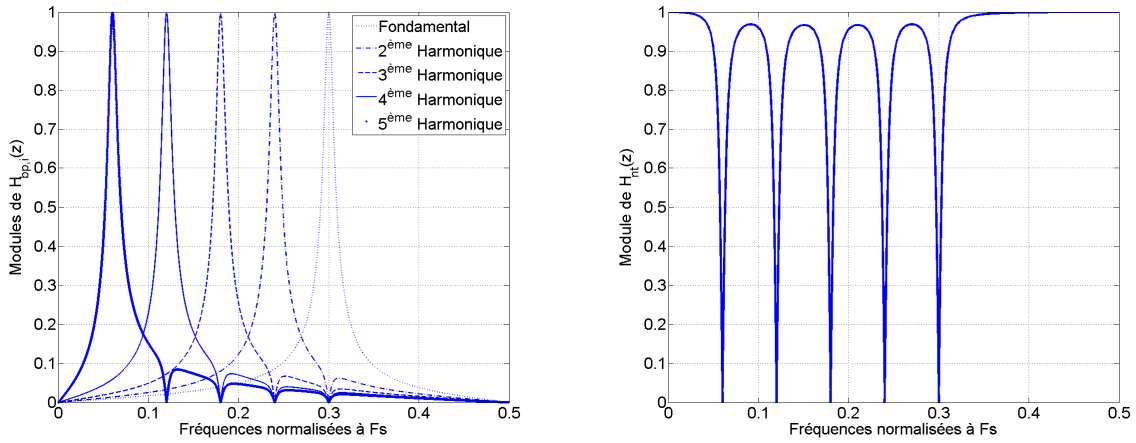
(b) Sortie coupe-bande

FIGURE 3.4 – Réponses en fréquence des sorties du banc cascadé avec un fondamental à $0.06Fs$.

La principale caractéristique de ces réponses en fréquence est l'élimination des fréquences au fur et à mesure des cellules. La sortie de la première cellule, sélectionnant le fondamental,

contient une partie de la puissance des harmoniques suivantes. De la même façon, la sortie de la deuxième cellule, sélectionnant le deuxième harmonique, contient une partie de la puissance des harmoniques suivantes à partir du troisième harmonique. Il est à noter que le fondamental est éliminé. Il en va ainsi de suite, jusqu'à la $N^{ième}$ sortie qui ne contient aucune partie de la puissance des autres harmoniques.

Il existe une alternative qui consiste à inverser l'ordre des filtres : plutôt que d'extraire le fondamental en premier, les harmoniques d'ordre élevé seront supprimés avant ceux d'ordre plus faible. La figure 3.5 détaille les réponses en fréquence pour chaque sortie passe-bande du banc de filtre inversé pour $N = 5$.



(a) Sorties passe-bandes

(b) Sortie coupe-bande

FIGURE 3.5 – Réponses en fréquence des sorties du banc cascadé inversé avec un fondamental à $0.06Fs$.

Nous avons ici un comportement similaire mais avec un résultat différent. Le deuxième harmonique contient une partie de la puissance du fondamental, le troisième harmonique contient une partie de la puissance du deuxième harmonique et du fondamental et ainsi de suite jusqu'au dernier harmonique qui contient une partie de la puissance de tous les harmoniques précédents.

Pour pouvoir mesurer précisément la puissance de chaque harmonique, il faut éliminer la puissance des autres harmoniques. En effet, selon l'importance et la position des harmoniques, l'estimation de la puissance de chacune des composantes sera plus ou moins biaisée. Une solution à ce problème consiste à utiliser le banc de filtres parallèles-cascadés, qui devrait permettre l'atténuation des résidus harmoniques.

3.2.2 Banc de filtres parallèles-cascadés

Les bancs de filtres cascadés ne permettent pas l'élimination des différentes composantes harmoniques sur toutes les sorties passe-bandes du banc de filtres. Seule la dernière sortie passe-bande $X_{bn,N}$ et le sortie coupe-bande X_{nt} ne contiennent aucune partie de la puissance des autres harmoniques. Pour remédier à ce défaut, une première solution est d'éliminer, pour chaque sortie passe-bande, la puissance des harmoniques supérieurs en utilisant des cellules supplémentaires : c'est le banc de filtres parallèles-cascadés (figure 3.6) [73].

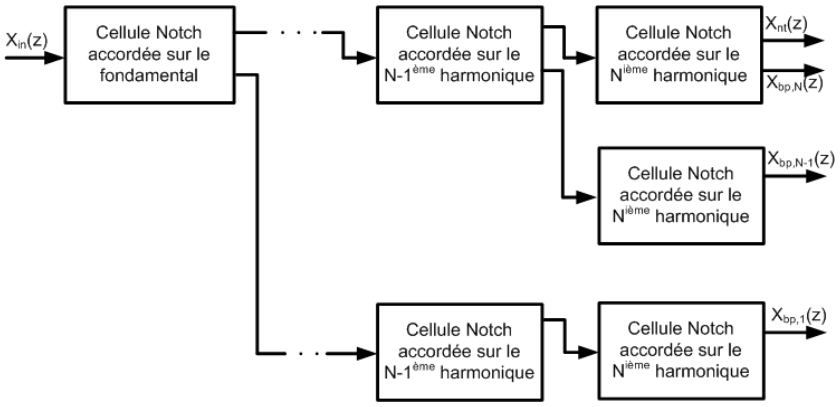


FIGURE 3.6 – Le banc de filtres parallèles-cascadés.

Certaines cellules possèdent deux sorties, la sortie passe-bande en bas et la sortie coupe-bande en haut, les autres cellules n'en possédant qu'une, la sortie coupe-bande. De plus, chaque cellule possède ses propres paramètres pour fixer la fréquence de résonance et surtout la largeur de la bande passante. Ainsi, il y aura un nombre différent de cellules accordées sur chaque harmonique. Il n'y a qu'une cellule accordée sur le même harmonique au premier niveau, alors qu'il y aura N cellules accordées sur le même harmonique au dernier niveau. De plus, comme chaque harmonique est éliminé des autres composantes spectrales du signal, l'ordre des cellules peut être modifié, la seule contrainte étant de filtrer, pour chaque sortie passe-bande, les harmoniques restants. Chaque sortie passe-bande du banc de filtres parallèles-cascadés sera alors libre de la puissances des autres harmoniques. Ce résultat est très intéressant lors de l'estimation des différents paramètres du CAN car il permet une meilleure estimation des paramètres spectraux par rapport à celle obtenue avec les bancs de filtres cascadés présentés dans la section précédente. Mais ce bénéfice a un coût : le banc de filtres contient à présent $\frac{N(N+1)}{2}$ cellules.
La fonction de transfert $H_{bp,i}$ de chaque sortie passe-bande $X_{bp,i}$ est donnée par :

$$H_{bp,i}(z) = H_i(z) \cdot \prod_{j=1,j\neq i}^{N} H_{nt,j} = H_i(z) \cdot \prod_{j=1,j\neq i}^{N} (1 - H_j(z)) \tag{3.13}$$

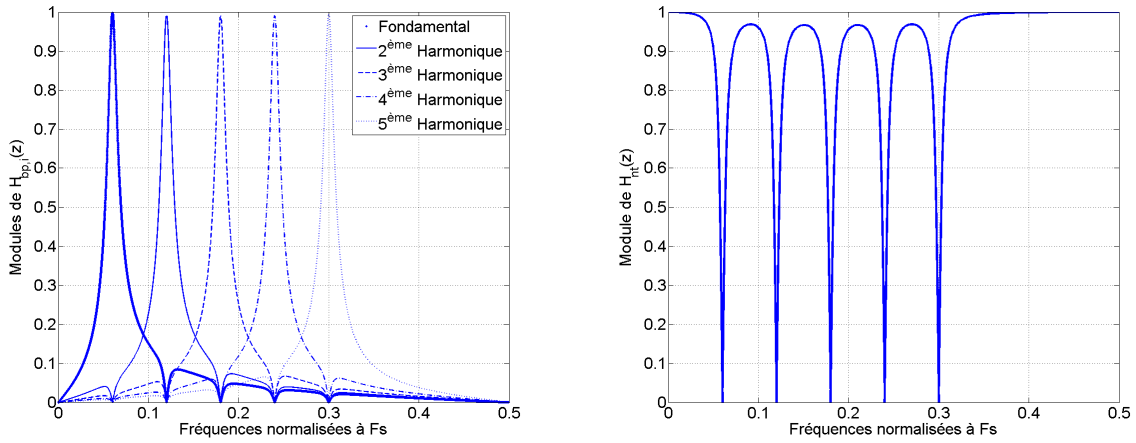
avec

$$H_i(z) = \frac{b_{0,i}(1 - z^{-2})}{1 + a_{1,i}z^{-1} + a_{2,i}z^{-2}} \tag{3.14}$$

où $b_{0,i}$, $a_{1,i}$ et $a_{2,i}$ sont les coefficients du filtre résonant à $f_{r,i}$. La fonction de transfert H_{nt} de la sortie coupe-bande X_{nt} est donnée par :

$$H_{nt}(z) = \prod_{i=1}^{N} H_{nt,i} = \prod_{i=1}^{N} (1 - H_i(z)) \tag{3.15}$$

H_{nt} est identique au banc de filtres cascadés. En effet, entre l'entrée du banc et la sortie X_{nt}, le signal passe par les mêmes cellules cascadés. Ces fonctions de transfert sont linéaires par rapport aux cellules de bases, le placement des pôles et des zéros de chaque cellule n'est pas modifié par les autres cellules. La figure 3.7 illustre la propriété essentielle de cette structure. En effet, elle représente les réponses en fréquence des sorties passe-bandes $X_{bp,i}$ correspondant aux raies harmoniques à $0.06F_s$, $0.12F_s$, $0.18F_s$, $0.24F_s$ et $0.30F_s$ (a) ainsi que la réponse en fréquence de la sortie coupe-bande X_{nt}. Nous constatons que pour chaque sortie $X_{bp,i}$, il y a un zéro de transmission sur toutes les raies harmoniquement liées à celle accordée sur la fréquence de résonance de la *ième* sortie. De plus, ces raies sont toutes éliminées de la sortie coupe-bande. Contrairement au banc de filtres cascadés, l'influence que pourrait avoir ces harmoniques sur le calcul de la puissance du signal des sorties $X_{bp,i}$ et X_{nt} est ainsi éliminée.



(a) Sorties passe-bandes

(b) Sortie coupe-bande

FIGURE 3.7 – Réponses en fréquence des sorties du banc parallèle-cascadé avec un fondamental à $0.06Fs$.

Les performances du banc de filtres parallèles-cascadés seront donc meilleures que celles des bancs de filtres cascadés mais le coût supplémentaire est clairement prohibitif dans le cadre du test embarqué. Il faut donc trouver une topologie de banc de filtres permettant d'atteindre de telles performances sans induire une trop forte augmentation des ressources nécessaires.

3.2.3 Banc de filtres parallèles

Cette topologie fut présentée par Martin et Sun [78]. Le banc est composé de N cellules biquadratiques (figure 3.8) parallèles délivrant N sorties passe-bandes $X_{bp,i}$ accordées sur les fréquences du fondamental et de ses harmoniques, et une sortie coupe-bande X_{nt} donnant la sortie associée au bruit.

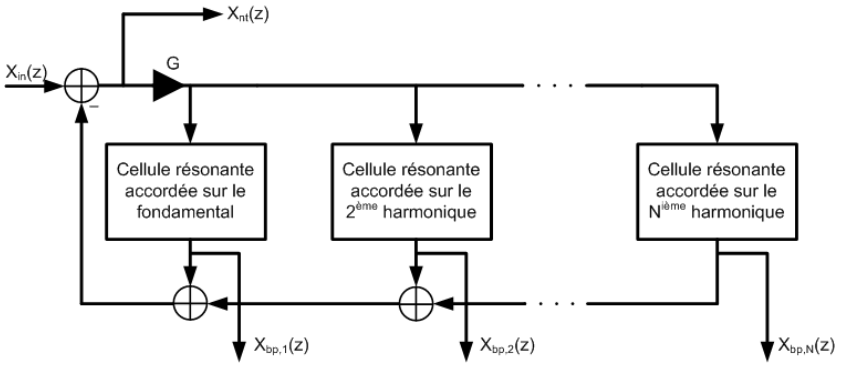


FIGURE 3.8 – Banc de filtre classique parallèles.

La fonction de transfert $H_{bp,i}$ de chaque sortie passe-bande est donnée par :

$$H_{bp,i}(z) = \frac{G.H_i(z)}{1 + G\sum_{j=1}^{N} H_j(z)} \tag{3.16}$$

où $H(i)$ correspond à la fonction de transfert associée à chaque cellule résonante. G est une constante multiplicative appelée "dumping factor" permettant d'assurer la stabilité du banc de filtres dont une valeur particulière ($G = 1/2N$) donne de bonnes propriétés statistiques des signaux de sorties [79]. La fonction de transfert H_{nt} de la sortie coupe-bande est donnée par :

$$H_{nt}(z) = \frac{G}{1 + G.\sum_{i=1}^{N} H_i(z)} \tag{3.17}$$

Ces fonctions de transfert ($H_{bp,i}$ et H_{nt}) ne sont pas des formes linéaires par rapport aux H_i, ce qui introduit de nouveaux pôles et zéros. Pour illustrer ces modifications, prenons comme exemple un banc composé de deux cellules :

$$H_{bp,i}(z) = \frac{G.H_i(z)}{1 + G(H_1(z) + H_2(z))} \tag{3.18}$$

Soient N_i et D_i, respectivement le numérateur et le dénominateur de la fonction de transfert H_i. En réécrivant $H_{bp,i}$ en fonction de N_i et D_i, il vient :

$$H_{bp,i}(z) = \frac{G.N_i(z).D_1(z).D_2(z)}{D_i(z).(D_1(z).D_2(z) + G[D_1(z).N_2(z) + D_2(z).N_1(z)])} \tag{3.19}$$

Nous pouvons alors constater que les zéros de $H_{bp,i}$ sont exactement les zéros de H_i ainsi que les pôles de H_1 et H_2. Quand aux pôles de $H_{bp,i}$, ce sont les pôles de H_i ainsi que les racines de $D_1(z).D_2(z) + G[D_1(z).N_2(z) + D_2(z).N_1(z)]$. Différentes structures de filtres LDI sont proposées par Padmanabhan et Martin [75, 79] pour la réalisation de la cellule résonante. En particulier, il y a la structure LDI undamped et une autre structure basée sur l'utilisation de filtres complexes conjugués du premier ordre. La structure LDI undamped permet la réalisation de deux fonctions de transfert, l'une (H_a) dont le facteur de qualité est indépendant du coefficient du filtre et l'autre (H_b) dont la largeur de bande est indépendante du coefficient :

$$H_{a,i}(z) = \frac{z^{-1}\left(1 - z^{-1}\right)}{1 - \left(2 - k_i^2\right) z^{-1} + z^{-2}} \tag{3.20}$$

$$H_{b,i}(z) = k_i \times H_{a,i}(z) = \frac{k_i z^{-1}\left(1 - z^{-1}\right)}{1 - \left(2 - k_i^2\right) z^{-1} + z^{-2}} \tag{3.21}$$

La pulsation de résonance de cette structure ne dépend que de k_i et vaut :

$$\Omega_{r,i} = 2\pi \frac{f_{r,i}}{F_s} = 2\sin^{-1}\left(\frac{k_i}{2}\right) \tag{3.22}$$

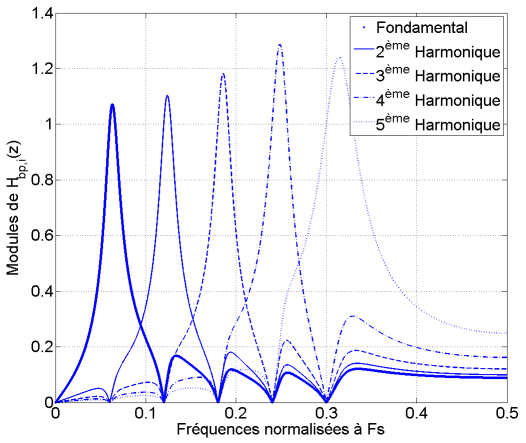
Le banc de filtres parallèles dont les cellules réalisent les fonctions de transferts H_a et H_b sont respectivement nommés banc de filtres parallèles A et B.

La deuxième structure proposée par Padmanabhan [75] réalise la fonction de transfert (H_c) suivante :

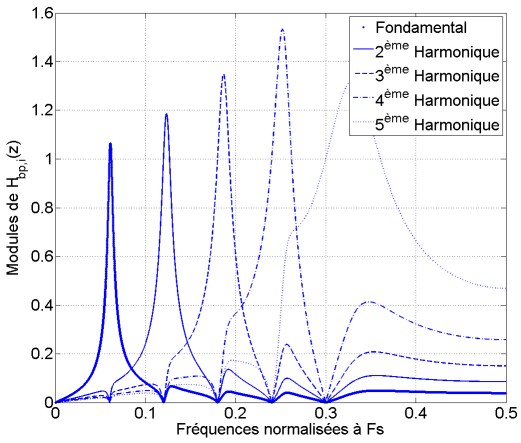
$$H_{c,i}(z) = \frac{2z^{-1}(a_i - z^{-1})}{1 - 2a_i z^{-1} + z^{-2}} \tag{3.23}$$

où $a_i = \cos\Omega_r = \cos(2\pi \frac{f_r}{F_s})$. On peut remarquer que $2a_i = 2 - k_i^2$. Le dénominateur des fonctions de transfert de ces structures sont donc identiques, ainsi que leurs pôles conjugués et donc leurs fréquences de résonance, si l'on ne tient pas compte des décalages dûs aux zéros de ces fonctions de transfert. Ce banc de filtres est stable à condition que $G < 1/N$ et $|a_i| < 1$ [78]. Le banc de filtres utilisant cette structure est nommé banc de filtres parallèles C. La figure 3.9 représente les réponses en fréquences de chaque sortie passe-bande des bancs de filtres parallèles A, B et C.

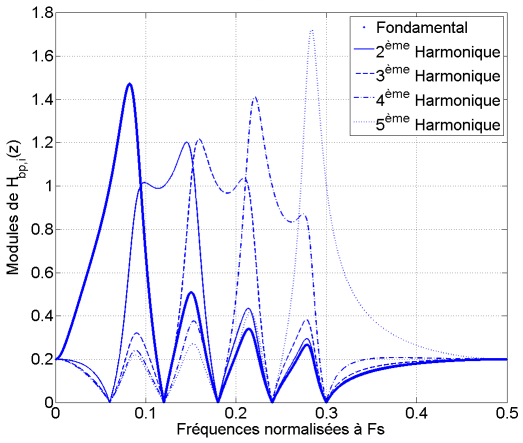
Le module de chaque fonction de transfert associée à chacune des sorties passe-bandes du banc de filtres vaux 1 à la fréquence de résonance de l'harmonique considéré et vaux 0 à la fréquence de résonance des autres harmoniques. Ainsi la séparation des puissances des différents harmoniques s'effectue correctement. Malheureusement, ces modules ont une amplitude supérieure à 1 pour de nombreuses fréquences, entraînant ainsi une amplification de la puissance du signal en dehors des harmoniques. En effet, chaque sortie passe-bande contiendra une partie de la puissance du bruit résiduel et la puissance calculée pour chacune de ces sorties passe-bandes sera alors sur-évaluée. De plus, toute la puissance du bruit résiduel incluse dans celle des harmoniques est déduite de la



(a) Banc A



(b) Banc B



(c) Banc C

FIGURE 3.9 – Réponses en fréquence des sorties passe-bandes des bancs de filtres parallèles A, B, et C accordées sur $0.06F_s$, $0.12F_s$, $0.18F_s$, $0.24F_s$ et $0.30F_s$, correspondant aux cinq premières harmoniques du spectre.

puissance du signal de bruit résiduel, entraînant ainsi une sous-évaluation de sa puissance calculée. Les estimations des paramètres spectraux seront alors fortement biaisées. Cette amplification n'est donc pas compatible avec le test Go/NoGo cherchant à évaluer le *SNR* ou le *THD* du CAN.

Pour pouvoir obtenir des réponses en fréquence comparable à celles du banc de filtres parallèles-cascadés, il faut revenir à une structure classique de filtre. La fonction de transfert de telles

structures est (cf. équation 3.8) :

$$H_{d,i}(z) = \frac{1-r^2}{2} \frac{(1-z^{-2})}{1-2r\cos\Omega_i z^{-1} + r^2 z^{-2}} \tag{3.24}$$

où Ω_i, la pulsation normalisée de résonance est proche de θ_i, l'argument des pôles conjugués et r correspond au module de ces pôles. Ici, la constante multiplicative G vaut 1. La réponse en fréquence associée à cette fonction de transfert admet un maximum (= 1) à proximité de la fréquence de résonance. Ce banc de filtres est nommé banc de filtres parallèles D. Malheureusement, l'utilisation directe de telles structures dans la topologie parallèle du banc de filtres, du fait de la rétroaction, les pôles et zéros des sorties passe-bandes sont modifiées. La séparation des différents harmoniques ne s'effectue pas correctement. La figure 3.10 représente les réponses en fréquences de chaque sortie passe-bande du banc de filtres parallèles D.

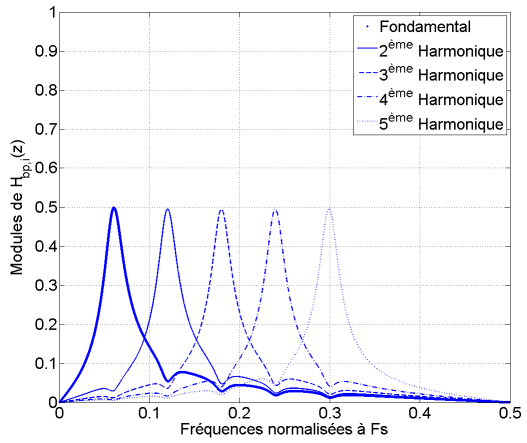


FIGURE 3.10 – Réponses en fréquence des sorties passes bandes du banc de filtres parallèles D, accordées sur $0.06F_s$, $0.12F_s$, $0.18F_s$, $0.24F_s$ et $0.30F_s$, correspondant aux cinq premières harmoniques du spectre.

Quelque soit la fréquence considérée, nous constatons que l'amplitude de la fonction de transfert est toujours plus petite que 1. Toutes les fréquences sont donc atténuées, y compris la fréquence de résonance. De plus, il n'y a pas de zéros de transmission au niveau des fréquences des autres harmoniques. Pour corriger ce défaut, il faut prendre en compte la rétroaction du banc de filtres pour calculer une nouvelle fonction de transfert pour chacune des cellules résonantes. La fonction de transfert pour chaque sortie passe-bande du banc de filtres que l'on souhaite obtenir est $H_{d,i}$. Par rapport aux fonctions de transfert des cellules résonantes $H_{e,i}$, $H_{d,i}$ s'exprime :

$$H_{d,i}(z) = \frac{H_{e,i}(z)}{1 + \sum_{j=1}^{N} H_{e,j}(z)} \tag{3.25}$$

où $H_{e,i}$ est la fonction de transfert de la cellule considérée et les $H_{e,j}$ sont les fonctions de transfert de toutes les cellules du banc de filtres (banc de filtres parallèles E). Malheureusement, prendre en compte l'influence des $N-1$ cellules sur tout le spectre rend le calcul complexe. C'est pourquoi, en première approximation, seule la cellule considérée est prise en compte. En effet, pour chaque fréquence de résonance d'une cellule, il y a un zéro de transmission dans la fonction de transfert des autres cellules. Son module est donc nul à ces fréquences :

$$H_{d,i}(z) = \frac{H_{e,i}(z)}{1 + H_{e,i}(z)} \tag{3.26}$$

Soient $N_{d,i}$, $D_{d,i}$, $N_{e,i}$ et $D_{e,i}$, respectivement les numérateurs et dénominateurs de la fonction de transfert que l'on souhaite réaliser pour chaque sortie passe-bande $X_{bp,i}$ du banc de filtres et de la fonction de transfert de la cellule considérée. Il vient alors :

$$\frac{N_{d,i}(z)}{D_{d,i}(z)} = \frac{\frac{N_{e,i}(z)}{D_{e,i}(z)}}{1 + \frac{N_{e,i}(z)}{D_{e,i}(z)}} \tag{3.27}$$

$$N_{e,i}(z) = N_{d,i}(z) \tag{3.28}$$

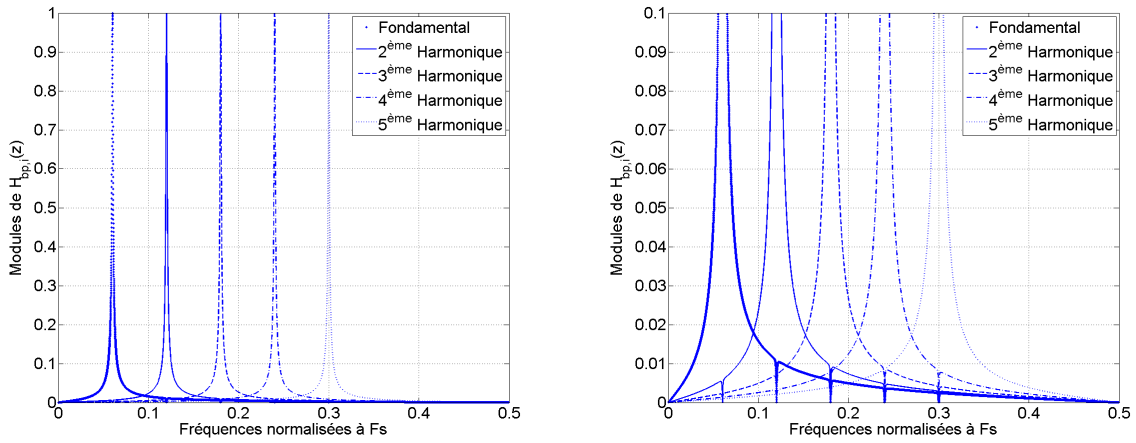
$$D_{e,i}(z) = D_{d,i}(z) - N_{d,i}(z) \tag{3.29}$$

ce qui correspond à :

$$H_{e,i}(z) = \frac{\frac{1-r^2}{1+r^2}(1 - z^{-2})}{1 - \frac{4r}{1+r^2}\cos\Omega z^{-1} + z^{-2}} \tag{3.30}$$

La figure 3.11 représente les réponses en fréquences de chaque sortie passe-bande du banc de filtres parallèles E ($G = 1/2N$). Ainsi, chaque sortie passe-bande ne contient que la puissance associée à l'harmonique considéré sans atténuation et coupe celle des autres harmoniques. Contrairement aux autres bancs de filtres, l'atténuation autour des fréquences de résonance est plus forte, ce qui pourrait s'expliquer par la structure même du banc de filtres en boucle qui augmente l'ordre de la fonction de transfert des sorties passe-bandes. En effet, d'après les équations 3.10 et 3.13, l'ordre des fonctions de transfert associées aux sorties passe-bandes des bancs de filtres cascadés et parallèles-cascadés sont respectivement $2k$ et $2N$, où k est le numéro de l'harmonique considéré et N le nombre total d'harmoniques pris en compte par le banc de filtres. Dans ces deux cas, l'ordre maximal de ces fonctions de transfert est $2N$. Dans le cas du banc parallèle, l'équation 3.19 montre que l'ordre des sorties passe-bandes est $2(N+1)$. Ainsi l'ordre du banc de filtres parallèle est supérieur à celui des autres bancs.

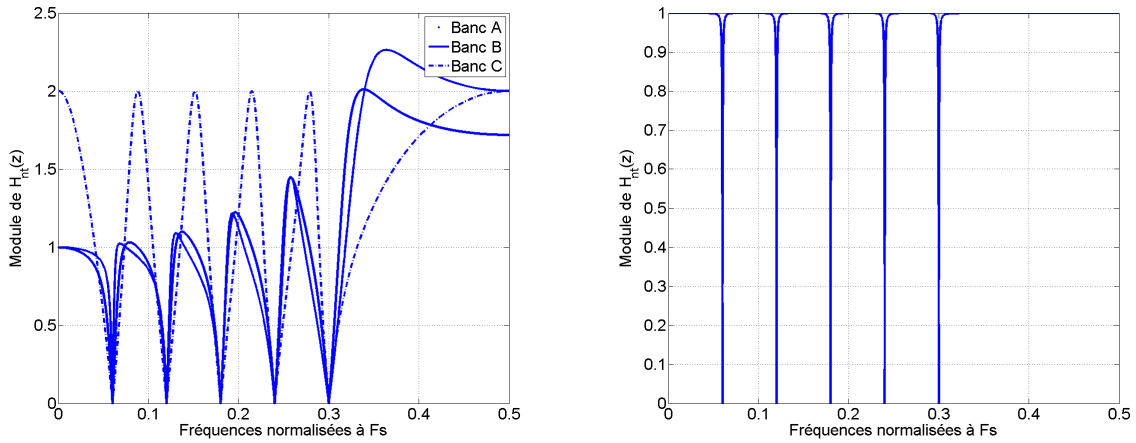
Intéressons nous maintenant aux fonctions de transfert des sorties coupe-bandes X_{nt} de ces bancs de filtres, représentées à la figure 3.12. Pour les bancs de filtres utilisant une structure LDI (figure 3.12(a)), bien que tous les harmoniques soient coupés, l'amplification de certaines fréquences entraînera un calcul erroné de la puissance de la sortie coupe-bande. A contrario, la



(a) Banc E (b) Banc E (Détail)

FIGURE 3.11 – Réponses en fréquence des sorties passe-bandes du banc de filtres parallèles E accordées sur $0.06F_s$, $0.12F_s$, $0.18F_s$, $0.24F_s$ et $0.30F_s$, correspondant aux cinq premières harmoniques du spectre.

figure 3.12(b) montre que les harmoniques sont coupées sans amplification ou atténuation du bruit résiduel pour le banc de filtres E.



(a) Banc A, B et C (b) Banc E

FIGURE 3.12 – Réponses en fréquence des sorties coupe-bandes des bancs de filtres parallèles A, B, C et E accordées sur $0.06F_s$, $0.12F_s$, $0.18F_s$, $0.24F_s$ et $0.30F_s$, correspondant aux cinq premières harmoniques du spectre.

Afin de quantifier les différents effets non souhaitables de ces différents bancs de filtres, on se propose de faire une simulation à partir d'un signal réel issu du CAN testé (et dont le spectre est représenté par la figure 3.13) et de calculer le *SNR* et le *THD* à partir de la puissance des harmoniques et du bruit résiduel. Ces *SNR* et *THD* calculés sont ensuite comparés aux *SNR* et *THD* obtenus par transformée de Fourier rapide (FFT). Six bancs ont été simulés en virgule flottante : le banc de filtres cascadés et cascadés inverse, le banc de filtres parallèles-cascadés, ces trois bancs utilisent des cellules classiques ; puis trois bancs de filtres parallèles (A, C et E). La table 3.1 résume les résultats de ces simulations.

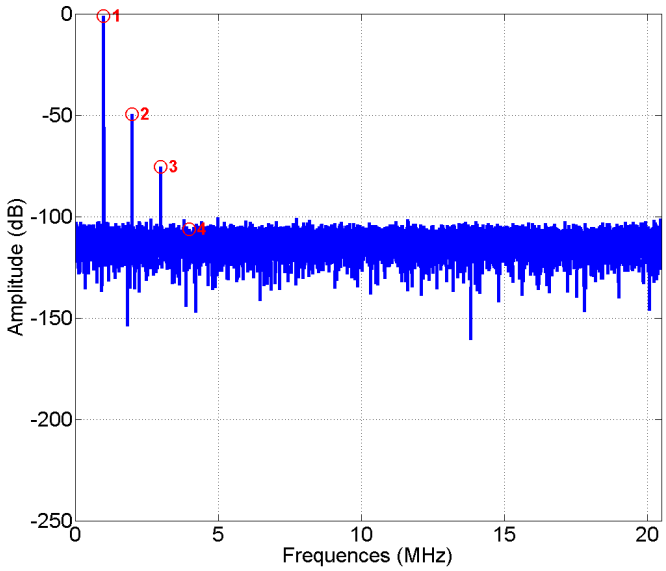


FIGURE 3.13 – Spectres du signal issu du CAN.

	FFT	cascadé	cascadés inverse	parallèle-cascadés	parallèle A	parallèle C	parallèle E
SNR	70,46	71,13	71,03	71,03	44,44	50,38	70,53
THD	-48,42	-48,74	-16,17	-48,79	-47,89	-48,14	-48,42

TABLE 3.1 – *SNR* et *THD* (en dB) obtenus par les différents bancs de filtres simulés.

La valeur très faible du THD obtenu à l'aide du banc de filtres cascadés inverse est due au fait que les derniers harmoniques contiennent une partie de la puissance des harmoniques précédents. Ces six bancs (mis à part le banc de filtres parallèles-cascadés) sont équivalant en terme de ressources à quelques opérateurs près. Plus précisément, un classement peut être établit en fonctions des ressources utilisées, du plus faible au plus élevé. Tout d'abord, les deux bancs de

filtres cascadés qui n'utilisent aucun opérateur entre les cellules, puis les trois bancs de filtres parallèles qui utilisent un multiplieur et $N+1$ additionneurs supplémentaires et enfin, le banc de filtres parallèles-cascadés qui utilise $N(N+1)/2$ cellules contre N pour les autres. Il s'agit donc de choisir celui qui permettrait d'avoir une estimation la plus proche du *SNR* et du *THD* calculés par FFT, tout en utilisant un minimum de ressources. Les deux bancs de filtres cascadés (inverse ou non) permettent une bonne estimation soit du *SNR*, soit du *THD* mais pas des deux en même temps. Bien que ces structures soient les plus simples, elles ne sont pas adéquates pour notre application. Le banc de filtres parallèle-cascadés permet de très bonnes estimations du *SNR* et du *THD*. Malheureusement, dans le cadre du test embarqué, la réalisation d'un tel banc de filtres est beaucoup trop pénalisante en termes de surface. Les banc de filtres parallèles A et C ne permettent pas d'avoir de bonnes estimations du *SNR* et du *THD*. Par contre, en utilisant des cellules réalisant la fonction de transfert H_e préalablement calculée, les estimations du *SNR* et du *THD* obtenues sont les plus proches des valeurs obtenues par FFT sans augmentation prohibitive des ressources nécessaire à sa réalisation. C'est donc le banc de filtres parallèles E que nous réaliserons pour la séparation des différentes composantes spectrales du signal issu du CAN dans la partie traitement du BIST.

3.3 Conclusion

Nous avons présenté, dans ce chapitre, le banc de filtres linéaires permettant l'extraction de paramètres spectraux du CAN. Dans un premier temps, une étude théorique du filtre rejecteur de composante harmonique (filtre Notch) a été détaillée en étudiant son élément fondamental : le filtre résonateur du second ordre.

Dans une seconde partie, trois topologies de banc de filtres ont été étudiées : filtres cascadés, parallèles-cascadés et parallèles. Après une première étude théorique sur les réponses en fréquence de ces structures, la structure parallèle a été choisie. Cette structure modifie sensiblement les fonctions de transfert passe-bandes du banc de filtres par rapport à celles des cellules. Une méthodologie a été présentée pour définir les nouvelles fonctions de transfert des cellules pour obtenir celles souhaitées au niveau des sorties passe-bandes. Une seconde étude, qualitative, a été réalisée pour comparer les performances d'estimation de deux paramètres dynamiques (le *SNR* et le *THD*). Cette étude a permis de sélectionner une topologie de banc (banc parallèle) ainsi qu'une structure de filtres dont la fonction de transfert est $H_{e,i}$. La topologie parallèle avec les cellules E est la seule permettant une bonne estimation de la puissance de l'harmonique considérée pour un coût minimum.

Nous allons nous intéresser dans le prochain chapitre à la réalisation matériel de ce banc de filtres.

Chapitre

4 Implémentation du banc de filtres linéaires

Sommaire

Introduction

Dans le chapitre précédent, nous avons décrit théoriquement le banc de filtres permettant une séparation efficace des différentes composantes spectrales du signal issu du CAN : le banc de filtres parallèles E. Nous allons maintenant voir comment réaliser une implémentation de ce banc de filtres.

Dans la première partie de ce chapitre, nous présentons différentes structures (structures classique basées sur la réalisation directe de la fonction de transfert et structures LDI (Lossless Discrete Integrator)) disponibles dans la littérature pour la réalisation de cette structure résonante.

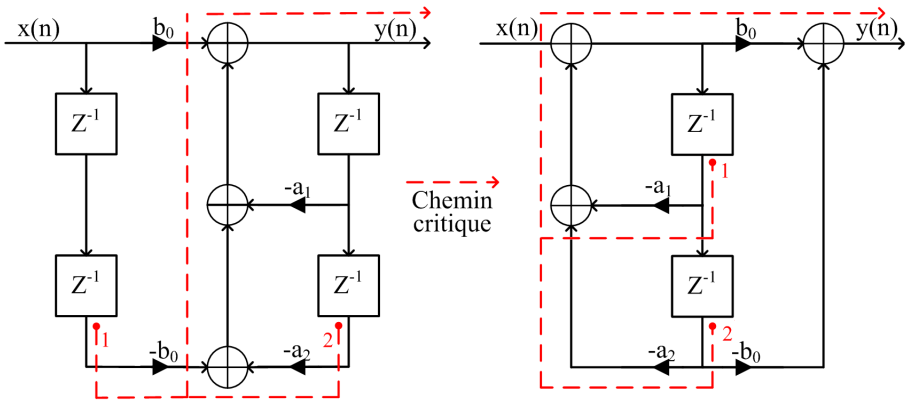
Puis nous présenterons, dans une seconde partie, l'implémentation des différentes structures de filtres ainsi que celle du banc choisi sur FPGA (Field Programmable Gate Array), circuit intégré programmable. Enfin, nous détaillerons les performances du banc de filtre conçu en réalisant une simulation post placement-routage du design sous ModelSim à partir d'une sinusoïde convertie par un CAN réel et un traitement des signaux de sortie sous MATLAB.

Enfin, dans une dernière partie, nous présenterons une étude du banc de filtres optimal ainsi qu'un algorithme décisionnel permettant d'obtenir toutes les informations nécessaires à la réalisation de ce banc de filtres optimal.

4.1 Etude des différentes structures d'implémentation d'une cellule

Il existe différentes structures, classiques ou non, pour la réalisation d'un filtre. Nous verrons ici les trois structures classiques (forme directe I et II et forme transposée) [77] ainsi que deux autres structures dites *Lossless Digital Integrator* (LDI) [80].

4.1.1 Structure en Forme Directe I & II



(a) Forme Directe I (b) Forme Directe II

FIGURE 4.1 – Structure du filtre résonateur en Forme Directe I & II

La fonction de transfert du filtre à implémenter est :

$$H_{bp}(z) = \frac{b_0(1 - z^{-2})}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{4.1}$$

avec $a_1 = -2r\cos\theta$ et $a_2 = r^2$. L'équation aux différences est donc :

$$y(n) = b_0 x(n) - b_0 x(n-2) - a_1 y(n-1) - a_2 y(n-2) \tag{4.2}$$

La Forme Directe I (Figure 4.1(a)) est construite à partir de l'équation aux différences 4.2. La sortie $y(n)$ est retardée en utilisant un délai (représenté par le symbole 'Z^{-1}') pour obtenir $y(n-1)$ qui lui-même est ensuite retardé d'une unité pour obtenir $y(n-2)$. Puis chaque sortie des délais est dirigée vers les additionneurs en utilisant les multiplicateurs adéquats (respectivement par $-a_1$ pour $y(n-1)$ et $-a_2$ pour $y(n-2)$). L'entrée $x(n)$ est multipliée par b_0 puis retardée conformément à l'équation aux différences 4.2 pour être dirigée vers les additionneurs comme le montre la figure 4.1(a).

La structure en Forme Directe I (Figure 4.1(a)) peut être vue comme deux sous-systèmes : une partie non-récursive, correspondant au numérateur de la fonction de transfert (à gauche) et une partie récursive, correspondant au dénominateur de la fonction de transfert (à droite). Or dans un système invariant dans le temps constitué de différents sous-systèmes cascadés, la relation entre l'entrée et la sortie est indépendante de l'ordre dans lequel les sous-systèmes sont cascadés. Il est donc possible de mettre la partie non-récursive en premier et de fusionner en un délai unique chaque couple de délais pour aboutir à la structure en Forme Directe II (figure 4.1(b)), réduisant ainsi la complexité du circuit.

Le chemin critique de la structure en Forme Directe I est composé d'un multiplieur et trois additionneurs (délai en bas à droite ou à gauche vers la sortie $y(n)$ sur la figure 4.1(a)), de même que celui de la structure en Forme Directe II (un des délais vers la sortie $y(n)$ en passant par l'additionneur le plus proche de l'entrée sur la figure 4.1(b)).

Ces structures ont un long chemin critique. De plus la structure en Forme Directe I utilise deux délais supplémentaires, augmentant ainsi la surface utilisée et donc la consommation d'un tel circuit. La structure en Forme Transposée devrait permettre de réduire la longueur du chemin critique tout en gardant un nombre minimal de délais. C'est ce qui est exposé dans la prochaine section.

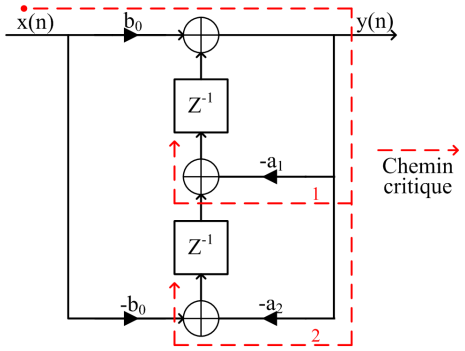


FIGURE 4.2 – Structure du filtre résonateur en Forme Transposée

4.1.2 Structure en Forme Transposée

La figure 4.2 décrit une forme modifiée des structures en Forme Directe : la Forme Transposée. Pour obtenir cette structure, il suffit d'écrire différemment l'équation aux différences 4.2 :

$$y(n) = b_0 x(n) - b_0 x(n-2) - a_1 y(n-1) - a_2 y(n-2) \tag{4.3}$$

$$y(n) = b_0 x(n) - b_0 x(n)\,\delta(n-2) - a_1 y(n)\,\delta(n-1) - a_2 y(n)\,\delta(n-2) \tag{4.4}$$

$$y(n) = b_0 x(n) + \left[(-b_0 x(n) - a_2 y(n))\,\delta(n-1) - a_1 y(n)\right]\delta(n-1) \tag{4.5}$$

où $\delta(n)$ est la fonction dirac. Ainsi les échantillons $x(n)$ et $y(n)$ sont multipliés par les coefficients associés avant d'être retardés. Le nombre de délais de cette structure est ainsi minimal (deux unités) et la longueur de son chemin critique est réduit à un multiplieur et seulement deux additionneurs (entrée vers un des délais en passant par un des multiplieurs par $-a_1$ ou $-a_2$ sur la figure 4.2).

Ces trois structures classique du résonateur sont sensibles à la troncature des coefficients lors de l'implémentation en virgule fixe. En effet, la représentation des coefficients $(a_1, a_2, b_0, b_1, b_2)$ sur un nombre fini de bits va considérablement modifier le placement des pôles et des zéros [81]. Une des solutions à ce problème consiste à utiliser une structure Lossless Discrete Integrator (LDI) moins sensibles à la troncatures des coefficients [75].

4.1.3 Structures Lossless Discrete Integrator

Les structures LDI du filtre résonateur, décrites par Padmanabhan [75, 80], permettent d'obtenir une très bonne sensibilité et une très bonne robustesse au bruit de quantification des coefficients. On distingue deux types de structures LDI possibles pour la cellule biquadratique.

4.1.3.1 LDI undamped

La première structure est basée sur une réalisation générale de la cellule du second ordre sans action ou commande de la largeur de bande B_{-3dB}. C'est un filtre résonateur dont la structure, proposée par [75], est dite "undamped" (non armortie) (Figure 4.3). Selon la sortie considérée

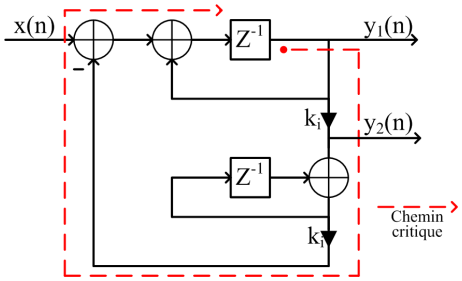


FIGURE 4.3 – Structure d'un LDI "undamped" de la cellule du second ordre.

Y_1 ou Y_2, la fonction de transfert associée sera respectivement :

$$H_1(z) = \frac{Y_1(z)}{X(z)} = \frac{z^{-1}\left(1 - z^{-1}\right)}{1 - \left(2 - k_i^2\right) z^{-1} + z^{-2}} \tag{4.6}$$

$$H_2(z) = k_i \times H_1(z) = \frac{Y_2(z)}{X(z)} = \frac{k_i z^{-1}\left(1 - z^{-1}\right)}{1 - \left(2 - k_i^2\right) z^{-1} + z^{-2}} \tag{4.7}$$

La sortie Y_1 permet d'avoir une bande passante approximativement indépendante du coefficient k_i alors que la sortie Y_2 permet d'avoir un facteur de qualité approximativement indépendant du coefficient k_i. Les fonctions de transfert associées à ces sorties sont celles qui ont été utilisées dans les bancs parallèles A et B au chapitre précédent (cf. section 3.2.3). Dans les deux cas, cette structure possède deux pôles sur le cercle unité lorsque $0 < k_i < 2$ et l'expression analytique de la pulsation de résonance ne dépend que de k_i :

$$\Omega_r = 2\pi \frac{f_r}{F_s} = 2 \sin^{-1}\left(\frac{k_i}{2}\right) \tag{4.8}$$

La structure résonne avec un coefficient de surtension qui dépend de k_i et aucune commande n'est disponible sur la largeur de la bande passante B_{-3dB} au niveau de la cellule. Dans le cas du banc de filtres parallèles, cette commande est réalisée par le coefficient G (figure 3.8). De plus, les zéros de cette structure sont positionnés en 0 et 1, contrairement à la fonction de transfert à implémenter (cf. équation 4.1).

4.1.3.2 LDI à largeur de bande constante

La deuxième structure étudiée est basée sur l'utilisation des filtres LDI permettant la commande de la largeur de bande [82]. Ce filtre LDI (Figure 4.4) permet un contrôle de la largeur de bande

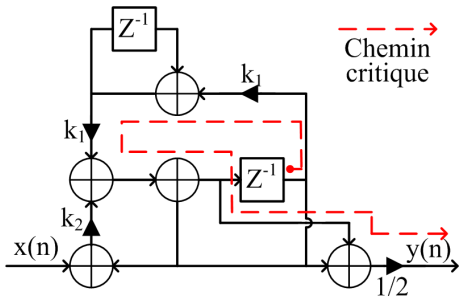


FIGURE 4.4 – Structure Lossless Discrete Integrator à bande passante constante du filtre résonateur

via un unique paramètre (k_2) qui est indépendant de celui déterminant la fréquence de résonance (k_1). Il est ainsi nommé "LDI à largeur de bande constante" dans la littérature [80]. La fonction de transfert de cette structure est :

$$H_{LDI}(z) = \frac{-k_2}{2} \times \frac{(1+z^{-1})(1-z^{-1})}{1-(2-k_2-k_1^2)z^{-1}+(1-k_2)z^{-2}} \tag{4.9}$$

Malheureusement, le coefficient associé à z^{-2} au dénominateur de cette fonction de transfert ne vaut jamais 1. En effet, dans ce cas précis, $k_2 = 1 - r^2$ est nul ce qui équivaux à $r = 1$. Le dénominateur de la fonction de transfert devient donc $2\cos\Omega - 1$ et le numérateur est nul. La fonction de transfert devient donc nulle, ce qui est une situation aberrante. Or la fonction de transfert que l'on cherche à réaliser possède un coefficient associé à z^{-2} au dénominateur valant 1. Nous ne pourrons donc pas utiliser cette structure dans notre banc.
Pour la réalisation de notre banc de filtres, nous utiliserons la structure en forme transposée utilisant un minimum de ressource et dont le chemin critique est le plus court.

4.2 Implémentation du banc de filtres

Après avoir présenté les différentes structures pour l'implémentation du filtre Notch, nous allons maintenant détailler l'implémentation de chaque cellule élémentaire, ainsi que celle du banc de filtres. Pour illustrer nos propos, nous utiliserons comme cas réel, le signal issu du CAN AD9042D d'Analog Devices [83]. L'entrée du filtre sera alors sur 12 bits, en format signé, échantillonnée à une fréquence $F_s = 41$ MHz.

4.2.1 Détermination des coefficients des filtres et leur codage

Pour le choix de la fréquence du fondamental dans le cadre du BIST, il existe trois possibilités :

– fixer arbitrairement la fréquence du fondamental et paramétrer le générateur de signal et le banc de filtres pour cette fréquence,
– rechercher la fréquence pour laquelle le générateur sera optimal,
– rechercher la fréquence pour laquelle le banc de filtres sera optimal.

Ces trois choix correspondent pour l'implémentation du banc de filtres à deux cas : réaliser l'implémentation du banc de filtres pour une fréquence arbitrairement fixée ou réaliser le banc de filtres optimal avec une fréquence choisie pour optimiser le codage des coefficients.
La première partie de cette étude va considérer le cas où la fréquence est choisie arbitrairement, ici $f_0 \approx 0.998$ MHz en mode cohérent sur 16384 points. L'expression des coefficients est rappelée dans le tableau suivant :

Filtre	b_0	b_1	b_2	a_1	a_2
transposé	$\frac{1-r^2}{1+r^2}$	0	$-b_0 = \frac{r^2-1}{1+r^2}$	$-2\cos\theta$	1

TABLE 4.1 – Coefficients des filtres résonateurs en fonction des coordonnées polaires des pôles de la fonction de transfert

Pour simplifier les différentes implémentations, b_0 et b_2 sont choisis sous la forme de puissance de 2, ainsi les multiplications par b_0 et b_2 seront réalisées par de simples décalages. Soit $k \in \mathbb{Z}^*, b_0 = 2^{-k} = \frac{1-r^2}{1+r^2}$. Alors $r^2 = \frac{1-2^{-k}}{1+2^{-k}}$. Avec $\theta = 2\pi\frac{f_0}{F_s}$, les coefficients deviennent alors :

Filtre	b_0	b_1	b_2	a_1	a_2
transposé	2^{-k}	0	-2^{-k}	$-2\cos(2\pi f_0)$	1

TABLE 4.2 – Coefficients des filtres résonateurs

Le coefficient b_0 dépend directement du module r des pôles de la fonction de transfert. Or, lorsque r est proche de 1, la largeur de bande passante devient de plus en plus étroite (équation 3.7). Il faut donc choisir une valeur de b_0 permettant d'avoir une valeur de r proche de 1 :

$$b_0 = 2^{-k} = \frac{1-r^2}{1+r^2} \tag{4.10}$$

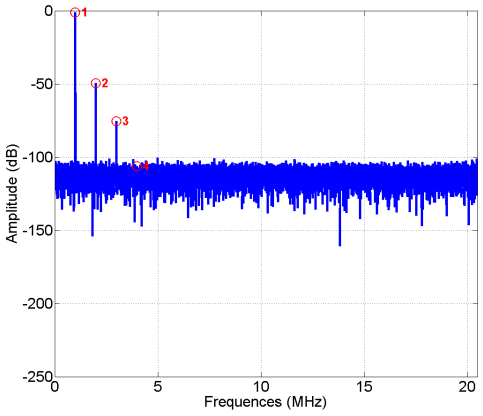
donc lorsque r tend vers 1, b_0 tend vers 0, ce qui équivaux à k tendant vers l'infini quand r tend vers 1.
Pour chaque valeur de k, la largeur de bande du filtre varie. Plus k est grand et plus la bande est étroite (r tendant alors vers 1). Or, plus la bande est étroite, plus la mesure de la puissance

de la fréquence filtrée sera précise puisque la puissance autour de la fréquence de résonance est atténuée. Mais, d'après l'équation 3.6, lorsque r tend vers 1, la position de la fréquence de résonance du filtre s'approche de la position recherchée diminuant le biais de la mesure [51]. Ces deux effets affecterons donc la mesure effectuée. Au final, plus la largeur de bande sera étroite, plus la précision sur la fréquence de résonance devra être élevée. Dans notre cas, nous cherchons à savoir si le CAN testé fonctionne dans une plage d'utilisation prédéfinie. D'après sa documentation, le CAN AD9042D d'Analog Devices est un convertisseur traitant 41 méga échantillons par seconde (MSPS). Sa résolution est de 12 bits et la valeur typique du *SNR* du signal issu du CAN est de 69 dB (datasheet).

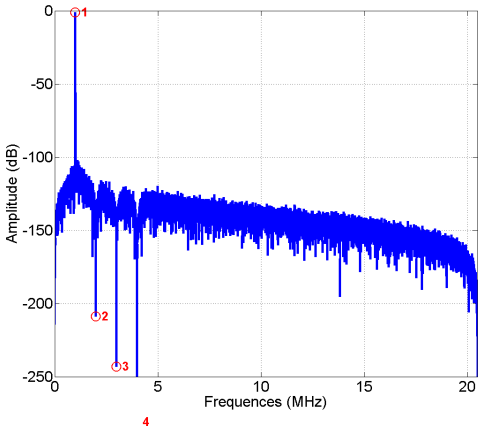
Pour déterminer la valeur de r, nous avons utilisé une méthode empirique qui consiste à tracer l'estimation du *SNR* en fonction de r. Puis selon l'intervalle de précision, nous choisirons la valeur de r la plus petite qui assure cette précision.

La première étape est d'obtenir une estimation de référence pour le *SNR*. Celle-ci est calculée à partir d'une FFT sur le signal issu du CAN (70.48 dB). Nous considérerons que ce CAN n'est pas défaillant tant que les défauts de conversion n'induisent pas une erreur supérieure à un demi *LSB*, ce qui correspond à une erreur de 3 dB sur l'estimation du *SNR*. Nous considérerons donc que pour toute valeur du *SNR* dont la mesure corrigée (non biaisée) est comprise entre 67.5 et 70.5 dB, le CAN testé est valide. Ensuite, dans le but de déterminer le biais de la mesure en fonction du module des pôles de la fonction de transfert r et donc de k, le banc de filtres est simulé en virgule flottante, codage offrant la plus grande précision, sous MATLAB/Simulink.

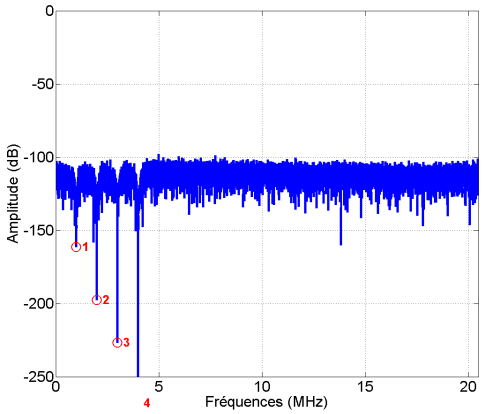
La figure 4.5 montre les spectres des signaux en entrée et en sortie du banc de filtres alors que la figure 4.6 montre le biais de l'estimation faite par filtrage par rapport à la mesure par FFT. On constate sur la figure 4.5 une bonne séparation des différentes composantes harmoniques du signal d'entrée et leur absence du bruit résiduel (figure 4.5(c)), ce qui valide la méthode de calcul du *SNR* en utilisant la puissance de chacun de ses signaux.



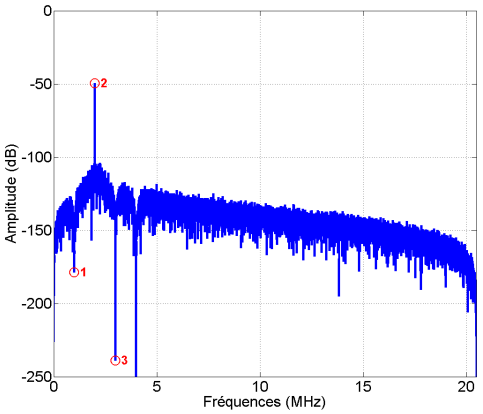
(a) Entrée



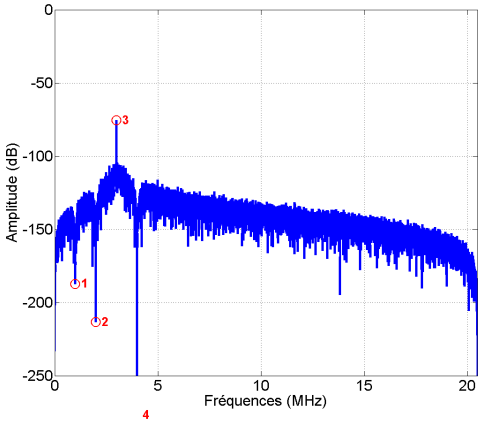
(b) Fondamental



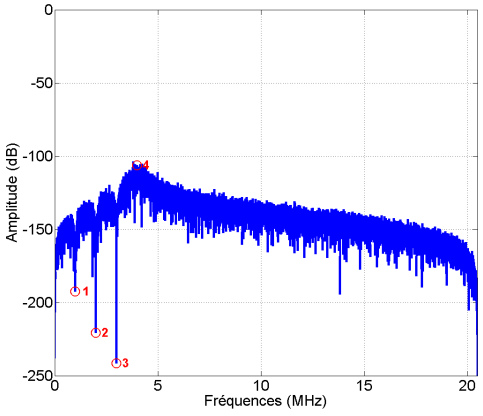
(c) Bruit



(d) $2^{ème}$ harmonique



(e) $3^{ème}$ harmonique



(f) $4^{ème}$ harmonique

FIGURE 4.5 – Spectres des signaux en entrée et sortie du banc de filtres

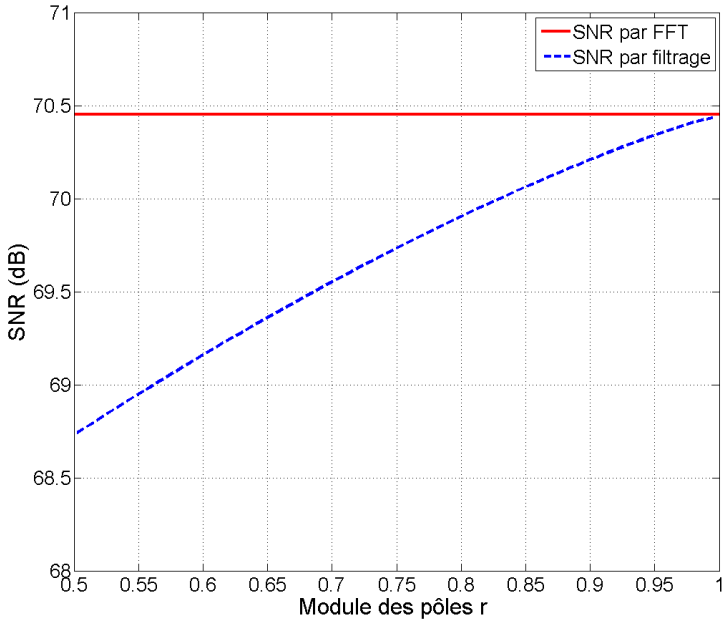


FIGURE 4.6 – Rapport signal à bruit en fonction du module des pôles de la fonction de transfert des cellules du banc de filtres.

Maintenant que le biais de mesure en fonction de r est connu, le domaine d'acceptabilité de l'estimation du *SNR* en fonction de r peut être défini. Nous tolérons 3 dB d'erreur sur l'estimation du *SNR* par rapport au *SNR* calculé par FFT. Pour chaque valeur de r, il faut donc enlever de cette marge de 3 dB le biais de l'estimation.

Parmi les coefficients, a_1 sera le seul à être approximé lors de l'implémentation en virgule fixe. Or la fréquence de résonance du filtre dépend de a_1 et va donc subir un décalage. Ce décalage affectera, tout comme la largeur de bande du filtre, l'estimation du *SNR* du signal filtré.
Le banc de filtres est maintenant simulé en virgule fixe, avec différentes tailles pour le codage des coefficients et les *SNR* correspondants sont calculés. Les résultats sont présentés sur la figure 4.7 en fonction de la taille du radix des coefficients pour différentes valeurs de k. Le radix d'un nombre binaire correspond au codage de la partie décimale de ce nombre. Nous constatons alors que le point appartenant au domaine d'acceptabilité minimisant la taille des coefficients est $k = 2$ pour une taille de 18 bits pour les radix des coefficients.
Les coefficients qui seront implémentés et leurs erreurs de quantification valent donc, avec une précision de 4 chiffres après la virgule :

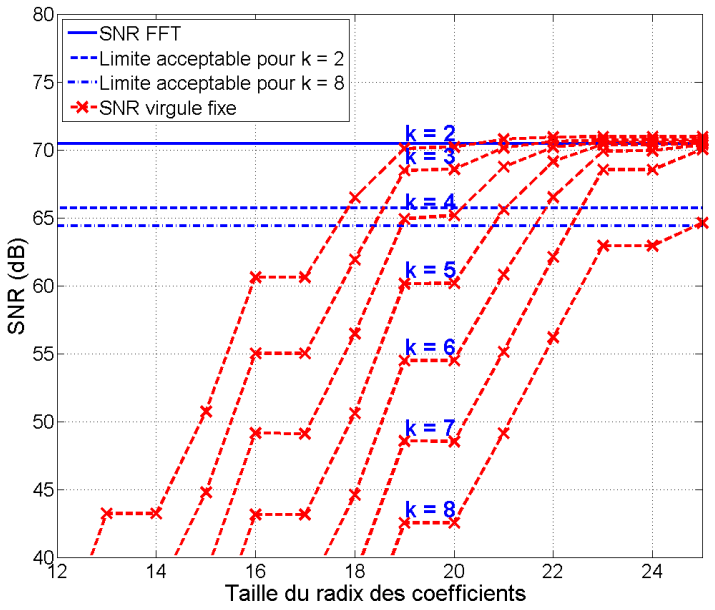


FIGURE 4.7 – Rapport signal à bruit en fonction de la taille du radix des coefficients et de k.

	b_0	b_1	b_2	a_1	a_2
	$2^{-2} = 0.25$	0	$-2^{-2} = -0.25$	≈ -1.9766	1
erreur	0	0	0	$\approx 2.22 \times 10^{-5}$	0

TABLE 4.3 – Coefficient des filtres résonateurs

Le radix de a_1 est donc codé sur 18 bits. Il faut y ajouter un bit pour la partie entière ($a_1 \in]-2, 2[$) et un bit de signe. Le format de codage de a_1 est donc $(20, 1, 18)$. La sortie du CAN AD9042D est un signal sur douze bits signé, le format d'entrée du banc de filtres est donc $(12, 11, 0)$. $-a_2$ vaut -1 et ne nécessite donc pas d'opérateur pour réaliser sa multiplication, seulement un inverseur. b_0 et $b_2 = -b_0 = -2^{-2}$) sont des puissances de 2 négative, ce qui implique que multiplier par b_0 et b_2 revient à faire un décalage à droite de deux bits. Les multiplications par b_0 et b_2 n'utilisent donc pas d'opérateur. De plus, le signal du bruit est multiplié par une constante normative avant l'entrée des cellules. Cette constante G vaut $1/8 = 2^{-3}$. Cette multiplication, comme les précédentes se fera donc par un décalage à droite de trois bits du signal du bruit. Une précision maximale pour les signaux internes (de manière à ne pas inclure de bruit de quantification supplémentaire) sera exigée. Il faut donc conserver tous les bits des radix des signaux multipliés. Ainsi leur radix sera sur 18 $(0 + 18)$ bits, la partie entière sera codée sur 12 $(1 + 11)$ bits et ainsi que ceux des sorties dont le format sera $(31, 12, 18)$. La sortie de chaque cellule étant sur 31 bits, l'entrée de la cellule suivant devra également être sur 31 bits. Le signal

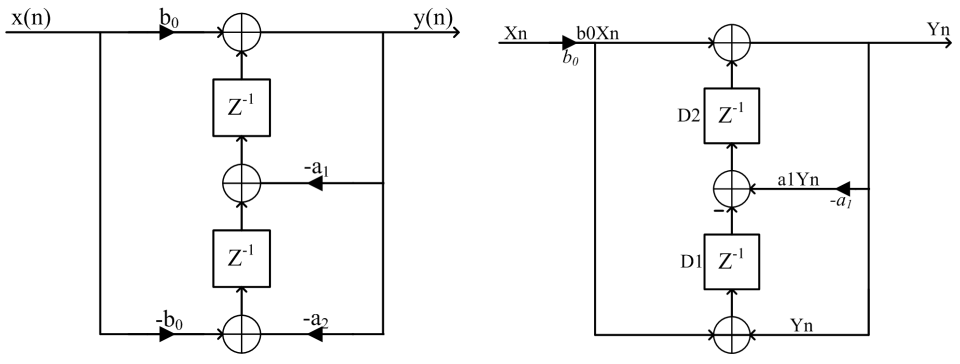
d'entrée aura donc un nouveau format : $(31, 12, 18)$.

4.2.2 Architecture pipeline ou architecture combinatoire ?

Il existe deux architectures possibles pour réaliser ces cellules, correspondants à deux types d'unité de calcul : combinatoire ou pipeline. Une unité de calcul combinatoire réalise sa fonction par propagation des signaux dans le circuit, alors que dans une unité de calcul pipeline, certains opérateurs sont séparés du suivant par un registre. Par rapport à une unité de calcul combinatoire, l'introduction de registres aux points critiques du circuit permet de réduire les chemins critiques et donc d'augmenter la fréquence d'horloge, mais le coût en ressources nécessaires le sera aussi, entraînant également une plus grande consommation d'énergie. Or, dans le cadre du BIST, une surface et une consommation minimale sont recherchées. Des unités de calculs combinatoires seront donc utilisées prioritairement pour la réalisation des filtres linéaires. Mais nous souhaitons traiter les échantillons à la volée, sans avoir à les mémoriser. Cette contrainte nous impose de calculer les valeurs des sorties en moins de $T_s = 1/F_s \approx$ 24.4 ns après l'arrivée de l'échantillon correspondant. Si les cellules combinatoires permettent d'atteindre de telles performances, nous les utiliserons, sinon, il faudra utiliser leur version pipeline.

4.2.3 Architecture d'une cellule Notch

Une cellule reçoit en entrée le signal à filtrer (Xn), une horloge ($clkS$) à 41 MHz, cadençant les signaux d'entrée et de sortie et un signal de reset asynchrone (rst). En sortie, la cellule délivre le signal d'entrée dont la composante spectrale considéré est isolée grâce au filtre résonateur (Yn) au même format que l'entrée. A partir de la structure en Forme Transposée présentée précédemment (figure 4.8(a)), diverses optimisations peuvent être apportés en vue de l'implémentation. La figure 4.8(b) présente la cellule résonante en Forme Transposée avec ces optimisations.



(a) théorique (b) implémentée

FIGURE 4.8 – Structure théorique et implémentée du filtre résonateur en Forme Transposée

– Les coefficients multiplicateurs de Xn (b_0 et b_2) sont égaux au signe près. Le signal Xn peut donc être multiplié par b_0 avant d'être séparé et envoyé vers les deux additionneurs et ainsi supprimer un multiplieur. De plus, b_0 est une puissance de 2, donc multiplier par b_0 revient à faire un décalage, ce qui n'utilise pas d'opérateur. Le signal ainsi obtenu est nommé *b0Xn* (figure 4.2) est donc maintenant disponible à l'entrée des additionneurs.
– Le coefficient a_1 est négatif, donc multiplier le signal Yn par $-a_1$ revient à faire une multiplication par $|a_1|$, ce qui est réalisé.
– Le coefficient $a_2 = 1$, donc pour réaliser la multiplication par $-a_2$, il suffit d'inverser le signal Y_n
– Le contenu du registre *D1* doit être égal à $(-b_0X_n) + (-Y_n)$. Mais pour l'instant, le contenu de *D1* vaut $b_0X_n + (-Y_n)$. Il suffit alors de ne pas multiplier Yn par -1 pour que le contenu de *D1* devient $b_0X_n + Y_n$. Par contre, il faut maintenant soustraire le contenu du registre *D1* au signal *a1Yn* pour que le contenu du registre *D2* vaille $-b_0X_n + (-Y_n) + (-a_1Y_n)$.
– Finalement, *D2* est ajouté à *b0Xn* pour obtenir *Yn*.

A chaque arrivée d'un nouvel échantillon à traiter (front montant sur *clkS*), les signaux *Xn*, *D1*, *D2* et *Yn* sont mis à jour. Pour pouvoir mettre à jour ces signaux, il faut que leurs valeurs respectives soient calculées, c'est à dire que les opérations sur le chemin critique (chemin le plus long entre deux retards ou entrée ou sortie) de cette architecture soient réalisées. Le chemin critique de cette structure est le chemin permettant d'avoir *D2* à partir de *Xn*. Il contient la multiplication par a_1 et deux additionneurs. Ces trois opérations devront donc être réalisées en un temps inférieur à 24.4 *ns*.

Chaque cellule est finalement décrite de façon comportementale en VHDL puis synthétisée avec l'outil ISE 10.1 de Xilinx pour un FPGA Xilinx Virtex IV. Ensuite,le fonctionnement de la cellule est vérifié par une simulation comportementale pré-synthèse sous ModelSim. Chaque synthèse est réalisée en n'utilisant que les slices, mais ni la RAM distribuée, ni les blocs DSP, ni les blocs de RAM disponibles dans le FPGA. On retrouvera la description du contenu des slices à la section 2.4.3.2. La table 4.4 résume les résultats de synthèse des cellules accordées sur chaque harmonique.

La fréquence d'horloge maximale de cette architecture est environ à 100 MHz, ce qui est conforme à notre contrainte ($F_{max} > 41$ MHz). La surface utilisée par une cellule est en moyenne de 129 slices, de 62 slices Flip-Flop et et 439 Look Up Table (LUT) à 4 entrées. Les opérations les plus consommatrices en surface sont les multiplications par a_1. Il est a noter que la surface d'un multiplieur dépend du nombre de '1' dans l'écriture binaire du coefficient multiplicateur ce qui

Harmonique	Nb de slices	Nb de slices FF	Nb de LUT4	Fréq. Max. (MHz)
Fondamental	209	62	398	105
Deuxième	219	62	420	106
Troisième	280	62	542	104
Quatrième	207	62	398	106

TABLE 4.4 – Résultat de la synthèse architecturale des cellules accordées sur chaque harmonique.

explique les différences constatées dans la table 4.4 pour l'occupation du FPGA des différentes cellules. Ce nombre de bits différents de '0' (ou égaux à '1') dans l'écriture binaire d'un nombre est appelé poids de Hamming (P_H). Les multiplieurs sont donc modifiés pour les rendre plus efficaces en terme de surface.

Puisque les coefficients du filtre sont constants, différentes méthodes peuvent être envisagées pour réaliser ces multiplications. Une première solution consiste à enregistrer préalablement dans une mémoire la valeur du produit pour chaque entrée, celle-ci servant à adresser la mémoire. Dans notre cas, il faudrait une mémoire de $31 * 2^{31}$ bits, soit plus de 66 Go, ce qui n'est pas adéquat à notre application. Une deuxième solution est l'utilisation des multiplieurs "shift-and-add" [84]. Ils sont plus rapides et moins consommateurs de surface que les multiplieurs classiques à base de LUT. Il existe différents algorithmes permettant d'obtenir un nombre minimal d'addition/soustraction. Le problème ayant pour but de trouver la décomposition en additions/soustractions et décalages d'une multiplication par une constante est connu sous le nom de *Multiplication par une constante unique*, qui est un problème NP-complet [85].
Pour résoudre ce problème, une première méthode naïve basée sur la méthode de multiplication manuelle est proposée : soit X, un mot binaire, A une constante codée sur N bits, $A(i)$ le $(i+1)^{ième}$ bits de A et $P_H(A)$ le poids de Hamming de A, alors :

$$A \times X = \sum_{i=0}^{N-1} (X << i).A(i) \tag{4.11}$$

Il y a $P_H(A)$ bits (les $A(i)$) non nuls, donc pour calculer $A \times X$, il faut $P_H(A)$ décalages et $P_H(A) - 1$ additions. Par exemple, si la constante A vaut 39, la multiplication de cette constante par X, réprésenté ci-dessous :

$$39_{10} \times X = 100111_2 \times X = X << 5 + X << 2 + X << 1 + X \tag{4.12}$$

nécessite uniquement trois additions. Cette solution, bien que simple, est rarement une solution optimale.

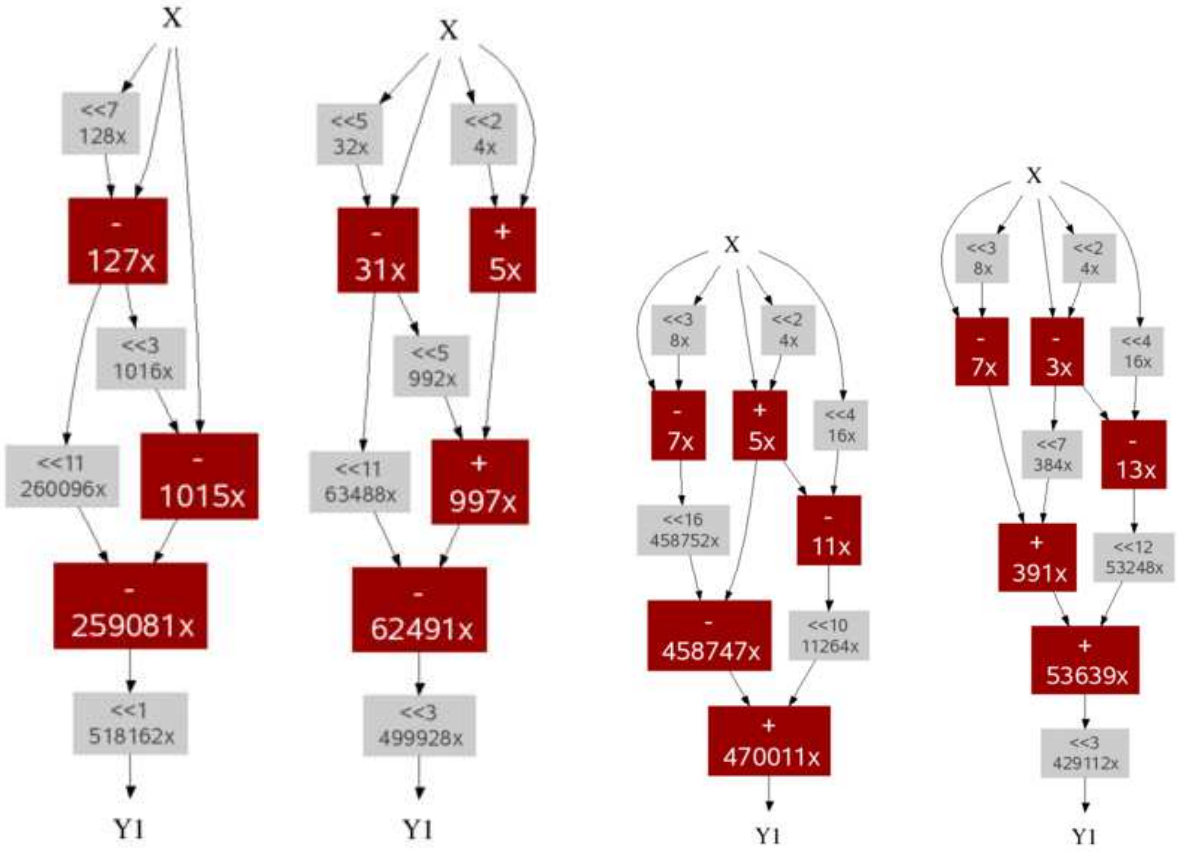
Une deuxième solution, permettant de réduire le nombre d'opérations nécessaires à la réalisation de la multiplication et donc la taille du multiplieur, est d'utiliser la représentation canonique CSD (Canonic Signed Digit) qui autorise des digits négatifs ($\overline{1}$). Toutes chaînes de '1' d'une longueur supérieure ou égale à 3 dans l'écriture binaire de ces coefficients peut être recodé en utilisant le recodage de Booth, réduisant le nombre d'additions dans les multiplieurs *Shift-and-Add*. En effet, le principe du recodage de Booth vient de la propriété suivante :

$$\forall i, j \in N^*, i < j, \sum_{k=i}^{j} 2^k = 2^{j+1} - 2^i \tag{4.13}$$

Ainsi, toute chaine de '1' de longueur $(j - i)$ dans l'écriture binaire d'un nombre peut être remplacer par une chaîne de '0' de longueur $(j - i - 1)$, plus un bit à '1' en poids fort et un bit à '-1' en poids faible. Pour chaque bit à '-1', l'addition correspondant sera remplacée par une soustraction. Notre exemple peut alors se réduire à deux additions/soustractions :

$$100111_2 \times X = 10100\overline{1}_{CSD} \times X = X << 5 + X << 3 - X \tag{4.14}$$

Il existe des algorithmes beaucoup plus complexes permettant de trouver une meilleure solution comme le BHM [86], le RAG-n [87] et le Hcub [84]. Nous avons utilisé le générateur en ligne du site web Spiral [1] proposant ces trois algorithmes pour définir nos multiplieurs. Pour chacune des quatre constantes, l'algorithme Hcub est utilisé avec deux optimisations : une profondeur de chemin minimale et une taille également minimale pour les additions/soustractions intermédiaires. La figure 4.9 montre la sortie de l'algorithme Hcub pour les multiplieurs des différentes cellules.



(a) Fondamental (b) Deuxième harmonique (c) Troisième harmonique (d) Quatrième harmonique

FIGURE 4.9 – Structure des multiplieurs générés par l'algorithme Hcub [1]

La table 4.5 donne les résultats de la synthèse des multiplieurs *Shift-and-Add* en fonction de l'harmonique et de l'algorithme considérés :

Pour les cellules accordées sur le fondamental et le quatrième harmonique, les multiplieurs obtenus par l'algorithme BHM sont ceux qui occupent le moins de place, alors que pour les cellules accordées respectivement sur les deuxième et troisième harmoniques, ce sont les multiplieurs obtenus par les algorithmes Hcub et RAG-n. Nous utiliserons ces structures optimale pour la réalisation de notre banc de filtres.

Harmonique	Algorithme	Nb de slices	Nb de LUT4	Délai Min. (ns)
Fondamental	BHM	67	129	10.218
	Hcub	67	129	10.218
	RAG-n	66	129	10.175
Deuxième	BHM	85	165	11.235
	Hcub	78	150	10.299
	RAG-n	82	158	11.344
Troisième	BHM	93	181	11.227
	Hcub	96	189	10.539
	RAG-n	78	153	11.473
Quatrième	BHM	76	149	11.210
	Hcub	89	170	10.358
	RAG-n	81	151	11.451

TABLE 4.5 – Résultat de la synthèse architecturale des multiplieurs *Shift-and-Add*.

4.2.4 Architecture du banc de filtres

Nous avons vu précédemment que l'entrée du banc se compose du signal issu du convertisseur et que les différentes sorties sont les différentes composantes harmoniques, ainsi que le bruit induit par la quantification et d'autres sources (thermique, jitter,...). L'entrée du banc de filtres est la sortie du CAN AD9042D d'Analog Devices [83] et elle est au format 12 bits signé, à une fréquence d'échantillonnage $Fs = 41\ MHz$. La sortie d'une cellule filtrante est au format 31 bits signé cadencée à $Ts = 1/Fs = 24.4\ ns$. De plus, seules les quatre premières harmoniques du signal sont considérées, il y a donc cinq signaux de sortie.

La structure interne du banc du filtre, décrite au paragraphe 3.2.3, se compose de quatre cellules biquadratiques résonant à une des fréquences harmoniques, de quatre additionneurs/soustracteurs et une multiplication par $1/8 = 2^{-3}$ pour le signal *Bruit*. Cette multiplication s'effectue par un décalage à droite de 3 bits. De plus, le signal *Bruit* décalé est le signal d'entrée des cellules, signal qui est également décalé mais de 2 bits vers la droite (multiplication par b_0). Tous ces décalages seront donc réalisés en même temps par un décalage de 5 bits vers la droite. La figure 4.10 décrit la structure interne du banc de filtres implémenté.

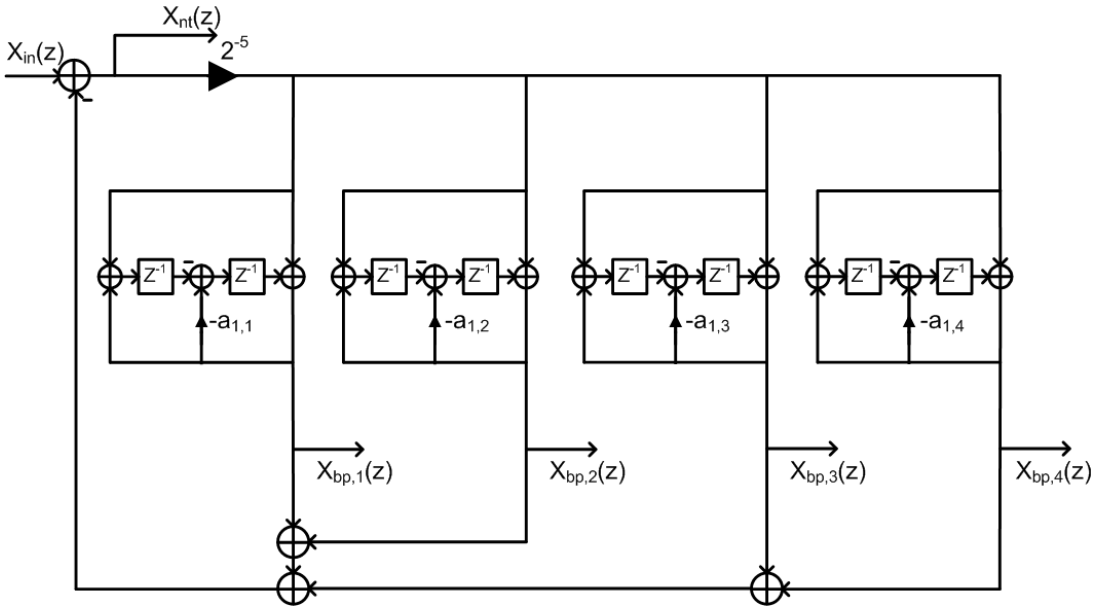


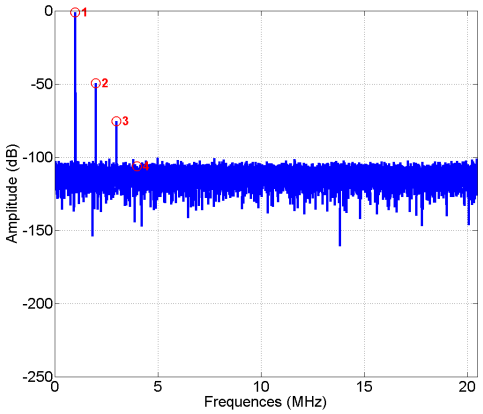
FIGURE 4.10 – Architecture interne du banc de filtres implémenté.

La table 4.6 résume les résultats de synthèse du banc de filtres utilisant les multiplieurs *Shift-and-Add* avec les optimisations décrites dans le paragraphe précedent.

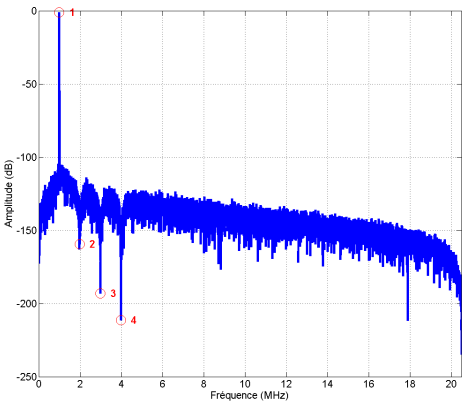
Nb de slices	Nb de slices FF	Nb de LUT4	Fréq. Max. (MHz)
345	274	652	130

TABLE 4.6 – Résultat de la synthèse architecturale du banc de filtres utilisant les multiplieurs *Shift-and-Add*.

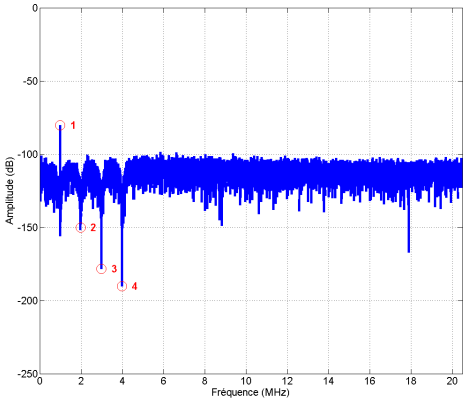
Pour valider notre conception du banc de filtres, une simulation post placement-et-routage du design a été réalisée sous ModelSim. Une sinusoïde issue du CAN AD9042D d'Analog Devices à la fréquence cohérente proche de 1 MHz est utilisé comme signal de test. Puis, les différents signaux de sortie du banc de filtres sont récupérés pour en faire une analyse spectrale (figure 4.11) à l'aide d'outils développés dans l'environnement MATLAB.



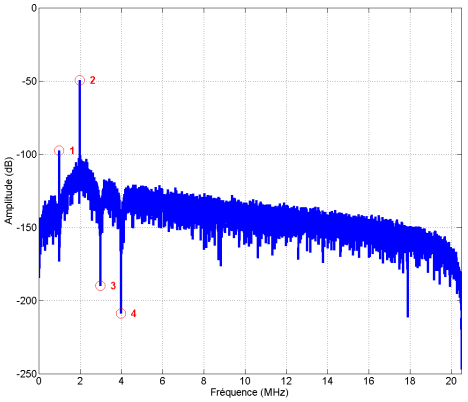
(a) Entrée



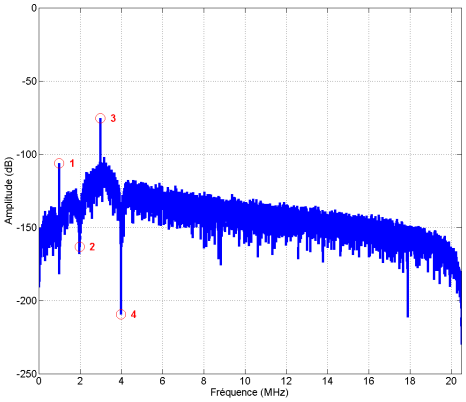
(b) Fondamental



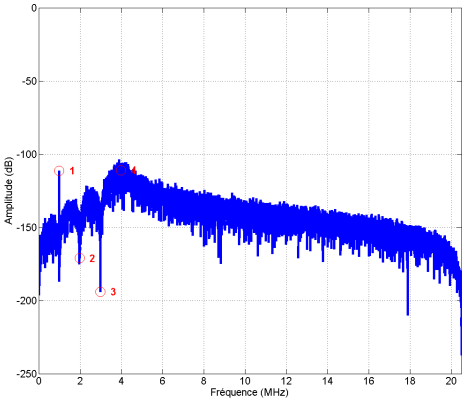
(c) Bruit



(d) $2^{ème}$ harmonique



(e) $3^{ème}$ harmonique



(f) $4^{ème}$ harmonique

FIGURE 4.11 – Spectres des signaux en entrée et sortie du banc de filtres

Le spectre du signal d'entrée (signal issu du CAN) est représenté sur la figure 4.5(a). Le plancher de bruit est d'environ −100 dB. Les trois premières harmoniques sont visibles avec des pics respectivement à 0 dB, −50 dB et −70 dB, soit une hauteur par rapport au plancher de bruit de, respectivement, 100 dB, 50 dB et 30 dB.
Le spectre de la sortie filtrée correspondant au fondamental est représenté à la figure 4.11(b). Seule la raie fondamentale reste à un niveau équivalent par rapport à son niveau d'entrée. Le plancher de bruit est en moyenne à −150 dB. Les autres raies harmoniques ont des niveaux inférieurs à −170 dB soit 20 dB sous le plancher de bruit.
Les spectres correspondants aux sorties filtrées des harmoniques 2, 3 et 4 sont représentés aux figures 4.11(d), 4.11(e) et 4.11(f). Le même type de spectre est alors observé sur les quatre figures. La raie spectrale correspondant à l'harmonique considérée a le même niveau que la raie correspondante à l'entrée du banc de filtres. Il est à remarquer que le fondamental a un niveau allant de −100 dB pour le deuxième harmonique à −120 dB pour le quatrième, ce qui correspond à une atténuation allant de 100 dB à 120 dB. La présence du fondamental à des niveaux supérieurs au plancher du bruit dans les spectres des harmoniques 2 à 4 et du bruit est un défaut du à l'implémentation en virgule fixe du banc de filtres. En effet, la valeur des coefficients implémentés est légèrement différente de la valeur théorique de ces mêmes coefficients. Les différences d'atténuation du fondamental dans ces signaux est due à la distance spectrale entre le fondamental et l'harmonique considéré. En effet, plus le fondamental va être proche de l'harmonique considéré et donc de la fréquence de résonance du filtre passe bande centré sur l'harmonique considéré, et plus l'atténuation du filtre sera faible.
Enfin, le spectre du bruit résiduel est représenté à la figure 4.11(c). Seul le fondamental à un niveau supérieur à celui du plancher de bruit, toutes les autres composantes harmoniques sont suffisamment amorties pour avoir un niveau inférieur. Ces résultats montrent que malgré ce défaut, le banc de filtres sépare correctement les différentes composantes spectrales du signal d'entrée. A partir des spectres des sorties du banc de filtres, le *SNR* obtenu est de 69.35 dB. La table 4.7 le compare aux *SNR* obtenus par d'autres méthodes :

Méthode	Datasheet	FFT	Banc de filtres idéal	Banc de filtres en virgule fixe
SNR (dB)	68	70.46	70.53	69.35

TABLE 4.7 – Rapports signal sur bruit en dB du CAN AD4096D.

Le *SNR* obtenu par le banc de filtres est proche de celui calculé par simulation en virgule flottante. Plus précisément, le *SNR* calculé à partir des signaux filtrés par le banc de filtres en virgule fixe codé sur un nombre limité de bits correspond au *SNR* calculé par FFT avec une erreur inférieure à 3 dB correspondant à un demi LSB. Cette précision est suffisante pour réaliser un test Go/NoGo

sur un convertisseur. De plus, ce banc de filtre peut fonctionner à une fréquence de 130 MHz, permettant ainsi d'étendre son utilisation à un CAN ayant une fréquence d'échantillonnage plus grande que le CAN AD4096D d'Analog Devices. Nous disposons donc maintenant d'un outil d'analyse fonctionnel d'un CAN dans le cadre du BIST.

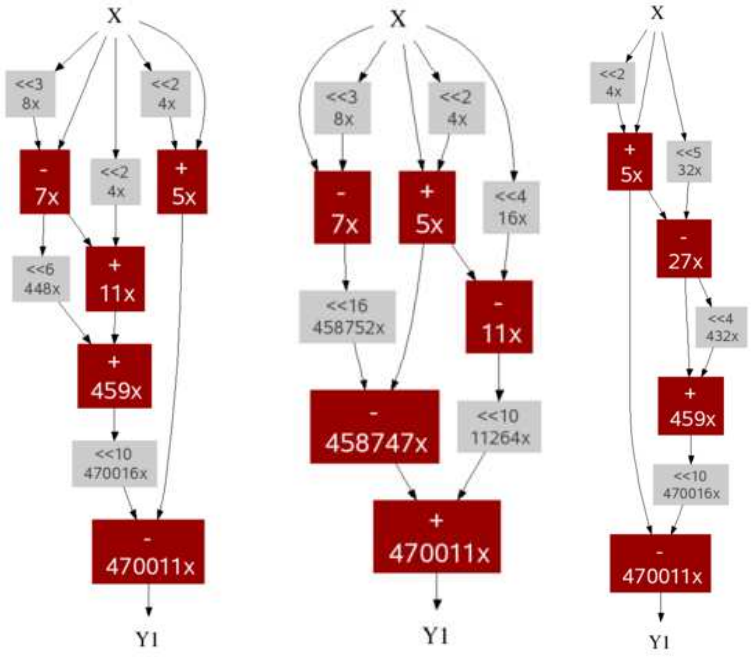
4.3 Etude du banc de filtres optimal

Après avoir réaliser un premier banc de filtres avec une fréquence du fondamental du signal issu du CAN choisie arbitrairement, nous allons maintenant présenter une méthodologie pour définir un banc de filtres optimal dans le cadre du test embarqué. Cela doit aboutir à une structure occupant le minimum de surface (et donc consommant le moins d'énergie).
La surface d'un filtre numérique est directement dépendante de trois critères : la taille des signaux internes, la taille et le type des opérateurs. Pour chaque type d'opérateur, les deux premiers critères sont liés à la précision des calculs effectués. Pour les additionneurs/soustracteurs, leur taille dépend principalement de la taille des signaux en entrées, c'est a dire de la précision des calculs. Pour les multiplieurs, la surface utilisée dépend aussi de la valeur du coefficient multiplicateur constant. Celle ci est définie en fonction de la largeur de bande et de la fréquence de résonance du filtre réalisé. Il n'existe pas de relation directe entre cette valeur et la surface du multiplieur optimal. Mais il est possible, en utilisant les algorithmes de résolution du problème de la *Multiplication par une constante unique*, de connaître le nombre d'opérateurs et la profondeur de l'arbre utilisés.
La surface d'une cellule dépend donc de la précision des calculs, de la largeur de bande et de la fréquence de résonance du filtre réalisé, soit trois degrés de liberté pour l'optimisation. Pour chaque valeur de la fréquence du fondamental, tout comme nous l'avons fait précédemment (cf. partie 4.2.1), les contraintes imposées au système de test vont imposer une largeur de bande et un codage pour les coefficients. A partir de ce codage, il conviendra ensuite de déterminer la structure minimale des multiplieurs (nombre d'additions/soutractions, profondeur de l'arbre) à l'aide des algorithmes Hcub, RAG-n et BHM.
Pour pouvoir faire un choix, il faut au préalable définir une métrique de comparaison. Nous proposons d'utiliser une fonction coût J définie comme étant la somme des dynamiques de sortie des additionneurs/soustracteurs intermédiaires. Par exemple, pour les multiplieurs de la figure 4.12, la valeur de leur fonction coût est :

$$\text{BHM} \quad : \quad 34[7x] + 33[5x] + 34[11x] + 40[459x] + 50[470011x] = 191 \tag{4.15}$$

$$\text{Hcub} \quad : \quad 34[7x] + 33[5x] + 33[11x] + 50[458747x] + 50[470011x] = 202 \tag{4.16}$$

$$\text{RAG-n} \quad : \quad 33[5x] + 36[27x] + 40[459x] + 50[470011x] = 161 \tag{4.17}$$



(a) BHM (b) Hcub (c) RAG-n

FIGURE 4.12 – Structure des multiplieurs générés par les algorithmes BHM, Hcub et RAG-n pour un multipliant de 470011

La table 4.8 indique les résultats de la synthèse de ces trois multiplieurs. On constate alors que plus leur fonction coût est faible, moins les ressources utilisées sont nombreuses.

Algorithme	Nb de slices	Nb de LUT4	Délai Min. (ns)
BHM	93	181	11.227
Hcub	96	189	10.539
RAG-n	78	153	11.473

TABLE 4.8 – Résultat de la synthèse architecturale des multiplieurs par 470011.

Il s'agit donc de choisir la fréquence qui permet de minimiser cette fonction coût. A partir des caractéristiques du CAN (résolution, fréquence d'échantillonnage) et d'un signal issu de ce même CAN, voici l'algorithme décisionnel permettant de déterminer la taille des signaux, la fréquence du fondamental et la structure des multiplieurs à utiliser pour la réalisation du banc de filtres optimal en fonction de la précision souhaitée :

Pour chaque fréquence testée, faire :

1. Déterminer le codage des coefficients en utilisant la méthodologie de la partie 4.2.1.
2. Coder chaque coefficient.
3. Calculer les valeurs de la fonction coût J de chaque multiplieur généré par les algorithmes BHM, Hcub et RAG-n.
4. Choisir les structures minimisant la fonction coût J pour chaque harmonique.
5. Additionner les valeurs de la fonction coût de chaque harmonique pour déterminer la fonction coût du banc du filtre.

Choisir la fréquence minimisant la fonction coût du banc de filtre.

4.4 Conclusion

Nous avons présenté dans ce chapitre l'implémentation du banc de filtres linéaires permettant l'extraction de paramètres spectraux du CAN. Dans un premier temps, cinq structures d'implémentation ont été présentées : les structures en Forme Directe I et II, la structure en Forme Transposée, la structure Lossless Discrete Integrator "undamped" sans commande de la largeur de bande et la structure Lossless Discrete Integrator à largeur de bande constante. Une des ces structures a ensuite été sélectionnée : la structure en Forme Transposée qui est la structure utilisant le moins de ressources.

L'implémentation du banc de filtres pour une fréquence du fondamental du signal issu du CAN fixée à 1 MHz a été ensuite détaillée dans la seconde partie. Dans un premier temps, la détermination du codage des coefficients a été réalisé, puis les architecture d'implémentation de la structure du banc de filtres ont été optimisée. Dans un troisième temps, l'architecture du banc a été réalisée sur FPGA puis validé à travers la simulation post-placement et routage, à partir d'une sinusoïde convertie par un CAN réel et un traitement des signaux de sortie sous MATLAB. Le *SNR* ainsi obtenu correspond au *SNR* obtenu en simulation en virgule flottante du banc de filtres avec une erreur de 1 LSB, précision suffisante pour permette de vérifier le fonctionnement correct du CAN.

Enfin, dans la cinquième et dernière partie, une étude du banc de filtres optimal ainsi qu'un algorithme décisionnel permettant d'obtenir toutes les informations nécessaires à la réalisation de ce banc de filtres optimal à partir de la résolution et de la fréquence d'échantillonnage du

convertisseur ont été présentés.

Chapitre

5 Filtrage adaptatif

Sommaire

Introduction

Dans le chapitre 3, l'unité d'extraction des paramètres spectraux du CAN a été réalisée par un banc de filtres linéaires. Lors de la réalisation des filtres du second ordre, la pulsation de résonance voulue subit un décalage, d'autant plus faible que le module des pôles de la fonction de transfert de ce filtre est proche de 1. Mais l'implémentation en virgule fixe du filtre implique une limitation de la précision de la valeur des coefficients, qui s'ajoute à la première approximation. De ce fait, le filtrage du signal issu du CAN n'est pas optimal. Pour améliorer celui-ci, nous proposons d'utiliser le principe du filtrage adaptatif. Le deuxième avantage de l'utilisation d'un

filtrage adaptatif est de permettre un suivi des raies spectrales si celles-ci sont décalées par des perturbation externes aux circuits du BIST.
Grâce au développement du traitement numérique du signal à partir des années 60 et à l'accroissement de la puissance des calculateurs, les méthodes adaptatives en traitement de signal ont connu un essor considérable permettant l'implémentation en temps réel d'algorithmes de plus en plus complexes à des cadences de plus en plus élevées [88].
Dans la première partie de ce chapitre, nous rappelons les bases théoriques du filtrage adaptatif et plus particulièrement la méthode du gradient stochastique ou LMS (Least Mean Square).
Nous présenterons, dans un second temps, les modification apportées aux différentes structures précédemment présentées pour obtenir des filtres adaptatifs.
Enfin dans une dernière partie, nous nous intéresserons à la simulation et à l'implémentation de ces différentes structures de filtres adaptatifs sur FPGA puis nous comparerons leurs performances par rapport à celles des structures linéaires (ou non adaptatives).

5.1 Rappels théoriques du filtrage adaptatif

Avant de décrire le principe du filtrage adaptatif, nous commencerons par exposer le principe du filtrage de Wiener sur lequel se base la filtrage adaptatif.

5.1.1 Principe et théorie du filtrage optimal de Wiener

Le filtrage optimal de Wiener consiste à rechercher un filtre linéaire optimal qui permet d'extraire d'un signal d'observation des composantes correspondant à un signal dit "utile". La figure 5.1 détaille le principe de fonctionnement de ce type de filtre. Les signaux utilisés sont : $x(n)$ processus aléatoire observé, $d(n)$ processus désiré ou signal de référence, $y(n)$ estimation de $d(n)$ obtenue par filtrage et $e(n)$ erreur d'estimation. Le filtre optimal de Wiener est le filtre

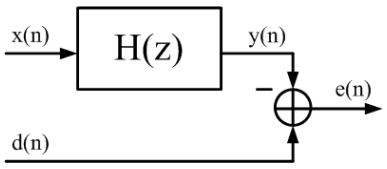


FIGURE 5.1 – Filtrage optimal de Wiener.

permettant d'avoir une erreur d'estimation "aussi petite que possible". Il convient de traduire l'expression "aussi petite que possible" dans un sens mathématique et plus particulièrement statistique. Il s'agit d'optimiser la conception du filtre en minimisant une fonction coût J ou index de performance. Il existe plusieurs définitions possibles pour cette fonction coût [89] :

1. la valeur quadratique moyenne de l'erreur d'estimation,
2. l'espérance de la valeur absolue de l'erreur d'estimation,
3. l'espérance de la puissance troisième ou plus, de la valeur absolue de l'erreur d'estimation.

La première option a un avantage par rapport aux deux autres, car elle permet l'utilisation d'outils mathématiques. En particulier, le choix de l'erreur quadratique moyenne implique une dépendance au second ordre de la fonction coût par rapport aux coefficients inconnus de la réponse impulsionnelle du filtre. De plus, cette fonction coût possède un minimum strict définissant de manière unique le filtre optimal. Ainsi, nous utiliserons cette définition pour la fonction coût.

Soit $h_n, n \in \mathbb{N}$ la réponse impulsionnelle du filtre. La sortie du filtre à l'instant n pour un signal d'entrée $x(n)$ s'exprime alors :

$$y(n) = \sum_{k=0}^{\infty} h_k x(n-k), \qquad n \in \mathbb{N} \tag{5.1}$$

Nous considérons que les signaux $x(n)$ et $d(n)$ sont des processus stochastiques stationnaires au sens large, ce qui implique que leur matrice d'autocorrélation est non-singulière. Nous considérons également que ces signaux sont à moyenne nulle. Si ce n'est pas le cas, il suffit de soustraire leurs moyennes aux signaux avant filtrage. Enfin, nous considérons ici des filtres à valeurs réelles, c'est à dire : $\forall n \in \mathbb{N},\ h_n \in \mathbb{R}$. L'estimation $y(n)$ de $d(n)$ est naturellement associée à une erreur d'estimation définie par la différence :

$$e(n) = d(n) - y(n) \tag{5.2}$$

La fonction coût J, que nous choisissons donc comme étant la valeur quadratique moyenne de l'erreur d'estimation, est :

$$J = E\Big[|e(n)|^2\Big] = E\Big[e^2(n)\Big] \tag{5.3}$$

où $E\Big[\_\Big]$ représente l'opérateur statistique de l'espérance. Il faut maintenant déterminer les conditions sous lesquelles J atteint son minimum. Ceci est obtenu lorsque l'équation 5.4 est satisfaite :

$$\nabla_k J = \frac{\partial J}{\partial h_k} = 0, \qquad k \in \mathbb{N} \tag{5.4}$$

Sous ces conditions, le filtre ainsi obtenu est dit optimal au sens de l'erreur quadratique moyenne. D'après l'équation 5.3, la fonction coût J est un scalaire indépendant de n. En utilisant l'expression de J dans l'équation 5.4, il vient :

$$\nabla_k J = E\Big[\frac{\partial e^2(n)}{\partial h_k}\Big] \tag{5.5}$$

$$= 2E\Big[e(n)\frac{\partial e(n)}{\partial h_k}\Big] \tag{5.6}$$

En utilisant l'équation 5.2, la dérivée partielle devient :

$$\begin{aligned}\frac{\partial e\,(n)}{\partial h_k} &= \frac{\partial}{\partial h_k}\Big(d\,(n) - y\,(n)\Big) && (5.7)\\ &= -\frac{\partial y\,(n)}{\partial h_k} && (5.8)\\ &= -\frac{\partial}{\partial h_k}\left(\sum_{k=0}^{\infty} h_k x\,(n-k)\right) && (5.9)\\ &= -x\,(n-k) && (5.10)\end{aligned}$$

En utilisant ce résultat dans l'équation 5.6, il vient :

$$\nabla_k J = -2E\Big[e(n)x(n-k)\Big] \tag{5.11}$$

Soit e_o, la valeur de l'erreur d'estimation lorsque le filtre est dans les conditions optimales ($\nabla_k J = 0$), il vient alors :

$$E\Big[e_o(n)x(n-k)\Big] = 0, \quad k \in \mathbb{N} \tag{5.12}$$

L'équation 5.12 décrit le principe d'orthogonalité, qui signifie que :

> *la fonction coût J atteint son minimum si et seulement si la valeur de l'erreur d'estimation correspondante $e_o(n)$ est orthogonale à tous les vecteurs d'entrée entrants dans l'estimation de la réponse désirée à l'instant n.*

En injectant les équations 5.1 et 5.2 dans celle du principe d'orthogonalité, il vient :

$$E\left[\left(d\,(n) - \sum_{i=0}^{\infty} h_{oi} x\,(n-i)\right) x\,(n-k)\right] = 0, \quad k \in \mathbb{N} \tag{5.13}$$

où h_{oi} est le $i$ième coefficient de la réponse impulsionnelle du filtre optimal. Cette réponse impulsionnelle h_o peut être finie ou infinie, d'où la borne supérieure de la somme valant l'infinie. En développant et réarrangeant les termes, il vient :

$$\sum_{i=0}^{\infty} h_{oi} E\Big[x(n-i)x(n-k)\Big] = E\Big[d(n)x(n-k)\Big], \quad k \in \mathbb{N} \tag{5.14}$$

Les deux espérances de l'équation 5.14 peuvent être interprétées comme suit :

- L'espérance $E\Big[x(n-i)x(n-k)\Big]$ est égale à la fonction d'autocorrélation de l'entrée du filtre pour un décalage de $i-k$. On la note $r(i-k)$.
- L'espérance $E\Big[d(n)x(n-k)\Big]$ est égale la fonction d'intercorrélation de l'entrée du filtre et de la réponse désirée pour un décalage de $-k$. On la note $p(-k)$.

Les fonctions d'autocorrélation et d'intercorrélation sont indépendantes du temps n car x et d sont des processus stochastiques discrets stationnaires. Ainsi l'équation 5.14 peut être réécrite :

$$\sum_{i=0}^{\infty} h_{oi} r(i-k) = p(-k), \quad k \in \mathbb{N} \tag{5.15}$$

Ce système d'équations définit les coefficients du filtre optimal et est appelé "équations de Wiener-Hopf". La résolution de ce système permet d'obtenir les valeurs optimales des coefficients de la réponse impulsionnelle du filtre. Dans le cas où la réponse impulsionnelle est finie ($\exists M \in \mathbb{N}, \forall m \geq M, h(m) = 0$), les équations 5.15 de Wiener-Hopf sont réduites à un système fini d'équations :

$$\sum_{i=0}^{M-1} h_{oi} r(i-k) = p(-k), \qquad k \in [\![0,\ M-1]\!] \tag{5.16}$$

Soit $\mathbf{x}(n) = [x(n),\ x(n-1),\ \ldots\ x(n-M+1)]^T$, le vecteur d'entrée. Soit $\mathbf{R} = E\Big[\mathbf{x}(n)\mathbf{x}^T(n)\Big]$, la matrice d'autocorrélation du vecteur d'entrée $\mathbf{x}(n)$ qui s'écrit de manière développée :

$$\begin{bmatrix} r(0) & r(1) & \cdots & r(M-1) \\ r(1) & r(0) & \cdots & r(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(M-1) & r(M-2) & \cdots & r(0) \end{bmatrix} \tag{5.17}$$

Soit $\mathbf{p} = E\Big[\mathbf{x}(n)d(n)\Big]$, le vecteur d'intercorrélation entre le vecteur d'entrée $\mathbf{x}(n)$ et la réponse désirée $d(n)$. Sous forme développée, $\mathbf{p}$ vaut :

$$\mathbf{p} = \Big[p(0),\ p(-1),\ \ldots\ p(1-M)\Big]^T \tag{5.18}$$

Enfin, soit $\mathbf{h}_o = [h_{o0},\ h_{o1},\ \ldots\ h_{o,M-1}]^T$, le vecteur des coefficients optimaux (au sens de l'erreur quadratique moyenne) de la réponse impulsionnelle du filtre. Les équations 5.16 de Wiener-Hopf peuvent alors se réécrirent sous forme matricielle de la façon suivante :

$$\mathbf{R}\mathbf{h}_o = \mathbf{p} \tag{5.19}$$

Pour résoudre ce système d'équations, nous considérons que la matrice d'autocorrélation $\mathbf{R}$ est non-singulière et donc inversible. Nous pouvons alors multiplier chaque coté de l'équation 5.19 par $\mathbf{R}^{-1}$, l'inverse de la matrice d'autocorrélation, pour obtenir les solutions :

$$\mathbf{h}_o = \mathbf{R}^{-1}\mathbf{p} \tag{5.20}$$

Le calcul du vecteur optimal $\mathbf{h}_o$ nécessite donc la connaissance de deux quantités :

1. la matrice d'autocorrélation $\mathbf{R}$ du vecteur d'entrée $\mathbf{x}(n)$,
2. le vecteur d'intercorrélation $\mathbf{p}$ entre le vecteur d'entrée $\mathbf{x}(n)$ et la réponse désirée $d(n)$.

Le calcul du vecteur d'intercorrélation nécessite donc la connaissance d'un vecteur de référence $d(n)$.

5.1.2 Principe et théorie du filtrage adaptatif

Nous avons vu au paragraphe précédent que la mise en oeuvre d'un filtre optimal de Wiener est donc possible si et seulement si :

– il est possible d'avoir la connaissance à priori des caractéristiques statistiques des signaux filtrés, notamment l'autocorrélation des signaux,
– ces caractéristiques sont stationnaires, c'est à dire qu'elles n'évoluent pas avec le temps.

Dans la pratique, ces deux conditions ne sont pas forcément remplies. Lorsque les informations à priori sur les caractéristiques du signal d'entrée ne sont pas complètement connues, il n'est pas possible de réaliser le filtre de Wiener ou celui qui est réalisé n'est pas optimal. Une approche simple pouvant être utilisée est l'utilisation d'une méthode "estimate and plug". C'est un processus en deux parties. Premièrement, le filtre estime les paramètres statistiques du signal considéré puis insère le résultat ainsi obtenu par une formule non récursive pour calculer les paramètres du filtre. Pour un fonctionnement temps réel, cette méthode a le désavantage de nécessiter un circuit très complexe et coûteux. Pour contrer cette limitation, nous pouvons utiliser un filtre adaptatif. Un tel système "s'auto-conçoit", c'est à dire que le filtre adaptatif repose sur un algorithme récursif, rendant possible le fonctionnement du filtre dans un environnement où la connaissance complète des caractéristiques du signal considéré n'est pas disponible. Cet algorithme démarre à partir d'un ensemble prédéterminé de conditions initiales, représentant ce que l'on sait sur l'environnement de fonctionnement du filtre. Ainsi, dans un environnement stationnaire, après un certain nombre d'itérations, l'algorithme converge vers la solution optimale de Wiener, au sens statistique du terme.

Il existe un grand nombre d'algorithmes d'adaptation mais on peut distinguer deux grandes familles d'algorithmes adaptatifs [89] :

– les algorithmes du gradient conduisant aux algorithmes de classe LMS (Least Mean Square)
– les algorithmes des moindres carrés conduisant aux algorithmes de classe RLS (Recursive Least Squares).

5.1.2.1 Méthode du gradient

La fonction coût, dans le cadre des algorithmes du gradient, est définie par l'erreur quadratique moyenne de la différence entre le signal désirée et la sortie du filtre. Cette fonction coût est une fonction du second ordre par rapport aux coefficients de la réponse impulsionnelle du filtre dont le minimum correspond aux coefficients de la solution optimale de Wiener. Ainsi, les coefficients de la réponse impulsionnelle du filtre vont être adaptés en fonction de l'erreur par une boucle de retour comme le montre la figure 5.2.

Ces signaux sont considérés comme étant stationnaire sur une courte durée permettant ainsi

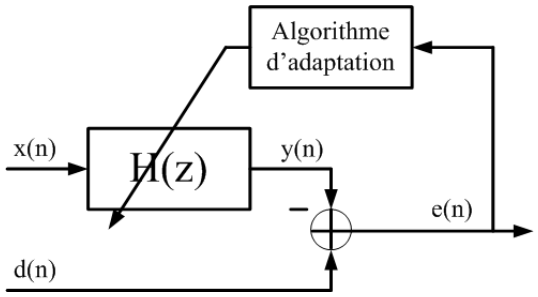


FIGURE 5.2 – Structure des filtres adaptatifs.

d'actualiser les coefficients du filtre. Nous considérons alors $\mathbf{x}(n) = [x(n), x(n-1), \ldots, x(n-M+1)]$ le vecteur des M derniers échantillons du signal d'entrée (M étant l'ordre du filtre), $y(n) = \mathbf{h}^T(n)\mathbf{x}(n)$ la réponse du filtre avec $\mathbf{h}(n)$, vecteur coefficients de la réponse impulsionnelle du filtre et $d(n)$ la réponse désirée. L'erreur d'estimation $e(n)$ est alors :

$$e(n) = d(n) - y(n) \tag{5.21}$$

$$= d(n) - \mathbf{h}^T(n)\mathbf{x}(n) \tag{5.22}$$

La détermination de $h(n)$ se fera alors à travers la minimisation de la fonction coût $J(\mathbf{h}(n))$, qui s'exprime par :

$$J(\mathbf{h}(n)) = E\left[e^2(n)\right] \tag{5.23}$$

$$= E\left[d^2(n)\right] - 2\mathbf{h}^T(n)\,E\left[\mathbf{x}(n)\,d(n)\right] + \mathbf{h}^T(n)\,E\left[\mathbf{x}(n)\,\mathbf{x}^T(n)\right]\mathbf{h}(n) \tag{5.24}$$

$$= \sigma_d - 2\mathbf{h}^T(n)\,\mathbf{p} + \mathbf{h}^T(n)\,\mathbf{R}\mathbf{h}(n) \tag{5.25}$$

où σ_d est la variance du signal désiré. Cette fonction J est une fonction continuement dérivable dont nous souhaitons trouver une solution optimale $\mathbf{h}_o$ qui satisfasse cette condition :

$$J(\mathbf{h}_o) \leq J(\mathbf{h}(n)), \quad \forall \mathbf{n} \tag{5.26}$$

L'équation 5.26 est l'expression mathématique d'une optimisation sans contrainte. Une classe d'algorithmes d'optimisation sans contrainte est particulièrement bien adaptée au filtrage adaptatif : la classe basée sur l'idée de "descente itérative locale" :

Définition 1 (Descente itérative locale) *A partir d'un estimation initiale* $\mathbf{h}(0)$*, générer une séquence de vecteurs* $\mathbf{h}(1)$, $\mathbf{h}(2)$, $\cdots$*, telle que la fonction coût* $J(\mathbf{h})$ *diminue à chaque itération de l'algorithme :*

$$J(\mathbf{h}(m+1)) < J(\mathbf{h}(m)) \tag{5.27}$$

où $\mathbf{h}(m)$ *est l'ancienne valeur des coefficients du filtre et* $\mathbf{h}(m+1)$ *est la nouvelle valeur des coefficients du filtre.*

En utilisant une forme simple de descente itérative, la méthode de la "plus grande pente", les ajustements successifs appliqués au vecteur des coefficients sont dans la direction de la plus grande pente, c'est à dire dans la direction opposée au vecteur gradient de la fonction coût $J(\mathbf{h})$, notée $\boldsymbol{\nabla} J(\mathbf{h})$:

$$\boldsymbol{\nabla} J(\mathbf{h}) = \frac{\partial J(\mathbf{h})}{\partial \mathbf{h}} \tag{5.28}$$

L'algorithme de la plus grande pente est décrit de manière formelle par :

$$\mathbf{h}(m+1) = \mathbf{h}(m) - \frac{1}{2}\mu \boldsymbol{\nabla} J(\mathbf{h}) \tag{5.29}$$

où m marque l'itération et μ est une constante positive appelée le coefficient de relaxation.
μ tient une place importante dans la convergence de l'algorithme de la plus grande pente. En effet, la condition nécessaire et suffisante pour que l'algorithme soit stable et qu'il converge est [89] :

$$|1 - \mu\lambda_k| < 1, \quad k \in [\![1, M]\!] \tag{5.30}$$

$$\Leftrightarrow 0 < \mu < \frac{2}{\lambda_k}, \quad k \in [\![1, M]\!] \tag{5.31}$$

$$\Leftrightarrow 0 < \mu < \frac{2}{\lambda_{max}} \tag{5.32}$$

où M est la dimension de la matrice $\mathbf{R}$ et λ_k ses valeurs propres. Plus μ sera proche de sa valeur limite, plus vite l'algorithme convergera et à l'inverse, plus la valeur de μ sera faible, plus l'algorithme sera lent à converger. En pratique, la connaissance des valeurs propres λ_k n'est pas accessible et donc le domaine de stabilité de l'algorithme est inconnu.

Pour appliquer l'algorithme de la plus grande pente (équation 5.29) au filtrage de Wiener (équation 5.25), il faut exprimer le gradient de la fonction coût $J(\mathbf{h}(n))$, plus simplement notée $\boldsymbol{\nabla} J(n)$:

$$\nabla J(n) = \frac{\partial J(n)}{\partial \mathbf{h}(n)} \tag{5.33}$$

$$= -2\Big[\mathbf{x}(n-k)e(n)\Big] \tag{5.34}$$

$$= -2\Big[\mathbf{x}(n-k)\Big(d(n) - \mathbf{h}(n)\mathbf{x}^T(n)\Big)\Big] \tag{5.35}$$

$$= -2\Big[\mathbf{x}(n-k)d(n)\Big] + 2\Big[\mathbf{x}(n-k)\mathbf{x}^T(n)\Big]\mathbf{h}(n) \tag{5.36}$$

$$= -2\mathbf{p} + 2\mathbf{R}\mathbf{h}(n) \tag{5.37}$$

Pour pouvoir appliquer l'algorithme de la plus grande pente, nous supposons que dans l'équation 5.37, la matrice d'autocorrélation $\mathbf{R}$ et le vecteur d'intercorrélation $\mathbf{p}$ sont connus, permettant ainsi le calcul du gradient $\nabla J(n)$ pour une valeur donnée du vecteur $\mathbf{h}(n)$. En substituant

l'équation 5.37 dans l'équation 5.29, la nouvelle valeur du vecteur $\mathbf{h}(n+1)$ en utilisant la simple relation de récursion :

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu\left(\mathbf{p} - \mathbf{R}\mathbf{h}(n)\right), \quad n \in \mathbb{N} \tag{5.38}$$

L'utilisation de l'algorithme de la plus grande pente nécessite donc la connaissance de la matrice d'autocorrélation $\mathbf{R}$ et du vecteur d'intercorrélation $\mathbf{p}$. Mais dans la pratique, il n'est pas concevable de calculer à chaque instant la matrice d'autocorrélation $\mathbf{R}$ et le vecteur d'intercorrélation $\mathbf{p}$ pour mesurer la valeur du gradient déterministe $\nabla J(n)$. Il faut donc estimer ce gradient en utilisant les données disponibles lorsque le filtre travail dans un environnement inconnu. Et c'est là que vient l'algorithme LMS pour estimer ce gradient.

Algorithme LMS Pour développer un estimateur du vecteur gradient $\nabla J(n)$, la stratégie la plus évidente est d'utiliser une estimation de la matrice d'autocorrélation $\mathbf{R}$ et du vecteur d'intercorrélation $\mathbf{p}$ dans l'équation 5.37. Le choix le plus simple pour un estimateur est d'utiliser des estimateurs instantanés basés sur les valeurs du vecteur d'entrée $\mathbf{x}(n)$ et de la réponse désirée $d(n)$. Ces estimateurs sont définis par :

$$\hat{\mathbf{R}}(n) = \mathbf{x}(n)\mathbf{x}^T(n) \tag{5.39}$$

$$\hat{\mathbf{P}}(n) = \mathbf{x}(n)d(n) \tag{5.40}$$

Ainsi, le vecteur gradient instantané $\hat{\nabla} J(n)$ est défini par :

$$\hat{\nabla} J(n) = -2\mathbf{x}(n)d(n) + 2\mathbf{x}(n)\mathbf{x}^T(n)\hat{\mathbf{h}}(n) \tag{5.41}$$

Généralement, cette estimateur est biaisé à cause du vecteur coefficient estimé $\hat{\mathbf{h}}(n)$. En effet, $\hat{\mathbf{h}}(n)$ est un vecteur aléatoire dépendant du vecteur d'entrée $\mathbf{x}(n)$. En utilisant l'estimation du gradient $\hat{\nabla} J(n)$ dans la formulation de l'algorithme de la plus grande pente (équation 5.38), il vient :

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu\mathbf{x}(n)\left(d(n) - \mathbf{x}^T(n)\hat{\mathbf{h}}(n)\right) \tag{5.42}$$

$$= \mathbf{h}(n) + \mu\mathbf{x}(n)e(n) \tag{5.43}$$

Cette dernière équation est l'expression de la forme réelle de l'algorithme adaptatif "Least Mean Square" (LMS). Il appartient à la famille des algorithmes du gradient stochastique. Puisque nous utilisons des estimateurs biaisés, le minimum de la fonction coût ne sera pas atteint et l'algorithme LMS va donner des solutions autour du point minimum avec une erreur quadratique moyenne supplémentaire. La figure 5.3 représente le graphe-flot de signal de l'algorithme

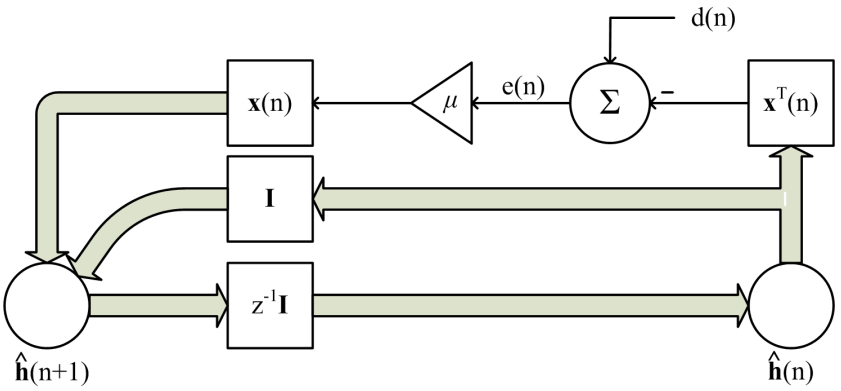


FIGURE 5.3 – Graphe-flot de signal de l'algorithme LMS.

LMS. Ce graphe-flot de signal illustre la simplicité de l'algorithme LMS. Ainsi à partir de la figure 5.3, on peut déduire que l'algorithme LMS nécessite seulement $2M + 1$ multiplications et $2M$ additions où M est le nombre de coefficients à adapter. En d'autres termes, la complexité d'exécution de l'algorithme LMS est en $O(M)$.

Il existe d'autres filtres adaptatifs LMS. On peut citer l'algorithme NLMS pour "Normalized LMS" [89]. Dans ce cas, l'équation 5.43 de mise à jour des coefficients du filtre devient :

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \frac{\mu}{\delta + \|\mathbf{x}(n)\|^2}\mathbf{x}(n)e(n) \tag{5.44}$$

où $\delta > 0$ est un paramètre de régulation. L'algorithme NLMS prend en compte la puissance instantanée du signal d'entrée pour l'adaptation des coefficients. Il illustre le principe de la perturbation minimale : à chaque itération, les coefficients sont modifiés de façon minimale.
On peut également citer l'algorithme LMS par blocs [89]. Nous ne détaillerons pas son fonctionnement ici car il nécessite la mémorisation d'un certain nombre d'échantillons en entrée et de nombreuse ressources pour le traitement par bloc, contrairement à notre contrainte d'utilisation minimale des ressources dans le cadre du BIST.

5.1.2.2 Méthode des moindres carrés

La deuxième approche dans le développement d'algorithmes adaptatifs est basée sur la méthode des moindres carrés. Pour rappel, on dispose des signaux suivant : $x(n)$ le signal d'entrée du filtre, $\mathbf{x}(n)$ le vecteur des M derniers échantillons du signal d'entrée, $y(n) = \mathbf{h}^T(n)\mathbf{x}(n)$ la réponse du filtre avec $\mathbf{h}(n)$ comme vecteur des M coefficients du filtre , $d(n)$ la réponse désirée et $e(n) = d(n) - \mathbf{h}^T(n)\mathbf{x}(n)$ l'erreur d'estimation. Ici la fonction coût est définie par la somme

du carrée des erreurs :

$$\mathcal{E} = \sum_{i=i_1}^{i_2} |e(i)|^2 \tag{5.45}$$

où i_1 et i_2 définissent les indices limites pour lesquels on minimise l'erreur. Cette somme peut être également vue comme une "énergie d'erreur". Il existe quatre méthodes de fenêtrage pour définir ces indices limites pour N échantillons d'entrée $[x(1), x(2), \ldots, x(N)]$:

1. la méthode de la covariance qui ne fait pas de supposition sur les données en dehors de l'intervalle $[\![1,\ \mathrm{N}]\!]$. Dans ce cas, $i_1 = M$ et $i_2 = N$.
2. la méthode de l'autocorrélation pour laquelle les données en dehors de l'intervalle $[\![1,\ \mathrm{N}]\!]$ sont nulles. Dans ce cas, $i_1 = 1$ et $i_2 = N + M - 1$.
3. la méthode de pré-fenêtrage pour laquelle les données précédant l'instant $i = 1$ sont nulles. Dans ce cas, $i_1 = 1$ et $i_2 = N$.
4. la méthode de post-fenêtrage pour laquelle les données suivant l'instant $i = N$ sont nulles. Dans ce cas, $i_1 = M$ et $i_2 = N + M - 1$.

Nous allons uniquement présenter la méthode de pré-fenêtrage, utilisant ainsi tous les échantillons d'entrée disponibles., puisque les autres choix de bornes ne change pas le principe mais seulement les calculs. Pour rappel, les différents signaux intervenants sont réels. Dans ce cas, l'équation 5.45 peut se réécrire :

$$\mathcal{E} = \sum_{i=1}^{N} |e(i)|^2 = \sum_{i=1}^{N} e^2(i) \tag{5.46}$$

Nous pouvons ainsi calculer le gradient de l'énergie d'erreur et en utilisant le même raisonnement que pour l'équation (5.6), la minimisation de la fonction coût $\mathcal{E}$ s'exprime par :

$$\nabla_k \mathcal{E} = -2 \sum_{i=1}^{N} x(i-k)e(i) = 0, \qquad k \in [\![0,\ M-1]\!] \tag{5.47}$$

Soit e_{min} la valeur de l'erreur d'estimation correspondant à la fonction coût minimale $\mathcal{E}_{min}$. Alors l'équation 5.47 est équivalente à :

$$\sum_{i=1}^{N} x(i-k)e(i) = 0, \qquad k \in [\![0,\ M-1]\!] \tag{5.48}$$

Soit $\hat{\mathbf{h}}$, la valeur particulière du vecteur $\mathbf{h}$ lorsque le filtre est optimisé au sens des moindres carrés. L'erreur d'estimation minimale est alors :

$$e_{min}(i) = d(i) - \sum_{j=0}^{M-1} \hat{h}_j x(i-j) \tag{5.49}$$

En substituant l'équation 5.49 dans l'équation 5.48, il vient :

$$\sum_{j=0}^{M-1} \hat{h}_j \sum_{i=1}^{N} x(i-k)x(i-j) = \sum_{i=1}^{N} x(i-k)d(i), \qquad k \in [\![0,\ M-1]\!] \tag{5.50}$$

Les deux sommes ayant pour indice i représentent à un facteur près des moyennes temporelles [1] :

- La moyenne temporelle $\sum_{i=1}^{N} x(i-k)x(i-j)$ est égale à la fonction d'autocorrélation moyennée dans le temps de l'entrée du filtre. On la note $\Phi(j,k)$ pour $(j,k) \in [\![0,\ M-1]\!]^2$.
- La moyenne temporelle $\sum_{i=1}^{N} u(i-k)d(i)$ est égale la fonction d'intercorrélation moyenné dans le temps de l'entrée du filtre et de la réponse désirée. On la note $z(-k)$.

Ainsi l'équation 5.50 peut être réécrite :

$$\sum_{j=0}^{M-1} \hat{h}_j \Phi(j,k) = z(-k), \qquad k \in [\![0,\ M-1]\!] \tag{5.51}$$

Soit $\mathbf{\Phi}$, la matrice d'autocorrélation moyennée dans le temps du vecteur d'entrée $\mathbf{x}(n)$, qui s'écrit de manière développée :

$$\begin{bmatrix} \Phi(0,0) & \Phi(1,0) & \cdots & \Phi(M-1,0) \\ \Phi(0,1) & \Phi(0,1) & \cdots & \Phi(M-1,1) \\ \vdots & \vdots & \ddots & \vdots \\ \Phi(0,M-1) & \Phi(1,M-1) & \cdots & \Phi(M-1,M-1) \end{bmatrix} \tag{5.52}$$

Soit $\mathbf{z}$, le vecteur d'intercorrélation moyenné dans le temps entre le vecteur d'entrée $\mathbf{x}(n)$ et la réponse désirée $d(n)$. Sous forme développée, $\mathbf{z}$ vaut :

$$\mathbf{z} = [z(0),\ z(-1),\ \ldots\ z(1-M)]^T \tag{5.53}$$

Enfin, soit $\hat{\mathbf{h}} = [\hat{h}_0,\ \hat{h}_1,\ \ldots\ \hat{h}_{M-1}]^T$, le vecteur des coefficients du filtre des moindres carrés. Les équations 5.50 peuvent alors se réécrirent de manière matricielle :

$$\mathbf{\Phi}\hat{\mathbf{h}} = \mathbf{z} \tag{5.54}$$

Pour résoudre ce système d'équations, nous considérons que la matrice $\mathbf{\Phi}$ est inversible [2] pour obtenir l'estimation au sens des moindres carrés des coefficients du filtre :

$$\hat{\mathbf{h}} = \mathbf{\Phi}^{-1}\mathbf{z} \tag{5.55}$$

1. Pour être totalement rigoureux, pour pouvoir utiliser le terme "moyenne temporelle", il faut diviser chaque somme par son nombre de termes N. Mais une telle opération n'a pas d'effet sur l'équation 5.50. Nous avons choisi d'ignorer ces facteurs pour plus de clarté.

2. La matrice d'autocorrélation d'un signal réel est symétrique et définie non-négative, et donc inversible si et seulement si son déterminant est non nul

Il s'agit alors de calculer l'inverse de la matrice $\mathbf{\Phi}$ et le vecteur $\mathbf{z}$ à chaque instant pour pouvoir mettre à jour les coefficients du filtres, ce qui n'est pas concevable dans la pratique. Il s'agit donc de trouver une solution permettant d'utiliser les informations préalablement calculées et ainsi obtenir une forme récursive du calcul de l'estimation au sens des moindres carrés des coefficients du filtre, ce que permet de faire l'algorithme RLS

Algorithme RLS Dans une implémentation récursive de la méthode des moindres carrés, nous commençons les calculs à partir d'un ensemble de conditions initiales puis l'information contenue dans les nouveaux échantillons est utilisée pour mettre à jour les anciennes estimations. Par conséquent, le nombre de données observables varient. Ainsi, la fonction coût à minimiser s'exprime comme $\mathcal{E}(n)$, où n est la longueur des données observables. De plus il est habituel d'introduire un facteur de pondération $\beta(n,i)$ dans la définition de $\mathcal{E}(n)$, qui s'écrit :

$$\mathcal{E}(n) = \sum_{i=1}^{n} \beta(n,i) \left|e(i)\right|^2 \tag{5.56}$$

Il est important de remarquer que le vecteur des coefficients de la réponse impulsionnelle du filtre $\mathbf{h}(n)$ ne varient pas pendant l'intervalle d'observation $1 \leq i \leq n$ pour lequel la fonction coût $\mathcal{E}(n)$ est définie. De plus, les facteurs de pondération sont strictement positifs et inférieur à $1 : 0 << \beta(n,i) \leq 1$. Ces facteurs de pondérations sont utilisés pour permettre "d'oublier" les données les plus anciennes et ainsi permettre au système de suivre les variations statistiques des données observables lorsque le filtre fonctionne dans un environnement non-stationnaire. Une forme particulière de pondération communément utilisée est le facteur de pondération exponentiel ou facteur d'oubli, définit par :

$$\beta(n,i) = \lambda^{n-i}, \qquad i \in [\![1,n]\!] \tag{5.57}$$

où λ est une constante positive proche mais inférieur à 1. Lorsque $\lambda = 1$, c'est la méthode standard des moindres carrées.
Ainsi, la fonction coût $\mathcal{E}(n)$ peut se réécrire :

$$\mathcal{E}(n) = \sum_{i=1}^{n} \lambda^{n-i} \left|e(i)\right|^2 \tag{5.58}$$

Puis, en développant l'équation 5.58 et en rassemblant les termes, les estimations au sens des moindres carrés de la matrice $M \times M$ d'autocorrélation moyennée dans le temps $\mathbf{\Phi}(n)$ et du vecteur d'intercorrélation moyenné dans le temps $\mathbf{z}(n)$ peuvent se réécrire :

$$\mathbf{\Phi}(n) = \sum_{i=1}^{n} \lambda^{n-i} \mathbf{x}(i)\mathbf{x}^T(i) \tag{5.59}$$

$$\mathbf{z}(n) = \sum_{i=1}^{n} \lambda^{n-i} \mathbf{x}(i)d(i) \tag{5.60}$$

D'après la méthode des moindres carrés présentée précédemment, la valeur optimale du vecteur des coefficients de la réponse impulsionnelle du filtre $\hat{\mathbf{h}}$, pour laquelle la fonction coût $\mathcal{E}(n)$ atteint son minimum, est définie par l'équation matricielle :

$$\mathbf{\Phi}(n)\hat{\mathbf{h}}(n) = \mathbf{z}(n) \tag{5.61}$$

où $\mathbf{\Phi}(n)$ et $\mathbf{z}(n)$ sont maintenant respectivement définis par les équations 5.59 et 5.60.
En isolant le terme correspondant à $i = n$ du reste de la somme de l'équation 5.59, nous pouvons écrire :

$$\mathbf{\Phi}\left(n\right) = \lambda \left[\sum_{i=1}^{n-1} \lambda^{n-1-i} \mathbf{x}\left(i\right) \mathbf{x}^T\left(i\right)\right] + \mathbf{x}\left(n\right) \mathbf{x}^T\left(n\right) \tag{5.62}$$

Par définition, l'expression entre crochets vaut $\mathbf{\Phi}\left(n-1\right)$. Ainsi, on obtient la formule de récurrence suivante :

$$\mathbf{\Phi}\left(n\right) = \lambda \mathbf{\Phi}\left(n-1\right) + \mathbf{x}\left(n\right) \mathbf{x}^T\left(n\right) \tag{5.63}$$

$\mathbf{\Phi}\left(n-1\right)$ est l'ancienne valeur de la matrice d'autocorrélation et le produit matriciel $\mathbf{x}\left(n\right) \mathbf{x}^T\left(n\right)$ est un terme de correction dans l'opération de mise à jour. De même, à partir de l'équation 5.60, on obtient une formule de récurrence sur le vecteur d'intercorrélation $\mathbf{z}(n)$:

$$\mathbf{z}(n) = \lambda \mathbf{z}\left(n-1\right) + \mathbf{x}(n)d(n) \tag{5.64}$$

Pour calculer l'estimateur des moindres carrés pour le vecteurs des coefficients $\mathbf{h}$ à partir de l'équation 5.61, il faut calculer l'inverse de la matrice $\mathbf{\Phi}\left(n\right)$. En pratique, le calcul de cette matrice inverse est très consommateur de temps et de ressources. C'est pourquoi le lemme d'inversion matricielle est utilisé.

Définition 2 (Lemme d'inversion matricielle (dans le cas réel)) *Soient* $\mathbf{A}$ *et* $\mathbf{B}$*, deux matrice réelle définies positives* $M \times M$*. Si il existe deux matrices réelles* $\mathbf{C}$ *et* $\mathbf{D}$*, avec* $\mathbf{C}$ *définie positive* $M \times N$ *et* $\mathbf{D}$ *définie positive* $N \times M$*, telles que :*

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}^{-1}\mathbf{C}^T. \tag{5.65}$$

alors l'inverse de la matrice $\mathbf{A}$ *vaut :*

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}\mathbf{C}\left(\mathbf{D} + \mathbf{C}^T\mathbf{B}\mathbf{C}\right)^{-1}\mathbf{C}^T\mathbf{B}. \tag{5.66}$$

En considérant la matrice $\mathbf{\Phi}\left(n\right)$ comme étant inversible [3], le lemme d'inversion matricielle peut

3. cf. note de pied de page 2

s'appliquer avec :

$$\mathbf{A} = \mathbf{\Phi}(n) \tag{5.67}$$
$$\mathbf{B}^{-1} = \lambda\mathbf{\Phi}(n-1) \tag{5.68}$$
$$\mathbf{C} = \mathbf{x}(n) \tag{5.69}$$
$$\mathbf{D} = 1 \tag{5.70}$$

Dans ce cas, on obtient la formule de récurrence suivante sur $\mathbf{\Phi}(n)$:

$$\mathbf{\Phi}^{-1}(n) = \lambda^{-1}\mathbf{\Phi}^{-1}(n-1) - \frac{\lambda^{-2}\mathbf{\Phi}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{\Phi}^{-1}(n-1)}{1+\lambda^{-1}\mathbf{x}^T(n)\mathbf{\Phi}^{-1}(n-1)\mathbf{x}(n)} \tag{5.71}$$

Soient $\mathbf{P}(n)$ et $\mathbf{k}(n)$ tels que :

$$\mathbf{P}(n) = \mathbf{\Phi}^{-1}(n) \tag{5.72}$$
$$\mathbf{k}(n) = \frac{\lambda^{-1}\mathbf{P}(n-1)\mathbf{x}(n)}{1+\lambda^{-1}\mathbf{x}^T(n)\mathbf{P}(n-1)\mathbf{x}(n)} \tag{5.73}$$

n'équation 5.66 peut alors se réécrire à partir de ces définitions :

$$\mathbf{P}(n) = \lambda^{-1}\mathbf{P}(n-1) - \lambda^{-1}\mathbf{k}(n)\mathbf{x}^T(n)\mathbf{P}(n-1) \tag{5.74}$$

La matrice $M \times M$ $\mathbf{P}(n)$ est dénommé matrice d'autocorrélation inverse. Le vecteur $\mathbf{k}(n)$ de longueur M est le vecteur gain. En réarrangeant l'équation 5.74, il vient :

$$\mathbf{k}(n) = \mathbf{P}(n)\mathbf{x}(n) \tag{5.75}$$

Or $\mathbf{P}(n) = \mathbf{\Phi}^{-1}(n)$, alors l'équation 5.75 devient :

$$\mathbf{k}(n) = \mathbf{\Phi}^{-1}(n)\mathbf{x}(n) \tag{5.76}$$

A partir de ces équations, il s'agit maintenant de développer une formule de récurrence pour mettre à jour le vecteur des coefficients de la réponse impulsionnelle du filtre $\mathbf{h}(n)$. Pour cela, nous utilisons les équations 5.60, 5.64 et 5.73 pour exprimer l"estimateur des moindres carrés du vecteur des coefficients à l'instant n :

$$\hat{\mathbf{h}}(n) = \mathbf{\Phi}^{-1}(n)\mathbf{z}(n) \tag{5.77}$$
$$= \mathbf{P}(n)\mathbf{z}(n) \tag{5.78}$$
$$= \lambda\mathbf{P}(n)\mathbf{z}(n-1) + \mathbf{P}(n)\mathbf{x}(n)d(n) \tag{5.79}$$

En substituant l'équation 5.72 pour $\mathbf{P}(n)$ dans le premier terme du membre de droite de l'équation 5.79, nous obtenons :

$$\hat{\mathbf{h}}(n) = \hat{\mathbf{h}}(n-1) - \mathbf{k}(n)\left[d(n) - \mathbf{x}^T(n)\hat{\mathbf{h}}(n-1)\right] \tag{5.80}$$

Enfin, puisque $\mathbf{k}(n) = \mathbf{P}(n)\mathbf{x}(n)$, nous obtenons la formule de récursion sur les coefficients pour l'algorithme des moindres carrés récursifs (RLS) :

$$\hat{\mathbf{h}}(n) = \hat{\mathbf{h}}(n-1) + \mathbf{k}(n)\xi(n) \tag{5.81}$$

où $\mathbf{k}(n) = \mathbf{\Phi}^{-1}(n)\mathbf{x}(n)$ est le vecteur de gain et $\xi(n) = d(n) - \hat{\mathbf{h}}^T(n-1)\mathbf{x}(n)$ est l'erreur d'estimation à priori. L'erreur d'estimation à priori $\xi(n)$ est, en général, différente de l'erreur d'estimation à posteriori $e(n) = d(n) - \hat{\mathbf{h}}^T(n)\mathbf{x}(n)$. Pour initialiser l'algorithme RLS, il faut choisir le vecteur des coefficients initial ainsi que la matrice d'autocorrélation initiale. La figure 5.4 représente le graphe-flot de signal de l'algorithme RLS. Ce graphe-flot de signal illustre la

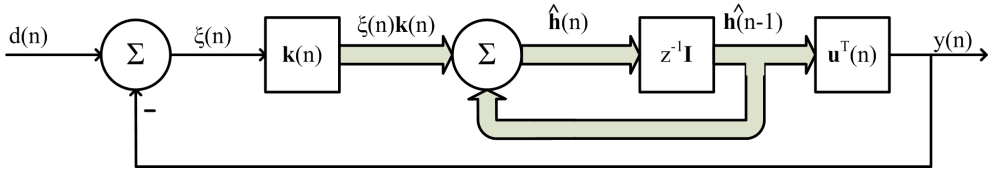


FIGURE 5.4 – Graphe-flot de signal de l'algorithme RLS.

relative complexité de l'algorithme RLS par rapport à celui de l'algorithme LMS.
La convergence du RLS est plus rapide que le LMS et l'erreur quadratique moyenne en excès tend vers 0 quand n tant vers ∞. En pratique, l'algorithme RLS nécessite plus de ressources que l'algorithme LMS puisqu'il faut calculer des opérations matricielles, donnant une complexité en $O(M^2)$ pour l'algorithme complet [9].
Il existe également d'autres algorithmes basés sur la méthode des moindres carrés comme le QR-RLS ou le QR-RLS inverse dont nous ne parlerons pas ici.

5.1.3 Algorithme adaptatif adapté à notre application

Comme nous l'avons vu au paragraphe précédent, la complexité de l'algorithme LMS est en $O(M)$ alors que celle de l'algorithme RLS est en $O(M^2)$, M étant le nombre de coefficients du filtre. Leur utilisation des ressources est directement proportionnel à leur complexité. L'implémentation de l'algorithme RLS nécessitera donc beaucoup plus de ressources que l'implémentation de l'algorithme LMS. Dans le cadre du BIST, où une utilisation de la surface minimale et un traitement temps réel sont visés, il est judicieux d'utilisé un algorithme LMS ou un algorithme de la famille des algorithmes du gradient pour l'adaptation des coefficients du filtres. Nous allons donc utiliser une adaptation de l'algorithme LMS sur notre cellule biquadratique définie au paragraphe 4.2.3 pour arriver à l'expression de l'équation de récurrence sur les coefficients du filtres.

5.1.3.1 Méthode de l'erreur en sortie

Soient $\mathbf{x}^T(n)$, le vecteur des échantillons et $\mathbf{h}^T(n)$, le vecteur des coefficients du filtre définis par [4] :

$$\mathbf{x}^T(n) = [y(n-1)\ y(n-2)\ x(n)\ x(n-1)\ x(n-2)] \tag{5.82}$$

$$\mathbf{h}^T(n) = [a_1(n)\ a_2(n)\ b_0(n)\ b_1(n)\ b_2(n)] \tag{5.83}$$

A chaque nouvelle sortie du filtre, les coefficients du filtres seront mis à jour. La réponse $y(n)$ du filtre s'écrit donc :

$$y(n) = \mathbf{h}^T(n)\mathbf{x}(n) \tag{5.84}$$

Le but de l'algorithme d'adaptation est de minimiser le fonction coût $J(n)$ définie par :

$$J(n) = \frac{1}{2}e^2(n) = \frac{1}{2}(d(n) - y(n)) \tag{5.85}$$

où $d(n)$ est le signal désiré, qui pour notre application correspond à la sinusoïde pure correspondant à l'harmonique filtré (cf. chapitre 3). Ainsi l'erreur $e(n)$ correspond au bruit résiduel. Nous pouvons alors réécrire l'équation 5.85 :

$$J(n) = \frac{1}{2}(x(n) - y(n)) \tag{5.86}$$

L'adaptation des coefficients se fait alors par :

$$\hat{\mathbf{h}}^T(n+1) = \hat{\mathbf{h}}^T(n) - \mu\frac{\partial J(n)}{\partial \hat{\mathbf{h}}(n)} \tag{5.87}$$

Les dérivées partielles $\frac{\partial J}{\partial \mathbf{h}}$ valent, dans une forme développée :

$$\frac{\partial J_k(n)}{\partial \mathbf{h}(n)} = e(n)\frac{\partial e(n)}{\partial \hat{h}_k(n)} \tag{5.88}$$

$x(n)$ étant indépendant de h, il vient :

$$\frac{\partial J_k(n)}{\partial \mathbf{h}(n)} = -e(n)\frac{\partial y(n)}{\partial \hat{h}_k(n)} \tag{5.89}$$

En remplaçant $y(n)$ par l'expression de la sortie du filtre, il vient :

$$\frac{\partial y(n)}{\partial a_i(n)} = y(n-i) - \sum_{k=1}^{2} a_k(n)\frac{\partial y(n-k)}{\partial a_i(n)} \tag{5.90}$$

$$\frac{\partial y(n)}{\partial b_i(n)} = x(n-i) + \sum_{k=1}^{2} a_k(n)\frac{\partial y(n-k)}{\partial b_i(n)} \tag{5.91}$$

4. les a_i de ce vecteur coefficient correspondent aux $-a_i$ de l'équation aux différences du filtre ($y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a_1y(n-1) - a_2y(n-2)$)

Soient α_i et β_i valant respectivement :

$$\alpha_i(n) = \frac{\partial y(n-k)}{\partial a_i(n)}, \quad i \in 1,\ 2 \tag{5.92}$$

$$\beta_i(n) = \frac{\partial y(n-k)}{\partial b_i(n)}, \quad i \in [\![0,\ 2]\!] \tag{5.93}$$

Chaque dérivée partielle est ensuite approximé par [90] :

$$\frac{\partial y(n-k)}{\partial a_i(n)} \approx \frac{\partial y(n-k)}{\partial a_i(n-k)} = \alpha_i(n-k) \tag{5.94}$$

$$\frac{\partial y(n-k)}{\partial b_i(n)} \approx \frac{\partial y(n-k)}{\partial b_i(n-k)} = \beta_i(n-k) \tag{5.95}$$

L'équation 5.91 peut être alors réécrite :

$$\alpha_i(n) \approx y(n-i) + \sum_{k=1}^{2} a_k(n)\alpha_i(n-k), \quad i \in 1,\ 2 \tag{5.96}$$

$$\beta_i(n) \approx x(n-i) + \sum_{k=1}^{2} a_k(n)\beta_i(n-k), \quad i \in [\![0,\ 2]\!] \tag{5.97}$$

A partir des équations 5.87, 5.92 et 5.93, il vient :

$$\hat{\mathbf{h}}^T(n+1) \approx \hat{\mathbf{h}}^T(n) + \mu e(n)\gamma(n) \tag{5.98}$$

où $\gamma(n) = \frac{\partial y(n)}{\partial \hat{\mathbf{h}}(n)} = [\alpha_1(n)\ \alpha_2(n)\ \beta_0(n)\ \beta_1(n)\ \beta_2(n)]^T$ est le vecteur régresseur filtré. Cette équation va donc permettre d'adapter les coefficients du filtre de manière récursive.

5.2 Implémentation du filtre adaptatif

La structures du filtre adaptatif en Forme Transposée est détaillée à partir de la structure préalablement présentée au chapitre 4.1. Notre structure réalise une fonction de transfert ayant ses zéros fixés à $z = -1$ et à $z = 1$, ainsi qu'une réponse en fréquence dont la largeur de bande est constante. Ainsi, seule la pulsation de résonance varie d'un filtre à l'autre.

5.2.1 Structure

Nous avons vu que les différents filtres du banc de filtres ont une largeur de bande fixée et identique pour maximiser l'énergie filtrée (cf. paragraphe 4.2.1). Or celle-ci ne dépendant que du module des pôles de la fonction de transfert, celui-ci est constant. De plus, les zéros sont fixés à $z = -1$ et $z = 1$. Les coefficients $b_0 = \frac{1-r^2}{1+r^2}$ sont donc constants et ne varierons pas avec l'adaptation. a_2 vaut toujours -1. Le seul coefficient dépendant de la pulsation de résonance est le coefficient a_1 qu'il faut adapté.

D'après l'algorithme présenté au paragraphe précédent, l'adaptation du coefficient a_1 est réalisée par :

$$a_1(n+1) = a_1(n) + \mu e(n)\alpha_1(n) \quad (5.99)$$

$$\alpha_1(n) = y(n-1) + a_1(n)\alpha_1(n-1) + a_2\alpha_1(n-2) \quad (5.100)$$

L'équation 5.100 peut être vue comme l'équation aux différences d'un filtre dont l'entrée est le signal $y(n-1)$, la sortie $\alpha_1(n)$ et $-a_1$ est le coefficients. Le calcul de $\alpha_1(n)$ se fera à travers un filtre dont la fonction de transfert est :

$$H_{\alpha_1} = \frac{z^{-1}}{1 - a_1 z^{-1} - a_2 z^{-2}} = \frac{z^{-1}}{1 - a_1 z^{-1} + z^{-2}} \quad (5.101)$$

La figure 5.5 présente la structure du filtre adaptatif.

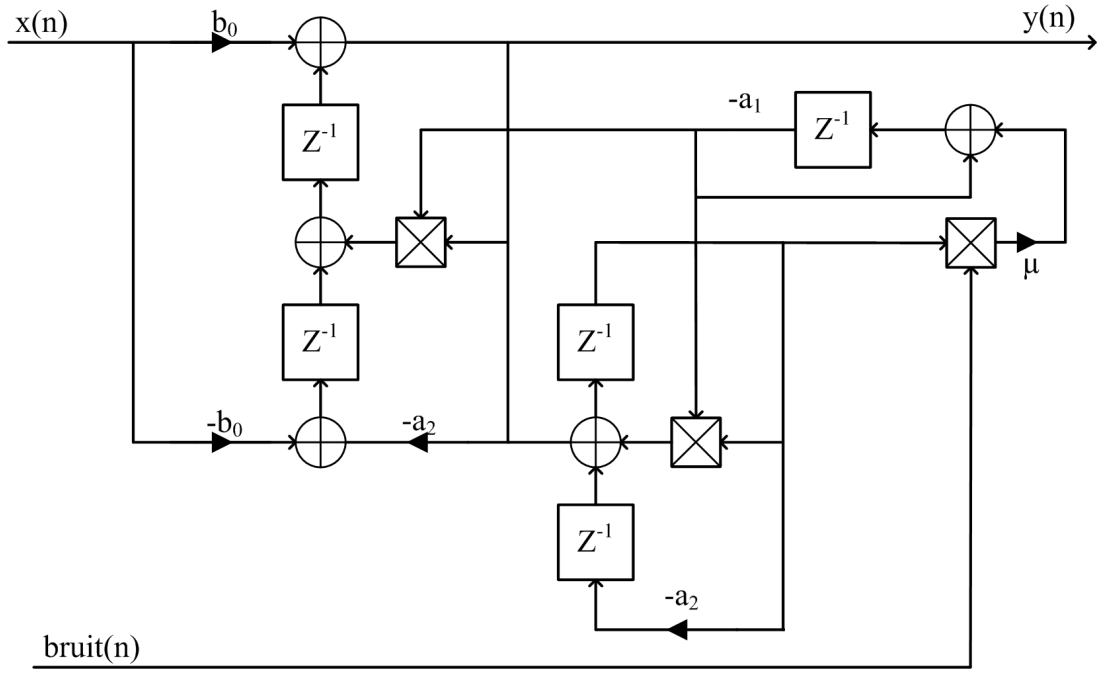


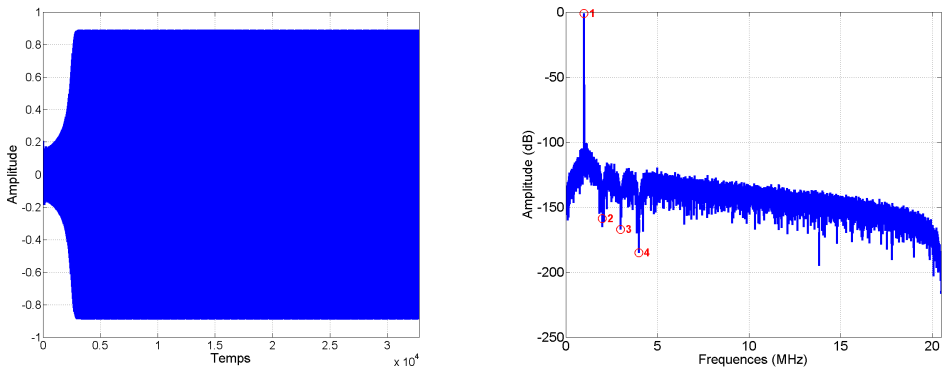
FIGURE 5.5 – Structure en Forme Transposée du filtre adaptatif.

Pour la réalisation des ces filtres adaptatifs, il reste à définir une valeur pour la constante d'adaptation μ. Celle-ci est choisie par simulation relativement faible pour satisfaire la condition de convergence (équation 5.32). De plus, dans l'objectif de réduire le coût d'implémentation de ces filtres, μ est choisie comme une puissance de 2 permettant la réalisation de sa multiplication associée par un simple décalage. Dans notre schéma de BIST, l'importance du temps de convergence du filtre vers le filtre optimal de Wiener est faible. En effet, pour que les puissances

estimées en sortie de filtres soient correctes, 4×2^{res} échantillons doivent être pris en compte soit 16384 points (cf. partie 1.3.3). Mais lors du démarrage de la procédure de test, il faut également attendre que le générateur de stimuli sorte du régime transitoire inhérent à tout oscillateur. Ainsi, 32768 points sont générés et les puissances des différents signaux sont calculées sur les 16384 derniers points.

5.2.2 Simulation

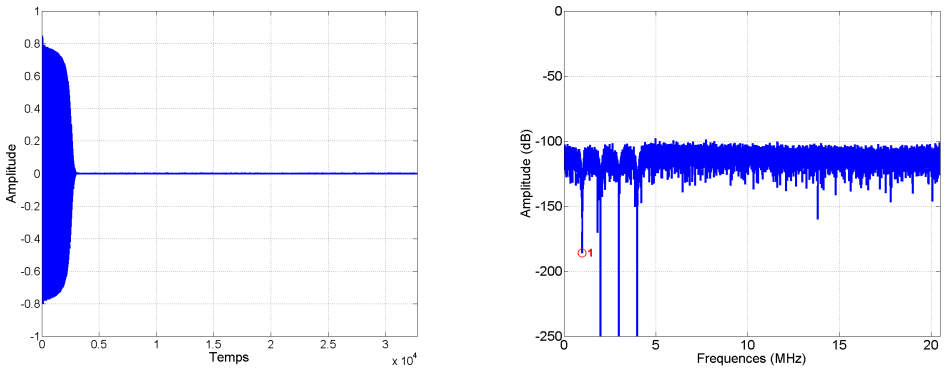
La structure du filtre adaptatif simulé est présentée à la figure 5.5. Le spectre du signal d'entrée des filtres adaptatifs comporte un fondamental d'amplitude d'environ 100 dB à 1 MHz par rapport au plancher de bruit. C'est le même signal converti par le CAN AD9042 que dans le chapitre 3. Les deuxième et troisième harmoniques respectivement d'amplitude d'environ 50 dB et 25 dB par rapport au plancher de bruit alors que les quatrième et cinquième harmoniques sont noyées dans le bruit. Le coefficient initial correspond à 95% de sa valeur calculée. Les résultats de la simulation sont présentées par les figures 5.6 et 5.7. La figure 5.6 montre le signal passe bande



(a) Signal (b) Spectre

FIGURE 5.6 – Signal passe bande et son spectre.

et son spectre. L'adaptation est brusque. Le fondamental présente une amplitude d'environ 100 dB par rapport au plancher de bruit dans la bande passante du filtre alors que le niveau du plancher de bruit est abaissé de 30 dB hors de la bande passante du filtre. Les raies harmoniques sont également atténuées faisant ainsi ressortir du plancher de bruit la quatrième harmonique. La sortie notch du filtre et son spectre sont représentés par la figure 5.7. Le fondamental y est fortement atténué, avec une perte d'environ 80 dB alors que les harmoniques d'ordre supérieur subissent une atténuation encore plus forte.



(a) Signal (b) Spectre

FIGURE 5.7 – Signal notch et son spectre.

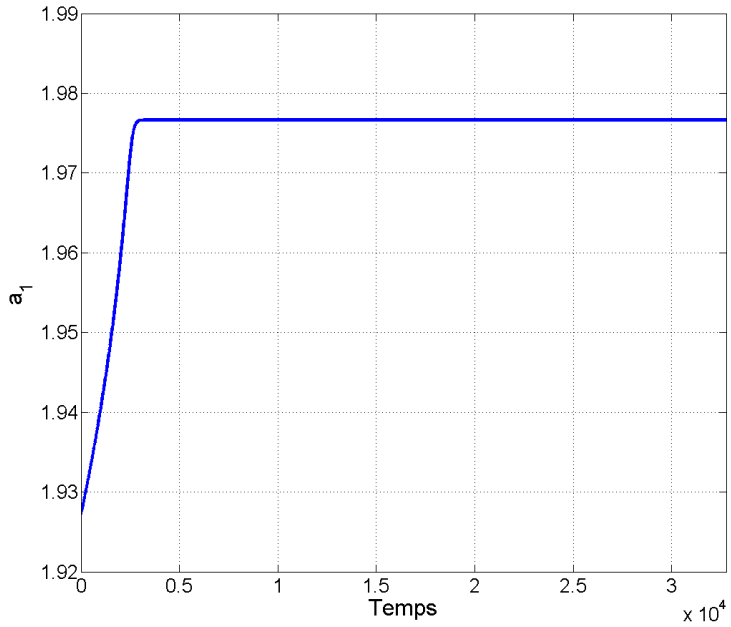


FIGURE 5.8 – Evolution du coefficient adapté du filtre en fonction du temps.

La figure 5.8 montre l'évolution du coefficient a_1 au cours du temps. On peut voir que le filtre converge en moins de 1500 itérations. Ce nombre est très inférieur au nombre d'échantillons non utilisés pour le calcul du *SNR* par le banc de filtre. Ainsi, l'adaptation du coefficient a_1 s'effectue correctement, avec un bonne atténuation en dehors de la bande passante.

5.2.3 Implémentation

L'implémentation des filtres adaptatifs s'effectuent à partir des cellules implémentées pour le bancs de filtres au chapitre 3. Maintenant que les a_1 sont recalculées au cours du temps, ils ne sont plus constant. Les multiplieurs *Shift-and-add* ne peuvent donc plus être utilisés. Ces multiplications seront donc réalisées par des multiplieurs à base de LUT permettant de multiplier deux signaux entre eux. La table 5.1 résume les résultats d'implémentation des cellules adaptatives et non-adaptatives utilisant les mêmes multiplieurs.

Structure	Nb de slices	Nb de slices FF	Nb de LUT4	Fréq. de fonctionnement (MHz)
Forme Transposée non-adaptative	209	62	398	105
Forme Transposée adaptative	1360	210	3432	47

TABLE 5.1 – Résultat des synthèses architecturales des différentes structures adaptatives ou non.

5.2.4 Comparaison des structures adaptatives et non adaptatives.

Comme attendu, les structures adaptatives utilisent beaucoup plus de ressources que les structures non-adaptatives : environ sept fois plus. Ce rapport de 7 s'explique par le fait du doublement du nombre de ressources pour la réalisation du filtre "sensitif" et l'utilisation d'un très grand multiplieur 31×31 pour le calcul du produit entre l'erreur et la sensibilité. De même, la fréquence maximale d'utilisation de filtre diminue mais reste toutefois supérieure à la fréquence d'échantillonnage du convertisseur.

5.3 Conclusion

Nous avons présenté dans ce chapitre les filtres adaptatifs permettant de minimiser la puissance du bruit dans l'extraction des composantes spectrales du signal issu du CAN. Dans un premier temps, plusieurs aspect théorique du filtrage optimal de Wiener ont été rappelés puis deux méthodes de filtrage adaptatif ont été étudiées : la méthode du gradient stochastique ou LMS (Least Mean Square) ainsi que la méthode des moindres carrés ou LS (Least-Square). Enfin, l'algorithme adaptatif adapté à notre application a été présenté.

Dans un second temps, les modifications apportées à la structure de filtre en forme transposée pour obtenir des filtres adaptatifs ont été détaillées.Puis la simulation et l'implémentation de cette structure de filtres sur FPGA a été présentés. Ses performances ont été comparées à celles de la structure non-adaptative. Ainsi, il a été montré que malgré la forte augmentation de lu besoin en surface pour une cellule, celle-ci reste assez performante au niveau de la fréquence de fonctionnement pour être utilisable pour tester le CAN ciblé.

Conclusion générale

Les travaux menés pendant cette thèse ont portés sur l'étude et la conception d'une structure BIST appliquée aux convertisseurs analogique numérique. Nous avons cherché à proposer une technique de BIST à bas coût appliquée aux CAN qui soit la plus générique possible dans le sens où elle ne se limite pas à une classe spécifique de convertisseur.

L'environnement de la conversion analogique numérique, l'étude des convertisseurs analogique numérique et le test des CAN ont été détaillés dans le premier chapitre. Les caractéristiques des CAN en terme d'erreurs statiques et dynamiques, telles que les erreurs d'offset, de gain, de non linéarité (paramètres de performance statiques) et les erreurs à l'ouverture, le taux de distorsion harmonique ainsi que le rapport signal à bruit (paramètres de performance dynamiques) ont été illustrées. La caractérisation dynamique d'un CAN a été illustrée par la description de quatre méthodes d'analyse (analyse par battement de fréquence, analyse temporelle, analyse statistique et analyse spectrale) permettant de déduire différents paramètres spectraux. Enfin, les principales techniques de test embarqués proposées dans la littérature ont été présentées.

A partir de l'étude de la conversion analogique numérique et de son test, une structure de BIST a été définie. Le premier point important, la génération in-situ d'un signal sinusoïdal analogique à partir d'un oscillateur numérique associé à un modulateur $\Sigma\Delta$ a ensuite été étudiée et réalisée. A partir d'un oscillateur numérique simple, ces performances ont été optimisées en utilisant un modulateur $\Sigma\Delta$ issu de la littérature après en avoir étudier plusieurs (générateur basé sur un oscillateur, modulateur passe bas du second ordre et modulateur MASH). Enfin, l'oscillateur $\Sigma\Delta$ passe bas a été implémenté sur FPGA. Celui présente un SNR de $72dB$ qui est supérieur au SNR supposé du CAN pour une occupation de 191 slices, ce qui représente environ le tiers de la surface totale occupé par le banc de filtre et le générateur de sinusoïdes.

L'unité d'extraction des paramètres spectraux a ensuite été définie au travers, dans un premier temps, d'une étude théorique des filtres résonateurs et notch. Dans un second temps, différentes topologies du banc de filtres (cascadés, parallèle-cascadés et parallèle) ont été présentées. Une

étude portant sur les fonctions de transfert des sorties du banc de filtres ont permis d'estimer leur capacité à mesurer avec un biais minimal le *SNR* du CAN testé. Nous avons, au cours de cette étude, présenté une méthodologie permettant de définir les fonctions de transfert des cellules filtrantes permettant la meilleure qualité de mesure, indépendamment du paramètre mesuré (*SNR* ou *THD*).

Dans un quatrième chapitre, l'implémentation du banc de filtres a été abordé. Différentes structures d'implémentation du filtre résonateur (structures en forme directe I et II, en forme transposée et structures *lossless discrete integrator*). Ensuite, l'implémentation d'un premier banc de filtres contraint au niveau de la fréquence du fondamental a été réalisé. L'objectif atteint était d'estimer le SNR du CAN avec une précision d'un demi LSB. Ce banc de filtres a été implémenté sur FPGA pour un fondamental à 1 MHz et occupe une surface de 395 slices. Nous avons également montré comment obtenir un banc de filtre optimal en terme de surface tout en conservant des performances adéquates au test en illustrant nos propos par le cas réel du CAN AD9042D d'Analog Devices. L'estimation du SNR par notre banc de filtre s'effectue avec une précision maîtrisée, précision qui peut être choisie pour être suffisante pour permette de savoir si le CAN fonctionne correctement.

Dans le dernier chapitre, nous sommes revenu à l'unité d'extraction des paramètres et plus précisément aux filtres pour en présenter une version adaptative. En effet, le calcul des coefficients des filtres se faisant par approximation, il existe un décalage entre la fréquence de résonance voulue pour un filtre et la fréquence de résonance obtenue lors de la réalisation matérielle de ce même filtre, décalage augmenté par le codage en virgule fixe des coefficients des filtres. De plus, des perturbation externes aux circuits du BIST peuvent introduire une distortion harmonique entre le signal généré et le signal transmit au CAN. L'utilisation d'une méthode adaptative a permit la correction de ces défauts. A partir d'une présentation théorique du filtrage optimal de Wiener puis d'algorithme de filtrage adaptatif dont la méthode du gradient stochastique (LMS) et des moindres carrés (LS), un algorithme adaptatif a été appliqué à notre application. Puis les modifications apportées à la structure définie précédemment ont été détailléesCette structure a été par la suite simulée et implémentée sur FPGA. Enfin, ses performances ont été comparées à celles de structure non-adaptative. Ainsi, il a été montré que les filtres adaptatifs utilisent sept fois plus de surface que les filtres non-adaptatifs mais gardaient une fréquence de fonctionnement suffisante pour son utilisation avec notre CAN cible. Nous avons ainsi montré que l'utilisation d'une méthode adaptative permet de suivre les décalages fréquentiels pouvant exister au niveau du signal issu du CAN.

La structure de BIST ainsi définie n'est pas directement utilisable. Premièrement, le bloc permettant le calcul et la comparaison des paramètres spectraux reste à réaliser. Même si celui-ci est relativement simple, il faut trouver une solution pour ne pas à avoir à calculer de quotient, comme le SNR par exemple. Une solution utilisable est de comparer la pseudo-puissance $P_0(s)$ du fondamental au produit du SNR minimum attendu et de la différence des pseudo-puissances $P_0(b)$ et $P_1(b)$ du bruit, $P_0(x)$ correspondant à la somme des carrés des échantillons $x(n)$ alors que $P_1(x)$ correspond au carré de la somme des échantillons normalisé par le nombre d'échantillons considérés. Dans un second temps, il faut réaliser un circuit intégré spécifique (ASIC) pour pouvoir atteindre une fréquence de suréchantillonnage suffisante pour le modulateur $\Sigma\Delta$ du générateur de sinusoïde.

Ce modulateur peut être également grandement amélioré. En effet en utilisant un modulateur $\Sigma\Delta$ passe haut, voir passe bande, il est possible de réduire la fréquence de suréchantillonnage. Il faut alors utiliser un filtre passe haut ou passe bande pour obtenir un signal analogique à partir du bitstream généré.

Le filtre adaptatif présenté est simple et non optimisé. Nous avons voulu montrer la faisabilité d'une telle méthode. Il existe d'autres algorithmes adaptatifs, LMS ou non, qui sont certainement moins coûteux.

Une autre perspective peut être envisagée : l'optimisation et l'automatisation de la génération du circuit de test. En effet, pour le banc de filtre, nous disposons actuellement d'un algorithme décisionnel permettant de choisir la structure des cellules du banc de filtres à partir de la fréquence d'échantillonnage et de la résolution du CAN tout en fournissant le codage binaire des coefficients. Cet algorithme peut être optimisé pour permettre, à partir des algorithmes de résolution du problème de *Multiplication par une constante unique*, de calculer automatiquement le coût associé à chaque multiplieurs et ainsi permettre le choix automatique de la meilleure structure. Enfin, il faut, à partir de ces structures, générer automatiquement la description VHDL du multiplieur *Shift-and-Add* correspondant puis, à l'aide d'un outil de synthèse haut-niveau par exemple, générer la description VHDL du banc de filtres complet.
De même que le banc de filtre, un logiciel peut être développé pour créer automatiquement le générateur de sinusoïde à partir de la qualité spectrale souhaitée pour le signal généré.
En combinant ces différentes perspectives, un outil global de génération automatique de structure BIST appliquée aux convertisseurs analogique numérique doit pouvoir être développé.

Bibliographie

[1] Spiral. Spiral Multiplier Block Generator.

[2] E. Allier. *Interface analogique numérique asynchrone : une nouvelle classe de convertisseur basés sur la quantification du temps.* PhD thesis, Ecole Nationale Supérieure d'Electronique et de Radio électricité de Grenoble TIMA, 2003.

[3] M. Bellanger. *Traitement numérique du signal : Théorie et pratique.* Dunod, 2002.

[4] M. Benkais. *Méthodologie de caractérisation des circuit de conversion de données : application aux convertisseurs analogique-numérique à facteur de bruit élevé. Mise en oeuvre dans le système CANTesT.* PhD thesis, Université Bordeaux 1, 1993.

[5] JESD99A.01. *Terms, Definitions, and Letter Symbols for Microelectronic Devices*, pages 2.36–2.58. JEDEC Solid State Technology Association, May 2003.

[6] IEEE Std 1241-2000. IEEE standard for terminology and test methods for analog-to-digital converters. *IEEE Std 1241-2000*, 2001.

[7] D. Dallet. *Contribution à la caractérisation des convertisseurs analogique-numérique : évaluation des méthodes et mise en oeuvre de nouveaux procédés.* PhD thesis, Université Bordeaux1, 1995.

[8] C. Recoquilloni. *Contribution au dessin et à l'integration d'un échantillonneur/démultipléxeur à très haut débit pour le projet ALMA.* PhD thesis, Université Bordeaux1, 2005.

[9] M. Jridi. *Etude, Modélisation et Amélioration des Performances des Convertisseurs Analogique Numérique Entrelacés dans le Temps.* PhD thesis, Université Bordeaux1, 2007.

[10] HP 5180A-2. Dynamic performance testing of A to D converters. *Hewlett Packard Product Note*, 1982.

[11] B. E. Peetz, A. S. Muto, and J. M. Neil. Measuring waveform recoder performance. *Hewlett Packard journal*, pages 21–29, November 1982.

[12] B. E. Peetz. Dynamic testing of waveform recoders. *IEEE Transactions on Instrumentation and Measurement*, 32(1) :12–17, March 1983.

[13] R. Potter. Least-squares estimation of sinusoidal parameters from measured data. Hewlett-Packard Internal Memo, June 1974.

[14] L. Ochs. Measurement and enhancement of waveform digitizer performance. presented at the IEEE Int. Convention, May 1976.

[15] H. U. Koller. New criterion for testing analog-to-digital converters for statistical evaluation. *IEEE Transactions on Instrumentation and Measurement*, 22 :214–217, September 1973.

[16] R. N. Bracewell. *The Fourier Transform and Its Application.* New York : McGraw-Hill, 1965.

[17] J. Kuffel, T.R. McComb, and R. Malewski. Comparative evaluation of computer methods for calculating the best-fit sinusoid to the digital record of high-purity sine wave. *IEEE Transactions on Instrumentation and Measurement*, 36(2) :418–422, June 1987.

[18] T.R. McComb, J. Kuffel, and B.C. Le Roux. A comparative evaluation of some practical algorithms used in the effectifs bits test of waveform recorders. *IEEE Transactions on Instrumentation and Measurement*, 38(1) :37–42, February 1989.

[19] J.Doernberg, H.S. Lee, and D.A. Hodges. Full-speed testing of a/d converters. *IEEE journal on Solid-State and Circuits*, 19(6) :820–827, December 1984.

[20] D.W. Doefler. High-resolution data acquisition system. *IEEE Transactions on Instrumentation and Measurement*, 35(4) :477–482, December 1986.

[21] S. Renaud. *Contribution à la caractérisation des circuits de conversion analogique-numérique. Conception et réalisation d'un système d'évaluation dynamique de ces dispositifs.* PhD thesis, Université de Bordeaux I, 1990.

[22] M. Vanden Bossche, J. Schoukens, and J.Renneboog. Dynamic testing and diagnostics of A/D converters. *IEEE Transactions on Circuits and Systems*, 33(8) :775–785, August 1986.

[23] J.H. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Procedings of the IEEE*, 66(1) :51–53, January 1978.

[24] Y.C. Jenq. Measuring harmonic distorsion and noise floor of an a/d converter using spectral averaging. *IEEE Transactions on Instrumentation and Measurement*, 37(4) :525–528, December 1988.

[25] O.M. Salomon. The effects of windowing and quantization error on the amplitude of frequency-domain functions. *IEEE Transactions on Instrumentation and Measurement*, 41(6) :932–937, December 1992.

[26] M.J. Ohletz. Hybrid Built-In Self-Test (HBIST) for Mixed Analog/Digital Integrated Circuits. In *Proc. European Test Conference*, pages 307–316, 1991.

[27] K. Damm and W. Anheier. HBIST of Nonliear Analog Building Blocks In Mixed-Signal Circuits. In *Proc. International Mixed-Signal Testing Workshop*, pages 257–262, 1995.

[28] G. Jervan. *Hybrid Built-In Self-Test and Test Generation Techniques for Digital Systems.* PhD thesis, Linköping University, 2005.

[29] K. Arabi and B. Kaminska. Oscillation-Test Strategy for Analog and Mixed-Signal Integrated Circuits. In *Proc. IEEE VLSI Test Symposium*, pages 476–482, 1996.

[30] K. Arabi and B. Kaminska. Testing Analog and Mixed-Signal Integrated Circuits Using Oscillation-Test Method. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, volume 16, pages 745–753, 1997.

[31] K. Arabi and B. Kaminska. Design for Testability of Embedded Integrated Operational Amplifiers. In *IEEE journal of Solid-State Circuits*, volume 33, pages 573–581, 1998.

[32] K. Arabi and B. Kaminska. Oscillation-Test Methodology for Low-Cost Testing of Active Analog Filters. In *IEEE Transactions on Instrumentation and Measurement*, volume 48, pages 798–806, 1999.

[33] K. Arabi and B. Kaminska. Efficient and accurate testing of analog-to-digital converters using oscillation-test method. In *European Design and Test Conference*, pages 348–352, 1997.

[34] G. Huertas and D. Vazquez and A. Rueda and J.L. Huertas. Oscillation-Based Test in Bandpass Oversampled AD Converters. In *Microelectronics journal*, number 34, pages 927–936, 2003.

[35] G. Huertas and D. Vazquez. Effective Oscillation-Based Test for application to a DTMF Filter Bank. In *Proc. IEEE International Test Conference*, pages 549–555, 1999.

[36] G. Huertas and D. Vazquez. Practical Oscillation-Based Test of Integrated Filters. In *IEEE Design and Test of Computers*, pages 64–72, 2002.

[37] S. Khaled and B. Kaminska and B. Courtois and M Lubaszewski. Frequency-based BIST for analog circuit testing. In *Proc. IEEE VLSI Test Symposium*, pages 54–59, 1995.

[38] D. Vazquez and G. Huertas and G. Leger and A. Ruena and J.L. Huertas. Practical Solutions for the Application of the Oscillation-Based-Test : Start-Up and On-Chip Evaluation. In *Proc. IEEE VLSI Test Symposium*, pages 433–438, 2002.

[39] D. Vazquez and G. Huertas and G. Leger and E. Peralias and A. Ruena and J.L. Huertas. On-Chip Evaluation of Oscillation-Based-Test Output Signals for Switched-Capacitor Circuits. In *Analog Integrated Circuits and Signal Processing*, pages 201–211, 2002.

[40] M.F. Toner and G.W. Roberts. Histogram-based Test for Distortion and Gain Tracking of a Mixed-Signal 8-bit PCM Chip. In *Proc. 6th Workshop on New Directions for Testing*, pages 97–112, 1992.

[41] A. Frish and T. Almy. HABIST : Histogram-based Analog Built-In-Self-Test. In *Proc. International Test Conference*, pages 760–767, 1997.

[42] M. Ehsanian and B. Kaminska and K. Arabi. A new on-chip digital BIST for analog-to-digital converters. *Microelectronics and reliability*, 38(3) :409–420, 1998.

[43] A. Frish and T. Almy. Histogram Based Testing of Analog Signals, 1998.

[44] F. Azais and S. Bernard and Y. Betrand and M. Renovell. Towards an ADC BIST scheme using the histogram test technique. In *Proceedings of the IEEE European Test Workshop*, pages 53–58, 2000.

[45] J.L. Huang and C.K. Ong and K.T. Cheng. A BIST scheme for on-chip ADC and DAC testing. In *Proceedings of Design, Automation and Test in Europe Conference and Exhibition.* , pages 216–220, 2000.

[46] R. de Vries and T. Zwemstra and E.M.J.G. Bruls and P.P.L. Regtien. Built-In Self-Test Methodology for A/D Converters. In *European Design and Test Conference*, pages 353–358, 1997.

[47] R. de Vries and B. Atzma. Method of Testing an Analog-to-Digital Converter, 1998.

[48] S. K. Sunter and N. Nagi. A simplified polynomial-fitting algorithm for DAC and ADC BIST. In *Proc. International Test Conference*, pages 389–395, 1997.

[49] S. Sunter and N. Nagi. Method and apparatus for testing DAC and ADC, 1997.

[50] M. F. Toner and G. W. Roberts. A BIST scheme for an SNR test of a sigma-delta ADC. In *International Test Conference*, pages 805–814, 1993.

[51] M. F. Toner and G. W. Roberts. A BIST scheme for an SNR, gain tracking, and frequency response test of a sigma-delta ADC. In *IEEE Transactions on Circuits and Systems - II*, volume 41, pages 1–15, January 1995.

[52] M.F. Toner and G.W. Roberts. A Frequency Response, Harmonic Distortion, and Intermodulation Distortion Test for BIST of a Sigma-Delta ADC. In *IEEE Trans. Circuits and Systems II*, volume 43, pages 608–613, 1996.

[53] C. Rebai. *Contribution à la Caractérisation des Convertisseurs Analogiques Numériques haute performances : Mise en oeuvre de nouveaux systèmes de traitement du signal pour le test in-situ.* PhD thesis, Université Bordeaux1, 2002.

[54] C.S. Turner. Recursive discrete-time sinusoidal oscillators. *IEEE Signal Processing Magazine*, 20(3) :103–111, May 2003.

[55] A. Lu and G. Roberts and D. Johns. A high-quality analog oscillator using oversampling D/A conversion techniques. In *IEEE Transactions on Circuits and Systems-II : Analog and Digital Signal Processing*, pages 437–444, Jul 1994.

[56] D. A. Johns and D. M. Lewis. IIR filtering on sigma-delta modulated signals. *Electronics Letters*, 27(4) :307–308, Feb. 1991.

[57] B. Leung. Theory of Sigma-Delta Analog to Digital Converter. *IEEE International Symposium on Circuits and Systems Tutorials*, pages 196–223, Dec. 1994.

[58] H. Inose and Y. Yasuda. A unity bit coding method by negative feedback. *Proceedings of the IEEE*, 51(11) :1524–1535, Nov. 1963.

[59] C.C. Cutler. *Transmission system employing quantization.* US Patent No. 2927962, 1960.

[60] G.R. Ritchie. *Higher order interpolation analog to digital converters.* PhD thesis, University of Pennsylvania, 1977.

[61] S. R. Norsworthy and R. Schreier and G. C. Temes. *Delta-Sigma Data Converters : Theory, Design, and Simulation.* IEEE Press, 1997.

[62] Y. Matsuya, K. Uchimura, A. Iwata, T. Kobayashi, M. Ishikawa T., and Yoshitome. A 16-bit oversampling a-to-d conversion technology using triple-integration noise shaping. *IEEE journal of Solid-State Circuits*, 22(6) :921–929, Dec 1987.

[63] X. Haurie and G. Roberts. Arbitrary-Precision Signal Generation for Bandlimited Mixed-Signal Testing. In *IEEE International Test Conference*, pages 78–86, 1995.

[64] X. Haurie and G. Roberts. Arbitrary-Precision Signal Generation for Mixed-Signal Built-In-Self Test. In *IEEE Trans. Circuits and Systems II : Analog and Digital Signal Processing*, volume 45, pages 1425–1432, 1998.

[65] P. Benabes. *Etude de nouvelles structures de convertisseurs sigma-delta passe-bande.* PhD thesis, Supélec, Université Paris XI, 1994.

[66] T. Hayashi, Y. Inabe, K. Uchimura, and T. Kimura. A multistage delta-sigma modulator without double integration loop. In *IEEE International Solid-State Circuits Conference. Digest of Technical Papers.*, volume XXIX, pages 182–183, Feb 1986.

[67] B. Bornoosh, A. Afzali-Kusha, R. Dehghani, M. Mehrara, S.M. Atarodi, and M. Nourani. Reduced complexity 1-bit high-order digital delta-sigma modulator for low-voltage fractional-n frequency synthesis applications. *Circuits, Devices and Systems, IEE Proceedings* -, 152(5) :471–477, Oct. 2005.

[68] Zhipeng Ye and M.P. Kennedy. Reduced complexity mash delta-sigma modulator. *Circuits and Systems II : Express Briefs, IEEE Transactions on*, 54(8) :725–729, Aug. 2007.

[69] Chia-Yu Yao and Chih-Chun Hsieh. Hardware simplification to the delta path in a mash 111 delta-sigma modulator. *Circuits and Systems II : Express Briefs, IEEE Transactions on*, 56(4) :270–274, April 2009.

[70] D.B. Ribner. A comparison of modulator networks for high-order oversampled Sigma-Delta analog-to-digital converters. *IEEE Transactions on Circuits and Systems*, 38(2) :145–159, 1991.

[71] C.M. Zierhofer. A multiplier-free digital sinusoid generator based on sigma-delta modulation. *Circuits and Systems II : Analog and Digital Signal Processing, IEEE Transactions on*, 43(5) :387–396, may 1996.

[72] D. Macii and P. Carbone and D. Petri. Stability Analysis of Oscillators based on a Delta-Sigma Topology, 2002.

[73] Ch. Rebai, D. Dallet, and Ph. Marchegay. LDI filter bank for ADC frequency domain analysis. In 9^{th} *IEEE International Conference on Electronics, Circuits and Systems*, pages 907–910, September 2002.

[74] Xilinx. Virtex-4 Family Overview DS112 (v3.0), 2007.

[75] M. Padmanabhan and W. Martin. Resonator-based filter-banks for frequency-domain applications. In *IEEE Transactions on Circuits and Systems*, volume 38, pages 1145–1159, October 1991.

[76] K. Steiglitz. A Note on Constant-Gain Digital Resonators. *Computer Music journal*, 18(4) :8–10, Winter 1994.

[77] J.G. Proakis and D.G. Manolakis. *Digital Signal Processing : Principles, Algorithms, and Applications*. Prentice-Hall, 1996.

[78] K. Martin and M. T. Sun. Adaptive filter suitable for real time spectral analysis. In *IEEE Transactions on Circuits and systems*, volume 33, pages 218–229, February 1986.

[79] W. Martin and M. Padmanabhan. Using IIR adaptative filter bank to analyse short data segment of noisy sinusoids. In *IEEE Transactions on Signal Processing*, volume 41, pages 2583–2590, August 1993.

[80] M. Padmanabhan and K. Martin. Filter Banks for Time-Recursive Implementations of Transforms. In *IEEE Transactions on Circuits and Systems*, volume 40, pages 41–50, January 1993.

[81] R. Beck and A. G. Dempster and I. Kale. Characterisation of finite-precision resonators used in recursive filter DFT implementations. In *journal of Signal Processing*, volume 80, pages 161–183, 2000.

[82] T. Kwan and K. Martin. Adaptive detection and enhancement of multiple sinusoids using a cascade IIR filter. In *IEEE Transactions on Circuits and Systems*, volume 36, pages 937–947, July 1989.

[83] Analog Devices. 12-Bit, 41 MSPS Monolithic A/D Converter, 1996.

[84] Yevgen Voronenko and Markus Püschel. Multiplierless multiple constant multiplication. *ACM Transactions on Algorithms*, 3(2), 2007.

[85] P.R. Cappello and K.Steiglitz. Some complexity issues in digital signal processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(5) :1037Ű1041, 1984.

[86] D.R. Bull and D.H. Horrocks. Primitive operator digital filters. *IEE Proceedings G*, 138(3) :401–412, 1991.

[87] A.G. Dempster and M.D. Macleod. Use of minimum-adder multiplier blocks in FIR digital filters. *IEEE Transactions in Circuits and Systems-II : Analog and Digital Signal Processing*, 42(9) :569–577, 1995.

[88] F. Michaut. *Méthodes adaptatives pour le signal : outils mathématiques et mise en oeuvre des algorithmes*. Hermes, 1992.

[89] S. Haykin. *Adaptive Filter Theory. Fourth Edition*. Prentice Hall, 2002.

[90] T.G. Xydis. A new iir adaptive filter design using tellegen's theorem. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 0 :2161–2164, 1991.

Publications

Conférence Internationale

[C1] N. Mechouk, D. Dallet, L. Bossuet and B. Le Gal, "A Low-Area Filter Bank Design Methodology for On-Chip ADC Testing," in *Proceeding of the 17th IEEE International Conference on Electronics, Circuits, and Systems (ICECS'10)*, Athens, Greece, December 2010.

Conférence Nationale

[N1] N. Mechouk, D. Dallet, L. Bossuet and B. Le Gal, "Unité de Gestion des Modes d'un IP Multimode," in *Journées Nationales du Réseau des Doctorants en Microélectronique (JNRDM'08)*, Bordeaux, France, May 14-16, 2008.

Annexe

A

Codes VHDL

A.1 Cellule Notch en Forme Transposée

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY Cellule_F is
generic (sizeRadix : integer);
PORT (clkS :  in Std_Logic;
      rst  : in Std_Logic;
      XnS  : in signed(sizeRadix+12 downto 0);
      Yn   : out signed(sizeRadix+12 downto 0));
END Cellule_F;


ARCHITECTURE Behavioral of Cellule_F is

signal bXn    : signed(sizeRadix+12 downto 0) := (others => '0');
signal D1     : signed(sizeRadix+12 downto 0) := (others => '0');
signal D2     : signed(sizeRadix+12 downto 0) := (others => '0');
```

```
signal YnTemp : signed(sizeRadix+12 downto 0) := (others => '0');
signal a1Yn   : signed(sizeRadix+12+sizeRadix+1 downto 0) := (others => '0');

COMPONENT multF_BHM is
PORT (X : in signed(sizeRadix+12 downto 0);
      Y : out signed(2*sizeRadix+13 downto 0));
END COMPONENT;

begin
      multA1Yn : multF_BHM
      port map(YnTemp,a1Yn);

process(rst, D2, XnS, YnTemp, bXn)--process(rst, D2, XnS, bXn)
      begin
      if rst = '0' then
            bXn    <= (others=>'0');
            YnTemp <= (others=>'0');
            Yn     <= (others=>'0');
      else
            bXn    <= XnS(XnS'left)&XnS(XnS'left)&XnS(XnS'left downto 2);
            YnTemp <= bXn - D2;
            Yn     <= YnTemp;
      end if;
end process;

process(clkS,rst)
      begin
      if rst = '0' then
            D1  <= (others=>'0');
            D2  <= (others=>'0');
      else
            if rising_edge(clkS) then
                  D1  <= bXn + YnTemp;
                  D2  <= D1 - a1Yn(2*sizeRadix+12 downto sizeRadix);
            end if;
```

```
      end if;
end process;

end Behavioral;
```

A.2 Banc de filtres

```
library IEEE;
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY Banc4P is
PORT (clkS  : in Std_Logic;
      rst   : in Std_Logic;
      Xin   : in Std_Logic_vector(11 downto 0);
      Bruit : out Std_Logic_vector(30 downto 0);
      Fond  : out Std_Logic_vector(30 downto 0);
      Harm2 : out Std_Logic_vector(30 downto 0);
      Harm3 : out Std_Logic_vector(30 downto 0);
      Harm4 : out Std_Logic_vector(30 downto 0));
END Banc4P ;

ARCHITECTURE Mixte of Banc4P is

COMPONENT Cellule_F is
generic (sizeRadix : integer);
PORT (clkS :  in Std_Logic;
      rst  : in Std_Logic;
      XnS  : in signed(30 downto 0);
      Yn   : out signed(30 downto 0));
end COMPONENT;

COMPONENT Cellule_H2 is
generic (sizeRadix : integer);
```

```
PORT (clkS :  in Std_Logic;
      rst  : in Std_Logic;
      XnS  : in signed(30 downto 0);
      Yn   : out signed(30 downto 0));
end COMPONENT;

COMPONENT Cellule_H3 is
generic (sizeRadix : integer);
PORT (clkS :  in Std_Logic;
      rst  : in Std_Logic;
      XnS  : in signed(30 downto 0);
      Yn   : out signed(30 downto 0));
end COMPONENT;

COMPONENT Cellule_H4 is
generic (sizeRadix : integer);
PORT (clkS :  in Std_Logic;
      rst  : in Std_Logic;
      XnS  : in signed(30 downto 0);
      Yn   : out signed(30 downto 0));
end COMPONENT;

signal XnS       : signed(11 downto 0) := (others => '0');
signal X30       : signed(30 downto 0) := (others => '0');
signal BruitTemp : signed(30 downto 0) := (others => '0');
signal entreeC   : signed(30 downto 0) := (others => '0');
signal sTemp1    : signed(30 downto 0) := (others => '0');
signal stemp2    : signed(30 downto 0) := (others => '0');
signal retour    : signed(30 downto 0) := (others => '0');
signal FondTemp  : signed(30 downto 0) := (others => '0');
signal Harm2Temp : signed(30 downto 0) := (others => '0');
signal Harm3Temp : signed(30 downto 0) := (others => '0');
signal Harm4Temp : signed(30 downto 0) := (others => '0');

begin
```

```
      Fond_Cell : Cellule_F
      generic map(18)
      port map(clkS,rst,entreeC ,FondTemp);

      Har2_Cell : Cellule_H2
      generic map(18)
      port map(clkS,rst,entreeC ,Harm2Temp);

      Har3_Cell : Cellule_H3
      generic map(18)
      port map(clkS,rst,entreeC ,Harm3Temp);

      Har4_Cell : Cellule_H4
      generic map(18)
      port map(clkS,rst,entreeC ,Harm4Temp);

process(rst, Xin, FondTemp, X30, Harm2Temp, Harm3Temp, Harm4Temp,...
 sTemp1, sTemp2, retour, BruitTemp)
begin
      if rst = '0' then
            XnS         <= (others=>'0');
            X30         <= (others=>'0');
            sTemp1      <= (others=>'0');
            sTemp2      <= (others=>'0');
            retour      <= (others=>'0');
            BruitTemp  <= (others=>'0');
      else
            XnS      <= signed(Xin);
            X30        <= XnS(XnS'left)&XnS&"0000000000000000";
            sTemp1      <= FondTemp  + Harm2Temp;
            sTemp2      <= Harm3Temp + Harm4Temp;
            retour      <= sTemp1    + sTemp2;
            BruitTemp  <= X30 - retour;
      end if;
end process;
```

```
process(clkS,rst)
begin
      if rst = '0' then
            Fond    <= (others=>'0');
            Harm2   <= (others=>'0');
            Harm3   <= (others=>'0');
            Harm3   <= (others=>'0');
            Harm4   <= (others=>'0');
            Bruit   <= (others=>'0');
            entreeC <= (others=>'0');
      else
            if rising_edge(clkS) then
                  Fond     <= std_logic_vector(FondTemp);
                  Harm2    <= std_logic_vector(Harm2Temp);
                  Harm3    <= std_logic_vector(Harm3Temp);
                  Harm4    <= std_logic_vector(Harm4Temp);
                  Bruit    <= std_logic_vector(BruitTemp);
                  entreeC  <= BruitTemp(BruitTemp'left)&BruitTemp(BruitTemp'left)...
                  &BruitTemp(BruitTemp'left)&BruitTemp(BruitTemp'left downto 3);
            end if;
      end if;
end process;

end Mixte;
```

A.3 Oscillateur $\Sigma\Delta$

A.3.1 Quantificateur

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY Quantificateur IS
generic(size : integer; sizeLSB : integer);
```

```
PORT (rst    : IN Std_Logic;
      entree : IN Std_Logic;
      bruit  : OUT Signed(size-1 DOWNTO 0));
END Quantificateur ;

ARCHITECTURE comportementale OF Quantificateur IS

constant onesMSB  : Signed(size-sizeLSB-1 downto 0) := (others=>'1');
constant zerosMSB : Signed(size-sizeLSB-2 downto 0) := (others=>'0');
constant zerosLSB : Signed(sizeLSB-1 downto 0) := (others=>'0');

begin

process(rst,entree)
begin
      if rst = '0' then
            bruit <= (others=>'0');
      else
            if (entree = '1') then
                  bruit <= onesMSB&zerosLSB;      -- '-1'
            else
                  bruit <= zerosMSB&"1"&zerosLSB;  -- '1'
            end if;
      end if;
end process;
end comportementale;
```

A.3.2 Modulateur $\Sigma\Delta$

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY SigmaDelta IS
generic(size : integer; sizeLSB : integer);
```

```
PORT (clk    : IN Std_Logic;
      rst    : IN Std_Logic;
      entree : IN Signed(size-1 downto 0);
      sortie : OUT Std_Logic);
END SigmaDelta;

ARCHITECTURE comportementale OF SigmaDelta IS

signal D1n      : signed(size-1 downto 0) := (others => '0');
signal D2n      : signed(size-1 downto 0) := (others => '0');
signal D3n      : signed(size-1 downto 0) := (others => '0');
signal D3nTemp1 : signed(size-1 downto 0) := (others => '0');
signal D3nTemp2 : signed(size-1 downto 0) := (others => '0');
signal D4n      : signed(size-1 downto 0) := (others => '0');
signal D1n1     : signed(size-1 downto 0) := (others => '0');
signal D2n1     : signed(size-1 downto 0) := (others => '0');
signal D3n1     : signed(size-1 downto 0) := (others => '0');
signal D4n1     : signed(size-1 downto 0) := (others => '0');
signal sub1     : signed(size-1 downto 0) := (others => '0');
signal add1     : signed(size-1 downto 0) := (others => '0');
signal add2     : signed(size-1 downto 0) := (others => '0');
signal add3     : signed(size-1 downto 0) := (others => '0');
signal add4     : signed(size-1 downto 0) := (others => '0');
signal add5     : signed(size-1 downto 0) := (others => '0');
signal add6     : signed(size-1 downto 0) := (others => '0');
signal add7     : signed(size-1 downto 0) := (others => '0');
signal bruit    : signed(size-1 downto 0) := (others => '0');
signal A1D1n    : signed(size-1 downto 0) := (others => '0');
signal A3D3n    : signed(size-1 downto 0) := (others => '0');
signal A2D2n1   : signed(size-1 downto 0) := (others => '0');
signal A4D4n1   : signed(size-1 downto 0) := (others => '0');
signal A1B1D1n  : signed(size-1 downto 0) := (others => '0');
signal A3B3D3n  : signed(size-1 downto 0) := (others => '0');
signal A2B2D2n1 : signed(size-1 downto 0) := (others => '0');
signal A4B4D4n1 : signed(size-1 downto 0) := (others => '0');
```

```
COMPONENT Quantificateur IS
generic(size : integer; sizeLSB : integer);
PORT (rst    : IN Std_Logic;
      entree : IN Std_Logic;
      bruit  : OUT Signed(size-1 DOWNTO 0));
END COMPONENT ;

begin

      Quant : Quantificateur
      generic map (size, sizeLSB)
      port map( rst, add7(add7'left), bruit);

      process(rst,entree,bruit,D1n,D1n1,D2n,D2n1,D3n,D3nTemp1,A1B1D1n,A1D1n,...
A3D3n,A3B3D3n,A2D2n1,A2B2D2n1,A4D4n1,A4B4D4n1,D3nTemp2,D3n1,D4n,D4n1,...
sub1,add1,add2,add3,add4,add5,add6,add7)

begin
      if rst = '0' then
            D1n1        <= (others=>'0');
            A1D1n       <= (others=>'0');
            A1B1D1n     <= (others=>'0');
            D2n1        <= (others=>'0');
            D3n1        <= (others=>'0');
            A3D3n       <= (others=>'0');
            A3B3D3n     <= (others=>'0');
            D3nTemp1    <= (others=>'0');
            D3nTemp2    <= (others=>'0');
            D4n1        <= (others=>'0');
            A2D2n1      <= (others=>'0');
            A2B2D2n1    <= (others=>'0');
            A4D4n1      <= (others=>'0');
            A4B4D4n1    <= (others=>'0');
            sub1        <= (others=>'0');
```

```
            add1        <= (others=>'0');
            add2        <= (others=>'0');
            add3        <= (others=>'0');
            add4        <= (others=>'0');
            add5        <= (others=>'0');
            add6        <= (others=>'0');
            add7        <= (others=>'0');
            sortie     <= '0';
      else
            sub1  <= entree - bruit;

            A1D1n  <= D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left)&...
D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left downto 8);
            A1B1D1n  <= D1n(D1n'left)&D1n(D1n'left)&D1n(D1n'left downto 2);

            A3D3n  <= D3n(D3n'left)&D3n(D3n'left)&D3n(D3n'left)&D3n(D3n'left)&D3n(D3n'left do
            A3B3D3n  <= D3n(D3n'left-6 downto 0)&"000000";

            add2  <= A1D1n - A3D3n;
            add4 <= A1B1D1n - A3B3D3n;

            D2n1  <= D2n + add2;
            D4n1  <= D4n - A3D3n;

            A2D2n1 <= D2n1(D2n1'left)&D2n1(D2n1'left)&D2n1(D2n1'left)&...
D2n1(D2n1'left)&D2n1(D2n1'left downto 4);
            A2B2D2n1 <= D2n1(D2n1'left-5 downto 0)&"00000";

            A4D4n1 <= D4n1(D4n1'left)&D4n1(D4n1'left)&D4n1(D4n1'left)&...
D4n1(D4n1'left)&D4n1(D4n1'left)&D4n1(D4n1'left)&D4n1(D4n1'left downto 6);
            A4B4D4n1 <= D4n1(D4n1'left-8 downto 0)&"00000000";

            add1  <= sub1 - A2D2n1;
            add3  <= A4D4n1 - A2D2n1;
            add5 <= A2B2D2n1 + A4B4D4n1;
```

```
            D1n1 <= add1 + D1n;
            D3n1 <= add3 + D3n;

            add6 <= add4 + add5;
            add7 <= add6 + entree;

            sortie  <= add7(add7'left);
      end if;
end process;

process(clk,rst)
begin
      if rst = '0' then
            D1n       <= (others=>'0');
            D2n       <= (others=>'0');
            D3n       <= (others=>'0');
            D4n       <= (others=>'0');
      else
            if rising_edge(clk) then
                  D1n <= D1n1;
                  D2n <= D2n1;
                  D3n <= D3n1;
                  D4n <= D4n1;
            end if;
      end if;
end process;
end comportementale;
```

A.3.3 Multiplexeur

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY Multiplexeur IS
```

```
generic(size : integer);
PORT (rst    : IN Std_Logic;
      cmd    : IN Std_Logic;
      sortie : OUT Signed(size-1 DOWNTO 0)) ;
END Multiplexeur ;

ARCHITECTURE comportementale OF Multiplexeur IS

      constant k0p : signed(size-1 downto 0) := "1111110101011000";
      constant k0m : signed(size-1 downto 0) := "0000010101010100";

begin
process(rst,cmd)
begin
      if rst = '0' then
            sortie     <= (others=>'0');
      else
            if (cmd = '1') then
                  sortie <= k0m;
            else
                  sortie <= k0p;
            end if;
      end if;
end process;
end comportementale;
```

A.3.4 SignalGenerator

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

ENTITY SignalGenerator is
PORT (clk : in Std_Logic;
      rst : in Std_Logic;
```

```
      Yn  : out Std_Logic);
END SignalGenerator ;

ARCHITECTURE Behavioral of SignalGenerator is

constant size    : integer := 18;
constant sizeLSB : integer := 17;

signal D1n    : signed(size-1 downto 0) := (others => '0');
signal D2n    : signed(size-1 downto 0) := (others => '0');
signal D1n1   : signed(size-1 downto 0) := (others => '0');
signal D2n1   : signed(size-1 downto 0) := (others => '0');
signal Coef   : signed(size-1 downto 0) := (others => '0');
signal sig1   : signed(size-1 downto 0) := (others => '0');
signal sig2   : signed(size+1 downto 0) := (others => '0');
signal cmd    : std_logic               := '0';


constant D1i  : signed(size-1 downto 0) := "001110100111110110"; -- Asin(2.pi.f0/Fs)
constant D2i  : signed(size-1 downto 0) := (others => '0');

COMPONENT Multiplexeur IS
generic(size : integer);
PORT (rst    : IN Std_Logic;
      cmd   : IN Std_Logic;
      sortie : OUT Signed(size-1 DOWNTO 0));
END COMPONENT ;

COMPONENT SigmaDelta IS
generic(size : integer; sizeLSB : integer);
PORT (clk    : IN Std_Logic;
      rst    : IN Std_Logic;
      entree : IN Signed(size-1 DOWNTO 0);
      sortie : OUT Std_Logic);
END COMPONENT ;
```

```
begin

      Mux : Multiplexeur
      generic map (size)
      port map(rst, cmd, Coef);

      SigDel : SigmaDelta
      generic map (size+2, sizeLSB)
      port map(clk, rst, sig2, cmd);

      sig1 <= D1n1(D1n1'left)&D1n1(D1n1'left)&D1n1(D1n1'left)...
      &D1n1(D1n1'left)&D1n1(D1n1'left)&D1n1(D1n1'left downto 5);
      sig2 <= D2n(D2n'left)&D2n(D2n'left)&D2n;

process(rst,D1n,Coef,D2n,sig1)
begin
      if rst = '0' then
            D1n1 <= (others => '0');
            D2n1 <= (others => '0');
      else
            D1n1  <= D1n + Coef;
            D2n1  <= sig1 + D2n;
      end if;
end process;

process(clk,rst)
begin
      if rst = '0' then
            D1n  <= D1i;
            D2n  <= D2i;
            Yn   <= '0';
      else
            if rising_edge(clk) then
                  D1n <= D1n1;
                  D2n <= D2n1;
```

```
                Yn  <= cmd;
            end if;
      end if;
end process;
end Behavioral;
```

Etude et conception d'une structure BIST
pour convertisseur analogique-numérique

Le test de circuits analogiques et mixtes est de plus en plus difficile du fait de l'intégration d'un nombre croissant de composants complexes au sein d'un même système. Les techniques de BIST permettent la réalisation d'un test efficace en intégrant au système les ressources nécessaires au test. Dans cette thèse, nous présentons une structure BIST pour les Convertisseurs Analogiques-Numériques (CAN) tout numérique. Le générateur de stimuli est un oscillateur Sigma-Delta numérique délivrant, après un simple filtrage analogique, une sinusoïde. L'analyse de la réponse se fait au moyen d'un banc de filtres numériques séparant les différentes composantes harmoniques du signal issu du CAN. A partir de ces composantes harmoniques, différents paramètres spectraux sont calculés. Afin de valider cette structure, différents prototypes ont été conçu sur FPGA. Les résultats expérimentaux confirment la capacité de notre structure à tester efficacement un CAN 12 bits ayant un SNR de 70 dB.

Mots clefs Test embarqué | Oscillateur $\Delta\Sigma$ | BIST
Filtrage numérique | Filtrage adaptatif | CAN

BIST structure study and design
dedicated to analog-to-digital converter

Analog and mixed circuit testing is more and more difficult because of the integration of a growing number of complex components in one system. BIST techniques allow the realization of an effective test system by integrating the test resources to the system. In this thesis, we present a BIST Structure for Analog to Digital Converters (ADC). The stimuli generator is a digital Sigma-Delta oscillator delivering a sine wave after a simple analog filtering. A bank of digital filters separating the different harmonic components of the signal from the ADC is used to analyze of the response. From these harmonic components, different spectral parameters are calculated. To validate this structure, different prototypes have been designed on FPGA. The experimental results confirm the ability of our structure to effectively test a 12-bit ADC with a 70-dB-SNR.

Keywords Embedded test | $\Delta\Sigma$ oscillator | BIST
Digital filtering | Adaptive filtering | ADC

Discipline : électronique

Université Bordeaux 1, CNRS UMR 5218, Laboratoire IMS — Bâtiment A31
351 cours de la libération — 33405 TALENCE