Proposition d'architectures radio logicielles fpga pour démoduler simultanément et intégralement les bandes radios commerciales, en vue d'une indexation audio

Brunel Happi Tietche

THESE DE DOCTORAT DE
L'UNIVERSITE PIERRE ET MARIE CURIE

Spécialité

Electronique et communications

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

M. Brunel HAPPI TIETCHE

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Proposition d'architectures radio logicielles FPGA pour démoduler simultanément et intégralement les bandes radios commerciales, en vue d'une indexation audio

soutenue le

devant le jury composé de : (préciser la qualité de chacun des membres).

Pr. Asoke K. NANDI	Rapporteur
Pr. Christophe JEGO	Rapporteur
Pr. Yves LOUËT	Examinateur
Pr. Bertrand GRANADO	Examinateur
Pr. Aziz BENLARBI-DELAÏ	Examinateur
Pr. Bruce DENBY	Directeur de thèse
Pr. Olivier ROMAIN	Co-directeur de thèse

RESUME

L'expansion de la radio et le développement de nouveaux standards enrichissent la diversité et la quantité de données contenues sur les ondes de radiodiffusion. Il devient alors judicieux de développer un moteur de recherches qui aurait la capacité de rendre toutes ces données accessibles comme le font les moteurs de recherche sur internet à l'image de Google. Les possibilités offertes par un tel moteur s'il existe sont nombreuses. Ainsi, le projet SurfOnHertz, qui a été lancé en 2010 et s'est terminé en 2013, avait pour but de mettre au point un navigateur qui serait capable d'indexer les flux audios de toutes les stations radios. Cette indexation se traduirait, entre autres, par de la détection de mots clés dans les flux audios, la détection de publicités, la classification de genres musicaux. Le navigateur une fois mis au point deviendrait le premier moteur de recherches de genre à traiter les contenus radiodiffusés. Relever un tel challenge nécessite d'avoir un dispositif pour capter toutes les stations en cours de diffusion dans la zone géographique concernée, les démoduler et transmettre les contenus audios à un moteur d'indexation. Ainsi, les travaux de cette thèse visent à proposer des architectures numériques portées sur une plateforme SDR pour extraire, démoduler, et mettre à disposition le contenu audio de chacune des stations diffusées dans la zone géographique du récepteur. Vu le grand nombre de standards radio existants aujourd'hui, la thèse porte principalement les standards FM et DRM30. Cependant les méthodologies proposées sont extensibles à d'autres standards.

C'est à base d'un FPGA que la majeure partie des travaux a été menée. Le choix de ce type de composant est justifié de par les grandes possibilités qu'il offre en termes de parallélisme de traitements, de maitrise de ressources disponibles, et d'embarquabilité. Le développement des algorithmes a été fait dans un souci de minimisation de la quantité de blocs de calculs utilisés. D'ailleurs, bon nombre d'implémentations ont été réalisées sur un Stratix II, technologie aux ressources limitées par rapport aux FPGAs d'aujourd'hui disponibles sur le marché. Cela atteste la viabilité des algorithmes présentés.

Les algorithmes proposés opèrent ainsi l'extraction simultanée de tous les canaux radios lorsque les stations ne peuvent occuper que des emplacements uniformément espacés comme la FM en Europe occidentale, et aussi, pour des standards dont la répartition des stations dans le spectre semble plutôt aléatoire comme le DRM30. Une autre partie des discussions porte sur le moyen de les démoduler simultanément.

ABSTRACT

The expansion of the radio and the development of new standards enrich the diversity and the amount of data carried by the broadcast radio waves. It becomes wise to develop a search engine that has the capacity to make these accessible as do the search engines on the internet like Google. Such an engine can offer many possibilities. In that vein, the SurfOnHertz project, which was launched in 2010 and ended in 2013, aimed to develop a browser that is capable of indexing audio streams of all radio stations. This indexing would result, among others, in the detection of keywords in the audio streams, the detection of commercials, the classification of musical genres. The browser once developed would become the first search engine of its kind to address the broadcast content. Taking up such a challenge requires to have a device to capture all the stations being broadcasted in the geographical area concerned, demodulate them and transmit the audio contents to the indexing engine. Thus, the work of this thesis aim to provide digital architectures carried on a SDR platform for extracting, demodulating, and making available the audio content of each broadcast stations in the geographic area of the receiver. Before the large number of radio standards which exist today, the thesis focuses FM and DRM30 standards. However the proposed methodologies are extensible to other standards.

The bulk of the work is FPGA-based. The choice of this type of component is justified by the great opportunities it offers in terms of parallelism of treatments, mastery of available resources, and embeddability. The development of algorithms was done for the sake of minimizing the amount of the used calculations blocks. Moreover, many implementations have been performed on a Stratix II technology which has limited resources compared to those of the FPGAs available today on the market. This attests to the viability of the presented algorithms.

The proposed algorithms thus operate simultaneous extraction of all radio channels when the stations can only occupy uniformly spaced locations like FM in Western Europe, and also for standards of which the distribution of stations in the spectrum seems rather random as the DRM30. Another part of the discussion focuses on the means of simultaneously demodulating it.

REMERCIEMENTS

Je souhaite, pour commencer, remercier Monsieur Asoke K. Nandi et Monsieur Christophe Jego pour avoir accepté de rapporter mes travaux malgré les reports de date de soutenance de ces derniers mois. De même, ma gratitude va vers Monsieur Yves Louët, Monsieur Bertrand Granado ainsi que Monsieur Aziz Benlarbi-Delaï, membres du jury, qui partageront la responsabilité d'évaluer mon travail.

Ensuite, je tiens à rendre à Monsieur Bruce Denby, directeur de cette thèse, ce qui lui est dû. Sans lui, il n'y aurait pas cette thèse. Je lui dois en grande partie les conférences et séminaires auxquels j'ai participé. De plus, sa rigueur dans la rédaction d'articles scientifiques m'aura énormément appris durant toutes ces années.

Pour continuer, je voudrais remercier Monsieur Olivier Romain qui, avec Monsieur Denby, ont su me procurer tout le matériel dont j'avais besoin. Il m'a aussi encadré de près, a pris à cœur mon avenir professionnel dès le début de ma thèse. Comme il aime à dire : « ne t'inquiète pas, ton avenir professionnel est tout tracé ».

Je n'oublie pas tous mes collègues ou anciens collègues des différents laboratoires par lesquels je suis passé : Ceux de Sigma (ESPCI), ceux de l'équipe SYEL du Lip6 et ceux du laboratoire ETIS. En particulier, mes pensées vont vers Lounis Zerioul, mon actuel collègue de bureau qui m'a apporté un soutien précieux ces dernières semaines, voire ces derniers mois. Je lui dis souvent : « ah Lounis…que ferais-je sans toi ? ».

Comment ne pas remercier aussi ma famille et mes amis ? Vous qui partout dans le monde me gardez dans vos cœurs. Puissé-je être pour vous un modèle en foi, en persévérance et en conduite.

A ma mère

La vie ne t'a pas épargné les dures épreuves…Combien de fois tu as dû penser que c'était la fin ? Courage ! Les épreuves sont passagères. Encore un peu de temps, et voici qu'arrive le pays où il n'y aura plus de pleurs, de douleurs ni d'alarmes. C'est là l'objectif à atteindre.

SOMMAIRE

ABRÉVIATIONS

AAC	Advanced Audio Coding
ADC / CAN	Analog-to-Digital Converter / Convertisseur Analogique Numérique
AF	Alternative Frequencies
ALUTs	Adaptive Look-Up Table
AM	Amplitude Modulation
AMR-WB	Adaptive Multi-Rate Wideband
AMSS	Amplitude Modulation Signalling System
ASIC	Application-Specific Integrated Circuit
AUPELF	Association des universités partiellement ou entièrement de langue française
AVC (MPEG-4 AVC/H.264)	Advanced Video Coding
Bande OIRT	65.8-74 MHz
BPSK	Binary PSK
C/N	Carrier-to-Noise ratio
CAM-D	Compatible Amplitude Modulation – Digital
CDC	Complex Down Converter
CELP	Code Excited Linear Prediction
CFCFO	Coarse Fractional Carrier Frequency Offset
CIC	Cascaded Integrator–Comb
CODEC	COdeur/DECodeur audio
CST	Coarse Symbol Timing
CT	Clock Time
CUC	Complex Up Converter
DAB	Digital Audio Broadcasting,
DAC / CNA	Digital-to-Analog Converter / Convertisseur Numérique Analogique
DARC	Data Radio Channel
DDC	Digital Down Converter
DFT	Discrete Fourier Transform
DoD	Department Of Defense

DQPSK	Differential Quadrature PSK
DRM	Digital Radio Mondiale
DSP	Digital Signal Processor
ECC	Extended Country Code
EPG	Electronic Program Guide
ER-BSAC	Error Resilient - Bit Sliced Arithmetic Coding
ESTER	Campagne d'Evaluation des Systèmes de Transcription enrichie d'Emissions Radiophoniques
ETSI	European Telecommunications Standards Institute
FAC	Fast Access Channel
FFCFO	Fine Fractional Carrier Frequency Offset
FFT	Fast Fourier Transform
FI	Fréquence Intermédiaire
FIFO	First In First Out
FIR	Finite Impulse Response
FM	Frequency Modulation
FPGA	Field-Programmable Gate Array
FT	Frame Timing
GO	Grandes Ondes (150 – 281 kHz)
GPGPU	General-Purpose computation on Graphics Processing Units)
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HDC	Hybrid Digital Coding
HE-AAC	High-Efficiency Advanced Audio Coding
HVXC	Harmonic Vector eXcitation Coding
IBOC	In-Band On-Channel
ICFO	Integer Carrier Frequency Offset
IHM	Interface Homme Machine
IP	Intellectual Property
ISDB-Tsb	Integrated Services Digital Broacasting Terrestrial Sound Broadcasting
LDC	Linguistic Data Consortium
LMSK	Level-controlled Minimum-Shift Keying
LNA	Low Noise Amplifier
LUT	Adaptive Look-Up Table

MFCCs	Mel-Frequency Cepstral Coefficients
MPEG	Moving Picture Experts Group
MPSoC	Multi Processor System-on-Chip
MSB	Most Significant Bit
MSC	Main Service Channel
NCO	Numerically Controlled Oscillator
NIST	National Institute of Standards and Technology
OC	Ondes Courtes (2.3 – 26.1 MHz)
Offset	Décalage ou erreur
OIRT	Organisation Internationale de Radiodiffusion et de Télévision
OL	Oscillateur Local
PDA	Personal Digital Assistant
PDFT	Polyphase DFT
PFFT	Pipelined FFT
PI	Program Indentification
PLL	Phase – Locked Loop
PO	Petites Ondes (531 – 1710 kHz)
PS	Program Service
PSK	Phase-Shift Keying
PTY	Program Type
PTYN	Program TYpe Name
QAM	Quadrature amplitude modulation
RBDS	Radio Data Broadcast System
RDS	Radio Data System
RF	Radiofréquences
RFI	Radio France International
RFID	Radio Frequency Identification
RMD	Robustness Mode Detection
ROM	Read Only Memory
RSB	Rapport Signal sur Bruit
RT	Radio Text
SBR	Spectral Band Replication
SCFO	Sampling Clock Frequency Offset
SDC	Service Description Channel
SDR	Software Defined Radio

SFDR	Spurious Free Dynamic Range
STO	Symbol Timing Offset
T-DMB	Terrestrial Digital Multimedia Broadcasting)
TA	Traffic Announcement
TF	Transformée de Fourier
TMC	Traffic Message Channel
TP	Traffic Program
VHF	Very High Frequency

GLOSSAIRE

AM HD Radio	Système de radio IBOC pour le standard AM développé par Ibiquity Digital
Bande de base / Zéro-FI	Désigne le spectre fréquentiel situé à la fréquence nulle
Bufferisé	Mis dans une mémoire tampon
Channelization	Procédé qui consiste à appliquer un banc de filtres à un signal
Channelizer	Ici, il s'agit de l'action d'opérer la channelization. Dans la littérature anglosaxonne, ce terme désignera plutôt le système qui opère la channelization
Channelizeur	Système qui opère la channelization
Chirp	Signal de test modulé en fréquence
Equiripple	Filtre dont les « ripples » dans la bande passante sont de même niveau et celles de la bande rejetée sont aussi de même niveau
FIFO « dual clock »	FIFO avec dont l'horloge d'entrée des échantillons est différente de l'horloge de sortie des échantillons
Filtre anti-repliement	Filtre qui permet, avant échantillonnage d'un signal, de rejeter les fréquences supérieures au double de la fréquence d'échantillonnage
Filtres en boucle (ou loop filter)	Filtres dont l'architecture intègre une structure de rebouclage
FM HD Radio	Système de radio IBOC pour le standard FM développé par Ibiquity Digital
FMeXtra	Système de radio IBOC pour le standard FM développé par Digital Radio Express, maintenant VuCast Media, Inc.
Frequency sampling-based filter	Filtre dont la réponse impulsionnelle est calculée à partir du gabarit souhaité du filtre
IP3	Point d'interception des produits d'intermodulation du 3ième ordre.
Itunes Tagging	procédé permettant à l'utilisateur (muni d'un récepteur HD Radio équippé d'un bouton spécial « Tag ») de tagger les musiques qu'il écoute à la radio afin de les voir référencées sur Itunes
kS/s	kilo-échantillons par seconde
Multiplication rate	Nombre de multiplications par instant d'horloge système d'une

	architecture numérique
Nyquist	Le critère de Nyquist stipule qu'un signal doit âtre échantillonné au moins deux plus rapidement que sa fréquence la plus élevée afin d'avoir une reconstruction correcte du signal
Open source	Package logiciel dont la licence offre des possibilités de libre redistribution, d'accès au code source et de création des travaux dérivés
Overlap (Overlap Add)	L' « Overlap » consiste à superposer plusieurs parties d'un signal temporel numérique sur lui même avec d'effectuer la sommation (« Add ») des différentes parties superposées.
Pige musicale, publicitaire, ou de programmes	Comptabilisation et relevé du nombre de titres, publicités ou programme diffusés par un média
Projet SPEAKeasy	Projet de l'armée américaine d'application de la radio logicielle
Re-échantillonnage	Procédé consistant à modifier la fréquence d'échantillonnage d'un signal numérisé
Ripple	ondulations du spectre. Elles peuvent apparaître, par exemple, dans la bande passante de la réponse en fréquence d'un filtre.
Sous-échantillonnage	Re-échantillonnage d'un signal à une fréquence inférieure à sa fréquence d'échantillonnage d'origine
Sous-échantillonnage (Undersampling)	Échantillonnage d'un signal à une fréquence inférieure au double de sa fréquence maximale sans replier le spectre du signal sur lui-même
Sparse channelization	Channelization dont les canaux à extraire ne sont pas uniformément espacés, mais plutôt aléatoirement
Stage (filtre polyphase)	Désigne un étage de la structure polyphasée du filtre
Sur-échantillonnage	Re-échantillonnage d'un signal à une fréquence supérieure à sa fréquence d'échantillonnage d'origine
Taps	Dans une ligne à retard numérique, il s'agit du nombre d'instants-horloges dont sera retardé chaque échantillon. Dans un filtre FIR polyphase, il s'agit d'un nombre de retards numériques instanciés dans un étage
Underflow	C'est lorsqu'on envoie une requête de lecture à une FIFO alors que celle-ci est vide

ViaVoice	Logiciels de reconnaissance vocale de IBM
Zlib / Deflate	« Zlib » est une librairies de fonctions de compression et décompression, incluant la vérification de l'intégrité des données décompressées. « Deflate() » est l'une de ses fonctions de compression.

1 Introduction

Les vecteurs de diffusion de l'information sont nombreux. Il y a la voie orale, le livre, la cassette ou plutôt, aujourd'hui, le CD, le DVD, la clé USB, la télévision, il y a aussi internet et enfin il y a la radio. La radio est un moyen de diffusion qui est bien connu de la plupart de citoyens du monde, qui est toujours présent dans différentes contrées où l'accès à internet ou la télévision est difficile, voire inexistant.

Une seule station radio, à elle seule, véhicule déjà une quantité d'information non négligeable. Par exemple, une chaîne FM en une journée peut diffuser de la musique, des journaux, des interviews, du sport, la météo, des informations sur le traffic etc. Si on considère maintenant toutes les autres stations FM qui diffusent en même temps que la première, on réalise la diversité des informations qui sont véhiculées en même temps, et ce du matin jusqu'au soir. Enfin, si on prend en compte le fait qu'il y a d'autres bandes radios comme par exemple l'AM, la DRM, le DAB ou le T-DMB, on arrive à la conclusion que les bandes radios, prises ensemble, constituent une mine d'informations.

Pourtant, la radio a un caractère très volatile, elle ne « parle pas deux fois ». A moins qu'une émission soit rediffusée sur la même station, l'auditeur qui n'écoutait pas la bonne station au moment de la diffusion ne bénéficiera pas de son contenu. Ainsi, il faut faire de la mine d'informations une base de données, dont les données sont répertoriées, voire sauvegardées et facilement accessibles.

Le monitoring des médias est le procédé qui consiste à capter, enregistrer, étiquetter et analyser les contenus des médias de diffusion. Le sociétés spécialisées dans ce domaine utilisent généralement de grandes installations contenant de nombreux récepteurs, situés sur des sites répandus dans toutes la zone de diffusion, pour suivre ce qui concerne le propriété intellectuelle pour leurs clients : statistiques sur les annonces commerciales ; problèmes de copyright ; statistiques sur les temps de diffusion des contenus musicaux ; statistiques sur les mentions et les utilisations des noms de sociétés ou de produits ; etc. Aujourd'hui, les techniques radio logicielles et le nombre important d'éléments logiques sur les FPGAs, Field Programmable Gate Array devices, FPGA, nous ont conduit à imaginer les potentialités d'une telle installation de monitoring de médias sur une seule puce. Non seulement cela procurerait une solution moins couteuse et plus gérable, mais en plus ça ouvrirait la porte à à de nouveaux axes de recherche, hybridation de la radio, profil comportemental, etc.

La capacité de capter, démoduler toutes les stations radios en temps réel sur un dispositif unique

et portable peut permettre à des applications telles que : des interfaces hommes-machines interactives reflétant le contenu ; la navigation intelligente du contenu de la radio ; les playlists organisées selon les goûts de l'utilisateur; l'identifications des genres musicaux ; la recherche de mots clés sur ce qui est diffusé ; la recherche de musiques ou de émissions particulières, et une panoplie d'autres possibilités. Ma thèse, qui s'est inscrite dans le cadre du projet ANR SurfOnHertz (ARPEGE 2009 - coordonnée par O. Romain – ETIS) dont un des objectifs est de développer un navigateur hertzien capable de rechercher des informations sur les flux radiodiffusés simultanéement.

Comme la radio FM reste la norme la plus diffusée dans le monde, et la radio DRM est destinée à remplacer progressivement la bande AM, les travaux présentés concernerons spécifiquement la bande FM (européenne) et la bande DRM30. La méthodologie développée pourra naturellement être utilisée pour toute bande FM similaire. En outre, nous avons choisi une implémentation de notre système sur un FPGA, compte tenu des possibilités offertes par ce type de dispositif pour créer des architectures massivement parallélisées. Le prototypage rapide reste aujourd'hui la première étape nécessaire vers une intégration monolithique.

Mes travaux ont été réalisés au sein de différents laboratoires : SIGMA de l'école ESPCI ParisTech (référent : B. Denby), Lip6 de l'UPMC (référent : O. Romain) et ETIS de l'université de Cergy Pontoise (référent : O. Romain). Ils ont été co-dirigés par le Professeur Olivier Romain, sous la direction du Professeur Bruce Denby. La majeure partie de ces travaux a été financée par l'ANR, sur le programme ARPEGE2009.

Le document est organisé en 4 parties. Dans la première partie il sera question de situer le contexte et la problématique de la thèse. Nous découvrirons le projet à son origine. On fera aussi un état de l'art des récepteurs multi-standards et/ou multi-canaux existants. La partie suivante, à savoir le chapitre 3 donnera une description des architectures proposées pour la FM et la DRM30.

Le re-échantillonnage sera une étape incontournable dans le principe de séparation et de démodulation des canaux FM. Ainsi, dans le quatrième chapitre, une nouvelle méthode de re-échantillonnage pour FPGAs sera décrite, présentant certains avantages par rapport aux méthodes existantes. Les prototypes et les résultats obtenus par les systèmes de réception proposés seront présentés dans le chapitre 5. Ces derniers seront accompagnés des premiers prototypes d'application d'indexation audio en temps réel de la radio. Ce sera d'ailleurs l'occasion de situer notre contribution par rapport à ce qui existe déjà. Enfin, nous concluerons sur le travail accompli et parcourerons des perspectives offertes par ce dernier.

2 Contexte, problématique et état de l'art

En France, plusieurs essais de diffusion de la radio via des standards numériques sont en cours. L'objectif est que, à terme, les bandes de radiodiffusion commerciales émettent les programmes en numérique sur les standards DRM (Digital Radio Mondiale) et T-DMB (Terrestrial Digital Multimedia Broadcasting) en plus de la diffusion analogique classique sur les bandes AM et FM. Ce changement aura pour conséquence d'enrichir non seulement la diversité des contenus, mais aussi les possibilités d'indexer ces contenus, soit avec les métadonnées qui accompagnent de plus en plus ces émissions (photo, titre artiste, etc.), soit via une analyse directe des données captées. Les bandes hertziennes représenteront alors de véritables sources riches, mais jusqu'ici sous-exploitées, de contenus multimédias.

A l'image d'internet et de Google, il sera alors indispensable de disposer d'un navigateur hertzien pour exploiter pleinement ces nouvelles sources de données. Seulement, la diversité de besoins dans les domaines d'exploitation de ces flux a jusqu'ici freiné le développement de nouveaux produits dans ce secteur naissant. On peut citer deux grandes classes d'applications :

1. **La création de nouveaux services** pour le consommateur, tels que l'affichage d'artiste, la recherche de mots clés, la recherche de morceaux ou l'identification du genre de musique. Il s'agit de la « radio à la demande » embarquée, dont les solutions proposées à ce jour restent rudimentaires, avec une interface mono-canal et une fonctionnalité très restreinte.
2. **Le monitoring des médias** via la pige musicale, publicitaire, ou de programmes. Dans ce cas les installations sont obligatoirement multi-canaux et d'une fonctionnalité très poussée. Et comme il faut dupliquer le matériel de captage/analyse sur chaque canal, elles sont de nécessité fixes, encombrantes, et de consommation d'énergie élevée.

2.1 Standards de radio diffusion existants et caractéristiques

2.1.1 Les différents standards de radiodiffusion

Les différents standards de radiodiffusion terrestres peuvent être regroupés en trois catégories, selon le type de modulation utilisée:

- Les standards analogiques
- Les standards numériques
- Les standards mixtes

Au sein de ces catégories, les standards se diversifient par les bandes de fréquence dans

lesquelles ils opèrent, par les modulations utilisées, par les types de compression mises en jeu ou encore par le type de métadonnées qu'ils transmettent en plus du son monophonique.

Les standards analogiques mettent en œuvre deux types de modulations : une d'amplitude AM sur les bandes GO (Grandes Ondes, 150 – 281 kHz), PO (Petites Ondes, 531 – 1710 kHz), OC (Ondes Courtes, 2.3 – 26.1 MHz) et, une modulation de fréquence (FM). Selon les régions du monde, la bande FM est localisée à différentes positions dans le spectre de radio diffusion. Ainsi, elle couvre la bande de fréquence de 76 à 90 MHz au Japon et dans le reste du monde, elle est comprise entre 87.5 et 108 MHz même si quelques pays en Europe de l'Est utilisent encore la bande OIRT (Organisation Internationale de Radiodiffusion et de Télévision, 65.8-74 MHz) comme la Russie, qui l'utilise conjointement à la bande 87.5-108 MHz.

Les deux autres catégories (la deuxième et la troisième catégorie) sont encore plus diversifiée que la première par leur nombre de standards plus élévé. Ainsi, parmi les standards numériques, il y a le DAB (Digital Audio Broadcasting), la DRM, le T-DMB et l'ISDB-Tsb (Integrated Services Digital Broacasting Terrestrial Sound Broadcasting). Les standards mixtes, i.e. ceux qui combinent à la fois les modulations analogiques et les modulations numériques sont appelées IBOC (In-Band On-Channel). Ils offrent la possibilité de transmettre simultanément des modulations analogiques et modulations numériques sur le même canal. Les standards IBOC les plus populaires sont le CAM-D (Compatible Amplitude Modulation – Digital), AM HD Radio, FM HD Radio et FMeXtra.

Ces standards offrent de façon générale plus de possibilités et de services que les standards analogiques sauf lors de mauvaises conditions de transmission (SNR faible, etc.). Dans ces cas là, la plupart de ces standards diffusent alors dans leur mode le plus robuste, privilégiant ainsi la diffusion du son monophonique ou de la parole, à d'autres informations.

Cependant, les standards analogiques disposent d'un système de transmission numérique de métadonnées. Les métadonnées sont des informations qui visent à améliorer l'expérience radiophonique, comme le nom de la station écoutée ou bien la langue du programme diffusé. Pour le standard AM, ce système c'est le AMSS (Amplitude Modulation Signaling System). Pour la FM, c'est le RDS (Radio Data System) ou le RBDS (Radio Data Broadcast System). Les types de métadonnées qui véhiculent des informations directement à l'utilisateur (via son récepteur) sont les suivantes :

- **PS (Program Service)**. Le nom de la station.
- **AF (Alternative Frequencies)**. La liste des fréquences des émetteurs voisins de la même station.
- **CT (Clock Time)**. L'heure courante.
- La date courante.
- Le pays d'émission.

- La langue de diffusion du programme.
- **TP (Traffic Program)**. Le drapeau qui indique si la station reçue est susceptible d'émettre des annonces routières.
- **TA (Traffic Announcement)**. Le drapeau TA indique que la station émet *en ce moment* une annonce routière.
- **RT (Radio Text)**. Il permet la diffusion de textes, comme les références des morceaux musicaux en cours de diffusion.
- **RT+**. Il sert à étiqueter certains des passages textuels des messages de radiotexte par des métadonnées décrivant leur nature. Par exemple, on peut indiquer qu'un certain fragment de message est un titre de chanson, un nom d'artiste.
- **PTY (Program TYpe) et PTYN (Program TYpe Name)**. Ils indiquent le type de programme diffusé.
- **PI (Program Indentification) et ECC (Extended Country Code)**. Le couple PI+ECC constitue un identifiant unique d'une station au niveau mondial.
- **TMC (Traffic Message Channel)**. Il permet la transmission pour les données structurées d'informations sur le trafic.

2.1.2 Caractéristiques des différents standards

Les bandes de transmission des différents standards cités jusqu'ici, à savoir les standards de radio terrestres, sont listées dans le tableau 2.1.

	AM	FM	T-DAB		DRM	
			DAB	DAB+	DRM 30	DRM+
Bande de diffusion	150-281 kHz 531-1710 kHz 2.3-26.1 MHz	65.8-74 MHz 76-108 MHz	174-239 MHz 1452-1467.5 MHz	174-239 MHz 1452-1467.5 MHz	3-30 MHz	30-174 MHz
	IBOC				ISDB-Tsb	T-DMB
	CAM-D	AM HD Radio	FM HD Radio	FMeXtra		
Bande de diffusion	150-281 kHz 531-1710 kHz 2.3-26.1 MHz	150-281 kHz 531-1710 kHz 2.3-26.1 MHz	87.8-108 MHz	87.8-108 MHz	188-192 MHz	174-239 MHz 1452-1492 MHz

Tableau 2.1 – Listes des différents standards de radios disponibles

Chaque standard radio ayant ses propres caractéristiques (espacement, type de modulation, etc.), celles-ci sont résumées dans le tableau 2.2.

Standard		Largeur des canaux	Type de modulation	Aspect numérique		
				Débit net	Type de Compression	Métadonnées ?
AM		5, 9 ou 10 kHz	AM + BPSK pour le AMSS (Numérique)	AMSS : 47 bits/s		Oui
FM		100 ou 200 kHz	FM +BPSK, LMSK resp. pour le RDS, DARC (Numérique)	RDS : 1.2 kbits/s		
				DARC : 16 kbits/s	Zlib/Deflate	
T-DAB	**DAB**	1.536 MHz	DQPSK	0.8-1.7 Mbits/s	MPEG-1 Audio Layer II	Oui Autres possibilités : EPG[1], fonction Pause/Rembobinage/Enregistrement, Séquences d'images, différents formats stéréo possibles (ex : 2.0, 3.2 ou 5.1 pour le DAB+)
	DAB+				HE-AAC v2 (AAC+)	
DRM	**DRM30**	4.5, 5, 9, 10, 18 ou 20 kHz	16, 64 QAM	4-70 kbits/s	MPEG-2 AAC, MPEG-2 CELP, MPEG-2 HVXC	Oui Autres possibilités : EPG, Séquences d'images, Son stéréo 5.1
	DRM+	96 kHz	4, 16 QAM	35-185 kbits/s	MPEG-2 AAC	
IBOC	**CAM-D**	15 kHz				
	AM HD Radio	20, 30 kHz	BPSK, 16/64 QAM (avec en plus AM+BPSK éventuellement)	20-60 kbits/s	HDC+SBR	Oui possibilités : Itunes Tagging, fonction Pause
	FM HD Radio	200, 400 kHz	FM, BPSK, QPSK	25-300 kbits/s	Basé sur MPEG4 HE-AAC	
	FMeXtra		QPSK, 8-PSK, 16-QAM, 32-QAM		AAC, AAC+, AAC+ v2, AMR-WB, AMR-WB+	Oui possibilités : Surround sound 5.1
ISDB -Tsb	**1-segment**	429.6, 430.6, 432.5 kHz	QPSK, DQPSK, 16QAM et 64QAM	842.55-5361.84 kbits/s	MPEG-2 AAC (audio), MPEG-4 AVC/H.264 (vidéo)	Oui Possibilités : vidéos, images.
	3-segment	1.287, 1.288, 1.29 MHz		280.85 - 1787.28 kbits/s		Oui Autres possibilités : vidéos, images, Son stéréo 5.1, diffusion simultanée en plusieurs langues.
T-DMB pour la radio		1.536 MHz	DQPSK		HE-AAC v2, ER-BSAC, MPEG-4 AVC/H.264 (images)	Oui Autres possibilités : EPG, logo, Séquences d'images, différents formats stéréo possibles (MPEG Surround. ex: 5.1 etc.)

(1) ***EPG :*** *Il délivre des informations sur le programme de radio présent et les émissions à venir.*

Tableau 2.2 – Caractéristiques des différents standards de radios disponibles

En plus des systèmes de radiodiffusion terrestres, il y a les systèmes de radiodiffusion par satellite. Ce sont des systèmes propriétaires dont les caractéristiques techniques sont moins accessibles. Le tableau 2.3 dresse la liste des systèmes satellitaires encore utilisés, avec leurs caractéristiques :

Standard satellitaire	**Bande utilisée**	**Largeur des canaux**	**Type(s) de modulation**	**Débit net**	**Type de compression**
ADR (Astra Digital Radio)	10,7-11,7 GHz	260 kHz	Frequency Modulated QPSK	192 kbits/s	MPEG-1 Layer II
1WorldSpace	1452-1492 MHz		QPSK	≤128 kbits/s	MPEG-1 Layer III
Sirius, XMRadio	2330-2345 MHz 2520-2670 MHz				

Tableau 2.3 – Caractéristiques des différents standards de radios par satellite utilisés aujourd'hui

En France, il était prévu dès 2011 que la radio numérique prendrait la place de l'analogique. Celle-ci devait alors disparaitre totalement à l'horizon 2015. Aussi, le Ministère de la Culture et de la Communication avait décidé par un arrêté de 2007 que les deux normes retenues pour la radio numérique seront la DRM et le T-DMB respectivement pour l'AM et la FM. Plutard, en 2013, le gouvernement a ajouté la norme DAB+ comme norme pour la Radio Numérique Terrestre (RNT).

1. **DRM.** La DRM (Digital Radio Mondiale) a été lancé officiellement le 16 Juin 2003 à Genève pour donner un nouveau souffle à la diffusion AM, dans les bandes de fréquences inférieures à 30 MHz. La DRM permet de diffuser un signal radio numérique sur les bandes de fréquences utilisées actuellement par la modulation d'amplitude. Actuellement, on dénombre 12 stations [DRMorg] qui émettent des programmes en Europe dont 1 en France (RFI, Radio France International). La DRM a été étendue jusqu'aux fréquences inférieurs à 174MHz. On parle alors de DRM30 quand il s'agit des fréquences inférieures de 30 MHz et de DRM+ pour celles qui sont supérieures.
2. **T-DMB.** Le DAB (Digital Audio Broadcasting) est un standard de radiodiffusion de contenu

multimédia qui a été accepté par plusieurs pays, et actuellement plus de 500 millions (www.worlddab.org) de personnes peuvent recevoir dans le monde, plus de 1000 services différents. Comme pour la DRM, la diffusion en DAB procure pour l´utilisateur de multiples avantages; le son est de qualité numérique et les données associées sous forme de texte (programme en cours, liste de programmes à venir, etc.) et d'image (pochette de l'album, logo de la radio, etc.). Depuis 2005, une extension du DAB a été normalisée par l'ETSI (European Telecommunications Standards Institute) permettant d'améliorer la réception des programmes et d'étendre les services à la diffusion de vidéo et de programmes télévisés. Ce complément de norme s'appelle le T-DMB (Terrestrial Digital Multimedia Broadcasting).

3. **DAB+.** Elle est aujourd'hui en vigueur dans plusieurs pays européens comme la Grande Bretagne, l'Allemagne ou la Suisse.

Il y a donc une grande diversité de standards et de canaux possibles. On effet, à partir de la largeur des canaux et de la largeur des bandes de chaque standard, on peut déduire le nombre maximal de stations pouvant coexister pour chaque standard (tableau 2.4). On peut alors imaginer la variété, la richesse de l'information que peut contenir chaque bande. Pour prendre en charge tout cela, il est nécessaire de mettre en œuvre une solution qui soit flexible pouvant gérer plusieurs canaux de différents standards. La radiologicielle est une technique permettant de concevoir un système adapté à ce genre de besoin. En effet, elle permet d'envisager, au sein du même dispositif, un démodulateur multi-stations multi-standard de par sa nature.

	AM	**FM**	**T-DAB**	**DRM**		**IBOC**		
				DRM 30	**DRM+**	**CAM-D**	**AM HD Radio**	**FM HD Radio**
Nombre de stations possibles	> 2700	> 200	> 360	> 2700	1500	> 1670	> 840	> 50

Tableau 2.4 – Nombre possible de stations pour quelques standards radios de radios disponibles

2.2 La radiologicielle.

2.2.1 Définition

La radio logicielle - également nommée "Software Defined Radio" ou SDR – est un système de réception et / ou d'émission dans lequel une partie (plus ou moins majoritaire) des traitements est faite ou contrôlée de façon logicielle. Dans le cadre de la réception, la radio logicielle est une approche de traitement du signal qui consiste à numériser tout ou partie des traitements qui sont destinés à un standard de radiocommunication ou de télécommunication donné. Un tel récepteur possède alors une partie « hardware » qui sont la circuiterie électronique de réception et une partie « software » logicielle.

Elle tire son origine dans les années 70 où un laboratoire du département de la Défense des États-Unis (ou DoD pour Department Of Defense) en Californie a créé un outil logiciel pour faire de l'analyse en bande de base et appelé Midas. On parlait alors de « digital receiver ». C'est plus tard, dès les années 90, que la radio logicielle a été rendue très populaire grâce à Joseph Mitola et au projet SPEAKeasy. En effet, en 1991, Mitola émit l'idée d'appliquer la combinaison d'un « digital receiver » et d'autres dispositifs au GSM, ouvrant ainsi le concept de la la radio logicielle aux applications civiles. SPEAKeasy est né d'un besoin de l'armée américaine d'avoir des dispositifs capables de communiquer avec différents standards de communications et capables d'évoluer avec les normes. En 1994, la démonstration a été faite grâce à un système constitué de plusieurs centaines de processeurs installé à l'arrière d'un camion [Lackey95].

La radio logicielle a ouvert la voie de la versatilité des dispositifs de réception.

2.2.2 La réception multistandards

Concrètement, un récepteur multi-standards radio logicielle peut être conçu suivant plusieurs différentes approches. Voici la plupart d'entre elles:

- L'approche idéale
- L'approche « à éléments en parallèle »
- L'approche homodyne
- L'approche super-hétérodyne

2.2.2.1 L'approche idéale

L'approche idéale pour faire un récepteur radio logiciel est sans doute celle qui consiste à numériser la totalité du signal issu à la sortie de l'antenne [Ken05] ou éventuellement juste après qu'il ait été amplifié par un amplificateur faible bruit (ou LNA pour Low Noise Amplifier) et filtré par un filtre anti-repliement. Tout le reste des traitements est alors effectué par une architecture programmée. Ainsi, l'extraction des différents canaux du signal, le mixage du signal pour le ramener à la bande de base, la démodulation du signal et toutes les autres opérations seront réalisées par des traitements numériques. La figure 2.1 illustre ce principe :

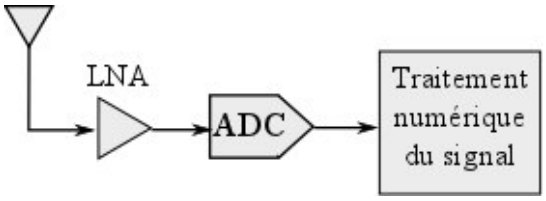


Figure 2.1 – Récepteur multi-standard radio logiciel idéal. Adapté de [Ken05]

2.2.2.2 L'approche « à éléments en parallèle »

L'approche idéale mentionnée peut s'avérer difficile à mettre en œuvre dans de nombreux cas. De nombreux problèmes techniques limitent cette approche (antennes adaptables, amplification IWB, numérisation, etc.). Par exemple, si la bande à traiter va de 15 MHz à 3 GHz, l'horloge qui cadence l'ADC devra avoir une fréquence supérieure ou égale à 6 GHz afin que la totalité du spectre soit conservée, selon le critère de Nyquist. Il faudra aussi que l'ADC soit d'une grande précision (voire supérieure à 20 bits effectifs) à cause de la grande disparité de puissance à différents endroits de ce large spectre.

En pratique, un tel ADC fonctionnant à une telle cadence entrainerait certainement une forte consommation et l'obligation d'avoir un dispositif qui génère un horloge aussi rapide. Ces conditions sont difficiles à remplir au sein d'un système embarqué. La technique d'un récepteur telle que proposée par Yoshida *et al* [Yos03] est un début de réponse pour ce genre de situations. En effet, elle inspire un modèle de récepteurs multi-standards aussi surnommé «récepteur à éléments en parallèle » [Lak09]. Il consiste, avant numérisation, à faire passer le signal par une voie, un « élément en parallèle », parmi plusieurs éléments prévus. Chaque élément parallèle peut correspondre à une bande précise du spectre, un standard précis, ou un regroupement de bandes précises. Un tel système peut considérablement réduire les contraintes sur l'ADC. La figure 2.2

donne un aperçu de cette approche.

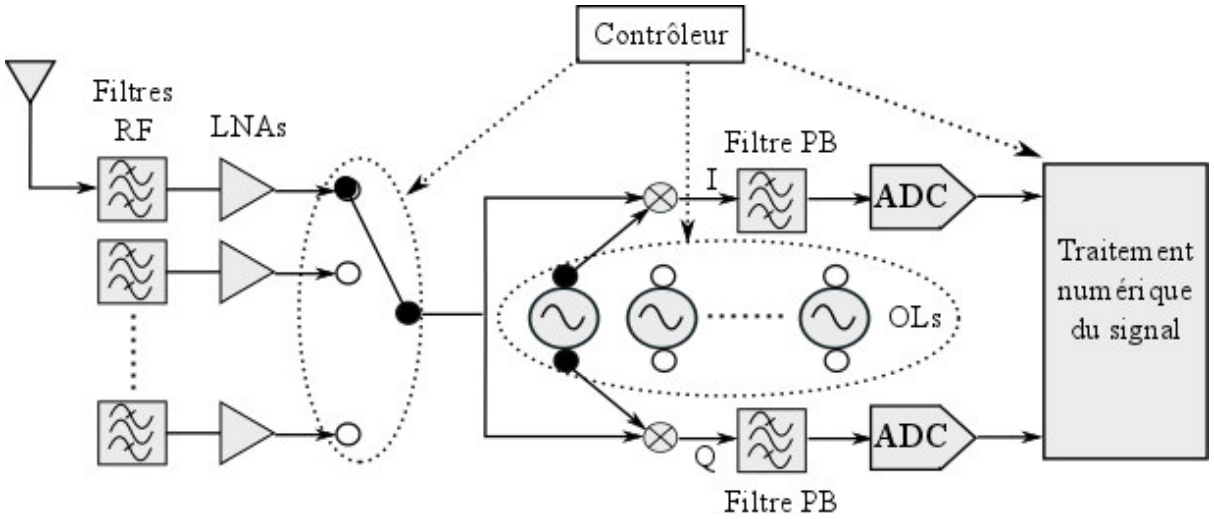


Figure 2.2 – Un récepteur multistandards à éléments en parallèle. Le contrôleur détermine par quels éléments parallèles passera le signal à traiter et quel sera le traitement numérique adapté

Les autres approches se distinguent chacune par le procédé avec lequel le signal reçu de l'antenne est ramené à la bande de base (zéro-FI).

2.2.2.3 L'approche par numérisation en RF avec sous-échantillonnage (undersampling)

Cette approche ne nécessite aucun étage de mixage analogique ou numérique. L'objectif ici est de choisir la fréquence d'horloge de l'ADC tel que le spectre du signal soit ramené à la bande de base par sous-échantillonnage (undersampling). Après quoi intervient directement le traitement numérique du signal. Bien sûr, il peut être nécessaire d'amplifier dès sa sortie de l'antenne, avant de l'envoyer à l'ADC (figure 2.3). Cette approche est sans doute l'une des plus proches de la radio logicielle idéale après l'approche idéale vue en section 2.2.2.1. Il n'y a pas de mixage analogique avant numérisation ce qui simplifie considérablement la partie analogique RF. Il faut néanmoins un filtrage analogique de bonne qualité pour éviter le repliement spectral après numérisation.

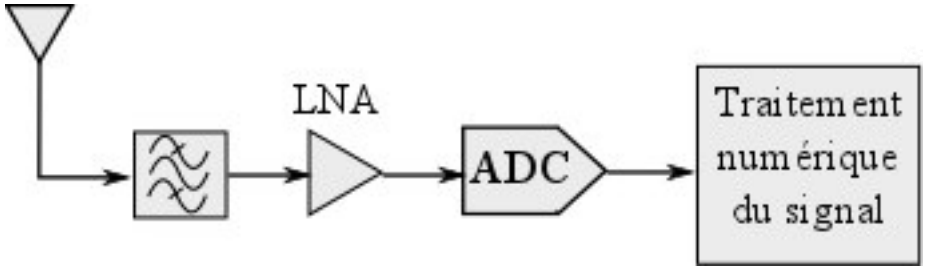


Figure 2.3 – Illustration de l'approche par numérisation en RF avec sous-échantillonnage (undersampling)

2.2.2.4 L'approche homodyne

Dans l'approche homodyne, on s'arrange à transposer le signal à la bande de base de façon analogique avant sa numérisation. On atténue ainsi les contraintes liées à la fréquence d'horloge de l'ADC. On appelle aussi de tels récepteurs « zéro-FI » ou « à conversion directe ». La figure 2.4 donne un exemple de récepteur homodyne.

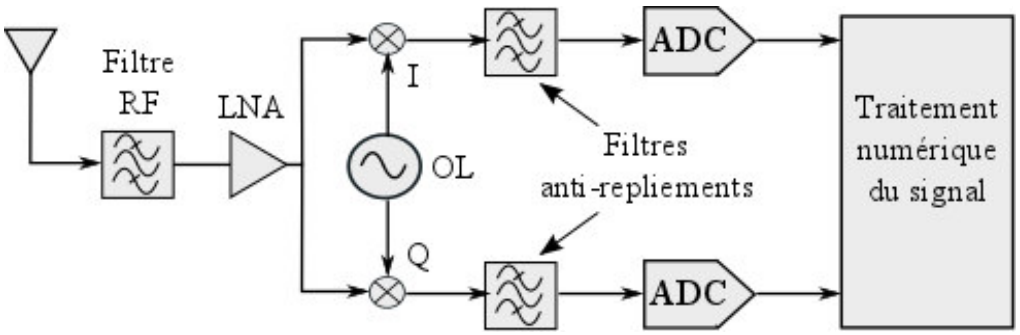


Figure 2.4 – Un récepteur homodyne

Dans cette approche, il y a peu d'étages entre le filtre RF et les ADC si ce n'est un LNA, un OL et des filtres anti-repliements (nécessaires après le mixage), ce qui facilite son intégration sur puce par rapport au récepteur hétérodyne (section 2.2.2.5) par exemple. Cependant, l'OL doit être à même de générer des signaux pour transposer tout type de porteuse, même si celle-ci est de fréquence très élevée. Ensuite, pour ce genre d'architecture, le risque qu'une portion de l'énergie issue de l'OL remonte jusqu'à l'antenne et se mélange de nouveau avec lui-même est élevé, générant un offset de tension et la saturation des étages d'amplification. La saturation se propageant jusqu'au CAN provoque la dégradation du rapport signal à bruit en réception. Il peut donc s'avérer nécessaire d'employer des algorithmes de compensation d'offset.

2.2.2.5 L'approche (super)hétérodyne

Quant à l'approche (super)hétérodyne, elle repose sur le fait que la transposition en bande de base passe d'abord par une transposition à une fréquence intermédiaire (FI). Après cette première transposition, il y a le filtrage analogique des fréquences images générées. Une fois les fréquences images supprimées (ou plutôt atténuées), on opère alors le passage de « fréquence intermédiaire » à « bande de base ». Ce passage peut lui aussi se faire par un mixage analogique – dans ce cas la numérisation du signal a lieu après mixage –, ou par un calcul numérique à l'aide d'une NCO – la

numérisation du signal intervient alors juste après la transposition à FI. La figure 2.5 illustre un récepteur super-hétérodyne avec numérisation en FI.

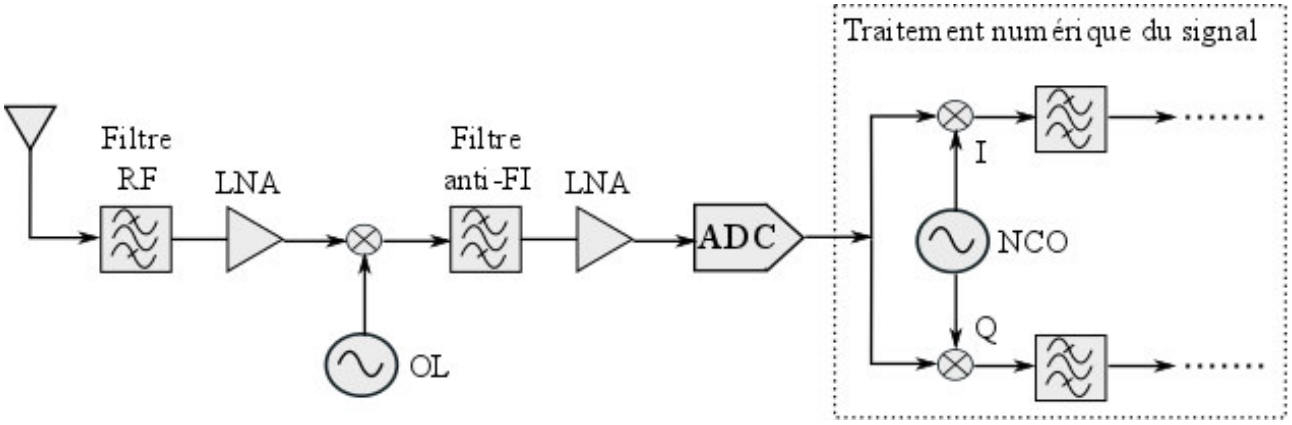


Figure 2.5 – Un récepteur super-hétérodyne avec numérisation en FI

Le récepteur superhétérodyne présente de bonnes performances en termes de sélectivité et de sensibilité. Grâce au filtrage progressif des interférences, les contraintes de linéarité du récepteur sont mieux prises en charge. Néanmoins, ce type de récepteur nécessite la réjection de ses fréquences images. On note également sa susceptibilité aux fréquences parasites proches de la fréquence intermédiaire et la nécessité d'utiliser des cellules d'adaptation d'impédance. Vu que les gabarits de filtres sont choisis en fonction de la bande à traiter, cette architecture n'est pas adaptée à un usage multi - bandes où plusieurs normes sont à prendre en considération.

On déduit de toutes les approches précédentes que, plus la partie « analogique » du système est importante, moins l'architecture générale est flexible et plus les contraintes liées à l'analogique s'intensifient (consommation, coût de fabrication plus élevé, etc.). En outre, cela réduit la charge de calcul du(des) processeur(s). Il revient donc à l'utilisateur de choisir l'architecture qui lui convient en fonction de l'application à mettre en œuvre.

Comme nous l'avons vu plus haut, la radio logicielle donne alors la possibilité de numériser le signal le plus proche possible de l'antenne. Dans cadre du projet SurfOnHertz, elle a été adoptée car elle permet d'envisager le traitement (démodulation et indexation) des radios soit au sein d'un même dispositif programmable, soit avec un assemblage de plusieurs dispositifs, tout en restant dans un contexte de système embarqué. Ces considérations permettent aisément d'écarter les architectures envisagées dans la section 2.2.2, à savoir un assemblage de plusieurs récepteurs indépendants qui alimenteraient une machine d'indexation indépendante.

2.3 SurfOnHertz : Développement d'un navigateur multimédia sur toutes les bandes commerciales radiodiffusées

2.3.1 Le projet SurfOnHertz et ses objectifs scientifiques

Le projet ANR SurfOnHertz (ARPEGE 2009), qui s'est déroulé sur la période de Janvier 2010 au 30 juin 2013, visait à réaliser un dispositif unique et novateur sous la forme d'un multi-tuner multi-standards capable d'indexer tous les canaux d'une même bande de fréquence en parallèle (figure 2.6).

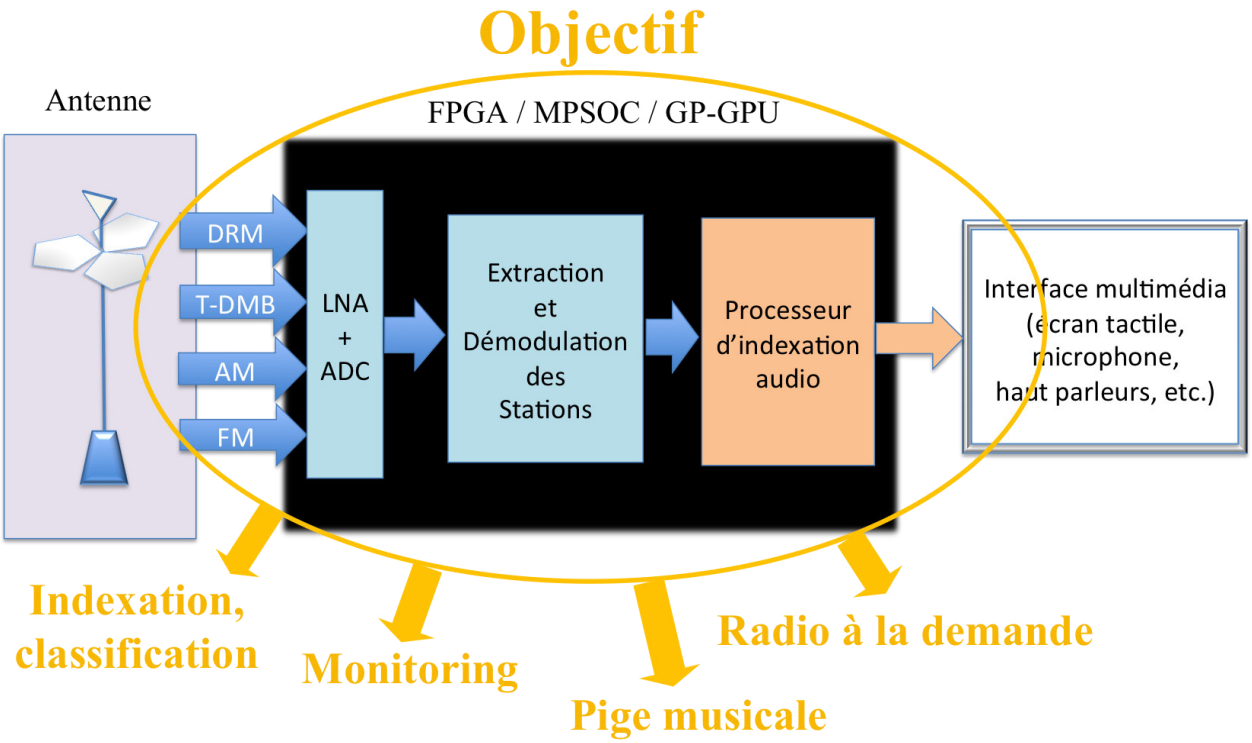


Figure 2.6 – Illustration du concept SurfOnHertz

Ce projet adressait alors quatre objectifs scientifiques et techniques :

1. **Front end** : Réaliser le front-end en identifiant les étages nécessaires à la réception des signaux diffusés sur 6 décades de 100kHz à 1.5GHz, de l'AM-GO au T-DMB en bande L, en prenant en compte des critères d'encombrement et de consommation.
2. **Récepteur parallèle intelligent** : Proposer une architecture de récepteur parallèle permettant de démoduler simultanément les 4 standards actuels (AM, FM, DRM et T-DMB). L'architecture retenue devait être réalisée sur la base d'une étude algorithmique préliminaire en prenant en compte de possibles modifications nécessaires à une exploitation future par d'autres standards (DAB, Digital Audio Broadcasting, par exemple). A ce niveau, les contraintes d'implémentation devaient être évaluées, sur la base de critères de puissance de

calcul, de débit d'entrée/sortie, de consommation et de flexibilité en matière de reconfiguration dynamique.

3. **Moteur d'indexation audio** : Identifier les algorithmes nécessaires à l'extraction et l'exploitation des programmes audio démodulés ainsi que les métadonnées associées. Les descripteurs algorithmiques devait être retenus sur les bases du scénario déterminé et des contraintes liées à une implémentation sur un système embarqué. Une estimation de celles-ci devait être établie en matière de puissance de calcul nécessaire et de taille mémoire nécessaire au stockage.
4. **Preuve de concept**: Démontrer la pertinence de l'approche retenue au moyen de la réalisation d'une preuve de concept démontant les objectifs scientifiques et techniques du projet. La démonstration des performances du système créé devait être effectuée sur quatre standards; deux analogiques (AM et FM) et deux numériques (DRM et T-DMB). Sur la base des scénarios de monitoring et de navigation, les informations extraites seront affichées sur une interface homme machine afin d'augmenter l'interaction.

2.3.2 Problèmes techniques

SurfOnHertz était un projet de recherche industriel situé au croisement des domaines de l'électronique, des radiocommunications et du traitement du signal. L'équipe du projet était composée de 6 partenaires, comprenant, 4 unités mixtes de recherches LIP6 (UMR7606), ESPCI (UMR7084), ENSEA–ETIS (UMR8051), Telecom-ParisTech-LTCI (UMR5141), ainsi que de l'Institut de Télécom (ParisTech et Sud-Paris), du CEA, et de la société Yacast (lors du lancement du projet en 2009, Yacast était, sur le plan national, un acteur majeur dans le monitoring des médias pour le compte de la SACEM).

Cependant, la réalisation de ce projet ambitieux nécessitait de lever 3 principaux verrous :

1. Le premier concerne le front end analogique en vu de l'acquisition des signaux diffusés. Les différents standards couvrent 1.5 GHz de bande, sur laquelle uniquement quelques MHz sont utiles.
2. Les contraintes d'implémentation représentent le verrou le plus difficile. En effet, la mise en place du démonstrateur nécessite d'implémenter deux blocs de traitement (réception et indexation en bande de base) complexes.
3. L'ultime verrou allait être la vérité terrain. L'expérimentation des fonctionnalités du prototype sur le terrain est incontournable afin d'en déterminer les performances obtenues en vue d'une exploitation industrielle.

2.3.3 Répartition des tâches dans le projet

Pour résoudre les problèmes soulevés, la répartition des tâches au sein de ses membres s'est inspirée d'une structure classique d'un dispositif radio logiciel, à savoir une antenne, suivi d'un front-end analogique pour amplifier/filtrer/mélanger le signal, un ADC et une plateforme numérique pour traiter les signaux numérisés. On a donc obtenu l'organisation suivante:

1. **La conception du Front End par le CEA / LIST**: L'objectif de cette tâche était de simuler, concevoir, et réaliser un front end analogique capable de transposer une bande de largeur et de fréquence centrale quelconque dans une bande de fréquence intermédiaire proche de la fréquence zéro. Dans ce projet les difficultés résidaient dans la multitude des configurations et la dispersion des paramètres largeur de bande et fréquence centrale. Par exemple en modulation d'amplitude analogique, dans la bande que l'on a coutume de nommer GO, largeur de bande et fréquence centrale sont de l'ordre de quelques centaines de kHz. A contrario en radio analogique (par exemple, la FM) ou numérique largeur de bande et fréquence centrale sont de l'ordre de plusieurs dizaines de MHz. La dispersion des paramètres touche aussi le nombre de canaux simultanément présents dans une bande de fréquence.

   Ainsi, on demandait au front end les performances suivantes :

   - Faible encombrement.
   - Faible consommation.
   - Performances en termes de facteur de bruit, d'intermodulation lors d'un traitement parallèle de multiples canaux.
2. **Les traitements radio par l'ESPCI** : Conceptuellement, l'objectif de cette tâche était de concevoir un ensemble d'algorithmes pour transformer le signal radio brut numérisé en un ensemble de flux parallèles en bande de base, chacun composé du signal audio d'une station radio et de ses méta données. Dans la pratique, comme le multi tuner de SurfOnHertz devait indexer les bandes AM, FM, DRM, et TMB-B, utilisant des modes de transmission très divers, il était prévu 4 architectures individuelles (1 architecture par standard). Devant l'ampleur de la tâche et compte tenu du temps imparti, il a été décidé de se focaliser prioritairement sur la FM et la DRM.
3. **La conception des architectures du récepteur radio par les laboratoires ETIS et Lip6** : Dans cette tâche, il fallait premièrement définir l'architecture électronique nécessaire à la démodulation simultanée des canaux radio et à la restitution des signaux en bande de base pour une post-indexation audio. Ensuite, il fallait porter les algorithmes de traitements développés dans la tâche précédente (tâche 2) sur la plate-forme de traitement retenue.

4. **L'indexation audio par l'Institut Mines-Télécom**: L'objectif de cette tâche était d'exploiter et d'adapter le savoir faire de l'Institut Mines-Télécom en indexation audio aux spécificités du projet SurfOnHertz. Il s'agissait aussi d'apporter les innovations nécessaires pour que les avancées en indexation audio réalisées durant ce projet puissent être utiles de manière générale aux applications de YACAST. Il fallait donc mettre en œuvre l'utilisation des outils ALISP (Automatic Language Independent Speech Processing), dont l'Institut-Télécom a la pérennité [Chollet1999], pour les besoins d'indexation et de pige musicale de YACAST, indépendamment de la plate-forme matérielle. Ensuite, il fallait adapter les algorithmes nécessaires pour la gamme d'application choisie pour le portage sur le dispositif embarqué, à savoir la radio à la demande et le monitoring.
5. **La conception des architectures temps-réels pour l'indexation audio par ETIS** : Le but recherché dans ce cas était l'intégration matérielle des différents blocs de traitement nécessaire aux algorithmes d'indexation, de classification et de navigation.
6. **La démonstration par ETIS**: L'équipe du projet devait faire la preuve de concept du projet en répondant d'une part aux objectifs scientifiques et techniques, et, d'autre part en levant les verrous mentionnés et rencontrés tout au long du projet. La preuve de concept devait être un démonstrateur de laboratoire de récepteur multi tuner intégrant des algorithmes d'indexation audio pour des applications de radio à la demande et de pige musicale.

   Pour y parvenir trois objectifs étaient à atteindre ici :

   - Intégrer les divers développements analogiques, numériques et algorithmiques pour la réalisation d'un démonstrateur.
   - Tester et qualifier le démonstrateur en laboratoire.
   - Tester et qualifier les performances du démonstrateur sur le terrain.

2.4 Problématique

Pour accomplir les tâches 2 et 3, les responsables à l'ESPCI et au Lip6 ont fini par arriver à la conclusion qu'il fallait un lien cordial entre ces deux tâches. Elle ne pouvait, en effet, pas évoluer l'une indépendamment de l'autre. La nécessité de cette union a poussé les deux tâches à fusionner, donnant naissance à une tâche commune. C'est pour accomplir cette nouvelle tâche, fusion des deux autres, que ma thèse est née.

La répartition des stations radios dans le spectre varie en fonction des standards. Ainsi, pour certains standards, les positions possibles des stations radios dans la bande sont uniformément espacées. C'est le cas de la FM en Europe occidentale pour qui les canaux possibles sont

régulièrement espacés de 100 kHz. Pour d'autres standards, l'espacement « entre-canaux-possibles » est si fin que les canaux semblent être choisis « n'importe où » dans la bande. L'espacement est alors non-uniforme. C'est le cas de la bande DRM30 où la granularité des stations est de l'ordre du kHz. On peut alors trouver une station située à 909 kHz et une autre station à 15.896 MHz [DRMorg].

La problématique adressée par ma thèse est qu'elle consiste à développer des procédés d'extraction et démodulation :

- De toutes les stations contenues dans une bande à distribution uniforme avec comme cas d'étude, la bande FM.
- De toutes les stations contenues dans une bande à distribution non-uniforme avec comme cas d'étude la bande DRM30.

Pour répondre à cette problématique, mon travail, a pour objectifs de :

- Développer des algorithmes pour démoduler simultanément toutes les stations contenues dans les bandes FM et DRM,
- Implémenter ces algorithmes dans une cible matérielle.
- Tester, valider et qualifier le système obtenu.
- Interfacer le système obtenu au reste du navigateur développé par les autres membres du projet.

Parmi toutes les architectures matérielles existantes, nous avons choisi de développer sur une cible de type FPGA en vue de pouvoir réaliser un SoC dans un futur proche. En effet, elles permettent de répondre aux besoins du projet SurfOnHertz en s'adaptant aux deux contextes considérés :

- Une embarquabilité du navigateur
- Une utilisation intensive pour la surveillance de l'ensemble des standards considérés

Elle offre aussi de grandes capacités en terme de parallélisme des traitements, ce qui est plus que nécessaire dans un contexte où on veut démoduler simultanément plusieurs dizaines de stations. En plus, non seulement les FPGAs offrent une maitrise de la quasi-totalité des ressources dont elles disposent, mais elles occupent en plus une surface relativement petite ce qui les rend envisageables dans un système embarqué.

2.5 Etat de l'art

La littérature révèle l'existence des différents projets ou des architectures qui se positionnent déjà comme précurseurs du projet SurfOnHertz. Pour commencer, il y a le dispositif introduit par Schatter, Zeller, et Eiselt [Schatter2008]. Son principe est illustré par la figure 2.7. Il s'agit du

concept d'une interface à reconnaissance vocale pour la recherche de contenu audio sur des flux issus d'un multiplex DAB. L'idée maitresse consiste à interfacer un récepteur DAB à une architecture matérielle de traitement comprenant un discriminateur musique/parole, un algorithme de reconnaissance de parole et de mot clé. L'interface homme machine comprend un synthétiseur vocal et un écran. Quoique présentant des similarités à certains aspects de SurfOnHertz, ce système n'adresse nullement les problèmes associés au traitement d'un très grand nombre de canaux en parallèle.

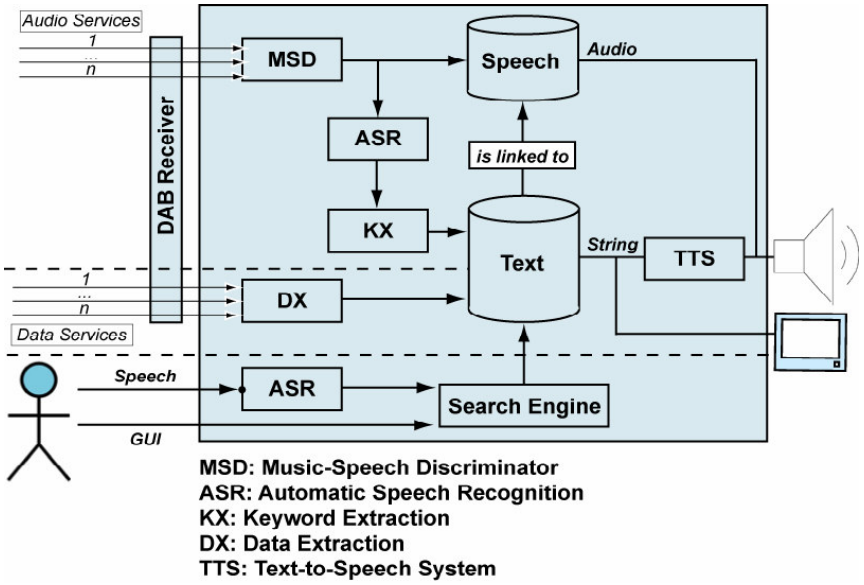


Figure 2.7 – Principe du récepteur radio de Schatter. Source : [Schatter2008]. « MSD » désigne le discriminateur musique/parole, « ASR » désigne la reconnaissance automatique de la parole. « KX » et « DX » représentent respectivement l'extraction de mots-clés et de données. « TTS » est le système de transcription de la parole et « Search Engine » désigne les constituants du moteur de recherche

Ensuite, il y a les travaux de deux des partenaires de SurfOnHertz (LIP6 et SIGMA-ESPCI) qui, ces dernières années, ont réalisé des prototypes d'un « Software Defined Broadcast Media Navigator ». Ces prototypes appliquent la technologie de la radio logicielle aux bandes commerciales de diffusion AM, FM [Denby2003A], [Denby2003B], [Denby2004], [Romain2004], [Romain2007A], [Romain2007B]. Fonctionnant en sous-échantillonnage (undersampling), ces dispositifs, à base de DSP ou de FPGA, numérisent une bande radio entière immédiatement après l'antenne, démodulent les canaux en parallèle, et appliquent un algorithme d'indexation audio. Toutefois, les fonctionnalités sont très limitées. Les circuits réalisés en FM, par exemple, ne permettent de démoduler que 5 stations de la même bande simultanément; et l'indexation audio réalisée se limite à l'application d'un simple discriminateur musique/parole sur un canal à la fois basé sur la recherche des « blancs » dans les signaux temporels démodulés. Le prototype AM quant à lui permet de démoduler simultanément les stations de la bande GO mais la faible bande passante

liée à la transmission série de la carte de prototypage (Shark ADSP2060), ne permet pas d'envoyer en temps réel à un PC distant les flux démodulés ni de les traiter localement.

Concevoir un système comme le récepteur multistandard SurfOnHertz mentionné dans la section 2.3.1 et dont les fonctionnalités sont montrées dans la figure 2.6 nécessite sans doute de le visualiser comme ayant 2 parties. La première partie est celle chargée de la réception et de la démodulation de tous les canaux des différents standards radios. La deuxième partie concerne l'indexation des canaux démodulés. Il est alors intéressant de constater que chacun de ces deux aspects fait l'objet de recherches depuis plusieurs années.

Nous avons vu que ma thèse se positionne dans la première partie. Par conséquent, passons en revue ce qui se fait dans la radio en termes de réception et de démodulation multi-stations et multi-standards.

2.5.1 Architectures de démodulation multi-canaux et multi-standards.

Différentes architectures de démodulation multi-canaux ou multi-standards ont donc vu le jour, Par exemple, Ihmig et Herkersdorf [Ihmig09] ont proposé une architecture radio logicielle basée sur FPGA. Cette architecture qui est flexible, multi-standards et multi-canaux, destinée à des applications audio et de gestion de trafic, est représentée dans la figure 2.8.

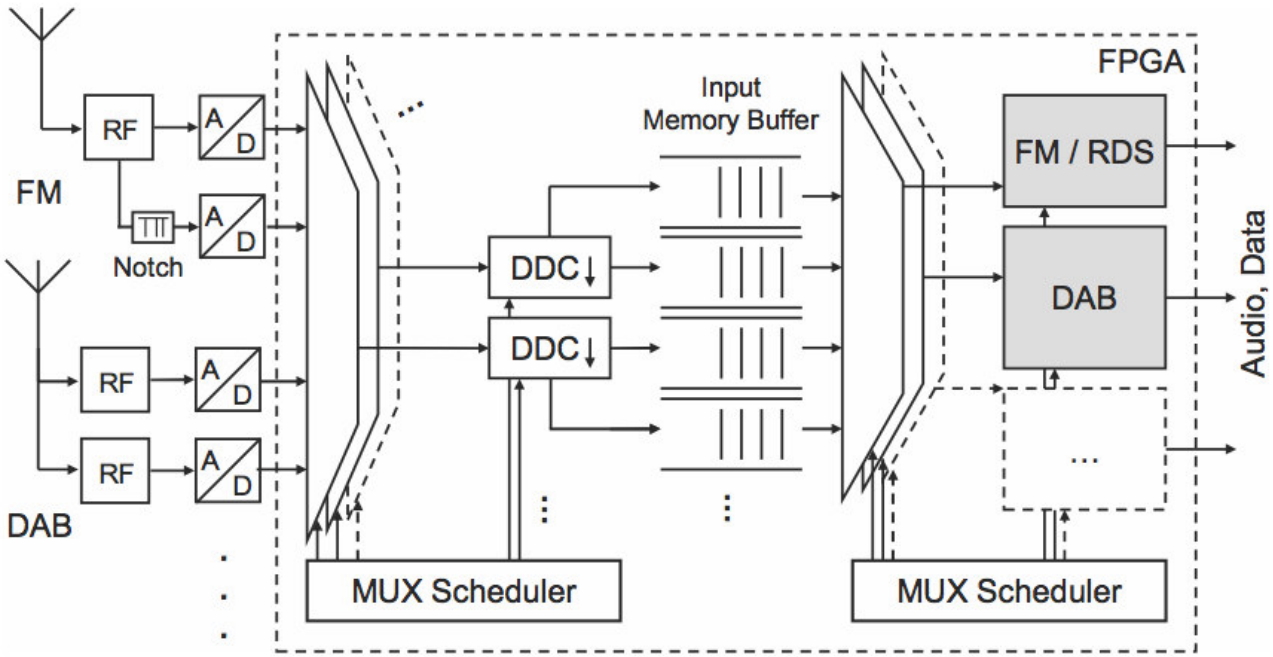


Figure 2.8 – Récepteur de Ihmig et Herkersdorf. Source : [Ihmig09]. « RF » représente un front end RF. « Notch » désigne un ensemble de filtres coupe-bandes. « MUX Scheduler » désigne un Multiplexeur/Ordonnanceur. « A/D » désigne un Convertisseur Analogique Numérique

Elle combine 2 chaînes de traitement. La première chaîne est dédiée à la bande FM et la seconde au DAB. Chaque chemin utilise 2 ADCs, un pour les stations puissantes et un pour les stations plus faibles. Chaque ADC a sa propre fréquence d'échantillonnage, et tous les traitements numériques sont prototypés sur un circuit FPGA. Les échantillons sont acheminés à deux DDCs via un multiplexeur programmable avant d'être "bufferisés". Chaque canal extrait, FM ou DAB, est bufferisé dans une mémoire de type FIFO. Le système possède seulement un processeur par standard, i.e. un pour la FM/RDS et un pour le DAB, géré par un Multiplexeur/Ordonnanceur. La performance du système est limitée par la fréquence maximale de traitement de la chaîne DDC et du nombre de DDCs pouvant être implémentés sur le FPGA. Dans [Ihmig09], un exemple d'implémentation avec 2 DDCs et une fréquence maximale de 100MHz est discuté. Le nombre de canaux pouvant être simultanément traité est, dans ce cas, limité à 2 pour la bande FM et 12 pour le DAB. La structure séquentielle présentée n'est, par conséquent, pas suffisamment performante pour traiter simultanément tous les canaux présents dans la bande.

Une autre architecture a été proposée par Sliskovic [Sliskovic08] pour la radiodiffusion. Il s'agit d'un système radio logiciel constitué de 2 voies qui débouchent sur un DSP (figure 2.9). L'étage d'entrée de chaque voie est un tuner qui convertit le signal RF à la fréquence intermédiaire 10.7 MHz. Les tuners prennent en charge les bandes AM, FM et la bande d'informations météos. Il est par conséquent possible d'y rajouter un récepteur DRM ou IBOC vu qu'il opèrera dans la même bande de fréquence que l'AM et la FM.

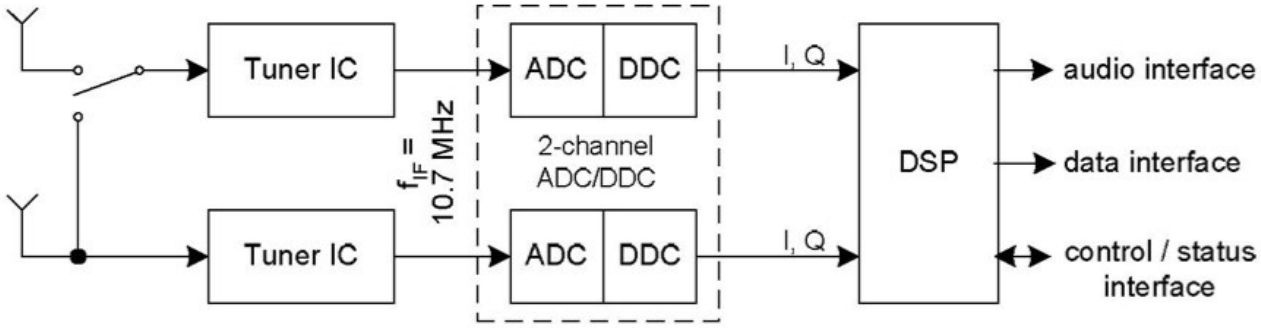


Figure 2.9 – Récepteur multi-standards de Sliskovic. Source : [Sliskovic08]. « IC » signifie circuit intégré

Les 2 voies permettent au récepteur de fonctionner en mode réception simultanée de 2 standards indépendants, ou bien en mode réception de 2 canaux d'un même standard ou encore en mode utilisation de la diversité des 2 voies pour améliorer la réception d'un canal. On ne peut donc traiter plus de 2 stations.

Il existe aussi différentes plateformes multi-standards [Luo03] pour lesquelles l'accent est d'avantage mis sur la structure logicielle plutôt que sur la partie matérielle. Le système présenté par

Luo *et al* [Luo03] est une plateforme qui, bien que faisant du multi-standards, est orientée communications et non pas radiodiffusion. La figure 2.10 en illustre le principe.

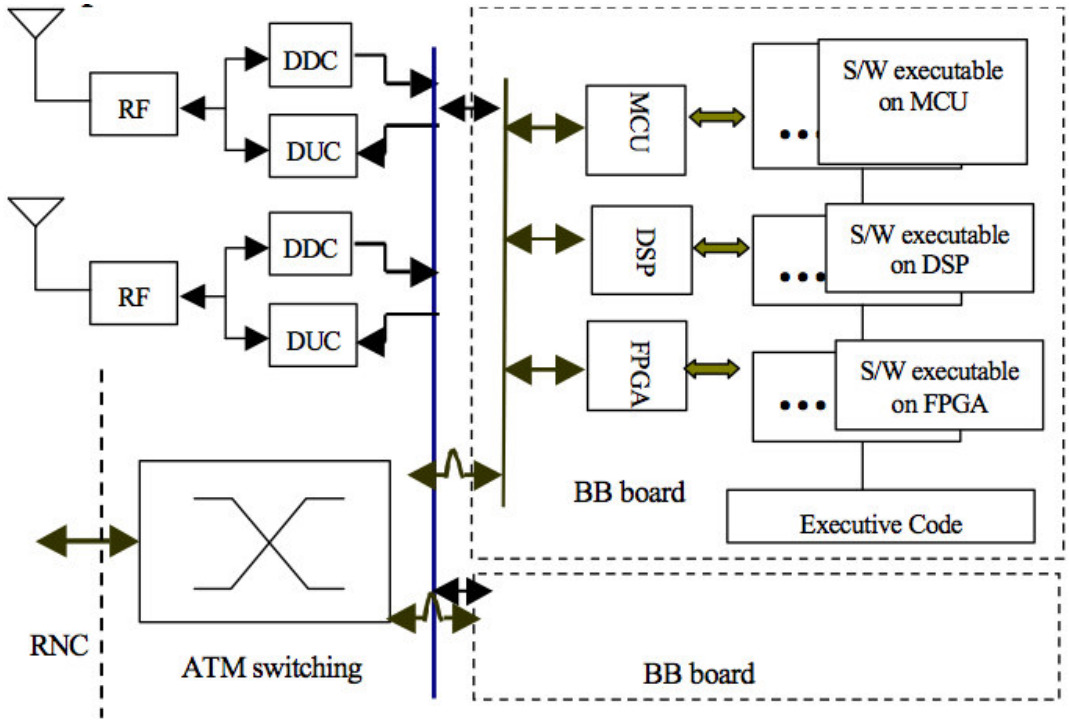


Figure 2.10 – Plateforme radio logicielle multi-standards de traitement en bande de base. Source : [Luo03]. « RF » représente un front end RF. « BB », « S/W » et « MCU » signifient respectivement « bande de base », « software » et « microcontrôleur ». « RNC » signifie littéralement Radio Network Controller tandis que « ATM » est le Automatic Transfer Mode

2.5.1.1 Architectures de démodulation logicielles

On trouve également des plateformes multi-standards qui sont essentiellement logicielles :

- **GNURadio** [GNUorg]. Le projet GNURadio qui, depuis 1998, vise à fournir aux développeurs de logiciels radio, un ensemble d'outils et de librairies libres indépendantes de tout support matériel. L'objectif de ce projet est qu'un PDA ou une plateforme nomade puisse être par exemple transformé en poste de radio pour les bandes analogiques et numériques, en poste de télévision via les normes DVB ou en lecteur RFID, uniquement en changeant le logiciel embarqué. Les bibliothèques développées sont indépendantes du matériel d'acquisition, ce qui augmente ainsi la portabilité.
- **WiNRADiO** [WINcom]: Il s'agit d'un ensemble de logiciels de réception, d'enregistrement et de stockage des radios AM, FM ou DRM. Ils accompagnent souvent des dispositifs RF externes (antennes, récepteurs etc.).
- **DREAM** [DREAMnet]: C'est un logiciel qui peut être utilisé en tant que récepteur AM, FM ou DRM et qui est disponible sous licence GNU – GPL (General Public License).

- **SoDiRa** [SoDiRa]: Il s'agit d'un logiciel capable de démoduler plusieurs standards, notamment le AM, FM, DRM, ainsi que le AMSS et le RDS.

Du côté des applications, il y a le concept « SDR Touch » qui vise à changer les smartphones ou les tablettes en tuner radio logiciel multi-standards [SDRTouch] grâce à une application qu'on installe sur l'appareil mobile. Le Smartphone est alors accompagné d'un dongle (une sorte de clé USB) en guise de front end à qui il est connecté par un câble USB OTG (On-The-Go) (figure 2.11). Avec ce système, l'utilisateur peut par exemple écouter les stations radio FM, les prévisions météos, les communications de la police, des pompiers ou des urgences. Il peut aussi recevoir certaines diffusions numériques et écouter des communications concernant les avions, la circulation des taxis, les bandes sons des programmes TV. Le système est fait pour l'écoute d'une station à la fois.

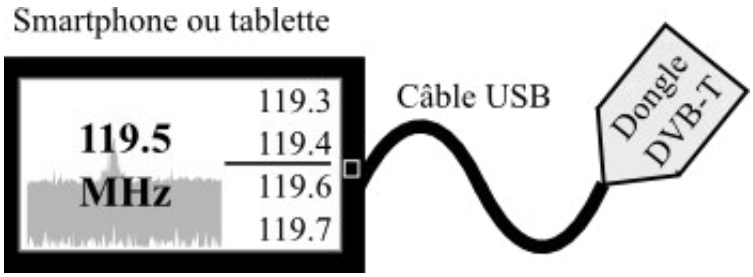


Figure 2.11 – Concept « SDR Touch »

Similairement au « SDR Touch », l'application « Wavesink » [Wavesink] permet quant à elle de démoduler la FM/RDS, le DAB et la DRM+. Son principe est le même que pour le « SDR Touch » à savoir que c'est une application fonctionnant sur Smartphone ou tablette accompagné d'un câble USB OTG et d'un dongle (incluant une puce RTL2832U).

Parmi les structures proposées qui peuvent extraire ou bien démoduler simultanément plus de 20 canaux, il y en a 2 qui sortent du lot compte tenu du nombre élevé de canaux qu'elles peuvent démoduler. La première est un dispositif commercial, la CDRX-3200 de FlexRadio [FlexRadio], capable d'extraire jusqu'à 32 canaux entre 100 kHz et 100 MHz simultanément (figure 2.12). C'est en fait un assemblage de 32 récepteurs qui possède chacun une bande passante allant jusqu'à 200 kHz, et qui peuvent être réglés de façon indépendante ou synchrone. Cependant, ce dispositif ne démodule pas les signaux FM, ce qui implique la nécessité d'un dispositif en plus pour que le système soit complet. La seconde structure est un récepteur FM basé sur un FPGA [Romain2007B] et qui capture 32 canaux et en démodule 5 simultanément (figure 2.13). Les canaux doivent néanmoins se situer à des fréquences fixes précises, si bien que le récepteur peut ne pas coïncider avec la distribution des stations désirées.

Figure 2.12 – Image du CDRX-3200 de FlexRadio. Source: [FlexRadio]

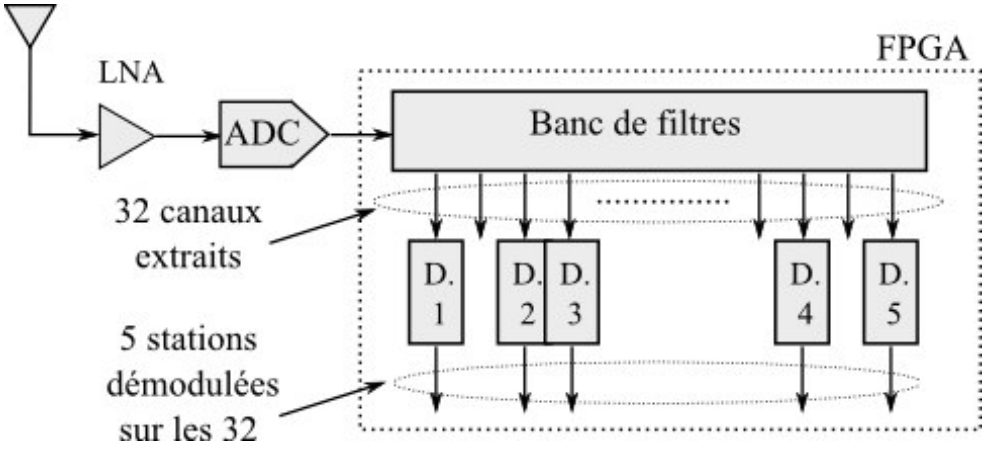


Figure 2.13 – Récepteur FM de Romain et Denby basé sur un FPGA. « D. i» indique la i^{ème} instance du module de démodulation

2.5.1.2 Solutions industrielles

Le monde industriel apporte lui aussi des solutions intéressantes. Il y a tout d'abord, le module RS500 de la société Radioscape [RS50006], qui est un système numérique permettant la réception de plusieurs standards, AM analogique, FM analogique incluant le RDS, la DRM et le DAB. C'est une architecture radio logicielle basée sur l'utilisation d'un front-end analogique, d'un ADC et d'un DSP Texas Instrument (le TMS320DRM350, capable de démoduler la DRM et la DAB [TIcom]). La démodulation est mono-canal et réalisée par programmation dynamique du DSP. Le module RS500 (figure 2.14), est le premier module à avoir été commercialisé et exploité dans des récepteurs grands publics [Sabel2007] (Morphy Richards 27094) pour la France, le Portugal, l'Espagne, la Hollande, l'Allemagne, la Belgique et l'Angleterre. Ce module ne permet pas d'exploiter tous les standards en même temps. D'autre part, il n'est plus commercialisé aujourd'hui.

Figure 2.14 – Module RS500 de la société Radioscape. Source : [*RS50006*]

La société NXP quant à elle produit des puces multi-standards, les SAF3560 [SAF3560] et SAF3561 [SAF3561]. La première est capable de prendre en charge la démodulation de la AM, la FM ou bien la HD Radio pour en extraire l'audio et les données numériques. La deuxième quant à elle prend en charge le DAB, le DAB+ et le T-DMB. Ces puces ne sont cependant pas conçues pour faire de la démodulation de plusieurs canaux simultanément. Elles ne permettent pas de répondre aux problèmes levés par le projet SurfOnHertz.

Le système MS-8118/BRL commercialisé par WiNRADiO est également très intéressant [WINcom2]. C'est un système d'enregistrement et de stockage des bandes AM, FM ou même DRM. Basé sur un ordinateur industriel monté sur un rack 19 pouces, il peut contrôler jusqu'à 8 stations radio simultanément et enregistrer le signal audio sur un disque dur en temps réel pour un post-traitement.

Quant à la société Etherwaves [ETHERcom], développe l'ensemble ClearSignal-SoC^{TM} qui est un ensemble d'IP matérielles et logicielle permettant d'ajouter une capacité multi-standard (DAB, DAB+, T-DMB, DRM et HD Radio) à un système sur puce en cours de conception. La figure 2.15 montre un exemple d'utilisation de cette solution.

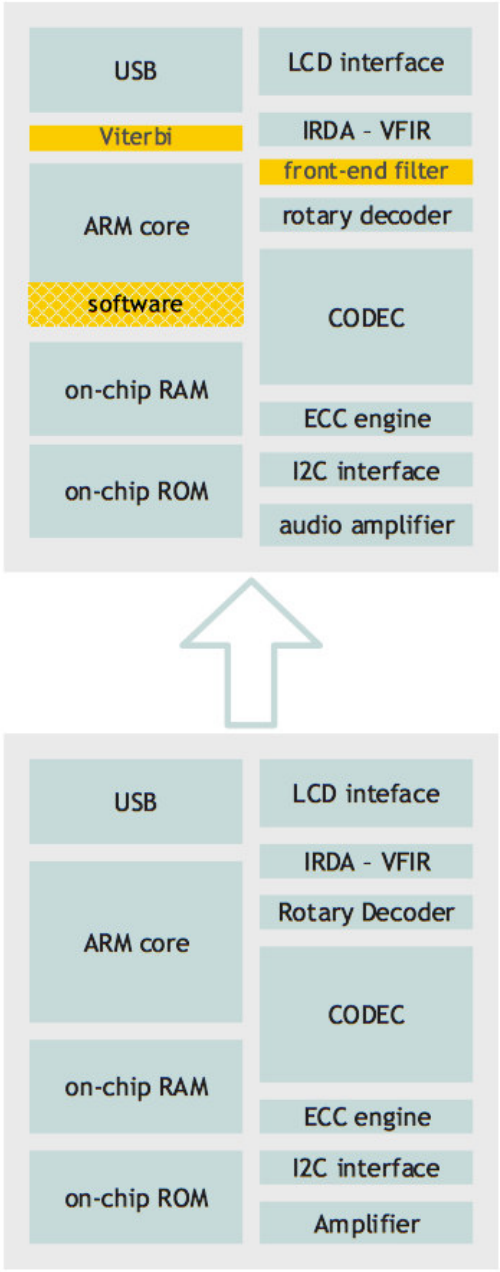


Figure 2.15 – Exemple d'utilisation de l'ensemble logiciel ClearSignal-SoC. Source : [ETHERcom02]. A gauche il y a le SoC original du client. A droite c'est le SoC du client auquel ont été rajoutées les fonctions de récepteur numérique.

2.5.2 Bilan comparatif et conclusion

Pour résumer les différentes méthodes mentionnées ci-dessus avec leurs caractéristiques, un bilan comparatif est dressé dans le tableau 2.5.

	Multi-standards ?	Multi-canaux ?	Toutes les stations ?	Combien de canaux simultanément ?	Standards Concernés
Ihmig et Herkersdorf (2009)	Oui	Oui	Non	14 (FM : 2, DAB : 12)	FM, DAB
Sliskovic (2008)	Oui	Oui	Non	2	AM, FM, DRM, IBOC
Luo *et al* (2008)	Oui	Plateforme orientée communications et non radio diffusion			
GNURadio	Oui	Non	Sans objet		AM, FM, DRM, DAB
WiNRADiO	Oui	Oui	Non	8 (MS-8118/BRL)	AM, FM, DRM
Dream	Oui	Non	Sans objet		
SoDiRa	Oui				AM, AMSS, FM/RDS, DRM
SDRTouch	Oui				AM, FM, etc.
Wavesink	Oui				FM/RDS, DAB, DRM+
CDRX-3200*	Oui	Oui	Non	32	FM, FM HD
Romain et Denby (2007)	Non	Oui	Non	5	FM
RS500	Oui	Non	Sans objet		AM, FM (RDS), DRM, DAB
SAF3560	Oui	Non			AM, FM, HD Radio
SAF3561	Oui	Non			DAB, DAB+ et T-DMB.
MS-8118/BRL	Oui	Oui	Non	8	AM, FM, DRM
ClearSignal-SoC™	Oui	Non	Sans objet		DAB, DAB+, T-DMB, DRM, HD Radio

* Le dispositif CDRX-3200 de FlexRadio capture et extrait les canaux sans les démoduler, contrairement aux autres systèmes repertories dans le tableau.

Tableau 2.5 – Récapitulatif de l'état de l'art sur les récepteurs multi-canaux et multi-standards

De toutes les architectures présentées dans cet état de l'art, il n'en est aucune qui soit à même d'extraire et de démoduler toutes les stations d'un standard radio en particulier et encore moins de plusieurs standards simultanément. Cela rend légitime la problématique de cette thèse expliquée

plus haut (section 2.4). Répondre à cette problématique revient finalement à proposer un système qui soit multi-standards (FM et DRM), et qui démodule tous les canaux de chacun des standards.

2.6 Bibliographie

[Chollet1999] G. Chollet, J. Cernocky, A. Constantinescu, S. Deligne, and F. Bimbot. Towards ASLIP: a proposal for Automatic Language Independent Speech Processing. In Keith Ponting, editor, 1259 NATO ASI: Computational models of speech pattern processing Springer Verlag, 1999.

[Denby2003A] B. Denby et S.A. Hariti, Towards a Software Radio Enabled Broadcast Media Navigator, EC-VIP-MC2003, Zagreb, Croatie, juillet 2003.

[Denby2003B] B. Denby, C. Kiesling, J.-C. Prévotet, P. Garda, B. Granado, A. Wassatch, Fast Triggering in High Energy Physics Experiments Using Hardware Neural Networks, IEEE Transactions on Neural Networks, Vol. 14 (2003), pp. 1010 – 1027.

[Denby2004] B. Denby, O. Romain, S.-A. Hariti, A Software Radio Approach to Commercial FM Content Indexing actes du 11th International Workshop on Systems, Signals and Image Processing, IWSSIP'04 Ambient Multimedia, September 13-15, 2004, Poznań, Poland.

[DREAMnet] <http://sourceforge.net/apps/mediawiki/drm/>. Dernier accès le 16 Août 2013.

[DRMorg] Site web du Digital Radio Mondiale. <http://www.drm.org>. Dernier accès : 15 Août 2013.

[ETHERcom] <www.etherwaves.com>. Dernier accès le 17 Août 2013.

[ETHERcom02] EtherWaves, « ClearSignal SoC Product Description ». <http://www.etherwaves.com/downloads/ClearSignal-SoC.pdf>. Dernier accès le 17 Août 2013.

[FlexRadio] FlexRadio Systems, 4616 W. Howard Lane, Austin, TX USA 78728. <http://www.flex-radio.com/Products.aspx?topic=CDRX-3200>. Accessed 26th July 2013.

[GNUorg] <http://www.gnuradio.org>. Dernier accès le 16 Août 2013.

[Ihmig09] M. Ihmig, and A. Herkersdorf, "Flexible multi-standard multi-channel system architecture for Software Defined Radio receiver," in *9th IEEE Intelligent Transport Systems Telecommunications, Int. Conf, ITST.*, Lille, France, Oct. 20 – 22, 2009, pp. 598-603.

[Ken05] KENINGTON, P. B.. RF and Baseband Techniques for Software Defined Radio.

Éditions Artech House, 2005, 340 pages. ISBN: 1580537936.

[Lackey95] R. L. Lackey and D. W. Upmal, "SPEAKeasy : The military software radio," IEEE Communications Magazine, vol. 33, pp. 51-61, May 1995.

[Lak09] LAKYS, Y., " Filtres à Fréquence Agile Totalement Actifs : Théorie Générale et Circuits de Validation en Technologie SiGe BiCMOS 0.25µm", Thèse de doctorat, Université Bordeaux 1, France, 2009.

[Luo03] Z. Luo, W. Li, Y. Zhang, and W. Guan, "A multi-standard SDR base band platform," in *IEEE Int. Conf. Computer Networks and Mobile Computing, ICCNMC 2003*, Shanghai, China, Oct. 20 – 23, 2003, pp. 461- 464.

[Romain2007A] O. Romain, B. Denby et J. Denoulet, « Récepteur Haute fréquence à traitement numérique multi-canaux », brevet (patent) : EP2145387 A2. Date de publication : 20 janvier 2010. Déposant : Université Pierre et Marie Curie – Paris 6.

[Romain2007B] O. Romain, and B. Denby, "Prototype of a Software Defined Broadcast Media Indexing Engine," in *IEEE Int. Conf. Acoustics, Speech and Signal Processing, ICASSP 2007*, Honolulu, Hawaii USA, Apr. 15 – 20, 2007, pp.II-813-II-816.

[RS50006] RadioScape, « RS500 Multi-Standard Digital Radio Module For DRM, DAB, AM & FM, » application datasheet, 2006.

[Sabel2007] L. Sabel, « Software-Defined Radio – the solution for multi-standard multimédia in the mobile environnement », EBU Technical Review, January 2007, <http://tech.ebu.ch/docs/techreview/trev_309-radioscape.pdf>. Accessed 18th Nov. 2013.

[SAF3560] NXP Semiconductors, "SAF3560 terrestrial digital radio processor," Product short data sheet, rev. 5 – 8 February 2013.

[SAF3561] SAF3561 Terrestrial Digital Radio Processor <http://www.nxp.com/products/audio_radio/digital_radio_processors/series/SAF3561.html>. Dernier accès le 17 Août 2013.

[Schatter2008] G. Schatter, B. Zeller, A. Eiselt, Navigating Spoken Broadcast Content of a Digital Radio DAB by Music-Speech Discrimination and Information Retrieval, 9th Digital Broadcasting Workshop, 18 et 19 septembre 2008, Erlangen, Allemagne.

[SDRTouch] <http://sdr.martinmarinov.info/>. Dernier accès le 24 Sept. 2013.

[Sliskovic08] M. Sliskovic, "Software Defined Automotive Receiver for Broadcasting Services," in *IEEE Int. Conf. Consumer Electronics, ICCE 2008. Digest of Technical Papers*, Las Vegas, NV, Jan 9 – 13, 2008, pp.1-2.

[SoDiRa] <http://www.dsp4swls.de/sodira/sodiraeng.html>. Dernier accès le 06 Oct. 2013

[TIcom] Texas Instruments, « TMS320DRM300/350 Digital Radio Mondiale Solution, »

Product Bulletin, 2005. <http://www.ti.com/lit/ml/sprt354/sprt354.pdf>. Dernier accès le 09 Oct. 2013

[Wavesink] <http://www.feilen-stolz.de/receiver.php>. Dernier accès le 06 Oct. 2013

[WINcom] <http://www.winradio.com/index.htm>. Dernier accès le 23 Sept. 2013.

[WINcom2] <http://www.winradio.com/home/ms8118-brl.htm>. Dernier accès le 18 Août 2013.

[Yos03] Yoshida, H., Kato, T., Tomizawa, T., Otaka, S., Tsurumi, H., "Multi-mode Software Defined Radio Receiver using Direct Conversion and Low-IF Principle: Implementation and Evaluation", Electro. Commun. in Japan, Vol. 86, 2003, pp. 55-65.

3 Méthodologie de channelization et de démodulation simultanée de l'intégralité de bandes de radiodiffusion

3.1 Architecture générique

Nous avons pu voir précédemment (section 2.1) qu'il existe des radios terrestres (AM, FM, DRM, T-DMB etc.) et des radios satellitaires (Astra, 1WorldSpace etc.). Tout le long de ce travail, nous allons nous intéresser uniquement aux radios terrestres. Ce choix est appuyé par le fait que plusieurs standards de radios satellitaires sont des systèmes propriétaires et donc non ouvertement accessibles.

Rappelons que ce qui est présenté dans la suite, à savoir ma contribution par rapport à l'état de l'art, est de développer une méthode radio logicielle, autour d'un FPGA, pour extraire et démoduler :

- Toutes les stations contenues dans une bande à distribution uniforme. Nous allons pour cela nous focaliser uniquement sur la bande FM.
- Toutes les stations contenues dans une bande à distribution non-uniforme. Nous allons pour cela nous focaliser sur la bande DRM30.

Pour ce faire, la solution retenue est illustrée de façon générale par le schéma suivant (figure 3.1).

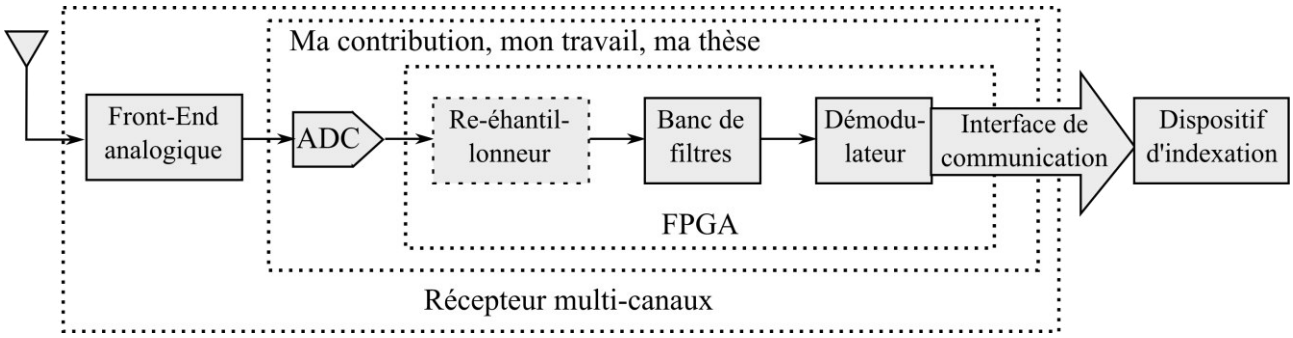


Figure 3.1 – Architecture d'extraction et de démodulation de toutes les stations

Dans cette architecture,

- le *Front-End* filtre et amplifie la bande cible, à savoir la bande DRM30 ou la bande FM dans notre cas précis. Il convient de rappeler que le front-end est réalisé par le partenaire CEA du projet SurfOnHertz.
- L'*ADC* joue un rôle déterminant. En effet, si la bande passante du standard est à zéro-FI ou proche de zéro-FI (ce qui est souvent le cas des standards avec distribution non-uniforme

comme la AM ou la DRM30), il numérise le signal à une fréquence d'échantillonnage suffisamment élevée pour respecter le critère de Nyquist. Si la bande passante se situe à fréquence intermédiaire (ce qui est les cas de certains standards avec distribution uniforme comme la bande FM), l'ADC fait un sous-échantillonnage (undersampling) tout en respectant le critère de Nyquist.

- Juste après échantillonnage, il peut s'avérer nécessaire de changer la fréquence d'échantillonnage du signal acquis. Le *re-échantillonneur* est implémenté pour cela.
- Le *Banc de filtres* : c'est le module qui extrait tous les canaux utiles de la bande et qui les ramène soit à la bande de base, soit proche de la bande de base selon les caractéristiques dont le module suivant a besoin pour la démoduler.
- Le *Démodulateur* : Il démodule les canaux pour en extraire l'audio.
- L'*Interface de communication* : Il achemine les données audio démodulées au dispositif d'indexation.
- Le *Dispositif d'indexation* : C'est lui qui fait l'indexation sur les stations démodulées.

Dans cette architecture, il apparaît clairement qu'aucune transposition de fréquence analogique n'est faite par le front-end. Il s'agit alors, dans le cas de la bande FM, d'une architecture radio logicielle par numérisation en RF avec sous-échantillonnage (undersampling, voir section 2.2.2.3). Dans le cas de la bande DRM30, l'architecture est proche de l'approche idéale (voir section 2.2.2.1). Sans plus tarder, nous allons examiner les différentes parties de la méthode proposée.

Dans la section 3 qui débute, il sera question des concepts et algorithmes utilisés pour les blocs mentionnés plus haut (voir figure 3.1) à l'exception du bloc « re-échantillonneur ». Ce dernier fera l'objet d'une section à part. Les détails sur les réalisations matérielles des autres blocs seront présentés dans la section 5.

3.2 Front-End analogique et ADC

3.2.1 Front-End analogique

Notre récepteur est directement alimenté par l'antenne. Le front-end s'occupe du filtrage et de l'amplification du signal reçu, ce qui constitue une étape nécessaire pour produire un signal correct aux autres parties du circuit. Plus précisément, le front-end analogique doit:

- Sélectionner la bande de fréquence appropriée. Par conséquent, le front-end fait premièrement fonction de filtrage, dont la fonction est d'optimiser le SNR.
- Amplifier le signal reçu de sorte qu'il soit compatible avec la plage d'entrée de l'ADC. On a besoin d'amplificateurs pour obtenir un bruit faible, un gain élevé et un point d'interception

élevé des produits d'intermodulation du 3ième ordre (IP3). A partir de la formule de Friis pour le facteur de bruit F d'une chaîne d'éléments en série, on a (3.1):

$$F = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + ..., \tag{3.1}$$

où F_1, F_2, … représente les facteurs de bruit des étages suivants et G_1, G_2, … leur gain, nous rappelant que, généralement, c'est l'étage d'entrée qui joue le rôle dominant.

Le front-end analogique ayant été réalisé par une « personne » extérieure à ma thèse, seuls les points principaux de l'architecture du front end seront reportés dans le présent document.

Pour avoir une idée des besoins du récepteur décrit ici, nous pouvons faire un simple calcul. Un nombre très répandu d'ADCs supporte en entrée un signal dont l'amplitude est limitée à 2 volts crête-crête. Un niveau de -107 dBm, en revanche, correspond à une tension effective de 1 µV sur une impédance de 50 ohms. L'amplification nécessaire sera alors de l'ordre de 100 dB, ce qui sera suffisant pour amplifier un signal de 10 µV (valeur typique pour un signal radio exploitable) à 1 V. Comme mentionné dans [TRDD12], le gain maximal d'une telle chaîne ne doit pas dépasser 60 dB. Cette considération limite alors les performances du front-end et donc celles du récepteur entier. De plus, dans notre système radio logiciel où plusieurs porteuses sont traitées en parallèle, les étages d'amplification doivent être extrêmement linéaires pour éviter l'intermodulation entre porteuses.

3.2.1.1 FM

Pour le récepteur FM, un certain nombre de configurations ont été simulées sous l'environnement Genesys d'Agilent avant la phase de prototypage, afin de parvenir au design final décrit ci-après, dont les paramètres sont les suivants : Gain = 60 dB ; F = 2.6 dB ; IP3 = 39 dBm.

La bande passante du front end est la bande 88 – 108 MHz.

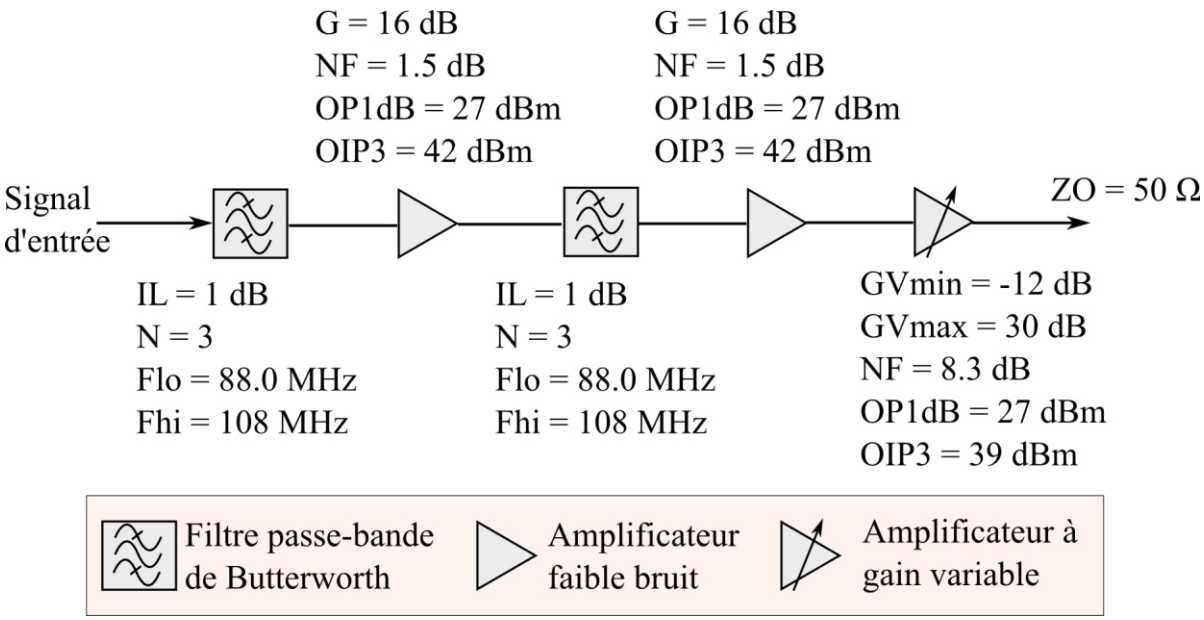


Figure 3.2 – Schéma global du front analogique développé pour la bande FM. IL est l'incertitude sur le gain dans la bande passante du filtre, N est l'ordre, Flo et Fhi sont respectivement les fréquences de coupures inférieures et supérieures. G est le gain, NF le facteur de bruit, OP1dB est le point de compression à 1 dB et OIP3 est l'IP3. GVmin et GVmax sont respectivement le gain minimal et maximal qu'on peut régler avec l'amplificateur à gain variable. ZO est l'impédande de sortie du front-end

3.2.1.2 DRM

Un deuxième front-end a été prototypé pour les bandes LHF et HF. Dans ce front-end, on coupe le signal d'entrée en deux sous-bandes. La première est issue du filtrage et de l'amplification des fréquences Grandes Ondes (GO) et Moyennes Ondes (MO) qui vont de 150 kHz à 1,6 MHz. La seconde est issue du filtrage et de l'amplification de la bande 2,3 – 30 MHz. La figure 3.3 schématise ce front-end.

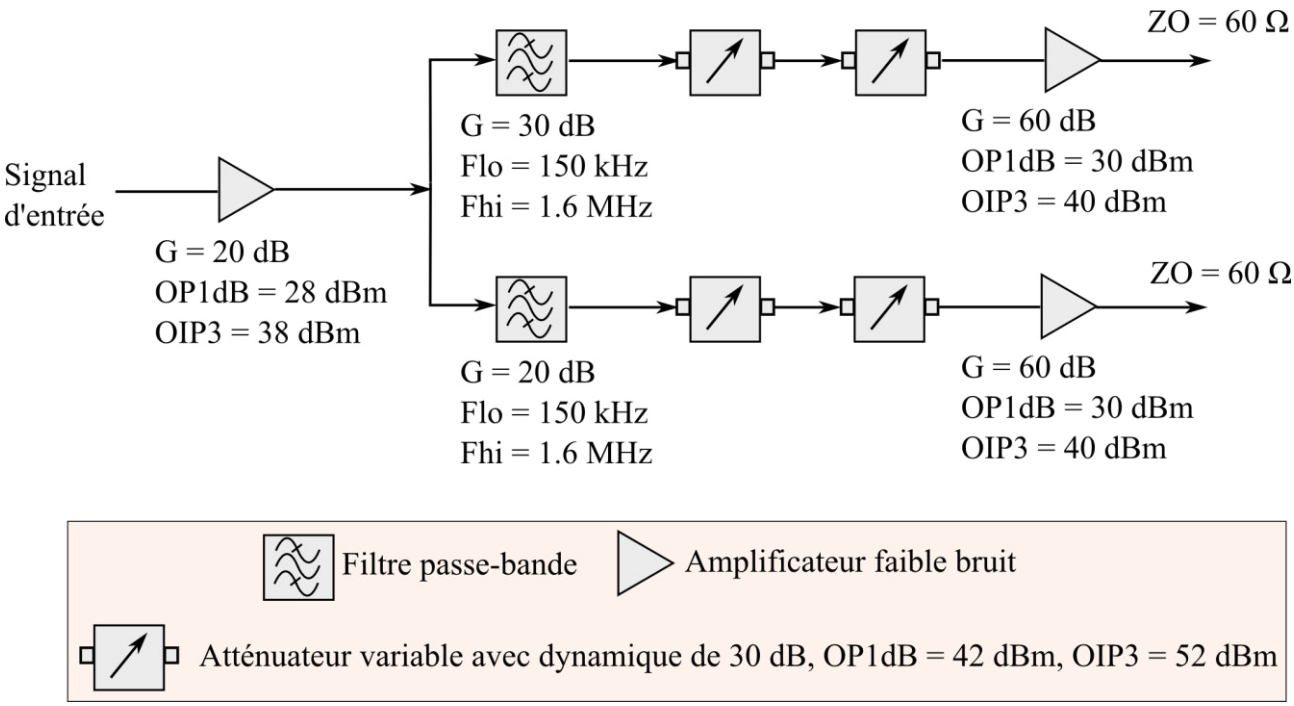


Figure 3.3 – Schéma synoptique du front-end AM – DRM30. Flo et Fhi sont respectivement les fréquences de coupures inférieures et supérieures. G est le gain, OP1dB est le point de compression à 1 dB et OIP3 est l'IP3. ZO est l'impédande de sortie du front-end

Les éléments du synoptique sont caractérisés par leur gain et leur point d'intermodulation. Ce montage permet de disposer d'un gain maximal de 110dB dans la branche GO+MO et 100 dB dans l'autre branche. On a aussi un IP3 de 40dBm. Au final, nous avons donc deux front-end : un pour la bande FM, et un autre pour la bande AM-DRM.

3.2.2 ADC

Comme on souhaite numériser chaque bande radio dans leur intégralité, la puissance variable des signaux reçus des stations radios doit être prise en compte. Dans le cadre de cette thèse, nous allons rester dans le cas de figure où la différence maximum entre les stations puissantes et les stations faibles est de 55dB. Pour s'assurer que les stations faibles ne disparaissent pas sous le seuil du bruit de quantification, et en ajoutant une marge de bruit de 15 dB pour démoduler de façon satisfaisante les signaux radios, on peut choisir un ADC de résolution 12-bits. Le seuil du bruit de quantification d'un échantillon codé sur 12 bits est donné par (3.2):

$$20 \log_{10}(2^{12}) \approx 72 \tag{3.2}$$

Par conséquent, comme 72 dépasse (55+15), un ADC 12-bits fournit un bruit de quantification qui remplit le précédent critère.

Maintenant que la précision de l'ADC est choisie, le prochain paramètre à régler est la fréquence d'échantillonnage. Nous avons vu plus haut que l'architecture adoptée pour traiter la bande FM est une architecture radio logicielle par numérisation en RF avec sous-échantillonnage (undersampling, voir section 2.2.2.3). Intéressons-nous à la bande FM en Europe occidentale qui va de 87.5 à 108 MHz. La largeur de cette bande étant 20.5 MHz, la fréquence d'échantillonnage doit être supérieure à 41 MHz (Nyquist). Aussi, on la fixe à 87 MHz afin :

- de transposer le signal à la bande de base par sous-échantillonnage (undersampling).
- de s'assurer que les stations demeurent dans le même ordre qu'avant la numérisation du signal.

Dans le cas de la bande DRM30, l'architecture est proche de l'approche radio logicielle idéale (voir section 2.2.2.1). Il suffit de choisir une fréquence supérieure ou égale à 2×30 MHz. Nous avons choisi la fréquence 81.92 Mhz car elle permet d'implémenter un banc de filtres efficace pour la bande DRM30 (voir section 3.4.3 pour plus de détails).

Les échantillons obtenus dans les 2 cas précédents sont ensuite envoyés dans le FPGA pour la suite des traitements.

3.3 Un procédé de re-échantillonnage optimisé

Une fois dans le FPGA, selon le standard, le signal numérisé peut être d'abord envoyé au module re-échantillonneur. C'est le cas pour le standard FM et vraisemblablement le cas pour les autres standards numérisés par sous-échantillonnage (undersampling). En effet, au cours de développement de notre architecture, il est apparu le besoin de procéder au re-échantillonnage du signal numérisé afin que le banc de filtre développé pour la FM puisse extraire tous les canaux de la bande (voir section 3.4.2 pour plus de détails). Le re-échantillonnage consiste à changer la fréquence d'échantillonnage d'un signal numérique donné (figure 3.4). Une partie de ma thèse a donc consisté à développer un algorithme de re-échantillonnage, facilement implémentable sur FPGAs. La section 4 présente en détails cet algorithme ainsi que ses implémentations matérielles. Nous avons voulu faire de cette méthode de re-échantillonnage un outil pratique de conception utilisable aussi dans des contextes de radio logicielle autres que SurfOnHertz. Partant du constat que dans la plupart des cas la conversion de fréquence requise n'est pas à facteur entier, nous avons caractérisé cette méthode « à ratio arbitraire », c-a-d, à même de prendre en charge des cas où le facteur de conversion n'est pas entier.

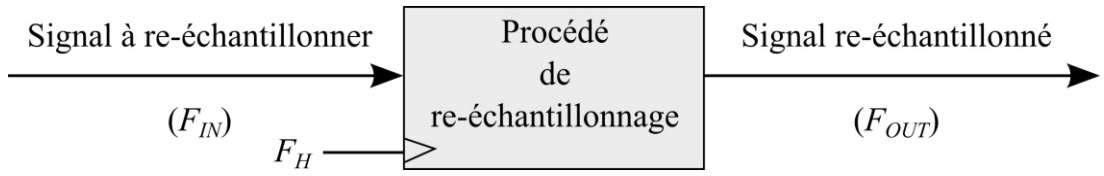


Figure 3.4 – Contexte du re-échantillonnage. F_{IN} est le fréquence d'échantillonnage du signal au départ. F_{OUT} est le fréquence d'échantillonnage du signal désirée F_H représente la fréquence d'horloge sur laquelle est cadencé le re-échantillonneur

Le re-échantillonneur sera intégré au récepteur FM mais pas dans le récepteur DRM30. Ce dernier n'en aura pas besoin. Dans le récepteur FM, il effectuera un re-échantillonnage de 87 MHz à 51.2 MHz. La fréquence 51.2 MHz permettra au bloc « bancs de filtres » suivant d'extraire tous les canaux existants de la bande FM et de les ramener à la bande de base.

La capacité d'extraire toutes les stations en parallèle et les démoduler ensuite est une capacité incontournable pour les récepteurs DRM30 et FM qui sont développés dans le cadre de cette thèse. Il faut alors lever l'obstacle de l'extraction simultanée des canaux – dans la suite du présent document, nous appellerons cette extraction « channelization », comme en anglais – et de leur démodulation simultanée. La position et la largeur des canaux dans les 2 bandes étant différentes, de même que le type de modulation (numérique pour la DRM30 et analogique pour la FM), conduisent à développer une méthodologie particulière et adaptée pour chaque bande.

La section suivante (section 3.4) présente un système novateur de channelization de l'intégralité des bandes radios visées. Une partie de ma thèse a été consacrée à son développement. Les problèmes auquel il répond qu'il est parfaitement adapté à une implémentation sur FPGA d'une part, et que d'autre part il utilise le minimum possible de ressources en calcul afin d'augmenter son embarquabilité.

La section d'après (section 3.5) présente ce que nous proposons pour faire leur démodulation.

3.4 Méthodologie de channelization simultanée de l'intégralité des bandes DRM30 et FM

3.4.1 Etat de l'art.

La « channelization » est le procédé qui permet d'extraire plusieurs canaux de la bande d'un signal traité. Comme nous nous intéressons à la radio, nous souhaitons donc que la channelization soit faite avec une phase linéaire afin de minimiser l'impact négatif que pourrait avoir ce procédé sur la démodulation de chaque station. De même, vu que les canaux d'un standard radio ont des largeurs égales ou de même ordre de grandeur (voir tableau 2.2), le travail présenté ici s'attaquera à une channelization avec canaux de largeur égale. On appele « channelizeur » un système qui fait la

channelization.

Diverses méthodes, applicables aux FPGAs, permettent de faire de la channelization d'un signal en utilisant un filtrage à phase linéaire avec largeur de bande égale. On peut classer ces méthodes en 3 catégories, selon la gestion des espacements inter-canaux que permettent ces techniques :

- **Channelizeurs avec espacements uniformes**. Ici, le channelizeur peut soit fournir tous les canaux uniformément espacés comme les techniques Weighted Overlap Add (WOLA) [L03][H02][WLW06], Pipelined FFT (PFFT) [L03]; Polyphase DFT filterbanks (PDFT) [L03][H02][HDR03][WLW06], soit permettre de sélectionner les canaux à extraire parmi un ensemble de canaux possibles uniformément espacés. Il s'agit par exemple de la technique Pipelined Frequency Transform (PFT) [L03] ou encore d'une architecture imaginée par Hentschel [H02], qui est l'assemblage d'un Overlap Add (OLA) et de plusieurs Goertzels [H02].
- **Channelizeurs avec espacements semi-variables**. Ces channelizeurs offrent une liberté limitée dans le choix des espacements. Dans ce type de channelizeurs figurent les techniques de Coefficient Decimation (CD) [MVLO11] et de Wajih et Gordon [WG04].
- **Channelizeurs avec espacements totalement variables**. Ils permettent de faire une « channelization parcimonieuse», c'est-à-dire avec une totale liberté sur les fréquences centrales des canaux extraits. Parmi ces méthodes, on peut noter les techniques Per Channel [HDR03, L03, MVLO11], Goertzel [H02], Tunable Pipelined Frequency Transform (TPFT) [L03], le Frequency Masking (FRM) [MVLO11, MVMP08] et la méthode Darak, Vinod et Lai [DVL12].

La figure 3.5 résume les catégories des différents channelizeurs.

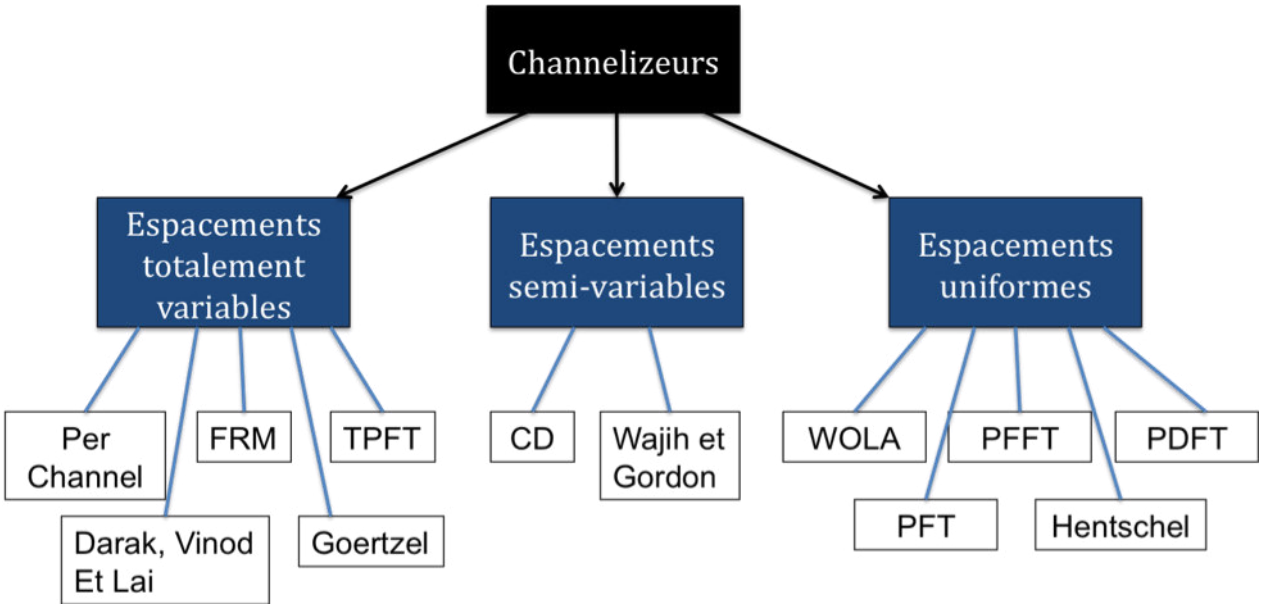


Figure 3.5 – Listes de channelizeurs pour FPGAs existants et catégories de leur appartenance

Les standards radios à traiter peuvent être à distribution non uniforme ou bien à distribution uniforme. Cela signifie que le channelizeur requis pour chaque standard appartient soit appartient à la 1^{ième} catégorie (pour les distributions uniformes comme la FM), soit à l'une des deux dernières catégories (pour les bandes à distribution non uniformes comme la DRM30).

Avant d'aller plus loin, rappelons la définition de la sélectivité qui nous sera utile dans la suite.

Sélectivité = Bande passante du canal / Bande totale de diffusion

3.4.1.1 Channelizeurs à distribution uniforme

Les techniques de channelization qui permettent d'extraire de façon uniforme peuvent être réparties en deux catégories. Il y a d'une part celles qui permettent de sélectionner les canaux à extraire parmi un ensemble de canaux possibles uniformément espacés, et puis il y a celles qui ne peuvent que fournir tous les canaux uniformément espacés.

Dans la première catégorie, on retrouve la technique Pipelined Frequency Transform (PFT) ou encore d'une architecture imaginée par Hentschel, qui est l'assemblage d'un Overlap Add (OLA) et de plusieurs Goertzels.

3.4.1.1.1 Pipelined Frequency Transform (PFT)

C'est la méthode dont est issue la technique TPFT (section 3.4.1.2.3). Basée sur une structure en "arbre", la technique PFT [L03] découpe et filtre successivement la bande spectrale afin d'obtenir

une résolution de plus en plus fine. C'est ce qu'on peut voir dans la figure 3.6-a. Chaque étage découpe les sous-bandes extraites par l'étage précédent en deux.

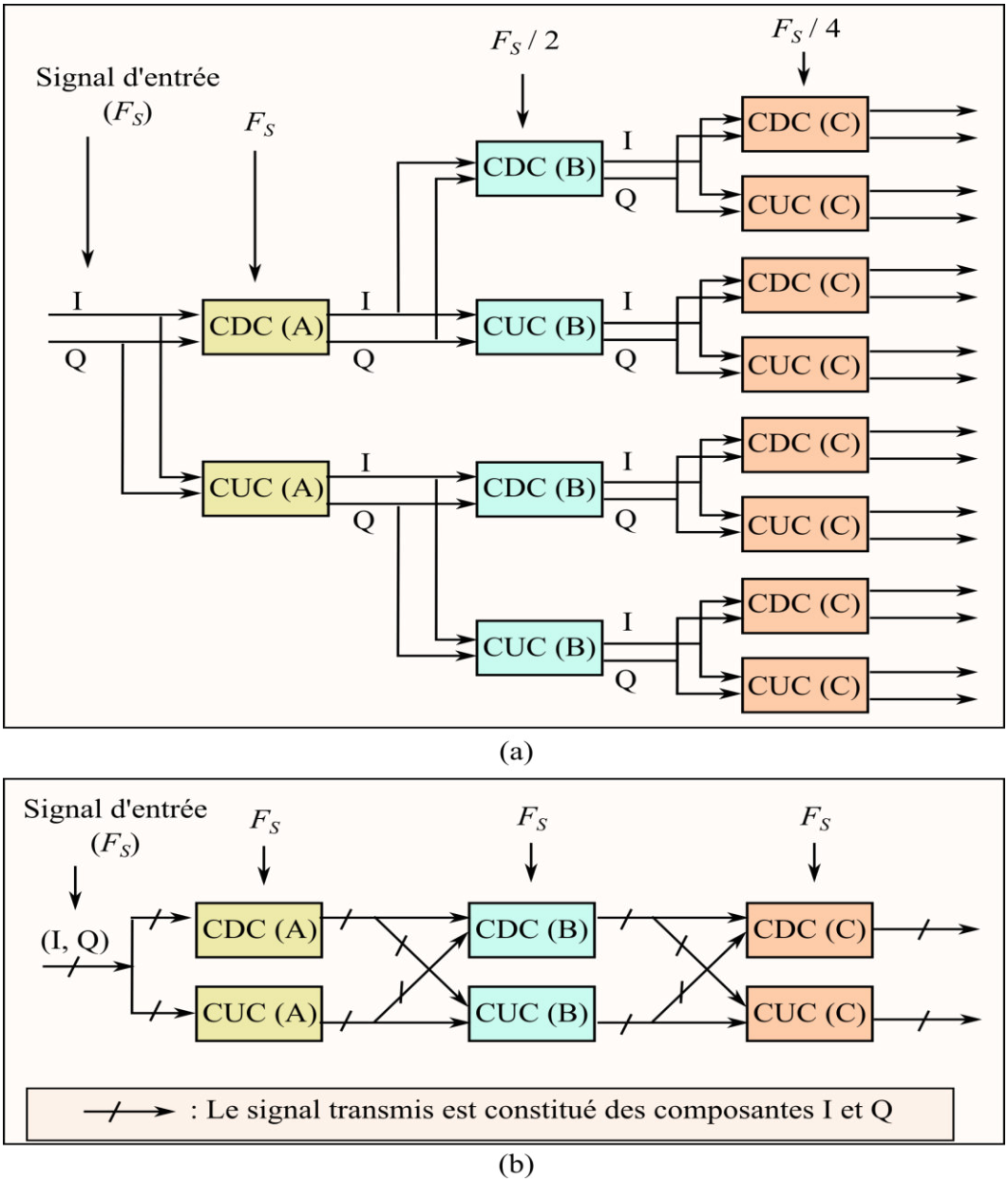


Figure 3.6 – Structure basique du channelizeur PFT. Source adaptée : [L03]. a) Illustration de la méthode. b) Version optimisée de la méthode. Les « CDC » et « CUC » sont respectivement les « complex downconverters » et les « complex upconverters »

La figure 3.6-a illustre clairement le fonctionnement de cette méthode, mais il est possible de simplifier l'implémentation en réduisant considérablement le nombre de blocs CDC et CUC. Cela se fait grâce à un entrelacement entre les différents étages (figure 3.6-b).

Le principe de la méthode est de diviser le signal en 2 sous-bandes. Les sous-bandes sont elles même divisées en 2 autres sous-bandes et ainsi de suite, jusqu'à arriver à un niveau où on peut extraire le canal souhaité du filtrage d'une sous-bande en utilisant un filtre d'ordre réduit. Il est

possible d'utiliser des filtres CIC à chaque étape du traitement pour effectuer le découpage en sous-bandes tout en gardant une phase linéaire. Mais il faudra ici aussi un filtre FIR à chaque étage pour compenser la perte de gain des CIC. La sélectivité pour le standard FM en Europe occidentale varie entre 10^{-3} et 10^{-2}. Elle entraine que le nombre global de multiplieurs requis pour implémenter ces filtres peut atteindre plusieurs milliers. Par exemple, si les filtres de compensations utilisés sont du type « equiripple » et qu'on souhaite obtenir 45 dB de rejection, le nombre de multiplications requis par instant d'horloge système est supérieur à 2500.

Néanmoins, cette méthode présente l'avantage que le nombre de multiplications qu'elle utilise ne dépend pas du nombre de canaux uniformes à extraire, peu importe ce nombre, mais uniquement de la sélectivité et du niveau de rejection souhaité.

3.4.1.1.2 Channelizeur de Hentschel

L'idée est de combiner l'algorithme WOLA (sans la FFT) avec l'algorithme Goertzel [H02]. L'un des avantages de l'algorithme WOLA est qu'il permet d'implémenter un banc de filtres à partir d'un filtre d'ordre élevé en utilisant peu de multiplieurs grâce à la technique du repliement. C'est cette caractéristique de WOLA qu'on exploite dans une architecture du type WOLA + Goertzel. Toutefois, on n'implémente pas la FFT qui est généralement intégrée à l'algorithme WOLA. En effet, l'algorithme WOLA+FFT permet d'extraire des sous-bandes uniformément espacées dont le nombre dépend de la précision de la FFT (2, 4, 8, …128 … 2048 etc.). A la place de la FFT, on met des filtres Goertzel. Chaque filtre Goertzel remplace un point d'extraction de la FFT que la FFT aurait produit et qui est un des canaux recherchés. Etant donné qu'avec la FFT, on extraira en plus des sous-bandes sans informations, cette technique permet de sélectionner uniquement celles qui nous intéressent. Pour cela il suffira de reconfigurer le coefficient 2πki / C où ki est le canal et C le nombre total de canaux possibles.

La figure 3.7 résume cette méthode.

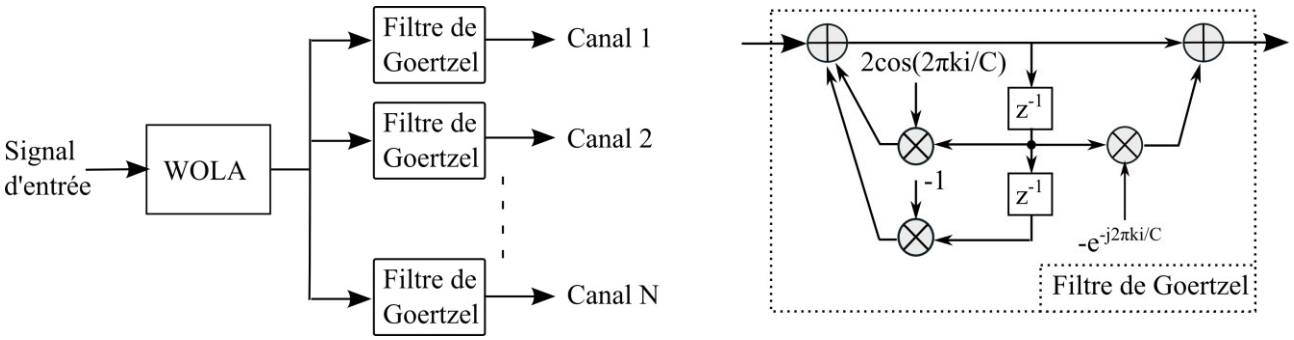


Figure 3.7 – Channelizeur WOLA + Goertzel de Hentschel

Pour ce qui est de la deuxième catégorie, les techniques de channelization optimisées pour les canaux régulièrement espacés avec sous-bandes à largeur égale sont principalement basées sur la FFT. En effet, la FFT permet une gestion hautement optimisée des ressources. En outre, les « Intellectual Property cores » (IP) pour les implémentations FPGA sont largement disponibles aujourd'hui, réduisant ainsi le cycle de conception du banc de filtres.

Pour cette raison, limitons notre investigation aux structures “Pipelined FFT” (PFFT) [L03], “Polyphase DFT” (PDFT) [L03][H02][HDR03][WLW06] et “Weighted Overlap Add » (WOLA) [L03] [H02][WLW06].

3.4.1.1.3 Architecture Pipelined FFT (PFFT)

La technique PFFT, qui traite le signal d'entrée directement avec une FFT, nécessite la pondération du signal d'entrée avec un filtre FIR passe bas pour améliorer la performance du filtrage (figure 3.8). La «pondération» du FIR doit être effectuée avant le traitement FFT, sinon la rejection du banc de filtre au niveau du premier lobe secondaire sera inférieure à 20 dB, ce qui est dû au fait que la réponse impulsionnelle de la FFT se rapproche d'une fenêtre rectangulaire, dont la transformée de Fourier est une fonction sinus cardinal (sinc) avec un premier lobe secondaire à -13 dB seulement. Le filtre de pondération est celui qui aurait été implémenté si on utilisait un « per channel ». Il peut être alors trop large si la rejection requise est élevée comme ce sera le cas pour des canaux FM parfaitement isolés. Par exemple, pour extraire un canal de 200 kHz d’une bande FM de 20 MHz, avec 40 dB de rejet et 10 kHz de bande de transition, il faudrait un filtre de Kaiser de 15000 coefficients.

Les approches PDFT ou WOLA semblent préférables pour notre application afin de mieux gérer les ressources matérielles. De plus, lorsqu’il s’agit de canaux régulièrement espacés avec sous-bandes à largeurs égales, elles se révèlent particulièrement efficaces en termes de complexité de calcul si plusieurs dizaines de canaux sont en jeu [L03][WLW06].

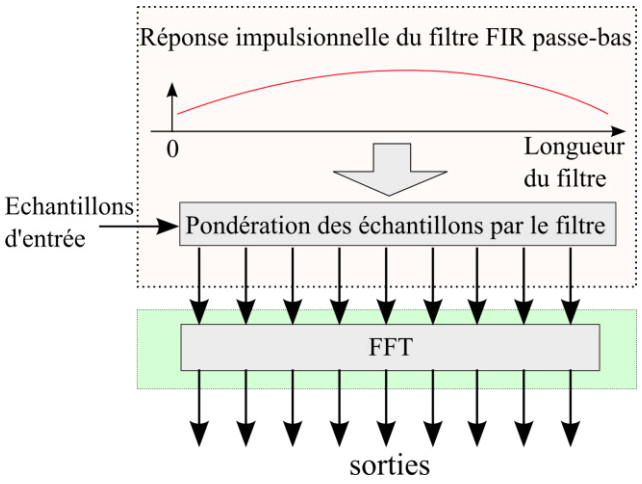


Figure 3.8 – Technique PFFT avec un filtre passe bas pour améliorer la performance de filtrage

3.4.1.1.4 Architectures PDFT et WOLA

Soit $H_P(n)$ la réponse impulsionnelle du filtre d'entrée de longueur L_P, D_P le facteur de décimation de la fréquence d'échantillonnage entre le signal d'origine et de sorties produites, et K_P le nombre de canaux devant être générés par le banc de filtres. Les figures 3.9 et 3.10 illustrent respectivement les architectures PDFT et WOLA.

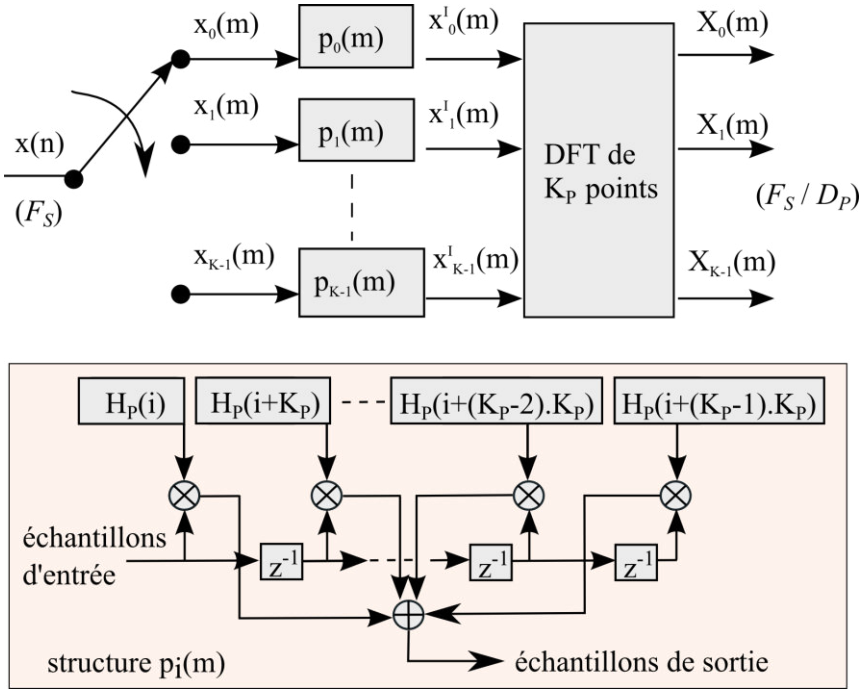


Figure 3.9 – Banc de filtres Polyphase DFT (PDFT). La figure de gauche représente le banc de filtre. La figure de droite esquisse la structure $p(m)$ où $p_i(m)$ représente la décomposition polyphase du filtre. $x(n)$ représente le signal à « channelizer ». $X_i(m)$ représente le $i^{ème}$ canal extrait

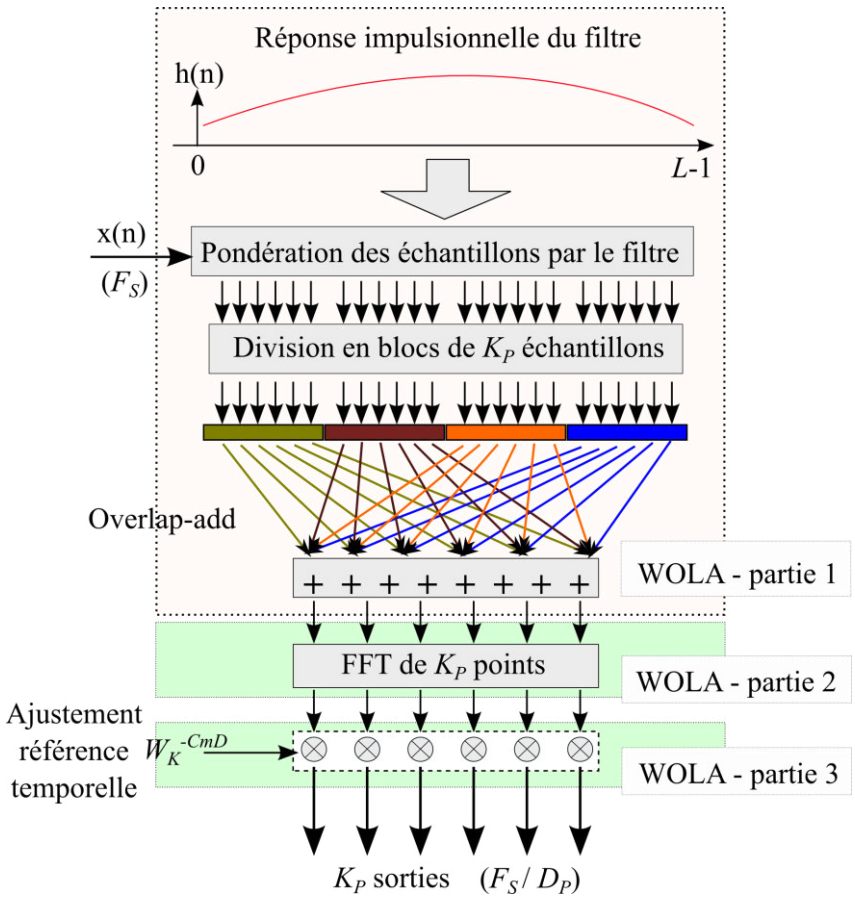


Figure 3.10 – Banc de filtre WOLA, adaptée de [WLW06]. $x(n)$ représente le signal à « channelizer », F_I représente la fréquence d'échantillonnage d'entrée, $W_K = e^{j2\pi/Kp}$, et C et m sont respectivement le numéro de canal extrait et le numéro de l'échantillon extrait. Les parties 1, 2 et 3 sont respectivement les étapes weight overlap add, la FFT, et l'ajustement temporel de l'algorithme WOLA

La structure PDFT basique exposée à travers la figure 3.9 utilise le même nombre de multiplieurs que la version fenêtrée de la PFFT discutée ci-dessus, mais sa mise en œuvre polyphasée permet de travailler à des fréquences beaucoup plus élevées que le PFFT. Cependant, comme la figure 3.9 le montre, l'application de la PDFT sous cette forme basique nécessite la création d'autant de filtres que de canaux à traiter. Cet inconvénient peut être contourné si une structure PDFT plus sophistiquée est implémentée, équivalente à une implémentation de l'algorithme WOLA adapté pour des situations dans lesquelles $D_P = K_P$ et L_P est un multiple de K_P.

La figure 3.11 illustre la structure WOLA – autrement dit PDFT améliorée – mentionnée ci-dessus. Le nombre de multiplieurs requis est beaucoup plus faible que ceux qu'un PDFT de base aurait exigé. En effet, sans compter la FFT, le PDFT de base de la figure 3.9 exige L multiplieurs alors que WOLA n'a besoin que de L_P / K_P multiplieurs. Cette implémentation de WOLA est

également bien adaptée au mode d'acquisition de données de l'IP FFT standard utilisé dans notre design, disponible dans la bibliothèque de composants, et qui envoie et reçoit les données en série. Ainsi, les échantillons de données d'entrée $x^I_0(m)$, $x^I_1(m)$.. $x^I_{K-1}(m)$ (voir figure 3.9 et 3.11) sont cadencés à raison d'un échantillon par cycle-d'horloge ; les échantillons de sortie sont générés de la même façon, c'est-à-dire consécutivement dans l'ordre, $X_0(1)$, $X_1(1)$...$X_{K-1}(1)$, $X_0(2)$, $X_1(2)$...$X_{K-1}(2)$, $X_0(3)$, $X_1(3)$...$X_{K-1}(3)$.

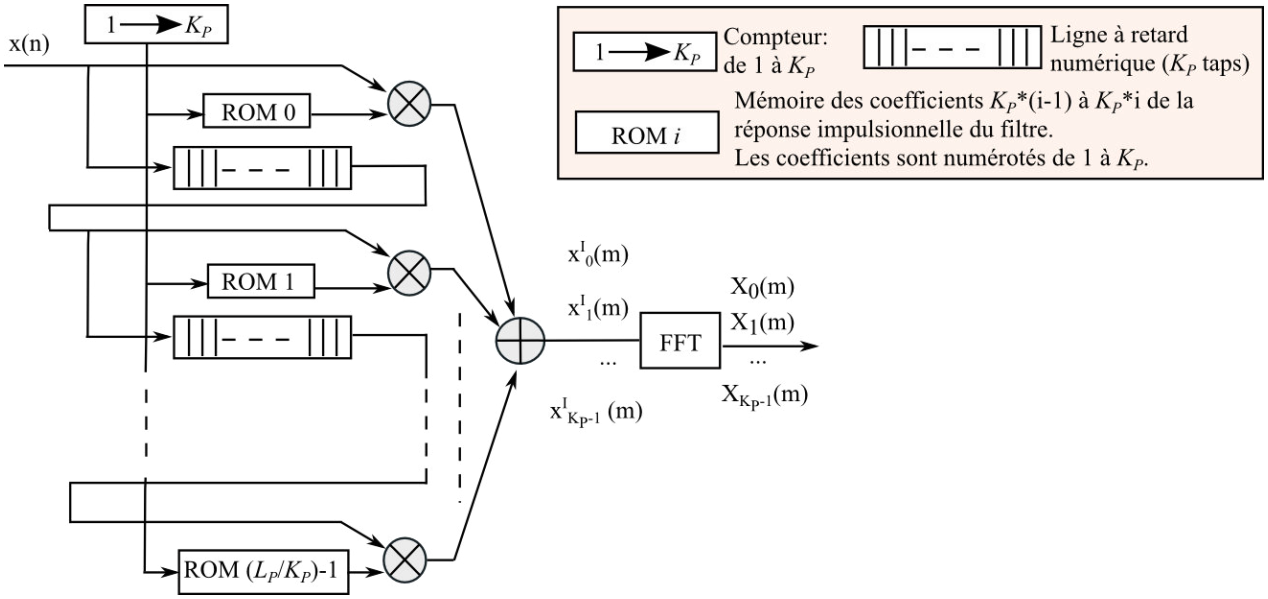


Figure 3.11 – Implémentation de WOLA lorsque $D_P = K_P$ et L_P est un multiple de K_P. Dans cette structure, tous les blocs sont cadencés à la même fréquence. Chaque « ligne à retard numérique » retarde les échantillons "$x(n)$" de K_P cycles d'horloge. Les blocs "ROM i" ont K_P cellules mémoires, numérotées de 1 à K_P. Le « counter block », qui indexe chaque bloc " ROM i" compte de 1 à K_P via une boucle infinie

Le tableau 3.1 récapitule les caractéristiques de toutes les différentes techniques passées en revue jusqu'ici. Elles sont toutes implémentables sur FPGA.

Channelizeur	**Nombre de canaux extraits**	**Gestion des ressources de calcul pour « channelizer » uniformément toute la bande**
PFT	1 à K_P	Non optimale
Hentschel	1 à K_P	Non optimale
PFFT	K_P	Adaptée
PDFT	K_P	Plus adaptée
WOLA	K_P	Optimale

Tableau 3.1 – Tableau récapitulatif des channelizeurs à distribution uniforme

L‘étude des différentes architectures présentée révèle que, pour ce qui est des architectures à distribution uniforme, nous avons vu que deux catégories existent : Celles qui permettent de sélectionner les canaux à extraire parmi un ensemble de canaux possibles uniformément espacés, et puis celles qui ne peuvent que fournir tous les canaux de la bande uniformément espacés. Pour démoduler la radio, nous allons nous aider de celles de la deuxième catégorie. En effet, elles ouvrent la porte à la conception d’un channelizeur qui extrait tous les canaux possibles contenus dans la bande. Pour se faire, comme on a pu le voir plus haut, les structures à base de FFT se révèlent particulièrement efficaces, surtout la technique WOLA qui permet d’utiliser moins de ressources en calcul que les autres pour cette tâche.

3.4.1.2 Channelizeurs à distribution non uniforme

Dans les bandes de radiodiffusion en dessous de 30 MHz, la largeur de chaque canal est souvent très étroite par rapport à la bande (sélectivité de l’ordre de 10^{-4}). C’est le cas du DRM30 que nous étudions ici où la largeur maximale du canal est 20 kHz alors que la bande est étendue sur plus de 25 MHz (3-30 MHz, voir tableaux 2.1 et 2.2). C’est dans ce contexte que nous étudierons les architectures pour faire de la channelization non uniforme.

Beaucoup de ces architectures existent aujourd’hui. Comme dans un FPGA les ressources intrinsèques – notamment les ressources en calcul – sont limitées, examinons-les sous l’angle du nombre de multiplieurs nécessaires pour leur bon fonctionnement. On trouve ainsi :

3.4.1.2.1 Per channel

La channelization par « Per channel » [HDR03, L03, MVLO11] est l'une des plus simples à mettre en œuvre qui soit, mais qui peut aussi s'avérer gourmande en ressources. En effet, elle consiste à faire passer le signal à channelizer par un DDC (Digital Down Converter) qui le transposera à la bande de base et puis à filtrer avec un passe-bas (généralement un FIR pour avoir une réponse linéaire) pour ne récupérer que le canal voulu (figure 3.12). On rajoute autant de DDCs et de filtres passe-bas que de canaux à extraire. Dans notre contexte de filtrage de bandes étroites, l'ordre du filtre est très élevé (> 40000 pour obtenir une atténuation de 45 dB des bandes rejetées avec un filtre de Kaiser) ce qui entraine une grande consommation de ressources si le nombre de canaux est grand.

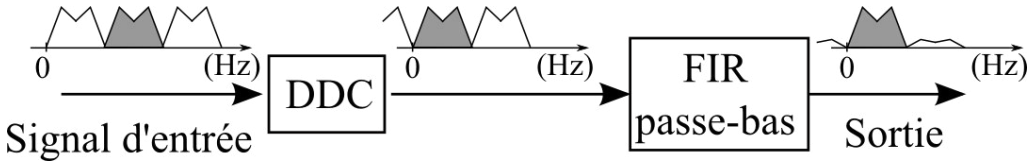


Figure 3.12 – Principe de la channelization « Per Channel » pour extraire 1 canal

3.4.1.2.2 Goertzel

La channelization parcimonieuse par Goertzel [H02] est déjà plus élaborée que la celle par « per channel ». Il ne s'agit pas de la structure OLA+Goertzel mentionnée dans la section 3.4.1 (qui fait une channelization à espacement uniforme et qui a été suggérée par le même auteur dans le même article [H02]) mais plutôt d'un assemblage FIR + Goertzel sans overlap temporel. Elle utilise l'expression donnée par Goertzel qui permet de calculer la transformée de Fourier (TF) d'un signal à une fréquence précise. Son utilisation pour la channelization passe aussi par l'utilisation d'un filtre FIR passe-bas qui servira de gabarit pour extraire chaque canal souhaité. Ce filtre peut être de même ordre que celui du channelizeur « Per Channel » mais il ne sera implémenté qu'une seule fois pour tous les canaux à extraire, peu importe leur nombre. Ensuite, le calcul de la TF opéré par la structure de Goertzel permettra, dans notre cas, de transposer chaque canal à la bande de base et de lui appliquer le FIR passe-bas. La figure 3.13 schématise la structure implémentée.

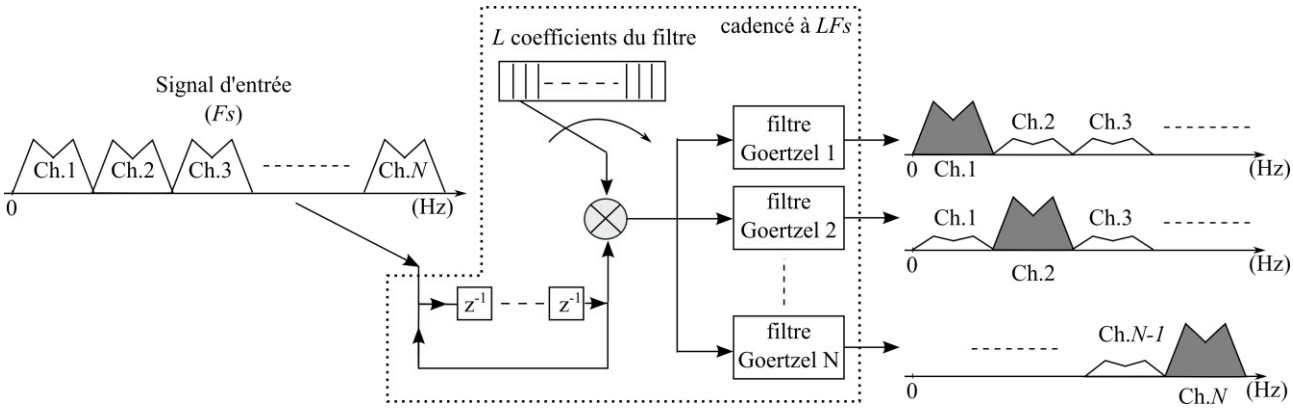


Figure 3.13 – La channelization par Goertzel. « Ch.*i* » signifie « $i^{ème}$ canal ».

L'inconvénient de cette structure est qu'elle ne peut pas être appliquée telle quelle si la fréquence d'échantillonnage du signal d'entrée est élevée. En effet, elle requiert qu'à chaque instant du signal d'entrée l'on utilise tous les coefficients du filtre pour pondérer autant d'échantillons d'entrée qu'il y a de coefficients. Cela impose qu'il faut disposer d'une horloge dont la fréquence est supérieure ou égale à LF_S (où L est l'ordre du filtre et F_S est la fréquence d'échantillonnage d'entrée) et d'un composant capable de la supporter. Si le produit LF_S est élevé, il n'est pas possible d'utiliser cette méthode, faute de matériel adéquat. Par exemple, les valeurs L = 600 et F_S = 30 MHz exigent

d'avoir une horloge supérieure ou égale à 18 GHz dans le FPGA, chose impossible à l'heure actuelle. Dans notre cas, la méthode est inutilisable telle quelle car nous avons besoin d'un L supérieur à 200 (compte tenu de la sélectivité des canaux) et d'un F_S supérieur à 30 MHz (pour respecter le critère de Nyquist lors de la numérisation des bandes).

3.4.1.2.3 Tunable Pipelined Frequency Transform (TPFT)

La technique TPFT [L03], directement issue de la PFT [L03] présentée en 3.4.1.1.1, consiste à diviser le signal en 2 sous-bandes. Les sous-bandes sont elles même divisées en 2 autres sous-bandes et ainsi de suite, jusqu'à arriver à un niveau où on peut extraire le canal souhaité du filtrage d'une sous-bande en utilisant un filtre d'ordre réduit. Entre chaque étage de division en sous-bandes, on peut utiliser une structure NCO + multiplieur pour transposer en fréquence les signaux issus de l'étage précédents avant de les envoyer à l'étage suivant. En effet, les NCOs (de l'anglais *numerically controlled oscillators*), sont des oscillateurs numériques qui génèrent des signaux sinusoïdaux utiles pour transposer un signal via l'équation (3.3) impliquant un signal temporel $s(t)$ et sa transformée de Fourier $S(f)$:

$$F_T\left(e^{+j2\pi f_0 t}\, s(t)\right) = S\left(f - f_0\right) \tag{3.3}$$

où F_T dénote l'opération de calcul de la transformée de Fourier.

Cela permet d'ajuster la position des canaux à extraire afin qu'ils soient correctement filtrés dans les étages suivants. La figure 3.14 schématise cette technique où tous les canaux à extraire sont de même largeur.

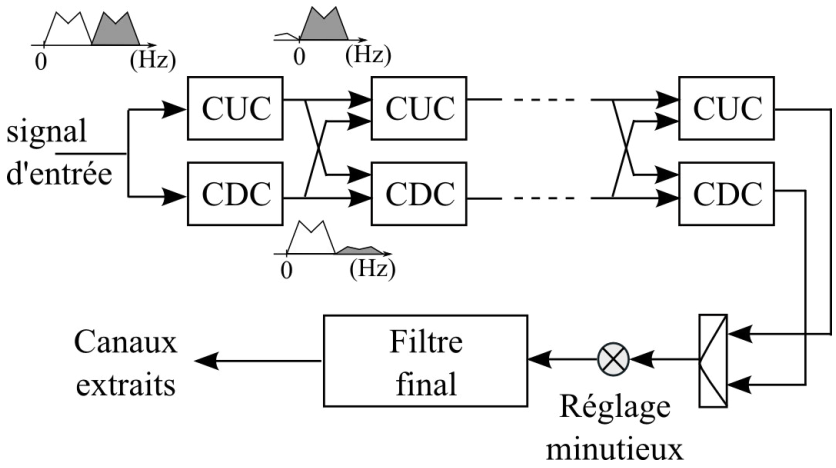


Figure 3.14 – La technique TPFT adaptée au cas où les canaux à extraire sont de même largeur. Le CDC (Complex Down Converter) et le CUC (Complex Up Converter) sont des blocks constitués d'un étage de mixage pour transposer le signal entrant et de filtres (CIC/FIR) pour extraire respectivement la moitié inférieure et la moitié supérieure du signal entrant

Avec cette structure, il n'y a pas besoin d'un CUC/CDC par sous-bande à traiter, mais il suffit d'un seul CUC/CDC cadencé à F_S pour traiter toutes les sous-bandes de l'étape précédente. Cela est dû au fait que les sous-bandes ont des fréquences d'échantillonnage inversement proportionnelles à F_S. Avec cette technique, comme avec la méthode PFT, il est possible d'utiliser des filtres CIC à chaque étape du traitement pour effectuer le découpage en sous-bandes tout en gardant une phase linéaire. Cependant, les filtres CIC présentent l'inconvénient d'avoir un gain non-constant décroissant dans la bande passante. Il faut donc un filtre FIR à chaque étage pour compenser cette perte de gain. Avec un nombre d'étages supérieur ou égal à 10 (ce qui est notre cas compte tenu de la sélectivité de l'ordre de 10^{-4}), le nombre global de multiplieurs requis pour implémenter ces filtres peut être de l'ordre de plusieurs milliers. Par exemple, si les filtres de compensations utilisés sont du type « equiripple » et qu'on souhaite obtenir 45 dB de rejection, le nombre de multiplications requis par instant d'horloge système est supérieur à 4800 d'après les simulations.

3.4.1.2.4 Frequency Masking (FRM)

L'objectif de la FRM est de constituer un filtre FIR, H_{FRM}, de bande de transition étroite [L86] dont la complexité est réduite. Cette technique a ensuite été adaptée en banc de filtres [MVLO11, MVMP08]. A partir d'un premier filtre FIR classique peu sélectif $Ha(z)$, appelé « modal filter », on génère deux autres filtres $Ha(z^M)$ et $Hc(z^M)$ en rajoutant des retards numériques de M-taps où $Hc = 1 - Ha$ et M est un entier choisi par l'utilisateur (figure 3.15). On appele L_{Ha} la longueur de Ha. Chacun de ces deux derniers filtres opère comme un banc de filtre dont l'espacement est uniforme et dont les sous-filtres sont identiques (figure 3.16). Ils offrent alors la possibilité d'effectuer un filtrage plus sélectif comme la largeur de la bande de transition des sous-filtres dépend du nombre

de lignes de retards numériques. Après, il faut utiliser d'autres filtres, des « filtres de masquage », pour « masquer » les répliques spectrales gênantes (figure 3.17). On appelle H_{aM} et H_{cM} les filtres de masquage. On a alors (3.4):

$$H_{FRM} = Ha(z^M)\, H_{aM}(z) + Hc(z^M)\, H_{cM}(z) \tag{3.4}$$

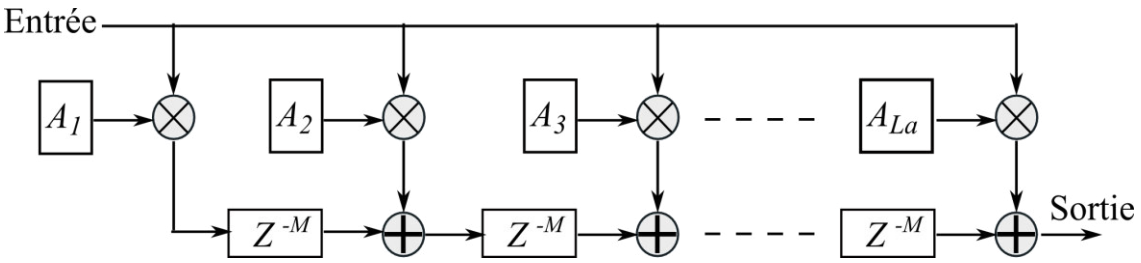


Figure 3.15 – Structure de $Ha(z^M)$. A_i représente le $i^{ème}$ coefficient de Ha. Z^M représente une ligne de retard numérique de M-taps

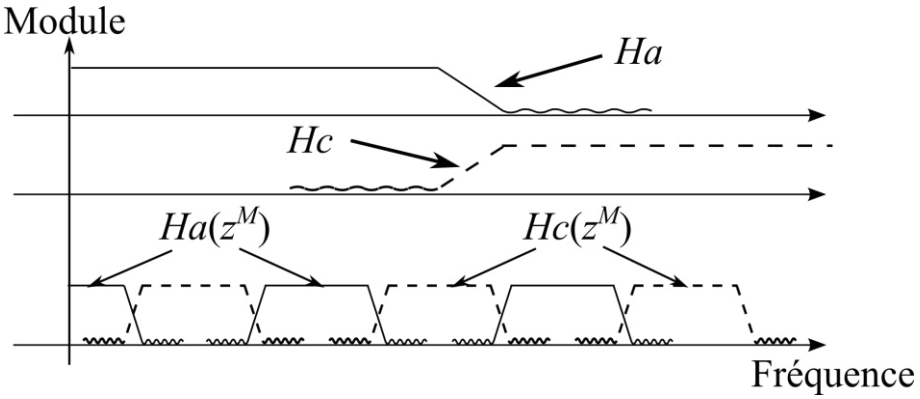


Figure 3.16 – spectres de $Ha(z^M)$ et $Hc(z^M)$

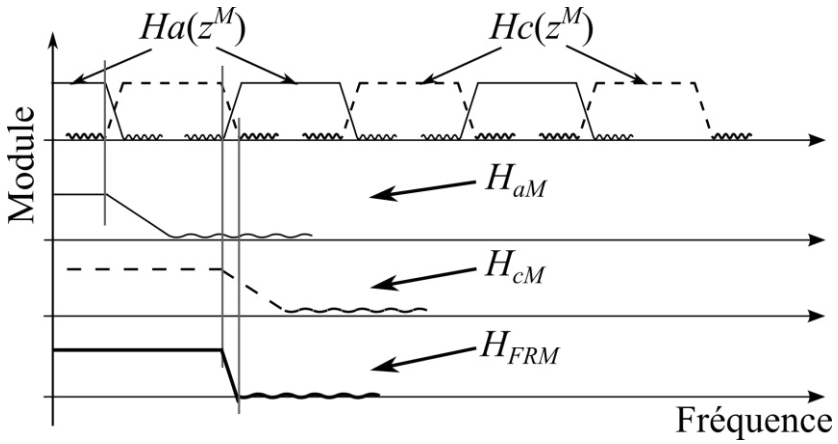


Figure 3.17 – spectres de H_{aM}, H_{cM} et H_{FRM}

Pour faire une channelization parcimonieuse avec des sous-bandes à largeurs égales, une implémentation de cette technique consiste à rajouter des NCOs qui ramèneront à la bande de base les sous-bandes issues de $Ha(z^M)$ et $Hc(z^M)$, avant de les envoyer aux filtres de masquage. Il faut ensuite rajouter autant de filtres de masquage que de canaux à extraire (figure 3.18).

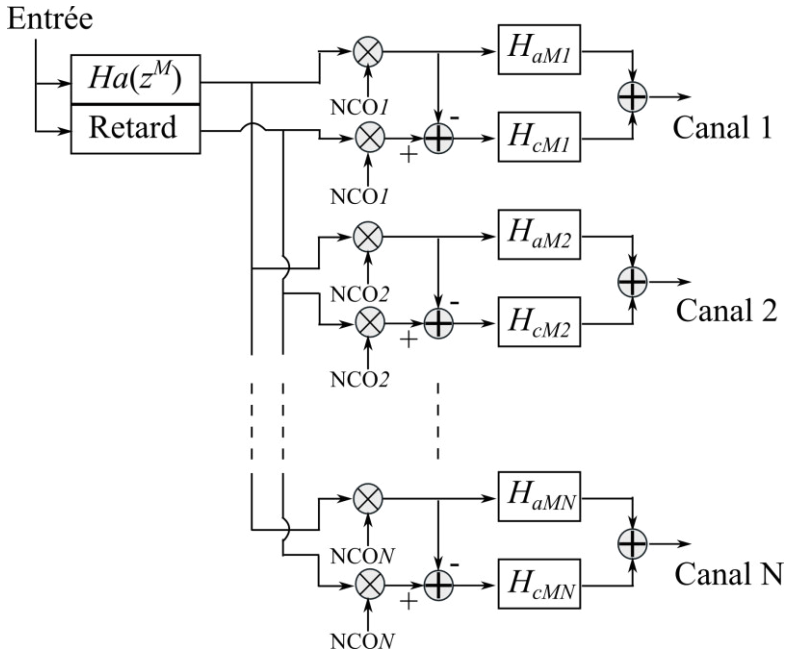


Figure 3.18 – Banc de filtres basé sur la technique FRM. « Retard » représente un retard numérique de $z^{-M(L_{Ha}-1)/2}$. NCO*i* représente les sinusoïdes générées par la $i^{ème}$ NCO

Comme nous sommes intéressés par une channelization avec sous-bandes à largeurs égales, on peut prendre tous les H_{cMi} identiques à un H_{cM} de longueur L_{HcM} et tous les H_{aMi} identiques à un H_{aM} de longueur L_{HaM}. Il s'agit sans doute de l'une des plus simples adaptations de cette technique à notre situation mais elle demeure fidèlement représentative des ressources à mettre en œuvre pour utiliser ce type de banc de filtres. Ainsi, cette technique permet certes d'élaborer des bancs de filtres qui selon les situations peuvent être moins consommateurs que d'autres techniques comme le « Per Channel » par exemple. Cependant, pour notre application où la sélectivité est faible, utiliser cette technique nous ferait certainement implémenter des filtres de masquage très sélectifs avec un certain nombre de coefficients (par exemple $L_{HaM} > 10000$ pour avoir 40 dB de rejection avec un filtre de Kaiser); ce qui se traduit par l'utilisation de beaucoup de multiplieurs.

3.4.1.2.5 Coefficient Decimation (CD)

Cette technique [MVLO11] se base sur la décimation par un facteur D de la réponse impulsionnelle d'un filtre modal. Cette décimation consiste garder intact un coefficient de la

réponse impulsionnelle sur *D* et à remplacer par zéro les autres coefficients. Le filtre modal dont il est question ici est le filtre à implémenter pour extraire un canal voulu de la bande à traiter. Ce filtre a la même complexité que celui qu'on implémenterait dans un banc de filtres « per channel » équivalent. Cette décimation permet de créer ainsi plusieurs répliques spectrales de la réponse en fréquence du filtre. Ces répliques sont centrées à des multiples entiers de *2π/D*. Ensuite, à l'aide d'une simple soustraction, on peut isoler la réplique spectrale qui contient le canal recherché (figure 3.19). Enfin, grâce à un NCO on peut ramener le canal extrait à la bande de base.

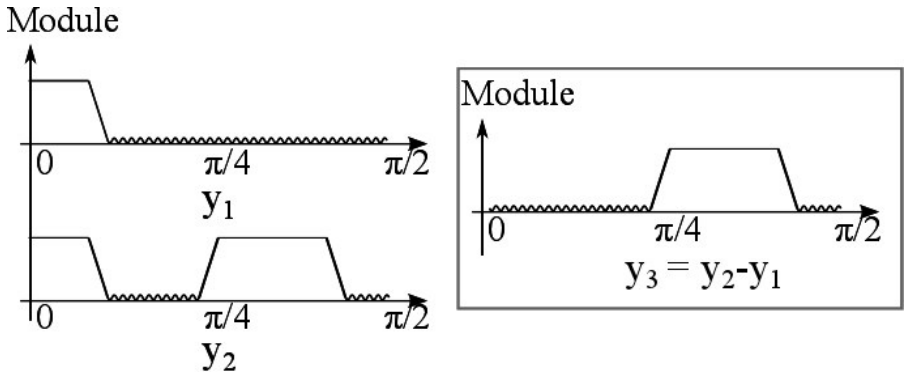


Figure 3.19 – Exemple avec *D* = 3. y_1 et y_2 sont respectivement les allures du filtre modal et du filtre décimé. y_3 est la réplique spectrale isolée

Bien que cette structure soit flexible, cette flexibilité reste limitée par trois points :

- Le facteur de décimation doit avoir une valeur minimale en dessous de laquelle apparaît un recouvrement spectral entre les sous-bandes extraites.
- Les sous-filtres du banc de filtre ne peuvent être centrés que sur des multiples entiers de *2π/D*.
- Le filtre modal de base doit être implémenté pleinement avec tous les multiplieurs requis.

Le dernier point pose un problème important dans notre cas. Ainsi, même si on trouvait dans une situation favorable où les deux premiers points sont résolus (c-à-d où *D* est suffisamment grand et où les sous-filtres sont centrés sur les bons canaux), on serait confronté au problème du nombre de multiplieurs nécessaires étant donné que le filtre modal ici a la même complexité que celui du « per channel ».

3.4.1.2.6 Channelizeur de Wajih et Gordon

L'idée de cette technique [WG04] est d'utiliser les « modulated perfect reconstruction (PR) filterbanks » pour channelizer. Plus précisément, il s'agit de reconstruire uniquement les canaux utiles après avoir décomposé le signal par un filtre polyphase. En effet, lorsqu'on fait une reconstruction parfaite, ou perfect reconstruction (PR), après une décomposition polyphase, on se

sert des sous-bandes générées par la partie décomposition – généralement appelée « analysis » – afin de reconstituer le signal dans une autre partie – appelée « synthesis ». Ce que propose Wajih et Gordon est de ne reconstituer que les canaux qu'on souhaite extraire plutôt que le signal entier. En d'autres termes, dans la partie synthesis, on n'utilise que les sous-bandes qui font partie des canaux à extraire pour reconstituer ces derniers. Il s'agit d'une idée astucieuse qui permet de faire des optimisations de ressources mais qui nécessite que la largeur des canaux extraits soit égale à un multiple entier des sous-bandes extraites dans la partie analysis. De plus, cela requiert que le canal à extraire soit centré soit au centre, soit à l'extrémité d'une des sous-bandes qui constituent le canal en question selon que ce dernier est constitué respectivement soit d'un nombre impair soit d'un nombre pair de sous-bandes.

3.4.1.2.7 Channelizeur de Darak, Vinod et Lai

Le channelizeur présenté par Darak, Vinod et Lai [DVL12] est basée sur une combinaison de la FRM et de la CD technique afin d'obtenir une architecture plus flexible que ces 2 techniques. L'architecture générale de ce channelizeur est proche de celle de la technique FRM c'est-à-dire qu'on retrouve des filtres interpolés $Ha(z^{FInt})$ – où $FInt$ est le facteur d'interpolation – issus d'un filtre modal et son complémentaire, des filtres de masquage et un additionneur. Les originalités de cette technique se font voir à plusieurs niveaux. Premièrement, le $FInt$ utilisé pour générer les filtres interpolés est rationnel ($FInt = M / D_C$). Ils sont créés par une interpolation de facteur M de la réponse impulsionnelle (comme dans le cas de la technique FRM) précédée d'une décimation de facteur D_C du filtre modal. Cela rend le choix de la largeur des sous-bandes plus libre. Deuxièmement, sans augmenter le nombre de multiplications, on peut implémenter plusieurs filtres interpolés correspondant à différentes valeurs du facteur d'interpolation (figure 3.20). Cela permet d'avoir un channelizeur à largeurs non uniformes, ce qui ne nous sera pas utile vu qu'on souhaite obtenir une largeur de sous-bande constante. Troisièmement, lorsque cela est possible, on génère les différents filtres de masquage à l'aide de la technique CD. On peut ainsi, à partir d'un filtre de masquage modal, générer plusieurs filtres de masquage sans augmenter la complexité en multiplications. Cette architecture peut donc s'avérer, dans certains cas, moins consommatrice en multiplieurs que la FRM présentée plus haut (section 3.4.1.2.4).

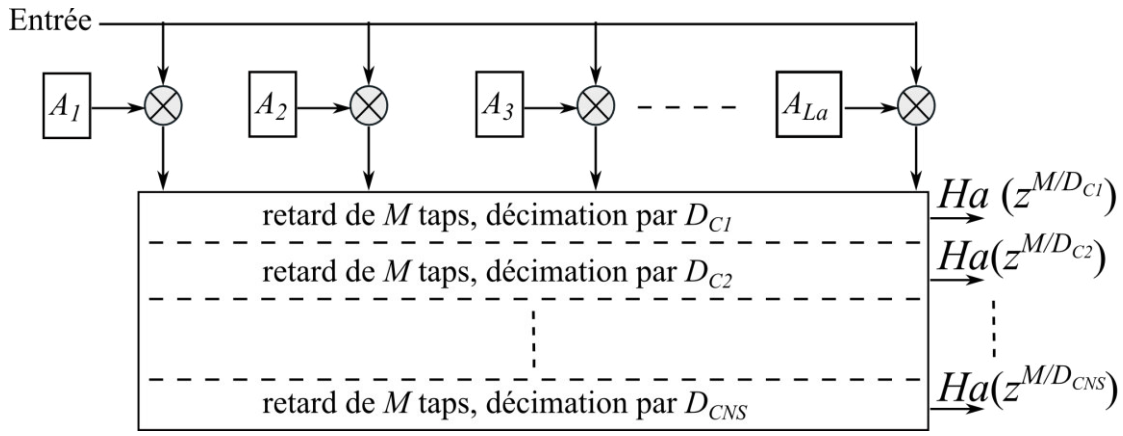


Figure 3.20 – Différents filtres interpolés dérivés du filtre modal *Ha*. « Retard de *M* taps » représente une ligne à retard numérique de z^{-M}. Chaque D_{Ci} est un entier correspondant à une largeur de sous-bande différente. *NS* est le nombre total de largeurs de sous-bandes différentes

Cependant, comme la structure FRM citée plus haut, ce channelizeur présente une faiblesse lorsqu'il s'agit d'obtenir des bandes étroites. En effet, l'étroitesse de la largeur des canaux oblige à avoir des filtres de masquage assez sélectifs. Du coup, comme on a pu le voir pour le channelizeur CD, soit la distribution des canaux à extraire nous est très favorable et il suffit d'implémenter un seul filtre de masquage modal et déduire les autres par CD, soit il faut implémenter plusieurs filtres de masquage comme ceux de la technique FRM. Dans tous les cas, il faut implémenter au moins un filtre de masquage. Cette solution peut s'avérer très consommatrice en ressources.

Le tableau 3.2 récapitule les caractéristiques de toutes les différentes techniques identifiées dans l'état de l'art.

Channelizeur	Flexibilité : Espacement entre canaux extraits		Implémentable si $Fs \geq 60$ MHz ?*	Nombre de multiplications probablement supérieur à 3926 ? **	Implémentation sur FPGA***
	Variable	Semi-variable			
Per Channel	X		Oui	Oui	Faisable
Goertzel	X		Non	Oui	Faisable
TPFT	X		Oui	Oui	Faisable
FRM	X		Oui	Oui	Faisable
CD		X	Oui	Oui	Faisable
Wajih et Gordon		X	Oui	Pas forcément	Pas traité
Darak, Vinod et Lai	X		Oui	Oui	Faisable

* 60 MHz est la fréquence minimale dont nous aurons besoin pour numériser la bande DRM30 (3-30 MHz) tout en respectant le critère de Nyquist.

** Il s'agit du nombre de multiplications requises à chaque instant $1/Fs$. Le nombre inclus les multiplications réelles et complexes. On considère qu'au delà de 3926, ce nombre est très élevé. En effet, on prend comme référence le Stratix V 5SGSD8 [ALTERA13], qui est l'un des FPGAs possédant l'une des plus grandes quantités de multiplieurs « câblés » à l'heure actuelle, chez le constructeur ALTERA. Ses ressources permettent d'instancier jusqu'à 3926 multiplieurs réels 18 x 18 bits.

*** le mot « faisable » indique que la méthode est implémentable sur un FPGA dans des conditions de sélectivité moins rude. Pour une sélectivité de 10^{-4} comme celle étudiée ici, il incombe à l'utilisateur de s'assurer que son FPGA est à même de supporter l'implémentation

Tableau 3.2 – Tableau récapitulatif des channelizeurs à distribution non uniforme

Le tableau 3.2 montre clairement que les méthodes proposées pour faire une channelization de l'intégralité de la bande DRM30, c-à-d la channelization avec espacement variable ou semi-variable, vont pour la plupart engendrer l'utilisation d'un grand pourcentage des multiplieurs instanciables dans le FPGA, voire de leur totalité. Cela peut-être gênant, si on veut effectuer d'autres traitements au sein du même FPGA comme la démodulation des canaux.

Ces considérations motivent notre choix de proposer une nouvelle méthode de channelization à espacement variable, qui a pour vocation d'utiliser moins de multiplications que celles présentées jusqu'à présent, lorsque la sélectivité est faible (10^{-4}). C'est l'objet de la section 3.4.3.

3.4.1.3 Conclusion sur l'état de l'art

Ainsi, pour la « channelization » de la radio,

- Dans le cas des standards à distribution uniforme (FM), nous proposerons un channelizeur qui, basé sur l'algorithme WOLA, permet d'extraire toutes les stations possibles contenues dans la bande radio.
- Dans le cas des standards à distribution non uniforme (DRM30), nous allons examiner une proposition d'architecture permettant de faire une channelization parcimonieuse tout en réduisant les ressources en calculs. Le système proposé sera basé sur la combinaison de plusieurs techniques de channelization mentionnées plus haut.

3.4.2 Channelization en FM, architecture proposée

Les signaux à « channelizer » sont maintenant supposés complexe au sens mathématique. Les architectures proposées seront donc aisément adaptables au cas où les signaux à « channelizer » sont réels.

En FM, les stations ont des bandes passantes de même largeur et sont situées à des fréquences données par $i \times 0.1$ MHz + B_{OFMB} MHz, où i est un entier et B_{OFMB} est le début de la bande FM en MHz. On se servira de l'algorithme WOLA comme cela a été discuté dans la section 3.4.1.1. Il faut donc maintenant fixer la bande passante, la rejection du filtre, la bande de transition et la longueur du filtre de fenêtrage. Pour le cas de la bande FM (qui est le standard qui nous servira de modèle pour les bandes en VHF II), la largeur de la bande désirée a été fixée à 100 kHz vu que ça correspond à la déviation fréquentielle maximale du signal FM.

Rappelons que nous avons appelé $H_P(n)$ la réponse impulsionnelle du filtre instancié par WOLA dont la longueur est L_P. D_P est le facteur de décimation de la fréquence d'échantillonnage entre le signal original et les sorties générées, et K_P le nombre de canaux devant être générés par le banc de filtres. Un filtre de Kaiser de longueur $L_P = 4096$ échantillons permet d'obtenir une rejection de 60 dB (figure 3.21).

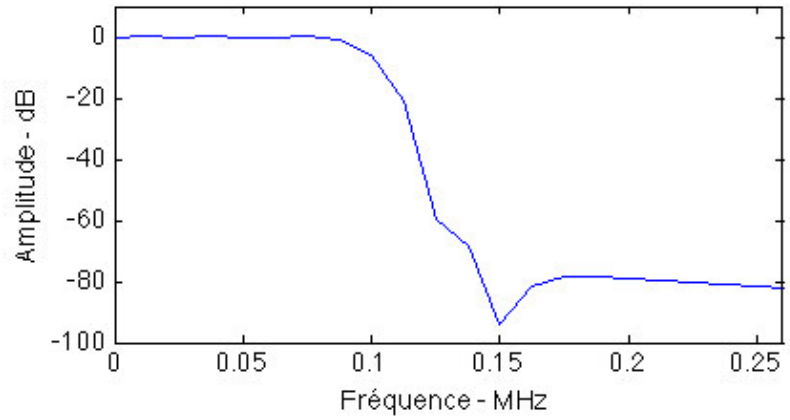


Figure 3.21 – Allure du spectre de H_P. Il s'agit d'un filtre de Kaiser de 4096 échantillons

Le banc de filtres WOLA transpose chaque canal extrait en bande de base. Soit F_i la fréquence d'échantillonnage du canal extrait. Pour respecter le critère de Nyquist, on doit avoir F_i /2 ≥ 100 kHz (cf. section 2.1.2 sur les différents standards). On doit aussi concevoir un banc de filtre qui prend en compte la position des canaux dans le spectre FM. Dans plusieurs pays, les stations FM sont centrés sur 87.5 + i × 0.1 (MHz), où $i \in \{0, 1, \ldots, 205\}$. Un exemple est donné dans la figure 3.22. Comme notre banc de filtres créera des sorties pour ces stations, il doit aussi, en principe, respecter l'espacement de 0.1 MHz.

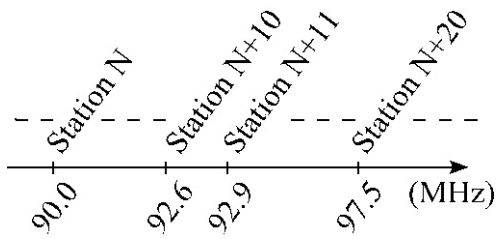


Figure 3.22 – Exemple de distribution des stations radio

Selon le critère de Nyquist, on doit avoir F_{SFM} / 2 ≥ B_{FM} où F_{SFM} est la fréquence d'échantillonnage du signal FM à channelizer et B_{FM} la largeur de la bande FM commerciale. B_{FM} est généralement égale à 20 MHz. Vu que $F_{SFM} = D_P \times F_i$ et F_i /2 ≥ 100 kHz, on doit avoir $F_{SFM} \geq 2 \times D_P \times 100$ kHz. Nous avons vu plus haut que le nombre de canaux sortis par l'IP FFT est K_P et que $D_P = K_P$. Les K_P canaux sortant ne sont pas indépendants. En effet, le nombre de canaux indépendants générés par l'IP FFT est K_P / 2 car l'IP génère les échantillons des composants négatifs et positifs d'un canal indépendant, par trame de sortie. Comme dans notre situation les échantillons des composants négatifs et positifs sont identiques, on peut considérer que la FFT génère seulement K_P /2 canaux utiles (c'est-à-dire indépendants). Par conséquent, si l'on considère un entier positif j tel que j × 0.1 MHz est l'espacement entre les centres des canaux extraits, on peut

déduire que (F_{SFM} / 2) / (K_P / 2) = $j \times 0.1$ MHz à cause du banc de filtre, où F_{SFM} / 2 fait référence à la bande passante qui doit être traitée et K_P / 2 est le nombre de canaux extraits utiles. De plus, l'IP FFT exige à K_P d'être un multiple de 2 (64, 128, 256, etc.). Les conditions sur F_{SFM} mentionnées plus haut sont listées après dans (3.5).

$$\begin{aligned} F_{SFM} &\geq 40 \\ F_{SFM} &\geq 0.2 \times D_P \\ F_{SFM} \ / \ K_P &= j \times 0.1 \end{aligned} \tag{3.5}$$

où l'unité de F_{SFM} est le MHz.

Des équations (3.5), en plus du fait qu'on a choisi $K_P = D_P$, on peut déduire que $j > 1$. Cela signifie qu'on ne peut pas, en utilisant l'architecture WOLA de base présentée jusqu'à présent, implémenter un banc de filtres capable de démoduler une distribution arbitraire de stations dont les fréquences sont centrées à des multiples de 0.1 MHz. Comme nous le verrons plus bas, il est cependant possible d'atteindre un résultat équivalent avec $j > 1$, pourvu qu'on adapte la structure WOLA de façon approprié. Une fois de plus, en utilisant les équations (3.5) et $K_P \in \{64, 128, 256 ... 16384\}$, on peut créer un tableau des valeurs autorisées de F_{SFM} en fonction de j et K_P (ou D_P) (tableau 3.3).

		j					
K_P (ou D_P)	K_P / 2	1	2	3	4	5	...
64	32	6.4	12.8	19.2	25.6	32	…
128	64	12.8	25.6	38.4	51.2	64	…
256	128	25.6	51.2	76.8	102.4	128	…
512	256	51.2	102.4	153.6	204.8	256	…
…	…	…	...	...	...		…

Tableau 3.3 – Quelques valeurs possibles de F_{SFM} en MHz. Les cases grises dénotent les possibilités non autorisées de par le fait que F_{SFM} doit être supérieur à 40 et $j > 1$

Le tableau 3.3 montre qu'il y a plusieurs possibilités de combinaisons de j, K_P et F_{SFM} qui

peuvent correspondre à notre application. Ceci étant, nous préférons implémenter un banc de filtre qui satisfait les conditions supplémentaires suivantes:

1. La plus basse fréquence d'horloge permise. En effet, les designs FPGA ont souvent une fréquence d'horloge maximale au delà de laquelle le bon fonctionnement du design n'est pas garanti.
2. L'utilisation des ressources matérielles la plus efficace. Il fut découvert durant le prototypage que, en général, plus K_P est faible dans le banc de filtre, plus il y a de ressources matérielles utilisées.
3. Une bonne qualité audio. Il fut aussi déterminé empiriquement que, plus K_P est faible, meilleure est la qualité audio des stations FM démodulées, vraisemblablement parce qu'on possède plus d'échantillons par hertz sur le signal audio de sortie.

Le jeu de paramètres $K_P = 128$, $j = 4$ et $F_{SFM} = 51.2$ MHz représente un bon compromis compte tenu des conditions mentionnées plus haut étant donné que: 51.2 MHz est l'une des plus faibles fréquences d'horloge possibles ; la qualité audio avec $K_P = 128$ est plutôt bonne selon les tests d'écoute, et elle est clairement meilleure que celle avec $K_P = 256$; et l'utilisation des ressources est réduite comparé à $K_P = 64$. Ces paramètres sont alors ceux qui ont été choisis pour le channelizeur que nous décrivons. La figure 3.23 illustre l'implémentation d'un banc de filtres pour $j = 4$.

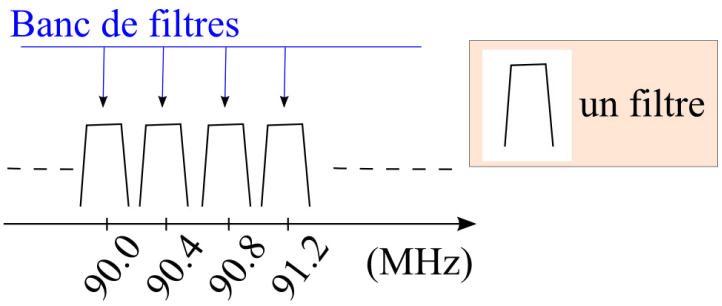


Figure 3.23 – Implémentation du banc de filtres pour $j = 4$. Les quatre flèches verticales indiquent quatre occurrences consécutives du banc de filtres implémenté. Notons que l'espacement entre deux occurrences consécutives est 0.4 MHz, ce qui est égal à $j \times 0.1$ MHz.

Afin d'implémenter cette solution avec notre jeu de paramètres, $j = 4$, $D_P = 128$, $F_{SFM} = 51.2$ MHz, notre architecture comprend un bloc re-échantillonneur (décrit dans les sections 4.2.2.1 et 4.2.3.1) qui change la fréquence d'échantillonnage du signal de 87 MHz à 51.2 MHz. Dans ce cas, on aura $F_i = F_{SFM} / D_P = 400$ kHz. On observe aussi que comme $D_P = K_P$, le facteur W_K^{-CmD} de la figure 3.10 est égal à 1. Cela fait que l'étape "ajustement référence temporelle" de l'algorithme WOLA n'a pas besoin d'être implémentée, menant ainsi à une simplification de notre architecture.

On retourne maintenant à la question de savoir comment construire un banc de filtre capable d'extraire n'importe que jeu de stations dont les fréquences centrales sont situées à $87.5 + i \times 0.1$ (MHz), $i \in \{0, 1, \ldots, 205\}$. Par exemple, considérons la distribution de stations dans la bande FM suivante (figure 3.24).

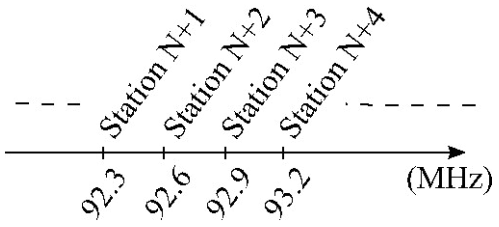


Figure 3.24 – Second exemple de distribution des stations radios

Le banc de filtres d'espacement 400 kHz ne peut pas extraire toutes les stations de cette distribution. D'ailleurs, aucune implémentation de WOLA avec $j > 1$ n'en est capable. Pour résoudre ce problème, la solution adoptée est simplement d'ajouter des bancs de filtres WOLA supplémentaires en parallèle, comme le montre la figure 3.25.

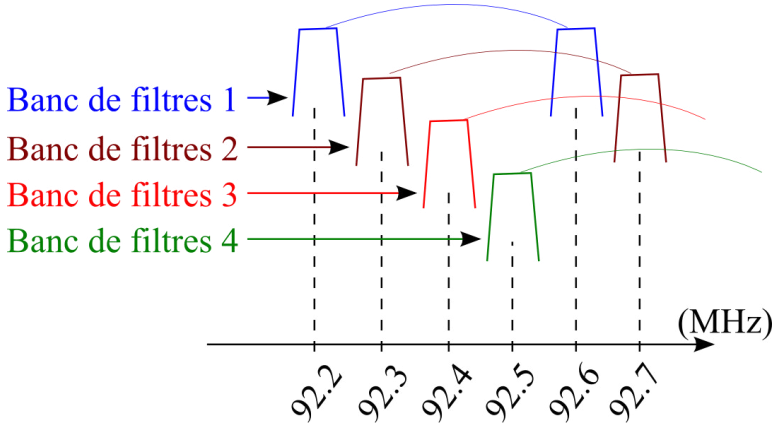


Figure 3.25 – Système de bancs de filtres pour une extraction entrelacée des canaux

Un moyen simple de créer des bancs de filtres entrelacés est de créer des copies du signal d'entrée transposées de 0, 100, 200 et 300 kHz, et d'envoyer les signaux résultants aux différentes copies du banc de filtres WOLA. Le channelizeur final et complet, est alors composé d'un re-échantillonneur suivi de 4 branches désignées par « filterbanks 1, 2 3 et 4 » dans la figure 3.25. La figure 3.26 donne un synopsis complet de sa structure où les branches 2, 3 et 4 sont équipées chacune d'un NCO pour générer les sinusoïdes de fréquences 100, 200 et 300 kHz.

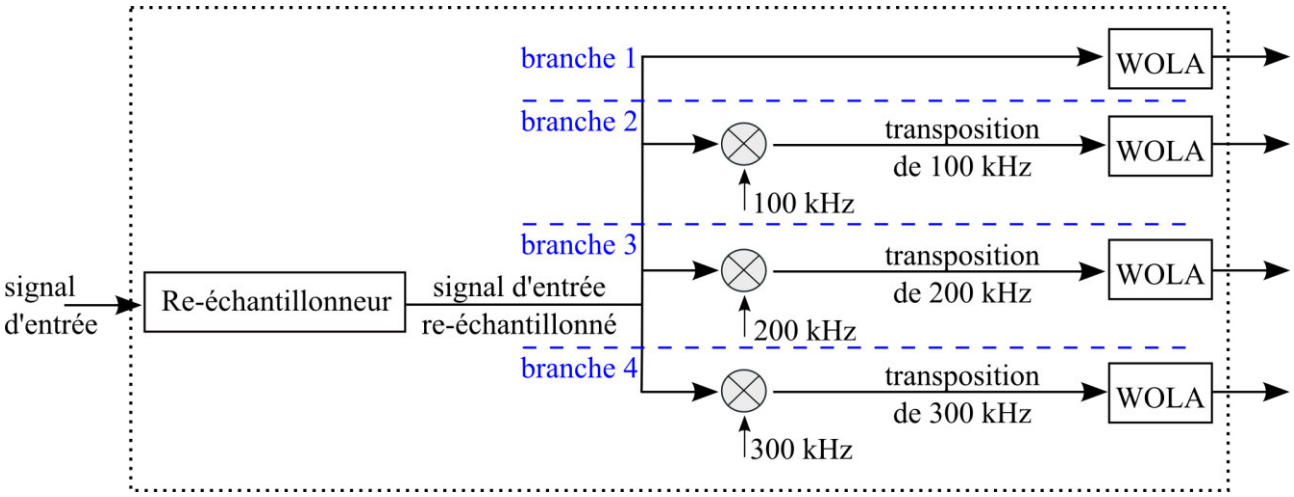


Figure 3.26 – Channelizeur final. Trois NCOs (non représentés sur la figure) génèrent les sinusoïdes de fréquences 100, 200 et 300 kHz

L'implémentation matérielle de ce channelizeur est présentée dans la section 5.1.4.

3.4.3 Channelization en LF, MF et HF

3.4.3.1 Architecture proposée

Cette architecture est bâtie sur l'idée que, l'un des moyens les plus efficaces pour opérer un filtrage à phase linéaire est d'utiliser un FIR. Rappelons que dans les bandes de radiodiffusion en dessous de 30 MHz, comme la DRM30 que nous étudions ici, la largeur de chaque canal est souvent très étroite (20 kHz) par rapport à la bande. Dans un tel contexte, comme nous l'avons vu précédemment, en utilisant les différentes méthodes existantes répertoriées dans la section 3.4.1.2, le nombre de coefficients requis pour un tel filtre tend à être très élevé. Afin d'utiliser efficacement les ressources que requiert ce filtre, nous proposons alors une architecture issue de la combinaison de plusieurs techniques de channelization existantes (figure 3.27). Il s'agit de WOLA, Goertzel et Per Channel. L'ensemble WOLA + Goertzel a pour objectif de générer des sous-bandes issues du signal à channelizer. Chaque sous-bande est 2 fois plus large que les canaux à extraire. Ensuite, avec une architecture de type Per Channel, on va extraire avec précision les canaux recherchés.

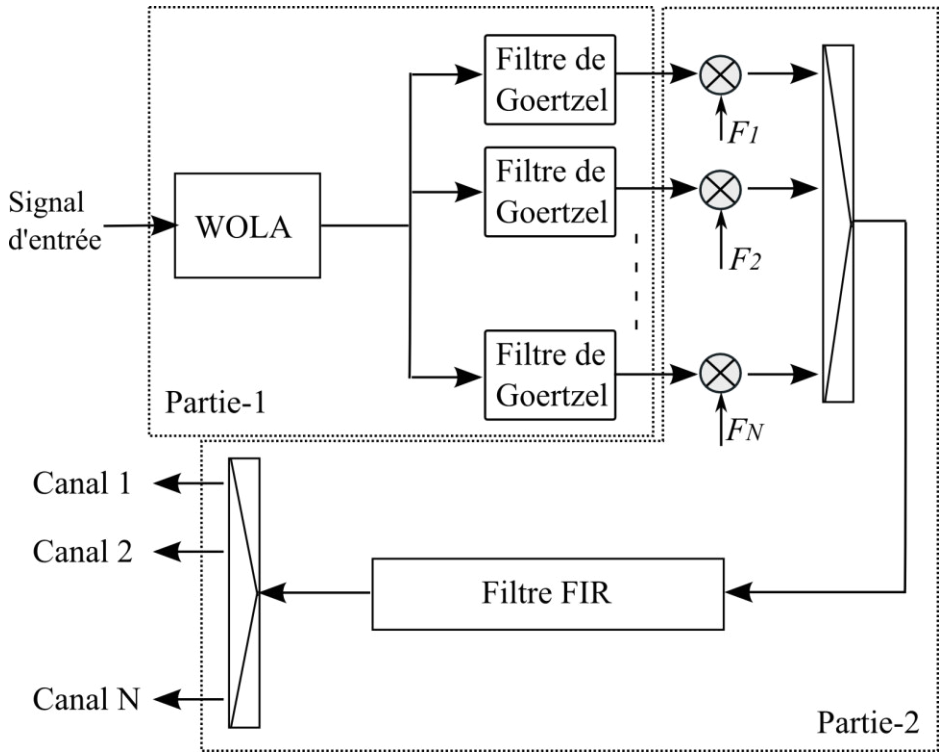


Figure 3.27 – Principe du channelizeur proposé. *Fi* représentent des signaux sinusoïdaux utiles pour extraire les canaux recherchés

3.4.3.1.1 Partie WOLA + Goertzel : Algorithme de Hentschel

On commence d'abord par implémenter le channelizeur proposé par Hentschel (voir la section 3.4.1.2.2) qui est un assemblage de WOLA et Goertzel. Comme précédemment, L_P est la longueur de la réponse impulsionnelle du filtre instancié par WOLA et K_P le nombre d'échantillons par bloc d'overlap [L03, WLW06]. On appelle aussi B_{IN} la largeur de la bande à channelizer.

Le concept de base du channelizeur de Hentschel veut que chaque sous-bande produite soit un des canaux recherchés. Ainsi, plus la résolution sur la position des canaux est fine, plus la résolution du bloc WOLA – c'est-à-dire le nombre K_P – est grande. Ainsi, par exemple, pour extraire d'une bande B_{IN} = 40 MHz des canaux dont les positions sont au kHz près, il faut $K_P = B_{IN}$ /1 kHz = 40000. Cela revient à implémenter 40000 multiplieurs (pour envoyer les 40000 échantillons pondérés, si possible, en parallèle au noyau FFT) ou bien réduire le nombre de multiplieurs au détriment de la bande passante maximale autorisée pour chaque canal (pour envoyer les échantillons en série au noyau FFT).

Pour pallier à ce problème, dans l'architecture proposée, la largeur de chaque sous-bande sélectionnée extraite par la partie 1 (figure 3.27) est fixée à 2 fois celle que les canaux recherchés. Le but recherché est que chaque sous-bande contiennent au moins un canal recherché. Reprenons l'exemple précédent et supposons chaque canal recherché large de 50 kHz. L'extraction des sous-

bandes (valant 2 × 50 = 100 kHz) dans ce cas fixe K_P à B_{IN}/100 kHz = 400. La position de l'ensemble WOLA – Goertzel dans l'architecture générale du channelizeur est donnée par la figure 3.27 – partie 1. Un descriptif plus détaillé de son implémentation est donné dans la figure 3.28, où l'architecture du bloc WOLA est celle de la figure 3.11 et l'architecture du bloc Goertzel est directement adaptée de ce qu'on peut trouver dans la littérature [H02, JL03].

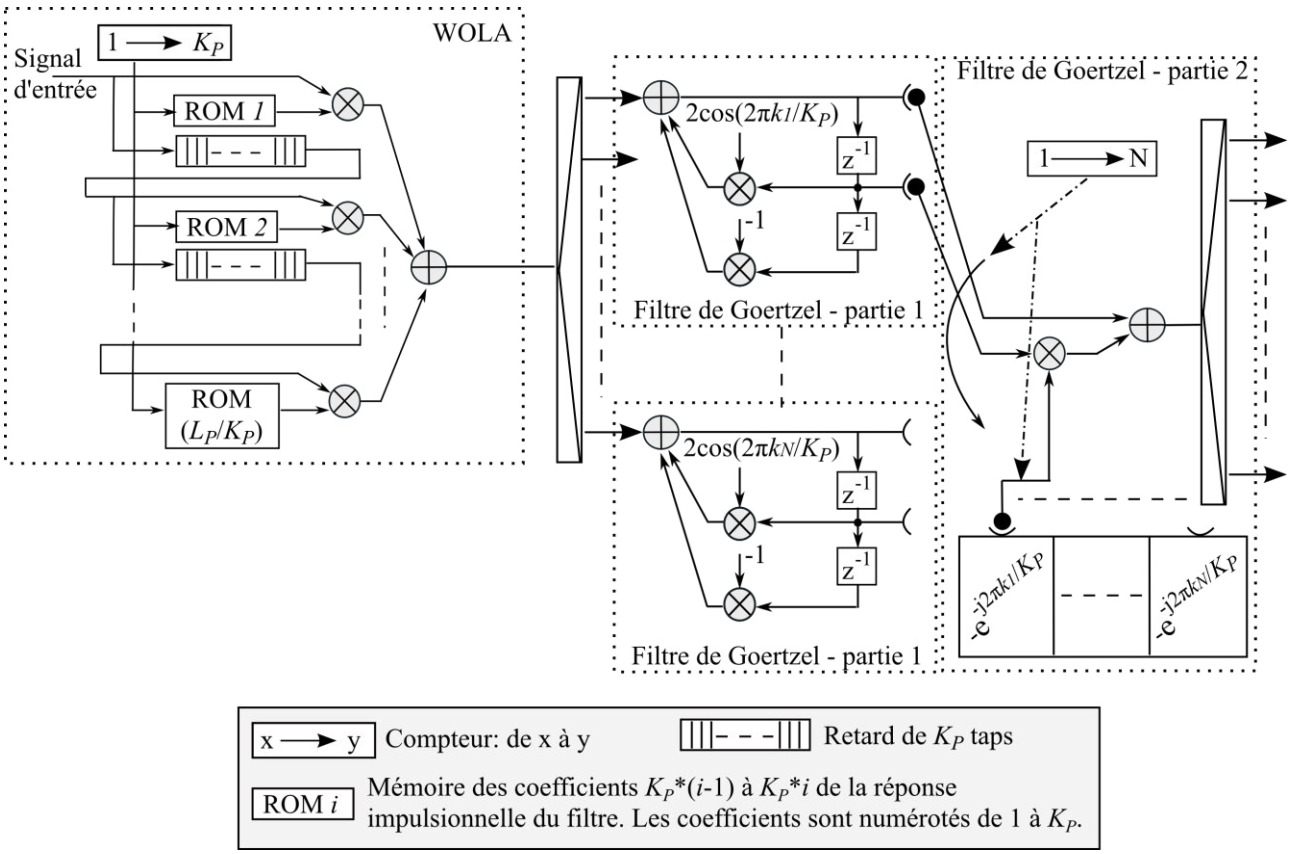


Figure 3.28 – Partie 1 du channelizeur proposé. *ki* est le numéro de la sous-bande extraite et *N* est le nombre de sous-bandes extraites. « Retard de K_P taps » représente une ligne à retard numérique de K_P cycles d'horloge

3.4.3.1.2 Partie "Per channel"

A ce niveau du procédé, on a extrait des canaux très réduits qui, chacun, contiennent un – ou éventuellement tout ou partie d'un deuxième – canal recherché. Il faut encore faire une opération qui va complètement isoler chaque canal recherché.

Pour cela, on va appliquer à chaque sous-bande précédente un filtre (nommé *Hg* et d'ordre *Lg*) dont la bande passante est la même que celle des canaux recherchés. Comme on avait déjà, grâce à WOLA, obtenu des canaux à largeur réduite, l'ordre de ce dernier filtre, à savoir *Lg*, sera lui aussi considérablement réduit. Par exemple, *Lg* = 115 suffit à prélever un signal avec une atténuation de

40 dB des bandes rejetées, avec un filtre de Kaiser et une largeur de bande de transition égale au dixième de celle du canal recherché.

Avant d'appliquer ce dernier filtre, le(s) canal(aux) contenu(s) dans chaque sous-bande est(sont) ramenée(s) à zéro-FI grâce au signal sinusoïdal émis par une NCO. Comme la position dans le spectre des canaux à extraire est connue *a priori*, il est facile de déterminer la fréquence de chaque signal sinusoïdal et de la fixer. La façon la plus évidente de faire cette transposition serait d'implémenter un NCO et un multiplieur par sous-bande. Cependant, pour réduire le nombre de multiplieurs, nous proposons de n'en utiliser qu'un seul pour toutes les sous-bandes (figure 3.29-a). Cela est possible par le fait que la fréquence d'échantillonnage des sous-bandes est faible (environ 10^{-2} F_S) et on peut donc cadencer le multiplieur à une fréquence suffisamment élevée pour que chaque sous-bande soit transposée à tour de rôle.

De même, afin de ne pas implémenter autant de filtres que de canaux, un seul filtre est implémenté pour tous les canaux. Ce filtre – que nous appelons « Filtre FIR Pipeliné » – est lui aussi cadencé à une fréquence suffisamment élevée pour que chaque canal soit filtré à tour de rôle grâce à des buffers qui alimentent chaque multiplieur (figure 3.29-b). La réalisation de cette version « pipelinée » du filtre est rendue envisageable grâce au fait que l'ordre du filtre est tel que $Lg \times$ (*fréquence d'échantillonnage des sous-bandes*) < F_S. Il est intéressant de remarquer que l'ensemble NCO + Filtre, constitue l'algorithme du « per channel » même si, dans notre cas, nous avons pipeliné le filtre afin d'en utiliser qu'un seul.

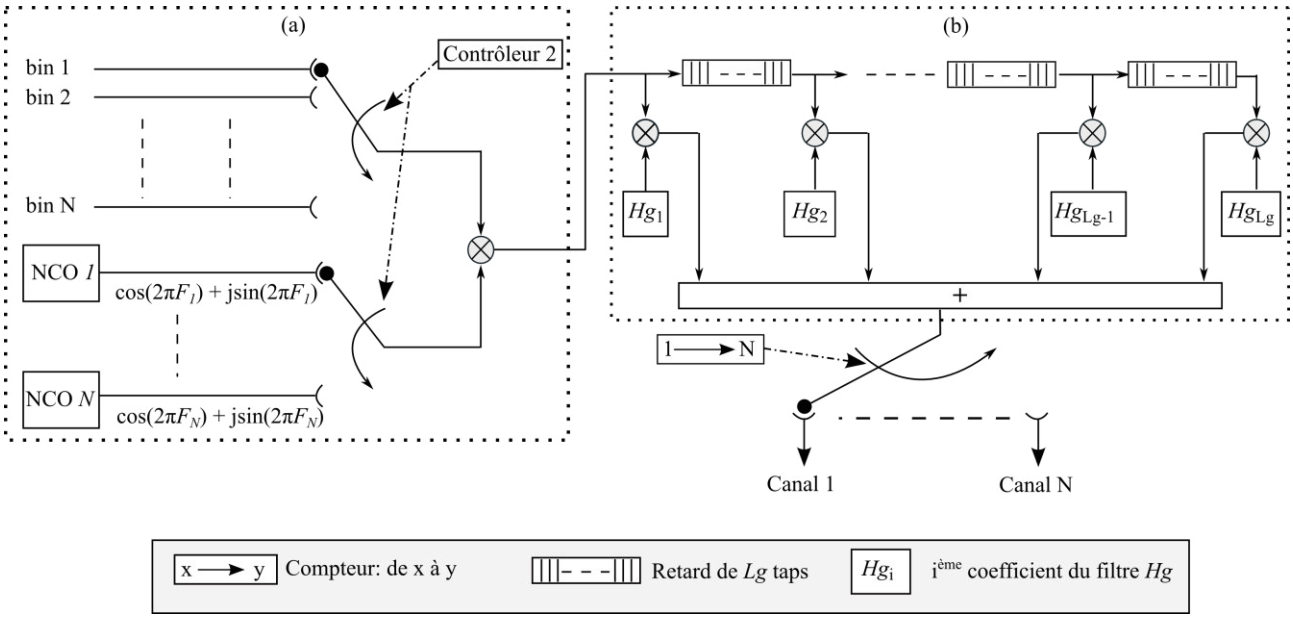


Figure 3.29 – Partie « Per channel » du channelizeur DRM proposé. a) transposition. b) Filtre FIR Pipeliné. « j » représente la racine carrée complexe de « -1 ». « Retard de *Lg* taps » représente une ligne à retard numérique de *Lg* cycles d'horloge. *Fi* représentent les fréquences des signaux sinusoïdaux utiles pour extraire les canaux recherchés

3.4.3.1.3 WOLA superposé

L'ensemble WOLA+Goertzel de l'architecture proposée mentionnée plus haut effectue un premier découpage en K_P bandes dont les positions possibles sont fixées et uniformément espacées. L'espacement de ces positions possibles étant F_S / K_P, cela revient à laisser, entre chaque bande découpée possible, un espace de la même largeur que la sous-bande découpée. Ainsi, il peut arriver que des canaux recherchés se situent (figure 3.30):

- soit dans l'espace entre 2 sous-bandes possibles,
- soit sur l'extrémité d'une sous-bande.

Si de tels canaux sont présents, on peut les extraire en rajoutant trois autres structures WOLA + Goertzel précédées chacune d'une NCO. Cette NCO génère un signal sinusoïdal qui sert à transposer en fréquence le signal d'entrée afin de permettre aux WOLA + Goertzel additionnels de se superposer au premier WOLA + Goertzel. On obtient alors un channelizeur constitué de 4 branches où chaque branche est un ensemble NCO + WOLA + Goertzel sauf dans la branche 1 où il n'y a pas de NCO. La figure 3.31 illustre clairement le channelizeur obtenu. En fixant la fréquence de chacune des 3 NCOs respectivement à $F_S / 4K_P$, $F_S / 2K_P$ et $F_S / 2K_P + F_S / 4K_P$, on peut extraire les canaux en question (figure 3.32).

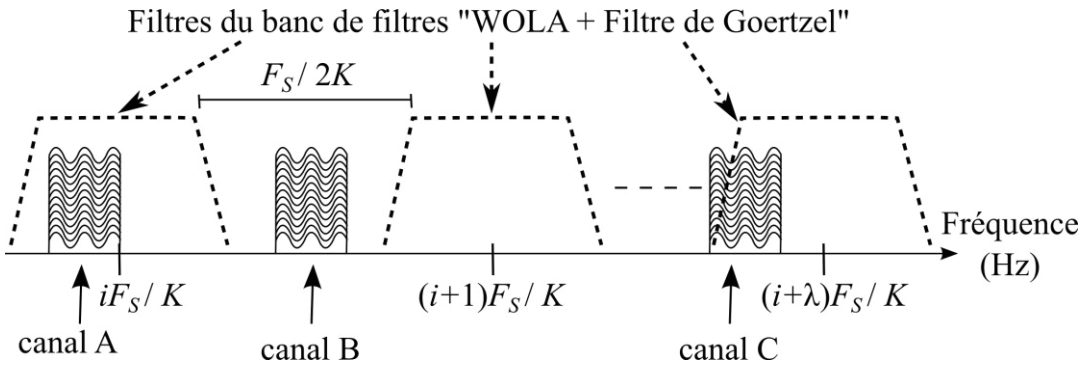


Figure 3.30 – Positions possibles des sous-bandes extraites par le banc de filtres « WOLA + Goertzel » et exemple de spectre d'un signal comportant quelques canaux à extraire. « i » et « λ » sont des entiers qui indiquent les centres des sous-bandes. Channel A est totalement inclus dans une sous-bande. Par contre, Channel B se trouve dans l'espace non couvert entre deux sous-bandes consécutives. Channel C quant à lui se situe à l'extrémité d'une sous-bande

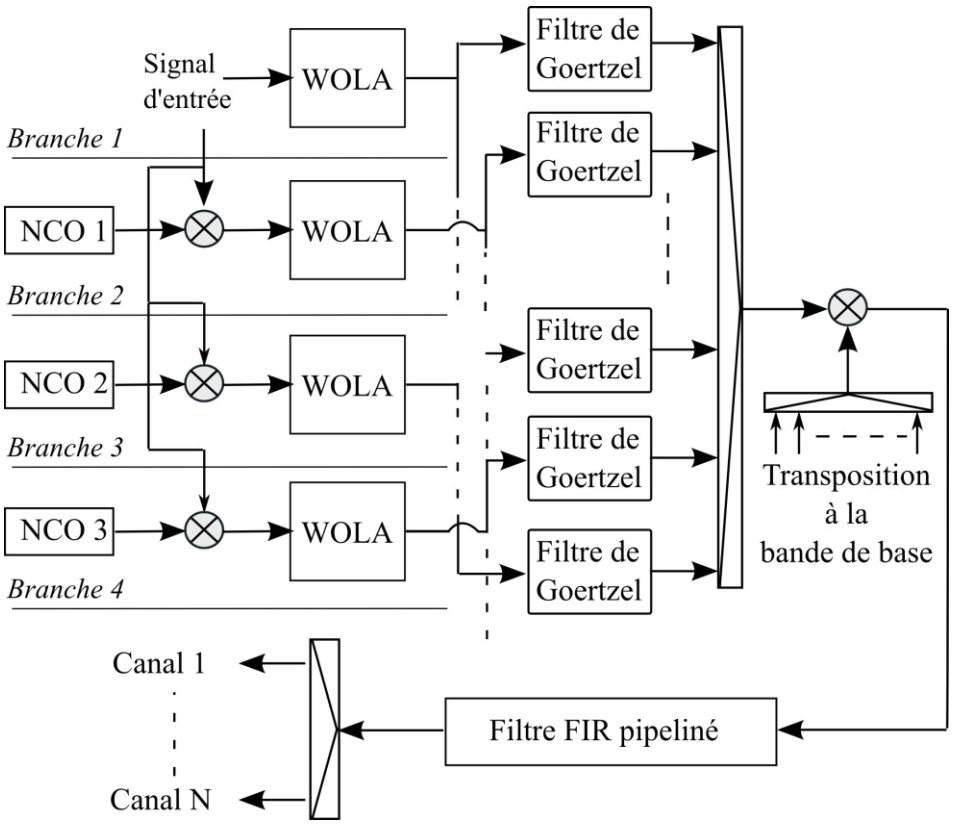


Figure 3.31 – Channelizeur complet avec ses 4 branches. Les NCOs 1, 2 et 3 génèrent respectivement les sinusoïdes de fréquences $F_S / 4K_P$, $F_S / 2K_P$ et $F_S / 2K_P + F_S / 4K_P$ (Hz). « Transposition à la bande de base » désigne l'ensemble des signaux générés par les NCOs de la figure 3.29-a

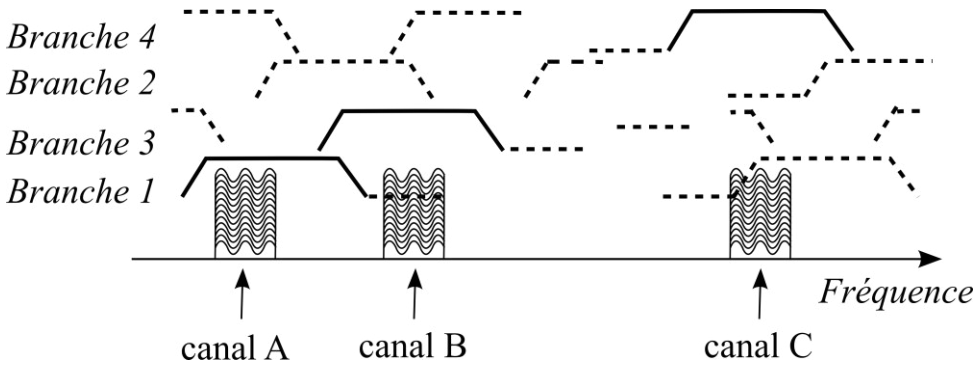


Figure 3.32 – 4 structures « WOLA + Goertzel » superposées. Chaque « branche i » indique les sous-bandes extraites par la branche correspondante. Canal A est correctement extrait par le premier « WOLA + Goertzel » mais pas Canal B et C. Par contre, le « WOLA + Goertzel » de la branche 3 extrait parfaitement Canal B et c'est le banc de filtre de la branche 4 qui extrait parfaitement Canal C

Dans tous les schémas descriptifs de l'architecture mentionnés en haut, les multiplieurs et les additionneurs opérent sur des nombres complexes. Cependant, certains composants – comme certaines catégories de FPGA – n'intègrent pas de modules de calculs complexes. Pour eux, il est

bien sûr possible de décomposer chaque opération complexe en opérations réelles, et d'adapter tous les précédents schémas en conséquence.

3.4.3.2 Comparaison avec d'autres architectures existantes

Nous nous proposons dans cette partie de comparer l'architecture proposée avec les autres channelizeurs mentionnés dans l'état de l'art. Notre critère de comparaison sera le nombre de multiplications par instant d'horloge système, le « multiplication rate », utilisé par chacun des channelizeurs. Les tableaux suivants (Tableaux 3.4 et 3.5) recensent le multiplication rate de chacune de ces méthodes. Le tableau 3.4 répertorie le nombre de multiplications complexes. Des informations sur son remplissage sont données ci-après pour chaque méthode de l'état de l'art. En complément, le tableau 3.5 répertorie le nombre de multiplications réelles équivalent. Ce nombre est déduit de la décomposition des multiplications complexes en série de multiplications réelles. B_{OUT} représente la largeur de chaque canal à extraire, L représente le nombre de coefficients nécessaire pour extraire un canal de largeur B_{OUT} de la bande B_{IN} et N est le nombre de canaux à extraire :

- *La méthode proposée* : Les tableaux 3.4 et 3.5 montrent les « multiplication rates » de la méthode proposée dans 2 cas. Le premier cas c'est celui où tous les canaux requis peuvent être extraits par la branche 1 seulement. Le calcul des « multiplication rates » est alors le même que le nombre de multiplieurs des figures 3.28 et 3.29 réunies. Le second cas (worst case design), à savoir que les 4 branches doivent être implémentées pour extraire les N canaux. Le « multiplication rate » est déduit de la figure 3.31.
- *Per channel* : Le « multiplication rate » de cette approche peut facilement se calculer compte tenu de la simplicité de la méthode. C'est le nombre de multiplieurs utilisés par les DDCs et par les filtres FIR (figure 3.12).
- *Goertzel* : Ici aussi, le « multiplication rate » consiste à comptabiliser le nombre de multiplications nécessaires pour pondérer les échantillons d'entrée par les L coefficients du filtre (figure 3.13), ainsi que tous les multiplieurs instanciés dans les filtres Goertzels.
- *TPFT* : Le nombre de multiplieurs affiché de la méthode TPFT est déduit du schéma explicatif standard de la méthode [L03] adapté au cas où les canaux à extraire sont tous de même largeur (figure 3.14). Aussi, L_1 représente le nombre de coefficient requis pour scinder chaque sous-bande issue de l'étape précédente en 2 sous-bandes. L_2 est le nombre de coefficients requis pour effectuer la dernière étape de channelization dans la TPFT. C'est l'étape appelée « filtre final » qu'on retrouve dans la figure 3.14 et dans la littérature

[L03] sous le nom de « final shaping filter ». C'est donc le filtre qu'il faut pour extraire B_{OUT} de B_{INTER}, où B_{INTER} est la bande minimale supérieure à B_{OUT} issue de la division répétée de B_{IN} par 2.

- *FRM* : Le « multiplication rate », qui est ici le nombre de multiplieurs instanciés, est simplement déduit de la figure 3.18.
- *CD* : Ici, on suppose être dans le cas idéal, c'est-à-dire que les canaux à extraire correspondent parfaitement aux contraintes imposées par cette méthode (voir la section 3.4.1.2.5).
- *Channelizeur de Wajih et Gordon* : L'implémentation matérielle de ce channelizeur n'ayant pas été traitée dans l'article référence [WG04], nous ne nous étendrons pas dessus.
- *Darak, Vinod et Lai* : Pour ce channelizeur, qui est le dernier de la comparaison, on suppose encore que les canaux à extraire sont dans la configuration idéale, c'est-à-dire que $NS = 1$ et qu'il suffit d'implémenter un filtre de masquage modal et déduire les autres par CD. On appelle L_{Ha2} la longueur du filtre modal *Ha* utilisé pour ce channelizeur, L_{HaM2} et L_{HcM2} sont les longueurs des filtres de masquage.

	Méthode proposée		Per Channel	Goertzel	TPFT
	branche 1	4 branches			
Filtres FIR	L_P/K_P+Lg	$4.L_P/K_P+Lg$	$N.L$	L	$2.L_1.\text{floor}(\log(B_{IN}/B_{OUT})/\log(2))+L_2$
DDCs	1	4	N		$2.\text{floor}(\log(B_{IN}/B_{OUT})/\log(2))+1$
Autres	$N+1$	$N+1$		$N.L+1$	
Total	$L_P/K_P + Lg + N+2$	$4.L_P/K_P + Lg + N+5$	$N.(L+1)$	$L.(N+1)+1$	$2.(L_1+1).\text{floor}(\log(B_{IN}/B_{OUT})/\log(2)) +L_2 +1$

	FRM	CD	Wajih et Gordon	Darak, Vinod et Lai
Filtres FIR	$L_{Ha}+(L_{HaM}+L_{HcM}).N$	L	Implémentation matérielle non traitée	$L_{Ha2}+L_{HaM2}+L_{HcM2}$
DDCs	$2.N$	N		N
Autres				
Total	$L_{Ha}+(2+L_{HaM}+L_{HcM}).N$	$L + N$		$L_{Ha2}+L_{HaM2}+L_{HcM2} + N$

Tableau 3.4 – Nombre de multiplications complexes requises par méthode. La fonction « floor » désigne la partie entière. « Filtres FIR » désigne les multiplications liées aux filtres FIR instanciés tandis que « DDCs » désigne les multiplications entre les signaux à transposer et les sinusoïdes issues de NCOs

	Méthode proposée		Per Channel	Goertzel	TPFT
	branch 1	4 branches			
Filtres FIR	$2.L_P/K_P+4.Lg$	$8.L_P/K_P+4.Lg$	$2.N.L$	$2.L$	$4.L_1.\text{floor}(\log(B_{IN}/B_{OUT})/\log(2)) + 2.L_2$
DDCs	4	16	$4.N$		$8.\text{floor}(\log(B_{IN}/B_{OUT})/\log(2))+4$
Autres	$2.N + 4$	$2.N+4$		$2.N.L + 4$	
Total	$2.L_P/K_P + Lg + 2.N+8$	$8.L_P/K_P + Lg+2.N+20$	$N.(2.L+4)$	$2.L(1+N) + 4$	$4.(L_1+2).\text{floor}(\log(B_{IN}/B_{OUT})/\log(2)) +2.L_2 +4$

	FRM	CD	Wajih et Gordon	Darak, Vinod et Lai
Filtres FIR	$2.L_{Ha}+(L_{HaM}+L_{HcM}).2.N$	$2.L$	Implémentation matérielle non traitée	$2.(L_{Ha2}+L_{HaM2}+L_{HcM2})$
DDCs	$8.N$	$4.N$		$4.N$
Autres				
Total	$2.L_{Ha}+(4+L_{HaM}+L_{HcM}).2.N$	$2.L + 4.N$		$2.(L_{Ha2}+L_{HaM2}+L_{HcM2}+2.N)$

Tableau 3.5 – Nombre de multiplications réelles requises par méthode. La fonction « floor » désigne la partie entière. « Filtres FIR » désigne les multiplications liées aux filtres FIR instanciés tandis que « DDCs » désigne les multiplications entre les signaux à transposer et les sinusoïdes issues de NCOs

Cas du DRM30

Prenons le cas de la channelization de canaux de type DRM30. On voudrait fixer la rejection du filtrage à 40 dB minimum pour bien isoler les canaux recherchés et garantir une bonne démodulation après channelization. Pour une comparaison plus juste entre les différentes méthodes, on choisit d'utiliser un même de filtres FIR pour toutes ces méthodes : « Kaiser». Il n'y a que pour le TPFT qu'on prendra un autre type de filtres. C'est une méthode qui exige que ses filtres FIR puissent compenser la perte de gain de ses filtres CICs. B_{OUT} ici vaut 20 kHz (c'est le

mode DRM qui utilise la bande la plus large). On prend B_{IN} = 30 MHz. Les valeurs des paramètres suivants permettent de répondre aux critères :

- Méthode proposée : L_P = 16384; K_P = 1024; Lg = 80; F_S = 81.92 MHz.
- Per channel, Goertzel et CD : L = 60000.
- TPFT : On fixe la rejection des différents étages consécutifs à 45 dB pour limiter le bruit dû à la longue chaîne de traitement. Pour cela, des filtres CIC de 15 sections et retards de 1-tap chacun sont utilisables. Ils peuvent être compensés avec des filtres « equiripple » de longueur L_1 = 247. On supposera dans notre cas qu'on n'a pas besoin du « final filter ». On prend alors L_2 = 0.
- FRM : Les paramètres M = 100; L_{Ha} = 100; L_{HaM} = 15000; L_{HcM} = 14650 permettent de répondre au problème posé.
- Darak, Vinod et Lai : Etant donné qu'on fait une channelization avec largeurs de bande égales, on ne tire pas pleinement profit des originalités (mentionnées dans la section 3.4.1.2.7) que présente cette structure par rapport au FRM. Des paramètres identiques à ceux du FRM sont alors utilisables : M = 100; D_C = 1 ; L_{Ha2} = 100; L_{HaM2} = 15000; L_{HcM2} = 14650.

Les figures 3.33 et 3.34 montrent le nombre de multiplications requises par chacune de ces méthodes pour un nombre de canaux compris entre 4 et 100.

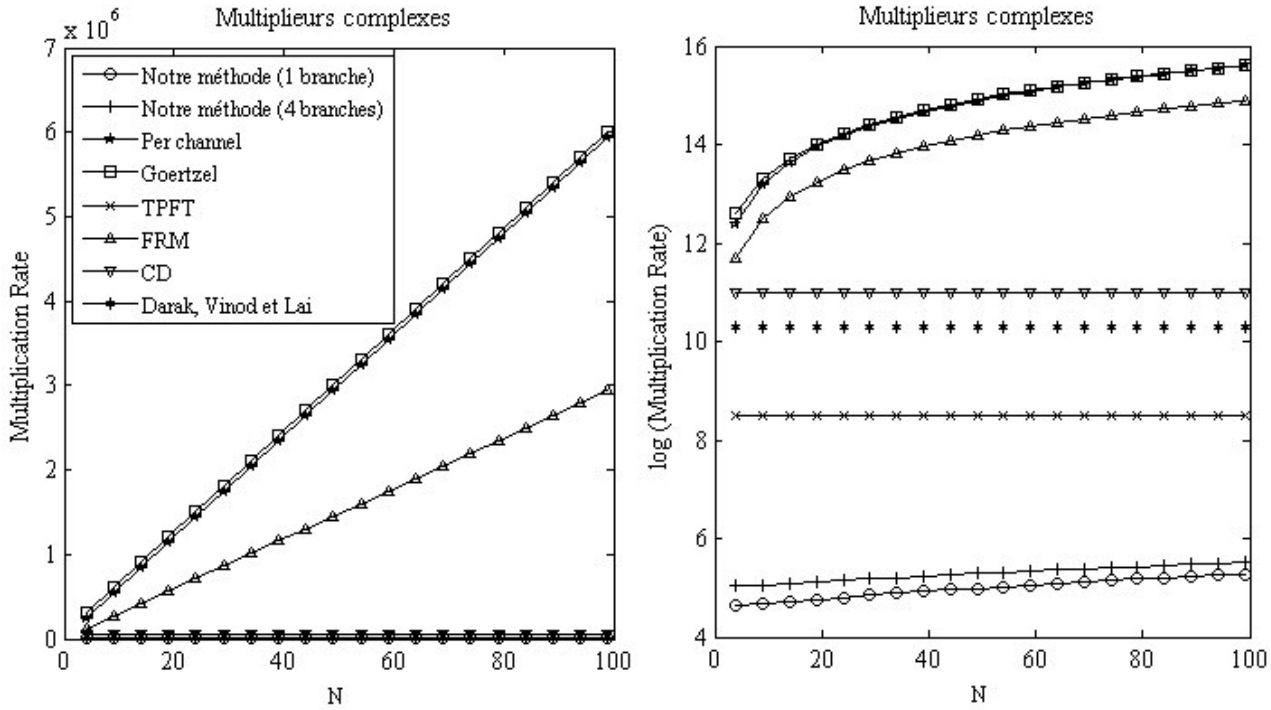


Figure 3.33 – Multiplication rate complexe pour extraire des canaux de 20 kHz dans une bande de 30 MHz. A gauche, les valeurs du multiplication rate sont exactes. A droite, il s'agit du logarithme népérien du multiplication rate

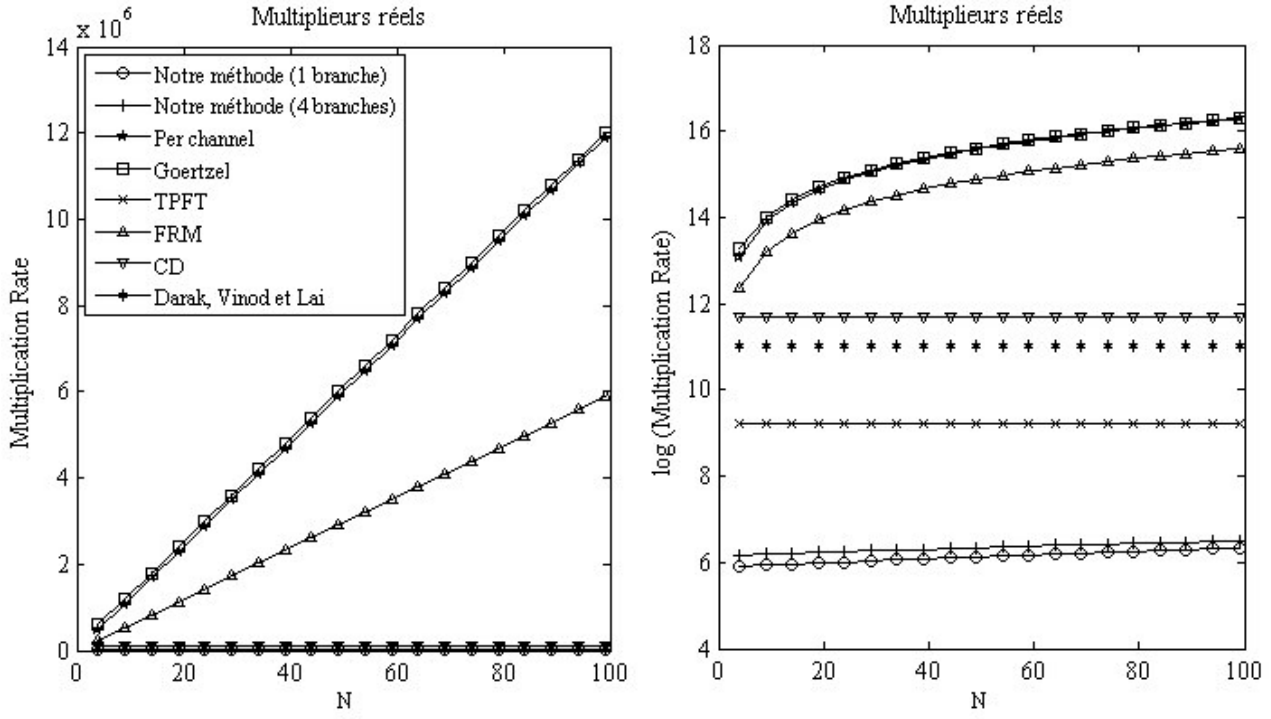


Figure 3.34 – Multiplication rate réel pour extraire des canaux de 20 kHz dans une bande de 30 MHz. A gauche, les valeurs du multiplication rate sont exactes. A droite, il s'agit du logarithme népérien du multiplication rate

Dans les 2 figures 3.33 et 3.34 ci-dessus, les courbes de droite sont les logarithmes népériens des valeurs calculées. Cela permet de mettre en évidence la différence d'utilisation de ressources entre les différentes méthodes. A part pour la technique Goertzel, le « multiplication rate » des

autres techniques est très proche du nombre de multiplieurs implémentées matériellement. Comme le montrent les figures 3.33 et 3.34, dans le cas de la channelization de canaux DRM30, l'architecture proposée dans cet article utilise grossièrement le moins de multiplieurs dans le composant. Cela rend notre architecture beaucoup plus intéressante en termes de multiplieurs dans beaucoup de cas.

Un exemple implémentation du channelizeur DRM30 sur le FPGA et les résultats de validations obtenus sont présentés respectivement dans les sections 5.2.3 et 5.2.6.

3.4.4 Conclusion sur la channelization

Nous avons vu les différentes techniques de channelization existantes et analysé leurs caractéristiques. A partir de plusieurs d'entre elles, nous avons développé des channelizeurs adaptés pour la FM et la DRM30. La section 5 contient les performances que nos récepteurs ont obtenues avec ces channelizeurs. La prochaine étape de notre périple consistera à développer la démodulation simultanée de tous les canaux « channelizés ». C'est le sujet de la section 3.5.

3.4.5 Bibliographie

[ALTERA13] Stratix V Device Overview. *ALTERA ®. 101 Innovation Drive. San Jose. CA 95134, USA*. 06 May 2013. <http://www.altera.com/literature/hb/stratix-v/stx5_51001.pdf>. Accessed 18th Nov. 2013.

[DVL12] Sumit Jagdish Darak, Achutavarrier Prasad Vinod, Edmund M.-K. Lai, *« A Low Complexity Reconfigurable Non-uniform Filter Bank for Channelization in Multi-standard Wireless Communication Receivers », Journal of signal processing systems*, 2012, vol. 68, no. 1, pp. 95-111.

[H02] T. Hentschel, *« Channelization for software defined base-stations », Annales des télécommunications*, May 2002, vol. 57, n° 5-6, pp. 386-420. DOI : 10.1007/BF02995169

[HDR03] F. J. Harris, C. Dick, and M. Rice, "Digital receivers and transmitters using polyphase filter banks for wireless communications," *IEEE Trans. Microwave Theory and Techniques*, vol. 51, no. 4, pp. 1395-1412, Apr. 2003.

[JL03] Jacobsen, E.; Lyons, R., "The sliding DFT," *Signal Processing Magazine, IEEE* , vol.20, no.2, pp.74,80, Mar 2003

[L03] J. Lillington, “Slice and dice chunks of radio spectrum,” *Wireless Systems Design*, vol. 8, p. 39, Nov. 2003.

[L86] Lim, Y. C. (1986). « *Frequency-response masking approach for the synthesis of sharp linear phase digital filters ». IEEE transactions on circuits and systems,* 1986

[MVLO11] R. Mahesh, A. P. Vinod, Edmund M-K. Lai, A, Omondi, *« Filter Bank Channelizers for Multi-Standard Software Defined Radio Receivers » J Sign Process Syst*, 2011.

[MVMP08] Mahesh, R.; Vinod, A.P.; Moy, C.; Palicot, Jacques, "A Low Complexity Reconfigurable Filter Bank Architecture for Spectrum Sensing in Cognitive Radios," *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, vol., no., pp.1 – 6, 15-17 May 2008

[TRDD12] Tietche, B.H.; Romain, O.; Denby, B.; Dieuleveult, F.D., "FPGA-based simultaneous multichannel fm broadcast receiver for audio indexing applications in consumer electronics scenarios," *IEEE Transactions on Consumer Electronics*, vol.58, no.4, pp.1153, 1161, Nov. 2012, doi: 10.1109/TCE.2012.6414980

[WG04] Abu-Al-Saud, W.A.; Stuber, G.L., "Efficient wideband channelizer for software radio systems using modulated PR filterbanks," *IEEE Transactions on Signal Processing*, vol.52, no. 10, pp. 2807-2820, Oct. 2004

[WLW06] H. Wang, Y. Lu, and X. Wang, “Channelized receiver with WOLA filterbank,” *IEEE Int. Conf. Radar*, *CIE’06*, Shanghai, China, Oct. 16 – 19, 2006, pp.1-3.

3.5 La démodulation

3.5.1 Démodulation massivement parallèle du standard FM

Pour ce qui est du standard FM, ici encore, l'objectif est de démoduler simultanément toutes les stations. Comme on ne sait pas à ce stade quels canaux issus de la FFT transportent des stations, l'approche adoptée sera de démoduler simultanément tous les canaux issus de la FFT.

Comme le channelizeur FM est compose de 4 branches entrelacées, où chaque branche extrait 128 canaux espacés uniformément avec un intervalle de 400 kHz (dont 64 sont indépendants), le nombre de canaux distincts que le channelizeur extraira sera 4×128/2 = 256. Ces canaux sont centrés aux fréquences 87.0 + i×0.1 (MHz), où $i \in \{0, 1, \ldots, 255\}$. Le banc de filtres du channelizeur couvre donc la plage de 87 à 112 MHz. Etant donné qu'il y a potentiellement 256 canaux à démoduler, deux implémentations du système de démodulation sont possibles:

- Générer 256 instances du système de démodulation qui travailleront en parallèle
- Pipeliner le système de telle sorte qu'un module unique suffise à démoduler tous les canaux d'une branche donnée.

Le premier choix est évidemment le plus simple pour notre application, vu que les FPGAs sont adaptés aux architectures parallélisées. Malheureusement, le coût du point de vue des registres logiques ou blocs DSPs atteint rapidement les limites des ressources du FPGA utilisées ici. Cette difficulté nous a conduit à choisir la seconde alternative.

Trois célèbres techniques de démodulation FM appropriées aux FPGAs sont résumées dans [RPN09]. Il s'agit de l'approche « feedback (PLL) », l'approche « arctangente-différencier » (« arctangent-differentiate approach », en anglais) et l'approche « différencier-diviser » (« differentiate-divide approach », en anglais). Leurs équations caractéristiques sont respectivement les équations (3.6), (3.7) et (3.8) et elles sont illustrées par la figure 3.35. Pour l'approche « feedback », le fait que la structure présentée contienne plusieurs boucles fait qu'il est difficile de réaliser une version pipelinée de ce système. Par conséquent, restreignons notre attention aux deux autres approches.

$$Y = K_1 + K_2 \int (I \sin(\int Y) + Q \cos(\int Y)) \tag{3.6}$$

$$Y = \mathrm{d}(Arctan(-Q/I))/\mathrm{dt} \tag{3.7}$$

$$Y = (I(\mathrm{d}Q/dt) - (\mathrm{d}I/dt)Q)/(I^2+Q^2) \tag{3.8}$$

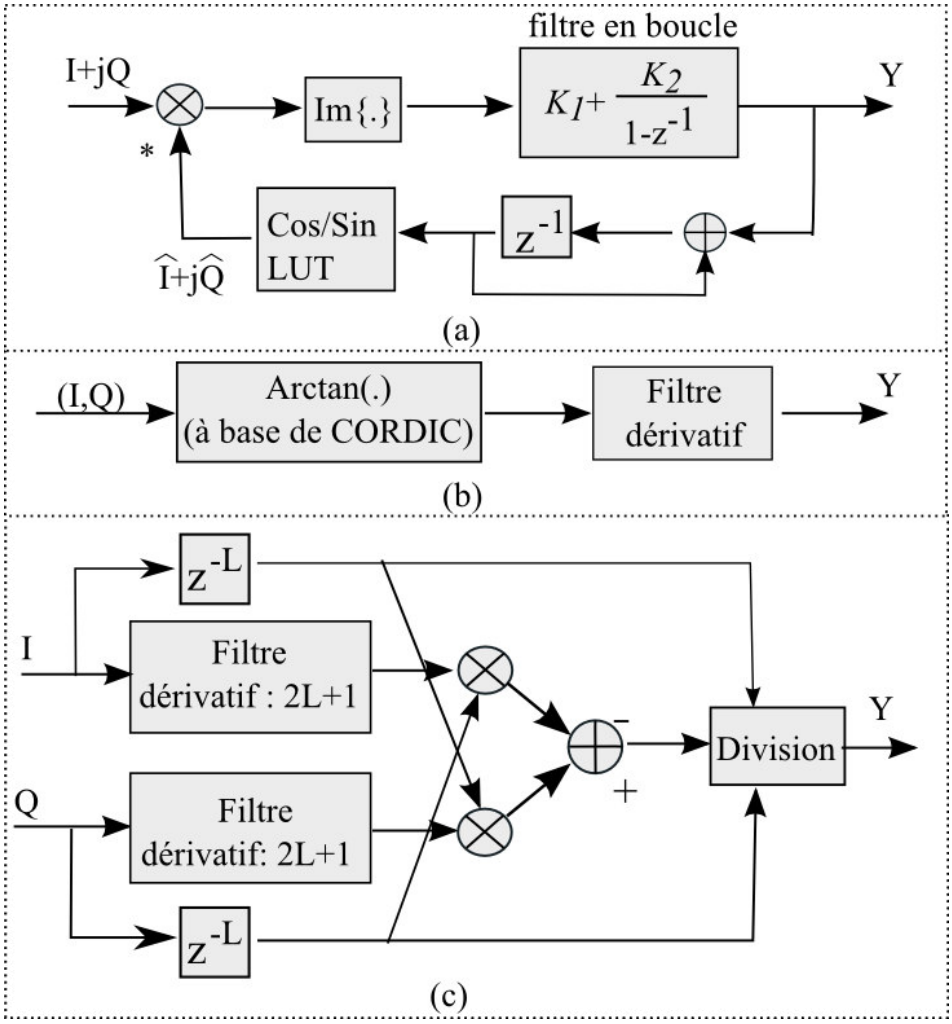


Figure 3.35 – 3 systèmes populaires de démodulation de la FM. Source modifiée : [RPN09]. a) Démodulateur de fréquence « Feedback ». K_1 et K_2 sont des constantes qui déterminent le filtre en boucle. "*" représente le conjugué complexe. « LUT » est une table de valeurs précalculées. b) Démodulateur « arctangente-différencier» (Arctangent-differentiate system). Le calcul de l'arctangente est fait par l'algorithme CORDIC [V59], [W71]. c) Démodulateur « différencier-diviser » (Differentiate-divide system)

L'objectif est de construire un démodulateur pipeliné et les figures 3.35-b et 3.35-C indiquent qu'un filtre dérivatif doit être utilisé. Il est clair que plus le filtre implémenté est simple, plus le pipelining sera facile à mettre en œuvre. Le moyen le plus direct de différencier est de simplement soustraire les échantillons consécutifs deux par deux. Cependant, avant d'adopter une structure particulière, on doit examiner les performances des différentes options.

3.5.1.1 Simulations avec Matlab

Le channelizeur décrit dans la section 3.4.2 extrait des canaux sortant à 400 kHz. La figure 3.36 montre une simulation de la démodulation d'un chirp linéaire modulé en fréquence et échantillonné à 400 kHz en utilisant les deux méthodes : « différencier-diviser » et « arctangente-différencier ».

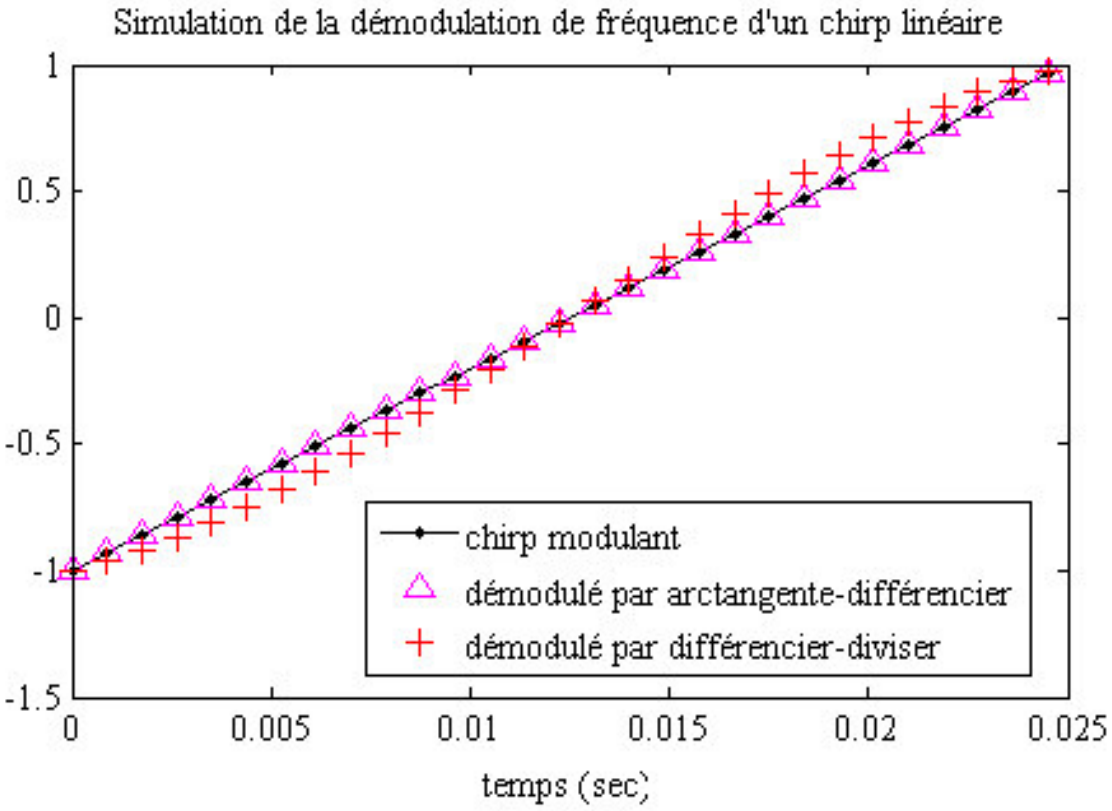


Figure 3.36– Simulation de la démodulation d'un chirp linéaire par les deux systèmes de démodulation, avec une déviation fréquentielle de 75 kHz. Le signal modulant, une rampe, est linéaire et varie de -1 à 1 durant la toute la simulation. Les résultats sont normalisés à l'intervalle [-1; 1] afin de les comparer aisément au signal modulant

La figure montre que la démodulation « différencier-diviser» dans sa forme la plus simple donne de mauvais résultats parce que le différentiateur deux-par-deux n'est pas une approximation de la dérivée suffisamment bonne aux fréquences qui nous intéressent. En effet, lorsque la fréquence d'échantillonnage est proche de la déviation fréquentielle maximale, comme dans notre application, deux échantillons consécutifs sont trop espacés dans le temps pour faire une bonne approximation de la dérivée (certaines simulations avec une fréquence d'échantillonnage bien plus élevée, 50 MHz par exemple, permettent d'obtenir de bons résultats avec l'architecture « différencier-diviser »). Des architectures « différencier-diviser » utilisant des filtrés dérivatifs plus élaborés ont été aussi testés, mais avec des résultats décevants. Pour obtenir de bonnes performances avec un tel différentiateur, il faut probablement utiliser un filtre très sophistiqué, ce

qui rend le pipelining difficile à mettre en œuvre. Ces raisons nous ont conduits à choisir la méthode « arctangente-différencier » pour notre application.

3.5.1.2 Structure du démodulateur pipeline « arctangente-différencier »

Le bloc de calcul de l'arctangente illustré dans la figure 3.35-b est supposé prendre en charge les 4 quadrants. Dans la pratique, cependant, un bloc de calcul CORDIC évalue la valeur d'un angle du premier quadrant, c'est-à-dire un angle de l'intervalle [0, π/2], en calculant la fonction *arctan(-Q/I)*. Dans notre implémentation, le bloc CORDIC est précédé par un bloc qui transpose premièrement tout angle à l'intervalle [0, π/2]. Ce bloc, que nous avons appelé algorithme « 4 quadrants à 1 quadrant », assigne un signal appelé "signal de contrôle" qui contient le numéro du quadrant original de l'angle traité. Soient (I_{IN}, Q_{IN}) et (I_{OUT}, Q_{OUT}) respectivement l'entrée et la sortie du bloc « 4 quadrants à 1 quadrant ». L'algorithme fonctionne comme décrit dans le tableau 3.6:

Si $I_{IN} \geq 0$ et $Q_{IN} \geq 0$ alors *(l'angle appartient au quadrant 1,[0, π/2])*
    $I_{OUT} = I_{IN}$; $Q_{OUT} = Q_{IN;}$
    control signal = 1
sinon si $I_{IN} < 0$ et $Q_{IN} \geq 0$ alors *(quadrant 2,[π/2, π]. On calcule : angle-π/2)*
    $I_{OUT} = I_{IN}$; $Q_{OUT} = - Q_{IN}$
    control signal = 2
sinon si $I_{IN} < 0$ et $Q_{IN} < 0$ alors *(quadrant 3,[π, 3π/2]. On calcule : angle-π)*
    $I_{OUT} = - I_{IN}$; $Q_{OUT} = - Q_{IN}$
    control signal = 3
else *(quadrant 4,[3π/2, 2π]. On fait angle-3π/2)*
    $I_{OUT} = - I_{IN}$; $Q_{OUT} = Q_{IN}$
    control signal = 4
fin si

Tableau 3.6 – L'algorithme « 4 quadrants à 1 quadrant »

Une fois que l'angle devient un angle du premier quadrant, le bloc CORDIC calcule *arctan(-Q/I)*, produisant ainsi la valeur de l'angle dans [0, π/2]. L'angle calculé est alors envoyé au bloc suivant, « 1 quadrant à 4 quadrants » (figure 3.37), qui ramène l'angle calculé à son quadrant original comme décrit dans le tableau 3.7.

Si control signal = 1 alors $ANG_{OUT} = ANG_{IN}$ sinon si control signal = 2 alors $ANG_{OUT} = ANG_{IN} + \pi/2$ sinon si control signal = 3 alors $ANG_{OUT} = ANG_{IN} + \pi$ sinon $ANG_{OUT} = ANG_{IN} + 3\pi/2$ fin si

Tableau 3.7 – L'algorithme « 1 quadrant à 4 quadrants ». ANG_{IN} et ANG_{OUT} sont respectivement l'entrée et la sortie du bloc

Le filtre dérivatif de la figure 3.35-b utilise une opération dite « unwrap » pour assurer la continuité de la phase. En effet, le bloc arctangente produit des angles entre 0 et 2π, et par conséquent des discontinuités entre deux angles consécutifs sont possibles. Par exemple, la dérivée des deux angles consécutifs $5\pi/2$ et $3\pi/4$, qui devrait donner $5\pi/2-3\pi/4=7\pi/4$, donnera plutôt $\pi/2-3\pi/4=-\pi/4$ parce que le calcul de l'arctangente associe $5\pi/2$ à $\pi/2$. Bien qu'ils représentent le même angle sur le cercle trigonométrique unitaire, les valeurs de leur phase sont différentes. Le bloc "unwrap" offre une solution à ce problème. Soient $Y_{1,N}$ et $Y_{1,\,N-1}$ deux angles consécutifs produits par le bloc arctangente. L'algorithme « unwrap » est décrit dans le tableau 3.8. Il utilise le fait que la différence entre deux angles consécutifs sera toujours inférieure à π, vu que le signal est échantillonné à 400 kHz et que la déviation maximale du signal FM considéré ici est seulement de 100 kHz:

Si $Y_{1,N} >= \pi$	
alors si $Y_{1,\,N-1} >= (Y_{1,N} - \pi)$ alors $Y = Y_{1,N} - Y_{1,\,N-1}$ sinon $Y = Y_{1,N} - 2\pi - Y_{1,\,N-1}$ fin si	sinon si $Y_{1,N} <= (Y_{1,N} + \pi)$ alors $Y = Y_{1,N} - Y_{1,\,N-1}$ sinon $Y = Y_{1,N} + (2\pi - Y_{1,\,N-1})$ sinon si
fin si	

Tableau 3.8 – L'algorithme « unwrap »

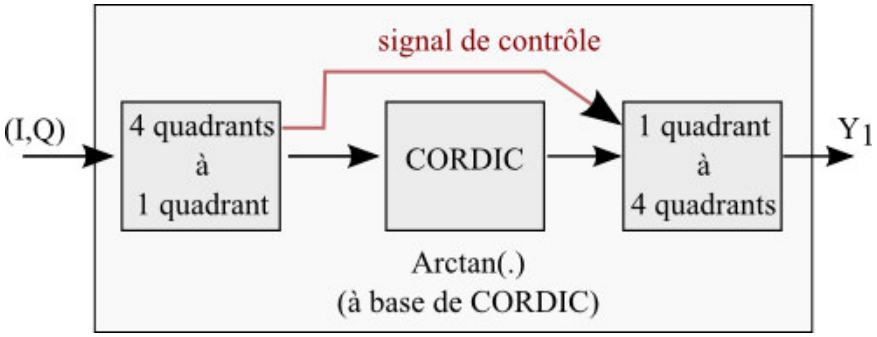


Figure 3.37 – Calcul de l'arctangente basé sur le CORDIC. Dans la figure, « signal de contrôle » indique au bloc « 1 quadrant à 4 quadrants » à quel quadrant l'angle généré par le bloc CORDIC appartient

L'algorithme CORDIC est itératif. Dans un système sans contraintes de temps tel qu'une calculatrice, on peut effectuer autant d'itérations qu'on le souhaite jusqu'à obtenir la précision voulue. Cependant, dans un système temps réel tel qu'un design basé sur FPGA, un nombre maximal d'itérations doit être fixé, en fonction de la quantité de ressources matérielles disponibles. Notre démodulateur pipeliné bénéficie de cela. En effet, comme il y'a un nombre fixe d'itérations, l'algorithme CORDIC est décomposé de manière à ce que chaque itération devienne un bloc à part entière, comme le montre la figure 3.38. De cette manière, toutes les paires d'échantillons I-Q qui arrivent au bloc CORDIC sont traitées indépendamment, ce qui permet d'éviter les retards peu importe le canal auquel elles appartiennent. En conséquence, l'angle correspondant est calculé à partir de chaque paire d'échantillons sans qu'aucun retard ne soit subi dans le processus.

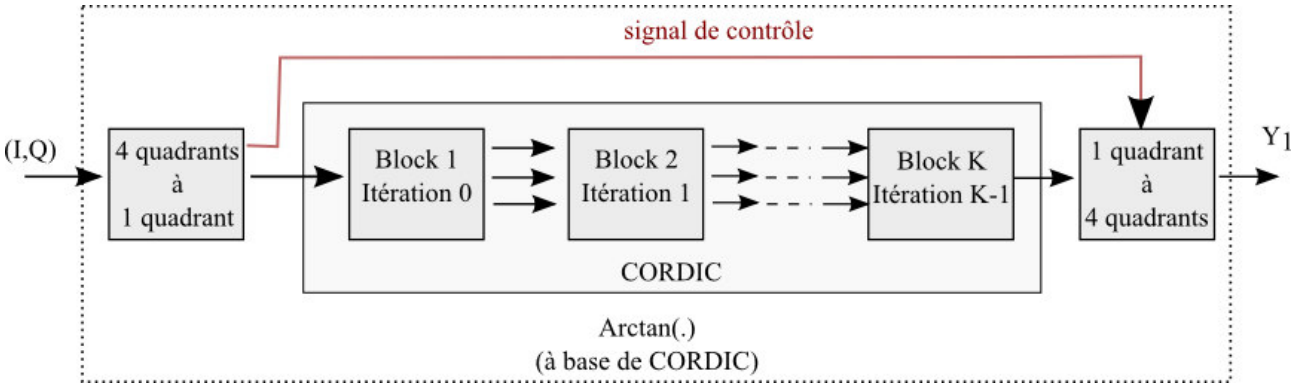


Figure 3.38 – Décomposition en blocs de l'algorithme CORDIC

Une fois que l'angle est calculé, la dernière étape pour compléter la démodulation est de faire la différentiation des angles obtenus (figure 3.39). Comme on a choisi de faire une simple différence, on a seulement besoin de rajouter un buffer supplémentaire au système, afin de retarder les échantillons de telle sorte que le différentiateur reçoive simultanément 2 échantillons consécutifs d'un canal donné.

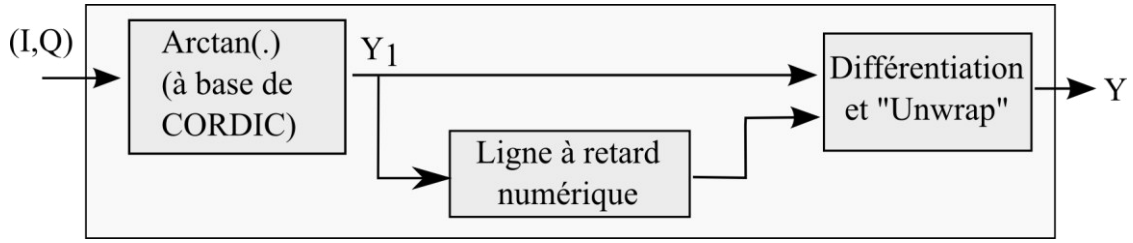


Figure 3.39 – Structure du démodulateur pipeliné « arctangente-différencier »

3.5.2 Démodulation par processeur embarqué pour le standard numérique DRM

La DRM est un standard numérique basé sur la technique OFDM. La DRM30, qui est la version du DRM pour les bandes de fréquences inférieures à 30 MHz, peut être diffusée selon 4 modes différents, dit « modes de robustesse ». Ces modes varient d'un mode de forte capacité de transmission avec faible robustesse (mode A) à un mode faible capacité avec forte robustesse (mode D) [CPWO06]. Ils correspondent chacun à des conditions de propagation différentes. La DRM+ qui est la version du DRM pour les bandes de fréquences supérieures à 30 MHz est diffusée dans un seule mode : le mode E. Dans le tableau 3.9 sont répertoriés les différents modes de robustesse et les conditions habituelles de propagation de leur utilisation. Le tableau suivant (tableau 3.10) quant à lui récapitule les caractéristiques OFDM des différents modes.

Mode de robustesse	Conditions typiques de propagation	
A	Canaux d'ondes de sol, avec peu de fading.	Ondes de sol
B	Canaux sélectifs en temps et fréquence, avec une plus grande dispersion des temps de propagation.	Ondes ionosphériques
C	Comme le mode de robustesse B, mais avec un étalement doppler plus élevé.	
D	Comme le mode de robustesse B, mais avec de sévères étalements des temps de propagation et de sévères étalements doppler.	

Tableau 3.9 – Modes de robustesse DRM30 et conditions habituelles de propagation [EBU08]

Liste des paramètres	Mode de robustesse				
	A	B	C	D	E
Tu (ms)	24	21+1/3	14+2/3	9+1/3	2.25
Tg (ms)	2+2/3	5+1/3	5+1/3	7+1/3	0.25
Tg/*Tu*	1/9	1/4	4/11	11/14	1/9
Ts = *Tu* + *Tg* (ms)	26+2/3	26+2/3	20	16+2/3	2.5
Δ*F* (Hz)	41+2/3	46+7/8	68+2/11	107+1/7	444.44

Tableau 3.10 – Paramètres des symboles OFDM pour la DRM30 (source : [CPWO06]) et la DRM+ (source : [DRM10]). *Ts* est la durée du symbole OFDM, *Tg* la durée de l'intervalle de garde, et *Tu* la durée de la partie utile d'un symbole OFDM (c'est-à-dire sans l'intervalle de garde). *ΔF*, l'espacement entre sous-porteuses, vaut 1 / *Tu*

La démodulation d'un tel standard numérique passe par plusieurs étapes nécessaires compte tenu de la complexité de ce type de modulation. La figure 3.40 en donne un synopsis.

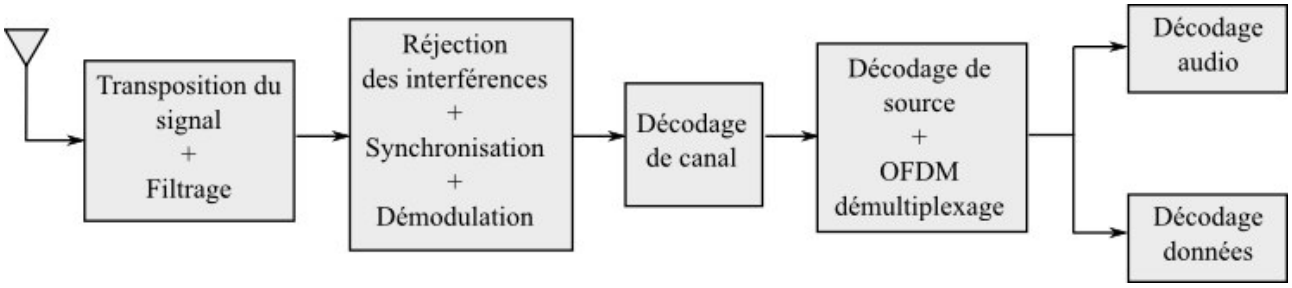


Figure 3.40 – Différentes étapes de la démodulation d'un standard numérique

Après, les méthodes adoptées pour faire l'une ou l'autre de ces étapes dépendent du concepteur. L'une des architectures de démodulation de la DRM les plus célèbres est celle proposée par Kurpiers et Fischer [KF03]. Ce sont les 2 auteurs à l'origine de l'open source Dream DRM dont nous parlerons plus loin. Les étapes qu'ils proposent sont représentées dans la figure 3.41 [KF03].

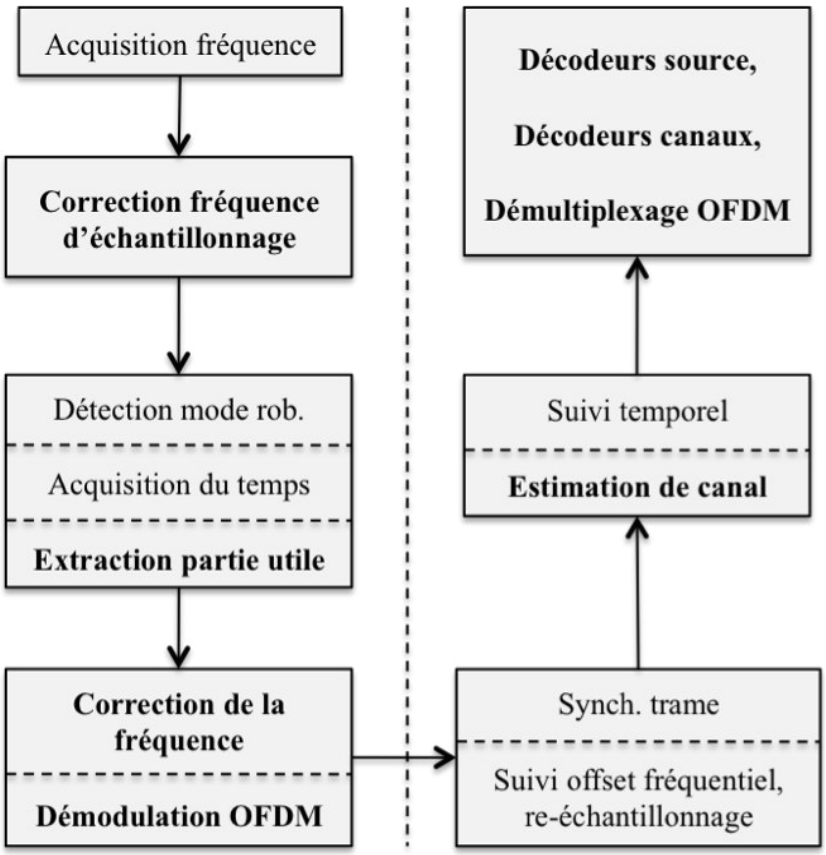


Figure 3.41 – Différentes étapes pour la démodulation du DRM, proposées par Kurpiers et Fischer. Source adaptée: [KF03]

Chacune de ses étapes sert à contrer soit l'interférence entre porteuses, soit l'interférence entre symboles.

- « Acquisition fréquence », « Correction de la fréquence » et « Suivi offset fréquentiel » consistent à déterminer le décalage – l'offset – fréquentiel du signal numérique reçu ainsi qu'à le corriger.
- « Correction fréquence d'échantillonnage» et « Suivi offset re-échantillonnage » servent à corriger la fréquence d'échantillonnage du signal acquis, lorsque celle-ci diffère de celle d'origine.
- « Détection mode rob.» consiste à déterminer dans quel mode de diffusion, (A, B, C ou D pour la DRM30) est diffusé le signal.
- « Acquisition du temps » et « Suivi temporel » servent à déterminer quel échantillon du signal est un début de symbole et à se synchroniser temporellement par rapport à ça.
- « Synch. trame » aide à déterminer le premier symbole de chaque trame.

Pour effectuer la démodulation du DRM sur FPGA, 2 types de solutions se présentent :

- Les solutions hardwares
- Les solutions softwares.

Les solutions hardwares sont les implémentations matérielles, en VHDL ou Verilog, qui effectuent toutes les opérations nécessaires à la démodulation. Les solutions logicielles sont les programmes développés pour des cœurs de processeurs implémentés dans le FPGA, comme les softcore Nios II et Microblaze, respectivement pour la marque Altera et Xilinx.

Les solutions matérielles présentent généralement l'avantage d'offrir au concepteur une grande maitrise des ressources utilisées. Il lui est alors possible de réaliser les optimisations nécessaires pour « consommer » le minimum de ressources possibles. Différents auteurs se sont penchés sur la question des solutions matérielles pour effectuer chacune des opérations susmentionnées, comme la synchronisation du signal [CPWO06] et la détection du mode du robustesse (RMD), l'estimation grossière du temps d'arrivée d'un symbole (CST), l'estimation grossière de l'offset fréquentiel (CFCFO-ICFO) et la détermination du premier symbole de la trame transmise (FT). D'autres méthodes, ensuite, permettent de resynchroniser [CPWOK07] en estimant soit l'offset fréquentiel (FFCFO), soit l'offset fréquentiel de l'horloge d'échantillonnage (SCFO), soit le décalage temporel d'un symbole.

Cependant, il reste quelques verrous à surmonter. Le premier est que les méthodes proposées sont développées pour le traitement d'une station à la fois or l'objectif que nous visons est un traitement multi-stations. Le deuxième challenge est qu'il faut développer un décompresseur audio MPEG-2 AAC qui peut traiter plusieurs canaux simultanément. Afin de réduire les temps de développement, nous nous sommes limités à une implémentation logicielle du décodeur DRM.

3.5.2.1 Etat de l'art des architectures logicielles pour démoduler la DRM.

Il existe 6 principales architectures logicielles qui permettent de démoduler un signal DRM, dont la plupart d'entres elles ont déjà été mentionnées dans la section 2.5.1.1: GNURADIO, WiNRADiO, Dream, SoDiRa, Diorama et Wavesink.

GNURADIO [GNUorg]: Un outil de développement logiciel libre qui fournit un environnement d'exécution de traitement du signal et des blocs de traitement pour implémenter les radios logicielles utilisant du matériel RF externe, bas coût et d'ores et déjà disponibles. Il est largement utilisé dans des environnements amateurs, académiques et commerciaux pour soutenir la recherche en communication sans fil ainsi que pour mettre en œuvre des systèmes de radio temps réel. Les applications GNURADIO sont premièrement écrites en langage Python, alors que les traitements et calculs sont implémentés en C++. Ainsi, le développeur est capable d'implémenter un système de radio temps-réel facile à utiliser. L'installation peut se faire soit par des fichiers précompilés, soit par la compilation manuelle utilisant le code source fourni. Il faut pour cela installer plusieurs fichiers externes. Actuellement, des efforts sont faits pour rendre GNU Radio compatible à d'autres

systèmes d'exploitation dont Microsoft Windows [GNUorg2].

WiNRADiO [WINcom]: Il s'agit d'un ensemble de logiciels de réception, d'enregistrement et de stockage des radios AM, FM ou DRM. Payants ou gratuits (versions de démonstration), ils accompagnent souvent des dispositifs RF externes (antennes, récepteurs etc.). Le code source de ces logiciels n'est pas disponible.

Dream [DREAMnet]: C'est un logiciel Open-Source qui peut s'exécuter en tant que récepteur AM, FM ou DRM et qui est disponible sous licence GNU – GPL (General Public License). Ce projet logiciel implémente au moins les caractéristiques minimales d'un récepteur DRM. Le langage de programmation est le C++. Le code fonctionne sur Mac OSX, Microsoft Windows et Linux, ce qui en fait un logiciel très portable. L'installation de ce logiciel se fait soit par fichier précompilé, soit manuellement à partir du code source.

SoDiRa [SoDiRa]: Il s'agit d'un logiciel capable de démoduler plusieurs standards, notamment le AM, FM, DRM, ainsi que le AMSS et le RDS. Pour le moment, les modes C et D de la DRM ne sont pas pris en compte. Le logiciel est un exécutable qui tourne sur le système d'exploitation Windows.

Diorama [Diorama]: « Diorama » est un récepteur logiciel temps réel de la DRM. C'est un open-source écrit pour s'exécuter sur Matlab. Ecrit en C, C++ et en langage Matlab (« .m », « .mat »), il est compatible avec Windows et Linux et il permet aussi d'enregistrer les flux audios démodulés.

Wavesink [Wavesink]: « Wavesink » est une solution logicielle complète pour réaliser une démodulation multi-standards de la bande VHF. La librairie est écrite en C++. Utilisée conjointement à un récepteur RF, « Wavesink » peut servir à démoduler le DAB, DRM+ et le FM-stéréo/RDS. Le récepteur est disponible comme une librairie ou avec un interface utilisateur pour afficher les informations sur le canal ou bien la démodulation. C'est une application très portable puisqu'elle peut tourner sur Windows, Linux, MacOS, Android, iOS.

Le tableau 3.11 résume la comparaison des 6 logiciels.

		Portabilité	
Logiciels	Open-source	Windows	Linux
GNURadio	O	Problèmes potentiels de compatibilité	O
WiNRADiO	×	O	O
Dream	O	O	O
SoDiRa	×	O	×
Diorama	O	O	O
Wavesink	×	O	O

Tableau 3.11 – Comparatif des 6 logiciels de démodulation de la DRM. O Disponible, × indisponible

On peut voir dans le tableau 3.11 que seuls deux packages logiciels présentent toutes les caractéristiques nécessaires pour une adaptation et une intégration au FPGA, à savoir :

- l'accès libre au code source
- la portabilité nécessaire

Il s'agit de Dream et de Diorama. Nous allons utiliser Dream dans la suite.

3.5.2.2 L'open source « Dream DRM »

L'architecture du logiciel Dream est donnée dans la figure 3.42.

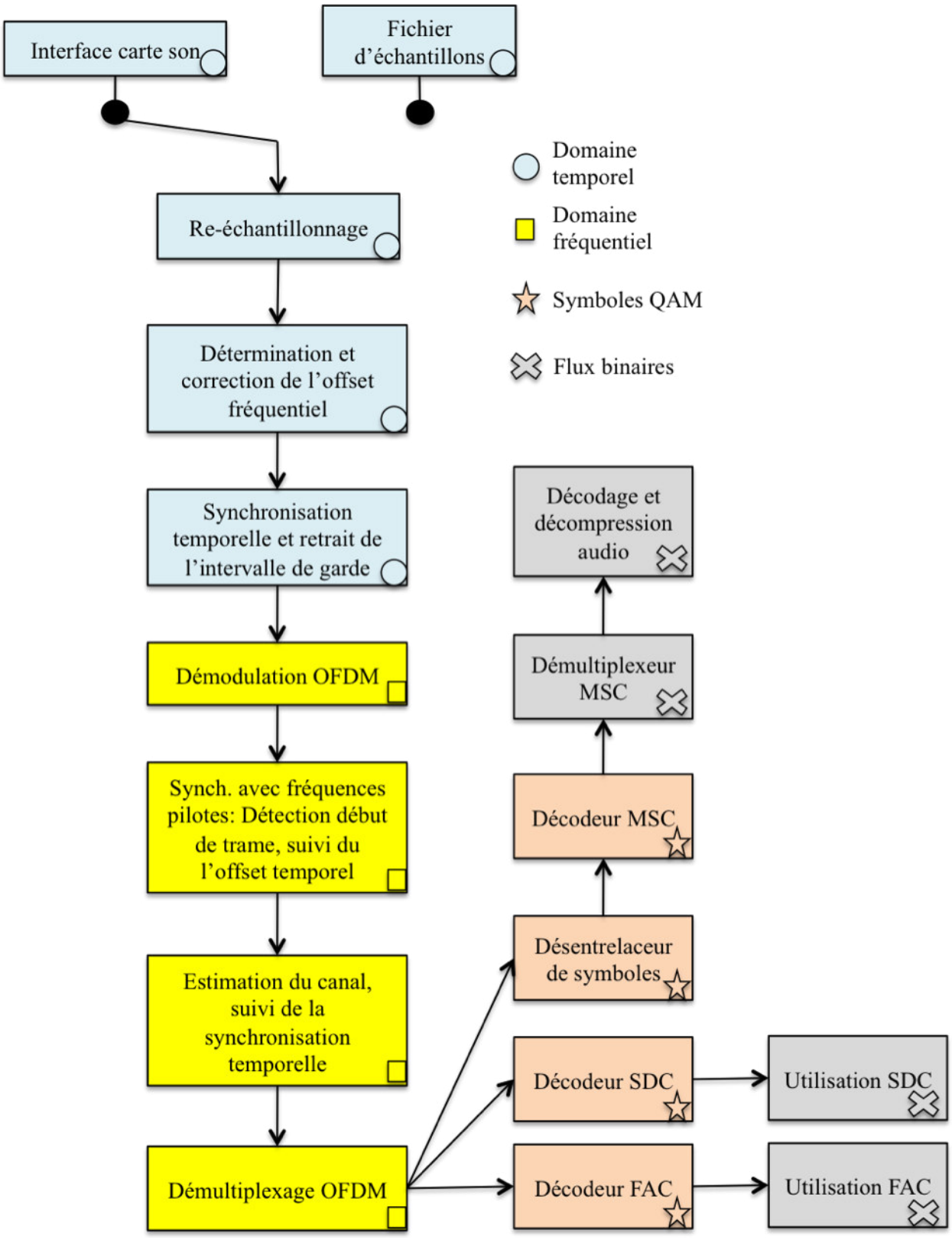


Figure 3.42 – Architecture du logiciel Dream DRM. Le « Fast Access Channel » (FAC) est utilisé par le récepteur pour obtenir des informations sur les propriétés du signal OFDM et sur la configuration SDC/MSC. Le « Service Description Channel » (SDC) contient les informations nécessaires pour le décodage du MSC, comme la structure de la trame multiplexée, et d'autres informations supplémentaires. Le « Main Service Channel » (MSC) encode la trame multiplexée générée par le multiplexeur et utilise les QAM 16 et 64. La carte son est utilisée pour la démodulation en temps réel

La figure 3.42 ne montre que la fonctionnalité « démodulation de la DRM » du logiciel Dream. Cependant, ce dernier est capable de se comporter en émetteur, il peut aussi gérer les standards AM et FM. De plus, c'est un logiciel qui intègre un simulateur de canaux de transmission ainsi qu'une interface graphique (ou GUI pour Graphical User Interface) codée avec les outils Qt [QTorg]. La bibliothèque Qt est une bibliothèque orientée objet utilisée pour créer des applications graphiques écrite en C++. Avec l'aide d'un stagiaire que j'ai co-encadré [MANFOUMBI12], nous avons désactivé ou supprimé les autres fonctionnalités afin de ne conserver qu'un noyau minimal du logiciel permettant de démoduler la partie audio du standard DRM. Le démonstrateur réalisé actuellement utilise le channelizeur DRM (cf. sections 5.2.3 et 5.2.6) et le flux est démodulé par un PC à l'aide de Dream.

La version « light » de Dream qui a été développée est en cours de portage sur un processeur embarqué sur FPGA (figure 3.43). Pour rendre cette solution multi-canaux, un moyen serait par exemple d'avoir une instanciation de plusieurs cœurs de processeurs (figure 3.44-a), ou bien d'utiliser un ou deux processeurs en mode temps partagé (figure 3.44-b).

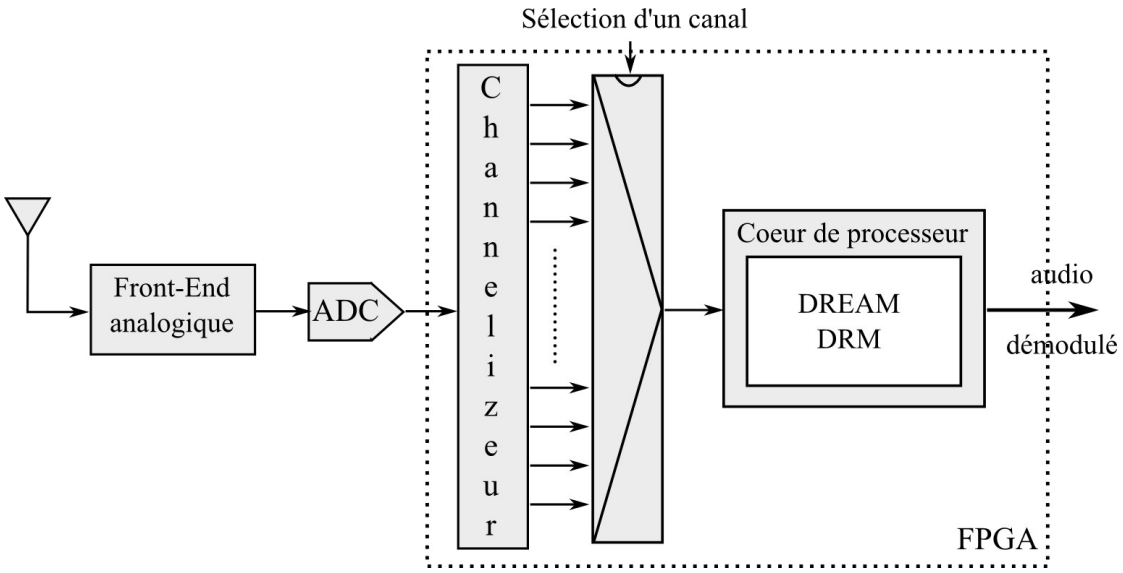


Figure 3.43 – Architecture du récepteur DRM mono-canal basé sur le logiciel Dream DRM

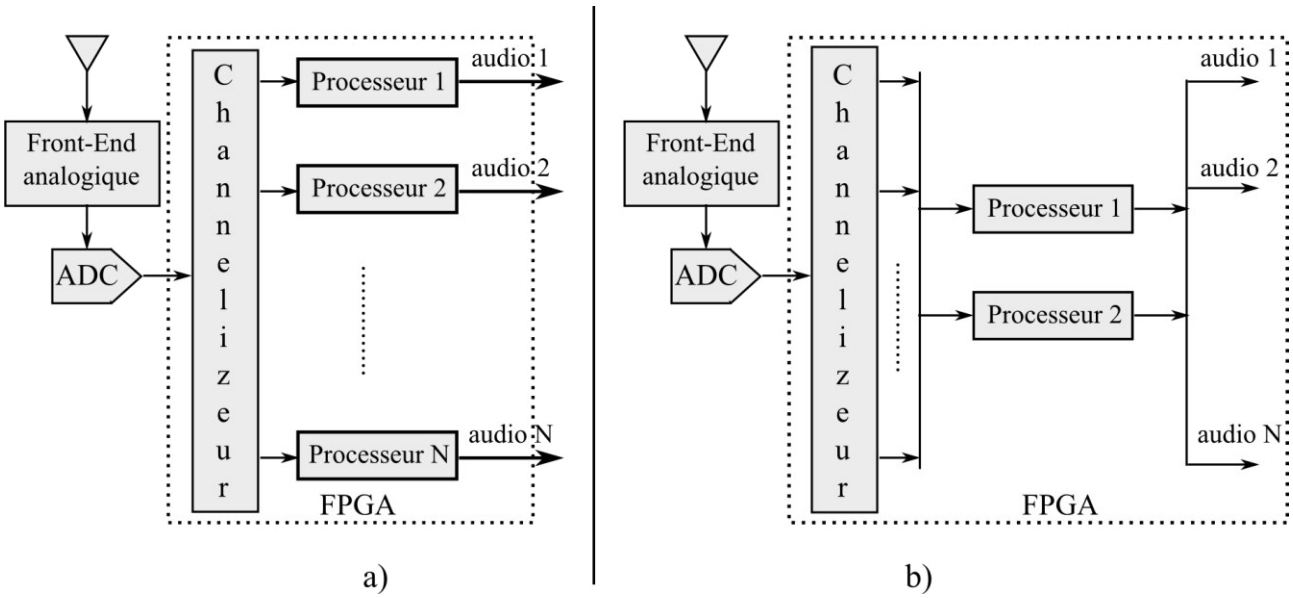


Figure 3.44 – Architecture du récepteur DRM multi-canaux basé sur le logiciel Dream DRM. a) plusieurs processeurs instanciés (un processeur par canal). b) Deux processeurs partagés en temps

Par ailleurs, la fonctionnalité démodulation DRM de l'audio que nous avons isolée de Dream nous sert à valider le channelizeur DRM que nous avons décrit dans la section 3.4.3. Le résultat du test de validation est donné dans la section 5.2.6.2.

3.6 Transmission des données démodulées pour un post-traitement

Après que les canaux DRM et FM aient été démodulés, ils sont transférés au système de traitement du contenu audio et des métadonnées. Un tel système peut bien sûr être implémenté dans un FPGA, GPU, CPU, MPSoC ou n'importe quel système hardware capable de traiter les signaux de manière appropriée. Quelque soit la cible matérielle de ce système, la transmission de données en série sera convenable peu importe le nombre de canaux à envoyer pourvu que la vitesse de l'horloge du système de transmission soit adéquate. Afin de garder la vitesse de l'horloge le plus bas possible, on choisit de ne transmettre que les canaux qui contiennent une station radio. Par exemple, à Paris, 51 chaines de radios FM ont été recensées. Dans ce cas, au lieu de transférer 256 canaux FM démodulés en plus des canaux DRM, on va uniquement transférer 51 stations FM ainsi que le nombre précis de stations DRM reçues. Cette démarche réduit de manière considérable les contraintes sur la fréquence d'horloge et permet une évaluation plus précise des protocoles et types de connexion qui répondent à nos besoins de transmission. Pour notre application, une connexion USB2 – dont le débit théorique maximal est 60 Mo/s – est adéquate (les considérations matérielles conduisant au choix de l'USB2 sont discutées dans la section 5.1.6.1).

Pour effectuer le transfert, pour la DRM, tous les canaux ciblés par notre système de

démodulation sont des canaux portant des stations existantes. Il suffit simplement de les mettre en série au moyen d'une FIFO. Pour la FM, un moyen simple consiste à sauvegarder dans des ROM du FPGA la cartographie des stations connues. Cette cartographie est une liste de drapeaux de 1 bit représentant chacun un canal: le drapeau 0 représente le canal 87.0 MHz; le drapeau 1 le canal 87.1 MHz, jusqu'au drapeau 255 pour 112.5 MHz (en général, la bande FM s'arrête à 108 MHz). Si une station existe sur un canal, le drapeau correspondant est mis à 1, sinon il est mis à 0. Les échantillons des canaux provenant de chaque branche du récepteur FM sont sauvegardés dans des buffers. Il y a un buffer par canal. Une FIFO dual clock sert d'interface entre les canaux démodulés la connexion USB2. Un bloc, appelé « contrôleur», lit alternativement chaque buffer et envoie l'échantillon à la FIFO d'interface, tandis que la carte contrôle quels échantillons sont écrits dans la FIFO. La figure 3.45 schématise le système de transfert complet.

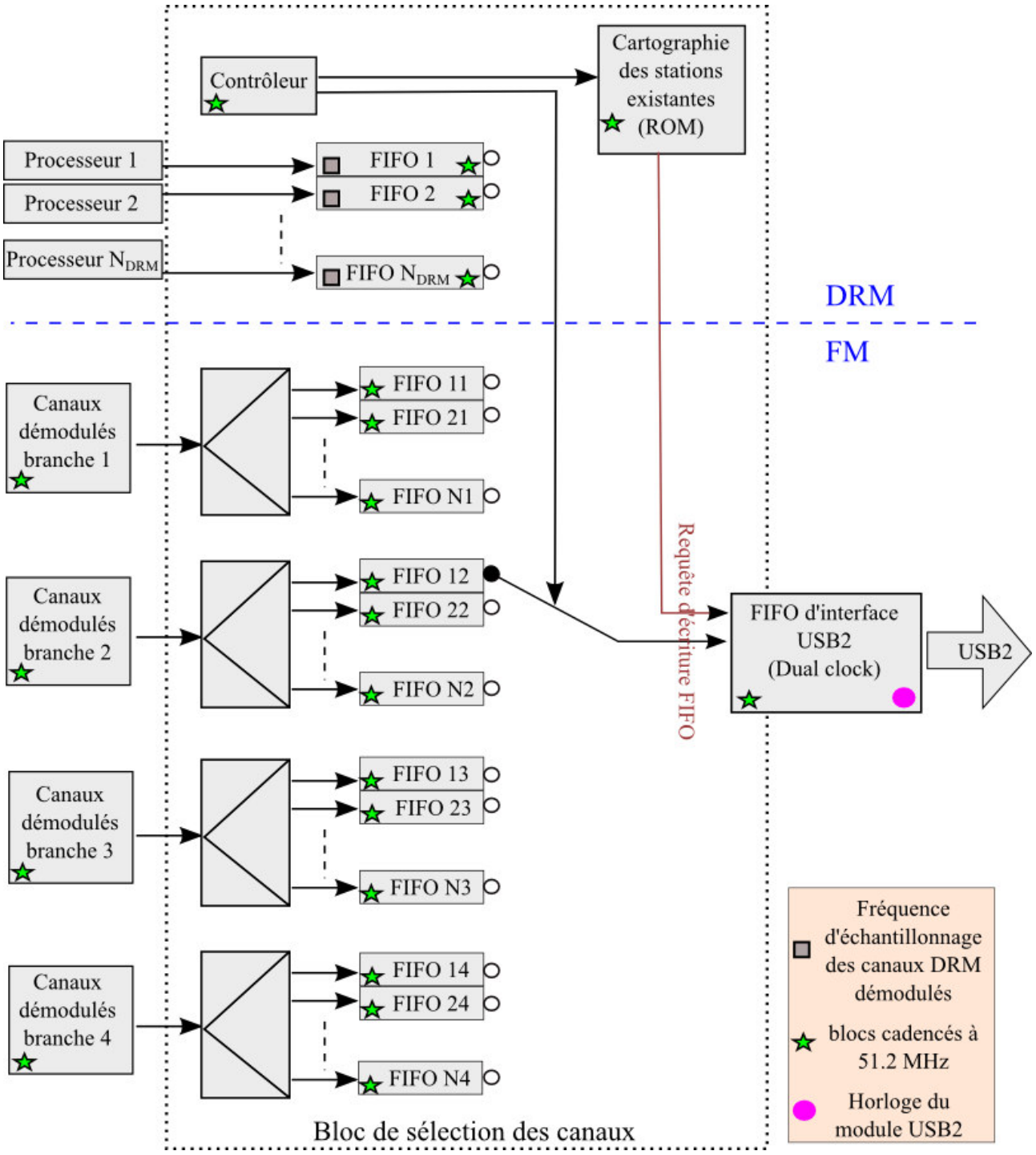


Figure 3.45 – Système de transmission des stations démodulées. On considère que la démodulation massive de tous les canaux DRM est faite par un processeur par canal (cf. figure 3.44-a). Chaque “FIFO i” est une FIFO dual clock qui sert de buffer à un canal DRM. Chaque “FIFO ij” est le buffer d’un canal FM.

3.7 Bibliographie

[CPWO06] Chen Chen; Beomjin Park; Lijun Wei; Hyun-Seok Oh, "Synchronization Acquisition Methods for DRM Systems," *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th* , vol., no., pp.1,5, 25-28 Sept. 2006

[CPWOK07] Chen Chen; Beomjin Park; Lijun Wei; Hyun-Seok Oh; Kyungho Kim, "Synchronization Tracking Methods for DRM Systems," *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE* , vol., no., pp.1965,1969, 11-15 March 2007

[Diorama] <http://nt.eit.uni-kl.de/static/diorama/index.html>. Dernier accès le 06 Oct. 2013.

[DREAMnet] <http://sourceforge.net/apps/mediawiki/drm/>. Dernier accès le 16 Août 2013.

[DRM10] DRM Consortium. « Digital Radio Mondiale (DRM) – A Broadcaster's Guide, Version 1.1 ». DRM Consortium. June 2010.

[EBU08] European Broadcasting Union (EBU). « Technical Bases for DRM Services Coverage Planning ». Source: B/LMS. Status: Report. Geneva, Switzerland, June 2008.

[GNUorg] <http://www.gnuradio.org>. Dernier accès le 16 Août 2013.

[GNUorg2] <http://gnuradio.org/redmine/projects/gnuradio/wiki/WindowsInstall>. Dernier accès le 23 Sept. 2013.

[KF03] Kurpiers, A.F.; Fischer, V., "Open-source implementation of a Digital Radio Mondiale (DRM) receiver," *HF Radio Systems and Techniques, 2003. Ninth International Conference on (Conf. Publ. No. 493)* , vol., no., pp.86,90, 23-26 June 2003

[MANFOUMBI12] M. Manfoumbi Mouziegou, « Contribution à la conception d'un récepteur parallèle pour la norme DRM sur FPGA ». Rapport de stage de fin d'études d'ingénieur, ESIR, Université de Rennes 1, Rennes, France, 2012.

[QTorg] <http://qt-project.org/>. Dernier accès le 24 Sept 2013.

[RPN09] M. Rice, M. Padilla, and B. Nelson, "On FM Demodulators in Software Defined Radios Using FPGAs," in *IEEE Military Communications Conference, MILCOM 2009*, Boston, Massachusetts, Oct. 18 – 21, 2009, pp.1-7.

[SoDiRa] <http://www.dsp4swls.de/sodira/sodiraeng.html>. Dernier accès le 06 Oct. 2013.

[V59] J. E. Volder. (1959, Sept.). The CORDIC Trigonometric Computing Technique. *IRE Trans. Electronic Computers*. *EC-8(3)*. pp. 330–334.

[W71] J. S. Walther, "A unified algorithm for elementary functions," in *Proc. American Federation of Information Processing Societies Spring Joint Computer Conference, AFIPS'71*, New York, May 18 – 20, 1971, pp. 379–385.

[Wavesink] <http://www.feilen-stolz.de/receiver.php>. Dernier accès le 06 Oct. 2013.

[WINcom] <http://www.winradio.com/index.htm>. Dernier accès le 23 Sept. 2013.

4 Une procédé de re-échantillonnage optimisé

Dans cette section, nous allons détailler le procédé d'échantillonnage que nous avons développé. Nous en avons fait référence dans la section 3.3. Ce procédé est né du besoin d'effectuer le re-échantillonnage du signal numérisé afin que le banc de filtre développé pour la FM puisse extraire tous les canaux de la bande (voir section 3.4.2 pour plus de détails) sans interférences (superpositions de canaux dans un canal). Le re-échantillonnage consiste à changer la fréquence d'échantillonnage d'un signal numérique donné (figure 3.4). Comme cela a été mentionné dans la section 3.3, nous avons voulu proposer une méthode de re-échantillonnage facilement implémentable sur FPGAs et qui soit un outil pratique de conception utilisable aussi dans des contextes de radio logicielle autres que SurfOnHertz. Partant du constat que, dans la plupart des cas la conversion de fréquence requise n'est pas à facteur entier, nous avons caractérisé cette méthode « à ratio arbitraire », c-à-d, à même de prendre en charge des cas où le facteur de conversion n'est pas entier.

Bien que les performances des FPGAs évoluent constamment, ces derniers souffrent encore de certaines limitations technologiques (par exemple les fréquences d'horloge maximales des blocs DSP internes, bandes passante des entrées/sorties, ressources internes, etc.) et aux performances des outils de placement et routage. Ces limites restreignent la plage des fréquences et la complexité des circuits qui peuvent être exploités en pratique dans une architecture FPGA, de sorte que certaines architectures paraissent viables d'un point de vue théorique mais ne le sont pas d'un point de vue matériel. D'un autre côté, si un FPGA supporte des fréquences compatibles avec ceux de la bande VHF, il se peut que l'équipement du concepteur soit incapable de générer une horloge avec une fréquence suffisamment élevée.

Le Spurious Free Dynamic Range, ou SFDR, défini comme le rapport d'un signal à son harmonique parasite la plus puissante, est un critère naturel de qualité des systèmes d'échantillonnage. Dans différentes situations, les harmoniques parasites doivent être prises en charge soigneusement puisqu'elles peuvent négativement influencer la sortie d'une chaine de traitement. Par exemple, dans le cadre de SurfOnHertz où on développe un système radio logicielle pour démoduler des stations FM, la différence d'amplitude entre les stations les plus puissantes et les stations faibles peut être de 50 dB ou plus. Le concepteur doit s'assurer que la procédure de re-échantillonnage n'introduit pas d'harmonique parasite qui peut masquer les stations faibles.

Le mécanisme de génération des harmoniques est intimement lié au mécanisme de re-échantillonnage que voici : le spectre du signal d'entrée échantillonné est un motif répété régulièrement tous les λF_{IN}, où λ est un nombre entier. La reconstruction d'une version analogique

du signal avant l'échantillonnage peut alors être interprétée comme le filtrage du signal numérique de façon à ne conserver qu'une seule réplique et supprimer les autres, comme le montre la figure 4.1-a. C'est ce signal analogique qui, échantillonné ensuite à F_{OUT}, devient le signal re-échantillonné, comme indiqué dans la figure 4.1-b.

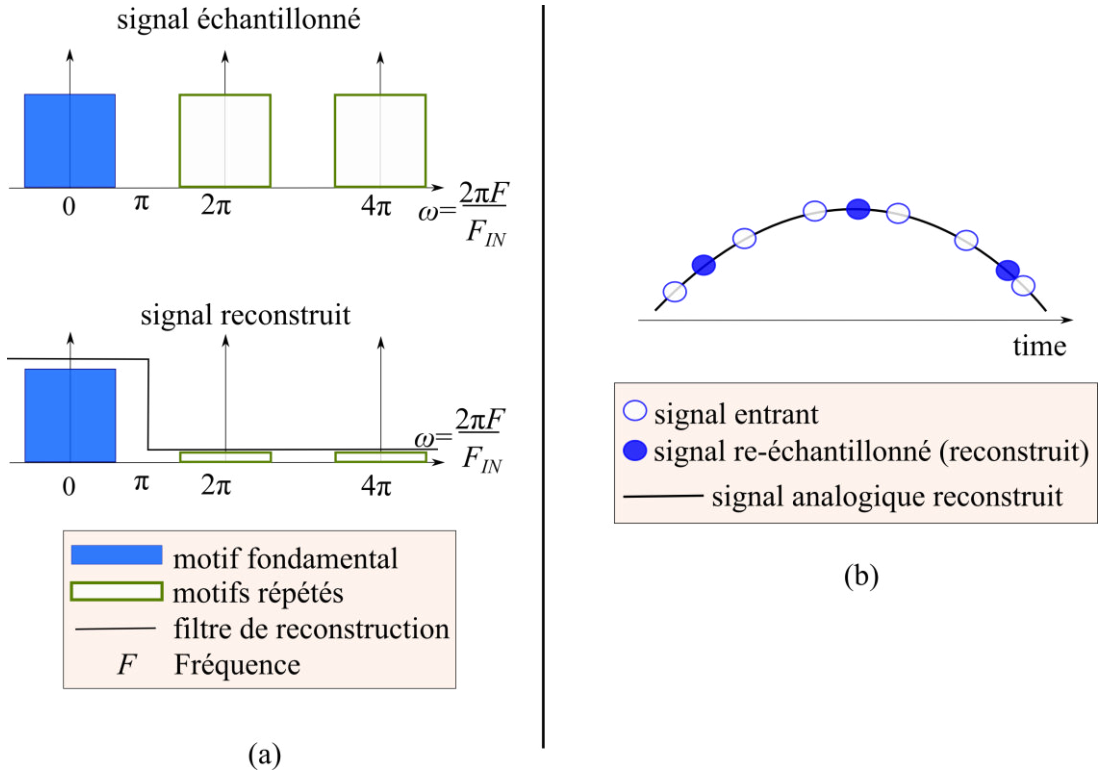


Figure 4.1 – Mécanisme de re-échantillonnage d'un signal. a) Spectre d'un signal avant et après reconstruction. b) Répresentation temporelle d'un signal re-échantillonné

Lorsqu'un signal est re-échantillonné, les harmoniques parasites proviennent de la rejection imparfaite des répliques spectrales au cours du processus de reconstruction. Les amplitudes de ces harmoniques dépendent du filtre de reconstruction utilisé dans le processus de re-échantillonnage. En règle générale, plus il y a de ressources matérielles dédiées à ce processus, plus les harmoniques parasites peuvent être réduites. La quantité de ressources disponibles sur un FPGA étant limitée, cela limite le SFDR qu'on peut atteindre, rendant ainsi les compromis nécessaires entre SFDRs voulus et ressources disponibles.

Dans toute cette section (section 4) la méthode proposée sera présentée. Il s'agit d'une solution de re-échantillonnage à ratio arbitraire pour des applications radio logicielle, basée sur FPGA. Elle permet au concepteur de contrôler le SFDR à partir d'une interface entre les domaines d'horloge d'entrée et de sortie simple, pratique, et qui ne nécessite aucune horloge en plus. Cela rend cette méthode appropriée aux designs FPGAs limités en terme d'horloge. Le re-échantillonneur prévu pour des fréquences allant jusqu'à la bande VHF (en dessous de 300 MHz environ), avec une

variante destinée au sur-échantillonnage, et une autre destinée au sous-échantillonnage. L'approche proposée, basée sur le SFDR, fournit une estimation de la quantité des ressources qui correspond fidèlement à l'utilisation des ressources une fois que le circuit FPGA a été implémenté.

4.1 Etat de l'art

Pour commencer, voyons quelles sont les méthodes de re-échantillonnage existantes en rapport avec notre travail. Depuis les premières structures d'interpolation polynomiale proposées par Farrow en 1988 [F88], les dernières décennies ont vu l'arrivée de nombreuses architectures de conversion de fréquence d'échantillonnage appropriées pour la radio logicielle et qui se prêtent bien à une implémentation sur FPGA. Celles-ci peuvent être groupées en quatre catégories: Les architectures à base de structure de Farrow; celles à base de filtres dits « Cascaded Integrator–Comb », ou CIC; celles à base de filtres FIR polyphases; et les méthodes hybrides. Dans la suite de ce document, F_{IN} désignera la fréquence d'échantillonnage du signal à traiter et F_{OUT} sera celle du signal re-échantillonné.

4.1.1 Méthodes à base de structures de Farrow

Les systèmes basés sur les structures Farrow fournissent une méthode efficace pour implémenter des interpolateurs polynomiaux de Lagrange ou des splines sur DSP ou FPGA. Ces systèmes sont souvent utilisés pour l'interpolation linéaire, parabolique ou cubique, ce qui est suffisant pour les applications pour modems [EGH93]. Cependant, des polynômes d'interpolation d'ordres supérieurs peuvent également être implémentés, par exemple dans [Q10], où les auteurs proposent une implémentation FPGA d'un système de re-échantillonnage complet basé sur des structures de Farrow, incluant un générateur d'intervalles fractionnaires. La structure de base est illustrée à la figure 4.2, où μ_k est la fraction d'une période d'échantillonnage d'entrée entre un échantillon de sortie k et l'échantillon m_k d'entrée qui le précède immédiatement, comme le montre la figure 4.3.

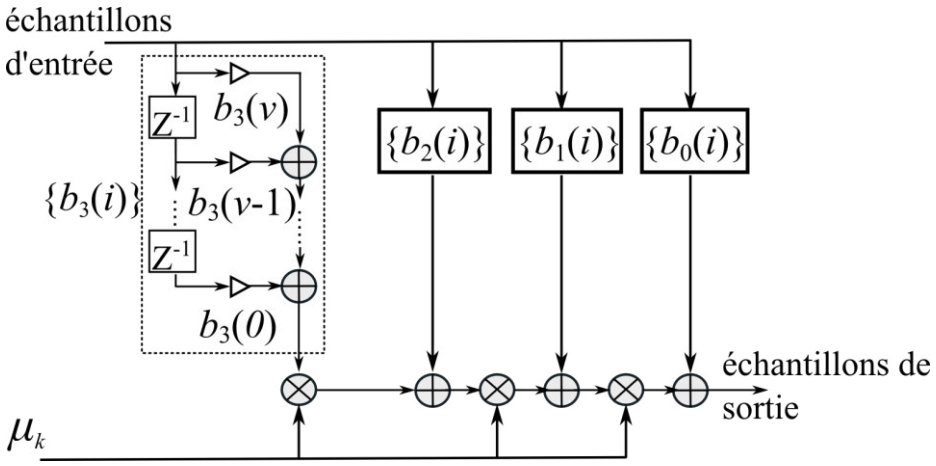


Figure 4.2 – Structure Farrow pour un interpolateur cubique. Source modifiée : [Q10]. "$b_l(j)$" représente les coefficients utilisés pour pondérée les échantillons entrants

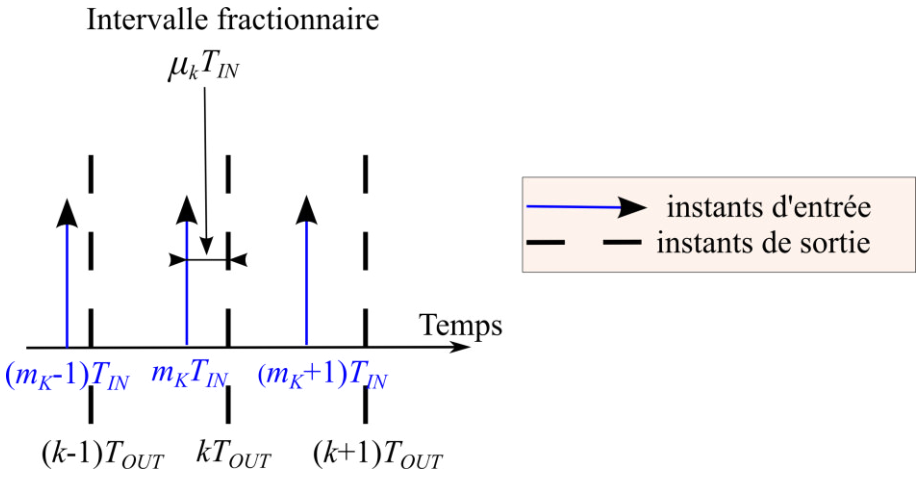


Figure 4.3 – Intervalle fractionnaire

L'implémentation matérielle des structures de Farrow d'ordre élevé conduit cependant à une difficulté: ces systèmes nécessitent une troisième horloge fonctionnant à F_C, qui, pour des applications radio logicielles à hautes fréquences, est exclue vu que le FPGA sera incapable de supporter une plus grande fréquence d'horloge. En particulier, l'intervalle fractionnaire μ_k est un paramètre essentiel dans toute opération de re-échantillonnage. Pour effectuer la conversion de fréquence d'échantillonnage arbitraire [Q10], c'est F_C qui sera utilisé pour calculer le μ_k de la figure 4.2. Dans le cas du sur-échantillonnage, nous avons $F_C=I_N.F_{IN}$, où I_N est un nombre entier pris suffisamment grand pour que $F_C>F_{OUT}$, alors que pour le sous-échantillonnage, $F_C=I_N.F_{OUT}$ et $F_C>F_{IN}$.

D'un autre côté, ces systèmes peuvent fonctionner comme des convertisseurs de fréquence d'échantillonnage rationnels avec un facteur de conversion donné par R/L. En général, les re-échantillonneurs rationnels sont des convertisseurs avec des étages de sur-échantillonnage distincts des étages de sous-échantillonnage. Lorsque les structures Farrow sont utilisées dans ce cas, F_C

dépend de *R* ou *L*. Bien que de nombreux systèmes ont été développés autour des structures Farrow modifiées ou évoluées, la question de la troisième horloge demeure, ce qui peut être gênant si les fréquences F_{IN} et F_{OUT} sont déjà proches des limites de ce que le FPGA utilisé peut supporter, ou si un générateur d'horloge capable d'atteindre la fréquence nécessaire n'est pas disponible.

4.1.2 Méthodes à base de CIC

Les filtres CIC (cascaded integrator – comb) sont constitués d'un ou plusieurs paires de filtres integrator/decimator – comb (figure 4.4). Il s'agit d'une classe de filtres qui peuvent être utilisés dans des applications d'interpolation et de décimation. La figure 4.4 montre un filtre CIC conventionnel. Pour les architectures à base de CIC, diverses méthodes ont été proposées pour améliorer les caractéristiques du filtre CIC et le rendre approprié aux applications radio logicielles [SR97][J-DM05][NA06][A-A-SS03][J-DM06][J-D07]. Ces méthodes permettent d'obtenir à une meilleure réjection des répliques spectrales et un meilleur SFDR.

En général, les filtres à base de CIC n'utilisent pas de multiplications, contrairement aux structures de Farrow. Cependant, la conception de filtres CIC pour re-échantillonnage arbitraire conduit à des problèmes avec les horloges utilisées dans les implémentations matérielles. Comme la figure 4.4 le montre, les méthodes basées sur CIC effectuent une conversion de fréquence d'échantillonnage rationnelle, c'est-à-dire que c'est une interpolation suivie d'une décimation. Cela signifie qu'une troisième horloge est nécessaire ici aussi, entre les étapes de sur-échantillonnage et de sous-échantillonnage. Ainsi, si *R* est élevé, la simple structure de l'exemple de la figure 4.4 n'est pas faisable. Des optimisations seront nécessaires durant la conception pour limiter la fréquence de la troisième horloge. Ces optimisations, qui peuvent rapidement devenir complexes, ne peuvent pas conduire à une suppression de la troisième horloge, mais seulement à une réduction de sa fréquence.

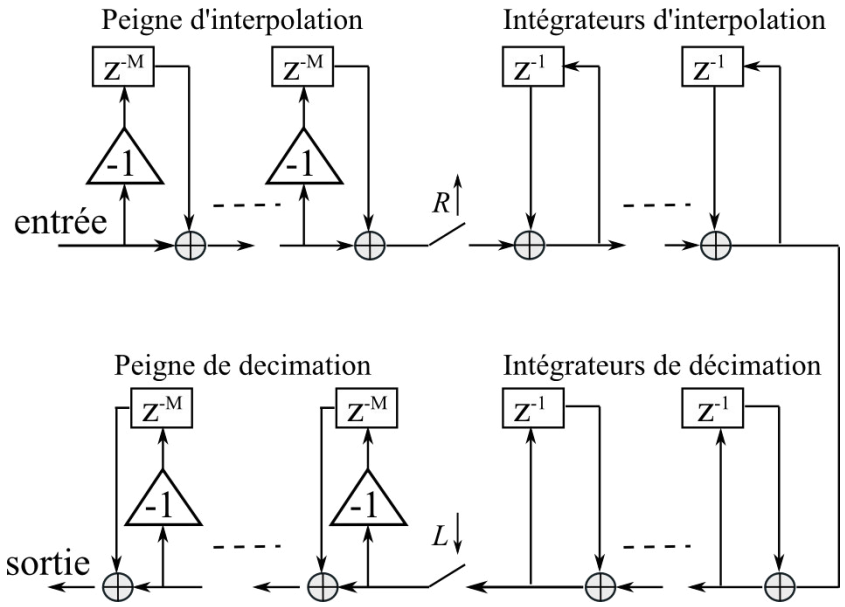


Figure 4.4 – Filtre CIC conventionnel pour la conversion de fréquence d'échantillonnage de ratio *R/L*. Source modifiée : [A-A-SS03]. Les mots « peigne » et « intégrateurs » sont les traductions respectives de « combs » et « integrators »

4.1.3 Méthodes à base de filtres polyphases FIR

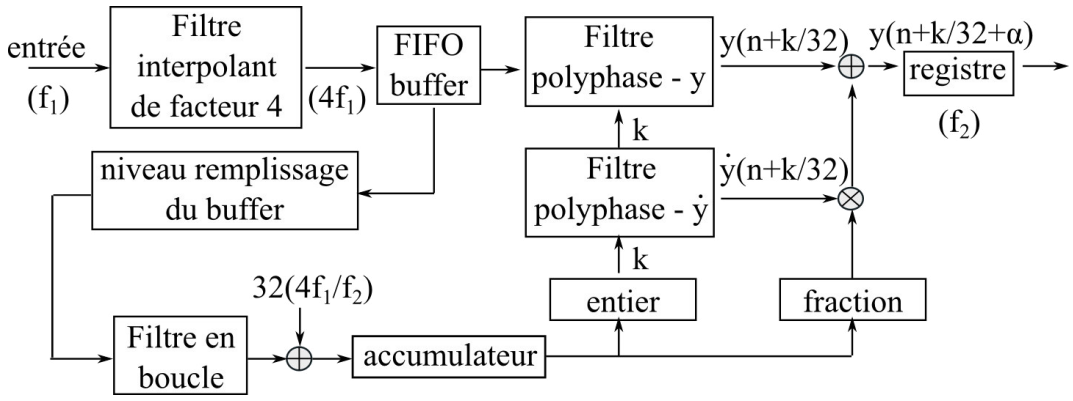
Les premières architectures utilisant des filtres FIR polyphasés ont été conçues pour la conversion de fréquence à rapport entier; certaines architectures pour la conversion de fréquence arbitraire ont été développées dans [DH04] [JPKP08] [B08] [XWS10]. Les performances en terme de SFDR de ces architectures sont généralement élevées, les rendant appropriées pour les applications radio logicielles, mais elles sont consommatrices de ressources mémoire. Deux architectures implémentables sur FPGA et basées sur des filtres FIR polyphasés se démarquent. Elles sont décrites ci-dessous.

4.1.3.1 Les re-échantillonneurs arbitraires de C.Dick et F.Harris

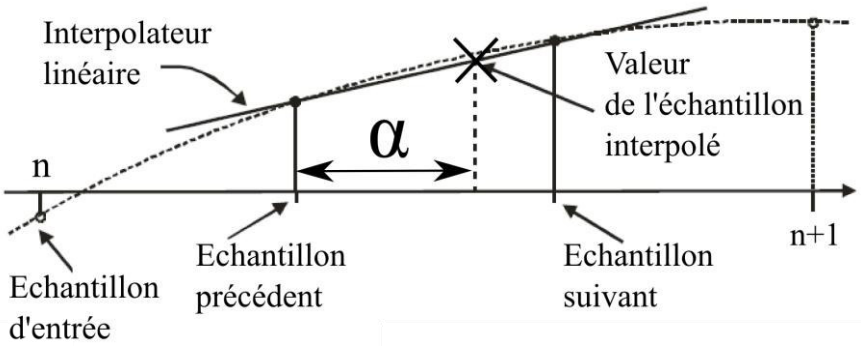
La référence [DH04] décrit cette méthode. Bien qu'elle ait été premièrement conçue pour le sur-échantillonnage, la méthode peut être utilisée pour le sous-échantillonnage. La structure de base est une cascade de deux étages de filtrage. Le premier étage (représenté sur la figure 4.5-a par le bloc "Filtre interpolant de facteur 4") est constitué par le filtre qui accepte les échantillons d'entrée et les sur-échantillonne par un facteur I (I est un facteur défini par l'utilisateur). Le processus de sur-échantillonnage nécessite évidemment une troisième horloge (multiple de F_{IN}) dont la fréquence est I fois plus élevée que la fréquence de l'horloge d'entrée. Ce sur-échantillonnage augmente l'espacement entre les images spectrales du signal échantillonné. Il permet de réduire la complexité (ressources mémoire et de calcul utilisées) de la structure de filtre suivante. La seconde opération de filtrage est effectuée par une paire de M_1 – stage filtres polyphases (représentée sur la figure 4.5-a

par les blocs "Filtre Polyphase") qui calculent la valeur de l'échantillon interpolé et la valeur de la première dérivée interpolée. Les filtres polyphases de la paire utilisent leur $k^{\text{ième}}$ branche pour calculer un échantillon et une valeur dérivée située à un $(k / M_1)^{\text{ième}}$ d'une période d'horloge, un offset α à partir de l'emplacement de l'échantillon souhaité, où α est l'intervalle fractionnaire entre l'échantillon précédent donné par le premier étage de filtrage (l'étage de sur-échantillonnage) et l'échantillon désiré (figure 4.5-b). Plus I est grand, plus la bande de transition de l'interpolateur est grande, ce qui permet d'avoir un filtre polyphase plus court avec moins de taps par stage. L'interpolation finale est effectuée par un développement en série de Taylor à deux termes de la fonction de temps à l'aide de la relation donnée dans (4.1). La structure intègre un filtre en boucle Proportionnel Intégral standard (figure 4.5-a), ce qui rend le re-échantillonneur en mesure de supporter une certaine dérive de la fréquence moyenne de l'horloge de sortie.

$$y(n + k / M_1 + \alpha) = y(n + k / M_1) + \alpha \dot{y}(n + k / M_1) \tag{4.1}$$



(a)



(b)

Figure 4.5 – Re-échantillonnage à ratio arbitraire. Source modifiée : [DH04]. (a) graphe de flux de données d'un re-échantillonneur à ratio arbitraire avec I = 4, M_1 = 32. Source modifiée : [DH04]. (b) L'offset, α, Source modifiée : [DH04]. Les échantillons "n" et "n+1" sont les échantillons d'entrée du re-échantillonneur. Les échantillons «précédent» et «suivant» sont ceux générés par le premier étage de filtrage. L'échantillon «valeur de l'échantillon interpolé» est la sortie

Cette structure peut atteindre de hauts SFDRs comme 80 dB, ce qui la rend convenable pour une grande variété d'applications. Néanmoins, par rapport aux autres méthodes considérées jusque-là, elle est un peu plus complexe à implémenter.

Un sous-échantillonneur semblable au système de Dick et Harris a été proposé par Jorgovanovic *et al* dans [JPKP08]. Comme il exécute un re-échantillonnage par le calcul du plus proche voisin, il utilise moins de blocs que le re-échantillonneur de Dick et Harris. Plus précisément, il n'y a ni «filtre interpolant» ni un «filtre polyphase - $\dot{y}$ ». La structure demeure basée sur un filtre en boucle pilotant un filtre polyphase, ce qui rend le sous-échantillonneur capable de supporter la dérive des fréquences d'horloge d'entrée et de sortie. Cependant, les performances obtenues par cette structure ne sont pas aussi bonnes que celles du système de Dick et Harris. Pour que ses performances soient de même niveau que ce dernier, les deux filtres mentionnés ci-dessus doivent être mis en œuvre, ce qui implique une fois encore l'utilisation d'une troisième horloge.

4.1.3.2 La structure de Barker

Dans [B08], Barker propose une approche simple, illustrée à la figure 4.6, qui peut être appropriée pour des implémentations sur FPGA. Dans [XWS10], les auteurs montrent une façon de réduire l'utilisation de la mémoire et de n'utiliser qu'un seul multiplieur dans ce type de structure. Selon [B08] et [XWS10], cette méthode peut atteindre de très grand SFDR. Néanmoins, la question de savoir précisément comment gérer l'interface entre les deux domaines d'horloge F_{IN} et F_{OUT} demeure. De plus, afin de compenser la réduction de la mémoire et du nombre de multiplieurs de [XWS10], une horloge complémentaire dont la fréquence est (N_T / 2) – fois la fréquence d'entrée est introduite, où N_T est le nombre de taps dans la structure originale de Barker. La figure 4.6 montre cette architecture avec les échantillons d'entrée arrivant à F_{IN} et l'accumulateur fonctionnant à F_{OUT}. Un tel système, s'il est implémenté tel que proposé ici, génèrerait des erreurs de calcul une fois implémenté vu qu'il n'y a pas d'interface appropriée entre les deux domaines d'horloge.

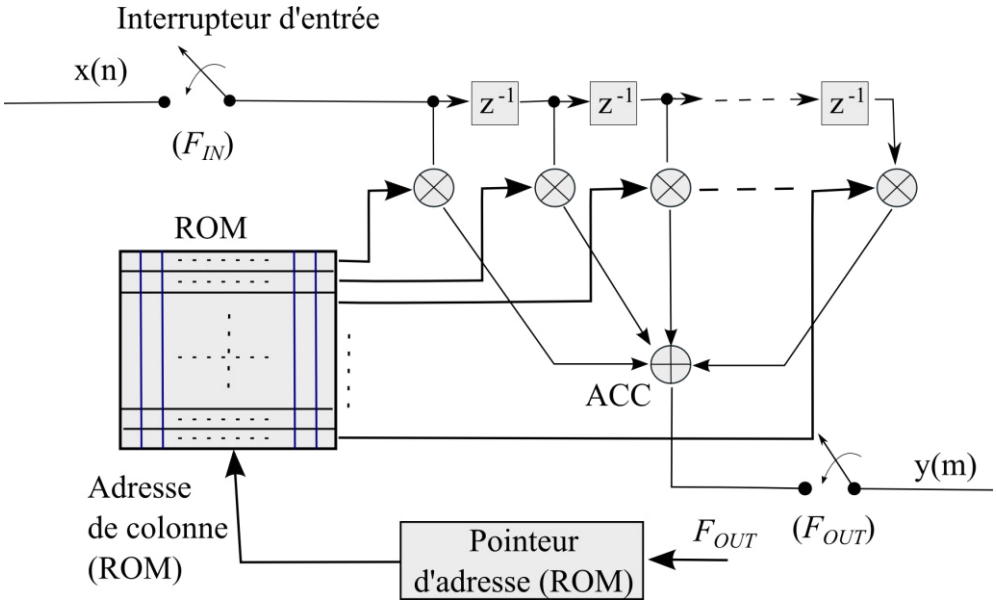


Figure 4.6 – Sur-échantillonneur basé sur un filtre-ROM polyphasé. Source modifiée : [B08]

4.1.4 Méthodes hybrides

La dernière classe de re-échantillonneurs comprend les systèmes de re-échantillonnage qui sont des combinaisons de procédés de re-échantillonnage différents. Deux architectures combinant des structures de Farrow et de CICs, par exemple, sont détaillées dans [BR05], tandis qu'une combinaison de sur-échantillonnage à ratio entier avec une interpolation de Lagrange apparaît dans [R84] [E03]. Dans tous ces cas, pour interpoler le signal et calculer les intervalles fractionnaires, les méthodes proposées exigent une troisième horloge dont la fréquence est un multiple de celle de l'horloge d'entrée ou de celle de l'horloge de sortie.

4.1.5 Conclusions sur l'état de l'art

Le tableau 4.1 résume les caractéristiques des différentes méthodes listées plus haut.

Catégorie	Références	Caractéristiques	Besoin d'une 3ième horloge ?
A base de structures de Farrow	[F88][EGH93] [Q10]	Peuvent être implémentés sans utiliser de ressource mémoire ([Q10])	Oui
A base de CIC	[SR97][J-DM05] [NA06][A-A-SS03] [J-DM06][J-D07]	Ne nécessitent pas de multiplication	Oui
A base de filtres FIR polyphases	Dick et Harris [DH04]	-- Peut produire des SFDRs élevés* -- Implémentation plus complexe -- Supporte la déviation de la fréquence moyenne de l'horloge de sortie	Oui
	Jorgovanovic *et al* [JPKP08]	-- Peut produire des SFDRs élevés* -- Supporte les déviations de la fréquence moyenne de l'horloge d'entrée par rapport à l'horloge de sortie	Oui (afin d'avoir des SFDRs élevés)
	Barker [B08]	-- Devrait produire des SFDRs élevés* -- Problème de l'implémentation FPGA non traité	Non
	Barker modifié [XWS10]	-- Devrait produire des SFDRs élevés* -- Problème de l'implémentation FPGA non traité	Oui
Types hybrides	[BR05][R84][E03]	Produit des SFDRs plus élevés que pour les catégories Farrow et CIC	Oui

* Ici, on entend pas « SFDR élevé » des SFDRs supérieurs ou égaux à 80 dB

Tableau 4.1 – Tableau de classification de plusieurs méthodes de re-échantillonnage existantes

Aucune de ces méthodes ne propose une implémentation claire permettant de s'affranchir de

l'utilisation d'une troisième horloge.

L'utilisation d'une troisième horloge pour gérer l'interface entre les deux domaines d'horloges est un problème dans le contexte visé ici, à savoir celui d'applications VHF. En effet, la fréquence de cette horloge pourrait rapidement atteindre les limites de ce que le FPGA peut prendre en charge, ou de ce qu'un oscillateur à quartz standard peut générer. Cependant, en utilisant une structure de re-échantillonnage similaire à celle de Barker [B08], tout en interfaçant proprement les deux domaines d'horloge, la technique classique du filtre FIR classique devrait être en mesure de fournir un modèle de re-échantillonneurs pratiques qui ne comprennent pas de troisième horloge.

Ce qui suit est l'implémentation détaillée d'une telle structure, un re-échantillonneur similaire à la structure Barker, mais basée sur un FIR classique plutôt qu'un FIR polyphase, et qui se distingue des autres structures de re-échantillonnage étudiées par le fait qu'il interface convenablement les deux domaines d'horloge sans utiliser de troisième horloge. Cette méthode a fait l'objet d'une publication dans la revue Springer « Journal of Signal Processing Systems » publiée en septembre 2013 [HRD13].

4.2 Re-échantillonneur pratique sans 3^{ième} horloge et avec contrôle du SFDR

De la théorie à la mise en œuvre matérielle, la méthode de re-échantillonnage arbitraire proposée est détaillée par la suite. Nous verrons aussi dans cette partie comment cette méthode permet de prédire le SFDR.

4.2.1 Considérations théoriques

4.2.1.1 Le re-échantillonneur implémenté

L'architecture proposée utilise des filtres FIR passe-bas comme filtres de reconstruction. Soit $H(n)$ la réponse impulsionnelle discrète du filtre de reconstruction qui va être utilisé dans le processus de re-échantillonnage. Il y a une réponse impulsionnelle continue équivalente $H_C(t)$ telle que si on échantillonne $H_C(t)$, on obtient $H(n)$. Etant donné que le filtrage d'un signal par un filtre FIR consiste à faire une convolution entre le signal et la réponse impulsionnelle du filtre, le re-échantillonneur proposé est basé sur $H_C(t)$. En effet, à partir de la valeur du rapport F_{IN} / F_{OUT} , il est possible de calculer avec précision la position temporelle de chaque échantillon de sortie à partir des positions temporelles des échantillons d'entrée. En d'autres termes, il est possible de calculer les

instants temporels de sortie à partir de μ_k. A partir de ces instants, avec $H_C(t)$, on peut déterminer les coefficients de la réponse impulsionnelle qui doivent pondérer les échantillons d'entrée (figure 4.7). De cette façon, le processus de filtrage peut générer un échantillon de sortie à n'importe quel instant de sortie voulu. L'architecture proposée fournit une implémentation matérielle de ce processus en utilisant une structure similaire au re-échantillonneur de Barker [B08].

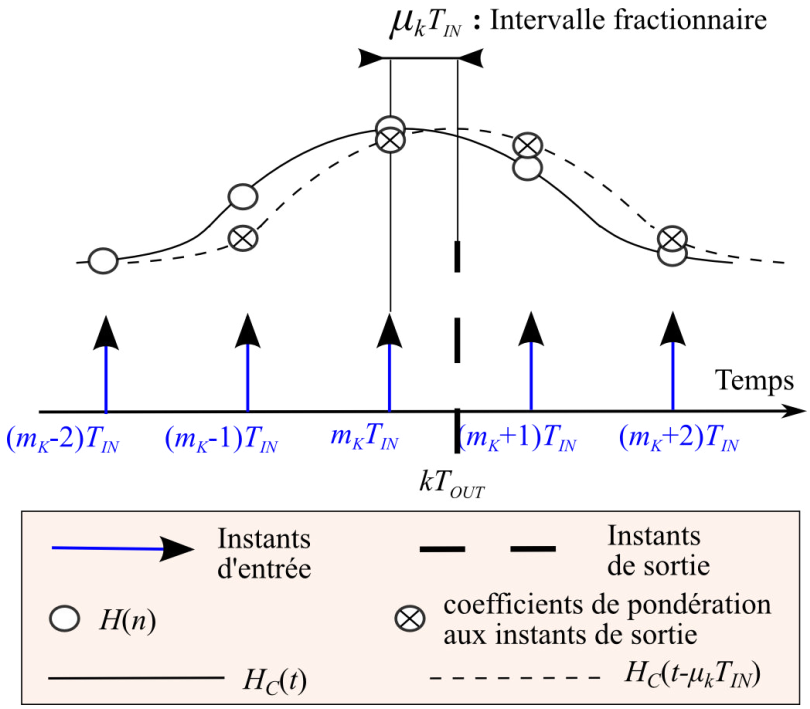


Figure 4.7 – $H(n)$ et $H_C(t)$. Les coefficients de pondération aux instants de sortie sont obtenus après que $H_C(t)$ ait été retardé de $\mu_k T_{IN}$

4.2.1.2 Problèmes

Soit B la moitié de largeur de la bande passante du signal d'entrée et β le rapport $2.B/F_{IN}$. A ce stade, 2 questions, que nous allons décrire dans ce qui suit, demeurent.

- Quel type de filtre allons-nous choisir, et avec quels paramètres?
- Comment allons-nous déterminer les coefficients de pondération à chaque instant de sortie?

Problème N°1: L'utilisateur peut implémenter tout type de filtre FIR passe-bas, à condition que les dimensions spectrales correspondent aux exigences de l'application visée. Par exemple, l'utilisateur peut choisir parmi les filtres de type sinus cardinal (Sinc) pondérés par des fenêtres comme Kaiser [K74], Blackman [OS99], Hann [OS89], fenêtres Nuttall [N81], parmi les filtres dits « frequency sampling-based filters » [M98], ou encore parmi les filtres conçus avec des méthodes d'optimisation tels que les moindres carrés [PB87], Parks-McClellan [A79], ou les moindres carrés contraints [SLB95]. Par exemple, si l'ordre du filtre est faible, les filtres Sinc pondérés par une

fenêtre de Blackman-Harris [H78] réalisent de très faibles ondulations (ripple) dans la bande passante (de l'ordre de 10^{-5} dB), mais avec une large bande de transition, tandis que l'algorithme de Parks-McClellan donne la possibilité de régler la largeur de bande de transition désirée.

Comme le critère d'évaluation dans les applications radio logicielle est souvent le SFDR, réduire le SFDR sera notre principal critère de décision dans le choix d'une architecture appropriée. La valeur du SFDR obtenue avec notre procédé de re-échantillonnage peut être calculée à partir du spectre de la version temps-continu de la réponse impulsionnelle du filtre, $H_C(t)$. En effet, le spectre de $H_C(t)$ s'étend à l'infini car $H_C(t)$ est continue. Ainsi, connaissant ce spectre, on peut évaluer l'influence du filtre sur les images spectrales du signal re-échantillonné, et estimer le SFDR à partir du niveau de réjection du filtre au niveau de chaque image spectrale. En déterminant la réponse impulsionnelle d'un filtre à l'aide d'un logiciel mathématique approprié, on obtient les coefficients de la réponse discrète $H(n)$. Pour approximer le SFDR, on peut sur-échantillonner $H(n)$ par un facteur I élevé afin d'obtenir une nouvelle réponse discrète, $H_I(t)$ qui se rapproche mieux de $H_C(t)$ (l'interpolation nécessaire peut être facilement réalisée avec un logiciel mathématique approprié, avec un I de l'ordre de 10000). Puis, une valeur approximative du SFDR est donnée par l'amplitude maximale de $H_I(t)$ au-delà de la fréquence $2\pi(1-\beta) / I$ (voir figure 4.8). Ainsi, un filtre peut être considéré comme satisfaisant selon le critère SFDR pour une application donnée lorsque, pour un ordre donné, le SFDR estimé est supérieur ou égal au SFDR minimum requis. Dans la pratique, il est prudent d'ajouter une marge de 10 dB pour tenir compte des bruits intervenant dans le processus.

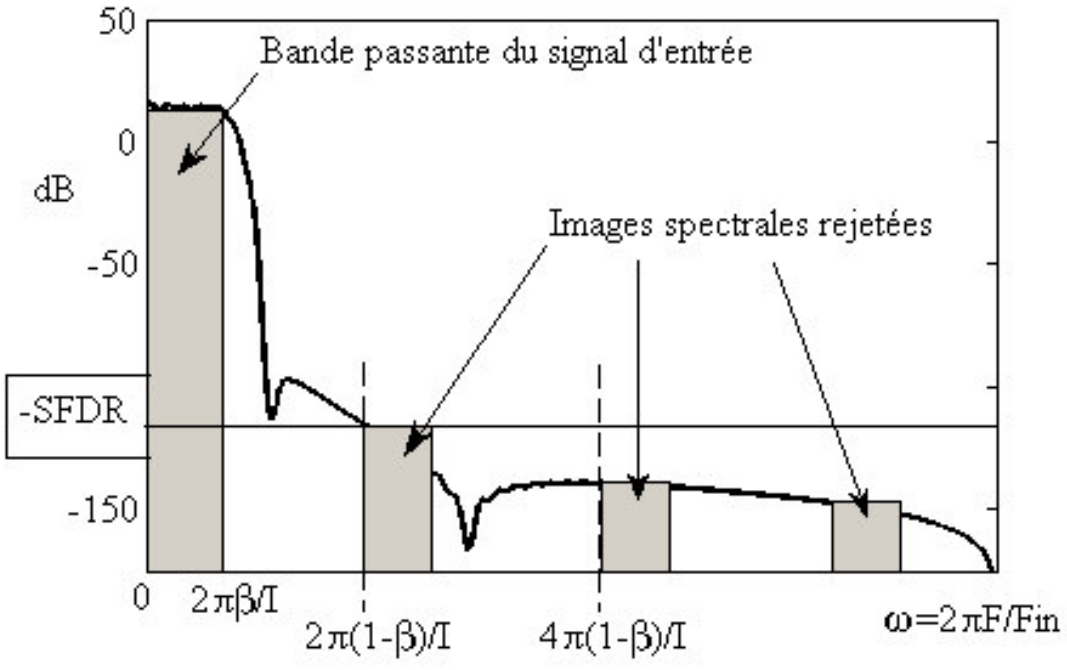


Figure 4.8 – Valeur approximée du SFDR. La courbe représente le spectre de la réponse impulsionnelle du filtre en dB

Problème N°2: Une fois que le filtre est configuré et son ordre déterminé, on utilise $H_I(t)$ à nouveau pour calculer les coefficients de pondération du filtre à chaque instant de sortie. Ces coefficients seront prélevés de $H_I(t)$ par la technique du plus proche voisin. Cependant, cette technique génère une erreur « stair-step » (litt. « en marches d'escalier ») constituée de l'erreur entre le coefficient qui aurait dû pondérer chaque échantillon de signal et le coefficient effectivement choisi. La figure 4.9 montre cette erreur pour l'échantillon $m_K T_{IN}$. Chaque coefficient choisi sera la valeur de l'échantillon de $H_I(t)$ le plus proche de la valeur idéale que nous aurions calculée avec $H_C(t\text{-}\mu_k T_{IN})$. Ainsi, dans ce cas, le facteur I utilisé pour déterminer $H_I(t)$ doit être assez élevé. En effet, nous devons garder le bruit de l'erreur « stair-step » sous le SFDR requis, de sorte que l'équation (4.2) soit satisfaite.

$$I > 0.5 \times 10^{SFDR/20} \tag{4.2}$$

Comme le signal est quantifié avec un nombre de bits limité b, il est parfois nécessaire que le SFDR soit maintenu en dessous du bruit lié à la gamme dynamique du signal codé sur b bits. Cette considération conduit à (4.3):

$$I > 2^{b-1} \tag{4.3}$$

Comme le calcul des coefficients est fait « offline », il n'y a aucune restriction sur la durée du traitement. On peut ainsi atteindre le facteur d'interpolation nécessaire avec un logiciel approprié comme Matlab. Puis, à partir de $H_I(t)$, on obtient les échantillons correspondant aux instants de sortie. Toutefois, pour des filtres dont la réponse impulsionnelle peut s'exprimer de façon analytique, on peut faire différemment. En fait, on peut classer les filtres FIR en deux catégories:

- Ceux qui ont une réponse impulsionnelle dont l'expression analyse est connue, tels que les filtres Sinc pondérés par des fenêtres usuelles comme Blackman, Nutall, Hamming, Kaiser, Tukey [T67] etc.
- Ceux dont l'expression analytique est inconnue: essentiellement ceux générés par une méthode d'optimisation tels que Parks-McClellan, les moindres carrés, etc.

Pour les filtres appartenant à la première catégorie, une fois les paramètres et l'ordre définis, il est possible de déduire les expressions analytiques des valeurs des coefficients à chaque instant de sortie [LVKL96]. Les résultats obtenus à partir de ces expressions analytiques sont ainsi plus précis que ceux obtenus par la méthode d'interpolation décrite précédemment. Afin d'avoir une comparaison non biaisée, l'obtention des coefficients de pondération souhaités s'est faite par interpolation pour les deux types de filtres.

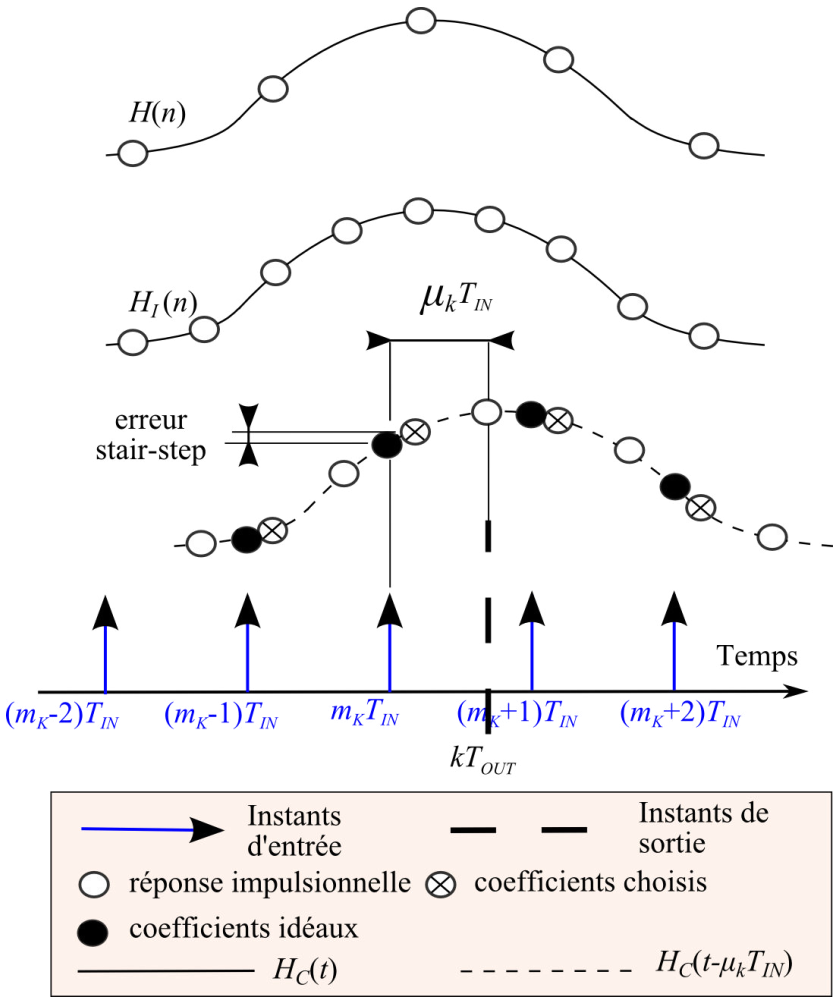


Figure 4.9 – Erreur « stair-step ». Les coefficients idéaux sont ceux qui pondéreraient les échantillons du signal d'entrée si nous avions $H_C(t)$. Les coefficients choisis sont les coefficients de $H_I(t)$ qui sont les plus proches voisins des coefficients idéaux et qui pondèrent les échantillons du signal d'entrée

4.2.1.3 Modélisation Matlab

Afin de vérifier que les SFDRs théoriques correspondraient bien à ceux obtenus après l'implémentation, nous avons déterminé un ensemble de valeurs de SFDRs en utilisant l'approche théorique décrite ci-dessus qui ont été confirmées par des expérimentations sur FPGA et des expérimentations sous Matlab. Ces tests permettent de vérifier la fiabilité de notre modélisation du SFDR.

La figure 4.10 montre le SFDR estimé avec le procédé illustré dans la figure 4.8 pour $\beta = 0.5$ et pour certains filtres connus en fonction de A. Nous verrons dans la section 4.2.3 que les résultats expérimentaux confirment les tendances prévues dans la figure 4.10. La figure 4.10 est donc valable pour toute application ayant $\beta = 0.5$, à condition que le bruit de la gamme dynamique du signal et

des coefficients du filtre soit maintenu en dessous du SFDR requis par le concepteur. Le bruit de la gamme dynamique est représenté sur la figure par la ligne horizontale, ce qui indique la limite au-delà de laquelle les harmoniques parasites sont plus faibles que le bruit de la gamme dynamique, comme le montre la figure 4.11. Au-delà de cette ligne, le SFDR mesurée après implémentation est la différence entre le signal et le bruit de la gamme dynamique.

Dans la suite, les paramètres utilisés pour tracer les courbes – ou pour calculer les coefficients – avec les méthodes Parks-McClellan et les moindres carrés ont été choisis pour s'approcher le plus possible de $\beta = 0.5$ tout s'assurant que l'algorithmes d'optimisation convergent. En effet, certains jeux de paramètres font diverger les algorithmes.

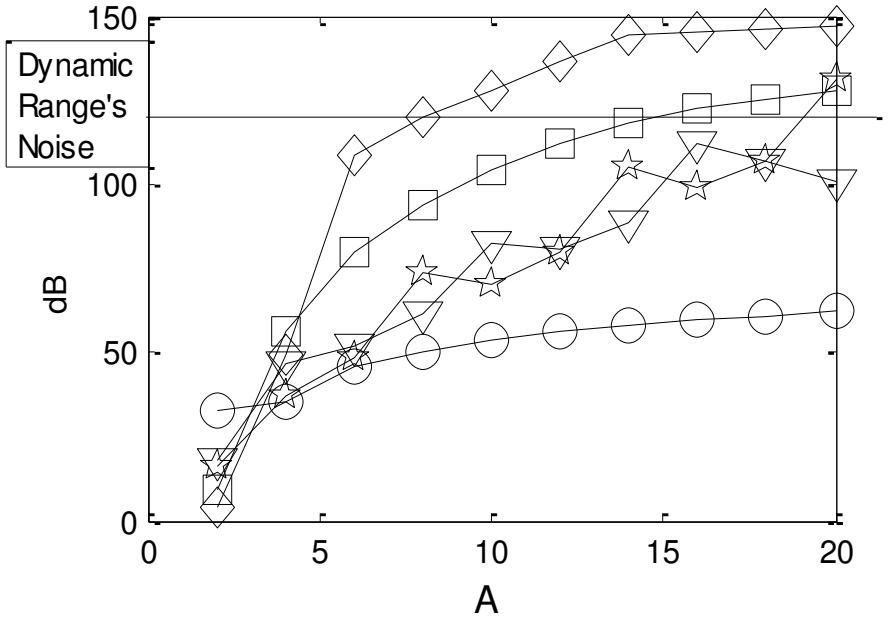


Figure 4.10 – SFDR estimé pour différents filtres connus. $A < 21$. O Rectangulaire, ☆ Parks-McClellan, □ Hann, ▽ Moindres carrés, ◇ Blackman Harris. La ligne horizontale "Dynamic Range Noise" indique la limite supérieure due à la précision du signal et des coefficients. La ligne, dont le niveau dépend de cette précision, fait office de SFDR maximum que les re-échanillonneurs implémentés peuvent atteindre. Les 10 dB de marge mentionnés plus haut n'apparaissent pas dans cette figure

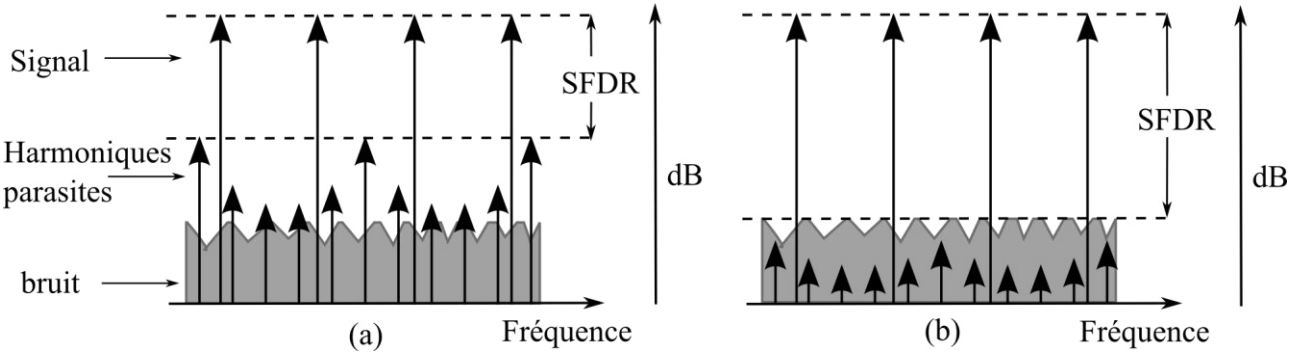


Figure 4.11 – Illustration du SFDR mesuré après implémentation. Les quatre flèches les plus hautes représentent les 4 porteuses qui composent le signal. Les autres flèches représentent les harmoniques parasites. Le gris représente le bruit de gamme dynamique. (a) Les harmoniques sont supérieures au bruit de gamme dynamique. Le SFDR mesuré est alors la différence entre le signal et les harmoniques. (b) Les harmoniques sont inférieures et mélangées au bruit. Le SFDR mesuré est alors la différence entre le signal et le plancher de bruit

4.2.1.4 Conclusion sur l'aspect théorique

Nous venons de développer un modèle théorique de re-échantillonneur qui permet d'estimer le SFDR qu'on obtiendra après re-échantillonnage. Le contrôle du SFDR était l'un des objectifs visés et annoncés pour cette méthode de re-échantillonnage. Maintenant, il nous faut concevoir l'architecture matérielle qui sera à même de mettre en œuvre ce modèle théorique sans utiliser de 3^{ième} horloge, c-à-d une horloge autre que F_{IN} et F_{OUT}. C'est le deuxième objectif annoncé à atteindre, et c'est ce qui fait l'objet de la section suivante (section 4.2.2).

4.2.2 Architecture

Soient Q et N respectivement le numérateur et le dénominateur de la version simplifiée de la fraction F_{IN}/F_{OUT}. On remarque que, lorsque k augmente de 0 à + ∞, les valeurs μ_k se répètent avec une période N. Par exemple, si on re-échantillonne un signal de 87 MHz à 51,2 MHz, nous aurons 87/51.2 = 435/256. Dans cet exemple, lorsque k augmente, μ_k évolue périodiquement avec une période de 256. Notre implémentation matérielle fait usage de la périodicité du μ_k. Par conséquent, il suffit de stocker les valeurs des coefficients du filtre correspondant à une période de μ_k. Soit A un entier tel que (2A +1) est l'ordre du filtre de reconstruction. Pour l'exemple précédent, nous aurions à stocker 256(2A +1) valeurs.

4.2.2.1 Le sous – échantillonneur

L'architecture proposée est présentée dans la figure 4.12. Les valeurs des coefficients sont

stockées dans (2A+1) mémoires mortes (ROM), dont chacune est de taille N. Nous rappelons d'après la figure 4.3 que chaque μ_k est associé à un échantillon de sortie ; on sait ainsi exactement la position des échantillons de sortie par rapport aux échantillons d'entrée pendant une période de μ_k. Par conséquent, afin de calculer un échantillon de sortie, chaque étage de ROM contient une valeur de coefficient qui pondère un échantillon d'entrée à un instant de sortie donné durant cette période. Autrement dit, au $i^{ème}$ instant d'une période, les $i^{èmes}$ emplacements des ROM 1 à (A+1) pondèrent les (A+1) échantillons d'entrée à gauche de l'échantillon de sortie en train d'être calculé, et les $i^{èmes}$ emplacements des ROM (A+2) à (2A+1) pondèrent les A échantillons d'entrée à droite. Les échantillons d'entrée pondérés sont ensuite additionnés par un additionneur parallèle pour produire l'échantillon de sortie.

On note que le calcul de l'échantillon de sortie est entièrement synchronisé sur F_{IN}. La FIFO "dual clock" fait passer les échantillons de sortie calculés à F_{IN} au domaine d'horloge F_{OUT}. Appelons D_{EP} la profondeur de cette FIFO. Afin de résoudre le problème du ratio arbitraire, le bloc "Contrôleur Maitre", qui est le cœur de la structure, contrôle précisément l'incrémentation des index des ROM quand un nouvel échantillon de sortie doit être calculé.

Le bloc Contrôleur Maitre permet également d'éviter le débordement de la FIFO. En effet la figure 4.12 montre que, si une donnée est stockée dans la FIFO à chaque front montant de F_{IN}, la FIFO débordera car $F_{IN} > F_{OUT}$. Pour éviter le débordement, le bloc Contrôleur Maitre garantit que seuls les échantillons de sortie valides entrent dans la FIFO via le signal « Requête Ecriture FIFO ». Pour remplir ces deux rôles, le bloc Contrôleur Maitre garde en mémoire une version simplifiée de la « carte de sous-échantillonnage » des échantillons de sortie par rapport aux échantillons d'entrée (voir la figure 4.13). Cette carte est un vecteur binaire de longueur Q. Comme le montre la figure 4.13, un élément de la carte est égal à 1 si l'instant d'entrée précède immédiatement un temps de sortie. Sinon, il est égal à 0. Le bloc Contrôleur Maitre utilise la carte pour augmenter les index des ROMs au moment approprié et pour assigner le signal « Requête Ecriture FIFO ». Ainsi, même si l'écriture dans la FIFO est cadencée par F_{IN}, elle se fait à cadence équivalente à F_{OUT}. Il est facile de déterminer la configuration appropriée de la carte avec un logiciel mathématique tel que Matlab.

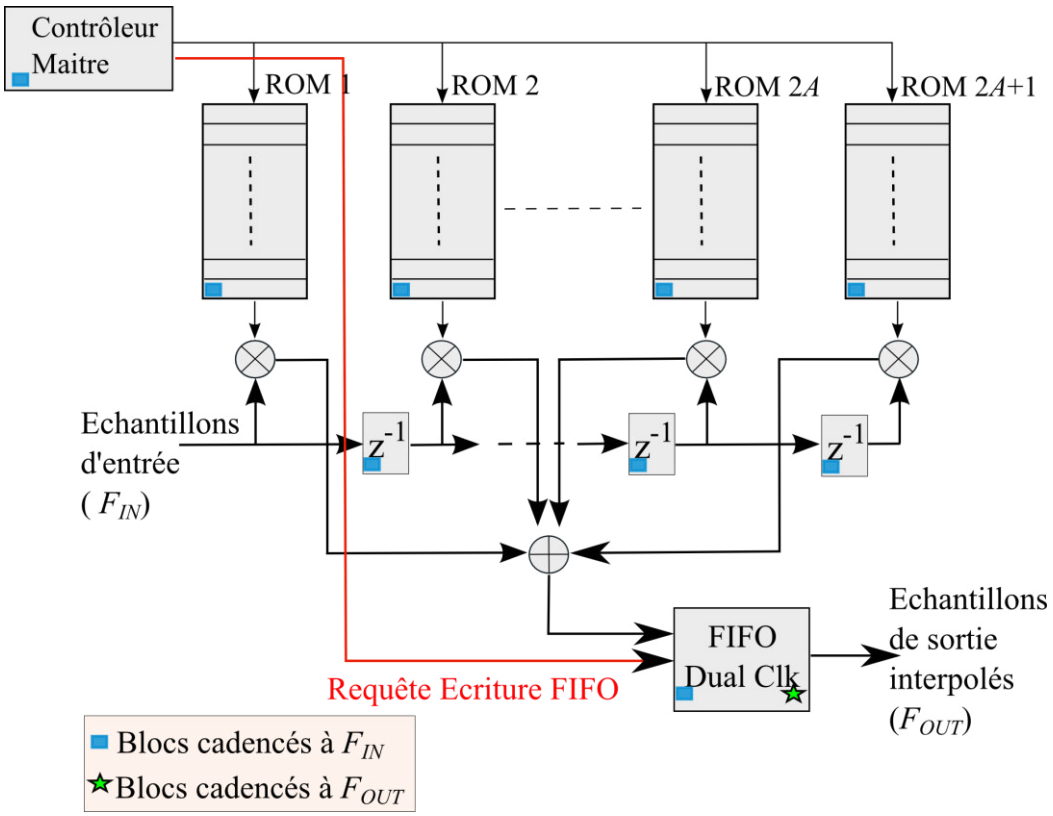


Figure 4.12 – Architecture du sous-échantillonneur arbitraire

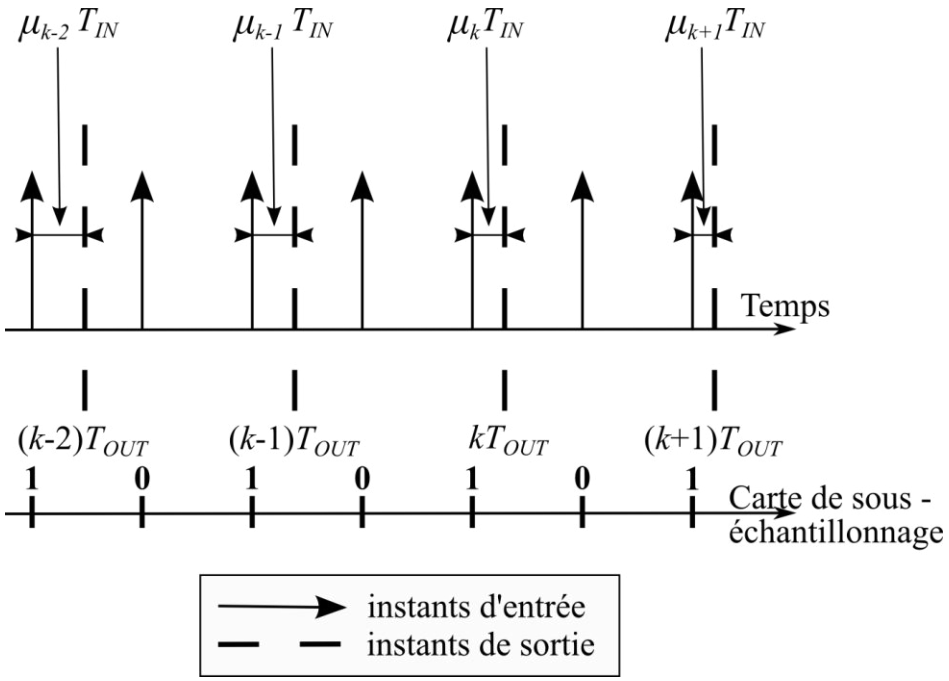


Figure 4.13 – Carte de sous-échantillonnage

4.2.2.2 Le sur – échantillonneur

La structure du sur-échantillonneur est similaire à celle du sous-échantillonneur (voir la figure

4.14). La FIFO conserve son rôle d'interface entre les deux domaines d'horloge. Le bloc Contrôleur Maitre gère la lecture de la FIFO de manière à éviter un « underflow ». En effet, en utilisant une «carte de sur-échantillonnage", le bloc Contrôleur Maitre garantit que les index des ROMs sont augmentés et que la FIFO est lue aux instants appropriés. La carte binaire de sur-échantillonnage n'est évidemment pas identique à celle du sous-échantillonnage (figure 4.13). Tandis que pour la carte du sous-échantillonnage est synchronisée sur les instants d'entrée, la carte du sur-échantillonnage est synchronisée sur les instants de sortie. Quand un élément de la carte de sur-échantillonnage est égal à 1, il faut incrémenter l'index des ROMs. Le bloc Contrôleur Maitre active également les registres "REG e" qui sont des registres avec une entrée « actif ». Quand « actif » est à 0, la sortie de "REG e" conserve sa valeur précédente. Sinon, il fonctionne comme une simple bascule D. Cela est utile quand on calcule plusieurs échantillons de sortie à partir des mêmes échantillons d'entrée.

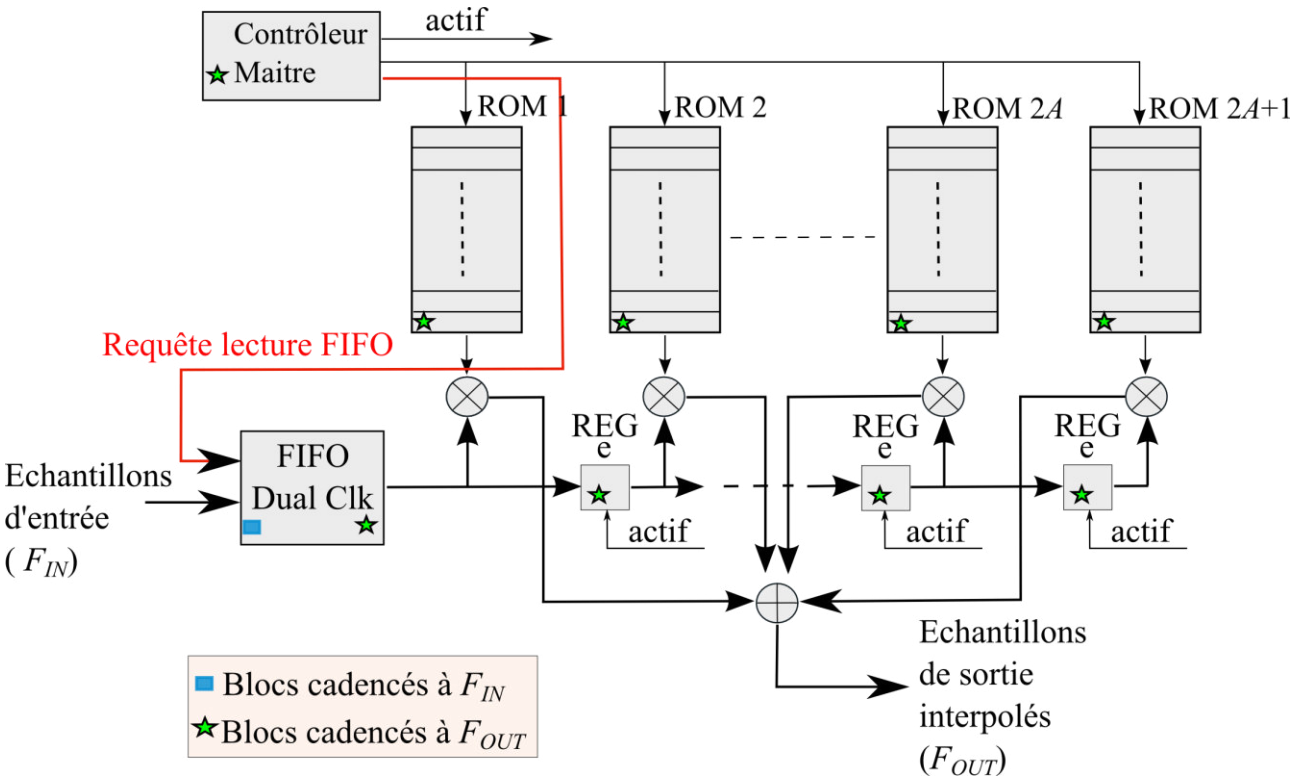


Figure 4.14 – Architecture du sur-échantillonneur arbitraire

4.2.2.3 Estimation des ressources avant implémentation

Pour résumer, le re-échantillonneur utilise $(2A+1)\times N+1+D_{EP}$ emplacements mémoires, répartis en $(2A+1)\times N$ pour les coefficients du filtre, 1 pour la carte et D_{EP} pour le FIFO « dual clock ». Au cours du processus de conception, il est facile d'estimer l'utilisation des ressources avant d'entrer

dans la phase d'implémentation. En effet, D_{EP} peut prendre n'importe quelle valeur fixée par le concepteur. Une fois que A est déduit de la figure 4.10, D_{EP} fixé, et N et Q déterminés, l'estimation des ressources requises peut être obtenu en utilisant le tableau 4.2 où les échantillons d'entrée et les coefficients du filtre sont codés sur b bits.

Mémoire pour le sous-échantillonnage (en bits)	$(2A+1) \times b \times N + Q + (2b + x) \times D_{EP}$
Mémoire pour le sur-échantillonnage (en bits)	$((2A+1) \times b + 1) \times N + b \times D_{EP}$
Multiplieurs $b \times b$ – bits	$2A+1$
Additionneurs	1 additionneur parallèle $(2A+1) \times 2b$ bits

Tableau 4.2 – Estimation des ressources avant implémentation. $2^{x-1}<2A<2^x$

Avant de conclure cette section, nous soulignons que, dans le tableau 4.2, la carte de "sous-échantillonnage" ou de "sur-échantillonnage" est considérée comme un élément mémoire. En outre, les multiplieurs de $b \times b$ – bits et les additionneurs parallèles peuvent bien sûr être implémentés de nombreuses manières différentes (avec des blocs DSP ou avec des éléments logiques), en fonction du type FPGA et l'implémentation choisie par le concepteur.

4.2.3 Tests et validation

Cette partie montre les tests que nous avons fait pour montrer la viabilité de la méthode. Il s'agit donc de tests expérimentaux ainsi que la comparaison des résultats obtenus avec les prévisions théoriques.

4.2.3.1 Le sous – échantillonneur

On se propose, par exemple, de mettre en place un sous-échantillonneur pour traiter la bande VHF II. Pour rester dans le cas déjà abordé dans la section 4.2.2, et dans le contexte du récepteur FM (section 3.3), prenons la bande FM en Europe occidentale qui s'étend de 87.5 à 108 MHz, en re-échantillonnant la bande 87 – 108.75 MHz. Toujours pour demeurer dans le cas de figure de la section 4.2.2 où l'ADC échantillonne à 87 MHz, nous faisons un sous-échantillonnage de 87 MHz à 51,2 MHz. Les échantillons d'entrée sont codés sur 12 bits. Comme la largeur de bande du signal d'entrée est de 21,75 MHz, nous avons $F_{IN} = 4.B$ (où B est la moitié de la largeur de bande du signal d'entrée) et donc $\beta = 0.5$. Comme $\beta = 0.5$, ce test se réalise dans les mêmes conditions que les prédictions de la figure 4.10. Fixons le SFDR au-dessous du plancher de bruit: SFDR $\geq$ 72 dB pour 12 bits. Ainsi, l'expression (4.3) donne $I_2 > 2048$. Selon la figure 4.10, on choisit $A = 6$ avec une

fenêtre de Blackman-Harris. Il est clair de calculer $N = 256$, et nous choisissons aussi de fixer D_{EP} à 512.

4.2.3.1.1 Implémentation sur FPGA

L'implémentation du design est réalisée sur un FPGA de type Altera Stratix II EP2S180F1020C5 utilisant une carte de développement Stratix II DSP. Le système est entièrement conçu avec le logiciel Altera Quartus II version 9.1, ainsi que la synthèse et l'implémentation. Le tableau 4.3 résume les ressources matérielles utilisées.

ALUTs	298/143.520 (<1%)
Registres logiques dédiés	560/143.520 (<1%)
Mémoire (en bits)	54.707/9.383.040 (<1%)
blocs DSP 9-bit	26/768 (3%)
F_{IN} maximal	120.83 MHz
F_{OUT} maximal	146.09 MHz

Tableau 4.3 – Ressources du sous-échantillonneur

4.2.3.1.2 Résultats

Le signal de test d'entrée est constitué de la somme de 4 sinusoïdes. Leurs fréquences porteuses respectives sont 88, 94, 100 et 108 MHz. Par repliement spectral, 88, 94, 100 et 108 MHz sont ramenés respectivement ramenées à 1, 7, 13 et 21 MHz après un échantillonnage à 87 MHz. Les échantillons d'entrée et les valeurs des coefficients du filtre sont sur 12 bits. La figure 4.15 montre le spectre de 65536 échantillons prélevés du signal de sortie obtenu de l'implémentation FPGA, via un analyseur logique.

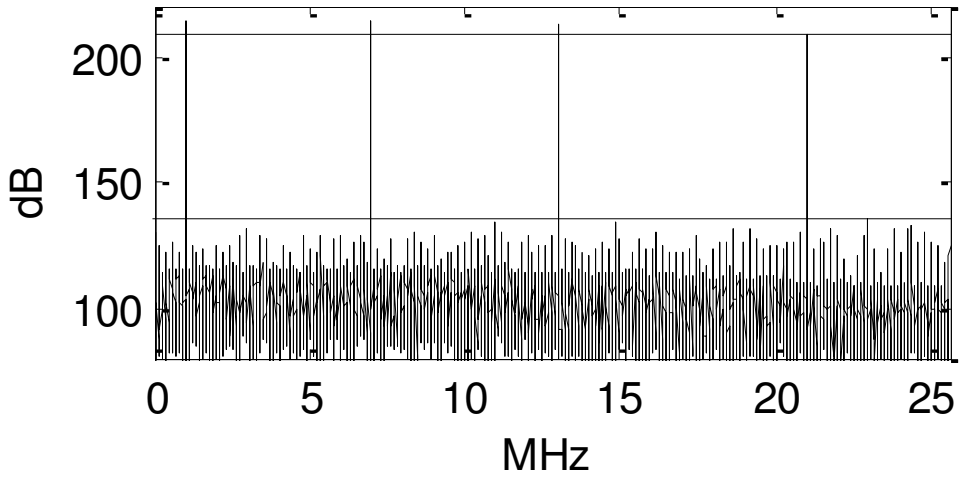


Figure 4.15 – Résultats d'implémentation montrant le spectre du signal sur-échantillonné lorsque F_{IN} = 87 MHz et F_{IN} = 51.2 MHz. « dy », qui est la différence de niveaux entre les curseurs horizontaux, est égale à 73.5 dB

4.2.3.1.3 Comparaison entre théorie et résultats expérimentaux

La comparaison sera basée sur deux points: le SFDR et la quantité de ressources utilisées. En ce qui concerne le premier, comme le montre la figure 4.15, on a dy ≈ 73.5 dB et les harmoniques parasites sont maintenues au-dessous du plancher de bruit, comme prévu. Ensuite, en accord avec le tableau 4.2, les ressources du sous-échantillonneur sont données dans le tableau 4.4.

Mémoire (en bits)	((2x6+1)×12)×256+435+(2×12+4) ×512=54707
Multiplieurs 12 x12 – bits	13
Additionneur parallèle 13 × 24 – bits	1

Tableau 4.4 – Comparaison entre la théorie et les résultats expérimentaux : Sous-échantillonneur

Selon le type FPGA et le design choisis par l'utilisateur, les multiplieurs sont implémentés différemment. Dans notre cas, nous avons implémentés les multiplieurs uniquement avec des blocs DSP. Sous Quartus II version 9.1, les blocs DSP 18x18 sont synthétisés pour remplacer les blocs DSP 12x12, qui ne sont pas disponibles dans le Stratix II. Un bloc DSP 18x18 est constitué de deux blocs 9x9. De plus, Quartus implémente des additionneurs parallèles sur Stratix II en utilisant des ALUTs (Adaptive Look-Up Table) et des registres logiques. Prenant en compte ces considérations, les prédictions du tableau 4.4 sont reformulées dans le tableau 4.5.

Mémoire (en bits)	54707
Multiplieurs 18x18 – bits	13
Multiplieurs 9x9 – bits	26
Additionneur parallèle 17x24 – bits	1

Tableau 4.5 – Tableau des ressources du sous-échantillonneur implémenté

Si l'on compare le tableau 4.5 avec les résultats de l'implémentation indiqués dans le tableau 4.3, nous voyons que le sous-échantillonneur implémenté utilise la même quantité de ressources mémoire et des multiplieurs que prévu. Les ressources utilisées par l'additionneur parallèle et toute autre partie du sous-échantillonneur sont reportées dans le tableau 4.3 dans les cases ALUTs et registres.

Deux séries de tests ont été effectuées sur le système de sous-échantillonnage pour confirmer les résultats de SFDR prédits par la théorie dans la figure 4.10, dans le cas où $A \leq 20$. La première série consistait à mesurer le SFDR généré par le précédent signal d'entrée (à savoir, la somme des quatre sinusoïdes) pour différentes implémentations FPGA du re-échantillonneur (c'est-à-dire différents types de filtres, avec $A \leq 20$). Les résultats sont présentés sur la figure 4.16.

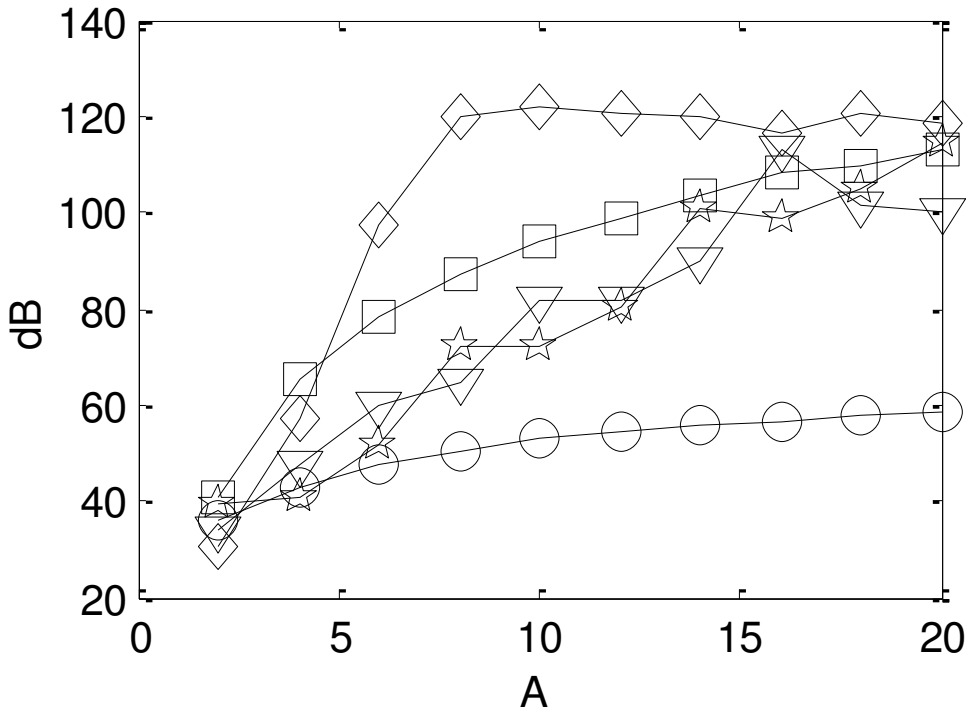


Figure 4.16 – SFDR expérimentaux obtenus pour le sous-échantillonnage lorsque $A \leq 20$. O Rectangulaire, ☆ Parks-McClellan, □ Hann, ▽ Moindres carrés, ◇ Blackman Harris. Le plancher de bruit de 116 dB pour une précision de 19 bits explique pourquoi les résultats obtenus sont plafonnés par ce niveau

Ces résultats expérimentaux peuvent ensuite être comparés aux prédictions théoriques. Avant

cela, cependant, afin de nous assurer que les résultats de la figure 4.16 ne sont pas le résultat d'un choix fortuit de fréquences F_{IN} et F_{OUT}, dans notre deuxième série de tests, nous avons construit un banc d'essai Matlab qui émule précisément notre système de sous-échantillonnage et donne donc des résultats identiques à ce que le matériel de sous-échantillonnage système donnerait. Ce banc de test a déterminé le SFDR produit par notre système de sous-échantillonnage pour 4 choix différents de signaux d'entrée, et pour 2 valeurs différentes du rapport F_{OUT} / F_{IN}, avec $\beta = 0.5$. Ici, chaque signal d'entrée est une somme de dix sinusoïdes dont les fréquences sont choisies de telle sorte que $F_{IN} = 4B$. Le signal d'entrée et les valeurs de pondération sont codés sur 19 bits et I_2 est choisie selon l'équation (4.3). Le tableau 4.6 résume les essais effectués.

	Fréquences des porteuses (MHz)	F_{IN} (MHz)	F_{OUT} (MHz)	F_{OUT} /F_{IN}
Signal 1	$1+C \times 2.2$	87	51.2	0.588506
Signal 2	$1.95+C \times 2.2$	87	51.2	0.588506
Signal 3	$0.8+ C \times 2.3$	86	74	0.860465
Signal 4	$0.3+ C \times 2.3$	86	74	0.860465

Tableau 4.6 – Tableau répertoriant les tests du sous-échantillonneur, avec $C = \{0,1\ldots,9\}$

Les résultats des deux séries de tests sont résumés dans la figure 4.17, où chaque ensemble de courbes est comparé au résultat théorique correspondant que nous avons vu dans la figure 4.10. Comme expliqué à la fin de la section 4.2.1.3, ce résultat théorique est plafonné par une valeur de SFDR imposée par le plancher de bruit, qui est d'environ 116 dB pour 19 bits. Pour chaque filtre testé, les erreurs entre les SFDRs résultants et les SFDRS théoriques sont tracées dans la figure 4.18. La figure 4.18 montre ainsi que les résultats donnés par le système de sous-échantillonnage sont globalement appréciables car ces erreurs vont rarement au-dessous de -10 dB. En outre, cela justifie le choix d'une marge de 10 dB, telle que mentionnée dans la section 4.2.1.3. Les résultats de l'expérimentation matérielle sont donc en accord avec la théorie.

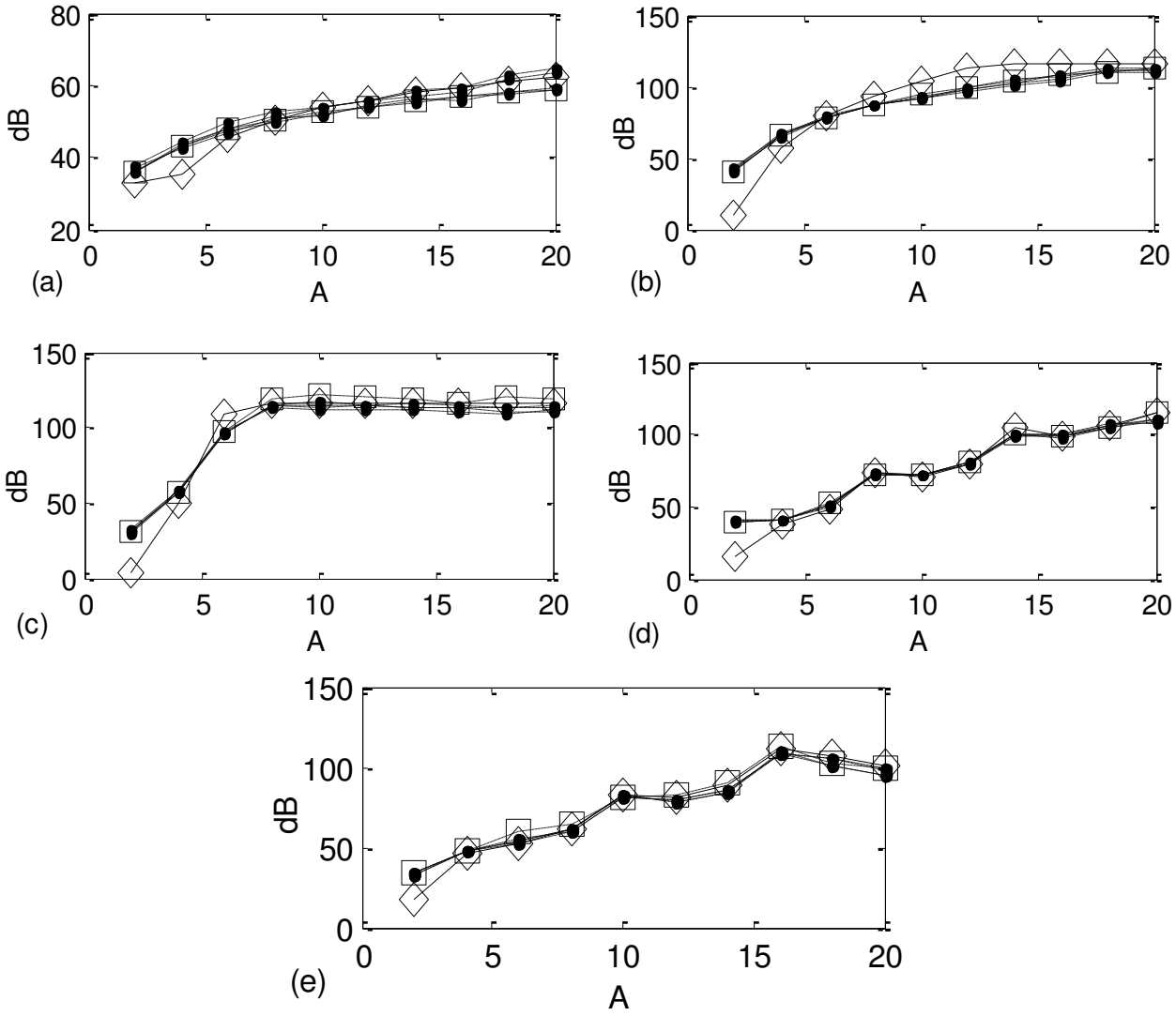


Figure 4.17 – SFDR expérimentaux et théoriques pour le sous-échantillonnage lorsque $A \leq 20$. □ Résultats expérimentaux du FPGA, ◇ SFDR théorique, ● résultats de la simulation. (a) Fenêtre rectangulaire. (b) Fenêtre de Hann. (c) Fenêtre de Blackman Harris. (d) Parks-McClellan. (e) Moindres carrés. Le plancher de bruit de 116 dB pour une précision de 19 bits explique pourquoi les résultats obtenus sont plafonnés par ce niveau

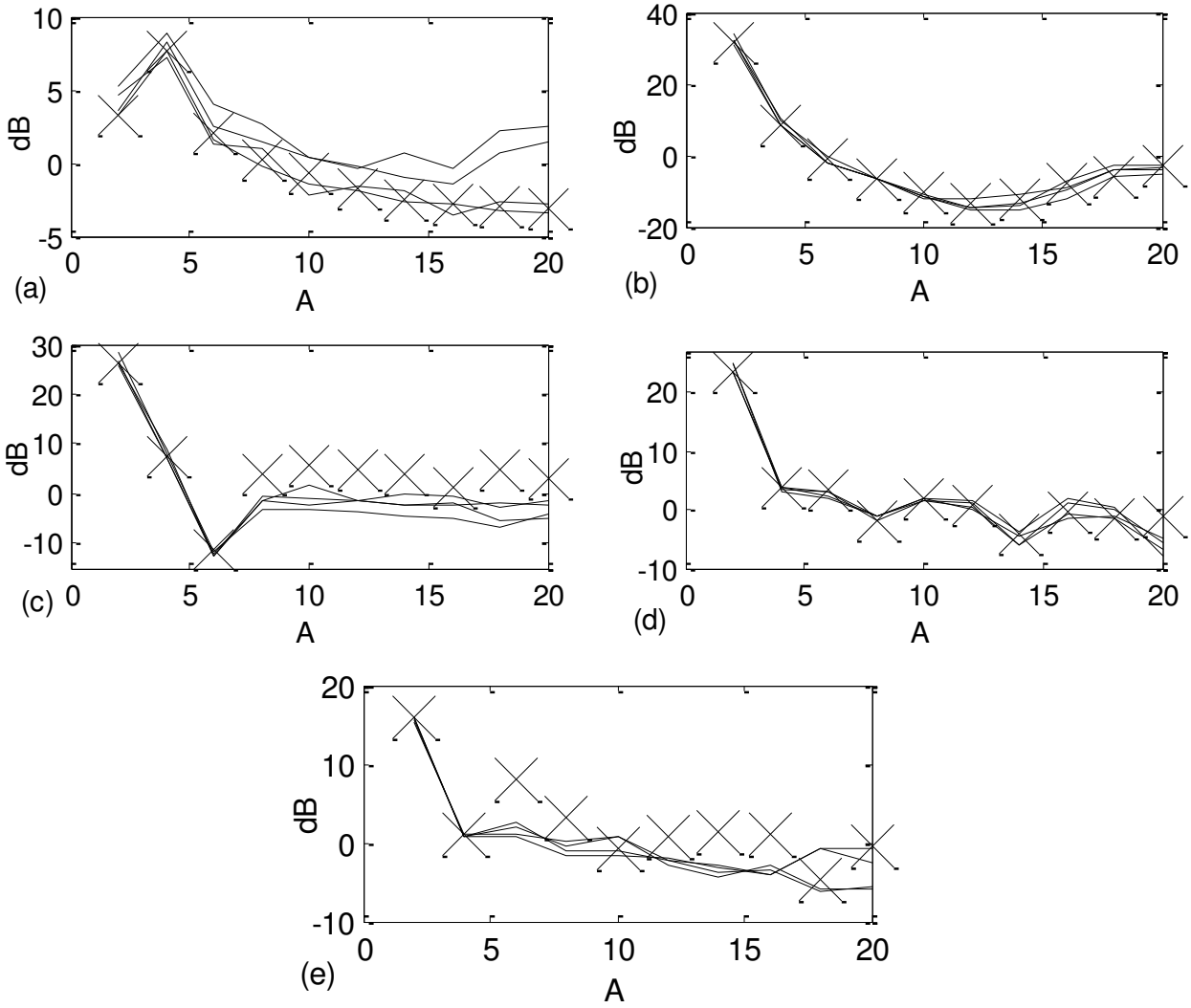


Figure 4.18 – Erreurs entre les SFDRs reportés dans la figure 4.17 et les SFDRs théoriques de la figure 4.10. × Erreur des résultats expérimentaux du FPGA, —— Erreur des résultats de simulation. (a) Fenêtre rectangulaire. (b) Fenêtre de Hann. (c) Fenêtre de Blackman Harris. (d) Parks-McClellan. (e) Moindres carrés

Le sous-échantillonneur proposé remplit bien les objectifs fixés, à savoir le contrôle du SFDR, la non utilisation d'une 3^{ième} horloge et la prédiction de la quantité des ressources requises. Néanmoins, il souffre de certaines faiblesses. La première est qu'il ne supporte pas de déviation entre les fréquences moyennes de l'horloge d'entrée et de l'horloge de sortie. La deuxième est que l'on peut se retrouver dans des cas où la quantité de mémoire à allouer aux ROMs est critique (voire supérieure à la quantité disponible) pour le FPGA utilisé. Cela est dû au fait que le nombre d'étages N dans chaque ROM est le numérateur de la version simplifiée du ratio F_{IN}/F_{OUT} et certains jeux de valeurs de F_{IN} et F_{OUT} peuvent induire une valeur de N très grande pour le FPGA.

4.2.3.2 Le sur – échantillonneur

Dans cette section, nous proposons un sur-échantillonneur pour traiter une partie du spectre qui s'étend de 0 à 10 MHz. Les paramètres de sur-échantillonnage sont F_{IN} = 43.5 MHz et F_{OUT} = 50

MHz et le signal d'entrée de test est le même que comme ci-dessus $\beta = 0.5$. Les échantillons d'entrée et les valeurs des coefficients du filtre sont ici sur 12 bits. Une fois de plus, on cherche à fixer le SFDR en dessous du plancher de bruit: SFDR ≥ 72 dB pour 12 bits. Utilisons maintenant un filtre conçu par une méthode d'optimisation. Selon la figure 4.10, on peut choisir $A = 10$ avec un filtre optimisé par la méthode des moindres carrés. De toute évidence, $N = 100$ et nous choisissons $D_{EP} = 512$.

4.2.3.2.1 Implémentation sur FPGA

Là encore est démontrée la légitimité de la conception d'un re-échantillonneur sans troisième horloge dont la fréquence est un multiple de celle de l'horloge d'entrée ou de sortie. Imaginons que le circuit ait été réalisé sur un FPGA de type Altera Cyclone II EP2C20F484C7N en utilisant une carte de développement « Cyclone II FPGA starter development board », populaire en milieu académique. Cette carte est équipée de trois oscillateurs dont celui qui a la plus haute fréquence génère une horloge de fréquence de 50 MHz. Ces oscillateurs sont les seules sources d'horloge possibles du FPGA. Bien que le Cyclone II puisse supporter des fréquences d'horloge plus élevées, nous aurions été, dans ce cas, confrontés à une situation où utiliser une horloge plus rapide que 50 MHz n'est pas possible. Vu que $F_{OUT} = 50$ MHz, utiliser une troisième horloge aurait été impossible.

En réalité, le circuit a été réalisé sur un FPGA de type Altera Stratix II EP2S180F1020C5 utilisant une carte de développement Stratix II DSP. Le système a été entièrement conçu avec le logiciel Altera Quartus II version 9.1, ainsi que la synthèse et l'implémentation. Le tableau 4.7 résume les ressources matérielles utilisées.

ALUTs	366/143,520 (<1%)
Registres logiques dédiés	625/143.520 (<1%)
Mémoire (en bits)	31.444/9.383.040 (<1%)
blocs DSP 9-bit	42/768 (3%)
F_{IN} maximal	265.96 MHz
F_{OUT} maximal	126.76 MHz

Tableau 4.7 – Ressources du sur-échantillonneur

4.2.3.2.2 Résultats

Le signal de test d'entrée est la somme de quatre sinusoïdes ayant pour fréquences respectives 0,05, 3,5, 6,5 et 10,87 MHz. La figure 4.19 montre le spectre de 65536 échantillons prélevés du

signal de sortie obtenu de l'implémentation FPGA.

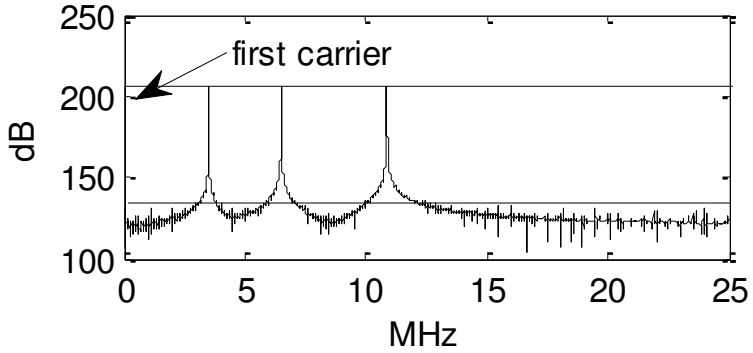


Figure 4.19 – Résultats d'implémentation montrant le spectre du signal sur-échantillonné lorsque F_{IN} = 43.5 MHz et F_{OUT} = 50 MHz. « dy », qui est la différence de niveaux entre les curseurs horizontaux, est égale à 72.9 dB

4.2.3.2.3 Comparaison entre théorie et résultats expérimentaux

A nouveau, la comparaison sera basée sur deux points: le SFDR et la quantité de ressources utilisées. En ce qui concerne le premier, comme le montre la figure 4.19, on a dy ≈ 73 dB et les harmoniques parasites sont maintenues au-dessous du plancher de bruit, comme prévu. Ensuite, en accord avec le tableau 4.2, les ressources du sur-échantillonneur sont données dans le tableau 4.8.

Mémoire (en bits)	((2x10+1)×12+1)×100+(12×512)=31444
Multiplieurs 12 x12 – bits	21
Additionneur parallèle 21 × 24 – bits	1

Tableau 4.8 – Comparaison entre la théorie et les résultats expérimentaux : Sur-échantillonneur

Les conditions de tests étant les mêmes que précédemment, les prédictions du tableau 4.8 sont reformulées dans le tableau 4.9 (voir la section 4.2.3.1.3 pour plus d'explications).

Mémoire (en bits)	31444
Multiplieurs 18x18 – bits	21
Multiplieurs 9x9 – bits	42
Additionneur parallèle 21 × 24 – bits	1

Tableau 4.9 – Tableau des ressources du sur-échantillonneur implémenté

Si l'on compare le tableau 4.9 avec les résultats de l'implémentation indiqués dans le tableau 4.7, nous voyons que le sur-échantillonneur implémenté utilise la même quantité de ressources mémoire et des multiplieurs que prévu. Les ressources utilisées par l'additionneur parallèle et toute

autre partie du sous-échantillonneur sont reportées dans le tableau 4.7 dans les cases ALUTs et registres.

Deux séries de tests, les mêmes que celles sur le système de sous-échantillonnage, ont été effectuées sur le système de sur-échantillonnage afin de confirmer les résultats de SFDR prédits par la théorie. La première série consistait à mesurer le SFDR généré par un signal constitué de la somme de quatre sinusoïdes pour différents types de filtres, avec $A \leq 20$. Les résultats de l'implémentation matérielle sont présentés sur la figure 4.20.

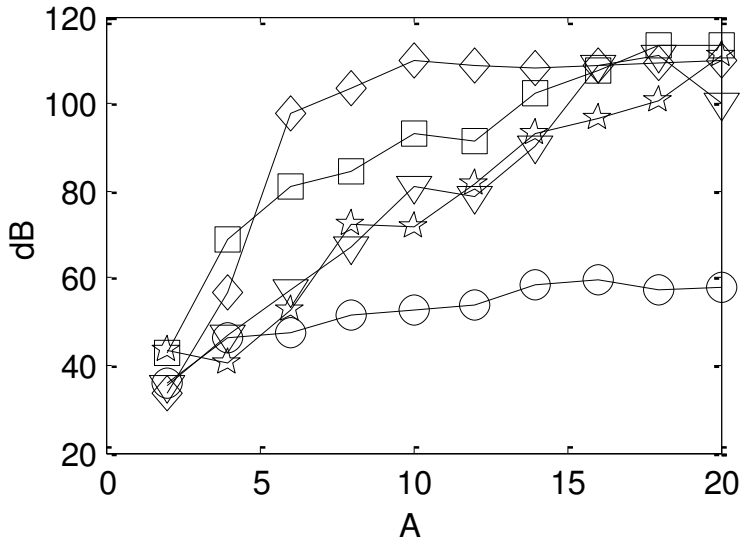


Figure 4.20 – SFDR expérimentaux obtenus pour le sur-échantillonnage lorsque $A \leq 20$. O Rectangulaire, ☆ Parks-McClellan, □ Hann, ▽ Moindres carrés, ◇ Blackman Harris

Le banc d'essai Matlab a déterminé le SFDR produit par le système de sur-échantillonnage pour 4 choix différents de signaux d'entrée, et pour 2 valeurs différentes du rapport F_{OUT} / F_{IN}, avec $\beta = 0.5$. Chaque signal d'entrée était à nouveau une somme de dix sinusoïdes dont les fréquences étaient choisies de telle sorte que $F_{IN} = 4B$. Le signal d'entrée et les valeurs de pondération ont été codés sur 19 bits. Le tableau 4.10 résume les essais effectués.

	Fréquences des porteuses (MHz)	F_{IN} (MHz)	F_{OUT} (MHz)	F_{OUT} / F_{IN}
Signal 1	$0.8+C \times 2.3$	86	148	0.58108
Signal 2	$0.3+D \times 2.3$, 19	86	148	0.58108
Signal 3	$1.95+ C \times 2.2$	87	100	0.87
Signal 4	$1.3+ C \times 2.2$	87	100	0.87

Tableau 4.10 – Tableau répertoriant les tests du sur-échantillonneur. On a $C = \{0,1\ldots,9\}$, $D = \{0,1\ldots,8\}$

Les résultats des simulations sont résumés dans la figure 4.21 pour chaque filtre testé. La figure 4.21 présente les résultats des tests en comparaison avec les SFDRs minimums prédits. Comme

pour le précédent cas du sous-échantillonnage, ces SFDRs théoriques sont plafonnés par une valeur imposée par le plancher de bruit, qui est d'environ 116 dB pour 19 bits, comme c'est expliqué à la fin de la section 4.2.1.3. Pour chaque filtre testé, les erreurs entre les SFDRs résultants et les SFDRS théoriques sont tracées dans la figure 4.22. La figure 4.22 montre ainsi que les résultats donnés par le système de sur-échantillonnage sont, ici aussi, globalement appréciables car ces erreurs vont rarement au-dessous de -10 dB. Une marge de 10 dB permet par conséquent d'avoir une estimation fiable du SFDR. Les résultats de l'expérimentation matérielle sont donc en accord avec la théorie.

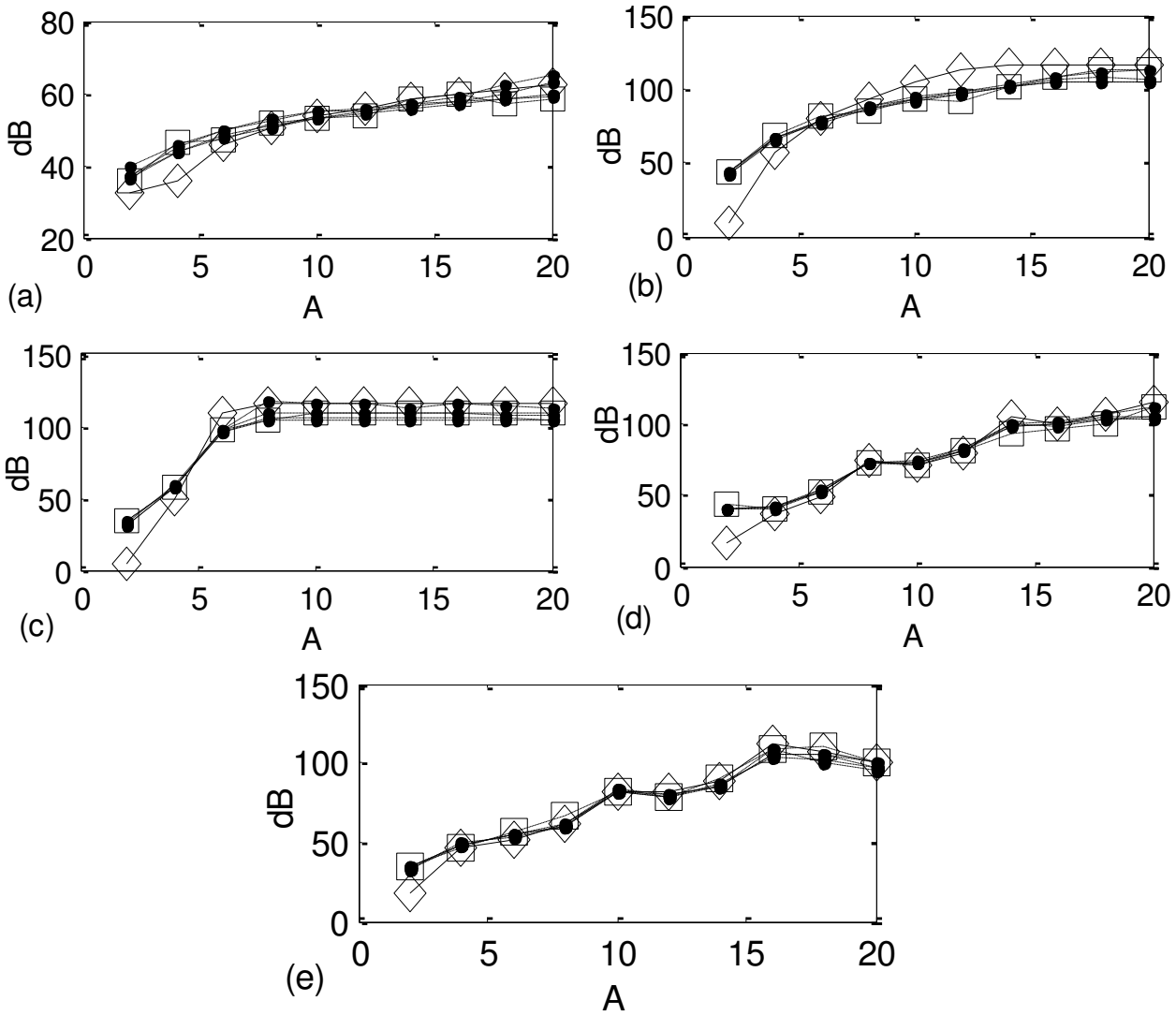


Figure 4.21 – SFDR expérimentaux et théoriques pour le sur-échantillonnage lorsque $A \leq 20$. □ Résultats expérimentaux du FPGA, ◇ SFDR théorique, ● résultats de la simulation. (a) Fenêtre rectangulaire. (b) Fenêtre de Hann. (c) Fenêtre de Blackman Harris. (d) Parks-McClellan. (e) Moindres carrés

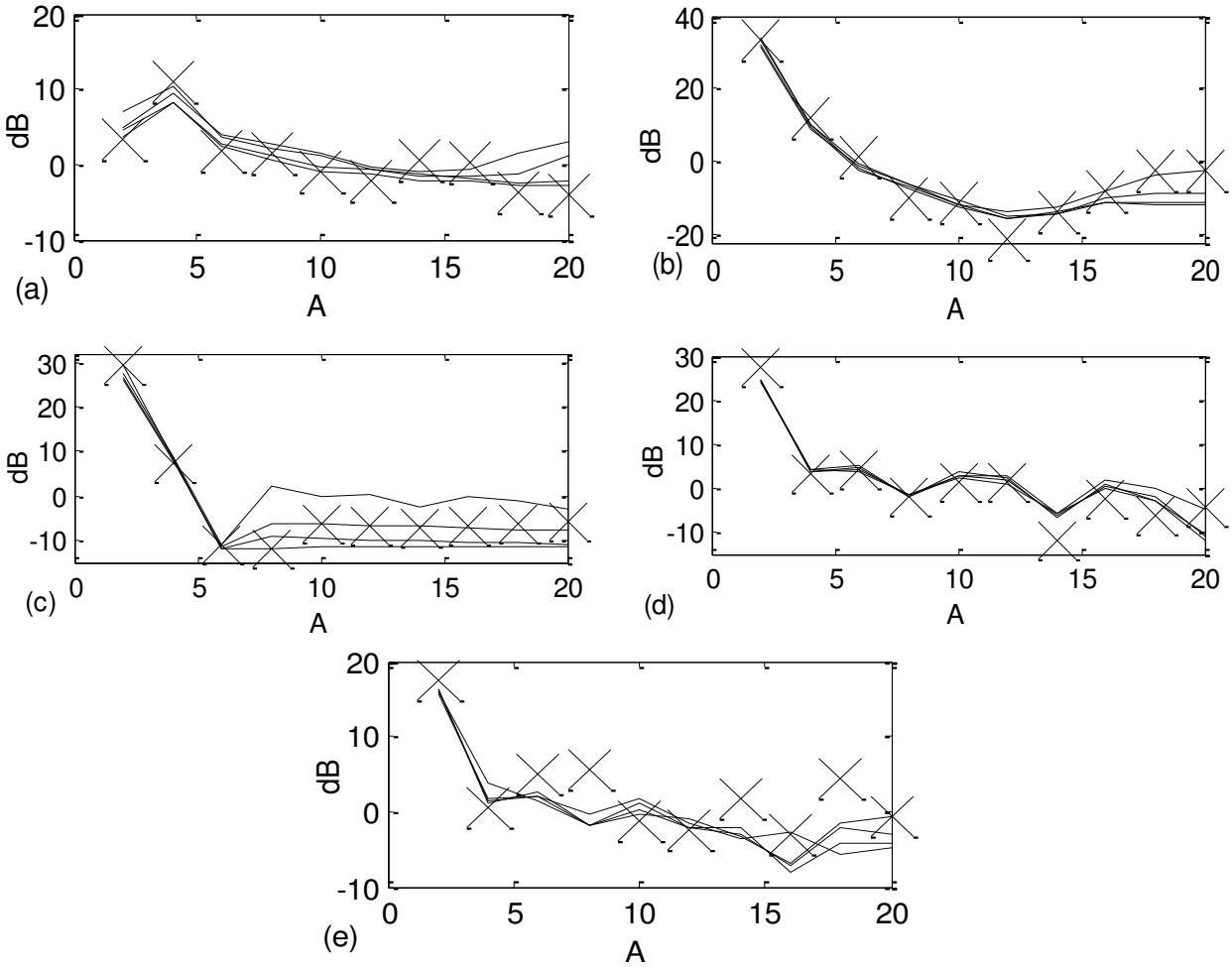


Figure 4.22 – Erreurs entre les SFDRs reportés dans la figure 4.21 et les SFDRs théoriques de la figure 4.10. × Erreur des résultats expérimentaux du FPGA, —— Erreur des résultats de simulation. (a) Fenêtre rectangulaire. (b) Fenêtre de Hann. (c) Fenêtre de Blackman Harris. (d) Parks-McClellan. (e) Moindres carrés

Comme ce fut le cas du sous-échantillonneur, le sur-échantillonneur proposé remplit lui aussi les objectifs fixés, à savoir le contrôle du SFDR, la non utilisation d'une 3^{ième} horloge et la prédiction de la quantité des ressources requises. Il souffre également des mêmes faiblesses. Autrement dit, il ne supporte pas de déviation entre les fréquences moyennes de l'horloge d'entrée et de l'horloge de sortie. De plus, l'utilisateur peut se retrouver dans des situations où la quantité de mémoire à allouer aux ROMs est critique (voire supérieure à la quantité disponible) pour le FPGA utilisé. Cela est dû au fait que le nombre d'étages N dans chaque ROM est le numérateur de la version simplifiée du ratio F_{IN}/F_{OUT} et certains jeux de valeurs de F_{IN} et F_{OUT} peuvent induire une valeur de N très grande pour le FPGA.

Le tableau 4.11 récapitule les performances et les caractéristiques de la méthode proposée (sous-échantillonnage et sur-échantillonnage inclus) ainsi que celles des re-échantillonneurs existants prévus pour une utilisation sur FPGA.

Catégorie	Références	Caractéristiques	Besoin d'une 3ième horloge ?
Méthode proposée	**[HRD13]**	**-- Peut produire des SFDRs élevés* -- Permet de contrôler le SFDR obtenu -- Permet de prévoir la quantité de ressources utilisée avant l'implémentation -- Implémentation simple et pratique -- Ne supporte pas de déviation de la fréquence moyenne de l'horloge d'entrée par rapport à l'horloge de sortie -- Peut entrainer une surconsommation de la mémoire disponible dans le FPGA**	**Non**
A base de structures de Farrow	[F88][EGH93] [Q10]	Peuvent être implémentés sans utiliser de ressource mémoire ([Q10])	Oui
A base de CIC	[SR97][J-DM05] [NA06][A-A-SS03] [J-DM06][J-D07]	Ne nécessitent pas de multiplication	Oui
A base de filtres FIR polyphases	Dick et Harris [DH04]	-- Peut produire des SFDRs élevés* -- Implémentation plus complexe -- Supporte la déviation de la fréquence moyenne de l'horloge de sortie	Oui
	Jorgovanovic *et al* [JPKP08]	-- Peut produire des SFDRs élevés* -- Supporte les déviations de la fréquence moyenne de l'horloge d'entrée par rapport à l'horloge de sortie	Oui (afin d'avoir des SFDRs élevés*)
	Barker [B08]	-- Devrait produire des SFDRs élevés* -- Problème de l'implémentation FPGA non traité	Non
	Barker modifié [XWS10]	-- Devrait produire des SFDRs élevés* -- Problème de l'implémentation FPGA non traité	Oui
Types hybrides	[BR05][R84][E03]	Produit des SFDRs plus élevés que pour les catégories Farrow et CIC	Oui

* Ici, on entend pas « SFDR élévé » des SFDRs supérieurs ou égaux à 80 dB

Tableau 4.11 – Positionnement de notre architecture de re-échantillonnage

4.3 Conclusion

Nous avons présenté une architecture FPGA de re-échantillonnage pratique et adaptée aux implémentations radios logicielles. Elle prend comme point de départ la structure Barker [B08]. Par rapport à d'autres systèmes de conversion de fréquence d'échantillonnage compatible avec les designs FPGA, la nôtre ne nécessite pas de troisième horloge dont la fréquence est un multiple de F_{IN} ou de F_{OUT}, comme c'est le cas de la plupart des méthodes citées dans la section 4.1. Elle convient donc même aux très hautes les fréquences d'échantillonnage. Le besoin en SFDR de l'application visée est directement incorporé dans le processus de conception via une simple procédure de calcul. Il est par ailleurs montré dans les tableaux 4.4, 4.5, 4.8 et 4.9 que la quantité des ressources prévue dans la phase de conception correspond fidèlement à l'utilisation réelle des ressources après implémentation matérielle. Même si elle n'est pas destinée à des applications dans lesquelles les horloges d'entrée et de sortie sont totalement découplées (voir par exemple [DH04] [JPKP08]), la structure proposée offre une alternative simple, pratique, à d'autres techniques de re-échantillonnage, pour les cas d'utilisation traités.

C'est cette méthode qui a servi à implémenter un re-échantillonneur indispensable à la conception du banc de filtre du récepteur multi-canaux FM (section 3.4.2).

4.4 Perspective

Ce travail offre une perspective intéressante, il peut donner suite à la création d'une IP paramétrable (correspondant à une « MegaCore Function » pour les FPGAs de type ALTERA) à disposition des utilisateurs de FPGA. Au même titre que les IP pour FFTs sont largement répandus et intégrés aux logiciels de développement, cette méthode pourrait donner lieu à un IP Core instanciable avec des paramètres variables (F_{IN}, F_{OUT}, A, type de filtre) selon le SFDR souhaité. Cela pourrait se faire après avoir développé un moyen pratique pour réduire la quantité de mémoires utilisées par les ROMs si N est très grand pour le FPGA ciblé.

4.5 Bibliographie

[A-A-SS03] W. A. Abu-Al-Saud, and G. L. Stuber. (2003, May). Modified CIC filter for sample rate conversion in software radio systems. *IEEE Signal Process. Lett. 10(5)*, pp. 152-154.

[A79] Programs for Digital Signal Processing, IEEE Press, New York, 1979, Algorithm

5.1

[B08] D. E. W. Barker. (2008, July). Efficient re-sampling implementations [DSP Tips & Tricks]. *IEEE Signal Processing Mag*. *25(4)*, pp. 114-117.

[BR05] D. Babic, and M. Renfors. (2005, Jan.). Power efficient structure for conversion between arbitrary sampling rates. *IEEE Signal Process. Lett*. *12(1)*, pp. 1-4.

[DH04] C. Dick, and F. Harris, "Options for arbitrary re-samplers in FPGA-based modulators," in *Conf. Rec. 38th IEEE Asilomar Conf. Signals, Systems and Computers*, Pacific Grove, CA, 2004, vol. 1, pp. 777-781.

[E03] G. Evangelista, (2003, Feb.). Design of digital systems for arbitrary sampling rate conversion. *EURASIP Signal Processing*. 83(2), pp. 377–387.

[EGH93] L. Erup, F. Gardner, and R. Harris. (1993, June). Interpolation in digital modems – Part II: Implementation and performance. *IEEE Trans. Commun*. vol. 41, pp. 998–1008.

[F88] C.W. Farrow, "A continuously variable digital delay element," in *IEEE Int. Symp. Circuits and Systems*, Espoo, Finland, 1988, vol. 3, pp. 2641-2645.

[H78] J.F. Harris, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform." Proceedings of the IEEE. Vol. 66 (January 1978). pp. 51-84.

[HRD13] B. Happi Tietche, O. Romain and B. Denby, "A Pratical FPGA-Based Architecture for Arbitrary Ratio Sample Rate Conversion", *Journal of Signal Processing Systems*, *Springer*, Sept. 2013, doi: 10.1007/s11265-013-0840-5 <Free online access: http://link.springer.com/article/10.1007%2Fs11265-013-0840-5>. Accessed 18th Nov. 2013

[J-D07] G. Jovanovic-Dolecek, "Modified CIC filter for rational sample rate conversion," in *IEEE Int. Symp. Communications and Information Technologies, ISCIT '07 Symp.*, Sydney, 2007, pp. 252-255.

[J-DM05] G. Jovanovic-Dolecek, and S. K. Mitra. (2005, July). A new two-stage sharpened comb decimator. *IEEE Trans. Circuits and Syst*ems *I. 52(7)*, pp. 1414-1420.

[J-DM06] G. Jovanovic-Dolecek, and S.K. Mitra, "Stepped triangular CIC filter for rational sample rate conversion," in *IEEE Asia Pacific Conf. Circuits and Systems, APCCAS Conf.*, Singapore, 2006, pp. 916-919.

[JPKP08] M. Jorgovanovic, M. Pajic, G. Kvascev and J. Popovic, "FPGA Design of Arbitrary Down-sampler," in *26th IEEE Int.. Conf. Microelectronics*, *MIEL 2008*, Nis, Serbia, 2008, pp. 391-394.

[K74] J.F. Kaiser, "Nonrecursive Digital Filter Design Using the I0- sinh Window Function," Proc. 1974 IEEE Symp. Circuits and Systems, (April 1974), pp.20-23.

[LVKL96] T.I. Laakso, V. Valimaki, M. Karjalainen and U.K. Laine, (1996, Jan). Splitting the unit delay: tools for fractional delay filter design, *IEEE Signal Processing Magazine*, 13(1), pp.30-60.

[M98] S.K. Mitra, Digital Signal Processing A Computer Based Approach, First Edition, McGraw-Hill, New York, 1998, pp. 462-468.

[N81] A. H. Nuttall. (1981, February). Some Windows with Very Good Sidelobe Behavior. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 29(1), pp. 84-91.

[NA06] S. B. Nerurkar, and K. H. Abed. (2006, April). Low-power decimator design using approximated linear-phase N-band IIR filter. *IEEE Trans. Signal Process*. *54(4)*, pp. 1550-1553.

[OS89] A.V. Oppenheim and R.W. Schafer, Discrete-Time Signal Processing, Prentice-Hall, 1989, pp. 447-448.

[OS99] A.V. Oppenheim and R.W. Schafer. Discrete-Time Signal Processing. Upper Saddle River, NJ: Prentice-Hall, 1999, pp. 468-471.

[PB87] T.W. Parks and C.S. Burrus, Digital Filter Design, John Wiley & Sons, 1987, pp. 54-83.

[Q10] D. X. Quang, "Digital resampling and timing recovery in QAM systems," M. S. thesis, Dept. Elect. Eng., Saskatchewan Univ., Saskatchewan, Canada 2010.

[R84] T. Ramstad. (1984, June). Digital methods for conversion between arbitrary sampling frequencies. *IEEE Trans. Acoust., Speech, Signal Process*. *32(3)*, pp. 577–591.

[SLB95] I.W. Selesnick, M. Lang, and C.S. Burrus, "Constrained Least Square Design of FIR Filters without Specified Transition Bands," Proceedings of the IEEE Int. Conf. Acoust., Speech, Signal Processing, Vol. 2 (May 1995), pp.1260-1263.

[SR97] T. Saramaki, and T. Ritoniemi, "A modified comb filter structure for decimation," in *Proc. IEEE Int. Symp. Circuits and Systems, ISCAS'97 Symp.*, Hong Kong, 1997, vol. 4, pp. 2353-2356.

[T67] J.W. Tukey, (1967). An introduction to the calculations of numerical spectrum analysis. Spectral Analysis of Time Series. Wiley ed. pp.25–46.

[XWS10] Yong-jian Xu, Hong-yuan Wang, and Zhi Shen, "Modified polyphase filter for arbitrary sampling rate conversion," in *6th IEEE Int. Conf. Wireless Communications Networking and Mobile Computing*, *WiCOM Conf.*, Chengdu, China, 2010, pp. 1-4.

5 Prototypage et résultats expérimentaux

Nous allons, dans cette partie, voir en détails les prototypes réalisés. Il convient sans doute de rappeler l'architecture générique qui reflète notre méthodologie de réception (figure 3.1).

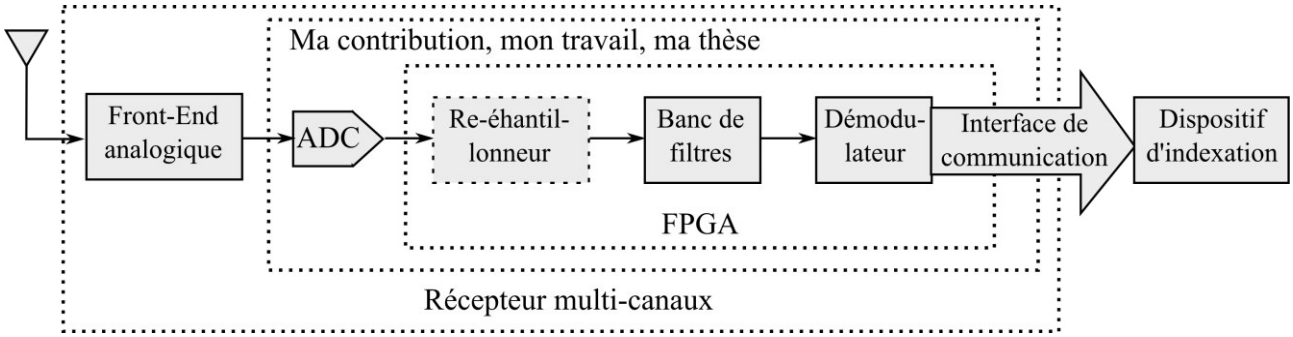


Figure 3.1 (rappel) – Architecture d'extraction et de démodulation de toutes les stations

Nous avons vu dans le chapitre 4 que cette architecture est le modèle utilisé pour la réception FM et pour la réception DRM. Dans le cas de la réception DRM, le re-échantillonneur n'est pas utilisé. Des prototypes de chaque bloc de l'architecture ont été réalisés, pour la FM et pour la DRM, aboutissant ainsi à des prototypes finalisés et opérationnels.

5.1 Prototypage pour la bande FM

5.1.1 Front-end analogique

Une photographie du prototype du front-end analogique développé par le CEA (partenaire du projet SurfOnHertz) pour la FM (voir section 3.2.1.1 pour plus de détails) est donnée dans la figure 5.1:

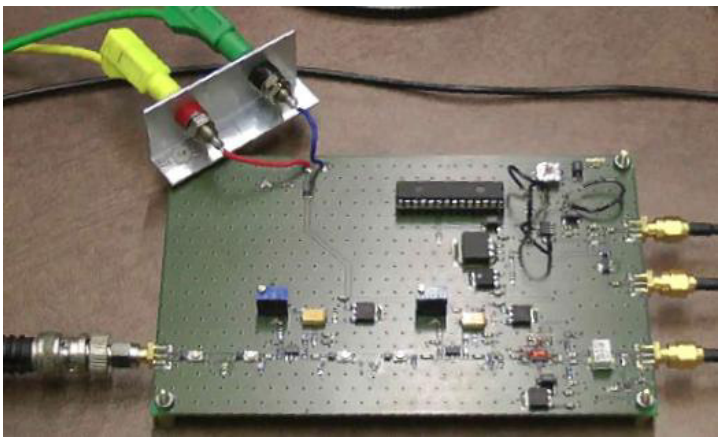


Figure 5.1 – Photographie du circuit réalisé du front-end analogique

Le résultat d'une simulation au cours de laquelle 20 porteuses ont été injectées à l'entrée du circuit est montré dans la figure 5.2. On remarque que le rapport porteuse sur bruit (ou C/N pour Carrier-to-Noise ratio) est de l'ordre de 70 dB. C'est un excellent résultat pour une modulation analogique.

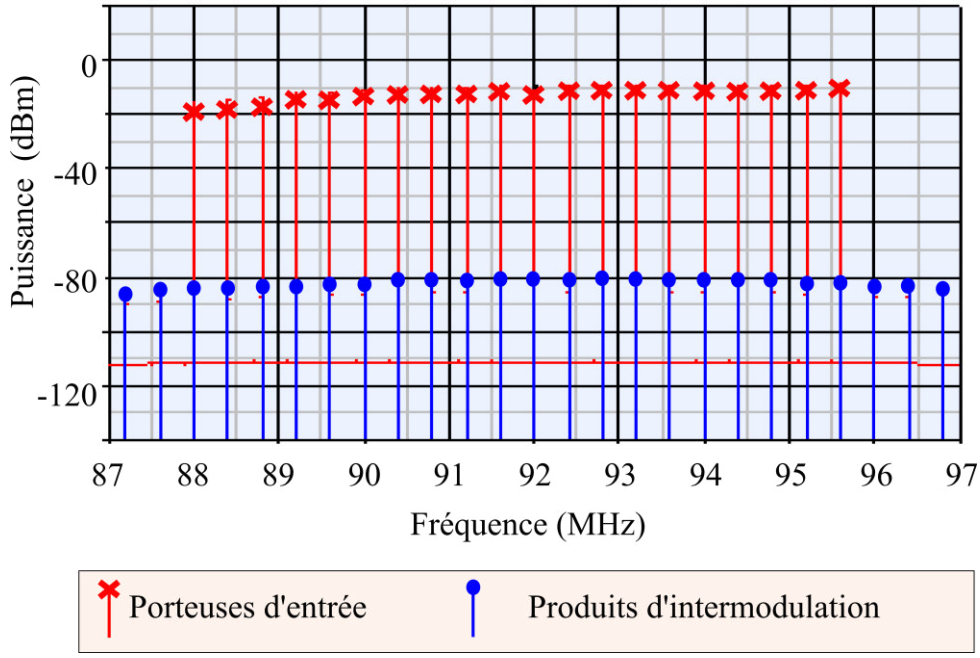


Figure 5.2– Simulation de l'intermodulation avec 20 porteuses en entrée

5.1.2 Re-échantillonneur

Le prototype du re-échantillonneur utilisé est discuté dans la section 4.2.3.1. On rappelle que l'implémentation de ce re-échantillonneur, comme tous les prototypes présentés ici, a été réalisée sur un FPGA de type Altera Stratix II EP2S180F1020C5 utilisant une carte de développement Stratix II DSP. Les ressources disponibles dans ce composant sont données par le tableau 5.1.

ALUTs	143.520
Registres logiques dédiés	143.520
Mémoire (en bits)	9.383.040
Eléments DSP	768
PPLs	12

Tableau 5.1– Ressources disponibles dans le FPGA Stratix II EP2S180F1020C5

5.1.3 Dynamic autoscale

Le flux de sortie généré par le re-échantillonneur est un flux d'échantillons de 28 bits résultant de l'augmentation de la largeur du flot de données due aux calculs opérés par le processus de re-échantillonnage, plus précisément la sommation des sorties de 13 multiplieurs 12 × 12 bits. Parce qu'une précision de 12 bits est suffisante pour notre application (voir la section 3.2.2), un module appelé « dynamic autoscale » est inséré à la sortie du re-échantillonneur afin de limiter la largeur de la sortie. Il est, de toutes façons, inévitable de réduire la largeur des échantillons avant la case WOLA car le noyau FFT utilisé plus loin par notre composant ne prend en entrée que des mots de largeur maximale 24 bits.

Le « dynamic autoscale » a été implémentée pour sélectionner et sortir les 12 bits les plus importants d'un flux d'échantillons. Le système fonctionne grâce à un curseur qui, sur les 28 bits (numérotés de 27 à 0) de l'échantillon, donne le numéro de celui à partir duquel le mot de 12 bits sera extrait. Par exemple, si le curseur est égal à 25, « dynamic autoscale » sortira un mot fait des bits 25 à 15 du mot d'entrée. Si la valeur du curseur est inférieure à 11, la sortie est complétée avec des zéros (figure 5.3). Le curseur est mis à jour une fois par seconde, ce qui est suffisant pour garantir un calcul correct du curseur pour une horloge de 51.2 MHz (51.2×10^6 cycles d'horloge).

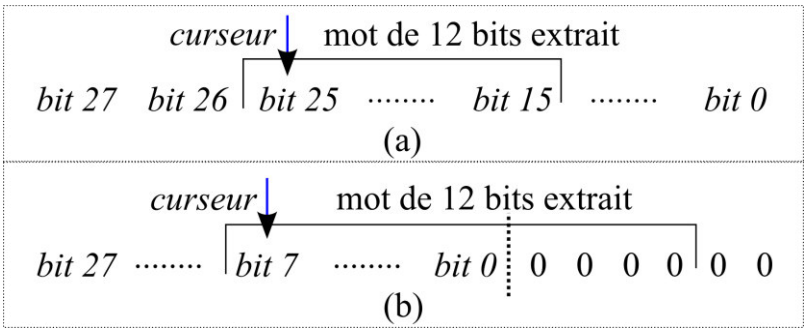


Figure 5.3 – Les bits 27 à 0 représentent l'échantillon de 28 bits. a) *Curseur* = 25, *curseur* est supérieur à 11. b) *Curseur* = 7, *curseur* est inférieur à 11. L'échantillon de sortie de 12 bits est complété par des zéros

Le curseur est initialisé à 27, qui est le numéro du bit de poids fort (ce bit est encore appelé « *MSB* », pour *most significant bit*) d'un échantillon de 28 bits. Les échantillons sont des nombres codés par complément à 2. Ils sont donc positifs lorsque $MSB = 0$, sinon ils sont négatifs. Une fois par seconde, « dynamic autoscale » exécute l'algorithme de la table 5.2 pour mettre à jour le curseur, où bit_E est le bit de numéro E dans un mot de 28 bits issu du re-échantillonneur. Ce procédé garantit qu'on choisit les 12 bits les plus importants du flux d'échantillons de 28 bits issus du re-échantillonneur à destination des blocs suivants.

Si, pendant 1 seconde, $bit_{curseur}$ a toujours été égal à $bit_{curseur-1}$ lorsque $MSB = 1$ et $bit_{curseur}$ a toujours été égal à $bit_{curseur-1}$ lorsque $MSB = 0$, alors
    Si $curseur > 0$ alors
        $Curseur = curseur - 1$
    Sinon
        $Curseur = curseur$
    Fin si
Sinon si, pendant 1 seconde, $bit_{curseur}$ a toujours été égal à $bit_{curseur+1}$ lorsque $MSB = 1$ et $bit_{curseur}$ a toujours été égal à $bit_{curseur+1}$ lorsque $MSB = 0$, alors
    Si $curseur < 27$ alors
        $Curseur = curseur + 1$
    Sinon
        $Curseur = curseur$
    Fin si
Sinon
    $Curseur = curseur + 1$
Fin si

Table 5.2 – Algorithme de mise à jour du curseur du module « dynamic autoscale »

5.1.4 Banc de filtres pour la FM

On a vu dans la section 3.4.2 que le channelizeur FM est cadencé à 51.2 MHz et est composé de 4 branches, chacune d'elle ayant un banc de filtres WOLA. Trois branches parmi les quatre utilisent un NCO pour transposer le signal à la gamme de fréquences appropriées (figure 3.26).

5.1.4.1 NCO

Les NCOs utilisées sont des IPs ALTERA disponibles dans la librairie des composants. Afin de mettre en œuvre la relation (3.3), étant donné que $e^{jr}=\cos(r) + j\sin(r)$, les NCOs ont deux sorties correspondant au sinus et au cosinus. En fonction de la branche, la fréquence de sortie est fixée soit à 100, soit à 200, ou encore à 300 kHz. La largeur des sorties sinus et cosinus sont fixées à 13 bits. Après multiplication du signal d'entrée par le sinus et le cosinus, les signaux résultants sont I et Q (voir figure 5.4) définis sur 25 bits. On utilise la routine « dynamic autoscale I,Q » pour sélectionner les 12 bits les plus importants à envoyer au banc de filtres WOLA (figure 5.4). « dynamic autoscale I,Q » opère la même mise à l'échelle sur I et Q vu que I et Q sont complémentaires. Cela signifie que si par exemple « dynamic autoscale I,Q » sort seulement les bits 19 à 7 sur les échantillons I, il sortira aussi les même bits sur les échantillons Q. Soient bit_{C1} et bit_{C2} les bits de poids fort des 12 plus importants bits de I et Q. Le bit le plus élevé entre bit_{C1} et bit_{C2} détermine en conséquence quels bits seront prélevés sur les échantillons I et Q.

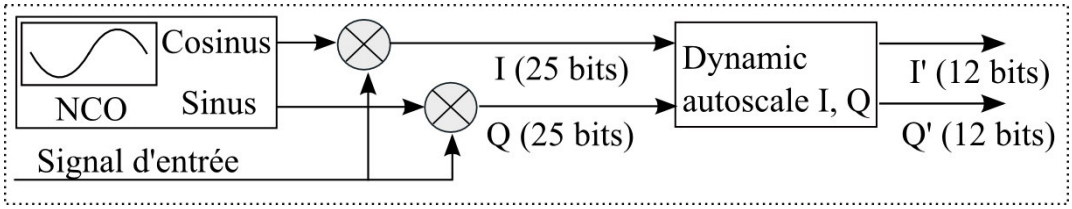


Figure 5.4 – Système de transposition du signal d'entrée

5.1.4.2 Banc de filtres WOLA

Le principe de WOLA a été résumé sur la figure 3.10. Comme expliqué dans la section 3.4.2, l'étape « ajustement référence temporelle» est sautée dans notre implémentation puisque cela revient à multiplier par 1 avec notre choix de jeu de paramètres. Ainsi, seules les parties 1 et 2 de l'algorithme WOLA (figure 3.10) ont besoin d'être implémentées. Les paramètres de notre implémentation sont les suivants : L_P = 4096 and $K_P = D_P$ = 128. Le banc de filtres est cadencé à 51.2 MHz (cf. section 3.4.2), et la réponse impulsionnelle du filtre composé de mots de 12 bits. En général, les logiciels de développement sur FPGA offrent les principaux IPs FFT standards. Chaque cœur de FFT dispose de deux entrées et deux sorties I et Q. Pour l'IP FFT utilisé dans notre application, les échantillons de données d'entrée sont fournis en mode série, c'est-à-dire qu'ils alimentent le noyau FFT un par un.

Lors de l'implémentation de la branche 1, l'entrée Q du noyau FFT est simplement mise à 0 puisqu'aucune transposition de fréquence n'a eu lieu. Cependant, les branches 2, 3 et 4 feront usage

de composants Q non nuls. L'implémentation matérielle du channelizeur FM complet est résumée dans la figure 5.5.

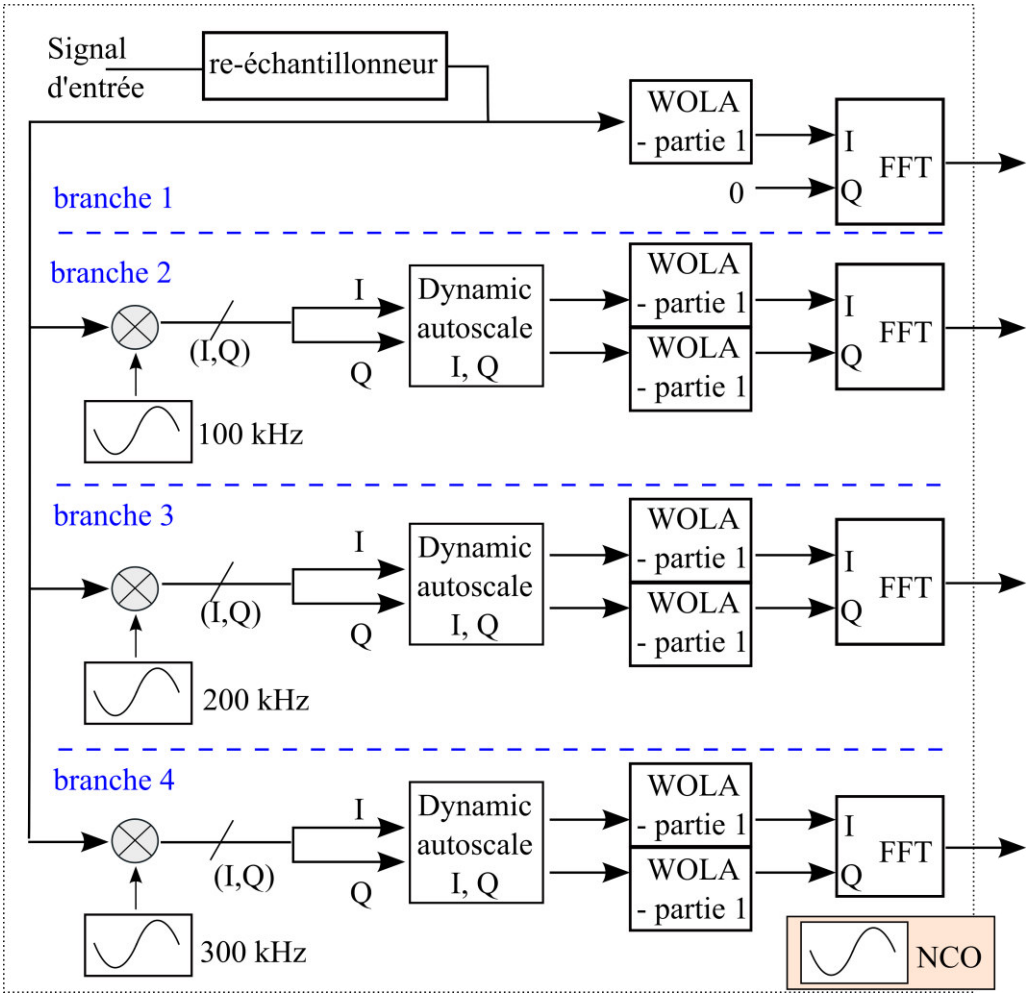


Figure 5.5 – Synopsis l'implémentation matérielle du channelizeur FM

Comme la réponse impulsionnelle du filtre et les échantillons du signal d'entrée sont des mots de 12 bits, les échantillons qui sont destinés à alimenter les entrées I et Q du noyau FFT sont des mots de 29 bits. Cela est dû aux calculs qui sont faits par le processus de Weighting Ovelap Add, c'est-à-dire, la somme des résultats issus de 32 multiplieurs 12 × 12 bits. Pour le noyau FFT utilisé, cependant, la largeur maximale de l'entrée est seulement de 24 bits. Pour cette raison, les échantillons qui alimentent le noyau FFT sont réduits à 24 bits. L'utilisation des ressources du FPGA par le channelizeur est résumée dans le tableau 5.3, après compilation sous Altera Quartus II 64-bit version 10.1:

	Re-échantillonneur 87 – 51.2 MHz	Branche 1	Branche 2	Branche 3	Branche 4 (NCOs inclus)	**Channelizeur FM entier**
ALUTs	298 (< 1 %)	6963 (5 %)	10525 (7 %)	5414 (4 %)	5416 (4 %)	**28616 (20 %)**
RLD	531 (< 1 %)	8022 (6 %)	10645 (7%)	9707 (7 %)	9709 (7 %)	**38614 (27 %)**
Mémoire (en bits)	54707 (<1%)	129592 (1 %)	238672 (3 %)	238672 (3 %)	238672 (3 %)	**900315 (10 %)**
Eléments DSP	26 (3 %)	112 (15 %)	158 (21 %)	236 (31 %)	236 (31 %)	**768 (100 %)**
PLLs	--	--	--	--	--	**1 (8%)**

Tableau 5.3 – Tableau des ressources consommées par le channelizeur. Les chiffres dans les parenthèses donnent les pourcentages par rapport aux ressources disponibles du FPGA. Les ressources disponibles du FPGA sont données dans le tableau 5.1. La colonne « channelizeur FM entier » résulte de la sommation « à la main » des ressources des autres colonnes du tableau, ces dernières étant directement recopiées du rapport de compilation de Quartus 10.1. « RLD » représente les registres logiques dédiés.

5.1.5 Démodulateur de fréquence

La carte FPGA utilisée dans notre implémentation possède un Convertisseur Numérique Analogique (CNA) – ou DAC pour Digital-to-Analog Converter – de 14 bits, ce qui est utile pour faire des tests de vérification comme contrôler les signaux des canaux démodulés un seul à la fois; par conséquent, on a choisi de fixer à 14 bits la précision des échantillons sortant du système de démodulation. Le nombre d'itération du CORDIC est alors de 14, vu qu'il n'est pas nécessaire d'implémenter plus d'itérations que la précision requise.

On a vu dans la section 3.4.2 que F_i = 400 kHz. En conséquence, pour un canal spécifique, chaque bloc FFT donne un nouvel échantillon tous les 128 échantillons produits, car le noyau FFT est cadencé à 51.2 MHz. Cela signifie que la longueur de la ligne à retard numérique de la figure 3.39 est égale à 128. L'utilisation des ressources du système est donnée dans le tableau 5.4:

	Bloc de démodulation de la branche 1	**Système de démodulation complet (4 blocs pour les 4 branches)**
ALUTs	1164 (< 1 %)	**4669 (3 %)**
Registres logiques dédiés	1034 (< 1 %)	**4147 (3 %)**
Mémoire (en bits)	1814 (<1%)	**7541 (< 1 %)**
Eléments DSP	0 (0 %)	**0 (0 %)**

Les chiffres entre parenthèses donnent les pourcentages d'utilisation des ressources disponibles de la puce. Les blocs de démodulation des branches 2, 3 et 4 utilisent une quantité de ressources proche de celui de la branche 1. La colonne « Système de démodulation complet » résulte de la sommation « à la main » des ressources utilisées par le démodulateur de chaque branche selon le rapport de compilation du logiciel Quartus 10.1.

Tableau 5.4– Ressources du démodulateur de fréquences pipeline et limitation de l'horloge.

5.1.6 Bloc de transfert des stations démodulées

L'implémentation matérielle du système de sortie (figure 3.45) dépend de deux choix fondamentaux.

- Le choix d'un protocole de transfert adéquat.
- Le paramétrage de la largeur et de la profondeur des FIFOs buffers des canaux et de la FIFO d'interface.

Nous allons maintenant examiner ces deux choix en détails.

5.1.6.1 Le choix d'un protocole de transfert adéquat

Deux critères motivent le choix du protocole de transfert. Le premier provient du fait que les données issues des stations démodulées doivent être exploitables par une variété de différentes structures matérielles de traitement de signal tels que les FPGAs, les GPUs, CPUs, etc. Les liaisons séries sont idéales pour cet usage. Le second critère est le débit de transfert requis qui dépend naturellement du nombre de stations radio à transmettre. Considérons la métropole française la plus large, la région parisienne, qui a environ 51 stations FM répertoriées. Chaque station démodulée qui au départ était un signal de mots de 14 bits échantillonné à 400 kHz (section 5.1.5) est décimé par 2 car la déviation maximale de la FM (en Europe occidentale) est 100 kHz. Si maintenant on considère qu'on capte en région parisienne toutes les stations DRM30 diffusée en Europe (bien que ce ne soit pas le cas), une douzaine, et qu'on ne se contente que de ne démoduler que l'audio

cadencé à moins de 48 kHz, la liaison de transmission doit être capable de supporter au moins 51 x 14 x 200000 + 12 x 14 x 48000 = 150.9 Mb/s. L'USB 2.0, avec une bande passante 480 Mb/s, répond à ces deux critères.

5.1.6.2 Le paramétrage des largeurs et des profondeurs des FIFOs

Les buffers des canaux sont de 14 bits et profonds de 32 mots. Cette profondeur est suffisante pour permettre au bloc « contrôleur » de la figure 3.45 d'éviter la lecture des buffers lorsqu'ils sont vides et leur remplissage complet.

Les caractéristiques de la FIFO d'interface dépendent du dispositif de connexion de l'USB 2.0 utilisé. On a choisi un adaptateur QuickUSB Santa Cruz, qui est complètement compatible avec notre carte de développement Altera Stratix II DSP. Comme l'adaptateur QuickUSB Santa Cruz est prévu pour une FIFO d'interface large de 16 bits, nous avons utilisé une FIFO de 16 bits et de 1024 mots dans notre implémentation. Chaque mot de 16 bits est constitué comme suit :

- Les 6 MSBs sont une étiquette assignée par le « contrôleur » qui donne le numéro de la station démodulée. La première station porte le numéro 0.
- Les 10 bits de poids faibles, LSBs pour « least significant bits » contiennent les 10 MSBs de l'échantillon de la station correspondante.

Les ressources utilisées par le système, y compris la FIFO d'interface, sont montrées dans le tableau 5.5.

ALUTs	9606 (7%)
Registres logiques dédiés	7619 (5%)
Mémoire (en bits)	98560 (1%)
Eléments DSP	0 (0%)

Les chiffres entre parenthèses donnent les pourcentages d'utilisation des ressources disponibles de la puce. Les chiffres donnés sont directement issus du rapport de compilation.

Tableau 5.5 – Utilisation des ressources du système de transmission de stations démodulées et limitation de l'horloge

5.1.7 Prototype du récepteur FM complet

Le récepteur complet est obtenu en assemblant tous les blocs décrits. Un synopsis fonctionnel de l'architecture est donné dans la figure 5.6.

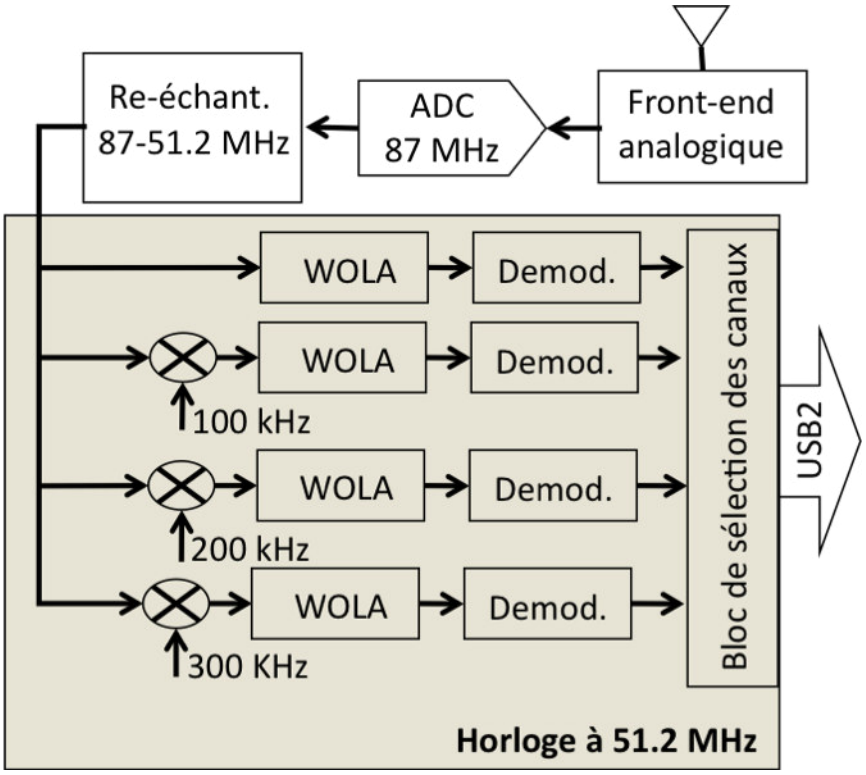


Figure 5.6 – Synopsis fonctionnel de l'architecture du récepteur global.

L'utilisation des ressources du FPGA (ALTERA Stratix II EP2S180) du prototype est résumée dans le tableau 5.6. La contribution de chaque bloc du système est individuellement donnée dans le tableau 5.7.

ALUTs	43656 (30 %)
Registres logiques dédiés	51533 (36 %)
Mémoire (en bits)	1207120 (13 %)
Eléments DSP	768 (100 %)
PLLs	1 (8 %)
Max 87	108.66
Max 51.2	76

Les chiffres entre parenthèses donnent les pourcentages d'utilisation des ressources disponibles de la puce. Max 87 et Max 51.2 sont les fréquences maximales auxquelles peuvent monter les horloges 87 MHz et 51.2 MHz sans entraver la bonne exécution des opérations. Les chiffres donnés sont directement issus du rapport de compilation.

Tableau 5.6 – Tableau d'utilisation des ressources du prototype

	Contribution de chaque bloc décrit plus haut en fonction des ressources utilisées par le prototype entier, à l'exception de "dynamic adapter"				Contribution des parties restantes (%)
	Channelizeur		Démodulateur de fréquences (%)	Système de transmission des stations (%)	
	Re-échantillonneur (%)	4 branches (%)			
ALUTs	< 1	65	11	22	2
RLD	1	74	8	15	2
Mémoire (en bits)	5	70	< 1	8	17
Eléments DSP	3	97	0	0	0
PLLs	100	0	0	0	0

Les pourcentages des contributions de chaque bloc sont les proportions calculées de leurs ressources par rapport aux ressources utilisées par le prototype entier. Les parties restantes désignent toute la logique, la mémoire et les blocs « dynamic adapter » utilisés pour inter-connecter les blocs décrits plus haut. « RLD » représente les registres logiques dédiés.

Tableau 5.7 – Contribution de chaque bloc

5.1.8 Résultats expérimentaux du récepteur FM

Le prototype du récepteur FM décrit ci-dessus (section 5.1) est opérationnel. Avant cette version du prototype, il y a des versions bêtas, non abouties, qui ont fait l'objet d'évaluations en France et de démonstrations en France [NuméANR13], en Europe [DASIP12] et en Afrique [DataMining12], ce qui n'est pas encore le cas de la version actuelle. Les résultats expérimentaux obtenus avec la version intermédiaire ont été publiés dans le journal IEEE Transactions on Consumer Electronics [TRDD12] et en conférence IEEE [HRDBV12]. Nous allons les présenter.

5.1.8.1 Signaux démodulés

Le spectre de la sortie audio d'une station française type, après la chaine complète de traitement par notre prototype intermédiaire, est présentée dans la figure 5.7-a. La figure 5.7-b donne un exemple de l'utilisation du spectre FM en France, pour aider à la compréhension du spectre obtenu expérimentalement au-dessus. Les bandes mono et stéréo, ainsi que le pilote de synchronisation à 19 kHz et la bande RDS, sont clairement identifiables dans le spectre expérimental issu de notre

prototype. Le dispositif fournit simultanément un signal de ce type pour toutes les chaines de radio FM présentes dans la zone de réception. Encore faut-il que les chaines en question possèdent toutes ces composantes.

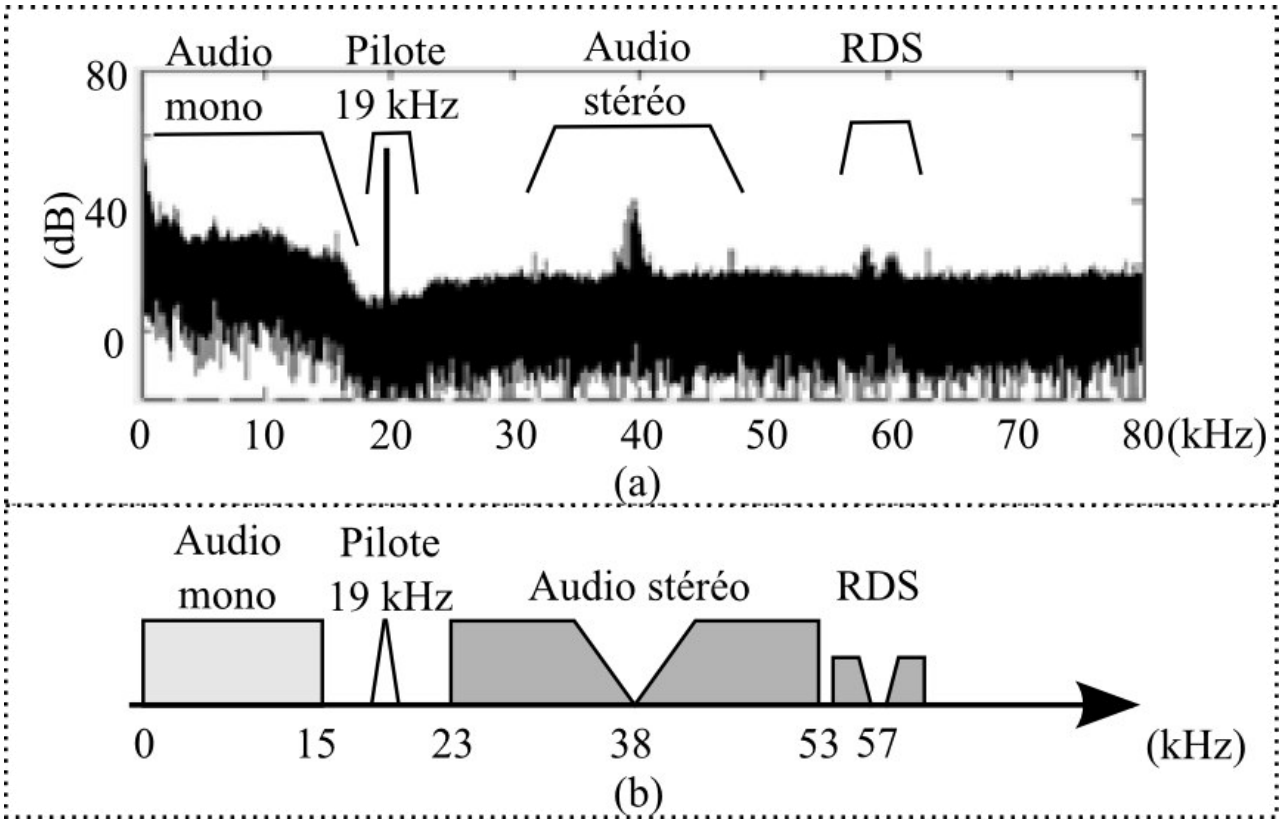


Figure. 5.7 Spectre d'une station radio FM avec transmission RDS. (a) Spectre d'une station radio française démodulée par notre prototype intermédiaire. (b) Répartition des informations dans le spectre d'une station radio FM standard.

Une mesure du RSB donne un meilleur aperçu de la qualité de l'audio produite par notre prototype intermédiaire. Le procédé expérimental de mesure était le suivant : Un canal démodulé arbitraire a été choisi, et le FPGA programmé pour sortir ce canal sur un connecteur stéréo jack par un codeur/décodeur audio, CODEC en anglais pour Audio coder/decoder, ce dernier étant disponible sur notre carte. Lorsqu'aucun signal FM n'était mis en entrée, le niveau du signal démodulé sortant a été mesuré sur le connecteur jack. Cela est devenu le niveau de bruit. Par la suite, pour différents gains du front-end analogique et différents niveaux d'une porteuse de 1 kHz modulée en fréquence avec une déviation de ± 75 kHz (comme pour une radio FM d'Europe occidentale standard), on a mesuré le niveau du signal de sortie démodulé. Le RSB a alors été calculé à partir de ces mesures.

Les résultats apparaissent dans la figure 5.8, qui montre les RSB pour le bruit audio en fonction du niveau du signal RF modulé en fréquence. Les courbes indiquent que notre prototype de récepteur est capable d'atteindre un RSB de 50 dB, ce qui est comparable à ce qui est obtenu dans les récepteurs analogiques traditionnels. Cependant, là où notre récepteur manifeste une faiblesse

par rapport à un récepteur analogique, c'est au niveau de la sensibilité RF globale. En effet, les récepteurs traditionnels (comme dans [Philips92] par exemple) peuvent maintenir la sensibilité jusqu'à -90 dBm, tandis que notre prototype – du moins sa version intermédiaire – rcoupe à -80 dBm (front-end y compris). Cette déficience est, d'une certaine manière, déterminée par la précision de l'ADC (12 bits), et peut en principe être améliorée par: a) utiliser deux ADCs, un pour les stations de haute puissance et un autre, accompagné par un circuit analogique dédié, pour les plus faibles comme proposé dans [IH09]; b) Simplement utiliser un ADC de plus d'une précision supérieur à 12 bits.

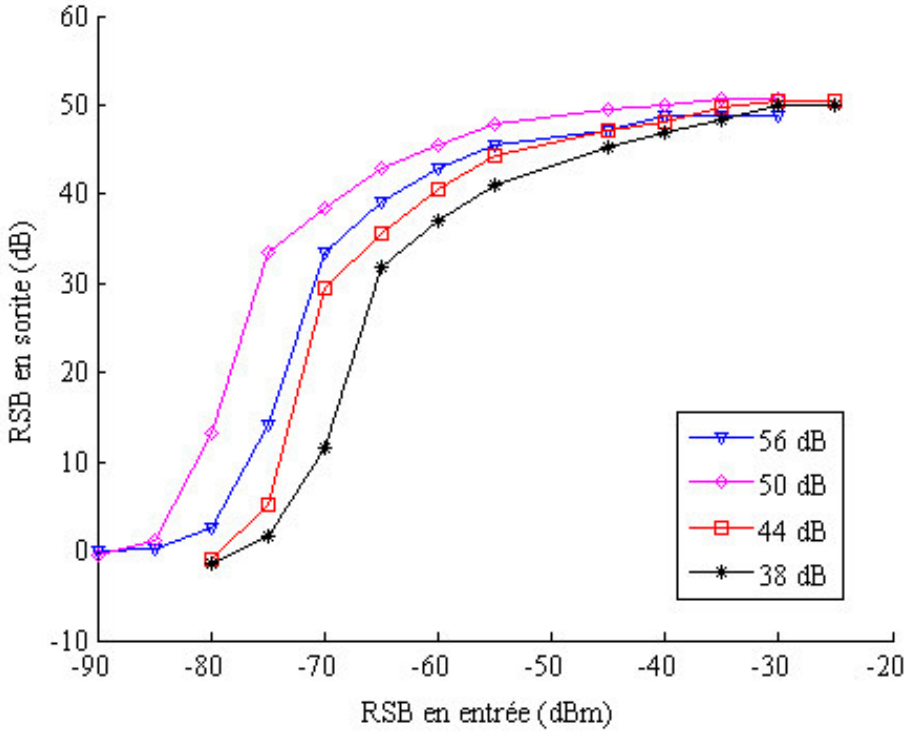


Figure 5.8 RSB donné par le prototype intermédiaire en fonction du niveau du signal d'entrée. Les valeurs dans le rectangle sont les différents gains d'amplification du front-end analogique.

Les stations sorties par notre prototype sont destinées à des applications d'indexation audios comme par exemple la classification parole/musique, l'identification du genre musical, le traitement de la parole et d'autres semblables. Il faut maintenant prouver que les performances du récepteur FM permettent d'atteindre cet objectif. C'est le but de la section suivante (section 5.1.8.2).

5.1.8.2 Indexation par le contenu audio

L'indexation audio se base sur des technologies qui ont fait l'objet de recherches depuis de nombreuses années [Lamel2008]. Des travaux sur la détection de parole ont tout d'abord été développés pour les communications téléphoniques. Il s'agit d'une tâche complexe à réaliser si le

niveau de bruit est important ou si, par exemple, la parole est superposée à un fond musical. En ce qui concerne la transcription automatique de la parole, les chercheurs s'efforcent d'améliorer les performances des algorithmes depuis les années 1960. Dans les années 90, les résultats obtenus en dictée automatique ont incité des sociétés telles qu'IBM à commercialiser des produits (ViaVoice par ex.). Les agences gouvernementales américaines ont menées une politique volontariste de structuration de la recherche en reconnaissance de la parole en organisant des campagnes d'évaluation sur des données publiques. Elles subventionnent le LDC (Linguistic Data Consortium) pour la collection et la distribution des données et le NIST (National Institute of Standards and Technology) pour l'organisation et le dépouillement des campagne d'évaluation. Il a été question, entre autres, de transcrire les bulletins d'information radiodiffusés en anglais. En moins de 10 ans, plus précisément de 1998 à 2004, le taux d'erreurs en mots est passé de 30% à 10%. En France, la première campagne d'évaluation des systèmes de traitement automatique pour la langue française a été menée dans le cadre de l'initiative AUPELF. On a aussi vu naitre la campagne ESTER [TECHNOLnet], puis plutard les évaluations ETAPE 2011 où il était question de détection et classification des entités nommées (noms propres, quantités, etc) et d'identification et de suivi de locuteurs (présentateurs radios, personnages politiques et/ou médiatiques, etc. La reconnaissance du locuteur, l'identification des langues et la traduction vocale ont aussi été abordées par le NIST depuis plus d'une dizaine d'années.

Dans le domaine de l'indexation musicale, différents travaux ont porté sur la classification du genre, la reconnaissance de pièces, la détection du rythme, la reconnaissance et la séparation des instruments, la transcription automatique de partitions [Essid] [Herrera2006] [Peeters2004].

Quant aux techniques d'indexation qui apprennent automatiquement à associer données et étiquettes, peu de recherches ont été effectuées dessus. L'une des approches phares de ce domaine est l'approche ALISP (Automatic Language Independent Speech Processing) qui a d'abord été développée pour le codage de la parole à très bas débit, et qui a été exploitée avec succès pour d'autres tâches, telles que la reconnaissance du locuteur ou de la langue [ElHannani2005] [ElHannani2006] [ElHannani2007]. Elle permet donc de réduire au maximum la tâche d'annotation par des opérateurs humains. D'autres travaux montrent que l'approche est également applicable à des signaux musicaux [Schwarz2004].

Chez Yacast, le partenaire industriel du projet SurfOnHertz, la captation des programmes est actuellement réalisée via des tuners FM, DVB-S ou DVB-T. Les flux analogiques audio et vidéo sont ensuite numérisés pour être ingérés dans les moteurs d'identification intégrant les algorithmes de « Finger Printing ». Il est nécessaire de déployer un tuner et un serveur d'identification pour chaque media radio ou télé. Chaque flux est analysé en temps réel 24h sur 24, 7 jours sur 7. Les contenus publicitaires et musicaux reconnus sont intégrés dans une base de données puis validés

manuellement par des opérateurs qui vérifient l'intégralité des programmes.

L'application d'indexation développée dans le cadre du projet SurfOnHertz est la classification de genre. Plus précisément, il s'agit d'identifier les flux audios démodulés en 5 classes :

- la classe « musique jazz ».
- la classe « variété ».
- la classe « musique classique ».
- la classe « bruit » : C'est la classe qui indique lorsque le son audio démodulé est très bruité.
- la classe « non musique », autrement dit la parole.

Pour avoir une idée intuitive de la viabilité d'une telle application, nous donnons les signaux de la composante mono de deux stations radios dans la figure 5.9, enregistrées à 200 kS/s à partir du récepteur « intermédiaire », dont les programmes étaient assez différents durant la période examinée (cinq secondes ; la stéréo, le RDS, et le pilote à 19 kHz ont été enlevées du signal pour plus de clarté). Une station diffusait alors de la musique tandis que l'autre diffusait la parole. La figure montre clairement que les signaux parole et musique sont différents. Par exemple, l'enveloppe du signal de la musique maintient des valeurs plus élevées sur des périodes de temps plus longues par rapport au signal de la parole. La fonction d'identification de la classe « Non musique (parole) » de l'application a été développée de façon à exploiter cette différence. D'autres différences peuvent être exploitées avec succès en utilisant les caractéristiques spectrales telles que coefficients MFCCs (Mel-Frequency Cepstral Coefficients), avec des systèmes de classification [voir, par exemple, Peeters07]. Les fonctions de classification des 3 genres musicaux (jazz, classique et variété) et du bruit sont basées dessus.

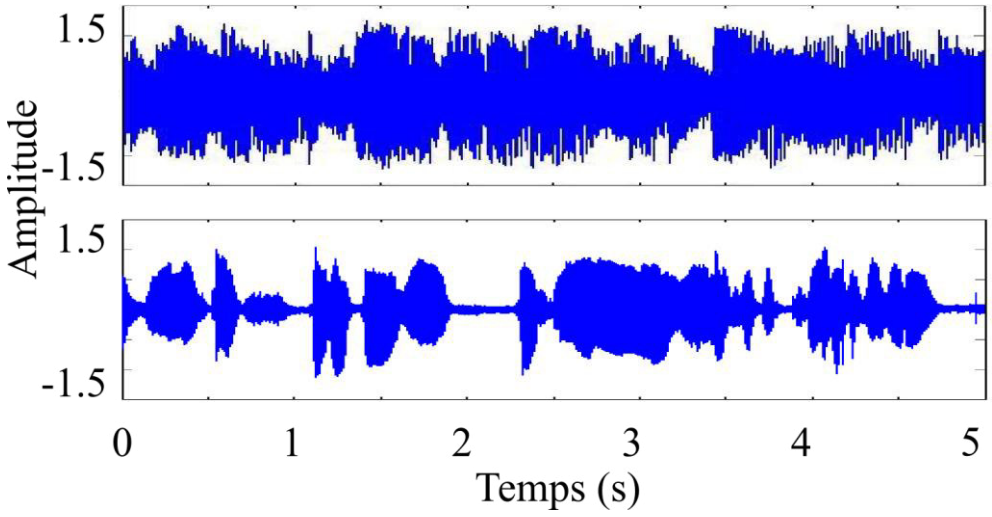


Figure 5.9 – Signaux de deux stations radios enregistrés à 200 kHz sur une durée de 5 secondes. L'enregistrement au dessus a été fait lorsque du jazz était diffusé, tandis que l'enregistrement du dessous montre le signal d'une diffusion de la parole.

L'assemblage de toutes composantes du système global (antenne, front end analogique, FPGA Stratix II, machine d'indexation) constitue la version actuelle du navigateur hertzien développé pour la FM. Une image en est donnée dans la figure 5.10. On peut y voir toutes les composantes constituant le navigateur. La machine d'indexation est un PC sur lequel tourne l'application d'indexation.

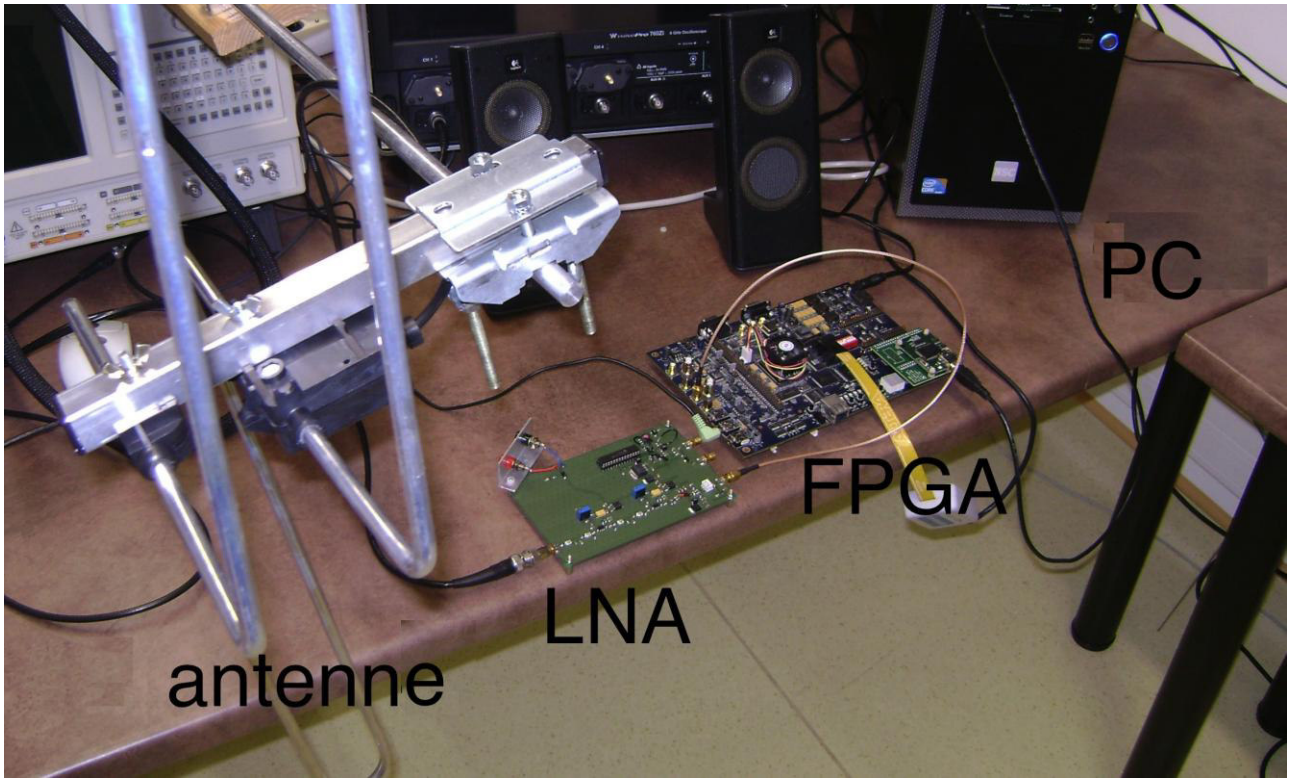


Figure 5.10 – Navigateur Hertzien SurfOnHertz pour la bande FM

C'est sur l'écran du PC qu'on voit apparaître les résultats de la classification des 5 genres en temps réel, comme le montre la copie d'écran de la figure 5.11. Ces résultats sont rafraîchis toutes les 12 secondes.

Figure 5.11 – Copie d'écran de l'IHM du navigateur. Les classes déterminées (jazz, classique, variété, bruit, non musique) sont affichées avec les logos des stations concernées

Avec une des versions bêtas de l'application d'indexation (version où seule la classification des genres 3 musicaux était implémentée), une série de tests visant à qualifier les algorithmes d'indexation sur nos flux démodulés a été publiée [HRDBV12]. Cette publication donne également des détails sur l'algorithme utilisé. Les figures suivantes (5.12 à 5.14) montrent les résultats de obtenus.

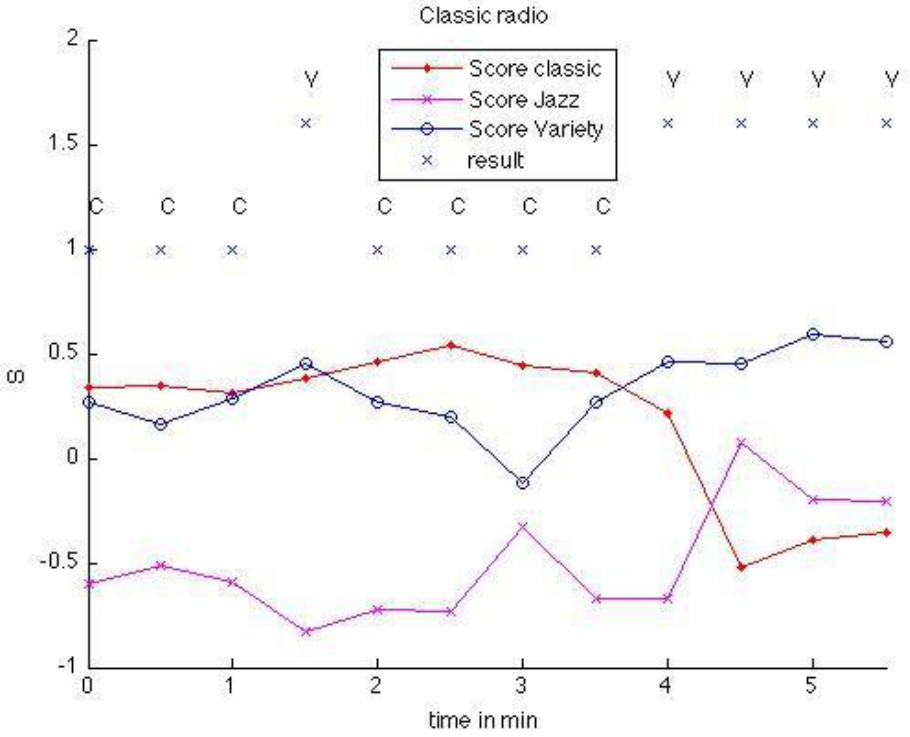


Figure 5.12 – Sorties du classifieur pour une radio de musiques classiques, à la fréquence 101.1 MHz à Paris. L'axe des abscisses est le temps en minutes et l'axe des ordonnées est le score normalisé. C, V et J représentent les résultats, respectivement le classique, la variété et le jazz

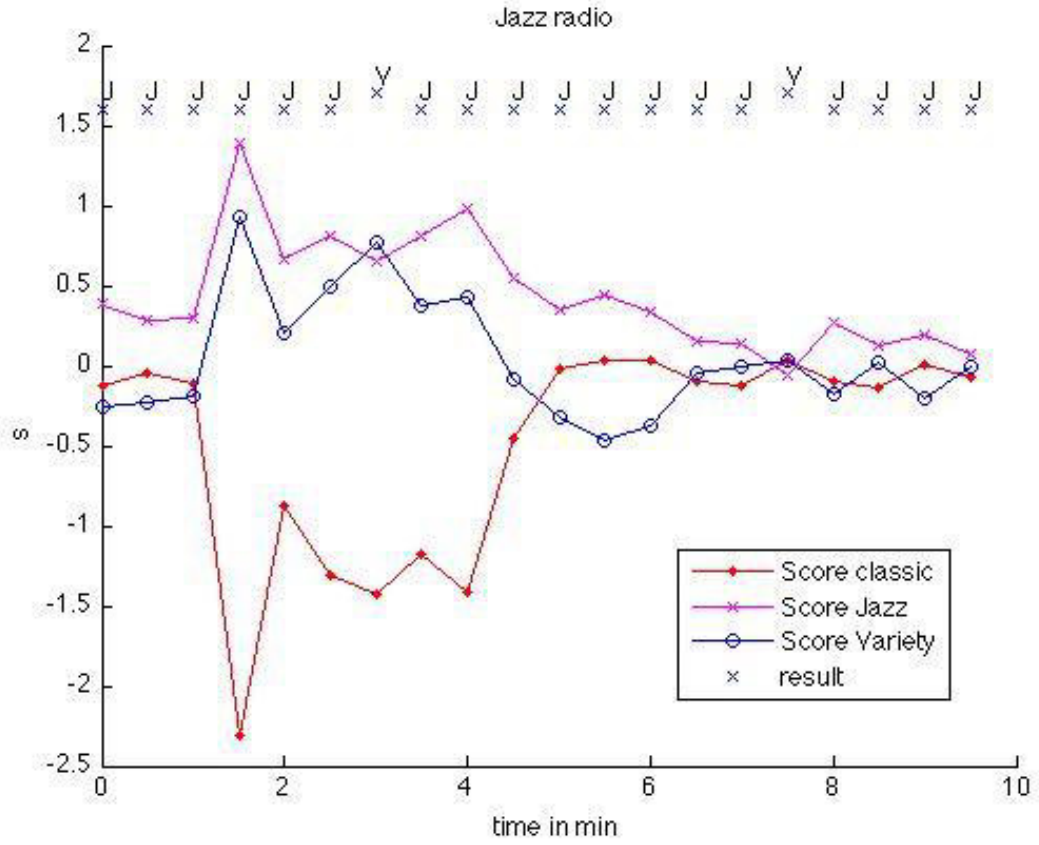


Figure 5.13 – Sorties du classifieur pour une radio de musiques jazz, à la fréquence 89.9 MHz à Paris. L'axe des abscisses est le temps en minutes et l'axe des ordonnées est le score normalisé. C, V et J représentent les résultats,

respectivement le classique, la variété et le jazz

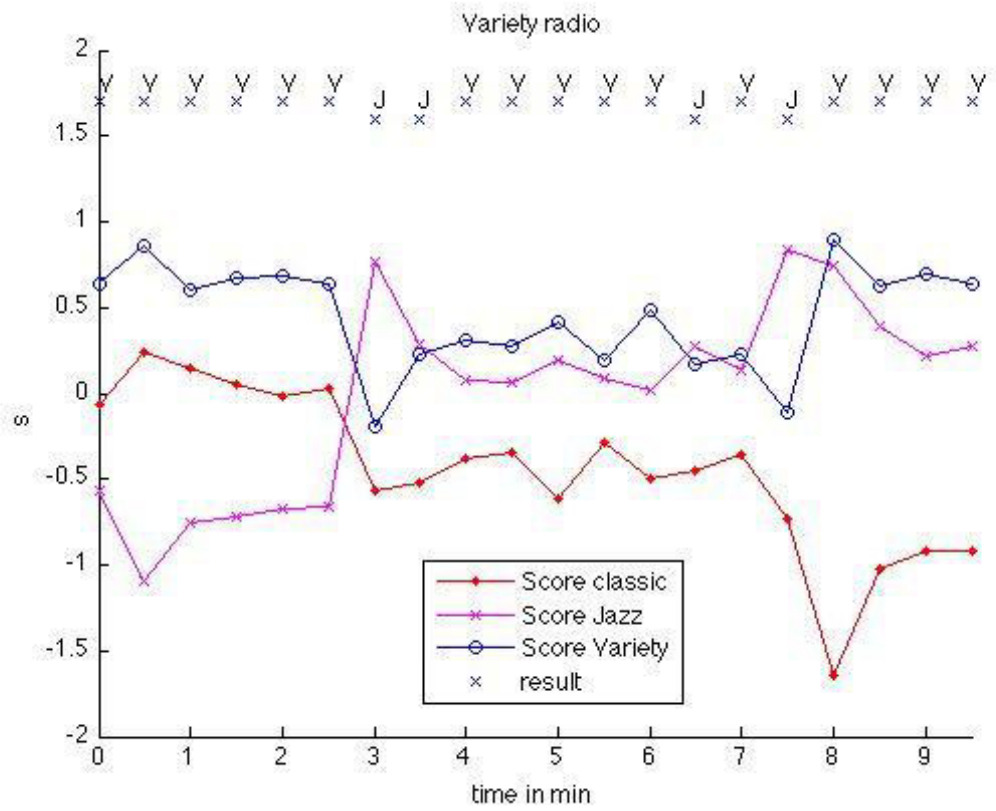


Figure 5.14 – Sorties du classifieur pour une radio de variétés, à la fréquence 96 MHz à Paris. L'axe des abcisses est le temps en minutes et l'axe des ordonnées est le score normalisé. C, V et J représentent les résultats, respectivement le classique, la variété et le jazz

Les résultats montrent que, dans la plupart des cas, la variété est bien détectée. Cependant, le classifieur de prototype est aussi très sensible aux bruits. Ainsi, lorsque le son grésille, le classifieur associe le classique à la variété ou au jazz comme dans la figure 5.12. Quant à la publicité, aucune classe spécifique n'a encore jusqu'aujourd'hui été créée pour elle. Ainsi, au moment où les tests ont été faits, la publicité ainsi que lorsque la parole diffusée avec un fond musical étaient souvent associée par le classifieur à la variété.

5.1.8.3 Conclusion

A partir des résultats de la classification, on déduit que le channelizeur/démodulateur FM fournit des flux démodulés dont la qualité permet l'exécution d'algorithmes d'indexations. Cependant, l'obtention d'une version finale du navigateur FM nécessitera des efforts supplémentaires pour :

- fournir un signal de la meilleure qualité possible au classifieur, pour éviter de fausser sa classification,

- améliorer l'algorithme de classification pour le rendre moins sensible au bruit, améliorer des résultats et rajouter une classe « publicité ».

5.2 Prototypage pour la bande DRM30

5.2.1 Front end analogique

Ce deuxième front-end, décrit dans la section 3.2.1.2, a été fabriqué pour la bande AM, qui est aussi la bande DRM30. Son montage permet selon les simulations de disposer d'un gain de 110 dB dans la branche GO+MO et 100 dB dans l'autre branche. Ainsi, il pourra amplifier des signaux de très faibles amplitudes (de l'ordre de 1µV) pour les envoyer à l'ADC. D'autre part, ce front-end est également très performant au niveau de la linéarité; il possède un IP3 de 40dBm.

Ces simulations, qui consistent en une entrée un signal de 10 porteuses, puis un signal de 20 porteuses donnent comme résultats respectifs un C/N est de l'ordre de 47 dB et 42 dB pour la branche GO+MO avec une amplification de 115 dB. Pour l'autre branche, les résultats sont respectivement 60 dB et 54 dB avec une amplification de près de 110 dB.

5.2.2 Re-échantillonneur

Le récepteur DRM proposé n'utilise pas de re-échantillonneur car il ne nécessite pas, comme le récepteur FM, de ramener le signal RF à la bande de base. Ici, la bande (3 – 30 MHz) est considérée comme étant très proche de la bande de base et il suffit de cadencer l'échantillonnage du signal à la fréquence qui conviendra au banc de filtre subséquent.

5.2.3 Banc de filtres pour la DRM

Le channelizeur est décrit dans la section 3.4.3. On cherche à obtenir une rejection de 40 dB. On a vu dans la section 3.4.3.2 que les paramètres suivants permettent d'y arriver : L_P = 16384, K_P = 1024, Lg = 80, F_S = 81.92 MHz.

On recense un certain nombre de stations DRM dont 12 [DRMorg] qui émettent des programmes en Europe et 1 en France [DRMorg]. Notre channelizeur est implémenté pour extraire les 10 canaux dont les stations ont les puissances d'émission les plus élevées (en réalité, en région parisienne – là où les développements ont eu lieu –, on réceptionne moins de 5 stations DRMs). Leurs fréquences de diffusion sont respectivement : 3955, 5875, 7355, 7390, 9450, 9600, 9760, 9780, 13720 et 15120 kHz. La valeur des fréquences des sinusoïdes générées par les NCOs

(10, 20 et 30 kHz) feront en sorte que 1 d'entre les stations seront pré-extraites par la première branche « WOLA + Goertzel », 4 par la 2ième branche, 4 par la 3ième branche et 1 par la dernière branche.

Dans le channelizeur implémenté, les blocs WOLA de toutes les branches sont approvisionnés par une mémoire unique de coefficients du filtre. Cela permet d'optimiser l'utilisation de la mémoire disponible dans le composant. Le tableau 5.8 résume les quantités des ressources utilisées après implémentation. Ces quantités sont déterminées par traitement des données affichées dans le rapport de compilation. Pour avoir une meilleure idée de ce qu'une branche utilise, le tableau 5.9 reporte les ressources utilisées par un channelizeur qui ne possède que la branche 3.

	Channelizeur avec 4 branches seulement (10 canaux extraits)				
	Channelizeur complet	Contribution de chaque partie			
		WOLA sans FFT	Goertzels	NCOs+Filtre pipeliné	Parties restantes
ALUTs	**39065**	6305	5369	26841	550
Registres logiques dédiés	**31225**	8837	5794	15988	606
Mémoire (en bits)	**1782688**	1695936	30720	51392	4640
Eléments DSP	**768**	304	192	272	0
Horloge de 81.92 MHz maximale	**95.72**	--	--	--	

Tableau 5.8 – Utilisation des ressources du channelizeur DRM 30. Les canaux sortants sont cadencés à 80 kHz. Les parties restantes désignent toute la logique, la mémoire et les blocs « dynamic adapter » utilisés pour inter-connecter les blocs décrits et implémenter le design sur la carte FPGA.

	Channelizeur avec branche 3 seulement et 4 canaux extraits				
	Channelizeur complet	Contribution de chaque partie			
		WOLA sans FFT	Goertzels	NCOs+Filtre pipeliné	Parties restantes
ALUTs	**16229**	1848	1709	12355	317
Registres logiques dédiés	**14142**	2433	2391	8982	336
Mémoire (en bits)	**613696**	573744	12288	25520	2144
Eléments DSP	**768**	80	96	592	0
Horloge de 81.92 MHz maximale	**94.48**	--	--		

Tableau 5.9 – Utilisation des ressources d'un channelizeur DRM 30 avec une branche seulement et 4 canaux extraits. Les canaux sortants sont cadencés à 80 kHz. Les parties restantes désignent toute la logique, la mémoire et les blocs « dynamic adapter » utilisés pour inter-connecter les blocs décrits et implémenter le design sur la carte FPGA.

On peut d'ailleurs noter que, bien que l'implémentation du channelizeur présentée ci-dessus concerne la bande DRM30, il va de soi qu'elle peut être adaptée à d'autres bandes comme la bande AM par exemple.

5.2.4 Démodulateur DRM30

Au moment de la rédaction du présent manuscrit, le développement du démodulateur DRM30 est encore en cours. Nous avons, dans le code C++ du logiciel DREAM, enlevé un bon nombre de fichiers et désactivé des fonctions qui ne contribuent pas à la démodulation de l'audio du DRM. Le code extrait fonctionne sur un PC, comme le montre le test de la section 5.2.6.2. Cependant, nous n'avons pas encore implémenté ce code dans le FPGA. Ce sera la prochaine étape, après l'avoir profilé, afin d'identifier les parties à accélérer matériellement et celles qui seront exécutées sur un processeur de type softcore. Par conséquent, comme le démodulateur DRM30 n'étant pas encore achevé, nous ne sommes pas allés plus dans le développement du récepteur DRM.

5.2.5 Prototype du récepteur DRM30 actuel

Compte tenu de notre avancement, la figure 5.15 illustre l'état actuel de notre prototype temps réel DRM30. Les canaux extraits, pour le moment, sont enregistrés dans des fichiers avant d'être

envoyés à un PC pour être démodulés « off line ». Ainsi, même si le channelizeur fonctionne en temps réel, ce n'est pas le cas de la chaine complète « channelizeur + démodulateur-PC ». La figure 5.16 est une photographie du prototype.

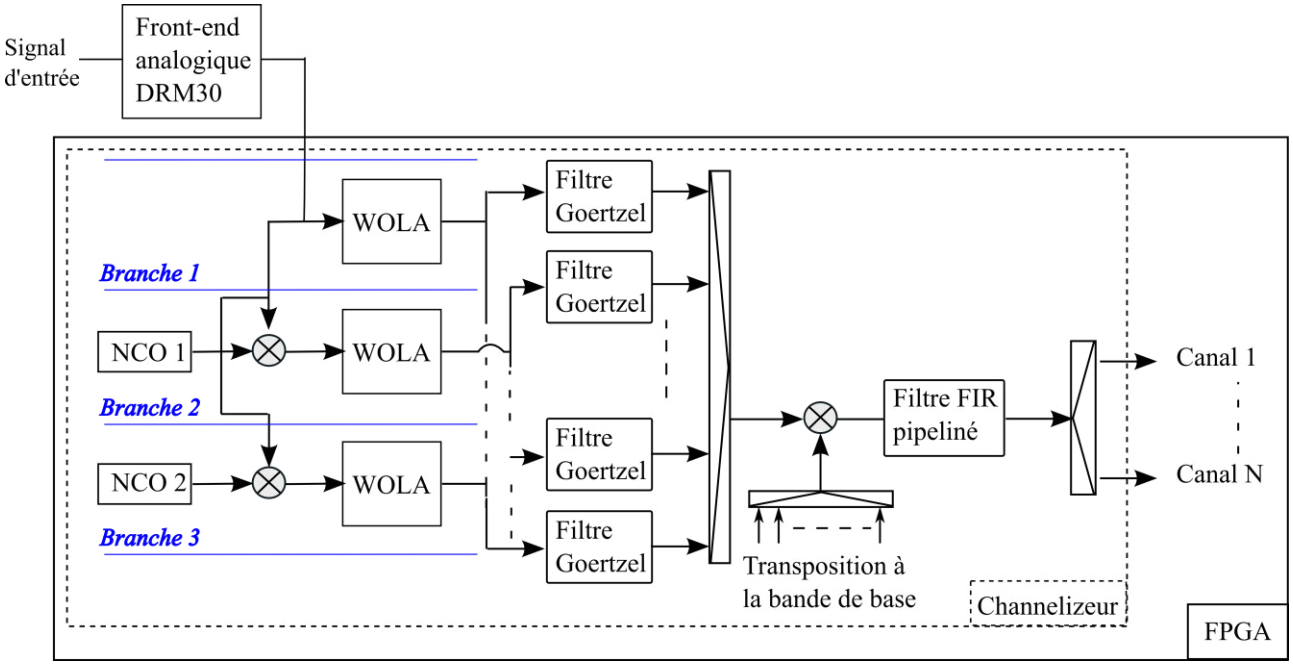


Figure 5.15 – Prototype temps-réel du récepteur DRM30 actuel

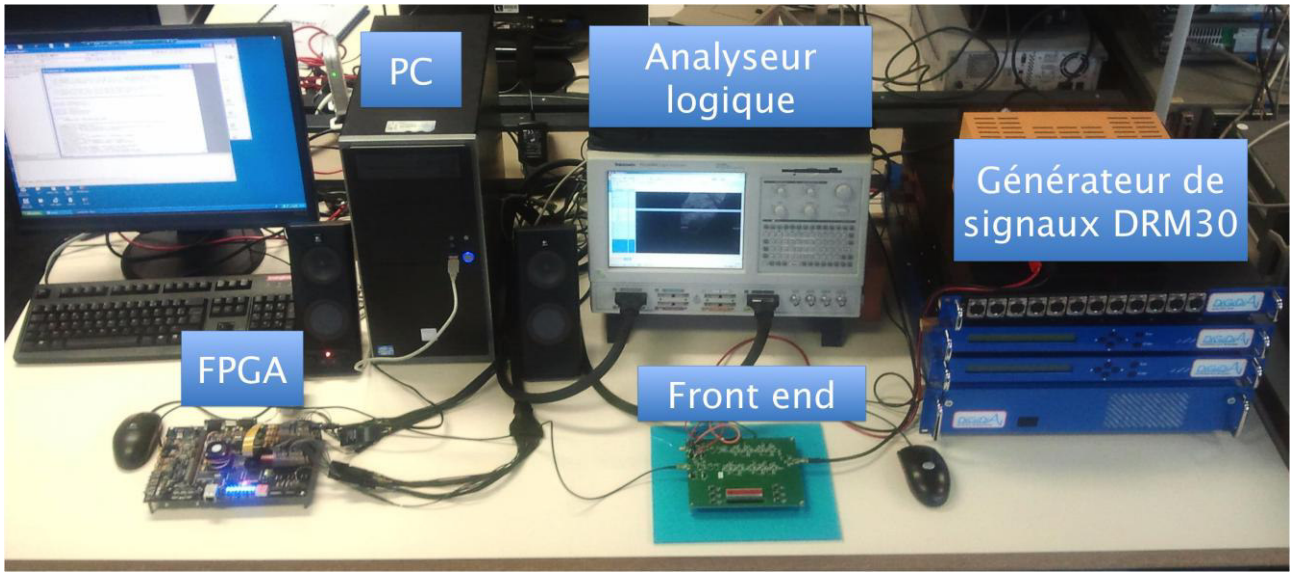


Figure 5.16 – Photographie du prototype du récepteur DRM30 actuel

5.2.6 Résultats expérimentaux sur le récepteur DRM

Les tests opérés sur le récepteur DRM visent à montrer que le channelizeur DRM extrait des

canaux de qualité suffisamment bonne pour être démodulés. Le récepteur complet (c'est-à-dire Acquisition+channelization+démodulation) n'étant pas opérationnel au moment de la rédaction du présent manuscrit, il n'est pas question ici de tester les algorithmes d'indexation.

Deux tests ont alors été faits sur le channelizeur DRM pour assurer sa validité :

- Le premier a consisté à injecter un signal constitué de 3 séries de 10 porteuses réparties dans le spectre pour « jouer de rôle » de 3 stations qui arrivent au channelizeur. Le but recherché est de vérifier que le channelizeur est bel et bien capable d'isoler chaque canal recherché.
- Le deuxième test a consisté à enregistrer dans un fichier un canal extrait en temps réel par le channelizeur sur un vrai signal DRM30, et ensuite à faire démoduler le fichier par le code DREAM modifié. Ce test montre alors que les canaux isolés sont aptes à la démodulation. Cela confirme aussi que l'objectif linéarité du processus est atteint. C'est-à-dire que la channelization conserve la linéarité minimale permettant ensuite de démoduler les stations DRM30.

5.2.6.1 Premier test

Les fréquences de chaque série sont choisies de façon à correspondre aux 3 canaux les plus rapprochés parmi les 10 de la section 5.2.3. De cette manière, on peut mettre le channelizeur dans un cas réel moins favorable que les autres. En effet, plus les canaux sont rapprochés les uns des autres, moins ils sont susceptibles d'être bien isolés par les mécanismes de channelization. Il devient alors plus aisé de mettre en évidence les défauts de ces mécanismes. On choisit les fréquences 9600, 9760, 9780. Le signal (dont le spectre est représenté dans la figure 5.17) est échantillonné à 81.92 MHz. Il est constitué de la somme de 30 NCOs instanciées dans le FPGA qui génèrent chacune une des 30 porteuses du signal. Ce signal est directement envoyé au channelizeur proposé. Le spectre tracé sur la figure 5.17 est une émulation Matlab du signal, car le matériel à disposition au moment du test ne permet pas d'enregistrer une quantité d'échantillons suffisante (plusieurs millions) du signal pour tracer son spectre détaillé. La figure 5.18 montre les canaux extraits par le channelizeur. On peut y voir que les canaux rejetés sont atténués d'au moins 40 dB comme souhaité dans la section 3.4.3.2. Cela confirme la capacité de notre channelizeur à isoler les stations recherchées.

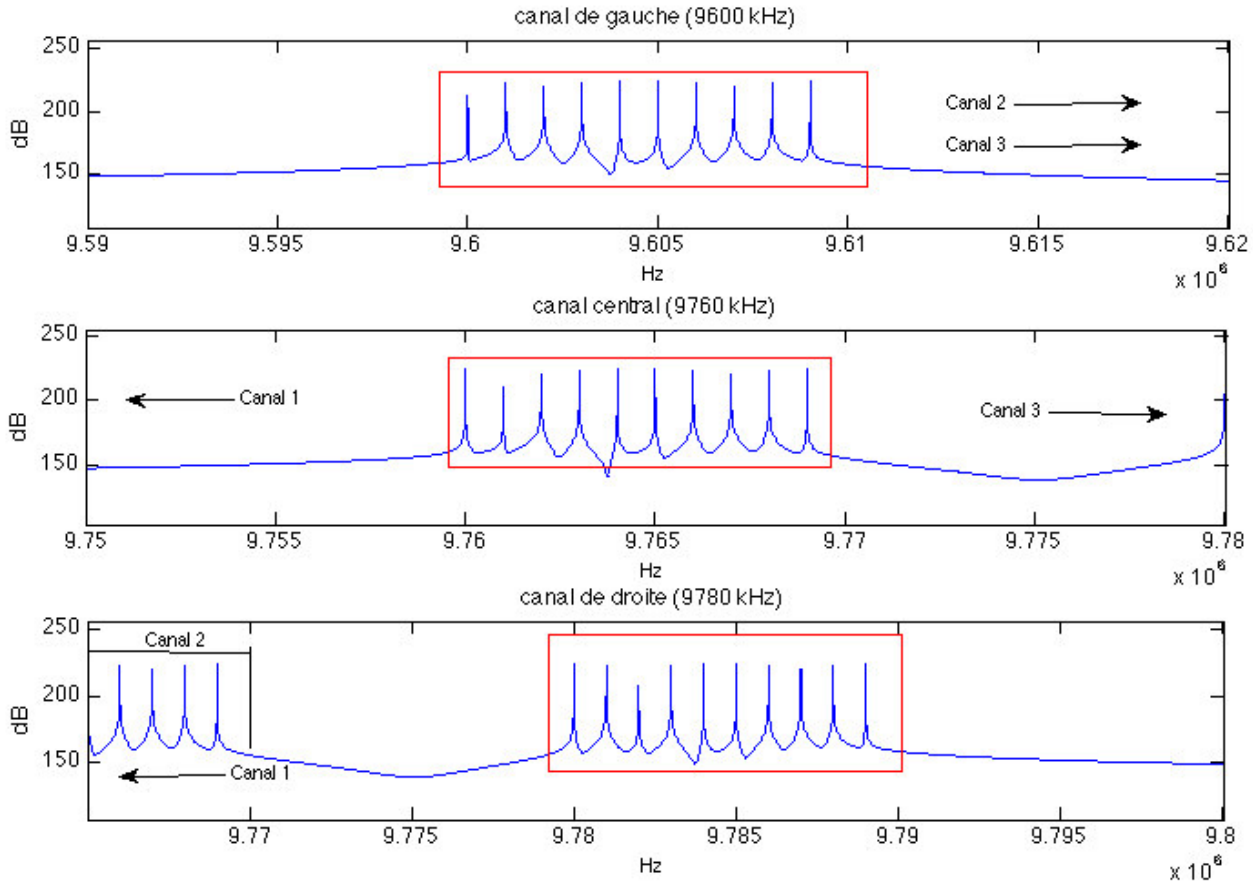


Figure 5.17 – Les 3 canaux qui constituent le signal à channelizer. Les carrés rouges représentent les stations larges de 10 kHz. Le canal alloué à chaque station est de 20 kHz (il s'agit de la largeur maximale allouée à un canal DRM30).

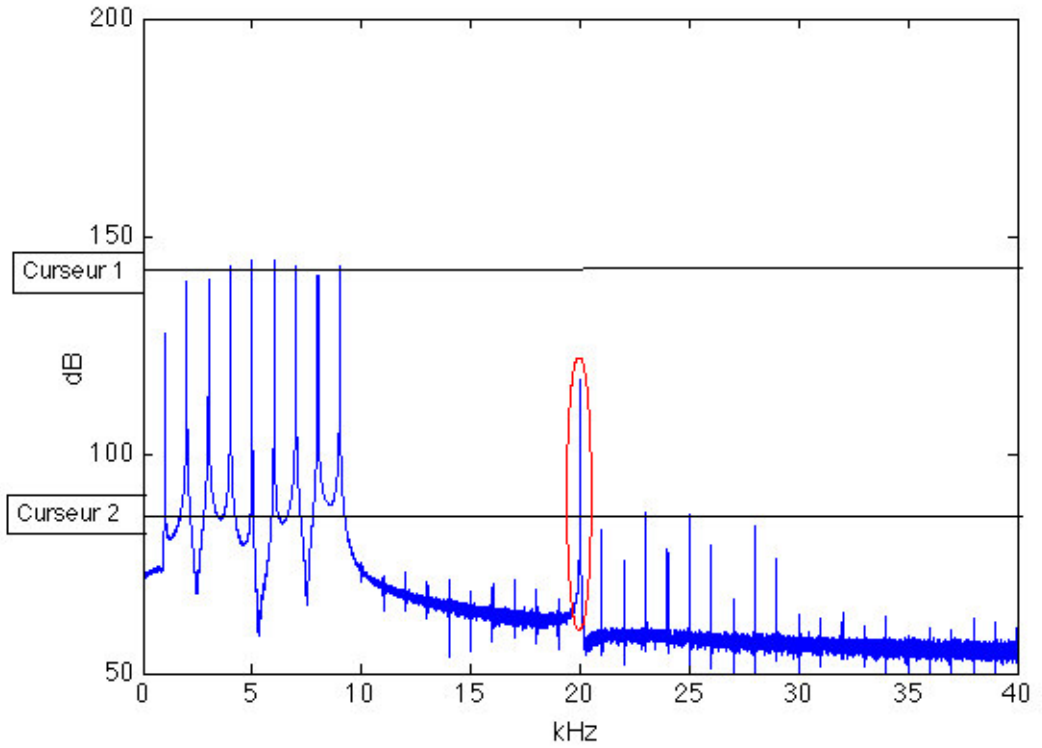


Figure 5.18 – canal 2 extrait après channelization. On constate que le rapport entre le signal de départ et les harmoniques (lignes horizontales) est largement supérieur à 40 dB. La porteuse entourée est un débordement de la station 3 dans le canal réservé à la station 2. Etant déjà présent avant la channelization, il ne s'agit donc pas d'une harmonique parasite proprement dite. On constate qu'il y a une forte rejection des porteuses suivantes – qui n'appartiennent pas au canal 2 – comme on le souhaitait.

5.2.6.2 Deuxième test

La figure 5.19 illustre le protocole expérimental utilisé.

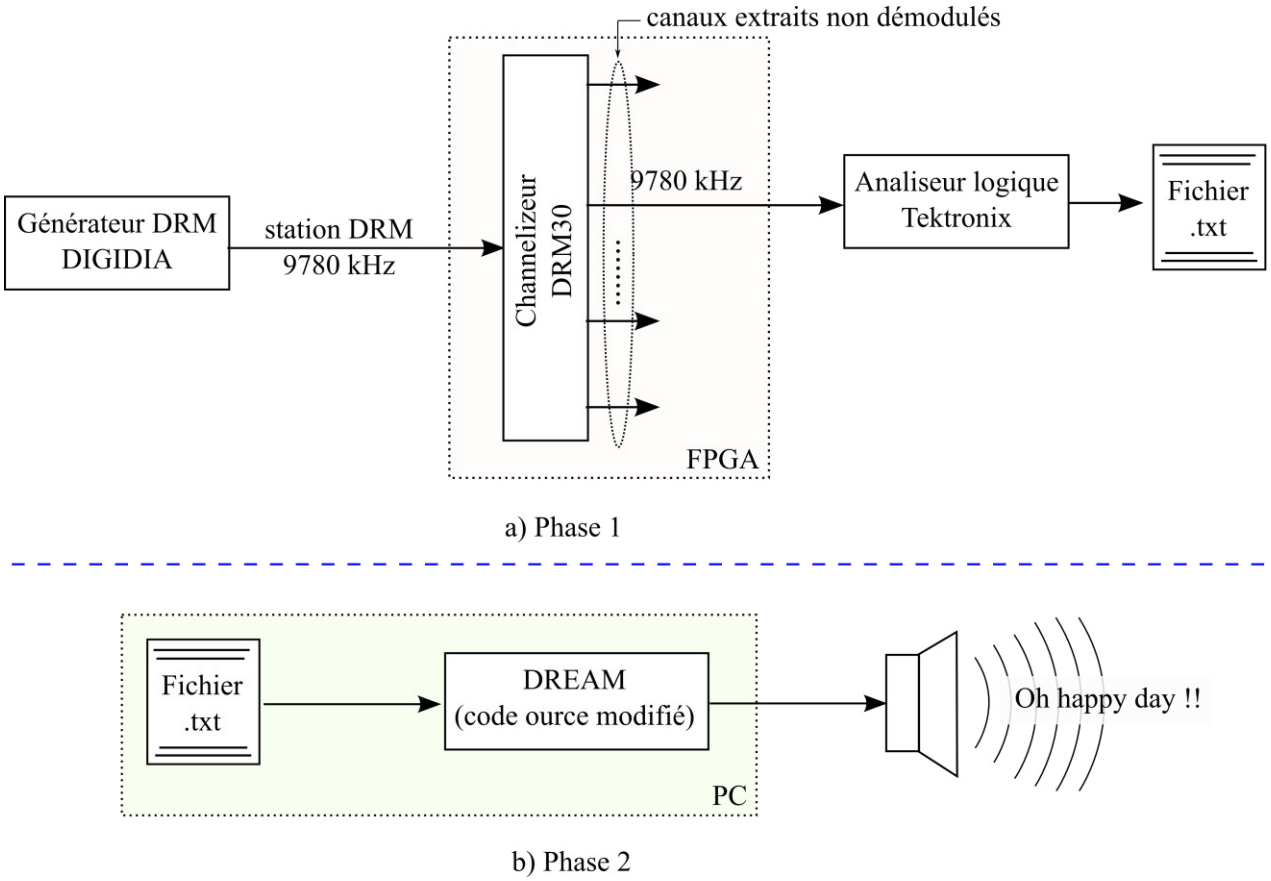


Figure – 5.19: Protocole expérimental du deuxième test sur la DRM.

Sur la figure 5.16 apparaissent tous les éléments utilisés pour le test. Celui-ci a consisté à générer une station DRM avec un générateur approprié (le générateur DIGIDIA) et à l'injecter dans le Stratix II dans lequel un channelizeur DRM est implémenté. Ici, la rejection de la channelization fixée dans le channelizeur est supérieure à 40 dB. Grâce à un analyseur logique (le Tectronix TL5400), le signal issu de channelizeur a été prélevé et stocké dans un fichier avant d'être transféreé dans le filtre FIR pipeliné. C'est la phase 1. Ensuite, dans la phase 2, le fichier a été utilisé comme source d'entrée pour le programme DREAM modifié par nos soins. Une fois l'exécution du programme lancée, on a pu écouter le son audio démodulé. Le test a donc été concluant, validant ainsi le fonctionnement du code DREAM modifié et attestant de la bonne linéarité du processus de channelization. En effet, on peut premièrement supposer que le filtre FIR pipeliné à phase linéaire ne dégradera pas significativement la linéarité du processus. Ensuite, il n'est pas non plus insensé de supposer que si l'un des deux (le fonctionnement du code modifié ou la channelization à phase linéaire) ne s'exécutait pas convenablement, nous n'aurions pas écouté le son démodulé.

5.3 Positionnement des prototypes par rapport aux récepteurs existants

Nous arrivons au terme de la présentation du travail effectué durant la thèse, il convient de montrer le positionnement de nos prototypes par rapport aux récepteurs multi-standards et / ou multi-canaux existants (ces derniers ont été présentés dans l'état de l'art, section 2.5). Pour se faire, le tableau 5.10 récapitule les performances obtenues par nos prototypes et celles des récepteurs existants. Ainsi, par rapport au premier objectif recherché qui était de faire la démodulation simultanée de toutes les stations des standards étudiés, nos prototypes montrent clairement que les méthodologies de démodulation proposées sont des avancées significatives qui se démarquent des systèmes existants.

	Multi-standards ?	**Multi-canaux ?**	**Toutes les stations ?**	**Combien de canaux simultanément ?**	**Standards Concernés**
Prototype du récepteur FM	**Non**	**Oui**	**Oui**	**> 200**	**FM**
Prototype du récepteur DRM30*	**Non**	**Oui**	**Oui****	**10****	**DRM30**
Ihmig et Herkersdorf	Oui	Oui	Non	14 (FM : 2, DAB : 12)	FM, DAB
Sliskovic	Oui	Oui	Non	2	AM, FM, DRM, IBOC
Luo *et al*	Oui	Plateforme orientée communications et non radio diffusion			
GNURadio	Oui	Non	Sans objet		AM, FM, DRM, DAB
WiNRADiO	Oui	Oui	Non	8 (MS-8118/BRL)	AM, FM, DRM
Dream	Oui	Non	Sans objet		
SoDiRa	Oui				AM, AMSS, FM/RDS, DRM
SDRTouch	Oui				AM, FM, etc.
Wavesink	Oui				FM/RDS, DAB, DRM+
CDRX-3200*	Oui	Oui	Non	32	FM, FM HD
Romain et Denby	Non	Oui	Non	5	FM
RS500	Oui	Non	Sans objet		AM, FM (RDS), DRM, DAB
SAF3560	Oui	Non			AM, FM, HD Radio
SAF3561	Oui	Non			DAB, DAB+ et T-DMB.
MS-8118/BRL	Oui	Oui	Non	8	AM, FM, DRM
ClearSignal-SoC™	Oui	Non	Sans objet		DAB, DAB+, T-DMB, DRM, HD Radio

* Le dispositif CDRX-3200 de FlexRadio et notre prototype DRM30 capturent et extraient les canaux sans les démoduler, contrairement aux autres systèmes répertoriés dans le tableau.

** Bien que notre prototype DRM30 a été programmé pour extraire 10 stations, la méthodologie développée permet d'extraire toutes les stations existantes (environ 60 répertoriés à ce jour [DRMorg]) mais encore faudrait-il les capter.

Tableau 5.10 – Positionnement de notre prototype par rapport aux récepteurs existants

5.4 Bibliographie

[DASIP12] Tietche, B.H.; Romain, O.; Denby, B.; Benaroya, L.; De Dieuleveult, F.; Granado, B.; Wassi, G.; Khemiri, H.; Chollet, G.; Petrovska, D.; Blouet, R.; Hachicha, K.; Viateur, S., "Prototype of a radio-on-demand broadcast receiver with real time musical genre classification," *2012 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, vol., no., pp.1,2, 23-25 Oct. 2012

[DataMining12] « Le projet Surf On Hertz - la démodulation simultanée de toute la bande FM », *Séminaire sur le Data Mining. Département d'Informatique de l'Université de Yaoundé I*, Yaoundé, Cameroun, 4 – 11 Juin 2012.

[DRMorg] Site web du Digital Radio Mondiale. <http://www.drm.org>. Dernier accès : 15 Août 2013.

[ElHannani2005] A. El Hannani and D. Petrovska-Delacrétaz. Exploiting high-level information provided by ALISP in speaker recognition. In the proceedings of the Non Linear Speech Processing Workshop (NOLISP), April 2005.

[ElHannani2006] A. El Hannani, D.T. Toledano, D. Petrovska-Delacrétaz, A. Montero-Asenjo, and J. Hennebert. Using data-driven and phonetic units for speaker verification. In the proceedings of the IEEE Workshop on Speaker and Language Recognition (Odyssey), June 2006.

[ElHannani2007] A. Elhannani. Text-independent Speaker Verification based on High Level Information Extracted with Data-driven Methods. PhD Thesis, Université de Fribourg and Institut National des Télécommunications Evry, 2007.

[Essid2005] S. Essid. Classification automatique des signaux audio-fréquences: reconnaissance des instruments de musique, Thèse Univ. Paris 6

[Herrera2006] P. Herrera, A. Klapuri, M. Davy. Chap.6 Automatic Classification of Pitched Musical Instrument Sounds. Signal Processing methods for Music transcription, Edited by A. Klapuri and M. Davy, Springer (2006).

[HRDBV12] Happi Tietche, B.; Romain, O.; Denby, B.; Benaroya, L.; Viateur, S., "FPGA-based radio-on-demand broadcast receiver with musical genre identification," *2012 IEEE International Symposium on Industrial Electronics (ISIE)*, vol., no., pp.1381,1385, 28-31 May 2012

[IH09] M. Ihmig, and A. Herkersdorf, "Flexible multi-standard multi-channel system architecture for Software Defined Radio receiver," in *9th IEEE Intelligent*

Transport Systems Telecommunications Int. Conf., ITST., Lille, France, Oct. 20 – 22, 2009, pp. 598-603.

[Lamel2008] Lori Lamel and Jean-Luc Gauvain. Speech processing for audio indexing, In Proceedings of the 6th International Conference on Natural Language Processing, GoTAL 2008 - Advances in Natural Language Processing, number 5221/2008 in Lecture Notes in Computer Science, pages 4-15. Springer Verlag, Berlin/Heidelberg, 2008.

[NuméANR13] « SurfOnHertz », *Rencontres du numériques ANR*. Centre des Congrès de la Cité des Sciences et de l'Industrie, Paris, France, 17 – 18 Avril 2013.

[Peeters07] G. Peeters, A Generic System for Audio Indexing: Application to Speech/Music Segmentation and Music Genre Recognition, in *Proc. of the 10th Int. Conference on Digital Audio Effects*, *DAFx-07*, Bordeaux, France, September 10-15, 2007.

[Peeters2004] G. Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Technical report, IRCAM (2004).

[Philips92] Philips Corporation, "TDA7000, FM radio circuit," datasheet, May 1992.

[Schatter2008] G. Schatter, B. Zeller, A. Eiselt, Navigating Spoken Broadcast Content of a Digital Radio DAB by Music-Speech Discrimination and Information Retrieval, 9th Digital Broadcasting Workshop, 18 et 19 septembre 2008, Erlangen, Allemagne.

[Schwarz2004] Diemo Schwarz. Data-Driven Concatenative Sound Synthesis. PhD Thesis in Acoustics, Computer Science, Signal Processing Applied to Music, Université Paris 6 - Pierre et Marie Curie, January 20, 2004

[TECHNOLnet] <http://www.technolangue.net/article60.html>. Dernier accès le 12 Oct. 2013

6 Conclusion générale et perspectives

6.1 Conclusion

Il existe une panoplie de récepteurs pour les différents standards de radio. Nous avons pu découvrir beaucoup d'entre eux dans le chapitre 2. Parmi eux, il y en qui permettent de recevoir plusieurs standards différents, d'autres permettent de recevoir à la fois plusieurs standards et plusieurs canaux simultanément. Cependant, parmi tous ces récepteurs, il n'y en a aucun qui a vocation à extraire et à démoduler simultanément toutes les stations existantes soit dans un standard radio particulier, soit dans plusieurs standards. Ces récepteurs ne sont donc pas aptes à répondre à un besoin d'indexation permanente de toutes les stations simultanément au sein d'un dispositif embarqué. Comme tel était l'objectif du projet SurfOnHertz, il était donc nécessaire de développer un nouveau type de récepteurs qui répondent à la fois au critère de l'embarquabilité et au critère « multi-canaux multi-standards ». Les travaux présentés dans ce manuscrit sont une avancée considérable dans la réalisation de ces nouveaux récepteurs.

Partant de travaux qui avaient été menés entre 2002 et 2007 [RD07][DR04][RD04][DH03] par les initiateurs du projet, nous avons d'abord établi la structure que devait avoir un tel récepteur dédié au standard FM. Cette architecture, comme nous l'avons présenté, est la combinaison d'un front-end analogique, un ADC, un re-échantillonneur, un channelizeur, un démodulateur multi-canaux et un protocole de transfert des stations démodulées.

La conception du front end analogique était une tâche qui incombait à un autre partenaire du projet. Pour réaliser les autres parties du récepteur, notre regard s'est porté sur une cible de type FPGA, qui offre de grandes capacités en termes de parallélisme, maitrise des ressources et embarquabilité.

L'ADC était déjà présent sur la carte du FPGA, il a suffit de le programmer pour qu'il conserve toute la bande FM et la ramène à une fréquence proche de la bande de base. Lors du développement du re-échantillonneur, il nous est paru judicieux de proposer une méthode de re-échantillonnage générale pour la proposer à tout concepteur de circuit sur FPGA faisant un traitement de signal dans une bande comprise entre la LF et la VHF. Cette méthode permet au concepteur de fixer le SFDR minimum de son processus de re-échantillonnage, de ne pas utiliser d'autre horloge que celles d'entrée et de sortie (c'est souvent le cas), et de donner au concepteur une idée fiable des ressources que le re-échantillonneur utilisera. Un article de revue a été publié à ce sujet [HRD13].

Quant au channelizeur, nous avons exploité l'idée de 4 bancs de filtres WOLA entrelacés pour le mettre au point. Ces quatre bancs de filtres, autrement dit ces quatres branches, permettent d'extraire et d'isoler toutes les canaux possibles contenus dans la bande FM étudiée. Il fallait ensuite pouvoir démoduler tous ces canaux simultanément. Nous avons essayé d'user de capacité de parallélisme du FPGA en dupliquant une IP de démodulation « arctangente+différencier » autant de fois qu'il y a de canaux extraits. Notre FPGA (Stratix II EP2S180) ne pouvait pas le supporter car n'ayant pas la quantité de ressources logiques requises. Nous avons alors opté pour une solution très efficace : une IP « pipelinée » par branche.

Une fois tous les canaux FM démodulés, nous nous sommes mis d'accord avec les autres partenaires du projet, ceux qui ont implémenté la machine d'indexation, pour définir un protocole de transfert des stations à cette dernière. Le prototype de navigateur hertzien pour la FM est ainsi né, et a été présenté lors de différents évènements en France [NuméANR13], en Allemagne où il a reçu la récompense du « Best Demo Award » [DASIP12] et en Afrique [DataMining12]. Quant au récepteur FM spécifiquement, il a fait l'objet d'une revue dans IEEE Transactions on Consumer Electronics [TRDD12].

Après avoir achevé le récepteur FM, il fallait s'attaquer à ce qui allait présenter des difficultés d'un autre genre : la conception du récepteur DRM30. Nous avons commencé par développer un channelizeur adapté à cette bande. La difficulté fût grande pour trois raisons. La première est que les stations DRM peuvent être situées « n'importe où » dans la bande avec une granularité de l'ordre 1 kHz (contrairement à 100 kHz dans la bande FM). La deuxième est que les canaux DRM30 sont très étroits par rapport à la bande. Il fallait donc un channelizeur très flexible possédant une capacité de filtrage très poussée. La troisième est que le filtrage devait être à phase linéaire, afin de ne pas nuire à la démodulation subséquente. Nous sommes parvenus à un channelizeur issu de la combinaison de plusieurs algorithmes (WOLA+Goertzel+DDC), et dont le processus utilise un minimum de multiplications.

Contrairement à channelizeur FM, le channelizeur DRM30 développé ne nécessitait pas d'être précédé d'un re-échantillonneur. Nous avons aussi entamé le développement d'un démodulateur DRM30 destiné à être intégré au FPGA. Pour cela, nous nous sommes servis de l'open source DREAM. Actuellement, nous avons épuré le code de plusieurs fichiers sources ou désactivé des fonctions que nous ne jugions pas importantes la démodulation audio. Le code obtenu n'a pas encore été embarqué dans le FPGA : ce sera la prochaine étape, après l'avoir profilé. Il a néanmoins servi à valider et attester que le channelizeur développé fonctionne.

Un navigateur hertzien basé sur la DRM30 n'est donc certes pas encore opérationnel, mais tous les ingrédients pour y arriver semblent présents. L'obstacle à franchir est le développement d'un démodulateur multi-canaux au sein du FPGA.

Ces travaux sont une avancée car ils débouchent sur un récepteur FM qui démodule toutes les stations de la bande, ainsi que sur un channelizeur qui peut extraire toutes les stations DRM30. A notre connaissance, aucune autre architecture n'en était capable jusqu'à présent. Ces travaux ont fait l'objet de deux publications dans des journaux et un troisième est en cours de rédaction sur la DRM. Il fait parallèlement l'objet d'un dépôt de brevet.

On peut aussi rajouter que ces travaux sont extensibles à la bande AM et DRM+. En effet, le channelizeur DRM30 est utilisable en AM et le channelizeur du récepteur FM est utilisable en DRM+. En plus, il est aisé d'implémenter un démodulateur AM « en hard ». Et même si ce n'était pas le cas, l'open source DREAM permet de démoduler l'AM. On pourrait alors épurer le code de DREAM pour qu'il ne traite que l'AM. Cependant, la démodulation du DRM+ nécessitera de se procurer (ou d'écrire) un code source le fait. En effet, à l'heure actuelle, ni DREAM, ni Diorama ne prennent en charge ce standard pour l'instant.

6.2 Perspectives

Afin d'amener le travail entamé à son aboutissement, il faut encore livrer quelques batailles à court et moyen terme :

- **Intégrer le démodulateur DRM30 au FPGA**. Pour cela, il faudra d'abord profiler le code source DREAM modifié prévu et après étudier son intégration dans la puce. Cela sera peut-être possible en implémentant un décodeur Viterbi en « hard » et deux 2 processeurs dans le FPGA pour les autres fonctions du démodulateur. C'est proche de ce qui s'est fait pour un autre projet [DiMITRI06]. Dans ce projet, il a été question d'utiliser 2 processeurs indépendants, un ARM920 et un ARM946, pour implémenter les fonctions réjection des interférences, synchronisation, démultiplexage, décodage de l'audio. Ces processeurs ont été utilisés conjointement à des DDCs et un décodeur Viterbi codés en « hard ». Notre travail sera alors d'implémenter toutes ces composantes au sein d'un même FPGA.
- **Rendre le démodulateur DRM30 capable de traiter plusieurs canaux simultanément.**
- **Faire cohabiter le récepteur DRM30 et le récepteur FM au sein d'un même FPGA**. Cela permettra d'attester de la faisabilité d'un récepteur unique embarqué multi-standards/toutes-stations-diffusées.
- **Etendre le dispositif de réception à la bande DRM+.** Le channelizeur requis ici sera le même que celui développé pour la FM. Il faudra alors modifier notre démodulateur DRM30 pour qu'il traite également la DRM+.

6.3 Bibliographie

[DASIP12] Tietche, B.H.; Romain, O.; Denby, B.; Benaroya, L.; De Dieuleveult, F.; Granado, B.; Wassi, G.; Khemiri, H.; Chollet, G.; Petrovska, D.; Blouet, R.; Hachicha, K.; Viateur, S., "Prototype of a radio-on-demand broadcast receiver with real time musical genre classification," *2012 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, vol., no., pp.1,2, 23-25 Oct. 2012

[DataMining12] « Le projet Surf On Hertz - la démodulation simultanée de toute la bande FM », *Séminaire sur le Data Mining. Département d'Informatique de l'Université de Yaoundé I*, Yaoundé, Cameroun, 4 – 11 Juin 2012.

[DH03] Denby, B.; Hariti, S.-A., "Towards a software-radio enabled broadcast media navigator," *Video/Image Processing and Multimedia Communications, 2003. 4th EURASIP Conference focused on* , vol.2, no., pp.817,822 vol.2, 2-5 July 2003

[DiMITRI06] Jérôme Quévremont, « "DiMITRI". SoC architectures for embedded applications: illustration with the DiMITRI example ». *UPC-ENSEA*, 23 Novembre 2006.

[DR04] B. Denby, O. Romain, S.-A. Hariti, "A Software Radio Approach to Commercial FM Content Indexing", IEEE IWSSIP04, pp. 63-66, Poznan, Pologne, 2004. ISBN : 83-906074-8-4.

[HRD13] B. Happi Tietche, O. Romain and B. Denby, "A Pratical FPGA-Based Architecture for Arbitrary Ratio Sample Rate Conversion", *Journal of Signal Processing Systems*, *Springer*, Sept. 2013, doi: 10.1007/s11265-013-0840-5 <Free online access: http://link.springer.com/article/10.1007%2Fs11265-013-0840-5>. Accessed 18th Nov. 2013.

[NuméANR13] « SurfOnHertz », *Rencontres du numériques ANR*. Centre des Congrès de la Cité des Sciences et de l'Industrie, Paris, France, 17 – 18 Avril 2013.

[RD04] O. Romain & B. Denby,"A Real Time Software Radio for the AM Long Wave Band",11th International Workshop on Systems, Signals and Image Processing, IWSSIP'04 Ambient Multimedia, September 13-15, 2004, Poznan, Poland. ISBN : 83-906074-8-4.

[RD07] O. Romain & B. Denby, "Prototype of a Software Defined Broadcast Media Indexing Engine", in proc IEEE International Conference on Acoustic Signal Speech and Signal Processing, ICASSP 2007, 15-20 April 2007, Honolulu, Hawaii USA. ISBN :1-4244-0727-3. Doi :10.1109/ICASSP.2007.366360

[TRDD12] Tietche, B.H.; Romain, O.; Denby, B.; Dieuleveult, F.D., "FPGA-based

simultaneous multichannel fm broadcast receiver for audio indexing applications in consumer electronics scenarios," *IEEE Transactions on Consumer Electronics*, vol.58, no.4, pp.1153, 1161, Nov. 2012, doi: 10.1109/TCE.2012.6414980

7 Publications

7.1 Revues / Journals

- **B. Happi Tietche**, O. Romain and B. Denby, "A Pratical FPGA-Based Architecture for Arbitrary Ratio Sample Rate Conversion", *Journal of Signal Processing Systems*, *Springer*, Sept. 2013, doi: 10.1007/s11265-013-0840-5
  <Free online access: http://link.springer.com/article/10.1007%2Fs11265-013-0840-5>. Accessed 18th Nov. 2013.

- **Tietche, B.H.**; Romain, O.; Denby, B.; Dieuleveult, F.D., "FPGA-based simultaneous multichannel fm broadcast receiver for audio indexing applications in consumer electronics scenarios," *IEEE Transactions on Consumer Electronics*, vol.58, no.4, pp.1153, 1161, Nov. 2012, doi: 10.1109/TCE.2012.6414980

7.2 Conférences internationales avec comité de lecture et actes / Peer reviewed international conferences papers

- **Tietche, B.H.**; Romain, O.; Denby, B.; Benaroya, L.; De Dieuleveult, F.; Granado, B.; Wassi, G.; Khemiri, H.; Chollet, G.; Petrovska, D.; Blouet, R.; Hachicha, K.; Viateur, S., "Prototype of a radio-on-demand broadcast receiver with real time musical genre classification," *2012 Conference on Design and Architectures for Signal and Image Processing (DASIP),* Karlsruhe, Germany, pp. 1,2, 23-25 Oct. 2012. **« Best Demo Award »**

- **Happi Tietche, B.**; Romain, O.; Denby, B.; Benaroya, L.; Viateur, S., "FPGA-based radio-on-demand broadcast receiver with musical genre identification," *2012 IEEE International Symposium on Industrial Electronics (ISIE),* Hangzhou, China, pp.1381, 1385, 28-31 May 2012, doi: 10.1109/ISIE.2012.6237292

- **Tietche, B.H.**; Romain, O.; Denby, B.; Benaroya, L.; De Dieuleveult, F.; Granado, B.; Khemiri, H.; Chollet, G.; Petrovska-Delacretaz, D.; Blouet, R.; Hachicha, K.; Viateur, S., "Software radio FM broadcast receiver for audio indexing applications," *2012 IEEE*

International Conference on Industrial Technology (ICIT), Athens, Greece, pp. 585,590, 19-21 Mar. 2012, doi: 10.1109/ICIT.2012.6210002

7.3 Colloques nationaux / National symposiums

- **Brunel Happi Tietche**, Olivier Romain, Bruce Denby, "Sous-échantillonneur à ratio arbitraire sur FPGA", *Groupement De Recherche System-On-Chip/System-In-Package, GDR SOC-SIP*, Lyon, France, 10-12 juin 2013.

- **Brunel Happi-Tietche**, Olivier Romain, Bruce Denby, et al, "Un démodulateur parallèle en SDR pour la détection des publicités sur la bande de radiodiffusion FM", *Groupement De Recherche System-On-Chip/System-In-Package, GDR SOC-SIP*, Paris, France, 13-15 juin 2012.

7.4 Animation de séminaires / Animation of seminars

- « SurfOnHertz », *Rencontres du numériques ANR*. Centre des Congrès de la Cité des Sciences et de l'Industrie, Paris, France, 17 – 18 Avril 2013.

- « Le projet Surf On Hertz - la démodulation simultanée de toute la bande FM », Séminaire sur le Data Mining. Département d'Informatique de l'Université de Yaoundé I, Yaoundé, Cameroun, 4 – 11 Juin 2012.