Architecture de circuit intégré reconfigurable, très haut débit et basse consommation pour le traitement numérique de l'OFDM avancé

C. Sahnine

INSTITUT POLYTECHNIQUE DE GRENOBLE

N° attribué par la bibliothèque
ISBN : 978-2-84813-130-6

THESE

pour obtenir le grade de

DOCTEUR DE L'Institut polytechnique de Grenoble

Spécialité : Micro et Nanoélectronique
préparée au laboratoire **TIMA de Grenoble**
et **France Telecom R&D Meylan**
dans le cadre de **l'Ecole Doctorale**
Electronique, Electrotechnique, Automatique, Télécommunications, Signal
Présentée et soutenue publiquement

par

Chawki Sahnine
le 30 Janvier 2009

Architecture de circuit intégré reconfigurable, très haut débit et basse consommation pour le traitement numérique de l'OFDM avancé

Directeur de thèse : Frédéric Pétrot
Co-directeur de thèse : Nacer-Eddine Zergainoh

JURY

M. Maurice Bellanger	, Président
M. Emmanuel Casseau	, Rapporteur
M. Jean-François Hélard	, Rapporteur
M. Frédéric Pétrot	, Directeur de thèse
M. Nacer-Eddine Zergainoh	, Co-directeur de thèse
M. Denis Callonnec	, Examinateur
M. Yves Mathieu	, Examinateur
M. Pierre Siohan	, Examinateur

Remerciements

Mes premières pensées et remerciements vont à mes parents, sans qui je n'aurais pas pu accomplir mes projets personnels, plus particulièrement cette thèse. Leur support et leur soutien ont été sans failles. Je leur dédie pleinement cette thèse avec un grand Merci !

Toute ma gratitude à M. Denis Callonnec, responsable de l'unité de R&D COST à Orange Labs à Meylan et l'instigateur de cette thèse, qui a accepté de me prendre comme stagiaire de master recherche en 2005 et qui m'a proposé cette thèse. Je le remercie pour sa confiance, sa disponibilité ainsi que sa relecture très attentive de ce manuscrit.

Je tiens à remercier sincèrement mes directeurs de thèse, M. Frédéric Pétrot, professeur au laboratoire TIMA de l'Institut Polytechnique de Grenoble, et M. Nacer-Eddine Zergainoh, maitre de conférence à l'université Joseph Fourier à Grenoble, pour leur très bon encadrement, à la fois scientifique et humain, leurs conseils et pour le temps consacré à la réalisation de cette thèse. Elle n'aurait pas pu aboutir sans leur aide !

J'adresse tous mes remerciements à l'ensemble des membres du jury. Je remercie M. Maurice Bellanger, professeur au CNAM, de m'avoir fait l'honneur de présider le jury. A M. Emmanuel Casseau, professeur à l'ENSSAT à Lannion et M. Jean-François Hélard, professeur à l'INSA de Rennes, qui ont accepté d'être les rapporteurs de cette thèse et pour leur commentaires sur mon manuscrit. A M. Yves Mathieu, professeur à l'ENST, et M. Pierre Siohan, chercheur à Orange Labs à Rennes, d'avoir accepté d'être les examinateurs de cette thèse.

Je remercie particulièrement mes stagiaires durant ces trois années de thèse. Julien Reinauld, Riad Kara-Falah ainsi que Diarga Fall. Cette thèse ne serait jamais allée aussi loin sans leur aide et leur implication.

Je remercie aussi mes collègues d'Orange Labs à Rennes pour leur support scientifique et le temps consacré à répondre à mes questions.

A mes collègues d'Orange Labs à Meylan, plus particulièrement l'équipe des thésards, toute ma sympathie pour l'ambiance sympathique de travail.

Une pensée particulière à ma famille, encore une fois à mes parents, à mon oncle (khalo !) pour ses conseils toujours des plus sages ! A ma sœur Assia, mon frère Riad, et mes cousins, Amine et Mounir, pour leur encouragement sans fin !

Finalement, j'adresse mes remerciements des plus sincères à tous mes amis, qui m'ont côtoyé quotidiennement ici en France, que ce soit en période de stress intense ou de grande fête ! A Nico, Yoan & Gladys, Abdul, Matt, Denis & Raïssa, Tarik, Christelle.

A mes amis outre atlantique et de l'autre coté de la méditerranée, pour leurs messages, mails et coups de téléphone de motivation et d'encouragement malgré la distance qui nous sépare. Pensée spéciale à Lydia, Riad et Zoubida.

Une pensée particulière à tous ceux qui de près ou de loin m'ont donné un jour ce petit rayon de soleil synonyme d'amour et de lumière par sa chaleur et son éclat.

Résumé

La majorité des réseaux locaux sans fil actuels et les futurs réseaux mobiles de 3G avancée et 4G font appel à de la transmission multi-porteuse OFDM, reposant elle-même sur un traitement numérique par transformée de Fourier rapide (TFR). Ces systèmes devraient couvrir des bandes passantes de l'ordre de quelques dizaines voire centaines de MHz.

Cette thèse a pour but d'étudier les architectures de circuits intégrés pour le traitement numérique de l'OFDM avancé, très haut débit et multi-standard. Ces architectures visent à développer à la fois des puissances de calculs plus élevées pour répondre aux exigences de débit, ainsi que des capacités de reconfiguration pour des applications multi-standard. De plus, ces architectures doivent respecter une contrainte de consommation réduite du fait de l'environnement embarqué des terminaux mobiles.

En termes de solutions avancées, nous considérons deux schémas différents de la modulation OFDM, l'OFDM/QAM et l'OFDM/OQAM. Ce dernier nécessite en pratique un filtre polyphase de mise en forme, qui se concrétise dans notre étude par la fonction IOTA. Nous considérons aussi les dispositifs de mise en œuvre de type SISO et MIMO.

Une analyse comparative des différentes algorithmes et architectures de la TFR a permis de déterminer la meilleure approche nous permettant d'obtenir une bonne adéquation algorithme architecture. Cette solution intègre aussi le filtrage de mise en forme par la fonction IOTA. Ainsi, nous proposons une architecture à base de mémoires utilisant un multiplexage temporel des opérations sur une matrice de calcul à gros grain optimisée pour le traitement de la TFR et le filtrage de mise en forme. Cette approche temporelle permet une réalisation de modulation OFDM avancée pour des valeurs du paramètre N, le nombre de sous-porteuses de la modulation, allant de 64 à 8192 et du paramètre L, la longueur de troncature pour le filtrage, égal à 2, 4 et 8. L'architecture de la matrice applique le même traitement sur deux ou quatre flux d'échantillons différents, pour les modes MIMO 2x2 et 4x4 respectivement. Nous proposons aussi une stratégie pour la gestion des mémoires. Elle consiste en la mise en œuvre de bancs de mémoires permettant d'obtenir des tailles variables et de désactiver les mémoires non nécessaires. Le calcul en virgule fixe induit sur les données des effets inévitables de troncature et d'arrondi qui se traduisent par du bruit de calcul. Ainsi, nous proposons une étude de quantification des données de l'architecture proposée.

Afin de valider le bon fonctionnement et la faisabilité de l'architecture, nous proposons un prototypage sur FPGA et une synthèse physique d'un circuit ASIC (*layout*) en technologie CMOS submicronique de 65 nm. Les performances obtenues permettent d'envisager l'utilisation de cette architecture pour couvrir les différentes normes à base de la modulation OFDM.

Mots clés : Modulation OFDM, QAM, OQAM, IOTA, MIMO, TFR, filtre polyphase, architecture de circuit intégré, haut-débit, reconfiguration, basse consommation, FPGA, synthèse physique ASIC.

Abstract

Title : Reconfigurable, high throughput and low power VLSI architecture for advanced OFDM digital processing

Most current wireless LANs and future Beyond 3G and 4G mobile networks involve the multi-carrier OFDM transmission, based itself on the digital processing of the fast Fourier transform. These systems should cover bandwidths in the order of several tens or even hundreds of MHz.

The aim of this thesis was to study the architectures of integrated circuit for a high speed and multi-standard OFDM digital processing. These architectures require both higher speed processing to meet the required throughput, and reconfiguration for multi-standard applications. Moreover, these architectures should meet the requirement of reduced power consumption due to the embedded environment of mobile terminals.

In terms of advanced solutions, one considers two different OFDM modulation patterns, the OFDM/ QAM and OFDM/OQAM. This latter requires a pulse shaping polyphase filter implemented in our study on the IOTA prototype function. One considers also SISO/MIMO functionalities.

A comparative analysis of various FFT algorithms and architectures has led to identify the best approach which gives a good algorithm architecture adequation. This solution also incorporates the pulse shaping filter, more precisely implementing the IOTA function. One has therefore proposed a memory-based architecture using a time multiplexed operations on a coarse grained matrix optimized for the treatment of the FFT and of the pulse shaping filtering. This time approach allows a realization of advanced OFDM modulation for values of the parameter N, the number of subcarrier, from 64 to 8192 and the parameter L, the truncation length for pulse shaping filter, equal to 2, 4 and 8. The architecture of the matrix applies the same treatment on two or four streams of different samples, for modes MIMO 2x2 and 4x4 respectively. A strategy to manage memories has also been proposed. It is based on a memory banks approach to obtain various memory sizes and to enable the turn off the unnecessary memories.

A first FPGA prototyping and an ASIC layout design have validated the functioning and the feasibility of the architecture. The FPGA prototyping platform used was the ML402 from Xilinx incorporating the FPGA XC4VSX35 from the Virtex-4 family. The ASIC layout design has been done using the submicronic 65 nm CMOS technology from STMicroelectronics. The performances obtained out of this architecture makes it a good candidate to cover the different standards based on OFDM modulation.

Key words : OFDM modulation, QAM, OQAM, IOTA, MIMO, FFT, polyphase filter, VLSI architecture, high-throughput, reconfiguration, low power, FPGA, ASIC layout.

Table des matières

Table des figures

Liste des tableaux

Introduction

Contexte de la thèse

Le monde des télécommunications a connu des avancées spectaculaires ces dernières décennies. Le développement de l'informatique a permis la numérisation de l'information, permettant la transmission des données de différentes natures, voix, images, vidéo, internet, etc. La pénétration des circuits intégrés, toujours plus performants, dans les techniques de télécommunications a rendu possible la réalisation de modulations de plus en plus complexes. L'apparition de la téléphonie mobile commerciale et plus récemment des réseaux d'accès sans fil large bande tel que le WIFI (*wireless Fidelity*) ou le Wimax(*Worldwide interoperability for microwave access*) [1–3], a accru l'intérêt pour des nouvelles techniques de transmission de l'information efficaces sur des canaux hertziens.

La transmission de l'information par les ondes électromagnétiques dans l'air a le désavantage de subir l'influence du milieu de propagation. Les obstacles tels que les immeubles, les voitures, et bien sûr la topographie du milieu absorbent et réfléchissent une certaine proportion des ondes radio. La canal de propagation radio est alors caractérisé par trois phénomènes physiques qui perturbent la nature du signal d'information, à savoir l'atténuation du signal avec la distance parcourue, l'effet de masque (*shadowing*), et l'évanouissement multi-trajets (*fading*). De plus, dans le cas d'une mobilité des usagers, des perturbations peuvent dégrader les performances du lien radio suite à l'effet Doppler. Ces distortions du signal dans le canal de propagation radio mobile doivent alors être compensées par diverses techniques, dont des modulations plus avancées, dans le but de récupérer efficacement l'information.

Une des modulations les plus utilisées dans les systèmes de communication sans fil et qui fait l'objet de cette étude, est le multiplexage par répartition en fréquences orthogonales (*Orthogonal Frequency Division Multiplexing*, OFDM) [4], aussi appelée DMT (*Discrete Multi Tone*) dans le cas des communications filaires [5]. L'OFDM, système de modulation multi-porteuses, est très répandu, que ce soit dans la diffusion numérique terrestre (DVB-T/H, *Digital Video Broadcasting - Terrestrial/Handheld*) [6–9], l'ADSL (*Asymmetric Digital Subscriber Line*) [10], le WIFI et récemment le Wimax pour l'accès large bande sur quelque dizaines de kilomètres.

La modulation OFDM couplée avec la technologie multi-antenne MIMO (*Multiple-Input Multiple-Output*) [11–13] a aussi été retenue pour la prochaine évolution des systèmes de communications mobile de $3^{ème}$ génération [14–18]. Connue sous le nom de 3GPP-LTE (*Third Generation Partnership Project - Long Term Evolution*), cette nouvelle évolution des systèmes UMTS (*Universal Mobile Telecommunications System*) vise à accroître l'efficacité spectrale et la capacité radio, réduire la latence et les frais d'exploitation des opérateurs et, à terme, fournir aux utilisateurs finaux de nouveaux services haut débit mobiles à haute performance. Elle a pour

objectif d'atteindre des débits de données supérieurs à 100 Mbits/s en liaison descendante et 50 Mbits/s en liaison ascendante.

Le système de communication utilisant la modulation OFDM et caractérisé par le plus grand débit est la technologie Ultra Large Bande (UWB, *Ultra WideBand*) du consortium Wimedia [19–23], aussi identifiée sous le nom MB-OFDM (*Multiband*-OFDM). Elle nécessite un débit de 480 Mbits/s pour les communications machine à machine à très courte distance (*Universal Serial Bus* (USB) sans fil).

La synthèse du signal OFDM est typiquement réalisée en utilisant la transformée de Fourier inverse dans laquelle les symboles modulants sont des symboles complexes QAM (*Quadrature amplitude modulation*) qui modulent chaque porteuse. Les normes de communications actuelles à base de la modulation OFDM utilisent une fonction de mise en forme du signal émis de type rectangulaire afin de limiter la durée d'un symbole OFDM dans le temps. Elles sont aussi caractérisées par l'insertion d'un intervalle de garde ou préfixe cyclique pour chaque symbole OFDM pour s'affranchir des interférences inter-symboles. Ce schéma de modulation est connu sous le nom de OFDM/QAM ou encore CP-OFDM (*Cycle Prefix*-OFDM) [24–27].

Un schéma particulier de la modulation OFDM, l'OFDM/OQAM (pour *Offset QAM*) présente des caractéristiques intéressantes à exploiter dans le cadre de communications radio-mobiles [28]. Ainsi, associée à une mise en forme du signal de type IOTA (*Isotropic Orthogonal Transform Algorithm*) [29], l'OFDM/OQAM présente de meilleures caractéristiques de localisation en temps et en fréquence que l'OFDM/QAM. En outre, cette version ne nécessite pas d'insertion d'intervalle de garde offrant ainsi une meilleur efficacité spectrale.

On assiste aussi à une convergence des différents réseaux de communications mobiles et sans fil large bande [30, 31]. L'utilisateur a alors la possibilité de se connecter à des réseaux d'accès différents avec le même terminal. Ceci impose une nécessité de reconfiguration des ressources matérielles afin de réaliser les différents schémas de modulation à base de la technique OFDM. En effet, les paramètres du schéma de modulation OFDM, tels que le nombre de porteuses modulées ou encore le débit de transmission, sont propres à chaque standard de communication.

Objectif de la thèse

Cette thèse a pour but d'étudier les architectures de circuits intégrés pour le traitement numérique de l'OFDM avancé, très haut débit et multi-standard. Ces architectures nécessitent à la fois des puissances de calculs plus élevées pour répondre au débit exigé, ainsi que des capacités de reconfiguration pour des applications multi-standard. Elles doivent pouvoir réaliser plusieurs modulations en parallèle pour les systèmes MIMO. De plus, ces architectures doivent être caractérisées par une consommation réduite du fait de l'environnement embarqué des terminaux mobiles.

En règle générale, les architectures VLSI (*Very-Large-Scale Integration*) des modulateurs OFDM sont dédiées à une application spécifique. Ceci permet d'optimiser les ressources (mémoires, opérateurs arithmétiques, puissance, etc.) disponibles dans le circuit.

Dans une perspective de développement à long terme, et d'anticipation de futures normes à très large bande, les recherches menées tenteront de déterminer la bande passante que pourrait couvrir un modulateur OFDM avancé, réalisant soit le schéma OFDM/QAM ou OQAM, dans un contexte multi-standard, très haut débit et basse consommation. On cherchera à déterminer les limites technologiques pour sa réalisation et les solutions envisageables.

Contribution de la thèse

La transformée de Fourier et le filtrage par la fonction prototype sont les principales étapes de la modulation OFDM avancée. Il existe plusieurs algorithmes réalisant la transformée de Fourier. Ces algorithmes de transformée de Fourier rapide (TFR) sont une succession d'opérations arithmétiques structurées, chacun utilisant des approches différentes et possédant des avantages et des inconvénients. Ils n'offrent pas tous les mêmes performances. La première contribution a consisté à répertorier un ensemble fini d'algorithmes, à déterminer leurs performances, et à réaliser un choix d'algorithme pour la TFR.

Chaque algorithme peut être implémenté de façons différentes selon le type d'architecture utilisée. Ces algorithmes donnent des performances différentes pour chaque type d'implémentation. Ainsi, un algorithme peut être très performant arithmétiquement parlant, et offrir des performances inférieures en termes de débit et de consommation lorsqu'il est réalisé suivant une architecture donnée. Une bonne adéquation algorithme-architecture permet de tirer parti des performances arithmétiques de l'algorithme et des performances en débit et en consommation de l'architecture. La deuxième contribution a été de déterminer quels types d'architectures sont les plus adéquates pour notre classe d'algorithme en tenant compte des fortes contraintes de débit, de reconfiguration et de consommation.

Le filtrage de mise en forme par la fonction prototype IOTA est un algorithme assez complexe et a déjà fait l'objet d'une optimisation algorithmique et architecturale dans le cadre de la thèse de doctorat de Mr. Jalali [32]. Toutefois, les contraintes de haut débit et de basse consommation influencent encore une fois cette architecture. Même s'il s'agit de deux processus différents et successifs, une conception optimisée ne peut se faire en considérant la transformée de Fourier et le filtrage séparément. Une étude globale du modulateur est à l'origine de la troisième et principale contribution qui a permis l'émergence d'une architecture prometteuse et innovante offrant un bon compromis entre flexibilité, consommation et haut débit.

Cette architecture intègre plusieurs solutions proposées pour répondre au cahier de charge du modulateur OFDM avancé. Ainsi, on propose une solution intégrant le système MIMO et permettant à l'architecture d'effectuer jusqu'à quatre modulations OFDM/QAM ou OFDM/OQAM en parallèle. L'architecture comprend une matrice de calcul à gros grain flexible permettant d'exécuter soit l'algorithme de la TFR ou celui du filtrage. Finalement, une organisation mémoires basse consommation, scalable selon les paramètres de la modulation, est proposée pour les mémoires des échantillons, des coefficients de la TFR et ceux de la fonction IOTA.

Plan et organisation du manuscrit

Le reste du manuscrit de la thèse est divisé en trois parties retraçant de façon logique les travaux menés pendant la thèse. Nous commençons, dans la première partie, par présenter l'état de l'art sur les architectures OFDM avancées. Elle est composée de trois chapitres. Le premier chapitre introduit la problématique traitée dans le cadre de cette thèse. Le deuxième chapitre présente les algorithmes et les architectures conventionnelles utilisés dans un modulateur OFDM avancé. Ainsi, nous analysons les différentes algorithmes de la TFR ainsi que les différentes architectures pour les mettre en oeuvre. Nous avons fait de même pour le filtrage de mise en forme. Le troisième chapitre présente la meilleure approche de conception offrant un bon compromis entre capacité de traitement, surface et puissance consommée pour un modulateur OFDM avancé multi-standard et haut débit. Les conclusions tirées de ce chapitre nous ont amenés à l'architecture proposée.

La deuxième partie présentant la conception de l'architecture proposée est divisée en cinq chapitres. Le premier chapitre constituant cette partie présente l'approche système de l'architecture pour une bonne compréhension de son fonctionnement. Les trois chapitres suivants présentent les différentes facettes constituant cette architecture, à savoir la matrice de calcul, la stratégie mémoire et le contrôle de l'architecture. Enfin, le dernier chapitre de la deuxième partie présente l'étude de quantification des données de l'architecture proposée.

La dernière partie présente les résultats d'implémentation et d'expérimentation de l'architecture proposée. Cette partie est composée de deux chapitres, le premier concerne le prototypage FPGA (*Field Programmable Gate Array*) et le deuxième la synthèse physique (*layout*) du circuit ASIC (*Application Specific Integrated Circuit*). Ces travaux ont été réalisés entre autres par trois stagiaires [33–35].

Pour finir, nous concluons le présent document par une discussion sur l'architecture proposée et par la proposition des perspectives.

Brevet, communications et publications

Les recherches menées ont abouti à un dépôt de brevet international numéro FR2008/050703 (du 18.04.2008) de l'architecture proposée par France Telecom R&D.

Cette thèse a aussi fait l'objet des trois conférences internationales suivantes :

- C. Sahnine, N. Zergainoh, D. Callonnec, F. Pétrot, "*Efficient Design Approach and Advanced Architectures for Universal OFDM Systems*", PhD research in Microelectronics and Electronics conference (PRIME 2007), Conference proceedings, p 33-36 Bordeaux, France, 2-5 Juillet, 2007.
- C. Sahnine, N. Zergainoh, D. Callonnec, F. Pétrot, "*Towards a High-Throughput and Low Power Reconfigurable Architecture of Advanced OFDM Modulator for Software-Defined Radio Systems*", Midwest Symposium on Circuits and Systems (MWSCAS 2007), Conference proceedings, 50th IEEE International Midwest Symposium on, p 1205-1208 Montréal, Canada, 5-8 Août, 2007.
- C. Sahnine, J.-P. Javaudin, G. Degoulet, B. Jahan, "*OFDM/OQAM Transceiver Implementation*", Design and Architectures for Signal and Image Processing (DASIP 2007),

Grenoble, France, 27-29 Novembre, 2007.

Deux articles de revues ont été soumis et sont en attente de réponse :

- C. Sahnine, N. Zergainoh, D. Callonnec, F. Pétrot, "*Configurable VLSI Architecture for Multi-Mode Advanced OFDM/OQAM Modulation*", soumis à IEEE Transactions on Very Large Scale Integration (VLSI) Systems.
- C. Sahnine, N. Zergainoh, D. Callonnec, F. Pétrot, "*Transformées de Fourier Rapides pour les modulations OFDM : État de l'art et avancées FFT for OFDM modulation : State of the Art and Beyond*", soumis à Technique et Science Informatiques (TSI) .

Première partie

État de l'art sur les architectures OFDM avancées

Chapitre 1

Problématique

Les systèmes et les normes de communications à base de la modulation OFDM sont de plus en plus nombreux. La figure 1.1 ci-dessous illustre l'augmentation de la mobilité des utilisateurs versus la montée en débit pour plusieurs systèmes de communications à base de la modulation OFDM.

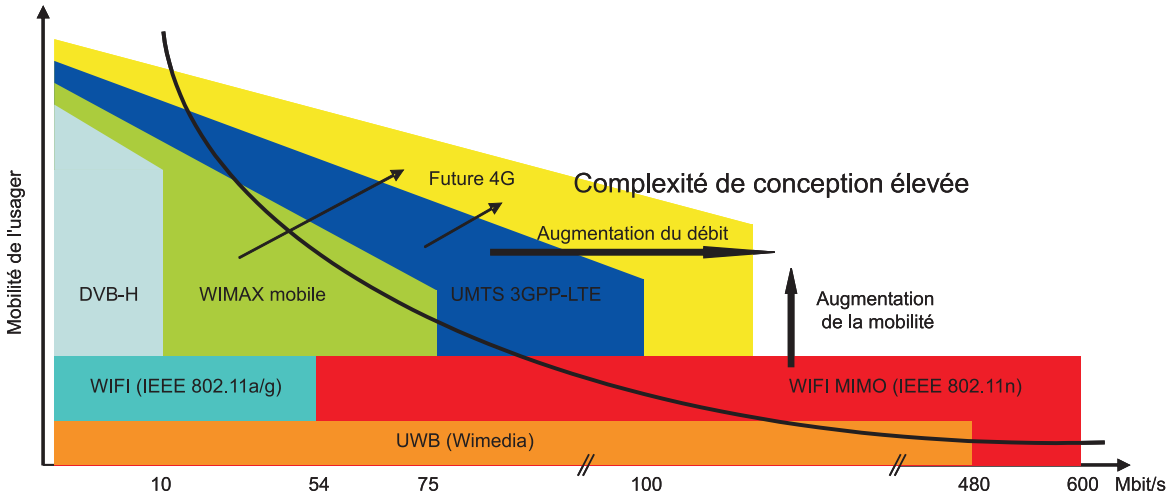

Figure 1.1 – Débit versus mobilité : zone d'opération de différents systèmes de communication OFDM sans fils

La croissance de ces deux paramètres, débit et mobilité, crée des obstacles à la réalisation d'interfaces radios optimisées. La complexité des algorithmes utilisés dans le traitement du signal augmente avec l'augmentation des contraintes. Dans le but de réaliser ces algorithmes, il est nécessaire de concevoir des circuits toujours plus complexes et toujours plus performants. La figure1.2 illustre l'écart entre la complexité des algorithmes et les performances actuelles des processeurs génériques (GPP : *General-Purpose Processor*) et des processeurs dédiés au traitement du signal (DSP : *Digital Signal Processor*). Pour combler cet écart, il est nécessaire d'utiliser des circuits très hautes performances, dédiés à la réalisation de ces algorithmes complexes.

On note sur la figure 1.2, que les prochaines générations de systèmes de communications mobiles, 3G et 4G, ainsi que la radio logicielle (SDR : *Software Defined Radio*) se situent dans cette zone de haute complexité algorithmique. Le concept de radio logicielle est de concevoir des radios flexibles, capables de s'adapter aux différents standards en téléchargeant un simple logiciel [37–41]. Pour atteindre cet objectif, les architectures des futurs terminaux mobiles de-

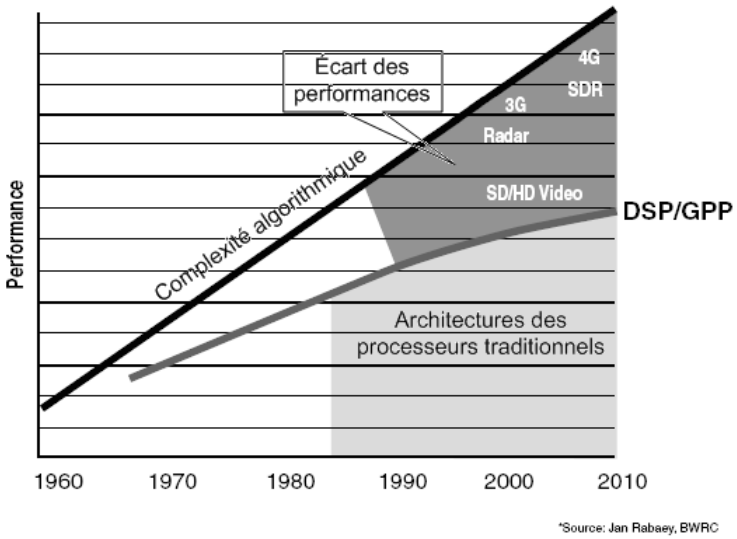

FIGURE 1.2 – Prévision de la complexité algorithmique et des performances des processeurs [36]

vront être reconfigurables, flexibles, programmables, à très forte capacité de calcul, très basse consommation, et à faible coût.

Le tableau 1.1 présente les différentes normes OFDM visées. Dans notre étude, nous considérons particulièrement la version avancée de la modulation OFDM, l'OFDM/OQAM. Cette version n'est pas encore utilisée commercialement mais elle est candidate aux futurs normes de communications mobiles et large bande. Le tableau 1.1 extrapole les paramètres système hypothétiques dans le cas de l'OFDM/OQAM. Nous verrons dans le chapitre suivant que le schéma OFDM/OQAM nécessite une génération d'un symbole OFDM en un laps de temps deux fois plus court que l'OFDM/QAM pour obtenir le même débit symbole. Ce schéma de modulation induit aussi une complexité accrue due au filtrage supplémentaire par la fonction IOTA.

En résumé, notre modulateur doit être capable de réaliser une TFR de 64 à 8192 points avec la contrainte de l'UWB qui est de réaliser une TFR à 128 points en 242 ns pour la modulation OFDM/QAM. Dans le cas de l'OFDM/OQAM, le traitement de la TFR de 128 points et le filtrage de mise en forme doivent être réalisés en 121 ns. Il doit pourvoir aussi réaliser plusieurs modulations en parallèle pour la technique MIMO.

Toutefois, notre objectif est beaucoup plus général. Même si nous considérons ces normes comme références pour notre modulateur, le principal objectif reste de concevoir un modulateur OFDM avancé basse consommation pour n'importe quel nombre de porteuses puissance de 2 et pour la plus grande largeur de bande possible. Il s'agit d'une anticipation de futures normes de communication à base de la technique de transmission OFDM dans le cadre de la radio logicielle.

Une conception basse consommation peut se faire à toutes les couches de conception d'un système intégré sur puce (SOC : *System On Chip*). La figure 1.3 illustre les différentes couches de conception d'un SOC [42]. Des efforts d'optimisation peuvent être entrepris à chacune de ces couches pour obtenir des gains en consommation plus importants.

Au niveau systèmes, des gains peuvent être obtenus en réalisant un meilleur partitionnement matériel/logiciel, en mettant hors tension certaines parties du circuit, en utilisant du *clock gating*, etc. Au niveau algorithmes, il est possible d'exploiter la régularité de l'algorithme, tirer profit

TABLE 1.1 – Paramètres système pour plusieurs normes OFDM/QAM et **extrapolation hypothétique** pour l'OFDM/OQAM, pour un même débit symbole

Norme	Largeur de bande (MHz)	N	Espace inter-porteuses (KHz)	Période d'un symbole τ_0* (μs)		MIMO
				OFDM		
				QAM	OQAM **Extrapolation hypothétique**	
3GPP-LTE	1,25	128	15	102,4	51,2	Oui
	2,5	256				
	5	512				
	10	1024				
	20	2048				
WIFI	20	64	312,5	3,2	1,6	Oui
Wimax Mobile	1,25	128	10.94	91,4	45,7	Oui
	5	512				
	10	1024				
	20	2048				
DVB-H	8	2048	4,464	224	112	Non
		4096	2,232	448	224	
		8192	1,116	896	448	
UWB MB-OFDM	528	128	4125	0,242	0,121	Non

*Le temps symbole est calculé sans l'intervalle de garde pour la modulation OFDM/QAM

de la localité des données, utiliser des représentations des données plus efficaces, etc. Au niveau architectures, on peut augmenter les opérations concurrentes grâce au parallélisme et au pipeline, diminuer la fréquence d'exécution et la tension d'alimentation.

Les deux derniers niveaux font plus références aux technologies de fabrications des transistors. Au niveau circuit, on peut jouer sur la taille des transistors, utiliser des portes logiques plus performantes, etc. Au niveau technologie, on peut faire varier la tension seuil V_{th} ou encore utiliser des technologies SOI (*Silicon-On-Insulator* ou silicium sur isolant).

Notre domaine de conception se situe au niveau algorithmes et architectures. Comme on le constate dans la figure 1.3, des optimisations importantes peuvent être obtenues.

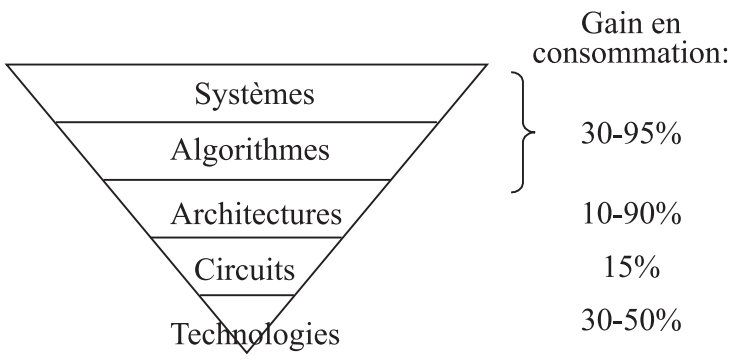

FIGURE 1.3 – Gain en consommation à différentes couches de conception d'un SOC [42]

Les objectifs de haut débit, de basse consommation et de reconfigurabilité imposent des restrictions à la fois contraignantes et opposées. En effet, l'objectif de haut débit aura tendance à augmenter les ressources arithmétiques et donc la puissance consommée. Il en est de même pour la reconfigurabilité qui engendre des ressources supplémentaires. La problématique de cette thèse est donc de trouver une certaine cohérence des solutions résolvant chacune de ces contraintes. Ainsi, les questions et les obstacles que nous tenterons de lever dans cette étude se présentent comme suit :

Quels types d'algorithmes peuvent être utilisés pour diminuer la complexité de la transformée de Fourier, le cœur de la modulation OFDM ? Quel est le degré de régularité de ces algorithmes ? Cette régularité peut-elle avantager une reconfiguration pour de nouveaux paramètres de la TFR ?

Quels types d'architectures peuvent satisfaire les contraintes de haut débit et de reconfigurabilité ? Quel est le degré d'adéquation entres ces architectures et les algorithmes utilisés ? Ces mêmes architectures permettent-telles de limiter la consommation et la taille du circuit ? Le cas échéant, quelle innovation est nécessaire pour résoudre ces problèmes, d'autant plus que la modulation OFDM avancée nécessite en plus de la TFR, un filtrage supplémentaire ? Quel est l'impact de la reconfiguration ?

Chapitre 2

Algorithmes et architectures d'un modulateur OFDM avancé

La modulation OFDM revient à répartir les informations à transmettre sur un grand nombre de porteuses orthogonales en parallèle, individuellement modulées à bas débit, grâce à une transformée de Fourier inverse [4, 43–47]. L'expression du signal OFDM transmis est :

$$s(t)_{OFDM} = \sum_{n=-\infty}^{+\infty} \sum_{m=0}^{2M-1} a_{n,m} e^{j\theta_{n,m}} g\left(t - n\tau_0\right) e^{j2\pi m v_0 t} \tag{2.1}$$

Où $a_{n,m}$ représente l'information envoyée sur la $m^{ème}$ porteuse du $n^{ème}$ symbole, $2M$ $(= N)$ est le nombre des sous-porteuses, υ_0 l'espace inter-porteuse et τ_0 la durée d'un symbole OFDM. Le terme $g\left(t - n\tau_0\right)$ représente la fonction prototype de mise en forme temporelle du signal décalée dans le temps tous les $n\tau_0$.

Dans notre étude, nous considérons deux schémas de modulations particuliers, l'OFDM/QAM et l'OFDM/OQAM. Dans la prochaine section 2.1, nous présentons la structure d'un modulateur pour les deux schémas de modulation. Les deux sections suivantes, 2.2 et 2.3, présentent respectivement les algorithmes et les architectures de TFR. Finalement, la dernière section 2.4 présente l'algorithme de mise en forme IOTA pour le schéma OFDM/OQAM ainsi que l'architecture de mise en œuvre.

2.1 Structure d'un modulateur OFDM avancé

La figure 2.1 illustre une chaine de communication simplifiée d'un système sans fil en émission-réception. À l'émission, les données sont codées par le bloc *Codage source* afin d'enlever la redondance d'information et ainsi diminuer la quantitié de données à transmettre. Le bloc *Couche MAC* ou couche de liaison de données (*medium access control*) organise les données en trames et permet entre autre de partager les ressources radios entre les différents utilisateurs dans le cas des réseaux mobiles et locaux sans fil. Par la suite, une stratégie de codage est appliquée par le bloc *Codage canal* dans le but de rendre plus robuste la transmission à travers le canal. Le bloc *modulateur* permet de représenter l'information binaire à transmettre sous une forme physique. Dans le cas de la modulation OFDM, le train binaire à l'entrée module les différentes porteuses du signal à transmettre. Le bloc *CNA* convertit le signal numérique en un signal analogique. Finalement, le bloc *Front-End* amplifie le signal et le transpose à la bonne fréquence d'émission. Le récepteur reprend le chemin inverse. Cette thèse se concentre sur la bloc *modulateur*.

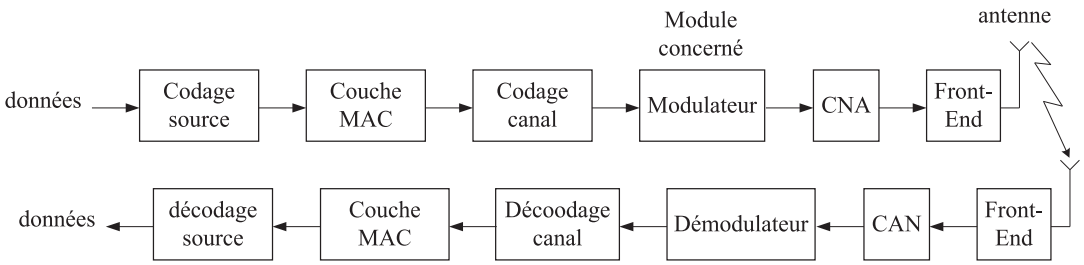

FIGURE 2.1 – Chaîne de transmission/réception

2.1.1 Schéma OFDM/QAM

Pour ce schéma de modulation [4, 43, 48, 49], le déphasage avant modulation par les informations $a_{n,m}$ de chaque sous-porteuse par rapport aux sous-porteuses adjacentes en temps et en fréquence est nul. Ainsi, on a $\theta_{n,m} = 0$ pour l'équation 2.1. On obtient alors pour la modulation OFDM/QAM :

$$s(t)_{OFDM/QAM} = \sum_{n=-\infty}^{+\infty} \sum_{m=0}^{2M-1} a_{n,m} g\left(t - n\tau_{0_{OFDM/QAM}}\right) e^{j2\pi m \upsilon_0 t} \tag{2.2}$$

Où $a_{n,m}$ est complexe et l'espace inter-porteuse υ_0 est $\frac{1}{\tau_{0_{OFDM/QAM}}}$. $\tau_{0_{OFDM/QAM}}$ est la durée d'un symbole OFDM/QAM.

La fonction prototype de mise en forme $g(t)$ correspond à la fonction rectangle et de ce fait aucun filtrage n'est nécessaire. Cette fonction est caractérisée par une orthogonalité complexe. Les porteuses sont par conséquent modulées par des données complexes obtenues après modulation en quadrature QAM (en général les constellations utilisées vont du BPSK au 64-QAM). De plus, en présence d'un canal multi-trajet, des versions décalées du symbole n s'additionnent entre elles, créant de l'interférence inter-symbole (ISI : *Inter-Symbole Interference*) entre les symboles OFDM/QAM. Pour absorber ce retard, on insère un préfixe cyclique ou intervalle de garde. Le débit utile est réduit dans les mêmes proportions.

La figure 2.2 illustre les étapes d'une modulation OFDM/QAM. Les données à l'entrée subissent une modulation QAM. Les symboles QAM qui en sont issus sont mappés sur un bloc de N points pour réaliser une TFR à N points. L'intervalle de garde est inséré par la suite.

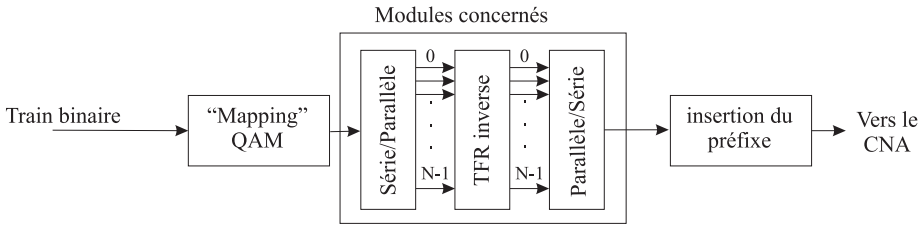

FIGURE 2.2 – Schéma classique du modulateur OFDM/QAM

La figure 2.3 illustre le principe de la modulation OFDM avec un schéma QAM. On y trouve la représentation temporelle avec l'intervalle de garde ainsi que son équivalence dans le domaine fréquentiel, soit la superposition de N fonctions sinus cardinal espacées de façon orthogonale tous

les υ_0. Lors de la démodulation d'une des sous-porteuses, la condition d'orthogonalité nous permet de récupérer toute l'information de la sous-bande sans interférer avec les données adjacentes. Toutefois, la modulation OFDM/QAM est très sensible à la désynchronisation de fréquence du démodulateur par rapport à celle du modulateur ce qui a pour conséquence qu'une partie de l'énergie de la bande adjacente est récupérée. Le spectre fréquentiel de la fonction prototype utilisée (sinus cardinal), a des lobes secondaires relativement larges, et une mauvaise localisation fréquentielle due à la décroissance lente de la fonction. Ceci entraîne donc, en cas de désynchronisation, des interférences entre porteuses (ICI : *Inter-Carrier Interference*). Afin d'éliminer les différentes sources d'interférences, ISI et ICI, le schéma OFDM/OQAM a été proposé.

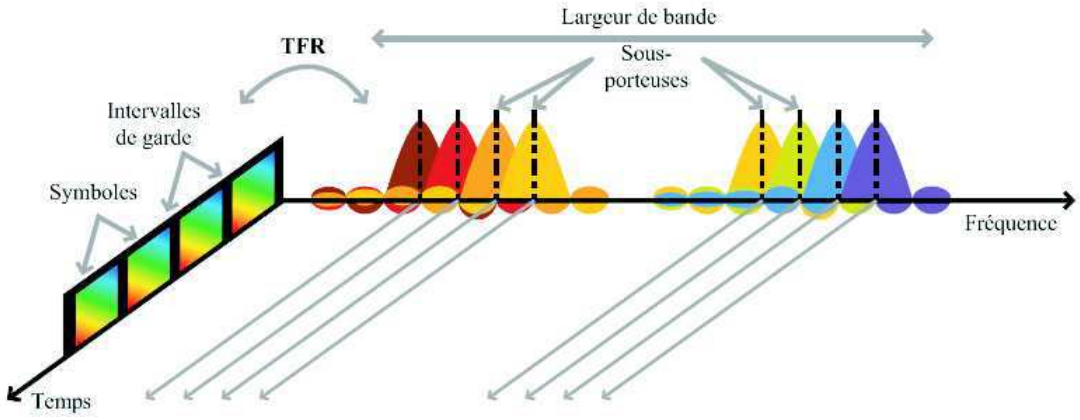

FIGURE 2.3 – Principe de la modulation OFDM, plan temps-fréquence du schéma OFDM/QAM [50]

2.1.2 Schéma OFDM/OQAM

Le schéma OFDM/OQAM utilise une fonction de mise en forme possédant une meilleure localisation en temps et en fréquence. Toutefois, cette bonne localisation à la fois en temps et en fréquence ne peut être obtenue que pour des fonctions possédant une orthogonalité réelle [28,29,32,51–59]. De ce fait, les symboles complexes $a_{n,m}$ de la version QAM doivent être séparés en deux symboles réels, un pour la partie réelle et l'autre pour la partie imaginaire. Chacune de ces deux données réelles module une porteuse à des instants séparés par la moitié de la durée symbole OFDM/QAM ($\frac{\tau_{0_{QAM}}}{2} = \tau_{0_{OFDM/QAM}}$), d'où l'origine du *offset*. Par conséquent, pour obtenir le même débit symbole utile, deux symboles OFDM/OQAM doivent être envoyés au lieu d'un symbole OFDM/QAM. Pour de plus amples explications sur les conditions d'orthogonalité du schéma OFDM/OQAM, le lecteur peut se référer à l'annexe A.

Dans la modulation OFDM/OQAM, les sous-porteuses sont espacées de $\upsilon_0 = \frac{1}{2\tau_{0_{OQAM}}}$ (réseau de densité 2). La figure 2.4, donne une comparaison du plan temps-fréquence du schéma OFDM/QAM et du schéma OFDM/OQAM. On constate que ce plan temps-fréquence est deux fois plus dense pour l'OFDM/OQAM que pour l'OFDM/QAM.

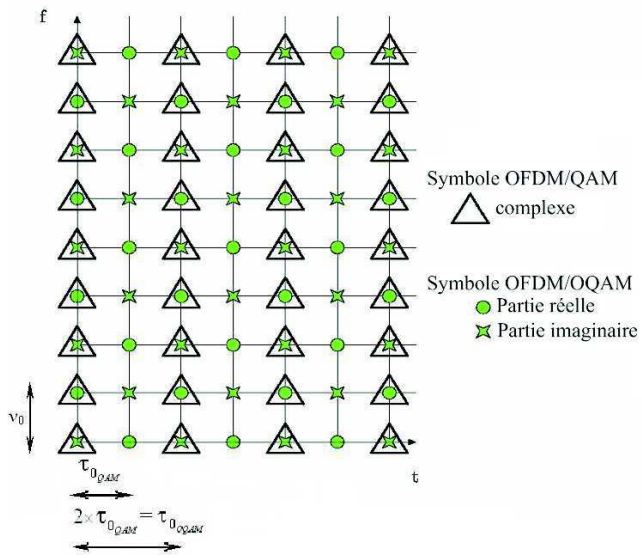

FIGURE 2.4 – Plan temps-fréquence du schéma OFDM/QAM et OFDM/OQAM [60]

Dans ce schéma OQAM, $\theta_{n,m} = (n+m)\frac{\pi}{2}$, les différentes porteuses adjacentes sont donc déphasées de $\frac{\pi}{2}$ en temps et en fréquence. L'équation 2.1 devient dans le cas de la modulation OFDM/OQAM :

$$s(t)_{OFDM/OQAM} = \sum_{n=-\infty}^{+\infty} \sum_{m=0}^{2M-1} a_{n,m} i^{n+m} g\left(t - n\tau_{0_{OFDM/OQAM}}\right) e^{j2\pi m\upsilon_0 t} \tag{2.3}$$

La fonction prototype $g(t)$ utilisée correspond à la fonction IOTA $\Im(t)$ [29,32,49,56,61–64]. Cette bonne localisation en temps et en fréquence de la fonction IOTA illustrée à la figure 2.5 nous permet de nous affranchir de l'intervalle de garde indispensable en OFDM/QAM. La figure 2.6 est une comparaison du spectre de la fonction IOTA avec le spectre de la fonction fenêtre rectangle. On remarque que l'atténuation de la fonction IOTA est plus prononcée que celle de la fonction rectangle, d'environ 20 dB de plus dès le premier lobe. De plus, la fonction IOTA a une atténuation plus continue et constante. Ainsi son influence sera beaucoup moindre sur les symboles OFDM adjacents.

La figure 2.7 illustre les étapes d'une modulation OFDM/OQAM. Après la modulation QAM, les données complexes sont séparées en deux données réelles. Une multiplication par i^{n+m} est réalisée par la suite. Ceci permet d'obtenir des échantillons complexes purement réels ou purement imaginaires assurant un déphasage de $\frac{\pi}{2}$ en temps et en fréquence de chaque porteuse garantissant ainsi une orthogonalité complète des sous porteuses. Finalement, une TFR est réalisée suivie par le filtrage polyphase de mise en forme par la fonction IOTA. On note sur la figure 2.7 que pour $N = 2M$ points à l'entrée de la TFR, seulement M points sont émis à la sortie du filtrage polyphase. Ainsi, comparativement au modulateur OFDM/QAM, le modulateur OFDM/OQAM doit opérer à une fréquence deux fois plus rapide après la décomposition des données complexes en deux données réelles pour avoir le même débit de transmission. Cette différence s'explique par la densité de 2 du plan temps-fréquence OFDM/OQAM.

Dans notre étude, nous ne considérons que les blocs encadrés aux figures 2.2 et 2.7, soit la TFR et le filtrage polyphase.

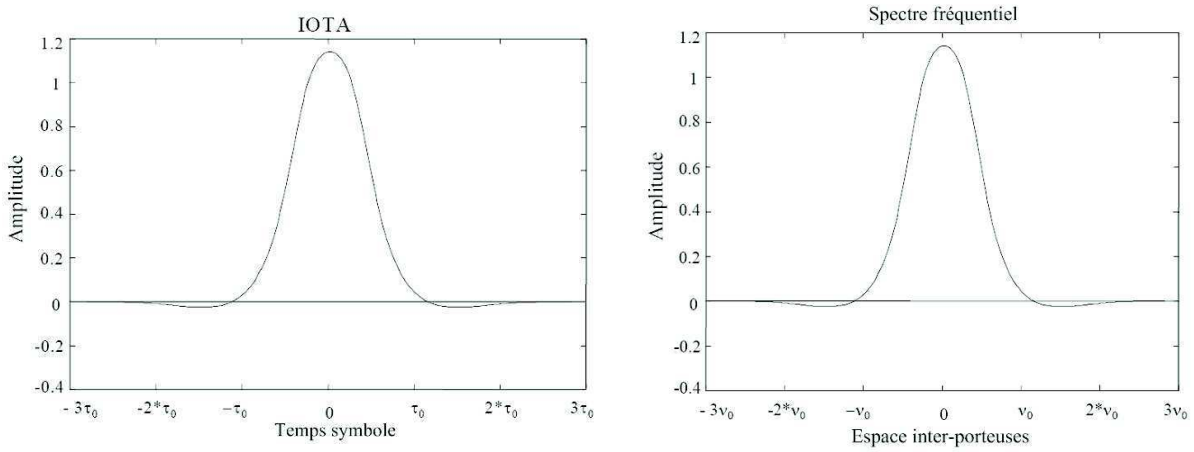

FIGURE 2.5 – Fonction IOTA et son spectre fréquentiel [61]

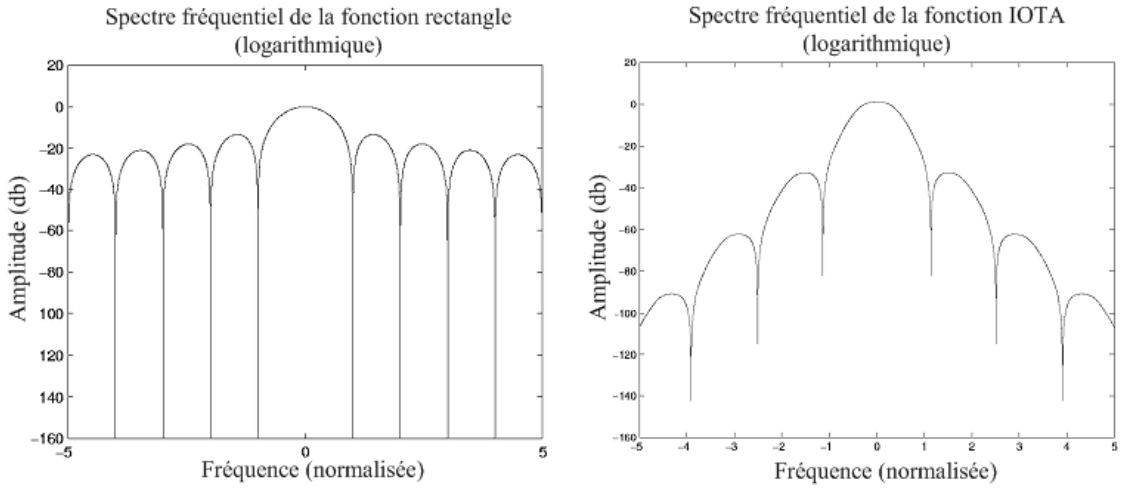

FIGURE 2.6 – Spectre fréquentiel logarithmique de la fonction rectangle et de la Fonction IOTA [60]

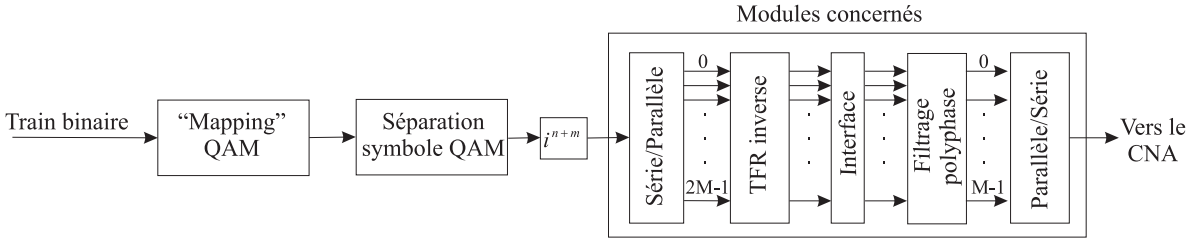

FIGURE 2.7 – Schéma classique du modulateur OFDM/OQAM

Il existe une alternative à la génération d'un signal OFDM/OQAM, autre que celle présentée jusqu'ici. En effet, il est possible d'utiliser des bancs de filtre en transmission, appelés aussi transmultiplexeur, afin d'effectuer un multiplexage des données en un seul signal [28, 65–67]. Cette approche agit sur la partie réelle, et le signal OFDM obtenu est réel. Elle permet de diminuer la complexité arithmétique en utilisant entre autre une TFR réelle au lieu d'une TFR complexe, ceci afin d'éliminer les opérations redondantes. Toutefois, cette approche n'a pas été retenue dans notre cas puisque nous cherchions aussi à réaliser une TFR complexe pour le schéma de modulation QAM.

2.2 Performances des algorithmes de la TFR

2.2.1 Brève introduction aux algorithmes de la TFR

La transformée de Fourier rapide est une façon très efficace de calculer la transformée de Fourier discrète (TFD). Le terme TFR ne désigne pas une nouvelle transformée, ni un algorithme, mais un ensemble d'algorithmes qui permettent d'accélérer le traitement de la TFD [68–77]. Soit l'équation de la TFD :

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \tag{2.4}$$

et son inverse :

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk} \tag{2.5}$$

avec les facteurs de rotations $W_N = e^{-j\frac{2\pi}{N}}$, $0 \leq n \leq N-1$, $0 \leq k \leq K-1$.

Les vecteurs $x(n)$, représentant les échantillons temporels, et $X(k)$, sa transformée dans le domaine fréquentiel, sont unidimensionnels, respectivement de taille $N \times 1$ et $K \times 1$ avec $K = N$. L'idée de l'approche *divide-and-conquer* à la base des algorithmes de la TFR, consiste à représenter ces vecteurs sur 2 dimensions (cas du radix-2, radix-4, radix-8, etc.) ou plus (cas du radix-2^{i} ou radix-4^{i}).

Pour chaque algorithme, il existe deux versions duales et de même complexité, à savoir l'entrelacement en fréquence (*Decimation-In-Frequency DIF*) et l'entrelacement en temps (*Decimation-In-Time DIT*). Dans la version DIF, les échantillons d'entrées $x(n)$ sont lus dans l'ordre naturel et les échantillons $X(k)$ sont obtenus dans l'ordre *bit reversed*. Dans la version DIT, c'est l'inverse. Dans cette étude, nous considérons la version DIF (l'étude peut facilement être étendue à la version DIT). La décomposition de la TFD pour l'obtention de différentes algorithmes de TFR est présentée à l'annexe B.

Pour une TFD à N points puissance de 2 (2, 4, 8, etc.), il existe au moins un facteur commun entre N_1 et N_2, avec $N = N_1 \times N_2 = r \times \frac{N}{r}$. Selon les valeurs de N_1 et N_2, on obtient plusieurs variantes appelées algorithmes *radix-r* (base-r). La valeur de r est le nombre de points de la plus petite TFR effectuée, appelée opérateur papillon (de l'anglais *Butterfly*), pour réaliser une TFR complète à N points. Ceci permet de diminuer la complexité arithmétique (nombre de multiplications/additions) de la TFD à N points de N^2 à $N \log_r N$.

La figure 2.8 illustre le schéma de base d'un papillon radix-r. Dans la suite de ce document, on identifie par croisillon la partie du papillon réalisant les additions complexes des modules

papillons radix-r présentés à l'annexe B. Le papillon englobe donc la partie croisillon et les multiplications complexes selon l'algorithme utilisé.

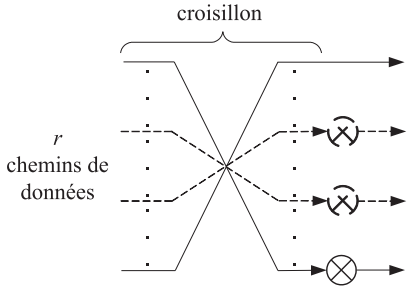

FIGURE 2.8 – Papillon radix-r

Un des algorithmes le plus utilisé est la TFR radix-2, qui est l'algorithme de Cooley et Tukey proprement dit [68]. Il transforme le calcul d'une TFD de N points en un certain nombre de calculs de TFD de 2 points. Cet algorithme est caractérisé par un flot de données régulier facilitant son intégration dans un circuit intégré dédié. Toutefois, du point vue complexité arithmétique, l'algorithme radix-2 n'est pas le plus performant. La section suivante présente une comparaison de la complexité arithmétique pour différents algorithmes.

2.2.2 Complexité arithmétique des algorithmes de TFR

Le nombre de multiplications/additions réelles nécessaires pour réaliser une TFR à N points dépend de la capacité de l'algorithme à générer des coefficients de rotation triviaux au lieu de valeurs complexes non triviales. Par coefficients triviaux on entend des facteurs ayant la valeur ± 1 ou $\pm j$, qui correspondent aux multiples de $W_N^{N/4}$. Dans ce cas, il s'agit simplement d'une permutation des valeurs réelles et imaginaires. Un gain supplémentaire en terme de multiplications triviales est obtenu en considérant les multiplications par les facteurs de rotation multiple impaire de $W_N^{N/8} = \frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2}$. Dans ce cas, la partie réelle et imaginaire du facteur de rotation sont les mêmes, nous permettant de diviser par deux le nombre de multiplications réelles nécessaires.

La complexité arithmétique est la même peu importe l'entrelacement utilisé, en temps ou en fréquence. En effet, dans les deux cas nous avons $\log_r N$ étages chacun ayant N/r modules papillons. Dans cette étude, nous considérons les multiplieurs complexes classiques constitués de 4 multiplications réelles et 2 additions réelles (qu'on désignera par 4M2A) et des multiplieurs basse consommation nécessitant seulement 3 multiplieurs réels mais au prix d'un additionneur réel supplémentaire (qu'on désignera par 3M3A). Les tableaux 2.1 et 2.2 illustrent les expressions mathématiques de la complexité arithmétique pour les différents algorithmes présentés et le type de multiplieur complexe utilisé. Les tableaux 2.3 et 2.4 donnent le nombre de multiplications/additions non triviales pour différentes valeurs de N.

L'algorithme split-radix offre les meilleurs performances en termes d'opérations arithmétiques, suivi par le radix-2^3 et le radix-2^2. Selon [78, 79], la complexité arithmétique des diverses versions de l'algorithme split-radix est la même (nombre d'additions réelles additionnées aux multiplications réelles) quelle que soit la combinaison radix-$2^r/2^{rs}$ utilisées. Ainsi, des al-

TABLE 2.1 – Complexité arithmétique des algorithmes pour un multiplieur 4M2A

Algorithmes	♯ de mult. réelles non triviales	♯ d'add. réelles non triviales
radix-2	$2N\log_2 N - 7N + 12$	$3N\log_2 N - 3N + 4$
radix-4 et 2^2	$\frac{3N}{2}\log_2 N - 5N + 8$	$\frac{11N}{4}\log_2 N - \frac{26N}{12} + \frac{8}{3}$
radix-8 et 2^3	$\frac{4N}{3}\log_2 N - \frac{59N}{14} + \frac{40}{7}$	$\frac{33N}{12}\log_2 N - \frac{57N}{28} + \frac{16}{7}$
split-radix	$\frac{4N}{3}\log_2 N - \frac{38N}{9} + \frac{2}{9}(-1)^{\log_2 N} + 6$	$\frac{8N}{3}\log_2 N - \frac{16N}{9} - \frac{2}{9}(-1)^{\log_2 N} + 2$

TABLE 2.2 – Complexité arithmétique des algorithmes pour un multiplieur 3M3A

Algorithmes	♯ de mult. réelles non triviales	♯ d'add. réelles non triviales
radix-2	$\frac{3N}{2}\log_2 N - 5N + 8$	$\frac{7N}{2}\log_2 N - 5N + 8$
radix-4 et 2^2	$\frac{9N}{8}\log_2 N - \frac{43N}{12} + \frac{16}{3}$	$\frac{25N}{8}\log_2 N - \frac{43N}{12} + \frac{16}{3}$
radix-8 et 2^3	$\frac{25N}{24}\log_2 N - \frac{350N}{112} + 4$	$\frac{146N}{48}\log_2 N - \frac{25N}{8} + 4$
split-radix	$N\log_2 N - 3N + 4$	$3N\log_2 N - 3N + 4$

TABLE 2.3 – Nombre de multiplications/additions non triviales pour un multiplieur 4M2A

	Radix-2		Radix-4 (et 2^2)		Radix-8 (et 2^3)		Split-radix	
N	Mult.	Add.	Mult.	Add.	Mult.	Add.	Mult.	Add.
4	0	16	0	16			0	16
8	4	52			4	52	4	52
16	28	148	24	144			24	144
32	108	388					84	372
64	332	964	264	920	248	928	248	912
128	908	2308					660	2164
256	2316	5380	1800	5080			1656	5008
512	5644	12292			3992	11632	3988	11380
1024	13324	27652	10248	25944			9336	25488
2048	30732	61444					21396	56436
4096	69644	135172	53256	126296	48280	126832	48248	123792
8192	155660	294916					107412	269428
16384	344076	638980	262152	595288			236664	582544

TABLE 2.4 – Nombre de multiplications/additions non triviales pour un multiplieur 3M3A

	Radix-2		Radix-4 (et 2^2)		Radix-8 (et 2^3)		Split-radix	
N	Mult.	Add.	Mult.	Add.	Mult.	Add.	Mult.	Add.
4	0	16	0	16			0	16
8	4	52			4	52	4	52
16	24	152	20	148			20	148
32	88	408					68	388
64	264	1032	208	976	204	972	196	964
128	712	2504					516	2308
256	1800	5896	1392	5488			1284	5380
512	4360	13576			3204	12420	3076	12292
1024	10248	30728	7856	28336			7172	27652
2048	23560	68616					16388	61444
4096	53256	151560	40624	138928	38404	136708	36868	135172
8192	118792	331784					81924	294916
16384	262152	720904	199344	658096			180228	638980

gorithmes split-radix utilisant des combinaisons de radix-2/4, radix-2/8, radix-2/16, ou encore radix-2/32, auront tous la même complexité en termes d'additions et multiplications. Finalement, il a été démontré [73] que peu importe l'amélioration obtenue par un algorithme radix-p^2, l'algorithme radix-p/p^2 donnera de meilleurs résultats qu'un algorithme radix-p ou même radix-p^2. Par exemple l'algorithme radix-4/16 aura de meilleures performances que le radix-16.

Toutefois, l'utilisation d'algorithmes de plus hauts degrés augmente la complexité d'intégration dans un circuit dédié. Même si le nombre d'additions et de multiplications réelles non triviales offre un excellent indice sur l'efficacité d'un algorithme, l'intégration matérielle tient compte d'autres critères de performances, qui sont la régularité de l'algorithme ainsi que la complexité de réalisation et de contrôle de l'architecture. Les gains obtenus par la diminution de multiplications/additions peuvent parfois être perdus par la complexité de contrôle induite et du surplus d'interconnexion. D'où l'intérêt des algorithmes radix-2^i. Ces derniers offrent une plus grande régularité architecturale pour une réalisation matérielle par rapport au radix-8 ou 4 et encore plus par rapport au split-radix. La classe d'algorithme radix-2^i offre la possibilité d'atteindre presque les performances du split-radix, et de garder la régularité du radix-2. En effet, selon la valeur de i (2 à 4), on aura soit le même nombre de multiplications et d'additions réelles que le radix-4 (cas pour $i = 2$), ou que le radix-8 (cas pour i =3), soit on se rapprochera des performances du split-radix (cas pour $i = 4$) et cela en gardant toujours la même architecture régulière du radix-2 (N.B. : le radix-2 est un cas particulier du radix-2^i pour $i = 1$).

Les figures 2.9, 2.10 et 2.11 illustrent les flots de données pour les algorithmes radix-2, radix-2^2 et radix-2^3 respectivement. On constate le même flot de données pour les trois algorithmes. Il est composé de $\log_2 N$ étages chacun constitué de $\frac{N}{2}$ modules papillons de base (croisillon radix-2). Seuls les emplacements des multiplications complexes et les valeurs des facteurs de rotation W sont différents.

Comme on l'a mentionné à l'annexe B, les algorithmes radix-2^i ne peuvent être utilisés que pour des tailles de TFR N puissance de 2^i. De ce fait, dans le cas du radix-2^2 et pour N puissance de 2 seulement, il est nécessaire d'utiliser en premier le radix-2 sur un étage afin de décomposer la TFR à N points en 2 sous TFRs de $\frac{N}{2}$ puissance de 4. Pour le radix-2^3 et une valeur de $\log_2 N$ modulo 3 égale à 1, la décomposition commence aussi par un étage radix-2 suivi par le radix-2^3. Pour uqne valeur de $\log_2 N$ modulo 3 égale à 2, la décomposition commence par un étage radix-2^2 (équivalent à deux étages radix-2). Cette association de plusieurs radix afin de réaliser toutes les tailles de TFR puissance de 2 est connu sous le nom d'algorithme *mixed-radix*.

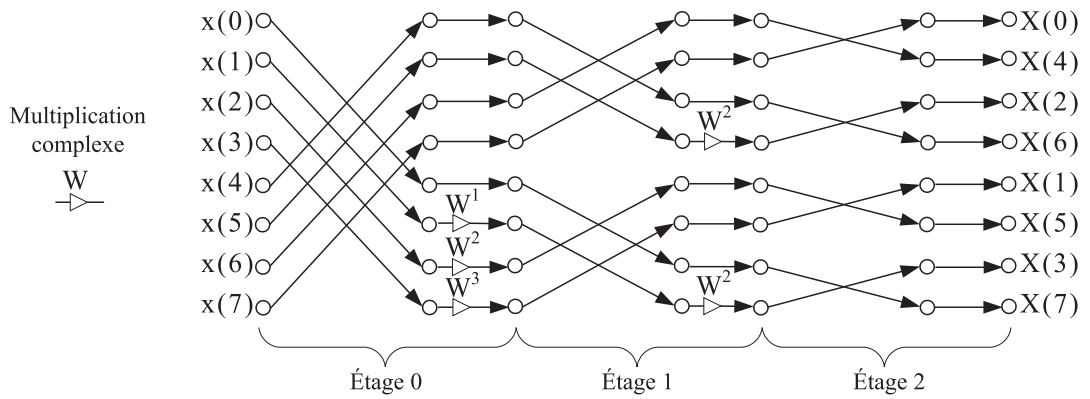

FIGURE 2.9 – TFR à $N = 8$ points utilisant le radix-2. N.B : les valeurs de W^x correspondent à W_8^x

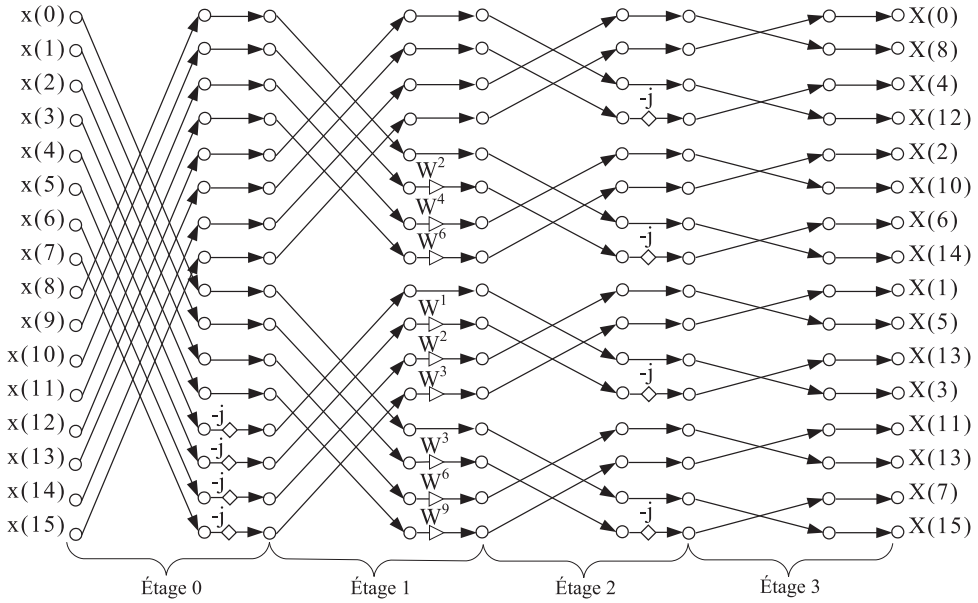

FIGURE 2.10 – TFR à $N = 16$ points utilisant le radix-2^2. N.B : les valeurs de W^x correspondent à W_{16}^x

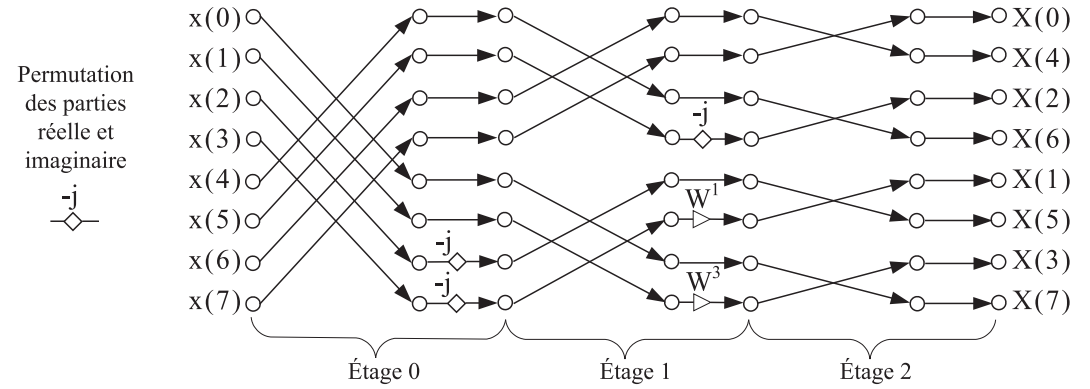

FIGURE 2.11 – TFR à $N = 8$ points utilisant le radix-2^3. N.B : les valeurs de W^x correspondent à W_8^x

2.3 Architectures conventionnelles de la TFR

La réalisation matérielle d'algorithmes de la TFR est l'aspect le plus important de l'étude des performances que pourrait obtenir un modulateur OFDM très haut débit multi-standard. Chacun des algorithmes considérés pour la TFR peut être réalisé par différents types d'architectures. Par conséquent, une conception performante en terme de débit et de puissance consommée passe par une bonne adéquation algorithme-architecture. Il existe principalement deux types d'architecture pour la réalisation de la TFR, les architectures en pipeline et les architectures à base de mémoires.

2.3.1 Architectures en pipeline

L'architecture en pipeline est caractérisée par un traitement continu des entrées séquentielles de la TFR. Une telle architecture est composée de $\log_r N$ modules de calcul (MC), un par étage, qui correspondent aux opérateurs papillon (multiplexage spatiale). La valeur de r correspond au radix de l'algorithme utilisé. Chaque MC traite N/r opérations papillon successives (multiplexage temporel). Par conséquent, le taille maximale réalisable pour ce type d'architecture est dictée par le nombre de MC.

Les architectures en pipeline ont une mémoire des échantillons distribuée sur tous les étages. En effet, à chacun des $\log_r N$ étages, il y a seulement $\frac{N}{2^\kappa}$ échantillons, κ étant le numéro de l'étage $1 \leq \kappa \leq \log_r N$. Il en est de même pour les mémoires des coefficients W, dont le nombre diffèrent d'un étage à un autre.

Les différentes architectures en pipeline diffèrent principalement selon deux points. La première différence est le nombre de chemins de données utilisés pour traiter les échantillons. On trouve ainsi deux types d'architectures, à chemin unique (*single-path*) ou à chemin multiple (*Multi-path*). Un plus grand nombre de chemins permet d'augmenter le débit de traitement mais au prix d'une augmentation des ressources nécessaires. La deuxième différence consiste en la stratégie de mémorisation pour créer les différent délais nécessaires à l'ordonnancement des échantillons.

Multi-path Delay Commutator

L'approche *Multi-path Delay Commutator* (MDC) est une des premières approches utilisées pour des architectures de la TFR en pipeline ; elle est encore largement utilisée [80, 81]. Elle comprend un commutateur pour permuter les échantillons et des délais pour synchroniser les mêmes échantillons. Selon l'algorithme radix-r utilisé, on aura $r-1$ délais avant et après chaque module papillon. De plus, cette architecture nécessite $r-1$ multiplieurs par module papillon. La figure 2.12a) illustre un exemple d'une architecture radix-2 MDC de 8 points. Les modules papillon, les multiplieurs et les délais ont un taux d'utilisation de 50% du temps.

La figure 2.12b) illustre le cas pour le radix-4. Cette architecture a un taux d'utilisation des multiplieurs de 25%. En général, on lui préfère la version SDC qui permet d'augmenter ce taux et de diminuer le nombre de multiplieurs.

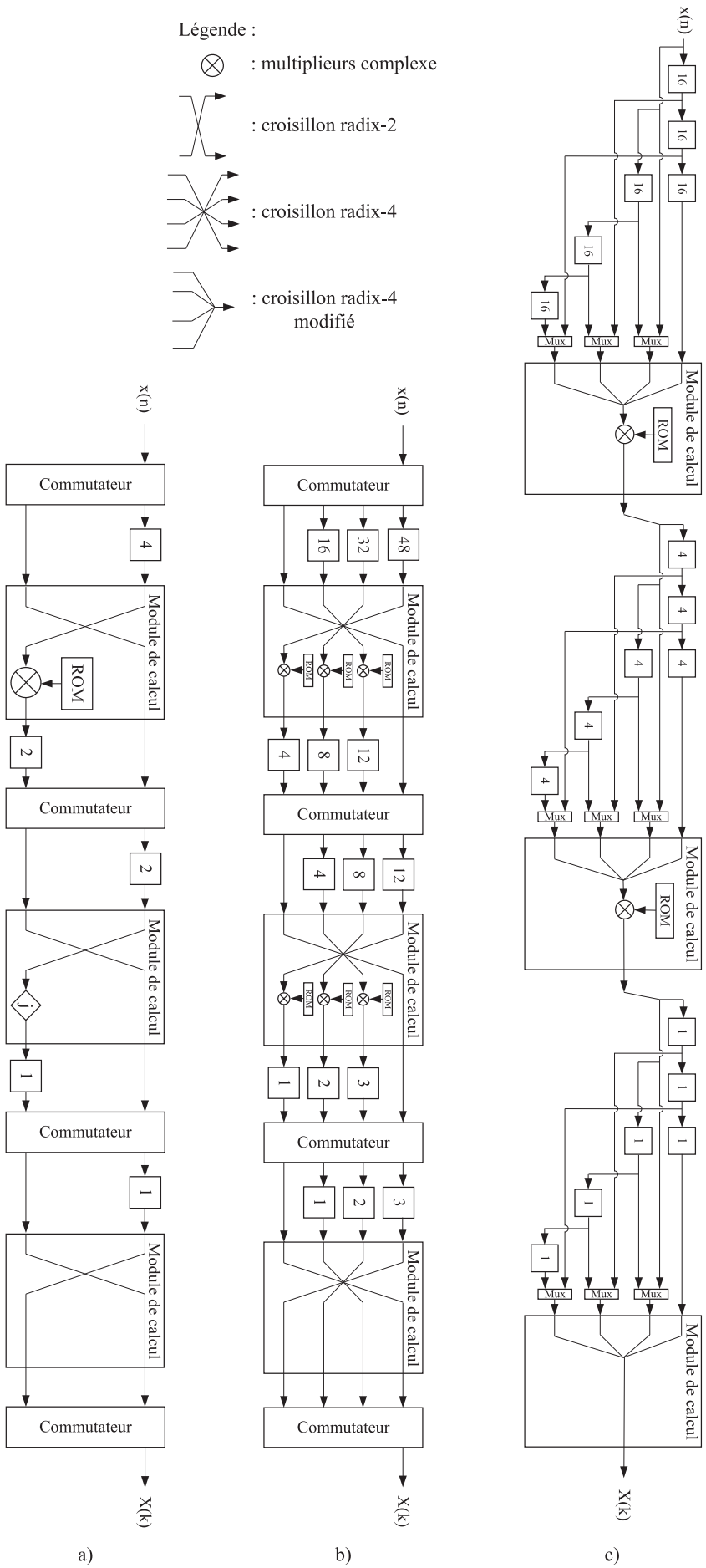

FIGURE 2.12 – Architectures de TFR en pipeline a) à 8 points radix-2 MDC b) à 64 points radix-4 MDC c) à 64 points radix-4 SDC

Single-path Delay Commutator

La version *Single-path Delay Commutator* (SDC) pour le radix-4 permet d'augmenter le taux d'utilisation du multiplieur à 75% en utilisant 1 multiplieur complexe au lieu de 3 multiplieurs complexes par étage [82,83]. Cette modification divise le papillon radix-4 en un papillon simplifié radix-4, traitant seulement une sortie du papillon radix-4 d'origine à la fois. Le papillon conventionnel radix-4 utilise 8 additionneurs/soustracteurs tandis que la version simplifiée utilise 3 additionneurs/soustracteurs. Cependant, cette modification engendre un contrôle plus complexe. La figure 2.12c) illustre l'architecture d'une TFR radix-4 SDC de 64 points.

Single-path Delay Feedback

Alors que les registres à délai sont utilisés à 50% du temps dans l'approche MDC, l'architecture *Single-path Delay Feedback* (SDF) permet d'optimiser le nombre de registres en utilisant deux fois moins de registres avec un taux d'utilisation de 100% [84–87]. Contrairement à l'approche MDC, au lieu de stocker les échantillons entrant et sortant du module papillon dans deux ensembles différents de $r-1$ registres (pour une algorithme radix-r), l'approche SDF utilise un seul ensemble de $r-1$ délais. La figure 2.13 illustre l'architecture d'une TFR radix-2 à 8 points utilisant l'approche SDF.

Afin d'exploiter le parallélisme et ainsi répondre à des débits plus importants, plusieurs chemins de données sont utilisés. On obtient ainsi des architectures en pipeline-parallèles [88,89]. La figure 2.14 illustre une architecture en pipeline SDF avec un parallélisme des chemins de données de $P=4$ proposée notamment pour l'UWB.

Comparaison

L'approche radix-r MDC est r fois plus rapide que l'approche radix-r SDF. Toutefois en termes de mémoires, l'approche SDF est plus économe que l'approche MDC (qui souffre d'un surplus de surface par rapport à l'approche SDF). En termes de ressources, les architectures radix-2^4 et radix-4^2 SDF sont plus performantes que n'importe quelle autre approche comme on le voit au tableau 2.5. Elles nécessitent toujours le moins de ressources arithmétiques et de mémoires. Ceci est dû entre autres au fait qu'elles utilisent plusieurs multiplieurs par des valeurs constantes optimisées et caractérisées par une complexité inférieure. En termes de complexité de contrôle, les architectures pour le radix-4-SDC et split-radix sont les plus complexes à contrôler.

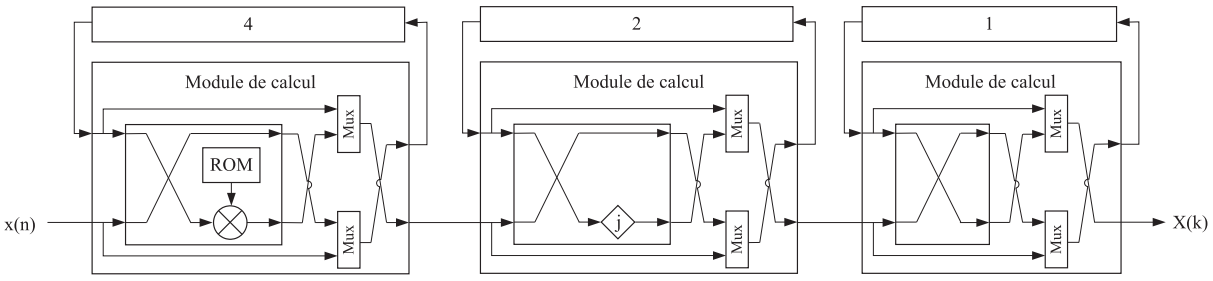

FIGURE 2.13 – FFT à 8 points radix-2 SDF

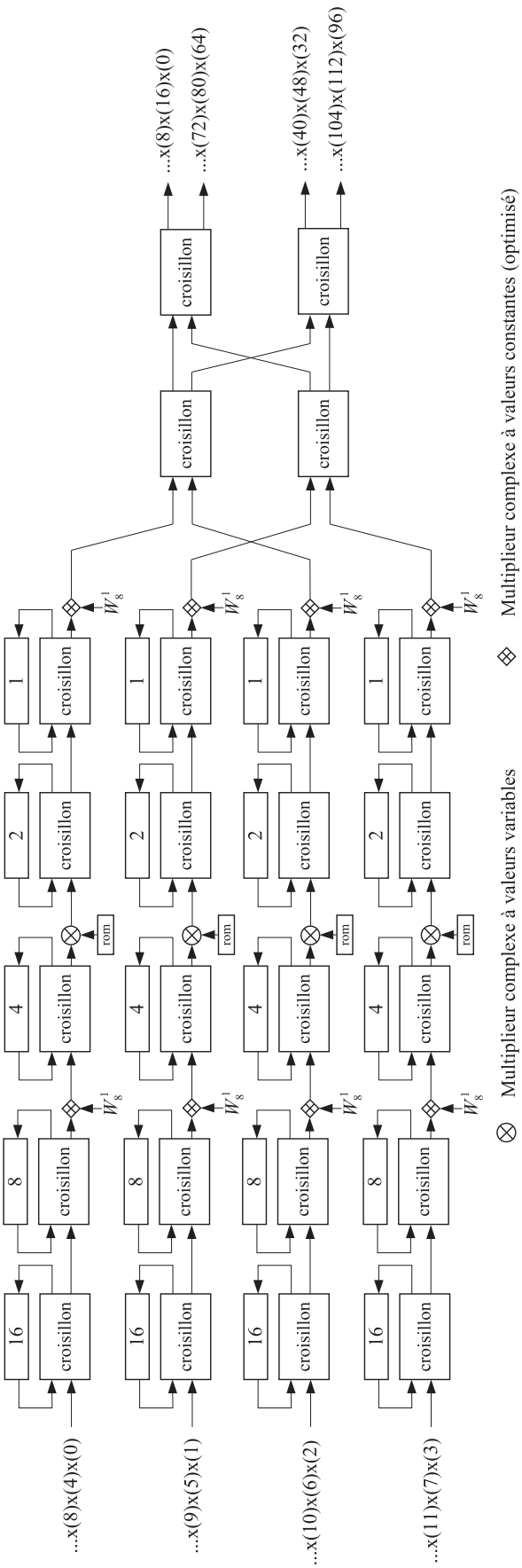

FIGURE 2.14 – Architecture en pipeline-parallèle SDF mixed-radix-$2^3/2^4$ de 128 points pour l'UWB

TABLE 2.5 – Ressources matérielles selon l'architecture en pipeline utilisée

Architecture	♯ Multiplieurs complexes	♯ additionneurs complexes	Mémoire	Contrôle	$f_{op}^{\dagger}$	Réf.[§]
R2 MDC	$2\log_4 N - 1$	$4\log_4 N$	$\frac{3N}{2} - 2$	simple	$\frac{f_{échant}^{\ddagger}}{r}$	[80]
R2 SDF	$2\log_4 N - 1$	$4\log_4 N$	$N - 1$	simple	$f_{échant}$	[90]
R4 MDC	$3\,(\log_4 N - 1)$	$8\log_4 N$	$\frac{5N}{2} - 4$	simple	$\frac{f_{échant}}{r}$	[80]
R4 SDF	$\log_4 N - 1$	$8\log_4 N$	$N - 1$	moyen	$f_{échant}$	[91]
R4 SDC	$\log_4 N - 1$	$3\log_4 N$	$2N - 2$	complexe	$f_{échant}$	[82]
R2^2 SDF	$\log_4 N - 1$	$4\log_4 N$	$N - 1$	simple	$f_{échant}$	[92]
R2^3 SDF	$\approx \log_4 N - 1$	$4\log_4 N$	$N - 1$	simple	$f_{échant}$	[92]
R2^4 SDF	$\leq 0.7\log_4 N - 1$	$4\log_4 N$	$N - 1$	simple	$f_{échant}$	[92]
R4^2 SDF	$\leq 0.7\log_4 N - 1$	$4\log_4 N$	$N - 1$	simple	$f_{échant}$	[92]
SR MDC	$4\,(\log_4 N - 1)$	$12\log_4 N - 8$	$\frac{3N}{2} - 2$	complexe	$\frac{f_{échant}}{r}$	[93]
SR SDF	$\log_4 N - 1$	$4\log_4 N$	$N - 1$	complexe	$f_{échant}$	[94]

[†]fréquence d'opération. [‡]fréquence d'échantillonnage. [§]Références bibliographiques

2.3.2 Architectures à mémoires

Les architectures à base de mémoires mémorisent en général les N échantillons à l'entrée avant d'entamer le traitement de la TFR. Elles sont composées d'un ou de plusieurs modules de calculs (papillons), d'unités de mémorisation et d'une unité de contrôle. Contrairement à l'approche en pipeline, la plus grande taille de TFR ne dépend que de la taille mémoire disponible et non du nombre de modules de calcul [95–103]. Du fait du multiplexage temporel du traitement de la TFR qui caractérise ce type de solution, la surface totale du circuit est inférieure à celle des architectures de type pipelinées qui sont caractérisées par un multiplexage spatial et qui nécessitent donc plus de ressources. La figure 2.15 illustre l'architecture à base de mémoires.

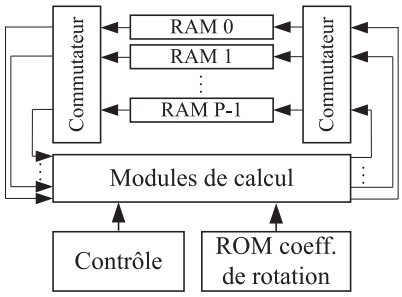

FIGURE 2.15 – Architecture à base de mémoires

Il est possible d'obtenir plusieurs degrés de parallélisme et de profondeur de pipeline selon le type de configuration des modules de calculs. La figure 2.16 illustre quatre configurations différentes des modules de calcul. La première approche décrite à la figure 2.16a) est la plus simple. Elle consiste en un module de calcul réalisant l'opérateur papillon radix-2. La fréquence d'opération de ce type de configuration est plus grande que dans le cas des architectures en pipelines.

L'augmentation du degré de parallélisme réduit la fréquence d'opération. Ceci peut être obtenu soit en utilisant plusieurs modules papillon en parallèle soit par la réalisation d'un module papillon d'un radix élevé, ce qui permet de traiter plusieurs données à la fois [104]. Les figures

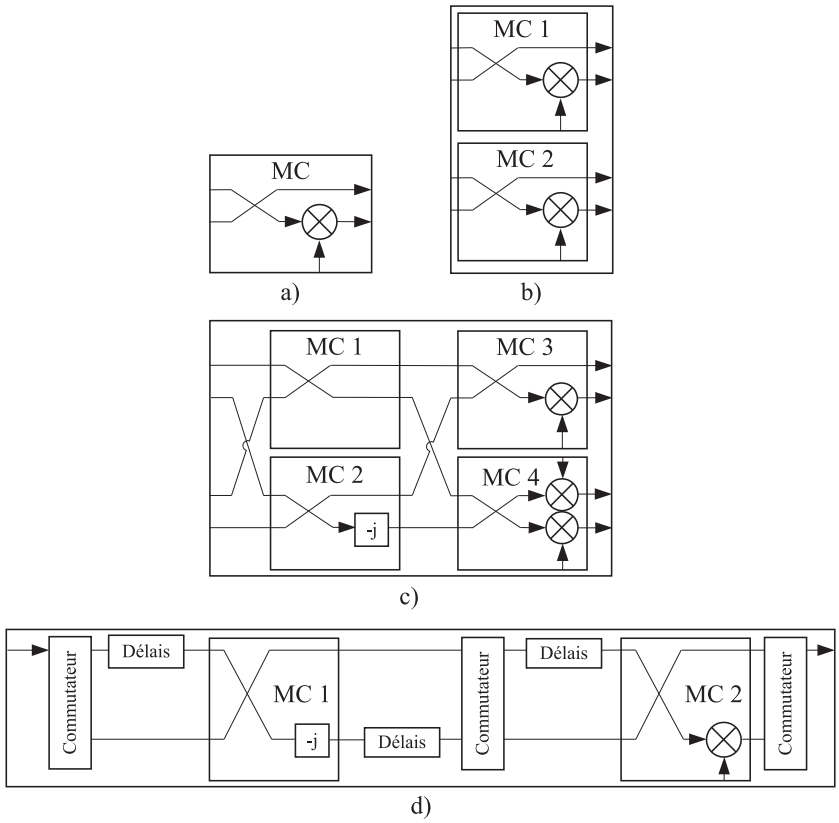

FIGURE 2.16 – Différentes configurations des modules de calculs des architectures à mémoires : a) un papillon radix-2 [97], b) deux papillons radix-2 en parallèle [98,102], c) un papillon radix-2^2 [99], d) un papillon radix-2^2 en pipeline MDC [100,101,103]

2.16b) et c) illustrent ces cas de figures avec la configuration de deux modules radix-2 en parallèle et d'un module radix-2^2 respectivement. Dans les deux cas, nous avons le même degré de parallélisme de 4. Néanmoins, le cas radix-2^2 à la figure 2.16c) exploite un plus grand multiplexage spatiale et réalise deux étages de suite.

La figure 2.16d) illustre une autre approche pour réaliser l'algorithme radix-2^2 dans une architecture à mémoires. Dans ce cas, les deux modules de calcul sont en pipeline (MDC) et aucun parallélisme intra-étage des modules n'est utilisé. La fréquence d'opération de cette architecture est moins élevée que celle du cas simple à la figure 2.16a) mais reste un peu plus élevé que celle de l'approche 2.16c). Toutefois, cette approche possède un bon compromis entre les solutions a) et c). En effet, la figure d) nécessite moins de ressources matérielles que l'approche c) tout en offrant de bonnes performances de puissance de calcul.

Un des avantages supplémentaires de l'utilisation d'un radix plus élevé dans les architectures à base de mémoires est la diminution des opérations de lecture/écriture dans les mémoires des échantillons. En effet, ces opérations s'effectuent tous les $\log_r N$ étages. Les configurations des types 2.16c) et 2.16d) auront donc besoin de moins d'accès mémoires comparativement aux configurations 2.16a) et 2.16b), ce qui permet de limiter grandement la puissance consommée.

Le tableau 2.6 donne le nombre de modules de calcul nécessaires et la fréquence d'opération selon la configuration choisie (pour des algorithmes radix-2^i). P_D correspond au parallélisme des données, c'est-à-dire le nombre d'échantillons à l'entrée des modules de calcul à chaque coup

TABLE 2.6 – Ressources matérielles et fréquences d'opération des architectures à base de mémoires. Configurations A : un papillon radix-2, B : deux papillons radix-2 en parallèle, C : un papillon radix-2^2 , D : un papillon radix-2^2 en pipeline MDC

Configuration	♯ MC	Parallélisme des données P_D	Fréquence d'opération
A, B et C	$P \times i \times \left(\frac{2^i}{2}\right)$	$P_D = P \times 2^i$	$\frac{\log_r N\left[\left(\frac{N}{P_D}-1\right)+T_{calcul}\right]}{N} f_{échant}$
D	$P \times \log_2 r$	$P_D = P \times P_C$	

d'horloge. P correspond au nombre de modules papillon radix-2^i en parallèle. P_C correspond au nombre de chemins de données pour la configuration 2.16d). P_C vaut 1 pour un module papillon SDF et 2 pour l'approche MDC (en considérant les algorithmes radix-2^i). T_{calcul} correspond au nombre de cycles de traitement des modules de calcul.

Les solutions à base de mémoires sont une approche adéquate pour la réalisation d'une architecture à gros grains à mi-chemin entre une intégration DSP et une intégration ASIC. En effet, cette approche offre de meilleures performances que des solutions à base de processeurs et est beaucoup plus flexible que les ASICs. Elle comprend un ensemble de jeu d'instructions et une matrice reconfigurable. Du fait que les reconfigurations se font au niveau des opérateurs arithmétiques (multipliers, additionneurs, etc.), le surplus de surface est limité. Ce type de réalisation est parfaitement adapté à des applications orientées données, tels que l'algorithme de TFR, caractérisé par des coeurs de boucles très réguliers.

L'architecture RaPiD [105] est une architecture hétérogène qui combine à la fois des composants à gros grains et à grains fins implémentant entre autre un récepteur OFDM. Elle est constituée par un chemin de données pipeliné reconfigurable et 16 unités fonctionnelles servant au traitement artihmétique des données. Il a été démontré que cette architecture réduit l'écart entre les performances des DSP programmables et les architectures ASICs d'un facteur 6 dans le cas du récepteur OFDM.

Afin d'augmenter les performances des architectures à mémoires, des solutions à base de mémoires caches ont été proposées [106, 107]. En effet, l'emploi de mémoires caches entre les modules de calculs papillons et les mémoires principales permet d'obtenir une augmentation de la vitesse de traitement et un rendement d'énergie plus efficace du fait que des mémoires de plus petite taille sont plus rapides et dissipent moins d'énergie. Comme le calcul de la TFR est déterministe, la mémoire cache n'a pas besoin d'étiquette, une opération sur la mémoire cache est réalisée directement via un mappage des adresses. Toutefois, leur utilisation augmente la complexité de contrôle de l'architecture.

2.4 Algorithme et architecture de mise en forme IOTA

Le filtrage de mise en forme augmente la complexité de l'architecture du modulateur OFDM avancé comparativement au modulateur OFDM/QAM classique. Il nécessite des ressources arithmétiques et des ressources mémoires pour les coefficients IOTA ainsi que pour les échantillons filtrés.

La fonction IOTA de la figure 2.5 est échantillonnée à une fréquence $\frac{M}{\tau_0}$ hertz pour un total de $2ML$ échantillons. Le paramètre L correspond à la longueur de troncature de la fonction IOTA. La fonction couvres L période de TFR inverse de N échantillons. Comme nous l'avons mentionné, pour chaque TFR à N points, seulement $M = \frac{N}{2}$ échantillons sont émis après le filtrage. La fonction IOTA filtre donc $2L$ symboles OFDM/OQAM de M échantillons. Les longueurs de troncatures L considérées pour cette étude sont 2, 4 et 8.

Durant le filtrage de mise en forme, $2L$ fonctions IOTA se superposent dans le domaine temporel. Chaque fonction est décalée par rapport aux autres de M échantillons, soit un temps symbole OFDM/OQAM. Ainsi, chaque échantillon émis résulte de la sommation de $2L$ échantillons de TFR inverses différentes pondérés par les différentes fonctions IOTA.

La figure 2.17 illustre l'algorithme de filtrage de mise en forme pour les $2L$ premiers symboles OFDM/OQAM. Cet exemple est donné pour $N = 8$ ($M = 4$) et $L = 2$. Les $2ML$ coefficients IOTA sont représentés par $\Im_g$ ($0 \leq g \leq 2ML - 1$). $C_{j,k}$ représente le $k^{ième}$ échantillon du $j^{ième}$ résultat de TFR inverse, soit $X_j(k)$. $S_{j,k}$ représente le $k^{ième}$ échantillon du $j^{ième}$ symbol OFDM/OQAM. À la sortie du modulateur, l'expression du signal discret est [32] :

$$S_{k+jM} = \sum_{q=0}^{2L-1} [\alpha_{k,q} C_{k,j-q} + \beta_{k,q} C_{k+M,j-q}] \tag{2.6}$$

avec

$$\alpha_{k,q} = \begin{cases} 0 & \text{si } q \text{ impair} \\ \Im_{k+qM} & \text{si } q \text{ pair} \end{cases} \qquad \beta_{k,q} = \begin{cases} \Im_{k+qM} & \text{si } q \text{ impair} \\ 0 & \text{si } q \text{ pair} \end{cases}$$

Avec $0 \leq k \leq M - 1$ et $0 \leq q \leq 2L - 1$ correspond aux $2L$ fonction IOTA superposées. L'indice j indique l'ordre du symbole OFDM dont il est question tout au long de la transmission et vaut $j \in Z$.

L'algorithme ainsi que l'architecture de filtrage peuvent être décrits de la façon suivante [32] : à la fin de la TFR inverse, le filtre polyphase reçoit à chaque cycle deux échantillons complexes, $C_{j,k}$ et $C_{j,k+M}$, qui représentent un échantillon des M premiers et un échantillon des M derniers résultats de la TFR inverse respectivement. Chacun de ces échantillons, $C_{j,k}$ et $C_{j,k+M}$, est pondéré par L coefficients différents de la fonction IOTA, respectivement $\Im_{k+2\ell M}$ et $\Im_{k+(2\ell+1)M}$ (avec $0 \leq \ell \leq L - 1$), grâce à $2L$ multiplieurs et additionnés aux $2L - 1$ résultats de TFR inverses précédents à $2M$ points. Finalement, M échantillons complexes sont émis par symbole OFDM/OQAM. Ainsi, l'algorithme de filtre nécessite $2L - 1$ blocs mémoires à M échantillons complexes pour mémoriser les résultats du filtrage non émis.

La figure 2.18 illustre l'implémentation matérielle du filtre polyphase IOTA selon le paramètre L. On y trouve $(2L - 1)$ FIFOs de M données complexes, $2L$ multiplieurs et additionneurs réels (fois deux pour la partie réelle et imaginaire). En utilisant $2L - 1$ FIFOs, on obtient une architecture reproduisant le chemin de données du filtrage polyphase avec un minimum de logique de contrôle.

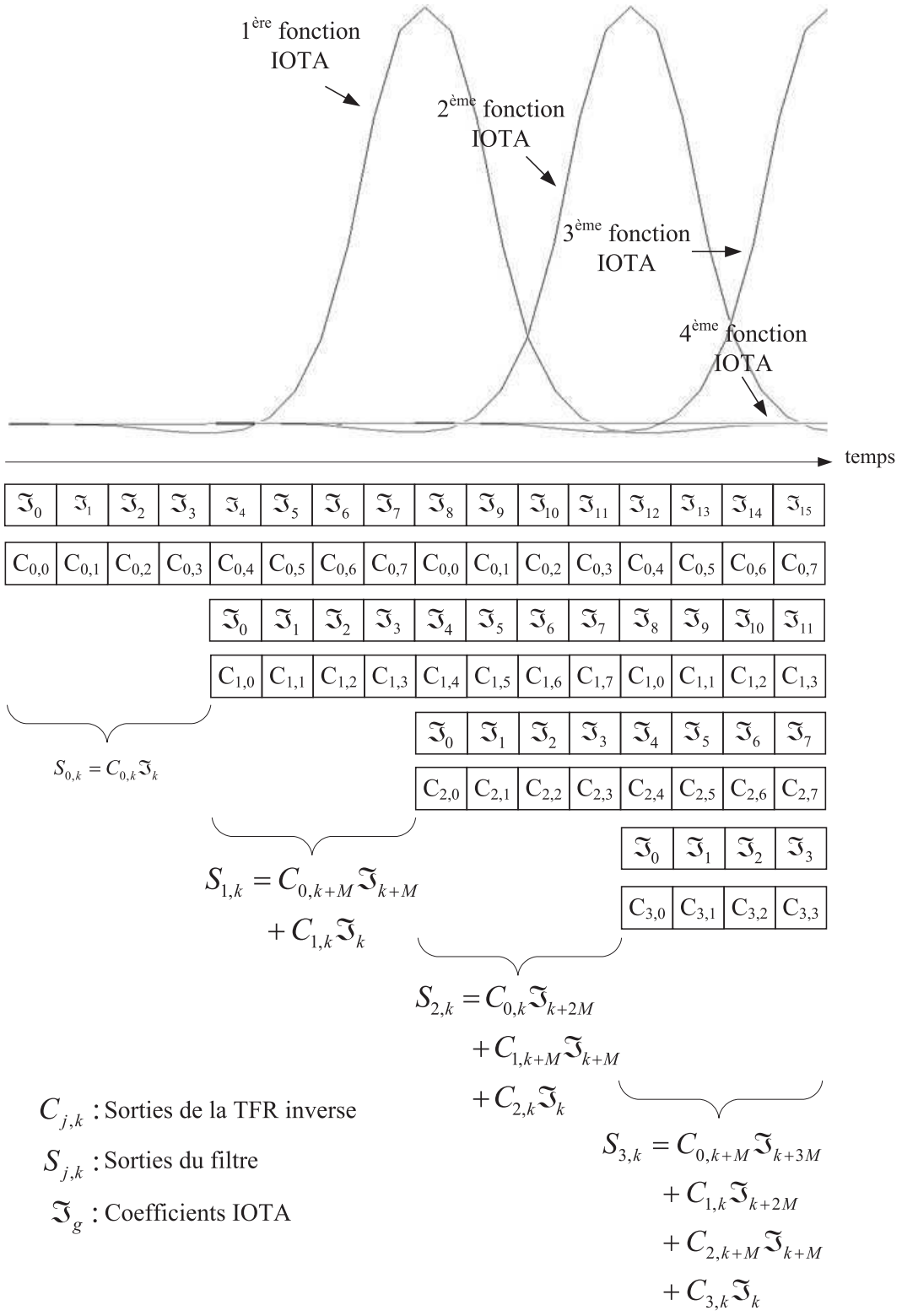

FIGURE 2.17 – Illustration de l'algorithme du filtre polyphase avec M=4 et L=2

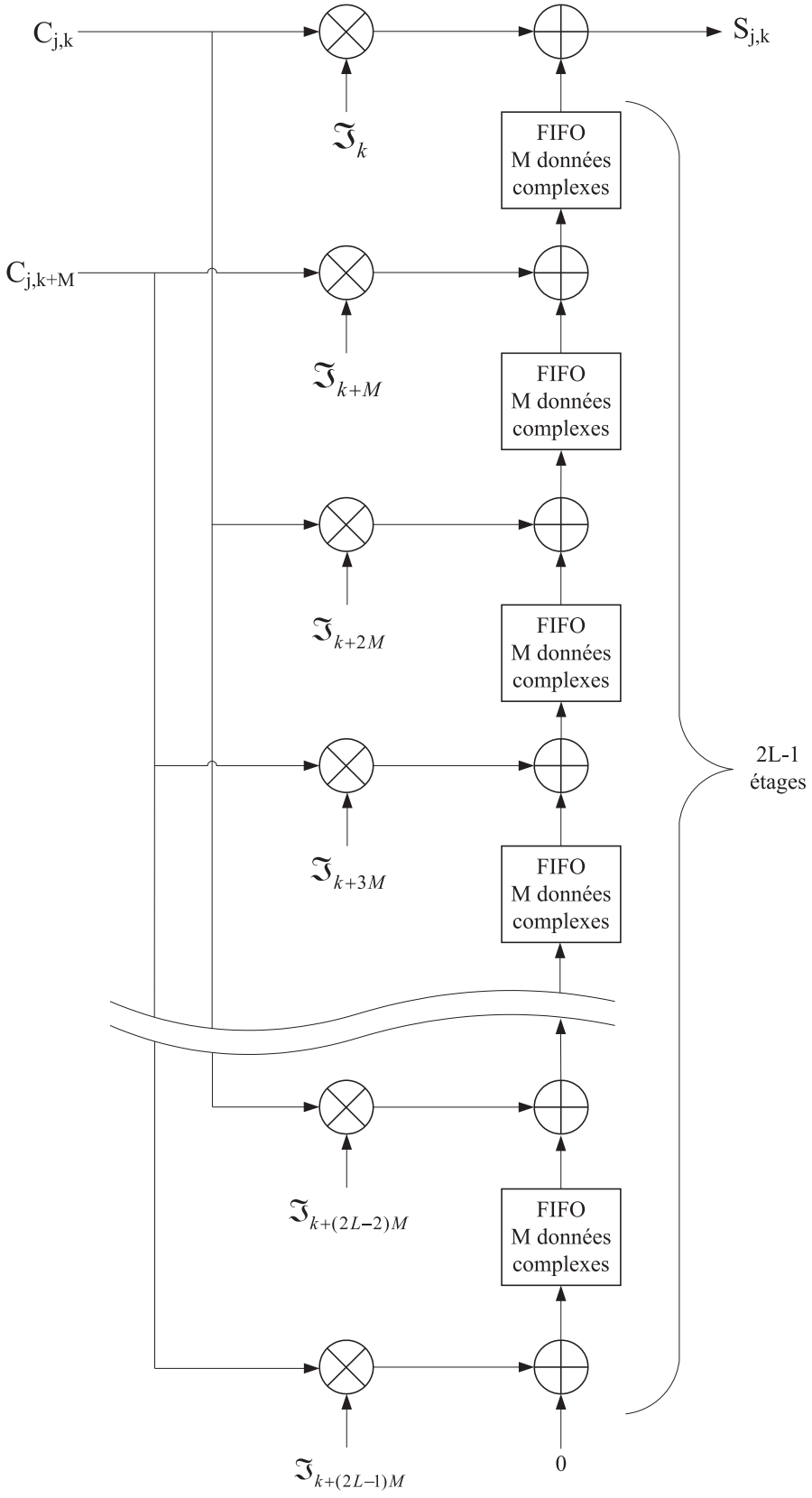

FIGURE 2.18 – Implémentation matérielle du filtre polyphase IOTA selon la valeur de L

Chapitre 3

Compromis entre débit, reconfigurabilité et consommation

Ce chapitre présente la meilleure approche de conception offrant un bon compromis entre capacité de traitement, surface et puissance consommée pour un modulateur OFDM avancé multi-standards et haut débit. Une analyse des différentes architectures présentées dans le chapitre précédant en fonction des contraintes considérées est réalisée.

Dans la prochaine section, nous comparons les différentes architectures pour trois normes, soit l'UWB, la norme DVB-H, et 3GPP-LTE. Les normes basées sur la modulation OFDM diffèrent principalement par leur temps symbole, ou en d'autre terme l'écart inter-porteuses, ainsi que par le nombre de porteuses N. L'UWB est la norme la plus contraignante en terme de temps symbole, qui est de 242,42 ns ($v_0 = 4,125MHz$) pour l'OFDM/QAM pour une TFR de 128 points (un temps symbole qui serait deux fois plus court pour l'OFDM/ OQAM). En terme de taille TFR, les normes de radiodiffusions numériques DVB-H nécessitent des tailles allant jusqu'a 8192 porteuses. La version de la norme 3GPP-LTE à 512 porteuses est moyennement contrainte en terme de débit et en terme de nombre de porteuses comparativement à l'UWB et DVB-H.

Cette étude est réalisée en considérant une même tension d'alimentation pour tous les différents types d'architectures. Par la suite, dans la section 3.2, la même analyse est réalisée mais en considérant, cette fois-ci, une gestion des tensions d'alimentation entre les différentes architectures. Finalement, à la section 3.3, nous présentons l'influence de la reconfigurabilité sur les architectures à pipeline et à mémoires et l'influences sur la consommation de puissance. Les conclusions tirées de ce chapitre nous ont menés à l'architecture qui sera proposée par la suite.

3.1 Comparaison des performances pour des tensions constantes

Les contraintes des normes existantes et les caractéristiques des normes futures engendrent une plus grande consommation de puissance. La diminution du temps symbole influe sur les architectures à base de mémoires par une fréquence d'opération plus grande et par un degré de parallélisme plus grand pour les architectures en pipeline. L'augmentation du nombre de porteuses influe sur les architectures en pipeline par un plus grand nombre de modules de calculs nécessaires. Les tableaux 3.1 à 3.3 illustrent les ressources nécessaires pour différents degrés de parallélisme P pour l'UWB à 128 porteuses, DVB-H à 8192 porteuses et le 3GPP-LTE à 512 porteuses selon une architecture en pipeline et à base de mémoires. Le nombre de multiplieurs complexes et d'additionneurs complexes des architectures en pipelines sont tirés du tableau 2.5.

Les solutions (1) et (2) sont des architectures en pipeline-parallèles tel qu'illustré à la figure 2.14. Les architectures (3) à (5) sont à base de mémoires radix-2^i. L'architecture (6) est une architecture en pipeline SDF radix-2^4.

La puissance consommée est proportionnelle à :

$$P \propto S f_{op} V^2 \tag{3.1}$$

où S représente la surface de circuit, f_{op} la fréquence d'opération, et V la tension d'alimentation. En normalisant la puissance par rapport à S, la fréquence d'échantillonnage $f_{échant}$ et V, on

TABLE 3.1 – Ressources des différentes architectures pour la norme UWB

Architectures	Parallélisme (P)	♯ UC	♯ Mult. complexes	♯ Addit. complexes	Fréquence d'opération ($f_{op} = k f_{échant}$)	♯ accès mémoire
(1) SDF radix-2^4	2	13	2.9	26	$0.5 f_{échant}$	7N
(2) SDF radix-2^4	4	24	5.8	48	$0.25 f_{échant}$	7N
(3) Mémoire radix-2^2	4	4	3	8	$1.09 f_{échant}$	4N
(4) Mémoire radix-2^2	8	8	6	16	$0.59 f_{échant}$	4N
(5) Mémoire radix-2^3	8	12	10	24	$0.48 f_{échant}$	3N

TABLE 3.2 – Ressources des différentes architectures pour la norme DVB-H avec $N = 8192$

Architectures	Parallélisme (P)	♯ UC	♯ Mult. complexes	♯ Addit. complexes	Fréquence d'opération ($f_{op} = k f_{échant}$)	♯ accès mémoire
(6) SDF radix-2^4	1	13	3.6	26	$f_{échant}$	13N
(3) Mémoire radix-2^2	4	4	3	8	$1.75 f_{échant}$	7N
(4) Mémoire radix-2^2	8	8	6	16	$0.88 f_{échant}$	7N
(5) Mémoire radix-2^3	8	12	10	24	$0.63 f_{échant}$	5N

TABLE 3.3 – Ressources des différentes architectures pour la norme 3GPP-LTE avec $N = 512$

Architectures	Parallélisme (P)	♯ UC	♯ Mult. complexes	♯ Addit. complexes	Fréquence d'opération ($f_{op} = k f_{échant}$)	♯ accès mémoire
(6) SDF radix-2^4	1	9	2.2	18	$f_{échant}$	9N
(3) Mémoire radix-2^2	4	4	3	8	$1.28 f_{échant}$	4.5N
(4) Mémoire radix-2^2	8	8	6	16	$0.65 f_{échant}$	4.5N
(5) Mémoire radix-2^3	8	12	10	24	$0.41 f_{échant}$	3N

obtient une échelle de comparaison entre les différentes architectures. Ceci en considérant que toutes les architectures utilisent les mêmes types de multiplieurs et d'additionneurs. L'équation 3.1 dans le cas des multiplieurs et des additionneurs devient alors :

$$P_{norm} = \frac{P}{S_{Mult/Add} f_{échant} V^2} \propto \sharp\text{éléments} \times k \tag{3.2}$$

Où $\sharp$ *éléments* représente le nombre de multiplieurs ou additionneurs complexes, $S_{Mult/Add}$ la surface d'un multiplieur ou d'un additionneur et k le coefficient de proportionnalité de f_{op} par rapport à $f_{échant}$ des tableaux 3.1 à 3.3. Pour la consommation des mémoires, elle est approximée par l'équation suivante :

$$P_{norm} = \frac{P}{S_{mémoire} f_{échant} V^2} \propto \sharp\text{accès mémoires} \times k \tag{3.3}$$

Les figures 3.1 à 3.3 illustrent les puissances normalisées relatives pour les différentes architectures pour les trois normes. Dans la cas de l'UWB, les multiplieurs des architectures en pipeline-parallèles (1) et (2) consomment moins de puissance que les architectures à base de mémoires (3) à (5). Ces dernières offrent de meilleures performances en termes de puissances consommées pour les additionneurs. Pour les mémoires, les architectures (2) en pipeline-parallèle et (5) à base de mémoires consomment le moins. Ceci s'explique par le fait que les fréquences d'opération sont relativement basses du fait du parallélisme, soit $0,25 f_{échant}$ pour l'architecture (2) et $0,48 f_{échant}$ pour l'architecture (5). Pour cette dernière, le nombre d'accès mémoire est aussi le moins élevé.

Il en est de même pour les cas du DVB-H et du 3GPP-LTE, où les multiplieurs de l'architecture en pipeline consomment moins que les architecture à base de mémoires. Toutefois, pour un grand nombre de porteuses comme c'est le cas du DVB-H, la consommation des mémoires est prépondérante du fait de la taille des mémoires. Les architectures à bases de mémoires associées à un radix élevé comme c'est le cas de l'architecture (5), permettent d'optimiser leur consommation du fait du nombre restreint d'accès mémoires.

Dans cette analyse, les puissances relatives des différents éléments, les multiplieurs, les additionneurs et les mémoires, sont à des échelles de grandeur différentes. Ainsi, il est difficile de comparer les contributions de ces différents éléments, à la consommation totale de l'architecture.

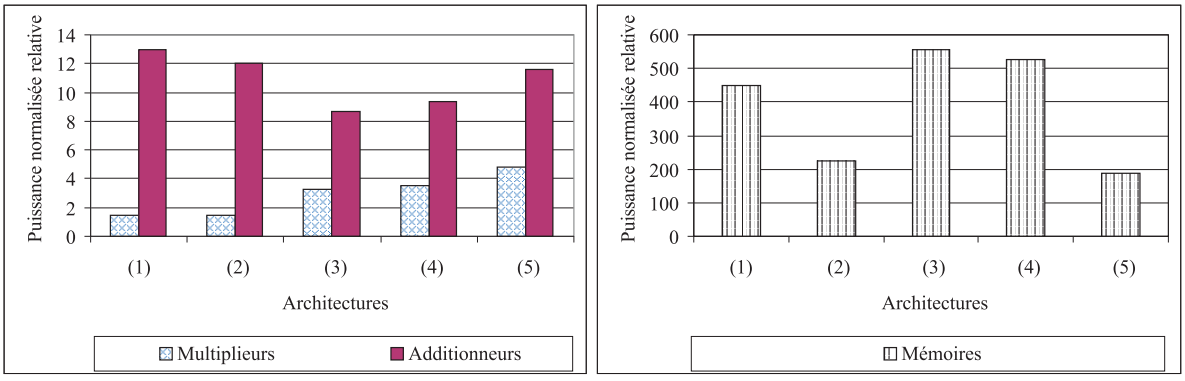

FIGURE 3.1 – Consommation normalisée relative des multiplieurs complexes, des additionneurs complexes et des mémoires de la TFR pour la norme UWB $N = 128$

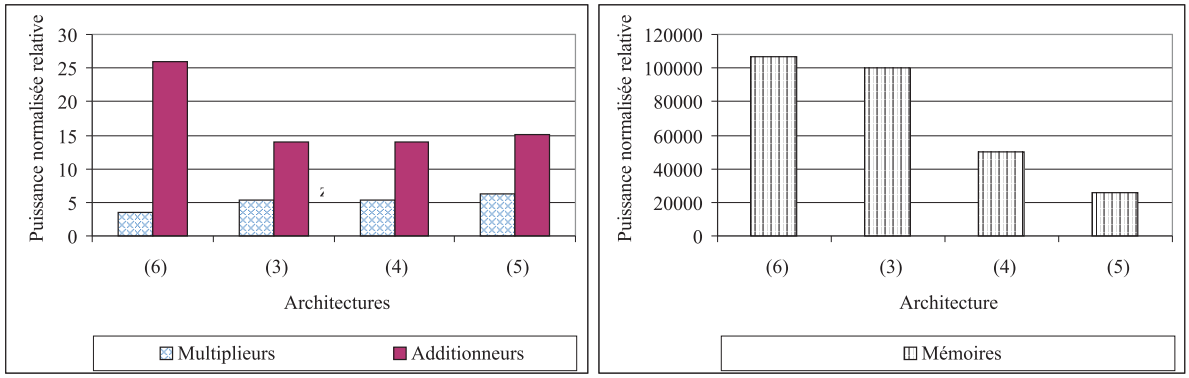

FIGURE 3.2 – Consommation normalisée relative des multiplieurs complexes, des additionneurs complexes et des mémoires de la TFR pour la norme DVB-H avec $N = 8192$

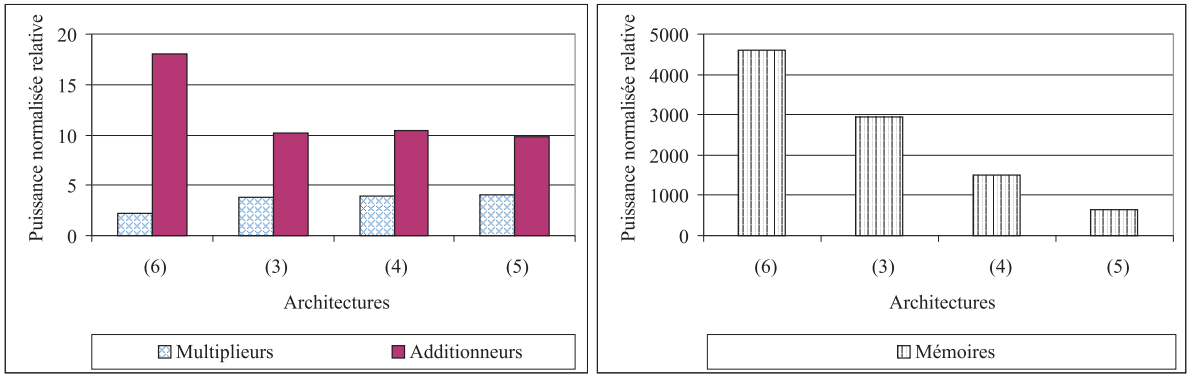

FIGURE 3.3 – Consommation normalisée relative des multiplieurs complexes, des additionneurs complexes et des mémoires de la TFR pour la norme 3GPP-LTE avec $N = 512$

3.2 Comparaison des performances pour des tensions variables

Le parallélisme permet d'opérer à des fréquences plus faibles. Un gain de puissance est obtenu en appliquant une baisse de tension proportionnelle à la baisse de fréquence lorsque la technologie d'implémentation le permet. En effet, selon [108], la fréquence maximale est proportionnelle à :

$$f_{max} \propto (V_{dd} - \alpha_1 - \alpha_2 V_{bs})^{\beta}, \tag{3.4}$$

avec V_{dd}, la tension d'alimentation du circuit, V_{bs}, la tension d'alimentation du substrat du transistor, $\alpha_1 > 0$, $\alpha_2 > 0$, $\beta \approx 1$ pour les technologies récentes. Ainsi, la tension d'alimentation est linéairement proportionnelle à la fréquence d'opération. Pour notre étude, on considère que la $f_{échant} = f_{max}$ pour une tension donnée V. Or, selon les tableaux 3.1 à 3.3, $f_{op} = k \times f_{échant}$ avec $k < 1$. En appliquant une tension kV, les équations 3.2 et 3.3 deviennent :

$$P_{norm} = \frac{P}{S_{Mult/Add} f_{échant} V^2} \propto \sharp\text{éléments} \times k^3 \tag{3.5}$$

$$P_{norm} = \frac{P}{S_{mémoire} f_{échant} V^2} \propto \sharp\text{accès mémoires} \times k^3 \tag{3.6}$$

Cette baisse de tension a été appliquée aux cas de figure des tableaux 3.1 et 3.2 où $k < 1$, soit pour toutes les architectures opérant à une plus basse fréquence et pour lesquelles une baisse

de la tension d'alimentation est possible. Les figures 3.4 à 3.6 illustrent les consommations obtenues. L'architecture (5) à base de mémoires et à fort parallélisme ($P = 8$) offre les meilleures performances pour les normes DVB-H et 3GPP-LTE. Dans le cas de l'UWB, l'architecture (2) en pipeline-parallèle avec un parallélisme de 4 offre de meilleures performances que l'architecture à base de mémoires (5).

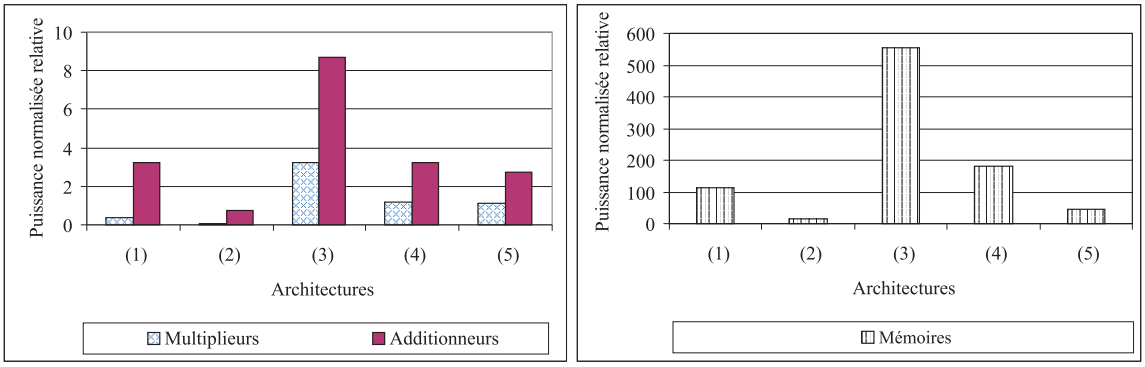

FIGURE 3.4 – Consommation normalisée relative pour la norme UWB $N = 128$ avec gestion de la tension

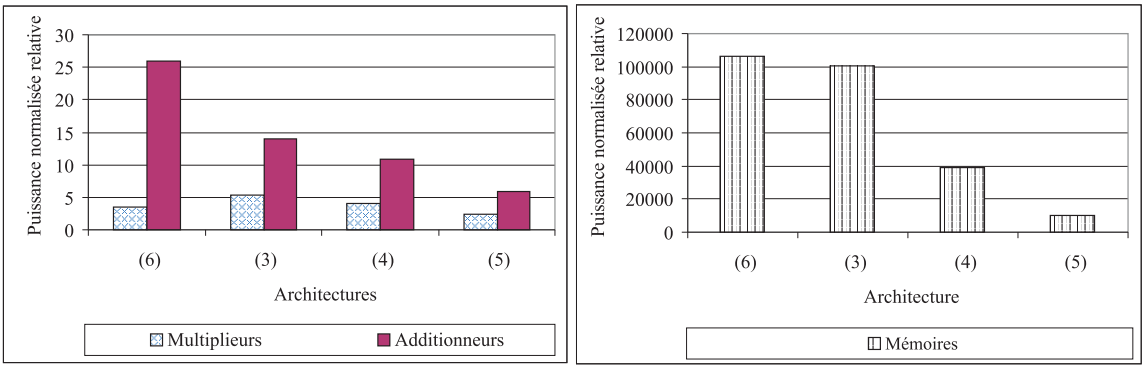

FIGURE 3.5 – Consommation normalisée relative pour la norme DVB-H avec $N = 8192$ et gestion de la tension

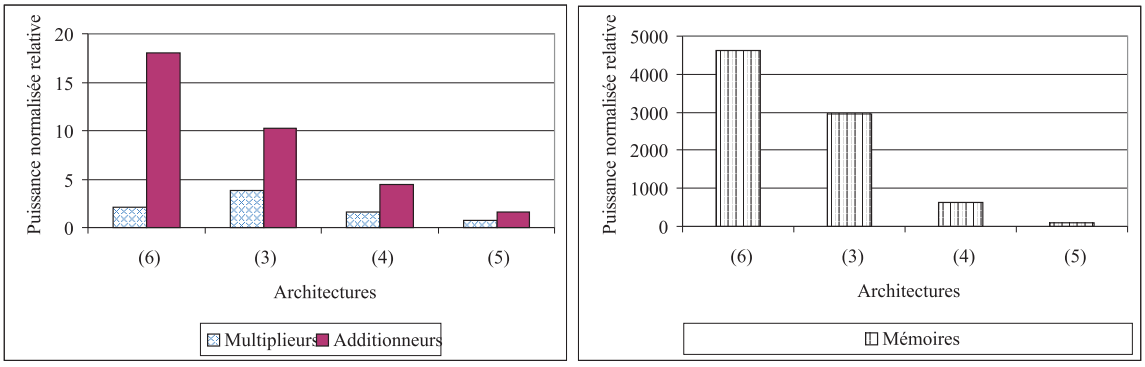

FIGURE 3.6 – Consommation normalisée relative pour la norme 3GPP-LTE avec $N = 512$ et gestion de la tension

3.3 Vers une architecture multi-standard

La figure 3.7 illustre les différentes normes OFDM selon l'écart inter-porteuses et le nombre de porteuses. La future norme de télédiffusion terrestre DVB-T2, dont le schéma de modulation OFDM/OQAM est candidat, y est aussi illustrée. Les différents types d'architectures y sont disposés selon leur domaine de performance. Pour les normes moyennement contraintes en terme de temps symbole (faible écart inter-porteuses) et de taille TFR, les architectures à base de pipeline peuvent être considérées. Leur grande régularité facilite l'implémentation, et ainsi réduit le temps de conception. Pour les normes fortement contraintes en terme de temps symbole et à faible nombre de porteuses telle que l'UWB, les architectures en pipelines-parallèles offrent les meilleures performances. Toutefois, les architectures à base de mémoires et à grand degré de parallélisme offrent les meilleures performances pour les normes à grand nombre de porteuses. La tendance des futures normes à utiliser un plus grand nombre de porteuses, comme c'est le cas pour la future norme DVB-T2 où jusqu'à 32768 porteuses peuvent être utilisées, renforce le choix des architectures à base de mémoires pour des grandes tailles de TFR. Ces dernières offrent aussi un très bon compromis dans les autres cas de figures et encore plus si une gestion de la tension d'alimentation est possible.

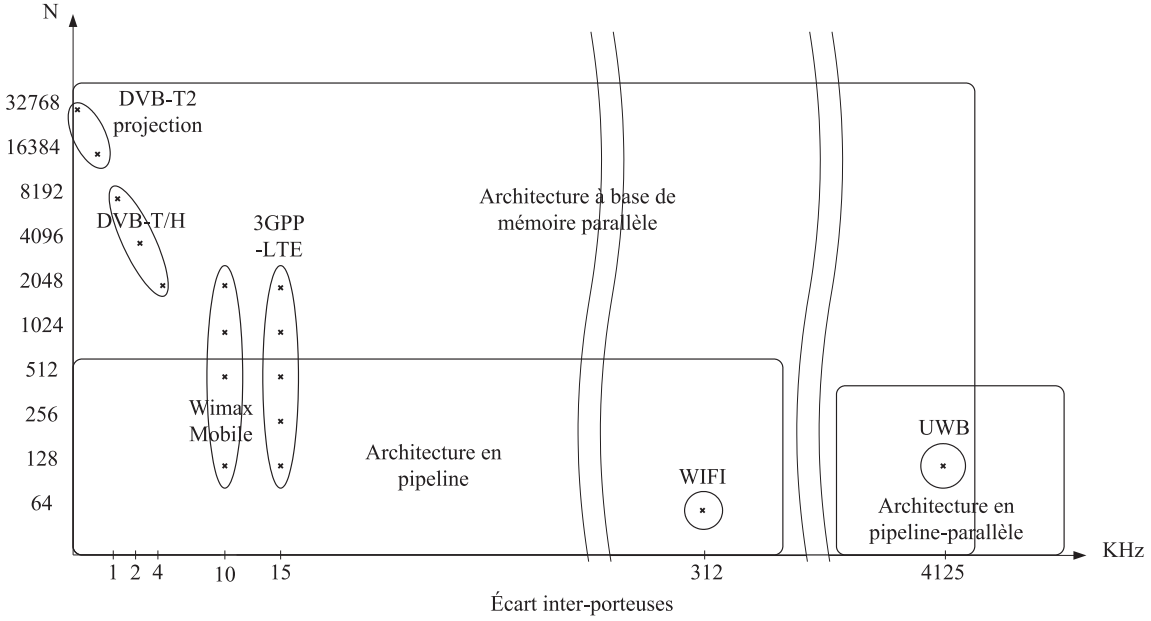

FIGURE 3.7 – Architectures selon différents écarts inter-porteuses et nombre de porteuses pour l'OFDM/QAM

Outre le fait que l'approche à base de mémoires permet de limiter le nombre de ressources arithmétiques, elle permet en plus de limiter le nombre d'accès mémoires lorsqu'elles sont associées à des algorithmes à degré élevé. Ainsi, à titre d'exemple, une architecture à base de mémoires radix-2^2 de la figure 2.16 aura $N \times \frac{\log_2 N}{2}$ accès mémoires pour une TFR à N points tandis qu'une architecture en pipeline nécessitera $N \times \log_2 N$ accès mémoires. De plus, les architectures en pipeline utilisent des mémoires à base de FIFO, ce qui engendre une surface beaucoup plus grande que des mémoires RAMs utilisées pour les architectures à base de mémoires, surtout pour des grandes valeurs de N [109].

En terme de capacité mémoire, le tableau 2.5 montre que les solutions SDF nécessitent la plus petite capacité mémoire parmi les architectures en pipeline, soit N points complexes. À cela, il faut ajouter deux mémoires RAM à N points complexes pour les buffers de sortie, soit une capacité totale de $3N$ données complexes . Dans chaque mémoire, les échantillons à la sortie de la TFR sont écrits dans un ordre *bit-reversed* et lus dans un ordre croissant naturel. Les deux buffers sont utilisés alternativement pour les différentes TFR successives. Dans le cas des architectures à mémoires, il existe des solutions à base d'une mémoire de traitement et à deux mémoires de traitement. Dans le premier cas, les différentes TFR successives utilisent alternativement un algorithme DIF et DIT afin d'éviter les conflits en écritures/lectures entre l'acquisition des N échantillons à l'entrée et l'envoi des N échantillons à la sortie. L'utilisation de deux mémoires permet d'alterner le traitement de la TFR et d'éviter les conflits d'écritures/lectures entre les différentes TFR. Toutefois, pour les deux approches, lorsque la fréquence de traitement est supérieure à la fréquence d'échantillonnage (d'acquisition des échantillons), il est alors nécessaire d'utiliser des buffers de sortie. Ces derniers permettent d'écrire les données à la fréquence de traitement et de les lire à la fréquence d'échantillonnage.

La modulation OFDM/OQAM nécessite un temps de traitement plus rapide que l'OFDM/QAM pour le même débit utile. Les normes présentées sur la figure 3.7 dans le cas de l'OFDM/OQAM se situeront donc plus à droite dans la plan écart interporteuse versus nombre de points.

L'analyse effectuée dans ce chapitre a été effectuée en considérant que chaque architecture ((1) à (6)) réalisent une seule norme. Dans un contexte multi-standard, une reconfigurabilité est nécessaire pour les architectures. Les architectures en pipeline sont très régulières et peuvent être facilement configurables. Ainsi, il est possible de concevoir une TFR en pipeline de 8192 points grâce à $\log_2 8192 = 14$ étages de calcul et de court-circuiter les étages non nécessaires pour les tailles de TFR plus petites. Néanmoins, ce type d'architecture est efficace pour des applications de l'ordre de quelque dizaines de MHz de largeur de bande. Ces architectures engendrent aussi un plus grand nombre de ressources arithmétiques du fait qu'elles doivent être conçues pour N_{max}. Ceci est d'autant plus important si l'on considère une architecture en pipeline-parallèle dans le cas de l'UWB.

De plus, dans l'hypothèse d'un traitement continu de la modulation OFDM/OQAM à très large bande pour l'UWB utilisant une architecture en pipeline-parallèle avec un degré de parallélisme P, l'architecture du filtre de mise en forme doit aussi utiliser une stratégie parallèle. Par conséquent, $2L \times P$ multiplieurs réels et $2L \times P$ additionneurs réels (pour la partie réelle et autant pour la partie imaginaire) doivent être utilisés.

Dans ce contexte de très haut débit et d'applications multi-standard, les architectures en pipelines sont donc moins efficaces que celles à base de mémoires. En effet, la reconfigurabilité des architectures à mémoires ne nécessite pas de ressources arithmétiques supplémentaires du fait du multiplexage temporel des opérations. Ceci nous permet aussi d'intégrer le filtrage de mise en forme en réutilisant des ressources arithmétiques disponibles pour le traitement de la TFR. Cette réutilisation des ressources n'est pas possible pour les architectures en pipeline de la TFR, du fait que les ressources arithmétiques sont utilisées sans discontinuité.

Conclusion

Dans cette première partie du manuscrit de thèse, nous avons présenté l'état de l'art des architectures OFDM avancées. Nous avons montré en quoi consiste la modulation OFDM d'un point de vue matériel et l'influence du filtrage du mise en forme dans le cas particulier de l'OFDM/OQAM.

Nous avons aussi illustré au chapitre l'influence du choix de l'algorithme de la TFR sur les différentes architectures possibles et montré comment obtenir une bonne adéquation algorithme architecture pour profiter à la fois des performances arithmétiques de l'algorithme et des avantages en termes de débit, surface et consommation des architectures. Nous avons montré que l'association des algorithmes de TFR radix-2^i avec des architectures en pipeline ou à mémoires offrent les meilleures performances. Il est à noter que dans cette analyse, les puissances relatives des différents éléments, les multiplieurs, les additionneurs et les mémoires, sont à des échelles de grandeur différentes. Ainsi, il est difficile de comparer les contributions de ces différents éléments, à la consommation totale de l'architecture. En termes de surface, nous nous intéressons plutôt au nombre de multiplieurs ou d'additionneurs qu'à la surface totale engendrée. Cette analyse nous a permis de déterminer les performances des trois principaux éléments contribuant grandement à la consommation d'une architecture donnée. Les autres éléments composant une architecture, tel que le contrôle, n'ont pas été abordés, du fait de leur contribution moindre en termes de puissance consommée.

Pour répondre à la contrainte de reconfigurabilité de l'architecture pour des applications multi-standard, notre choix s'est porté sur les architectures à mémoires radix-2^i et à grand degré de parallélisme qui offrent un meilleur compromis entre débit, reconfiguration et basse consommation que les architectures en pipeline ou pipeline-prallèle. En effet, le multiplexage temporel des architectures à mémoires nous permet de limiter les ressources arithmétiques peu importe la taille de la TFR. Ce choix résout en même temps l'intégration du filtrage du mise en forme en réutilisant les ressources de la TFR. Ainsi, nous obtenons un processeur de TFR-Filtre à base de mémoires répondant à nos très fortes contraintes.

Dans la prochaine partie, nous allons expliquer la conception d'une telle architecture. Nous y présenterons les différentes solutions permettant de résoudre les différents obstacles à sa réalisation.

Deuxième partie

Architecture d'un modulateur OFDM avancé haut-débit, reconfigurable, et basse consommation

Chapitre 4

Approche système

Dans ce chapitre, nous introduisons une architecture optimisée pour la modulation OFDM avancée pour des communications large bande et multi standard. Elle permet de réaliser soit une modulation OFDM/QAM classique qui consiste en une TFR, soit une modulation OFDM/OQAM qui nécessite en plus de la TFR, un filtrage de mise en forme. La solution proposée est une architecture à base de mémoires qui exploite une stratégie de réutilisation des ressources et combine à la fois un parallélisme à gros grains et à grains fins. En plus du parallélisme, elle utilise une approche en pipeline dans le traitement des données afin d'augmenter le multiplexage spatial permettant ainsi de traiter un plus grand nombre de données à la fois. Elle permet avec les mêmes ressources du circuit, de réaliser soit une TFR, soit le filtrage de mise en forme par la fonction IOTA.

Le processeur OFDM avancé est paramétrable pour des TFR de taille 64 points à 8192 points et peut réaliser un filtrage de mise en forme avec des longueurs de troncature égales à 2, 4 ou 8. Jusqu'à quatre modulations peuvent être réalisées en même temps dans le cas d'un système MIMO. Il utilise l'algorithme radix-2^i avec un entrelacement en fréquence pour la TFR afin de diminuer le nombre de multiplications complexes tout en gardant un chemin de données identique.

Afin de générer un signal OFDM pour différents standards, plusieurs reconfigurations à l'intérieur de l'architecture sont possibles. Ces reconfigurations se font à différents niveaux incluant les mémoires, les interconnexions, et la matrice de calcul qui permet de traiter les données. Ces reconfigurations sont dictées par le nombre de points à traiter, la longueur de troncature de la fonction IOTA et le mode MIMO.

L'utilisation d'une architecture à mémoires et la réutilisation des ressources disponibles permettent de limiter les ressources arithmétiques et diminuent grandement la taille du processeur. Selon la configuration choisie, seules les ressources nécessaires sont utilisées tandis que les ressources non nécessaires sont désactivées. De plus, grâce au degré de parallélisme, le processeur peut générer un débit symbole de plusieurs centaines de Mb/s, tout en limitant la consommation d'énergie. La solution offre un bon compromis entre reconfigurabilité, performance et consommation.

La figure 4.1 illustre l'architecture système de la solution. On peut diviser l'architecture en quatre grandes parties :

1. Les ressources mémoires spécifiques à la TFR qui englobent les mémoires des échantillons et des coefficients de rotation W.

2. Les ressources mémoires spécifiques au filtrage IOTA qui englobent les mémoires des échantillons et des coefficients de la fonction IOTA.
3. Les ressources arithmétiques formant une matrice de calcul, qui permettent de réaliser à la fois l'algorithme de la TFR et le filtrage par la fonction IOTA.
4. Le module de contrôle permettant de gérer le fonctionnement global de l'architecture selon les différents paramètres.

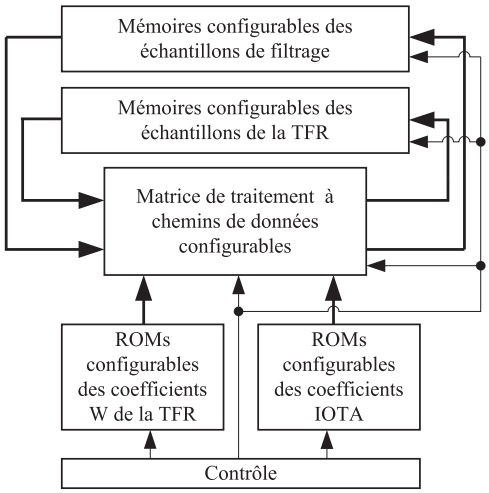

FIGURE 4.1 – Architecture simplifiée du processeur OFDM avancé proposé

4.1 Caractéristiques de l'architecture

Cette section illustre les principales caractéristiques de l'architecture, à savoir sa reconfigurabilité, son parallélisme et sa stratégie basse consommation.

Architecture reconfigurable

La reconfiguration de l'architecture se fait à plusieurs niveaux et selon trois paramètres à savoir le nombre de points N, la longueur de troncature L, et le mode MIMO. Les reconfigurations se situent aux niveaux suivants :

- *Les mémoires* :
  - Les tailles des mémoires varient de 64 à 8192 points. Selon le paramètre N, on désactive les mémoires non nécessaires. De plus, pour les mémoires de filtrage, certaines mémoires sont désactivées en plus, selon le paramètre L. Les mémoires sont donc constituées de bancs de mémoires concaténables grâce au vecteur d'adresse.
- *Les modules de calculs* :
  - Les interconnexions entre les différents modules de calculs constituant la matrice représentent le premier niveau de configuration au niveau de la matrice de calcul. Ceci nous permet de reproduire les deux flots de données nécessaires pour un modulateur OFDM avancé, à savoir le flot de donnée pour le traitement de la TFR et celui pour le filtrage par la fonction IOTA.
  - Il existe une reconfiguration aussi selon l'algorithme utilisé. La classe d'algorithmes de la TFR choisie est celle des algorithmes radix-2^i. La topologie du flot de données est

toujours la même quelque soit i (i varie de 1 à 3), et a la même régularité : il s'agit de celui d'une TFR radix-2. Seul le nombre de modules de calcul nécessaires varie. Par conséquent, selon l'algorithme utilisé, on désactive les modules de calculs non nécessaires.

– Finalement, le dernier niveau de reconfiguration se situe à l'intérieur même de chaque module de calcul. En effet, selon qu'on effectue une opération de TFR ou filtrage, les modules de calcul devront effectuer soit des opérations d'additions/multiplications complexes appelées opérateur papillon (cas de la TFR), soit des additions/multiplications réelles (cas du filtrage).

Cette réutilisation des modules de calcul pour le filtrage IOTA constitue un caractère innovant de l'architecture.

Architecture parallèle

Le degré de parallélisme est dicté par l'algorithme de TFR utilisé le plus élevé à savoir le radix-2^3. En effet, pour un algorithme radix-2^i, on manipule 2^i échantillons à la fois sur i étage. Dans le cas de notre architecture, on manipule 8 échantillons à l'entrée en même temps sur 3 étages maximum pour le mode TFR. On garde le même nombre de parallélisme pour le filtrage IOTA.

Ce taux de parallélisme nous permet aussi de réaliser 2 ou 4 TFRs en parallèle. En effet, dans le mode MIMO 2x2, on manipule 4 échantillons par TFR pour un total de 8 échantillons. Dans le cas du mode MIMO 4x4, on manipule 2 échantillons par TFR pour un total toujours égal à 8. Dans ce mode, le processeur est équivalent à 2 ou 4 modulateurs indépendants réalisant le même traitement sur différents vecteurs de données.

Architecture basse consommation

Le choix d'algorithme de haut degré pour la TFR est en lui-même un choix permettant de diminuer la consommation. Plus le degré est haut, plus le nombre de multiplications complexes triviales augmente. Ceci permet une consommation moindre. En effet, les multiplieurs sont un des composants les plus gourmands en termes d'énergie. Dans une optique de basse consommation, on tire parti des multiplications triviales : lorsque les facteurs de rotation sont égaux à ± 1 ou $\pm j$, on désactive les multiplieurs et les additionneurs dans les modules de calcul.

Le taux de parallélisme nous permet aussi d'opérer à des fréquences très basses et de ce fait, une tension réduite peut être appliquée au modulateur, nous permettant d'obtenir un gain de la puissance consommée.

De plus, les algorithmes de haut degré pour la TFR exigent un accès mémoire moins fréquent ce qui diminue par conséquent la consommation. Pour les algorithmes radix-2^i, on accède aux mémoires tous les i étages. Ainsi, pour le radix-2^3, on accède trois fois moins fréquemment aux mémoires que pour le radix-2, et deux fois moins pour le radix-2^2 toujours par rapport au radix-2.

Finalement, comme on l'a expliqué dans la partie reconfiguration, la désactivation de certaines mémoires ou modules de calcul selon les paramètres N, L et le mode MIMO permettent d'optimiser la consommation de notre architecture. Ceci est fait grâce à la désactivation des horloges (*clock gating*) pour un prototypage FPGA. Pour une implémentation sur circuit dé-

dié ASIC, il serait possible de couper complètement la tension des parties non utilisées, ce qui coupera toute consommation à la fois dynamique mais aussi et surtout statique.

4.2 Multiplexage temporel des opérations

Les architectures à base de mémoires utilisent un multiplexage temporel des opérations. Dans notre cas, un premier multiplexage consiste à réaliser la TFR et le filtrage en deux étapes successives. De plus, chacune de ces deux étapes est elle même réalisée en plusieurs passages dans la matrice de calcul. La figure 4.2 illustre l'ordonnancement des opérations pour une modulation OFDM/OQAM. Le mode de fonctionnement est illustré pour chacun des composants du modulateur, à savoir, les mémoires de la TFR, les mémoires de filtrage et la matrice de calcul.

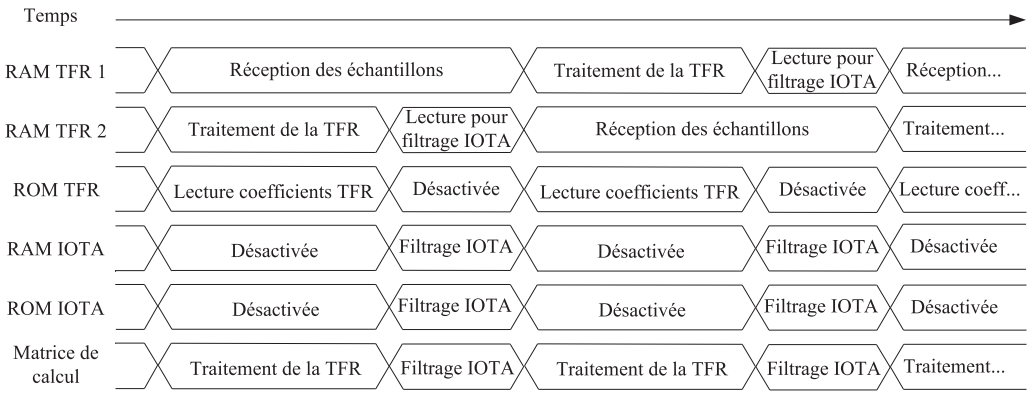

FIGURE 4.2 – Ordonnancement des opérations pour différents composants du processeur pour une modulation OFDM/OQAM

La mémoire des échantillons pour la TFR est composée de deux bancs de RAMs à N points. Chacun de ces deux bancs sert alternativement à la réception des N échantillons à l'entrée du modulateur et au traitement des données. Ainsi, lorsque la RAM 1 est en mode réception, des opérations d'écriture des échantillons sont effectuées à la fréquence d'échantillonnage. Des opérations d'écriture/lecture à la fréquence de traitement du processeur sont effectuées sur la RAM 2 lors du traitement de la TFR, et seules des opérations de lecture sont effectuées durant le mode filtrage (dans le cas d'une modulation OFDM/OQAM), toujours à la même fréquence, afin de transférer les résultats de la TFR vers la matrice pour l'opération de filtrage. Les rôles sont inversés à chaque nouvelle entrée de N échantillons. Les mémoires ROM des coefficients de rotation W servent seulement lors du traitement de la TFR et sont désactivées durant le mode IOTA et inversement pour les mémoires ROM des coefficients IOTA. Des opérations d'écriture/lecture sont réalisées sur les mémoires RAMs IOTA lors du filtrage de mise en forme et sont désactivées autrement.

La matrice de calcul reçoit durant le traitement de la TFR les échantillons d'une des deux mémoires RAMs de la TFR ainsi que les facteurs de rotations stockés dans les ROMs afin de réaliser une opération papillon radix-2^i. Durant le filtrage de mise en forme, la matrice de calcul reçoit les résultats de la TFR stockés dans une des deux mémoires RAMs de la TFR, les coefficients des mémoires ROMs IOTA ainsi que les échantillons filtrés précédemment mémorisés dans les RAMs de filtrage. Le chapitre suivant décrit en détail le fonctionnement et l'architecture de la matrice.

Chapitre 5

Matrice de calcul

Ce chapitre décrit l'architecture et le fonctionnement de la matrice de calcul. Elle constitue une des principales innovations du modulateur. Ce caractère innovant provient de sa reconfigurabilité élevée qui lui permet tout d'abord d'effectuer du multi-radix et le filtrage de mise en forme IOTA. Elle est constituée de 12 modules de calcul (MC) agencés en une matrice de 4 par 3. Les étages (colonne de la matrice) non nécessaires peuvent être désactivés grâce à la technique de *clock gating*. Chaque module de calcul peut être configuré afin d'être utilisé comme croisillon lors d'une TFR ou comme module de filtrage lors de la mise en forme IOTA. La matrice peut réaliser une TFR radix-2, radix-2^2 ou encore radix2^3.

Les premières sections illustrent de façon simplifiée et conceptuelle le fonctionnement de la matrice pour les différents modes, soit le mode TFR, le mode filtrage et le mode MIMO, pour une meilleure compréhension de l'architecture. Par la suite, une description détaillée et théorique de l'architecture de la matrice sera présentée pour les différents modes.

5.1 Fonctionnement de la matrice

5.1.1 Fonctionnement du mode TFR

La figure 5.1 illustre la configuration de la matrice pour le mode TFR et le mode SISO. Dans le cas de l'algorithme radix-2^3, toute la matrice est utilisée. Le flot de données correspond à une TFR à 8 points et décompose donc une TFR à N points (avec N puissance de 8) en une série de TFR à 8 points. Elle reçoit les coefficients de rotation tel qu'illustré à la figure B.5 de l'annexe B.

Afin de pouvoir réaliser toutes les tailles de la TFR, autres que les tailles puissance de 8, il est nécessaire d'opérer une première décomposition grâce à l'algorithme radix-2^2 pour les valeurs de $\log_2 N$ modulo 3 égales à 2, ou une première décomposition grâce à l'algorithme radix-2 pour les valeurs de $\log_2 N$ modulo 3 égales à 1. Ainsi, après chaque décomposition radix-2^2 ou radix-2, on obtient respectivement 4 ou 2 sous-TFR de taille puissance de 8 qui peuvent être traitées grâce à l'algorithme radix-2^3.

En désactivant la première colonne de la matrice, on obtient deux chemins de données d'une TFR à 4 points. Ceci correspond à deux papillons radix-2^2 en parallèle. La matrice reçoit donc le même nombre d'échantillons à chaque coup d'horloge que lors du radix-2^3. Les deux papillons reçoivent les coefficients de rotation de différents indices tel qu'illustré à la figure B.4. En désactivant les 2 premières colonnes de la matrice, on obtient 4 modules papillon en parallèle traitant

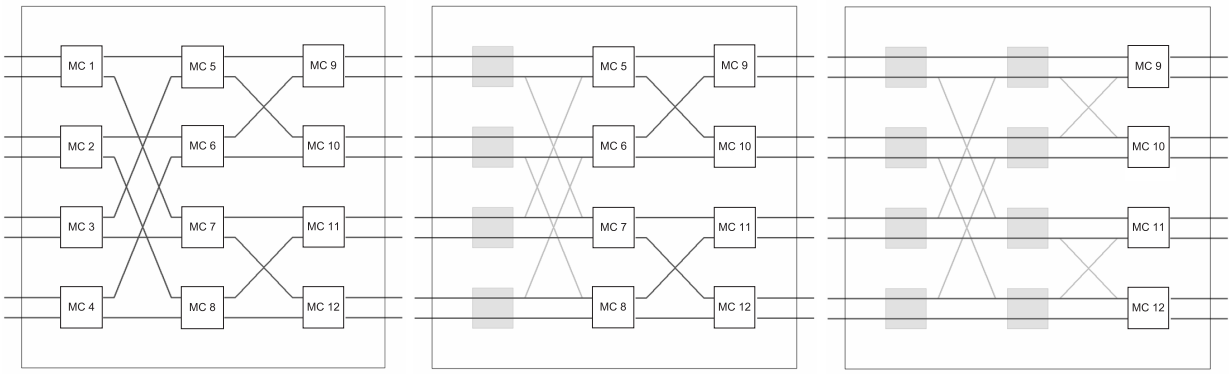

FIGURE 5.1 – Interconnexion des modules de calcul de la matrice en mode SISO pour les algorithmes de la TFR a) radix-2^3 b) radix-2^2 c) radix-2

toujours un total de 8 données et recevant les coefficients de rotation de différents indices tel qu'illustré à la figure B.1.

Les figures 5.2 illustrent l'ordonnancement des opérations pour la matrice pour les différents cas présentés. La TFR est réalisée en $\frac{\log_2 N}{3}$ passages dans la matrice pour N puissance de 8. Pour une valeur de $\log_2 N$ modulo 3 égale à 2 et 1, la TFR est réalisée en $\frac{\log_2(N/4)}{3}+1$ et $\frac{\log_2(N/2)}{3}+1$ passages respectivement dans la matrice. Chaque passage a une durée de $N/8$ coups d'horloge afin de traiter les N points de la TFR. Ainsi, on réalise $N/2$ croisillons à deux points sur $\log_2 N$ étages.

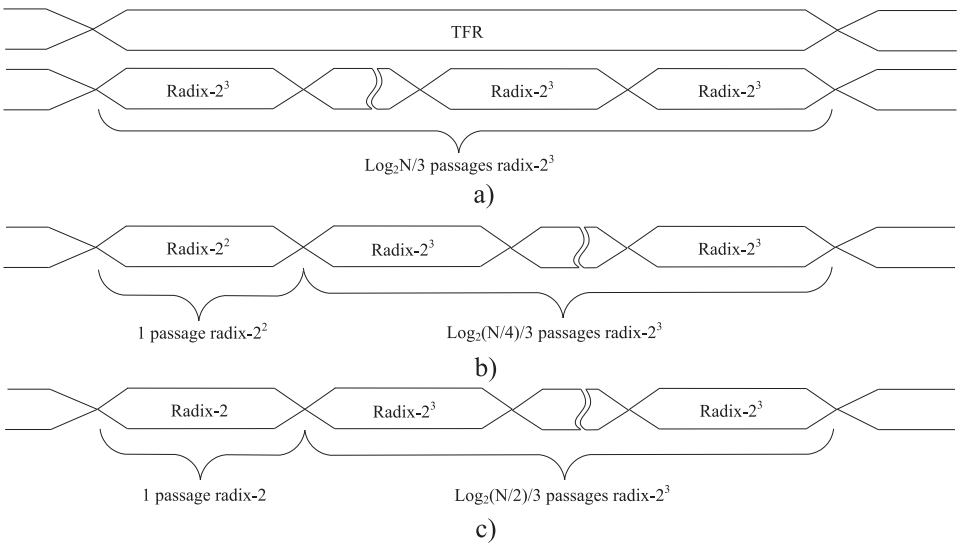

FIGURE 5.2 – Ordonnancement des opérations de la matrice en mode SISO pour une TFR à N points a) puissance de 8 (64, 512, 4096) b) $\log_2 N$ modulo 3 égale à 2 (256, 2048) c) $\log_2 N$ modulo 3 égale à 1 (128, 1024, 8192)

5.1.2 Fonctionnement du mode filtrage de mise en forme

L'opération de filtrage de mise en forme consiste en des multiplications/additions. La première colonne de la matrice ne contient que des additionneurs du fait qu'aucun algorithme radix-2^i utilisé n'a besoin de multiplication par les facteurs de rotation à la première colonne. Seuls les modules de calcul des deux dernières colonnes sont utilisés tandis que la première colonne est toujours inactive. Certains multiplieurs ont été ajoutés dans ces modules de calcul afin de réaliser le filtrage de mise en forme. Ils ne sont donc actifs que dans ce mode.

Chaque module de calcul est indépendant et filtre un certain nombre d'échantillons des résultats de la TFR. Deux chemins de données coexistent donc dans la matrice. Un pour le traitement de la TFR et le deuxième pour le filtrage. Pour ce dernier cas, chaque module est connecté aux mémoires des échantillons de TFR à filtrer, aux mémoires ROMs des coefficients de la fonction IOTA, ainsi qu'aux mémoires des échantillons précédents filtrés.

Le filtrage de mise en forme nécessite $2 \times 2LM$ multiplications/additions, en tenant compte des parties réelles et imaginaires, pour chaque symbole OFDM/OQAM émis. La figure 5.3 illustre un schéma simplifié de la matrice en mode filtrage. Comme on le constate, les modules de calcul 5, 6, 9 et 10 filtrent la partie réelle des échantillons de la TFR, tandis que les modules de calcul 7, 8, 11, et 12 filtrent la partie imaginaire. Selon la valeur de L, $2L$ modules de calcul sont utilisés afin de filtrer les parties réelles et imaginaires des échantillons. Ainsi, lorsque $L = 2$, les 4 modules de la dernière colonne sont utilisés et les autres colonnes sont désactivées. Pour $L = 4$, les 8 modules des deux dernières colonnes sont utilisés. Le cas $L = L_{max} = 8$ est une exception. En effet, pour ce cas de figure, 16 modules de calcul auraient été nécessaires. Afin de limiter les ressources, le cas $L = L_{max}$ est réalisé par multiplexage temporel en deux grandes étapes. Ce cas de figure utilise donc les mêmes ressources que le cas $L = 4$ mais d'une durée deux fois plus longue. Chaque module de calcul filtre $\frac{4LM}{2L} = 2M = N$ échantillons au total à un rythme de 8 échantillons à chaque coup d'horloge pour L égal à 2 ou 4. Le filtrage est réalisé donc en $\frac{N}{8}$ cycles. Dans le cas $L = L_{max}$, le filtrage est réalisé en $\frac{N}{4}$ cycles.

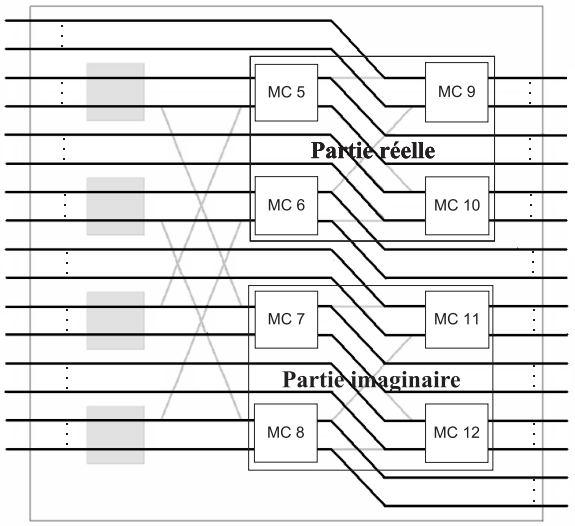

FIGURE 5.3 – Schéma simplifié de la matrice en mode filtrage de mise en forme

5.1.3 Fonctionnement du mode MIMO

Le mode MIMO tire parti du parallélisme des papillons radix-2^2 et radix-2 présents dans la matrice afin de réaliser plusieurs TFR en parallèles. En effet, comme nous l'avons montré plus tôt, il existe deux papillons radix-2^2 et 4 papillons radix-2 en parallèle dans la matrice. L'idée est qu'au lieu d'utiliser ces modules en parallèle pour une TFR, on les utilise pour réaliser plusieurs TFR en parallèle mais avec un parallélisme des papillons de moindre degré. En effet, afin de réaliser le mode MIMO 2x2, on utilise les 2 papillons radix-2^2 séparément, chacun traitant une TFR différente. Chaque papillon reçoit les mêmes coefficients de rotations mais différents échantillons d'entrée. Puisque l'algorithme radix-2^2 traite des tailles de TFR puissance de 4 seulement, un premier passage radix-2 est réalisé grâce aux modules de la dernière colonne de la matrice pour les tailles de TFR puissance de 2 seulement. Dans le cas du MIMO 4x4, on utilise les 4 papillons radix-2 séparément, chacun traitant une TFR différente comme pour le cas MIMO 2x2. La figure 5.4 illustre les différentes configurations de la matrice dans le mode MIMO lors de la TFR.

La figure 5.5 illustre l'ordonnancement des opérations dans la matrice pour le mode MIMO. Pour le cas du MIMO 2x2, 2 TFRs sont réalisées en $\frac{\log_2 N}{2}$ passages dans la matrice pour N puissance de 4. Pour N puissance de 2 seulement, les 2 TFRs sont réalisées en $\frac{\log_2(N/2)}{2} + 1$ passages dans la matrice. Chaque passage a une durée de $N/4$ cycles. Finalement, dans le mode MIMO 4x4, 4 TFRs sont réalisées en $\log_2 N$ passages dans la matrice d'une durée de $N/2$ cycles chacun.

Il est à mentionner que du fait que les mémoires sont conçues pour $N_{max} = 8192$ points, les tailles maximales des TFRs pour le mode MIMO 2x2 est de 4192 points et 2048 points pour le mode MIMO 4x4.

La configuration de la matrice pour le mode MIMO lors du filtrage de mise en forme est semblable au cas SISO. En effet, dans ce dernier cas, chaque module de calcul traitait 8 échantillons de la TFR à chaque cycle. Dans les cas MIMO 2x2 et 4x4, chaque module reçoit 8 échantillons de 2 et 4 TFRs différentes respectivement. De plus, une gestion des coefficients IOTA (lors de leur lecture) est effectuée pour réaliser les 2 ou 4 filtrages en parallèle.

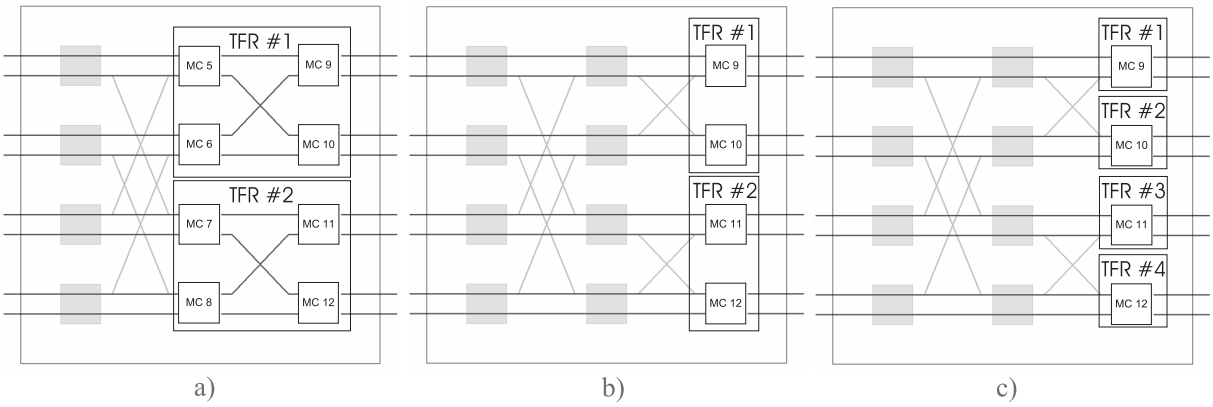

FIGURE 5.4 – Configuration de la matrice pour le mode a) MIMO 2x2 radix-2^2 b) MIMO 2x2 radix-2 c) MIMO 4x4 radix-2

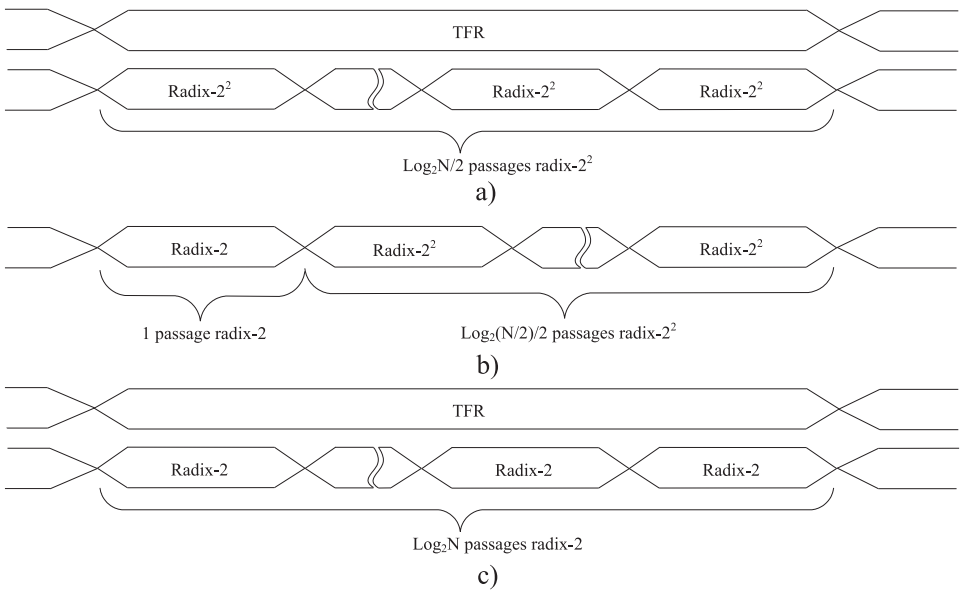

FIGURE 5.5 – Ordonnancement des opérations de la matrice pour le mode a) MIMO 2x2 et N puissance de 4 b) MIMO 2x2 et N puissance de 2 seulement c) MIMO 4x4

5.2 Description détaillée de la matrice

Dans cette section, nous présentons le cheminement qui a dicté au choix du degré maximal de l'algorithme radix-2^i. Nous verrons que le choix du degré de l'algorithme dicte le parallélisme de l'architecture, ainsi que les ressources nécessaires.

Le standard UWB est la contrainte la plus élevée en terme de puissance de calcul. L'objectif est de voir si une architecture reconfigurable, offrant un certain degré de flexibilité, permet d'atteindre des débit aussi élevés. L'UWB requiert la réalisation d'une TFR à 128 points dans une largeur de bande de 528 MHz, soit le calcul de la TFR en 242,42 ns pour une modulation OFDM/QAM classique. Dans le cas de la modulation OFDM/OQAM, il est nécessaire de réaliser la TFR et le filtrage de mise en forme dans un laps de temps deux fois plus rapide du fait de l'orthogonalité réelle de la fonction prototype IOTA. Afin d'éviter le recouvrement des traitements des symboles OFDM, le temps de traitement requis pour la modulation OFDM/OQAM doit être inférieur ou égal au temps d'acquisition des échantillons à l'entrée. Nous avons alors pour un algorithme radix-r :

$$\overbrace{f_{op}\frac{N}{rP_{P,radix}}\log_r N}^{TFR} + \overbrace{f_{op}\frac{2ML}{rP_{P,radix}}}^{Filtrage} \leq \overbrace{f_{op}N}^{Acquisition} \tag{5.1}$$

Où $P_{P,radix}$ représente le parallélisme des papillons radix-r, soit le nombre de papillons radix-r en parallèle. Le tableau 5.1 donne le nombre de papillons en parallèle pour un algorithme radix-r, différentes valeurs de L, et une TFR à 128 points, soit le cas de l'UWB. La fréquence d'opération du circuit à été limitée à la fréquence d'échantillonnage, soit $f_{op} = f_{echant}$. On constate que ce cas de figure nécessite seulement un module radix-16 ou un module radix-8 sauf pour $L = 8$. Toutefois, le radix-16 nécessite des ressources matérielles plus importantes que le radix-8. Pour cette raison, le degré de l'algorithme utilisé a été limité au radix-8, soit en réalité le radix-2^3 ($r = 8 = 2^{i_{max}}$, $i_{max} = 3$, $P_{P,R2^3} = 1$).

TABLE 5.1 – Parallélisme des papillons $P_{P,R2^i}$ pour différents algorithmes-r, valeurs de L, $N = 128$ et $f_{op} = f_{echant}$

Radix \ L	2	4	8
2	5	6	8
4	2	2	3
8	1	1	2
16	1	1	1

5.2.1 Approche théorique du mode TFR

L'architecture proposée exploite plusieurs niveaux de parallélisme. Pour une meilleure compréhension, nous allons d'abord les présenter en détail.

Le premier niveau est le parallélisme d'un papillon de l'algorithme utilisé P_{radix}. Ainsi, pour un algorithme radix-2^i, on manipule $P_{radix} = r = 2^i$ échantillons par cycle. En second, nous avons le parallélisme des papillons $P_{p,radix}$ du tableau 5.1, soit le nombre de papillons radix-2^i en parallèle. Le troisième niveau de parallélisme détermine le nombre de données manipulées pour une TFR, soit $P_{D_{TFR}} = P_{radix}P_{p,radix}$. Selon le mode MIMO utilisé, plusieurs TFRs peuvent être réalisées simultanément, nous avons donc un parallélisme du nombre de TFR, identifié par P_{MIMO}. Finalement, le parallélisme global des données à l'entrée de la matrice à chaque cycle est de $P_D = P_{radix}P_{p,radix}P_{MIMO} = P_{D_{TFR}}P_{MIMO}$. Le tableau 5.2 illustre les différents niveaux de parallélisme selon la configuration.

TABLE 5.2 – Divers degrés de parallélisme selon la configuration de la matrice

Configuration	P_{radix}	$P_{p,radix}$	$P_{D_{TFR}}$	P_{MIMO}	P_D
SISO radix-2^3	8	1	8	1	8
SISO radix-2^2	4	2	8	1	8
SISO radix-2	2	4	8	1	8
MIMO 2x2 radix-2^2	4	1	4	2	8
MIMO 2x2 radix-2	2	2	4	2	8
MIMO 4x4 radix-2	2	1	2	4	8

La matrice de l'architecture proposée est donc caractérisée par une grande scalabilité permettant au système de favoriser soit un haut degré de parallélisme des données pour une TFR, soit un plus grand nombre de TFR. Dans tous les cas, le parallélisme global des données P_D reste inchangé et est égal à 8.

Pour une algorithme radix-2^i, un papillon nécessite $\frac{P_{radix}}{2}$ modules de calcul sur i étages, c'est-à-dire $\sharp MC_{R2^i,papillon} = \frac{2^i}{2}i$. Le rapport $\frac{1}{2}$ découle du fait que chaque module de calcul manipule 2 échantillons à la fois. Le tableau 5.3 donne le nombre de modules de calcul nécessaires selon les différentes configurations. $\sharp MC_{TFR}$ et $\sharp MC_{MIMO}$ correspondent respectivement au nombre de modules de calcul pour une TFR et selon le mode MIMO, c'est-à-dire en tenant compte du parallélisme des papillons et du parallélisme des TFR pour le mode MIMO.

Le tableau 5.3 illustre un autre type de parallélisme, le parallélisme spatiale. En effet, après propagation des données dans la matrice et atteinte d'un régime stationnaire, P_D échantillons

TABLE 5.3 – Nombre de modules de calcul nécessaires selon les différentes configurations

Configuration	$\sharp MC_{R2^i,papillon} = \frac{2^i}{2}i$	$\sharp MC_{TFR} = \sharp MC_{R2^i,papillon} P_{p,radix}$	$\sharp MC_{MIMO} = \sharp MC_{TFR} P_{MIMO}$
SISO radix-2^3	12	12	12
SISO radix-2^2	4	8	8
SISO radix-2	1	4	4
MIMO 2x2 radix-2^2	4	4	8
MIMO 2x2 radix-2	1	2	4
MIMO 4x4 radix-2	1	1	4

sur i étages, pour un algorithme radix-2^i, sont manipulés par la matrice. L'approche en pipeline de la matrice permet d'augmenter la vitesse de traitement des échantillons.

Le fait de conserver le même degré de parallélisme pour une TFR ($P_{D_{TFR}}$) quel que soit l'algorithme utilisé nous permet de faciliter la génération des adresses et de limiter la complexité de contrôle de l'architecture. En effet, les indices x_p des $P_{D_{TFR}}$ échantillons sont à équidistance et séparés par $\frac{N}{P_{D_{TFR}}}$ indices. On obtient alors :

$$x_p = x + p\frac{N}{P_{D_{TFR}}} \tag{5.2}$$

avec $0 \leq x_p \leq N-1$, $0 \leq x \leq \frac{N}{P_{D_{TFR}}} - 1$ et $0 \leq p \leq P_{D_{TFR}} - 1$. De ce fait, la valeur de l'indice x correspond à la valeur du compteur de papillon.

La figure 5.6 donne un schéma détaillé de la matrice en mode TFR. Étant donné que les entrées-sorties pour le filtrage sont séparées des entrées-sorties pour la TFR au niveau de chaque module, le multiplexage de données au niveau de la matrice se limite à envoyer les échantillons d'entrée vers les modules concernés selon le radix en cours. On a donc des multiplexeurs au niveau des entrées échantillons des modules 5 à 12 (pour les colonnes 2 et 3) pour sélectionner la provenance des données, soit des entrées de la matrice, soit de l'étage précédant.

Le contrôle de la matrice est réalisé grâce aux signaux *TFR_IOTA_mode* représenté sur 1 bit et qui détermine le mode de traitement de la matrice, *Radix* représenté sur 2 bits, et finalement *Dernier_étage* représenté aussi sur 1 bit et qui indique s'il s'agit du dernier étage du traitement de la TFR. En effet, le dernier étage d'une TFR n'implique que des multiplications triviales par 1 et par conséquent les multiplieurs peuvent être désactivés.

5.2.2 Approche théorique du mode filtrage de mise en forme

Lors du filtrage de mise en forme, chaque échantillon de résultat de la TFR subit L multiplications réelles et L additions réelles à la fois pour la partie réelle et imaginaire de l'échantillon comme expliqué à la section 2.4. Nous avons donc besoin de $2LP_D$ multiplieurs et additionneurs réels afin de traiter les $P_D = 8$ échantillons à l'entrée de la matrice. Soit 32 multiplieurs/additionneurs pour $L = 2$, 64 pour $L = 4$ et 128 pour $L_{max} = 8$. Nous verrons dans la prochaine section que chaque module de calcul est composée de 8 paires de multiplieurs/additionneurs réels. Chaque module de calcul reçoit les $P_D = 8$ échantillons et réalise une opération de pondération/sommation pour une valeur de ℓ, avec $0 \leq \ell \leq L-1$. Ainsi, le

filtrage nécessite $2L$ modules de calcul. Pour $L = L_{max}$, le filtrage est dans un premier temps effectué pour $0 \leq \ell \leq \frac{L_{max}}{2} - 1$, et dans un deuxième temps pour $\frac{L_{max}}{2} \leq \ell \leq L_{max} - 1$. On peut noter que pour $L = 2$ $(0 \leq \ell \leq 1)$, seuls les MCs de la dernière colonne sont utilisés comme mentionné précédemment.

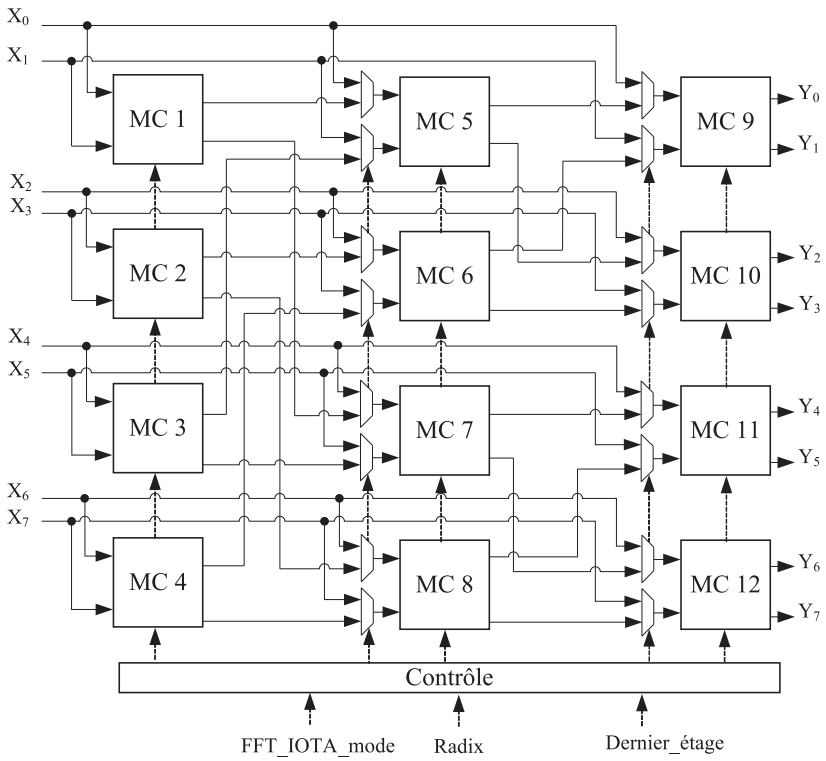

FIGURE 5.6 – Schéma détaillé de la matrice en mode TFR

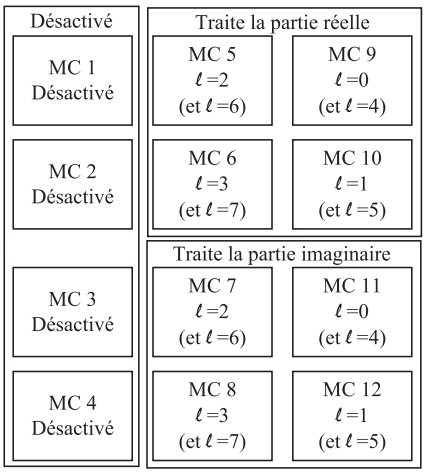

FIGURE 5.7 – Répartition des opérations de pondération/sommation dans la matrice selon la valeur de ℓ lors du filtrage

5.2.3 Architecture des modules de calcul

Chaque module de calcul de la première colonne est composé de deux additionneurs complexes soit 4 additionneurs réels au total. Pour les 2 dernières colonnes, les modules de calcul sont composés de 2 additionneurs complexes et 2 multiplieurs complexes afin de réaliser un croisillon radix-2^i de la TFR. En utilisant un schéma de 4 multiplieurs réels et 2 additionneurs réels pour la multiplication complexe, on obtient un total de 8 additionneurs/multiplieurs réels par modules de calcul pour les 2 dernière colonnes.

Dans le cas du filtrage de mise en forme IOTA, l'équation 2.6 donne l'expression du signal émis, c'est-à-dire les échantillons du résultat de la TFR à chaque temps symbole j filtrés par les M premiers coefficients de la fonction IOTA ($\Im_0$ à $\Im_{M-1}$). Si nous considérons tous les échantillons filtrés (soient les échantillons émis et les échantillons mémorisés) à chaque temps symbole j, nous obtenons $2ML$ échantillons à l'indice $g(k,s,\ell) = n+2\ell M = k+sM+2\ell M$ avec $0 \leq g(k,s,\ell) \leq 2ML-1$, $0 \leq k \leq M-1$ et $0 \leq n \leq N-1$. Le paramètre s indique s'il s'agit des M premiers ou des M derniers échantillons du résultat de la TFR et est donc égale à $0 \leq s \leq 1$. En insérant $g(k,s,\ell)$ dans l'équation 2.6, on obtient l'expression générale des N échantillons du résultat de la TFR filtrés par la fonction prototype IOTA.

$$F_{j,g(k,s,\ell)} = \sum_{q=0}^{2L-1} \left[\alpha_{q,g(k,s,\ell)} C_{j-q,k+sM} + \beta_{q,g(k,s,\ell)} C_{j-q,k+(-1)^s M+M}\right] \tag{5.3}$$

avec

$$\alpha_{q,g(k,s,\ell)} = \begin{cases} 0 & \text{si } q \text{ impair} \\ \Im_{k+sM+2\ell M+qM} & \text{si } q \text{ pair} \end{cases} \qquad \beta_{q,g(k,s,\ell)} = \begin{cases} \Im_{k+sM+2\ell M+qM} & \text{si } q \text{ impair} \\ 0 & \text{si } q \text{ pair} \end{cases} \tag{5.4}$$

Notez que $\alpha_{q,g(k,s,\ell)}$ et $\beta_{q,g(k,s,\ell)} = 0$ si $k+sM+2\ell M+qM \geq 2ML$ et $0 \leq q \leq 2L-1$, soit les $2L$ fonctions IOTA superposées.

L'expression du signal émis $S_{j,k}$ de l'équation 2.6 représente le cas $F_{j,g(k,0,0)}$. Considérant l'approche parallèle de la matrice en insérant l'équation 5.2 dans 5.3 et en isolant les échantillons filtrés au temps symbole j (c'est-à-dire pour $q=0$) des échantillons filtrés précédemment(soit $q>0$) et mémorisés dans les mémoires de filtrage, on obtient :

$$F_{j,g(x,p,\ell)} = \underbrace{\Im_{x+p\frac{N}{P_{D_{TFR}}}+2M\ell} C_{j,x+p\frac{N}{P_{D_{TFR}}}}}_{F_{j,g(x,p,\ell)}|_{q=0}} + F_{j,g(x,p,\ell)}|_{q>0} \tag{5.5}$$

Cette expression représente l'opération de filtrage à chaque temps symbole j, c'est-à-dire la multiplication des échantillons du résultat de la TFR $C_{j,x+p\frac{N}{P_{D_{TFR}}}}$ par les coefficients de la fonction IOTA $\Im_{x+p\frac{N}{P_{D_{TFR}}}+2M\ell}$ ainsi que la sommation par les échantillons filtrés au temps symbole précédent $F_{j,g(x,p,\ell)}|_{q>0}$. La figure 5.8 illustre cette opération selon la valeur des indices x, p et ℓ. L'opération de pondération/sommation est réalisée par la paire de multiplieur/additionneur p du module de calcul traitant une valeur de ℓ comme expliqué à la section 5.2.2. Il est alors possible d'exprimer l'équation 5.5 en fonction des ressources disponibles :

$$\begin{aligned} &RAM\_Filtre_{ecriture}(F_{j,g(x,p,\ell)}) = \\ &MC_{(\ell,p)}\left[C_{j,x+p\frac{N}{P_{D_{TFR}}}}, \Im_{x+p\frac{N}{P_{D_{TFR}}}+2M\ell}, RAM\_Filtre_{lecture}\left(F_{j,g(x,p,\ell)}|_{q>0}\right)\right] \end{aligned} \tag{5.6}$$

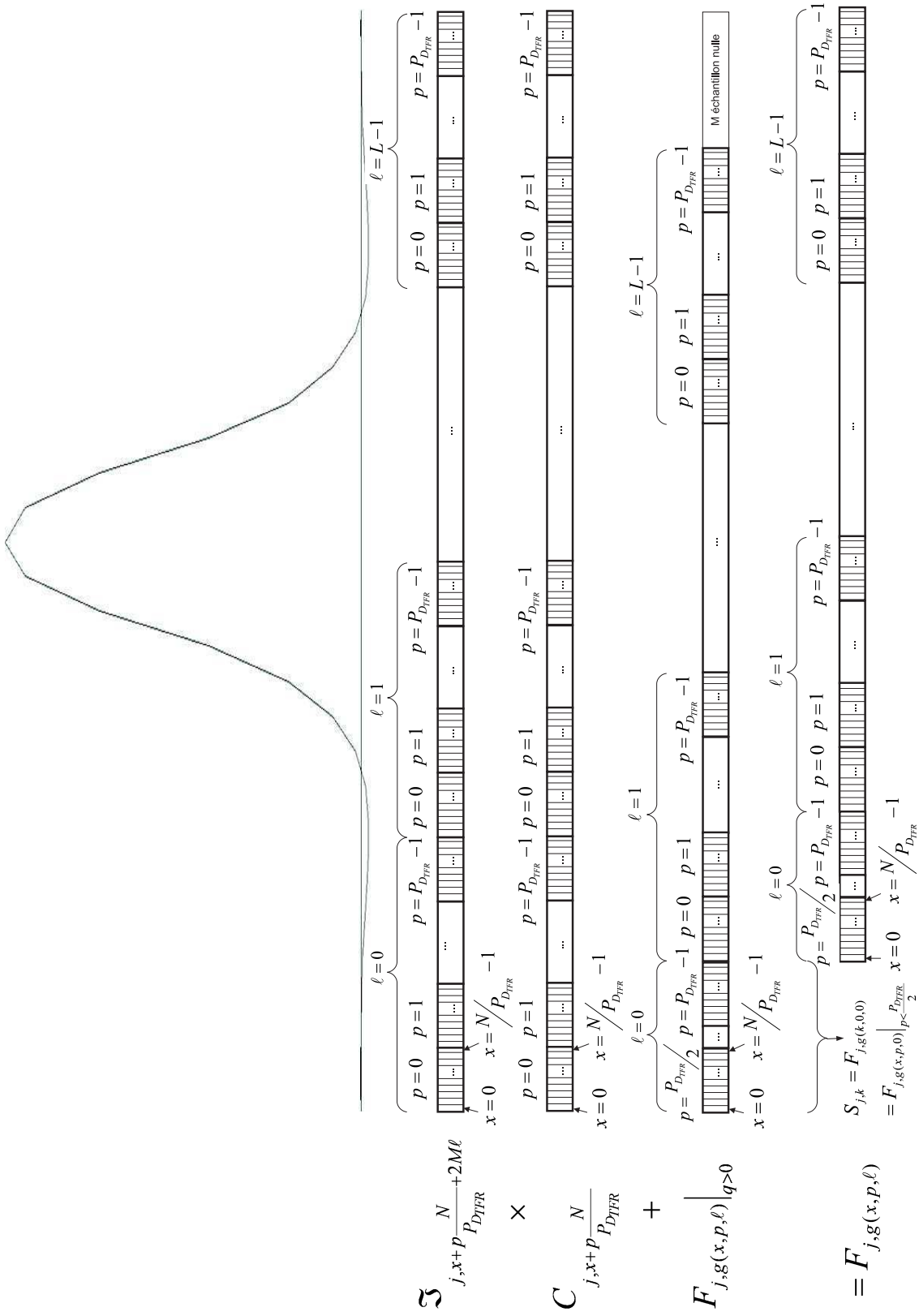

FIGURE 5.8 – Illustration de l'opération de filtrage selon les indices x, p et ℓ

La figure 5.9 illustre les entrées/sorties des données d'un module de calcul des deux dernières colonnes de la matrice. Il est composé de deux blocs de calcul (BC). Chaque bloc consiste en quatre multiplieurs réels et quatre additionneurs réels. Les signaux F_{out} et F_{in} représentent respectivement $F_{j,g(x,p,\ell)}$ et $F_{j,g(x,p,\ell)}|_{q>0}$.

La figure 5.10 illustre l'architecture d'un bloc de calcul. Selon le mode de traitement, le bloc de calcul est configuré soit en un additionneur et un multiplieur complexes, soit en quatre multiplieurs/additionneurs réels. Les deux configurations des blocs de calcul sont illustrées à la figure 5.11. Le mode TFR permet de calculer Y_1 de l'opérateur papillon grâce au BC1 et Y_2 grâce au BC2. Dans le mode filtrage de mise en forme, la paire de multiplieur/additionneur permet de réaliser l'opération de pondération/sommation de l'équation 5.6 pour différentes valeurs de p.

La valeur de p étant comprise en 0 et $P_{D_{TFR}}$ selon le mode MIMO, les valeurs des indices des coefficients IOTA (déterminées par $x+p\frac{N}{P_{D_{TFR}}}+2M\ell$) et des échantillons de résultat de la TFR (déterminées par $x+p\frac{N}{P_{D_{TFR}}}$) diffèrent donc aussi selon le mode MIMO. Ainsi, par rapport à la matrice de calcul, les connexions restent inchangées ; seules les valeurs reçues des échantillons et des coefficients IOTA changent, nous permettant de réaliser soit un filtrage, soit deux filtrages en parallèle soit quatre filtrages en parallèle. La figure 5.11 illustre les différentes valeurs de p pour les quatres paires de multiplieur/additionneur d'un bloc de calcul.

Les deux facteurs de rotation nécessaires dans chacun des blocs de calcul dépendent de l'algorithme radix-2^i utilisé et peuvent être triviaux (± 1 ou $-j$). Afin de tenir compte de ces valeurs et de désactiver les multiplieurs au besoin, des changements de signe et des permutations des parties réelles et imaginaires sont réalisés. Des inverseurs et des multiplexeurs sont incorporés dans les schémas de la figure 5.11. Le tableau 5.4 illustre les différents cas selon les valeurs des facteurs de rotation (identifiés par 1 ou $-j$ pour les valeurs triviales et W pour des valeurs non triviales) et en tenant compte des multiplications par les coefficients IOTA (identifiés par $\Im$). On obtient ainsi 7 versions différentes des modules de calcul en tenant compte de ceux du premier étage. Pour les modules des deux dernières colonnes, les différentes versions diffèrent légèrement et ont donc la même complexité. La figure 5.12 illustre l'emplacement des différents types de modules de calcul dans la matrice. Des schémas détaillés de chaque module de calcul de type A à G sont présentés à l'annexe D.

TABLE 5.4 – Différents cas de figure des modules de calcul selon les multiplications triviales ou non triviales

	cas A	cas B	cas C	cas D	cas E	cas F	cas G
BC1	1	1	1,$\Im$	1,$\Im$	1,W,$\Im$	1,$\Im$	1,W,$\Im$
BC2	1	-j	1,$\Im$	-j,$\Im$	-j,W,$\Im$	1,W,$\Im$	1,W,$\Im$

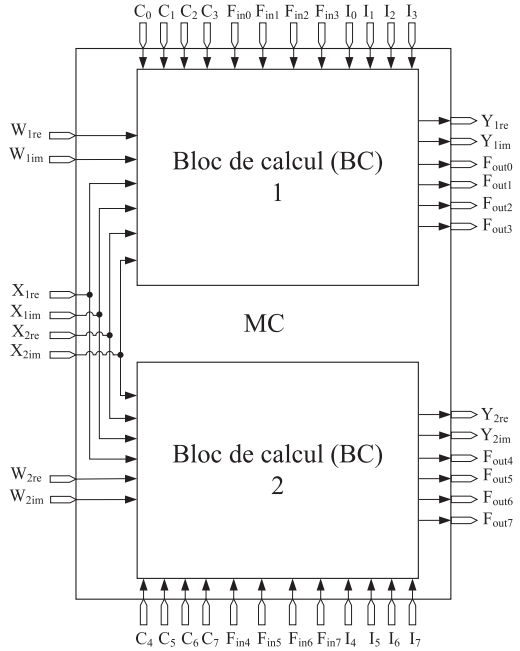

FIGURE 5.9 – Entrées/Sorties d'un module de calcul des deux dernières colonnes

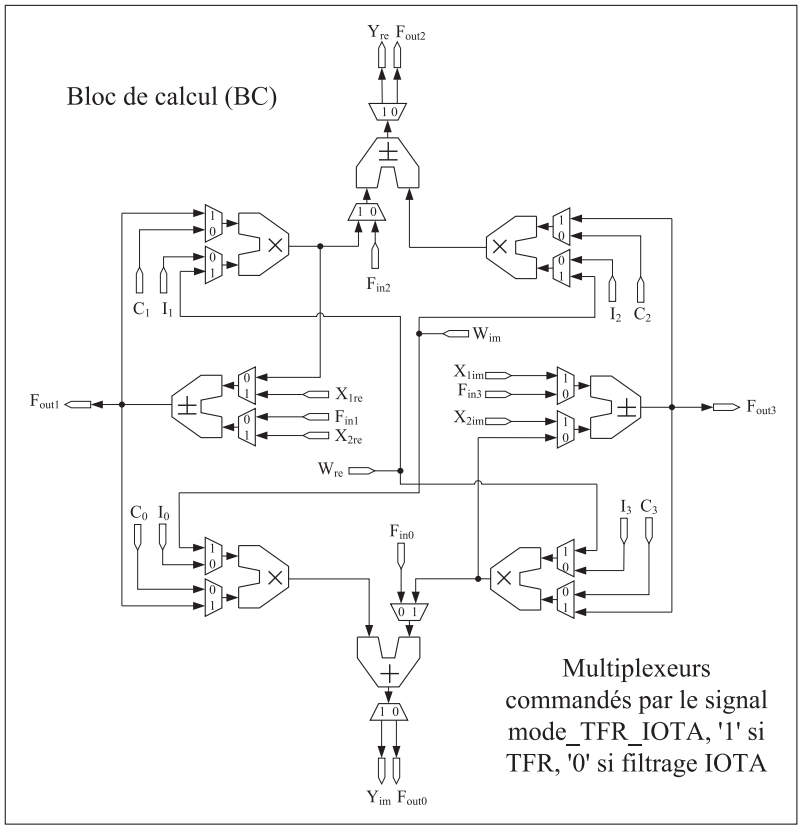

FIGURE 5.10 – Architecture d'un bloc de calcul des deux dernières colonnes

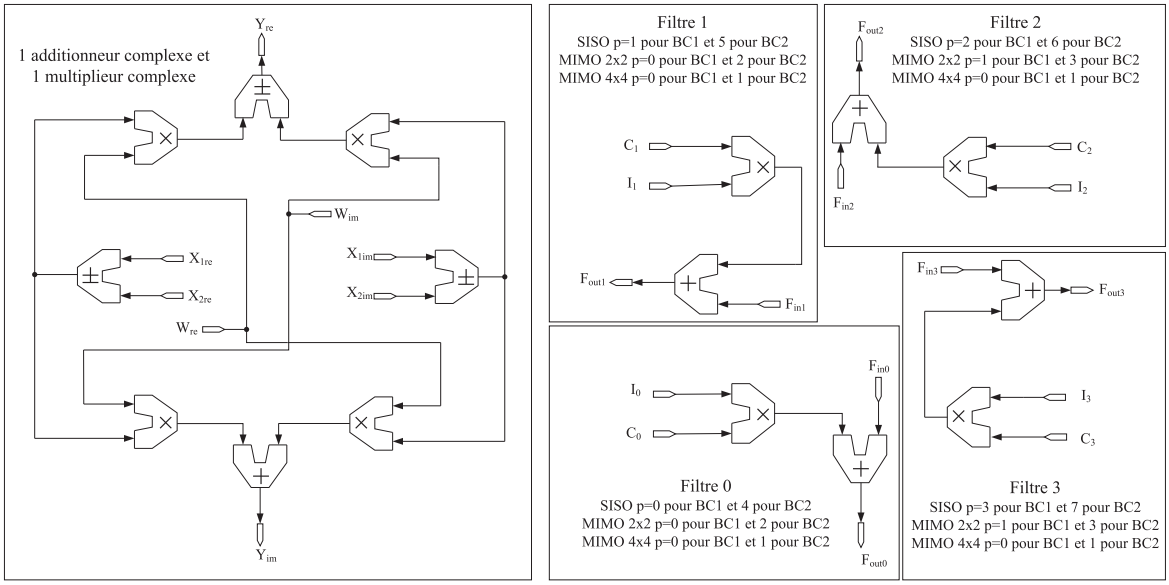

FIGURE 5.11 – Les deux configurations d'un bloc de calcul a) mode TFR b) mode filtrage de mise en forme

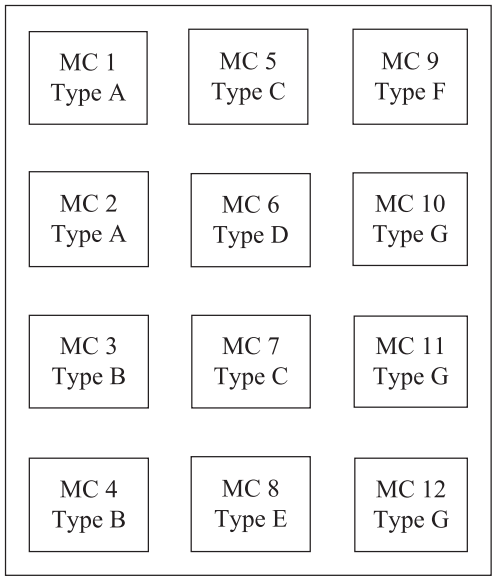

FIGURE 5.12 – Emplacement des différents types (A à G) de modules de calcul dans la matrice

Chapitre 6

Stratégie mémoire

La capacité de reconfiguration de l'architecture proposée induit la nécessité de tailles mémoires variables selon la valeur de L, N, et MIMO, et donc du standard de communication. Du fait du parallélisme des données à l'entrée de la matrice, on doit accéder à $P_D = 8$ données parmi N échantillons à chaque cycle. Par conséquent, l'architecture possède des blocs mémoire constitués de P_D bancs de mémoires RAM de taille $\frac{N_{max}}{P_D} = 1024$ points complexes dans le but de réaliser P_D lectures/écritures simultanément. La figure 6.1 illustre un bloc mémoire à $N_{max} = 8192$ données complexes.

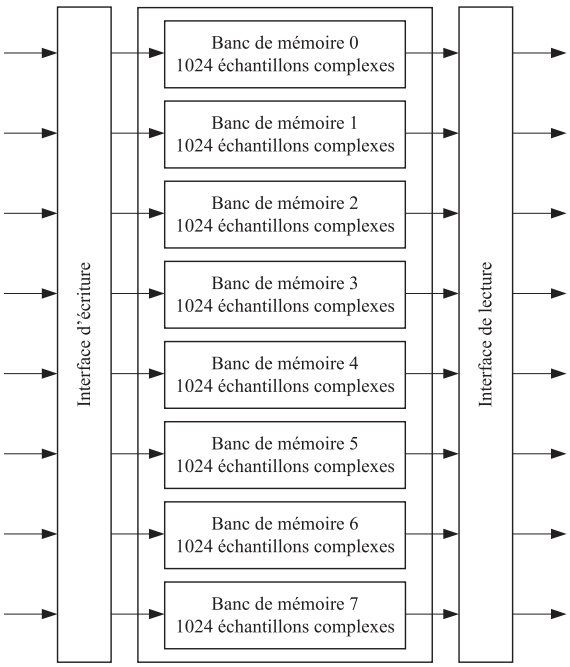

Figure 6.1 – Bloc mémoire à $N_{max} = 8192$ données complexes

Les bancs de mémoire permettent d'obtenir des tailles de mémoire variables selon le paramètre N et le mode MIMO. Le tableau 6.1 montre les différentes tailles mémoires de chacun des P_D bancs de mémoire RAM à taille variable. Selon la configuration, chaque banc de mémoire aura une taille de $\frac{N}{P_{D_{TFR}}}$ échantillons complexes.

Chacun des bancs de mémoire est constitué de 8 sous RAMs de taille de 512, 256, 128, 64, 32, 16 et deux de 8 échantillons complexes. L'idée principale est de concaténer des sous RAMs *double port* grâce au décodage des adresses. Ainsi, nous obtenons toute les tailles mémoires entre 8 et 1024 points complexes pour les bancs de mémoires. Seule une mémoire à 8 points complexes

est active en tout temps. Les 7 autres sous-mémoires sont désactivées selon la valeur de N et du mode MIMO. La figure 6.2 illustre l'architecture d'un banc de mémoire à taille variable. Le décodage aussi bien en lecture qu'en écriture est un décodage classique, les bits de poids le plus faible servant à adresser les 8 sous-mémoires. On sélectionne une des sorties des sous-mémoires RAM grâce aux multiplexeurs contrôlés par les bits du poids le plus fort du vecteur d'adresse. Ce dernier est sur 10 bits, nous permettant d'adresser des valeurs allant de 0 à 1023.

TABLE 6.1 – Tailles des bancs mémoire selon le paramètre N et le mode MIMO

N	SISO $P_{D_{TFR}} = 8$	MIMO 2x2 $P_{D_{TFR}} = 4$	MIMO 4x4 $P_{D_{TFR}} = 2$
8192	1024	-	-
4096	512	1024	-
2048	256	512	1024
1024	128	256	512
512	64	128	256
256	32	64	128
128	16	32	64
64	8	16	32

Le délai de propagation le plus long dans les multiplexeurs correspond au cas où l'échantillon voulu est situé dans la dernière mémoire RAM 8. Toutefois, le délai de décodage de l'adresse, dans ce cas de figure, est le plus court, du fait de la petite taille de la mémoire. Dans le cas où l'échantillon désiré est situé dans la plus grande mémoire RAM 1, le délai de décodage est plus long mais l'échantillon se propage dans un seul multiplexeur. Le délai de décodage et de propagation dans les différents multiplexeurs est sensiblement le même quelle que soit la RAM concernée.

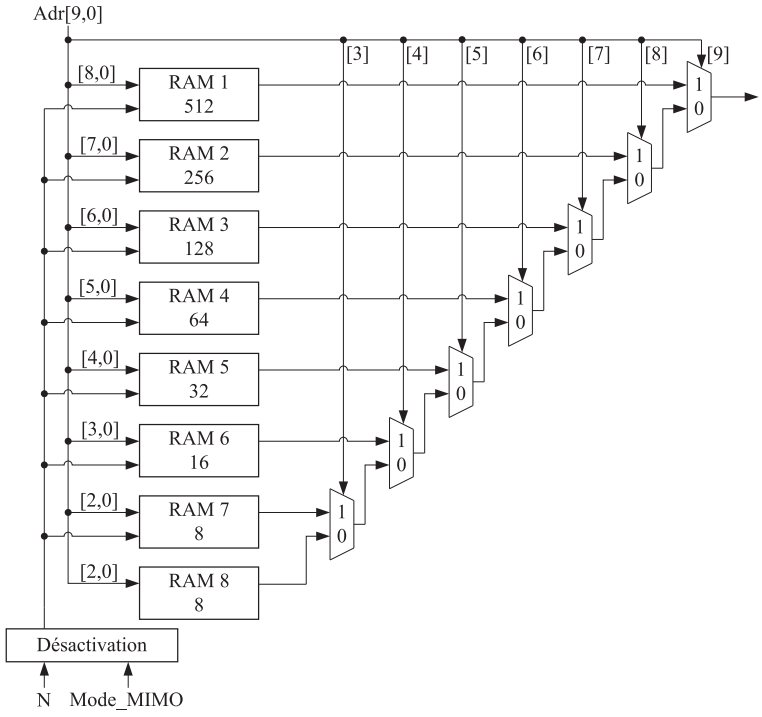

FIGURE 6.2 – Banc de mémoire à taille variable entre 8 et 1024 échantillons complexes

6.1 Mémoires des échantillons de la TFR

Les deux mémoires à $N_{max} = 8192$ échantillons complexes de la TFR correspondent au bloc mémoire de la figure 6.1. L'algorithme de la TFR génère $\frac{N}{P_{D_{TFR}}}$ séquences comprenant $P_{D_{TFR}}$ échantillons complexes à chacun des $\log_r N$ étages. La figure 6.3 illustre un exemple d'allocation mémoire d'une TFR à 64 points utilisant l'algorithme radix-4. Au premier des trois étages de la TFR, une des séquences nécessaires implique les échantillons d'indice 0, 16, 32, et 48. Au deuxième étage, les échantillons d'indice 0, 4, 8, et 12. Finalement, au dernier étage, les échantillons 0, 1, 2 et 3. Les quatres échantillons de chacune de ces séquences doivent être donc dans différentes mémoires afin d'éviter les conflits d'adresses. Il s'agit de trouver une fonction qui à l'indice x_p d'un échantillon de l'expression 5.2 associe un banc mémoire (compris entre 0 et $P_D - 1$) et une adresse dans ce banc mémoire (comprise entre 0 et $\frac{N}{P_{D_{TFR}}} - 1$).

RAM 0	RAM 1	RAM 0	RAM 1
0	1	2	3
7	4	5	6
10	11	8	9
13	14	15	12
19	16	17	18
22	23	20	21
25	26	27	24
28	29	30	31
34	35	32	33
37	38	39	36
40	41	42	43
47	44	45	46
49	50	51	48
52	53	54	55
59	56	57	58
62	63	60	61

FIGURE 6.3 – Exemple d'allocation des échantillons pour une TFR radix-4 de 64 points

Soit, la représentation de l'indice x_p de l'équation 5.2 d'un échantillon de la TFR en base r_{MIMO} :

$$x_p = \left[x_{p_{\log_{r_{MIMO}} N-1}} \; x_{p_{\log_{r_{MIMO}} N-2}} \ldots x_{p_1} \; x_{p_0} \right]_{r_{MIMO}} \tag{6.1}$$

avec

$$r_{MIMO} = P_{D_{TFR}}. \tag{6.2}$$

Alors l'échantillon à l'indice x_p sera mémorisé dans le banc mémoire et à l'adresse déterminés par les équations suivantes [110] :

$$\text{Banc mémoire} = \left(x_{p_{\log_{r_{MIMO}} N-1}} + x_{p_{\log_{r_{MIMO}} N-2}} + \ldots + x_{p_0} \right) \; \textit{Modulo}\; r_{MIMO}, \tag{6.3}$$

$$\text{Adresse mémoire} = \left[x_{p_{\log_{r_{MIMO}} N-1}} \; x_{p_{\log_{r_{MIMO}} N-2}} \ldots x_{p_1} \right]_{r_{MIMO}}. \tag{6.4}$$

La valeur de r_{MIMO} correspond au degré de parallélisme des données d'une TFR, $P_{D_{TFR}}$, et non la base de l'algorithme radix-2^i. Ceci est dû au fait que peu importe l'algorithme utilisé, le nombre d'indices générés à chaque cycle correspond à $P_{D_{TFR}}$ et dépend donc du mode MIMO.

Par exemple, dans la configuration MIMO 2x2 radix-2, $P_{D_{TFR}} = 4$ échantillons sont générés à chaque cycle et non 2 (si on tient compte du radix). Les interfaces en lecture et écriture des blocs mémoires des échantillons de la TFR intègrent donc les deux équations 6.3 et 6.4 pour les trois valeurs de r_{MIMO}, soit 2, 4 et 8.

6.1.1 Générateur d'indice à base variable et à taille variable

La génération des indices nécessite une gestion selon la taille de la TFR et de la base r_{MIMO}. La génération des indices x_p est réalisée tous les i étages pour un algorithme radix-2^i. Dans [111], un générateur d'indice à base de multiplexeurs, nommé *SIB* (*shift-insert-bypass*), est proposé pour une architecture radix-2^2. Notre solution repose sur le même principe mais est généralisée pour tous les cas de radix-2^i.

Soit κ, le numéro de l'étage en cours compris entre 0 et $\log_2 N_{max} - 1$. Les r_{MIMO} indices x_p impliqués à chaque étage κ suivant un flot de données radix-2^i ne diffèrent que pour les $\log_2 r_{MIMO}$ bits situés à la position $\left(\log_2 \frac{N}{r_{MIMO}} - \kappa + \omega\right)$ avec $0 \leq \omega \leq \log_2 r_{MIMO} - 1$. Les autres bits identiques correspondent au compteur de papillon b. La solution consiste donc à insérer $\log_2 r_{MIMO}$ bits différents (qu'on identifiera sous le terme symbole $S = [S_\omega - 1, \ldots, S_0]$) à la bonne position selon la valeur de l'étage pour le bon indice x_p. La valeur du symbole S est comprise entre 000 et 111 pour le mode SISO, 00 et 11 pour le mode MIMO 2x2 et 0 ou 1 pour le mode MIMO 4x4. A tous les i étages, les symboles S sont décalés vers la droite de i positions.

Soit, à une fin d'illustration, une TFR à 64 points utilisant le radix-2^2 comme algorithme pour un système MIMO 2x2, $r_{MIMO} = 4$. Du fait qu'on lit 4 échantillons par cycle, on réalise donc 2 croisillons par cycle. Le compteur papillon est compris donc entre 0 et $\frac{N/2}{2} = 16$ sur 4 bits. Le compteur d'étage κ est sur 3 bits et varie entre 0 et 5. Finalement, la valeur de ω est sur 1 bit. On génère donc 16 séquences de 4 points aux étages 0, 2 et 4. La figure 6.4 illustre la valeur du vecteur des indices x_p en représentation binaire. Ainsi, au premier étage $\kappa = 0$, les 16 séquences nécessaires de 4 échantillons x_p sont $x_0 = 00b_3b_2b_1b_0$ ($S = 00$), $x_1 = 01b_3b_2b_1b_0$ ($S = 01$), $x_2 = 10b_3b_2b_1b_0$ ($S = 10$), $x_3 = 11b_3b_2b_1b_0$ ($S = 11$). La première séquence lorsque le compteur papillon b est à 0 est alors 0, 16, 32, 48.

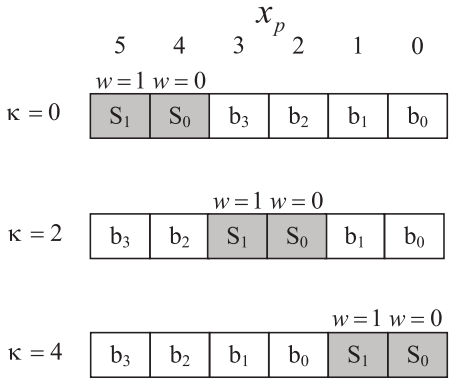

FIGURE 6.4 – Valeurs des indices x_p pour une TFR à 64 points radix-2^2 en mode MIMO 2x2

Le $v^{\text{ième}}$ bit de l'indice x_p prend donc comme valeur soit un bit du compteur de papillon ou un bit du symbole S. La figure 6.5 illustre un multiplexeur permettant de déterminer la valeur du bit v ($0 \leq v \leq \log_2 N_{max} - 1$) de l'indice x_p. On obtient les cas suivants :

$$x_{p_v} = \begin{cases} S_w & \text{si } v = \log_2 \frac{N}{r_{MIMO}} - \kappa + \omega \\ b_v & \text{si } v < \log_2 \frac{N}{r_{MIMO}} - \kappa \\ b_{v-\log_2 r_{MIMO}} & \text{si } v > \log_2 \frac{N}{r_{MIMO}} - \kappa + \log_2 r_{MIMO} - 1 \end{cases} \tag{6.5}$$

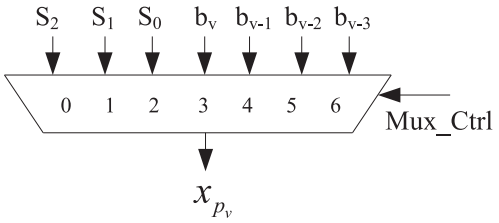

FIGURE 6.5 – Multiplexeur des modules SIB

Ainsi, si le $v^{\text{ième}}$ bit correspond à un des bits qui diffèrent, il prendra un des bits de symbole S. Si le $v^{\text{ième}}$ bit est à droite des bits qui diffèrent, alors il prendra le $v^{\text{ième}}$ bit du compteur de papillon b. Finalement, si le $v^{\text{ième}}$ bit est à gauche, il prendra un des bits d'indice $v-1$, $v-2$ ou $v-3$ selon la valeur de r_{MIMO}, soit b_{v-1}, b_{v-2} ou b_{v-3}.

La figure 6.6 illustre l'architecture complète du module SIB composé de $\log_2 N_{max} = 13$ multiplexeurs chacun sélectionnant un bit de l'indice x_p. Le générateur d'indices comportera donc 8 modules SIB générant chacun un indice. Chaque générateur recevra une valeur différente du symbole S. La figure 6.7 illustre l'architecture du générateur d'indice pour la TFR à base variable et à taille variable. Il sera configuré par N, MIMO et radix, ainsi que les signaux *start* et *shift* pour signaler le début de la TFR et le début de chaque nouvel étage radix-2^i. Il prendra en entrée la valeur b du compteur papillon et produira 8 adresses x_p en sortie. En mode SISO, on utilisera les 8 adresses, en mode MIMO 2x2 on utilisera uniquement les adresses x_0, x_1, x_2 et x_3 qui seront dupliquées pour les 2 sous-mémoires et enfin en mode MIMO 4x4 on utilisera uniquement les adresses x_0 et x_1 qui seront dupliquées pour les 4 sous-mémoires.

Le contrôle des $\log_2 N_{max} = 13$ multiplexeurs de chacun des 8 SIB se fait grâce à un seul séquenceur du fait que le fonctionnement est le même, seuls les symboles S changent. Le séquenceur est composé de registres à décalage avec entrée en série et sorties parallèle à base de 13 registres de 3 bits. La figure 6.8 illustre l'architecture d'un des 13 étages du séquenceur. Les registres à décalage sont initialisés selon les valeurs de N et du mode MIMO afin de configurer les multiplexeurs de manière adéquat. À la fin de chaque étage radix-2^i, le contenu de la FIFO est décalé de i positions grâce au signal *shift* qui agit comme une horloge sur les registres. Ainsi, comme on le constate sur la figure 6.8, selon la valeur de i, le multiplexeur sélectionne la sortie d'un registre précédant Mux_CTRL_{v-i}. La structure complète est illustrée à la figure 6.9.

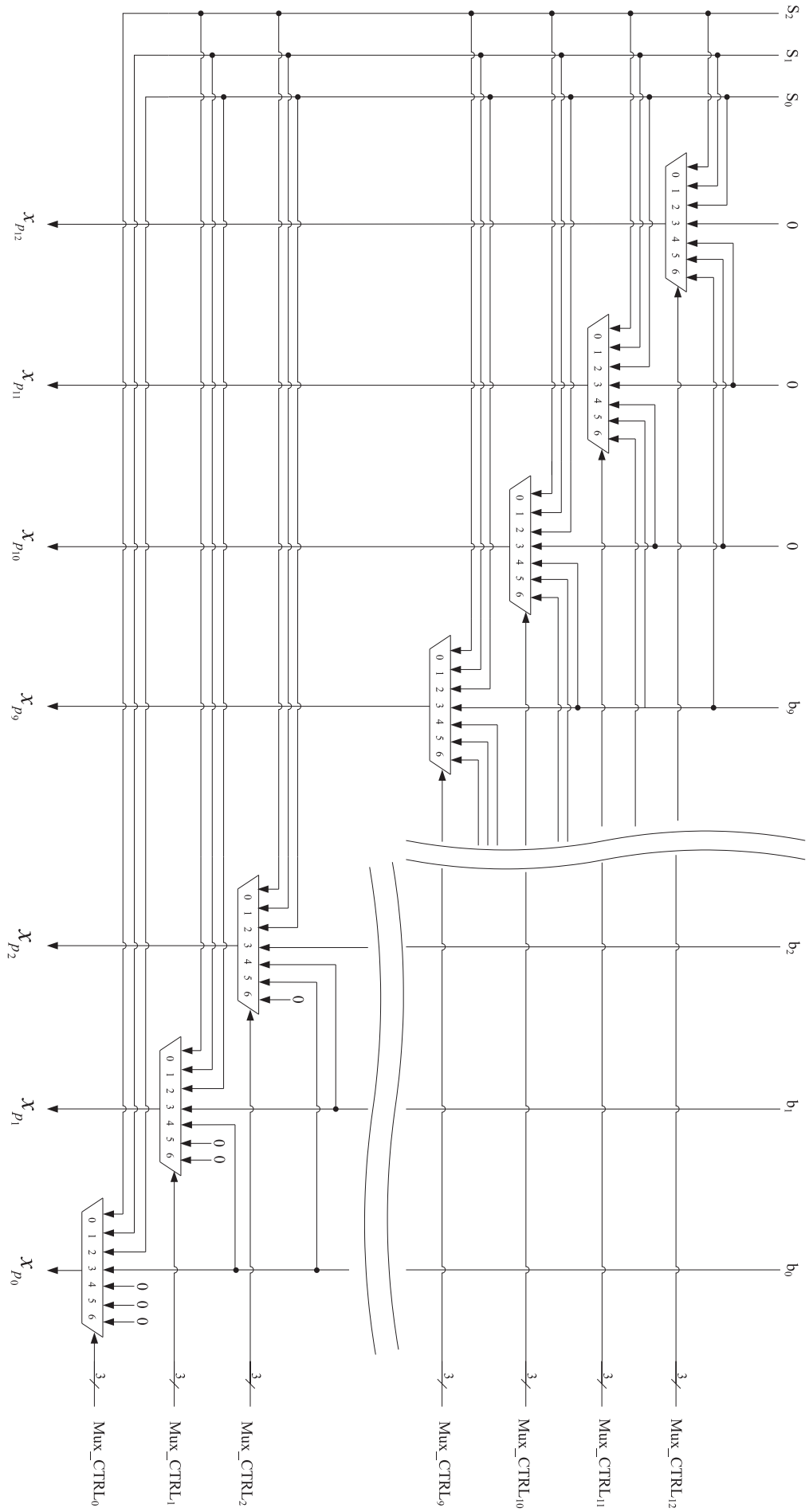

FIGURE 6.6 – Architecture d'un module SIB

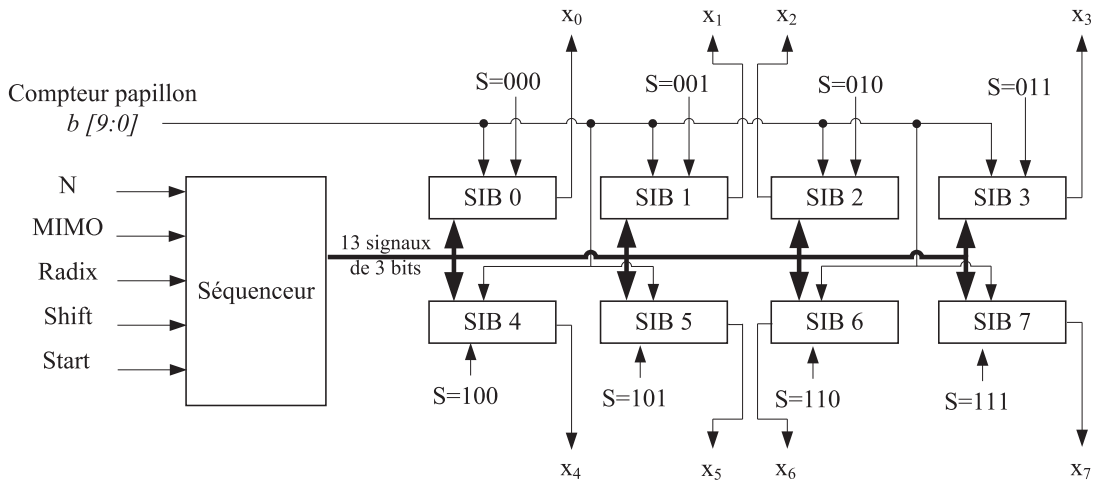

FIGURE 6.7 – Architecture du générateur d'indices pour la TFR

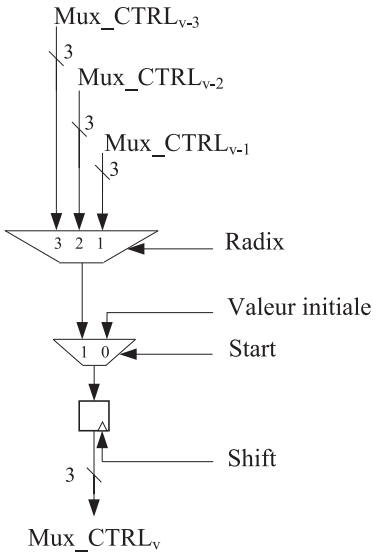

FIGURE 6.8 – Architecture d'un étage du séquenceur des modules SIB

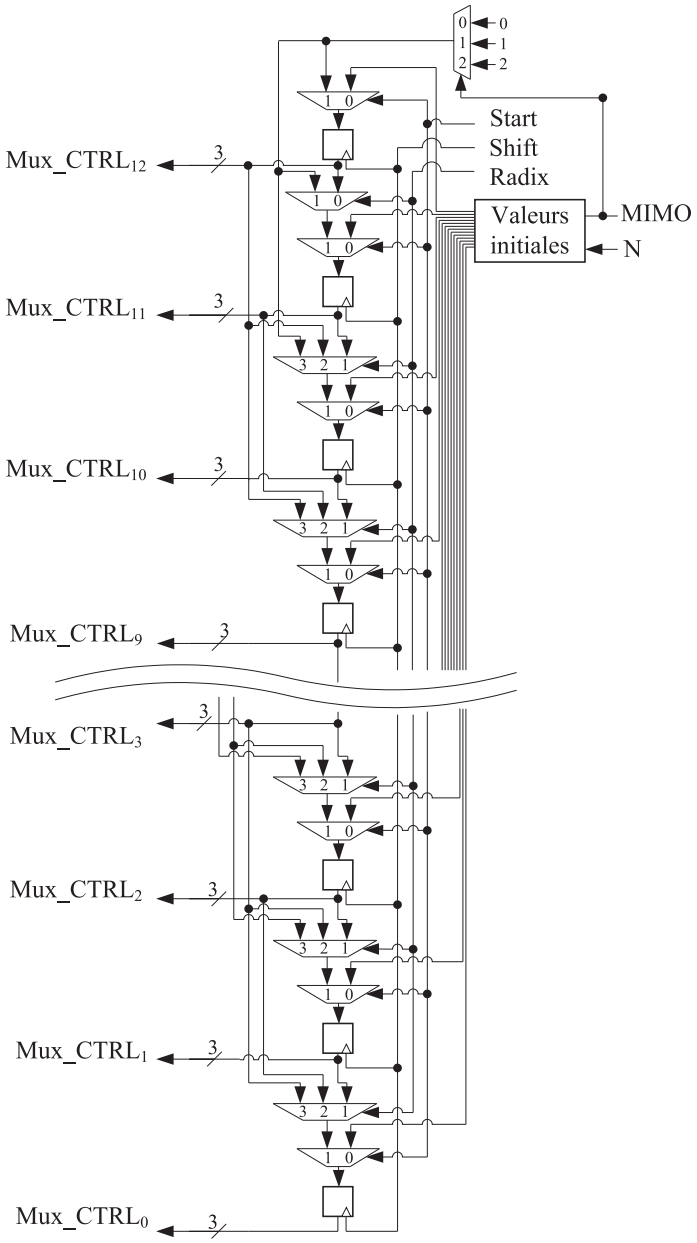

FIGURE 6.9 – Architecture du séquenceur des modules SIB

6.2 Mémoires des coefficients de rotation W de la TFR

La principale contrainte pour la génération des coefficients des facteurs de rotation W, vient de la stratégie parallèle utilisée par l'architecture. Le cas du radix-2^3 impose la plus grande contrainte. La figure 6.10 illustre l'emplacement et la valeur des multiplications complexes dans la matrice pour les 3 algorithmes en mode SISO. Soit, les 2 dernières colonnes pour le radix-2^3 et la dernière colonne pour le radix-2^2 et le radix-2.

Pour le radix-2^3, les multiplieurs de la deuxième colonne ne varient pas en fonction de N et correspondent toujours à $W_8^1 = \frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2}$ et $W_8^3 = -\frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2}$. La difficulté se pose donc pour le dernier étage où 7 coefficients doivent être obtenus de la mémoire des coefficients W à chaque cycle. Les valeurs sont : W_N^n, W_N^{2n}, W_N^{3n}, W_N^{4n}, W_N^{5n}, W_N^{6n}, W_N^{7n}, n correspondant au compteur papillon b, soit $W_N^{\gamma_{R23}b}$ ($1 \leq \gamma_{R23} \leq 7$, $R23$ pour radix-2^3).

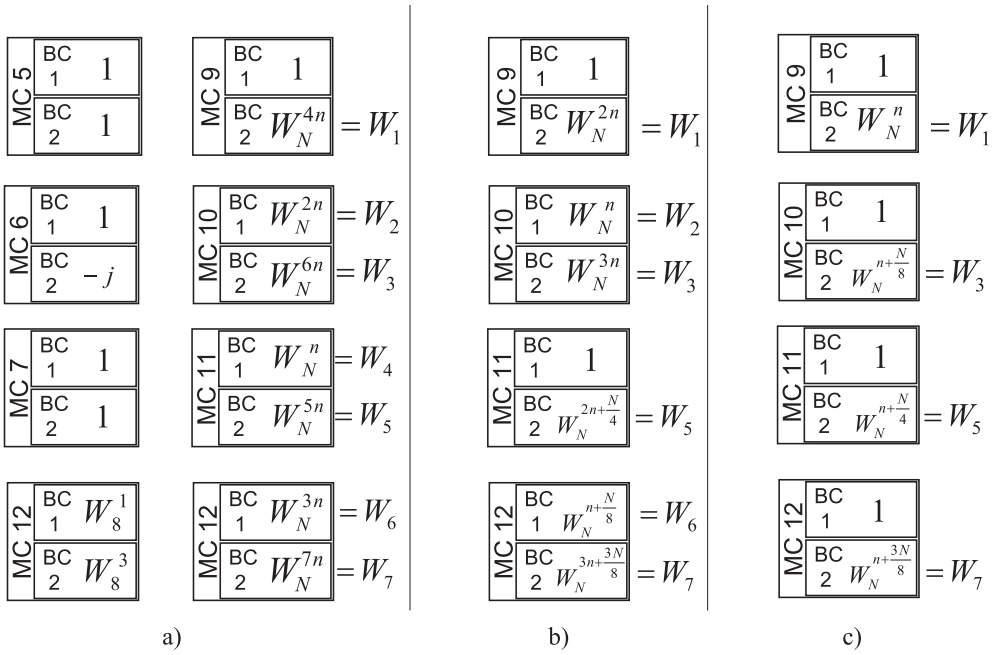

FIGURE 6.10 – Emplacement des multiplications non triviales dans la matrice en mode SISO pour le a) radix-2^3 b) radix-2^2 c) radix-2

L'algorithme radix-2^3 est adopté seulement pour une configuration SISO pour $\log_2 N$ multiple de 3, c'est-à-dire $N = 2^6 = 64$, $N = 2^9 = 512$ et finalement $N = 2^{12} = 4096$. Pour ces valeurs de N, on doit obtenir les 7 valeurs W citées plus haut. Puisque la conception et donc la taille des mémoires est faite pour N_{max}, les coefficients W pour $N < N_{max}$ correspondent à un sous ensemble des coefficients W. Ceci revient, en réalité, à sous-échantillonner les coefficients W pour $N < N_{max}$. Une correspondance est donc nécessaire entre les 7 coefficients $W_N^{\gamma_{R23}b}$ et $W_{N_{max}}^{\delta_{R23}}$ afin de déterminer l'emplacement des coefficients dans les ROMs. Ainsi, les 7 coefficients correspondent à $W_{N_{max}}^{\delta_{R23}} = W_{N_{max}}^{\frac{N_{max}}{N}\gamma_{R23}b}$. L'indice $\delta_{R23}(\gamma_{R23}) = \frac{N_{max}}{N}\gamma_{R23}b$ permet de déterminer l'adresse physique du coefficient dans la mémoire ROM.

En observant le plan complexe des facteurs de rotation à la figure 6.11, on constate que l'algorithme radix-2^3 nécessite $\frac{7N}{8}$ coefficients contrairement à $\frac{N}{2}$ pour le radix-2 et $\frac{3N}{4}$ pour le radix-2^2.

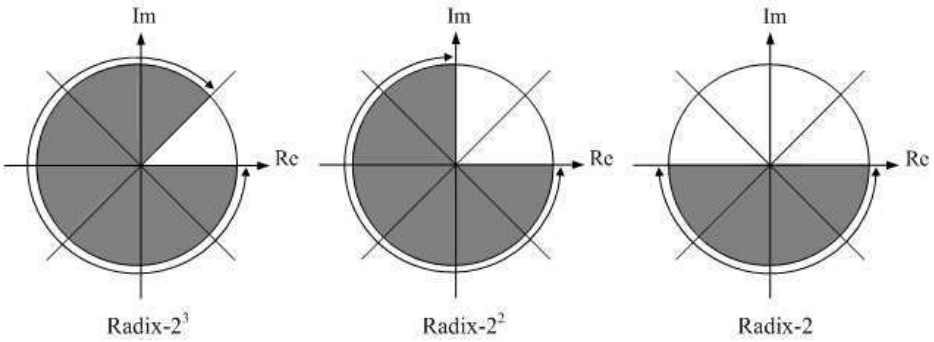

FIGURE 6.11 – Domaine des coefficients W selon l'algorithme radix-2^i dans le plan complexe pour une TFR directe

Afin d'éviter tout conflit de mémoire lors de la lecture des 7 coefficients, le plus évident serait bien sûr d'avoir sept mémoires contenant chacune tout le plan complexe. Cette solution n'est évidemment pas souhaitable. Toutefois, il est possible d'obtenir ce schéma en limitant aussi le nombre de coefficients mémorisés. Grâce aux relations trigonométriques, il est possible d'obtenir n'importe quel point du plan complexe en utilisant seulement $\frac{N}{8}$ coefficients et en réalisant des opérations de permutation des parties réelles et imaginaires ou/et d'inversion de signe. Pour accéder aux 7 coefficients en parallèle, on utilise 7 mémoires à $\frac{N_{max}}{8}$ points (on verra en réalité qu'une petite optimisation peut être obtenue pour limiter la taille de certaines mémoires).

La figure 6.12 illustre le plan complexe ainsi que les coefficients réellement mémorisés situés dans la partie en noire. Les 3 bits illustrés représentent les huit parties à $\frac{N_{max}}{8}$ points du plan complexe dans le cas d'une TFR directe (et TFR inverse). En réalité, ces trois bits sont les trois bits les plus significatifs (*MSB : Most Significant Bit*) de l'indice de l'échantillon désiré, soit $\delta[12, 10]$.

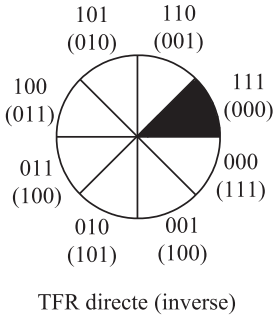

FIGURE 6.12 – Les 8 sections du plan complexe pour une TFR directe (et inverse) et ainsi que les coefficients mémorisés (en noir)

Soient $cos(\alpha)$ et $sin(\alpha)$, les valeurs lues dans la ROM de $\frac{N_{max}}{8}$ points complexes, le tableau 6.2 donne les valeurs finales des coefficients à obtenir en fonction des trois bits $\delta[12, 10]$ pour la TFR directe et inverse. Les parties réelles des facteurs de rotation sont les mêmes, alors que les parties imaginaires des facteurs sont simplement inversées quand on passe de la TFR directe à la TFR inverse (et vice versa). En réalité, les coefficients W du demi-quadrant $\delta[12, 10]$ pour une TFR directe correspondent aux coefficients du demi-quadrant du complément de $\delta[12, 10]$ pour une TFR inverse.

TABLE 6.2 – les valeurs des facteurs de rotation pour la TFR directe et inverse selon les bits $\delta[12, 10]$ de l'indice du coefficient

$\delta[12, 10]$	TFR directe		TFR inverse	
	réelle	imaginaire	réelle	imaginaire
000	$cos(\alpha)$	$-sin(\alpha)$	$cos(\alpha)$	$sin(\alpha)$
001	$sin(\alpha)$	$-cos(\alpha)$	$sin(\alpha)$	$cos(\alpha)$
010	$-sin(\alpha)$	$-cos(\alpha)$	$-sin(\alpha)$	$cos(\alpha)$
011	$-cos(\alpha)$	$-sin(\alpha)$	$-cos(\alpha)$	$sin(\alpha)$
100	$-cos(\alpha)$	$sin(\alpha)$	$-cos(\alpha)$	$-sin(\alpha)$
101	$-sin(\alpha)$	$cos(\alpha)$	$-sin(\alpha)$	$-cos(\alpha)$
110	$sin(\alpha)$	$cos(\alpha)$	$sin(\alpha)$	$-cos(\alpha)$
110	$cos(\alpha)$	$sin(\alpha)$	$cos(\alpha)$	$-sin(\alpha)$

Le schéma 6.13 illustre l'architecture générant n'importe quel point du plan complexe à partir d'une mémoire à $\frac{N_{max}}{8}$ points complexes. Le bit $\delta[10]$ permet de déterminer l'adresse physique du coefficient dans le demi-quadrant mémorisé de la figure 6.12, soit $\delta[9,0]$ si $\delta[10]=0$, soit l'inverse de l'adresse $\delta[9,0]$ si $\delta[10]=1$. Les valeurs lues dans les deux ROMs contenant les parties réelles et imaginaires subiront des inversions et des permutations selon le signal de contrôle suivant :

$$\begin{aligned} Ctrl\_Mux[2] &= \delta[12] \oplus TFR\_Direct\_Inverse \\ Ctrl\_Mux[1] &= \delta[11] \oplus TFR\_Direct\_Inverse \\ Ctrl\_Mux[0] &= \delta[10] \oplus TFR\_Direct\_Inverse \end{aligned} \tag{6.6}$$

Où le signal $TFR\_Direct\_Inverse$ est un signal de contrôle du système qui vaut '1' ou '0' si l'on effectue une TFR directe ou inverse respectivement. Il s'agit tout simplement de prendre le complémentaire des bits $\delta[12,10]$ quand on passe d'une TFR inverse à une TFR directe.

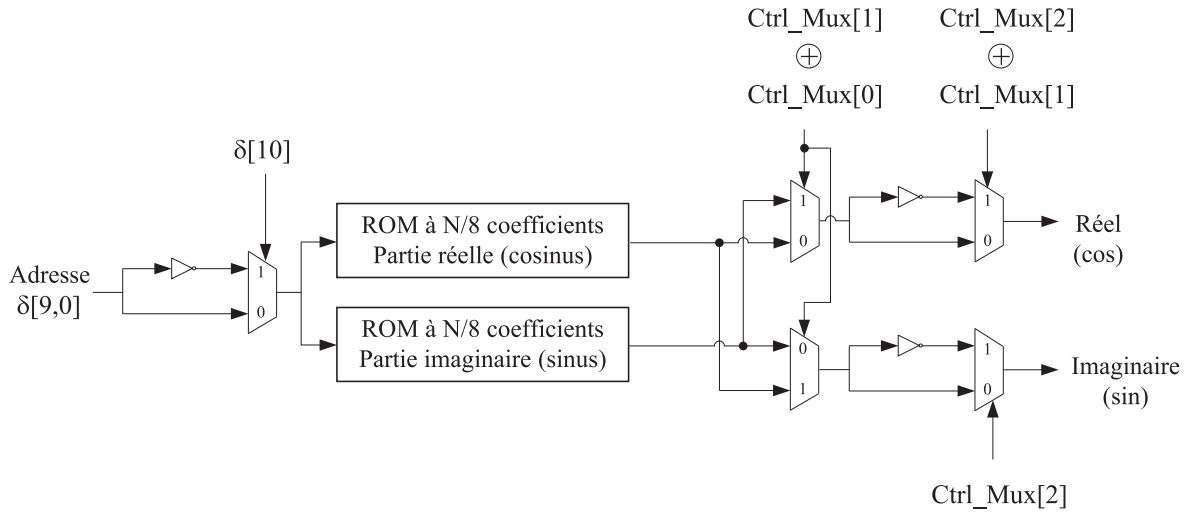

FIGURE 6.13 – Architecture ROM à $\frac{N_{max}}{8}$ points complexes réalisant la génération d'un coefficient W du plan complexe

Les algorithmes radix-2^2 et radix-2 nécessitent respectivement 6 et 4 coefficients W par cycle en mode SISO comme illustré à la figure 6.10. En tenant compte du faite que les mémoires sont prévues pour N_{max}, les coefficients W pour l'algorithme radix-2^2 sont alors $W_{N_{max}}^{n\frac{N_{max}}{N}}$, $W_{N_{max}}^{2n\frac{N_{max}}{N}}$, $W_{N_{max}}^{3n\frac{N_{max}}{N}}$, $W_{N_{max}}^{\left(n+\frac{N}{8}\right)\frac{N_{max}}{N}}$, $W_{N_{max}}^{\left(2n+\frac{N}{4}\right)\frac{N_{max}}{N}}$ et $W_{N_{max}}^{\left(3n+\frac{3N}{8}\right)\frac{N_{max}}{N}}$. L'indice des coefficients est alors, avec $n=b$, $\delta_{R22}(\gamma_{R22})=\gamma_{R22}b\frac{N_{max}}{N}$ pour les 3 premiers coefficients $1\leq\gamma_{R22}\leq 3$ et $\delta_{R22}(\gamma_{R22})=(\gamma_{R22}-3)\left(b\frac{N_{max}}{N}+\frac{N_{max}}{8}\right)$ pour les 3 derniers coefficients, $4\leq\gamma_{R22}\leq 6$.

De la même façon, les coefficients W pour radix-2 en mode SISO sont égaux à : $W_{N_{max}}^{\frac{N_{max}}{N}n}$, $W_{N_{max}}^{\left(n+\frac{N}{8}\right)\frac{N_{max}}{N}}$, $W_{N_{max}}^{\left(n+\frac{N}{4}\right)\frac{N_{max}}{N}}$ et $W_{N_{max}}^{\left(n+\frac{3N}{8}\right)\frac{N_{max}}{N}}$. L'indice des coefficients est alors $\delta_{R2}(\gamma_{R2})=b\frac{N_{max}}{N}+\frac{\gamma_{R2}N_{max}}{8}$ avec $n=b$ et $0\leq\gamma_{R2}\leq 3$.

Dans le cas du radix-2, une optimisation est possible en ne cherchant que les deux premiers facteurs dans les mémoires ($\gamma_{R2}=0$ et 1). Les deux derniers seront ($\gamma_{R2}=2$ et 3) obtenus par simple inversion-permutation des deux premiers. Nous avons alors pour une TFR directe $W_N^{\left(n+\frac{N}{4}\right)\frac{N_{max}}{N}}=jW_N^{\frac{N_{max}}{N}n}$ et $W_N^{\left(n+\frac{3N}{8}\right)\frac{N_{max}}{N}}=jW_N^{\left(n+\frac{N}{8}\right)\frac{N_{max}}{N}}$. Et pour la TFR inverse $W_N^{-\left(n+\frac{N}{4}\right)\frac{N_{max}}{N}}=-jW_N^{-\frac{N_{max}}{N}n}$ et $W_N^{-\left(n+\frac{3N}{8}\right)\frac{N_{max}}{N}}=-jW_N^{-\left(n+\frac{N}{8}\right)\frac{N_{max}}{N}}$.

Afin de réaliser cette optimisation, nous avons besoin d'un multiplexage de données pour les facteurs W_5 et W_7 (notation correspondant à la figure 6.10) qui sont calculés à partir des coefficients W_1 et W_3 dans le cas de radix-2. Le calcul peut être directement fait sur les données lues dans les mémoires en jouant sur les signaux de contrôle de la permutation de données pour W_5 et W_7. En fait, la multiplication par j (-j) pour la TFR directe (inverse) revient tout simplement à tourner le coefficient de $\frac{\pi}{2}$, qui peut être obtenu en ajoutant la valeur "010" à $\delta[12,10]$. En d'autres termes, on peut utiliser le bit $radix[1]$ (qui est à 0 uniquement pour radix-2) afin d'obtenir les signaux de contrôle des multiplexeurs. Les équations 6.6 deviennent alors pour W_5 et W_7 :

$$\begin{aligned} Ctrl\_Mux[2]_{W_5/W_7} &= \left[\delta[12]_{W_1/W_3} \oplus \left(\delta[11] \cdot \overline{radix[1]}\right)\right] \oplus TFR\_Direct\_Inverse \\ Ctrl\_Mux[1]_{W_5/W_7} &= \left(\delta[11]_{W_1/W_3} \oplus \overline{radix[1]}\right) \oplus TFR\_Direct\_Inverse \\ Ctrl\_Mux[0]_{W_5/W_7} &= \delta[10]_{W_1/W_3} \oplus TFR\_Direct\_Inverse \end{aligned} \tag{6.7}$$

La plus grande taille de TFR utilisant le radix-2 est 8192 points pour le mode SISO. En effet, 8192 est une puissance de 2 seulement et une première décomposition radix-2 est nécessaire. Donc, du point de vue des unités de stockage, le radix-2 nécessite 2 mémoires de $\frac{N_{max}}{8}$ points. Une autre optimisation est donc possible en considérant les 5 mémoires supplémentaires (pour un totale de 7 mémoires). En effet, dans le cas du radix-2^3, la plus grande taille de TFR utilisée est de 4096 points pour le mode SISO. Dans le cas du radix-2^2, la plus grande taille est de 2048 points dans le mode SISO, et 4096 points pour le mode MIMO 2x2. Les coefficients nécessaires correspondent donc aux coefficients aux indices pairs. Par conséquent, du fait que les coefficients aux indices impairs peuvent être ignorés, les 5 mémoires restantes auront une capacité de $\frac{N_{max}}{16}$. La capacité totale sera alors de : $2\frac{N_{max}}{8} + 5\frac{N_{max}}{16} = \frac{9N_{max}}{16}$, soit 4608 coefficients complexes.

La génération des coefficients de rotation W en fonction du radix et du mode MIMO, s'effectue de la façon suivant :

1. Retarder le compteur papillon b selon le radix afin de synchroniser au niveau des modules de calcul de la matrice les sorties des coefficients W et l'arrivée des données au niveau des multiplieurs.
2. Déterminer la correspondance du compteur pour $N_{max} = 8192$ selon N et le mode MIMO.
3. Déterminer les indices δ nécessaires en fonction du radix, du mode MIMO et du numéro de l'étage κ.
4. Accéder aux mémoires (de la figure 6.13) nécessaires selon le radix.
5. Dupliquer au besoin les coefficients selon le monde MIMO.

L'architecture obtenue est illustrée à la figure 6.14. Comme on le constate, la correspondance pour N_{max} dépend aussi du mode MIMO. En effet, le compteur papillon b varie de 0 à $\frac{N}{P_{D_{TFR}}} - 1$. Pour le mode SISO, la valeur maximale de N est de 8192 et $P_{D_{TFR}} = 8$. Pour le mode MIMO 2x2, la valeur maximale de N est de 4096 et $P_{D_{TFR}} = 4$. Pour le mode MIMO 4x4, la valeur maximale de N est de 2048 et $P_{D_{TFR}} = 2$. La valeur maximale du compteur papillon est donc de 1023, soit sur 10 bits en représentation binaire, peu importe le mode MIMO. Toutefois, pour une même valeur de N, la valeur maximale du compteur papillon sera respectivement 2 fois et 4 fois plus élevées que la valeur maximale du mode SISO pour les modes MIMO 2x2 et 4x4 respectivement.

De ce fait, la correspondance pour N_{max} doit aussi tenir compteur du mode MIMO. Le tableau 6.3 illustre la valeur du compteur B' qui est sur 12 bits, soit après correspondance à N_{max}.

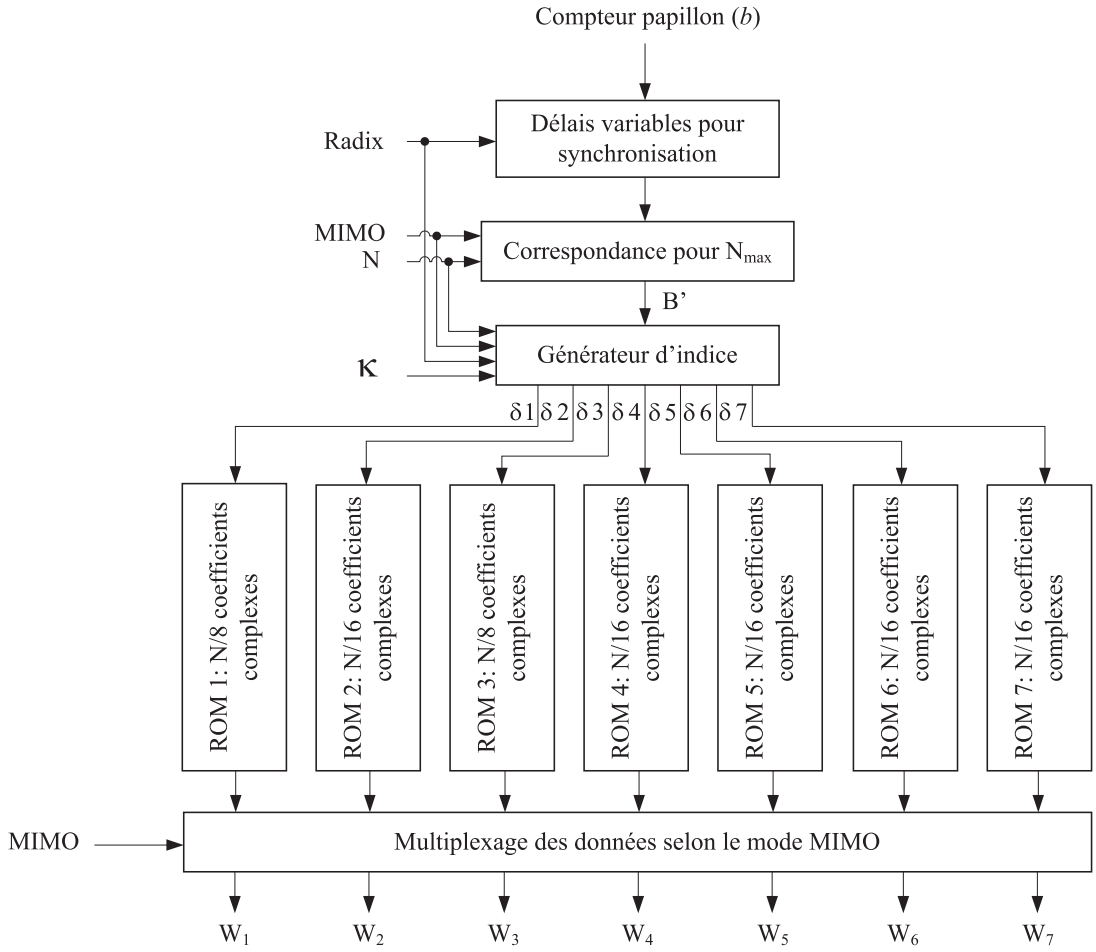

FIGURE 6.14 – Architecture du générateur des coefficients de rotation W

TABLE 6.3 – Valeur du compteur B', après correspondance du compteur papillon b selon N et le mode MIMO

N	$B'[11,0]$		
	mode SISO	mode MIMO 2x2	mode MIMO 4x4
8192	[00b[9,0]]	-	-
4096	[00b[8,0]0],	[0b[9,0]0]	-
2048	[00b[7,0]00]	[0b[8,0]00]	[b[9,0]00]
1024	[00b[6,0]000]	[0b[7,0]000]	[b[8,0]000]
512	[00b[5,0]0000]	[0b[6,0]0000]	[b[7,0]0000]
256	[00b[4,0]00000]	[0b[5,0]00000]	[b[6,0]00000]
128	[00b[3,0]000000]	[0b[4,0]000000]	[b[5,0]000000]
64	[00b[2,0]0000000]	[0b[3,0]0000000]	[b[4,0]0000000]

L'architecture du générateur d'indices δ est illustrée à la figure 6.15. A chaque étage radix-2^i, nous avons 2^κ sous TFR à $\frac{N}{2^\kappa}$ points, et donc nous devons générer 2^κ mêmes ensembles de coefficients W. À une fin d'illustration, prenons l'exemple d'une TFR à 8 points utilisant

l'algorithme radix-2. A chacun des $\log_2 N = 3$ étages, $\frac{N}{2} = 4$ coefficients W sont nécessaires tel qu'illustré au tableau 6.4. Afin de reproduire ce schéma, on considère seulement les $10 - \kappa$ bits les plus significatifs du vecteur B' après décalage de κ bits vers la gauche. Ainsi, le compteur obtenu à la sortie des multiplexeurs d'entrée (selon mode MIMO) de la figure 6.15 comptera 2^κ fois jusqu'à $\frac{N/P_{D_{TFR}}}{2^\kappa} - 1$. Chaque valeur du compteur (et donc de l'indice) aura été multipliée par 2^κ après décalage vers la gauche comme illustré au tableau 6.4.

TABLE 6.4 – Exemple coefficients W pour $N = 16$ avec l'algorithme radix-2

	Étage 0	(Étage 1 ($\kappa = 1$)		Étage 2 ($\kappa = 2$)	
	($\kappa = 0$)	Avant décalage	Après décalage	Avant décalage	Après décalage
W_1	W_8^0	W_4^0	W_8^0	W_2^0	W_8^0
W_2	W_8^1	W_4^1	W_8^2	W_2^1	W_8^4
W_3	W_8^2	W_4^0	W_8^0	W_2^0	W_8^0
W_4	W_8^3	W_4^1	W_8^2	W_2^1	W_8^4

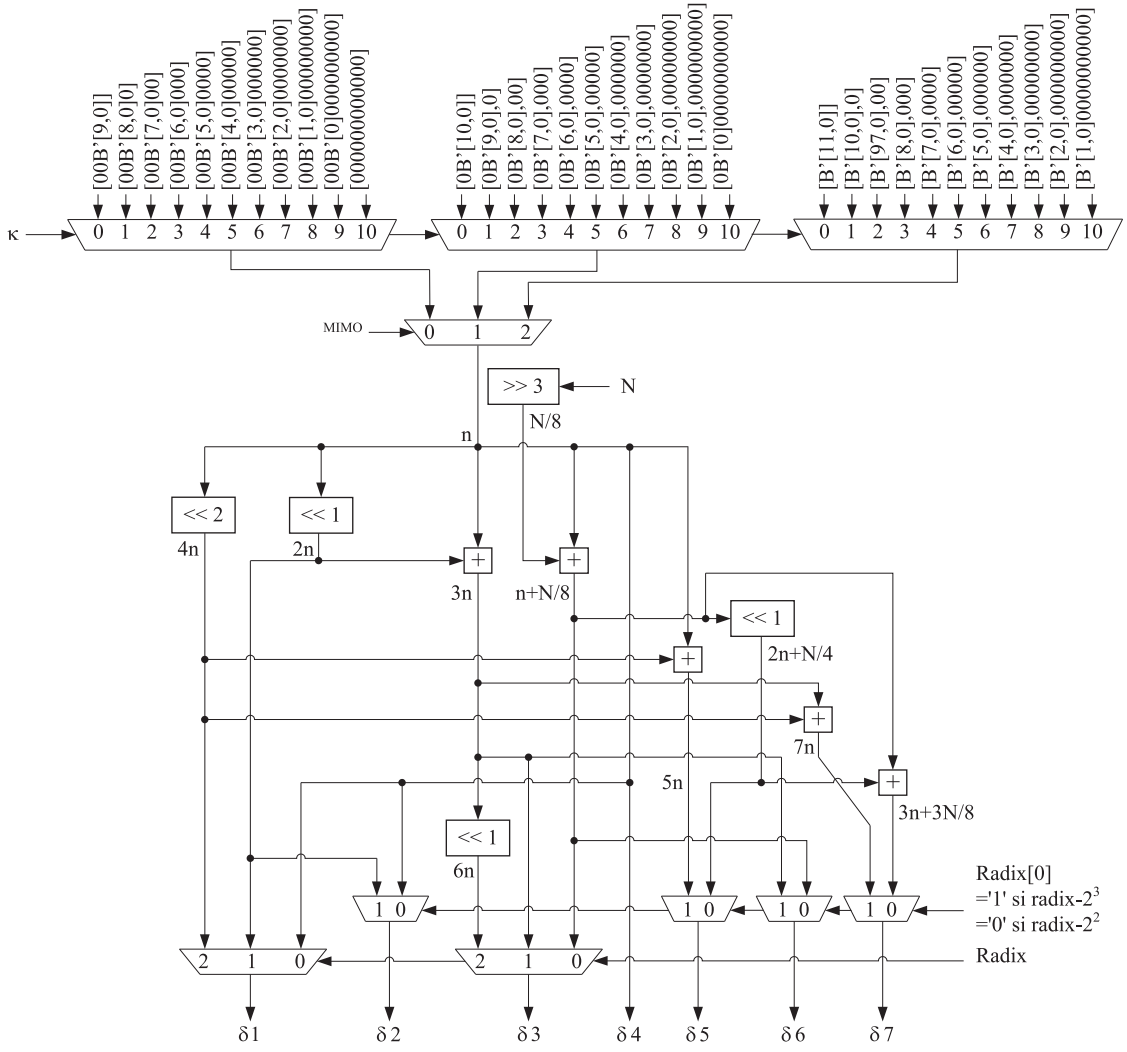

FIGURE 6.15 – Architecture du générateur d'indices δ des coefficients de rotation W

Le dernière étape de la génération des coefficients de rotation W consiste à considérer seulement les coefficients nécessaires. Dans le cas du mode MIMO 2x2, les algorithmes utilisés sont le radix-2^2 et le radix-2. Pour le radix-2^2, seuls les coefficients W_1, W_2 et W_3 doivent être sélectionnés puisque ces mêmes coefficients sont dupliqués pour la deuxième TFR en parallèle pour un total de 6 coefficients. Pour le radix-2, seuls les coefficients W_1 et W_3 doivent être sélectionnés puisque ces mêmes coefficients sont dupliqués pour la deuxième TFR en parallèle pour un total de 4 coefficients. Finalement, dans le cas du mode MIMO 4x4, seul le coefficient W_1 du radix-2 doit être sélectionné et répliqué 4 fois pour les quatre TFR en parallèle. La figure 6.16 illustre l'architecture du multiplexage des coefficients de rotation W selon le mode MIMO.

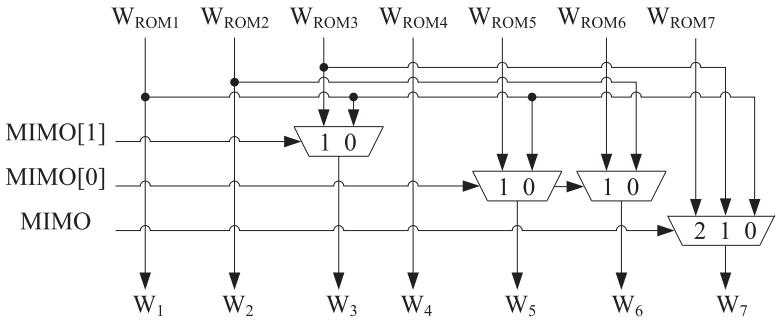

FIGURE 6.16 – Architecture du multiplexage des coefficients de rotation W selon le mode MIMO

6.3 Mémoires des coefficients de mise en forme IOTA

La fonction IOTA a une longueur de $2M_{max}L_{max}$ (ou $N_{max}L_{max}$) coefficients. Grâce à la symétrie de la fonction, nous pouvons stocker que $M_{max}L_{max}$ coefficients. Les valeurs de ces coefficients dépendent de la valeur de L, N et τ_0 qui correspond à un temps symbole OFDM/OQAM.

À chaque cycle, $P_{D_{TFR}}L$ coefficients doivent être lus dans les mémoires des coefficients pour $L = 2$ ou 4 et $P_{D_{TFR}}\frac{L_{max}}{2}$ pour $L = L_{max} = 8$. Pour rappel, le filtrage pour L_{max} se fait en deux temps. De plus, $P_D L = 8L$ coefficients sont envoyés vers la matrice. Il y a donc $\frac{P_D}{P_{D_{TFR}}}$ fois plus de coefficients envoyés vers la matrice que de coefficients obtenus du fait de la duplication des coefficients selon le mode MIMO. Le tableau 6.5 donne le nombre de coefficients lus dans les mémoires des coefficients selon les différentes configurations.

Les $M_{max}L_{max}$ coefficients sont mémorisés dans $L_{max} = 8$ blocs mémoires de M_{max} points. La figure 6.17 illustre la demi-fonction IOTA et les différents blocs mémoires. Selon la valeur

TABLE 6.5 – Nombre de coefficients lus dans les mémoires des coefficients à chaque cycle selon la configuration

	Longueur de troncature		
Mode MIMO	$L = 2$	$L = 4$	$L = 8$
SISO ($P_{D_{TFR}} = 8$)	16	32	32
MIMO 2x2 ($P_{D_{TFR}} = 4$)	8	16	16
MIMO 4x4 ($P_{D_{TFR}} = 2$)	4	8	8

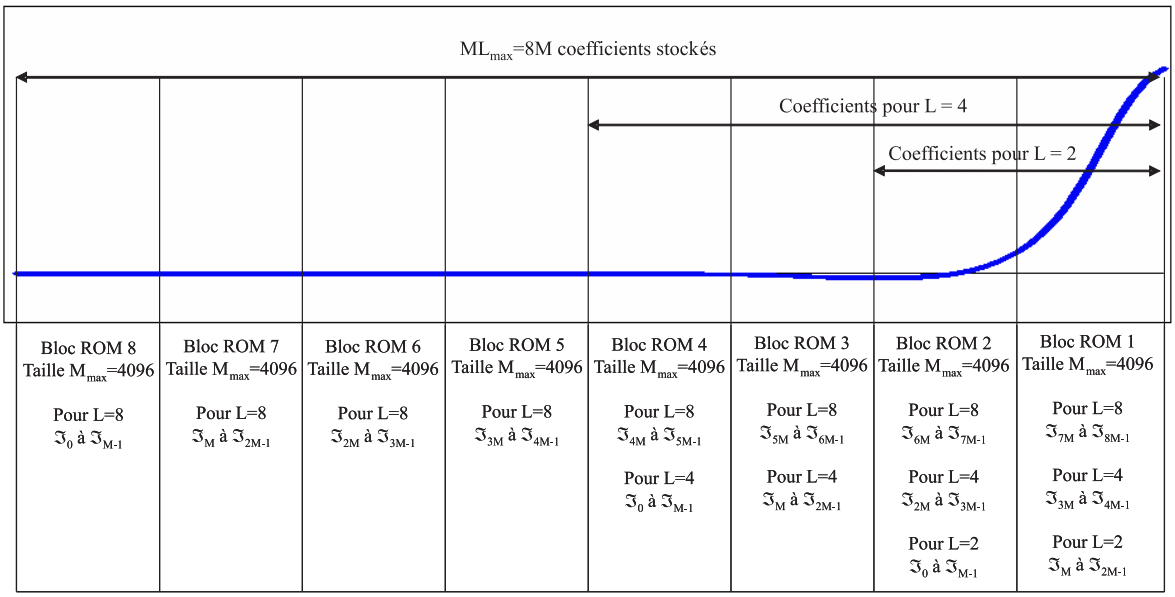

FIGURE 6.17 – Demi fonction IOTA mémorisée dans les $L_{max} = 8$ blocs ROM et les valeurs des indices des coefficients IOTA selon L

de L, les coefficients sont mémorisés dans les blocs 1 à L, les autres blocs étant désactivés. De plus, pour une valeur de L donnée, le coefficient IOTA $\Im_{indice}$ sera situé dans des blocs mémoires différents. Par exemple, le premier coefficient $\Im_0$ sera mémorisé dans le bloc 2, 4 ou 8 selon la valeur de L. La figure 6.17 illustre les indices des coefficients selon L.

Lors de l'adressage des mémoires pour L égal à 2 ou à 4, une correspondance est effectuée par rapport à la référence, soit aux indices de $L_{max} = 8$. Ainsi, selon la valeur de L, on ne considère que les coefficients dont la valeur des indices est située entre $(L_{max} - L)M$ et $(L_{max} + L)M$, c'est-à-dire $(8 - L)M$ et $(8 + L)M$. De plus, selon la valeur de N, on ne considère que les coefficients aux indices multiples de $\frac{N_{max}}{N} = \frac{8192}{N}$. Ainsi, pour $N = 4096$, on ne considère que les coefficients aux indices multiples de 2. Ceci correspond à sous-échantillonner les coefficients IOTA pour des valeurs de N inférieures à N_{max}. Les coefficients sont mémorisés pour une valeur normalisée de τ_0.

Dans chaque bloc mémoire, $P_{D_{TFR}}$ coefficients doivent être obtenus par cycle pour $L = 2$ ou 4. Ainsi, selon le mode MIMO, 2, 4 ou 8 (valeur de $P_{D_{TFR}}$) coefficients seront obtenus dans chaque bloc. En réalité, $\frac{P_{D_{TFR}}}{2}$ coefficients correspondent à la première moitié de la fonction IOTA, et $\frac{P_{D_{TFR}}}{2}$ à la seconde moitié de la fonction. La figure 6.18 présente le cas simple de la fonction IOTA pour $N = 8$ et $L = 4$ pour illustration, soit 32 coefficients. Dans cet exemple, on considère que $P_{D_{TFR}} = 2$; de ce fait $2L = 8$ coefficients sont lus à chaque cycle. Cette figure montre la correspondance des coefficients de la deuxième moitié de la fonction vers la première moitié de la fonction.

Dans cet exemple, les coefficients IOTA voulus de la première moitié de la fonction au premier cycle sont $\Im_0$, $\Im_4$, $\Im_8$, $\Im_{12}$. Ceux de la deuxième moitié sont $\Im_{16}$, $\Im_{20}$, $\Im_{24}$, $\Im_{28}$. On constate que par la symétrie de la fonction IOTA, lire les coefficients de la première moitié de la fonction consiste à parcourir les blocs mémoires dans le sens croissant des adresses, soit de 0 à $M-1$, tandis que pour les coefficients de la deuxième moitié, cela revient à parcourir les blocs mémoires dans

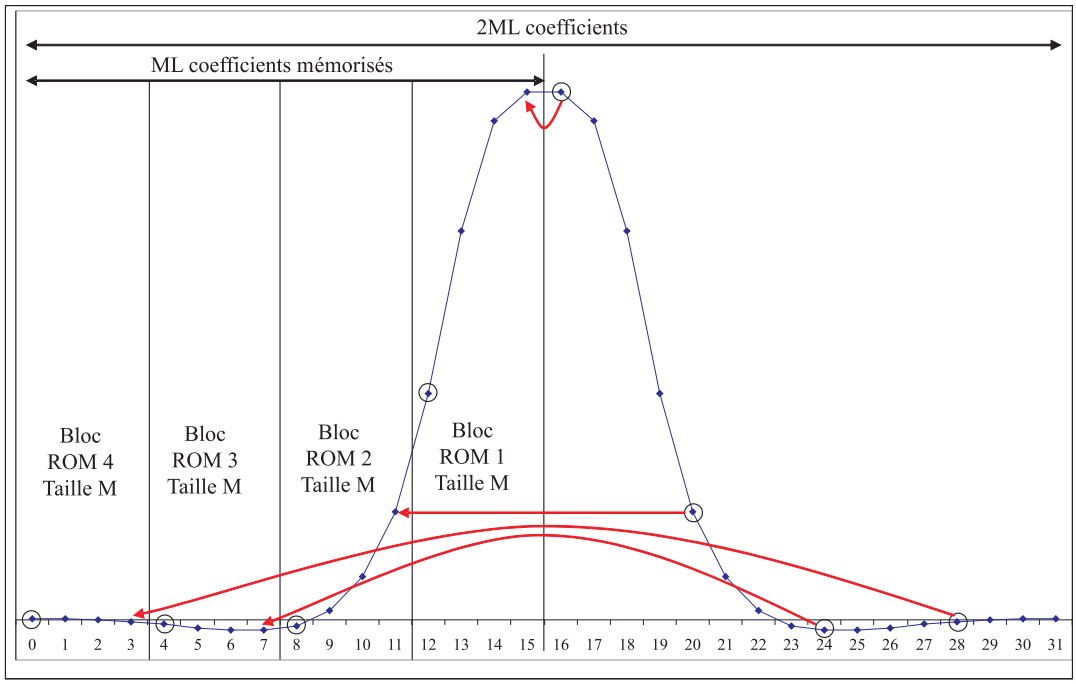

FIGURE 6.18 – Détermination des coefficients IOTA par symétrie (illustration pour $N = 16$ ($M = 8$) et $L = 4$

le sens décroissant de l'adresse, ce qui considérant l'adresse représentée en binaire, correspond au complément de l'adresse. Par exemple, à la figure 6.18, deux coefficients doivent être lus dans le bloc ROM 4 au premier cycle, $\Im_0$ pour la première moitié de la fonction, et $\Im_3$ pour la deuxième moitié de la fonction. Ceci correspond respectivement aux adresses 00 et 11 du bloc ROM 4.

Pour L_{max}, du fait que le filtrage est réalisé en deux temps, $\frac{P_{D_{TFR}}}{2}$ coefficients doivent être lus par cycle dans chaque bloc mémoire. Dans un premier temps, les coefficients de la première moitié sont envoyés vers la matrice, et par la suite ceux de la deuxième moitié. Les blocs mémoires 5 à 8 n'étant utilisés que pour L_{max}, deux types de blocs sont utilisés dans l'architecture. Les blocs mémoires de type 1 (appelé Bloc mémoire ROM à 8 sorties) concernent les blocs 1 à 4. Pour ces derniers, jusqu'à 8 coefficients peuvent être lus pour L égale à 2 ou 4, et maximum 4 pour L_{max}. Les blocs mémoires de type 2 (appelé Bloc mémoire ROM à 4 sorties) concernent les blocs 5 à 8, et jusqu'à 4 coefficients peuvent être lus à chaque cycle. La figure 6.19 illustre l'architecture globale des mémoires des coefficients IOTA. Pour une meilleure compréhension, nous allons d'abord présenter dans la sous-section suivante, la version triviale, soit le bloc mémoire ROM à 4 sorties. Les signaux de contrôle de la figure 6.19 seront aussi abordés.

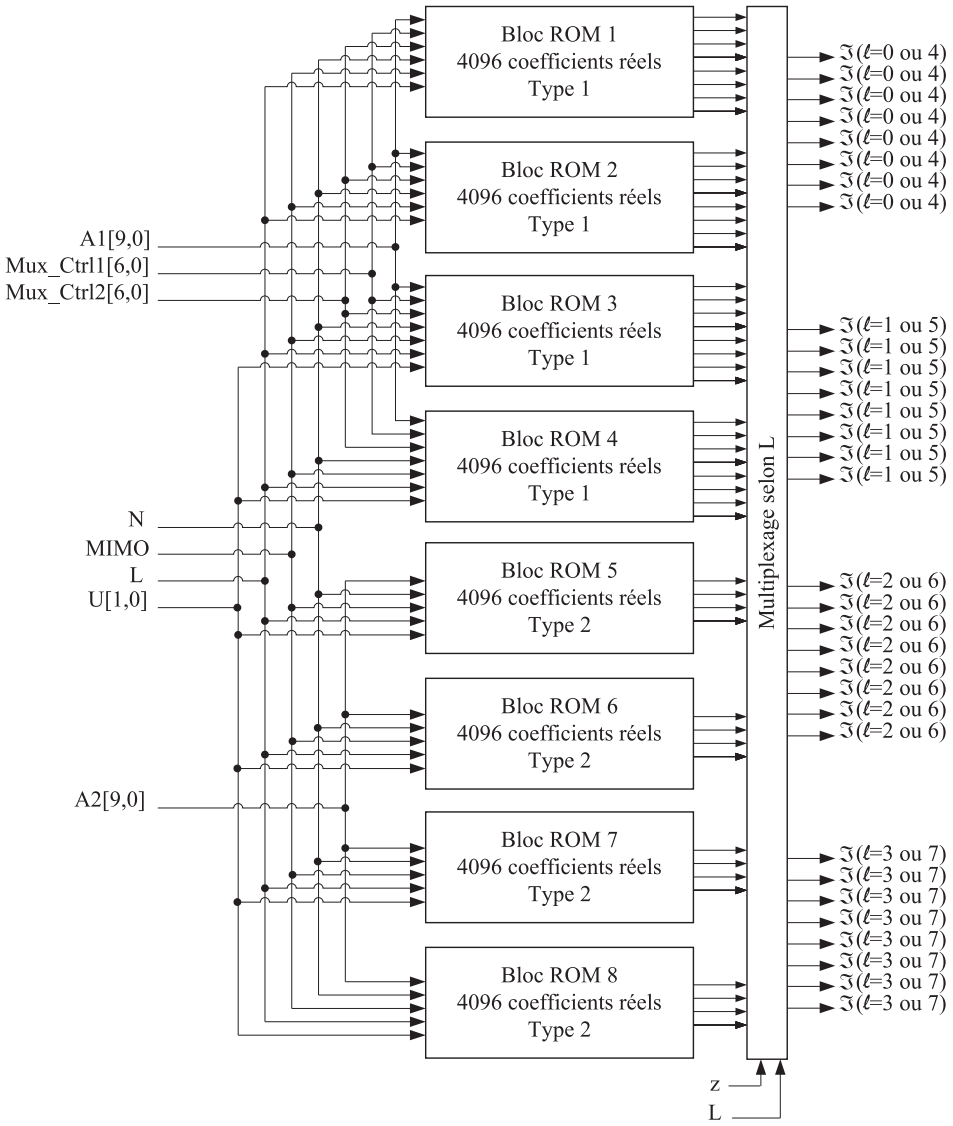

FIGURE 6.19 – Architecture globale des mémoires des coefficients IOTA

6.3.1 Bloc mémoire ROM IOTA à 4 sorties (Type 2)

Afin de lire jusqu'à 4 coefficients par bloc mémoire, il est nécessaire de subdiviser chaque bloc en 4 bancs de mémoires de $\frac{M_{max}}{4}$ coefficients, donnant ainsi la possibilité de lire un coefficient par banc. La figure 6.20 illustre l'architecture d'un bloc de type 2.

L'équation 5.5 présente les coefficients IOTA impliqués dans l'opération de filtrage, soit $\Im_{x+p\frac{N}{P_{D_{TFR}}}+2M\ell}$. En tenant compte de la symétrie de la fonction, les $P_{D_{TFR}}L$ coefficients désirés seront situés dans les différents bancs des blocs 1 à ℓ à l'indice x, soit le compteur papillon. L'adresse des coefficients $A2$ des figures 6.19 et 6.20 est dérivée en partie du compteur papillon après correspondance à N_{max}, soit B' du tableau 6.3. On a :

$$A1[9,0] = \begin{cases} B'[9,0] & \text{si } z = 0 \\ B''[9,0] & \text{si } z = 1. \end{cases} \tag{6.8}$$

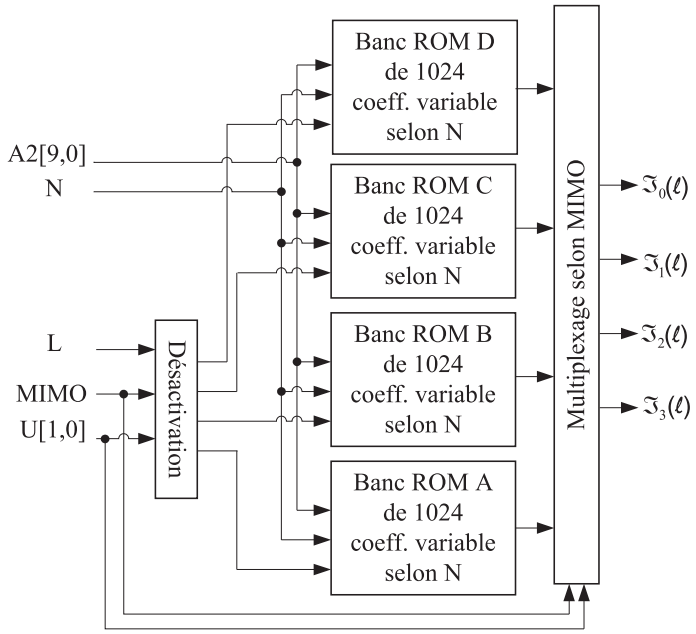

FIGURE 6.20 – Architecture d'un bloc mémoire de type 2 (blocs 5 à 8)

Le signal B'' est obtenu après correspondance à N_{max} du complément du compteur papillon b. Le tableau 6.6 illustre la valeur de B''. Le signal z indique pour $L = L_{max}$, s'il s'agit du filtrage par la première partie de la fonction IOTA ($z = 0$), ou le filtrage par la deuxième moitié de la fonction ($z = 1$). Ainsi, on parcourt les bancs de mémoires dans le sens croissant pour lire les coefficients de la première moitié de la fonction, et dans l'ordre décroissant pour ceux de la deuxième moitié.

TABLE 6.6 – Valeur du compteur B''

N	$B''[11,0]$		
	mode SISO	mode MIMO 2x2	mode MIMO 4x4
8192	$[00\overline{b[9,0]}]$	-	-
4096	$[00\overline{b[8,0]}0]$,	$[0\overline{b[9,0]}0]$	-
2048	$[00\overline{b[7,0]}00]$	$[0\overline{b[8,0]}00]$	$[\overline{b[9,0]}00]$
1024	$[00\overline{b[6,0]}000]$	$[0\overline{b[7,0]}000]$	$[\overline{b[8,0]}000]$
512	$[00\overline{b[5,0]}0000]$	$[0\overline{b[6,0]}0000]$	$[\overline{b[7,0]}0000]$
256	$[00\overline{b[4,0]}00000]$	$[0\overline{b[5,0]}00000]$	$[\overline{b[6,0]}00000]$
128	$[00\overline{b[3,0]}000000]$	$[0\overline{b[4,0]}000000]$	$[\overline{b[5,0]}000000]$
64	$[00\overline{b[2,0]}0000000]$	$[0\overline{b[3,0]}0000000]$	$[\overline{b[4,0]}0000000]$

Selon la valeur de $P_{D_{TFR}}$ et donc du mode MIMO, les 4 bancs auront les 3 configurations suivantes :

1. Mode SISO : Les 4 bancs sont en parallèle et un coefficient est obtenu par banc.
2. Mode MIMO 2x2 : Les 4 bancs sont groupés 2 par 2 (respectivement A-B et C-D) en série et un coefficient est obtenu par groupe. On obtient ainsi l'équivalent de 2 bancs de taille $\frac{M_{max}}{2}$ en parallèles.
3. Mode MIMO 4x4 : Les 4 bancs sont considérés en série et un coefficient est obtenu à chaque cycle. On obtient ainsi l'équivalent d'un banc de taille M_{max}

Ces différentes configurations s'expliquent par le fait que selon le degré de parallélisme $P_{D_{TFR}}$, le bloc mémoire de M_{max} points, est subdivisé en 2 ou 4 mémoires de tailles 2 ou 4 fois plus petites respectivement tel qu'illustré à la figure 6.21. Les 4 bancs sont actifs simultanément seulement dans le mode SISO. Dans le mode MIMO 4x4, les coefficients sont d'abord lus dans le banc A, puis B, C et finalement D. Ainsi, les quatre bancs ne sont pas actifs au même moment et peuvent être désactivés par *clock gating*. Le signal U sur 2 bits permet de sélectionner un des 4 bancs et d'activer seulement l'horloge du banc nécessaire. Ce signal correspond aux bits [10] et [11] du signal B'. Soit l'équivalent du modulo $\frac{M_{max}}{4}$ du compteur papillon décallé selon N . On obtient alors $U[1,0] = 00$ pour le banc A, 01 pour B, 10 pour C et finalement 11 pour le banc D. Pour le mode MIMO 2x2, les bancs A et C sont actifs au même moment, soit pour $U[0] = 0$, et B-D lorsque $U[0] = 1$,

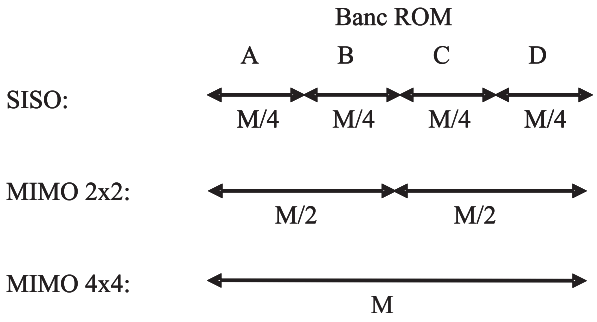

FIGURE 6.21 – Configuration d'un bloc mémoire de type 2 selon le mode MIMO

La figure 6.22 illustre le multiplexage selon le mode MIMO de la figure 6.20. Pour le mode SISO, les 4 sorties des bancs sont sélectionnées. Pour le mode MIMO 2x2, les sorties des bancs A et C sont sélectionnées lorsque $U[0] = 0$ et les bancs B et D lorsque $U[0] = 1$. Ces deux sorties sont dupliquées pour un total de 4 sorties par bloc mémoire. Pour le mode MIMO 4x4, les 4 bancs sont sélectionnés alternativement selon la valeur du signal U. On obtient ainsi un coefficient qui est répercuté sur les 4 sorties du bloc mémoire ROM.

Chaque banc de ROM de taille $\frac{M_{max}}{4} = 1024$ coefficients est constitué de 8 sous ROM afin d'obtenir des tailles mémoires allant $\frac{M_{min}}{4} = 8$ à 1024 selon les différentes configurations possibles tel que présenté au tableau 6.1. Pour cela, nous utilisons la même approche que pour les RAM des échantillons présentée à la figure 6.2. La figure 6.23 illustre l'architecture du banc de ROM de type 2. Comme on le constate, contrairement au banc RAM de la figure 6.2, l'adressage et le contrôle des multiplexeurs de sortie se font selon les bits les plus significatifs du vecteur d'adresse. En effet, nons proposons de mémoriser les coefficients de la fonction IOTA suivant une certaine stratégie (indices pairs et impairs séparés) nous permettant d'avoir selon N, le nombre voulu de coefficients IOTA (sous-échantillonnage des coefficients). Ainsi, l'adressage est effectué selon la parité de l'adresse et non selon l'ordre naturel (croissant) du vecteur d'adresse.

La première mémoire de 512 données contient tous les coefficients $\Im_{indice}$ des $\frac{M_{max}}{4} = 1024$ coefficients aux indices impairs. Les coefficients aux indices pairs restants sont placés dans les 7 mémoires restantes. Selon la parité de l'indice modulo 2, la mémoire de 256 données contient les coefficients impairs des 512 échantillons restant et les 6 mémoires restantes contiennent les coefficients pairs et ainsi de suite. Ainsi, pour chaque taille N, les coefficients nécessaires seront dans les mémoires non désactivées.

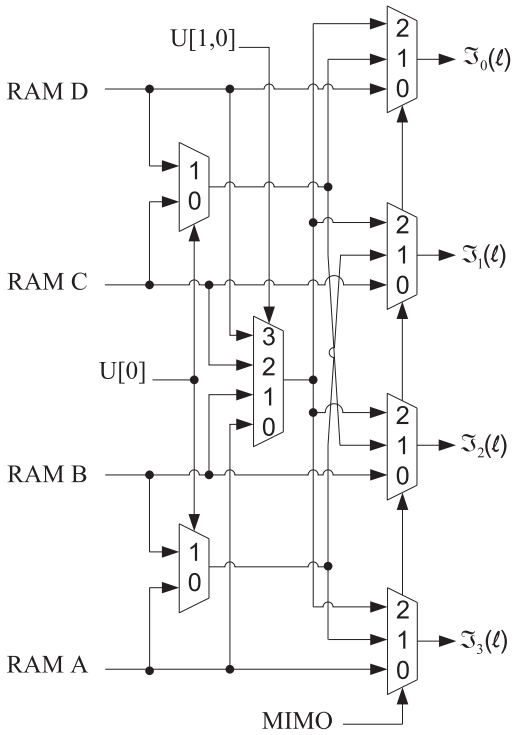

FIGURE 6.22 – Multiplexage des sorties selon le mode MIMO pour les blocs mémoires de type 2

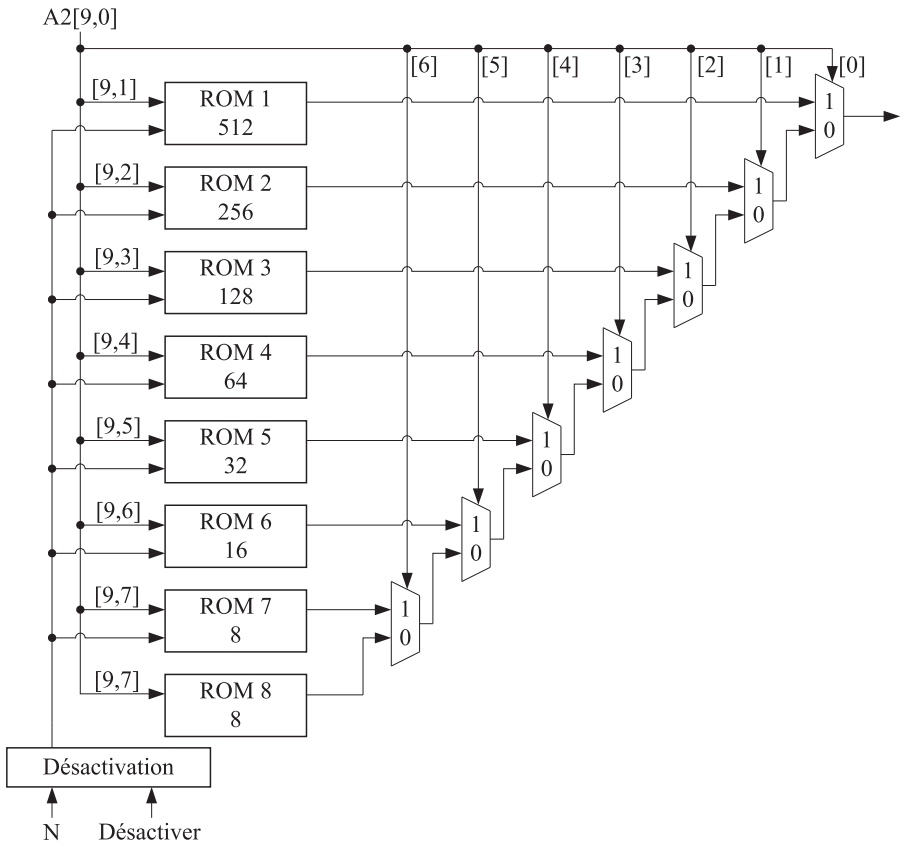

FIGURE 6.23 – Architecture d'un banc de ROM pour les coefficients IOTA de type 2

Par exemple, pour $N = 64$, tous les coefficients nécessaires sont dans les dernières mémoires de 8 points. Pour illustration, prenons l'exemple où $N_{max} = 128$, soit $\frac{M_{max}}{4} = 16$, les coefficients d'un même bloc ROM sont alors mémorisés dans 3 ROM selon les schémas du tableau 6.7. Ainsi, tous les coefficients de la fonction IOTA pour la plus petite valeur de N, sont dans la ROM 3, soit tous les coefficients aux indices multiples de $\frac{N_{max}}{N_{min}} = \frac{128}{32} = 4$ ($N_{min} = 32$ dans cet exemple).

TABLE 6.7 – Stratégie de mémorisation des coefficients IOTA pour les mémoires ROM de type 2, exemple avec $N_{max} = 128$

ROM 1	
Adresse $A2$	Coef.
0	$\Im_1$
1	$\Im_3$
2	$\Im_5$
3	$\Im_7$
4	$\Im_9$
5	$\Im_{11}$
6	$\Im_{13}$
7	$\Im_{15}$

ROM 2	
Adresse $A2$	Coef.
0	$\Im_2$
1	$\Im_6$
2	$\Im_{10}$
3	$\Im_{14}$

ROM 3	
Adresse $A2$	Coef.
0	$\Im_0$
1	$\Im_4$
2	$\Im_8$
3	$\Im_{12}$

6.3.2 Bloc mémoire ROM IOTA à 8 sorties (Type 1)

Dans le cas des blocs mémoires de type 1, jusqu'à 8 coefficients peuvent être obtenus en sorties. La figure 6.24 illustre l'architecture d'un bloc mémoire ROM de type 1. Afin de lire 2 coefficients dans chaque banc A à D, le deuxième coefficient correspondant au coefficient obtenu par symétrie, une première approche serait de subdiviser les différentes sous-mémoires des bancs en deux mémoires plus petites. Par exemple deux mémoires de 256 coefficients fonctionnant comme une mémoire de 512 coefficients.

Nous proposons toutefois d'utiliser un schéma particulier et original pour résoudre cette problématique de lectures multiples dans les bancs mémoires. En effet, l'architecture utilise plutôt des mémoires de taille deux fois plus petites, mais ayant des longueurs de mots deux fois plus longues. Ainsi, on aura par exemple une mémoire ROM de 256 points dont la longueur de chaque point est de $2b_W$ au lieu d'une ROM de 512 points de longueur b_W. Ceci nous permet, entre autres, d'avoir une surface plus petite, de fait qu'on aura un seul décodeur d'adresse au lieu de deux. Le décodeur d'adresse d'une mémoire deux fois plus large sera plus petit en surface que deux décodeurs d'adresses de mémoires deux fois moins larges.

Pour cela, il faut considérer les coefficients IOTA dans chaque bloc ROM A à D comme étant regroupés deux par deux. À titre d'exemple, le tableau 6.8 reprend le cas de figure du tableau 6.7 pour $N_{max} = 128$, mais selon l'approche mémoire de type 1.

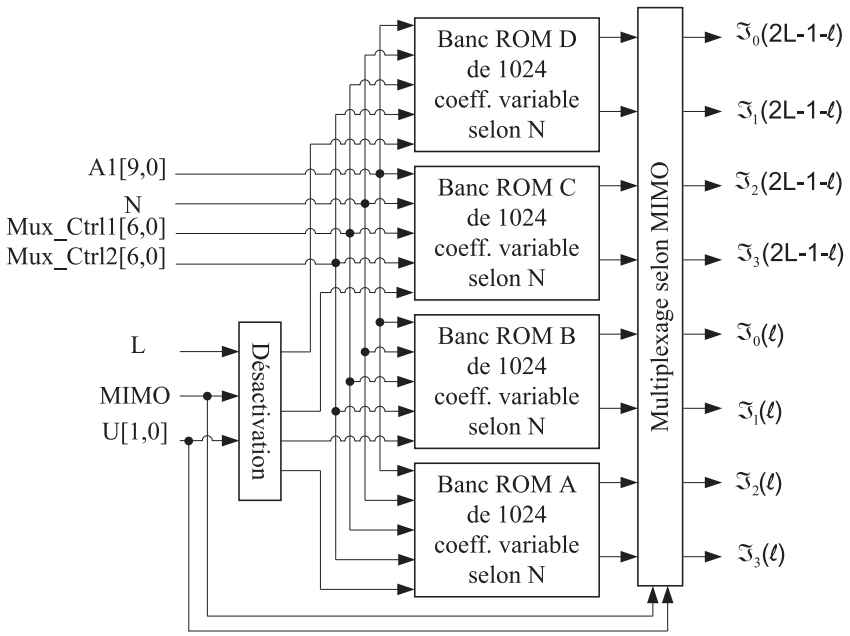

FIGURE 6.24 – Architecture d'un bloc mémoire de type 1 (blocs 1 à 4)

TABLE 6.8 – Stratégie de mémorisation des coefficients IOTA pour les mémoires ROM de type 1, exemple avec $N_{max} = 128$

ROM 1	
Adresse $A1$	Coef.
0	$(\Im_{15}; \Im_1)$
1	$(\Im_{13}; \Im_3)$
2	$(\Im_{11}; \Im_5)$
3	$(\Im_9; \Im_7)$

ROM 2	
Adresse $A1$	Coef.
0	$(\Im_{14}; \Im_2)$
1	$(\Im_{10}; \Im_6)$

ROM 3	
Adresse $A1$	Coef.
0	$(\Im_{12}; \Im_0)$
1	$(\Im_8; \Im_4)$

Pour la plus petite valeur de $N = 32$, tous les coefficients aux indices multiples de $\frac{N_{max}}{N_{min}} = \frac{128}{32} = 4$ sont dans la dernière mémoire, et deux coéfficients peuvent être lus par cycle. En effet, dans ce cas, nous avons besoin des coefficients dans l'ordre suivant :

Compteur papillon b en binaire $b_3b_2b_1b_0$:	0000	0001	0010	0011
B' en binaire $B'_3B'_2B'_1B'_0$:	0000	0100	1000	1100
Adresse $A1$:	0	1	1	0
Coeff. de la $1^{ère}$ moitié de la fonction :	$\Im_0$	$\Im_4$	$\Im_8$	$\Im_{12}$
ROM numéro :	3	3	3	3
Coeff. de la $2^{ème}$ moitié de la fonction (par symétrie) :	$\Im_{12}$	$\Im_8$	$\Im_4$	$\Im_0$
ROM numéro :	3	3	3	3

Dans cet exemple, le compteur papillon b compte de 0 à $\frac{N}{P_{D_{TFR}}} - 1 = \frac{32}{8} - 1 = 3$. Le signal B' correspond au compteur après décalage vers la gauche de $\log_2\left(\frac{N_{max}}{N}\right) = \log_2\left(\frac{128}{32}\right) = 2$ positions. L'adresse $A1$ est égale à B'_2 si $B'_3 = 0$ et $\overline{B'_2}$ si $B'_3 = 1$.

On constate bien que tous les coefficients désirés pour la plus petite valeur de N sont déjà regroupés deux par deux dans la plus petite mémoire, soit les couples $(\Im_{12}; \Im_0)$ et $(\Im_8; \Im_4)$ pour cet exemple. Toutefois, il faut permuter l'ordre à la moitié du compteur papillon du fait de la symétrie de la fonction. On constate aussi qu'il est nécessaire afin d'obtenir tous les coefficients

désirés, de parcourir la mémoire dans le sens croissant pour $b = 0$ à $\frac{\frac{N}{P_{D_{TFR}}}}{2} - 1$, soit la première moitié du compteur, et dans le sens décroissant pour la second moitié.

Pour des valeurs de N différentes de N_{min}, les deux coefficients désirés ne sont plus situés dans la même mémoire mais deux mémoires différentes. Pour $N = 64$ à titre d'exemple, les coefficients désirés sont les suivants (indices pairs) :

$b_3b_2b_1b_0$:	0000	0001	0010	0011	0100	0101	0110	0111
$B'_3B'_2B'_1B'_0$:	0000	0010	0100	0110	1000	1010	1100	1110
Adresse $A1$:	0	0	1	1	1	1	0	0
Coeff. de la $1^{ère}$ moitié :	$\Im_0$	$\Im_2$	$\Im_4$	$\Im_6$	$\Im_8$	$\Im_{10}$	$\Im_{12}$	$\Im_{14}$
ROM numéro :	3	2	3	2	3	2	3	2
Coeff. de la $2^{ème}$ moitié :	$\Im_{14}$	$\Im_{12}$	$\Im_{10}$	$\Im_8$	$\Im_6$	$\Im_4$	$\Im_2$	$\Im_0$
ROM numéro :	2	3	2	3	2	3	2	3

Dans ce cas, le compteur papillon b compte de 0 à $\frac{N}{P_{D_{TFR}}} - 1 = \frac{64}{8} - 1 = 7$. Le signal B' correspond au compteur après décalage vers la gauche de 1 position et l'adresse $A1$ est toujours égale à B'_2 si $B'_3 = 0$ et $\overline{B'_2}$ si $B'_3 = 1$.

Un multiplexage des coefficients est donc nécessaire afin de sélectionner les deux coefficients nécessaires à la sortie des sous-ROMs dans chacun des bancs A à D. La figure 6.25 illustre le banc de mémoire de type 1 utilisant des ROMs double largeur et de taille deux fois plus petites. Le vecteur d'adresse $A1$ est égale à :

$$A1[9,0] = \begin{cases} [B'[9,0]] & \text{si } B'[9] = 0 \\ [B'[9]B''[8,0]] & \text{si } B'[9] = 1. \end{cases} \tag{6.9}$$

On remarque que les deux coefficients lus dans chaque ROM sont multiplexés grâce au bit le plus significatif du vecteur d'adresse ($A1[9]$). Ceci correspond à la permutation des coefficients pour la seconde moitié du compteur papillon correspondant aussi à l'adresse de lecteure $A1$. Les signaux Mux_Ctrl1 et Mux_Ctrl2 permettent de contrôler les multiplexeurs de sortie de chaque banc de ROM. Ils sont égaux à :

$$Mux\_Ctrl1[6,0] = B'[6,0] \tag{6.10}$$

$$Mux\_Ctrl2[6,0] = B''[6,0] \tag{6.11}$$

La figure 6.26 illustre le multiplexage des coefficients selon le mode MIMO à la sortie des bancs de ROM A à D tel qu'illustré à la figure 6.24. Ce multiplexage fonctionne de la même manière que ceux du type 2. Il faut toutefois noter que les deuxièmes coefficients $\Im_2$ des quatres bancs doivent être inversés à l'entrée des multiplexeurs comparativement aux premiers coefficients $\Im_1$ tel qu'illustré à la figure 6.26 . Ceci s'explique par la symétrie de la fonction IOTA, le début de la première moitié de la fonction IOTA correspondant à la fin de la deuxième moitié de la fonction et vice versa.

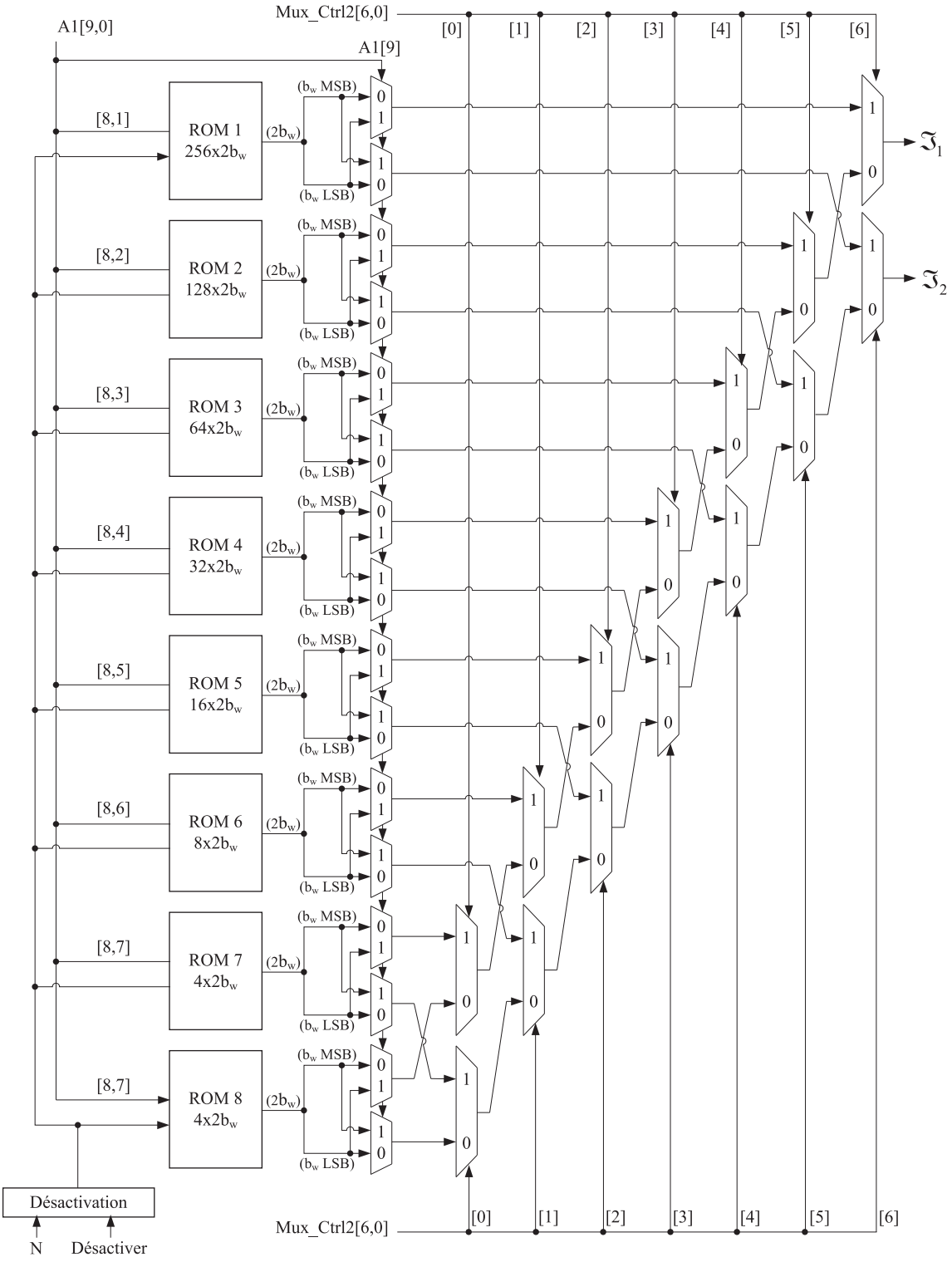

FIGURE 6.25 – Architecture d'un banc de ROM pour les coefficients IOTA de type 1

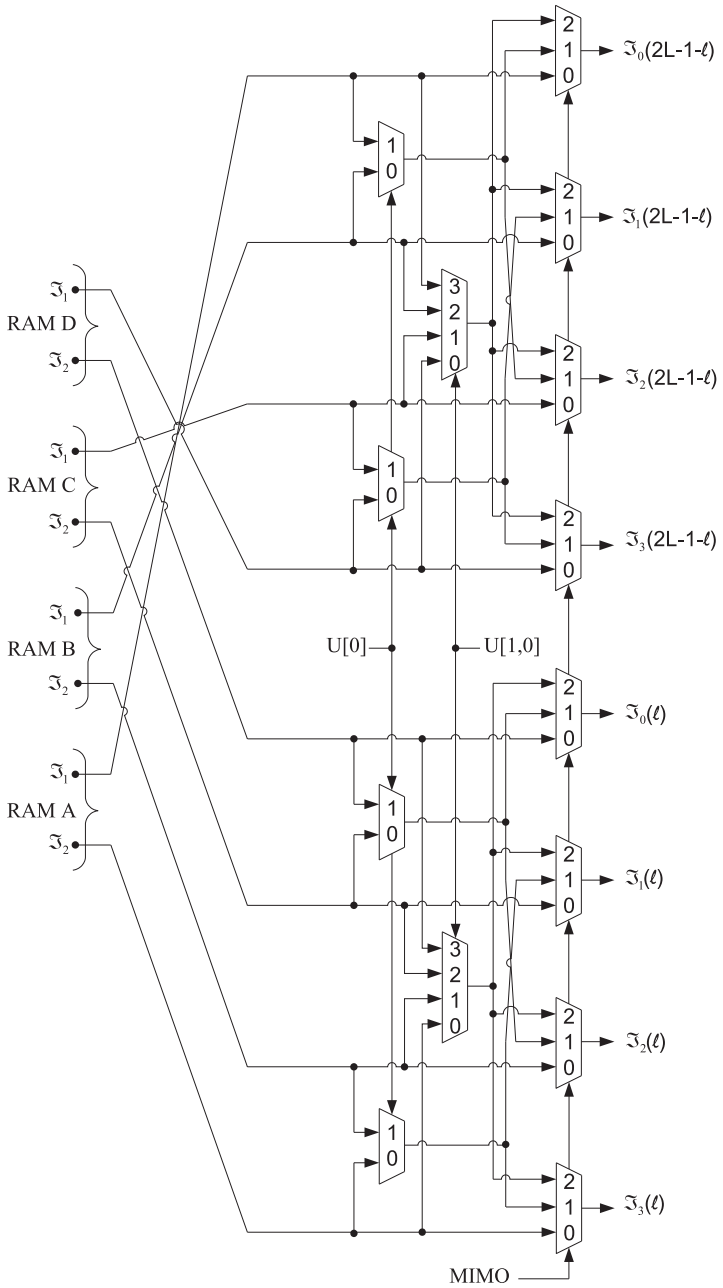

FIGURE 6.26 – Multiplexage des sorties selon le mode MIMO pour les blocs mémoires de type 1

6.3.3 Interface de sortie des mémoires IOTA

Les différents coefficients obtenus dans chaque bloc mémoire doivent être orientés vers les bons modules de calcul de la matrice. À cette fin, un multiplexage selon le paramètre L est effectué à la sortie des blocs comme illustré à la figure 6.19. La figure 6.27 illustre l'architecture du module de multiplexage. Le premier multiplexage selon le signal z permet de sélectionner, dans le cas $L = L_{max}$, les coefficients de la première moitié de la fonction IOTA pour $z = 0$, et ceux de la deuxième moitié pour $z = 1$. Finalement, selon la valeur de L, la dernière colonne de multiplexeurs permet d'orienter les coefficents désirés vers les modules de calcul de la matrice.

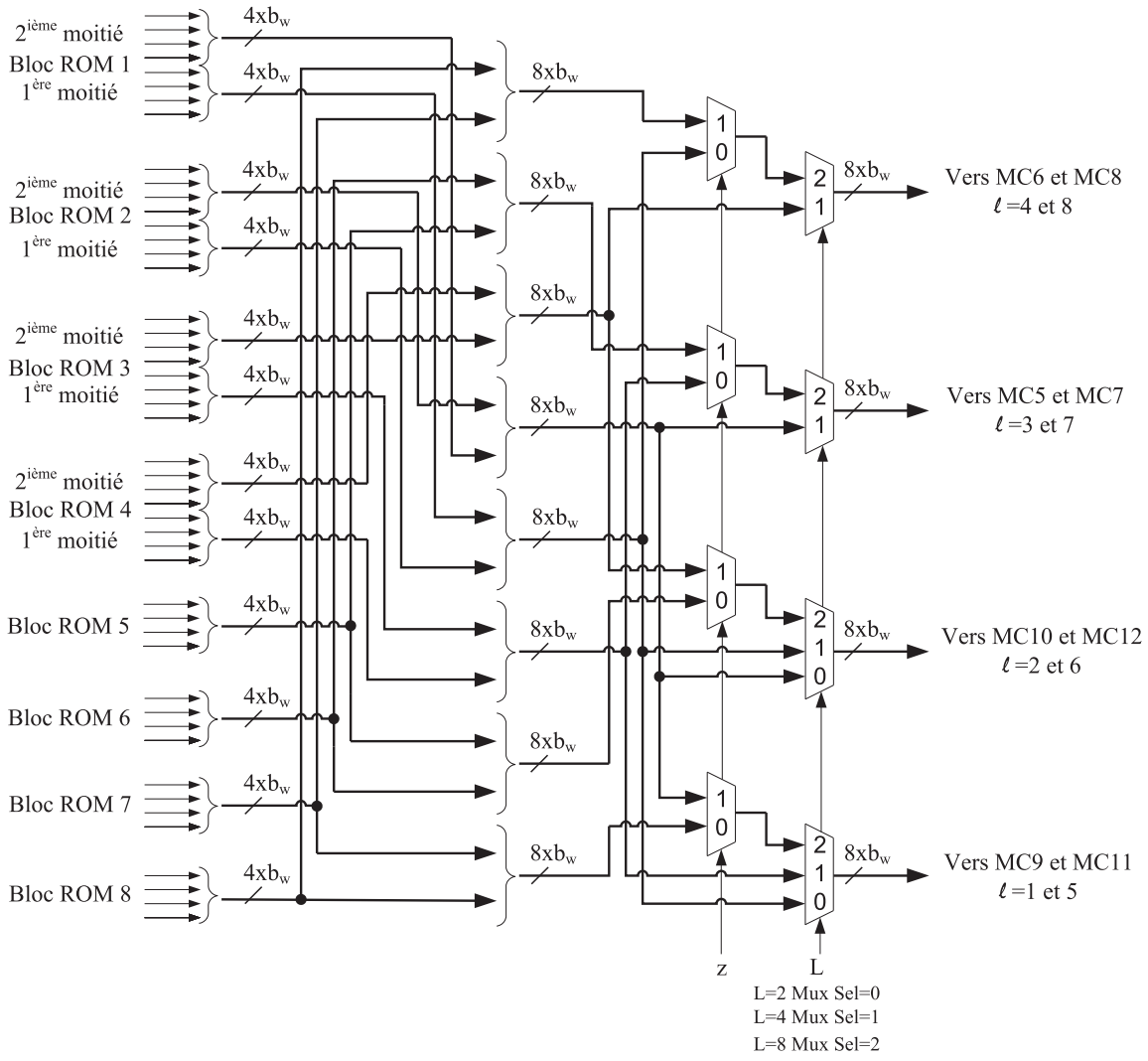

FIGURE 6.27 – Multiplexage de sortie selon L des mémoires de coefficients IOTA

6.4 Mémoires de filtrage de mise en forme des échantillons

À la figure 2.18 de la section 2.4, nous avons présenté l'architecture classique de filtrage composée de FIFO pour la mémorisation des échantillons filtrés. Toutefois, cette architecture des mémoires tient compte de valeurs de L et N fixes.

L'approche classique filtre $2L$ échantillons à chaque cycle. Elle utilise $2L-1$ FIFOs de M données complexes. Une opération de lecture/écriture est réalisée par cycle dans chaque FIFO. Dans notre cas, $P_{D_{TFR}}L$ sont filtrés à chaque cycle, soit $\frac{P_{D_{TFR}}}{2}$ plus d'échantillons. Par conséquent, jusqu'à $\frac{P_{D_{TFR}}}{2}$ opérations de lecture/écriture sont réalisées par cycle dans chaque bloc mémoire de filtrage. Du fait de la valeur maximale de $P_{D_{TFR}}$ de 8, chaque bloc est composé donc de 4 bancs mémoires RAM de taille $\frac{M_{max}}{4}$ échantillons au lieu d'une mémoire FIFO à M point. La figure 6.28 illustre l'analogie de la mémoire de filtrage entre les deux approches. L'architecture est donc composée de $(2L_{max}-1)$ blocs mémoires. Selon la valeur de L, seulement $(2L-1)$ blocs sont activés.

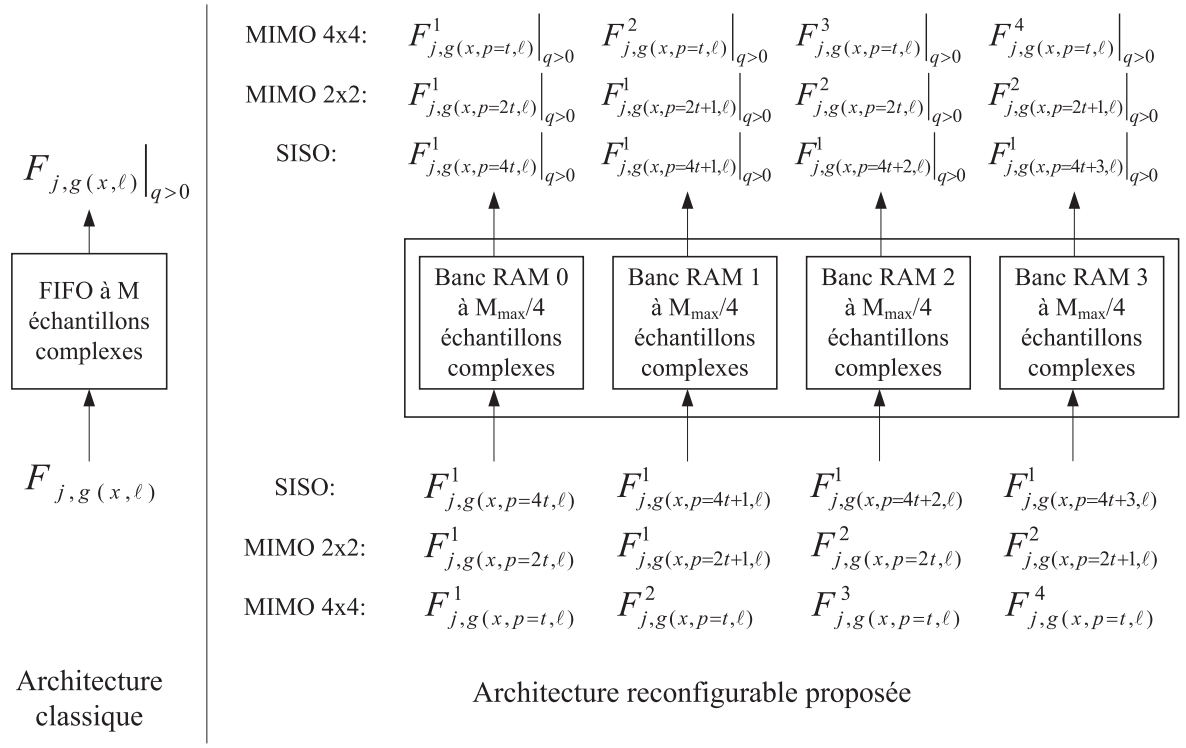

FIGURE 6.28 – Analogie entre l'architecture d'un bloc mémoire de filtrage classique et proposé ($t=0$ si $p \leq \frac{P_{D_{TFR}}}{2}-1$ sinon 1)

Selon le mode MIMO, les quatres bancs mémorisent à chaque cycle soit 4 échantillons de la même TFR ($F^1_{j,g(x,p,\ell)}$ pour $p=0$ à 3, ou 4 à 7), soit 2 échantillons de deux TFR différentes ($F^1_{j,g(x,p,\ell)}$ et $F^2_{j,g(x,p,\ell)}$ pour $p=0$ à 1, ou 2 à 3), soit 1 échantillon de quatre TFR différentes ($F^1_{j,g(x,p,\ell)}$, $F^2_{j,g(x,p,\ell)}$, $F^3_{j,g(x,p,\ell)}$, $F^4_{j,g(x,p,\ell)}$ pour $p=0$ ou 1). Chaque banc RAM est composé de l'architecture de la figure 6.2 afin d'obtenir des tailles allant de 8 à 1024 échantillons.

L' architecture utilise des RAMs *double port* mais avec le fonctionnement d'une FIFO. Ceci est fait grâce à l'adressage des RAMs. Ainsi, la dernière adresse lue correspond à la prochaine

adresse d'écriture. L'écriture est toutefois décallée d'un nombre de cycles égal au nombre de cycle de filtrage d'un échantillon. Les adresses d'écriture et de lecture correspondent à la valeur du compteur papillon b et ne nécessitent donc aucun générateur d'adresse particulier. Outre le fait que la mémoire RAM occupe moins de surface que la FIFO en particulier pour des valeurs de N pouvant atteindre 8192 [109], le cas du L_{max} augmente la complexité de l'architecture à base de FIFO, du fait qu'un traitement en deux temps est nécessaire.

La figure 6.29 illustre les connexions entre les additionneurs des modules de calcul et les mémoires de filtrage pour $L = 2$ et mode SISO. Pour une valeur de x, p avec $p < \frac{P_{D_{TFR}}}{2}$ et ℓ, l'échantillon du résultat du filtrage $F_{j,g(x,p,\ell)}$ est obtenu à partir de l'échantillon aux indices x, $p + \frac{P_{D_{TFR}}}{2}$ et ℓ. Pour $p > \frac{P_{D_{TFR}}}{2}$, le résultat est obtenu à partir de l'échantillon aux indices x, $p - \frac{P_{D_{TFR}}}{2}$ et $\ell + 1$. Il est à noter que lors du filtrage, les échantillons des résultats de la TFR sont lus dans un ordre *bit reversed*, du fait de l'algorithme entrelacement en fréquence.

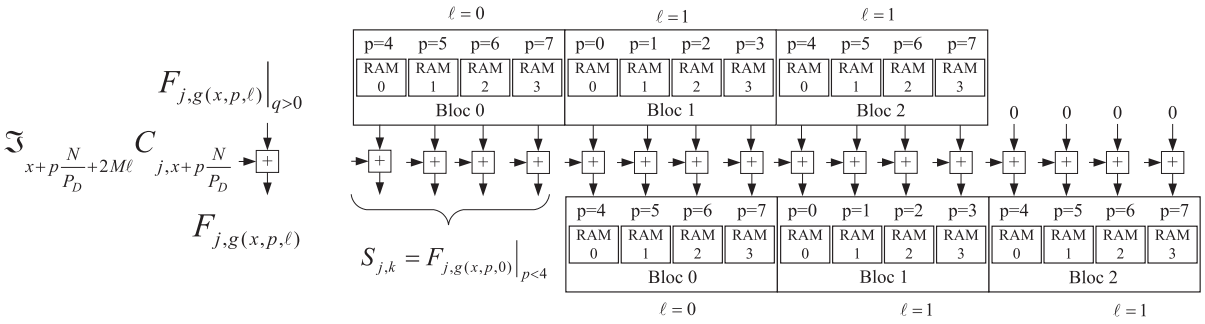

FIGURE 6.29 – Connexion entre les additionneurs des modules de calcul et les mémoires de filtrage pour $L = 2$ et mode SISO ($P_{D_{TFR}} = 8$)

On constate deux particularités pour les blocs mémoires situés aux deux extrémités. En effet, les échantillons $F_{j,g(x,p,\ell)}$ avec $\ell = 0$ et $p < \frac{P_{D_{TFR}}}{2}$ obtenus après addition des échantillons filtrés au symbole précédent $F_{j,g(x,p,\ell)}|_{q>0}$ avec $\ell = 0$ et $p \geq \frac{P_{D_{TFR}}}{2}$ sont envoyés directement à la sortie. Les échantillons $F_{j,g(x,p,\ell)}$ avec $\ell = L - 1$ et $p \geq \frac{P_{D_{TFR}}}{2}$ sont obtenus en additionnant des valeurs nulles.

La figure 6.30 illustre les connexions entre les modules de calculs de la matrice en tenant compte des différentes valeurs de L. De plus, il faut noter que pour L_{max}, les additionneurs des blocs de calcul traitant une valeur de ℓ, reçoivent des échantillons de deux blocs différents. Il en est de même de la mémorisation du résultat de l'addition. Lors du filtrage par la première moitié de la fonction IOTA ($z = 0$), on considère en lecture les blocs 0 à 7 et 0 à 6 en écriture. Pour la deuxième moitié du filtrage ($z = 1$), on considère en lecture les blocs 8 à 14 et 7 à 14 en écriture. À noter qu'en écriture, les deux groupes de blocs mémoires (0 à 6 et 7 à 14) reçoivent tous deux les échantillons, toutefois seulement le groupe concerné est actif en écriture.

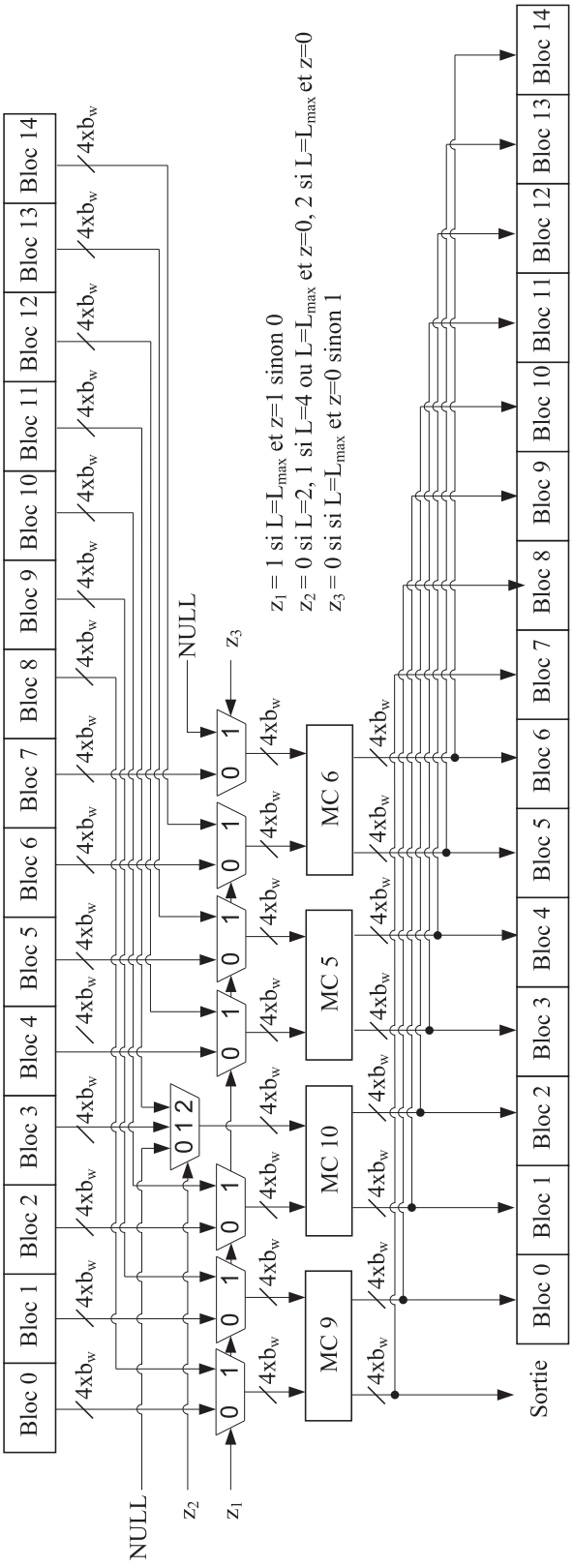

FIGURE 6.30 – Connexion entre les modules de calcul et les mémoires de filtrage pour les différentes valeurs de L (échantillons réels)

Chapitre 7

Contrôle de l'architecture

Le module de contrôle a pour but de générer tous les signaux de contrôle des modules. Ces signaux permettent, d'une part, de configurer le fonctionnement de tous les modules et notamment la matrice de calcul, et d'autre part, de commander la génération d'adresses de lecture/écriture pour toutes les mémoires de l'architecture.

La figure 7.1 illustre l'architecture du module de contrôle du modulateur OFDM avancé. Ce dernier est constitué principalement de deux modules. Le premier module, *séquenceur TFR/Filtre*, permet d'ordonnancer les opérations de la TFR et du filtrage. Le second module contrôle le traitement de la TFR. Les signaux d'entrée hors les paramètres N, L et le mode MIMO sont :

- RST : réinitialise le système.
- $START$: enclenche la modulation d'un symbole OFDM (QAM ou OQAM).
- $TFRD/TFRI$: détermine le sens de la TFR, directe ou indirecte.
- $QAM/OQAM$: détermine le schéma de modulation, 0 si QAM, 1 si OQAM.

En sortie, les signaux sont :

- $Data\_out$: contrôle les buffers de sortie.
- z : détermine le nombre d'étapes de filtrage, $z = 0$ ou 1 si L_{max} sinon $z = 0$.
- TFR_IOTA_mode : vaut 1 si mode TFR ou 0 si mode filtrage IOTA.
- κ : détermine le numéro de l'étage.
- $Dernier\_étage$: détermine s'il s'agit du dernier étage de la TFR.
- $Shift$: Détermine la fin d'un étage pour le contrôle du séquenceur des modules SIB.
- b : détermine le numéro du papillon de l'étage radix-2^i.
- $Radix$: détermine l'algorithme radix-2^i de l'étage en cours de traitement.
- $Flushed$: détermine si la matrice est vide après le traitement d'un étage radix-2^i de la TFR ou après le filtrage.

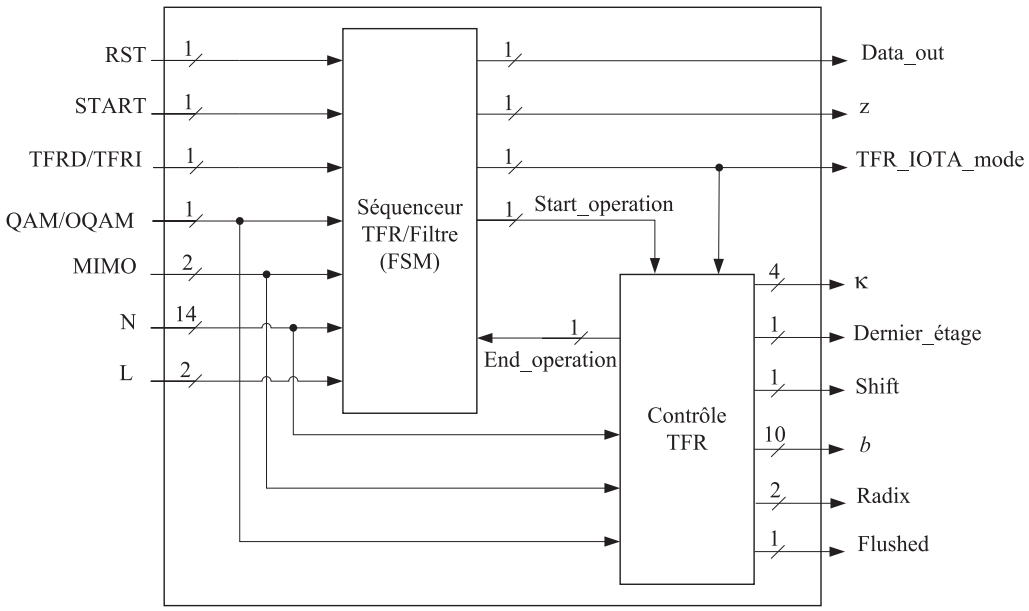

FIGURE 7.1 – Module de contrôle de l'architecture proposée

7.1 Ordonnancement entre les opérations de la TFR et du filtrage

La figure 7.2 illustre la machine à états finis (FSM ou *Finite State Machine*) du module *séquenceur TFR/Filtre*. Au signal $START$, la FSM passe de l'état *Attente* à l'état *Début TFR*. Cet état permet de mettre le signal *Start_operation* à '1' afin de lancer le fonctionnement du module *contrôle TFR*. La FSM passe par la suite à l'état *TFR* afin de réinitialiser le signal *Start_operation* pendant le traitement. Le signal *End_operation* déterminé par le module *contrôle TFR* dicte la fin de la TFR. Selon le signal End_IOTA, la FSM passe soit à l'état *début IOTA* afin de mettre à nouveau le signal *Start_operation* à '1' et ainsi lancer l'opération de filtrage ou à l'état *Data_out*. L'état *IOTA* permet de réinitialiser le signal *Start_operation* durant le traitement du filtrage.

Le signal End_IOTA détermine la fin du filtrage. Ce signal est obtenu à partir du signal $QAM/OQAM$ et d'un compteur sur 2 bits contrôlé par les signaux RST_IOTA et INC_IOTA permettant respectivement la réinitialisation et l'incrémentation du compteur. Si le signal $QAM/OQAM$ vaut '0', alors End_IOTA ='1' en tout temps et aucun filtrage n'est effectué. Sinon End_IOTA='1' si $L = L_{max}$ et la valeur du compteur égal à 2, c'est-à-dire après les deux étapes de filtrages, ou pour une valeur du compteur égale à 1 pour les autres valeurs de L correspondant à une étape de filtrage sinon End_IOTA='0'.

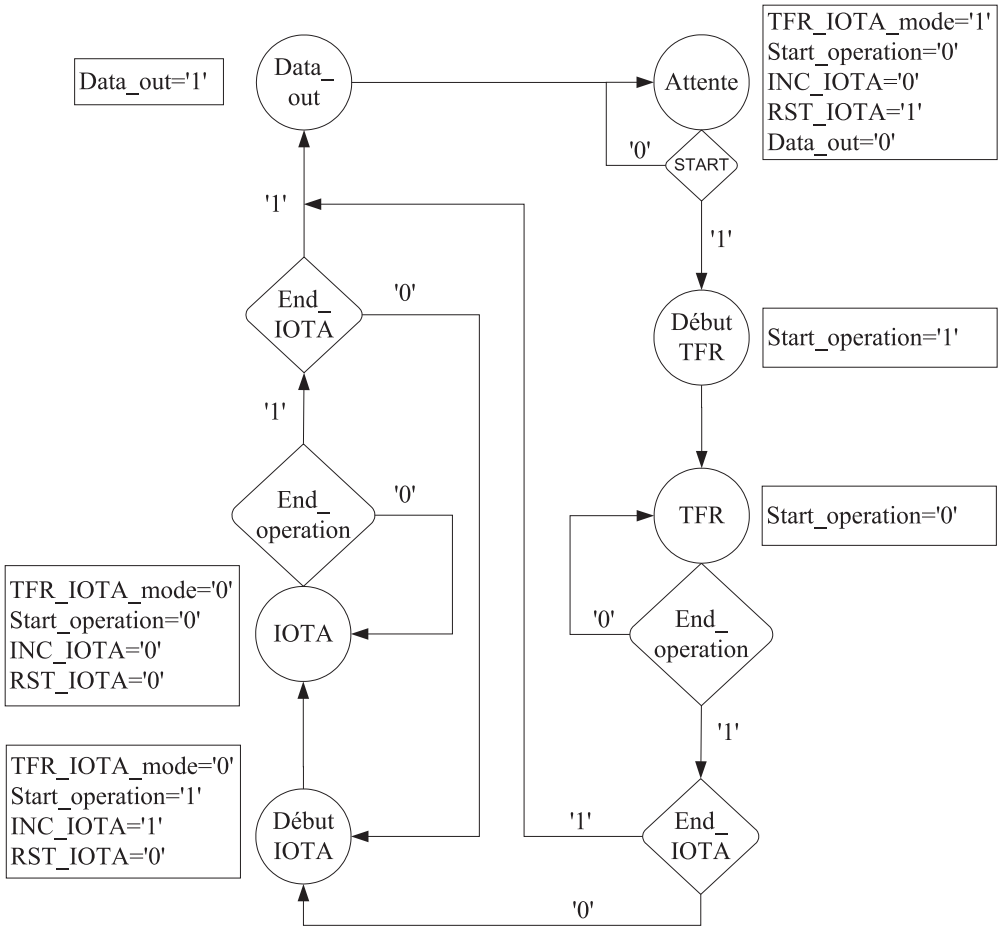

FIGURE 7.2 – Machine à états du séquenceur TFR/Filtre

7.2 Contrôle de la TFR

La figure 7.3 illustre le module de contrôle de la TFR. Ce dernier est constitué de deux FSM et de trois compteurs. La première FSM, *séquenceur TFR*, permet d'ordonner le fonctionnement de la TFR, tandis que la seconde FSM permet de calculer le radix de l'algorithme utilisé.

Le compteur κ_cmpt détermine le numéro de l'étage en cours. À la fin d'un étage radix-2^i, la valeur du compteur qui est sur 4 bits est incrémentée d'une valeur i grâce au signal κ_INC. Le compteur est remis à zéro à la fin de la TFR grâce au signal κ_RST. Le signal $Dernier\_étage$ est obtenu en comparant la valeur du compteur à κ_{max} qui dépend des paramètres N et du mode MIMO.

Le compteur b_cmpt est le compteur de papillon. À chaque cycle sa valeur est incrémentée de 1 grâce au signal b_INC. Le compteur est remis à zéro à la fin d'un étage grâce au signal b_RST. Le signal $Dernier\_papillon$ est mis à '1' si le compteur atteint la valeur maximale, soit $\frac{N}{P_{D_{TFR}}} - 1$.

Le troisième compteur, F_cmpt, permet de déterminer que les derniers échantillons ont été traités par la matrice et que cette dernière est vide. En effet, le calcul d'un nouvel étage ne peut commencer tant que tous les résultats de l'étage courant n'ont pas été mémorisés.

La figure 7.4 illustre la machine à état du séquenceur de la TFD. Au signal $START$, la FSM

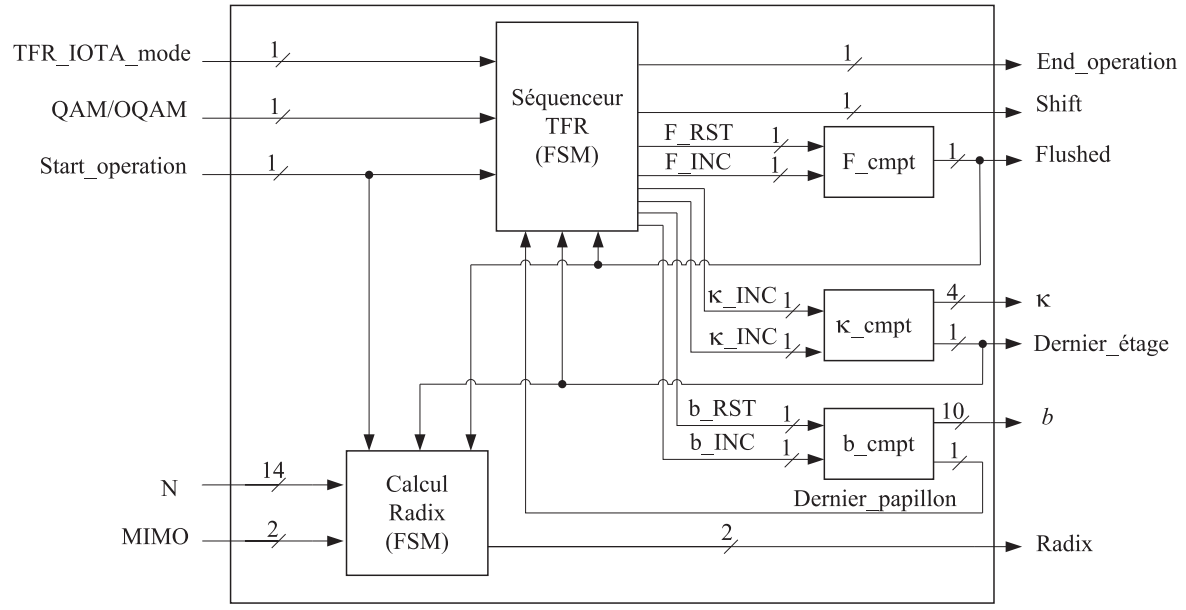

FIGURE 7.3 – Module de contrôle de la TFR

passe de l'état *Attente* à l'état *Running_ TFR*. Le signal *Shift* est mis à '1' afin d'agir comme front montant sur les bascules du séquenceur des modules SIB. S'il s'agit du dernier étage et si la matrice est vide, alors le séquenceur revient en mode attente d'une nouvelle modulation. La figure 7.5 illustre la machine à état du calcul du radix permettant de décomposer la TFR à N points en une succession de sous TFR à $\frac{N}{2^i}$ points.

Finalement, il faut noter que lors du traitement du filtrage de mise en forme, la valeur de κ_{max} est de 1, permettant ainsi de générer la lecture des échantillons des résultats de la TFR dans l'ordre *bit reversed*.

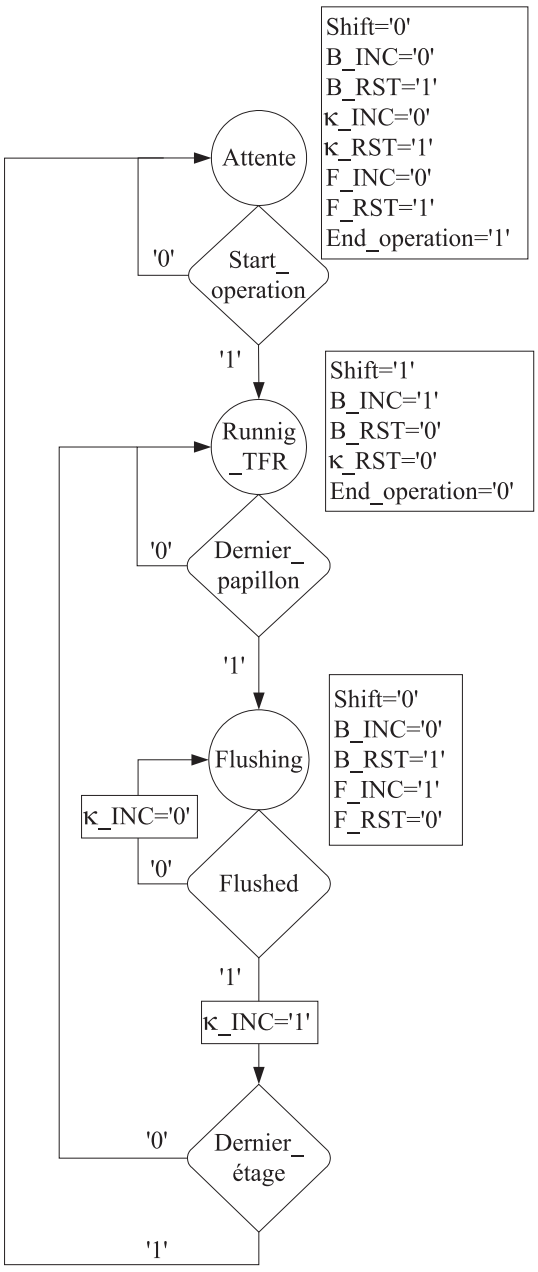

FIGURE 7.4 – Machine à état du séquenceur TFR

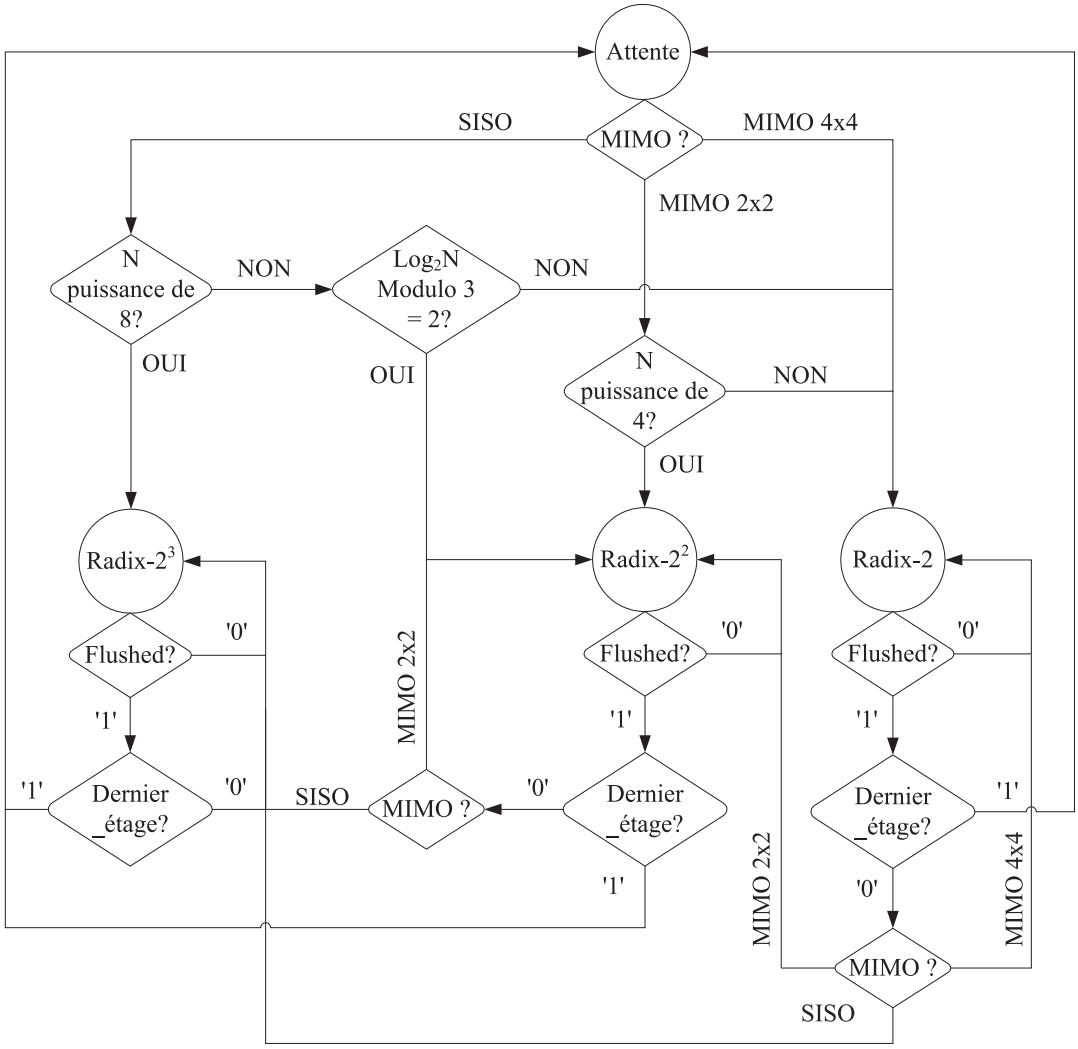

FIGURE 7.5 – Machine à état du calcul du radix

Chapitre 8

Erreur de quantification

La réalisation d'un circuit intégré dédié apporte des limitations dans le calcul arithmétique du fait de la représentation des données en virgule fixe ce qui est nécessaire pour satisfaire les contraintes de coût et de consommation. L'utilisation de l'arithmétique virgule fixe se traduit par la présence de sources de bruits liées à l'élimination de bits lors d'un changement de format. Ces bruits se propagent au sein du système et modifient la précision des calculs en sortie de l'application. La dégradation de la précision des calculs doit être maîtrisée afin de garantir l'intégrité de l'algorithme et les performances de l'application. La précision est évaluée en déterminant l'expression du Rapport Signal à Bruit de Quantification (RSBQ). Cette analyse permet ainsi de limiter les ressources nécessaires, telles que la taille des mémoires des données des échantillons, des coefficients de rotation W et de la fonction IOTA, ainsi que la taille des opérateurs arithmétiques nécessaires tout en garantissant un RSBQ adéquat.

Nous supposons que les calculs se font avec une représentation des données en complément à 2. Ces données sont représentées par $b+1$ bits dont le bit de poids le plus fort désigne le signe. Le pas de quantification, c'est-à-dire la plus petite valeur représentable, est de 2^{-b}. Dans ce chapitre, nous analyserons l'erreur induite par la quantification des données. Dans un premier temps, nous identifions les sources du bruit du modulateur proposé. Par la suite, une approche théorique est présentée afin de déterminer l'expression analytique du RSBQ du modulateur OFDM avancé proposé. Finalement, des résultats de simulation du modèle SystemC du modulateur seront exposés.

8.1 Sources de bruits

On peut identifier plusieurs sources de bruit. En premier lieu, la quantification des données à traiter, avant même toute manipulation arithmétique, présente une erreur de quantification selon le nombre de bits qui leur sont octroyés. Nous pouvons identifier trois sources de bruit initiales :

1. Les erreurs dues aux échantillons à moduler en entrée du circuit.
2. Les erreurs dues aux coefficients W de la TFR.
3. Les erreurs dues aux coefficients de la fonction IOTA lors du filtrage de mise forme dans le cas de l'OFDM/OQAM.

Nous considérons que ces données sont obtenues par arrondi. Soit b_1 et b ($b < b_1$) les nombres de bits à droite de la virgule binaire respectivement avant et après les opérations d'arrondi.

L'erreur maximale a une amplitude de :

$$-\frac{1}{2}\left(2^{-b}-2^{-b_1}\right) \leq e_A \leq \frac{1}{2}\left(2^{-b}-2^{-b_1}\right) \tag{8.1}$$

Sans perte de généralité, nous ne tenons pas compte de ces sources de bruits des échantillons à l'entrée à ce stade et nous considérons que les échantillons quantifiés représentent leur valeur exacte. Nous nous efforcerons d'analyser les erreurs introduites dans le calcul lors de la TFR et du filtrage de mise en forme.

8.1.1 Sources de bruits pour la TFR

L'analyse théorique présentée dans ce chapitre a été réalisée pour le cas de la TFR directe. Toutefois, les résultats s'appliquent aussi pour la version inverse. En effet, les deux versions utilisent la même architecture et les mêmes algorithmes radix-2^i avec un entrelacement en fréquence, la différence étant dans le signe des valeurs des coefficients de rotation W. Les manipulations des données est la même peu importe la version de la TFR, directe ou inverse, et les opérations de quantifications et de troncations se font aux mêmes emplacements.

Dans une TFR de longueur N radix-2^i, nous avons $\log_2 N$ étages de calculs, chaque étage comprenant $N/2$ croisillons. Pour un entrelacement fréquentiel, nous avons les relations suivantes :

$$\begin{aligned} X_{\kappa+1}(\theta) &= (X_\kappa(\theta) + X_\kappa(\phi))\, W_N^{r_1} \\ X_{\kappa+1}(\phi) &= (X_\kappa(\theta) - X_\kappa(\phi))\, W_N^{r_2} \end{aligned} \tag{8.2}$$

Où $X_\kappa(*)$ et $X_{\kappa+1}(*)$ représentent respectivement les entrées et sorties du croisillon du κ^{ime} étage. Ces équations montrent que le module maximal des nombres croît d'un étage à un autre et peut générer un débordement (*overflow*). Il a été démontré que le module maximal des nombres n'augmente que d'un facteur inférieur ou égal à deux à chaque étage de calcul [70,112,113]. Ceci conduit donc à l'encadrement suivant du module sortant d'un étage de calcul :

$$max\left(|X_\kappa(\theta)|, |X_\kappa(\phi)|\right) \leq max\left(|X_{\kappa+1}(\theta)|, |X_{\kappa+1}(\phi)|\right) \leq 2\ max\left(|X_\kappa(\phi)|, |X_\kappa(\phi)|\right) \tag{8.3}$$

Ce débordement peut être évité en incorporant un recadrage par un facteur de $1/2$ dans le croisillon. Ce recadrage consiste seulement en un décalage d'un bit vers la droite des bits de données avec une élimination d'un bit par troncature. Ce bit ayant une valeur de 0 ou 1 avec une probabilité de $1/2$, il en résulte une puissance de bruit égale à $\frac{2^{-2b}}{8}$ sur chaque nombre réel et $\frac{2^{-2b}}{4}$ sur chaque nombre complexe [70,77]. La valeur maximale de cette erreur de troncature est de :

$$-\left(2^{-b}-2^{-b_1}\right) \leq e_T \leq 0 \tag{8.4}$$

Une autre source d'erreur est dûe aux multiplications qui génèrent un nombre de bits de résultat supérieur au nombre de bits des valeurs d'entrée. Ainsi, les résultats d'une multiplication de deux nombres à $b+1$ bits doivent être ramenés de $2b+1$ bits à $b+1$. Dans le cas d'une multiplication réelle, l'erreur de quantification est uniforme sur l'intervalle de $\left[\frac{-2^{-b}}{2}, \frac{-2^{-b}}{2}\right]$ en considérant l'arrondi [70]. Dans ce cas, cette source d'erreur (*roundoff noise*) a une variance de $\frac{2^{-2b}}{12}$. Considérant qu'une multiplication complexe est constituée de quatre multiplications réelles, la variance de l'erreur pour une multiplication complexe est donc de $\frac{2^{-2b}}{3}$.

La figure 8.1 illustre l'emplacement du recadrage et de la quantification d'un croisillon radix-2^i.

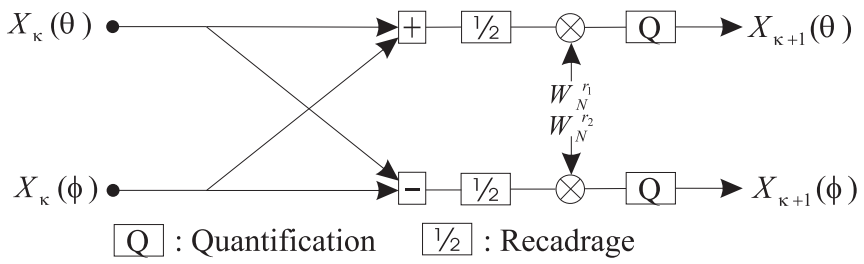

FIGURE 8.1 – Emplacement du recadrage et de la quantification d'un croisillon radix-2^i

8.1.2 Sources de bruits pour le filtrage de mise en forme

Comme nous l'avons expliqué dans les chapitres précédents, le filtrage de mise en forme consiste en L pondérations parallèles de $2M$ données sortant de la TFR inverse par la fonction prototype IOTA (pour un total de $2ML$ pondération), à additionner les résultats aux échantillons des mémoires de filtrage et finalement à émettre M échantillons. Les valeurs des coefficients de la fonction IOTA sont tous égaux ou inférieurs à 1. La valeur maximale des échantillons filtrés est obtenue lorsque toutes les entrées du filtre sont égales à 1 (c'est-à-dire lorsque le résultat de la TFR inverse est un signal continu d'une amplitude de 1). L'échantillon d'indice k à l'instant j à la sortie du filtre, soit $S_{j,k}$ de l'équation 2.6, est alors égal à la sommation des coefficients IOTA d'indice $k + qM$ représentée par l'équation 8.5 :

$$S_{j,k} = |\Im_k| + |\Im_{k+M}| + |\Im_{k+2M}| + \ldots + \left|\Im_{k+(2L-2)M}\right| + \left|\Im_{k+(2L-1)M}\right| \tag{8.5}$$

$S_{j,k}$ est toujours supérieur à 1 mais inférieur à 2 et un débordement peut donc survenir. Un recadrage par 1/2 des données à l'entrée ou à la sortie du filtre serait nécessaire. Toutefois, une alternative est de diviser les coefficients du filtre par un facteur $\sqrt{2}$, tout en gardant l'amplitude maximale du filtre entre 0.5 et 1 (≈ 0.7) [32]. Dans ce cas, le résultat de l'équation 8.5 devient inférieur à 1 et aucun recadrage n'est plus nécessaire.

Les sources de bruit dans le cas du filtrage de mise en forme proviennent donc seulement de la quantification après les pondérations par le fonction IOTA. Du fait que la fonction IOTA est réelle, la pondération consiste en deux multiplications réelles, chacune ayant une variance de l'erreur égale à $\frac{2^{-2b}}{12}$.

La figure 8.2 illustre l'emplacement de la quantification lors du filtrage de mise en forme.

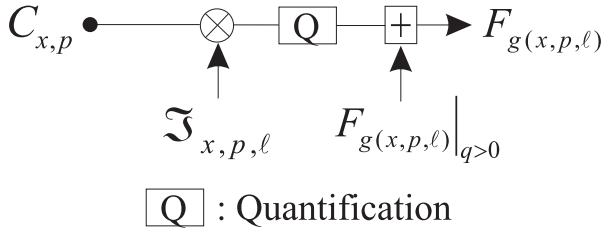

FIGURE 8.2 – Emplacement de la quantification lors du filtrage de mise en forme

8.2 Approche théorique de l'erreur de quantification

L'analyse du bruit de quantification pour un algorithme de TFR radix-2 a été largement présentée dans la littérature scientifique [112–118]. Dans cette section, nous reprenons ces résultats pour les généraliser au cas des algorithmes radix-2^i à des fins de comparaison. Nous considérons les manipulations suivantes des données :

1. Un recadrage avec troncature d'un bit est réalisé après l'addition complexe du croisillon de la TFR avec une variance de l'erreur de $\sigma_t^2 = \frac{2^{-2b}}{4}$.
2. Une quantification est réalisée après chaque multiplication complexe du croisillon de la TFR avec une variance de l'erreur de $\sigma_a^2 = \frac{2^{-2b}}{3}$.
3. Une quantification est réalisée après chaque multiplication réelle lors du filtrage de mise en forme avec une variance de l'erreur de $\sigma_{IOTA}^2 = \frac{2^{-2b}}{6}$ pour les partie réelles et imaginaires des échantillons filtrés.

Afin d'obtenir une expression simple de l'erreur de quantification, nous supposons que :

1. Toutes les entrées de N échantillons complexes sont non corrélées entre elles.
2. Toutes les entrées de N échantillons complexes sont uniformément distribuées dans l'intervalle $\left[\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$ respectant la condition $|x(n)| < 1$ pour tout n.
3. Les erreurs de quantification, multiplication et recadrage, ne sont pas corrélées.
4. Il n'y a pas de corrélation entre les erreurs et les entrées.
5. Les erreurs dues aux quantifications des coefficients W et de la fonction IOTA sont ignorées.
6. Les entrées ne contiennent pas d'erreur.

Dans les sous-sections suivantes, nous ne présentons que les équations de départ et finales du développement des expressions mathématiques du RSBQ des algorithmes de TFR radix-2^i. Le développement complet des différentes expressions sont présentées à l'annexe C.

8.2.1 Approche théorique pour la TFR radix-2 DIF

Puisque toutes les sources de bruit sont considérées non corrélées, la variance du bruit à chacune des N sorties de la TFR du chemin de données de la figure 2.9 est la somme de chaque source de bruit qui se propage à cette sortie. Considérons la variance de l'erreur due à la troncation lors du recadrage. Pour une TFR radix-2 à 2^m points, nous avons $\frac{N}{2^{\kappa+1}}$ erreurs de troncature complexes pour un étage κ, qui se propagent sur $(m-1-\kappa)$ étages pour chaque sortie $X(k)$. Nous obtenons alors :

$$\sigma_T^2 = \sigma_t^2 \left[\frac{N}{2} \left(\frac{1}{4}\right)^{m-1} + \frac{N}{2^2} \left(\frac{1}{4}\right)^{m-2} + \ldots + \frac{N}{2^m} \left(\frac{1}{4}\right)^{m-m} \right] \tag{8.6}$$

Où le facteur $\frac{1}{4}$ reflète le fait que l'erreur de troncation dûe au recadrage de l'étage κ est divisée par deux à chaque étage, et de ce fait, sa variance est divisée par 4. On obtient alors après simplification :

$$\sigma_T^2 = 2\sigma_t^2 \left[1 - \frac{1}{N}\right] = \frac{2^{-2b}}{2} \left[1 - \frac{1}{N}\right] \tag{8.7}$$

La deuxième source d'erreur qui se propage à chaque échantillon de sortie est due à l'erreur d'arrondi après les multiplications complexes dans chaque croisillon. Dans un premier temps, nous considérons que toutes les multiplications complexes par les facteurs de rotation W génèrent un bruit de quantification. Pour une TFR radix-2 à 2^m points, nous avons $\frac{N}{2}$ erreurs d'arrondi complexes par étage κ, et chaque erreur se propage à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$. De plus, chaque erreur subit $(m-1-\kappa)$ recadrages. Le bruit total dû aux multiplications par les facteurs W à chaque sortie est alors :

$$\begin{aligned}\sigma_W^2 &= \sigma_a^2 \frac{N}{2}\frac{1}{N}\left[\frac{N}{2}\left(\frac{1}{4}\right)^{m-1} + \frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \ldots + \frac{N}{2^m}\left(\frac{1}{4}\right)^{m-m}\right] \\ &= \sigma_a^2\left[1-\frac{1}{N}\right] = \frac{2^{-2b}}{3}\left[1-\frac{1}{N}\right] \end{aligned} \tag{8.8}$$

Les multiplications complexes triviales par les facteurs de rotation W égaux à ± 1 et $\pm j$ ne génèrent pas de bruit de quantification. En effet, dans ce cas les multiplieurs sont multiplexés et les données sont simplement envoyées vers la sortie du croisillon (pour $W = \pm 1$) ou les parties réelles et imaginaires permutées ($W = \pm j$). Il y a $2^{\kappa+1}$ multiplications triviales par étage κ pour les $m-1$ premiers étages qui se propagent à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$. Le dernier étage a $\frac{N}{2}$ multiplications triviales par 1. Nous obtenons alors :

$$\begin{aligned}\sigma_{W_{tr}}^2 &= \frac{\sigma_a^2}{N}\left[2\frac{N}{2}\left(\frac{1}{4}\right)^{m-1} + 2^2\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \ldots + 2^{m-1}\frac{N}{2^{m-1}}\left(\frac{1}{4}\right) + \frac{N}{2}\frac{N}{2^m}\right] \\ &= \sigma_a^2\left[\frac{5}{6} - \frac{4}{3N^2}\right] = \frac{2^{-2b}}{3}\left[\frac{5}{6} - \frac{4}{3N^2}\right] \end{aligned} \tag{8.9}$$

La variance de l'erreur de sortie globale pour l'algorithme radix-2 DIF est alors égale à :

$$\sigma_{E_{R2DIF}}^2 = \sigma_T^2 + \sigma_W^2 - \sigma_{W_{tr}}^2 = 2^{-2b}\left[\frac{5}{9} - \frac{5}{6N} + \frac{4}{3N^2}\right] \tag{8.10}$$

Nous nous intéressons au paramètre RSBQ qui correspond à la moyenne quadratique de l'erreur $E\left[|E(k)|^2\right]$ sur la moyenne quadratique du signal de sortie $E\left[|X(k)|^2\right]$. Selon [113], la moyenne quadratique de $X(k)$ est :

$$E\left[|X(k)|^2\right] = \sum_{k=0}^{N-1} E\left[|x(n)|^2\right]\left|W^{kn}\right|^2 = \frac{1}{3N} \tag{8.11}$$

Le RSBQ pour l'algorithme radix-2 DIF est alors :

$$RSBQ_{R2DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b}\left[\frac{5N}{3} - \frac{5}{2} + \frac{4}{3N}\right] \tag{8.12}$$

Ce résultat correspond au résultat obtenu par Oppenheim [113] et Sundaramurthy [116]. Dans le cas des algorithmes radix-2^2 et radix-2^3, les formules du RSBQ présentées dans les sous-sections suivantes sont, pour autant que nous le sachions, inédites.

8.2.2 Approche théorique pour la TFR radix-2^2 DIF

Avec la même démarche, on peut déterminer le RSBQ de l'algorithme radix-2^2. Du fait que le chemin de données est le même peu importe l'algorithme utilisé, le recadrage/troncature des données se fait donc aux mêmes emplacements. L'erreur due au recadrage est donc égale à l'expression 8.7. La différence se situe dans les valeurs des facteurs de rotation et l'emplacement des multiplications complexes. Pour l'algorithme radix-2^2 à 2^m points (m étant un multiple de 2), tous les étages κ pair sont des multiplications triviales par ± 1 ou $\pm j$. En ne considérant que les étages impairs, sachant qu'il y a $\frac{3N}{4}$ multiplications complexes par étage, on obtient :

$$\begin{aligned}\sigma_W^2 &= \sigma_a^2 \frac{3N}{4}\frac{1}{N}\left[\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \frac{N}{2^4}\left(\frac{1}{4}\right)^{m-4} + \frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{N}{2^m}\left(\frac{1}{4}\right)^{m-m}\right] \\ &= \sigma_a^2\left[1-\frac{1}{N}\right] = \frac{2^{-2b}}{3}\left[1-\frac{1}{N}\right] \end{aligned} \tag{8.13}$$

Or, pour chaque étage κ impair, nous avons $4^{\kappa'}$ (avec $\kappa' = \frac{\kappa+1}{2}$) multiplications triviales et $\frac{3N}{4}$ au dernier étage. L'erreur de quantification en moins dues aux multiplications complexes triviales est égale à :

$$\begin{aligned}\sigma_{W_{tr}}^2 &= \frac{\sigma_a^2}{N}\left[4\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + 4^2\frac{N}{2^4}\left(\frac{1}{4}\right)^{m-4} + 4^3\frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{3N}{4}\right] \\ &= \sigma_a^2\left[\frac{49}{60} - \frac{16}{15N^2}\right] = \frac{2^{-2b}}{3}\left[\frac{49}{60} - \frac{16}{15N^2}\right] \end{aligned} \tag{8.14}$$

Par la relation 8.10, on obtient :

$$\sigma_{E_{R2^2DIF}}^2 = 2^{-2b}\left[\frac{101}{180} - \frac{5}{6N} + \frac{16}{45N^2}\right] \tag{8.15}$$

Finalement, l'expression du RSBQ pour l'algorithme radix-2^2 est égale à (avec la moyenne quadratique de $X(k)$ égale à l'expression 8.11) :

$$RSBQ_{R2^2DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b}\left[\frac{101N}{60} - \frac{5}{2} + \frac{16}{15N}\right] \tag{8.16}$$

8.2.3 Approche théorique pour la TFR radix-2^3 DIF

Une décomposition radix-2^3 se réalise en trois étages. Le premier étage n'est composé que de multiplications triviales. Les deuxième et troisième étages possèdent $\frac{N}{4}$ et $\frac{7N}{8}$ multiplications complexes respectivement. Chaque erreur d'arrondi se propage à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$ et traverse $m-2-\kappa$ étages. L'expression de l'erreur d'arrondi due aux multiplications complexes par les facteurs W pour une TFR à 2^m points (m étant un multiple de 3) est égale alors à :

$$\begin{aligned}\sigma_W^2 &= \sigma_a^2\frac{N}{4}\frac{1}{N}\left[\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \frac{N}{2^5}\left(\frac{1}{4}\right)^{m-5} + \ldots + \frac{N}{2^{m-1}}\left(\frac{1}{4}\right)^{m-(m-1)}\right] \\ &\quad + \sigma_a^2\frac{7N}{8}\frac{1}{N}\left[\frac{N}{2^3}\left(\frac{1}{4}\right)^{m-3} + \frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{N}{2^m}\left(\frac{1}{4}\right)^{m-m}\right] \\ &= \frac{8\sigma_a^2}{7}\left[1-\frac{1}{N}\right] = 2^{-2b}\frac{8}{21}\left[1-\frac{1}{N}\right] \end{aligned} \tag{8.17}$$

Il existe $8^{\kappa-1}$ (avec $\kappa' = \frac{\kappa+1}{3}$) multiplications triviales pour κ correspondant au troisième étage de chaque décomposition radix-2^3. Les expressions 8.9 et 8.14 deviennent alors dans le cas du radix-2^3 :

$$\begin{aligned}\sigma^2_{W_{tr}} &= \frac{\sigma_a^2}{N}\left[8\frac{N}{2^3}\left(\frac{1}{4}\right)^{m-3} + 8^2\frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{7N}{8}\right] \\ &= \sigma_a^2\left[\frac{449}{504} - \frac{64}{63N^2}\right] = \frac{2^{-2b}}{3}\left[\frac{449}{504} - \frac{64}{63N^2}\right] \end{aligned} \tag{8.18}$$

En remplaçant les expressions 8.7, 8.17 et 8.18 dans 8.10, on obtient :

$$\sigma^2_{E_{R2^3DIF}} = 2^{-2b}\left[\frac{37086}{63504} - \frac{37}{42N} + \frac{64}{189N^2}\right] \tag{8.19}$$

Et le RSBQ pour l'algorithme radix-2^3 est alors (avec la moyenne quadratique de $X(k)$ égale à l'expression 8.11) :

$$RSBQ_{R2^3DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b}\left[\frac{37086N}{21168} - \frac{37}{14} + \frac{64}{63N}\right] \tag{8.20}$$

La figure 8.3 montre les courbes théoriques du RSBQ pour différentes longueurs des échantillons pour une TFR de 4096 points. La figure 8.4 montre quant à elle la courbe de RSBQ en fonction du nombre de points de la TFR pour des longueurs d'échantillons de 16 bits. Les deux figures comparent le RSBQ pour les trois algorithmes radix-2^i. On remarque une superposition quasi-parfaite entre les trois algorithmes. En effet, l'algorithme radix-2^2 offre des performances en terme de RSBQ inférieur de seulement 0.05 dB par rapport au radix-2. Les performances diminuent de seulement 0.21 dB dans le cas du radix-2^3 par rapport au radix-2. Les plus grandes erreurs de quantifications sont introduites par le recadrage des données. Le chemin de données identiques entre les trois algorithmes explique cette similarité dans le RSBQ. Ainsi, toutes les algorithmes radix-2^i offrent les mêmes performances que l'algorithme radix-2. On remarquera que l'augmentation d'un bit de la longueur des échantillons, améliore le RSBQ d'environ 6 dB. Tandis que pour une valeur de N deux fois plus grande, ce qui correspond à ajouter un étage de calcul, le RSBQ se détériore d'environ 3 dB. Ainsi, pour avoir le même RSBQ, il faut ajouter 1/2 bit au calcul, soit 1 bit pour une valeur quadruple de N. Les expressions du RSBQ obtenus ne tiennent pas compte de tous les paramètres, tels que les erreurs de quantification des échantillons à l'entrée ou encore l'influence des coefficients de rotation W. Toutefois, elles nous donnent une bonne approximation des performances atteignables du modulateur OFDM avancé.

8.2.4 Approche théorique pour le filtrage de mise en forme

Lors du filtrage, nous avons $2 \times 2ML$ multiplications réelles et donc $4ML$ sources de bruit. De plus, chaque échantillon à la sortie du filtre est la somme de $2L$ termes de pondération et donc le bruit de sortie est la somme de $2L$ sources de bruit. La variance de l'erreur introduite lors du filtrage est alors :

$$\sigma^2_{E_{IOTA}} = \frac{2^{-2b}}{6}2L = \frac{2^{-2b}}{3}L \tag{8.21}$$

Ainsi, quand la valeur de L double, le RSBQ croît d'environ 3 dB, soit une détérioration de 6 dB lors d'un passage d'un filtrage avec $L = 2$ à $L = 8$.

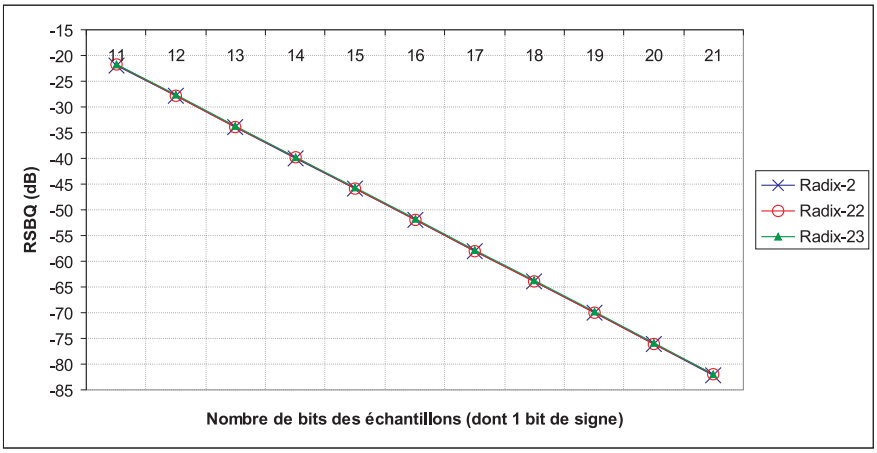

FIGURE 8.3 – RSBQ théorique en dB pour différentes longueurs des échantillons de la TFR à 4096 points

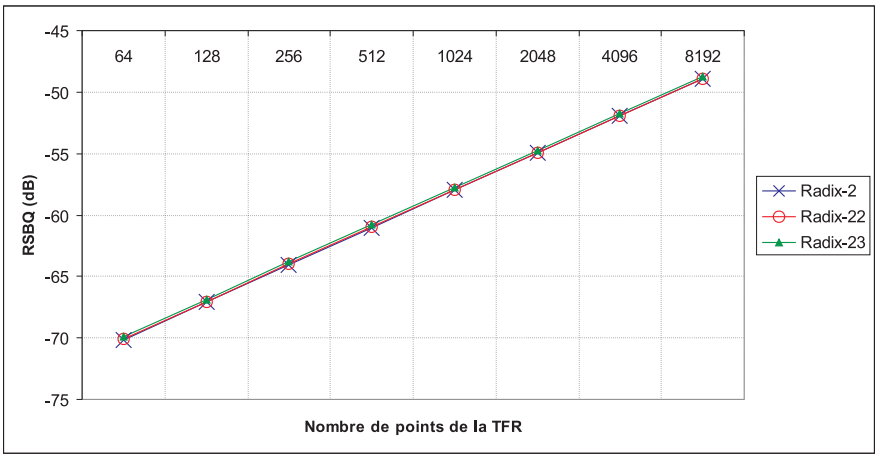

FIGURE 8.4 – RSBQ théorique en dB pour différentes tailles de TFR et une longueur des échantillons de 16 (dont un bit de signe)

8.3 Résultats des simulations pour l'étude de l'erreur de quantification

Nous avons effectué une série de simulations pour analyser l'erreur de quantification de l'architecture OFDM avancée proposée. Pour cela, nous avons programmé en SystemC [119] le modulateur OFDM avancé avec une représentation en virgule fixe complément à deux des données, et nous avons comparé les résultats obtenus avec une représentation des données en virgule flottante double précision avec une dynamique des nombres entre 10^{-308} et 10^{308}.

La fonction de mise en forme IOTA possède une très bonne localisation fréquentielle et temporelle. Afin d'exploiter cette caractéristique, et profiter ainsi de la pureté spectrale de la modulation OFDM/OQAM, particulièrement dans le cadre de la radio cognitive et la radio opportuniste, un RSBQ de -50 dB est nécessaire pour toutes les valeurs de N et de L, soit jusqu'à 8192 points et $L = 8$. Les contraintes sont donc plus élevées que dans le cas de la modulation OFDM/QAM, où un RSBQ de -30 db est suffisant.

Comme on l'a vu précédemment, l'augmentation du nombre de points influence négativement le RSBQ. De plus, le filtrage de mise en forme peut avoir une influence encore plus négative sur l'erreur de quantification surtout pour une grande valeur de L. Une architecture utilisant une représentation en virgule fixe ne permet pas d'obtenir un RSBQ constant pour différentes valeurs de N ou encore différentes valeurs de L. Des longueurs de données variables, permettant d'augmenter le nombre de bits selon le nombre d'étages de la TFR, peut améliorer les performances mais au prix de ressources supplémentaires, notamment la tailles des opérateurs arithmétiques. Afin d'assurer un RSBQ de -50 dB pour toutes les configurations possibles, l'analyse a d'abord été effectuée pour le pire cas, soit une TFR de 8192 points et une longueur de troncature de 8. Les valeurs des échantillons à l'entrée sont inférieures à 1 et codées sur 6 bits. La position de la virgule se situe donc après le bit de signe.

La figure 8.5 illustre l'évolution du RSBQ d'une TFR à 8192 points pour différentes longueurs des échantillons et des coefficients de rotation W. On constate qu'à partir d'une certaine longueur des coefficients W, le RSBQ atteint une valeur minimale et ne varie plus. Cette longueur est de 13, 14, 15, et 16 bits pour des longueurs des échantillons de 18, 19, 20 et 21 bits respectivement. On obtient ainsi un RSBQ maximale de -57 dB pour une longueur des échantillons de 18 bits et une valeur minimale de -75 dB pour une longueur des échantillons de 21 bits. Ainsi, l'augmentation de 1 bit des échantillons de la TFR permet d'obtenir un gain de 6 dB.

La figure 8.6 montre les performances obtenues pour une modulation OFDM/OQAM avec une longueur de troncature de la fonction IOTA de 8. La longueur des échantillons pour le filtrage est égale à celle des échantillons de la TFR. La figure 8.6 illustre le RSBQ pour une longueur des échantillons de 19 et 20 bits seulement. Une longueur des échantillons de 21 bits engendrerait une capacité mémoire plus importante. En effet, il est plus intéressant de limiter la longueur des échantillons que la longueur des coefficients afin d'atteindre le RSBQ voulu. On constate sur la figure 8.6 que le RSBQ ne varie plus à partir d'une longueur des coefficients IOTA supérieure à 10 bits pour une longueur des échantillons sur 19 bits. Pour une taille des échantillons de 20 bits, le RSBQ atteint une valeur minimale pour une longueur des coefficients IOTA de 11 bits dans le cas d'une taille des coefficients W de 14 bits, et 13 bits dans le cas d'une taille des coefficients W de 15 bits. On constate que les -50 dB sont atteint seulement pour des longueurs des échantillons sur 20 bits, une longueur des coefficients W de 15 bits, et 13 bits au moins pour les coefficients IOTA. Le RSBQ passe de -69 dB pour une TFR à 8192 points, à -50 dB en appliquant le filtrage de mise en forme avec $L = 8$. Les pertes de performances dues au filtrage avec cette longueur de troncature est donc de 19 dB. En obtenant un RSBQ de -50 dB pour le pire cas, on s'assure ainsi d'atteindre les performances souhaitées pour des valeurs de N inférieures à 8192 porteuses, et une valeur de L inférieure à 8.

La figure 8.7 illustre le RSBQ pour différentes tailles de TFR, pour l'OFDM/QAM et différentes longueurs de troncature dans le cas de l'OFDM/OQAM. On constate que la contrainte de -50 dB est toujours atteinte pour tous les cas de figures. L'application du filtrage avec $L = 2$ détériore le RSBQ de 10 dB pour des tailles de TFR allant jusqu'à 1024 points, et de 11 dB pour des TFR plus grandes. Pour des valeurs de N inférieures ou égales à 256, les performances diminuent de 3 dB à chaque fois que la valeur de L double. Pour des valeurs de N plus élevées que 256, cette diminution est de 4 dB. Ces résultats concordent avec la théorie présentée à la section 8.2.4. Ainsi, le filtrage IOTA dégrade le RSBQ de 10 à 11 dB pour $L = 2$ selon les tailles

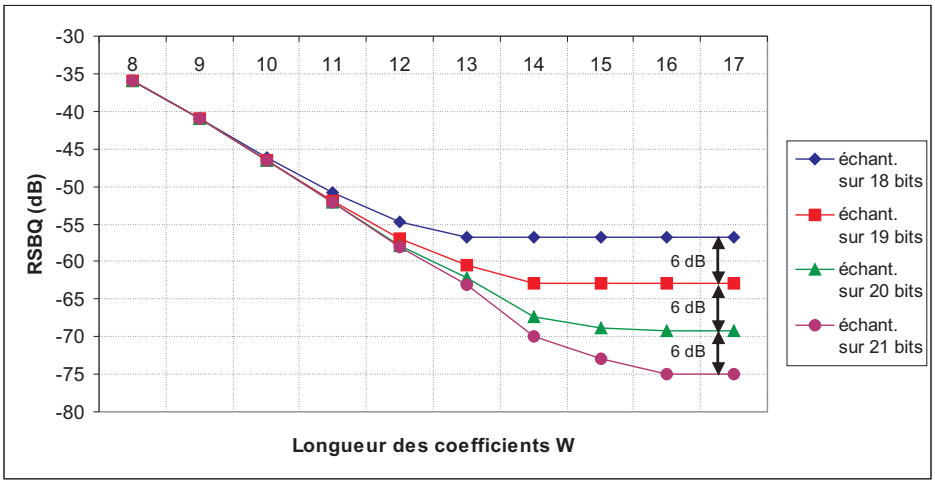

FIGURE 8.5 – RSBQ pour une TFR de 8192 points différentes longueurs des coefficients W et des échantillons (dont un bit de signe)

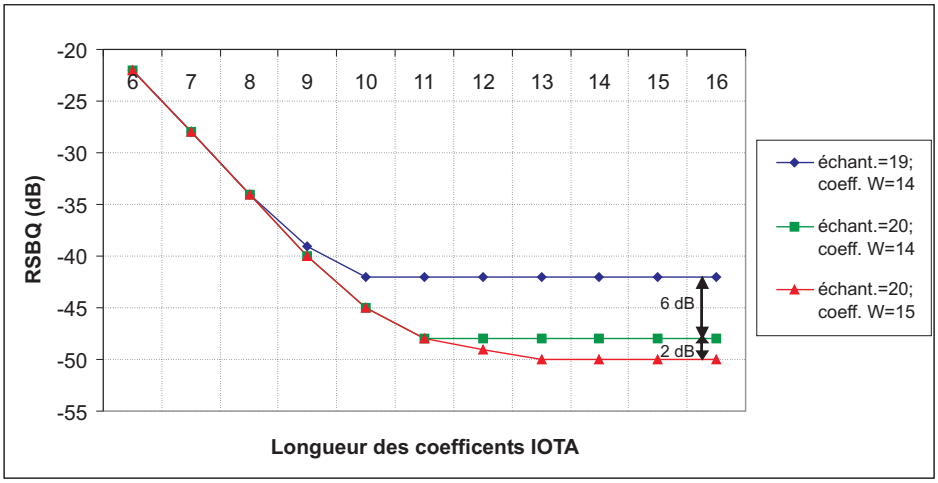

FIGURE 8.6 – RSBQ pour l'OFDM/OQAM pour $N = 8192$ et $L = 8$ en fonction des échantillons TFR, des coefficients W et des coefficients IOTA (longueur avec un bit de signe)

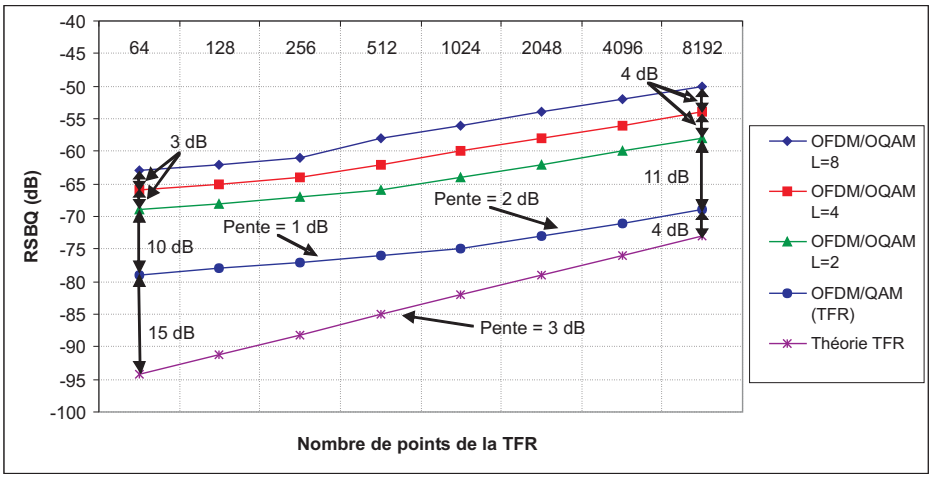

FIGURE 8.7 – RSBQ en fonction de N, avec une longueur des échantillons de 20 bits, les coefficients W sur 15 bits et les coefficients IOTA sur 13 bits (longueur avec un bit de signe)

de la TFR. Pour $L = 4$, la dégradation est de 13 à 15 dB, et finalement 16 à 19 dB pour $L = 8$.

On constate aussi sur la figure 8.7 une différence entre la courbe théorique et simulée pour la TFR. Cette différence est de 15 dB pour $N = 64$ et 4 dB pour une taille de TFR de 8192 points. De plus, la variation de RSBQ en fonction de N est de 3 dB pour les résultats théoriques, et de 1 ou 2 dB, selon la valeur de N, pour les résultats simulés. Cette différence s'explique par le fait que l'expression théorique du RSBQ ne tient pas compte des facteurs de rotations W.

La figure 8.8 illustre le RSBQ pour différentes tailles de TFR en fonction de la longueur des coefficients W. La longueur des échantillons de la TFR est fixée à 20 bits. On remarque que le RSBQ se stabilise à partir de différentes valeurs des longueurs des coefficients W, selon la valeur de N. Notre approche a consisté à réaliser l'analyse de l'erreur de quantification pour le pire cas, soit 8192 points. Pour ce nombre de porteuses, le RSBQ se stabilise à partir d'une longueur des coefficients W de 15 bits. Or, pour cette valeur, le RSBQ n'est pas encore constant pour des tailles de TFR plus petites comme on le constate sur la figure 8.8. Pour ce cas de figure, la différence du RSBQ à chaque multiplication par 2 du nombre de porteuses (ce qui correspond à un étage supplémentaire de traitement), est de 1 à 2 dB. Ceci correspond aux résultats expérimentaux de la figure 8.7. Pour des longueurs des coefficients W égales ou supérieures à 18 bits, le RSBQ atteint une valeur constante pour toutes les valeurs de N. Dans ce cas, la différence du RSBQ à chaque multiplication par 2 du nombre de porteuses est de 3 dB, ce qui rejoint les résultats théoriques. Ainsi, la dégradation du RSBQ à chaque ajout d'un étage supplémentaire de traitement avec la longueur des coefficients W pour le pire cas , nous permet d'avoir une dégradation du RSBQ moins importante.

La figure 8.9 illustre la courbe de RSBQ théorique et simulée en fonction du nombre de points de la TFR. La longueur des données est fixée à 20 pour les échantillons et 18 pour les coefficients W. On constate que la variation du RSBQ en fonction de N est de 3 dB pour les deux courbes.

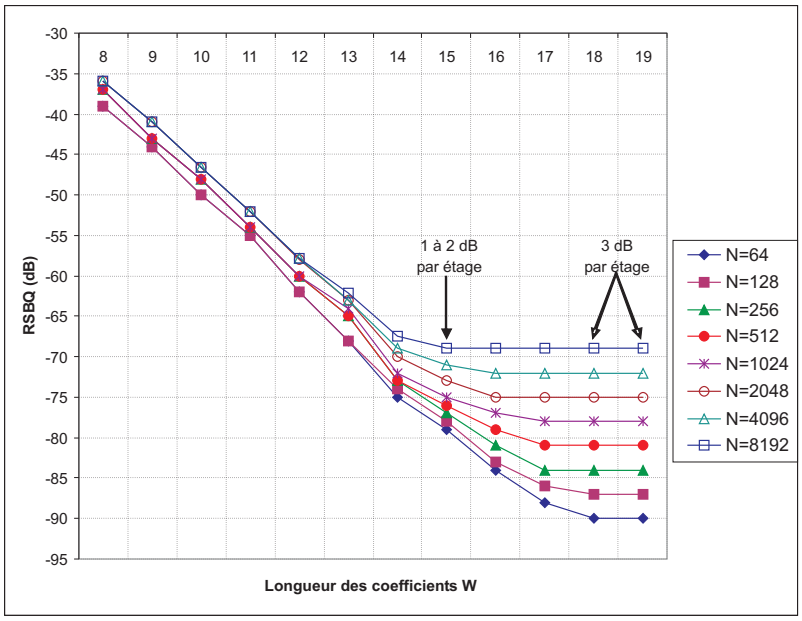

FIGURE 8.8 – RSBQ pour une TFR en fonction des coefficients W et de N pour une longueur des échantillons de 20 (dont un bit de signe)

Il existe un écart de 4 dB entre les résultats théorique et simulées. Cet écart s'explique entre autres par le fait que les données en entrée présentent déja une erreur de quantification. De plus, une certaine corrélation existe en réalité entre les erreurs de troncation [120]. Cette corrélation a été négligée dans les hypothèses de départ.

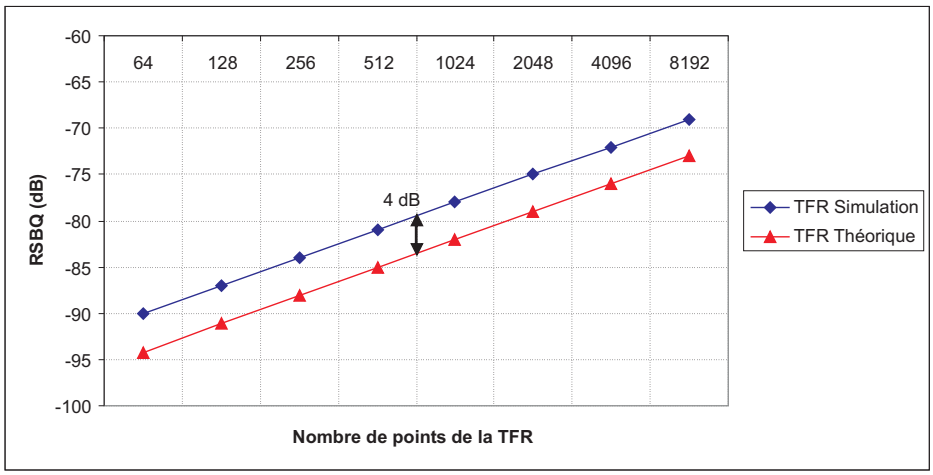

FIGURE 8.9 – Comparaison du RSBQ simulé et théorique pour une TFR en fonction de N, une longueur des échantillons de 20 bits et 18 bits pour les coefficients W (dont un bit de signe)

8.4 Discussion

Nous avons étudié, dans ce chapitre, la quantification des données et les erreurs induites. Pour profiter de la pureté spectrale de la fonction IOTA, la modulation OFDM/OQAM nécessite des performances en termes de RSBQ assez élevé, de l'ordre de -50 dB, comparé à l'OFDM dit classique. Afin d'atteindre cet objectif, nous avons constaté qu'une représentation des données en complément à deux de 20 bits pour les échantillons (à la fois pour la TFR et le filtrage IOTA), 15 bits pour les coefficients de rotation W et 13 bits pour les coefficients de la fonction IOTA, permettent d'atteindre le RSBQ de -50 dB pour un nombre de porteuses pouvant aller jusqu'à 8192 et une longueur de troncature pour le filtrage allant jusqu'à $L = 8$. Les mémoires nécessaires pour le modulateur OFDM avancé sont présentées au tableau 8.1. Comme on le constate, la capacité mémoire des échantillons de la TFR et du filtrage est plus importante que la capacité mémoire des coefficients de rotation W et de la fonction IOTA.

TABLE 8.1 – Tailles mémoires des données d'après l'étude de quantification

Taille mémoire pour la TFR		Taille mémoire pour le filtrage	
RAM échantillons	ROM coefficients	RAM échantillons	ROM coefficients
$2 \times 2N_{max}b_{echant}$ = 640 Kbits	$2\left(\frac{9N_{max}}{16}\right)b_{coef_W}$ = 135 Kbits	$2\left(2L_{max}-1\right)M_{max}b_{echant}$ = 2.4 Mbits	$L_{max}M_{max}b_{IOTA}$ = 416 Kbits

Une autre méthode permettant d'améliorer le RSBQ sans avoir à augmenter la longueur de données internes est d'utiliser une mise à l'échelle des données. Un exemple de réalisation de cette approche est la technique dite *block floating point* (BFP) ou facteur de mise à l'échelle [121, 122].

Il s'agit d'une approche à mi-chemin de la virgule fixe et la virgule flottante. En effet, elle combine l'avantage d'utiliser l'arithmétique simple de la virgule fixe, et une dynamique des nombres proche de celle de la virgule flottante.

La représentation en BFP est constituée par un exposant et une mantisse comme dans le cas de la virgule flottante. En règle générale, l'exposant est partagé par la partie réelle et imaginaire d'une donnée complexe ou par un ensemble de données complexes. À chaque étage, si une donnée devient trop grande pour être représentée par les bits de la mantisse disponible, une mise à l'échelle vers le bas de toutes les données est effectuée. Si la plus grande valeur des données est petite, une mise à l'échelle vers le haut de toutes les données est réalisée afin d'utiliser toute la largeur disponible de la mantisse. Dans les deux cas, l'exposant est utilisé afin de compter le nombre de fois que les données ont été décalées.

Il serait intéressant d'étudier les améliorations possibles par une technique BFP. Ce genre d'approche augmente la complexité de l'architecture. Elle demande une vérification de chaque donnée à chaque étage, des registres à décalages supplémentaires, et une mémorisation des décalages effectués sur les données. Une étude de compromis devrait être réalisée, afin de déterminer la diminution des largeurs des données obtenue (et donc de la taille des mémoires), et la complexité introduite en terme de contrôle et ressources supplémentaires.

Conclusion

Dans cette partie, nous avons présenté une architecture optimisée pour la modulation OFDM avancée ainsi qu'une étude de la quantification des données. L'architecture proposée est complètement reconfigurable selon les paramètres N, L et le mode MIMO, permettant de n'utiliser que les ressources nécessaires. L'architecture est très régulière facilitant ainsi son contrôle.

Cette architecture à gros grain possède une forte capacité de calcul qui nous permet de répondre au très haut débit demandé pour les applications, surtout si on tient compte du filtrage de mise en forme IOTA qui impose une fréquence d'opération deux fois supérieure à une modulation OFDM/QAM dite classique. Sa modularité et son niveau de flexibilité nous donne la possibilité de réaliser plusieurs algorithmes performants radix-2^i ainsi que le filtrage par la fonction prototype avec une même architecture.

Des solutions originales pour différents problèmes ont été proposées. Les caractéristiques de l'architecture sont :

1. Un multiplexage temporel des opérations de traitement de la TFR et du filtrage.
2. Une matrice de calcul reconfigurable.
3. Un degré de parallélisme variable selon les paramètres d'entrée.
4. Une stratégie de mémorisation des échantillons selon le paramètre N pour les échantillons de la TFR et les paramètres N et L pour les échantillons de filtrage.
5. Un schéma de mémorisation des coefficients de rotation W selon l'algorithme utilisé.
6. Un schéma de mémorisation des coefficients de la fonction IOTA selon la parité des indices et une gestion des mémoires selon les paramètres N et L.

D'autres fonctions de mise en forme peuvent être utilisées. Il suffit de remplacer les valeurs des coefficients de mise en forme mémorisées. Toutefois, la fonction utilisée doit être complètement symétrique.

Troisième partie

Implémentation et expérimentation

Chapitre 9

Prototypage FPGA

Ce chapitre présente les travaux de prototypage sur plateforme FPGA ainsi que les résultats obtenus. L'architecture proposée a d'abord été codée en langage VHDL et simulée afin de vérifier le bon fonctionnement du modulateur à différentes étapes du prototypage et ainsi valider l'architecture proposée.

La plateforme de prototypage utilisée est la ML402 de la société Xilinx [123] intégrant le FPGA XC4VSX35 [124] de la famille Virtex-4. Ce dernier est composé de 192 blocs arithmétiques appelés DSP48 facilitant l'intégration d'applications de traitement du signal. Pour une bonne conception sur FPGA, il est nécessaire de bien connaître les caractéristiques du FPGA ciblé. Une partie du protypage FPGA a donc consisté à intégrer les blocs DSP48 dans l'architecture de la matrice de calcul. Dans la prochaine section, nous présenterons l'architecture et le fonctionnement des DSP48. Bien que très performants, ces blocs manquent de flexibilité. Des modifications à l'architecture de la matrice ont été apportées afin de pouvoir utiliser efficacement ces blocs.

Dans la seconde section, nous verrons les contraintes en mémoires rencontrées lors du prototypage. Comme nous l'avons présenté dans la partie précédente, l'architecture proposée utilise une granularité assez fine des blocs mémoires afin de réaliser des TFR allant de 64 points à 8192 points. La stratégie mémoire a été elle aussi légèrement modifiée afin de permettre le prototypage sur le FPGA choisi. Finalement, la dernière section présentera les performances obtenus.

Il est à noter que le prototypage FPGA a été réalisé avant l'étude de quantification des données. De ce fait, toutes les longueurs des données ont été arbitrairement fixées à 16 bits, que ce soit les échantillons ou les coefficients de la TFR ou ceux du filtrage. Pour ces longueurs des données, le RSBQ obtenu en simulation est de -25 dB pour le pire cas, soit une modulation OFDM/OQAM avec 8192 porteuses et une longueur de troncature de 8. Pour le cas le moins contraint, c'est-à-dire une modulation OFDM/QAM avec 64 porteuses, le RSBQ obtenu en simulation est de -66 dB.

9.1 Intégration des blocs DSP48 dans la matrice de calcul

Les FPGAs possèdent une architecture à grain fin permettant une optimisation au bit près. Ce type d'architecture permet une flexibilité totale au niveau des reconfigurations des chemins de données mais est par contre défavorable aux traitements arithmétiques. En intégrant des blocs DSP cablés, les FPGAs acquièrent une capacité de reconfiguration au niveau opérateur arithmétique plus efficace. Pour des applications de traitement du signal caractérisées par une grande capacité de calcul, ce compromis entre grain fin et grain épais offre de meilleures performances.

La figure 9.1 illustre l'architecture d'un bloc DSP48. Il est composé d'un multiplieur complément à deux entrées de 18 x 18 bits, suivi par un additionneur/soustracteur/ accumulateur signé à trois entrées de 48 bits. Il est possible de cascader directement un résultat d'une cellule DSP48 à l'autre afin de réaliser des opérations plus complexes sans l'utilisation d'interconnexion générique du FPGA. Cette caractéristique permet d'optimiser les connexions entre les différents opérateurs arithmétiques ce qui permet aux blocs DSP d'être cadencés à une fréquence maximale de 500 MHz. La consommation de puissance d'un DSP48 est de 2.3 mW/100 MHz.

Les registres présents dans le DSP48 permettent la synchronisation des données en provenance des différentes entrées et sont utiles dans un scénario pipeline. L'utilisation ou non d'un registre se configure statiquement grâce à des paramètres génériques de l'entité VHDL correspondant aux modules DSP48, tandis que l'activation ou non du registre se fait par un signal d'entrée *clock enable* propre à chaque registre, ce qui permet une programmation basse consommation.

La reconfigurabilité du bloc DSP48 est assurée par différents multiplexages internes ce qui permet d'effectuer de nombreux calculs fortement utilisés dans le domaine du traitement du signal. Chaque bloc DSP48 peut donc réaliser une opération de multiplication, multiplication/addition, multiplication/accumulation ou simplement une addition à deux ou trois entrées. Toutefois, il n'est pas possible de réaliser une opération d'addition suivie d'une multiplication par le même bloc DSP. Ce manque de flexibilité dans l'ordre d'utilisation des opérateurs arithmétiques d'un bloc DSP48 modifie légèrement l'architecture des blocs de calculs à l'annexe D.

La figure 9.2 illustre un schéma simplifié (les multiplexages selon les types de bloc A à G du tableau 5.4 ont été omis) d'un bloc de calcul (BC1) intégrant des blocs DSP48 et réalisant l'opération $(X_1 + X_2)W$. Les DSPs 1 et 2 réalisent l'addition complexe $X_1 + X_2$, tandis que les DSPs 3 à 6 réalisent la multiplication complexe par le coefficient W. Les additionneurs des DSP 3 et 4 ne peuvent être utilisés pour réaliser l'addition complexe du fait que les multiplieurs de ces mêmes DSPs sont utilisés au cycle suivant pour réaliser une partie de la multiplication complexe. Sans cette contrainte de l'ordre d'utilisation des opérateurs d'un même bloc DSP, un bloc de calcul aurait nécessité 4 DSP48 au lieu de 6, soit les DSPs 3 à 6. Ces derniers sont utilisés pour réaliser le filtrage de mise en forme comme illustré à la figure 9.3. Des schémas détaillés de

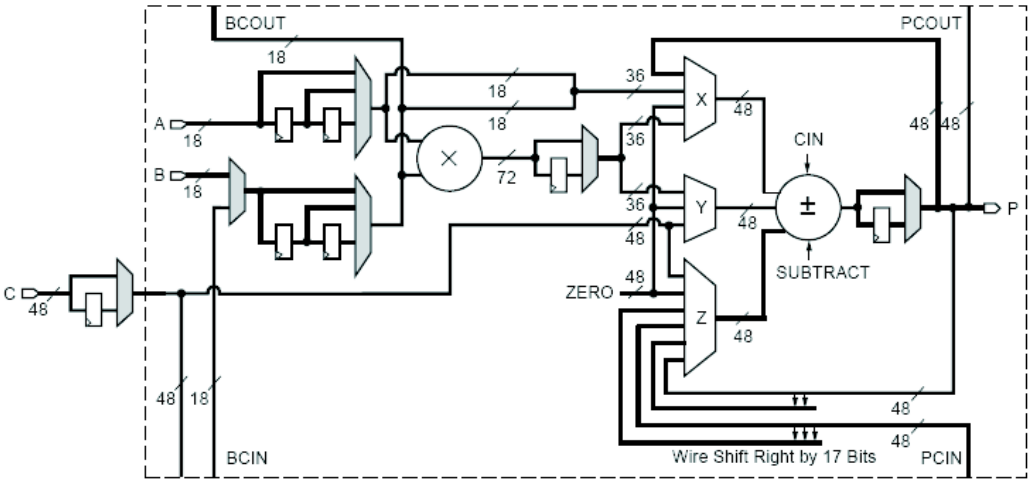

FIGURE 9.1 – Architecture d'un bloc DSP48 [36]

chaque module de calcul de type A à G intégrant les blocs DSP48 tels qu'intégrés dans le FPGA sont présentés à l'annexe E.

Une synthèse logique de la matrice seule a été réalisée afin d'obtenir une idée des performances de la matrice de calcul composée des blocs DSP48. La conclusion de cette synthèse est que la matrice peut fonctionner avec des fréquences qui dépassent les 450 MHz pour la famille Virtex-4, utilise 33% des ressources logiques et 113 blocs DSP des 192 disponibles. Ces essais de synthèse ont également révélé un point important sur l'architecture de la matrice. Il s'agit de la grande connectivité dans ce type de design. À titre d'exemple, pour des données codées à 16 bits, on a plus de 4000 connexions pour relier la matrice aux différentes mémoires et modules annexes.

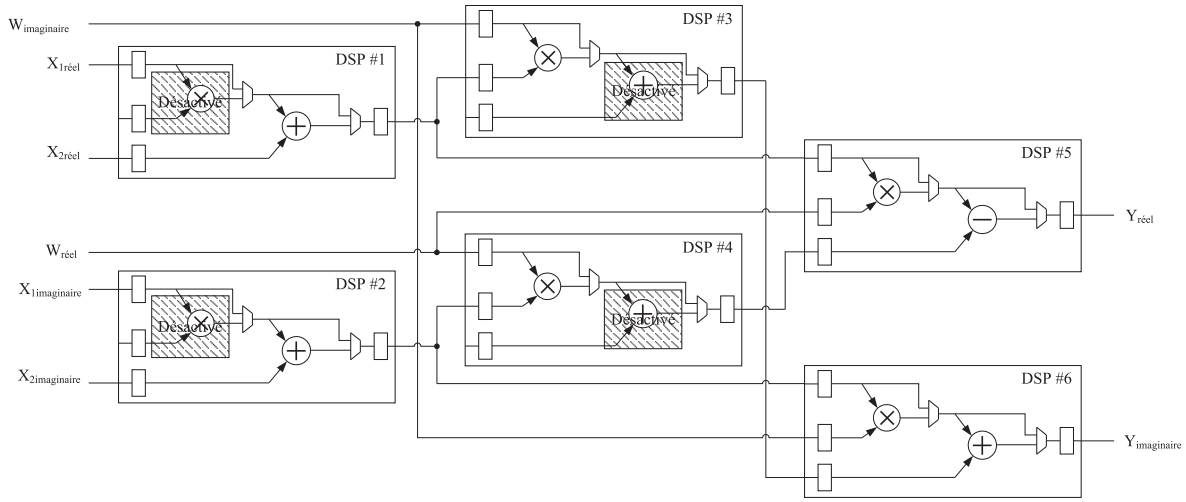

FIGURE 9.2 – Architecture d'un bloc de calcul (BC1) intégrant des DSP48 en mode TFR

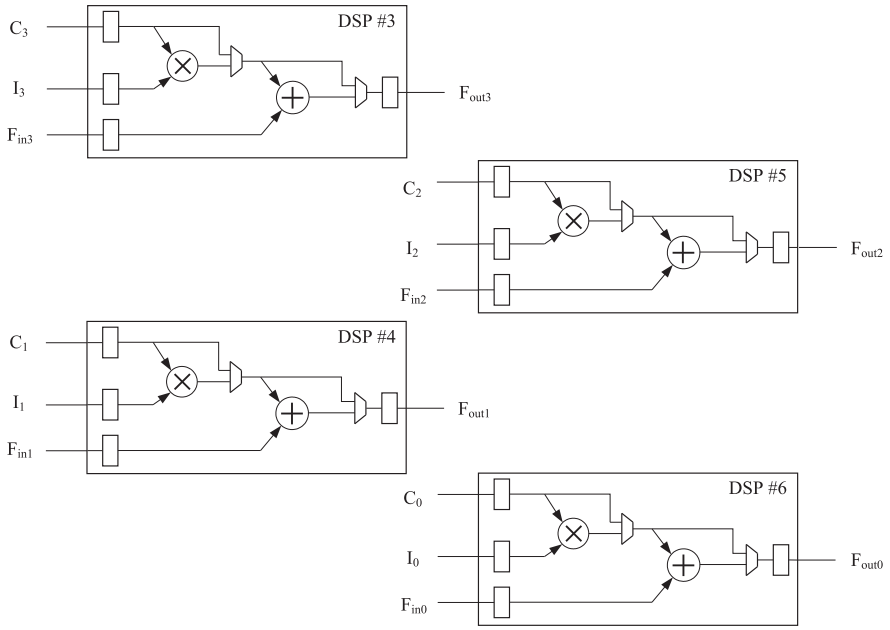

FIGURE 9.3 – Architecture d'un bloc de calcul (BC1) intégrant des DSP48 en mode filtrage

9.2 Contraintes mémoires pour une intégration FPGA

Le FPGA XC4VSX35 de Xilinx possède 192 blocs RAM de 18 Kbits (18432 bits) pour un total de 3456 Kbits de mémoire RAM. De plus, jusqu'à 240 Kbits supplémentaires (mémoire distribuée) peuvent être obtenus grâce à la logique combinatoire disponible dans le FPGA. Toutefois, cette logique combinatoire ne peut être utilisée en totalité comme mémoire RAM du fait qu'elle est utilisée par les différents composants du modulateur OFDM avancé. Comme nous l'avons vu dans la section précédente, la matrice de calcul à elle seule utilise le tiers des ressources combinatoires.

Chaque bloc RAM de 18 Kbits possède deux ports symétriques, interchangeables et complètement indépendants. Ceci donne la possibilité d'utiliser le bloc RAM soit comme deux RAMs *single port*, une RAM *double port*, une ROM ou encore une FIFO. Chaque port peut être utilisé dans une configuration de taille mémoire allant de 16Kx1 à 512x36 en respectant la capacité totale du bloc RAM de 18 Kbits. Ainsi, il est possible d'utiliser par exemple un bloc RAM comme une RAM *double port* de 512 mots de 16 bits pour un total de 8 Kbits. Toutefois la mémoire restante de 10 Kbits du bloc RAM ne peux être réutilisée à d'autre fin.

L'architecture proposée utilise des blocs mémoires RAM *double port*. Les tableaux 9.1 et 9.2 illustrent respectivement le nombre de blocs et la capacité mémoire nécessaire pour la réalisation de l'architecture (avec des données sur 16 bits), soit 1742 blocs RAM *double port* de taille pouvant varier de 8 à 512 données pour un total de 3600 Kbits de capacité mémoire. Or, seulement 192 blocs RAM *double port* sont disponibles dans le FPGA.

TABLE 9.1 – Nombre de blocs mémoires de l'architecture proposée

Nombre de blocs mémoires				
TFR		Filtrage		Buffer de sortie
RAM échant.	ROM coeff.	RAM échant.	ROM coeff.	RAM échant.
$2 \times 2 \times 8 \times 8$ $= 256$	2×7 $= 14$	$2 \times (2L_{max} - 1) \times 4 \times 8$ $= 920$	$L_{max} \times 4 \times 8$ $= 256$	$2 \times 2 \times 8 \times 8$ $= 256$
Total : 1742 blocs RAMs				

TABLE 9.2 – Capacité mémoire de l'architecture proposée avec une longueur des données b_W de 16 bits

TFR		Filtrage		Buffer de sortie
RAM échant.	ROM coeff.	RAM échant.	ROM coeff.	RAM échant.
$2 \times 2N_{max}b_W$ $= 512$ Kbits	$2\left(\frac{9N_{max}}{16}\right)b_W$ $= 144$ Kbits	$2\left(2L_{max} - 1\right)M_{max}b_W$ $= 1.920$ Mbits	$L_{max}M_{max}b_W$ $= 512$ Kbits	$2 \times 2N_{max}b_W$ $= 512$ Kbits
Capacité mémoire totale : 3600 Kbits				

Le Virtex-4 XC4VSX35 dispose d'une capacité mémoire qui suffit à notre architecture mais n'offre pas assez de flexibilité au niveau des blocs mémoires RAMs. Notre architecture est taillée sur mesure en vue d'une basse consommation et par conséquent engendre un gros nombre de mémoires. Afin de réaliser le prototypage FPGA, une version allégée de l'architecture proposée a été élaborée.

Le prototypage a été réalisé avec une longueur de troncature L égale à 2 seulement dans le but de réduire le nombre de blocs mémoires nécessaires. De plus, chaque banc mémoire RAM

à taille variable de 8 à 1024 points de la figure 6.2 des mémoires des échantillons de la TFR et de filtrage de mise en forme a été remplacé par un seul bloc à 1024 points. Les stratégies mémoires pour les coefficients de rotation W pour la TFR et celle des coefficients de la fonction IOTA restent inchangées. Ainsi, pour cette version, la gestion des mémoires selon la taille N n'est réalisée que pour les mémoires des coefficients W et IOTA.

Les tableaux 9.3 et 9.4 illustrent respectivement le nombre de blocs et la capacité mémoire nécessaire pour la réalisation de la version allégée de l'architecture. Pour cette version de l'architecture, le nombre de blocs mémoires total est de 170 blocs sur 192 disponibles pour une capacité mémoire totale de 1680 Kbits.

TABLE 9.3 – Nombre de blocs mémoires de la version allégée de l'architecture proposée

Nombre de blocs mémoires				
TFR		Filtrage		Buffer de sortie
RAM échant.	ROM coeff.	RAM échant.	ROM coeff.	RAM échant.
$2 \times 2 \times 8$ $= 32$	2×7 $= 14$	$2 \times (2L-1) \times 4$ $= 24$	$L \times 4 \times 8$ $= 68$	$2 \times 2 \times 8$ $= 32$
Total : 170 blocs RAMs				

TABLE 9.4 – Capacité mémoire de la version allégée de l'architecture proposée avec une longueur des données b_W de 16 bits

TFR		Filtrage		Buffer de sortie
RAM échant.	ROM coeff.	RAM échant.	ROM coeff.	RAM échant.
$2 \times 2N_{max}b_W$ $= 512$ Kbits	$2\left(\frac{9N_{max}}{16}\right)b_W$ $= 144$ Kbits	$2\,(2L-1)\,M_{max}b_W$ $= 384$ Kbits	$LM_{max}b_W$ $= 128$ Kbits	$2 \times 2N_{max}b_W$ $= 512$ Kbits
Capacité mémoire totale : 1680 Kbits				

9.3 Simulations et résultats

Plusieurs simulations ont été réalisées à différentes étapes du flot de conception sur FPGA, à savoir des simulations RTL (*Register Transfer Level*) au niveau porte, des simulations post-synthèse et finalement post placement-routage. Ceci nous a permis de vérifier le bon fonctionnement du modulateur à différentes étapes du prototypage et ainsi de valider l'architecture proposée.

Les figures 9.5 à 9.8 illustrent le résultats des simulations selon différents paramètres systèmes. Elles ont été réalisées sur la version complète de l'architecture et non sur la version allégée. Ces simulations au niveau comportemental nous ont permis de vérifier tous les cas de figures des paramètres d'entrée.

La figure 9.5 présente les résultats pour une modulation OFDM/QAM à 1024 porteuses en mode SISO. Il s'agit d'une wobulation, c'est-à-dire que des porteuses successives, soit les porteuses 0 à 4, sont stimulées successivement pour les différents symboles 0 à 4. Les différentes porteuses modulées pour chaque symbole sont illustrées sur la figure 9.5. On constate le déphasage de 90 degrés entre la partie réelle et imaginaire. Pour le premier symbole, la porteuse 0 donne en sortie un signal continu pour la partie réelle et un signal nul pour la partie imaginaire.

Les figures 9.6 à 9.8 illustrent le cas d'une modulation OFDM/OQAM. Afin de vérifier le bon fonctionnement du filtrage, nous n'avons modulé qu'une seule porteuse d'un symbole tous les $2L$ symboles. Ainsi, le signal en sortie selon l'équation 2.6 est égale à $S_{j,k} = \Im_{k+sM+2\ell M} C_{k+sM}$, soit les résultats de la TFR, C_{k+sM}, pondérés par la fonction IOTA. Pour la porteuse continue 0, on obtient ainsi en sortie la fonction IOTA elle même sur 2L symboles. Pour les autres porteuses 16, 32 et 48, on obtient des sinusoïdales de fréquences 16, 32 et 48 fois plus rapides que la fréquence fondamentale pondérée par la fonction IOTA. Finalement, il est à noter que la partie imaginaire pour le cas de la porteuse 32, soit la moitié de $N = 64$, est nulle comme pour le cas de la porteuse 0.

On constate qu'il existe un grand intervalle nul entre les différentes sinusoïdes modulées par la fonction IOTA à la figure 9.6. Ceci est dû au fait que la fonction IOTA pour $L = 8$ est nulle sur une grande partie des $2ML$ coefficients comme illustré à la figure 6.17. Les bancs mémoires des coefficients IOTA contenant des valeurs nulles ont été gardés afin d'obtenir une architecture générique et adaptable à différentes fonctions de mise en forme symétriques, autres que la fonction IOTA.

Cet écart entre les différentes sinusoïdes modulées par la fonction IOTA diminue pour les valeurs de L égales à 4 et 2 comme illustré sur les figures 9.7 et 9.8 respectivement. La figure 9.7 représente l'exemple d'une modulation à 256 points pour $L = 4$ en mode MIMO 2x2. La figure 9.8 représente l'exemple d'une modulation à 512 points pour $L = 2$ en mode MIMO 4x4. Les différentes porteuses modulées sont indiquées sur les figures.

Le tableau 9.5 indique le nombre de cycles pour la réalisation d'une TFR directe ou inverse ainsi que pour le filtrage de mise en forme. Ces résultats concernent à la fois la version complète de l'architecture proposée et la version allégée pour le cas $L = 2$. La latence au niveau de la deuxième et troisième colonne de la matrice est de 6 pour le calcul de la TFR, tandis qu'elle est de 2 pour la première colonne. Ainsi, le temps de parcours de la matrice pour un étage radix-2^3 est de 14 cycles, 12 cycles pour le radix-2^2 et finalement 6 cycles pour le radix-2. Dans le cas du filtrage de mise en forme, la latence est de 2 pour les deux dernières colonnes.

La figure 9.4 reprend les résultats du tableau 9.5 et indique le nombre de cycles pour la réalisation d'une modulation OFDM avancée selon les différents paramètres. On remarque l'influence du taux de parallélisme $P_{D_{TFR}}$ qui dicte directement la valeur de r_{MIMO}. L'emploi d'un radix élevé tel que le radix-2^3 diminue significativement le nombre de cycles nécessaires pour réali-

TABLE 9.5 – Nombre de cycles pour la TFR et le filtrage selon N, L et le mode MIMO

N	SISO ($P_{D_{TFR}} = 8$)			MIMO 2x2 ($P_{D_{TFR}} = 4$)			MIMO 4x4 ($P_{D_{TFR}} = 2$)		
	TFR	Filtre $L =$		TFR	Filtre $L =$		TFR	Filtre $L =$	
		2 ou 4	8		2 ou 4	8		2 ou 4	8
64	45	9	18	85	17	34	229	33	66
128	83	17	34	171	33	66	491	65	130
256	137	33	66	305	65	130	1073	129	258
512	235	65	130	695	129	258	2359	257	514
1024	561	129	258	1341	257	514	5181	513	1026
2048	1079	257	514	3139	513	1026	11331	1025	2050
4096	2105	513	1026	6217	1025	2050	-	-	-
8192	5183	1025	2050	-	-	-	-	-	-

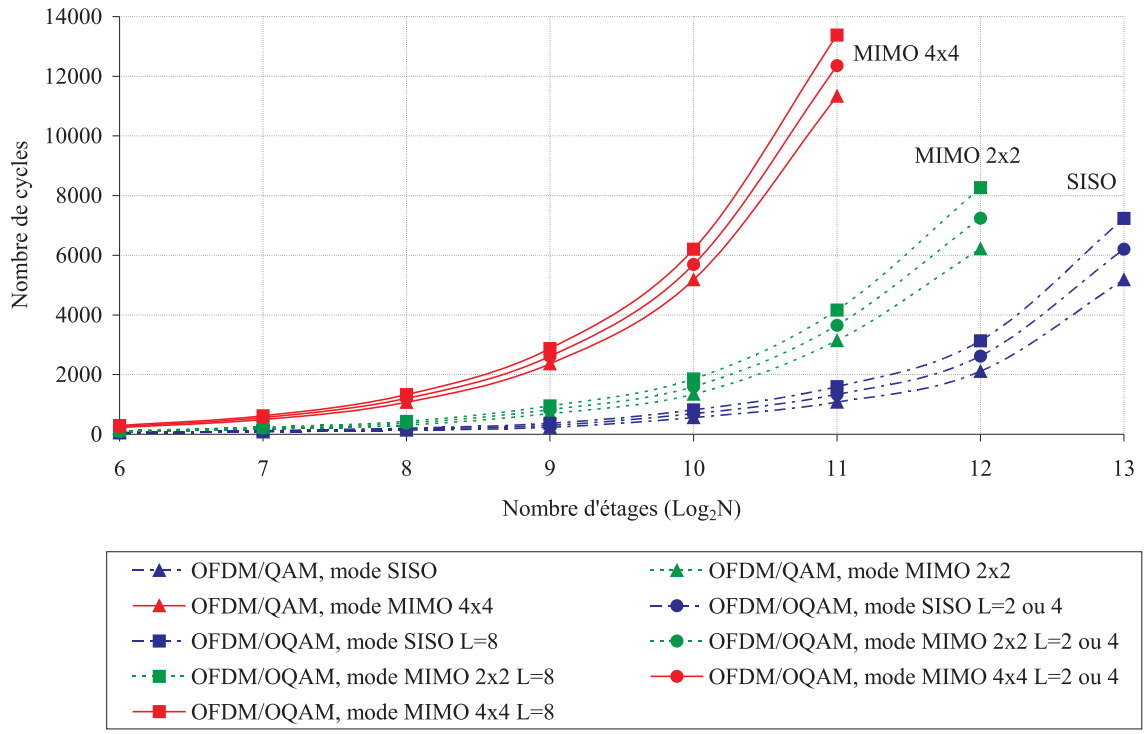

FIGURE 9.4 – Nombre de cycles d'une modulation d'un symbole OFDM/QAM ou OQAM selon N, L et le mode MIMO

ser la TFR. Une architecture reconfigurable multi-radix est donc particulièrement efficace. Pour des radix inférieurs comme le radix-2, les performances sont inférieures, toutefois, quatre TFRs peuvent être effectuées en même temps, ce qui procure une flexibilité supérieure à l'architecture, qui peut ainsi répondre à plusieurs paramètres différents.

La contribution du filtrage de mise en forme en terme de cycles de traitement supplémentaires par rapport à la TFR pour le mode SISO est en moyenne de 23% pour $L = 2$ ou 4 et de 46% pour $L = 8$. Pour le mode MIMO 2x2, il y a en moyenne 19% de cycles supplémentaires par rapport à la TFR pour $L = 2$ ou 4 et 37% pour $L = 8$. Finalement, pour le mode MIMO 4x4, la moyenne est de 12% et 23% selon les deux cas de figures. On remarque que la contribution du filtrage de mise en forme en termes de cycles de traitement diminue pour les modes MIMO 2x2 et 4x4. Cette diminution s'explique par le fait que le nombre de cycles pour le traitement de la TFR augmente significativement pour les modes MIMO 2x2 et 4x4.

Le tableau 9.6 illustre les ressources utilisées du FPGA XC4VSX35 de Xilinx par la version allégée de l'architecture. Les ressources logiques des FPGA de Xilinx sont constituées de blocs logiques configurables eux-mêmes constitués de briques de base appelées *Slice*. On constate que 84% des ressources logiques du FPGA sont utilisées, 33% pour la matrice de calcul et 51% pour les divers éléments restants tels que les générateurs d'indices et les modules de contrôle de l'architecture. Chaque *Slice* est constitué de deux tables de transcodage appelées *Look Up Table* (LUT) et des registres. La synthèse logique montre que seulement 3% des LUTs sont configurées en registres à décalage. Ces derniers sont principalement utilisés pour les divers décalages de bit réalisés dans la génération des adresses des échantillons et des coefficients.

TABLE 9.6 – Ressources utilisées du FPGA XC4VSX35 de Xilinx

Nombre de Slices	12949 sur 15360 (84%)
Nombre de Slices Flip Flops	1714 sur 30720 (5%)
Nombre de LUT à 4 entrées	23907 sur 30720 (78%)
Nombre utilisés comme logique	22922 (75%)
Nombre utilisés comme registre à décalage	985 (3%)
Nombre de bloc RAM	162 sur 192 (84%)
Nombre de DSP48	113 sur 192 (58%)
Nombre de IOBs	281 sur 448 (62%)

L'architecture utilise 162 blocs RAM de 18 Kbits au lieu des 170 blocs annoncés au tableau 9.3. En effet, les plus petites mémoires ROM de taille $4 \times 2b_W$ des bancs des coefficients IOTA (voir figure 6.25) utilisent des RAMs distribuées réalisées grâce à la logique combinatoire. La version de l'architecture intégrée dans le FPGA réalise le filtrage avec une valeur de L égale à 2 seulement. Nous avons donc 2 blocs mémoires de type 1 contenant chacun quatre bancs ROM (A à D), soit un total de 8 mémoires utilisant des ROMs distribuées. Finalement, l'architecture utilise 281 des entrées/sorties (*Input/output block IOB*) des 448 disponibles.

La fréquence d'opération maximale sur le FPGA est de 70 MHz. On constate donc une diminution importante entre la fréquence maximale lors de la synthèse logique de la matrice de calcul et celle de l'architecture complète. Le chemin critique est situé entre le compteur papillon et la génération des adresses des coefficients de rotation W. Comme on le constate sur le générateur d'indice des coefficients W à la figure 6.15, plusieurs additionneurs sont utilisés afin d'obtenir les différents indices. Ces additionneurs sont synthétisés grâce à la logique combinatoire. Il est possible d'augmenter la fréquence maximale en insérant des registres afin de diminuer le délai sur le chemin le plus long. Une autre solution serait d'utiliser les blocs DSP48 pour les diverses additions du générateur d'indice. Dans les deux cas, il est nécessaire de resynchroniser tous les chemins de données. Les performances en fréquences peuvent donc être améliorées de quelques dizaines de MHz. Toutefois, cet effort d'optimisation n'a pu être réalisé lors du prototypage.

La consommation de puissance a été déterminée grâce à l'outil logiciel *Xpower Analyser* de Xilinx. La consommation statique est de 122 mW et la consommation de puissance dynamique est de 599 mW à la fréquence maximale, soit 8,6 mW/MHz. Toutefois, ce logiciel peut présenter des erreurs d'estimation de l'ordre de 10% de la consommation de puissance réelle pour une TFR radix-2 [125]. Il n'a pas été aussi possible de mesurer la consommation de l'architecture selon les différentes paramètres N et du mode MIMO.

Le tableau 9.7 illustre la fréquence minimale nécessaire au circuit proposé pour diverses normes à base de la modulation OFDM/QAM. On considère aussi d'hypothétiques normes l'OFDM/OQAM avec une longueur de troncature L de 2. Dans les deux cas, on considère un seul flux de données (mode SISO). Ces fréquences ont été obtenues à partir du nombre de cycles de traitement du tableau 9.5. On constate qu'à l'exception de la norme UWB, le système est capable de répondre aux différentes exigences en fréquences des différentes normes. Pour ces dernières, les fréquences d'opération sont particulièrement basses par rapport à la fréquence maximale. La diminution de la puissance dynamique sera donc proportionnelle au gain de fréquence par rapport à la fréquence maximale.

On peut noter que pour la norme DVB-H, la fréquence d'opération pour la version à 4096 points est légèrement inférieure à la version à 2048 points. Le temps symbole est aussi deux fois plus long pour la version à 4096 points. Or, pour cette dernière version, l'architecture n'utilise que l'algorithme radix-2^3 ($\log_2 2048 = 12$ est un multiple de 3), tandis que pour la version à 2048 points, il est nécessaire d'utiliser l'algorithme radix-2^2 avant le radix-2^3. Ceci explique la fréquence légèrement supérieure pour la version à 2048 points.

TABLE 9.7 – Fréquence d'opération du modulateur proposé pour différentes normes existantes en mode SISO et extrapolation pour une modulation OFDM/OQAM avec $L = 2$

Norme	Largeur de bande (MHz)	Période d'un symbole* (μs)	N	Fréquence d'opération (MHz)	
				OFDM	
				QAM	OQAM
3GPP-LTE	1,25	102,4	128	0,8	1,9
	2,5		256	1,3	3,3
	5		512	2,4	5,9
	10		1024	5,5	13,5
	20		2048	10,5	26,1
WIFI	20	3,2	64	14	33,7
Wimax Mobile	1,25	91,4	128	0,9	2,2
	5		512	2,6	6,6
	10		1024	6,1	15,1
	20		2048	11,8	29,2
DVB-H	8	224	2048	4,8	11,9
		448	4096	4,7	11,7
		896	8192	5,8	13,9
UWB MB-OFDM	528	0,242	128	342,4	825

*Le temps symbole est calculé sans l'intervalle de garde pour la modulation OFDM/QAM

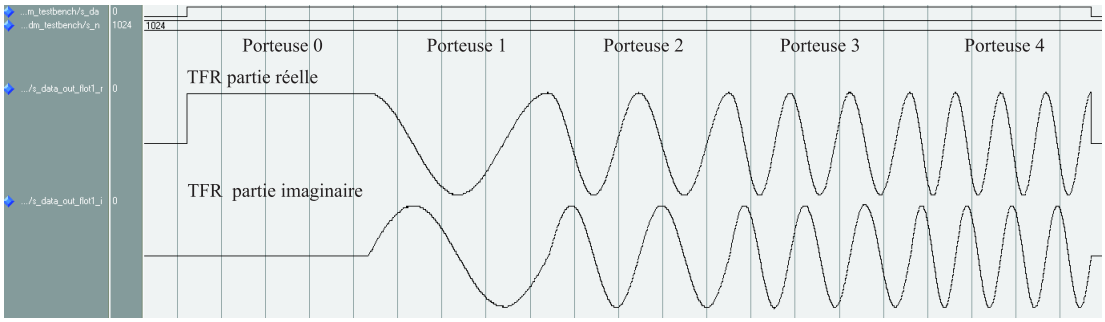

FIGURE 9.5 – Simulation pour une modulation OFDM/QAM pour $N = 1024$ en mode SISO

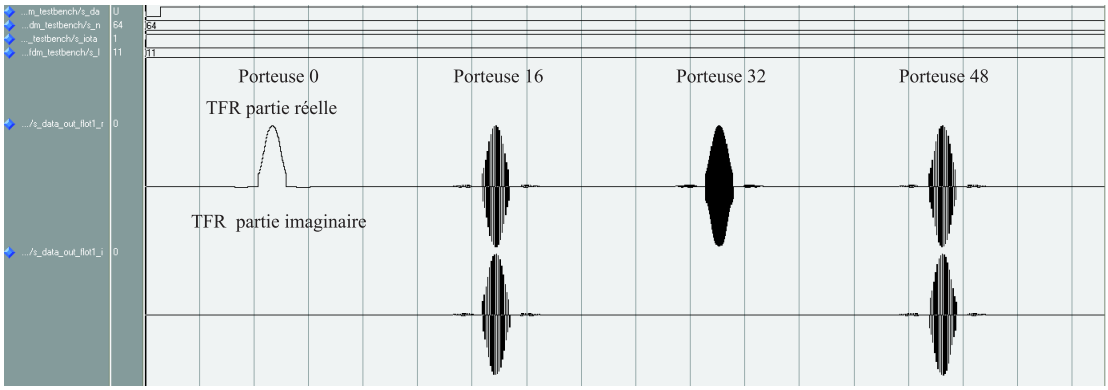

FIGURE 9.6 – Simulation pour une modulation OFDM/OQAM pour $N = 64$ et $L = 8$ en mode SISO. N.B : L'allure de la fonction IOTA est légèrement altérée pour des raisons d'affichage.

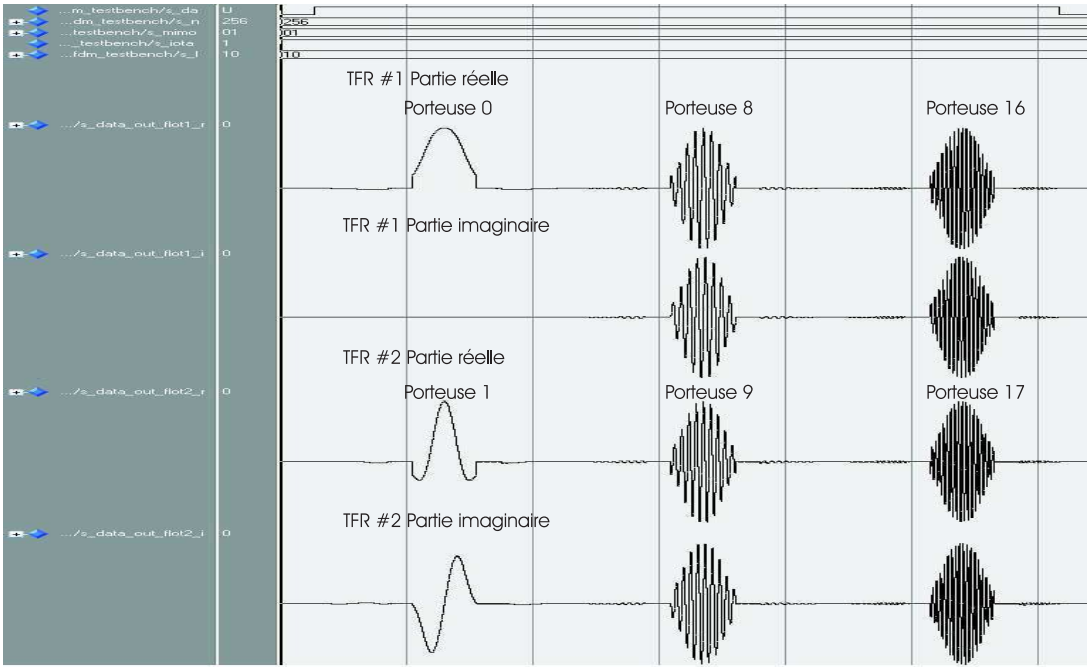

FIGURE 9.7 – Simulation pour une modulation OFDM/OQAM pour $N = 256$ et $L = 4$ en mode MIMO 2x2. N.B : L'allure de la fonction IOTA est légèrement altérée pour des raisons d'affichage.

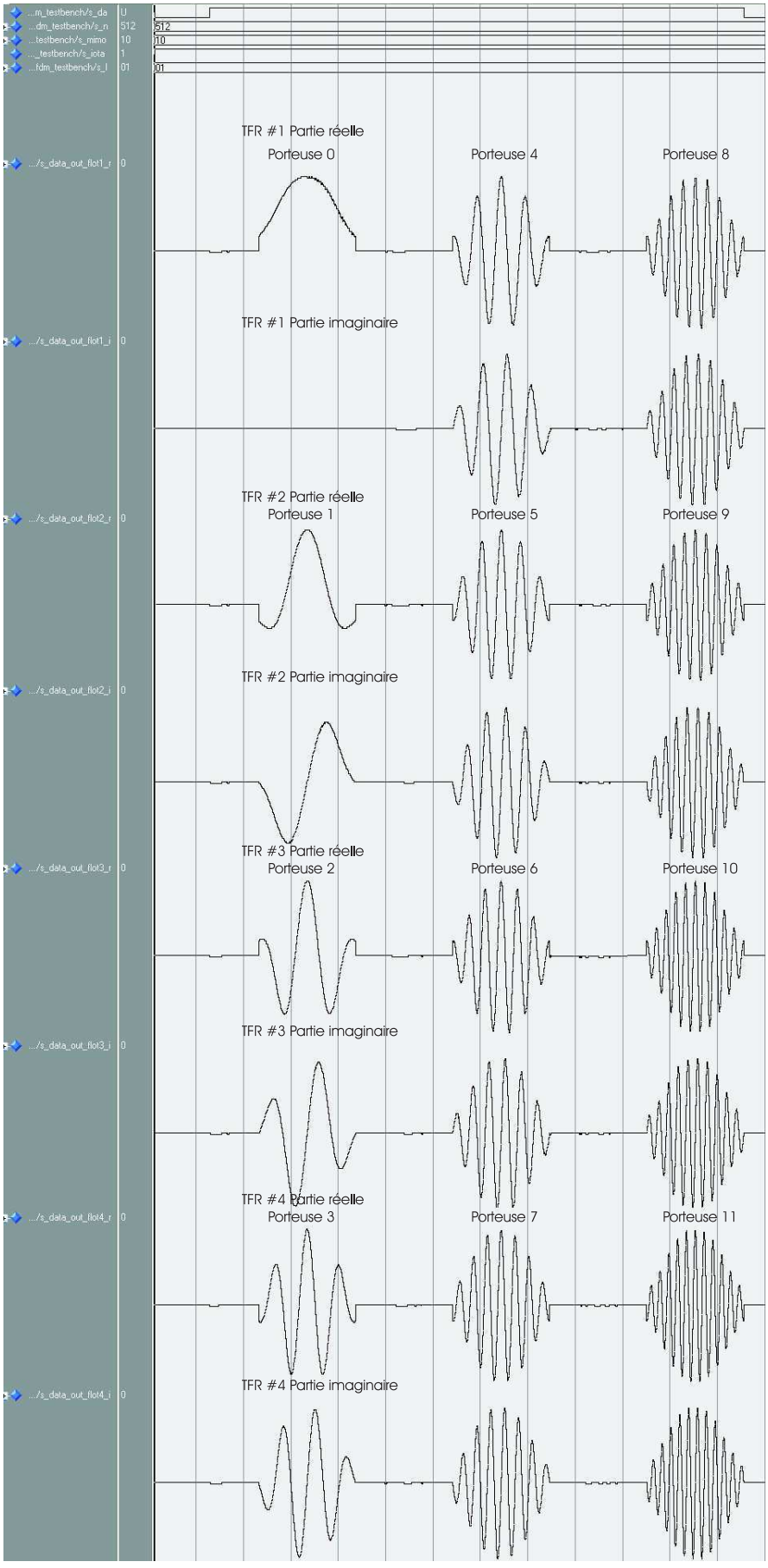

FIGURE 9.8 – Simulation pour une modulation OFDM/OQAM pour $N = 512$ et $L = 2$ en mode MIMO 4x4. N.B : L'allure de la fonction IOTA est légèrement altérée pour des raisons d'affichage.

Chapitre 10

Synthèse physique du circuit ASIC

Ce chapitre présente les résultats de la synthèse physique (*layout*) du circuit dédié ASIC du modulateur OFDM avancé proposé. Comparativement au FPGA, la technologie ASIC fournit une optimisation maximale des paramètres du circuit en termes de surface, vitesse et consommation de puissance. Le prototypage FPGA nous a permis de tester au préalable le circuit sur un support programmable avant de s'intéresser à la synthèse logique du circuit dédié ASIC. Cela donne rapidement une première idée du comportement réel du système et des ressources matérielles nécessaires à la mise en oeuvre de l'architecture proposée.

Contrairement à l'intégration FPGA, la synthèse physique du circuit ASIC nous a permis de réaliser la version complète de l'architecture, c'est-à-dire pour toutes les valeurs de L. De plus, les mémoires des échantillons de la TFR et du filtrage de mise en forme utilisent la stratégie mémoire proposée tel qu'illustrée à la figure 6.2. Elle est composée d'un banc de ROM de taille allant de 8 à 1024 échantillons. La matrice de calcul a été implémentée de façon générique et n'utilise donc pas les blocs DSP48 présents dans le FPGA de Xilinx. Des schémas détaillés des modules de calcul sont présentés à l'annexe D. Toutes les données sont sur 16 bits comme pour le prototypage FPGA.

Les simulations et certains résultats présentés au chapitre précédent sur le prototypage FPGA restent valables pour le synthèse physique du circuit ASIC. Ainsi, les nombres de cycles nécessaires à la réalisation de la TFR et du filtrage de mise en forme présentés au tableau 9.5 correspondent aussi à l'architecture dédiée ASIC. Il en est de même des fréquences d'opération selon les différentes normes présentées au tableau 9.7.

Le circuit dédié a été réalisé en technologie de gravure CMOS submicronique de 65 nm de STMicroelectronic. Les résultats présentés tiennent donc compte d'une des technologie d'intégration silicium les plus récentes. La synthèse logique a été réalisée grâce au logiciel Design Vision de Synopsys. Le placement routage a été réalisé par le logiciel SoC Encounter de Cadence.

10.1 Résultats d'implémentation

La figure 10.1 illustre le plan d'implantation (*Floorplan*) du circuit après le placement des différents blocs de l'architecture proposée. Les figures 10.2 et 10.3 illustrent respectivement les schémas de l'arbre d'horloge et de routage de l'architecture proposée.

La surface totale du circuit proposé avec une technologie de gravure de 65 nm est de 12,5 mm^2. La logique combinatoire occupe 0,62 mm^2, soit 5% de la surface totale. Le tableau 10.1 présente les surfaces pour les différentes parties du circuit. On constate que les ressources mémoires

TABLE 10.1 – Surface pour les principales parties de l'architecture en technologie CMOS 65 nm de STMicroelectronic

Composants	Surface	
	en mm^2	en %
Ressources de la TFR		
RAM TFR 1	1,05	8,4
RAM TFR 2	1,05	8,4
ROM TFR	0,07	0,6
Surface totale pour les mémoires de la TFR	2,17	17,4
Ressources du filtrage		
RAM IOTA	7,73	62
ROM IOTA	0,14	1
Surface totale pour les mémoires du filtrage	7,87	63
Ressources arithmétiques		
Matrice de calcul	0,31	2,5
Ressources pour le contrôle		
Contrôle Système	0,001	0,008
Autres		
Buffer de sortie 1	1,05	8,4
Buffer de sortie 2	1,05	8,4
Divers	0,02	0,2
Surface totale	12.47	100

allouées au filtrage de mise en forme représentent 63% de la surface totale. Ceci s'explique par la grande valeur de $L_{max} = 8$ qui engendre une capacité mémoire de $(2L_{max} - 1)M_{max}$ échantillons. Cette capacité mémoire est due à l'algorithme de filtrage et non à l'architecture choisie. On remarque aussi que les coefficients de la fonction IOTA ne représentent que 1% de la surface. Les ressources mémoires de la TFR ne représentent que 17,4 % de la surface. En tenant compte des buffers de sorties, l'architecture pour $L_{max} = 8$ est donc à 97,2 % composée de mémoire. La surface restante est occupée à 2,5 % par la matrice, et à plus de 0,2 % par le module de contrôle et divers composants. Une conception avec $L_{max} = 4$, aurait donné une surface totale de 8,3 mm^2, soit une diminution de 34 % de la surface par rapport à $L_{max} = 8$. Dans ce cas, la surface aurait été composé à 96 % par les mémoires et à 3,8 % par la matrice de calcul. Pour $L_{max} = 2$, la surface totale aurait été de 6,2 mm^2, soit une diminution de 50,5 % de la surface par rapport à $L_{max} = 8$. Dans ce cas, la surface aurait été composé à 94,5 % par les mémoires et à 5 % par la matrice de calcul.

La matrice de calcul occupe donc une part négligeable de la surface, malgré les nombreuses ressources arithmétiques et les nombreux multiplexeurs présents dans la matrice. En effet, la matrice est constituée de 8 modules de calcul à 8 multiplieurs réels et 8 additionneurs réels, et 4 modules de calculs à 4 additionneurs réels, soit un total de 64 multiplieurs réels et 80 additionneurs réels. Ceci nous assure un très grand débit de traitement. L'approche fortement parallèle rendue possible grâce à ces opérateurs arithmétiques nous permet d'opérer à très basse fréquence comme illustré au tableau 9.7 tout en limitant la surface occupée.

La fréquence maximale d'opération du circuit est de 270 MHz. Le chemin critique est situé dans les interfaces de lecture des buffers de sortie. Ces interfaces instancient les équations 6.3 et

6.4 qui déterminent respectivement le banc mémoire et l'adresse mémoire des échantillons selon la valeur de r_{MIMO}. Comme dans le cas du prototypage FPGA, le chemin critique peut être diminué. Un travail d'optimisation de l'architecture permettrait d'augmenter encore la fréquence maximale au prix d'une augmentation légère du nombre de cycles de traitement.

La technologie avancée CMOS 65 nm autorise une alimentation en tension basse de 0,9 V. La consommation de puissance dynamique est de 378,5 mW à la fréquence maximale, ce qui représente une variation de $1,4mW/MHz$, soit plus de 6 fois moins que le prototypage FPGA. En valeur absolue, la consommation dynamique a diminué de 37% par rapport à la consommation dynamique du prototypage FPGA. Quant aux courants de fuites, ils représentent une perte de 1,23 mW.

Le design ASIC de l'architecture proposée est capable de répondre à toutes les normes OFDM existantes à l'exception du cas de l'UWB. Il est aussi en mesure de réaliser la version OFDM/OQAM pour ces mêmes normes. À première vue, la consommation de puissance totale après la synthèse physique de l'ASIC semble assez importante. Néanmoins, elle est donnée pour la fréquence maximale. Or, pour les normes visées et réalisables par le circuit proposé, les fréquences d'opération sont de quelques MHz voire même inférieures dans certains cas. Par conséquent, la consommation totale du circuit est de l'ordre de quelques mW.

10.2 Comparaison des performances

Selon nos connaissances, il n'existe pas à l'heure actuelle, dans l'état de l'art, d'architectures réalisant une modulation OFDM/OQAM à des fins de comparaison. Toutefois, nous pouvons réaliser une comparaison qualitative de l'architecture en mode TFR par rapport à d'autres architectures. Nous considérons la surface de l'architecture proposée sans les ressources mémoires pour le filtrage de mise en forme. Les buffers de sorties ne sont pas considérés non plus pour cette comparaison. Nous avons alors pour l'architecture de TFR une surface de 2,5 mm^2. Il est difficile de comparer des architectures ayant été réalisées avec des technologies différentes. Pour cela, il est nécessaire d'utiliser des métriques normalisées.

Nous avons comparé l'architecture proposée dans un premier temps en termes de débit maximal de transmission qui est égale à $\frac{N_{max}}{\tau_{min}}$, où τ_{min} représente le temps d'exécution obtenu à la fréquence d'opération maximale. Nous avons aussi normalisé la surface des différentes architectures de comparaison pour une technologie de gravure de 65 nm selon l'équation proposée dans [106]. Nous avons :

$$Surface_{Normalise} = \frac{Surface}{\left(\frac{Technologie}{65nm}\right)^2} \tag{10.1}$$

Nous constatons au tableau 10.2 que la surface de l'architecture proposée est la plus importante. Toutefois, les architectures reconfigurables [126] et [127] ayant les plus petites surfaces normalisées réalisent une taille de TFR maximale de 2048 et 1024 points respectivement. Ainsi, pour réaliser une TFR de 8192 points, les surfaces seraient deux et quatre fois plus importantes, soit approximativement égales à la surface normalisée reconfigurable (2K/4K/8K) de l'architecture [103]. On constate aussi une différence entre les surfaces normalisées de l'architecture non reconfigurable de 8192 points [102] et l'architecture reconfigurable [103]. Cette différence est due

aux ressources nécessaires à la reconfiguration. Ainsi, la surface de l'architecture proposée est induite par la taille maximale de TFR de 8192 points, la reconfigurabilité de l'architecture, ainsi que par le nombre de ressources arithmétiques. De plus, le format des données varient d'une architecture à l'autre. Dans notre cas, les données sont 16 bits, au lieu de 12 bits maximum pour les autres architectures.

En terme de débit à la sortie du circuit, l'architecture proposée permet une transmission plus de 8 fois plus rapide en mode SISO que l'architecture de comparaison la plus rapide [127]. Pour le mode MIMO 2x2, le débit est plus de 3 fois plus rapide et deux TFR sont réalisés. Finalement, pour le mode 4x4, le débit est relativement le même mais quatre TFRs sont réalisées dans le cas de l'architecture proposée.

En ce qui concerne la puissance consommée, la consommation obtenue pour notre architecture englobe aussi les ressources du filtrage IOTA et des buffers de sortie. Il est donc difficile de comparer les différentes architectures en terme de consommation. À titre d'indication, le tableau 10.2 illustre la variation de la consommation par MHz. Il est à noter que cette variation est donnée pour des architectures réalisées avec des technologies différentes, et donc différentes tensions d'alimentation.

TABLE 10.2 – Comparaisons des performances du traitement de la TFR selon plusieurs architectures différentes

	Proposée mode MIMO			[102]	[103]	[126]	[127]
	SISO	2x2	4x4				
Architecture	à mémoire					pipeline	
Algorithmes	Multi-radix (MR) $2\text{-}2^2\text{-}2^3$			radix-2	MR $2\text{-}2^3\text{-}2^4$	MR 2-4-8	MR $2\text{-}2^2$
♯ MC	12			4	4	11	10
Technologie	65 nm			0,18 μm		0,35 μm	
Tension (V)	0,9			-	-	2,3	2
N	64 à 8K	64 à 4K	64 à 2K	8K	2K/4K /8K	512/1K /2K	32 à 1K
f_{max} (MHz)	270			172	-	45	50
b_w (bits)	16			12	-	12	8 à 14
Surface (mm^2)	2,504235			3,4965	6,34	13,05	8,5
Surface normalisée				0,456	0,827	0,4656	0,2932
Débit max. (Méchan./s)	426,7	2× 177,8	4× 48,8	30,45	36,56 à f_{op} = 5 MHz	45	50
Puissance (mW)	378,5065 à 270 MHz			31,1 à 22 MHz	-	176 à 17,8 MHz	76 à 25 MHz
$\Delta mW/MHz$	1,40			1,41	-	9,88	3,04

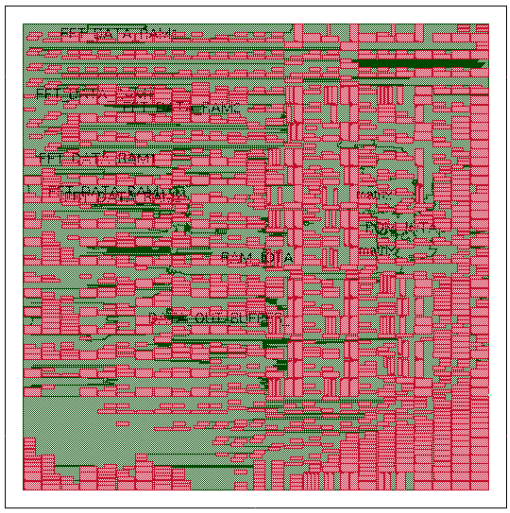

FIGURE 10.1 – Schéma du plan d'implantation (*Floorplan*) de l'architecture proposée

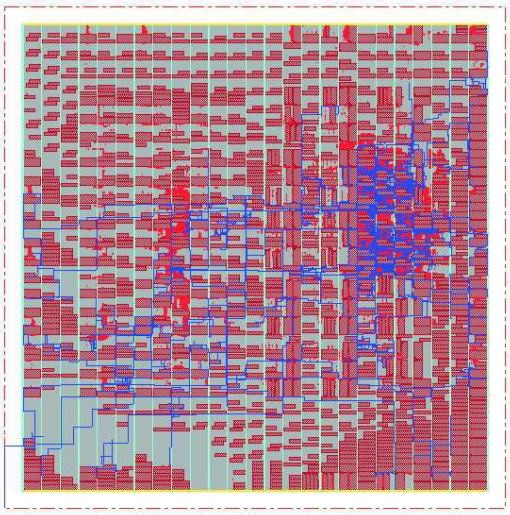

FIGURE 10.2 – Schéma de l'arbre d'horloge de l'architecture proposée (en bleu)

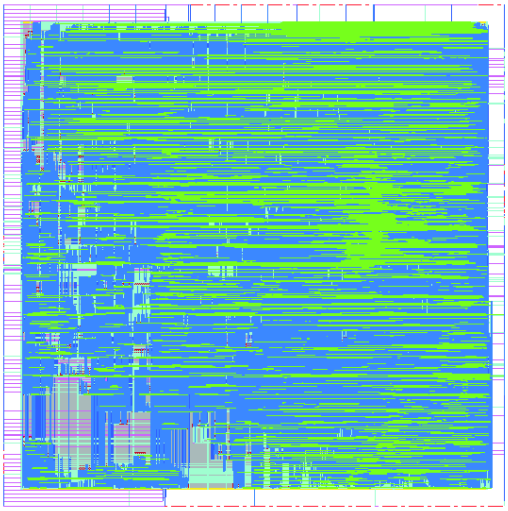

FIGURE 10.3 – Schéma du routage de l'architecture proposée

Conclusion

Cette partie nous a permis de présenter les résultats d'implémentation et d'expérimentation de l'architecture proposée. Ces travaux d'intégration dans une plateforme FPGA et la synthèse physique du circuit ASIC ont montré le bon fonctionnement et la faisabilité d'une telle architecture. Plusieurs optimisations sont possibles afin d'augmenter les performances, telles que la fréquence maximale du circuit, ou encore un plus grand effort d'optimisation de la surface pour l'intégration ASIC. Comme nous l'avons mentionné, ces travaux ont été réalisés par trois stagiaires que j'ai encadrés [33–35], et les diverses optimisations n'ont pas pu être réalisées au cours de ce projet. Les résultats obtenus représentent donc une limite inférieure des performances atteignables par ce type d'architecture.

Le prototypage FPGA nous a permis de constater entre autre les limites d'intégration de l'architecture. La granularité mémoire proposée joue un rôle important et n'est pas adaptée à l'architecture mémoire des FPGAs. Toutefois, en terme de puissance de traitement arithmétique, la matrice de calcul tire avantage de la présence des blocs DSP dans les FPGAs.

Les résultats obtenus pour la synthèse physique de l'ASIC nous ont montré que la surface du circuit est à dominante mémoire à plus de 97 % (pour $L_{max} = 8$) malgré le nombre important de ressources arithmétiques. Une conception avec $L_{max} = 4$, aurait donné une surface totale de 8,3 mm^2, soit une diminution de 34 % de la surface par rapport à $L_{max} = 8$. Dans ce cas, la surface aurait été composée à 96 % par les mémoires et à 3,8 % par la matrice de calcul. Pour $L_{max} = 2$, la surface totale aurait été de 6,2 mm^2, soit une diminution de 50,5 % de la surface par rapport à $L_{max} = 8$. Dans ce cas, la surface aurait été composée à 94,5 % par les mémoires et à 5 % par la matrice de calcul.

Comme attendu, le fort degré de parallélisme permet d'opérer à des fréquences beaucoup plus basses que les largeurs de bandes couvertes par les différentes normes. Cela aura un impact positif sur la consommation de puissance du circuit du fait que la puissance consommée par MHz est très faible, soit $1,4mW/MHz$. Le débit maximal obtenu est de 426,7 méga échantillons par seconde.

Conclusion générale

Cette thèse avait pour but d'étudier les architectures de circuits intégrés pour le traitement numérique de l'OFDM avancé multi-standard, très haut débit et basse consommation. Deux schémas de modulation différents ont été pris en compte, soit l'OFDM/QAM et l'OFDM/OQAM. Au delà de l'aspect matériel de la modulation OFDM, nous avons donc intégré l'impact du filtrage de mise en forme dans le cas particulier de l'OFDM/OQAM.

Une analyse comparative des différentes algorithmes et architectures de la TFR a permis de déterminer la meilleure approche nous permettant d'obtenir une bonne adéquation algorithme architecture. Ainsi, nous avons montré que l'association des algorithmes de TFR radix-2^i avec des architectures en pipeline ou à mémoires offrent les meilleures performances. Toutefois, pour répondre à la contrainte de reconfigurabilité de l'architecture pour des applications multi-standard, nous avons démontré que les architectures à base de mémoires à grand degré de parallélisme offrent le meilleur compromis entre les contraintes de débit, de reconfiguration et de basse consommation. Cette approche temporelle qui caractérise les architectures à base de mémoires permet de réutiliser les ressources arithmétiques pour le filtrage de mise en forme IOTA ce qui n'est pas possible avec les architectures en pipeline.

Nous avons donc proposé une architecture à base de mémoires utilisant un multiplexage temporel des opérations sur une matrice de calcul à gros grain optimisée pour le traitement de la TFR et le filtrage de mise en forme. Cette approche temporelle permet une réalisation de modulation OFDM avancée pour des valeurs du paramètre N, le nombre de porteuses de la modulation, allant de 64 à 8192 et du paramètre L, la longueur de troncature pour le filtrage, égal à 2, 4 et 8. La matrice de calcul exploite au maximum le parallélisme des données et le parallélisme spatiale permettant d'atteindre l'objectif de très haut débit, surtout si on tient compte du filtrage de mise en forme IOTA, qui impose une fréquence d'opération deux fois supérieure à une modulation OFDM/QAM. La matrice est aussi configurable selon les paramètres de modulation afin de désactiver les ressources non nécessaires. En mode MIMO, l'architecture de la matrice applique le même traitement sur deux ou quatre flux d'échantillons différents, pour les modes MIMO 2x2 et 4x4 respectivement. On obtient ainsi deux modulations en parallèle de taille maximale de 4096 porteuses ou quatre modulations en parallèle de taille maximale de 2048 porteuses.

Afin de répondre à la contrainte de basse consommation et de reconfigurabilité, une stratégie pour la gestion des mémoires a été proposée. Elle consiste en des bancs de mémoires permettant d'obtenir des tailles variables et de désactiver les mémoires non nécessaires. Cette stratégie a été appliquée à toutes les mémoires des échantillons de la TFR et de filtrage, ainsi qu'aux coefficients IOTA. Pour ces derniers, l'utilisation de mémoires scalables a été rendue possible

grâce à l'élaboration d'un schéma de mémorisation des coefficients IOTA selon la parité de leur indice. Ce schéma permet de mémoriser les coefficients IOTA désirés selon les valeurs de N et L dans les mémoires activées. Une organisation mémoire a aussi été proposée pour les coefficients des facteurs de rotation W. Cette organisation permet de désactiver les mémoires non nécessaires selon l'algorithme utilisé.

Notre solution utilise deux mémoires de traitement pour la TFR et deux buffers de sortie, pour une capacité totale de $4N_{max} = 32768$ données complexes, plus $(2L_{max} - 1)M_{max} = 61440$ données complexes pour filtrage de mise en forme IOTA. Dans le cas du filtrage, la capacité nécessaire dépend de l'algorithme de filtrage et aucune optimisation n'est possible. L'utilisation de nouvelles fonctions de mise en forme permettant des valeurs de troncature L plus petites diminuerait donc la ressource mémoire totale du filtrage. Des solutions existent par contre pour diminuer la capacité des mémoires de traitement et des buffers de sortie. Des approches utilisant alternativement des algorithmes en DIT et DIF pour le traitement de la TFR, ou utilisant des schémas de mémorisation particuliers pour les buffers de sortie, devraient être explorées.

Une étude théorique, complétée par des simulations, a porté sur l'aspect quantification des chemins des données et de calcul. Au préalable, et pour profiter des avantages de la pureté spectrale de la fonction IOTA, nous avons fixé un cahier de charges très contraignant sur l'impact de la quantification, à savoir un rapport signal sur bruit de quantification de l'ordre de -50 dB. Afin d'atteindre cet objectif, nous avons constaté qu'une représentation des données en virgule fixe en complément à deux de 20 bits pour les échantillons (à la fois pour la TFR et le filtrage IOTA), 15 bits pour les coefficients de rotation W et 13 bits pour les coefficients de la fonction IOTA, permettent d'atteindre le RSBQ de -50 dB pour un nombre de porteuses pouvant aller jusqu'à 8192 et une longueur de troncature pour le filtrage allant jusqu'à $L = 8$. Il serait intéressant d'étudier d'autres alternatives à la représentation des données. Des solutions alliant une représentation virgule fixe et virgule flottante offrent une meilleures dynamique des nombres avec une longueur des données limitée. Toutefois, ces performances sont au prix d'une complexité accrue et engendrent des ressources supplémentaires. Il est donc important de comparer les différentes alternatives pour la quantification des données avec celle proposée.

La spécification de l'architecture pour la synthèse RTL a été faite en VHDL. Un premier prototypage FPGA et une première synthèse physique d'un circuit ASIC (*layout*) ont permis de valider le bon fonctionnement et la faisabilité d'une telle architecture. La plateforme de prototypage FPGA utilisée était la ML402 de la société Xilinx intégrant le FPGA XC4VSX35 de la famille Virtex-4. Dans le cas de la conception du modèle physique ASIC, la technologie de gravure utilisée était la technologie CMOS submicronique de 65 nm de STMicroelectronic. Les performances obtenues pour le prototypage FPGA et pour la synthèse physique ASIC permettent de réaliser une solution hautement configurable couvrant la cible initiale de toutes les normes considérées sauf le standard UWB très haut débit. Un travail d'optimisation du modèle physique du circuit ASIC permettrait d'atteindre la fréquence de 342,4 MHz nécessaire à la norme UWB pour le schéma QAM. Toutefois, le cas du schéma OQAM pour la norme UWB impose une fréquence élevé de 825 MHz. Le débit maximal obtenu est de 426,7 méga échantillons par seconde en mode SISO.

La conception du circuit physique ASIC a montré que la surface de l'architecture est à 97% composée de mémoires pour une valeur de L égale à 8, les ressources arithmétiques ne représentant que 2,5% de la surface. Une conception avec $L_{max} = 4$, aurait donné une diminution de 34 % de la surface par rapport à $L_{max} = 8$. Dans ce cas, la surface aurait été composée à 96 % par les mémoires et à 3,8 % par la matrice de calcul. Pour $L_{max} = 2$, la diminution de la surface totale aurait été de 50,5 % par rappor $L_{max} = 8$. Dans ce cas, la surface aurait été composée à 94,5 % par les mémoires et à 5 % par la matrice de calcul.

Avec le recul, cette dominance mémoire peut amener à poser la question du choix de la limitation des ressources arithmétiques. Cette limitation a été obtenue par le choix d'une architecture à mémoire. Si on tient compte de la taille de TFR de 8192 points, du parallélisme nécessaire pour le haut débit, et des ressources supplémentaires dûes au filtrage IOTA, le choix d'une architecture en pipeline aurait nécessité un nombre de modules de calcul plusieurs fois supérieur à celui de l'architecture proposée et aurait donc engendré une surface pour les ressources arithmétiques beaucoup plus importante. Notre choix d'une architecture à mémoires est d'autant plus justifié si on tient compte des différentes configurations de l'architecture selon les paramètres N et L. En effet, les mémoires composant notre architecture sont activées selon ces deux paramètres. Par conséquent, en terme de consommation de puissance, la contribution des mémoires à la consommation totale dépend donc de N et L. À titre d'exemple, pour $N = 64$ pour le schéma OFDM/QAM, toutes les ressources mémoires pour le filtrage et la majorité des ressources mémoires pour la TFR sont désactivées. Pour cette configuration, la part de la consommation de puissance des ressources arithmétiques est prépondérante. L'architecture à base de mémoires permet donc une consommation limitée pour des valeurs de N et L plus petites.

Nous avons montré au chapitre 3.3 qu'une gestion efficace de la tension d'alimentation permettrait d'obtenir des gains en consommation importants. Toutefois, cette solution nécessite une couche supérieure de gestion de la tension [128–131]. Des travaux de recherche doivent être menés afin d'intégrer une telle couche dans l'architecture proposée. Cette technique permettra au modulateur proposé d'opérer à une tension et à une fréquence optimales pour les différents standards OFDM possibles selon les contraintes en débit imposées. L'ajustement en tension peut se faire de deux façons. La première consiste à ajuster la tension à des valeurs prédéfinies. Dans ce cas de figure, l'architecture nécessite d'être caractérisée en tension et en fréquence pour chaque norme considérée. Ceci rend plus difficile l'évolution de l'architecture pour de futures normes. La deuxième approche permet de se dissocier de cette contrainte en intégrant un système dynamique d'ajustement de la tension et de la fréquence. Une telle solution a été proposée pour une architecture OFDM/QAM à base de TFR en pipeline (SDF) radix-2^4 [86]. Elle est basée sur une technique d'ajustement en boucle fermée. Selon le mode d'opération, le gain de consommation pour cette architecture est de 18% à 43% par rapport à une architecture sans gestion de la tension et de la fréquence. La même étude devrait être appliquée à notre architecture.

La contrainte haute de couverture du cas de l'UWB nous a permis de déterminer les limites en termes de débit qu'un modulateur OFDM avancé multi-standard pourrait atteindre. Ces limites et ces performances ont été obtenues pour une version de l'architecture utilisant un degré de parallélisme des données, $P_D = 8$ et un algorithme *mixed* radix-$2/2^2/2^3$. L'architecture est aussi dictée par différents degrés de parallélisme des papillons et par un parallélisme des TFR pour le mode MIMO. Il serait intéressant d'étudier d'autres valeurs des différents degrés de parallélisme

afin de comparer les performances des différentes versions de l'architecture. Il en est de même de l'approche du traitement en parallèle du mode MIMO comparée à une approche séquentielle.

La meilleure façon de réaliser ces études de comparaison est de développer un environnement complet d'aide à la conception. En effet, nous avons présenté dans ce manuscrit toutes les équations permettant de décrire l'architecture, que ce soit pour la génération des adresses, pour le fonctionnement de la matrice ou encore le contrôle. Cette macro-génération devrait être orientée par les contraintes dictées par l'utilisateur de l'outil. Ainsi, l'outil recevrait en entrée la largeur de bande de la modulation désirée, le nombre de points puissance de 2 considéré ou encore la longueur de troncature. Le concepteur pourrait aussi fournir en option les degrés de parallélisme et le choix des algorithmes radix-2^i. À titre d'exemple, le choix du concepteur d'un parallélisme des données $P_D = 8$ et un algorithme *mixed* radix-$2/2^2$ (parallélisme de l'algorithme $P_{radix} = 4$) génèrerait une matrice à 8 modules de calcul avec un parallélisme des papillons $P_{p,radix} = 2$, soit la configuration de l'architecture proposée pour le mode MIMO 2x2. L'outil permettrait donc de paramétrer un modèle de base afin de générer une architecture finale.

Le développement d'un tel outil dans le cadre spécifique de la conception d'un modulateur permettrait un gain de temps pour la conception d'un modulateur OFDM avancé basé sur notre modèle d'architecture. Le concepteur aurait alors le choix entre plusieurs versions de l'architecture proposée lui permettant de déterminer celle répondant aux contraintes et aux performances désirées. Il serait d'ailleurs intéressant de comparer les performances de l'architecture proposée (et de ses alternatives éventuelles), avec une architecture obtenue par une approche de synthèse de haut niveau [132]. D'après les auteurs de cette approche, elle permet d'obtenir des bonnes performances en termes de temps de reconfiguration, de surface et de consommation de puissance pour des applications multimodes. Elle permettrait également d'explorer un espace plus grand d'architectures.

Un des verrous à la réalisation de la radio logicielle est la capacité à réaliser un front-end radio-fréquence (RF) flexible, permettant la transmission du signal sur une grande plage de fréquences et selon différentes interfaces air [133]. En particulier, les convertisseurs numérique/analogique et analogique/numérique, les transpositions en fréquence et les différents étages de filtrage imposent une forte contrainte sur la conception d'un front-end RF pour la radio-logicielle. Des études devraient être menées afin de déterminer les facultés de ce traitement numérique très large bande de la modulation OFDM/OQAM, associé à la grande pureté spectrale qui caractérise la fonction de mise en forme IOTA, à lever ces contraintes sur la synthèse du signal RF.

Annexes

ANNEXE A

CONDITIONS D'ORTHOGONALITÉS DE LA MODULATION OFDM/OQAM

Pour pouvoir récupérer le symbole OFDM/OQAM, le signal doit avoir une orthogonalité réelle en temps et en fréquence. Mathématiquement parlant, on doit avoir :

$$\left\langle s_{n,m}(t) \mid s_{n',m'}(t) \right\rangle = \Re \int_{-\infty}^{+\infty} s_{n,m}(t) s^*_{n,m}(t) = \begin{cases} 1 & \text{si } n = n' \text{ et } m = m \\ 0 & \text{ailleurs} \end{cases} \tag{A.1}$$

Avec $s(t) = e^{j(n+m)\frac{\pi}{2}} g\left(t - n\tau_0\right) e^{j2\pi m \upsilon_0 t}$, le signal OFDM/OQAM, $\upsilon_0 = \frac{1}{2\tau_0}$, et $g\left(t - n\tau_0\right)$ paire et réelle. Ici, τ_0 correspond à la période d'un symbole OFDM/OQAM.

Le terme $e^{j(n+m)\frac{\pi}{2}} (= i^{n+m})$ nous assure d'avoir une orthogonalité en temps et en fréquence pour toutes les porteuses prises deux à deux. En effet, soit :

$$\begin{aligned}
\left\langle s_{n,m}(t) \mid s_{n',m'}(t) \right\rangle &= \Re \int_{-\infty}^{+\infty} s_{n,m}(t) s^*_{n',m'}(t) \\
&= \Re \left[\int_{-\infty}^{+\infty} \left(e^{j(n+m)\frac{\pi}{2}} g\left(t - n\tau_0\right) e^{j2\pi m \upsilon_0 t} \right) \right. \\
&\quad \left. \cdot \left(e^{-j(n'+m')\frac{\pi}{2}} g\left(t - n'\tau_0\right) e^{-j2\pi m' \upsilon_0 t} \right) dt \right] \\
&= \Re \left[\int_{-\infty}^{+\infty} e^{j(n+m-n'-m')\frac{\pi}{2}} e^{j2\pi(m-m')\upsilon_0 t} g\left(t - n\tau_0\right) g\left(t - n'\tau_0\right) dt \right] \\
&= \Re \left[e^{j(n+m-n'-m')\frac{\pi}{2}} \int_{-\infty}^{+\infty} e^{j2\pi(m-m')\upsilon_0 t} g\left(t - n\tau_0 + (n'-n')\frac{\tau_0}{2}\right) \right. \\
&\quad \left. \cdot g\left(t - n'\tau_0 + (n-n)\frac{\tau_0}{2}\right) dt \right] \\
&= \Re \left[e^{j(n+m-n'-m')\frac{\pi}{2}} \int_{-\infty}^{+\infty} e^{j2\pi(m-m')\upsilon_0 t} \right. \\
&\quad \cdot g\left(t - (n+n')\frac{\tau_0}{2} - (n-n')\frac{\tau_0}{2}\right) \\
&\quad \left. \cdot g\left(t - (n+n')\frac{\tau_0}{2} + (n-n')\frac{\tau_0}{2}\right) dt \right]
\end{aligned}$$

On pose comme changement de variable : $t' = t - (n+n')\frac{\tau_0}{2}$ et $dt' = dt \Rightarrow t = t' + (n+n')\frac{\tau_0}{2}$.

$$\begin{aligned}
&= \Re \left[e^{j(n+m-n'-m')\frac{\pi}{2}} \int_{-\infty}^{+\infty} e^{j2\pi(m-m')\upsilon_0 \left(t' + (n+n')\frac{\tau_0}{2}\right)} g\left(t' - (n-n')\frac{\tau_0}{2}\right) \right. \\
&\quad \left. \cdot g\left(t' + (n-n')\frac{\tau_0}{2}\right) dt' \right] \\
&= \Re \left[e^{j(n+m-n'-m')\frac{\pi}{2}} e^{j2\pi(m-m')\upsilon_0 (n+n')\frac{\tau_0}{2}} \int_{-\infty}^{+\infty} g\left(t' - (n-n')\frac{\tau_0}{2}\right) \right. \\
&\quad \left. \cdot g\left(t' + (n-n')\frac{\tau_0}{2}\right) e^{j2\pi(m-m')\upsilon_0 t'} dt' \right]
\end{aligned}$$

Avec $\Re\left(e^{j\theta}\right) = \Re\left(e^{-j\theta}\right)$

$$\begin{aligned}
&= \Re\left[e^{j(n+m-n'-m')\frac{\pi}{2}} e^{j2\pi(m-m')\upsilon_0(n+n')\frac{\tau_0}{2}} \int_{-\infty}^{+\infty} g\left(t' - (n-n')\frac{\tau_0}{2}\right)\right. \\
&\quad \left. \cdot\, g\left(t' + (n-n')\frac{\tau_0}{2}\right) e^{-j2\pi(m-m')\upsilon_0 t'} dt'\right] \\
&= \Re\left[e^{j(n+m-n'-m')\frac{\pi}{2}} e^{j\pi(m-m')(n+n')\upsilon_0\tau_0} A_{gg'}\left((n-n')\tau_0, (m-m')\upsilon_0\right)\right] \\
&= \Re\left[e^{j((n-n')+(m-m')+(n+n')(m-m'))\frac{\pi}{2}} A_{gg'}\left((n-n')\tau_0, (m-m')\upsilon_0\right)\right] \qquad \text{(A.2)}
\end{aligned}$$

avec $\upsilon_0 = \frac{1}{2\tau_0}$.
$A_{gg'}\left((n-n')\tau_0, (m-m')\upsilon_0\right)$ est la fonction d'ambiguïté qui est définie comme :

$$A_{gg'} = \int_{-\infty}^{+\infty} g_{n,m}\left(t + \frac{\tau_0}{2}\right) g^*_{n',m'}\left(t - \frac{\tau_0}{2}\right) e^{-j2\pi\upsilon_0 t} \qquad \text{(A.3)}$$

La fonction d'ambiguïté détermine pour quels couples $(\tau_0, \upsilon_0)_{n,n',m,m'}$ on obtient une valeur nulle de l'intégrale d'orthogonalité en fournissant une mesure du degré de ressemblance entre le signal analysé et ses diverses translatées en temps et en fréquence.

Dans le cas de la fonction IOTA, on obtient une valeur nulle de l'intégrale A.3 seulement pour les porteuses séparées en temps ou en fréquence d'un multiple de 2 (sauf pour n=m=0) (voir figure A.1), soit $A_{gg'}(2n\tau_0, 2m\upsilon_0)$, c'est-à-dire une orthogonalité complexe telle la modulation OFDM/OQAM que sur 4 sous réseaux.

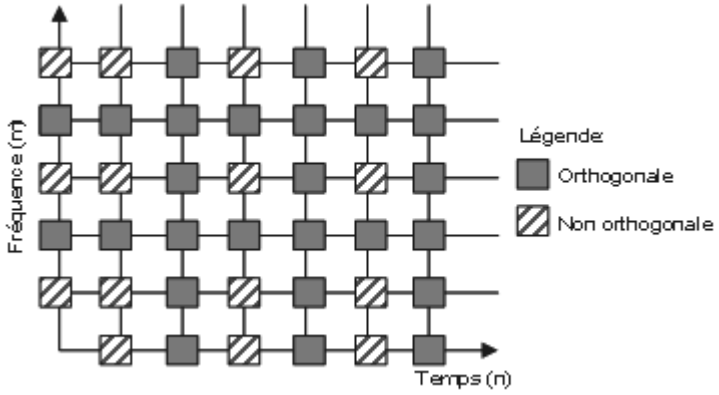

FIGURE A.1 – Orthogonalité de la modulation OFDM/OQAM due au terme $A_{gg}\left((n-n')\tau_0, (m-m')\upsilon_0\right)$

Dans le cas où le terme m aurait été négligé dans $e^{j(n+m)\frac{\pi}{2}}$, c'est-à-dire $e^{jn\frac{\pi}{2}}$, on aurait obtenu pour l'équation A.2 :

$$\left\langle s_{n,m}(t) \mid s_{n',m'}(t)\right\rangle = \Re\left[e^{j((n-n')+(n+n')(m-m'))\frac{\pi}{2}} A_{gg}\left((n-n')\tau_0, (m-m')\upsilon_0\right)\right] \qquad \text{(A.4)}$$

On remarque sur la figure A.2 que pour des valeurs de $(m-m')$ impair et $(n-n')$ impair ou égale à 0, l'orthogonalité n'aurait pas été obtenue.

Finalement, on constate sur la figure A.3 que l'expression $e^{j((n-n')+(m-m')+(n+n')(m-m'))\frac{\pi}{2}}$ $\cdot A_{gg}\left((n-n')\tau_0, (m-m')\upsilon_0\right)$ nous assure une orthogonalité réelle temps-fréquence complète.

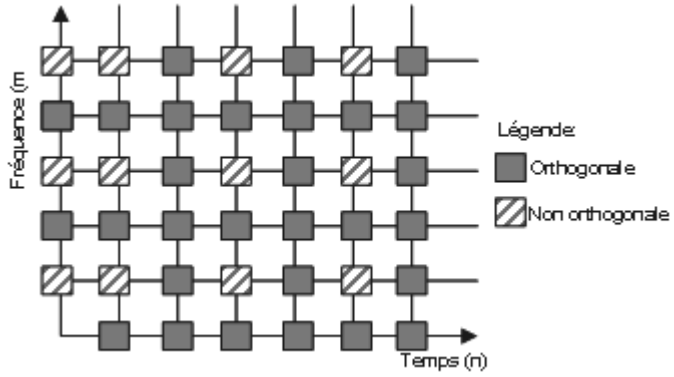

FIGURE A.2 – Orthogonalité de la modulation OFDM/OQAM due au terme $e^{j((n-n')+(m-m')(n+n'))\frac{\pi}{2}}\ A_{gg}\left((n-n')\,\tau_0,(m-m')\,\upsilon_0\right)$

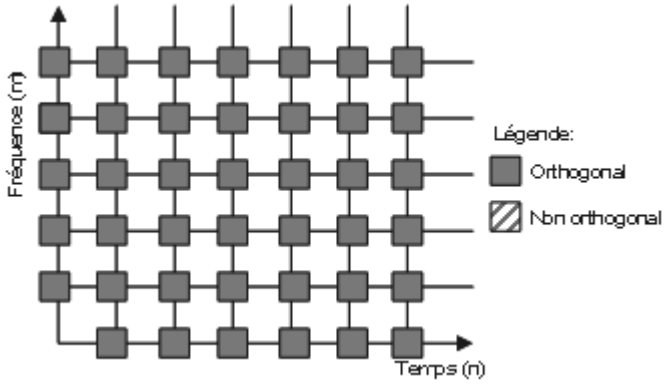

FIGURE A.3 – Orthogonalité de la modulation OFDM/OQAM due au terme $e^{j((n-n')+(m-m')+(m-m')(n+n'))\frac{\pi}{2}}\ A_{gg}\left((n-n')\,\tau_0,(m-m')\,\upsilon_0\right)$

Annexe B

Algorithmes de la TFR

Soit l'équation de la TFD :

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \tag{B.1}$$

et son inverse :

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk} \tag{B.2}$$

avec les facteurs de rotations $W_N = e^{-j\frac{2\pi}{N}}$, $0 \leq n \leq N-1$, $0 \leq k \leq K-1$.

Les vecteurs x(n), représentant les échantillons temporels, et X(k), sa transformée dans le domaine fréquentiel, sont unidimensionnels, respectivement de taille $N \times 1$ et $K \times 1$ avec $K = N$. L'idée de l'approche *divide-and-conquer* à la base des algorithmes de la TFR, consiste à représenter ces vecteurs sur 2 dimensions (cas du radix-2, radix-4, radix-8, etc.) ou plus (cas du radix-2^i). Pour chaque algorithme, il existe deux versions duales et de même complexité, à savoir l'entrelacement en fréquence (*Decimation-In-Frequency DIF*) et l'entrelacement en temps (*Decimation-In-Time DIT*). Dans la version DIF, les échantillons d'entrées $x(n)$ sont lus dans l'ordre naturel et les échantillons obtenus $X(k)$ dans l'ordre *bit reversed*. Dans la version DIT, l'ordre s'inverse, c'est-à-dire dans l'ordre *bit reversed* pour $x(n)$ et naturel pour $X(k)$. Dans cette étude, nous considérons la version DIF (l'étude peut facilement être étendue à la version DIT).

B.0.1 Ré-indexation sur deux dimensions

Soit $N = N_1 \times N_2$ (et $K = K_1 \times K_2$), deux nombres ayant un facteur commun. La fonction linéaire réalisant la ré-indexation est alors :

$$\begin{aligned} \text{en temps : } n &= N_2 n_1 + n_2 \\ \text{en fréquence : } k &= k_1 + K_1 k_2 \end{aligned} \tag{B.3}$$

avec $0 \leq n_1 \leq N_1 - 1$, $0 \leq n_2 \leq N_2 - 1$, $0 \leq k_1 \leq K_1 - 1$, $0 \leq k_2 \leq K_2 - 1$.

En introduisant l'équation B.3 dans la formule de la TFD directe B.1, on obtient :

$$X(k_1 + K_1 k_2) = \underbrace{\sum_{n_2=0}^{N_2-1} \underbrace{\left[W_N^{n_2 k_1} \underbrace{\left[\underbrace{\sum_{n_1=0}^{N_1-1} x(n_1, n_2) W_{N_1}^{n_1 k_1}}_{TFD \text{ à } N_1 \text{ points}} \right]}_{\text{rotation}} \right]}_{TFD \text{ à } N_2 \text{ points}} W_{N_2}^{n_2 k_2}} \tag{B.4}$$

La formule B.4 illustre le fonctionnement de la TFR, soit la décomposition de la TFR à N points en N_2 TFR à N_1 points.

Algorithme Radix-2 entrelacement en fréquence (Decimation-In-frequency, DIF)

Soit $N = r^p = 2^p$, pour N puissance de 2 avec p un entier positif, utilisant un algorithme radix-2 [76]. En évaluant pour n_1 l'équation B.4 avec $N_1 = 2$ et $N_2 = N/2$, on obtient alors :

$$X(k_1 + 2k_2) = \sum_{n_2=0}^{\frac{N}{2}-1} \left[x(n_2) + (-1)^{k_1} x(n_2 + N/2) W_N^{n_2 k_1} \right] W_{N/2}^{n_2 k_2} \tag{B.5}$$

On obtient ainsi une décomposition des N points en deux ensembles, soit les $N/2$ premiers points et les $N/2$ derniers points. La figure B.1 illustre le module papillon radix-2 DIF.

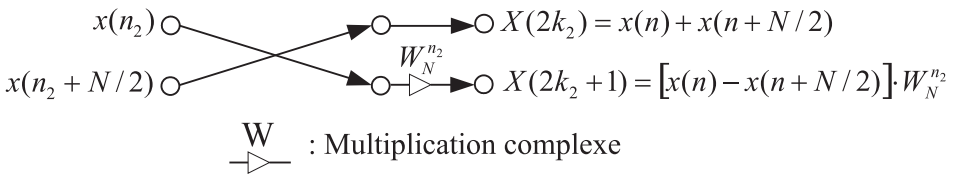

FIGURE B.1 – Module papillon du radix-2 DIF

Radix-4 Entrelacement en fréquence (Decimation-In-frequency, DIF)

Soit $N = r^p = 4^p$, pour N puissance de 4 avec p un entier positif, utilisant un algorithme radix-4 [76]. En évaluant pour n_1 l'équation B.4 avec $N_1 = 4$ et $N_2 = N/4$, on obtient alors :

$$\begin{aligned} X(k_1 + 4k_2) &= \sum_{n_2=0}^{\frac{N}{4}-1} \Big\{ \Big[x(n_2) + (-j)^{k_1} x(n_2 + N/4) + (-1)^{k_1} x(n_2 + N/2) \\ &+ W_4^{3k_1} x(n_2 + 3N/4) \Big] W_N^{n_2 k_1} \Big\} W_{N/4}^{n_2 k_2} \end{aligned} \tag{B.6}$$

On obtient ainsi une décomposition des N points en quatre séquences de $N/4$ points : $x(n)$, $x(n + N/4)$, $x(n + N/2)$, $x(n + 3N/4)$, avec $0 \leq n \leq N/4 - 1$. La figure B.2 illustre le module papillon radix-4 DIF.

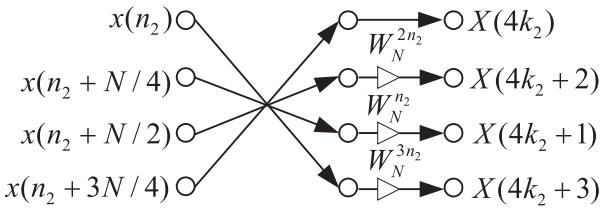

FIGURE B.2 – Module papillon du radix-4 DIF

Algorithme Split-Radix

L'algorithme split-radix [72, 76] exploite les propriétés des algorithmes de plus hauts degrés pour diminuer le nombre d'opérations arithmétiques. En observant l'algorithme radix-2 DIF B.5, on constate que les points pairs peuvent être traités indépendamment des points impairs.

$$X(2k_2) = \sum_{n_2=0}^{N/2-1} [x(n) + x(n_2 + N/2)]\, W_{N/2}^{k_2 n_2} \tag{B.7}$$

$$X(2k_2+1) = \sum_{n_2=0}^{N/2-1} \left\{[x(n_2) - x(n_2 + N/2)]\, W_N^{n_2}\right\} W_{N/2}^{k_2 n_2} \tag{B.8}$$

Avec $n_2 = k_2 = 0, 1, \cdots, \frac{N}{2} - 1$.

L'idée est d'utiliser l'algorithme radix-2 pour réaliser la partie paire de la TFR, et l'algorithme radix-4 pour la partie impaire, qui nécessite des multiplications supplémentaires par les facteurs de rotation $W_N^{n_2}$. L'algorithme split-radix-$2/4$ divise une TFR de N points en une TFR à $N/2$ points et 2 TFRs à $N/4$ points à chaque étage. L'expression de la partie impaire de la TFR radix-4 DIF est alors :

$$\begin{aligned} X(4k_3+1) &= \sum_{n_3=0}^{N/4-1} \{[x(n_3) - x(n_3 + N/2)] \\ &- j\,[x(n_3 + N/4) - x(n_3 + 3N/4)]\}\, W_N^{n_3} W_{N/4}^{k_3 n_3} \end{aligned} \tag{B.9}$$

$$\begin{aligned} X(4k_3+3) &= \sum_{n_3=0}^{N/4-1} \{[x(n_3) - x(n_3 + N/2)] \\ &+ j\,[x(n_3 + N/4) - x(n_3 + 3N/4)]\}\, W_N^{3n_3} W_{N/4}^{k_3 n_3} \end{aligned} \tag{B.10}$$

Avec $n_3 = k_3 = 0, 1, \cdots, \frac{N}{4} - 1$.

Ceci donne une allure de "L" au module papillon, du fait que la partie impaire nécessite 2 étages pour décomposer la TFR. La figure B.3 illustre le module papillon pour le split-radix.

B.0.2 Ré-indexation sur *i+1* dimensions

Les algorithmes radix-2^i [134–136] réindexe le vecteur de la TFR à N points sur $i+1$ dimensions. Ces algorithmes ont l'avantage d'avoir une grande régularité et flexibilité architecturale comparable à l'algorithme radix-2 tout en diminuant le nombre de multiplications réelles nécessaires au calcul d'une TFR. Ainsi, on peut obtenir la même complexité arithmétique que le radix-4, ou même mieux (complexité presque comparable à l'algorithme split-radix) selon la valeur de i.

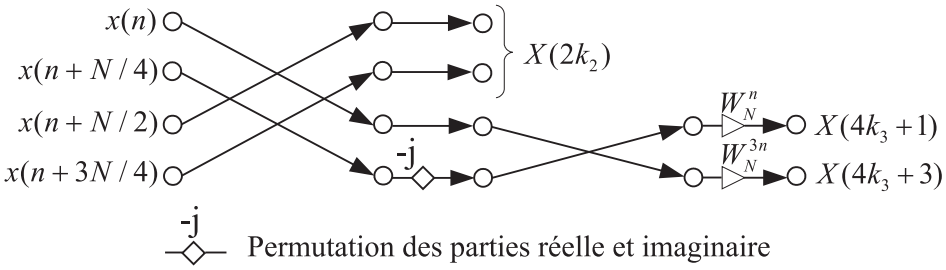

FIGURE B.3 – Module papillon du split-radix DIF

Algorithme Radix-2^2

Soit $N = r^p = 4^p$, pour N puissance de 4 avec p un entier positif, utilisant un algorithme radix-2^2. Considérant le développement de l'algorithme radix-2 DIF en appliquant cette fois-ci une indexation à 3 dimensions. La fonction linéaire réalisant cette transformation est :

$$\begin{aligned} \text{en temps} : n &= \frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3 \\ \text{en fréquence} : k &= k_1 + 2k_2 + 4k_3 \end{aligned} \tag{B.11}$$

avec $\{n_1, n_2 = 0, 1, n_3 = 0, \cdots, \frac{N}{4} - 1\}$ et $\{k_1, k_2 = 0, 1, k_3 = 0, \cdots, \frac{N}{4} - 1\}$. En intégrant B.11 dans l'équation de la TFD B.1 et en évaluant pour n_1, on obtient :

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{\frac{N}{4}-1} \sum_{n_2=0}^{1} B_{N/2}^{k_1}\left(\frac{N}{4}n_2 + n_3\right) W_N^{(\frac{N}{4}n_2+n_3)(k_1+2k_2+4k_3)} \tag{B.12}$$

Où

$$B_{N/2}^{k_1}\left(\frac{N}{4}n_2 + n_3\right) = x\left(\frac{N}{4}n_2 + n_3\right) + (-1)^{k_1} x\left(\frac{N}{4}n_2 + n_3 + \frac{N}{2}\right) \tag{B.13}$$

$B_{N/2}^{k_1}\left(\frac{N}{4}n_2 + n_3\right)$ de l'équation B.13 représente le premier croisillon (un étage) qui consiste seulement en des additions/soustractions complexes. En développant l'équation B.12 pour n_2, et en simplifiant l'expression du facteur de rotation on obtient :

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{\frac{N}{4}-1} \left\{ H_{N/4}^{k_1 k_2}(n_3)\, W_N^{n_3(k_1+2k_2)} \right\} W_{N/4}^{n_3 k_3} \tag{B.14}$$

Où

$$H_{N/4}^{k_1 k_2}(n_3) = B_{N/2}^{k_1}(n_3) + (-1)^{k_2}(-j)^{k_1} B_{N/2}^{k_1}\left(n_3 + \frac{N}{4}\right) \tag{B.15}$$

$H_{N/4}^{k_1 k_2}(n_3)$ représente le deuxième croisillon (un deuxième étage) qui consiste lui aussi en des multiplications complexes triviales par 1 ou j. Les échantillons ainsi obtenus subissent une rotation dans le plan complexe de l'ordre de $W_N^{n_3(k_1+2k_2)}$. Comme on le constate à l'équation B.14, on obtient après décompositions sur 2 étages, quatre TFRs à $N/4$ points. La figure B.4 illustre le module papillon pour le radix-2^2.

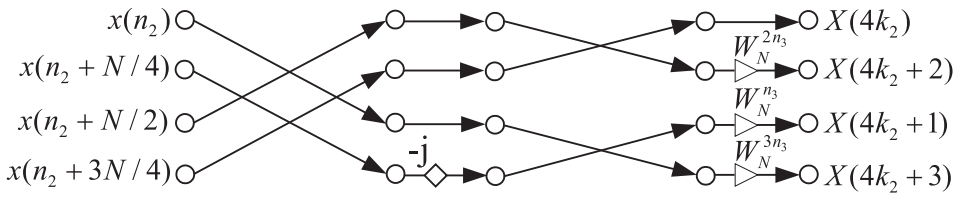

FIGURE B.4 – Module papillon du radix-2^2 DIF

Algorithme Radix-2^3

Soit $N = r^p = 8^p$, pour N puissance de 8 avec p un entier positif, utilisant un algorithme radix-2^3. En procédant de la même manière que le radix-2^2, mais en indexant sur 4 dimensions, on obtient :

$$\begin{aligned} \text{en temps : } n &= \frac{N}{2}n_1 + \frac{N}{4}n_2 + \frac{N}{8}n_3 + n_4 \\ \text{en fréquence : } k &= k_1 + 2k_2 + 4k_3 + 8k_4 \end{aligned} \tag{B.16}$$

avec $\{n_1, n_2, n_3 = 0, 1, n_4 = 0, \cdots, \frac{N}{8} - 1\}$ et $\{k_1, k_2, k_3 = 0, 1, k_4 = 0, \cdots, \frac{N}{8} - 1\}$. En intégrant B.16 dans l'équation de la TFD B.1 et en considérant les indices n_1, n_2 et n_3, on obtient :

$$X(k_1 + 2k_2 + 4k_3 + 8k_4) = \sum_{n_4=0}^{\frac{N}{8}-1} \{T_{N/8}^{k_1k_2k_3}(n_4)\, W_N^{n_4(k_1+2k_2+4k_3)}\} W_{N/8}^{n_4k_4} \tag{B.17}$$

Avec,

$$T_{N/8}^{k_1k_2k_3}(n_4) = H_{N/4}^{k_1k_2}(n_4) + (-1)^{k_3} W_8^{(k_1+2k_2)} H_{N/4}^{k_1k_2}\left(n_4 + \frac{N}{8}\right) \tag{B.18}$$

$T_{N/8}^{k_1k_2k_3}(n_4)$ représente le troisième croisillon (troisième étage), les deux premiers étant $B_{N/2}^{k_1}\left(\frac{N}{4}n_2 + n_3\right)$ et $H_{N/4}^{k_1k_2}(n_3)$. Une rotation est opérée sur les échantillons de rotation $W_N^{n_4(k_1+2k_2+4k_3)}$. On obtient ainsi un ensemble de 8 TFRs à $N/8$ points après chaque décomposition sur 3 étages d'une TFR à N points. La figure B.5 illustre le module papillon pour le radix-2^3.

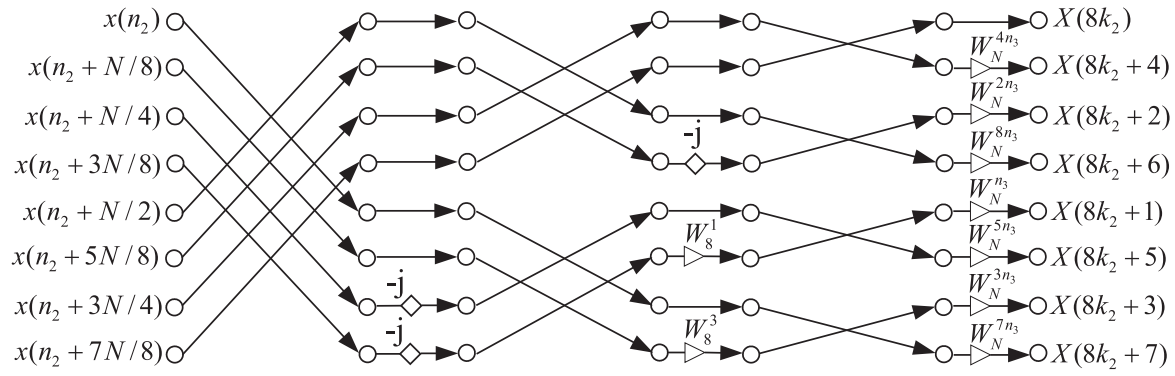

FIGURE B.5 – Module papillon du radix-2^3 DIF

Cette décomposition peut continuer pour des valeurs de i plus grande que 3 selon la même démarche. De même, il est possible de cascader sur i étages des algorithmes d'ordre supérieur tels que le radix-4^2 [92] ou le radix-4^3 [137].

Annexe C

Expression de RSBQ pour les algorithmes de TFR radix-2^i

C.1 Cas du radix-2

Pour une TFR radix-2 à 2^m points, nous avons $\frac{N}{2^{\kappa+1}}$ erreurs de troncature complexes pour un étage κ, qui se propagent sur $(m-1-\kappa)$ étages pour chaque sortie $X(k)$. Nous obtenons alors :

$$\begin{aligned}
\sigma_T^2 &= \sigma_t^2 \left[\frac{N}{2} \left(\frac{1}{4}\right)^{m-1} + \frac{N}{2^2} \left(\frac{1}{4}\right)^{m-2} + \ldots + \frac{N}{2^m} \left(\frac{1}{4}\right)^{m-m} \right] \\
&= \sigma_t^2 N \sum_{y=0}^{m-1} \left(\frac{1}{2}\right)^{y+1} \left(\frac{1}{4}\right)^{m-1-y} \\
&= \sigma_t^2 N \sum_{y=0}^{m-1} \left(\frac{1}{2}\right)^{2m-1-y} \\
&= \sigma_t^2 N \left(\frac{1}{2}\right)^{2m-1} \left[\sum_{y=0}^{m-1} 2^y \right] \\
&= \sigma_t^2 N \left(\frac{2}{N^2}\right) \left[\frac{1-2^m}{1-2} \right] \\
&= 2\sigma_t^2 \left[1 - \frac{1}{N} \right] \\
&= \frac{2^{-2b}}{2} \left[1 - \frac{1}{N} \right] \qquad \text{(C.1)}
\end{aligned}$$

Dans un premier temps, nous considérons que toutes les multiplications complexes par les facteurs de rotation W génèrent un bruit de quantification due à l'erreur d'arrondi. Pour une TFR radix-2 à 2^m points, nous avons $\frac{N}{2}$ erreurs d'arrondi complexes par étage κ, et chaque erreur se propage à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$. De plus, chaque erreur subit $(m-1-\kappa)$ recadrages. Le bruit total dû aux multiplications par les facteurs W à chaque sortie est alors :

$$\begin{aligned}
\sigma_W^2 &= \sigma_a^2 \frac{N}{2} \frac{1}{N} \left[\frac{N}{2} \left(\frac{1}{4}\right)^{m-1} + \frac{N}{2^2} \left(\frac{1}{4}\right)^{m-2} + \ldots + \frac{N}{2^m} \left(\frac{1}{4}\right)^{m-m} \right] \\
&= \sigma_a^2 \frac{N}{2} \left[\sum_{y=0}^{m-1} \left(\frac{1}{2}\right)^{y+1} \left(\frac{1}{4}\right)^{m-1-y} \right]
\end{aligned}$$

$$\begin{aligned}\sigma_W^2 &= \sigma_a^2 \frac{N}{2}\frac{2}{N^2}\left(\frac{1-2^m}{1-2}\right) \\ &= \sigma_a^2\left[1-\frac{1}{N}\right] \\ &= \frac{2^{-2b}}{3}\left[1-\frac{1}{N}\right] \end{aligned} \tag{C.2}$$

Il y a $2^{\kappa+1}$ multiplications triviales par étage κ pour les $m-1$ premiers étages qui se propagent à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$. Le dernier étage a $\frac{N}{2}$ multiplications triviales par 1. Nous obtenons alors :

$$\begin{aligned}\sigma_{W_{tr}}^2 &= \frac{\sigma_a^2}{N}\left[2\frac{N}{2}\left(\frac{1}{4}\right)^{m-1} + 2^2\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \ldots + 2^{m-1}\frac{N}{2^{m-1}}\left(\frac{1}{4}\right) + \frac{N}{2}\frac{N}{2^m}\right] \\ &= \sigma_a^2\left[\sum_{y=0}^{m-2}\left(\frac{1}{4}\right)^{m-1-n} + \frac{1}{2}\right] \\ &= \sigma_a^2\left[\left(\frac{1}{4}\right)^{m-1}\left(\sum_{y=0}^{m-2}4^n\right) + \frac{1}{2}\right] \\ &= \sigma_a^2\left[\frac{4}{N^2}\left(\frac{1-4^{m-1}}{1-4}\right) + \frac{1}{2}\right] \\ &= \sigma_a^2\left[\frac{4}{N^2}\left(\frac{\frac{N^2}{4}-1}{3}\right) + \frac{1}{2}\right] \\ &= \sigma_a^2\left[\frac{1}{3} - \frac{4}{3N^2} + \frac{1}{2}\right] \\ &= \sigma_a^2\left[\frac{5}{6} - \frac{4}{3N^2}\right] = \frac{2^{-2b}}{3}\left[\frac{5}{6} - \frac{4}{3N^2}\right] \end{aligned} \tag{C.3}$$

La variance de l'erreur de sortie globale pour l'algorithme radix-2 DIF est alors égale à :

$$\sigma_{E_{R2DIF}}^2 = \sigma_T^2 + \sigma_W^2 - \sigma_{W_{tr}}^2 = 2^{-2b}\left[\frac{5}{9} - \frac{5}{6N} + \frac{4}{3N^2}\right] \tag{C.4}$$

Le RSBQ pour l'algorithme radix-2 DIF est alors :

$$RSBQ_{R2DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b}\left[\frac{5N}{3} - \frac{5}{2} + \frac{4}{3N}\right] \tag{C.5}$$

Avec la moyenne quadratique de $X(k)$ égale à [113] :

$$E\left[|X(k)|^2\right] = \sum_{k=0}^{N-1} E\left[|x(n)|^2\right]\left|W^{kn}\right|^2 = \frac{1}{3N} \tag{C.6}$$

C.2 Cas du radix-2^2

Pour l'algorithme radix-2^2 à 2^m points (m étant un multiple de 2), tous les étages κ pair sont des multiplications triviales par ± 1 ou $\pm j$. En ne considérant que les étages impairs, sachant qu'il y a $\frac{3N}{4}$ multiplications complexes par étage, on obtient :

$$\begin{aligned}
\sigma_W^2 &= \sigma_a^2 \frac{3N}{4}\frac{1}{N}\left[\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \frac{N}{2^4}\left(\frac{1}{4}\right)^{m-4} + \frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{N}{2^m}\left(\frac{1}{4}\right)^{m-m}\right] \\
&= \sigma_a^2 \frac{3N}{4}\left[\sum_{y=0}^{\frac{m}{2}-1}\left(\frac{1}{2}\right)^{2y+2}\left(\frac{1}{4}\right)^{m-2-2y}\right] \\
&= \sigma_a^2 \frac{3N}{4}\left(\frac{1}{2}\right)^{2m-2}\left[\sum_{y=0}^{\frac{m}{2}-1} 2^{2y}\right] \\
&= \sigma_a^2 \frac{3N}{4}\left(\frac{4}{N^2}\right)\left(\frac{1-4^{\frac{m}{2}}}{1-4}\right) \\
&= \sigma_a^2 \frac{3}{N}\left[\frac{N-1}{3}\right] \\
&= \sigma_a^2\left[1-\frac{1}{N}\right] \\
&= \frac{2^{-2b}}{3}\left[1-\frac{1}{N}\right] \qquad \text{(C.7)}
\end{aligned}$$

Or, pour chaque étage κ impair, nous avons $4^{\kappa'}$ (avec $\kappa' = \frac{\kappa+1}{2}$) multiplications triviales et $\frac{3N}{4}$ au dernier étage. L'erreur de quantification en moins dues aux multiplications complexes triviales est égale à :

$$\begin{aligned}
\sigma_{W_{tr}}^2 &= \frac{\sigma_a^2}{N}\left[4\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + 4^2\frac{N}{2^4}\left(\frac{1}{4}\right)^{m-4} + 4^3\frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{3N}{4}\right] \\
&= \sigma_a^2\left[\sum_{y=0}^{\frac{m}{2}-2} 4^{y+1}\left(\frac{1}{2}\right)^{2y+2}\left(\frac{1}{4}\right)^{m-2-2y} + \frac{3}{4}\right] \\
&= \sigma_a^2\left[\left(\frac{1}{4}\right)^{m-2}\left(\sum_{y=0}^{\frac{m}{2}-2} 16^y\right) + \frac{3}{4}\right] \\
&= \sigma_a^2\left[\frac{16}{N^2}\left(\frac{1-16^{\frac{m}{2}-1}}{1-16}\right) + \frac{3}{4}\right] \\
&= \sigma_a^2\left[\frac{16}{15N^2}\left(\frac{N^2}{16}-1\right) + \frac{3}{4}\right] \\
&= \sigma_a^2\left[\frac{1}{15} - \frac{16}{15N^2} + \frac{3}{4}\right] \\
&= \sigma_a^2\left[\frac{49}{60} - \frac{16}{15N^2}\right] \\
&= \frac{2^{-2b}}{3}\left[\frac{49}{60} - \frac{16}{15N^2}\right] \qquad \text{(C.8)}
\end{aligned}$$

Par la relation C.4, et avec σ_T^2 égale à l'équation C.1, on obtient :

$$\sigma_{E_{R2^2DIF}}^2 = 2^{-2b}\left[\frac{101}{180} - \frac{5}{6N} + \frac{16}{45N^2}\right] \qquad \text{(C.9)}$$

Finalement, l'expression du RSBQ pour l'algorithme radix-2^2 est égale à (avec la moyenne quadratique de $X(k)$ égale à l'expression C.6) :

$$RSBQ_{R2^2DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b}\left[\frac{101N}{60} - \frac{5}{2} + \frac{16}{15N}\right] \tag{C.10}$$

C.3 Cas du radix-2^3

Une décomposition radix-2^3 se réalise en trois étages. Le premier étage n'est composé que de multiplications triviales. Les deuxième et troisième étages possèdent $\frac{N}{4}$ et $\frac{7N}{8}$ multiplications complexes respectivement. Chaque erreur d'arrondi se propage à $\frac{N}{2^{\kappa+1}}$ sorties $X(k)$ et traverse $m-2-\kappa$ étages. L'expression de l'erreur d'arrondi due aux multiplications complexes par les facteurs W pour une TFR à 2^m points (m étant un multiple de 3) est égale alors à :

$$\begin{aligned}
\sigma_W^2 &= \sigma_a^2\frac{N}{4}\frac{1}{N}\left[\frac{N}{2^2}\left(\frac{1}{4}\right)^{m-2} + \frac{N}{2^5}\left(\frac{1}{4}\right)^{m-5} + \ldots + \frac{N}{2^{m-1}}\left(\frac{1}{4}\right)^{m-(m-1)}\right] \\
&\quad +\sigma_a^2\frac{7N}{8}\frac{1}{N}\left[\frac{N}{2^3}\left(\frac{1}{4}\right)^{m-3} + \frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{N}{2^m}\left(\frac{1}{4}\right)^{m-m}\right] \\
&= \sigma_a^2\frac{N}{4}\left[\sum_{y=0}^{\frac{m}{3}-1}\left(\frac{1}{2}\right)^{3y+2}\left(\frac{1}{4}\right)^{m-2-3y}\right] + \sigma_a^2\frac{7N}{8}\left[\sum_{y=0}^{\frac{m}{3}-1}\left(\frac{1}{2}\right)^{3y+3}\left(\frac{1}{4}\right)^{m-3-3y}\right] \\
&= \sigma_a^2\frac{N}{4}\left(\frac{1}{2}\right)^{2m-2}\left[\sum_{y=0}^{\frac{m}{3}-1}2^{3y}\right] + \sigma_a^2\frac{7N}{8}\left(\frac{1}{2}\right)^{2m-3}\left[\sum_{y=0}^{\frac{m}{3}-1}2^{3y}\right] \\
&= \sigma_a^2\frac{N}{4}\left(\frac{4}{N^2}\right)\left[\frac{N-1}{7}\right] + \sigma_a^2\frac{7N}{8}\left(\frac{8}{N^2}\right)\left[\frac{N-1}{7}\right] \\
&= \frac{\sigma_a^2}{7}\left[1-\frac{1}{N}\right] + \sigma_a^2\left[1-\frac{1}{N}\right] \\
&= \frac{8\sigma_a^2}{7}\left[1-\frac{1}{N}\right] \\
&= 2^{-2b}\frac{8}{21}\left[1-\frac{1}{N}\right]
\end{aligned} \tag{C.11}$$

Il existe $8^{\kappa-1}$ (avec $\kappa' = \frac{\kappa+1}{3}$) multiplications triviales pour κ correspondant au troisième étage de chaque décomposition radix-2^3. Les expressions C.3 et C.8 deviennent alors dans le cas du radix-2^3 :

$$\begin{aligned}
\sigma_{W_{tr}}^2 &= \frac{\sigma_a^2}{N}\left[8\frac{N}{2^3}\left(\frac{1}{4}\right)^{m-3} + 8^2\frac{N}{2^6}\left(\frac{1}{4}\right)^{m-6} + \ldots + \frac{7N}{8}\right] \\
&= \sigma_a^2\left[\sum_{y=0}^{\frac{m}{3}-2}\left(\frac{1}{4}\right)^{m-3-3y} + \frac{7}{8}\right] \\
&= \sigma_a^2\left[\left(\frac{1}{4}\right)^{m-3}\left(\sum_{y=0}^{\frac{m}{3}-2}4^{3y}\right) + \frac{7}{8}\right] \\
&= \sigma_a^2\left[\left(\frac{64}{N^2}\right)\left(\frac{1-4^{3(\frac{m}{3}-1)}}{1-4^3}\right) + \frac{7}{8}\right]
\end{aligned}$$

$$\begin{aligned}\sigma^2_{W_{tr}} &= \sigma^2_a \left[\left(\frac{64}{N^2} \right) \left(\frac{\frac{N^2}{64} - 1}{63} \right) + \frac{7}{8} \right] \\ &= \sigma^2_a \left[\frac{1}{63} - \frac{64}{63N^2} + \frac{7}{8} \right] \\ &= \sigma^2_a \left[\frac{449}{504} - \frac{64}{63N^2} \right] \\ &= \frac{2^{-2b}}{3} \left[\frac{449}{504} - \frac{64}{63N^2} \right] \end{aligned} \tag{C.12}$$

En remplaçant les expressions C.1, C.11 et C.12 dans C.4, on obtient :

$$\sigma^2_{E_{R2^3DIF}} = 2^{-2b} \left[\frac{37086}{63504} - \frac{37}{42N} + \frac{64}{189N^2} \right] \tag{C.13}$$

Et le RSBQ pour l'algorithme radix-2^3 est alors (avec la moyenne quadratique de $X(k)$ égale à l'expression C.6) :

$$RSBQ_{R2^3DIF} = \frac{E\left[|E(k)|^2\right]}{E\left[|X(k)|^2\right]} = 2^{-2b} \left[\frac{37086N}{21168} - \frac{37}{14} + \frac{64}{63N} \right] \tag{C.14}$$

Annexe D

Schéma des modules de calcul

Table D.1 – Différents cas de figure des modules de calcul selon les multiplications triviales ou non triviales

	cas A	cas B	cas C	cas D	cas E	cas F	cas G
BC1	1	1	1,$\Im$	1,$\Im$	1,W,$\Im$	1,$\Im$	1,W,$\Im$
BC2	1	-j	1,$\Im$	-j,$\Im$	-j,W,$\Im$	1,W,$\Im$	1,W,$\Im$

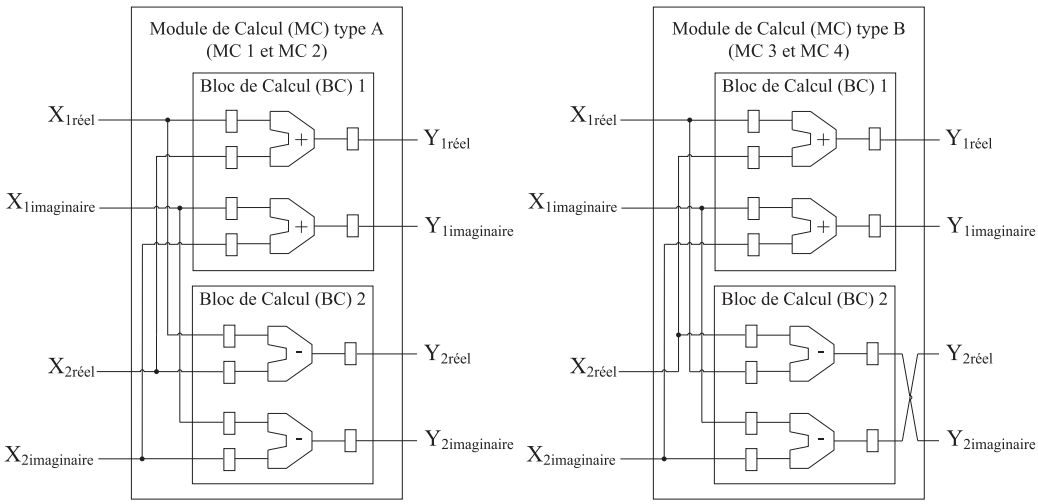

Figure D.1 – Architecture détaillée des modules de calcul de type A et B

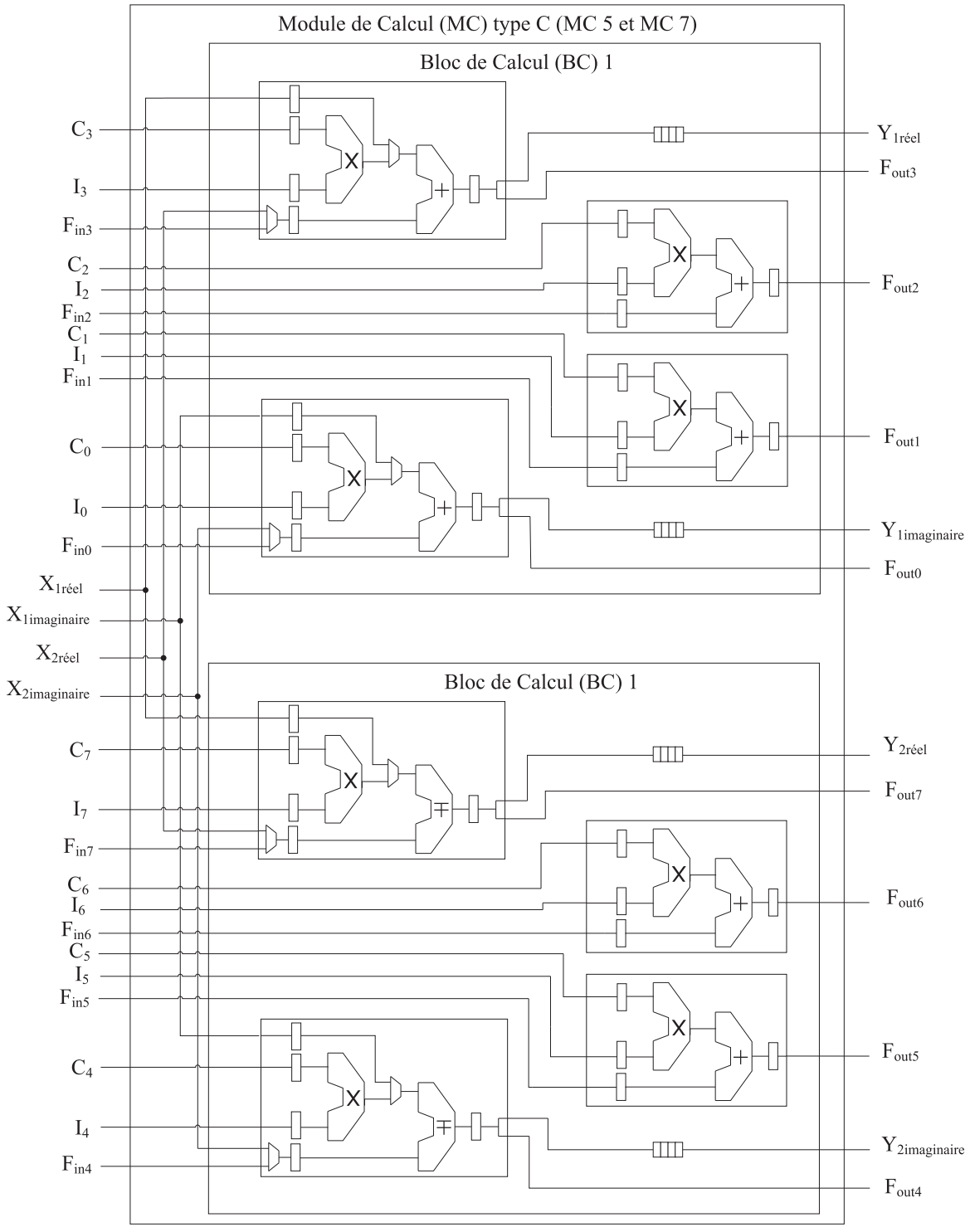

FIGURE D.2 – Architecture détaillée du module de calcul de type C

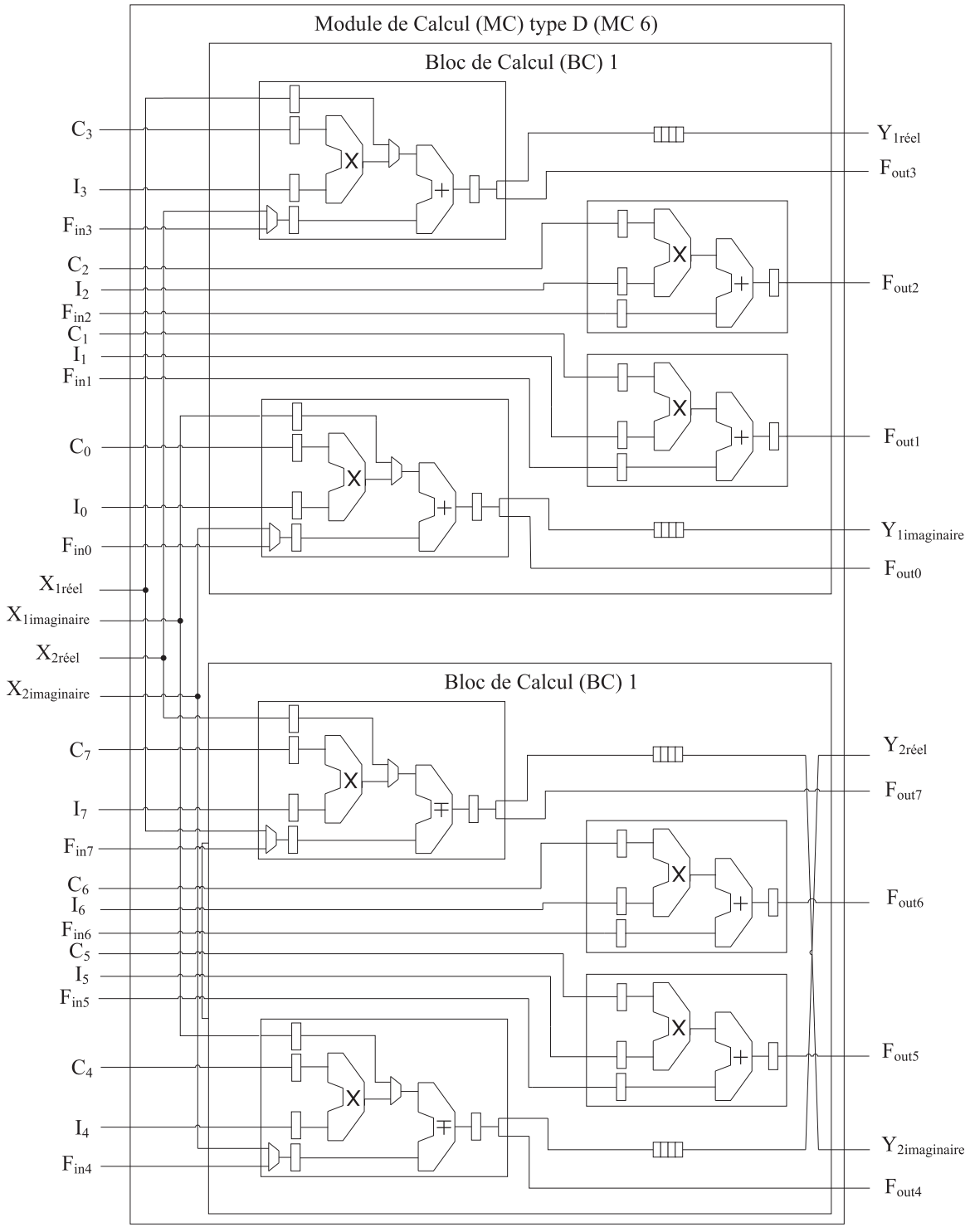

FIGURE D.3 – Architecture détaillée du module de calcul de type D

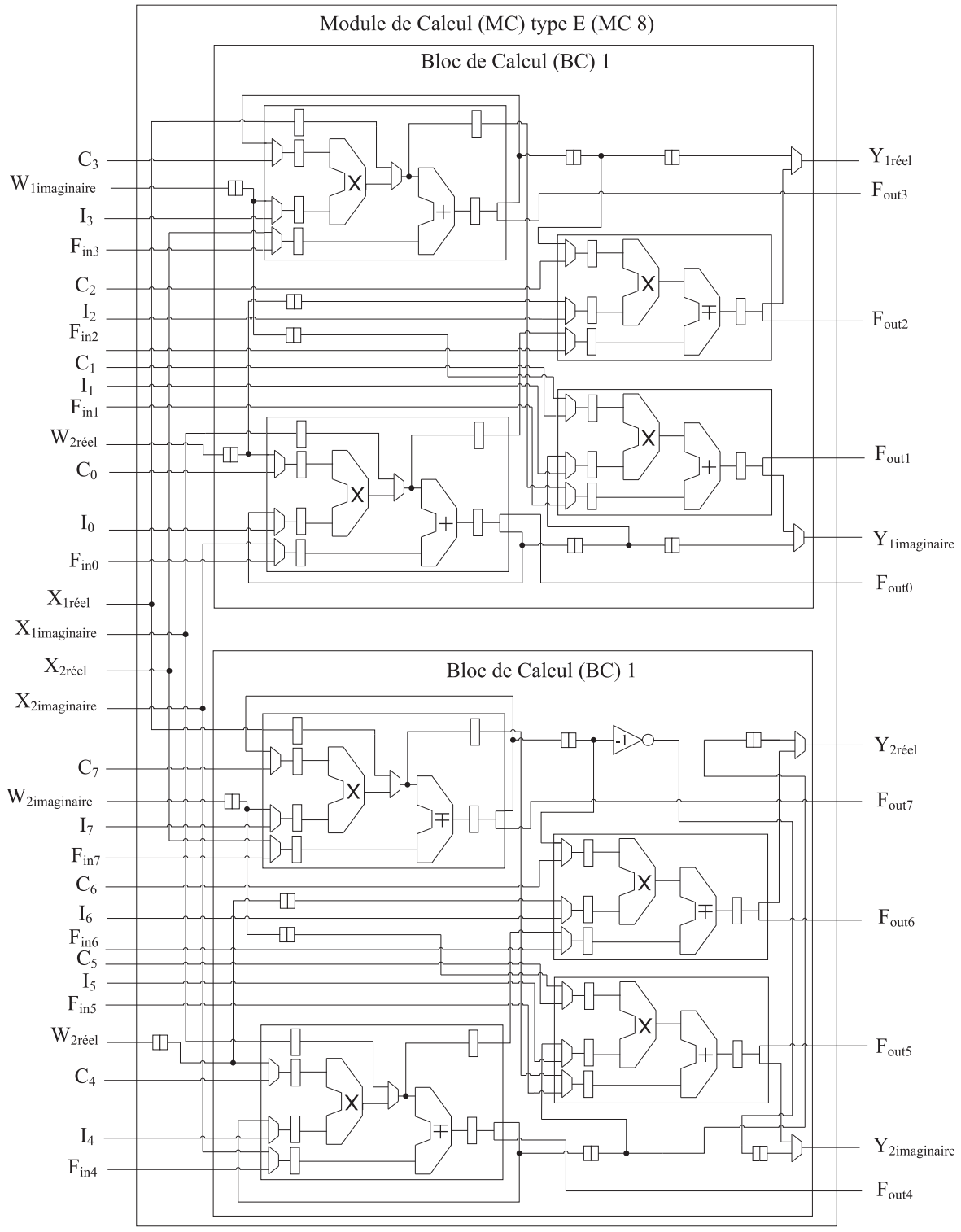

FIGURE D.4 – Architecture détaillée du module de calcul de type E

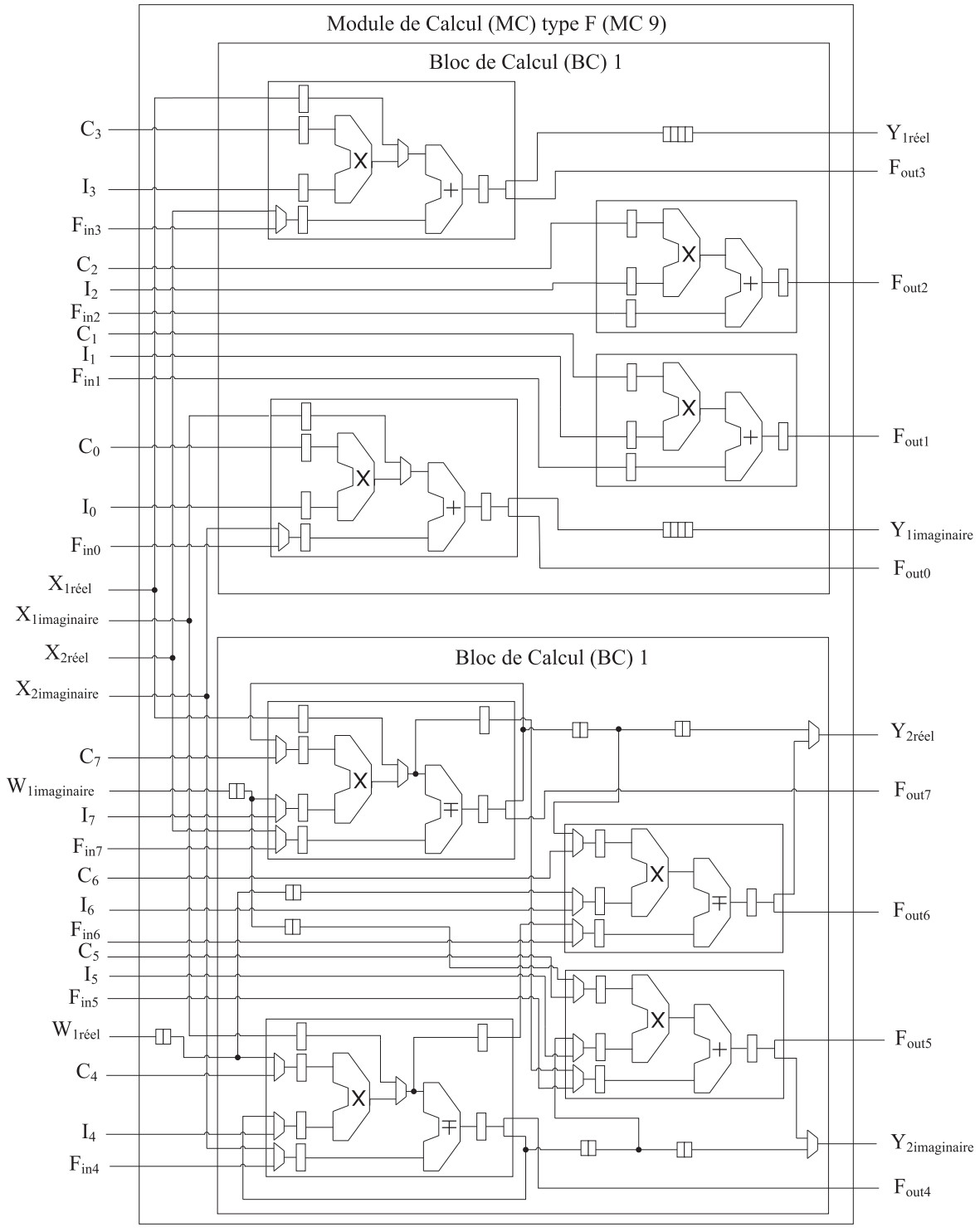

FIGURE D.5 – Architecture détaillée du module de calcul de type F

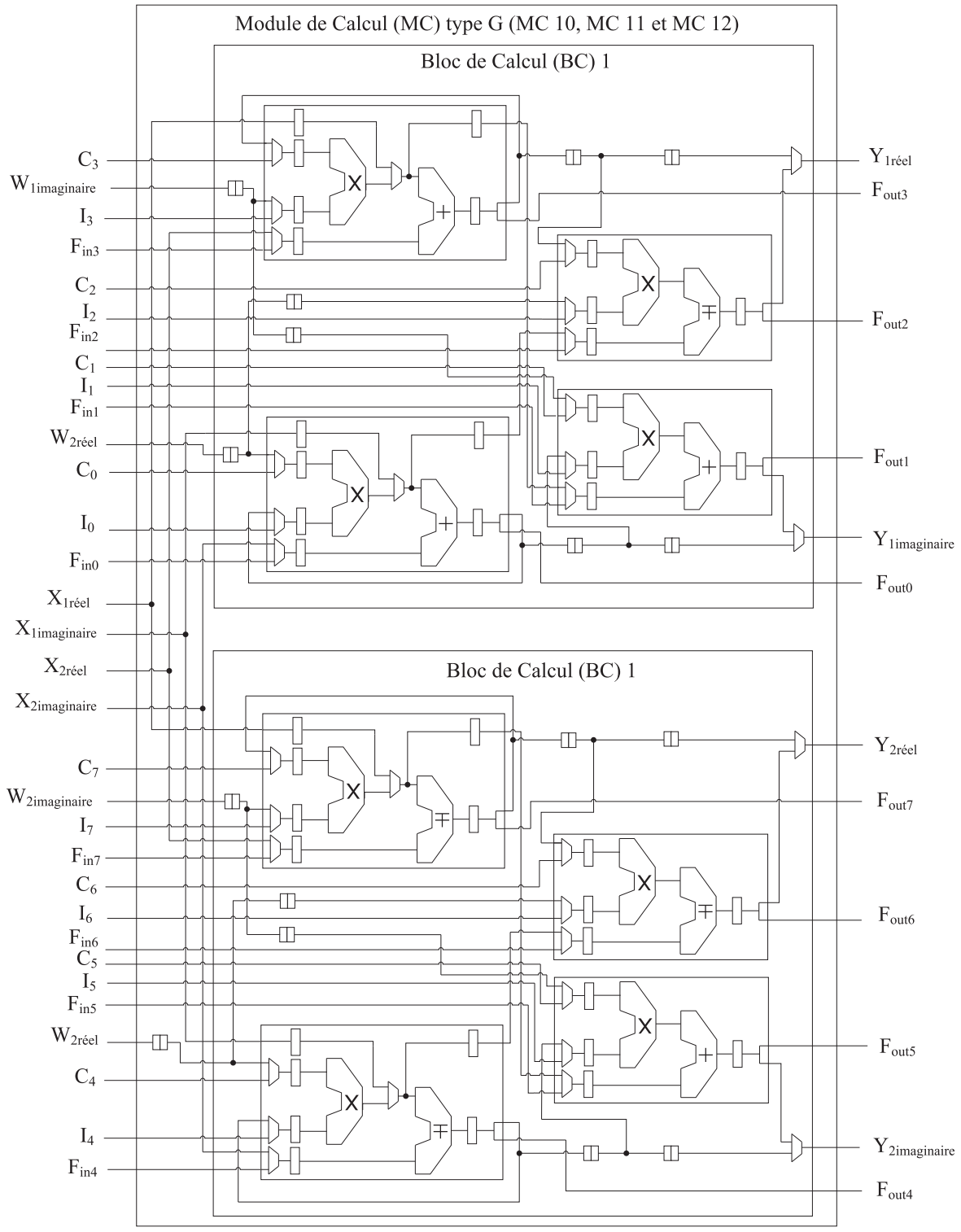

FIGURE D.6 – Architecture détaillée du module de calcul de type G

Annexe E

Schéma des modules de calcul implémentés sur FPGA

Table E.1 – Différents cas de figure des modules de calcul selon les multiplications triviales ou non triviales

	cas A	cas B	cas C	cas D	cas E	cas F	cas G
BC1	1	1	1,$\Im$	1,$\Im$	1,W,$\Im$	1,$\Im$	1,W,$\Im$
BC2	1	-j	1,$\Im$	-j,$\Im$	-j,W,$\Im$	1,W,$\Im$	1,W,$\Im$

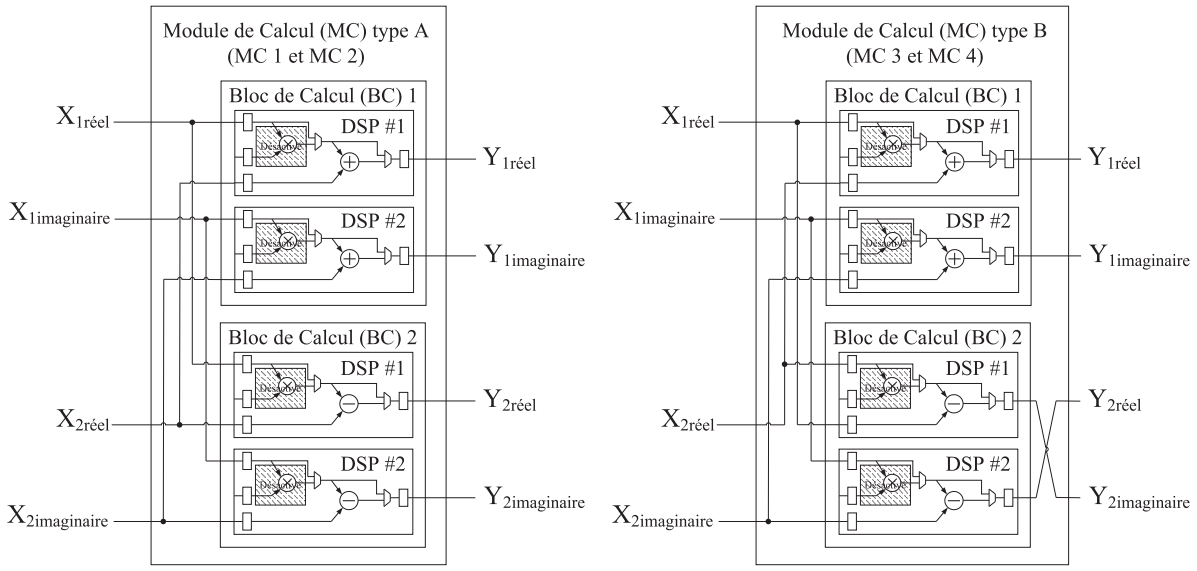

Figure E.1 – Architecture détaillée des modules de calcul de type A et B implémentés sur FPGA et utilisant des DSP48

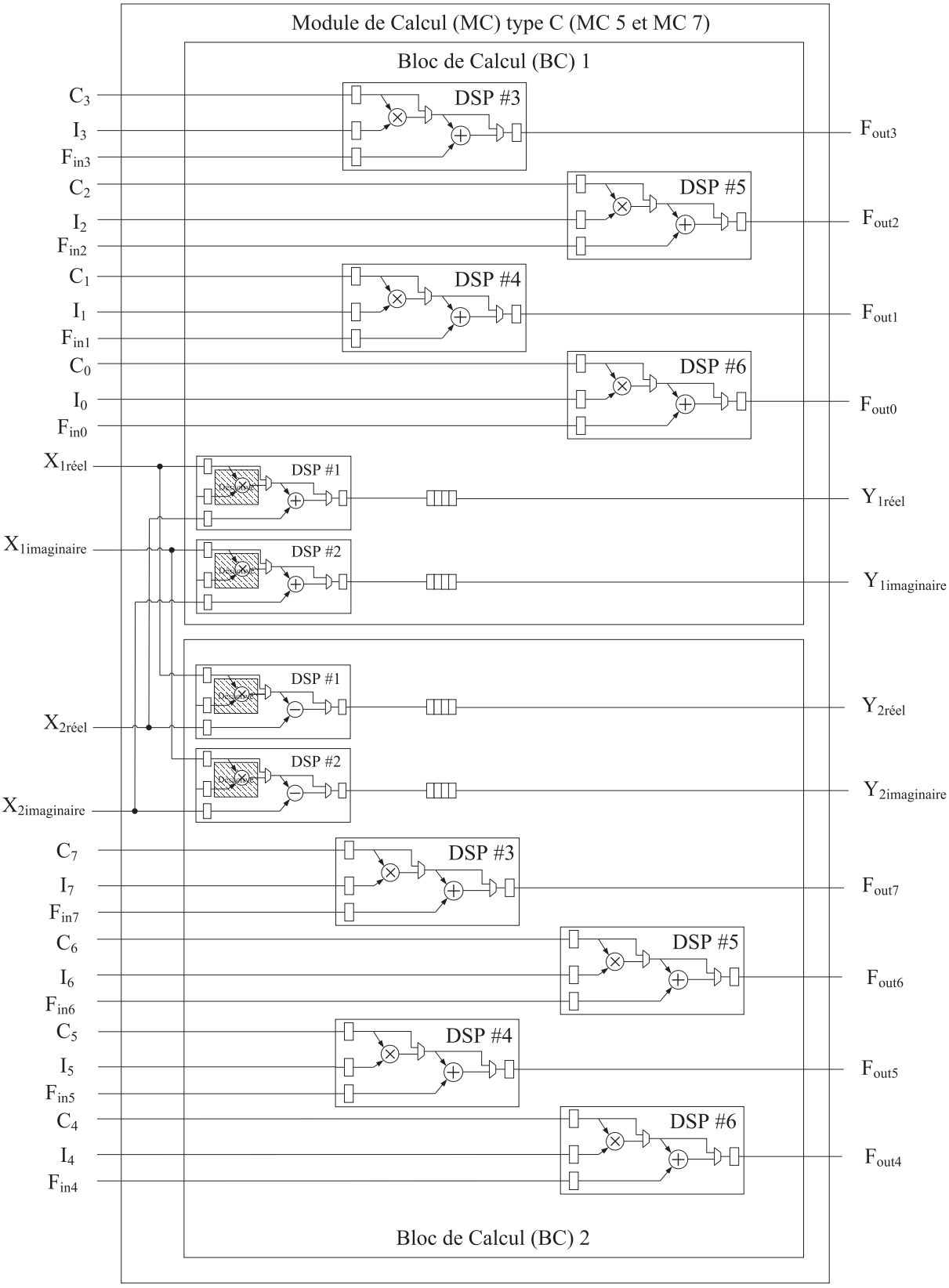

FIGURE E.2 – Architecture détaillée du module de calcul de type C implémenté sur FPGA et utilisant des DSP48

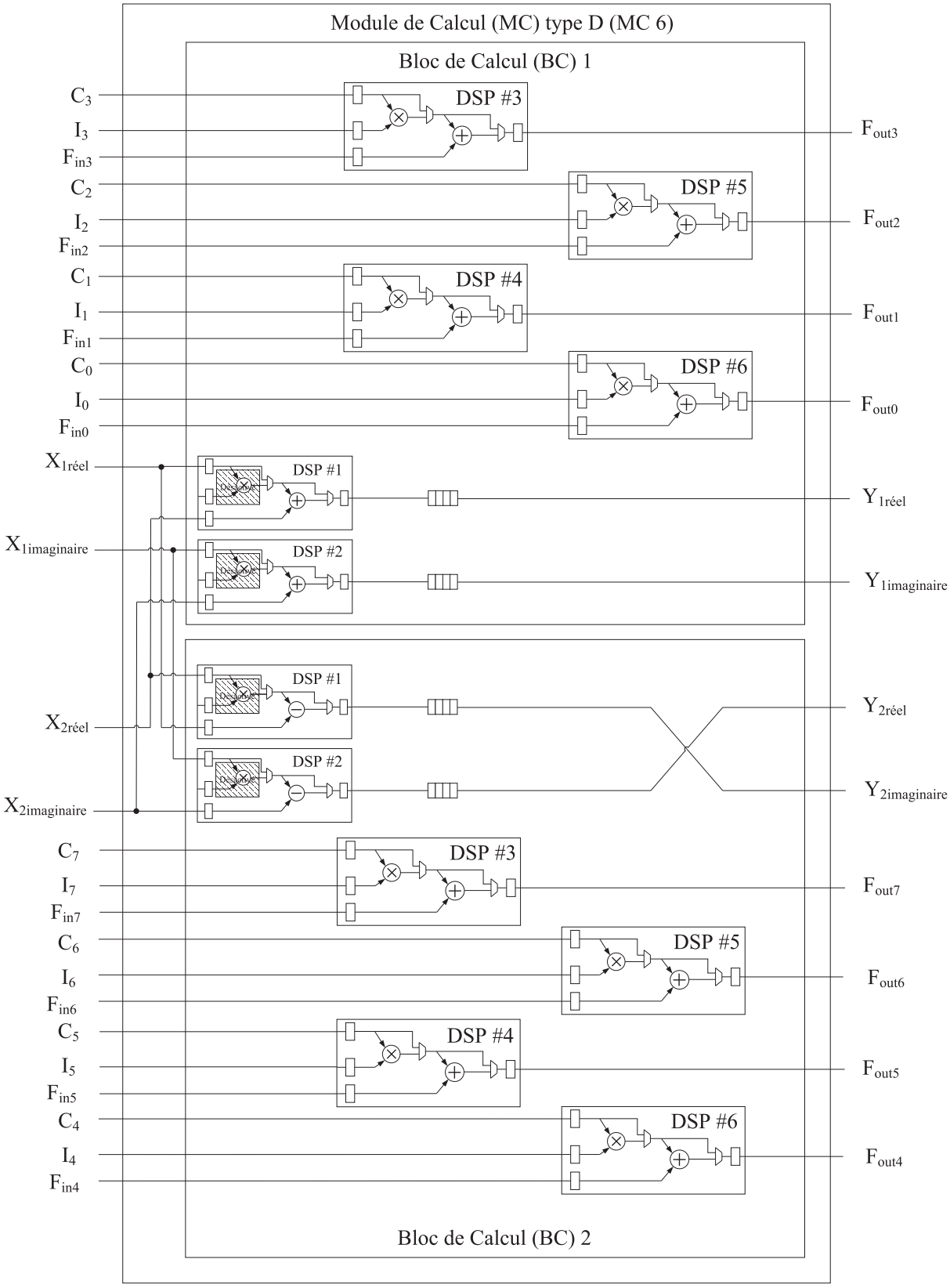

FIGURE E.3 – Architecture détaillée du module de calcul de type D implémenté sur FPGA et utilisant des DSP48

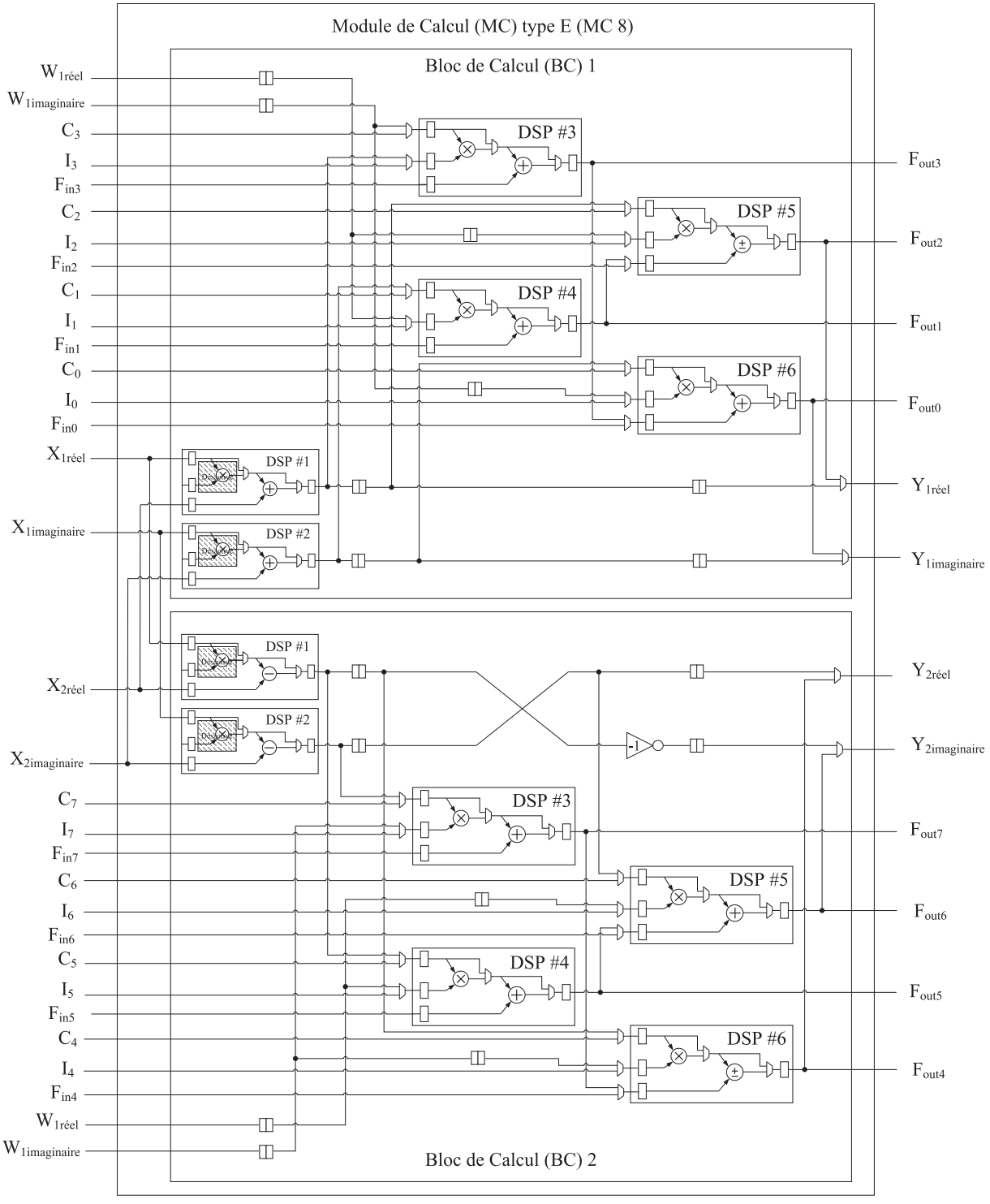

FIGURE E.4 – Architecture détaillée du module de calcul de type E implémenté sur FPGA et utilisant des DSP48

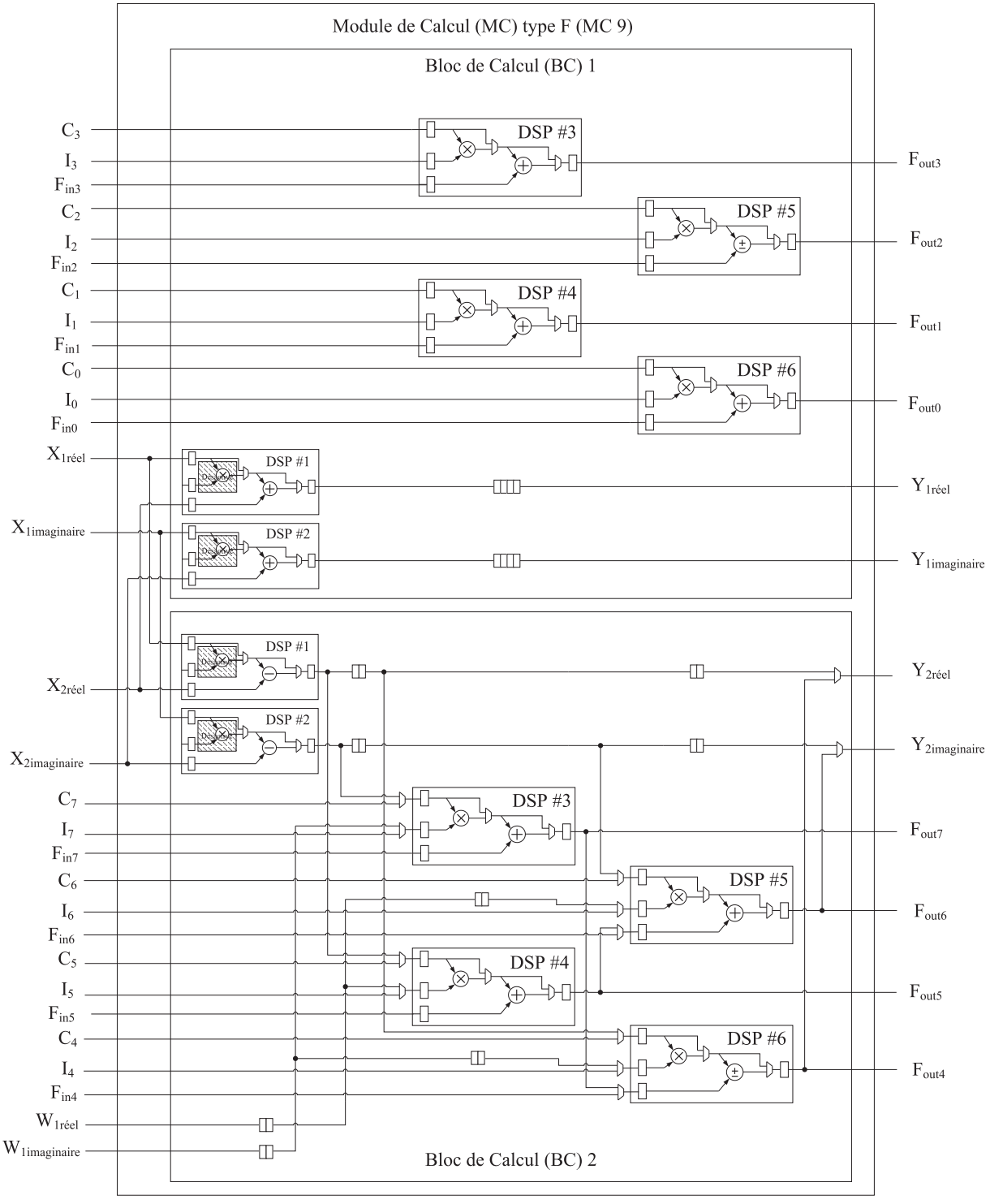

FIGURE E.5 – Architecture détaillée du module de calcul de type F implémenté sur FPGA et utilisant des DSP48

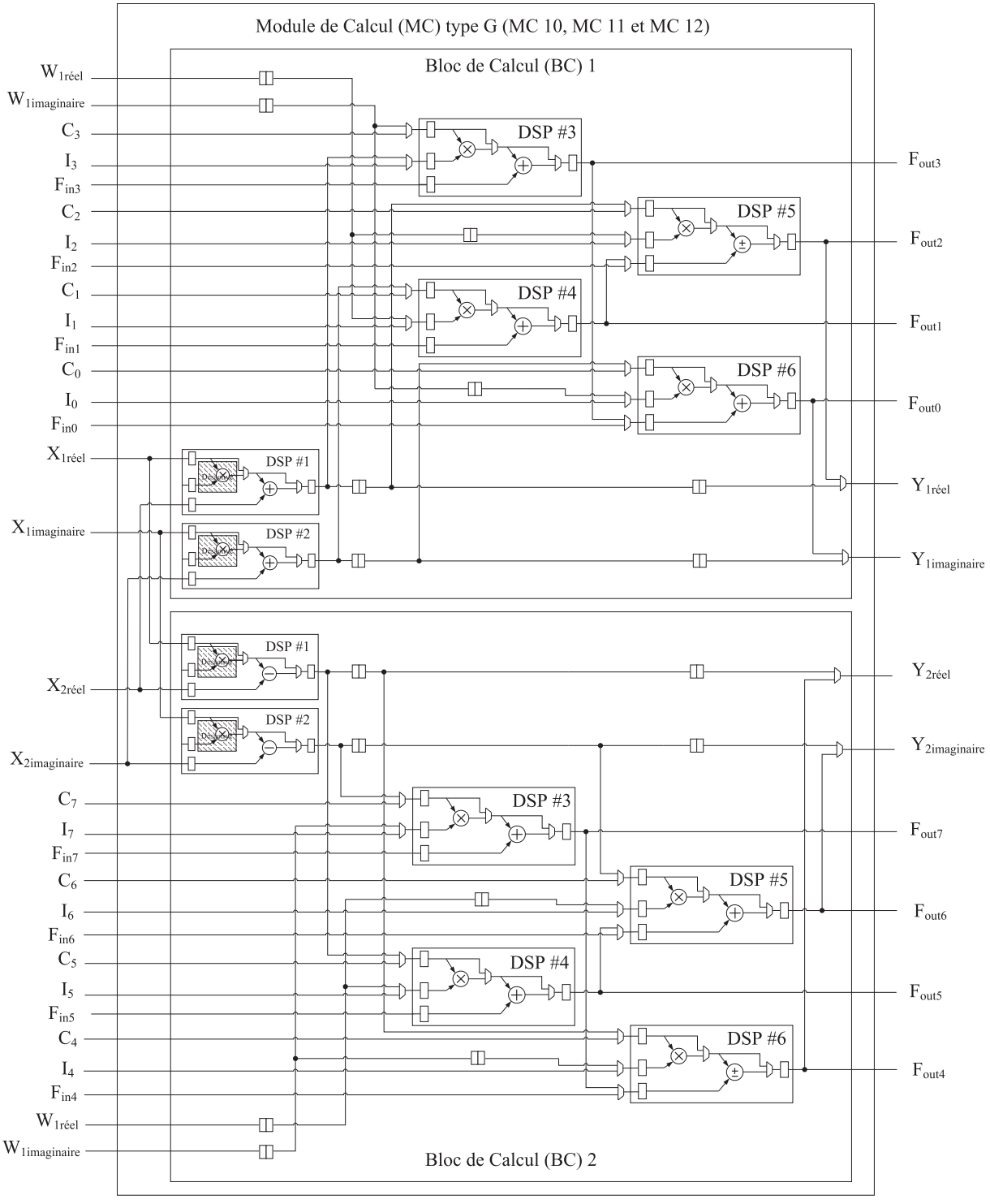

FIGURE E.6 – Architecture détaillée du module de calcul de type G implémenté sur FPGA et utilisant des DSP48

Bibliographie

[1] R. Olexa, *Implementing 802.11, 802.16, and 802.20 Wireless Networks Planning, Troubleshooting and Operations.* Elsevier, 2005.

[2] J. G. Andrews, A. Ghosh, and R. Muhamed, *Fundamentals of WiMAX Understanding Broadband Wireless Networking.* Prentice Hall, 2007.

[3] K. H. Teo, Z. Tao, and J. Zhang ;, "The mobile broadband WiMAX standard," IEEE *Signal Processing Magazine*, vol. 24, no. 5, pp. 144–148, Sept. 2007.

[4] A. R. S. Bahai, B. R. Saltzberg, and M. Ergen, *Multi-carrier Digital Communications Theory and Applications of* OFDM, 2nd ed. Springer science, 2004.

[5] P. Antoine, W. D. Wilde, C. Gendarme, S. Schelstraete, and P. Spruyt, "VDSL : la transmission de données sur paire de cuivre à la vitesse de la fibre," *Revue des Télécommunications d'Alcatel*, vol. 4e trimestre, 2000.

[6] "Digital video broadcasting (DVB) ; framing structure, channel coding and modulation for digital terrestrial television," ETSI, Tech. Rep., Novembre 2004.

[7] "Digital video broadcasting (DVB) ; DVB-H implementation guidelines," ETSI, Tech. Rep., Novembre 2005.

[8] M. Kornfeld and U. Reimers, "DVB-H the emerging standard for mobile data communication," *EBU Technical Review*, vol. 301, Janvier 2005.

[9] G. Faria, J. Henriksson, E. Stare, and P. Talmola, "DVB-H : Digital broadcast services to handheld devices," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 194–209, Janvier 2006.

[10] J. A. C. Bingham, *ADSL, VDSL, and Multicarrier Modulation*, J. G. Proakis, Ed. John Wiley & Sons, 2001.

[11] G. Tsoulos, MIMO *System technology for wireless communications.* Lavoisier, 2006.

[12] A. Naguib, N. Seshadri, and A. Calderbank, "Increasing data rate over wireless channels," *IEEE Signal Processing Magazine*, vol. 17, no. 3, pp. 76–92, 2000.

[13] A. F. Naguib, V. Tarokh, N. Seshadri, and A. R. Calderbank, "A space-time coding modem for high-data-rate wireless communications," *IEEE Journal on Selected Areas in Communication*, vol. 16, no. 8, Octobre 1998.

[14] R. Bachl, P. Gunreben, S. Das, and S. Tatesh, *The long term evolution towards a new* 3GPP *air interface standard.* Wiley Periodicals, 2007.

[15] Qualcomm, "Whitepaper : 3GPP long-term evolution (LTE)," Jan. 2008.

[16] Motorola, "Whitepaper : Long term evolution (LTE) a technical overview," 2007.

[17] Freescale, "Whitepaper : Overview of the 3GPP long term evolution physical layer," 2007.

[18] "3gpp tr 25.814 : Physical layer aspects for evolved universal terrestrial radio access (release 7)," Third Generation Partnership Project, Tech. Rep., 2006.

[19] W. P. Siriwongpairat and K. J. R. Liu, *Ultra-Wideband Communications Systems : Multiband* OFDM *Approach.* Wiley-IEEE Press, 2007.

[20] D. Leenaerts, R. van de Beek, J. Bergervoet, H. Kundur, and G. van der Weide, "Wimedia UWB technology : 480Mb/s wireless USB," in IEEE *International Workshop on Radio-Frequency Integration Technology*, Dec. 2007, pp. 8–12.

[21] C. Mishra, A. Valdes-Garcia, F. Bahmani, A. Batra, E. Sánchez-Sinencio, and J. Silva-Martinez, "Frequency planning and synthesizer architectures for multiband OFDM UWB radios," *IEEE Transactions on Microwave Theory and Techniques*, vol. 53, no. 12, pp. 3744–3756, 2005.

[22] K. Mandke, H. Nam, L. Yerramneni, C. Zuniga, and T. Rappaport, "The evolution of ultra wide band radio for wireless personal area networks," *High Frequency Electronics*, Septembre 2003.

[23] A. Stephann, E. Guéguen, M. Crussière, J.-Y. Baudais, , and J.-F. Hélard, "Optimization of linear precoded OFDM for high-data-rate UWB systems," *EURASIP Journal on Wireless Communications and Networking*, 2007.

[24] Y. G. LI and G. Stuber, Eds., *Orthogonal frequency division multiplexing for wireless communication.* Springer Science, 2006.

[25] H. Liu and G. Li, OFDM-*Based Broadband Wireless Networks Design and Optimization.* John Wiley & Sons, 2005.

[26] R. Prasad, OFDM *for Wireless Communications Systems.* Artech House, 2004.

[27] H. Schulze and C. Luders, *Theory and Applications of* OFDM *and* CDMA *Wideband Wireless Communications.* John Wiley & Sons, 2005.

[28] P. Siohan, C. Siclet, and N. Lacaille, "Analysis and design of OFDM/OQAM systems based on filterbank theory," *IEEE Transactions on Signal Processing*, vol. 50, no. 5, pp. 1170 – 1183, May 2002.

[29] B. Le Floch, M. Alard, and C. Berrou, "Coded orthogonal frequency division multiplex," *Proceedings of the IEEE*, vol. 83, no. 6, 1995.

[30] E. Gustafsson and A. Jonsson, "Always best connected," *IEEE Wireless Communication*, vol. 10, pp. 49–55, février 2003.

[31] U. Javaid, D. E. Meddour, T. Rasheed, and T. Ahmed, "Towards universal convergence in heterogeneous wireless networks using ad hoc connectivity," *in Proceedings of WPMC'06*, Septembre 2006.

[32] A. Jalali, "Etude et architecture d'un modem numérique destiné aux modulations multi-porteuses de densité 2," Ph.D. dissertation, Université de Rennes 1, Avril 1998.

[33] J. Reinauld, "Etude, développement et mise en oeuvre d'un modulateur OFDM avancé reconfigurable, haut débit et basse consommation," Master's thesis, Ecole Nationale Supérieure d'Electronique et de Radioélectricité de Grenoble ENSERG, Département Télécommunications, Juin 2007.

[34] R. Kara-Falah, "Réalisation d'une architecture reconfigurable d'un modulateur ofdm avancé haut débit et basse consommation," Master's thesis, Laboratoire TIMA, Septembre 2007.

[35] D. Fall, "FPGA and ASIC prototyping of an advanced OFDM modulator," Master's thesis, Polytech'Grenoble, Septembre 2008.

[36] DSP *: Designing for Optimal Results High-Performance* DSP *Using Virtex-4* FPGA*s*, Xilinx, Mars 2005.

[37] J. Mitola, "Software radio architecture : a mathematical perspective," IEEE *Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 514–538, April 1999.

[38] ——, "Technical challenges in the globalization of software radio," IEEE *Communications Magazine*, vol. 37, no. 2, pp. 84–89, Feb. 1999.

[39] P. Burns, *Software Defined Radio for 3G.* Artech House Mobile Communications Series, 2003.

[40] B. Haberland, W. Koenig, A. Pascht, and U. Weiss, "Software defined radio : A promising technology for multi-standard base stations," Alcatel Telecommunications Review, Tech. Rep., 2nd Quarter 2005.

[41] S. Ashley, "Cognitive radio," *Scientific American*, Mars 2006.

[42] R. David, "Architectures reconfigurables et faible consommation," in *Ecole thématique ECoFac Conception faible consommation de système temps réel*, 2006.

[43] J. G. Proakis and M. Salehi, *Fundamentals of communication systems.* Pearson Prentice Hall, 2005.

[44] B. R. Saltzberg, "Performance of an efficient parallel data transmission system," *IEEE Transactions on Communications Technology*, vol. 15, no. 6, pp. 805–811, 1967.

[45] R. W. Chang, "Synthesis of band-limited orthogonal signals for multi-channel data transmission," *Bell System Technical Journal*, vol. 46, pp. 1775–1796, 1966.

[46] R. W. Chang and R. A. Gibbey, "A theoretical study of performance of an orthogonal multiplexing data transmission scheme," *IEEE Transactions on Communications Technology*, vol. 16, no. 4, pp. 529–540, 1968.

[47] S. Weinstein and P. Ebert, "Data transmission by frequency-division multiplexing using the discrete fourier transform," *IEEE Transactions on Communications*, vol. 19, no. 5, pp. 628– 634, Oct. 1971.

[48] B. Hirosaki, "An orthogonally multiplexed QAM system using the discrete fourier transform," *IEEE Transactions on Communications*, vol. 29, no. 7, pp. 982– 989, Jul. 1981.

[49] M. Muck and J.-P. Javaudin, "Advanced OFDM modulators considered in the IST-WINNER framework for future wireless systems," *14th IST Mobile and Wireless Communications Summit*, June 2005.

[50] D. Rouffet, S. Kerboeuf, L. Cai, and V. Capdevielle, "4G Mobile," Alcatel Telecommunications Review, Tech. Rep., 2nd Quarter 2005.

[51] A. B. Salem, M. Siala, and H. Boujemâa, "Performance comparison of OFDM and OFDM/OQAM systems operating in highly dispersive radio-mobile channels," *IEEE International Conference on Electronics, Circuits and Systems, Gammarth*, Dec. 2005.

[52] J.-P. Javaudin, D. Lacroix, and A. Rouxel, "Pilot-aided channel estimation for OFDM/OQAM," *The 57th IEEE Semiannual Publication, Vehicular Technology Conference*, pp. 1581– 1585, April 2003.

[53] J.-P. Javaudin, C. Dubuc, D. Lacroix, and M. Earnshaw, "An OFDM evolution for the UMTS high speed downlink packet access," *60th IEEE Vehicular Technology Conference VTC2004-Fall*, vol. 2, pp. 846–850, Sept 2004.

[54] J. Du and S. Signell, "Comparison of CP-OFDM and OFDM/OQAM in doubly dispersive channels," *Future generation communication and networking (fgcn 2007)*, vol. 2, pp. 207–211, Dec 2007.

[55] M. El-Tabach, J.-P. Javaudin, and M. Helard, "Spatial data multiplexing over OFDM/OQAM modulations," *IEEE International Conference on Communications ICC '07*, pp. 4201–4206, Juin 2007.

[56] J.-P. Javaudin and P.-J. Bouvet, "Use of signals in quadrature over OFDM/OQAM," *IEEE 65th Vehicular Technology Conference VTC2007-Spring*, pp. 1891–1895, Avril 2007.

[57] B. Jahan, M. Lanoiselee, G. Degoulet, and R. Rabineau, "Full synchronization method for OFDM/OQAM and OFDM/QAM modulations," *IEEE 10th International Symposium onSpread Spectrum Techniques and Applications ISSSTA '08*, pp. 344–348, Juil. 2008.

[58] A. Skrzypczak, P. Siohan, N. Chotkan, and M. Djoko-Kouam, "OFDM/OQAM : An appropriate modulation scheme for an optimal use of the spectrum," *3rd International Symposium on Control and Signal Processing ISCCSP 2008*, pp. 405–410, Mars 2008.

[59] H. Bölcskei, P. Duhamel, and R. Hleiss, "Orthogonalization of OFDM/OQAM pulse shaping filters using the discrete Zak transform," *Signal Processing*, vol. 83, no. 7, pp. 1379–1391, 2003.

[60] 3GPP, "*3GPP TR 25.892* : Technical specification group radio access network ; feasibility study for orthogonal frequency division multiplexing (OFDM) for UTRAN enhancement (release 6)," the 3rd Generation Partnership Project, Technical specification, 2004.

[61] D. Lacroix, N. Goudard, and M. Alard, "OFDM with guard interval versus OFDM/offsetQAM for high data rate UMTS downlink transmission," *Vehicular Technology Conference Fall*, 2001.

[62] P. Jung, G. Wunder, and C.-S. Wang, "OQAM/IOTA downlink air interface for UMTS HSDPA evolution," *9th International OFDM-Workshop*, pp. 153–157, 2004.

[63] P. Jung, "OQAM/IOTA downlink air interface for 3G/4G," 2004.

[64] T. Tamura and Y. Sanada, "Fractional sampling OFDM/OQAM-IOTA on multipath channel with long delay spread," *IEEE Pacific Rim Conference on Computers and Signal Processing*, pp. 510–513, Juil. 2007.

[65] G. Cariolaro and F. Vagliani, "An OFDM scheme with a half complexity," *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, vol. 13, no. ç, Dec. 1995.

[66] L. Vangelista and N. Laurenti, "Efficient implementations and alternative architectures for OFDM-OQAM systems," *IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. 49, NO. 4, APRIL*, vol. 49, no. 4, Avril 2001.

[67] N. J. Fliege, "Computational efficiency of modified DFT polyphase filter banks," *The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, 1993.

[68] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, p. 297, April 1965.

[69] J. Cooley, P. A. W. Lewis, and P. D. Welch, "Historical notes on the fast fourier transform," *Proceedings of the* IEEE, vol. 55, no. 10, pp. 1675–1677, October 1967.

[70] L. R. Rabiner and B. Gold, *Theory and application of digital signal processing*, E. Cliffs, Ed. NJ : Prentice Hall, 1975.

[71] P. Duhamel and H. Hollmann, ""split-radix FFT algorithm," *Electron. Lett.*, vol. 20, no. 1, pp. 14–16, 1984.

[72] P. Duhamel, "Implementation of "split-radix" FFT algorithms for complex, real, and real-symmetric data," IEEE *Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 2, pp. 285 – 295, April 1986.

[73] M. Vetterli and P. Duhamel, "Split-radix algorithms for length-p^m DFTs," *International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1415 – 1418, 11-14 April 1988.

[74] Z. J. Mou and P. Duhamel, "In-place butterfly-style FFT of 2-D real sequences," IEEE *Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 10, pp. 1642 – 1650, Oct. 1988.

[75] P. Duhamel, "Algorithms meeting the lower bounds on the multiplicative complexity of length-2^n DFTs and their connection with practical algorithms," IEEE *Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 9, pp. 1504 – 1511, Sept. 1990.

[76] M. Vetterli and P. Duhamel, "Fast fourier transforms : a tutorial review and a state of the art," *Signal Processing*, vol. 19, pp. 259–299, 1990.

[77] M. Bellanger, *Traitement numérique du signal, théorie et pratique.* Masson, 1980.

[78] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "A new radix-2/8 FFT algorithm for length-q x 2^m DFTs," IEEE *Transaction on circuits and systems-I : Regular papers*, vol. 51, no. 9, Sept. 2004.

[79] ——, "Arithmetic complexity of the split-radix FFT algorithms," IEEE *International Conference on Acoustics, Speech, and Signal Processing (*ICASSP *'05)*, vol. 5, pp. v/137 – v/140, Sept. 2004.

[80] C. Meletis, P. Bougas, G. Economakos, P. Kalivas, and K. Pekmestzi, "High-speed pipeline implementation of radix-2 DIF algorithm," *International* Journal *of* Signal Processing, vol. 4, no. 1, 2004.

[81] C. Cheng and K. K. Parhi, "High-throughput VLSI architecture for FFT computation," IEEE *Transactions on Circuits and Systems*, vol. 54, no. 10, pp. 863 – 867, Oct. 2007.

[82] W. Han, T. Arslan, A. T. Erdogan, and M. Hasan, "Low power commutator for pipelined FFT processors," IEEE *International Symposium on Circuits and Systems (*ISCAS *2005)*, pp. 5274–5277, May 2005.

[83] S. Saponara, L. Serafini, and L. Fanucci, "Low-power FFT/IFFT VLSI macro cell for scalable broadband VDSL modem," *The 3rd* IEEE *International Workshop on System-on-Chip for Real-Time Applications*, pp. 161 – 166, June-July 2003.

[84] S. Lee and S.-C. Park, "Modified SDF architecture for mixed DIF/DIT FFT," IEEE *International Symposium on Circuits and Systems*, pp. 2590 – 2593, May 2007.

[85] Y.-T. Lin, P.-Y. Tsai, and T.-D. Chiueh, "Low-power variable-length fast fourier transform processor," *Computers and Digital Techniques*, vol. 152, no. 4, pp. 499 – 506, July 2005.

[86] Y. Chen, Y.-W. Lin, Y.-C. Tsao, and C.-Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1260–1273, 2008.

[87] T. Lenart and V. Öwall, "Architectures for dynamic data scaling in 2/4/8k pipeline FFT cores," *IEEE Transactions on Very Large Scale Integration (VLSI) systems*, vol. 14, no. 11, 2006.

[88] J. Lee, H. Lee, S. in Cho, and S.-S. Choi, "A high-speed, low-complexity radix-2/sup 4/ FFT processor for MB-OFDM UWB systems," IEEE *International Symposium on Circuits and Systems*, p. 4, May 2006.

[89] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A 1-GS/s FFT/IFFT processor for UWB applications," *Journal of solid-state circuits*, vol. 40, no. 8, pp. 1726– 1735, Aug. 2005.

[90] E. Wold and A. Despain, "Pipeline and parallel-pipeline FFT processors for VLSI implementation," IEEE *Trans. Comput.*, vol. C-33, no. 5, pp. 414 – 426, 1984.

[91] W. Li, Y. Ma, and L. Wanhammar, "Word length estimation for memory efficient pipeline FFT/IFFT processors," *Conference on Signal Processing Applications and Technology*, Nov. 1999.

[92] J. yeol Oh and M. seob Lim, "Area and power efficient pipeline FFT algorithm," IEEE *Workshop on Signal Processing Systems Design and Implementation*, pp. 520 – 525, Nov. 2005.

[93] J. Garcia, J. Michell, and A. Buron, "VLSI configurable delay commutator for a pipeline split radix FFT architecture," IEEE *Transactions on Signal Processing*, vol. 47, no. 11, pp. 3098 – 3107, 1999.

[94] W.-C. Yeh and C.-W. Jen, "High-speed and low-power split-radix FFT," IEEE *Transactions on Signal Processing*, vol. 51, no. 3, pp. 864 – 874, March 2003.

[95] B. Jo and M. Sunwoo, "New continuous-flow mixed-radix (CFMR) FFT processor using novel in-place strategy," IEEE *Transactions on Circuits and Systems*, vol. 52, no. 5, pp. 911 – 919, May 2005.

[96] C.-C. Wang, J.-M. Huang, and H.-C. Cheng, "A 2k/8k mode small-area FFT processor for OFDM demodulation of DVB-T receivers," IEEE *Transactions on Consumer Electronics*, vol. 51, no. 1, pp. 28 – 32, Feb 2005.

[97] C.-L. Wey, W.-C. Tang, and S.-Y. Lin, "Efficient VLSI implementation of memory-based FFT processors for DVB-T applications," IEEE *Computer Society Annual Symposium on VLSI*, pp. 98 – 106, March 2007.

[98] C.-H. Sung, K.-B. Lee, and C.-W. Jen, "Design and implementation of a scalable fast fourier transform core," IEEE *Asia-Pacific Conference on* ASIC, pp. 295 – 298, Aug. 2002.

[99] C.-P. Hung, S.-G. Chen, and K.-L. Chen, "Design of an efficient variable-length FFT processor," *Proceedings of the 2004 International Symposium on Circuits and Systems*, vol. 2, pp. II – 833–6, May 2004.

[100] S.-Y. Lee and C.-C. Chen, "VLSI implementation of programmable FFT architectures for OFDM communication system," *Proceeding of the 2006 international conference on Communications and mobile computing*, pp. 893 – 898, 2006.

[101] P.-Y. Tsai, T.-H. Lee, and T.-D. Chiueh, "Power-efficient continuous-flow memory-based FFT processor for WiMax OFDM mode," *International Symposium on Intelligent Signal Processing and Communications*, pp. 622 – 625, Dec. 2006.

[102] C.-L. Wey, S.-Y. Lin, W.-C. Tang, and M.-T. Shiue ;, "High-speed, low cost parallel memory-based FFT processors for OFDM applications," *14th IEEE International Conference on Electronics, Circuits and Systems ICECS 2007*, pp. 783–787, 11-14 Dec. 2007.

[103] K. H. Chen and Y. S. Li, "A multi-radix FFT processor using pipeline in memory-based architecture (PIMA) for DVB-T/H systems," *15th International Conference on Mixed Design of Integrated Circuits and Systems, MIXDES*, pp. 549–553, June 2008.

[104] C.-H. Su and J.-M. Wu, "Reconfigurable FFT design for low power OFDM communication systems," IEEE *Tenth International Symposium on Consumer Electronics*, pp. 1 – 4, June 2006.

[105] C. Ebeling, C. Fisher, X. Guanbin, S. Manyuan, and L. Hui, "Implementing an OFDM receiver on the RaPiD reconfigurable architecture," IEEE *Transactions on Computers*, vol. 53, no. 11, pp. 1436 – 1448, Nov. 2004.

[106] B. M. Baas, "A low-power, high-performance, 1024-point fft processor," IEEE *Journal of Solid-State Circuits (*JSSC*)*, pp. 380–387, March 1999.

[107] G. Zhong, F. Xu, and A. N. Willson, "A power-scalable reconfigurable FFT/IFFT IC based on a multi-processor ring," IEEE *Journal of Solid-State Circuits*, vol. 41, no. 2, pp. 482 – 495, February 2006.

[108] S. Martin, K. Flautner, T. Mudge, and D. Blaauw, "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," IEEE/ACM *International Conference on Computer Aided Design*, pp. 721 – 725, 10-14 Nov. 2002.

[109] T. Lenart and V. Öwall, "A 2048 complex point FFT processor using a novel data scaling approach," *Proceedings of the 2003 International Symposium on Circuits and Systems*, vol. 4, pp. 45–48, Mai 2003.

[110] H.-F. Lo, M.-D. Shieh, and C.-M. Wu, "Design of an efficient FFT processor for DAB system," *The 2001 IEEE International Symposium on Circuits and Systems ISCAS 2001*, vol. 4, pp. 654–657, 6-9 Mai 2001.

[111] C.-P. Hung, S.-G. Chen, and K.-L. Chen, "Design of an efficient variable-length FFT processor," *Proceedings of the 2004 International Symposium on Circuits and Systems (ISCAS '04)*, vol. 2, pp. 833–836, Mai 2004.

[112] P. D. Welch, "A fixed-point fast Fourier transform error analysis," *IEEE Transactions on Audio and Electroacoustics*, vol. 17, pp. 151–157, Juin 1969.

[113] A. V. Oppenheim and C. Weinstein, "Effects of finite register length in digital filtering and the fast Fourier transform," *Proceedings of the IEEE*, vol. 60, no. 8, pp. 957–976, Juillet 1972.

[114] V. U. Reddy and M. Sundaramurthy, "New results in fixed-point fast Fourier transform error analysis," *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '76*, vol. 1, pp. 120–125, 1976.

[115] Tran-Thong and B. Liu, "Fixed-point fast Fourier transform error analysis," *IEEE textsc-Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 6, pp. 563–573, Decembre 1976.

[116] V. U. Reddy and M. Sundaramurthy, "Some results in fixed-point fast Fourier transform error analysis," *IEEE Transactions on Computers*, vol. C-26, no. 3, pp. 305–308, Mars 1977.

[117] Y. Ma, "An accurate error analysis model for fast Fourier transform," *IEEE Transactions on Signal Processing*, vol. 45, no. 6, pp. 1641–1645, Juin 1997.

[118] A. Zergainoh, "Algorithmes rapides de filtrages fir adaptés à une implantation sur dsp," Ph.D. dissertation, Paris 11, Déc. 1994.

[119] *User's Guide*, Update for systemc 2.0.1 ed., SystemC Open Source License.

[120] V. U. Reddy and M. Sundaramurthy, "Effect of correlation between transactions errors on fixed-point fast Fourier transform analysis," *IEEE Transactions on Circuits and Systems*, vol. 27, no. 8, pp. 712–716, Juillet 1980.

[121] Y.-W. Lin, H.-Y. Liu, and C.-Y. Lee, "A dynamic scaling FFT processor for DVB-T applications," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 11, pp. 2005–2013, Novembre 2004.

[122] H.-G. Kim, J.-S. Youn, K.-T. Yoon, and J.-R. Choi, "8k-point pipelined FFT/IFFT with compact memory for DVB-T using block floating-point scaling technique," *5th International Conference on Information and Communications Technology*, pp. 41–44, Decembre 2007.

[123] *ML401/ML402/ML403 Evaluation Platform User Guide*, Xilinx, Mai 2006.

[124] *Virtex-4 FPGA User Guide*, Xilinx, Juin 2008.

[125] D. Elleouet, Y. Savary, and N. Julien, "An FPGA power aware design flow," *16th international workshop on power and timing modeling, optimization and simulation (PATMOS)*, pp. 415–424, 2006.

[126] Y.-T. Lin, P.-Y. Tsai, and T.-D. Chiueh, "Low-power variable-length fast fourier transform processor," *IEE Proceedings Computers and Digital Techniques*, vol. 152, no. 4, pp. 499–506, July 2005.

[127] F. Kristensen, P. Nilsson, , and A. Olsson, "A generic transmitter for wireless OFDM systems," *Personal, Indoor and Mobile Radio Communications PIMRC 2003*, September 2003.

[128] K. J. Nowka, G. D. Carpenter, E. W. MacDonald, H. C. Ngo, B. C. Brock, K. I. Ishii, T. Y. Nguyen, and J. L. Burns, "A 32-bit powerPC system-on-a-chip with support for dynamic voltage scaling and dynamic frequency scaling," *IEEE J. Solid-State Circuits*, vol. 11, pp. 1441–1447, 37.

[129] T. Fujiyoshi, S. Shiratake, S. Nomura, T. Nishikawa, Y. Kitasho, H. Arakida, Y. Okuda, Y. Tsuboi, M. Hamada, H. Hara, T. Fujita, F. Hatori, T. Shimazawa, K. Yahagi, H. Takeda, M. Murakata, F. Minami, N. Kawabe, T. Kitahara, K. Seta, M. Takahashi, Y. Oowaki, and T. Furuyama, "A 63-mw H.264/MPEG-4 audio/visual codec LSI with module-wise dynamic voltage/frequency scaling," *IEEE J. Solid-State Circuits*, vol. 41, no. 1, pp. 54–62, Jan. 2006.

[130] M. Nakai, S. Akui, K. Seno, T. Meguro, T. Seki, T. Kondo, A. Hashiguchi, H. Kawahara, K. Kumano, and M. Shimura, "Dynamic voltage and frequency management for a low-power embedded microprocessor," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 28–35, Jan. 2005.

[131] M. Nomura, Y. Ikenaga, K. Takeda, Y. Nakazawa, Y. Aimoto, and Y. Hagihara, "Delay and power monitoring schemes for minimizing power consumption by means of supply and threshold voltage control in active and standby modes," *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 805–814, Apr. 2006.

[132] C. Andriamisaina, E. Casseau, and P. Coussy, "Synthesis of multimode digital signal processing systems," *Second NASA/ESA Conference on Adaptive Hardware and Systems, 2007. AHS 2007*, pp. 318–325, Juil. 2007.

[133] P. B. Kenington, *RF And Baseband Techniques for Software Defined Radio*. Artech House, 2005.

[134] S. He and M. Torkelson, "A new approach to pipeline FFT processor," *The 10th International Parallel Processing Symposium*, pp. 766 – 770, April 1996.

[135] ——, "Designing pipeline FFT processor for OFDM (de)modulation," *International Symposium on Signals, Systems, and Electronics*, pp. 257 – 262, 29 Sept.-2 Oct. 1998.

[136] J.-Y. Oh and M.-S. Lim, "New radix-2 to the 4th power pipeline FFT processor," *Institute of* E*lectronics,* I*nformation and* C*ommunication* E*ngineers*, vol. E88, no. 8, 2005.

[137] K. Babionitakis, K. Manolopoulos, K. Nakos, N. Reisis, N. Vlassopoulos, and V. Chouliaras, "A high performance VLSI FFT architecture," *13th* IEEE *International Conference on Electronics, Circuits and Systems*, pp. 810 – 813, Dec. 2006.