HAL
open science

Etude et conception d'architectures haut-débit pour la modulation et la démodulation numériques

Gérald Arnould

To cite this version:

Gérald Arnould. Etude et conception d'architectures haut-débit pour la modulation et la démodulation numériques. Autre. Université Paul Verlaine - Metz, 2006. Français. NNT : 2006METZ032S . tel-01752369

HAL Id: tel-01752369

https://hal.univ-lorraine.fr/tel-01752369

Submitted on 29 Mar 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE LORRAINE

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4
Code de la Propriété Intellectuelle. articles L 335.2- L 335.10
http://www.cfcopies.com/V2/leg/leg_droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

Laboratoire Interfaces Capteurs et
Microélectronique

Université Paul Verlaine - Metz

École Doctorale IAEM - Lorraine
Département de Formation Doctorale Électronique - Électrotechnique

THÈSE DE DOCTORAT

Discipline : Electronique

Etude et Conception d'Architectures Haut-Débit pour la Modulation et la Démodulation Numériques

Par
Gérald ARNOULD

Soutenue le 8 Décembre 2006

Président	
Francis BRAUN	Professeur à l'Université Louis Pasteur - Strasbourg
Rapporteurs	
Mohamad SAWAN	Professeur à l'École Polytechnique de Montréal
Amara AMARA	Professeur à l'ISEP - Paris
Directeur de thèse	
Abbas DANDACHE	Professeur à l'Université Paul Verlaine - Metz
Co-Directeur de thèse	
Fabrice MONTEIRO	Maître de Conférence HDR à l'Université Paul Verlaine - Metz
Examinateurs	
Bernard LEPLEY	Professeur à l'Université Paul Verlaine - Metz

LICM - 7 rue Marconi, 57070 Metz Technopôle, France

Table des matières

Table des matières **iv**

Liste des tableaux **viii**

Table des figures **x**

Introduction **xiii**

1 Transmission numérique de l'information **1**

1.1 Chaîne de transmission de l'information 3

1.1.1 Introduction 3

1.1.2 Les modèles de référence 3

1.1.3 Organisation fonctionnelle de la couche physique 9

1.1.4 Atouts et contraintes d'un traitement numérique 12

1.2 Transmission d'un signal numérique en bande transposée 13

1.2.1 Échantillonnage des signaux 14

1.2.2 Cas des signaux sinusoïdaux 14

1.2.3 Quantification d'un signal 15

1.2.4 Production numérique de signaux en bande transposée 16

1.3 Architectures analogiques pour la modulation et la démodulation numériques 17
1.3.1 Modulateur QAM analogique 17
1.3.2 Démodulateur QAM analogique 18
1.3.3 Incompatibilités avec le traitement numérique 19
1.4 Circuits FPGA dans le cadre de la modulation et la démodulation 20
1.4.1 Architecture interne 22
1.4.2 Les blocs logiques LAB 24
1.4.3 Architecture d'une cellule logique 26
1.4.4 Limitations et avantages pour le traitement numérique du signal 28
1.5 Conclusion 30

2 Architectures pour les modulations numériques 31

2.1 Architecture d'un modulateur numérique 33
2.1.1 Différents types de modulations numériques 33
2.1.2 Structure générale d'un modulateur numérique 34
2.1.3 Conclusion 37
2.2 Générateurs de sinusoïdes : architectures existantes 38
2.2.1 Introduction 38
2.2.2 Utilisation de mémoires et d'échantillons pré-calculés 39
2.2.3 Compression de ROM 40
2.2.4 Utilisation couplée de filtres interpolateurs 41
2.2.5 Conclusion 44
2.3 Processeur CORDIC modifié 45
2.3.1 Introduction 45
2.3.2 Principe du processeur CORDIC 46

2.3.3 Application à la génération paramétrique de sinusoïdes 49
2.3.4 Architecture CORDIC itérative 52
2.3.5 Architecture CORDIC pipeline 54
2.3.6 Approximation de la valeur des $\arctan 2^{-i}$ 56
2.3.7 Optimisation de CORDIC pour la génération de sinusoïdes . . . 59
2.3.8 Architecture parallèle 62
2.3.9 Conclusions 65
2.4 Implantation et résultats 65
2.4.1 Méthodologie 65
2.4.2 Logiciel de génération automatique 66
2.4.3 Modulateur QAM 66
2.4.4 Performances comparées 67

3 Architectures pour la démodulation numérique 73
3.1 Registres à décalage à rétroaction linéaire 75
3.1.1 Principe mathématique 76
3.1.2 Périodicité, polynôme minimal et polynôme primitif 77
3.1.3 Algorithme de Berlekamp-Massey 78
3.1.4 Architectures matérielles pour l'implantation de LFSR 79
3.1.5 Compteurs à base de LFSR 80
3.1.6 Décompteur apparié 81
3.1.7 Conclusion 82
3.2 Application à la démodulation de fréquence 82
3.2.1 Architecture du démodulateur 84
3.2.2 Architecture parallèle 87
3.2.3 Implantation et résultats 90

3.3 Architecture pour la démodulation entièrement numérique de la phase . 93

3.3.1 Architecture du démodulateur 94

3.3.2 Mesure du déphasage 97

3.3.3 Implantation et résultats 101

Conclusion générale **107**

Références **111**

Liste des acronymes **121**

Liste des tableaux

1.1 Comparaison des FPGA par rapport aux autres architectures de traitement du signal [MESU05] 21

2.1 Performances comparées des différentes architectures pour la modulation QAM. 70

Table des figures

1.1 Modèle de Référence OSI 4

1.2 Modèle de référence TCP/IP en parallèle avec le modèle OSI 7

1.3 Modèle de référence UIT-T 8

1.4 Représentation fonctionnelle des couches physique et liaison d'un système de transmission comprenant une paire émetteur - récepteur 9

1.5 Représentation d'une ligne de transmission, séparation analogique numérique 11

1.6 Architecture d'un modulateur QAM analogique 18

1.7 Architecture d'un démodulateur QAM analogique 19

1.8 Architecture globale d'un FPGA de type FLEX10KE 22

1.9 Architecture d'un bloc logique 25

1.10 Architecture d'une cellule logique 26

1.11 Élément logique en mode de fonctionnement «normal» 27

1.12 Élément logique en mode de fonctionnement arithmétique 28

2.1 Architecture de l'accumulateur de phase 36

2.2 Architecture générale d'un générateur de sinusoïdes à base de ROM 38

2.3 Décompositions en plusieurs micro-rotations 48

2.4 Cellule élémentaire d'un processeur CORDIC 51

2.5 Générateur de sinusoïdes à base de processeur CORDIC itératif. 53

2.6 Générateur de sinusoïdes à base de processeur CORDIC pipeline. 54
2.7 Deux architectures de décaleurs. 55
2.8 Principe du recodage de l'angle. 61
2.9 Architecture de génération de sinusoïdes utilisant un recodage de l'angle. . 62
2.10 Production en parallèle de plusieurs échantillons. 63
2.11 Fréquence d'échantillonnage maximale atteinte avec les différentes architectures. 67
2.12 Nombre de cellules logiques (LC) utilisées sur le circuit FPGA. 68
2.13 Fréquence d'échantillonnage en fonction du nombre de cellules logiques occupées. 69

3.1 Les deux architectures de LFSR. 75
3.2 Principe d'un compteur basé sur un LFSR 80
3.3 Normalisation d'un signal M-FSK. 84
3.4 Architecture du démodulateur de fréquence à un étage. 84
3.5 Exemple d'une mesure de fréquence avec deux LFSR. 85
3.6 Architecture parallèle du démodulateur de phase. 88
3.7 Utilisation d'un désérialiseur rapide. 90
3.8 Fréquence d'échantillonnage pour différents types d'architectures de démodulation de fréquence. 90
3.9 Nombres de cellules logiques occupées en fonction de la résolution binaire. 91
3.10 Fréquence d'échantillonnage en fonction du nombre de cellules logiques occupées. 92
3.11 Durées des états haut et bas pour un signal non déphasé, et un signal avec un déphasage. 95
3.12 Architecture du démodulateur de phase 95

3.13 Détection d'un déphasage - premier chronogramme 98
3.14 Détection d'un déphasage - second chronogramme 99
3.15 Détection d'un déphasage - troisième chronogramme 100
3.16 Fréquence de fonctionnement maximale pour différents types d'architectures de démodulation de phase. 101
3.17 Nombres de cellules logiques occupées. 102
3.18 Fréquence d'horloge maximale atteinte en fonction du nombre de cellules logiques occupées. 103

Introduction

Le domaine des télécommunications et des réseaux a connu de profonds changements, et une évolution rapide pendant les années 1990, avec par exemple le développement des systèmes informatiques grand public. Sur le plan technologique en particulier, les réseaux de transmission ont vu leur capacité s'accroître, notamment concernant le débit supporté. Mais la demande est toujours plus importante, et suit l'évolution des contenus transportés. Car en plus des données traditionnelles, les contenus dits *multimédia*, comportant entre autres du son et de la vidéo de haute qualité, nécessitent toujours plus de vitesse, ainsi que des capacités de traitement du signal accrues.

Concernant les télécommunications, différents niveaux de traitement peuvent être considérés. Entre le niveau applicatif, qui concerne directement l'utilisateur, et le médium physique de transmission, pour lequel seuls des signaux et les composantes élémentaires des données sont pris en compte, les algorithmes mis en oeuvre n'ont pas les mêmes objectifs, ni les mêmes contraintes de performances. À l'extrémité du dispositif de télécommunication se trouvent les couches les plus basses de la chaîne de transmission, celles qui sont chargées de produire un signal représentatif des données à transmettre et adapté au médium, permettant sa transmission vers son destinataire. Ce sont ces couches basses qui sont les plus sensibles aux changements de protocoles de transmission, car elles doivent produire un signal dont les caractéristiques physiques et temporelles sont très dépendantes des spécifications du canal de communication. Aussi,

ces éléments requièrent pour leur conception, une attention toute particulière afin de ne pas brider la vitesse de traitement qu'ils sont capables d'atteindre, ce qui limiterait leur champ d'application et les possibilités d'évolution.

Des systèmes complexes de codage et d'adaptation des signaux sont donc apparus pour exploiter au maximum les capacités des média de transmission, et les architectures de modulation et de démodulation qui en découlent ont elles aussi dû évoluer pour supporter les cadences de traitement plus élevées.

Habituellement, ces architectures reposent sur des composants électroniques analogiques compatibles avec les hautes fréquences de fonctionnement. Mais ces technologies sont chères et peu adaptées à l'évolution des normes, des algorithmes ou des protocoles utilisés.

La démocratisation des circuits programmables de type FPGA[1] et leur adéquation aux petites séries ainsi que l'augmentation de leurs performances font qu'ils ne sont plus seulement utilisés à des fins de prototypage avant implantation sur silicium, mais aussi comme cible finale d'une chaîne de conception. Ils présentent de nombreux avantages, en particulier leur faible coût mais aussi le fait d'offrir une capacité d'évolution importante aux systèmes, permettant par conséquent de s'adapter rapidement aux changements de protocoles fréquents dans le domaine des télécommunications. En outre, ils s'intègrent parfaitement dans la chaîne de conception d'un système, où la réutilisation de blocs fonctionnels devient primordiale, avec l'augmentation de la complexité de ceux-ci, et des coûts et temps de développement inhérents.

Cependant, les algorithmes de traitement du signal capables de tirer le meilleur parti des spécificités structurelles de ces composants sont encore peu nombreux. À plus forte raison, les solutions de modulation ou de démodulation conçues à l'origine pour fonctionner sur des systèmes majoritairement analogiques ne sont pas forcément adaptées à

[1] *Field Programmable Gate Array*

ces nouveaux supports, ni à un traitement purement numérique.

Plusieurs tentatives ont été effectuées pour transposer sous forme de circuits numériques les solutions technologiques ayant fait leurs preuves dans le domaine analogique [TIER71, MATT93, CURT00]. Cependant, la transposition des fonctionnalités analogiques en numérique introduit des chemins critiques difficiles à réduire, qui limitent la fréquence de fonctionnement et restreignent l'emploi de tels systèmes aux applications à faible débit. C'est notamment le cas pour les boucles à verrouillage de phase, présentes dans la plupart des modulateurs et démodulateurs analogiques, et dont les fonctionnalités sont très difficiles à transposer efficacement en numérique [CARD02, BELL00, FRAI99].

Lors de l'implantation sur circuit numérique d'algorithmes de traitement du signal, il est nécessaire de prendre en considération l'influence de l'échantillonnage sur la fréquence maximale des signaux que ceux-ci peuvent traiter. Car d'après le critère établi par Nyquist et Shannon, la fréquence d'échantillonnage doit être supérieure au double de la fréquence réelle des signaux que notre circuit va prendre en compte. Or, les circuits considérés fonctionnant selon une logique synchrone, il n'est pas possible, dans le cas d'un traitement séquentiel direct, d'échantillonner à une fréquence supérieure à la fréquence d'horloge du système sur lequel ils sont implantés [SHAN48, BELL02]. Mais les protocoles utilisés dans les communications haut-débit nécessitent le plus souvent des signaux dont la fréquence est bien supérieure à la fréquence d'horloge des circuits numériques qui nous intéressent ici. Pour pouvoir prétendre utiliser ces signaux, il faut donc concevoir des architectures capables de fonctionner à une fréquence aussi proche que possible de la fréquence maximale admise par la technologie numérique employée, et présentant un degré de parallélisme suffisant pour atteindre les débits requis par les protocoles que nous nous proposons de traiter.

C'est ici que la méthodologie traditionnelle qui consiste, en partant d'un modèle ou d'un algorithme fonctionnant parfaitement dans le domaine analogique, à tenter de l'adapter en utilisant les éléments du monde numérique montre ses limites. Le plus souvent, les performances du système résultant sont décevantes, et très en deçà de celles qui peuvent être obtenues avec un système analogique. Car certains éléments fonctionnels analogiques ne peuvent être traduits en numérique que par le biais d'opérations combinatoires ou séquentielles complexes. Plus particulièrement, cette méthodologie introduit des chemins critiques préjudiciables aux performances des architectures numériques. Il apparaît alors qu'une simple transposition n'apporte aucun bénéfice en terme de performances. Il devient par conséquent intéressant de reconsidérer dès la source la méthodologie de conception des circuits numériques pour permettre une traduction directe de la fonctionnalité souhaitée en un algorithme adapté aux circuits numériques, sans passer par une étape analogique susceptible d'introduire des contraintes en contradiction avec la démarche de conception de ces circuits.

Dans cette thèse sont étudiées deux nouvelles architectures adaptées à la parallélisation, compatibles avec l'évolution des débits et des fonctionnalités des protocoles actuels. L'une concerne la modulation, l'autre la démodulation de signaux. L'architecture pour la modulation est basée sur une version modifiée de l'algorithme CORDIC[2], originellement conçu pour la génération itérative de fonctions trigonométriques [VOLD59, VOLD00]. L'architecture de démodulation utilise des registres à décalage à rétroaction linéaire (LFSR[3]) afin de directement mesurer la phase ou la fréquence des signaux qui lui sont fournis. Ces deux architectures fonctionnent sur un FPGA de type Flex10KE200E à la fréquence de 150 MHz et, après parallélisation, permettent d'atteindre un débit d'échantillons de sinusoïdes de 1 GHz, ce qui correspond à une amélio-

[2] *COrdinate Rotation DIgital Computer*
[3] *Linear Feedback Shift Register*

ration d'un facteur 3 par rapport à la fréquence maximale obtenue avec les architectures classiques implantées sur le même support, et jusqu'à un facteur 8 pour le débit binaire admissible en entrée. L'utilisation de logiciels de simulation permettra de nous assurer de la validité de chaque algorithme, puis leur description en langage VHDL et leur implantation sur un système cible permettra d'évaluer leurs performances.

Le premier chapitre pose les bases et fondements d'un système de transmission, et présente son architecture générale, en insistant sur les spécificités des systèmes purement numériques. Ensuite, il met en lumière les principes fondamentaux du traitement du signal, comme le théorème de l'échantillonnage qui influe sur notre méthodologie de conception d'une architecture entièrement numérique. Nous présentons également les solutions les plus utilisées dans les architectures analogiques, en nous focalisant sur les difficultés de leur transposition vers des systèmes entièrement numériques, afin d'extraire une nouvelle méthode pour nous affranchir des limitations introduites par la transposition des architectures analogique sur circuit numérique.

Le second chapitre porte sur la réalisation d'une architecture purement numérique pour un modulateur. Les différentes méthodes permettant de générer un signal modulé y seront présentées, et deux d'entre elles seront plus particulièrement au centre de nos attentions. Une première méthode, utilisant des échantillons de sinusoïde pré-calculés, est souvent employée car très simple à mettre en oeuvre. Une version hybride, mettant en oeuvre des filtres numériques interpolateurs est également examinée. Enfin, notre nouvelle architecture construite sur la base de l'algorithme CORDIC est présentée et confrontée aux deux précédentes. L'implantation sur circuit FPGA de ces solutions permet de mettre en évidence la meilleure adéquation de cette dernière architecture aux systèmes numériques qui constituent notre cible.

Parmi les différentes architectures de démodulation les systèmes utilisant des boucles

à verrouillage de phase (PLL[4]) sont traditionnellement employés dans le monde analogique. Dans le cadre d'une implantation sous forme d'architecture synchrone, ce type de système présente plusieurs limites, qui ne permettent pas son intégration dans des systèmes demandant un haut débit de données. Par conséquent je propose dans le troisième chapitre une nouvelle architecture parallèle utilisant plusieurs bancs de registres à décalage. Son adéquation avec les systèmes numériques est mise en valeur, sur la base de comparaisons avec des architectures de démodulation existantes.

[4] *Phase Locked Loop*

Chapitre 1

Transmission numérique de l'information

Ce chapitre présente en premier lieu une chaîne de transmission de signaux numériques, afin de situer et préciser l'importance des fonctions de modulation et de démodulation, replacées dans leur contexte. Ces deux éléments se situent à des positions déterminantes (au deux extrémités d'un canal de transmission) ce qui implique qu'ils sont les plus sujets à des évolutions importantes lorsque le débit imposé par les protocoles de communication ou la fréquence du signal transportant les informations augmentent.

Pour comprendre et aborder les difficultés que doivent surmonter les architectures numériques lors de la production d'un signal à haute fréquence, les principes régissant la conversion entre le monde numérique et le monde analogique, tels que le théorème de l'échantillonnage, sont rappelés. De là sont déduits les critères constituant les contraintes de conception pour des modulateurs ou démodulateurs numériques équivalents à leurs contreparties analogiques. Les avantages des architectures *tout numérique*

pour les systèmes de transmission sont mis en avant en particulier en terme d'évolution, de coût et de ressources. À cet égard, l'architecture générale et le fonctionnement d'un circuit de type FPGA sont analysés, ainsi que leur adéquation avec les algorithmes utilisés pour la modulation et la modulation de signaux.

1.1 Chaîne de transmission de l'information

1.1.1 Introduction

Les systèmes de transmission numérique de l'information véhiculent des données entre deux entités élémentaires que sont la source et le destinataire. Ces données circulent par le biais d'un support physique qui peut être un câble, de la fibre optique ou un faisceau hertzien [TANE03, PUJO03].

1.1.2 Les modèles de référence

Des efforts de modélisation ont été effectués pour permettre de séparer les différents niveaux de fonctionnalité d'un système de traitement numérique de l'information, plus particulièrement dans le cadre de transmissions réseau. Il existe trois principaux modèles :

- le modèle OSI[1] est souvent employé pour décrire cette segmentation, bien que peu de protocoles réels soient effectivement basés sur lui [TANE03] ;
- le modèle TCP/IP[2], antérieur au précédent, dispose d'une architecture modulaire ressemblant à celle d'OSI. Il est à la base du réseau mondial Internet [COME00] ;
- enfin, le modèle UIT-T[3] est né pour satisfaire aux besoins spécifiques des réseaux à haut-débit [GAGN01]. Il est plus particulièrement utilisé dans les réseaux de nouvelle génération [PUJO03].

Le modèle OSI

Il se compose de sept couches distinctes, et est présenté sur la figure 1.1.

[1] *Open System Interconnection*
[2] Transmission Control Protocol / Internet Protocol
[3] Union Internationale des Télécommunications

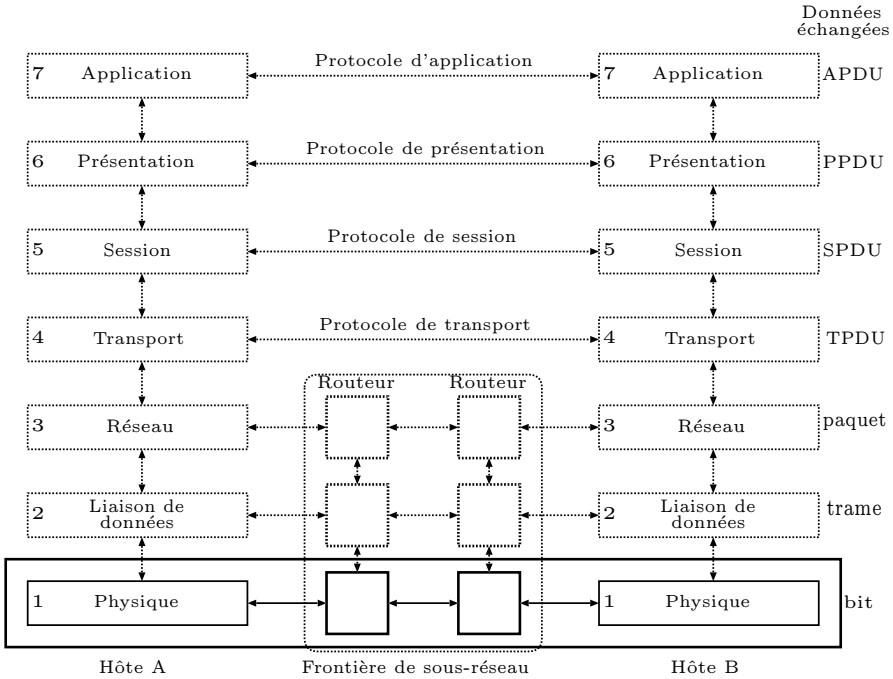


FIG. 1.1: Modèle de Référence OSI

Le modèle OSI décrit des niveaux de transmission mais non les protocoles proprement dits. Il divise l'ensemble des protocoles en sept couches indépendantes entre lesquelles sont définis deux types de relations : les relations verticales entre les couches d'un même système (interfaces) et les relations horizontales relatives au dialogue entre deux couches de même niveau (les protocoles). Les couches 1, 2, 3 et 4 sont orientées transmission et les couches 5, 6 et 7 sont orientées traitement.

- La couche *application* (7) fournit les services et interfaces de communication aux utilisateurs ;
- La couche *présentation* (6) s'occupe de la syntaxe des données. Cette couche permet à deux machines de communiquer même lorsqu'elles utilisent des représentations de données différentes. Elle gère des structures de données haut niveau pour accomplir cette tâche ;

- La couche *session* (5) permet d'établir une connexion logique entre deux applications. Elle assure l'organisation et la synchronisation du dialogue ;
- La couche *transport* (4) permet l'établissement, le maintien et la rupture des connexions. L'une des tâches principales de cette couche est d'accepter des données de la couche supérieure et de les diviser en unités plus petites : il s'agit de l'opération de fragmentation. Elle offre un service réel de bout en bout de la source à la destination, indépendant du chemin effectif utilisé entre les machines. Dans les couches plus basses, les protocoles établissent des relations entre une machine et ses voisins immédiats et non entre les machines source et destination. En effet, les couches 1 à 3 sont chaînées alors que les couches 4 à 7 sont de bout en bout ;
- La couche *réseau* (3) assure la commutation et le routage des paquets entre les nœuds du réseau. Pour le modèle OSI, il existe deux méthodes principales d'acheminement : la commutation de circuits et la commutation de paquets. C'est cette couche qui gère les congestions sur les nœuds du réseau ;
- La couche *liaison* (2) a pour but de transmettre les données sans erreur. Elle décompose les données de l'émetteur en trames de données puis les envoie de façon séquentielle. Différentes méthodes permettant de protéger les données contre les erreurs sont utilisées, comme le codage de détection et de correction d'erreur. Dans les réseaux à canal partagé, cette couche s'occupe aussi de contrôler l'accès au canal par l'intermédiaire d'une sous-couche MAC[4] ;
- La couche *physique* (1) s'occupe de la connexion physique sur le réseau. Elle se charge de la transmission de bits à l'état brut sur un canal de transmission. Les problèmes de conception de cette couche concernent les interfaces électriques, la synchronisation, ainsi que les spécifications du support physique de transmis-

[4] *Medium Access Control*

sion. En mode émission, elle est chargée de s'assurer de la bonne transmissions de données fournies par la couche liaison qui la précède sur le canal de transmission. En mode réception, l'intégrité des données reçues est prise en compte. La gestion d'éventuels mécanismes de synchronisation et de communications bidirectionnelles est dévolue à la couche physique, le cas échéant. La transition entre la représentation des informations sous forme de signal, puis sous forme de bits, est effectuée dans cette couche. C'est à ce niveau de fonctionnement que nous nous intéressons plus particulièrement dans la suite, car c'est dans celui-ci qu'interviennent en particulier les opérations de modulation et de démodulation [TANE03].

Le modèle d'abstraction OSI permet donc de situer les opérations de modulation et de démodulation dans un système de transmission d'information pouvant éventuellement s'inscrire dans le cadre plus global d'un réseau à grande échelle. Les étapes de traitement effectuées dans cette couche physique sont exposées dans la section 1.1.3.

Le modèle TCP/IP

Au contraire d'OSI, TCP/IP est né après les protocoles qu'il modélise, et il ne fait que peu de distinctions entre les concepts de services, d'interfaces ou encore de protocoles [BUSB00]. Par conséquent, la segmentation en couches indépendantes d'OSI n'est pas présente de façon aussi stricte dans TCP/IP. Le modèle TCP/IP permet simplement de positionner les protocoles existants et futurs dans un cadre théorique.

- La couche *application* de ce modèle contient entre autres tous les protocoles de haut niveau, comme Telnet ou FTP[5] ;
- la couche *transport* permet aux applications d'échanger des données indépendam-

[5] *File Transfer Protocol*

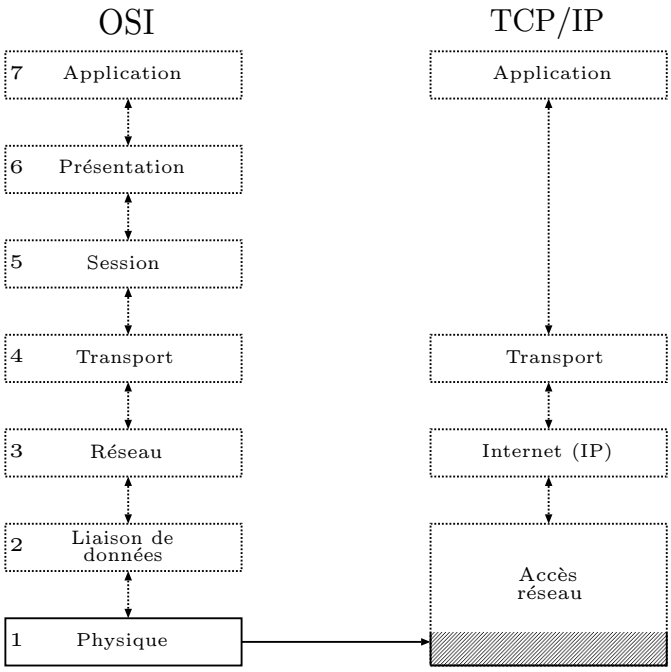


FIG. 1.2: Modèle de référence TCP/IP en parallèle avec le modèle OSI

ment du réseau utilisé, grâce aux protocoles TCP et UDP[6] ;

– la couche *internet*, ou réseau, est la pierre angulaire de cette architecture, et permet aux hôtes d'envoyer des paquets élémentaires indépendants les uns des autres, sans se préoccuper des détails concernant leur acheminement vers l'hôte destination ;
– enfin la couche *d'accès au réseau* est mal définie par le protocole. Elle regroupe tous les éléments nécessaires pour accéder à un réseau physique, quel qu'il soit. Elle contient en particulier les spécifications concernant la transmission de données sur le réseau physique, tout comme la première couche du modèle OSI.

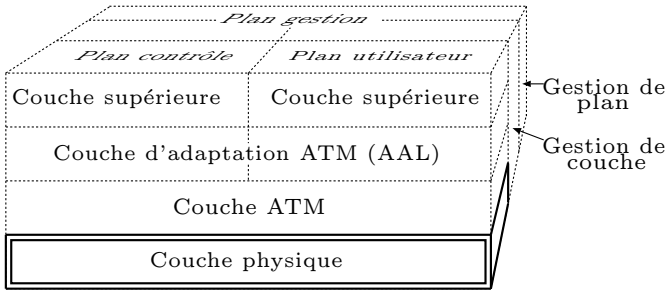


FIG. 1.3: Modèle de référence UIT-T

Le modèle UIT-T

L'architecture fonctionnelle du modèle UIT-T est présentée sur la figure 1.3, dans le cas d'ATM[7]. Il a été spécifiquement conçu pour permettre aux nouveaux réseaux de prendre en compte les applications multimédia, ce qui se traduit par l'existence de trois «plans» se partageant la même ressource physique par multiplexage. Il utilise de petits paquets de longueur fixe de type ATM appelés cellules [KOFM99]. Il présente de nombreuses similarités avec le modèle OSI, en particulier :

- La couche AAL[8] gère l'interface avec les couches supérieures et regroupe une partie des fonctionnalités de la couche 4 du modèle OSI (il lui manque les opérations de fragmentation et réassemblage) ;
- La couche ATM est responsable de l'acheminement des cellules de l'émetteur au destinataire. Elle est équivalente à la couche 3 du modèle OSI ;
- La gestion du protocole dépendant du support physique est confiée à la couche physique de ce modèle. Elle est donc responsable de la transmission au niveau bit des informations. Celle-ci s'occupe en plus de la reconnaissance des paquets ATM. Elle est donc équivalentes aux couches 1 et 2 du modèle OSI.

[6] *User Datagram Protocol*
[7] *Asynchronous Transfer Mode*
[8] *ATM Adaptation Layer*

1.1.3 Organisation fonctionnelle de la couche physique

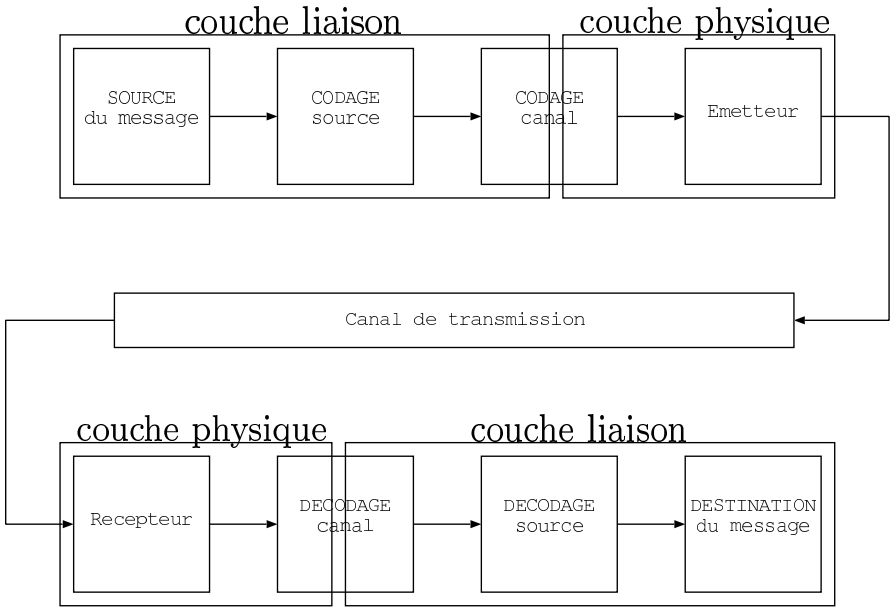


FIG. 1.4: Représentation fonctionnelle des couches physique et liaison d'un système de transmission comprenant une paire émetteur - récepteur

La figure 1.4 permet de présenter les fonctions de base d'un système de transmission numérique remplissant le rôle d'interface entre le signal continu adapté au médium de transmission et les informations binaires transmises aux couches supérieures. Cette figure présente à la fois des fonctions appartenant à la couche physique, et d'autres à la couche liaison de données. Mais certaines, comme le codage canal peuvent appartenir aux deux niveaux simultanément.

La source du message correspond aux données fournies par la couche liaison de l'émetteur, et réciproquement, la destination du message correspond à la couche liaison du destinataire se trouvant à l'autre extrémité de la ligne de transmission.

Le codeur source peut éventuellement modifier les éléments binaires à transmettre. Ceci est par exemple utile pour appliquer des opérations de compression permettant de réduire le nombre de bits effectivement présents dans la séquence à envoyer vers le ré-

cepteur. Cette étape repose en grande partie sur une bonne connaissance des paramètres statistiques du signal transmis.

Le codage de canal permet entre autres d'adapter cette séquence de manière à lui offrir une meilleure résistance aux perturbations et autres bruits qui peuvent intervenir lors du transit sur le canal de transmission. Ce codage intervient sous la forme d'algorithmes à base de codes, comme les codes cycliques, le code de Hamming ou encore les turbo-codes, et constitue une fonctionnalité de la couche liaison [BADR02, HANZ02].

Les données numériques à émettre sont présentes sous la forme de grandeurs abstraites n'ayant de signification logique que pour le système numérique sur lequel elle sont produites. Il est donc nécessaire de leur associer une représentation physique concrète, qui peut se matérialiser par exemple, sous forme d'un signal électrique. La fonction de l'émetteur correspondante est la modulation, qui associe à chaque groupe de n éléments du message source un signal de durée $T = n \cdot T_s$ [FRAI99]. Cette association s'effectuer au niveau de l'amplitude, de la fréquence, ou bien de la phase du signal transmis. Il est également possible d'utiliser une combinaison de plusieurs de ces paramètres. Le nombre et les genres de modifications qu'il est possible d'apporter au signal de sortie sont définis par le type de modulation choisie. Si les données sont groupées par paquets de n bits, il y aura en tout $M = 2^n$ états possibles. L'émetteur est également chargé d'effectuer une fonction de filtrage, et le cas échéant de transposition de fréquence afin d'éliminer les fréquences parasites, et de centrer le signal modulé autour d'une fréquence centrale souhaitée compatible avec le support de transmission [PUJO03].

Le figure 1.5 permet de mettre en valeur la séparation entre les deux types de traitements qui sont effectués sur une ligne de transmission numérique. Le bloc ETTD[9] correspond à l'ensemble des fonctionnalités qui ne sont pas directement liées à la pro-

[9]Équipement Terminal Transmetteur de Données

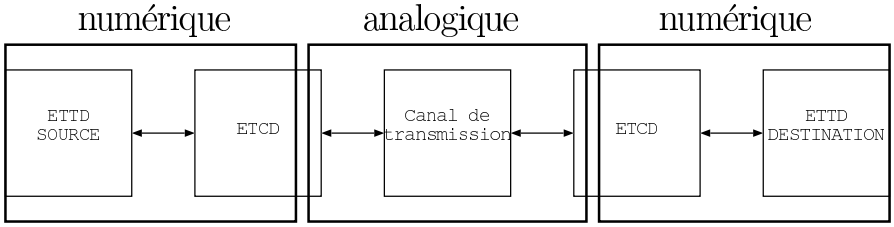


FIG. 1.5: Représentation d'une ligne de transmission, séparation analogique numérique

duction des signaux à transmettre sur le canal. Le bloc ETCD[10] quant à lui est chargé de la fonction de transformation des signaux binaires en signaux numérique.

L'ETCD est souvent désigné par le terme de "modem", contraction de modulateur / démodulateur [STEI91]. L'adaptation au support nécessite souvent la modulation et la démodulation d'une fréquence porteuse. Car le signal produit par la source dans sa forme brute a une occupation spectrale infinie, et le transmettre sans modification, en bande de base, n'est souvent pas compatible avec le canal de diffusion. Il est nécessaire de lui apporter des modifications pour qu'il corresponde aux caractéristiques de ce canal. L'opération de modulation vise en particulier à transposer la bande de fréquences occupée par ce signal dans une plage adaptée au médium. Pour une liaison numérique bi-directionnelle simultanée ou à l'alternat, il est nécessaire que chaque ETCD prenne en charge les deux fonctions. Les modems disposent d'une interface analogique avec le canal de transmission, et d'une interface numérique avec l'ETTD auquel ils sont associés. Le passage d'un mode de gestion numérique des données à un mode analogique s'effectue dans l'ETCD et, en ce qui concerne les architectures traditionnelles, en amont de la fonction de modulation. Cette transition est effectuée à l'aide de convertisseurs analogique / numérique.

[10]Équipement Terminal de Circuit de Données

1.1.4 Atouts et contraintes d'un traitement numérique

Une partie importante de l'équipement de transmission de données est réalisée à l'aide de circuits ou de composants analogiques [FRAI99]. Les architectures qui sont implantées de cette façon atteignent des fréquences de fonctionnement importantes. Mais ce sont surtout les avancées technologiques et physiques qui permettent de repousser plus haut encore ces vitesses. Par contre, les circuits considérés ne se prêtent pas à une évolution des protocoles ou des modulations liées, et il est nécessaire de produire de nouveaux circuits à chaque changement de norme.

Les circuits FPGA, tout comme les ASIC, fonctionnent à l'aide de signaux échantillonnés et doivent se conformer aux contraintes de l'échantillonnage que se traduisent en particulier les critères de Nyquist et Shannon, lesquels seront étudiés un plus loin dans la section 1.2. Ces contraintes font que les circuits numériques ne sont pas aussi rapides que les circuits analogiques correspondants, et leurs performances brutes s'en ressentent. Par contre, ils disposent d'autres avantages :

- il n'y a pas de dérive des caractéristiques en fonction des conditions extérieures ;
- la précision des informations traitées est garantie par le nombre de bits choisi pour représenter les nombres ;
- il est possible d'implanter directement des fonctions de filtrage avancées qui ne sont pas accessibles dans le domaine analogique ;
- l'intégration avec les étapes de codage ou de préparation des signaux est bien meilleure.

Pour pouvoir traiter des signaux dans une bande passante importante, il est nécessaire de disposer d'une vitesse de calcul élevée [GLAV96]. Les circuits de type ASIC bénéficient de technologies de fabrication autorisant des performances relativement élevées. Mais leur structure interne est figée : ils ne possèdent donc pas les mêmes possibilités

d'évolution que les circuits FPGA. Ces derniers disposent d'atouts spécifiques :

- les possibilités de prototypage sont infinies, et la simulation logicielle aisée ;
- il est possible de modifier l'architecture de traitement, ou le type de modulation utilisée pour s'adapter à de nouveaux protocoles sans changer de composant ;
- les coûts inhérents au développement d'applications, et ceux des composants sont très inférieurs à ceux des ASIC.

1.2 Transmission d'un signal numérique en bande transposée

La transmission d'un signal numérique peut être faite en bande de base[11], en utilisant un simple codage adapté. Cependant, l'encombrement spectral de tels signaux est théoriquement infini, bien qu'il soit possible de le réduire à un intervalle fini à l'aide d'un filtrage adapté [BELL02]. Mais la bande de fréquence occupée par un signal en bande de base dépend directement du type de codage utilisé et du débit de symboles binaires choisi. Ce type de signal est particulièrement sensible aux phénomènes d'atténuation et de distorsion lesquels sont d'autant plus prononcés que la distance entre l'émetteur et le récepteur est grande. La distorsion limite à terme la fréquence de transmission possible en bande de base [STEI91].

Les communications à haut débit utilisent un autre mode de transmission, bien plus efficace, mais qui nécessite d'adapter le signal à l'aide d'une modulation [IEEE99, GRAS01]. Cette opération a pour but de produire un signal dont le spectre est transposé pour le rendre conforme à la bande passante du support de transmission. Elle permet aussi le multiplexage fréquentiel permettant à plusieurs systèmes d'accéder simultané-

[11] C'est-à-dire sans modifier la bande de fréquence occupée par signal après son codage.

ment à un même canal [FRAI99].

1.2.1 Échantillonnage des signaux

L'échantillonnage est l'opération qui permet de représenter de façon discrète un signal continu dans le temps. Le signal échantillonné correspond aux valeurs du signal original, prélevées à des intervalles de temps fixes, multiples d'une durée donnée et qui est appelée période d'échantillonnage T_e.

Le théorème de l'échantillonnage [SHAN48] présente les conditions dans lesquelles un signal continu peut être complètement représenté par la suite discrète de ses échantillons. En d'autres termes, un signal est complètement représenté si à partir de ses échantillons prélevés à intervalles T_e, il est possible de reconstruire l'intégralité du signal initial.

Ce théorème s'exprime de la façon suivante : un signal dont le spectre ne contient aucune composante de fréquence supérieure ou égale à une fréquence maximale f_{max} peut être entièrement représenté par la suite de valeurs constituée par ses échantillons, à condition que ceux-ci soient prélevés à des intervalles régulièrement espacés, dont la durée T_e n'excède pas

$$T_e = \frac{1}{2 f_{max}}. \tag{1.1}$$

On peut alors définir la fréquence d'échantillonnage par $f_e = \frac{1}{T_e}$.

1.2.2 Cas des signaux sinusoïdaux

L'opération d'échantillonnage présente un intérêt tout particulier en ce qui concerne les signaux sinusoïdaux. Car c'est de cette opération que découlent les architectures d'un grand nombre de générateurs de signaux numériques : en utilisant une quantité

limitée d'échantillons stockés dans des mémoires, ils est possible de reconstruire un ensemble complet de signaux de fréquence variée [BELL02].

Soit le signal sinusoïdal suivant :

$$s(t) = \cos(\omega t + \phi). \tag{1.2}$$

L'échantillonnage de s conduit à la production de la suite $s(n)$ telle que

$$s(n) = \cos(\omega n + \phi), \text{ avec } T_e = \frac{1}{f_e} \text{la période d'échantillonnage.} \tag{1.3}$$

1.2.3 Quantification d'un signal

Un signal échantillonné se présente comme une suite d'impulsions dont les amplitudes peuvent prendre une infinité de valeurs distinctes. La quantification consiste à réduire le nombre de valeurs que peut prendre cette amplitude, en attribuant à chaque échantillon une valeur choisie parmi un ensemble fini de valeurs prédéterminées. Ces valeurs sont les niveaux de quantification. Tous les échantillons ayant une amplitude réelle différente, mais se trouvant dans une même plage d'échelonnage se verront attribuer la même valeur de quantification.

La quantification d'un signal introduit une erreur, qui produit du bruit de haute fréquence lors de la reconstruction du signal analogique à la fin du traitement numérique. Ce bruit de quantification $\varepsilon_q(t)$ correspond à l'équation :

$$\varepsilon_q(t) = m(t) - m_q(t), \tag{1.4}$$

où $m(t)$ correspond au signal source, et $m_q(t)$ au signal quantifié.

De plus, certaines techniques de modulation utilisent des approximations de la va-

leur de quantification des échantillons pour simplifier les calculs sur ceux-ci ou pour réduire la surface occupée par une fonction logique. Ces approximations peuvent en partie être absorbées par le filtre passe-bas en sortie de l'étage numérique. Celui-ci permet, entre autres, d'éliminer une partie des fréquences indésirables. Cependant, plus les erreurs et approximations sont élevées, plus l'ordre du filtre de correction doit être grand. Ceci peut engendrer un coût important, d'autant plus que dans le domaine des transmissions haut débit, ce filtre doit pouvoir fonctionner avec des signaux de fréquence très élevée.

1.2.4 Production numérique de signaux en bande transposée

La génération de signaux en bande transposée consiste, à partir d'un signal numérique ou analogique donné, à transposer sa fréquence centrale pour que son spectre se retrouve dans une bande de fréquence bien définie. Cette bande de fréquence dépend principalement des caractéristiques physiques du canal de transmission. Les protocoles utilisés et les formes de modulations qu'ils préconisent sont conçus pour se conformer au maximum à ces caractéristiques [BELL02, GLAV96, OFDM00, BREN96].

La production à l'aide d'architectures numériques de signaux en bande transposée doit se conformer aux règles de l'échantillonnage et de la quantification. Les performances d'un système numérique dédié à par exemple la modulation seront donc dépendantes :

- de la vitesse maximum de traitement des échantillons numériques ;
- de la précision avec laquelle les calculs sur ces échantillons sont faits.

Un système numérique fonctionnant à une vitesse d'horloge élevée permettra donc d'atteindre un débit d'échantillons important. Plus la résolution en nombre de bits des échantillons est importante, plus les opérations arithmétiques qui sont effectuées sur ces

échantillons pour produire un signal modulé, ou pour le démoduler, ont une influence néfaste sur la vitesse de fonctionnement du circuit numérique [GLAV96].

Pour obtenir les meilleures performances, il est donc primordial d'utiliser autant que possible des opérateurs et des fonctions adaptées spécifiquement à l'architecture numérique cible.

1.3 Architectures analogiques pour la modulation et la démodulation numériques

Dans cette section, nous présentons la structure d'un démodulateur QAM analogique et du démodulateur associé. Bien que la modulation QAM ne soit qu'une des méthodes existantes pour transmettre un signal numérique sur une fréquence porteuse, elle illustre, de par la complexité des opérateurs qu'elle met en œuvre, les contraintes que pourra poser à terme son implantation sur une architecture numérique.

1.3.1 Modulateur QAM analogique

Une modulation QAM peut être produite de deux façons distinctes :

- elle peut être considérée comme une modulation d'amplitude et de phase, et combine alors les techniques utilisées pour ces deux types de modulation ;
- il est également possible de moduler en amplitude deux porteuses en quadrature, puis de reconstituer le signal résultant à l'aide d'un additionneur. C'est cette technique qui est la plus fréquemment utilisée [FRAI99] et qui est présentée ci-après.

Le modulateur correspondant est présenté sur la figure 1.6. Il est notamment constitué d'un bloc de codage qui associe à chaque série de M bits en provenance de l'émet-

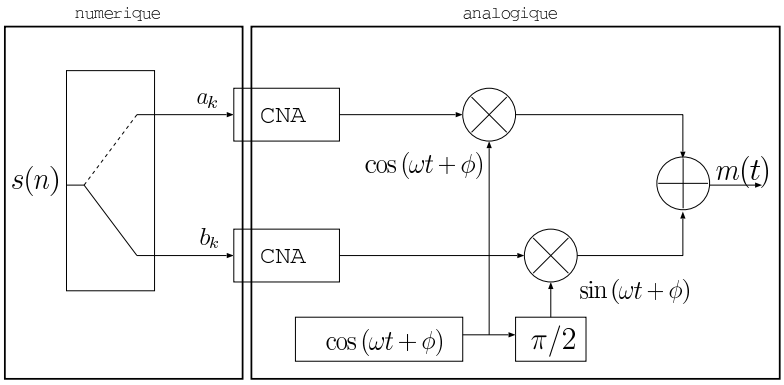


FIG. 1.6: Architecture d'un modulateur QAM analogique

teur un couple de valeurs correspondant aux amplitudes des composantes sinusoïdales en phase et en quadrature.

Les fonctions de génération des sinusoïdes, représentées par le bloc $\cos(\omega t+\phi)$ sont le plus souvent implantées à l'aide de boucles à verrouillage de phase, ou à l'aide de filtres d'ordre élevé. Les termes d'amplitude sont appliqués à l'aide de deux multiplieurs, l'un pour la composante en phase, l'autre pour la composante en quadrature. Les deux sinusoïdes sont alors combinées à l'aide d'un additionneur pour produire le signal modulé à émettre $s(t)$.

1.3.2 Démodulateur QAM analogique

Le démodulateur QAM correspondant est présenté sur la figure 1.7. Deux chemins de calculs sont utilisés, l'un pour la composante en phase, l'autre pour celle en quadrature. Plus précisément, les deux composantes sont séparées grâce aux multiplications par $\cos(\omega t+\phi)$ et $\sin(\omega t+\phi)$. Les blocs de prise de décision sont ensuite chargés d'effectuer une discrimination parmi l'ensemble discret des différentes valeurs possibles d'amplitude. La valeur résultante est communiquée à un bloc numérique qui reconstitue la suite de bits correspondante.

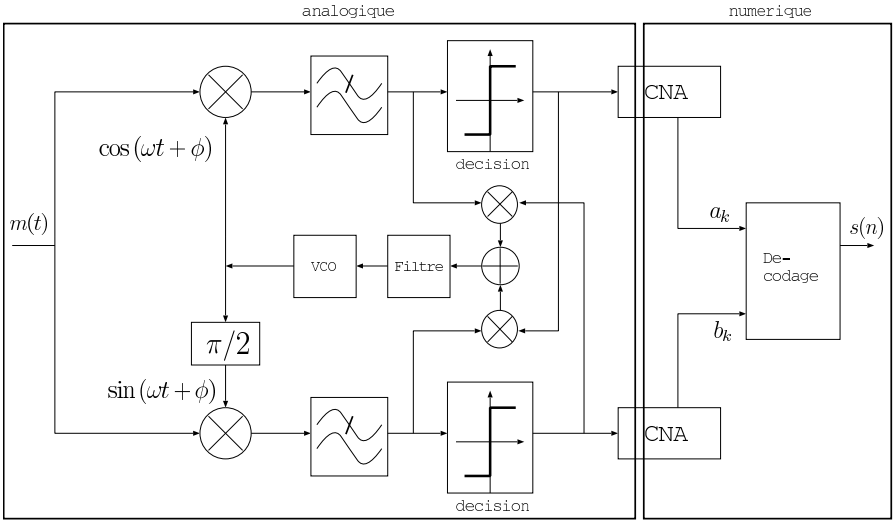


FIG. 1.7: Architecture d'un démodulateur QAM analogique

La structure du démodulateur est beaucoup plus complexe que celle du modulateur QAM, en particulier parce que le spectre des signaux modulés QAM ne contient aucune raie à la fréquence de l'onde porteuse. La boucle utilisée dans cet exemple pour récupérer la phase de l'onde porteuse, est appelée boucle de Costas. Elle est composée de deux multiplieurs, d'un additionneur, d'un filtre et d'un oscillateur contrôlé en tension (VCO[12]). La sinusoïde en phase correspondant à l'onde porteuse est reconstituée par interpolation et corrections successives à partir des valeurs numériques produites par les deux blocs de décision. Deux filtres passe-bas placés en amont de ces blocs de décision permettent de supprimer les composantes à fréquence élevées qui subsistent après la multiplication.

1.3.3 Incompatibilités avec le traitement numérique

Aussi bien dans le démodulateur que dans le modulateur, on trouve des fonctionnalités dont la mise en œuvre sur un circuit numérique limite excessivement la vitesse de

[12] Voltage Controled Oscillator

fonctionnement, tout en occupant une surface ou un nombre de cellules logiques importants. Les difficultés sont posées principalement par le générateur de sinusoïdes, représenté sous la forme d'un bloc $\cos(\omega t + \phi)$ sur la figure 1.6, et par l'oscillateur contrôlé en tension, noté *VCO* sur la figure 1.7. Dans un circuit numérique utilisant une logique synchrone le chemin le plus lent entre deux registres est appelé «chemin critique». Or, l'ensemble des opérations arithmétiques effectuées dans ces boucles se traduit par un chemin critique très long si elles sont implantées sur une architecture numérique. La fréquence de fonctionnement d'une architecture numérique dépend essentiellement de la taille des chemins critiques [TIER71]. Une architecture numérique utilisant ce type de boucle serait sans doute contrainte de recourir à un composant analogique externe pour permettre des performances acceptables.

De plus, les multiplieurs utilisés pour supprimer les porteuses en quadrature et en phase dans le démodulateur, ou pour produire ces même composantes dans le modulateur, doivent fonctionner à une fréquence élevée pour garantir une précision suffisante au signal de sortie et être compatibles avec des fréquences d'échantillonage élevées.

1.4 Circuits FPGA dans le cadre de la modulation et la démodulation

Les nouvelles génération de circuits FPGA fournissent sur une même puce l'équivalent de plusieurs centaines de milliers de portes logiques, ainsi qu'un certain nombre de bancs de mémoire vive [FLEX03, STRA05] Les avantages des FPGA sur les systèmes de type ASIC[13] sont doubles :

– ils proposent tout d'abord un flot de conception plus court ne nécessitant pas

[13] *Application Specific Integrated Circuit*

d'étape industrielle de production de circuit ;

– ensuite, il est possible de passer directement du prototype au produit final en utilisant dans les deux cas la même plate-forme.

D'un autre côté, les ASICs permettent d'atteindre des performances supérieures, en particulier en ce qui concerne les fréquences de fonctionnement, et les fonctionnalités qu'il est possible d'y implanter. Cependant, les FPGA s'avèrent être une solution de compromis intéressante pour systèmes produits en petit volume.

	ASIC	FPGA	CPU ou DSP [PAGE92, PHIL98]
Performances	Très élevées	Élevées	Faibles
Taille	Faible	Moyenne	Élevée
Consommation	Faible	Moyenne	Très Élevée
Intégration	Système sur une puce	Système sur une puce	Composants annexes nécessaires
Souplesse	Fonctions figées	Reconfigurable	Programmable
Mise en œuvre	Complexe	Moyennement complexe	Faiblement à moyennement complexe
Coût composants	Très élevé	Moyen	Faible

TAB. 1.1: Comparaison des FPGA par rapport aux autres architectures de traitement du signal [MESU05]

Le tableau 1.1 présente les avantages et inconvénient respectifs des architectures les plus couramment utilisées pour le traitement numérique du signal. Les circuits FPGA apparaissent comme un bon compromis, tant du point de vue de leur prix de revient limité que de leurs performances leur permettant d'être intégrés dans les architectures de traitement du signal à haut-débit. Les sections suivantes présentent à titre d'exemple l'architecture interne d'un FPGA Altera FLEX10KE200[FLEX03], qui est le circuit cible pour l'ensemble des architectures proposées dans ce document.

1.4.1 Architecture interne

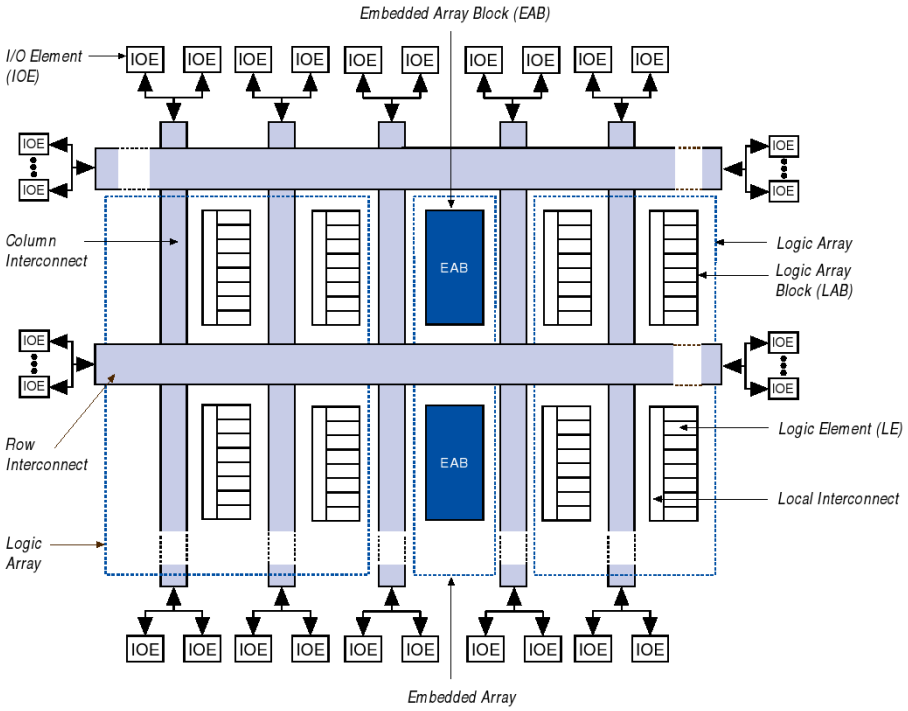


FIG. 1.8: Architecture globale d'un FPGA de type FLEX10KE

La figure 1.8 présente l'architecture interne d'un circuit FPGA de type Flex10KE, produit par Altera. Cette structure est composée de trois types d'élément de base, ainsi que de structures d'interconnexion et de cellules gérant les entrées-sorties (IOE[14]). Les trois éléments de bases sont :

- les éléments logiques élémentaires, notés LE[15] ;
- les matrices d'éléments logiques élémentaires, notés LAB[16] ;
- des blocs de mémoire versatiles, notées EAB[17].

[14] *Input/Output Element*
[15] *Logical Element*
[16] *Logical Array Bloc*
[17] *Embedded Array Block*

Problématique

La structure interne des FPGA est une contrainte pour le développement au niveau RTL[18] d'architectures adaptées. Mais le concepteur ne choisit pas la façon d'implanter les fonctionnalités dans les EAB, LAB ou LE. Ceci est la tâche du logiciel de synthèse : il se charge de répartir dans les ressources disponibles sur le FPGA ciblé les différents éléments logiques composant l'architecture de l'application développée.

Cependant, une bonne connaissance du type et de la quantité de ressources disponibles permet d'orienter les stratégies de conception pour choisir les fonctions les plus aptes à être implantées efficacement [CARD02, CARD03, DICK02, UKEI93, VALL04].

Les blocs de mémoire EAB

Ils sont composés d'éléments de mémoire basiques, auxquels sont adjoints des ports pour de gérer les entrées et les sorties. Ces ports disposent en plus de registres, ce qui permet d'utiliser ces mémoires pour implanter des fonctions logiques potentiellement complexes. La taille de ces blocs permet d'y intégrer par exemple des multiplieurs ou des circuits de correction d'erreur. Dans le cas de fonctions logiques, un schéma correspondant à une LUT[19] est implanté au moment de la configuration du FPGA dans ces blocs de RAM, ce qui permet de calculer les fonctions de logique combinatoire par une recherche directe des résultats dans la table, plutôt que de les évaluer à l'aide de portes logiques. Cette technique permet un gain de vitesse pour certaines fonctions logiques complexes.

Les blocs de mémoire peuvent aussi être utilisés sous forme plus classique pour supporter les fonctions de mémoire vive ou morte.

[18] *Register Transfer Level*
[19] *Look Up Table*

Ces blocs permettent d'obtenir de bonnes performances pour des fonctions logiques relativement complexes, mais :

- ils sont disponibles en nombre limité (24 sur un FLEX10K200E par exemple). Une fois l'ensemble des EAB occupés, les fonctions complexes seront implantées à l'aide de LE ;
- la complexité des fonctions pouvant être «pré-calculées» dans un EAB est limitée (8 entrées et 16 sorties binaires pour un EAB dans un FLEX10K200E). La réalisation de fonctions plus complexes nécessite de combiner plusieurs blocs. Dans ce cas, des délais dûs aux interconnexions apparaissent.

Il est possible de prévoir dès l'étape de conception de restreindre au maximum le nombre de fonctions complexes constituant notre architecture afin d'éviter de dépasser ces limites.

1.4.2 Les blocs logiques LAB

Ils sont composés de huit blocs logiques élémentaires de type LE. La figure 1.9 présente l'architecture interne de l'un de ces blocs. Ils comportent également des chemins optimisés pour la propagation de retenues, utiles en particulier dans le cadre de l'implantation de fonctions arithmétiques comme les additionneurs. De plus, des chemins d'interconnexion rapides et simplifiés entre les différents LE d'un même LAB sont disponibles. Des signaux de contrôle programmables sont fournis et peuvent servir à acheminer un signal d'horloge ou de type reset asynchrone.

Les LAB facilitent la combinaison de plusieurs LE et donc la réalisation de fonctions combinatoires ou séquentielles plus complexes. En profitant des chemins de propagation intégrés, les fonctions implantées dans un seul LAB bénéficient de délais de propagation réduits entre les différents LE intégrés. Mais un LAB ne regroupe qu'un nombre

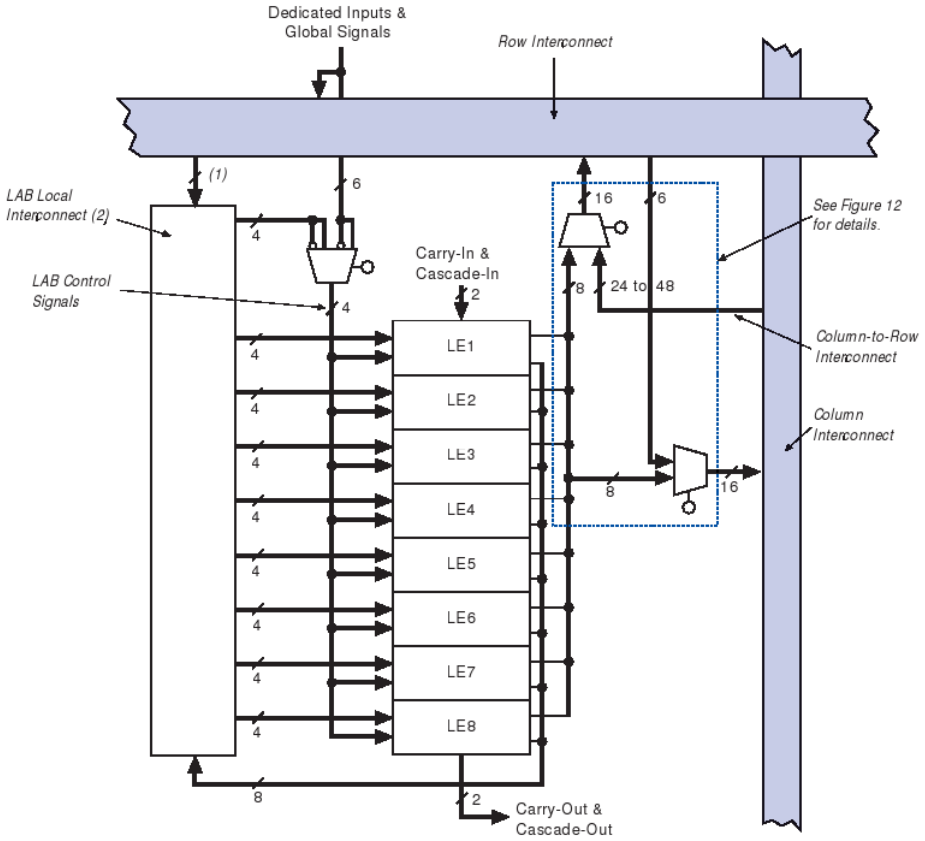


FIG. 1.9: Architecture d'un bloc logique

restreint de LE (8 sur l'architecture FLEX10KE). Les fonctions trop complexes, ou qui nécessitent un trop grand nombre d'entrées/sorties sont donc implantées sur plusieurs LAB distincts, et les délais de propagation entre cellules logiques s'en trouvent dégradés.

Le plus petit élément structurel de cette architecture est l'élément logique décrit dans la section suivante.

1.4.3 Architecture d'une cellule logique

Les éléments logiques LE correspondent à la plus fine granularité pour le type de FPGA décrit ici. Leur structure est présentée sur la figure 1.10. Ils sont composés d'une

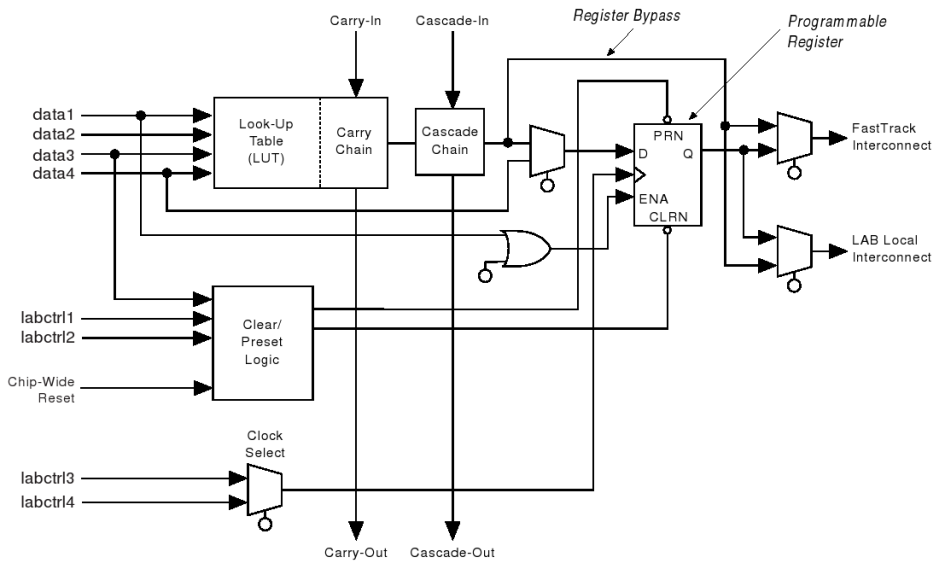


FIG. 1.10: Architecture d'une cellule logique

table LUT à quatre entrées permettant de réaliser des fonctions combinatoires à quatre variables. Le second élément important est un registre programmable, autorisant avec la table LUT la gestion de fonctions de logique synchrone. Enfin, des chemins spécifiques sont intégrés. Ils sont dédiés à la propagation des retenues des opérations arithmétiques de base et à l'utilisation en cascade de plusieurs LE pour réaliser des fonctions comportant plus de quatre entrées.

Il faut remarquer que ces cellules ont quatre modes distincts de fonctionnement :

- mode normal ;
- mode arithmétique ;
- mode compteur / décompteur ;
- mode compteur avec remise à zéro synchrone.

Les modes de fonctionnement les plus intéressants pour les algorithmes de traitement du signal, et en particulier pour ceux qui touchent à la modulation et la démodulation de signaux, sont les deux premiers modes.

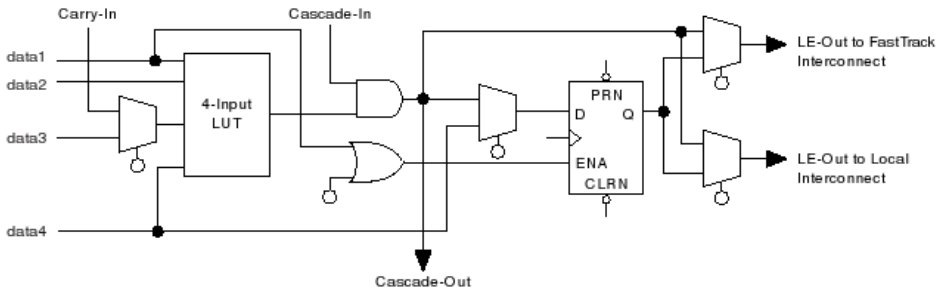


FIG. 1.11: Élément logique en mode de fonctionnement «normal»

Mode de fonctionnement normal

Ce mode propose une table LUT à quatre entrées permettant de créer des fonctions combinatoires classiques. Le schéma structurel d'un élément logique fonctionnant dans ce mode est présenté sur la figure 1.11. La présence d'un chemin d'interconnexion dédié et rapide entre les LE d'un même LAB permet de réaliser des fonctions combinatoires plus complexes utilisant plus de quatre entrées. Ce chemin est repéré par les entrées *Cascade In* et *Cascade Out* sur le schéma de la figure 1.11.

Le mode «normal» ne propose pas de chemin optimisé pour les propagations de retenues. Il est donc plus spécifiquement utilisé dans le cas de fonctions combinatoires simples et isolées.

Le mode arithmétique propose deux LUT à trois entrées au lieu de la seule LUT à quatre entrées du mode normal. Cette architecture est particulièrement indiquée pour la réalisation d'additionneurs, ou de fonctions équivalentes. En effet, dans ce cas l'une des LUT peut effectuer la somme de trois signaux (en tenant compte de la retenue éventuelle

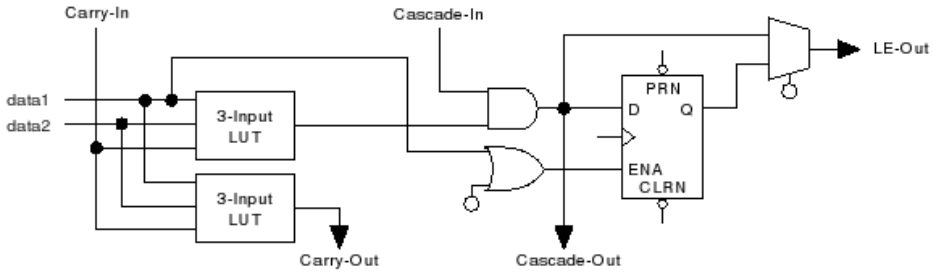


FIG. 1.12: Élément logique en mode de fonctionnement arithmétique

provenant d'un autre étage du compteur), pendant que la seconde table est utilisée pour calculer la valeur de la retenue qui sera propagée aux étages suivants du compteur. Comme dans le mode normal, différents LE peuvent être cascadés pour réaliser des fonctions combinatoires plus complexes.

Ce mode comble l'une des lacune du mode «normal» en proposant un chemin adapté à la propagation rapide de retenues. En contrepartie, les fonctions logiques que le LE est capable de traiter sont limitées à deux entrées de données. De plus, les chemins optimisés pour les retenues sont cantonnés à un même LAB. Dans le cas d'une fonction complexe s'étendant sur plusieurs LAB, le bénéfice est perdu.

1.4.4 Limitations et avantages pour le traitement numérique du signal

La structure générale d'un FPGA en fait un outil de premier plan pour implanter des algorithmes de traitement du signal et notamment, les algorithmes de modulation et de démodulation qui nous intéressent. Cependant, nous avons vu dans les sections 1.3.1 et 1.3.2 que les architectures analogiques pour la modulation et la démodulation numériques mettent en œuvre des fonctions complexes et en particulier un grand nombre de multiplieurs et/ou de diviseurs.

Un multiplieur peut être implanté en utilisant par exemple le *mode arithmétique* des éléments logiques. Il est par exemple possible de l'implanter en tirant partie des chemins optimisés qui permettent de cascader plusieurs cellules pour réaliser une fonction complexe, et des propagations de retenues rapides. Mais la complexité dans le domaine des circuits FPGA est d'abord mesurée par le nombre d'élément logiques, qui constituent la partie élémentaire du circuit. Ceci est différent dans les circuits de type ASIC, où cette complexité est évaluée en nombre de portes. Une seule cellule logique peut être équivalente à une architecture ASIC comprenant de une à 12 portes. De plus pour les FPGA, le nombre de cellules logiques élémentaires disponible est limité dès la conception du circuit. Lorsque le taux de remplissage du FPGA approche de sa capacité limite, la vitesse de fonctionnement se trouve très affectée par les délais d'interconnexion entre les différents éléments logiques.

Par conséquent, lors de la conception d'une architecture numérique dont la cible est un circuit FPGA, il est nécessaire de limiter la complexité des fonctions arithmétiques et logiques dans les limites imposées par la taille des structures élémentaires disponibles. Autant que possible, les fonctions ne doivent pas s'étendre sur plusieurs LAB, ni dépasser la taille des EAB. Les cellules élémentaires de ces circuits sont composées d'une fonction combinatoire simple suivie d'un registre. Cela favorise les architectures tendant à reproduire cette structure, et ce sont celles-ci qui présenteront les meilleures performances sur ce type de circuit [ANDR98].

Les fonctions arithmétiques mises en œuvre dans les architectures analogiques décrites plus haut sont trop complexes pour permettre une utilisation optimale d'un FPGA. Il est donc nécessaire de définir de nouvelles fonctions équivalentes, mais mieux adaptées à la structure interne de ces circuits.

1.5 Conclusion

Nous avons présenté dans ce chapitre les fondements théoriques qui régissent la transmission numérique des informations. La position et le rôle des modulateurs et des démodulateurs dans une chaîne de transmission a été explicité, ainsi que les difficultés d'implantation posées aux architectures numériques par la transmission d'un signal numérique en bande transposée. Deux exemples d'architectures de modulation et de démodulation de signaux QAM nous ont permis de mettre en lumière les éléments fonctionnels de ces architectures qui empêchent leur transcription directe sur un circuit numérique.

En effet, les performances atteintes par les systèmes numériques synchrones, et les FPGA en particulier dépendent fortement de la complexité des opérations arithmétiques à réaliser, et de la longueur maximale du chemin critique entre deux registres. L'analyse de la structure interne d'un FPGA nous a permis de comprendre les avantages, mais aussi les contraintes posées par ces circuits.

C'est pour cette raison qu'une nouvelle méthodologie de conception est proposée pour la conception de systèmes numériques pour la transmission de signaux à haut débit. Elle propose de transcrire dans le domaine numérique les fonctionnalités nécessaires à la modulation ou la démodulation, plutôt que de transcrire les architectures analogiques existantes, trop complexes et mal adaptées au numérique.

Le chapitre suivant présente une illustration de cette méthode, en proposant une architecture adaptée aux circuit numériques pour la modulation de signaux QAM.

Chapitre 2

Architectures pour les modulations numériques

L'UTILISATION de circuits de type FPGA pour la modulation numérique de signaux présente de nombreux avantages, en particulier celui d'une validation pratiquement immédiate des fonctionnalités, et d'une facilité d'adaptation inégalée en cas de changement de protocole ou de type de modulation. Cependant, l'implantation sur FPGA nous rend également tributaires de la vitesse limitée de ce type de circuit et de la quantité de ressources (sous forme de cellules élémentaires) disponibles. Ces limitations peuvent être compensées en parallélisant de façon importante les algorithmes utilisés, et en utilisant des fonctions arithmétiques simples, qui tiennent compte des spécificités architecturales de ces composants.

Ce chapitre présente donc quelques techniques permettant d'obtenir une modulation numérique de signaux, en nous focalisant sur la modulation QAM qui présente la particularité de nécessiter le contrôle de deux paramètres de la sinusoïde de sortie, à savoir, sa phase et son amplitude. Elle est donc l'une des modulations les plus complexes utilisées de nos jours dans les protocoles de transmission. Mais les solutions proposées ici

restent valides pour d'autres types de modulations, et un logiciel de génération automatique présenté en section 2.4.2 permet d'adapter automatiquement la description VHDL au type de modulation désiré.

Il est aussi fait mention d'architectures reposant sur des mémoires stockant tout ou une partie seulement des échantillons et qui privilégient la simplicité par rapport à la surface consommée. Une solution permettant d'améliorer la précision de la sinusoïde produite et qui utilise des filtres interpolateurs sera ensuite étudiée. Nous examinerons l'adéquation de ces systèmes avec une architecture reconfigurable, et la facilité avec laquelle il est possible de les adapter à des modifications de protocoles induisant des changements de fréquence ou de type de modulation. Une architecture utilisant en son cœur un algorithme de type CORDIC modifié est ensuite présentée puis analysée dans la section 2.3.

2.1 Architecture d'un modulateur numérique

Parler de modulateur entièrement numérique peut-être considéré comme un abus de langage puisqu'à proprement parler, certains des composants nécessaires à la génération d'un signal modulé ne peuvent être implantés que sur une architecture analogique. Il s'agit du convertisseur numérique analogique final, ainsi que de l'élément de filtrage permettant de supprimer les composantes à haute-fréquence résiduelles provenant de l'échantillonnage et de la numérisation des informations traitées [SHAN48, BELL02]. Cependant, à l'exception de ces deux parties, l'ensemble des autres fonctions élémentaires du modulateur peuvent être traitées à l'aide de composants numériques.

2.1.1 Différents types de modulations numériques

Le but de la modulation est d'adapter les informations à émettre au canal de transmission par l'intermédiaire d'un signal porteur sinusoïdal [PROA95] dont l'équation générale est :

$$S(t) = A \cdot \cos(\omega t + \phi). \tag{2.1}$$

Les paramètres pouvant être modifiés sont :

- l'amplitude du signal A ;
- la phase ϕ ;
- la pulsation ω.

L'ensemble de ces paramètres peut servir à coder les informations que nous souhaitons transmettre sur le canal. Puisque nous nous intéressons ici aux modulations numériques, l'information à transmettre se présente sous la forme d'une suite de bits qu'il est possible de grouper par paquets de longueur définie. On parlera alors de modulation M-aire, où M est le nombre de bits présents dans chaque symbole [STEI91].

Les types de modulation les plus utilisées sont :

– les modulations numériques d'amplitude, ou ASK[1] : $S(t) = A(n) \cdot \cos(\omega t + \phi)$;

– les modulations numériques de phase, ou PSK[2] : $S(t) = A \cdot \cos(\omega t + \phi(n))$;

– les modulations numériques de fréquence, ou FSK[3] : $S(t) = A \cdot \cos(\omega(n)t + \phi)$.

Dans chacun de ces exemples, n est un entier représentant le n^{e} symbole envoyé. D'autres types de modulation existent. C'est notamment le cas lorsque l'information est codée par des variations discrètes de l'amplitude sur deux porteuses sinusoïdales en quadrature (QAM[4]). Ce dernier type de modulation peut également être considéré comme une modulation mixte d'amplitude et de phase. En effet, l'expression du signal modulé est telle que

$$S(t) = A(n)\cos(\omega t + \phi) - B(n)\sin(\omega t + \phi) \tag{2.2}$$

$$S(t) = |A(n) + jB(n)\exp^{j\omega t + \phi}| \tag{2.3}$$

$$S(t) = \sqrt{A^2(n) + B^2(n)}\cos\left(\omega t + \phi + \arctan(\frac{B(n)}{A(n)})\right) \tag{2.4}$$

Nous nous intéresserons plus particulièrement par la suite aux architectures permettant de produire des signaux modulés selon le schéma QAM.

2.1.2 Structure générale d'un modulateur numérique

Un modulateur numérique est composé de quatre parties principales qui sont [TIER71] :

– l'accumulateur de phase, dont le rôle est de produire le terme de phase $\omega t + \phi$ dans l'équation 2.1. Ce terme correspond à la phase instantanée de l'échantillon de sinusoïde produit par le modulateur ;

[1] *Amplitude Shift Keying*
[2] *Phase Shift Keying*
[3] *Frequency Shift Keying*
[4] Quadrature Amplitude Modulation

- le codeur qui permet de convertir les séries de M-bits composant un symbole, pour les rendre compatibles avec le type de modulation choisi. Pour une modulation d'amplitude par exemple, une série de M-bits sera remplacée par la valeur correspondante de l'amplitude du signal sinusoïdal à produire en sortie du modulateur ;
- le générateur numérique de fréquence qui permet de produire les échantillons numériques de sinusoïdes à partir des paramètres que lui fournissent le codeur et l'accumulateur de phase ;
- le convertisseur numérique analogique, et le filtre passe-bas associé sont les derniers éléments de la chaîne de traitement. Ils convertissent les échantillons en signaux adaptés au médium de transmission qui peut être filaire, hertzien ou optique par exemple. Il s'agit de la seule partie de l'architecture fonctionnant en mode analogique.

Codeur

Le codeur est le plus souvent constitué par une mémoire ROM[5] de petite taille, dont est la taille d'adressage est égale au nombre de bits M par symbole de la modulation considérée. Cette ROM est adressée avec des séquences de M bits d'information fournies par les couches supérieures de l'architecture. En sortie de cet étage, on obtient, en fonction du type de modulation l'un ou plusieurs des paramètres suivants :

- F_{cw}, le mot de contrôle de la fréquence ;
- ϕ_n, le déphasage ;
- A, l'amplitude.

[5] *Read Only Memory*

Accumulateur de phase

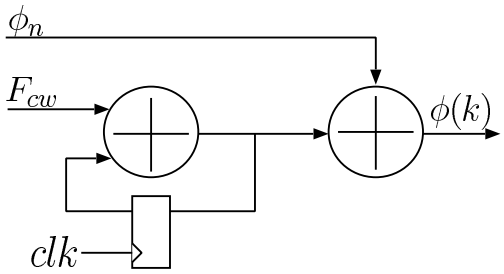


FIG. 2.1: Architecture de l'accumulateur de phase

L'architecture générale de l'accumulateur de phase est présentée sur la figure 2.1. Elle est composée principalement de deux additionneurs et d'un registre. L'accumulateur de phase est piloté par un mot de contrôle de la fréquence F_{cw} correspondant à l'incrément de phase appliqué à chaque période d'horloge du système.

F_{cw} détermine la fréquence f effective résultante du signal modulé, en fonction de la fréquence d'horloge f_{clk} de l'architecture hôte et selon l'équation :

$$f = \frac{f_{clk} \cdot F_{cw}}{2\pi}. \tag{2.5}$$

L'accumulateur de phase applique également, le cas échéant, un *déphasage* ϕ_n lorsque la modulation considérée utilise ce paramètre pour transmettre des informations. Le k^{e} mot de phase produit par cet étage vaut :

$$\phi(k) = k \cdot F_{cw} + \phi_n. \tag{2.6}$$

Il faut noter que cette opération est effectuée modulo 2π, et il est donc garanti que $0 \leq \phi(k) < 2\pi$.

Générateur de sinusoïdes

La sortie de l'accumulateur de phase est reliée à la partie la plus critique de l'architecture de modulation numérique à savoir, le générateur de sinusoïdes, aussi connu sous le nom de générateur de fréquence, qui doit calculer le cosinus du terme de phase produit par l'étage accumulateur. C'est dans cette partie que se trouve en général le chemin critique de l'architecture de modulation [VANK97], et c'est sur elle que se portera toute notre attention par la suite. À la k^{e} période d'horloge du système hôte, la valeur

$$S'(k) = \cos(\phi_k), \tag{2.7}$$

se trouvera en sortie du générateur de sinusoïdes, où ϕ_k est le mot de phase calculé précédemment par l'accumulateur de phase.

Application du terme d'amplitude

Suivant le type de modulation considéré, il est parfois nécessaire d'adjoindre aux étapes précédentes un multiplicateur. Celui-ci a pour rôle d'appliquer, le cas échéant au signal sinusoïdal produit par le générateur de sinusoïdes, le terme d'amplitude issu du codeur, noté A. Cette dernière étape permet d'obtenir un signal final de la forme :

$$S(k) = A\cos(k \cdot F_{cw} + \phi_n). \tag{2.8}$$

2.1.3 Conclusion

L'examen de l'architecture générale d'un modulateur numérique nous a permis de circonscrire les difficultés pouvant apparaître lors de l'implantation de ce type de système. Les performances des systèmes numériques étant très dépendantes de la longueur

du chemin critique, c'est dans le générateur de sinusoïdes que vont se trouver les éléments bridant la fréquence d'horloge et par conséquent, la fréquence de la sinusoïde produite. Il nous faut donc apporter un soin tout particulier à la réalisation de cet élément qui sera étudié en détail dans la section suivante.

2.2 Générateurs de sinusoïdes : architectures existantes

Plusieurs publications et produits industriels s'efforcent de résoudre le problème posé par les architectures numériques permettant de générer des signaux modulés. En particulier pour les modulations complexes, comme la QAM qui contrôle à la fois la phase et l'amplitude du signal considéré, il faut faire face à plusieurs difficultés afin de produire un échantillon binaire de signal modulé. Le premier écueil se trouve dans le calcul du cosinus de la phase de l'échantillon considéré, le second dans la multiplication par le coefficient d'amplitude.

2.2.1 Introduction

De nombreux travaux existent concernant la synthèse numérique de sinusoïdes.

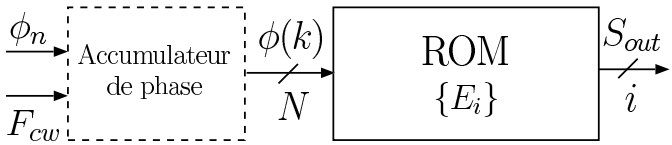


FIG. 2.2: Architecture générale d'un générateur de sinusoïdes à base de ROM

Ce problème est souvent présenté dans les publications comme la conversion d'une valeur de phase en l'amplitude d'une sinusoïde, et plusieurs architectures ont été proposées pour le résoudre. Les principales utilisent des ROM, dans lesquelles sont stockés

les échantillons pré-calculés ou une décomposition en série de Taylor de la fonction cosinus pour la traiter à l'aide de fonctions plus simples [SODA01, DECA02, LANG02], voire encore des méthodes hybrides utilisant à la fois des mémoires et des méthodes d'approximation polynomiales [LIU01]. Un schéma de l'architecture basique d'un générateur à base de ROM est représenté sur la figure 2.2.

2.2.2 Utilisation de mémoires et d'échantillons pré-calculés

Le type le plus couramment utilisé dans les circuits industriels est basé sur des mémoires en lecture seule qui contiennent les valeurs d'une sinusoïde pré-calculées lors de la conception du circuit. Dans le cas théorique idéal, en accédant à l'adresse 2^i de cette ROM, on obtient la valeur

$$E_i = \cos\left(i * \frac{2\pi}{2^N}\right), \tag{2.9}$$

où $i \in \{0, 2^N - 1\}$. Cette valeur échantillonnée est directement présentée au convertisseur numérique / analogique en sortie, puis au filtre passe bas qui ôte les composantes haute-fréquence résiduelles.

Dans le domaine numérique, la précision des échantillons stockés en ROM est forcément limitée, aussi les valeurs des E_i calculées précédemment sont limitées à l bits en pratique. Le nombre de bits de la ROM résultante est donc de $2^N \cdot l$. Ces conventions étant posées, le k^{e} échantillon disponible à la sortie numérique du générateur de sinusoïdes à base de ROM sera donc de la forme :

$$S_{out}(k) = \cos\left(k \cdot \frac{2\pi}{2^N}\right). \tag{2.10}$$

Le nombre d'échantillons stockés dans la ROM permettra de contrôler l'erreur affectant la phase du signal. En effet, plus il y a d'échantillons calculés, plus l'intervalle de

phase entre deux échantillons successifs est faible. De même, la valeur de l contrôle la précision en amplitude du signal généré. Plus l est grand, plus l'erreur de quantification commise est faible. Pour obtenir une bonne précision, à la fois en phase et en amplitude, il est souhaitable d'utiliser une ROM aussi importante que possible. Cela a cependant des conséquences néfastes sur la fréquence d'horloge maximale que peut atteindre le circuit cible et sur la surface occupée.

2.2.3 Compression de ROM

Il est possible en pratique de réduire la surface occupée par les ROM chargées de produire les échantillons de sinusoïde. La méthode la plus directe consiste à ne pré-calculer d'échantillons que pour un seul cadrant du cercle trigonométrique, et à exploiter les symétries de la fonction cosinus pour évaluer les angles situés dans les autres quadrants, grâce à des opérations combinatoires simples [NICH98].

La représentation binaire signée classique des nombres est en complément à deux, ce qui signifie que le bit de poids fort représente notamment le signe. Le second bit, conjointement avec le premier, nous donne une indication sur le sens de variation de l'amplitude. Ces deux bits permettent de détecter le quadrant dans lequel se trouve l'angle dont il faut calculer le cosinus. Les angles se trouvant dans le second et le quatrième quadrant passent par un étage de complémentation à deux leur permettant d'adresser correctement la ROM qui contient les échantillons du premier quadrant. Une seconde étape de complémentation à 2 est effectuée en sortie de la ROM pour les angles se trouvant dans les quadrants trois et quatre, afin de rétablir la valeur correcte du signe.

Ces deux techniques couplées permettent de réduire la taille de la ROM d'un facteur 4, c'est-à-dire de limiter la taille théorique à $2^{N-2} \cdot l$ bits.

Il est encore possible de réduire la taille requise de la ROM utilisée en plaçant dans

la ROM non plus directement les valeur des échantillons de sinusoïde correspondant aux angles du premier quadrant, mais les éléments de la fonction différence suivante [NICH98] :

$$\Delta(k) = \sin\left(\frac{\pi}{2^{N-1}} \cdot k\right) - \frac{\pi}{2^{N-1}} \cdot k. \tag{2.11}$$

En considérant de plus l'égalité

$$\cos x = \sin(\pi/2 - x), \tag{2.12}$$

ce choix permet de réduire de deux bits la taille des échantillons stockés dans la ROM, car

$$\max\left(\sin\left(\frac{\pi}{2^{N-1}} \cdot k\right) - \frac{\pi}{2^{N-1}} \cdot k\right) \approx 0.212 \max\left(\sin\left(\frac{\pi}{2^{N-1}} \cdot k\right)\right) \tag{2.13}$$

Il est par contre nécessaire de rajouter un additionneur supplémentaire en sortie de l'étage de calcul. Il est possible de paralléliser le calcul effectué par l'additionneur en utilisant par exemple une structure en pipeline, ce qui autorise des gains substantiels en termes de vitesse de fonctionnement. *A contrario*, il est extrêmement complexe de transformer la ROM utilisée en une architecture pipeline présentant un chemin critique court.

2.2.4 Utilisation couplée de filtres interpolateurs

En contrepartie de concessions sur la qualité spectrale du signal généré, il est possible de réduire encore la taille de la mémoire utilisée pour produire les échantillons de sinusoïde.

Approximation polynomiale

[SONG04] Afin de réduire plus avant la taille des ROM utilisées pour le calcul de la valeur de la sinusoïde, il est possible de ne placer dans les ROM qu'une partie des échantillons réellement utiles, puis d'interpoler les valeurs des échantillons intermédiaires à l'aide de polynômes approchant le comportement de la fonction cosinus pour des valeurs d'angle comprises dans un intervalle restreint [DECA02].

Ainsi, les fonctions sinus et cosinus sont décrites respectivement par les polynômes suivants

$$\begin{aligned} A(n) &= a_k \cdot C(n)^K + a_{k-1} \cdot C(n)^{K-1} + \ldots + a_0 \\ B(n) &= b_k \cdot C(n)^K + b_{k-1} \cdot C(n)^{K-1} + \ldots + b_0 \end{aligned} \tag{2.14}$$

Cette approximation introduit une erreur sur la valeur de l'amplitude de la sinusoïde calculée. Cette erreur peut être réduite en augmentant le degré du polynôme mais ceci rend également l'architecture plus complexe. Les travaux présentés dans [DECA02] concluent qu'un polynôme de degré 2 ou 3 permet d'atteindre des performances satisfaisantes, pour les applications où l'erreur résultante ne doit pas être supérieure à 60 dBc[6].

Décomposition en intervalles multiples

Le principe de cette technique consiste à considérer l'intervalle angulaire $[0, \frac{\pi}{2}]$ comme la réunion d'un ensemble de sous-intervalles de dimension fixe ou variable [LIU01, LANG02].

La fonction sinus est alors approchée sur chacun de ces intervalles, par une expression affine du type

$$s_i(x) = m_i \cdot (x - \frac{i}{8}) + y_i, \quad i \in [0,7]. \tag{2.15}$$

[6] *Decibel Relative to Carrier*

Les coefficients m_i et y_i sont évalués à l'aide d'un algorithme génétique, afin de minimiser les erreurs et le bruit résultant sur les échantillons de sinusoïde produits par ce système.

Décomposition polynomiale en série de Taylor

Une autre technique pour calculer les échantillons de sinusoïdes, avec une légère perte sur la précision de la phase résultante, consiste à séparer la phase en deux parties distinctes [QUAL91].

L'expression de la phase obtenue par les techniques de compressions de ROM est séparée en deux entités :

$$\phi = \frac{\pi}{2^{N-1}} \cdot k = \left(\frac{\pi}{2^{N-1}} \cdot k - \phi_a\right) + \phi_a. \tag{2.16}$$

La première partie, dite supérieure et notée ϕ_a, est traitée distinctement de la partie inférieure $\frac{\pi}{2^{N-1}}k$. En considérant que $\phi_a \approx \alpha$, on effectue un développement en série de Taylor de $\sin\left(\frac{\pi}{2^{N-1}}k\right)$ autour de la valeur ϕ_a, au lieu de le faire autour de α. On obtient de ce fait

$$\sin\left(\frac{\pi}{2^{N-1}}k\right) = \sin(\phi_a) + C_1(\frac{\pi}{2^{N-1}}k - \alpha)\cos(\phi_a) - \frac{C_2(\frac{\pi}{2^{N-1}}k - \alpha)}{2}\sin(\phi_a) + \ldots. \tag{2.17}$$

Les valeurs $\frac{\pi}{2^{N-1}}k - \alpha$, $1/2 \cdot \frac{\pi}{2^{N-1}}k - \alpha$, $\ldots$ sont des phases, et leur unité est donc le *radian*. Pour conserver l'homogénéité de l'équation 2.17 des coefficients correcteurs $\{C_i\}$, exprimés en rad^{-1}, sont utilisés.

Il s'agit ensuite d'approcher l'équation 2.17 en ne conservant que ses trois premiers termes. Il a été montré dans [WEAV90] que prendre des termes supplémentaires n'a que très peu d'influence sur la précision globale du système.

Les 7 bits de poids fort du mot de contrôle produit par l'accumulateur de phase (précédant le générateur de sinusoïdes) sont utilisés pour adresser deux ROM distinctes :

- la première contient les valeurs du sinus pré-calculées pour les angles codés sur 7 bits ;
- la seconde contient les valeurs du cosinus, pré-calculées selon les mêmes critères, en tenant compte également des facteurs de correction C_i.

Ces deux ROM sont ensuite compressées en utilisant la technique différentielle présentée dans la section 2.2.3. La ROM contenant les sinus permet de calculer le premier terme du développement de Taylor présenté avec l'équation 2.17. Le second terme est calculé à partir de la ROM contenant les cosinus : la valeur lue dans cette ROM est multipliée par le reste des bits du mot de contrôle de la phase :

$$\frac{\pi}{2^{N-1}}k - \alpha. \tag{2.18}$$

Cette opération permet de produire le second terme du développement de Taylor. Enfin, le troisième terme de cette transformation est obtenu par le biais d'une troisième ROM contenant les valeurs pré-calculées de $-\frac{C_2(\frac{\pi}{2^{N-1}}k-\phi_a)}{2}\sin(\phi_a)$. Celle-ci est adressée en utilisant les deux mots $\frac{\pi}{2^{N-1}}k - \phi_a$ et ϕ_a. Les trois termes obtenus sont combinés à l'aide d'un additionneur, éventuellement pipeliné pour augmenter les performances globales.

2.2.5 Conclusion

Il n'est pas possible, sur les circuits FPGA disponibles actuellement, d'atteindre les fréquences d'échantillonage nécessaires à la génération des sinusoïdes à haute-fréquence utilisées pour les transmissions haut-débit. Aussi, il est nécessaire d'utiliser des architectures fortement parallèles afin de dupliquer le nombre d'échantillons produits à chaque cycle d'horloge.

Les architectures que nous venons de décrire sont conçues pour ne produire dans le meilleurs des cas qu'un seul échantillon de sinusoïde par cycle d'horloge. Ceci n'est pas suffisant pour répondre aux besoins des transmissions haut débit, car celles-ci nécessitent des signaux de fréquence bien supérieure à celles qu'il est possible d'atteindre sur des circuits FPGA.

Il est donc nécessaire de développer des architectures adaptées à une parallélisation ultérieure le permettant d'atteindre des débits suffisamment élevés.

2.3 Processeur CORDIC modifié

La principale difficulté dans la conception d'un modulateur numérique se trouve dans la partie de génération numérique de fréquence, qui a pour rôle de calculer les échantillons de sinusoïde. Il faut donc limiter autant que possible le nombre de fonctions arithmétiques complexes intervenant dans la chaîne de génération d'échantillons, ou transformer ces opérations en sous-éléments plus simples pouvant être calculés en parallèle. L'algorithme CORDIC [VOLD59, VOLD00] permet de calculer facilement les valeurs des échantillons d'une sinusoïde sans utiliser de multiplication ni de boucle de rétroaction complexe.

2.3.1 Introduction

CORDIC correspond à l'origine à un algorithme permettant le calcul par approximations successives de fonctions trigonométriques ou bien hyperboliques. Il a été développé par Jack E. Volder dans [VOLD59], mais a bénéficié de nombreuses améliorations et extensions par la suite. L'intérêt de cet algorithme est qu'il permet l'évaluation de fonctions complexes sans employer de multiplieurs. Il a donc un usage tout

trouvé sur les architectures des microcontrôleurs simples ou sur les FPGA pour lesquels, nous l'avons vu, l'implantation d'un multiplieur, même avec des cellules dédiées, reste coûteuse en ressources et limite grandement la fréquence de fonctionnement du circuit cible. On trouvera dans [WALT71, HAMP94, MENC00] de nombreux exemples de fonctions analytiques pouvant être approchées grâce à CORDIC, mais nous nous limiterons ici aux fonctions `sinus` et `cosinus` qui ont une application directe dans la génération d'un signal modulé.

2.3.2 Principe du processeur CORDIC

Le principe de l'algorithme CORDIC comme décrit par Volder, est relativement simple.

Fondements mathématiques

Il s'agit d'effectuer la rotation matricielle d'angle θ du vecteur $\vec{v} = \binom{0}{1}$ sur le cercle trigonométrique. Cette rotation peut s'écrire sous la forme suivante :

$$\overrightarrow{v'} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \vec{v}, \tag{2.19}$$

En plaçant $\cos\theta$ en facteur, cela peut également s'écrire

$$\overrightarrow{v'} = \cos\theta \cdot \begin{pmatrix} 1 & -\tan\theta \\ \tan\theta & 1 \end{pmatrix} \cdot \vec{v}. \tag{2.20}$$

L'idée réside dans le fait de ne considérer ensuite que les rotations d'angles tels que

$$\theta = \arctan 2^i,\ i \in \mathbb{Z}. \tag{2.21}$$

L'équation 2.19 peut alors être simplifiée, et devient

$$\overrightarrow{\upsilon'} = \cos(\arctan 2^i) \cdot \begin{pmatrix} 1 & -2^i \\ 2^i & 1 \end{pmatrix} \cdot \vec{\upsilon}. \tag{2.22}$$

Une fois implantée sur un circuit fonctionnant en logique binaire, ou sous forme de code machine pour un microcontrôleur, l'opération de rotation correspond à un simple décalage binaire à droite ou à gauche, en fonction du signe de i. Il faut encore cependant multiplier le vecteur résultant par une constante valant $\cos(\arctan 2^i)$. Par conséquent, ce type de calcul est particulièrement bien adapté pour les systèmes qui nous intéressent, puisqu'il substitue à une opération trigonométrique complexe, une multiplication par une puissance de deux, soit un simple décalage en binaire.

La situation décrite ci-dessus ne considère qu'un sous ensemble particulier d'angles. Il est cependant possible d'étendre la méthode quelle que soit la valeur de l'angle considéré.

Généralisation

En effet, pour une valeur de θ prise dans l'intervalle $]-\frac{\pi}{2};\frac{\pi}{2}[$, Volder a démontré qu'il existe une suite de valeurs d_i, avec $d_i \in \{-1,1\}$, pour laquelle

$$\theta = \sum_{i=0}^{\infty} d_i \cdot \arctan 2^{-i}. \tag{2.23}$$

En termes géométriques, présentés sur la figure 2.3, la rotation d'angle θ peut être décomposée en une série de micro-rotations successives qui obéissent aux critères de l'équation 2.21, et pour lesquelles seul le *sens* de la rotation effectuée dépend de θ.

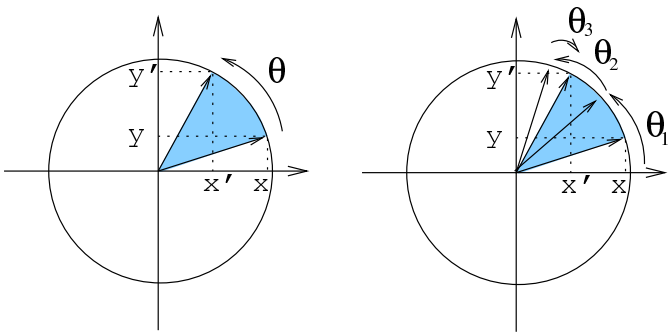


FIG. 2.3: Décompositions en plusieurs micro-rotations

Ainsi, effectuer une rotation d'angle θ peut s'écrire sous la forme matricielle suivante :

$$\overrightarrow{\upsilon'} = \prod_{i=0}^{\infty} \cos(d_i \cdot \arctan 2^{-i}) \cdot \prod_{i=0}^{\infty} \begin{pmatrix} 1 & -d_i \cdot 2^{-i} \\ d_i \cdot 2^{-i} & 1 \end{pmatrix} \cdot \vec{\upsilon}. \tag{2.24}$$

Pour simplifier cette expression, on peut poser :

$$\begin{aligned} C_\infty &= \textstyle\prod_{i=0}^{\infty} \cos(d_i \cdot \arctan 2^{-i}) \\ &= \textstyle\prod_{i=0}^{\infty} \cos(\cdot \arctan 2^{-i}), \text{ car la fonction } \cos \text{ est paire} \\ &\approx 0,6073. \end{aligned} \tag{2.25}$$

En d'autres termes, si $\vec{\upsilon}_k = \binom{x_k}{y_k}$ est le vecteur obtenu à la k^{e} itération des produits décrits dans l'équation ci-dessus, la micro-rotation permettant de passer de $\vec{\upsilon}_k$ à $\vec{\upsilon}_{k+1}$ est équivalente au système :

$$\begin{cases} x_{k+1} &= x_k - d_k \cdot y_k \cdot 2^{-i} \\ y_{k+1} &= y_k + d_k \cdot x_k \cdot 2^{-i} \end{cases}. \tag{2.26}$$

Il est donc possible d'effectuer de façon itérative la rotation complète d'angle θ, en n'utilisant que des opérations simples d'addition et de décalage à gauche ou à droite.

Cette suite récurrente converge vers la matrice solution du problème initial, à savoir

$$\overrightarrow{\upsilon'} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \upsilon. \tag{2.27}$$

2.3.3 Application à la génération paramétrique de sinusoïdes

Le principe de la génération de sinusoïdes consiste à produire en sortie de notre architecture une onde de la forme $S(t) = A \cdot \cos(\omega t + \phi(t))$. L'expression échantillonnée de ce signal est donc $S(n) = A \cdot \cos(\omega T_e \cdot n + \phi(n))$, si $\frac{1}{T_e}$ est la fréquence d'échantillonnage considérée. Il est donc nécessaire de calculer pour chacun des échantillons, la valeur du cosinus correspondant. L'algorithme CORDIC présente donc un très gros avantage, puisque comme nous l'avons vu, il n'utilise aucune opération arithmétique complexe pour parvenir au résultat final. En choisissant un vecteur $\vec{\upsilon}$, et un angle θ tels que

$$\vec{\upsilon} = \begin{pmatrix} A \\ 0 \end{pmatrix} \tag{2.28}$$

et

$$\theta = 2\pi \cdot \omega t + \phi(t) \tag{2.29}$$

l'algorithme produit une rotation qui conduit au vecteur résultant

$$\overrightarrow{\upsilon'} = \begin{pmatrix} A\cos(2\pi \cdot \omega t + \phi(t)) \\ A\sin(2\pi \cdot \omega t + \phi(t)) \end{pmatrix}. \tag{2.30}$$

Cet algorithme récursif présente cependant trois types de limitations dans le cadre de l'implantation sous forme de circuit numérique.

Approximation de la fonction trigonométrique

La première est qu'il ne fournit un résultat exact qu'au bout d'un nombre *infini* de rotations successives. Cet aspect doit être pris en considération, mais n'a que peu d'effet sur les architectures numériques qui nous intéressent. Car la précision des opérations arithmétiques effectuées est dans tous les cas bridée par la résolution binaire du microprocesseur ou de la logique implantée. Il suffit donc d'effectuer suffisamment de micro-rotations pour que l'erreur de calcul effective ne dépasse pas la résolution du circuit cible. Si on arrête le calcul au bout de N étapes, soit N micro-rotations effectuées, l'erreur commise sur l'angle de la rotation réellement effectuée est telle que

$$\sum_{i=N}^{\infty} \arctan(2^{-i}) \leq \varepsilon_N, \tag{2.31}$$

et pour N suffisamment grand, $\arctan(2^{-i}) \approx 2^{-i}$. Ce qui conduit à

$$2^{-N} \underbrace{\sum_{i=0}^{\infty} 2^{-i}}_{\lim_{\to\infty}=2} \leq \varepsilon_N \tag{2.32}$$

$$2^{-N+1} \leq \varepsilon_N. \tag{2.33}$$

La condition nécessaire et suffisante pour obtenir un résultat à une précision donnée de n_b bits est donc que $N \geq n_b + 1$.

Calcul du sens des micro-rotations

La seconde limitation de cet algorithme est de nécessiter la connaissance *a priori* du sens de chacune des micro-rotations successives. La suite $\{d_i\}_{i\in\mathbb{N}}$ correspondant à chacun des angles pour lesquels le cosinus doit être évalué peut être stockée dans une mémoire. Cependant, cette solution n'est pas envisageable dans le cadre d'une implanta-

tion numérique sur ASIC ou FPGA, de par l'occupation de surface trop importante. Mais il est possible de prévoir directement en ligne le sens de la prochaine micro-rotation à effectuer, en rajoutant au système 2.26 une équation dont le rôle est de conserver en permanence *l'angle de la rotation qu'il est encore nécessaire d'effectuer pour aboutir à une rotation d'angle* θ [NAHM97]. Après chaque micro-rotation, cette angle est augmenté – ou diminué suivant le sens de cette micro-rotation – de la valeur 2^{-i}. Si l'angle restant est supérieur à 0, alors la prochaine rotation se fera dans le sens anti trigonométrique. Dans le cas contraire, elle se fera dans le sens trigonométrique. Ces considérations se traduisent par une modification des équations 2.26 qui deviennent :

$$\begin{cases} x_{k+1} &= x_k - d_k \cdot y_k \cdot 2^{-i} \\ y_{k+1} &= y_k + d_k \cdot x_k \cdot 2^{-i} \\ z_{k+1} &= z_k - d_k \cdot \arctan 2^{-i} \\ d_{i+1} &= -1 \text{ si } z_{k+1} \geq 0,\ 1 \text{ sinon.} \end{cases} \tag{2.34}$$

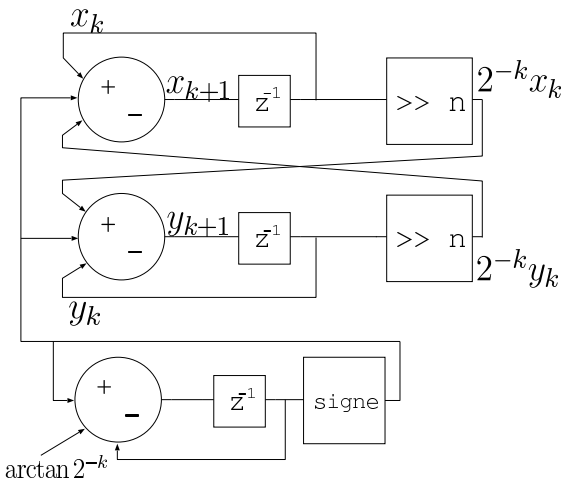


FIG. 2.4: Cellule élémentaire d'un processeur CORDIC

La figure 2.4 présente la cellule de base du processeur CORDIC que nous venons de

décrire. On y retrouve les trois chemins de calcul X, Y et Z. La complexité arithmétique se trouve dans les additionneurs soustracteurs, et dans les décaleurs, comme nous le verrons plus tard.

Facteur d'échelle

Enfin, la dernière limitation provient du terme C_∞ qui correspond à une erreur effectuée sur la norme du vecteur $\overrightarrow{v'}$ résultant. Mais cette valeur, appelée facteur d'échelle, ne dépend que du nombre de micro-rotations effectuées, et non pas de leur sens. Il est donc possible de la compenser *a priori*, en choisissant un vecteur $\vec{v}$ tel que

$$\vec{v} = \begin{pmatrix} C_\infty \cdot A \\ 0 \end{pmatrix}. \tag{2.35}$$

On remarquera que dans ce cas qu'il est possible de prendre en compte directement le terme d'amplitude du signal produit en sortie, et cela *sans nécessiter de multiplieur supplémentaire* à la sortie de l'étage de génération de sinusoïdes.

2.3.4 Architecture CORDIC itérative

L'algorithme présenté plus haut peut être implanté *en l'état* sur un circuit numérique. C'est d'ailleurs bien souvent sous cette forme qu'il est utilisé sur les microcontrôleurs pour permettre le calcul de fonctions trigonométriques, voir hyperboliques.

L'architecture proposée pour le générateur numérique de sinusoïdes est présentée sur la figure 2.5. Il est composé de trois éléments principaux :

- une ROM, contenant les valeurs pré-calculées des $\arctan 2^{-i}$;
- un bloc combinatoire, chargé de calculer les valeurs successives de x_i et y_i ;
- un bloc combinatoire, chargé d'évaluer l'angle restant.

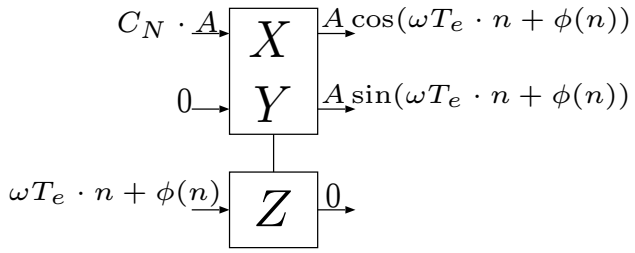


FIG. 2.5: Générateur de sinusoïdes à base de processeur CORDIC itératif.

Au début du fonctionnement du circuit, les blocs X et Y sont pré-chargés avec les valeurs $C_N \cdot A$ et 0 respectivement, et Z est chargé avec la valeur de l'angle dont il faut calculer le cosinus, à savoir θ. À chaque itération, les nouvelles valeurs de X et Y sont calculées en fonction des valeurs de l'étape précédente, et du sens de la rotation donné par le signe de Z. La valeur de Z est modifiée pour refléter la rotation qu'il reste à effectuer pour atteindre la rotation d'angle θ. Le calcul est terminé lorsque $Z = 0$, à la précision près du circuit sur lequel cette architecture est implantée. Il faut environ $n+1$ itérations pour obtenir la valeur d'un échantillon de sinusoïde avec une précision de n bits.

La position des registres est telle qu'à chaque cycle d'horloge, X Y et Z sont mis à jour une fois. Par conséquent, $n+1$ cycles d'horloges seront nécessaires pour produire un échantillon. La fréquence d'échantillonnage théorique correspondante sera donc de $\frac{f}{n+1}$, où f est la fréquence de fonctionnement du circuit sur lequel est implantée cette architecture. De plus, cette architecture utilise :

- trois additionneurs soustracteurs complets sur n bits ;
- une ROM comportant $n+1$ valeurs $\arctan 2^{-i}$, chacune ayant une précision de n bits, soit n^2+1 bits au total ;
- deux décaleurs à barillet sur n bits, dont la complexité est comparable à celle d'un additionneur de même taille.

L'architecture présentée ici a l'avantage d'être très proche de l'algorithme CORDIC, et d'occuper relativement peu de place [HENH92, DUPR93]. Mais elle fait appel à des décaleurs à barillet qui engendrent un bloc combinatoire volumineux et dont le chemin critique est important. De plus, nous cherchons à obtenir la plus grande fréquence d'échantillonnage possible, et cette architecture nécessite plusieurs cycles d'horloge pour produire un seul échantillon, ce qui n'est généralement pas satisfaisant.

2.3.5 Architecture CORDIC pipeline

La principale limitation de l'architecture précédente est qu'elle restreint fortement la fréquence d'échantillonnage théoriquement possible, d'un facteur au moins égal à la précision en bits des échantillons de sortie. Il est cependant possible, mais au prix d'une occupation de surface plus élevée, de produire un échantillon de sinusoïde par cycle d'horloge, en utilisant une version pipeline de l'algorithme CORDIC [KANG03, KANG06].

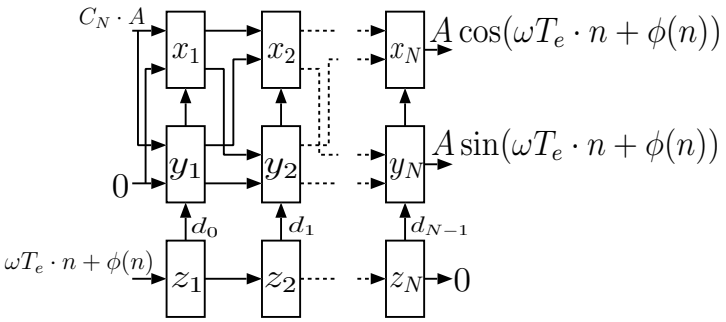


FIG. 2.6: Générateur de sinusoïdes à base de processeur CORDIC pipeline.

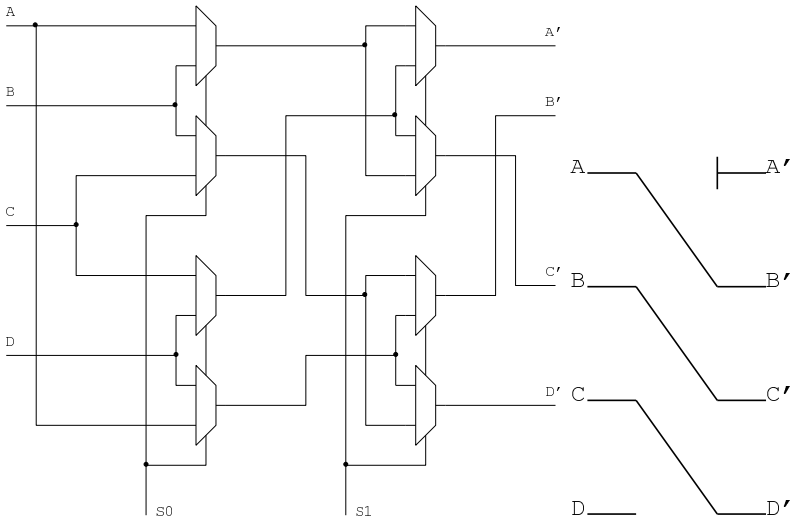
L'architecture modifiée est présentée sur la figure 2.6. Elle comporte un pipeline dont la profondeur ne dépend que de la précision souhaitée pour les échantillons de sinusoïde. Il y a $n+1$ étages sur le schéma présenté, pour une précision théorique de n bits. Chaque étage du pipeline n'est chargé d'effectuer qu'une seule micro-rotation, toujours

la même, dont seul le sens peut être modifié. C'est-à-dire que le k^{e} étage correspond au système d'équations :

$$\begin{cases} d_k &= signe(z_k) \\ x_{k+1} &= x_k - d_k \cdot y_k \cdot 2^{-k} \\ y_{k+1} &= y_k - d_k \cdot x_k \cdot 2^{-k} \\ z_{k+1} &= z_k - d_k \cdot \arctan 2^{-k} \end{cases} . \tag{2.36}$$

Cette fois, la fin du calcul n'est plus déterminé par le fait que le contenu de Z vaut 0, mais par le fait que $n+1$ micro-rotations ont été effectuées. D'après les résultats présentés dans la section 2.3.3, cela est suffisant pour obtenir un échantillon de sinusoïde à la précision binaire désirée.

En plus du fait de produire un échantillon à chaque cycle d'horloge, cette architecture possède l'avantage de ne pas utiliser de décaleurs à barillet puisque pour chaque étage du pipeline, le décalage à effectuer est connu par avance et ne dépend que de k.



(a) Décaleur à barillet (b) Décaleur câblé

FIG. 2.7: Deux architectures de décaleurs.

Comme le montre la figure 2.7, ce décalage peut être implanté par câblage sans nécessiter d'élément logique supplémentaire. Le chemin critique de cette architecture n'est plus affecté par cet élément mais uniquement par les additionneurs nécessaires au calcul des valeurs de X, Y et Z. Par conséquent, ce chemin n'est pas plus long que celui d'un additionneur complet sur N bits. En résumé, l'architecture pipeline utilise :

- $N+1$ additionneurs soustracteurs complets pour chacun des trois chemins de calcul (X, Y et Z) ;
- les valeurs des $\arctan 2^{-i}$ sont implantés sous forme de constantes, et ne consomment donc pas de surface ;
- $2(N+1)$ décalages, ne nécessitant pas d'élément de logique combinatoire.

La fréquence théorique de production d'échantillons de sinusoïde est donc égale à la fréquence d'horloge du circuit sur lequel est implantée cette architecture.

La version pipeline de l'algorithme CORDIC présente un avantage indéniable sur la version itérative, si le gain de vitesse est notre priorité [DUPR93, PHAT98, SING03]. Mais la version pipeline occupe une surface importante qu'il est encore possible de réduire en supprimant l'accumulateur d'angles Z.

2.3.6 Approximation de la valeur des $\arctan 2^{-i}$

Afin de pouvoir supprimer l'accumulateur d'angles, il est nécessaire de trouver une représentation simplifiée des $\arctan 2^{-i}$ stockés jusqu'à maintenant sous forme de constantes. On peut pour cela constater que, pour une valeur x suffisamment petite

$$\arctan x \approx x. \tag{2.37}$$

Pour un système numérique dont la résolution est finie et déterminée par le nombre de bits n utilisés pour exprimer les valeurs numériques, ceci est équivalent à

$$|\arctan x - x| < 2^{-n+1}, \tag{2.38}$$

2^{-n+1} étant le plus petit nombre qu'il est possible de représenter sur n bits. Un développement en série de Taylor de $\arctan x$ nous permet d'écrire que

$$\arctan x = \sum_{i=0}^{\infty} \frac{(-1)^i}{2n+1} x^{2n+1} \tag{2.39}$$

$$= x - \frac{x^3}{3} + \frac{x^5}{5} \ldots. \tag{2.40}$$

Ce développement est valable pour les valeurs de x telles que $|x| < 1$. On peut déduire de cela l'inégalité

$$|\arctan x - x| < \frac{|x|^3}{3}, \tag{2.41}$$

pour $x \neq 0$. Cette condition, appliquée à l'équation 2.38, nous permet d'affirmer que l'approximation est valable pour les valeurs de x telles que

$$\frac{|x|^3}{3} < 2^{-n+1}. \tag{2.42}$$

En appliquant ce résultat aux angles particuliers utilisés dans les architectures CORDIC qui sont de la forme $x = 2^{-k}$, l'équation précédente devient

$$\frac{|2^k|^3}{3} < 2^{-n+1}. \tag{2.43}$$

Cette expression peut être simplifiée de la façon suivante

$$\ln\left(\frac{2^{-3k}}{3}\right) < \ln\left(2^{-n+1}\right)\frac{n-1}{3} - \frac{\ln 3}{3\ln 2} < k. \tag{2.44}$$

Par conséquent, l'équation

$$k > \frac{n-1}{3} \tag{2.45}$$

est une condition suffisante pour que l'approximation $\arctan 2^{-k} = 2^{-k}$ soit acceptable à la précision binaire du circuit utilisé.

Extension de l'approximation

On souhaite utiliser des rotations liés directement aux bits de l'angle, mais sans pour autant introduire de multiplications. Pour cela, on cherche à utiliser l'approximation $\arctan 2^{-k} = 2^{-k}$ pour décider du sens de rotation à effectuer dans le processeur CORDIC, quelle que soit la valeur de k. Pour $k \leq \frac{n-1}{3}$, une erreur est donc introduite dans la micro-rotation correspondante. Afin de s'affranchir de cette erreur, il est possible d'exprimer, pour $k < \frac{n}{3}$, la valeur de 2^{-k} en fonction des $\arctan 2^{-i}$, pour $i \leq n$. Par exemple, pour $n = 8$:

$$\begin{cases} 2^{-1} &= \arctan 2^{-1} + \arctan 2^{-5} + \arctan 2^{-8} \\ 2^{-2} &= \arctan 2^{-2} + \arctan 2^{-5} + \arctan 2^{-8} \\ 2^{-3} &= \arctan 2^{-3} \\ \dots & \\ 2^{-8} &= \arctan 2^{-8} \end{cases} \tag{2.46}$$

Ainsi, pour effectuer une rotation d'angle 2^{-2} au lieu d'une rotation d'angle $\arctan 2^{-2}$, il faut effectuer en tout trois rotations intermédiaires. La première d'angle $\arctan 2^{-2}$, la

seconde d'angle $\arctan 2^{-5}$, et la dernière d'angle 2^{-8}. À partir de $k \geq 3$, il n'est plus nécessaire d'effectuer de rotations de correction.

Mais chacune des micro-rotations rajoutées modifie le facteur d'échelle d'une quantité égale à $\sqrt{1+2^{-2i}}$ pour une rotation d'ordre i. Il est nécessaire d'étudier si cela a une influence sur le facteur d'échelle global.

Influence sur le facteur d'échelle

La contribution au facteur d'échelle global des rotations de rang supérieur ou égal à k est négligeable à la précision binaire n si :

$$B_k = \prod_{i=k}^{n} \sqrt{1+2^{-2i}} \leq 1+2^{-n} \tag{2.47}$$

En utilisant la même méthode que dans le cas de l'approximation $\arctan 2^{-k} = 2^{-k}$, on démontre que l'équation 2.47 est équivalente à :

$$k \geq \frac{n-1}{2} \tag{2.48}$$

Toutes les micro-rotations de l'exemple présenté plus haut répondent à ce critère, elles n'ont donc aucune influence sur le facteur d'échelle global.

Ces approximations, et les quelques micro-rotations supplémentaires qu'elles impliquent, vont nous permettre de supprimer l'accumulateur d'angle utilisé dans l'algorithme CORDIC au prix d'un pipeline un peu plus long.

2.3.7 Optimisation de CORDIC pour la génération de sinusoïdes

[ARNO05]

La suppression de l'accumulateur d'angle nécessite de connaître pour chaque étage du pipeline décrit ci-dessus, le sens de la micro-rotation à effectuer pour un angle donné. En lieu et place du signe de z_k pour déterminer le sens de la rotation d'ordre k, on se propose d'utiliser le chiffre d'ordre k dans la représentation binaire de l'angle θ. Par exemple, pour un angle dont la représentation binaire est la suivante $0_1 1_1 1_2 0_3 1_4 0_5 1_6$, la première rotation sera contrôlée par le chiffre 0, la seconde par 1, ..., la sixième et dernière par 1 également.

La différence fondamentale par rapport aux architectures CORDIC décrites plus haut est que le chiffre contrôlant le sens de la rotation prend ses valeurs dans l'intervalle $\{0,1\}$ au lieu de $\{-1,1\}$. Il est donc nécessaire de modifier en profondeur l'algorithme utilisé pour qu'il prenne en compte ce nouveau paramètre.

Reconsidérons tout d'abord la décomposition angulaire présentée avec l'équation 2.23. En considérant qu'à la k^{e} étape de l'algorithme, la micro-rotation correspondante peut :

- soit être effectuée dans le sens trigonométrique, si d_k, le chiffre d'ordre k de θ, vaut 1 ;
- soit ne pas être effectuée si d_k vaut 0.

Cette équation peut-être modifiée de la façon suivante :

$$\theta = \sum_{i=1}^{\infty} d_i \cdot 2^{-i} \tag{2.49}$$

$$\theta = \sum_{i=1}^{\infty} (1 + c_i) \cdot 2^{-i} \tag{2.50}$$

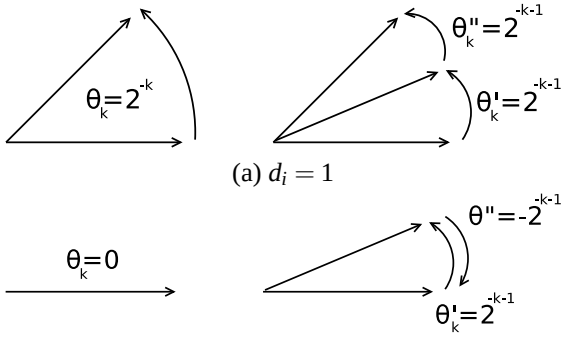
$$\theta = \sum_{i=1}^{\infty} 2^{-i} + \sum_{i=1}^{\infty} c_i, \tag{2.51}$$

avec $c_i \in \{-1,1\}$. On pose $\alpha_0 = \sum_{i=1}^{\infty} 2^{-i}$. Dans ce cas, α_0 ne dépend pas de θ, et les coefficients $\{c_i\}_{i\in\mathbb{N}}$ prennent leurs valeurs dans l'intervalle $\{-1,1\}$. Nous obtenons

donc une nouvelle décomposition de l'angle θ telle que :

$$\theta = \alpha_0 + \sum_{i=0}^{\infty} c_i \cdot 2^{-i}. \tag{2.52}$$

D'un point de vue géométrique, chaque micro-rotation utilisée dans l'algorithme CORDIC conventionnel est décomposée en deux micro-rotations d'ordre inférieur, l'une d'angle 2^{-k-1} et l'autre d'angle $c_k 2^{-k-1}$, c_k valant -1 ou 1. C'est cette seconde rotation qui transporte une information sur le sens de la micro-rotation à effectuer, et qui pourra donc être utilisée dans l'algorithme CORDIC.



(a) $d_i = 1$

(b) $d_i = 0$

FIG. 2.8: Principe du recodage de l'angle.

Les deux cas envisageables sont présentés sur la figure 2.8. Dans le cas 2.8a, la valeur du chiffre de rang k de θ est 1. Alors, deux micro-rotations d'ordre $k+1$ sont effectuées pour simuler une rotation complète d'ordre k. Dans le cas 2.8b, la valeur du chiffre de rang k de θ est zéro. Aucune rotation ne doit être effectuée, mais pour conserver une architecture homogène, et ne pas modifier le facteur d'échelle, on effectue en fait deux rotations antagonistes de rang $k+1$, dont la résultante est une rotation nulle.

Ainsi, en utilisant les chiffres de l'angle θ pour contrôler chaque étape de micro-rotation de l'algorithme CORDIC, on obtient en sortie du pipeline les valeurs de $\cos(\theta - \alpha_0)$, où α_0 est une constante qui ne dépend pas de la valeur de θ. Il est donc nécessaire

de compenser ce décalage angulaire qui se traduit par un déphasage indésirable mais constant en sortie de l'étage de production de sinusoïdes. Cette compensation peut-être effectuée en amont du processeur CORDIC, en modifiant directement la valeur de l'angle.

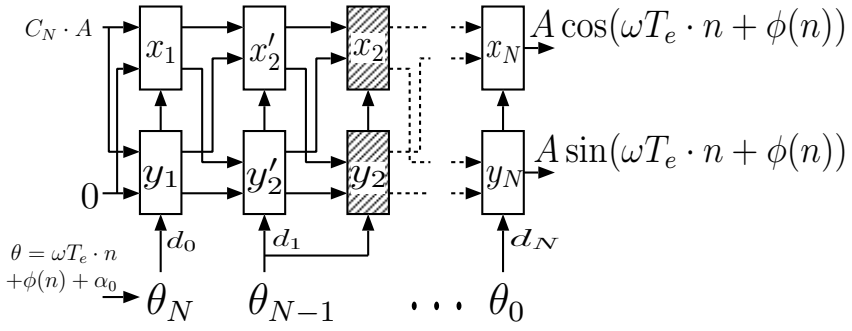


FIG. 2.9: Architecture de génération de sinusoïdes utilisant un recodage de l'angle.

L'architecture résultante est présentée sur la figure 2.9. La partie grisée correspond à l'une des micro-rotations ajoutées pour compenser l'erreur sur le facteur d'échelle.

L'architecture exposée ici permet au circuit sur lequel elle est implantée d'atteindre une fréquence d'horloge supérieure à celle obtenue avec les autres algorithmes présentés, car elle réduit le chemin critique de la partie combinatoire de chacun des étages du pipeline en supprimant l'étage de détection de signe, et l'un des additionneurs / soustracteurs. Mais au vu de la faible surface occupée, il est possible sur un seul FPGA de placer plusieurs étages CORDIC capables de générer en parallèle plusieurs échantillons de sinusoïde.

2.3.8 Architecture parallèle

Du fait de la fréquence maximale limitée des circuits numériques qui nous intéressent ici, il est nécessaire de générer plusieurs échantillons de sinusoïde en parallèle, afin d'atteindre une fréquence d'échantillonnage apparente supérieure à la fréquence

d'horloge. Ceci peut être effectué en multipliant le nombre de processeurs CORDIC utilisés, et en confiant à chacun la tâche consistant à générer des échantillons décalés.

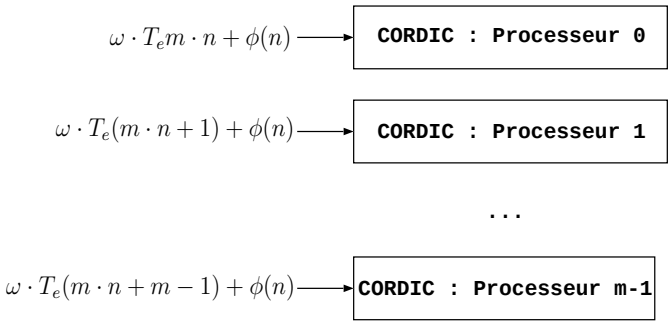


FIG. 2.10: Production en parallèle de plusieurs échantillons.

Le principe de cette architecture est présenté sur la figure 2.10. Pour permettre la production d'échantillons de façon décalée, il est nécessaire de modifier le terme de phase fourni à chacun des pipelines de la façon suivante :

$$\phi' = \omega \cdot T_e \cdot (m \cdot n + i) + \phi(n). \tag{2.53}$$

Dans cette équation, i représente l'ordre du processeur CORDIC considéré. De cette façon, chaque étage produira une sinusoïde dont la fréquence apparente est m fois la fréquence de la sinusoïde produite globalement. Le signal de sortie du modulateur est reconstitué en utilisant successivement les échantillons produits par chacun des étages.

La conséquence est que le système dans son ensemble est capable de produire m fois plus d'échantillons d'une même sinusoïde qu'un seul processeur CORDIC. Cette architecture présente cependant quelques limitations, car il est nécessaire que les composants chargés de la sérialisation et de la désérialisation des données à l'entrée et à la sortie de l'étage modulateur puisse fonctionner à la fréquence apparente de fonctionnement du système. Ceci afin que chacun des processeurs CORDIC puisse être alimenté

avec un nombre suffisant d'échantillons. Certains circuits FPGA comme ceux de la famille Altera Stratix II disposent en série des sérialiseurs / désérialiseurs rapides sur 8 bits [STRA05]. Il est donc possible d'éviter de recourir à un composant externe analogique pour implanter cette architecture parallèle.

2.3.9 Conclusions

Nous avons présenté dans cette section une architecture de génération de sinusoïdes utilisant un processeur CORDIC modifié pour répondre aux exigences d'une implantation optimale sur un circuit numérique de type FPGA. L'architecture résultante autorise l'utilisation efficace d'un pipeline, qui permet d'obtenir un chemin critique court et donc une fréquence de fonctionnement du circuit FPGA plus importante. Enfin, la simplification de l'algorithme par la suppression de l'étage accumulateur d'angle, permet de réduire la surface occupée, et d'utiliser en parallèle plusieurs étages CORDIC. Cette dernière architecture multiplie le nombre d'échantillons effectivement produits en un seul cycle d'horloge, et augmente d'autant l'efficacité du modulateur.

2.4 Implantation et résultats

2.4.1 Méthodologie

Les différentes architectures décrites dans la section précédente sont ici implantées sur circuit FPGA. Les outils logiciels utilisés pour cette synthèse sont le synthétiseur Synplify de Synplicity, et le synthétiseur Quartus II d'Altera pour la phase de placement routage. Afin de permettre d'utiliser des outils de conception provenant de tiers différents, l'ensemble des architectures ont été décrites en VHDL[7] au niveau RTL[8]. Les descriptions VHDL permettent de garder un bon niveau de compatibilité avec la plupart des logiciels de synthèse existants. Ainsi les performances des circuits synthétisés par la chaîne d'outils Quartus d'Altera, utilisée seule, peuvent être comparées à celles obtenues en utilisant un outil de synthèse externe, comme Synplify.

[7] *Very High Speed Integrated Circuit Hardware Description Language*
[8] *Register Transfer Level*

Il s'est avéré que les résultats obtenus grâce à l'utilisation de Synplify étaient supérieurs à ceux obtenus avec Quartus seul. Quartus n'est donc utilisé dans notre cas que pour effectuer le placement routage final, et pour la génération du micro-code permettant de programmer le FPGA.

2.4.2 Logiciel de génération automatique

Afin de s'affranchir de la tâche répétitive consistant à écrire pour chaque architecture et chaque paramétrage une nouvelle description en VHDL, un logiciel écrit en Perl à été développé. Son rôle est de produire automatiquement les fichiers de description des architectures en fonction des paramètres qui sont :

- le type de modulation ;
- la précision souhaitée pour le signal de sortie ;
- la fréquence souhaitée pour le signal de sortie ;

Le logiciel calcule automatiquement pour chaque architecture :

- le contenu des ROM, le cas échéant ;
- la valeur des arctan ;
- la profondeur des pipelines ;
- le degré de parallélisme nécessaire pour obtenir les performances voulues ;
- l'architecture du codeur correspondant à la modulation choisie.

2.4.3 Modulateur QAM

L'implantation du modulateur QAM complet diffère peu de celle du générateur de sinusoïdes. En effet, il suffit de rajouter la partie codeur manquante, dont le seul rôle est de convertir les données M-aires transportant les informations issues des couches supérieures du système de communication en informations sur la phase et l'amplitude

du signal sinusoïdal de sortie. La ROM correspondante est automatiquement calculée et générée par le logiciel décrit dans le paragraphe précédent.

2.4.4 Performances comparées

Les résultats pour l'ensemble des architectures présentées dans cette partie sont résumés sur le tableau 2.1. Le circuit FPGA de destination est de type Altera FLEX10K200E-2. La fréquence maximale annoncée par le constructeur pour ce type de circuit est de 250 MHz, ce qui correspond, dans un cas idéal, à une fréquence de production d'échantillons de sinusoïde de 250 méga-échantillons par seconde également, sur une architecture non parallèle.

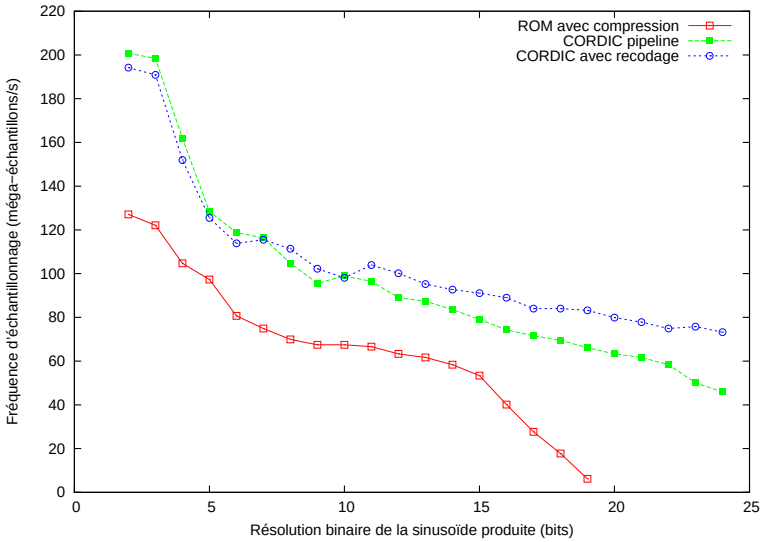


FIG. 2.11: Fréquence d'échantillonnage maximale atteinte avec les différentes architectures.

Les figures 2.11 et 2.12 présentent respectivement les performances en termes de fréquence d'échantillonnage de la sinusoïde pour la plus performante des architectures à base de ROM (qui correspond à celle utilisant la décomposition en série de Taylor),

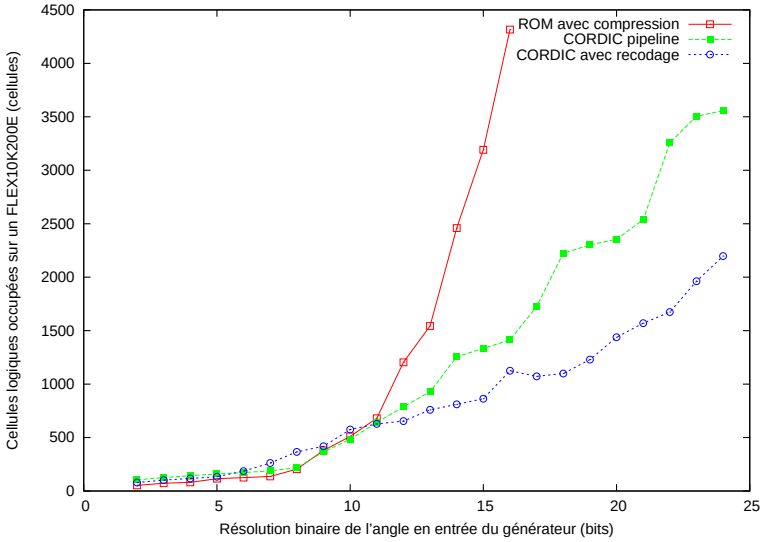


FIG. 2.12: Nombre de cellules logiques (LC) utilisées sur le circuit FPGA.

pour l'architecture CORDIC pipeline, et pour notre architecture modifiée utilisant un recodage de l'angle.

On constate que l'occupation en surface de l'architecture à base de ROM augmente très rapidement, et dépasse le nombre de cellules logiques disponibles dans le FPGA utilisé, bien que celui-ci dispose de structures (EAB) permettant l'implantation aisée de blocs de type mémoire. Ces cellules, qui se trouvent en quantité limitée, sont d'ailleurs utilisées principalement pour les faibles résolutions binaires. Au delà, les mémoires sont implantées à l'aide de portes logiques diminuant l'efficacité de l'architecture. La conséquence directe de cette augmentation est la chute très importante de la vitesse de fonctionnement du circuit. Cette diminution est principalement due aux délais introduits par le routage entre les différents éléments.

Les architectures à base de processeurs CORDIC sont les seules permettant d'atteindre des fréquences d'échantillonnage acceptables sur FPGA, en particulier au-delà d'une résolution de la phase de 12 bits. Aussi, la surface restant disponible sur le FPGA

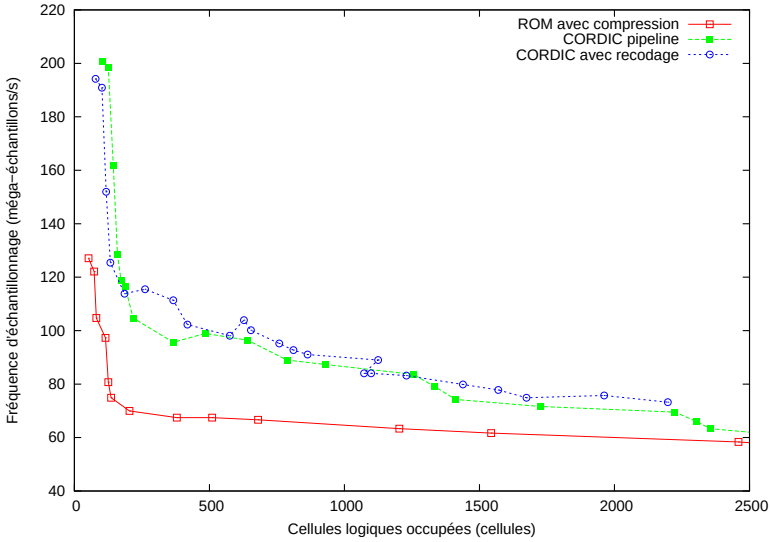


FIG. 2.13: Fréquence d'échantillonnage en fonction du nombre de cellules logiques occupées.

nous autorise à intégrer quatre étages de calcul en parallèle, ce qui conduit à une fréquence d'échantillonnage apparente de 420 méga-échantillons par seconde, au lieu de 110 méga-échantillons constatés pour une architecture à un seul étage.

Le tableau 2.1 résume les résultats pour un angle à l'entrée du modulateur codé sur 16 bits, et pour des échantillons produits codés sur 16 bits également.

Architecture	Cellules		Fréquence d'échantillonnage	
	Théoriques	Effectives	Théorique	Effective (MSample/s)
ROM compressée	-	5543	-	-
Interp. Linéaire	-	4502	-	31
Intervalles	-	4302	-	36
Taylor	-	3996	-	49
CORDIC Itératif	N_{cell}[1]	230	$\frac{f}{n+1}$[2]	13
CORDIC Pipeline	$(n+1) \cdot N_{cell}$	1430	f	101
CORDIC avec nouveau codage	$\frac{2}{3}(n+1) \cdot N_{cell}$	980	f	110
CORDIC Parallèle	$4 \cdot \frac{2}{3}(n+1) \cdot N_{cell}$[3]	4140	$4 \cdot f$	420

[1] Nombre de cellules occupées par un étage CORDIC.
[2] n étant la précision des échantillons produits, en bits.
[3] 4 est le degré de parallélisme choisi ici.

TAB. 2.1: Performances comparées des différentes architectures pour la modulation QAM.

Conclusion

Dans ce chapitre, nous avons présenté plusieurs architectures permettant de générer un signal sinusoïdal échantillonné dont l'amplitude, la fréquence et la phase peuvent être contrôlées. En plus des architectures à base de ROM contenant tout ou une partie des échantillons à produire, ou de celles qui reposent sur des filtres ou des systèmes d'interpolation, une nouvelle architecture parallèle utilisant un processeur CORDIC a été présentée.

Ces architectures, exceptées celles basées sur des processeurs CORDIC, utilisent des éléments qu'il est difficile d'intégrer de façon satisfaisante sur un circuit numérique, et

à plus forte raison s'il est de type FPGA. Les ROM consomment une grande quantité de ressources, d'autant plus que la précision de la sinusoïde produite est élevée. De plus, afin de pouvoir gérer des fréquences d'échantillonnage de sinusoïdes supérieures, il est nécessaire de paralléliser au maximum les architectures numériques. Or la parallélisation d'architectures à base de ROM nécessite la duplication de celles-ci, mais la surface occupée devient alors bien trop importante pour permettre une implantation sur circuit FPGA.

Les architectures de type CORDIC présentent au contraire de bonnes possibilités d'évolution. Ceci est particulièrement vrai pour celles utilisant le système de recodage de l'angle que nous avons introduit. En effet :

- il est aisé de produire une architecture de type pipeline utilisant un processeur CORDIC avec recodage de l'angle, ce qui permet de réduire le chemin critique entre chaque étage du pipeline à celui d'un simple additionneur ;
- les modifications de l'algorithme que nous avons présentées permettent de simplifier le processeur CORDIC de façon à diminuer la surface occupée ;
- à précision égale, un processeur ainsi modifié occupe beaucoup moins de surface qu'une architecture à base de ROM équivalente, il peut donc être dupliqué pour autoriser le traitement en parallèle de plusieurs échantillons.

Chapitre 3

Architectures pour la démodulation numérique

TOUT comme pour les modulateurs, les architectures numériques apportent des avantages certains aux démodulateurs de signaux notamment dans le cas de l'implantation sur circuits de type FPGA. Les éléments fonctionnels présents dans un démodulateur sont du même ordre de complexité que ceux qui composent un modulateur et on retrouve, en particulier dans les architectures les plus couramment utilisées, des boucles à verrouillage de phase et des générateurs programmables de sinusoïdes.

Dans l'optique de s'affranchir des algorithmes complexes se trouvant dans les architectures de démodulation numérique, les registres à décalage à rétroaction linéaire, ou LFSR[1], sont introduits. Leurs atouts pour le traitement de signaux sur architecture FPGA sont particulièrement importants. Leur utilisation en tant que compteurs rapides est étudiée et une nouvelle méthode pour générer facilement une paire compteur/décompteur associés est proposée.

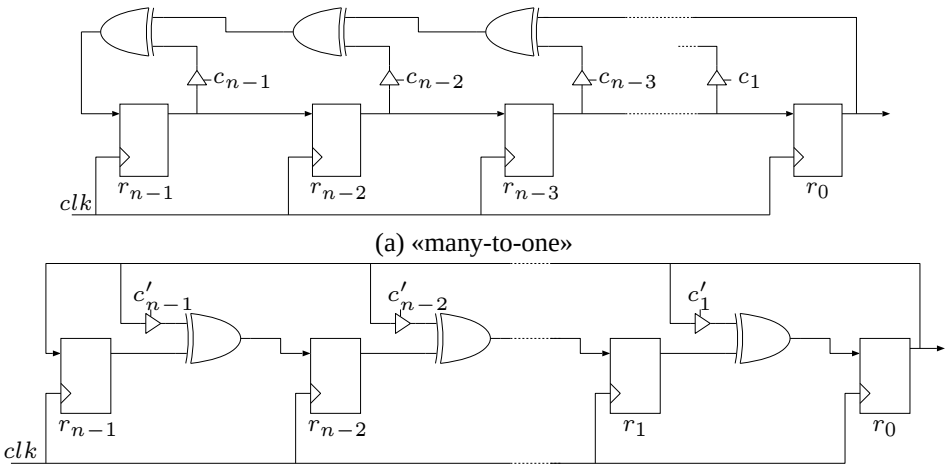
Ces registres sont ensuite utilisés au cœur de deux nouvelles architectures, l'une

[1] *Linear Feedback Shift Register*

pour la démodulation de fréquence, l'autre pour la démodulation de phase. Ces deux systèmes sont particulièrement bien adaptés aux circuits numériques synchrones. Ils atteignent en particulier de très bonnes performances sur les circuits FPGA.

3.1 Registres à décalage à rétroaction linéaire

Dans cette section sont présentés les principes et les fondements mathématiques d'une structure couramment utilisée dans les architectures permettant le traitement de codes correcteurs d'erreur [LIN83, BERL68], ou plus généralement pour la production de séquences pseudo-aléatoires [SCHO60, THOM05]. Un registre à décalage à rétroaction linéaire – ou LFSR[2] est un type de registre dont l'état des entrées ne dépend que d'une combinaison linéaire de l'état de ses cellules à l'instant précédent.



(a) «many-to-one»

(b) «one-to-many»

FIG. 3.1: Les deux architectures de LFSR.

Un registre est contrôlé par une horloge qui lui est externe. Dans notre cas, il s'agira de l'horloge système du circuit sur lequel cette architecture est implantée. Ces registres ont des propriétés particulièrement intéressantes, qui permettent entre autre d'envisager leur utilisation en tant que compteurs programmables. Les deux types d'architectures permettant de réaliser des LFSR sont présentées sur la figure 3.1. Un choix judicieux des coefficients $\{c_i\}_{1\leq i<n}$ et $\{c'_i\}_{1\leq i<n}$ permet d'obtenir de ces deux architectures un comportement équivalent, mais l'architecture de la figure 3.1b est la plus adaptée à

[2]Linear Feedback Shift Register

l'implantation sur circuit numérique [GORE02]. Les détails concernant ce dernier point sont présentés dans la section 3.1.4.

Les LFSR peuvent être utilisés pour des opérations de démodulation, en tirant parti de leurs propriétés de périodicité qui sont examinées dans les sections suivantes.

3.1.1 Principe mathématique

L'ensemble des principes présentés dans cette section utilisent l'architecture *many-to-one* de la figure 3.1a car celle-ci permet une présentation plus simple des équations. Les propriétés énoncées dans ce paragraphe peuvent également être appliquées à la forme *one-to-many* en utilisant les relations d'équivalence [GORE02].

Le registre est initialisé avec la séquence $R_0 = (r_0, r_1, \ldots, r_{n-1})_0$. À chaque itération, le contenu des cellules est décalé vers la droite, et celui de la cellule la plus à gauche, r_{n-1}, est remplacé par une combinaison linéaire des contenus de certaines des cellules du registre.

En conservant les notations présentées sur la figure 3.1a, il est possible d'écrire l'état R_k des cellules d'un LFSR après k itérations en utilisant le système d'équations récurrentes suivant :

$$\begin{cases} r_{k+1,i} = r_{k,i+1}, \text{ pour } i \leq 0 \leq n-2 \\ r_{k+1,0} = \sum_{j=1}^{n-1} c_j \cdot r_{j,0} \end{cases} . \tag{3.1}$$

Ce système peut également être représenté sous la forme matricielle équivalente :

$$R_{k+1} = T \cdot R_k \tag{3.2}$$

$$\begin{bmatrix} r_{k+1,0} \\ r_{k+1,1} \\ \vdots \\ r_{k+1,n-2} \\ r_{k+1,n-1} \end{bmatrix} = \begin{bmatrix} 1 & c_1 & \dots & c_{n-2} & c_{n-1} \\ 0 & 0 & \dots & 0 & 1 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_{k,0} \\ r_{k,1} \\ \vdots \\ r_{k,n-2} \\ r_{k,n-1} \end{bmatrix}. \tag{3.3}$$

Avec cette notation les k itérations effectuées sur le vecteur R_0 contenant les valeurs initiales du LFSR se traduisent par l'opération matricielle :

$$R_k = T^k \cdot R_0. \tag{3.4}$$

3.1.2 Périodicité, polynôme minimal et polynôme primitif

Les vecteurs R_i, dans le cas d'une implantation sur circuit numérique, prennent leurs valeurs dans le corps de Galois $\mathbb{GF}(2^n)$, où n est la longueur du registre utilisé. Il s'agit d'un corps *fini* et les opérations récurrentes sur cet ensemble sont périodiques [LIDL86].

Plus particulièrement l'opération réalisée par le LFSR est périodique, et il existe par conséquent une valeur $m \in \mathbb{N}$ telle que $R_m = R_0$, où m définit la période du LFSR.

Il est possible de garantir que cette période soit maximale [SCHO60, LIN83, ALFK95]. À cet effet, on définit un polynôme $P(X)$ sur ce corps, appelé *polynôme caractéristique* d'un LFSR de la façon suivante :

$$P(X) = X^n + \sum_{i=0}^{n-1} c_i \cdot X^i. \tag{3.5}$$

Les caractéristiques de la séquence $\{R_i\}_{i \geq 0}$ dépendent de celles de $P(X)$. En particulier, la période de cette séquence est maximale dans le cas où P est un polynôme primitif sur $\mathbb{GF}(2)$ [WATS62]. Cela signifie que P est de degré minimal (en l'occurrence n) et

qu'il est irréductible (il n'a aucun autre diviseur que l'unité et lui-même) [GOLO82]. La période maximale d'une telle séquence vaut alors :

$$m_{max} = 2^{n-1}. \tag{3.6}$$

Le plus petit polynôme caractéristique, et donc le plus petit LFSR, permettant de générer la suite de $\{R_i\}_{0 \leq i < m}$ peuvent être calculés si au moins n éléments r_i consécutifs sont connus. Cette opération utilise un l'algorithme de Berlekamp - Massey [BERL68, MASS69] présenté dans la section suivante.

Dans le cas d'un LFSR associé à un polynôme générateur minimal et primitif, le registre parcours, pendant l'ensemble des itérations, une et une seule fois chacun des éléments de l'espace vectoriel $\mathbb{GF}(2^n)$ excepté l'élément nul. L'ordre de parcours ne dépend que du polynôme P.

3.1.3 Algorithme de Berlekamp-Massey

Afin de générer des registres LFSR capables de produire une séquence de bits déterminée, Berlekamp [BERL68] a mis au point un algorithme itératif permettant de décoder les codes BCH[3]. Massey, dans [MASS69], a proposé une amélioration de cet algorithme dans le contexte plus spécifique des codes linéaires et a démontré qu'il pouvait être utilisé pour trouver le plus petit LFSR engendrant une suite linéaire donnée $\{r_i\}_{0 \leq i < n}$. Des améliorations de cet algorithmes peuvent être trouvées dans [ARNA04].

À l'itération d'ordre j cet algorithme produit le polynôme caractéristique du LFSR minimal permettant de générer les j premiers éléments de la séquence $\{R_{i,j}\}$. Il utilise pour cela le polynôme déjà calculé permettant de générer les $j-1$ premiers éléments.

[3] Bose, Ray-Chaudhuri, Hocquenghem

Cet algorithme produit à la demande un LFSR dont la sortie binaire correspond au mieux à l'application dans laquelle il est utilisé. Il est plus particulièrement utilisé dans le logiciel développé pour la description automatique de nos architectures de démodulation, afin de constituer des LFSR de périodicité donnée.

3.1.4 Architectures matérielles pour l'implantation de LFSR

Nous avons présenté sur la figure 3.1a l'une des architectures utilisées pour l'implantation matérielle de registres LFSR. Cette architecture est dite *externe* ou *many-to-one* [GORE02] puisque le fan-in de la fonction combinatoire de rétroaction peut être de 1 à n entrées en fonction de la complexité du polynôme caractéristique.

Ceci pose des problèmes dans le cas d'une optimisation architecturale, car les performances d'un tel système sont alors dépendantes du polynôme générateur $P(X)$. Sur le FPGA utilisé pour nos implantations, chaque cellule logique élémentaire est structurée sous forme d'un bloc combinatoire, comportant trois ou quatre entrées associé à un registre [ALFK95]. Les polynômes complexes risquent d'occuper plusieurs cellules et les délais de routage entre ces cellules limitera la fréquence de fonctionnement de l'architecture globale, et à plus forte raison les performances du système résultant.

Aussi, il est possible d'implanter la même fonction séquentielle que celle réalisée par le LFSR de la figure 3.1a en utilisant une architecture pour laquelle la complexité logique de la fonction combinatoire de rétroaction se trouve répartie de façon équitable entre toutes les cellules du registre. Cette architecture est dite *interne* ou *one-to-many* et est également présentée dans [GORE02] et sur la figure 3.1b.

Le polynôme représentatif de l'état des cellules d'un registre de longueur n, après i itérations, devient

$$R'_i(X) = \sum_{j=0}^{n-1} r'_j \cdot X^j. \tag{3.7}$$

Une transformation polynomiale simple permet de passer d'une représentation à l'autre, de façon bijective [GORE02], en écrivant

$$\begin{cases} r'_j &= c_{j+1} \cdot r_0 + r_{j+1}, \text{ si } j < n-1, \\ r'_{n-1} &= r_0, \text{ sinon.} \end{cases} \tag{3.8}$$

C'est ce type d'implantation qui sera exclusivement choisi pour la suite puisqu'il est mieux adapté aux circuits FPGA.

3.1.5 Compteurs à base de LFSR

Les LFSR, de par leur structure à base de registres et de fonctions logiques simples, sont particulièrement bien adaptés à une implantation sur FPGA [ALFK95], car leur architecture est proche de la structure interne des cellules logiques des FPGA présentés dans 1.4 et [FLEX03]. De plus, le fait qu'ils soient capables de produire des séquences de longueur fixes et périodiques, comme cela a été démontré dans la section 3.1.2, en fait un choix pertinent pour une utilisation en tant que *compteurs* [STAN97].

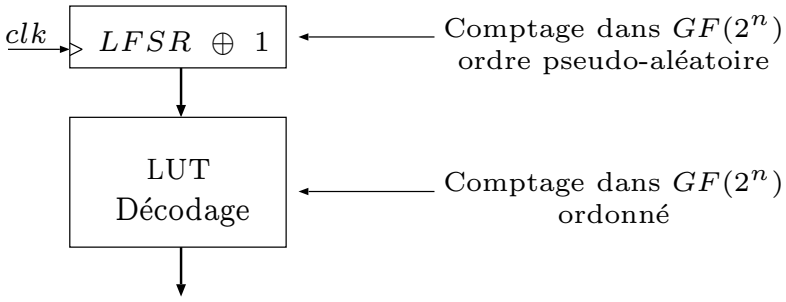


FIG. 3.2: Principe d'un compteur basé sur un LFSR

L'organisation fonctionnelle de l'un de ces compteur est présentée sur la figure 3.2. Il est composé de deux parties principales :

– un registre LFSR contrôlé par un polynôme primitif, de degré n ;

– une ROM de décodage.

Le rôle de cette ROM est de permettre la conversion entre l'ordre de parcours des éléments de $\mathbb{GF}(2^n)$ et l'ordre «traditionnel» croissant utilisé dans les compteurs classiques. La seule contrainte pour que ce compteur soit fonctionnel est de choisir un état initial R_0 des cellules du registre différent de l'élément nul. Dans le cas contraire, le contenu du registre n'évolue jamais.

Une architecture respectant ces critères se comporte comme un compteur modulo $2^n - 1$. Ce compteur, si le LFSR utilisé est de type *interne*, a l'avantage de ne pas présenter de contraintes liées aux délais de propagation des retenues affectant les compteurs classiques.

3.1.6 Décompteur apparié

Selon le même principe que les compteurs présentés dans la section précédente, j'ai mis au point à l'aide de LFSR des décompteurs ayant la particularité de passer *exactement* par les mêmes états logiques qu'un compteur donné mais avec un sens de parcours inversé.

En l'occurrence, si un compteur $C^{\oplus}$ passe par les états successifs $R_0, R_1, R_2, \ldots R_{2^n-1}$ après avoir été initialisé avec la valeur R_0, le décompteur $C^{\ominus}$ apparié comptera lui de la façon suivante :

$$R_{2^n-1},\ R_{2^n-2},\ \ldots\ R_0, \tag{3.9}$$

si ses cellules ont été initialisées avec la valeur R_{2^n-1}.

Pour un LFSR défini par la matrice T de l'équation 3.3, la matrice

$$T^{\ominus} = \begin{bmatrix} 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 1 & 0 & \dots & 0 & 0 \\ c_1 & c_2 & \dots & c_{n-1} & 1 \end{bmatrix}, \tag{3.10}$$

est caractéristique du décompteur associé.

En effet, le produit de T par $T^{\ominus}$ conduit à la matrice unité I. Par conséquent, $T^{\ominus}$ est la matrice caractéristique du LFSR décompteur apparié à un LFSR compteur défini par la matrice de l'équation 3.3.

Ce type de LFSR peut être utilisé en complément des LFSR compteurs présentés plus haut.

3.1.7 Conclusion

Les registres à décalage à rétroaction linéaire permettent en particulier de remplacer avantageusement les compteurs dans les architectures numériques dans lesquelles ils sont utilisés. De plus, s'ils sont présents sous leur forme «one-to-many», ils correspondent très bien à la structure interne des circuits FPGA qui nous intéressent ici.

La section suivante démontre l'intérêt de leur utilisation pour les systèmes de démodulation de signaux numériques modulés en phase et en fréquence.

3.2 Application à la démodulation de fréquence

[ARNO06]

On se propose d'implanter les bases d'un système capable de traiter les signaux de type M-FSK[4] à l'aide de registres

Les LFSR n'acceptent que des signaux de contrôle binaires. Or, les signaux que nous nous proposons de démoduler sont échantillonnés et quantifiés : la première étape avant tout traitement est donc la conversion du signal modulé en un signal image présentant la même fréquence et la même phase que le signal original, mais dont l'amplitude ne peut prendre d'autres valeurs que 0 ou 1. En d'autres termes, il s'agit de convertir le signal modulé en un signal purement binaire.

Cette opération est effectuée à l'aide d'un système de détection de seuil intégré au convertisseur analogique numérique présent en amont du circuit numérique sur lequel notre architecture est implantée.

Un signal modulé en fréquence répond à l'équation

$$S(k) = A\cos\left(2\pi f_q \cdot t + \phi_0\right), \tag{3.11}$$

avec $q \cdot \Delta t \leq t \ < (q+1)\Delta t$. Dans cette équation Δt est la durée d'un symbole FSK, ϕ_0 le déphasage du signal et f_q la fréquence du signal pour ce symbole.

Après traitement par le système à détection de seuil, le signal devient :

$$S'(i) = \begin{cases} 1, \text{ si } \frac{1}{2\pi f_q}\left(2m\pi - \phi_0\right) \leq t < \frac{1}{2\pi f_q}\left((2m+1)\pi - \phi_0\right) \\ 0, \text{ dans les autres cas,} \end{cases} \tag{3.12}$$

où $m \in \mathbb{Z}$.

Le résultat de cette transformation est présenté sur la figure 3.3.

[4] *M-ary Frequency Shift Keying* - Codage par Déplacement de Fréquence M-aire.

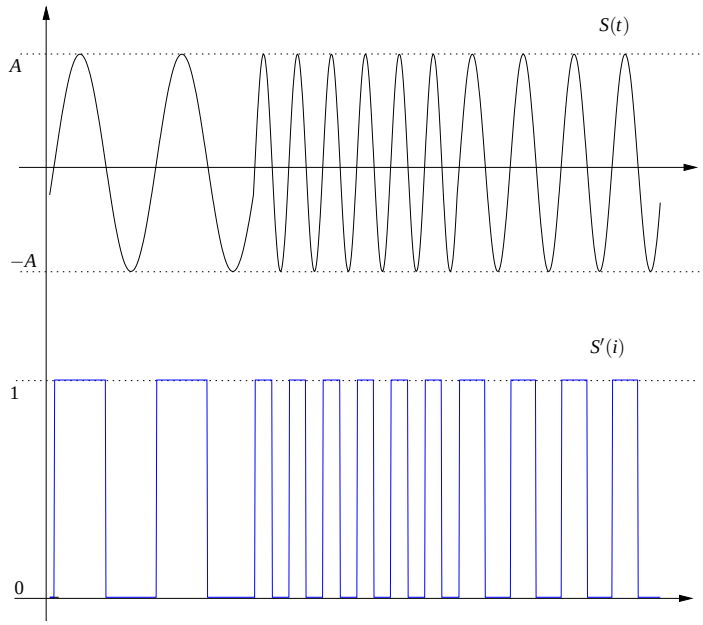


FIG. 3.3: Normalisation d'un signal M-FSK.

3.2.1 Architecture du démodulateur

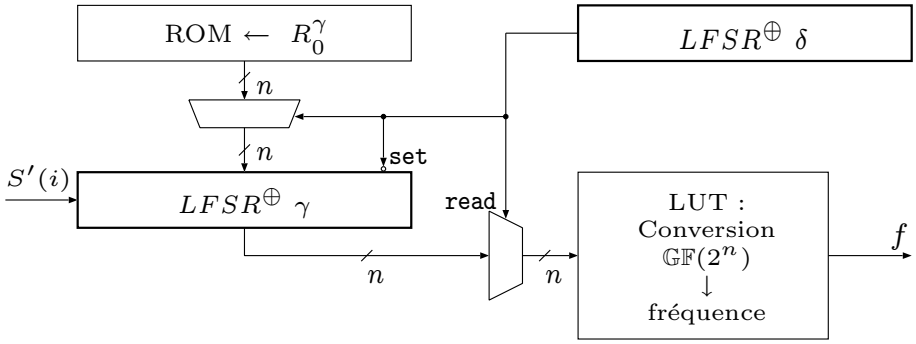


FIG. 3.4: Architecture du démodulateur de fréquence à un étage.

L'architecture proposée pour le démodulateur est présentée sur la figure 3.4. Elle est composée de quatre parties principales, la première étant un registre LFSR, noté $LFSR^\oplus$ γ. Le polynôme générateur de ce registre est de degré n, et prend donc successivement toutes les valeurs d'un sous groupe de $GF(2^n)$ comportant γ^{max} éléments. Il

pourra donc passer par $\gamma^{max} - 1$ états différents, c'est à dire $2^n - 1$ états dans le cas où le polynôme considéré est primitif. Le signal de comptage qui anime le LFSR γ est le signal à démoduler. Celui-ci est noté $S'(i)$ sur la figure 3.4.

Le registre est associé à une petite ROM contenant l'état initial des cellules, noté R_0. Cette ROM est utilisée pour réinitialiser le LFSR γ à sa valeur d'origine. La taille en bits de cette table est $n \cdot f_{res}$, où f_{res} est la résolution binaire du mot de fréquence. Cette fonction est équivalente à la fonction *reset* que l'on peut trouver sur certains compteurs traditionnels.

Le couple formé par le LFSR γ et cette ROM permet de réaliser un comptage programmable dont la durée est fixée par un second compteur implanté avec un LFSR noté $LFSR^{\oplus}\ \delta$. Le polynôme générateur associé à ce second registre est de degré m et permet de délimiter le temps de mesure de la fréquence. La durée totale d'une mesure dépend du nombre δ^{max} d'états par lesquels le LFSR δ va passer pour effectuer un cycle complet. En fonction de la nature du polynôme générateur de ce second LFSR, δ^{max} peut varier entre 1 et $2^m - 1$. Par conséquent, la durée d'une mesure bénéficie d'une grande amplitude de variation.

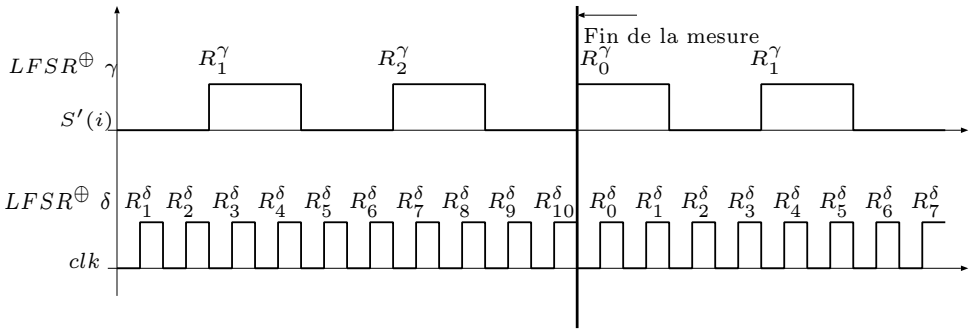


FIG. 3.5: Exemple d'une mesure de fréquence avec deux LFSR.

Une fois que le LFSR δ a effectué un cycle complet, la valeur présente dans le LFSR γ est lue puis est utilisée pour adresser une LUT qui contient une table de correspon-

dance entre l'ensemble des valeurs possibles pour le LFSR γ et la fréquence associée.

Un exemple d'une mesure est présenté sur la figure 3.5. L'état des deux LFSR y est présenté, $LFSR^{\oplus}\ \gamma$ étant incrémenté sur les fronts montants du signal $S'(i)$ et $LFSR^{\oplus}\ \delta$ sur ceux du signal d'horloge clk. R_0^{γ} et R_0^{δ} correspondent à la valeur initiale des cellules de $LFSR^{\oplus}\ \gamma$ et $LFSR^{\oplus}\ \delta$ respectivement, R_1^{γ} et R_1^{δ} à leur valeur après une itération, etc... Dans cet exemple, la période de comptage de δ est fixée à 11 itérations, la mesure prend donc fin une fois que les cellules de ce registre ont atteint la valeur $R_1 1^{\delta}$. La LUT de la figure 3.4 est ainsi adressée avec la valeur R_3^{γ} et la valeur de la fréquence mesurée résultante fournie par cette LUT est $f_{clk}/4$.

La résolution fréquentielle de cette architecture dépend des deux paramètres γ^{max} et δ^{max}. En particulier, la plus petite fréquence mesurable avec ce système est telle que le LFSR γ ne subit qu'un seul et unique décalage pendant toute la durée de la mesure, c'est-à-dire pendant un cycle complet de δ.

Dans ce cas, la fréquence du signal d'entrée $S(t)$ est égale à

$$f_{min} = \frac{1}{T_{clk} \cdot \delta^{max}} = \frac{f_{clk}}{\delta^{max}}, \tag{3.13}$$

T_{clk} correspondant à la période de l'horloge interne du circuit numérique sur laquelle le démodulateur est implanté, et f_{clk} sa fréquence. Avec ces notations, la durée d'une mesure de fréquence est

$$T_{clk} \cdot \delta^{max}. \tag{3.14}$$

De même, la fréquence maximale qui peut-être mesurée par cette architecture est telle que le LFSR γ passe par l'ensemble des états admissibles pour ce registre, c'est à dire γ^{max} états distincts, pendant la durée d'une mesure $T_{clk} \cdot \delta^{max}$. Dans ce cas, la

fréquence mesurée est égale à :

$$f_{max} = \frac{\gamma^{max}}{T_{clk} \cdot \delta^{max}} = \frac{\gamma^{max}}{\delta^{max}} f_{clk}. \tag{3.15}$$

Dans tous les cas, la fréquence maximale admissible en entrée ne peut excéder celle de l'horloge interne du circuit numérique cible, puisque l'ensemble de l'architecture fonctionne en mode *synchrone*, ni celle définie par le critère de Shannon [SHAN48] présenté dans la section 1.2.1.

Il est possible d'augmenter l'intervalle de résolution fréquentielle, en particulier en augmentant la durée de mesure, et donc le paramètre δ^{max}. Mais dans ce cas, la fréquence maximale du signal admise en entrée de l'architecture se trouve d'autant plus réduite que δ^{max} est grand, comme le montre l'équation 3.15.

L'architecture proposée dans le paragraphe suivant correspond à une version parallèle de celle exposée ici. Elle permet d'augmenter la largeur de l'intervalle dans lequel la fréquence du signal d'entrée peut varier, sans pour autant diminuer la borne supérieure de cette plage.

3.2.2 Architecture parallèle

La figure 3.6 présente une nouvelle architecture, qui en lieu et place d'un seul LFSR γ permettant de mesurer la fréquence de S, utilise un banc de N LFSR numérotés de $LFSR^{\oplus}$ γ_0 à $LFSR^{\oplus}$ γ_{N-1} Le LFSR de rang i est piloté par un polynôme de degré n_i, et il peut prendre en tout γ_n^i états distincts. Comme précédemment γ_n^i est maximal dans le cas où le polynôme caractéristique est minimal et primitif. Dans ce cas $\gamma_n^i = 2^{n_i} - 1$.

Le compteur $LFSR^{\oplus}$ δ n'est pas modifié, par contre la table LUT chargée de la conversion entre les valeurs mesurées par les $\{R_j^{\gamma_i}\}_{\substack{0 \leq i < N \\ 0 \leq j \leq \gamma_n^i}}$ et les valeurs de fréquences

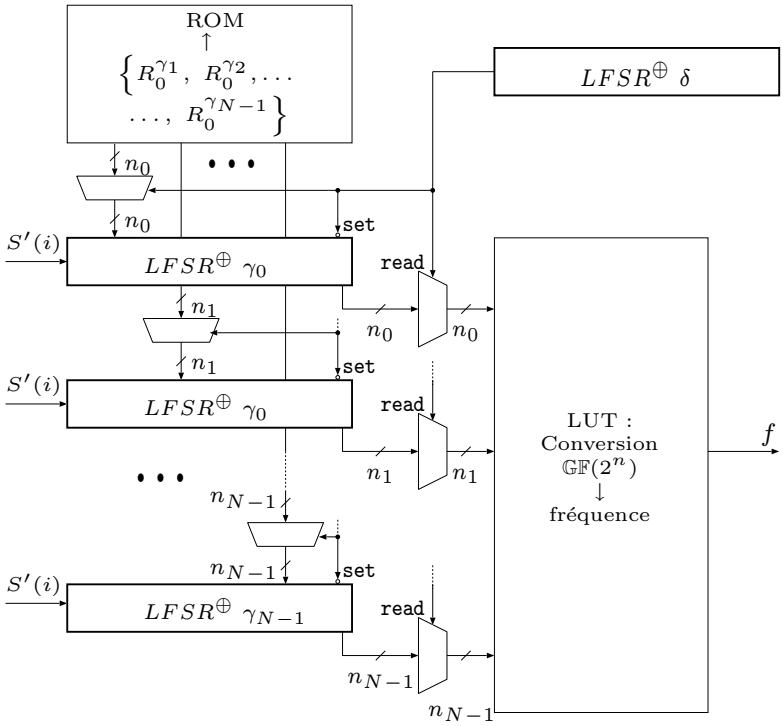


FIG. 3.6: Architecture parallèle du démodulateur de phase.

correspondantes se trouve agrandie. Elle comporte maintenant

$$t'_{LUT} = f_{res} \cdot \sum_{i=0}^{N-1} n_i \text{ bits.} \tag{3.16}$$

Les N registres présentés dans cette architecture sont déclenchés simultanément par le signal d'entrée modulé et converti selon l'équation 3.12, $S'(i)$. Les LFSR γ_j parcourent pendant un cycle complet γ_0^{max}, γ_1^{max}, $\ldots \gamma_{k-1}^{max}$ éléments de $\mathbb{GF}(2^j)$ respectivement. En notant $R_i^{\gamma_j}$, l'état du registre d'ordre j après i itérations, l'état $\{R\}_i$ du système composé de l'ensemble des registres après ces mêmes i itérations est représenté par le

système :

$$\{R\}_i = \left\{ \begin{array}{c} R_i^{\gamma_0} \pmod{\gamma_0^{max}} \\ R_i^{\gamma_1} \pmod{\gamma_1^{max}} \\ \cdots \\ R_i^{\gamma_{N-1}} \pmod{\gamma_{N-1}^{max}} \end{array} \right\}. \tag{3.17}$$

Si les polynômes générateurs des différents LFSR sont choisis de façon à ce que les $\{\gamma_i^{max}\}_{0 \leq i < N}$ soient premiers entre eux, le système dans son ensemble peut prendre un nombre maximal d'états distincts égal à :

$$\Gamma^{max} = \prod_{i=0}^{N-1} \gamma_i^{max}. \tag{3.18}$$

La nouvelle fréquence maximale que cette architecture est capable d'atteindre est donc

$$f'_{max} = \frac{\Gamma^{max}}{T_{clk} \cdot \delta^{max}}. \tag{3.19}$$

Comme précédemment, pour respecter les limites de Shannon il est nécessaire que $\Gamma^{max}/\delta^{max} < 1/2$.

À condition de disposer de sérialiseurs/désérialiseurs rapides, cette architecture parallèle peut être transformée pour mesurer des fréquences supérieures à la fréquence d'horloge du système hôte.

En effet, dans le cas où les échantillons du signal d'entrée sont répartis successivement sur chacun des LFSR comme le présente la figure 3.7, la fréquence «apparente» que chaque LFSR doit mesurer sera divisée par le nombre de LFSR utilisés. Si f est la fréquence effective du signal $S(t)$ et f_i la fréquence apparente mesurée par $LFSR^{\oplus}\ \gamma_i$, on aura :

$$f = \sum_{i=0}^{N-1} f_i. \tag{3.20}$$

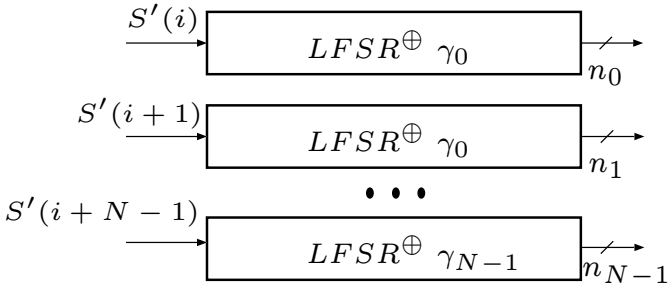


FIG. 3.7: Utilisation d'un désérialiseur rapide.

Avec ce système, il est envisageable de mesurer des fréquences couvrant la plage

$$\frac{1}{\delta_{max}} \cdot f_{clk} \leq f \leq \frac{1}{N} \cdot \frac{\gamma_{max}}{\delta_{max}} \cdot f_{clk}. \tag{3.21}$$

3.2.3 Implantation et résultats

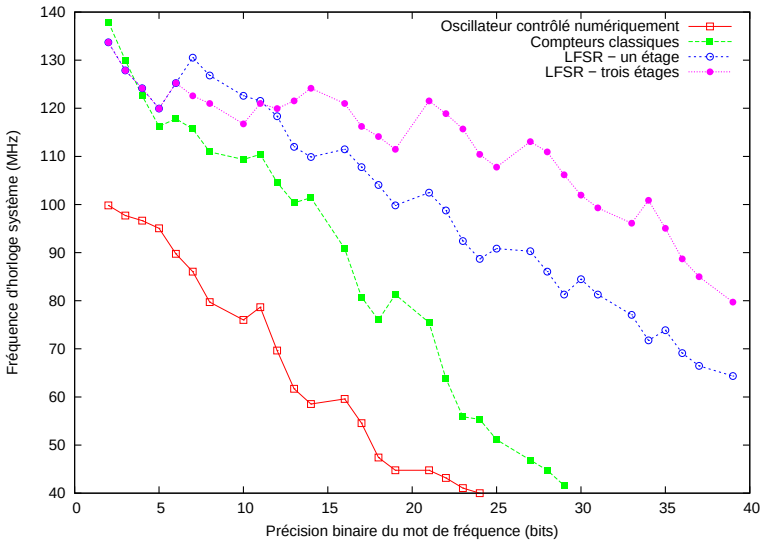


FIG. 3.8: Fréquence d'échantillonnage pour différents types d'architectures de démodulation de fréquence.

Le démodulateur de fréquence a été implanté sur un FPGA de type FLEX10K200E, et les résultats de la synthèse sont présentés sur les figures 3.8, 3.9 et 3.10. Pour fournir

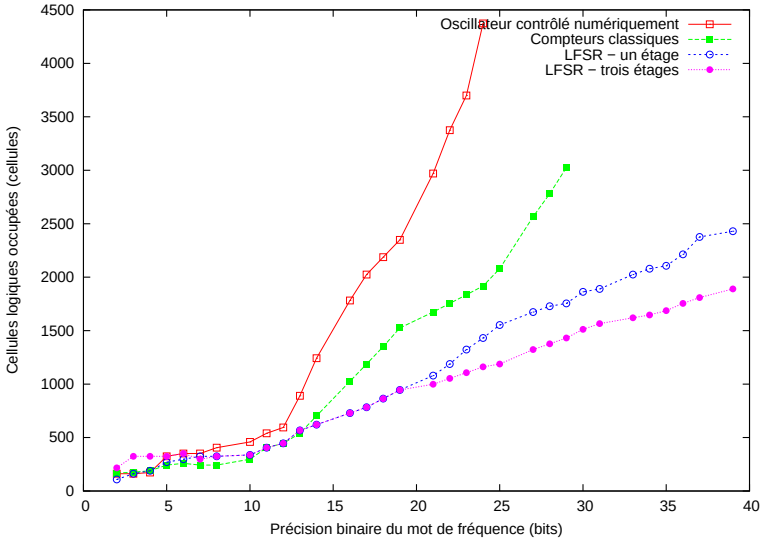


FIG. 3.9: Nombres de cellules logiques occupées en fonction de la résolution binaire.

des bases de comparaison, d'autres architectures de démodulation de fréquence on également été implantée sur le même type de FPGA : une architecture performante utilisant un générateur numérique de sinusoïdes [LEE06], une architecture utilisant des compteurs «classiques» et nos deux architectures : à un «étage» et parallèle à trois «étages».

Le logiciel présenté dans la section 2.4.2 est utilisé pour générer automatiquement les descriptions VHDL adéquates et pour calculer le contenu des LUT de décodage. Il utilise l'algorithme de Berlekamp-Massey pour calculer les plus petits polynômes caractéristiques permettant d'obtenir les compteurs requis.

La figure 3.8 présente la fréquence système maximale atteinte par les différents systèmes, lorsqu'on modifie la résolution en bit de la fréquence mesurée. La fréquence d'horloge maximale atteinte par les deux architectures à base de LFSR reste supérieure à celle atteinte par les autres architectures dans toute la gamme de précision binaire du mot de fréquence. La structure simple des étages de mesure à base de LFSR, proche de celle des éléments logiques composant un FPGA, explique les bonnes performances

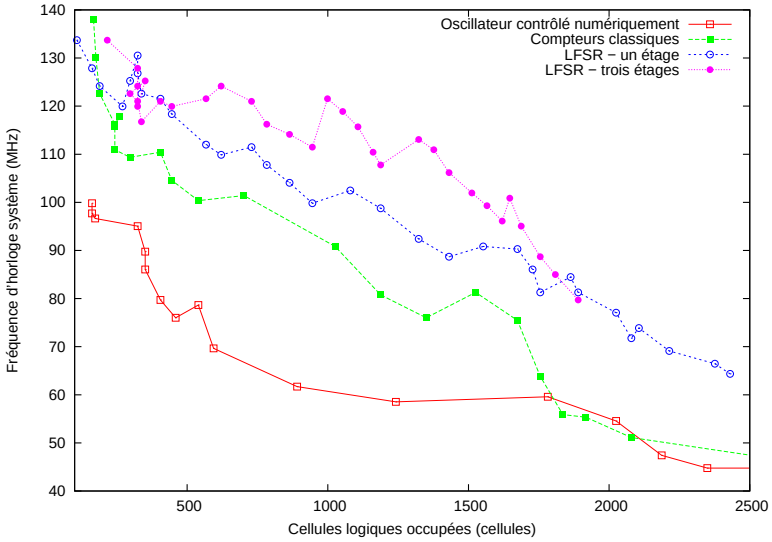


FIG. 3.10: Fréquence d'échantillonnage en fonction du nombre de cellules logiques occupées.

obtenues. À partir d'une précision de 12 bits les mécanismes de propagation de retenues utilisés dans les compteurs classiques commencent à limiter de façon importante la fréquence d'horloge atteinte. Enfin, l'architecture utilisant un oscillateur contrôlé numériquement souffre des mêmes contraintes que celles présentées dans le chapitre 2 : le nombre d'opérations arithmétiques complexes à effectuer et la taille des blocs combinatoires nécessaires pour implanter ces opérations s'accroissent avec la précision demandée. La fréquence de fonctionnement du FPGA cible est donc amoindrie.

La figure 3.9 montre le nombre de cellules logiques élémentaires consommées par les différentes architectures, en fonction de la précision du mot de fréquence. Les trois architectures à base de compteurs (classiques ou basés sur des LFSR) occupent un nombre comparable de cellules logiques. L'architecture à un seul étage est bien sûr avantagée par sa simplicité. Par contre, l'architecture utilisant un générateur de sinusoïdes est très gourmande en ressources comparativement aux trois autres. Là aussi, la quantité et la taille des opérateurs arithmétiques utilisés expliquent cette différence im-

portante. Notre architecture parallèle s'avère être plus efficace que l'architecture simple pour des précisions élevées (plus de 12 bits). Ceci s'explique par le fait qu'elle utilise trois registres de petite taille en lieu et place d'un seul gros registre pour effectuer ses mesures et l'implantation de ces trois petits registres consomme moins de ressources.

Enfin, la figure 3.10 la fréquence maximale atteinte en fonction de la quantité de cellules logiques occupées. Celle-ci est utilisée comme indicateur de performances pour les architectures testées. Les deux architectures à base de LFSR ont une progression pratiquement linéaire et sont particulièrement performantes lorsque le nombre de cellules occupées est supérieur à 1000. De même, l'architecture parallèle montre son intérêt en présentant le meilleur rapport fréquence atteinte par rapport aux nombre de cellules occupées, à partir de 1200 cellules utilisées environ, ce qui est cohérent avec les résultats de la figure 3.9.

3.3 Architecture pour la démodulation entièrement numérique de la phase

Sur le modèle de l'architecture de démodulation de signaux M-FSK présentée dans la section précédente, on se propose de définir une nouvelle architecture de mesure de la phase d'un signal modulé selon une modulation de type M-PSK[5].

Un signal modulé M-PSK répond à l'équation :

$$S(t) = A\sin\left(2\pi f \cdot t + \phi_q\right), \tag{3.22}$$

avec $q \cdot \Delta t \leq t < (q+1) \cdot \Delta t$. Comme précédemment, Δt est la durée d'un symbole M-PSK et ϕ_q la phase associée à ce symbole.

[5] *M-ary Phase Shift Keying* - Codage M-aire par Déplacement de Phase

Le signal d'entrée est tout d'abord converti en une forme purement binaire, qui autorise un traitement simplifié sur un circuit numérique à l'aide d'opérations combinatoires simples. La nouvelle équation définissant le signal $S(t)$ après échantillonnage, quantification et cette dernière transformation est :

$$m(n) = \begin{cases} 1, \text{ si } \frac{2k\pi-\phi_q}{2\pi f} \leq t < \frac{(2k+1)\pi-\phi_q}{2\pi f} \\ 0, \text{ dans les autres cas,} \end{cases} \tag{3.23}$$

Après transformation $m(n)$ conserve la même fréquence et les mêmes phases que le signal original avec la précision permise par la fréquence d'échantillonnage utilisée.

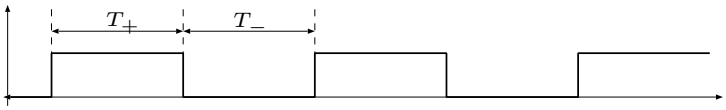
3.3.1 Architecture du démodulateur

Le démodulateur de phase utilise les mêmes principes de base que le démodulateur de fréquence présenté plus haut. Il utilise des registres LFSR fonctionnant en mode compteur ou décompteur pour mesurer la *régularité* du signal $m(n)$ qu'il est chargé de traiter. En d'autres termes un changement de phase sur un signal binaire carré se traduira par une rupture de la régularité des périodes pendant lesquelles le signal est à l'état haut et celles pendant lesquelles il est à l'état bas. Dans le cas d'un signal ne subissant aucun déphasage la durée de ces périodes doit être identique, ou pratiquement identique du fait des artefacts pouvant être introduits par l'opération d'échantillonnage.

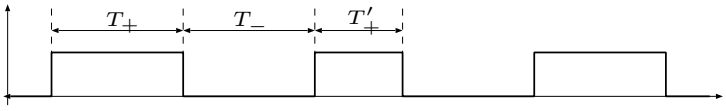
Cette différence est présentée sur la figure 3.11. Sur le chronogramme 3.11a, $T_+ = T_-$ le signal conserve la même phase sur toute la durée du graphe. Par contre, sur la figure 3.11b, $T'_+ > T_-$ ce qui dénote un déphasage $\Delta\phi$. Celui-ci peut-être évalué de la façon suivante

$$\Delta\phi = 2\pi f \cdot (T'_+ - T_-) \tag{3.24}$$

L'architecture proposée pour notre démodulateur de phase est présentée sur la fi-



(a) Signal non déphasé



(b) Signal avec un déphasage

FIG. 3.11: Durées des états haut et bas pour un signal non déphasé, et un signal avec un déphasage.

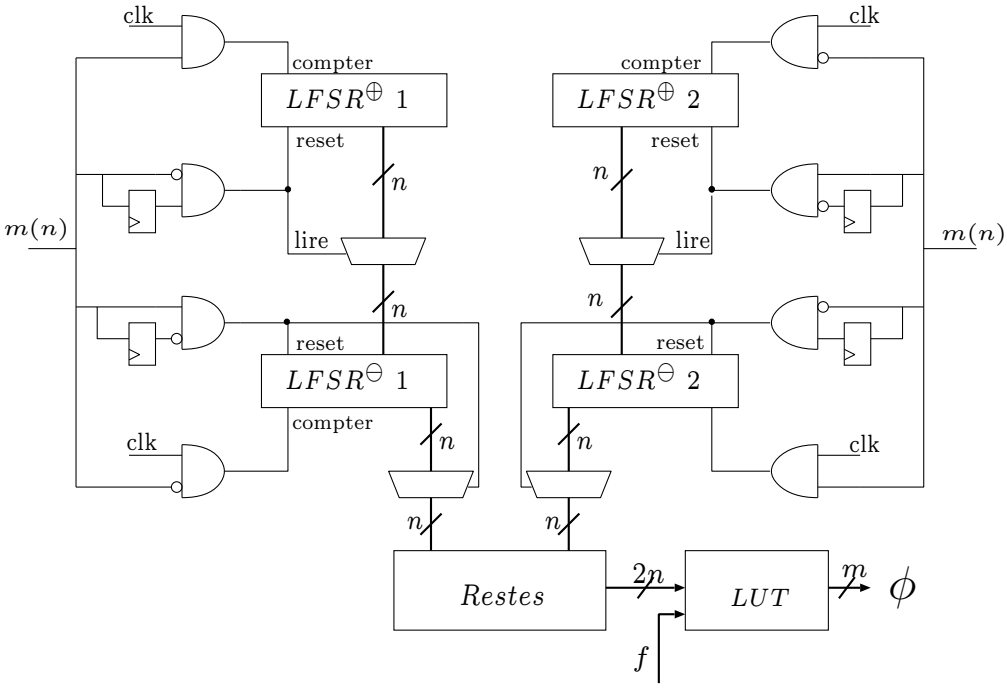


FIG. 3.12: Architecture du démodulateur de phase

gure 3.12. Ce démodulateur est composé principalement de quatre registres LFSR, notés $LFSR^{\oplus}$ 1, $LFSR^{\oplus}$ 2, $LFSR^{\ominus}$ 1 et $LFSR^{\ominus}$ 1.

Les $LFSR^{\oplus}$ 1 et $LFSR^{\oplus}$ 2 fonctionnent tous les deux en mode compteur, et partagent le même polynôme générateur. Par conséquent, leur cycle de comptage est identique. La valeur initiale du contenu des cellules de ces deux LFSR est notée R_0, et la valeur après

i itérations R_i.

Les $LFSR^{\ominus}$ 1 et $LFSR^{\ominus}$ 2 fonctionnent en mode décompteur. Comme dans le cas des deux LFSR compteurs, ils sont identiques, et partagent le même polynôme générateur. De plus, ils sont *appariés* aux deux LFSR $\oplus$ 1 et $\oplus$ 2, ce qui signifie qu'à chaque itération, leur contenu sera modifié d'un facteur R_{-1} : si à l'itération i, le contenu de l'un de ces registres est R_k, il sera R_{k-1} après l'itération suivante.

- Le $LFSR^{\oplus}$ 1 fonctionne en tant que compteur synchrone. Il n'est incrémenté que dans le cas où $m(n)$ est à l'état logique 1. Dans ce cas, à chaque période T_{clk} de l'horloge interne du circuit numérique sur lequel cette architecture est implantée, $LFSR^{\oplus}$ 1 subit une itération. Si $m(n)$ reste à l'état logique 1 pendant k périodes d'horloge, le contenu de ce registre sera :

$$\{LFSR^{\oplus}\ 1\} = R_k. \tag{3.25}$$

  Le contenu de ce registre est remis à sa valeur initiale R_0 après détection d'un front descendant de $m(n)$. Ce LFSR permet donc de mesurer la durée pendant laquelle $m(n)$ reste à l'état haut.
- Le $LFSR^{\oplus}$ 2 a un comportement exactement similaire à celui de $LFSR^{\oplus}$ 1. Simplement, il est incrémenté lorsque $m(n)$ est à un état logique bas, et est remis à son état initial R_0 lors de la détection d'un front montant de $m(n)$. À la différence du premier compteur, celui-ci permet de mesurer la durée d'un état logique bas de $m(n)$, et l'état de ses cellules pour un signal $m(n)$ ayant duré $k \cdot T_{clk}$ est noté $\{LFSR^{\oplus}\ 1\} = R_k$.
- Le $LFSR^{\ominus}$ 1 est sensible aux états logiques bas de $m(n)$, tout comme $LFSR^{\oplus}$ 2. Mais il s'agit d'un décompteur, qui est *initialisé* avec le contenu du registre $LFSR^{\oplus}$ 1 au moment où le signal $m(n)$ passe de l'état haut à l'état bas. Par exemple, si

$LFSR^{\oplus}$ 1 contenait la valeur R_k juste avant le front descendant dénotant pour $m(n)$ le passage de l'état haut à l'état bas et donc le début du décomptage par $LFSR^{\oplus}$ 2, ce dernier sera initialisé à R_k et prendra successivement les valeurs R_k, $R_{k-1}, \ldots$.

– Le $LFSR^{\ominus}$ 2 est un décompteur dont le comportement est proche de celui de $LFSR^{\ominus}$ 1. Par contre il réagit aux états logiques hauts de $m(n)$ en lieu et place des états logiques bas et la valeur initiale de ses cellules correspond à celle des cellules de $LFSR^{\oplus}$ 2 juste avant un front montant de $m(n)$.

3.3.2 Mesure du déphasage

Un déphasage est détecté si, avec l'une des deux paires de compteurs/décompteurs, une irrégularité dans la durée des périodes à l'état haut ou à l'état bas du signal $m(n)$ est constatée.

Pour le premier couple de compteurs/décompteurs, une période de $m(n)$ commence par un front montant indiquant le passage de ce signal à l'état haut. Cette période se termine au front montant suivant. Si aucun déphasage n'intervient pendant un cycle de comptage sur une période de $m(n)$ dont la durée est $2 * k$ cycles :

– $LFSR^{\oplus}$ 1 compte de R_1 à R_k pendant la demi-période où $m(n)$ est à l'état haut ;

– $LFSR^{\ominus}$ 1 décompte de R_k à R_1 pendant la demi-période où $m(n)$ est à l'état bas.

Il y a déphasage si le contenu de $LFSR^{\ominus}$ 1 est différent de R_1 au moment du front montant de $m(n)$ marquant le début d'une nouvelle période.

Le comportement de la seconde paire de compteur est symétrique de celui décrit ci-dessus : pour cette paire, une période de $m(n)$ est délimitée par deux fronts descendants au lieu de deux fronts montants. Là aussi, si le contenu de $LFSR^{\ominus}$ 2 n'est pas égal à R_1 à la fin d'une période, un déphasage a été détecté.

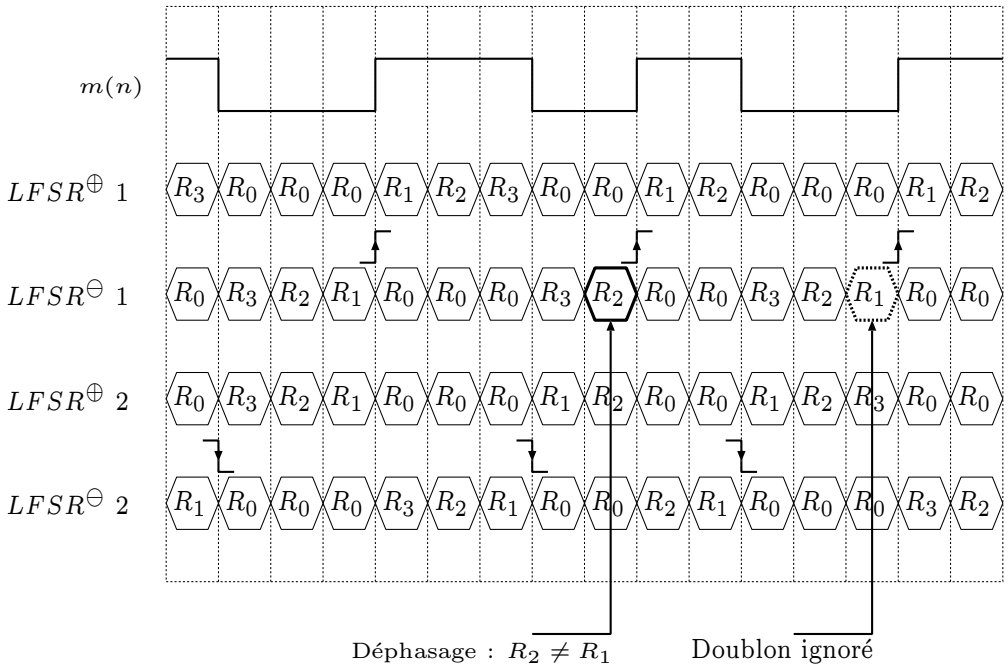


FIG. 3.13: Détection d'un déphasage - premier chronogramme

Les figures 3.13 à 3.15 présentent les trois cas de détection d'un déphasage.

Un exemple d'irrégularité ne pouvant être détectée que par la paire de compteurs $(LFSR^{\oplus}1, LFSR^{\ominus}1)$ est présenté sur la figure 3.13. Au moment du second front montant de $m(n)$ le contenu de $LFSR^{\ominus}$ 1 est R_2, ce qui indique un déphasage. Le même déphasage est détecté par la paire de compteurs 1 au moment du front montant suivant : il sera ignoré par la logique de décodage de l'architecture présentée sur la figure 3.12.

Le déphasage présenté sur la figure 3.14 ne peut être détecté que par la seconde paire de compteurs, puisque la première ne perçoit pas d'irrégularité dans la succession d'états hauts et bas de $m(n)$. À l'instant du second front descendant de $m(n)$ le contenu des cellules de $LFSR^{\ominus}$ 2 est R_2 au lieu de R_1. Un déphasage est donc détecté à ce moment. Comme dans le cas précédent, la seconde détection du même déphasage est ignorée par la logique de décodage.

Enfin, le déphasage présenté sur la figure 3.15 est détecté tout d'abord par la seconde

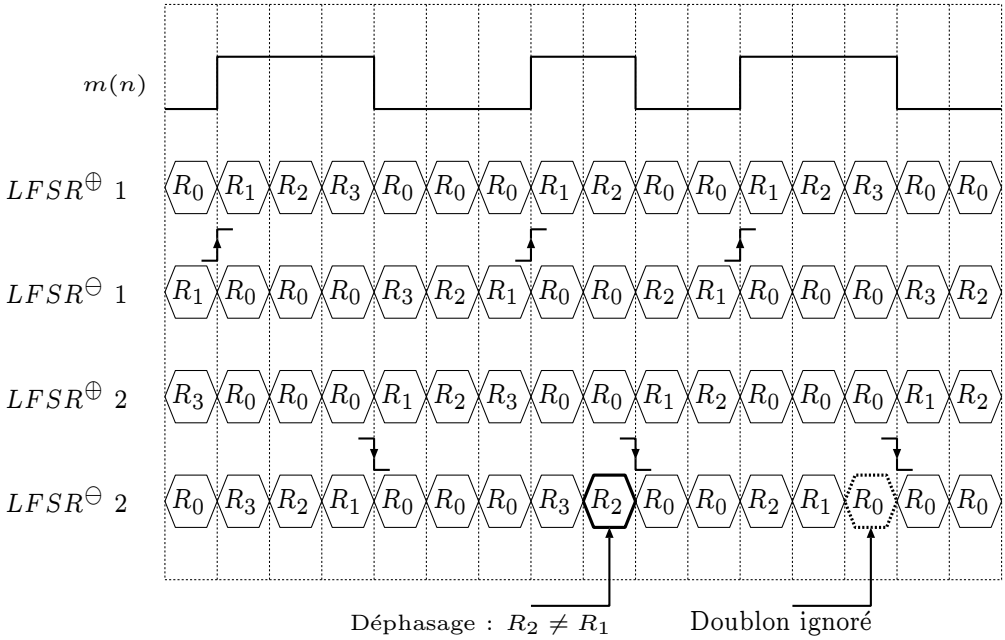


FIG. 3.14: Détection d'un déphasage - second chronogramme

paire de compteurs/décompteurs, puis par la première. Là encore, la seconde détection n'est pas prise en compte par le décodeur.

Lorsqu'un déphasage est détecté le contenu des cellules de $LFSR^{\ominus}$ 1 et $LFSR^{\ominus}$ 2 constituant les «restes» du décomptage permettent de calculer le déphasage de $m(n)$, à l'aide de l'équation 3.24. Ce déphasage dépend encore de la fréquence du signal $m(n)$, mais celle-ci peut-être évaluée à l'aide de l'architecture de démodulation de fréquence présentée dans la section précédente utilisée conjointement avec les valeurs de «restes» pour évaluer le déphasage de $m(n)$. Cette conversion finale est effectuée par une LUT contenant la table de conversion entre les couples de valeurs (R_i, f) et la phase ϕ.

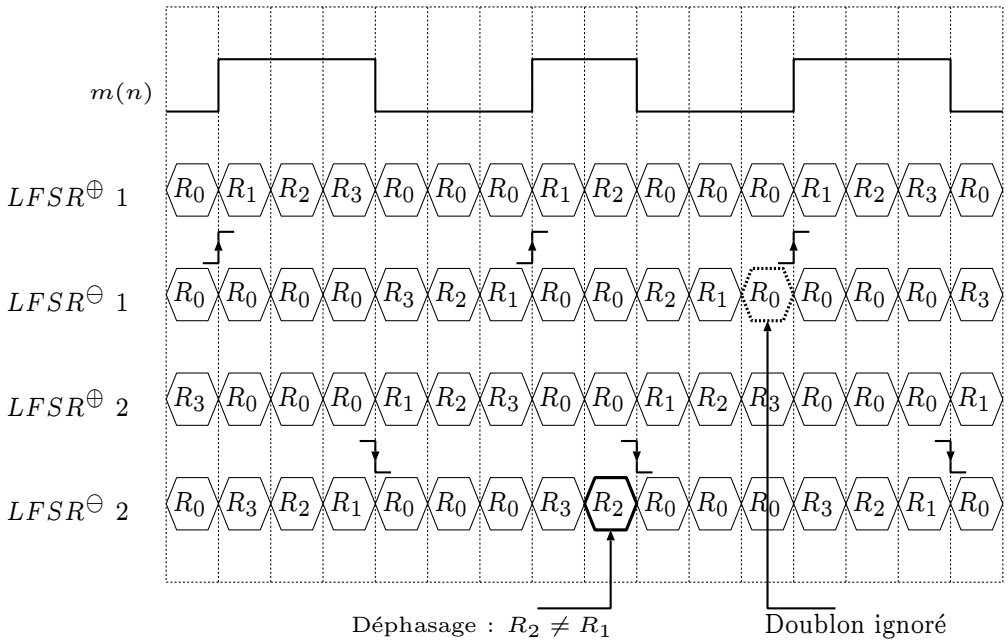


FIG. 3.15: Détection d'un déphasage - troisième chronogramme

3.3.3 Implantation et résultats

Les résultats de l'implantation de l'architecture de démodulation de phase sur un FPGA FLEX10K200E sont présentés dans cette section. Ils sont comparés avec les performances d'autres architectures numériques de démodulation présentes dans la littérature en particulier celles de [SAMU90, HENT93, PRIY03], qui utilisent un oscillateur contrôlé numériquement pour supprimer la fréquence porteuse du signal modulé, et celle présentée dans [XUE05] qui utilise un système de sur-échantillonnage.

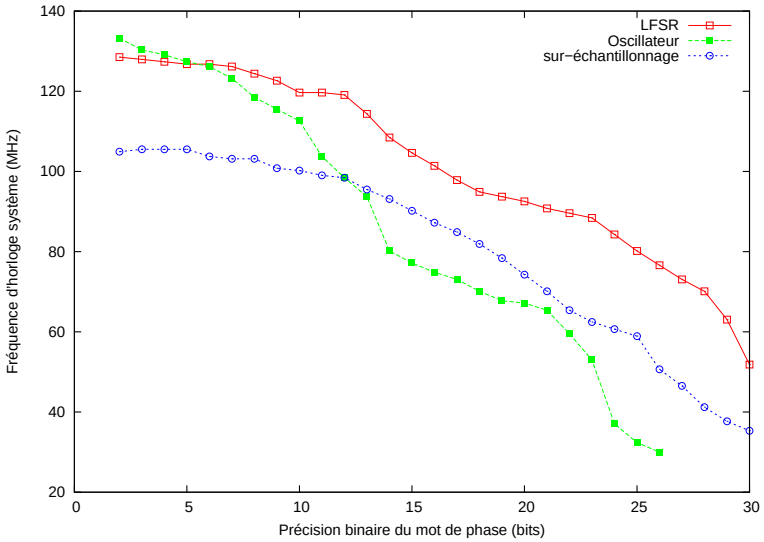


FIG. 3.16: Fréquence de fonctionnement maximale pour différents types d'architectures de démodulation de phase.

La figure 3.16 présente la fréquence maximale de fonctionnement du circuit, en fonction de la précision souhaitée pour la mesure de la phase. Notre architecture à base de LFSR permet d'obtenir une fréquence supérieure aux deux autres architectures testées au delà d'une précision de 8 bits. L'écart entre les architectures s'agrandit au fur et à mesure que la précision requise augmente. L'architecture à base d'oscillateur numérique présente de bonnes performance pour les précisions inférieures à 12 bits, mais sa fréquence de fonctionnement chute rapidement pour les précisions de plus de 16 bits.

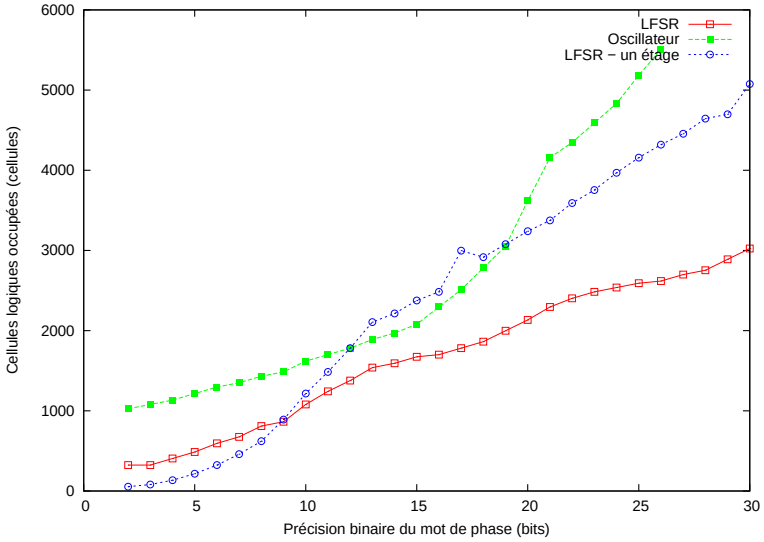


FIG. 3.17: Nombres de cellules logiques occupées.

La figure 3.17 compare le nombre de cellules logiques occupées par les trois architectures. L'architecture à base de LFSR voit sa consommation progresser linéairement avec la précision du mot de phase. La vitesse d'accroissement du nombre de cellules occupées est pratiquement identique à celle de l'architecture utilisant un oscillateur local, pour les précisions inférieure à 12 bits. Par contre, au delà de 16 bits, seule l'architecture à base de LFSR présente encore une consommation raisonnable, qui permet d'envisager une parallélisation éventuelle en multipliant les étages de calcul.

Enfin, la figure 3.18 présente l'efficacité des différentes architectures, en termes de fréquence en fonction de la surface. L'architecture à base de LFSR reste la plus efficace : elle permet d'assurer une fréquence d'horloge supérieure à 100 MHz, avec moins de 2000 cellules logiques consommées, pour une phase pouvant aller jusqu'à 16 bits de précision.

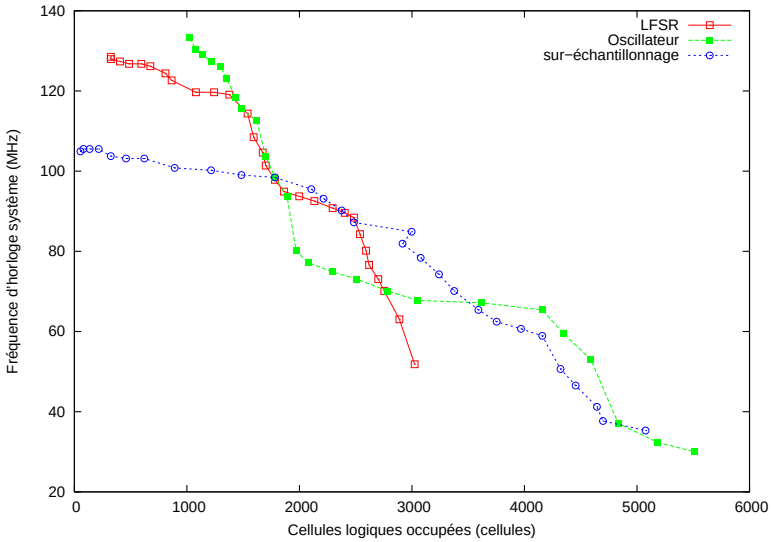


FIG. 3.18: Fréquence d'horloge maximale atteinte en fonction du nombre de cellules logiques occupées.

Conclusion

Dans ce chapitre, nous avons présenté deux nouvelles architectures de démodulation numérique de signaux modulés en fréquence ou en phase. Elles utilisent toutes deux des compteurs efficaces, basés sur des registres LFSR, dont la dimension et le polynôme caractéristique permettent de contrôler la durée du comptage.

Un démodulateur de fréquence utilisant des LFSR a été décrit, et implanté sur FPGA. Il présente la particularité d'effectuer une mesure directe de la fréquence, plutôt que d'utiliser une boucle de synchronisation. Ses performances permettent d'envisager son utilisation sous forme parallèle dans une architecture haut-débit.

Une méthode pour créer un décompteur apparié à un compteur donné a été proposée et mise en application pour réaliser une architecture pour la démodulation de phase, utilisant quatre LFSR compteurs/décompteurs. Cette architecture, tout comme son homologue pour la démodulation de fréquence, atteint des performances meilleures que les architectures existantes et occupe un nombre restreint de cellules logiques.

Les LFSR sont des systèmes particulièrement bien adaptés à la structure interne des cellules logiques élémentaires de processeurs FPGA. Leur utilisation dans ces deux architectures de démodulation a permis d'obtenir de bonnes performances sur ce type de circuit, ce qui démontre l'intérêt d'une méthodologie de conception prenant en compte très en amont la technologie cible.

Conclusion générale

LES systèmes numériques présentent de nombreux avantages dans le domaine de la transmission de l'information. Plus particulièrement, les circuits reconfigurables de type FPGA[6] sont séduisants, par leur faible coût, leurs possibilités d'évolution importantes et leur intérêt économique pour les productions en petite série. Ils permettent de plus de passer directement de l'étape de prototypage à l'étape de produit fini. Mais le domaine des transmissions est dominé par les architectures analogiques, en particulier en ce qui concerne les couches les plus basses des protocoles, qui sont chargées de produire le signal à transmettre sur le médium. Car ces couches, dont les plus crtiques sont celles chargées de la modulation ou de la démodulation, nécessitent des circuits capables de produire des signaux dont la fréquence peut atteindre plusieurs giga hertz. Les fréquences de fonctionnement des circuits numériques synchrones restant encore très inférieures à celles des systèmes analogiques dédiés. Pour combler en partie cette différence, il est nécessaire de concevoir de nouveaux algorithmes permettant le traitement en parallèle des signaux numériques. Une simple transposition en numérique des algorithmes utilisés dans le domaine analogique ne suffit pas à atteindre les performances requises par les protocoles haut-débit, car cette transposition introduit des chemins critiques difficiles à réduire, et des fonctionnalités dont la parallélisation est ardue.

[6] *Field-Programmable Gate Array*

La nouvelle méthodologie de conception que j'ai décrite dans ce mémoire s'efforce de s'affranchir des difficultés posées par le passage du domaine analogique au domaine numérique. Elle propose de nouvelles architectures traduisant directement les fonctionnalités nécessaires à la modulation et la démodulation de signaux en algorithmes adaptés aux circuits numériques, et aux FPGA en particulier.

Notre modulateur QAM[7], utilisant en son cœur un algorithme de type CORDIC[8] modifié et parallélisé, a été décrit en VHDL et implanté sur un circuit FPGA en utilisant cette nouvelle méthodologie de conception. Ses performances sont jusqu'à huit fois supérieures à celles des architectures analogiques directement transposées en numérique. Notre démodulateur de phase qui utilise plusieurs bancs de registres de type LFSR[9], atteint des performances du même ordre de grandeur.

Ces deux systèmes ont été conçus de façon à tirer au mieux parti de la structure interne spécifique des circuits FPGA de type FLEX10K200E sur lesquels ils sont implantés. En conséquence, le circuit peut fonctionner à une fréquence d'horloge qui s'approche de la fréquence maximale de 250 MHz donnée par le constructeur. De plus, les algorithmes sont fortement parallèles, et permettent donc d'atteindre une cadence de traitement des données binaires pouvant atteindre quatre fois la fréquence d'horloge effective.

Des travaux restent à effectuer sur ces architectures, en particulier pour étudier et prendre en compte différents points qui n'ont pas encore étés abordés dans ce manuscrit. Des paramètres comme la consommation n'ont pas été analysés mais restent

[7] *Quadrature Amplitude Modulation*
[8] *COordinate Rotation DIgital Computer*
[9] *Linear Feedback Shift Register*

primordiaux pour pouvoir intégrer ces algorithmes sur des systèmes embarqués. Enfin, nous nous sommes focalisés plus sur le type de modulation et sur les algorithmes numériques capables de résoudre les problèmes d'implantation d'un modulateur sur FPGA que sur les protocoles réseaux existants. Une prochaine étape de cette étude doit prendre en compte plus précisément les spécifications des protocoles courants, comme par exemple la distorsion en phase ou en amplitude autorisée pour le signal produit. Il est déjà possible d'agir sur ces paramètres, en changeant la profondeur des pipelines utilisés dans l'architecture CORDIC du modulateur, ou sur la longueur des registres utilisés dans le démodulateur.

Le projet peut-être avantageusement étendu pour permettre la gestion en ligne, sans nécessiter de reprogrammation, de protocoles utilisant plusieurs types de modulation pour s'adapter par exemple à la qualité de la ligne de transmission. Dans l'état actuel de notre architecture, il n'est pas possible, sans réécrire le code VHDL associé au modulateur ou au démodulateur, de traiter simultanément ces types de signaux différents, bien que le logiciel de génération automatique de code simplifie grandement cette étape. À plus forte raison, les différents blocs fonctionnels décrits ici peuvent être intégrés dans une bibliothèque de fonctions élémentaires pour le traitement bas-niveau du signal dans le domaine des transmissions numériques. Cette intégration doit permettre à terme d'inclure les architectures pour la modulation et la démodulation dans un processeur réseau complet.

En conclusion, nous avons démontré dans ce mémoire la faisabilité de l'implantation numérique d'algorithmes de traitement du signal utilisés pour les transmissions à haut débit, en nous focalisant plus particulièrement sur les étapes de modulation et de

démodulation. Nous avons développé une méthodologie de conception différente, qui permet de transcrire les fonctions composant ces étapes en algorithmes optimisés pour une implantation sur un circuit numérique de type FPGA, plutôt que de reproduire les algorithmes existants dans le monde analogique. Nous avons proposé des pistes d'amélioration pour permettre d'étendre cette méthodologie à des protocoles plus complexes, et d'intégrer les blocs fonctionnels conçus ici dans le cadre plus global d'un processeur réseau.

Références

[ALFK95] P. Alfke, "Efficient shift register, LFSR counters, and long pseudo-random sequence generators", Application Note, Xilinx, 1995.

[ANDR98] R. Andraka, "A survey of cordic algorithms for FPGA based computers", in *Proc. Sixth ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '98)*, ACM/SIGDA, ACM, 1998.

[ARNA04] F. Arnault, T. P. Berger, and A. Necer, "Feedback with carry shift registers synthesis with the Euclidean algorithm", *IEEE Trans. Inform. Theory*, vol. 50, no. 5, pp. 910–917, May 2004.

[ARNO05] G. Arnould, F. Monteiro, A. Dandache, and Ph. Jean, "Digital frequency synthesizer architectures", in *Proc. DCIS'2005*, Sept. 2005.

[ARNO06] G. Arnould, F. Monteiro, and A. Dandache, "A digital frequency shift keying demodulator", in *Proc. ICECS'06*, Nice, Dec. 2006.

[BADR02] J. Badrikian, *Mathématiques pour téléinformatique : codes correcteurs : principes et exemples*, Ellipses, Paris, 2002.

[BELL00] A. Bellaouar, Michael S. O'Brecht, A. M. Fahim, and M. I. Elmasry, "Low-power direct digital frequency synthesis for wireless communication", *IEEE J. Solid-State Circuits*, vol. 35, no. 3, pp. 385–390, March 2000.

[BELL02] M. Bellanger, *Traitement numérique du signal – théorie et pratique*, 7ème édition, Masson, Paris, 2002.

[BERL68] E. R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, 1968.

[BREN96] P. Brenner, "A technical tutorial on the IEEE 802.11 protocol", Breezecom Wireless Communications, July 1996.

[BUSB00] M. Busby, *Introduction à TCP/IP*, Osman Eyrolles Multimedia, Paris, 2000.

[CARD02] F. Cardells-Tormo, J. Valls-Coquillat, V. Almenar-Terre and V. Torres-Carot, "Efficient FPGA-based QPSK demodulation loops : application to the DVB standard", in *Proc. 12th Int. Conf. on Field-Programmable Logic and Applications (FPL 2002)*, Sept. 2002, pp. 112–121.

[CARD03] F. Cardells-Tormo and J. Valls-Coquillat, "Area-optimized implementation of quadrature direct digital frequency synthesizers on LUT-based FPGAs", *IEEE Trans. Circuits Syst. – II*, vol. 50, pp. 135–138, Mar. 2003.

[CHEU89] S. W. Cheung, "Performance of a CE16QAM modem in a regenerative satellite system", *Int. J. of Satellite Communications*, vol. 7, pp. 425–434, Dec. 1989.

[COME00] D. E. Comer, *TCP/IP architecture, protocoles, applications*, 3me édition, Dunod, Paris, 2000.

[CURT00] F. Curticapean and J. Niittylahti, "Low-power direct digital frequency synthesizer", in *Proc. 43rd IEEE Midwest Symp. Circuits and Systems*, vol. 2, Aug. 2000, pp. 822–825.

[CURT01] F. Curticapean and J. Niittylahti, "A hardware efficient direct digital frequency synthesizer", in *Proc. 8th IEEE Int. Conf. on Electronics, Circuits and Systems (ICECS 2001)*, vol. 1, Sept. 2001, pp. 51–54.

[DECA02] D. De Caro, E. Napoli, and A. G. M. Strollo, "ROM-less direct digital frequency synthesizers exploiting polynomial approximation", in *Proc. 9th Int. Conf. on Electronics, Circuits and Systems (ICECS 2002)*, Sept. 2002, pp. 481–484.

[DICK02] C. Dick, F. Harris, "FPGA QAM demodulator design", in *Proc. 12th Int. Conf. on Field-Programmable Logic and Applications (FPL 2002)*, Sept. 2002, pp. 112–121.

[DUPR93] J. Duprat and J. Muller, "The CORDIC algorithm : new results for fast VLSI implementation", *IEEE Trans. Comput.*, vol. C-42, pp. 168–178, Feb. 1993.

[FLEX03] "FLEX 10KE embedded programmable logic device", Data Sheet, ver. 2.5, Altera Corp., Jan. 2003 ; www.altera.com/literature/ds/dsf10ke.pdf.

[FRAI99] Ph. Fraisse, R. Protière, D. Marty-Dessus, *Télécommunications 1 : Transmission de l'information*, Ellipses, Paris, 1999.

[GAGN01] M. Gagnaire, *Boucles d'accès haut débits*, Eyrolles, Paris, 2001,

[GLAV96] A. Glavieux, M. Joindot, *Communications numériques*, Masson, Paris, 1996.

[GOLO82] S. W. Golomb, *Shift Register Sequences*, Aegean Park Press, Laguna Hills, CA, 1982.

[GORE02] M. Goresky and A. Klapper, "Fibonacci and Galois representation of feedback with carry shift registers", *IEEE Trans. Inform. Theory*, vol. 48, pp. 2816–2836, Nov. 2002.

[GRAS01] E. Grass *et al.*, "On the single-chip implementation of a HiperLan/2 and IEEE 802.11a capable modem", *IEEE Personal Communications Mag.*, vol. 8, pp. 48–57, Dec. 2001.

[GREE04] D. H. Green, "Linear complexity of modulo-m power residue sequences", *IEE Proc. – Computers and Digital Techniques*, vol. 151, no. 6, pp. 385–390, Nov. 2004.

[HAMP94] G. Hampson and Paplinski, "A VHDL implementation of a CORDIC arithmetic processor chip", Tech. Rep. 94-9, Monash Univ., Clayton, Victoria, Australia, 10 Oct. 1994.

[HANZ02] L. Hanzo, T. H. Liew, and B. L. Yeap, *Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission over Fading Channels*, IEEE Press and John Wiley & Sons, Ltd., 2002.

[HENT93] V. O. Hentinen, P. P. Laiho, and R. M. Särkilahti, "A digital demodulator for PSK signals", *IEEE Trans. Communications*, vol. COM-21, no. 12, pp. 1352–1360, Dec. 1993.

[HENH92] Y. Hen Hu, "CORDIC-based VLSI architectures for digital signal processing", *IEEE Signal Processing Magazine*, pp. 16–35, Juil. 1992.

[IEEE99] IEEE standard 802.11 : "Wireless LAN medium access control and physical layer specifications", IEEE Computer Society, August 1999.

[KANG03] C. Y. Kang and E. E, Swartzlander, Jr., "A digit-pipelined direct digital frequency synthesis architecture", in *Proc. IEEE Workshop on Signal Processing Systems (SIPS 2003)*, Aug. 2003, pp. 224–229.

[KANG06] C. Y. Kang and E. E, Swartzlander, Jr., "Digit-pipelined direct digital frequency synthesis based on differential CORDIC", *IEEE Trans. Circuits and Systems I : Regular Papers*, vol. 53, no. 5, pp. 1035–1044, May 2006.

[KOFM99] D. Kofman and M. Gagnaire, *Reseaux haut debit : Tome I Reseaux ATM et reseaux locaux*, 2me édition, Dunod, Paris, 1999.

[KOZA01] M. Kozak and I. Kale, "A pipelined noise shaping coder for fractional-N frequency synthesis", *IEEE Trans. on Instrumentation and Measurement*, vol. 50, no. 5, pp. 1154–1161, Oct. 2001.

[LANG02] J. M. P. Langlois and D. Al-Khalili, "Hardware optimized direct digital frequency synthesizer architecture with 60 dBc spectral purity", in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS 2002)*, vol. 5, Kingston, Ontario, Canada, May 2002, pp. V-361–V-364.

[LEE06] T. C. Lee and C. C. Chen, "A mixed-signal GFSK demodulator for Bluetooth", *IEEE Trans. Circuits and Systems–II : Express Briefs*, vol. 53, no. 3, pp. 197–201, March 2006.

[LIDL86] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge University Press, Cambridge, UK, 1986.

[LIN83] S. Lin and D.J. Costello, Jr., *Error Control Coding : Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

[LIU01] S.-I. Liu, T.-B. Yu, and H.-W. Tsao, "Pipeline direct digital frequency synthesiser using decomposition method", *IEE Proc. – Circuits, Devices and Systems*, vol. 148, pp. 141–144, March 2001.

[MASS69] J. L. Massey, "Shift-register synthesis and BCH decoding", *IEEE Trans. Inform. Theory*, vol. IT-15, no. 1, pp. 122–127, Jan. 1969.

[MATT93] W. E. Mattis, "A new modem structure for data transmission", *IEEE Trans. on Consumer Electronics*, vol. 39, no. 4, pp. 878–886, Nov. 1993.

[MENC00] O. Mencer, L. Séméria, M. Morf, and J.-M. Delosme, "Application of reconfigurable CORDIC architectures," *J. VLSI Signal Processing*, vol. 24, no. 2–3, pp. 211–221, Mars 2000.

[MESU05] "Les composants FPGA s'invitent dans le traitement du signal", *Mesures*, vol. 775, pp. 54–56, Mai 2005.

[NAHM97] S. Nahm and W. Sung, "A fast direction sequence generation method for CORDIC processors", in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'97)*, vol. I, Munich, 21–24 April 1997, pp. 635–638.

[NICH98] H. T. Nicholas, H. Samueli, and B. Kim, "The optimization of direct digital frequency synthesizer performance in the presence of finite word length effects", in *Proc. 42nd Annual Frequency Control Symposium*, 1998, pp. 357–363.

[OFDM00] OFDM-based 802.16.3 sub-11 GHz BWA Air Interface Physical Layer proposal, IEEE 802.16 Broadband Wireless Access Working Group Oct. 2000 ; http ://ieee802.org/16.

[PAGE92] J. H. Page *et al.*, "An application of DSP to voiceband modems", *BT Technology J.*, vol. 10, no. 1, pp. 80–100, Jan. 1992.

[PHAT98] D. S. Phatak, "Double step branching CORDIC : a new algorithm for fast sine and cosine generation", *IEEE Trans. Comput.*, vol. 47, no. 5, pp. 587–602, May 1998.

[PHIL98] S. Philip, "Etude et développement d'une nouvelle architecture de processeur DSP dédiée aux applications modem en télécommunications sur câble TV", Manuscrit de thèse, LICM, Université de Metz, 1998.

[PRIY03] B. E. Priyanto, C. L. Law, and Y. L. Guan, "Design and implementation of all digital I-Q modulator and demodulator for high speed wlan in FPGA", in *Proc. 2003 IEEE Pacific Rim Conf. on Communications, Computers and Signal Processing (PACRIM 2003)*, vol. 2, August 2003, pp. 659–662.

[PROA95] J. G. Proakis, *Digital Communications*, McGraw Hill, 1995.

[PUJO03] G. Pujolles, *Les réseaux*, 3ème édition, Eyrolles, Paris, 2003.

[QUAL91] "Qualcomm Q2334", Technical Data Sheet, Qualcomm Ltd., June 1991.

[RIST65] M. P. Ristenblatt, "Pseudo-random binary coded waveforms", *Modern Radar*, Ed. R. S. Berkowitz, New York : Wiley, pp. 274–314, 1965.

[SAMU90] H. Samueli and B. C. Wong, "A VLSI architecture for a high-speed all-digital quadrature modulator and demodulator for digital radio applications", *IEEE J. Selected Areas in Communications*, vol. 8, no. 8, pp. 1512–1519, Oct. 1990.

[SARR04] K. Sarrigeorgidis, and J. Rabaey, "Ultra low power CORDIC processor for wireless communication algorithms," *J. VLSI Signal Processing*, vol. 38, no. 2, pp. 115–130, Sept. 2004.

[SAVI92] J. Savir and W. H. McAnney, "A multiple seed linear feedback shift register", *IEEE Trans. Computers*, vol. 41, no. 2, pp. 250–252, Feb. 1992.

[SCHO60] P. H. R. Scholefield, "Shift registers generating maximum length sequences", *Electronic Technology*, vol. 37, pp. 389–394, Oct. 1960.

[SHAN48] C. E. Shannon, "A mathematical theory of communication", *Bell System Technical J.*, vol. 27, pp. 379–423, 623–656, July, October 1948.

[SING03] A. Singh *et al.*, "Comparison of branching CORDIC implementations", in *Proc. IEEE Int. Conf. on Application-Specific Systems, Architectures, and Processors*, June 2003, pp. 215–225.

[SODA01] A. M. Sodagar and G. R. Lahiji, "A pipelined ROM-less architecture for sine-output direct digital frequency synthesizers using the second-order parabolic approximation", *IEEE Trans. Circuits and Systems, Part II : Analog and Digital Signal Processing,*, vol. 48, no. 9, pp. 850–857, Sept. 2001.

[SONG04] Y. Song and B. Kim, "Quadrature direct digital frequency synthesizers using interpolation-based angle rotation", *IEEE Trans. VLSI Syst.*, vol. 12, pp. 701–710, July 2004.

[STAN97] M. R. Stan, "Synchronous up/down counter with clock period independent of counter size", in *Proc. 13th IEEE Symp. Computer Arithmetic (ARITH-13)*, 1997, p. 274–281. vol. 1, April 1997, pp. 635–638.

[STRA05] "Stratix II device family data sheet", Data Sheet, Altera Corp., Jan. 2005 ; www.altera.com/literature/hb/stx2/stx2_sii5v1_01.pdf.

[STEI91] M. Stein, *Les modems pour la transmission de données numériques*, 2ème édition, Masson, Paris, 1991.

[TANE03] A. Tanenbaum, *Réseaux*, 4ème édition, Eyrolles, Paris, 2003.

[THOM05] D. B. Thomas and W. Luk, "High quality uniform random number generation for massively parallel simulations in FPGAs", in *Proc. 2005 Int. Conf. Reconfigurable Computing and FPGAs (ReConfig'05)*, Puebla City, Mexico, 28-30 Sept 2005, pp. 84–91.

[TIER71] J. Tierney, C. M. Rader, and B. Gold, "A digital frequency synthesizer", *IEEE Trans. Audio Electroacoust.*, vol. 19, pp. 48–57, Mar. 1971.

[TORO01] A. Torosyan and A. N. Willson Jr., "Analysis of the output spectrum for direct digital frequency synthesizers in the presence of phase truncation and finite arithmetic precision", in *Proc. 2nd Int. Symp. on Image and Signal Processing and Analysis (ISPA'2001)*, Pula, Croatia, 19–21 June 2001, pp. 458–463.

[UKEI93] R. L. Ukeiley, *Field Programmable Gate Arrays (FPGAs)*, PTR Prentice Hall, Englewood Cliffs, 1993.

[VALL04] J. Valls, M. Kuhlmann, and K. K. Parhi, "Evaluation of CORDIC Algorithms for FPGA Design," *J. VLSI Signal Processing*, vol. 32, no. 3, pp. 207–222, Sept. 2004.

[VANK97] J. Vankka, "Methods of mapping from phase to sine amplitude in direct digital synthesis", *IEEE Trans. Ultrason. Ferroelec. Freq. Contr.*, vol. 44, no. 2, pp. 526–534, March 1997.

[VILL97] J. Villasenor and W. H. Mangione-smith, *Configurable Computing*, Scientific American, Juin 1997.

[VOLD59] J. E. Volder, "The CORDIC trigonometric computing technique", *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 330–334, Sept. 1959.

[VOLD00] J. E. Volder, "The birth of CORDIC," *J. VLSI Signal Processing*, vol. 25, no. 2, pp. 101–105, June 2000.

[WALT71] J. S. Walther, "A unified algorithm for elementary functions", in *Proc. AFIPS Spring Joint Comput. Conf.*, vol. 38, Atlantic City, NJ, USA, May 1971, pp. 379–385.

[WANG97] S. Wang, V. Piuri, and E. E. Swartzlander, "Hybrid CORDIC algorithms", *IEEE Trans. Comput.*, vol. 46, no. 11, pp. 1202–1207, Nov. 1997.

[WATS62] E. J. Watson, "Primitive polynomials (mod 2)", *Math. Comp.*, vol. 16, pp. 368—369, 1962,

[WEAV90] L. A. Weaver and R. J. Kerr, "High resolution phase to sine amplitude conversion", US. Patent 4 905 177, 27 Feb. 1990.

[XU98] J. Xu and A. Klapper, "Feedback with carry shift registers over $z/(n)$", in *Proc. Int. Conf. on Sequences and Their Applications (SETA'98)*, Singapore, Springer-Verlag, New York, Dec. 1998.

[XUE05] R. Xue, Q. Xu, K. F. Chang, and K. W. Tam, "A new method of an IF I/Q demodulator for narrowband signals", in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS 2005)*, May 2005.

Liste des acronymes

ALU	–	Arithmetic-Logic Unit
ASIC	–	Application Specific Integrated Circuit
ASK	–	Amplitude Shift Keying
AAL	–	Asynchronous Transfer Mode Adaptation Layer
ATM	–	Asynchronous Transfer Mode
BCH	–	Bose, Ray-Chaudhuri, Hocquenghem
CORDIC	–	COrdinate Rotation DIgital Computer
CPU	–	Cenral Processing Unit
dBc	–	Decibel Relative to Carrier
DSP	–	Digital Signal Processing
EAB	–	Embedded Array Block
ETCD	–	Équipement Terminal de Circuit de Données
ETTD	–	Équipement Terminal Transmetteur de Données
FPGA	–	Field-Programmable Gate Array
FTP	–	File Transfer Protocol
HTTP	–	HyperText Transfer Protocol
FSK	–	Frequency Shift Keying
GF	–	Galois Field
IOE	–	Input/Output Element
LAB	–	Logical Array Bloc
LE	–	Logical Element
LFSR	–	Linear Feedback Shift Register

LAB	–	Logical Array Bloc
LUT	–	Look-Up Table
MAC	–	Medium Access Control
M-FSK	–	M-ary Frequency Shift Keying
M-PSK	–	M-ary Phase Shift Keying
OSI	–	Open System Interconnection
PLL	–	Phase-Locked Loop
PSK	–	Phase Shift Keying
QAM	–	Quadrature Amplitude Modulation
RAM	–	Random Access Memory
ROM	–	Read Only Memory
RTL	–	Register Transfer Level
TCP	–	Transmission Control Protocol
IP	–	Internet Protocol
UDP	–	User Datagram Protocol
UIT-T	–	Union Internationale des Télécommunications
VCO	–	Voltage Controled Oscillator
VHDL	–	Very High Speed Integrated Circuit Hardware Description Language
VLSI	–	Very Large Scale Integration

Résumé

Ce mémoire porte sur l'adaptation au domaine tout numérique des techniques de modulation et de démodulation utilisées pour les transmissions de données dans les protocoles de communications modernes. Le passage au «tout numérique» provoque de nombreux changements, à la fois dans la méthodologie de conception, et dans les méthodes d'adaptation des algorithmes à l'architecture. En effet, la conversion des techniques analogiques en numérique s'avère générer de nombreuses contraintes qui brident les performances du système résultant. Aussi, ce travail définit de nouveaux algorithmes, qui s'affranchissent des structures complexes présentant des chemins critiques difficiles à réduire, telles les boucles de rétroaction très souvent utilisées dans les architectures analogiques et qui ne sont pas toujours adaptées aux composants numériques. L'architecture détaillée d'un modulateur de phase et d'amplitude, et celle d'un démodulateur associé sont présentées et implantées sur un circuit FPGA. Leurs performances peuvent être supérieures d'un facteur 8 à celles obtenues par simple transposition des architectures analogiques existantes sur un circuit numérique. Elles présentent en outre, de bonnes capacités d'évolution. En effet, la structure parallèle de ces architectures permet d'atteindre un débit de données élevé, et leur conception modulaire permet de les réutiliser par blocs dans le cadre d'un processeur réseau complet.

Mots-clés : Architectures numériques haut débit, modulation et démodulation totalement numériques, CORDIC, modélisation RTL, applications télécom.

Abstract

The aim of this thesis is to adapt the modulation and demodulation techniques used in modern data transmission protocols so that they could be used in a purely digital environment. The main issues are important modifications that must be incorporated to the design methodology and the methods of implementing algorithms in hardware. This is because the analogue to digital conversion usually imposes several constraints that worsen performance of the resulting system. In this work are proposed new algorithms that allow to avoid using feedback loops that usually appear in purely analogue architectures, and which are not suitable for digital implementations. The detailed architectures for a phase and amplitude modulator and for the corresponding demodulator are presented and implemented on FPGA. These new digital circuits significantly outperform their existing digital counterparts which could be obtained by simple analogue to digital conversion. They also show good scalability. In particular, the parallel structure of these architectures enables to achieve high-throughput, whereas their modular design allows to reuse some blocks directly in a complete network processor.

Keywords: High-throughput digital architectures, all-digital modulation and demodulation, CORDIC, RTL modelisation, telecommunication applications.