ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

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

PAR
FUTCHA-DJAHEU, Sandrine

EXTRACTION MANUELLE DE PRIMITIVES ISSUES D'APPLICATIONS EN TÉLÉCOMMUNICATION POUR UNE PLATEFORME MONOPROCESSEUR

MONTRÉAL, LE 9 AVRIL 2008

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. François Gagnon, directeur de mémoire
Département de génie électrique à l'École de technologie supérieure

M. Claude Thibeault, codirecteur
Département de génie électrique à l'École de technologie supérieure

M. Jean-François Boland, président du jury
Département de génie électrique à l'École de technologie supérieure

M. Yvon Savaria, examinateur externe
Département de génie électrique à l'École Polytechnique

EXTRACTION MANUELLE DE PRIMITIVES ISSUES D'APPLICATIONS EN TÉLÉCOMMUNICATION POUR UNE PLATEFORME MONOPROCESSEUR

FUTCHA-DJAHEU, Sandrine

RÉSUMÉ

L'implémentation d'algorithmes complexes sur des plateformes matérielles est un processus long. De ce fait, il devient pertinent de vérifier au tout début de la conception, si une plateforme donnée possède la capacité de traiter de nouveaux algorithmes. Ceci dans le but de réduire les efforts de programmation liés à leur implémentation. Le temps d'exécution d'une application ou son nombre de cycles, le taux d'utilisation de la mémoire et la consommation de puissance sont de bons exemples de métriques dont la prédiction vaut le peine. Le défi auquel cette dernière fait face s'accroît, d'autant plus que l'application choisie exhibe une variabilité en termes de temps d'exécution.

Le but de ce projet de recherche est d'élaborer une méthodologie de prédiction du nombre de cycles d'un algorithme de télécommunication quelconque sur des puces de traitement de signal (DSP). L'approche globale qui est nôtre prend appui sur l'élaboration d'une librairie. Cette approche tire profit de la richesse des librairies existentes comme celle offerte par Matlab/Simulink® pour évaluer cette métrique. Comme Matlab® est souvent utilisé pour modéliser les systèmes de télécommunications et les valider, il a été choisi pour ce projet. L'idée consiste à pré-caractériser les cellules fréquemment employées dans ces librairies pour estimer le temps de calcul de l'unité centrale de traitement (CPU) à partir des modèles Matlab®. De là, des équations générales de variabilité sont établies. Celles-ci décrivent le comportement de la métrique d'intérêt dans le processeur à travers l'étude de trois applications spécifiques.

Les applications visées sont le Zero Forcing Sorted QR Decomposition (ZF-SQRD), le récepteur Rake et la transformée rapide de Fourier (FFT). La première étape vers l'élaboration des équations de prédiction consiste à décomposer ces applications en blocs décrivant les opérations récurrentes opérées en télécommunication, nommées primitives fonctionnelles. Ensuite s'établit un lien entre ces opérations et celles équivalentes du côté matériel (primitives structurelles) pour compléter la librairie. Finalement, la validation des équations du temps CPU déduites en se servant de la librairie couronne le travail par l'implémentation dans un processeur. La précision des résultats repose sur la configuration des options offertes par le compilateur, en particulier sur la sélection ou non du pipeline logiciel. Lorsque ce dernier n'est pas sélectionné, les prédictions et les implémentations sont similaires, bonifiant ainsi la librairie établie. La sélection du pipeline logiciel introduit un rapport autour d'un pour deux entre les prédictions et les implémentations.

MANUAL EXTRACTION OF PRIMITIVES FROM TELECOMMUNICATION APPLICATIONS FOR A MONOPROCESSOR PLATFORM

FUTCHA-DJAHEU, Sandrine

ABSTRACT

Implementing complex algorithms on hardware platforms is time consuming. Therefore, it becomes very relevant to verify at the early stage development if a given digital signal platform is able to adequately run new algorithms in order to reduce the software efforts needed to implement them. The application execution time or number of cycles, the memory usage, the power consumption are good metrics' illustrators on which the prediction may ly to be effective. The challenge which face the forecasts, rise particularly when the target application exhibits variability in terms of CPU requirements.

The aim of our research project is to elaborate a methodology to predict the number of clock cycles for any telecommunication algorithm on digital signal processors (DSPs) devices. Our global approach is library based one. It takes advantage of the richness of existing libraries as the one offered by Matlab/Simulink® to elaborate such metric. As Matlab® is often utilized to model telecommunications systems and validate them, it was selected for the project. The idea is to pre-characterize the commonly used cells of these libraries to estimate the CPU time from Matlab® systems models. From this, generalized variability equations to depict the behavior of the former metric in a processor are established using three specific applications.

The target applications are Zero Forcing Sorted QR Decomposition (ZF-SQRD), Rake Receiver and Fast Fourier Transform (FFT) algorithms. The first step toward the elaboration of the predictions' equations, consists in the decomposition of those applications into blocks describing recurrent operations performed in telecommunication, namely functionnal primitives. Then a link between them and the equivalent hardware operations (structural primitives) is performed to complete the library. Finally comes the validation of the equations of the CPU time sorted using the library through atual implementation. The results accuracy rely on the compiler's configuration settings, precisely on the selection or not of the option involving software pipelining. When it is not selected, the implementations and the forecasts are quite similar, making the library reliable. When selected, the software pipelining introduces a ratio between predictions and implementations around one of two.

REMERCIEMENTS

J'aimerai exprimer ma profonde gratitude et mes sincères remerciements aux Professeurs, François Gagnon et Claude Thibeault, respectivement directeur et codirecteur de ce mémoire pour leur expertise, leur disponibilité et les conseils prodigués tout au long de la réalisation de ce travail.

Je ne saurai oublier l'appui inconditionnel de ma famille. Je remercie tout particulièrement Innocent, Pauline, Serge et Cyrille qui ont été des piliers face aux durs moments d'épreuves que j'ai dû affronter. Je remercie aussi mes bonnes amies : Isabelle et Élise. Vous m'avez donné le courage nécessaire pour foncer et ne pas baisser les bras ! ! ! Je vous dédie ce travail qui serait sans doute resté inachevé sans vous.

Finalement, je remercie tous les membres passés et présents du Laboratoire de Communications et d'Intégration de la Microélectronique (LACIME) auprès de qui j'ai trouvé un bel environnement humain pour la recherche et bien plus qu'un lieu de travail. Je pense spécialement à Basile, Safa et Marie-Ève. Merci pour tous vos bons conseils.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET SIGLES

ALU	**A**rithmetic and **L**ogic **U**nit (Unité arithmétique et logique)
ARIB	Japan's **A**ssociation of **R**adio **I**ndustry and **B**usiness (Association japonaise de radio, d'industrie et de commerce)
ATIS	**A**lliance for **T**elecommunications **I**ndustry and **S**olutions (Alliance pour les solutions en télécommunication industrielle)
BER	**B**it **E**rror **R**ate (Taux d'erreur binaire)
BLAST	**B**ell **L**aboratories **LA**yered **S**pace - **T**ime (Codage spatio-temporel par couches initié par les laboratoires de Bell)
C	Capacité du canal en $bits/s$
CCS	**C**ode **C**omposer **S**tudio
CDMA	**C**ode of **D**ivision and **M**ultiple **A**ccess (Accès Multiple par Répartition de codes)
COFF	**C**ommon **O**bject **F**ile **F**ormat (Fichier objet de format commun)
CPU	**C**entral **P**rocess **U**nit(Unité centrale de traitement)
DBLAST	**D**iagonal **BLAST** (Codage spatio-temporel diagonal par couches)
DIF	**D**ecimation **I**n **F**requency (Décimation fréquentielle)
DIT	**D**ecimation **I**n **T**ime (Décimation temporelle)
DP	**D**ouble **P**recision (Double précision)
DS-CDMA	**D**irect **S**equence CDMA (CDMA à séquence directe)
DSP	**D**igital **S**ignal **P**rocessor (Processeur numérique du signal)
DFT	**D**iscrete **F**ourier **T**ransform (Transformée de Fourier discrète)
DRAM	**D**ynamic **R**andom **A**ccess **M**emory (Mémoire dynamique à accès aléatoires)

EMIF	**E**xternal **M**emory **I**nter **F**ace (Interface pour la mémoire)
EPROM	**E**rasable **P**rogrammable **R**ead **O**nly **M**emory (Mémoire effaçable et programmable à lecture seule)
ETSI	**E**uropean **T**elecommunications **S**tandards **I**nstitute (Institut des standards en télécommunication européen)
FDD	**F**requency of **D**ivision **D**uplex (Duplexage par Répartition en Fréquences)
FFT	**F**ast **F**ourier **T**ransform (Transformée rapide de Fourier)
FDMA	**F**requency of **D**ivision **M**ultiple **A**ccess (Accès Multiple par Répartition en Fréquences)
GSM	**G**lobal **S**ystem for **M**obile (Système global pour mobile)
GPS	**G**lobal **P**ositionning **S**ystem (Système de positionnement global)
H	atténuation du signal dans un canal à évanouissement ou gain du canal
HPI	**H**ost **P**ort **I**nterface (Port Parallèle)
IDFT	**I**nverse **D**iscrete **F**ourier **T**ransform (Transformée de Fourier discrète inverse)
IFFT	**I**nverse of **F**ast **F**ourier **T**ransform (Transformée rapide de Fourier inverse)
IMI	**I**ntervalle **M**inimal d' **I**tération
IMT2000	**I**nternational **M**obile **T**elecommunications for the year 2000 (Télécommunications Mobiles Internationales pour l'année 2000)
IRAM	**I**nternal **R**ead **A**ccess **M**emory (Mémoire interne à accès aléatoires)
ISI	**I**nter **S**mbole **I**nterference (Interférence inter-symbole)
ITU	**I**nternational **T**elecommunication **U**nion (Union Internationale de Télécommunication)
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers (Institut des ingénieurs électriques et électronique)

LAN	**L**ocal **A**rea **N**etwork (Réseau local)
LOS	**L**ign **O**f **S**ight (Ligne de vue directe)
LUT	**L**ook **U**p **T**able (Table de transcodage)
MAC	**M**edium **A**ccess **C**ontrol (Medium de contrôle d'accès)
McBSP	**M**ultichannel **B**uffered **S**erial **P**ort (Bus série)
MIMO	**M**ultiple **I**nput **M**ultiple **O**utput (Entrées et sorties multiples)
MIPS	**M**illion **I**nstruction **P**er **S**econd (Million d'instructions par seconde)
MISO	**M**ultiple**I**nput **S**ingle **O**utput (Entrées multiples et sortie unique)
MMSE	**M**inimum **M**ean **S**quared **E**rror (Erreur quadratique moyenne)
N	Le bruit blanc complexe additif
N_0	Densité spectrale de puissance monolatérale du bruit blanc
N_r	Nombre d'antennes au récepteur
N_t	Nombre d'antennes au transmetteur
NLOS	**N**on **L**ign **O**f **S**ight (Ligne de vue indirecte)
OFDM	**O**rthogonal **F**requency **D**ivision **M**ultiplexing (Multiplexage par division en fréquences)
OFDMA	**O**rthogonal **F**requency **D**ivision **M**ultiplexing **A**ccess (Accès Multiple par Répartition en Fréquences Orthogonales)
OSI	**O**pen **S**ystems **I**nterconnection (Interconnexion de systèmes ouverts)
P	Puissance moyenne du signal sur la durée d'observation
PLL	**P**hase **L**ocked **L**oop (Boucle à verrouillage de phase)
PN	**P**seudo **N**oise (Séquence Pseudo Aléatoire)
QPSK	**Q**uadrature **P**hase **S**hift-**K**eying (Modulation par quadrature de phase)
r	Le vecteur colonne d'observations

RAM	**R**ead **A**ccess **M**emory (Mémoire à accès aléatoires)
ρ	Le rapport signal à bruit moyen par antenne de réception.
ROM	**R**ead **O**nly **M**emory (Mémoire à lecture seule)
RISC	**R**educed **I**nstruction **S**et **C**omputer (Ordinateur à instructions réduites)
RRC	**R**adio **R**esource **C**ontrol (Contrôle radio de ressources)
RTW	**R**eal **T**ime **W**orkshop (Outil de traitement temps-réel)
s	Le vecteur colonne à l'émetteur
SDMA	**S**patial of **D**ivision and **M**ultiple **A**ccess (Accès Multiple par Répartition dans l'espace)
SDRAM	**S**ynchronous **D**ynamic **R**andom **A**ccess **M**emory (Mémoire dynamique synchrone à accès aléatoires)
σ^2	La variance du bruit
SIMO	**S**ingle **I**nput **M**ultiple **O**utput (Entrée unique et sortie multiples)
SISO	**S**ingle **I**nput **S**ingle **O**utput (Entrée et sortie uniques)
SMG	**S**pecial **M**obile **G**roup (Groupe mobile spécial)
SNR	**S**ignal to **N**oise **R**atio (Rapport Signal à Bruit)
SP	**S**ingle **P**recision (Simple précision)
SRAM	**S**tatic **R**andom **A**ccess **M**emory (Mémoire statique à accès aléatoires)
STC	**S**pace - **T**ime **C**ode (Code spatio-temporel)
TDD	**T**ime of **D**ivision **D**uplex (Duplexage par Répartition en Temps)
TDMA	**T**ime of **D**ivision and **M**ultiple **A**ccess (Accès Multiple par Répartition dans le Temps)
TTC	**T**elecommunications **T**echnology **C**ommittee (Comité en technologie des télécommunications)

UMTS	**U**niversal **M**obile and **T**elecommunication **S**ystems (Systèmes de télécommunication mobile universel)
UTRA	**U**MTS **T**errestrial and **RA**dio interface (Interface radio terrestre de l'UMTS)
VBLAST	**V**ertical **BLAST** (Codage spatio-temporel vertical par couches)
VLIW	**V**ery **L**ong **I**nstruction **W**ord (Mot à instruction très longue)
W	Bande passante limitée du signal
WCDMA	**W**ideband CDMA (CDMA à large bande)
ZF	**Z**ero **F**orcing (Forçage à zéro)

INTRODUCTION

Au niveau historique, il aura fallu 15 ans supplémentaires à partir 1978, pour que les processeurs numériques de signal (en anglais DSP) deviennent des composants incontournables à l'électronique du grand public. Les DSP ont été initialement développés pour des applications de radars militaires et de télécommunications cryptées dans les années 70. C'est Texas Instruments qui, pour la première fois, en 1978, introduit un DSP pour la synthèse de la voix pour des applications courantes. Depuis lors, le traitement numérique du signal est un domaine en plein essor.

L'implémentation des algorithmes de traitement numérique du signal en temps réel est une tâche qui fait intervenir à la fois des aspects matériels et logiciels. Ceux-ci font partie intégrante d'un ensemble appelé plateforme microélectronique formée de processeurs numériques. L'implémentation d'un algorithme dans une plateforme microélectronique, en fonction de sa complexité, est un processus long, complexe et coûteux (Papadimitriou P. D. et al., 1997). À cet effet, les étapes à franchir vont du développement de l'algorithme lui-même à la production et la mise en marché de celui-ci en passant par les tests de simulation, la conception et le prototypage du circuit. Afin de faire face à la concurrence de plus en plus féroce dans ce domaine, les industries cherchent dans un premier temps, à simplifier et à réduire le temps d'achèvement de l'ensemble de ces opérations. Une fois cette étape franchie, leurs regards se tournent vers le perfectionnement des méthodologies courantes de design. Ces dernières gravitent autour du développement et de la maintenance de la plateforme pour laquelle elles sont destinées en vue de l'amélioration de ses capacités de réutilisation et de reconfiguration.

Le principal objectif du projet est de diminuer considérablement le temps d'estimation des ressources et, éventuellement, d'optimisation des performances d'une plateforme multiprocesseur programmable développée par l'entreprise Octasic . Il s'agit dans ce travail d'adapter les technologies d'Octasic à des applications radio autres que le traitement de

la voix, telles les télécommunications. C'est un processus itératif qui consiste, sans avoir à implémenter l'algorithme étudié, à prédire différentes métriques nécessaires à l'estimation des ressources requises par la plateforme et, de ces prédictions, à identifier le goulot d'étranglement s'il y en a un, de l'application puis à apporter des modifications afin de satisfaire aux contraintes de la plateforme si la puce programmable ne répond pas aux exigences du nouvel algorithme. Cette estimation pourvoiera ainsi à l'élaboration de designs dérivés et à l'amélioration de la flexibilité de la plateforme pour des développements futurs. En outre, le processus d'émulation s'en trouvera accéléré.

La méthodologie proposée consiste à développer une infrastructure dynamique et logicielle pour analyser et compléter les blocs d'opérations préconçus dans les simulateurs existants, dont Matlab/Simulink® est un exemple. Ces derniers ne répondent pas tous au besoin d'intégration des nouveaux algorithmes. La méthode employée permettrait ainsi de mieux prédire la capacité de la puce à recevoir le nouvel algorithme et d'en déduire les ressources nécessaires pour des designs dérivés. Pour ce faire, les étapes suivies du côté algorithmique prévoient de développer manuellement une librairie de blocs d'opérations primaires (dites primitives) récurrentes en télécommunication, d'établir des équations de variabilité des algorithmes étudiés en termes de temps CPU et de ressources mémoires, et, de produire une solution logicielle qui vise à automatiser l'extraction de ces primitives. Du côté matériel, il est prévu d'établir des équations de variabilité de la plateforme, d'appliquer des techniques d'ordonnancement et de partitionnement des tâches en contexte multi-processeurs et de générer dynamiquement un graphe d'exécution de l'application dans chacun des processeurs. Tout cela sera couronné par l'élaboration d'une méthodologie de vérification (formelle ou dynamique) des précédentes étapes.

Ce mémoire s'inscrit dans le cadre du traitement de signal dans les DSP. Il s'attèle à présenter les résultats du développement d'une librairie de primitives fonctionnelles (opérations récurrentes du côté application) et structurelles (opérations récurrentes du côté processeur). Cette étape servira de tremplin pour la réalisation des autres étapes, puisque

le but ultime est l'obtention d'une estimation-prédiction la plus fidèle possible, des risques technologiques qu'encourait l'implémentation de l'application choisie, sans avoir chaque fois à implémenter l'algorithme dans la plateforme d'intérêt. Le document se décompose en trois parties principales. La première partie est consacrée à une présentation sommaire des applications étudiées. La seconde traite de l'architecture du processeur et permet de faire le lien entre les opérations logicielles et celles matérielles lors de l'implémentation. La troisième présente les résultats obtenus.

Ce mémoire de maîtrise est constitué de cinq chapitres. Ils sont organisés comme suit :

Le chapitre premier s'attarde sur deux catégories importantes de systèmes regroupant les algorithmes étudiés, soit, d'une part, les systèmes de communication multi-émetteurs, multi-récepteurs (MIMO) et d'autre part, les liaisons point à multipoints en l'occurence l'accès multiple par répartition de code (CDMA) et l'accès multiple par répartition en fréquences orthogonales (OFDMA). Les premiers permettent d'augmenter l'efficacité spectrale d'un transmetteur en augmentant la diversité spatiale et les seconds permettent de partager un canal commun entre plusieurs utilisateurs, afin de rendre optimal l'utilisation des ressources d'un système.

Le deuxième chapitre présente de manière spécifique, les trois algorithmes qui ont fait l'objet de ce mémoire. Le multiplexage spatial avec emphase sur le codage par couches vertical vient en premier. Il est une catégorie d'algorithme MIMO permettant d'atteindre des niveaux élevés de capacités de transmission dans un système de communication. Ensuite, l'algorithme du récepteur Rake est présenté, dans un contexte d'accès multiple par codes à séquence directe selon le standard Universal Mobile Telecommunications Systems (UMTS). Enfin, ce chapitre fait cas de l'algorithme radix-2 de la transformée rapide de Fourier (FFT) avec décimation temporelle.

L'architecture du processeur utilisé, le TMS320C6711 de Texas Instruments, fait l'objet du chapitre 3. C'est un atout incontournable pour faciliter l'établissement d'une corrélation

entre les opérations du côté application et celles du côté matériel. De ce fait, la maîtrise de l'architecture de la puce utilisée est d'un grand apport pour exposer en détail la constitution de sa mémoire, ses modes d'adressage, le format représentation des nombres dont elle se sert (point flottant), son jeu d'instructions et ses techniques d'optimisation.

L'avant dernier chapitre étale la méthodologie d'extraction de primitives qui constitue le cœur de ce travail de recherche. Cette méthodologie s'organise en trois étapes. La première est la compréhension de l'architecture du processeur-cible pour en déduire un plan d'adressage qui minimise les accès mémoires inutiles. Ensuite, vient la compréhension des algorithmes étudiés pour en extraire une librairie d'opérations récurrentes en télécommunications, qui bien que non exhaustive, se veut être l'instrument de prédilection pour lier les opérations du côté application à celles du côté matériel. Finalement, la procédure de validation partiellement automatisée vient confirmer la pertinence des équivalences au sein de la librairie.

Le dernier et cinquième chapitre fait mention des résultats du travail accompli. Il s'attarde à les illustrer sur la base des applications ciblées et d'en faire une analyse. Ces résultats de simulation sont fortement dépendants du choix adéquat des options de simulation. La difficulté est donc de trouver la bonne combinaison d'options qui fournira des résultats plausibles.

CHAPITRE 1

LES DIFFÉRENTS SYSTÈMES DE COMMUNICATION ÉTUDIÉS

Les trois catégories d'application qui ont fait l'objet de ce travail de recherche appartiennent à la grande famille des télécommunications et sont classifiées comme suit : les algorithmes génériques, les systèmes de communication point à multipoints et les systèmes de communication à antennes multiples. Nous nous attarderons sur les deux derniers systèmes.

1.1 Les systèmes de communication MIMO

Les systèmes de communication à antennes multiples sans fils ont constitué un domaine de recherche d'intérêt ces dernières années au sein des réseaux de télécommunication. Les techniques MIMO (Multiple Input Multiple Output) permettent en effet de pourvoir à des taux de transfert de données élevés (Winters et al., 1994 ; Chuah et al., 1998, Foschini et Gans, 1998) et ainsi améliorer l'accès aux ressources en canaux de transmission (téléphonie mobile). Ils permettent aussi de répondre à la recrudescence des demandes d'accès internet et de transmission d'images via les systèmes de communication sans fils. Ces techniques sont alors utilisées dans plusieurs standards de communication tels : le MC-CDMA (Multi-Carrier Code Division Multiple Access), les réseaux locaux Wi-fi ou WLAN (IEEE 802.11) en Amérique et HiperLAN et UMTS en Europe ...

1.1.1 Le principe d'un système de communication MIMO

La raison d'être des systèmes MIMO s'articule autour de la résolution des problèmes d'encombrement et de limitation des réseaux sans fils en termes de débits et de ressources spectrales. L'idée de base dans les systèmes MIMO est le traitement spatio-temporel de l'information, où le temps, dimension naturelle, est complété par une dimension spatiale inhérente à l'utilisation de plusieurs antennes. Dans le cas des systèmes MIMO-OFDM

(MIMO-Orthogonal Frequency Division Multiplexing), un troisième concept se rajoute, celui de la fréquence, qui permet la conversion d'un canal MIMO sélectif en fréquence en un ensemble de canaux parallèles fixes en fréquence dits « flat fading ». Dans ce type de systèmes, l'information est répartie sur un grand nombre de porteuses toutes othogonales les unes aux autres et modulées à faibles débits. Cette technique est prisée par les nouvelles normes telles que le WiMax, qui est l'une des applications visées dans ce projet de recherche. Selon la Figure 1, le principe de fonctionnement d'un système MIMO consiste à transmettre de l'information à travers plusieurs antennes émettrices pour combattre le bruit de propagation présent dans le canal de transmission et ensuite à détecter les signaux originels en réception par différentes techniques. La capacité du canal s'en trouve accrue malgré la kyrielle de perturbations.

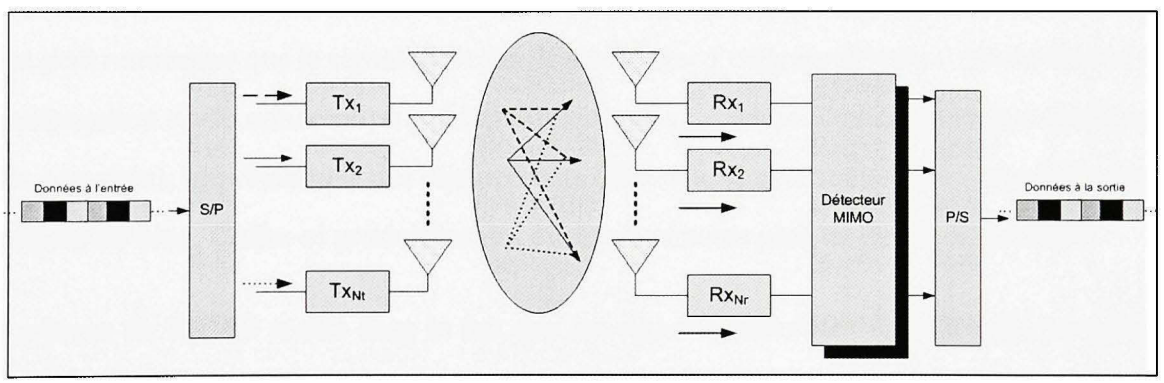


Figure 1.1 Principe de fonctionnement d'un système MIMO.

À l'émetteur, le signal est souvent inadapté à la transmission directe et il doit donc être modifié avant sa propagation. Ce processus peut comporter les opérations énumérées ci-contre :

a. Le codage, qui permet de minimiser les effets de sources dérangeantes inhérentes au système de communication,

b. L'entrelacement des données pour renforcer la robustesse du signal face au bruit dans le medium de propagation,

c. La modulation qui permet de transformer le message issu de la source en un signal adapté au canal de transmission par changement du signal binaire originel en un signal électrique en bande de base ou sur une fréquence porteuse. La technique de modulation employée dans le présent mémoire est la modulation par quadrature de phase (QPSK). Une fois modulées, les données sont envoyées au canal de transmission.

Traditionnellement considérée comme un inconvénient de communication, la propagation multi-trajets est au contraire la propriété clef des systèmes MIMO. Ces derniers, au lieu de la combattre, l'exploitent pour estimer au mieux le signal émis . En effet, sur la Figure 2 , on peut remarquer que le signal subit des dégradations d'origines diverses : phénomènes de propagation sur le milieu physique (affaiblissements), environnement électrique (présence de parasites), imperfections des équipements (distorsions non linéaires) en direction de la station de base. Celles-ci produisent des évanouissements plus ou moins importants.

La force du MIMO réside dans le fait que les signaux provenant de différents transmetteurs (minimalement deux), ne subissent pas nécessairement la même atténuation ou le même évanouissement au même moment à cause des distorsions induites par des phénomènes physiques tels que la réflexion ou la diffusion. Ainsi, il est plus probable qu'à un instant donné, de forts évanouissements que subirait un signal issu d'une antenne donnée, soient compensés par des conditions de canal plus favorables pour le(s) signal(aux) issu(s) de(s) autre(s) transmetteurs. Il existe différents types d'évanouissements qui seront discutés subséquemment. Les applications implémentées dans cette étude considèrent tous des modèles de canal de type Rayleigh. Prenons l'exemple d'un système MIMO à deux antennes en émission et deux antennes en réception, soit ayant quatre coefficients de canal. Il est possible de constater, en modélisant le canal par une distribution de Rayleigh qu'à un instant donné (0.8s dans notre cas), les coefficients de gain du canal $H1$ et $H3$ ne subissent

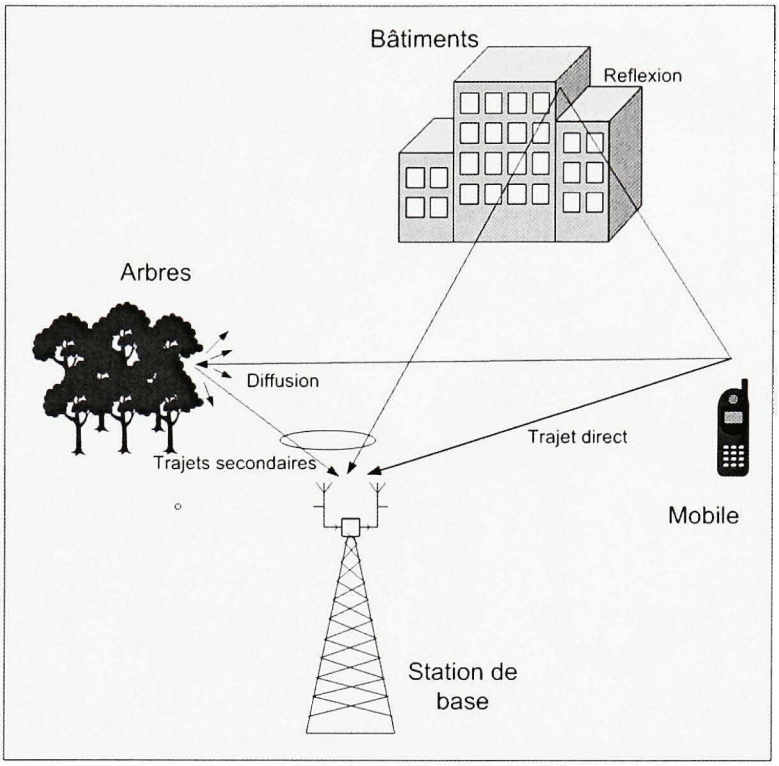


Figure 1.2 Phénomène multi-trajets d'un canal radio mobile.

de dégradations tandis que les deux autres sont atténués respectivement de -20 dB pour $H2$ et -10 dB pour $H4$ comme l'illustre la Figure 3.

Les éléments constitutifs du récepteur sont la détection, par un détecteur linéaire ou non linéaire (Chaboub, 2003) utilisant différents algorithmes tels le forçage à zéro(ZF), la minimisation de l'erreur quadratique moyenne (MMSE)..., la démodulation, le désentrelacement et le décodage des bits reçus. Ce sont donc les opérations inverses à la transmission qui y sont opérées. Dans ce mémoire, nous n'aborderons que l'étude d'un détecteur linéaire utilisant le critère de forçage à zéro ; soit le Zero-Forcing BLAST.

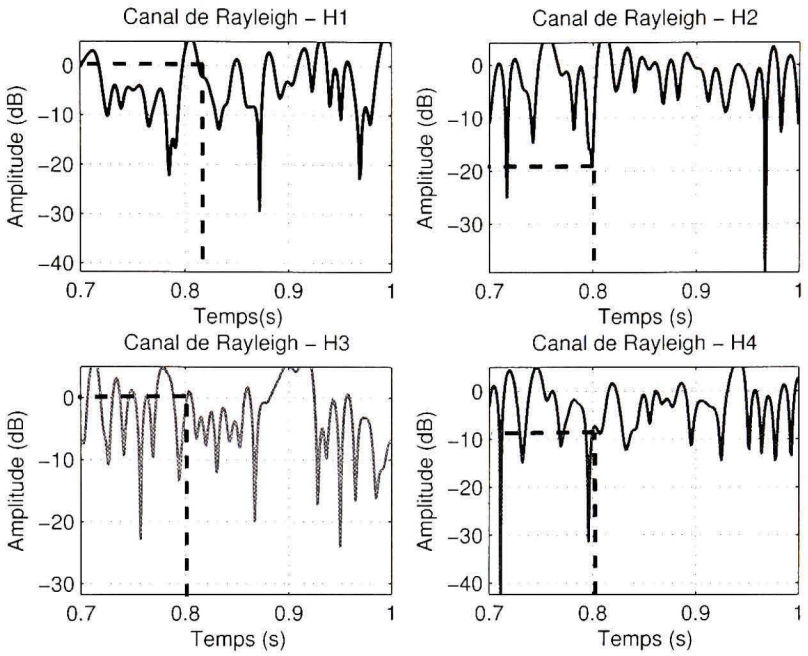


Figure 1.3 Gains d'un canal MIMO à distribution de Rayleigh ayant 4 coefficients.

1.1.2 Le canal de communication

Un canal de communication est un médium physique qui est utilisé pour la transmission d'un signal du transmetteur au récepteur. Dans les communications sans fil, le mode de transmission privilégié est l'onde électromagnétique. Des perturbations aléatoires non prévisibles affectent toujours le signal transmis. Il s'agit alors de la présence de bruit. L'effet du bruit peut être atténué par l'augmentation de la puissance du signal transmis en utilisant un amplificateur radio-fréquences. Toutefois, certaines limitations, à l'exemple de la bande passante allouée à la communication, réduisent son utilisation inconditionnée. Dans cette section, nous considérons particulièrement le canal à bruit blanc additif gaussien et le canal radio mobile à évanouissement de type Rayleigh.

1.1.2.1 Le canal à bruit blanc additif gaussien

Le canal gaussien est fréquemment utilisé pour étudier les performances d'un système de transmission. Le signal reçu $s(t)$ est la résultante de l'addition du signal originel $s(t)$ avec un bruit blanc gaussien modélisant une fonction de probabilité gaussienne nommée $n(t)$ comme le montre l'équation 1.1 :

$$r(t) = s(t) + n(t) \tag{1.1}$$

Le bruit blanc additif gaussien est un bon modèle pour les communications par satellite et sur de grandes distances. Il modélise les imperfections des équipements, le bruit de l'antenne ... Employé seul, il est moins pertinent pour les liaisons terrestres car celles-ci sont fortement influencées par les phénomènes multitrajets,les interférences, les non linéarités, la sélection en fréquences ... Par contre, sa combinaison en tant que bruit de fond avec un modèle de canal multitrajets est employée dans les systèmes de communication terrestres modernes. C'est elle que nous avons choisie pour les deux premières applications étudiées dans ce mémoire : le ZF-VBLAST et le récepteur Rake . Le calcul du rapport signal-à-bruit (SNR) ne dépendra que de la variance du bruit $SNR = \frac{1}{N_0} = \frac{1}{\sigma^2}$. La comparaison des performances se fera préférentiellement à l'aide de cette métrique. Le bruit blanc est caractérisé par un processus aléatoire centré, donc de moyenne nulle et de densité spectrale bilatérale correspondant à $N_0/2$.

Le récepteur optimal, celui qui donne le meilleur taux d'erreur, est constitué d'un filtre de mise en forme $g(t)$, suivi d'un échantillonneur dont le pas est T et d'un détecteur de seuil (Wozencraft et al.,1965). De plus, il faut que $h(t) = g(t) * g^*(-t)$ vérifie le critère de Nyquist au pas T, c'est-à-dire que $h(kT) = \delta_{k,0}$. La Figure 4 illustre ce modèle et la partie en pointillé quant à elle met en exergue le modèle du bruit blanc.

Le signal émis $s(t)$ est issu d'un filtre de mise en forme $g(t)$ pour respecter la finitude de la largeur de bande du canal et produit des éléments binaires d_k après chaque T secondes.

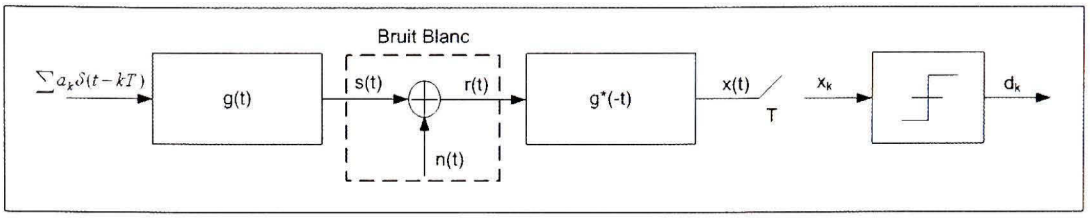


Figure 1.4 Récepteur optimal pour un canal gaussien.

Il admet l'écriture présentée à l'équation 1.2 :

$$s(t) = \sum a_k g(t - kT) \tag{1.2}$$

On démontre alors que le signal x_k à la sortie selon la Figure 4 , dépend de la fréquence d'opération de l'échantillonneur et obéit à l'équation 1.3

$$x_k = a_k + n_k, \tag{1.3}$$

où n_k correspond à l'échantillonnage aux instants kT du bruit blanc filtré par $g^*(-t)$.

Les échantillons n_k sont indépendants, uniformément distribués, répartis selon une gaussienne centrée et les symboles de sorties x_k sont à temps discret et à valeurs continues. La règle de décision optimale d_k s'obtient en prenant les valeurs seuil de x_k.

$$x_k = h(0)a_k + \sum_{l \neq 0} h(lT)a_{k-l} + n_k \tag{1.4}$$

Le terme $\sum_{l \neq 0} h(lT)a_{k-l}$ représente alors l'interférence intersymboles et celle-ci complique l'estimation du signal émis a_k. D'où l'importance de la combattre par des techniques d'égalisation telles que le forçage à zéro utilisé dans le multiplexage spatial présenté dans le prochain chapitre.

Dans le cas où la réponse impulsionnelle générale du canal h n'obéit pas au critère de Nyquist, autrement dit, lorsque la durée des symboles est plus longue que l'instant d'échantillonnage T, la sortie x_k est fonction de symboles supplémentaires distincts de a_k. Dans ce cas, l'expression de x_k est donnée par la relation 1.4.

1.1.2.2 Le canal à évanouissement

La multiplicité des facteurs impliqués dans la propagation dans un environnement cellulaire mobile conduit à l'application des techniques statistiques afin de décrire la variation des signaux. En réception d'un canal mobile, le signal obtenu provient de la superposition de plusieurs répliques du signal émis. Ces répliques proviennent soit de la réflexion, soit de la diffusion ou de la réfraction de l'onde électromagnétique originelle causant ainsi des évanouissements (en anglais 'fading').

On classifie les évanouissements en deux principales catégories en fonction de la proximité ou non entre l'émetteur et le récepteur. La première est l'évanouissement à grande échelle qui se caractérise par une diminution de la puissance moyenne du signal à la réception. Cette situation survient lorsque le transmetteur et le récepteur sont très éloignés et lorsque des obstacles saillants entre les deux (immeubles, arbres...) sont présents. La seconde est celle à petite échelle. Cette dernière résulte de changements infimes au niveau de l'amplitude et de la phase du signal reçu. Les trajets multiples occasionnent alors un étalement temporel, différence entre le plus grand et le plus court des retards entre les répliques et le signal initial, ce qui permet de caractériser par une seule variable la dispersion temporelle du canal. Les trajets multiples définissent aussi la sélectivité fréquentielle du dit signal.

Dans un système à bande passante étroite, les signaux transmis occupent généralement une bande passante plus petite que la bande passante de cohérence. Cette dernière est l'intervalle de fréquences dans lequel le processus d'évanouissement du canal est corrélé ou, en d'autres termes, il s'agit de la bande à l'intérieur de laquelle toutes les compo-

santes spectrales du signal transmis subissent la même atténuation (Vucetic, 2003). Le type d'évanouissement subi est alors qualifié de non sélectif en fréquence (frequency non selective ou frequency flat). La bande de cohérence,B_c, est du même ordre de grandeur que l'inverse de l'étalement temporel : $B_c \sim \frac{1}{T_m}$. Par ailleurs, si la bande passante du signal émis, B_s, est plus grande que la bande de cohérence, les composants spectraux des données émises subissent des distorsions non corrélées. Le spectre du signal reçu est alors diforme à cause de la non-homogénéité de la relation entre ses composantes spectrales et celles du signal originellement transmis (Vucetic, 2003). On qualifie cet état de sélectivité en fréquence (frequency selective). Les systèmes à large bande sont généralement sélectifs en fréquence. Le tableau I résume la distinction entre la sélectivité en temps et la sélectivité en fréquence.

Tableau I

Sélectivité en temps et en fréquence pour un canal radio

Sélectif en fréquence	Non sélectif en fréquence
$B_s \gg B_c$	$B_s \ll B_c$
Sélectif en temps	Non sélectif en temps
$T_s \gg T_c$	$T_s \ll T_c$
Non sélectif ni en fréquence ni en temps	$T_m \ll T_s \ll T_c$

Lorsqu'on se trouve en ligne de vue directe (LOS) et que le nombre de trajets est important, le signal reçu obéit à un modèle de canal de Rice. Par contre, lorsque le trajet direct fait défaut (NLOS), le signal suit une distribution de Rayleigh. Nous allons restreindre nos propos au cas du canal de Rayleigh car les simulations effectuées dans ce mémoire sont réalisées à l'aide de ce canal qui sera supposé parfaitement connu du récepteur. Auparavant, nous présenterons sommairement l'effet Doppler.

L'effet Doppler

L'effet Doppler est le décalage constant entre la fréquence de l'onde émise et celle de l'onde reçue lorsque l'émetteur et le récepteur sont en mouvement l'un par rapport à l'autre avec une vitesse radiale constante. Dans un contexte de propagation à un seul trajet où une porteuse f_c est transmise et où le signal en réception consiste en une onde arrivant avec un angle d'incidence θ par rapport à la direction de déplacement du mobile, l'effet Doppler du signal reçu, f_d est défini par l'équation 1.5 (Vucetic, 2003) :

$$f_d = \frac{v f_c}{c} cos\theta \tag{1.5}$$

où v est la vitesse du mobile et c la célérité de la lumière.

Dans un environnement multi-trajets, un signal transmis avec fréquence unique subit une dispersion fréquentielle due à l'inconstance des caractéristiques du canal durant la propagation, résultant en un signal reçu avec spectre non nul. L'effet Doppler entraîne alors un étalement de la bande de fréquence occupée par le signal dans un intervalle compris entre $f_c - f_{d_{max}}$ et $f_c + f_{d_{max}}$ où $f_{d_{max}}$ est la fréquence maximale de l'effet Doppler encore appelée taux d'évanouissement maximal, définie par l'équation 1.6 (Vucetic, 2003) :

$$f_{d_{max}} = \frac{v f_c}{c} \tag{1.6}$$

L'effet Doppler peut être considéré comme le pendant fréquentiel de l'étalement temporel. On peut alors définir l'étalement fréquentiel B_m correspondant à la différence entre le plus grand et le plus petit décalage en fréquence inhérent aux multiples trajets et le temps de cohérence du canal, T_c, comme étant l'intervalle durant lequel les distortions temporelles du canal restent négligeables. Traditionnellement ce temps est du même ordre de grandeur que l'inverse de l'étalement fréquentiel $T_c \sim \frac{1}{B_m}$.

Canal de Rayleigh

Le canal de Rayleigh décrit les variations du signal dans un environnement multitrajet à bande passante étroite. Comme mentionné plus haut, le passage du signal émis à travers le canal produit des répliques issues de la réflexion, de la diffusion ou de la réfraction du signal de départ. Lorsque le nombre d'ondes réfléchies est élevé, selon le théorème central limite, deux composants en quadrature du signal reçu sont des variables aléatoires gaussiennes non corrélées de moyenne nulle et de variance σ_s^2. Ainsi, l'amplitude instantanée du signal reçu $R \sim Rayleigh(\sigma)$, obéit à une distribution de Rayleigh si $R = \sqrt{X^2 + Y^2}$ où $X \sim N(0, \sigma^2)$ et $Y \sim N(0, \sigma^2)$ sont deux distributions normales. La présence des deux composantes normales explique la raison pour laquelle l'enveloppe du signal reçu obéit à une distribution de probabilité de Rayleigh dans le domaine temporel et sa phase suit une distribution uniforme comprise entre $-\pi$ et π. La fonction de densité de probabilité d'une distribution de Rayleigh est donnée par 1.7 (Vucetic, 2003) :

$$p(a) = \frac{a}{\sigma_s^2}.e^{-a^2/2\sigma_s^2} \quad si \quad a \geq 0, \qquad p(a) = 0 \quad si \quad a < 0 \tag{1.7}$$

La valeur moyenne, m_a, et la variance, σ_a^2 d'une variable aléatoire de type Rayleigh sont données par 1.8 (Vucetic, 2003) :

$$m_a = \sqrt{\frac{\pi}{2}}\sigma_s = 1.2533\sigma_s, \quad \sigma_a^2 = (2 - \frac{\pi}{2})\sigma_s^2 = 0.4292\sigma_s^2 \tag{1.8}$$

Une fois normalisée, la fonction de densité présentée en 1.7 devient 1.9 (Vucetic, 2003) :

$$p(a) = 2ae^{-a^2} \quad si \quad a \geq 0, \quad p(a) = 0 \quad si \quad a < 0 \tag{1.9}$$

La valeur moyenne et la variance quant à elles deviennent 1.10 (Vucetic, 2003) :

$$p(a) = 2ae^{-a^2} \quad si \quad a \geq 0, \quad p(a) = 0 \quad si \quad a < 0 \tag{1.10}$$

La fonction de densité de probabilité d'une distribution de Rayleigh normalisée est illustrée à la Figure 5 :

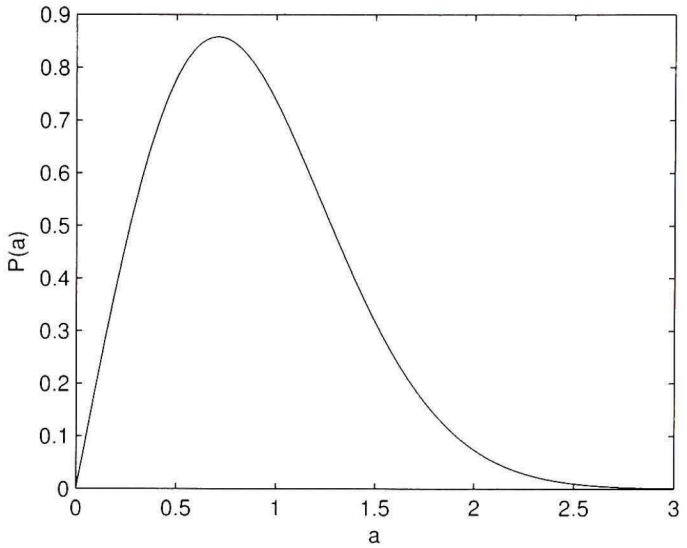


Figure 1.5 La fonction de densité de probabilité d'une distribution de Rayleigh.

Dans les canaux à évanouissements ayant un décalage Doppler valant $f_{d_{max}}$, les signaux reçus subissent un étalement fréquentiel dans une bande limitée entre $f_c \pm f_{d_{max}}$. En supposant une antenne omnidirectionnelle recevant des ondes sur le plan horizontal, un grand nombre d'ondes réfléchies et une puissance uniformément répartie selon l'angle incident, la densité spectrale de puissance de l'amplitude dégradée par évanouissemnts, notée $|P(f)|$, est donnée par l'équation 1.11 (Vucetic, 2003) :

$$|P(f)| = \frac{1}{2\pi\sqrt{f_{d_{max}}^2 - f^2}} \quad si \quad |f| \leq |f_{d_{max}}|, \quad |P(f)| = 0 \quad sinon, \qquad (1.11)$$

où f est la fréquence et $f_{d_{max}}$ est le taux d'évanouissement maximal. La valeur de $f_{d_{max}} T_s$ est le taux d'évanouissement maximal normalisé par le taux de symboles. Il sert de mesure à la mémoire du canal.

L'intérêt de l'emploi des techniques de transmission MIMO réside dans la possibilité d'accroître les performances d'un système de communication numérique donné grâce à la technique de diversité présentée ci-dessous.

1.1.3 La notion de diversité

La diversité est une technique employée pour combattre les évanouissements. Son principe consiste recevoir plusieurs répliques, M de la même information sur plusieurs canaux ayant des puissances comparables et des évanouissements indépendants pour accroître la probabilité de véhiculer une information de bonne qualité c'est-à-dire très faiblement atténuée. Autrement dit, leurs rapports signal-à-bruit (SNR), ρ_i, se trouvent tous en dessous du seuil de fonctionnement du système, ρ_{seuil}, qui est d'autant plus faible que le nombre M est élevé comme le montre l'équation 1.12 :

$$Pr(\rho_1 < \rho_{seuil}, \rho_2 < \rho_{seuil}, ..., \rho_M < \rho_{seuil}) = \Pi_{i=0}^{M-1} Pr(\rho_i < \rho_{seuil}), \tag{1.12}$$

où Pr est la probabilité qu'une observation soit inférieure à la valeur seuil.

L'une des techniques de diversité consiste à transmettre explicitement ou directement plusieurs répliques du même message à partir de l'émetteur. C'est le cas dans le codage spatio-temporel où la diversité des versions des données reçues suite à leur transmission en copies multiples à travers des antennes est employée pour améliorer la fiabilité de transmission de ces données (Alamouti S., 1998). L'autre consisterait à n'envoyer qu'une copie du signal émis mais à compter sur les trajets multiples presqu'inévitables lors de la transmission, pour obtenir plusieurs versions du signal envoyé. On trouve dans la littérature différentes formes de diversité (Benedetto S. et al., 1999 ; Biglieri E., 2007 ; Naguib et al., 2000) permettant d'obtenir une observation du SNR au-dessus du seuil de fonctionnement du système.

1.1.3.1 La diversité spatiale

La séparation physique de plusieurs antennes par une distance suffisante pour assurer la décorrélation des atténuations des signaux transmis individuellement fait surgir la notion de diversité spatiale. La taille de chacune des antennes, l'environnement de propagation ainsi que la fréquence, sont des éléments-clés qui déterminent la séparation nécessaire. Elle est habituellementde l'ordre de 10λ (λ étant la longueur d'onde du signal) pour la station de base (mais peut aussi être inférieure) et beaucoup moindre pour une station mobile. Dans ce cas, la redondance de l'information survient au récepteur dans le domaine spatial et ne réduit pas l'efficacité de l'utilisation de la bande passante. La diversité en transmission et celle en réception font partie intégrante de la diversité spatiale, selon que l'on se trouve respectivement à l'émetteur ou au récepteur. La diversité de polarisation et la diversité angulaire sont deux exemples de diversité spatiale.

Dans le premier cas, les polarisation horizontale et verticale sont transmises par deux antennes différentes polarisées et reçues par deux antennes différentes et polarisées (Vucetic, 2003). Le fait de choisir des polarisations différentes assure le fait que les deux signaux ne sont pas corrélés et ce, sans avoir à placer les antennes très éloignées l'une de l'autre. Cette technique permet d'atteindre une diversité d'ordre 2 pour des signaux de polarisations orthogonales lorsque le canal présente des évanouissements décorrélés. Les antennes doivent être utilisées par paires car il n'existe que deux plans de polarisation. Un avantage majeur de cette technique par rapport à la diversité spatiale est l'absence de contrainte sur l'écartement relatif entre les deux antennes ; ce qui est hautement attrayant pour les unités mobiles.

La diversité angulaire, quant à elle, est employée pour des transmissions dont la fréquence porteuse est supérieure à 10 GHz. Dans ce cas, comme les signaux émis sont suffisamment séparés angulairement, les signaux reçus sont indépendants les uns des autres ou décorrélés.

Il est possible de classer la diversité spatiale en deux catégories, en fonction de la multiplicité des antennes utilisées au transmetteur et au récepteur (Liu et al., 2001). La diversité en réception suppose que les multiples antennes réceptrices sélectionnent différentes copies des signaux transmis afin d'accroître le rapport signal à bruit global et de réduire les évanouissements dûs aux multi-trajets. Dans les techniques de diversité en transmission, les signaux émis sont entrelacés dans l'espace avant d'arriver au récepteur dans le but de renforcer la robustesse de ces signaux face au bruit dans le canal. La diversité en transmission peut accroître considérablement la capacité du canal.

1.1.3.2 La diversité temporelle

Elle est atteinte lorsque l'on transmet un signal sur plusieurs intervalles temporels espacés par au moins le temps de cohérence du canal dans le but de réduire la corrélation entre les signaux évanouis au récepteur. Cette technique est très effective dans les environnements à évanouissements rapides parce que pour ceux-ci, le temps de cohérence du canal étant négligeable, il est nécessairement plus petit que les intervalles de temps nécessaires à la transmission des données. Cependant, le principal inconvénient de cette approche découle des pertes en bandes de fréquences. L'entrelacement combiné avec un code correcteur d'erreur est un exemple de diversité temporelle (Vucetic, 2003). Dans ce cas, des répliques du signal transmis sont reçues au récepteur sous forme de redondance dans le domaine temporel par le code correcteur (Naguif A.F. et al., 2000). La séparation temporelle entre les répliques du signal transmis est alors atteinte en utilisant l'entrelacement temporel pour obtenir des évanouissements indépendants à l'entrée du décodeur.

1.1.3.3 La diversité fréquentielle

Elle revient à transmettre un signal sur plusieurs fréquences porteuses différentes dont l'écartement fréquentiel est d'au moins la bande de cohérence du canal. Elle se caractérise donc par la transmission du même message en différentes bandes de fréquences. Pour limiter la corrélation entre les signaux bruités reçus, la séparation fréquentielle devrait

être en accord avec la bande de cohérence. Comme la diversité temporelle, la diversité en fréquence introduit une utilisation sous-optimale de la bande passante, cette fois-ci due à la redondance des informations transmises dans le domaine fréquentiel.

1.1.3.4 La combinaison des répliques

Les fluctuations du rapport signal à bruit autour de sa moyenne, les variations brutales de la probabilité d'erreur d'un signal et la présence des évanouissements de canal lors de la transmission d'un signal ont rendu nécessaire l'introduction de la notion de diversité. Celle-ci est couramment utilisée dans les canaux sans fils pour combattre les évanouissements. Son principe s'attèle à exploiter le nombre de répliques indépendantes du signal après qu'il ait subi des évanouissements. Les précédents paragraphes ont permis la classification des techniques de diversité en fonction du domaine où la diversité était introduite. Or, à la réception, les antennes multiples combinent les signaux dupliqués de sorte que le signal résultant possède une variation d'amplitude moins prononcée comparativement aux mêmes conditions avec une antenne réceptrice. La combinaison de ces répliques peut se faire de plusieurs manières (Vucetic, 2003) :

a. On peut procéder par la commutation. Tout en respectant une valeur seuil, la sélection se circonscrit à la branche ayant le plus fort rapport signal à bruit.

b. Une autre possibilité est celle qui se sert de l'approche à gain maximal selon 1.13 (Vucetic, 2003) :

$$r_i = A_i e^{j\theta_j} s + n_i,\ r = \sum_{i=1}^{L} A_i e^{-j\theta_i} r_i \tag{1.13}$$

Avec r_i correspondant au signal reçu sur l'antenne i et $A_i e^{-j\theta_i}$ représentant le facteur de pondération reçu à l'antenne i. Dans ce cas, la variable de décision est une combinaison linéaire de plusieurs répliques. Les coefficients de pondération des répliques sont déterminés de façon à maximiser le rapport entre l'énergie instantanée

et la densité spectrale du bruit et lorsqu'il est optimal, il correspond à l'atténuation associée.

c. Enfin, il existe le gain égal selon 1.14 (Vucetic, 2003) :

$$r_i = A_i e^{j\theta_j} s + n_i, \; r = \sum_{i=1}^{L} e^{-j\theta_i} r_i \tag{1.14}$$

Avec r_i correspondant au signal reçu sur l'antenne i et $A_i e^{-j\theta_i}$ représentant le facteur de pondération reçu à l'antenne i.

Ici, la variable de décision se limite à la somme des répliques. C'est une méthode sous-optimale mais linéaire et simple. Sa complexité d'implémentation est moindre comparativement à celle de la méthode à gain maximal, avec des performances comparables.

1.1.4 La notion de capacité

Shannon (Shannon C.E., 1948) a mis en évidence le bruit comme étant la source des erreurs de transmission. Il a à cet effet proposé d'ajouter de la redondance au message à transmettre pour lutter contre ce phénomène et introduit une métrique nouvelle : la capacité du canal. Celle-ci mesure la quantité d'information maximale ou le débit maximal qui peut être transmis(e) à travers un canal.

1.1.5 La capacité d'un canal gaussien à bande limitée

Un système mono-antenne (SISO) étant perturbé seulement par un bruit blanc additif gaussien peut atteindre une capacité maximale de canal (Shannon, 1948 ; Proakis, 1995 ; Telatar E., 1999) donnée par l'équation $\frac{P}{N_0 W}$:

$$C = W log_2(1 + \frac{P}{N_0 W}) \quad bits/s, \tag{1.15}$$

où C est la capacité en $bits/s$, W la bande passante limitée du signal, P la puissance moyenne du signal sur la durée d'observation, N_0 la densité spectrale monolatérale du bruit et $\frac{P}{N_0 W}$ le SNR en réception.

La variation de la capacité normalisée C/W (appelée efficacité spectrale) d'un canal gaussien en fonction du SNR est présentée ci-dessous (Figure 6). La capacité est donc une fonction strictement croissante du SNR.

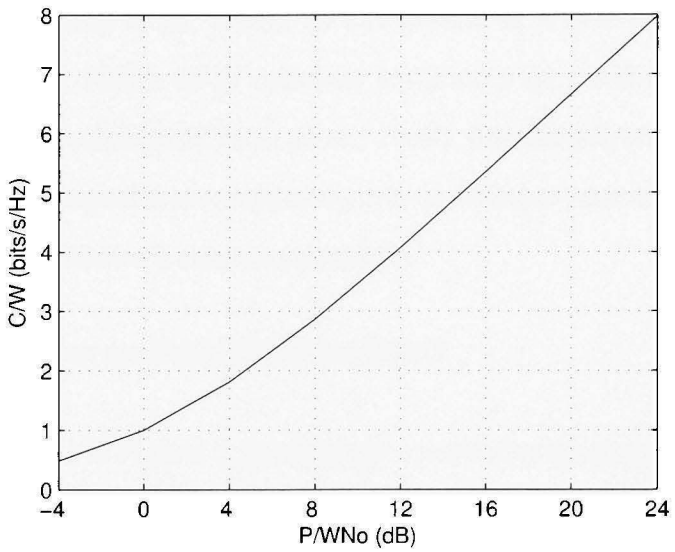


Figure 1.6 C/W en fonction du SNR.

1.1.6 La capacité d'un canal de Rayleigh à bande limitée

Dans le cas des canaux à évanouissements, le module de l'atténuation correspond à une variable de Rayleigh (section 1.1.2.2). En considérant un seul trajet d'atténuation réelle, il est possible d'introduire un gain gaussien h d'amplitude complexe et de puissance unité et en normalisant l'expression de la capacité par W, la capacité instantanée étant donnée par l'équation (1.16) :

$$C = log_2(1 + |h|^2 . \frac{P}{N_0 W}) \quad bits/s/Hz \tag{1.16}$$

Il est judicieux de remarquer, selon la Figure 6, que pour augmenter la capacité de 1 bit/s/Hz, il faut un gain de 3dB au niveau du SNR, soit une augmentation de la puissance de l'émetteur du double. Or, il n'est pas possible d'accroître indéfiniment la puissance du signal émis pour pourvoir à des débits de transferts élevés. En effet, plusieurs scientifiques ont essayé de déterminer la capacité du système dans un contexte de propagation moins indulgent que celui considéré en théorie de l'information par Foschini (Foschini G.J. et al.,1998) et Telatar (Telatar E., 1999), en tenant compte de la corrélation entre les évanouissements (Shiu et al., 2000), de l'existence de trajets directs en ligne de vue (LOS) (Driessen et al., 1998), de la variation temporelle du canal ou encore d'une allocation dynamique de puissance (Chuah et al., 1998). En considérant tous ces facteurs, la valeur théorique de la capacité est amenuisée. Ces problèmes sont mieux combattus grâce à l'introduction des systèmes à antennes multiples.

1.1.7 La capacité d'un système SIMO ou MISO

Pour le cas d'un SIMO à la Figure 7, le canal est constitué respectivement de N_r (N_t) coefficients distincts, $h = [h_{11},\ h_{21}, h_{31},\ ...,\ h_{N_r1}]$ ($h = [h_{11},\ h_{12}, h_{13},\ ...,\ h_{1N_t}]$) où h_{i1} (h_{1i}) est le coefficient de canal entre l'antenne d'émission et l'antenne de réception i (l'antenne d'émission i et l'antenne de réception). Le cas d'un système MISO peut être facilement déduit à partir d'un système SIMO. Nous limiterons notre propos au système SIMO. La capacité de 1.16 peut alors être généralisée avec l'équation 1.17 :

$$C = log_2(1 + \frac{P}{N_0 W}.h.h^*) \quad bits/s/Hz \tag{1.17}$$

1.1.8 La capacité d'un système MIMO

Considérons le système MIMO possédant N_t antennes émettrices ($Tx_i\ avec\ i \in \{1, ..., N_t\}$) et N_r antennes réceptrices ($Rx_i\ avec\ i \in \{1, ..., N_r\}$) tel que représenté à la Figure 8 :

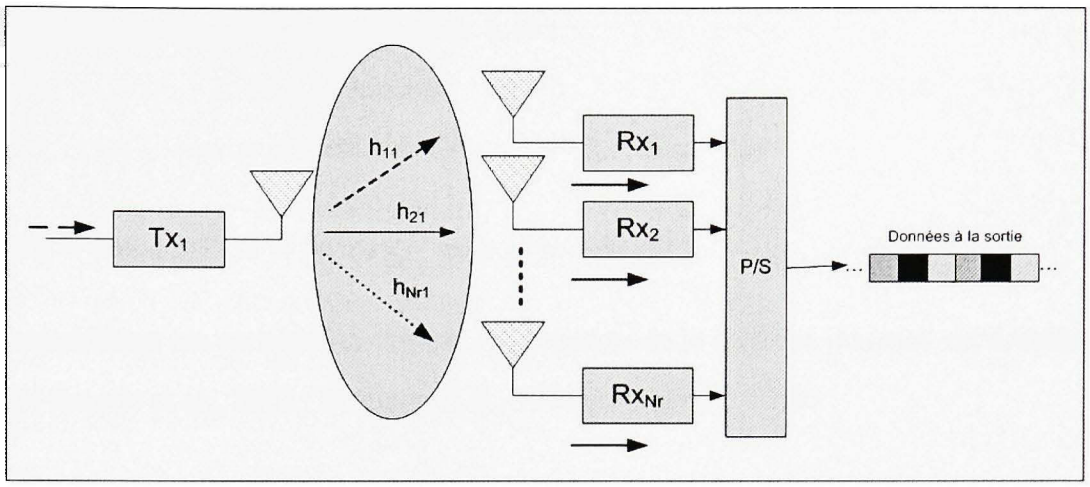


Figure 1.7 Schéma d'un système SIMO.

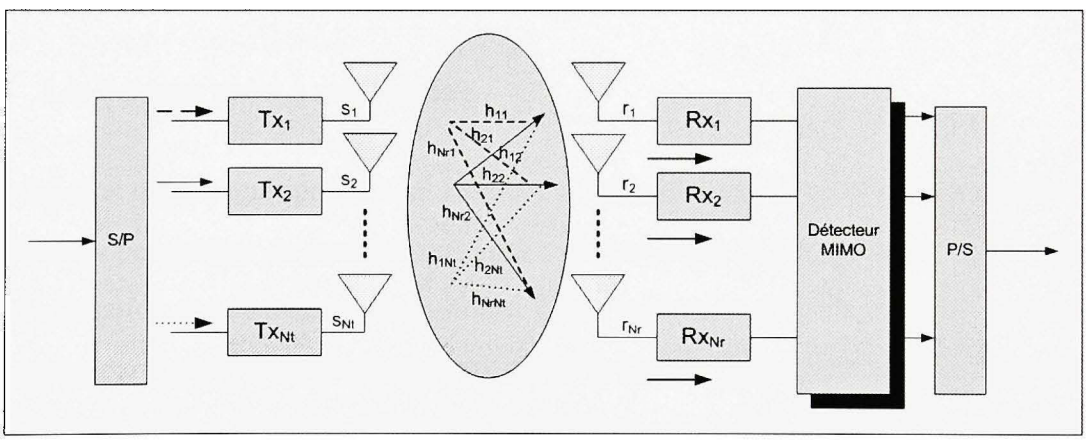


Figure 1.8 Schéma d'un système MIMO.

L'expression de la matrice de canal est donnée par l'équation 1.18 :

$$H = \begin{bmatrix} h_{11} & \dots & h_{1N_t} \\ . & & . \\ . & & . \\ . & & . \\ h_{N_r1} & \dots & h_{N_rN_t} \end{bmatrix} \tag{1.18}$$

Le signal reçu correspond à la relation 1.19 :

$$r = Hs + n \tag{1.19}$$

avec $r = [r_1 ... r_{N_r}]^T$, $s = [s_1 ... s_{N_t}]^T$ et $n = [n_1 ... n_{N_r}]^T$

En considérant les variables ci-dessus, l'expression de la capacité du canal est donnée par 1.20 (Vucetic et al., 2003 ; Shannon C.E., 1948 ; Telatar E., 1999) :

$$C = log_2(I_{N_r} + \frac{P}{N_t.(N_0 W)}.H.H^*) \ \ bits/s/Hz \tag{1.20}$$

On démontre qu'elle croît de n bits/s avec $n = min(N_t, N_r)$ lorsque le SNR croît de 3 dB.

1.2 Les liaisons point à multipoints

Les liaisons point à multipoints sont employées dans les techniques d'accès multiples. Ces techniques permettent de partager un canal commun entre plusieurs usagers pour optimiser l'utilisation des resources du système. Toute technique d'accès multiple est assortie d'un type particulier de duplexage permettant l'échange d'informations entre le mobile et la station de base et vice-versa. En effet, les liaisons sont généralement bidirectionnelles : de la station de base vers le terminal (liaison descendante) ou le contraire (liaison montante). Lorsque les débits sur les deux voies sont identiques, le service est dit symétrique. Autrement, on qualifie d'asymétrique tout service dont le débit descendant est supérieur au débit montant. Les deux techniques de duplexage les plus fréquentes sont : le duplexage en fréquence (FDD pour Frequency Division Duplex) et le duplexage dans le temps (TDD pour Time Division Duplex). Le mode FDD utilise deux bandes de fréquences indépendantes séparées par un écart fréquentiel spécifié par le système. Il est qualifié de système $full\ duplex$ (Krüger et al., 2004) car il permet d'envoyer et de recevoir des informations en même temps. Le duplexage temporel quant à lui permet aux deux voies d'utiliser la

même fréquence porteuse mais à des instants différents synchronisés. Il est aussi qualifié de $half\ duplex$. L'interface radio du standard UMTS utilise ces deux types de duplexage, tel que nous le verrons dans le prochain chapitre.

Parmi les nombreuses techniques d'accès multiples existantes seront passées en revue dans les prochains paragraphes d'une part l'accès multiple par repartition en fréquences (FDMA), l'accès multiple par repartition dans le temps (TDMA)et l'accès multiple par repartition de codes (CDMA). D'autre part, l'accès multiple par repartition dans l'espace (SDMA) et l'accès multiple par répartition en fréquences orthogonales (OFDMA) seront également présentés brièvement. Dans le cas du CDMA, nous introduirons la notion d'étalement de spectre par séquence directe (DS-CDMA) exploitée dans les systèmes UMTS.

1.2.1 Les systèmes FDMA

La Figure 9a illustre la technique d'accès multiple à répartition en fréquences. Celle-ci consiste à découper la bande passante disponible en plusieurs sous-bandes distinctes. Chacun des utilisateurs se voit alors attribuer une sous-bande associée à une fréquence porteuse. L'usager transmet continûment à la fréquence particulière Δf et l'ensemble des différentes fréquences porteuses issues des différents émetteurs constitue le canal. Le récepteur choisit la porteuse qui lui est appropriée pour déchiffer les données qui lui sont destinées.

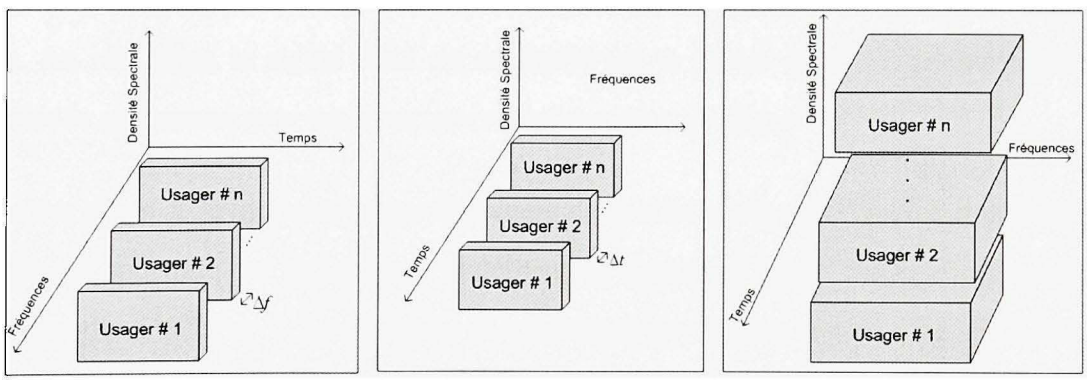
1.2.2 Les systèmes TDMA

Dans cette technique, la bande passante est utilisée par tous les utilisateurs mais leur répartition se fait de manière temporelle. Chaque utilisateur envoie des informations sous forme de trames sur un intervalle de temps Δt en occupant toute la bande passante. Ces informations une fois mises en rafales, sont transmises à des intervalles de temps disjoints dits "slots". Chacun des intervalles correspond à un usager tel qu'indiqué à la Figure 9b. Le canal se comporte donc comme la succession des slots en provenance des différents utili-

sateurs. Au récepteur, la synchronisation est nécessaire afin bien détecter les informations émises.

1.2.3 Les systèmes CDMA

Cette solution permet d'établir simultanément plusieurs liaisons dans le même intervalle de temps et dans la même bande de fréquences. Ainsi, plusieurs usagers souhaitant établir une communication, peuvent avoir accès aux mêmes ressources grâce à l'attribution d'un code d'étalement à chacun d'entre eux (Figure 9c). Théoriquement, l'orthogonalité des signatures assure une séparation rigoureuse des canaux. Cependant, le niveau d'intercorrélation entre ces signatures est faible en pratique et le niveau d'interférence s'en trouve minimisé. Dans le cas de l'étalement par séquence directe, les codes sont des séquences pseudo-aléatoires. Ils sont utilisés dans les systèmes radio mobiles de troisième génération.



(a) Répartition par fréquences (b) Répartition dans le temps (c) Répartition par codes

Figure 1.9 Différentes techniques d'accès multiples de troisième génération.

1.2.3.1 L'étalement de spectre

L'étalement de spectre a pour but de répartir, sur une large bande de fréquences, l'information originellement contenue dans un spectre de fréquences limité afin de diminuer les effets du bruit dans le système. Ce résultat dérive de la multiplication du signal binaire

d'origine (de fréquence relativement faible) par un signal, généralement pseudo-aléatoire (PN) dont les composants sont appelés *chips* ou bribes, de fréquence beaucoup plus élevée. Le facteur d'étalement est alors défini comme étant le rapport du signal après étalement par le signal non étalé (Kruger, 2004). C'est aussi le rapport entre le taux de chips par le taux de symboles (Kruger, 2004). Ainsi, la puissance du signal se retrouve répartie sur toute la nouvelle largeur de bande du signal. Le signal est immergé dans le bruit. Dans les communications cellulaires, ce dernier pourrait être issu de l'interférence mutuelle entre les utilisateurs de la même cellule (interférence intra-cellulaire) dû à la présence de trajets multiples, à la non-synchronisation et à l'indépendance de la transmission entre les utilisateurs qui occasionnent l'imperfection de l'orthogonalité des codes. Il pourrait aussi provenir des interférences dues aux signaux envoyés par les mobiles des cellules voisines. Il s'agit de l'interférence inter-cellulaire. Le principe de l'étalement de spectre est illustré à la Figure 10. Au transmetteur, le signal d'entrée est multiplié par une séquence pseudo-aléatoire pour produire un signal étalé dans le domaine spectral et prêt pour la transmission. Au récepteur, le signal étalé est multiplié par la même séquence pseudo-aléatoire employée au transmetteur pour retrouver l'information de départ.

L'un des effets de la modulation par étalement de spectre sur le signal reçu est de diminuer sa densité spectrale par le facteur d'étalement de spectre (Kruger, 2004) comme montré aux équations 1.21 et 1.22 . Le gain de traitement (en anglais *processing gain*) est exprimé par $Processing\ Gain = 10 * log(Spreading\ Factor)$. Chaque chip du code étalé, transporte donc l'énergie de chacun des symboles fractionné par le facteur d'étalement. Enfin, l'équation 1.23 (Kruger, 2004) marque le lien entre le rapport signal-à-bruit (SNR) et le rapport signal-à-interférence ($\frac{E_b}{N_0}$).

$$E_c = \frac{E_b}{Spreading\ Factor} \tag{1.21}$$

$$E_{c_{dB}} = E_{b_{dB}} - 10 * log(Spreading\ Factor) \tag{1.22}$$

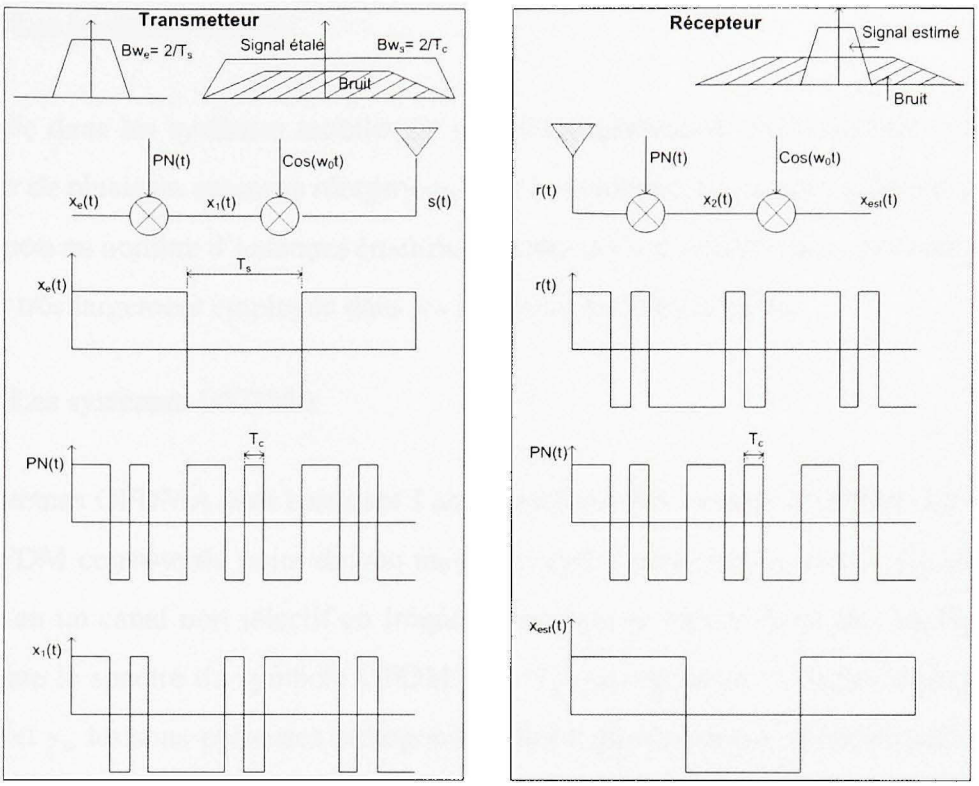


Figure 1.10 Principe de l'étalement par séquence directe.

$$\frac{E_b}{N_0} = \frac{W}{R}.SIR \tag{1.23}$$

avec W qui est largeur de bande du signal après étalement et R, la largeur de bande avant l'étalement.

Au récepteur, l'estimation du signal originel est obtenue en multipliant le signal reçu par la même séquence pseudo-aléatoire qu'à l'émetteur. Parmi les multi-trajets reçus, seul le signal qui s'était vu attribuer la même séquence PN au départ verra sa largeur de bande réduite tandis que le bruit demeurera étalé sur la largeur de bande avec les autres signaux non détectés.

1.2.4 Les systèmes SDMA

Exploitée dans les systèmes mobiles de quatrième génération, cette méthode consiste à disposer de plusieurs antennes réceptrices. Plus le nombre d'antennes réceptrices est élevé par rapport au nombre d'antennes émettrices moins il y a d'interférences entre les usagers. Elle est très largement employée dans les systèmes multiantennaires.

1.2.5 Les systèmes OFDMA

Les systèmes OFDMA sont basés sur l'accès multiple des symboles OFDM. La modulation OFDM consiste du point de vue mathématique à convertir un canal sélectif en fréquence en un canal non sélectif en fréquence comme le montre la figure. La Figure 11 représente le spectre du symbole OFDM avec T_s qui représente le temps d'un symbole OFDM et s_i, les sous-porteuses orthogonales. Sur le plan pratique, la conversion en canal non sélectif est atteinte en modulant un certain nombre de sous-porteuses orthogonales avec un taux de symboles réduit.

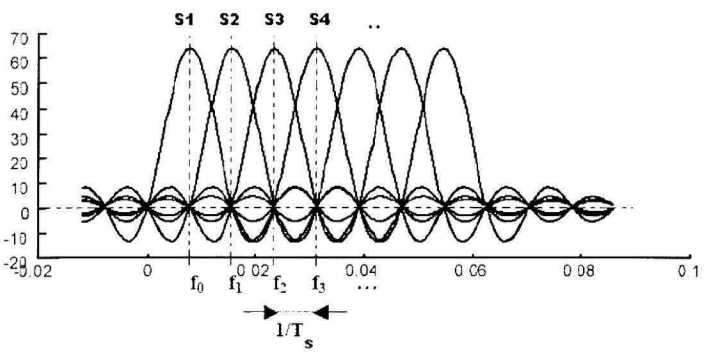


Figure 1.11 Représentation spectrale d'un symbole.

Ce type de modulation décompose un système large bande en plusieurs sous-systèmes à bande étroite. L'opération de *mapping* (Figure 12) consiste à moduler les sous-porteuses en amplitude et en phase. La superposition de toutes les sous-porteuses modulées, appelées

symboles OFDM est réalisée par l'inverse de la transformée rapide de Fourier (IFFT). Celle-ci transforme le signal du domaine fréquentiel au domaine temporel. Le temps de garde, est généralement une copie de la fin de la trame OFDM, (Figure 13). Cependant, l'ajout de l'intervalle de garde fait perdre l'orthogonalité entre les sous-porteuses, car le temps d'intégration de la FFT ne se fait plus dans un nombre entier de périodes. La solution employée pour y remédier est l'insertion du préfixe cyclique qui permet aussi de contrer l'interférence intersymbole (ISI). La durée d'un symbole OFDM détermine l'espacement entre chaque sous-porteuse et le temps disponible pour effectuer l'IFFT. Par ailleurs, le nombre de sous-porteuses détermine la taille de la FFT/IFFT (Yaghoobi, 2004). La FFT est l'une des applications étudiées dans ce travail. Elle sera présentée dans le prochain chapitre.

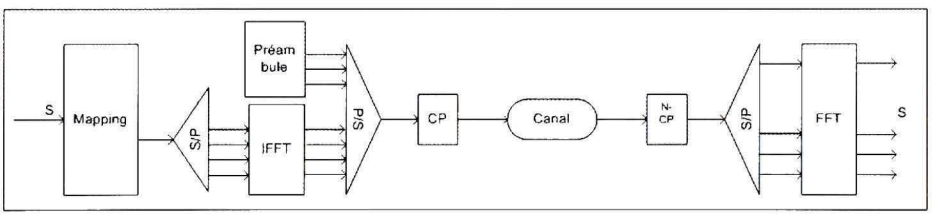


Figure 1.12 Principe de la modulation OFDM.

Avec T_g qui représente le temps de garde et T_b est la durée pendant laquelle est émise l'information d'un symbole OFDM.

Les deux composantes des systèmes OFDM, à savoir les symboles temporels OFDM et les sous-porteuses fréquentielles, peuvent être combinées en sous-canaux pour la gestion des usagers multiples. C'est l'OFDMA (Orthogonal Frequency Division Multiple Access).

L'OFDMA alloue de manière dynamique des groupes de sous-porteuses à différents utilisateurs pour autoriser une gestion plus souple du spectre de fréquences. Cette technique présente plusieurs avantages tels la réduction d'interférences et l'amélioration de la capa-

cité de transmission. Elle obéit au même principe que l'OFDM. Les sous-poteuses obtenues dans le domaine temporel sont de trois types tels que montrés à la Figure 14.

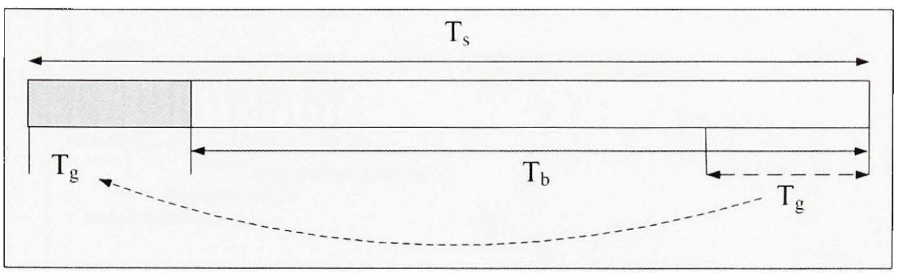


Figure 1.13 Structure d'un temps symbole OFDM.

Les sous-porteuses de données pour la transmission d'information, les sous-porteuses pilotes qui servent à la synchronisation et à l'estimation du canal et les sous-porteuses nulles utilisées comme des bandes de garde. En OFDMA, les sous-porteuses actives, qui peuvent être adjacentes, sont divisées en plusieurs sous-ensembles de sous-porteuses, chaque sous-ensemble se nomme un sous-canal.

1.3 Conclusion

L'enjeu des futurs systèmes de communication radio sera l'augmentation de l'efficacité spectrale. La solution-phare actuelle est le système MIMO. Il a été question de mettre en exergue cela à travers un bref survol sur les concepts de diversité et de capacité des systèmes MIMO. Le choix du type de modélisation du canal fait aussi partie intégrante de ces enjeux.

L'allocation de spectre est une denrée très prisée en communication malgré la rareté des ressources spectrales à cause de la diversité des applications existantes. Le recours aux techniques d'accès multiple permet d'optimiser le partage des ressources entre plusieurs utilisateurs. Parmi les techniques décrites, l'accès multiple à répartition par codes (CDMA) et le multiplexage par fréquences orthogonales (OFDMA) ont été privilégiés. Le

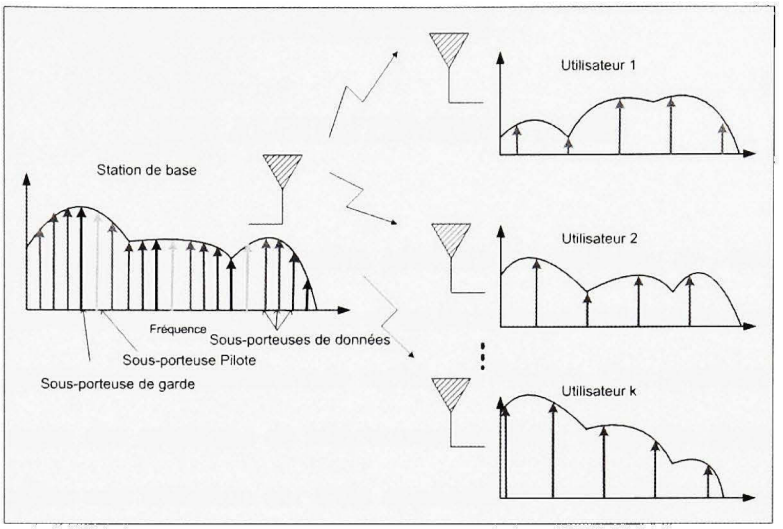


Figure 1.14 Principe de l'OFDMA.

CDMA est employé dans les systèmes UMTS et est basé sur l'étalement de spectre par séquence directe.

Après avoir présenté brièvement les systèmes MIMO et les liaisons point à multipoints, il est pertinent de parcourir de manière spécifique les algorithmes, qui tout en faisant partie des grandes catégories ci-dessus énoncées, ont fait l'objet de notre étude : le multiplexage spatio-temporel par couches verticales (VBLAST) et le récepteur Rake. Une troisième application, appartenant à la famille des algorithmes génériques et au coeur des systèmes OFDM, y sera présentée : la transformée rapide de Fourier (FFT). Ils sont l'objet du prochain chapitre.

CHAPITRE 2

LES ALGORITHMES ÉTUDIÉS

Le chapitre premier a introduit de manière générale les systèmes de communication étudiés, soit les systèmes à antennes multiples et les liaisons point à multipoints dont l'OFDMA, qui utilise l'algorithme de la transformée rapide de Fourier. Ces systèmes sont tous deux des fruits de l'essor des systèmes de télécommunications radio des dernières décennies. Ce chapitre focalise son attention sur trois applications particulières des grands groupes précédemment introduits. Ces applications, distinctes les unes des autres et sans aucun lien étroit, ont été choisies à cause de leur utilisation fréquente en télécommunications.

Le propos débute par la présentation des systèmes à codage spatio-temporel par couches (BLAST). Le BLAST est un algorithme de détection placé au récepteur des systèmes à antennes multiples, premier système introduit au chapitre précédent. Dans cette présentation, l'accent est mis sur ses différentes méthodes de détection employées dans le multiplexage spatial, avec emphase sur le détecteur à annulation successive d'interférences employant le forçage à zéro d'une part, et le critère de minimisation de l'erreur quadratique moyenne d'autre part. Après une comparaison de ces détecteurs, une justification du choix de l'architecture sélectionnée pour étude est illustrée. Celle-ci repose sur un compromis entre la complexité algorithmique et les performances désirées.

Le récepteur Rake est le deuxième algorithme étudié. Il fait référence aux systèmes se servant des liaisons points à multipoints de type CDMA. Une description non exhaustive des exigences du standard UMTS introduit cette seconde section, puisque le récepteur a été réalisé en respectant les critères dudit standard. À cela s'ajoute une description détaillée de cette forme d'algorithmes, très prisés dans un contexte de propagation multi-trajets.

La transformée rapide de Fourier (FFT) vient en troisième lieu. Elle est très demandée dans les systèmes OFDMA, précurseurs de la quatrième génération des réseaux de télé-

phonie mobile. Cette partie traite brièvement de l'évolution de la transformée de Fourier et s'attarde à décrire deux formes particulières de FFT : l'algorithme radix 2 à décimation temporelle et l'algorithme radix 2 à décimation fréquentielle.

2.1 Le codage spatio-temporel par couches, BLAST

Le codage spatio-temporel (STC) est une technique utilisée en communications sans fil pour transmettre des copies multiples de données à travers plusieurs antennes. Le codage est réalisé simultanément dans les domaines spatial et temporel dans le but d'introduire une corrélation entre les signaux émis par des antennes différentes à des intervalles de temps distincts (Vucetic et al., 2003). Cette corrélation spatio-temporelle est d'une grande utilité dans l'exploitation des évanouissements du canal MIMO afin de minimiser les erreurs de transmission au récepteur. Dans les systèmes multi-antennes non codés, ce type de codage permet d'obtenir un grand gain de puissance et la capacité d'atteindre la diversité en transmission sans pour autant sacrifier la bande passante. Les quatre catégories de codes spatio-temporels dénombrées (Vucetic et al., 2003) sont : les codes espace-temps en bloc, les codes espace-temps en treillis, les codes spatio-temporels turbo en treillis et, enfin, les codes espace-temps par couches, qui sont ceux qui focalisent notre propos. Le multiplexage spatial ou codage spatio-temporel par couches (en anglais layered space-time code) a été présenté pour la première fois par un chercheur, M.Foschini , des laboratoires de la firme de télécommunication Bell, aux États-Unis, dans l'une des publications de la compagnie-hôte (Foschini, 1996). La majeure partie des études traitant de ce sujet considère un système à un seul utilisateur disposant de N_t antennes émettrices. Ce contexte est également le nôtre.

Le principe du multiplexage spatial consiste à diviser une trame de données à l'entrée du transmetteur en plusieurs sous-trames. Par la suite, ces dernières seront acheminées au récepteur en utilisant des antennes différentes. Cette architecture favorise alors l'augmentation de la capacité du système. L'algorithme introduit par M. Foschini, le Bell La-

boratories Space-Time (BLAST) est une approche qui maximise l'utilisation de la bande passante d'un système de communication. Elle prend en effet avantage de la diversité spatiale acquise en transmettant et en détectant un nombre indépendant de données qui partagent un même canal sur plusieurs antennes. Le BLAST, comme c'est le cas des systèmes MIMO en général, exploite les effets multitrajets des canaux comme des alliés pour atteindre des hautes efficacités spectrales de l'ordre de 20 à 40 bps/Hz. Ces résultats ont été obtenus grâce à des tests de laboratoire de Bell effectués pour un système de huit antennes à l'émission et de douze antennes au récepteur (Golden et al., 1999). Les efficacités spectrales recueillies avoisinnent alors la limite de Shannon pour des canaux sans fil à évanouissements. Ceci est une avancée significative en comparaison avec les valeurs des capacités traditionnelles atteintes en communication sans fils. Ces dernières gravitent d'une part autour de 1 à 5 bps/Hz pour un cellulaire mobile et, d'autre part, autour de 8 à 10 bps/Hz pour des systèmes micro-ondes fixes employant une liaison point à point.

Kyrielle d'algorithmes se dénombrent dans une détection employant le multiplexage spatial : le multiplexage spatial diagonal, le Diagonal BLAST ou DBLAST (Foschini, 1996), le multiplexage spatial vertical, le Vertical BLAST ou VBLAST (Jiao, 2006) et le multiplexage spatial selon le principe turbo, le turbo BLAST ou TBLAST (Sellathurai M. et al., 2000) ... Les deux premiers algorithmes seront présentés en détail plus loin. En effet, le DBLAST est le premier algorithme à avoir vu le jour dans cette famille et le VBLAST est l'une des applications-cibles de ce travail. Ce chapitre introduit introduit aussi l'algorithme de décomposition QR, qui fait partie des algorithmes à multiplexage spatial vertical mais qui est préféré au VBLAST sous sa formulation première car plus facile d'implémentation. Les raisons de ce choix y seront mentionnées.

2.2 Diagonal Bell laboratories LAyered Space -Time, DBLAST

La première version des codes spatio-temporels par couches est appelée Diagonal BLAST (DBLAST) (Foschini, 1996). L'encodeur, illustré à la figure 1, se sert d'un arrangement

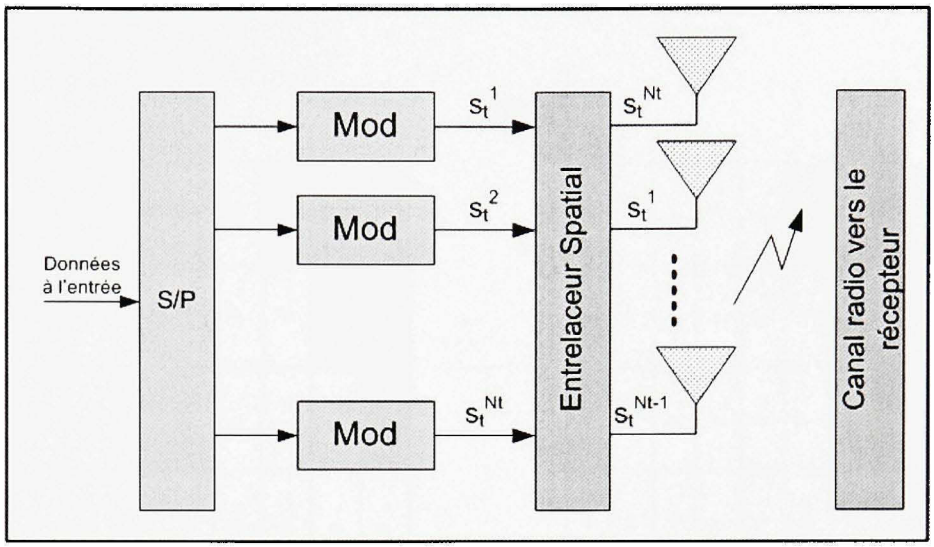


Figure 2.1 Architecture du DBLAST.

spatio-temporel qui correspond à un étalement diagonal des données au transmetteur. Au départ, la trame de bits émise provenant d'une source est démultiplexée en plusieurs sous-trames (décomposition série-parallèle) qui sont à leur tour encodées séparément, par un encodeur spatial. L'encodeur spatial est un entrelaceur qui dispose les symboles diagonalement à travers les antennes, dans le temps. La figure 2 en fait cas. Chaque couche peut avoir plus de symboles que le nombre d'antennes émettrices et la trame peut être très longue. La procédure d'encodage du DBLAST impose par elle-même une légère sous-utilisation du domaine espace-temps due à la présence des zéros (figure 2) dans la première couche à détecter. C'est la raison pour laquelle le DBLAST ne peut atteindre parfaitement la capacité de Shannon, puisque cette perte est engendrée chaque fois qu'un nouveau groupe de couches doit être transmis.

Cependant, le déploiement des symboles à travers les antennes lui confère l'avantage de posséder une diversité en transmission. Cette caractéristique élimine par le même fait la contrainte d'avoir un nombre d'antennes réceptrices au moins égal au nombre d'antennes émettrices qui apparaît lorsque la diversité en transmission est absente.

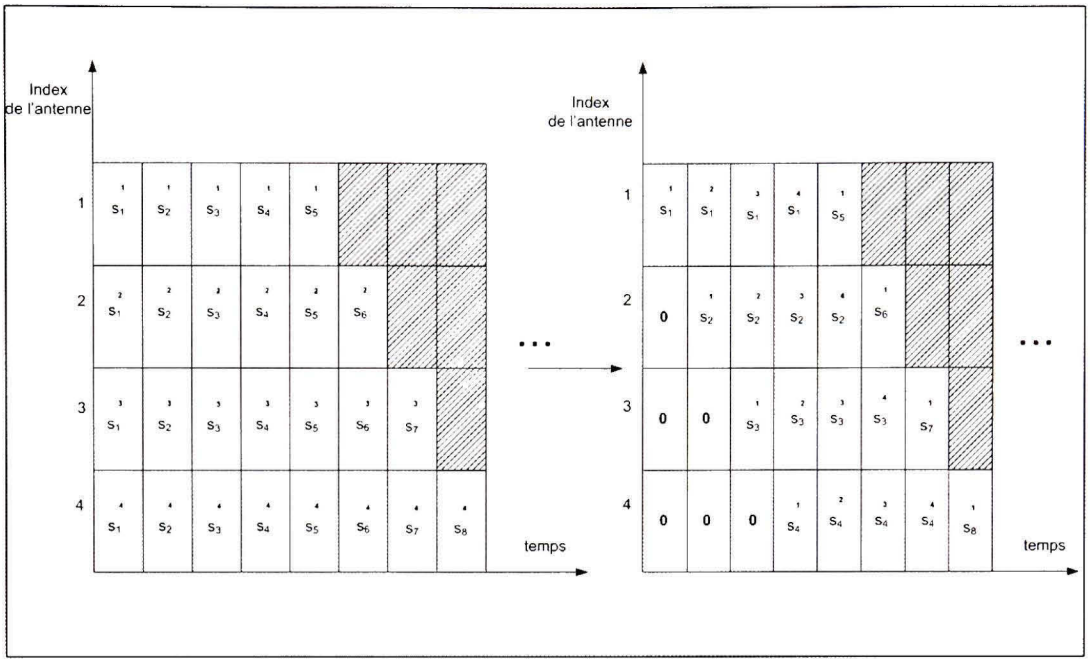


Figure 2.2 Codage diagonal par couches.

Le décodeur effectue le décodage couche après couche tel que présenté à la figure 3. C'est un processus itératif qui arrive à son terme au dernier index du réseau d'antennes disponibles. La détection se fait en considérant que les symboles qui étaient auparavant destinés à être transmis par une antenne donnée sont étalés diagonalement et transmis par différentes antennes. La figure 4 expose de manière détaillée les étapes du décodage. De la première à la quatrième étape, la détection se fait diagonalement en tenant compte chaque fois des signaux interférants. Le premier bloc n'a aucun interférant. Il possède alors une forte probabilité d'être détecté sans erreur car il est transmis seul. Plus on s'éloigne de lui, plus la détection est susceptible d'être erronée. Ensuite, les blocs suivants sont démodulés et détectés. Une fois tous les blocs de la première couche démodulés, la sous-trame associée à ladite couche est décodée. Le décodage devrait se faire sans faute sinon il y a propagation de l'erreur et toute la sous-trame est mal décodée. Pour amoindrir le nombre d'erreurs, la trame doit être longue et son encodage spatial robuste. Enfin, la couche dé-

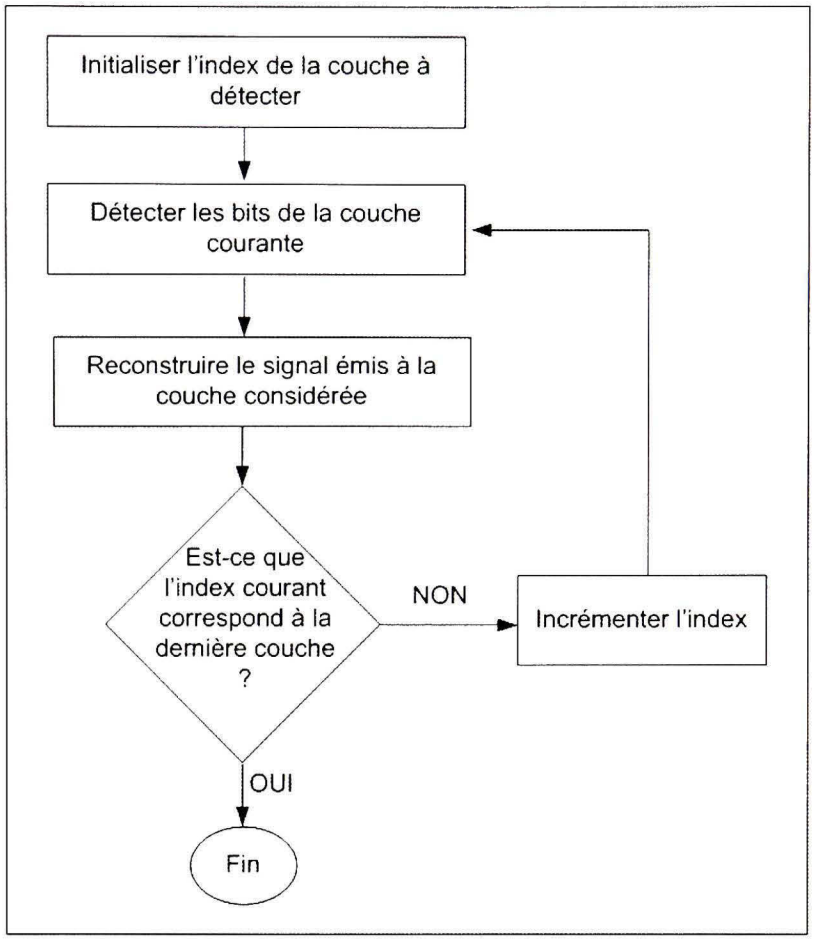


Figure 2.3 Vue temporelle du traitement successif des couches spatio-temporelles.

codée est soustraite du groupe global pour faire place à la suivante qui subira la même procédure.

Cette structure, bien que performante, demeure cependant complexe pour une implémentation en temps réel. C'est la raison pour laquelle d'autres types d'architectures moins exigentes pour les considérations pratiques et permettant un meilleur compromis entre une implémentation matérielle et des performances acceptables ont vu le jour. L'une d'elles est le multiplexage spatial par couches verticales.

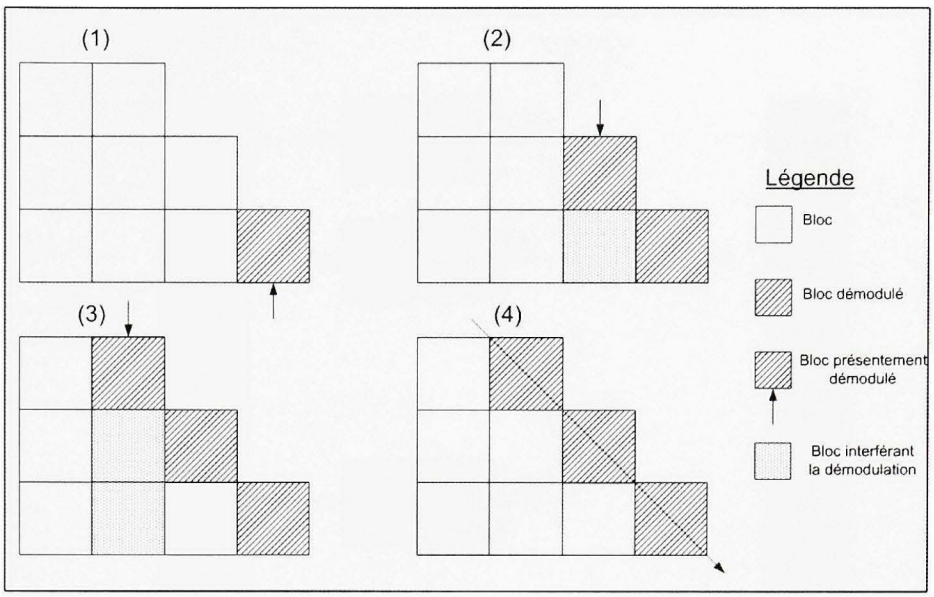


Figure 2.4 Principe du décodage par DBLAST.

2.3 Vertical Bell laboratories LAyered Space - Time, VBLAST

Comme dans le cas du DBLAST, la trame de bits est séparée en sous-trames. C'est de sa structure verticale de transmission des données dont dérive le VBLAST car tous les symboles d'une sous-trame donnée sont transmis par la même antenne (chaque sous-trame est affectée à une antenne). Ceci élimine les pertes spatio-temporelles engendrées dans le DBLAST, au détriment de la diversité en transmission. Celle-ci est perdue par le fait même que chaque sous-trame soit reliée à une antenne précise puisque le VBLAST ne possède pas d'entrelaceur spatial pour renforcer la robustesse des données avant leur transmission en environnement multitrajet (figure 5).

Pour N_t antennes à l'émetteur et N_r antennes au récepteur, le gain de diversité est de l'ordre de : $N_r + N_t - 1$ (Vucetic et al., 2003). Conventionnellement le nombre d'antennes en réception dans ce type de système doit être minimalement égal au nombre d'antennes en émission. Au décodeur, les symboles reçus sont démodulés. Il existe plusieurs algorithmes de détection pour cette architecture.

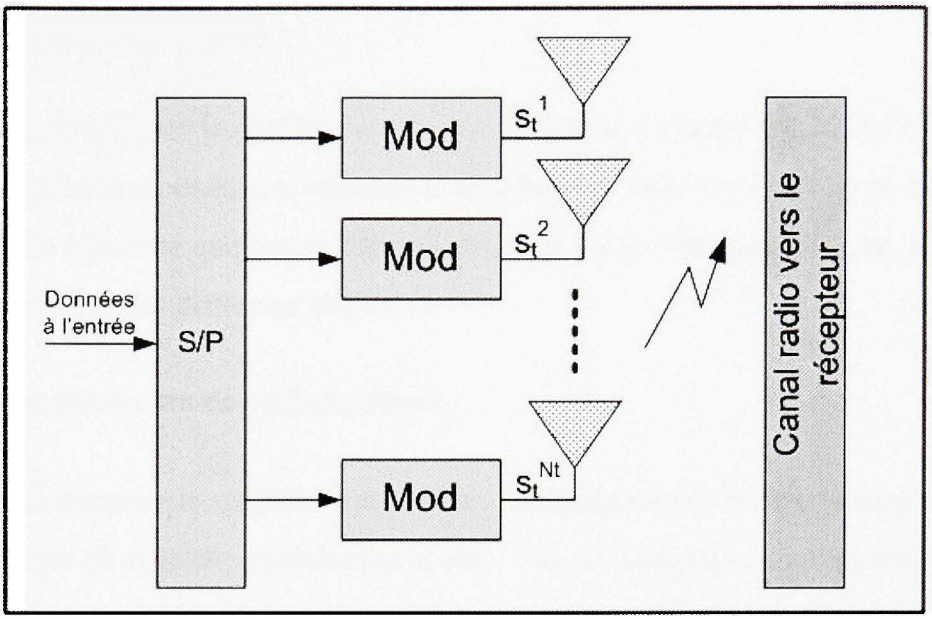


Figure 2.5 Architecture du VBLAST.

2.3.1 Les méthodes de détection

Plusieurs méthodes de détection sont employées par le VBLAST (Jiao, 2006) pour estimer le signal transmis et quelques unes sont présentées ci-dessous. Nous considérerons un système MIMO à N_t entrées et N_r sorties. Le vecteur colonne d'observations sera noté $r = (r_j),\ 1 \leq j \leq N_r$, le vecteur colonne émis, $s = (s_i),\ 1 \leq i \leq N_t$, la matrice d'atténuation du canal, $H = (H_{ij}),\ 1 \leq i \leq N_r; 1 \leq j \leq N_t$ et $N = (N_j),\ 1 \leq j \leq N_r$, le bruit blanc complexe additif de variance σ^2 et de moyenne nulle. Les hypothèses de travail sont les suivantes :

- $r = Hs + n$,
- la réalisation du canal H est connue,
- $\tilde{s}$ est le vecteur des symboles estimés,
- $\rho = \frac{P}{\sigma^2}$ est le rapport signal à bruit moyen par antenne de réception.

2.3.1.1 Les filtres adaptés

Cette méthode effectue le produit du signal reçu avec la transposée-conjuguée du coefficient de canal correspondant et compare le résultat avec tous les symboles possibles. Cependant, elle nécessite que les coefficients de canal soient orthogonaux pour éviter toute interférence issue des différents chemins.

2.3.1.2 Le maximum de vraisemblance

Ce détecteur compare le symbole reçu à toutes les combinaisons de symboles possibles en tenant compte du type de constellation choisie. Ses performances sont les meilleures en détection. Cependant, ce récepteur possède aussi la plus grande complexité de calculs et celle-ci croît exponentiellement avec le nombre de points de la constellation considérés. En supposant un bruit gaussien et un état de canal parfaitement connu au récepteur, le détecteur à maximum de vraisemblance estime les symboles émis grâce à la formule 2.1 :

$$\hat{s} = \underset{s}{argmin} \|r - Hs\|^2 \tag{2.1}$$

où $argmin \| \quad \|^2$ représente le calcul du minimum du carré de la norme de chaque ligne des matrices considérées.

La recherche de la distance minimale s'effectue sur tous les symboles possibles de la constellation.

2.3.1.3 Le forçage à zéro

Ce type de récepteur linéaire simple cherche à annuler les contributions des autres émetteurs. Pour cela, il faut que la matrice H des coefficients du canal soit carrée et inversible. L'estimation du signal originellement transmis conduit à effectuer le produit de l'inverse de la matrice de canal avec le signal reçu. Si H est reversible, le vecteur de symboles

estimés est alors donné par la relation 2.2 :

$$\tilde{s} = H^{-1} r \tag{2.2}$$

Le récepteur ZF fournit des résultats corrects lors de l'estimation des symboles s_i transmis dans le même canal pour des rapports signal-à-bruit élevés. Lorsque ces derniers sont faibles, la contribution du bruit dans l'estimation des symboles perturbe fortement le récepteur. Par ailleurs, en pratique, H n'étant pas nécessairement carrée c'est-à-dire que $N_t \neq N_r$, il devient pertinent d'utiliser la matrice pseudo-inverse de la matrice de canal $H^{\dagger}$ (Moore, 1920 ; Penrose, 1955 et Golub et al., 1996) pour estimer les symboles. Dans ce cas, le vecteur estimé s'écrit selon l'équation 2.3 (Golub et al., 1996) :

$$\tilde{s} = H^{\dagger} r = \left(H^{*} H\right)^{-1} H^{*} r \tag{2.3}$$

où :

- La matrice de canal H possède des éléments indépendants et identiquement distribués,
- $H^{\dagger} = \left(H^{*} H\right)^{-1} H$ est la matrice pseudo-inverse encore dite de Moore-Penrose de la matrice de canal H,
- H^{*} est la matrice complexe conjuguée et transposée de H.

L'inconvénient majeur de ce critère est l'augmentation du niveau de bruit à cause de l'inversion de la matrice de canal qui fait fi de la variance du bruit. Le critère qui minimise l'erreur quadratique moyenne permet de pallier à ce défaut.

2.3.1.4 La minimisation de l'erreur quadratique moyenne

La minimisation de l'erreur quadratique moyenne (en anglais Minimum Mean Square Error (MMSE)) amenuise l'erreur globale due à la contribution du bruit et à l'interférence mutuelle des signaux. Ce récepteur résiste donc mieux au bruit que le détecteur par forçage

à zéro. L'expression des symboles estimés est donnée par l'équation 2.4 :

$$\tilde{s} = (HH^* + \frac{N_t}{\rho} I_{N_t})^{-1} H^*.r \tag{2.4}$$

où I_{N_t} est la matrice identité d'ordre N_t et rappelons-le ρ est le rapport signal-à-bruit moyen par antenne de réception.

Il est possible de noter que le récepteur MMSE tend vers le récepteur ZF à haut SNR ou de façon équivalente lorsque σ^2 tend vers 0.

2.3.1.5 Le détecteur par annulation successive d'interférence

La méthode des filtres adaptés impose la contrainte de l'orthogonalité des coefficents de la matrice de canal. Le maximum de vraisemblance est très performant mais nécessite un nombre de tests exorbitant. Le critère de forçage à zéro n'est pas toujours satisfaisant dans la mesure où il introduit une amplification de la contribution du bruit sur certaines voies lorsqu'on multiplie le signal reçu par la matrice de canal pseudo-inverse. Le SNR s'en trouve alors dégradé. Bien que la méthode de détection directe utilisant le critère de minimisation de l'erreur quadratique moyenne soit moins sensible au bruit, son principal inconvénient est qu'elle ne parvient pas à enlever toute l'interférence entre symboles. Le recours à l'annulation successive des interférences permet de pallier à ces désavantages. Son principe consiste à détecter d'abord les voies qui réunissent les conditions les plus favorables, à soustraire leur interférence puis à répéter la procédure jusqu'aux signaux les plus perturbés. La structure proposée par Foschini (Foschini, 1996 ; Golden et al., 1999), le V-BLAST, emploie la structure à annulation successive d'interférences basée sur une égalisation soit par forçage à zéro ou par minimisation de l'erreur quadratique moyenne. Cet algorithme augmente les performances de l'estimateur au détriment d'une complexité accrue au récepteur.

Algorithme 1 : ZF-VBLAST

1: $i = 1$
2: $G_1 = \tilde{H}^{\dagger}$
3: $k_1 = \underset{j}{argmin} \left\| (G_1)_j \right\|^2$
4: $w_{k_i} = (G_i)_{k_i}$
5: $y_{k_i} = w_{k_i}^T r_i$
6: $\tilde{s}_{k_i} = Q(y_{k_i})$
7: $r_{i+1} = r_i - \tilde{s}_{k_i} \left(\tilde{H} \right)_{k_i}$
8: $G_{i+1} = \left(\tilde{H}_{k_i}^{-} \right)^{\dagger}$
9: $k_{i+1} = \underset{j \notin (k_1 \ldots k_i)}{argmin} \left\| (G_{i+1})_j \right\|^2$
10: $i = i + 1$

Le multiplexage spatial par couches verticales cherche à obtenir pour chaque voie, le meilleur SNR. Pour ce faire, il faut commencer l'estimation par la voie de données ayant la plus forte puissance. C'est la technique d'ordonnancement. Des voies transmises, celle qui possède le plus fort SNR doit être détecté en premier. Cette voie correspond à celle qui minimise la matrice pseudo-inverse G_i . Elle est ensuite suivie par la détection de la voie possédant le plus fort SNR parmi celles qui restent à détecter après l'annulation de son interférence au sein des voies restantes. La procédure se poursuit jusqu'à ce que toutes les voies soient détectées. Les étapes de détection sont résumées à travers l'Algorithme 1.

Dans les étapes ci-dessus énumérées, les notations utilisées sont :

- $(G_i)_j$ est la j^{me} ligne de G_i et k_i désigne l'ordre de détection des symboles,
- Q symbolise le processus de quantification par lequel l'estimation des symboles émis est faite selon le type de modulation choisi,
- $(H_{k_i}^{-})\dagger$ indique l'annulation de la contribution des k_i premiers émetteurs. Les colonnes de H sont alors remplacées par des zéros.

2.3.2 Quelques considérations pratiques pour une implémentation du VBLAST

Implanter l'algorithme V-BLAST en pratique, sous forme matérielle ou sous forme logicielle, n'est pas besogne de tout repos. Plusieurs interrogations surgissent à cet effet dont quelques unes sont énumérées ci-dessous :

- Quelle est la meilleure méthode pour implémenter la pseudo-inverse de la matrice de canal tout en conservant la stabilité numérique de l'algorithme ainsi qu'une rapidité acceptable ?
- Quel est l'ordre de grandeur de la complexité algorithmique du V-BLAST ?
- Quels débits peut-on atteindre avec les processeurs de traitement de signal modernes ?
- Quelles sont les opérations les plus gourmandes en ressources matérielles ?
- La quantité mémoire du processeur choisi sera-t-elle suffisante pour répondre aux exigences requises par celle du V-BLAST ?
- Comment procéder pour limiter la dégradation des performances en terme de taux d'erreur binaire ?

Plusieurs de ces questions ont en commun la volonté de maintenir un bon compromis entre la performance de la transmission, le débit des données et la complexité des calculs. Une telle analyse est d'une importance fondamentale puisqu'elle fournira des critères de base préalables à l'implémentation du récepteur et permettra de savoir si les contraintes économiques très souvent sévères, peuvent être respectées. Plus souvent qu'autrement, le goulot d'étranglement de l'implémentation se circonscrit à la réduction de la complexité algorithmique, soit au nombre total d'opérations qu'un algorithme doit réaliser avant de conclure sa tâche. L'une des opérations coûteuses en ressources de calcul est l'inversion de matrice. En effet, l'opération la plus complexe du VBLAST est le calcul de la pseudo-inverse de Moore-Penrose de la matrice de canal. Il existe plusieurs méthodes pour effectuer ce type

d'opérations au nombre desquelles se trouvent selon (Golub et al., 1996) : la méthode itérative par calcul de la conjuguée du gradient, la méthode de Gauss-Seidel, la méthode de Schultz-Hotelling, la décomposition par valeurs singulières, la décomposition QR ... La décomposition de la matrice de canal par la méthode QR avec ordonnancement a été choisie car elle est répandue dans le contexte de cette application (Wübben , 2001) d'autant plus qu'elle offre la double possibilité d'amenuiser la complexité des calculs tout en donnant des performances relativement semblables à celles de l'algorithme VBLAST traditionnel.

2.3.3 Algorithme Sorted QR Decomposition avec détection selon le critère de forçage à zéro, ZF-SQRD

Dans l'Algorithme 2, les lignes 1 à 11, tirées de (Wübben , 2001) exposent la décomposition QR ordonnancée. Les lignes subséquentes (12 à 22) complètent le processus de détection du signal de départ. En effet, la matrice de canal H est décomposée en un produit de deux matrices en utilisant l'orthogonalisation de Gram-Schmidt : d'une part, R, la matrice triangulaire supérieure de taille (N_t, N_t) et, d'autre part, Q la matrice unitaire de taille (N_r, N_t) . k_i détermine l'ordre de détection des symboles ; p_{est}, le vecteur de permutation défini par $(1, N_t)$; P est la matrice identité de taille $(N_t,\ N_t)$; $Q\{.\}$ est le processus de quantification à partir duquel le détecteur estime les symboles reçus ; ch_{out} est le signal bruité à la sortie du canal à évanouissement.

L'algorithme énoncé à travers l'Algorithme 2 illustre deux particularités du ZF-SQRD : l'ordonnancement dans la décomposition QR et le principe de l'annulation successive des interférences. Le processus de décomposition avec ordonnancement (étapes 2 à 11) consiste à transformer graduellement la matrice de canal H en un produit de deux matrices : Q et R. Les termes de ces dernières sont déterminées respectivement colonne après colonne pour Q et ligne après ligne pour R. L'ordre de la décomposition est dicté par la détermination de la colonne de Q (initialisée au départ comme étant la matrice de

Algorithme 2 : ZF-SQRD

1: $R = 0,\ Q = H,\ p = 1, ..., N_t,\ P = I$
2: **for** $i = 1 : N_t$ **do**
3: $k_i = argmin_{j=1,...,Nt}|Q_j|^2$
4: permuter les colonnes i et k_i dans Q, R et p
5: $R_{ii} = |Q_i|$
6: $Q_i = Q_i / r_{ii}$
7: **for** $j = i + 1 : N_t$ **do**
8: $r_{i,j} = Q_i^H * Q_j$
9: $Q_j = Q_j - R_{i,j} * Q_i$
10: **end for**
11: **end for**
12: $r = Q' * ch_{out}$
13: $r(N_t, :) = r(N_t, :)/R(N_t, N_t)$
14: $Q_{N_t}\{.\}$
15: **for** $i = N_t - 1 : -1 : 1$ **do**
16: **for** $j = i + 1 : N_t$ **do**
17: $x(i, :) = x(i, :) + R(i, j) * Q_j\{.\}$
18: **end for**
19: $r(i, :) = \frac{(r(i,:)-x(i,:))}{R(i,i)}$
20: $Q_i\{.\}$
21: **end for**
22: $Q\{.\} = P * Q\{.\}$

canal) dont la norme est la plus faible à l'étape 3. Cela équivaut alors à signifier laquelle des antennes a transmis le signal avec le plus fort rapport signal à bruit. Ensuite, l'estimation du signal transmis est exécutée entre les étapes 12 et 22. L'annulation successive des interférences, semblable à celle décrite à la section 2.3.1.5 est illustrée à l'étape 19 par la soustraction du signal le moins affecté par le bruit du reste des données à détecter.

2.3.4 Comparaison des détecteurs

Compte tenu de la complexité importante de la structure à annulation successive d'interférences basée sur le critère de minimisation de l'erreur quadratique moyenne, nous ne considérerons que la décomposition QR qui utilise le critère de forçage à zéro pour l'implémentation. D'une part, nous allons comparer cette structure à l'architecture ZF-VBLAST tel que proposé par Foschini ainsi qu'à l'architecture MMSE-VBLAST. D'autre

part, nous comparerons la même structure avec l'architecture SQRD employant le critère MMSE pour différents cas.

Dans toutes nos simulations, nous transmettons des blocs de taille $(N_t, 122)$ modulés par quadrature de phase (QPSK) en les acheminant au récepteur au moyen d'un canal de Rayleigh non sélectif en fréquence (flat fading). Nous considérons par ailleurs que ce canal a une fréquence Doppler fixée à $60\ Hz$, une fréquence d'observation de ses coefficients à $1\ kHz$ et que la fréquence d'observation du signal est de $2\ MHz$.

2.3.5 Complexité qualitative

La décomposition de la matrice H (le calcul de la pseudo-inverse de H) en un produit de matrices Q et R domine la complexité du SQRD et du VBLAST. Le nombre d'émetteurs et de récepteurs limite donc l'utilisation de cette structure. Le type de modulation a peu de poids sur la complexité de cette dernière et est considérée comme fixe dans nos résultats.

2.3.6 Comparaison ZF-SQRD, ZF-VBLAST, MMSE-SQRD et MMSE-VBLAST

Dans cette section, les performances entre le VBLAST et le SQRD sont comparées. Les performances classiques d'un système mono-émetteur (Single Input) mono-récepteur (Single Output) dit SISO $((1,1))$ ont été également reportées. De plus, il faut noter que l'augmentation du nombre d'antennes émettrices, entraîne une augmentation proportionnelle à l'efficacité spectrale.

La figure 6 considère les cas (1,1),(2,2) et (2,4). Il faut noter que sur cette figure les courbes (2,4) MMSE-VBLAST et (2,4) MMSE-SQRD sont confondues. Il y apparaît que les performances sont relativement comparables. C'est la raison pour laquelle les études futures se focaliseront autour de l'algorithme SQRD qui est moins complexe à implémenter.

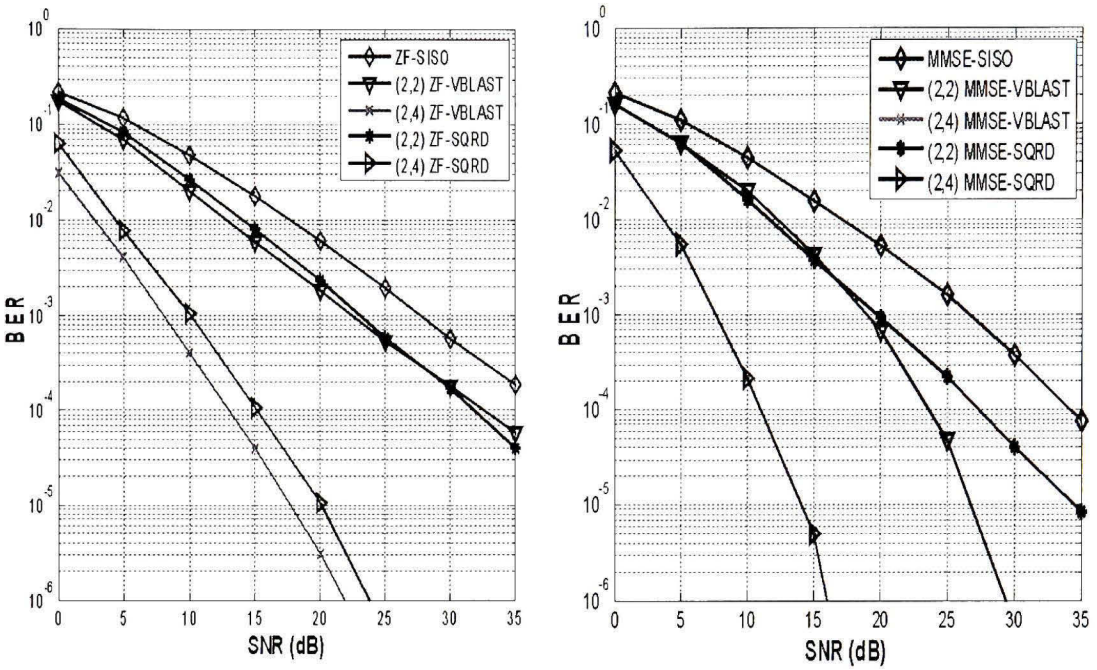


Figure 2.6 Performances du VBLAST et du SQRD par forçage à zéro et par minimisation de l'erreur quadratique moyenne.

Par ailleurs, la figure 7 illustre les taux d'erreurs escomptés par l'algorithme SQRD selon que le critère d'égalisation soit le forçage à zéro ou la minimisation de l'erreur quadratique moyenne. On peut constater que le second critère d'égalisation affiche des performances nettement supérieures à celles du premier, de l'ordre de 5 dB. Malgré cet avantage que présente l'égaliseur MMSE, le choix du forçage à zéro demeure vainqueur car il nécessite moins d'efforts lors des considérations pratiques.

L'influence du nombre d'émetteurs et de récepteurs est non négligeable pour obtenir des bonnes performances. Les courbes obtenues à la Figure 8 montrent l'amélioration introduite par l'accroissement du nombre de récepteurs en terme de SNR. En effet, rajouter des antennes en réception permet d'accroître la puissance utile reçue, donc la capacité à effectuer une bonne détection. Ceci est observé lorsque l'on passe d'un système (2,2) à

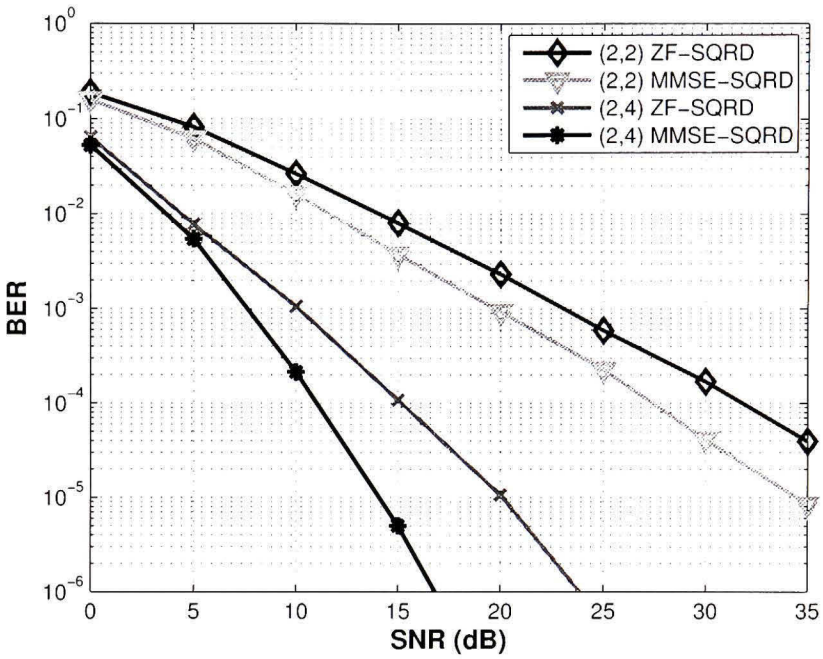


Figure 2.7 Performances du SQRD par forçage à zéro et par minimisation de l'erreur quadratique moyenne.

un système (2,4) où le gain avoisinne 10 dB pour un SNR plus petit que 15 dB et croît pour des valeurs plus grandes. Par contre l'augmentation du nombre d'émetteurs pour un nombre de récepteurs constant, dégrade le taux d'erreurs. C'est le cas lors du passage d'un système (2,4) à (4,4) où les pertes sont environ de 10 dB en dessous de 25 dB de SNR. Ces remarques permettent de vérifier l'équation de la diversité théorique du SQRD, soit $N_r - N_t + 1$.

Une fois terminée la description de la première application, les paragraphes subséquents, s'attèlent à discourir sur la seconde application indépendante de la première, soit le récepteur Rake.

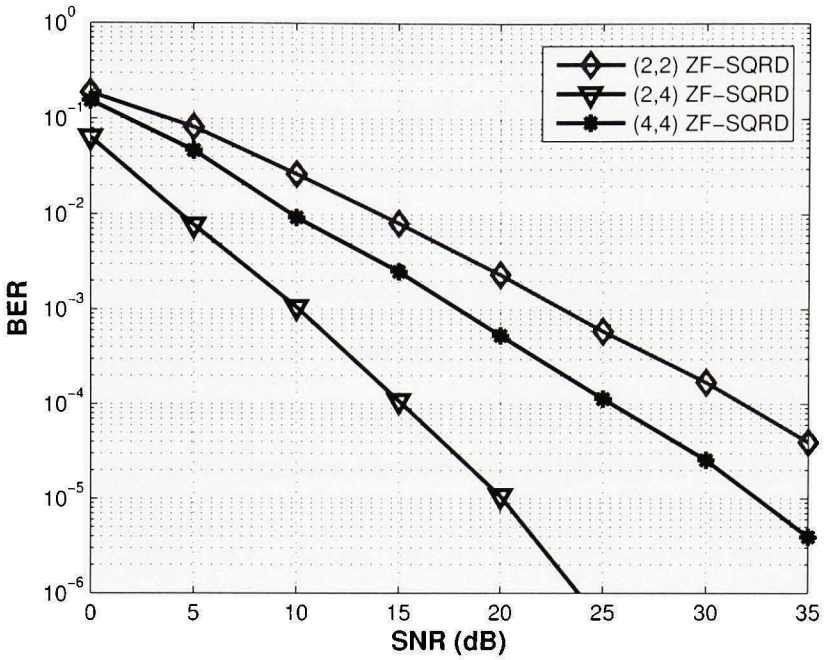


Figure 2.8 Influence du nombre d'émetteurs et de récepteurs sur les performances du SQRD.

2.4 Le récepteur Rake

Le principe du récepteur Rake a été présenté pour la première fois en 1958 par les chercheurs Price et Green pour lutter efficacement contre la combinaison des trajets multiples aléatoires avec les effets du bruit additif inévitables dans toute propagation et qui rendent difficile la détection des signaux émis(Price R. et al., 1958). L'implémentation du récepteur dans le cadre de ce mémoire a été réalisée en conformité avec le standard UMTS en considérant une liaison montante.

Ce qui suit débutera par une présentation générale des systèmes UMTS. La description des exigences du cahier de charges et des paramètres de l'interface radio prendra le relais afin de mieux cerner les caractéristiques de la couche primaire dite physique . Les couches supérieures de l'interface radio recevront les données de cette dernière via des canaux,

soit de transport ou physiques. Leur constitution se verra étalée à ce moment. Revenant à la couche physique, centre des intérêts dans le contexte qui est nôtre, seront tour à tour développés le principe de l'étalement spectral et de l'embrouillage des codes qui sont des moyens efficaces pour lutter contre le bruit de transmission et les interférences entre les usagers. Par la suite, la modulation numérique est introduite sous sa forme QPSK. Elle permet de véhiculer de l'information binaire d'un signal de référence selon la phase dans ce cas. Par ailleurs, le contrôle de puissance est un atout pour combattre les variations de la puissance du signal transmis d'une part. D'autre part, le transfert de la connexion d'une cellule à l'autre dit « handover »améliore la qualité de la communication. D'où la nécessité d'en toucher mot. Enfin, une description détaillée du récepteur Rake bouclera cette section.

2.4.1 Brève présentation des systèmes UMTS

L'une des approches les plus prometteuses de la troisième génération des systèmes de télécommunication est la combinaison du CDMA à large bande (WCDMA) avec le réseau fixe GSM. À cet effet, plusieurs propositions ont vu le jour au sein de l'Union Internationale de Télécommunication (ITU) et de l'initiative qu'elle a suscitée à l'année 2000 nommée Télécommunications Mobiles Internationales (IMT2000). Parmi les organisations qui se sont réunies pour définir un système mobile unique, peuvent être citées : Japan's Association of Radio Industry and Business (ARIB), Alliance for Telecommunications Industry Solutions (ATIS), European Telecommunications Standards Institute (ETSI) à travers Special Mobile Group (SMG), Telecommunications Technology Committee (TTC) ... Toutes ces institutions se sont penchées sur la possibilité de migrer progressivement vers les techniques radio WCDMA sans ignorer les nombreux avantages des réseaux GSM déjà existants. Le standard qui en est émergé est la norme ETSI basée sur le système universel en télécommunication mobile (en anglais Universal Mobile Telecommunication System (UMTS)) communément appelée UMTS Terrestrial Radio Access (UTRA). L'accès au standard UTRA s'effectue par Direct Sequence Code Division Multiple Access (DS-

CDMA). L'information est étalée sur une bande large de 5 MHz ; d'où le nom CDMA à large bande (WCDMA). Le WCDMA peut opérer en deux modes : le duplexage en fréquence (en anglais Frequency Division Duplex (FDD)) ou le duplexage en temps (en anglais Time Division Duplex (TDD)), décrits à la section 1.2.

L'information véhiculée par les systèmes UMTS sont de nature variée : son, données, images fixes ou animées. Les appareils vecteurs de cette information sont multitâches. Un téléphone mobile peut servir à la fois de visiophone, d'avertisseur pour la gestion d'un agenda ou d'une messagerie. Par ailleurs, l'utilisateur des systèmes de troisième génération bénéficie de plusieurs facilités, entre autres de la possibilité de filtrer des appels et de gérer ses communications dans le temps. Ceci nécessite une définition claire du cahier des charges et des contraintes de l'interface radio, essentielles pour toute implémentation en conformité avec le standard UMTS.

2.4.1.1 Les exigences de l'interface radio

L'interface radio répond à des critères exigeants en terme de qualité de service. En effet celle-ci doit supporter des débits élevés pouvant aller jusqu'à 2 Mbits/s ; ce qui est largement supérieur à celui d'un système GSM par exemple qui est de l'ordre de quelques kbps. Les services offerts doivent pouvoir être exécutés avec un débit maximal imposé par la vitesse de déplacement du mobile ainsi que son environnement. Ils doivent aussi être flexibles en termes de variabilité de débit, d'asymétrie entre la liaison montante et celle descendante. En outre, l'utilisateur doit pouvoir accéder à des applications multiples. L'UMTS subdivise les services disponibles en quatre classes. la première est le type conversationel qui se caractérise par la transmission de la voix et des images. Ce type de service n'accuse pas de variation de délai et dispose d'un temps d'exécution lent. Ensuite, vient la transmission par la vidéo qui entraîne un délai important, de l'ordre des secondes. Une troisième application est la communication interactive, par l'internet qui nécessite une liaison entre un ordinateur et un serveur web. Les usagers de ce type d'applications

s'attendent à des faibles délais de traitement mais toute variation de délai est critique. Enfin, les applications de second plan telles le courrier électronique ne se préoccupent pas du délai mais requièrent un faible taux d'erreurs.

La pluralité des prestations offertes et la couverture globale font d'un réseau cellulaire, un réseau de type multi-couches. Les macro-cellules (0.5 à 10 km de rayon) sont caractérisées par une couverture en environnement extérieur rural à grande mobilité, de l'ordre de 500 km/h et par un débit maximal de 144 kbps. Les micro-cellules (50 à 500 m de rayon) quant à elles disposent d'une forte densité de trafic, sont à mobilité réduite (au maximum 120 km/h) avec un débit maximal de 384 kbps. Enfin les pico-cellules (5 à 50 m de rayon) sont employées pour une couverture à l'intérieur des bâtiments, possèdent une mobilité réduite (inférieure à 10 km/h) et un débit maximal de 2 Mbps. Un changement de cellules transparent permet d'éviter les pertes de communication et l'utilisation de spectre de fréquences s'en trouve maximisée.

2.4.1.2 Le multiplexage et le spectre fréquentiel

Les bandes de fréquences utilisées dans le système UMTS sont représentées à la Figure 9 inspiré des données fournies par (ETSI, 2004). On peut y distinguer deux bandes appairées (liaisons montante-descendante) de 2×60 MHz qui utilisent le duplexage en fréquences (FDD) et deux bandes non appairées (liaisons montante-descendante) de 35 MHz rattachées au duplexage temporel (TDD), toutes attribuées aux applications terrestres. En outre, deux bandes de 2×30 MHz sont allouées aux applications satellites.

L'accès multiple W-CDMA fonctionne en mode FDD tandis que l'accès multiple TD-CDMA est actif en mode TDD. La principale différence entre les deux modes est leur souplesse face à l'asymétrie des liaisons. En effet, le mode FDD est adapté à plusieurs cellules mais n'est pas flexible aux liaisons aymétriques. Le mode TDD par contre, tient compte de l'asymétrie des échanges entre les liaisons montante et descendante pour adapter les débits de transmission tout en exigeant en contrepartie la synchronisation des sta-

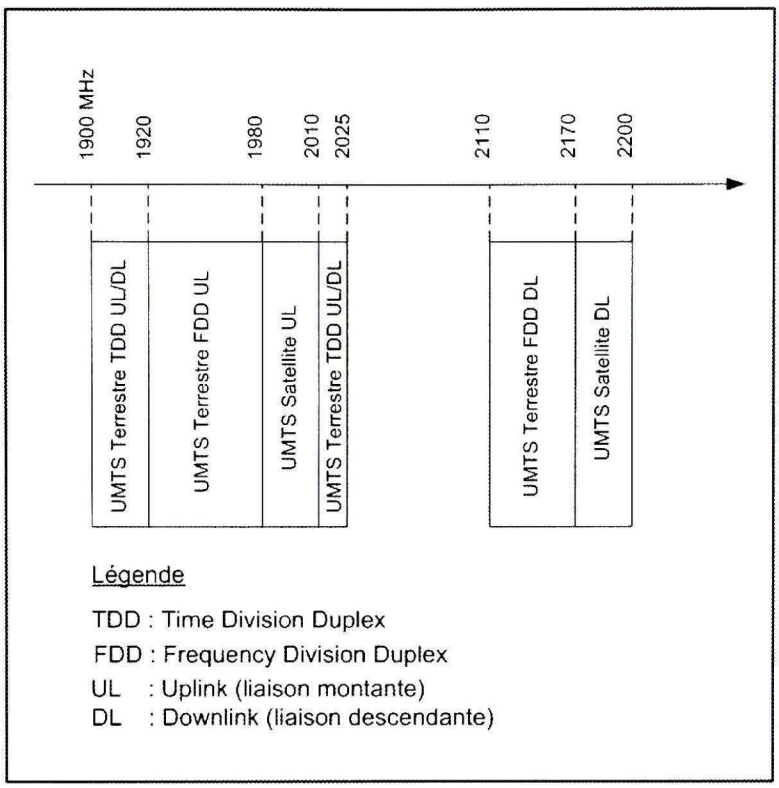


Figure 2.9 Bandes de fréquences UMTS.

tions de base. La solution W-CDMA en mode FDD est celle choisie dans notre étude et s'applique bien aux macro et aux micro cellules.

2.4.1.3 Les paramètres de l'interface radio terrestre du standard UMTS

La couche physique, la couche de contrôle d'accès du médium (en anglais medium access control (MAC)) et la couche de contrôle de ressources radio (en anglais radio resource control (RRC)) sont les principales couches de l'interface radio. La Figure 10 présente son architecture.

Également, sur cette figure, on aperçoit le canal de transport qui s'intéresse au mode de transfert des informations sur l'interface radio et le canal logique caractérisé par le type d'informations qui lui sont transférées. La couche physique est garante des services de

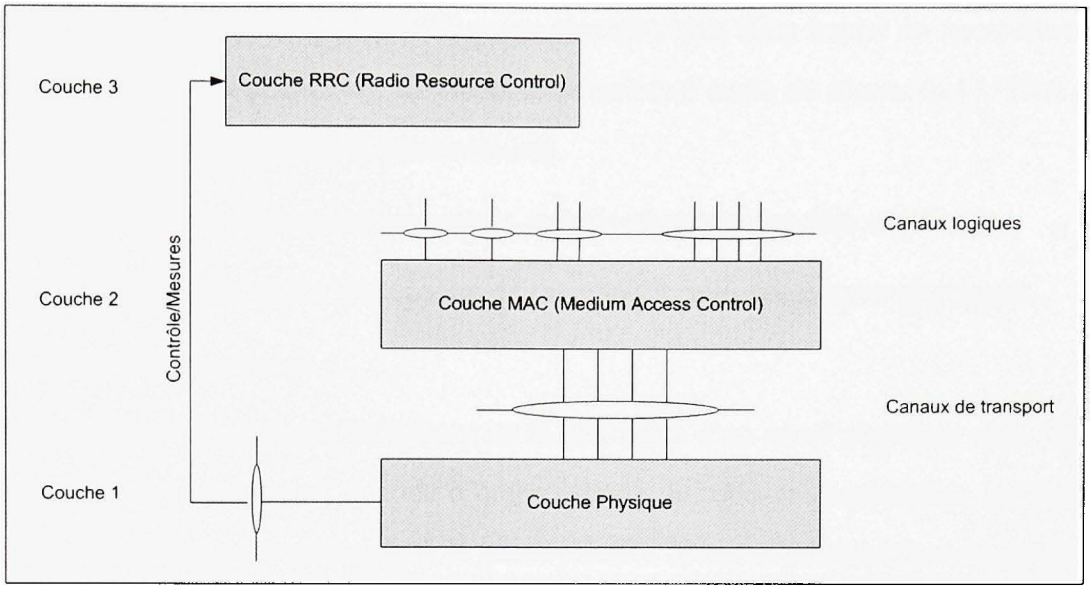


Figure 2.10 Architecture de l'interface radio.

transport de données aux couches supérieures. C'est aussi à ce niveau que l'implémentation du récepteur Rake se situe, d'où notre intérêt à son sujet. Elle s'assure de la préparation convenable des données à transmettre à travers les opérations de codage et de modulation.

Dans le cas particulier du CDMA, la couche physique s'attèle à effectuer les tâches d'étalement et d'embrouillage des données. Les fonctions qu'elle assure sont les suivantes (Krüger et al., 2004) :

- Synchronisation fréquentielle et temporelle.
- Contrôle de synchronisation.
- Mesures des caractéristiques radio.
- Pondération des puissances et combinaison des canaux physiques.
- Modulation et étalement/désétalement et démodulation des canaux physiques.
- Codage et décodage correcteur d'erreurs des canaux de transport.

- Combinaison et distribution en macrodiversité (état dans lequel un équipement du mobile est en liaison avec au moins deux points d'accès du réseau de l'UTRA pour améliorer la qualité de la transmission).
- Détection des erreurs sur les canaux et indication aux couches supérieures.
- Adaptation des débits des canaux de transport à celui des canaux physiques .
- Formation des voies...

Les éléments qui font partie intégrante de la définition d'un canal physique sont : la fréquence porteuse spécifique, le code d'embrouillage, le code de canalisation, les instants de début et de fin de transmission et la phase relative (0 ou $\pi/2$) dans le cas d'une liaison montante. Selon les normes du standard UMTS, une trame radio dure 10 ms et se compose de 15 fenêtres temporelles (« time slots »). Dans le cadre de ce mémoire, ni le codage de canal, ni le facteur de suréchantillonnage (oversampling factor) ne sont considérés, car le but visé est la décomposition en primitives d'une application-cible et non l'étude exhaustive de ses performances. Par ailleurs, l'implémentation considère le cas d'une liaison montante (uplink) en mode DS-CDMA. Pour ce mode, un time slot correspond à 2560 chips et contient un nombre de bits fonction du facteur d'étalement. Le Tableau II inspiré de (Krüger et al., 2004 ; ETSI, 2004) résume quelques uns des paramètres de l'interface radio pour une liaison montante associée au standard UMTS.

Tableau II

Paramètres de l'interface radio d'une liaison montante selon le standard UMTS

Mode	FDD	TDD
Accès multiple	DS-CDMA	TD-TDMA
Débit chip	3.84 Mchip/s	3.84 ou 1.28 Mchip/s
Espacement entre porteuses	4.4 à 5 MHz	idem
Pas de l'espacement	200 kHz	idem
Durée d'une trame radio	10 ms	idem
Structure d'une trame radio	15 time slots	idem
Modulation	QPSK	idem
Facteur d'étalement	4 à 256	1 à 16

2.4.1.4 Les canaux de transport

Les canaux de transport déterminent la manière par laquelle les données sont transmises à travers l'interface radio (Krüger et al., 2004). Ils sont accessibles par la couche physique et par les couches supérieures des systèmes à interconnexions ouvertes (Open Systems Interconnection (OSI)). Parmi eux sont classifiés les canaux dédiés identifiant l'équipement de l'utilisateur. On distingue aussi les canaux communs employant un adressage explicite de l'équipement de l'utilisateur si besoin est.

Dans la catégorie des canaux de transport dédiés, il n'y a qu'un seul canal, le canal DCH (Dedicated Channel), bidirectionnel transmis en partie ou en totalité dans une cellule. Les canaux physiques DPDCH et DPCCH sont utilisés pour transporter les données. Les canaux de transport communs sont au nombre de six : Broadcast CHannel (BCH), le Forward Acess Channel (FACH), le Paging CHannel (PCH), le Random Access CHannel (RACH), le CPCH (Common Packet CHannel) et, enfin, le Downlink Shared Channel (DSCH) et sont décrits dans (ETSI, 2005) et (Krüger et al., 2004).

2.4.1.5 Les canaux physiques

En ce qui concerne les canaux physiques dédiés, en s'attardant sur la liaison montante les types de canaux dénombrés sont :

a. Canal de données physique dédié montant (en anglais uplink Dedicated Physical Data CHannel (DPDCH)) véhiculant le canal de transport DCH.

b. Canal de contrôle physique dédié montant (en anglais uplink Dedicated Physical Control CHannel (DPCCH)), transportant les informations de contrôle générés par la couche physique à savoir : les bits pilotes servant à l'estimation de canal pour la détection cohérente, les commandes de contrôle de puissance (Transmit Power Control (TPC)), les informations de retour (FeedBack Information (FBI)) pour assister les techniques nécessitant une boucle de retour et les indicateurs de combi-

naisons de format (Transport Format Combination Indicator (TFCI)) pour identifier et indiquer différents services opérant simultanément. Le nombre de bits exact de chacun de ces éléments est donné dans (ETSI, 2005).

La voie en phase (voie I) transporte les données tandis que les informations de contrôle le sont par la voie en quadrature (voie Q). La structure des trames radio pour la voie montante des canaux physiques dédiés est illustrée à la Figure 11. Chaque trame de 10 ms est subdivisée en fenêtres temporelles. Chaque fenêtre temporelle possède une longueur de 2560 chips, correspondant à une période du contrôle de puissance. La supertrame, équivalente à 72 trames, opère pendant 720 ms. Sur cette même figure, k détermine le nombre de bits dans chaque fenêtre temporelle. Il est relié au facteur d'étalement (spreading factor (SF)) du canal physique selon l'équation 2.5. Ce dernier varie entre 256 et 4 selon le débit des données.

$$SF = \frac{256}{2^k} \tag{2.5}$$

Avec k qui représente un entier déterminant le nombre de bits dans chaque fenêtre temporelle.

Les canaux physiques communs de la liaison montante sont dénommés : Physical Random Access CHannel (PRACH) pour le support de transport du Random Access CHannel (RACH) et Physical Common Packet CHannel (PCPCH), utile pour transmettre le canal CPCH.

2.4.1.6 L'étalement

Le processus d'étalement de spectre, décrit au chapitre 1, se situe à la couche physique. Cette procédure comprend une opération de canalisation et une opération d'embrouillage. Au cours de la canalisation, les symboles de données de chacune des voies I et Q sont multipliés indépendamment par un code OVSF (Orthogonal Variable Spreading Factor).

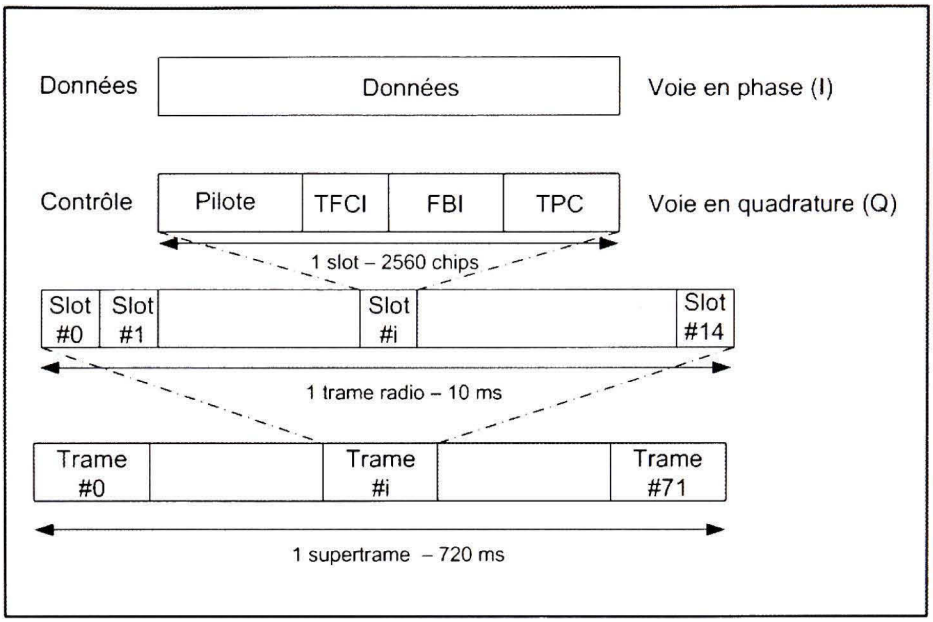


Figure 2.11 Trame radio des canaux DPDCH et DPCCH de la voie montante.

Pendant l'opération d'embrouillage, les signaux résultants sont multipliés par un code d'embrouillage à valeurs complexes. La Figure 12 illustre son principe.

Les codes de canalisation sont des codes à facteur d'étalement variable orthogonaux dits OVSF dont le rôle est d'assurer l'orthogonalité entre les différents canaux physiques d'un ou de plusieurs utilisateurs. Les codes OVSF peuvent être définis à partir d'un arbre comme celui de la Figure 13 . Les codes de canalisation correspondant à un facteur d'étalement SF sont définis à chaque niveau dans l'arbre. Un code peut être utilisé si aucun autre code sur le chemin entre ce code et la racine de l'arbre ou dans le sous-arbre n'a été utilisé dans la cellule considérée. En d'autres termes, dès qu'un code a été alloué à un service, ni ce code, ni ses fils ne peuvent être attribués à un service distinct. La base orthogonale des codes est la même d'une cellule à l'autre.

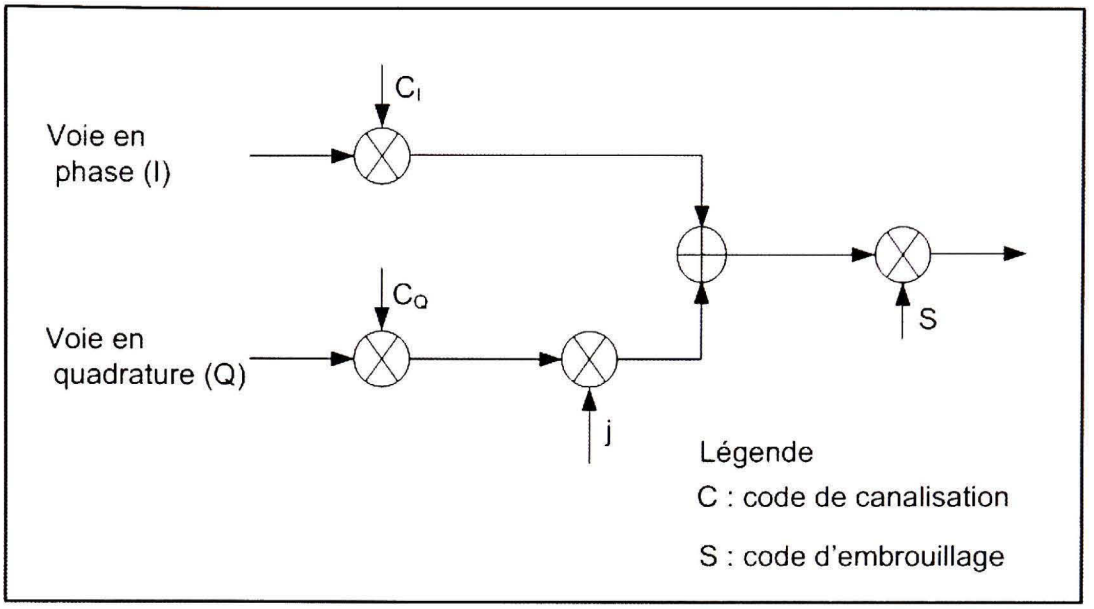


Figure 2.12 Technique d'étalement de spectre.

2.4.1.7 Les codes d'embrouillage

Ce code garantit l'orthogonalité entre les cellules et maintient ainsi la séparation entre les différentes stations de base. Pour une liaison montante, il y a la possibilité de générer des codes d'embrouillage longs ou courts. Les seconds sont recommandés pour des stations de base équipées de récepteurs de pointe pour une détection multi-usagers. La présente étude considère un simple récepteur Rake dans un contexte mono-usager. Par conséquent, le choix du premier type de codes s'impose.

Les codes d'embrouillage longs sont construits à partir de segments de longueur 38400 issus de deux séquences binaires. Ces séquences sont générées à partir de deux polynômes générateurs de degré 25. La description suivante s'inspire de (ETSI, 2005). Les deux séquences binaires x et y sont générées en utilisant respectivement les polynômes générateurs $X^{25} + X^3 + 1$ et $X^{25} + X^3 + X^2 + X + 1$. Ils sont représentés à la Figure 14 . La séquence résultante constitue les segments d'une séquence de Gold.

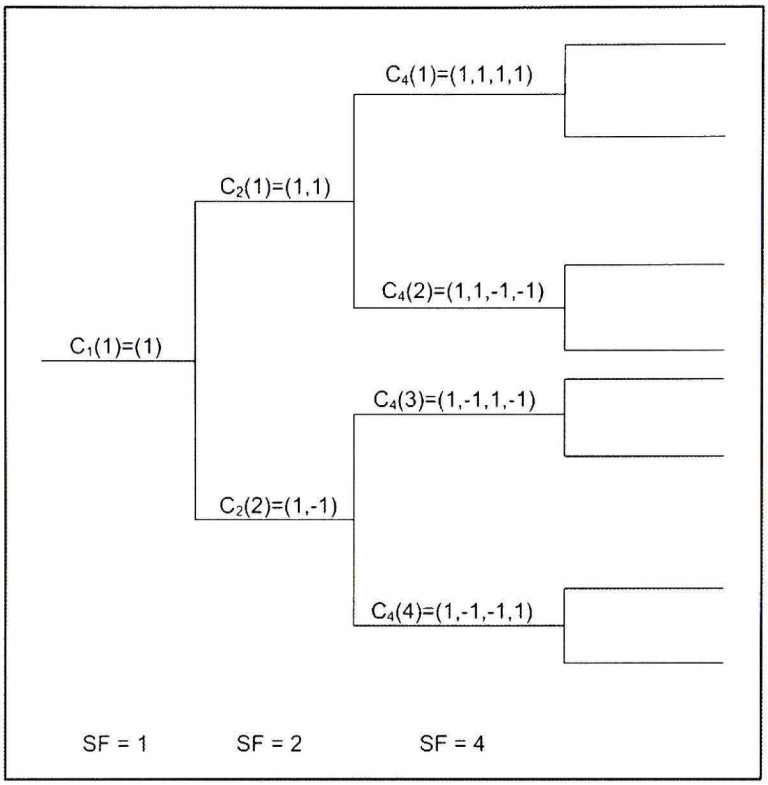


Figure 2.13 Arbre de génération des codes OVSF.

Supposons que $n_{23}...n_0$ est la représentation binaire du code d'embrouillage décimal n. Dans cette représentation, n_0 est le bit le moins significatif. La séquence x dépend du choix de la valeur décimale x_n du code d'embrouillage. Par ailleurs, $x_n(i)$ et $y(i)$ dénotent le i^{eme} symbole des séquences x_n et y respectivement. Les séquences m sont construites comme suit :

- Les conditions initiales sont données par les équations 2.6 et 2.7 (ETSI, 2005) :

$$x_n(0) = n_0,\ x_n(1) = n_1,\ ...,\ x_n(22) = n_{22},\ x_n(23) = n_{23},\ x_n(24) = 1 \quad (2.6)$$

$$y(0) = y(1) = ... = y(23) = y(24) = 1 \quad (2.7)$$

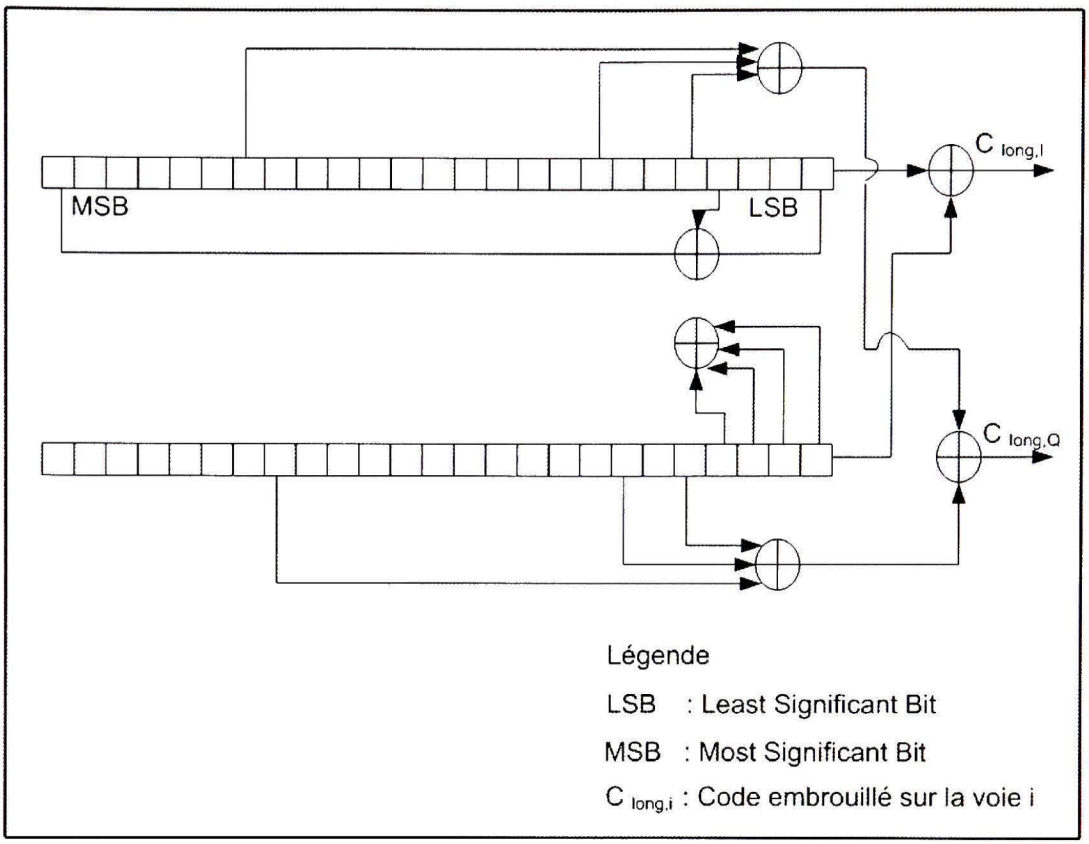


Figure 2.14 Configuration du générateur de codes d'embrouillage pour une liaison montante.

– Les symboles subséquents sont générés de manière récursive selon les équations 2.8 et 2.9 :

$$x_n(i+25) = (x_n(i+3) + x_n(i))_{mod2},\ i = 0,\ 1,\ ...,\ 2^{25} - 27 \tag{2.8}$$

$$y(i+25) = (y(i+3)+y(i+2)+y(i+1)+y(i))_{mod2},\ i = 0,\ 1,\ ...,\ 2^{25}-27 \tag{2.9}$$

– Enfin, les codes $C_{1,n}$ et $C_{2,n}$ du n^{ieme} code d'embrouillage sont définis par les équations 2.10 et 2.11 :

$$C_{1,n} = (x_n(i) + y(i))_{mod2},\ i = 0,\ 1,\ ...,\ 2^{25} - 27 \tag{2.10}$$

$$C_{2,n} = (x_n(i+4)+(x_n(i+7)+(x_n(i+18)+y(i+4)+y(i+6)+y(i+17))_{mod2} \quad (2.11)$$

2.4.1.8 La modulation

En général, la réalisation de la modulation QPSK se déroule de manière à séparer les parties réelles et imaginaires de la séquence nouvellement étalée avant d'être mise en forme par un filtre (facteur de mise en forme ou "roll-off" en anglais valant 0.22). Le but de l'utilisation du filtre de mise en forme est d'éliminer l'interférence inter-symboles. La Figure 15 expose la procédure de modulation avec $f(t)$ qui représente le filtre de mise en forme et s le signal complexe à la sortie de l'embrouilleur. L'étape de mise en forme n'a pas été considérée lors de l'implémentation car elle n'est pas incontournable pour implanter l'algorithme du récepteur Rake dans un processeur numérique.

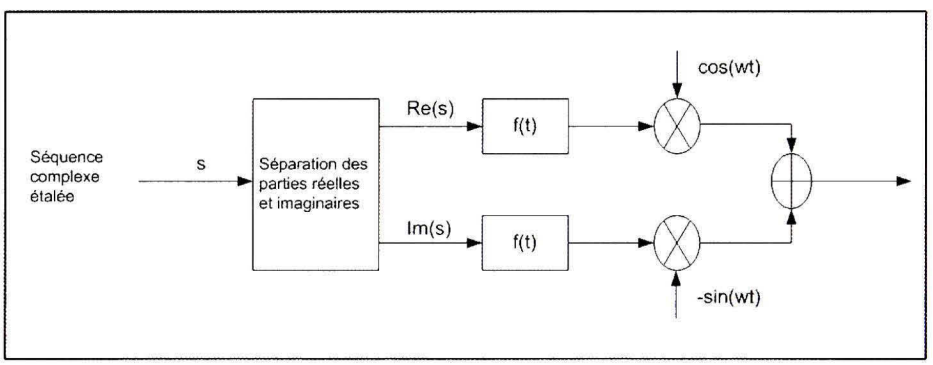


Figure 2.15 Modulation en quadrature de phase de la séquence étalée.

2.4.1.9 Le contrôle de puissance

Le contrôle de puissance permet de contrer les fluctuations de puissance causées par le phénomène d'évanouissement (Kruger et al., 2004). En effet, ce problème survient lorsque la station de base reçoit des signaux de différents utilisateurs situés à des endroits différents, donc ayant des atténuations différentes et que l'un des signaux envoyés éblouisse tous les autres. Pour contrer cet effet, le contrôle de puissance s'assure que le taux d'erreurs binaires soit supérieur à un certain seuil critique afin de bien décoder le signal reçu. Il

veille à ce que chaque mobile ou chaque station de base envoie le minimum de puissance nécessaire pour amoindrir les interférences.

Les deux types de contrôle de puissance utilisés dans les systèmes CDMA sont le contrôle de puissance en boucle ouverte (open-loop) et le contrôle de puissance en boucle fermée (closed-loop). Le contrôle de puissance en boucle ouverte concerne uniquement le mobile (sens montant) et son but est de déterminer le niveau de puissance du signal à transmettre avant d'entrer en communication avec la station de base. Ce niveau de puissance est calculé en fonction de l'atténuation des trajets. Enfin, le contrôle de puissance en boucle ouverte permet de lutter contre les évanouissements à grande échelle (évanouissements dû à la présence d'arbres, de collines ou d'immeubles ...).

Le contrôle de puissance en boucle fermé est utilisé pour compenser les évanouissements rapides. Il diffère du contrôle de puissance en boucle ouverte, dans la mesure où le récepteur concerné calcule des commandes de contrôle et les envoie à la source émettrice pour que celle-ci règle sa puissance d'émission. Une fois que le mobile a établi un lien de communication avec la station de base, le contrôle de puissance en boucle fermé est activé. Il a lieu dans les deux sens, c'est-à-dire dans les voies montante et descendante. Sur la liaison descendante, il est rapide pour limiter les évanouissements à petite échelle.

2.4.1.10 Le changement de cellules

La procédure de changement de cellules ou « handover »en anglais survient lorsqu'une station de base prend le relais d'une autre station de base avec laquelle un mobile est en communication en se déplaçant. Cette opération vise alors à améliorer la qualité de la communication pour un utilisateur en mouvement. Deux types de changement de cellules peuvent survenir : d'une part, le changement de cellules intra-fréquentiel qui conserve la fréquence de la porteuse, et, d'autre part, le changement de cellules inter-fréquentiel qui la modifie. Le premier type de changement de cellules retient notre attention car il est plus usuel .

Il existe deux catégories de changement de cellules intra-fréquentiel : le changement de cellules doux (soft) et le changement de cellules plus doux (softer) . Le changement de cellules doux permet à un mobile de communiquer avec différentes stations de bases sans subir de rupture physique. En effet, pendant la procédure de changement de cellules, le mobile doit interrompre la communication avec une station de base avant d'en établir une autre avec une station de base différente. C'est le cas dans la plupart des systèmes fondés sur le FDMA et le TDMA. Au contraire, dans un système CDMA où les cellules voisines utilisent la même fréquence porteuse, le mobile peut conserver une liaison radio avec plusieurs stations de base simultanément sans coupure : d'où le qualificatif doux. Le principe du changement de cellules doux est représenté à la Figure 16. En mode actif, la station mobile recherche en permanence de nouvelles stations de base émettant sur la fréquence porteuse courante. Une liste de priorité est établie pour séquencer l'ordre dans lequel les codes d'embrouillage descendants doivent être cherchés pour diminuer de manière considérable, le temps et les efforts à fournir. Cette liste est actualisée en continu de façon à réfléter le voisinage en évolution de la station mobile en mouvement. Le critère de mise à jour est la comparaison du niveau du signal émis avec un ensemble de seuils. L'information est ensuite envoyée à la station de base pour effectuer les mises à jour : soit le rajout ou l'enlèvement de telle ou telle station de base de la liste.

Le changement de cellules plus doux est un cas particulier de changement de cellules doux. Il intervient entre des secteurs ou cellules dont les stations de base sont colocalisées. Dans ce cas le mobile peut établir simultanément une communication avec une station de base gérant deux secteurs lorsqu'il se trouve dans la surface où les deux secteurs se chevauchent. Au lieu d'une combinaison par sélection, il permet une combinaison à gain maximal (telle que décrite à section 1.1.3.4) pour atteindre de meilleures performances.

Ainsi le changement de cellules, en version douce ou plus douce, permet d'accroître les performances de la liaison en y ajoutant une forme de diversité par combinaison des répliques. Il s'avère cependant nécessaire de minimiser les situations de changement doux

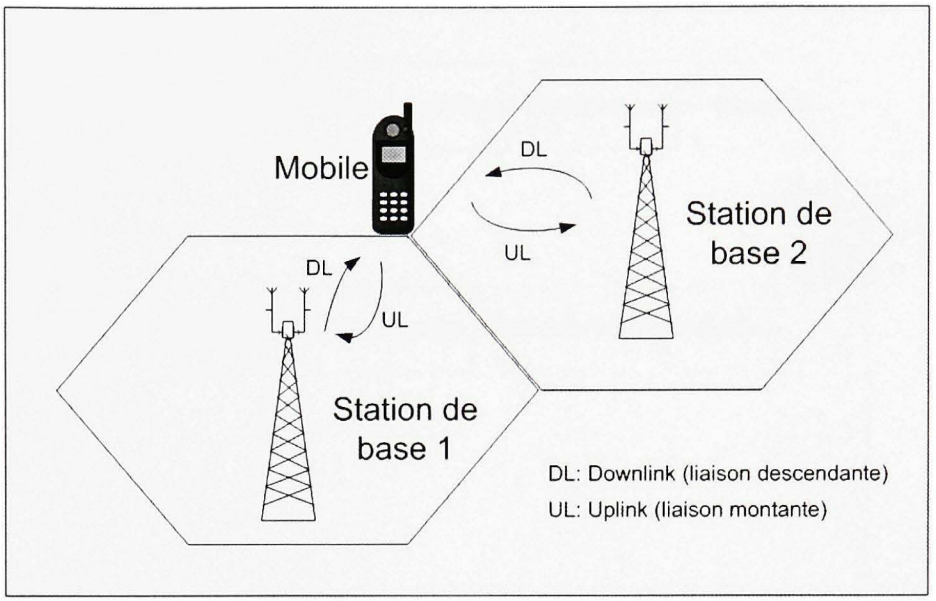


Figure 2.16 Principe du changement de cellules doux dans un système CDMA.

de cellules pour ne pas gaspiller les ressources et par ricochet, diminuer la capacité du système.

2.4.2 Description détaillée du récepteur Rake

Le récepteur naturel pour une modulation à étalement de spectre est le récepteur Rake (Price et al., 1958). Il consiste à exploiter la diversité offerte par les trajets multiples afin d'améliorer les performances d'un système de communication. La recombinaison de l'énergie du signal en utilisant de multiples récepteurs à corrélation est réalisée par un doigt (ou finger en anglais) du récepteur Rake. Chacun d'eux doit détecter les changements de phase et d'amplitude des signaux, et, les corriger toutes les millisecondes, en concordance avec les instants d'échantillonnage du canal. La forme de ces doigts donne au récepteur l'allure d'un râteau d'où dérive son nom. La Figure 17 met en exergue de manière simplifiée les principales étapes à franchir pour avoir une estimation du signal émis.

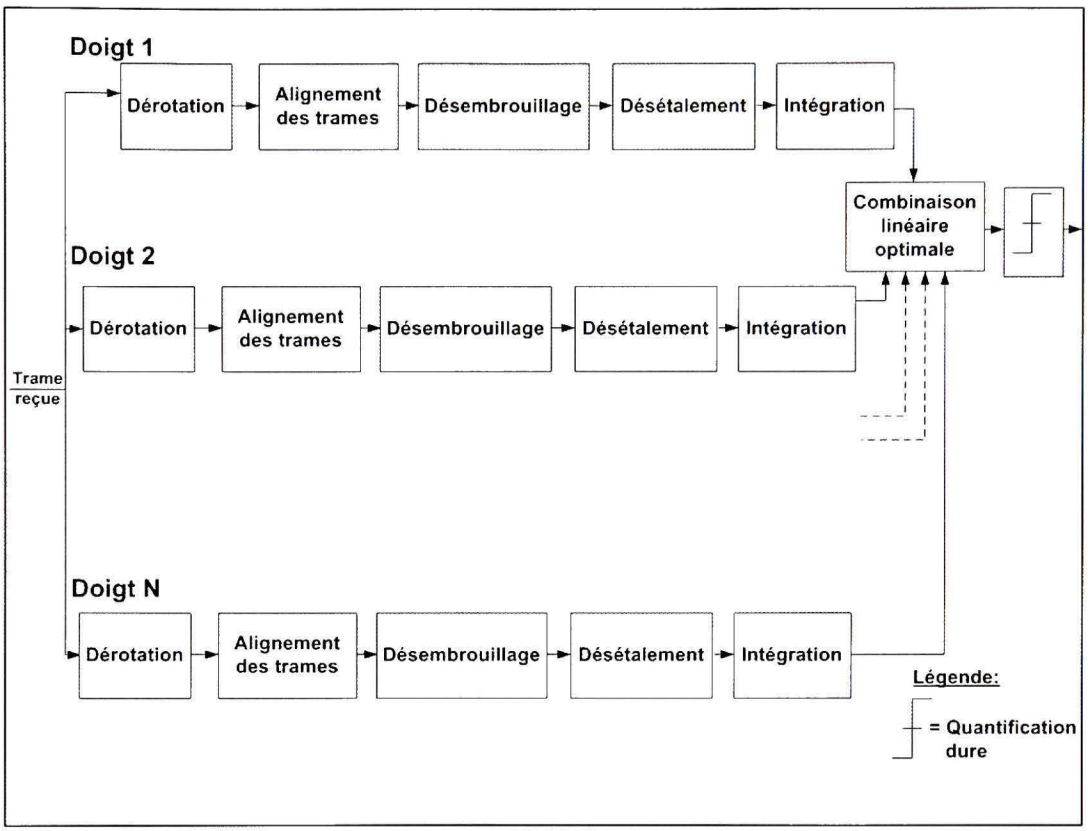


Figure 2.17 Principe du récepteur Rake.

L'application conçue considère le cas d'un seul utilisateur et une réalisation du canal H parfaitement connue. Les étapes de détection sont les suivantes :

a. La dérotation de la phase du signal complexe provenant du canal : Cette étape permet d'éliminer le déphasage introduit par les coefficients de gains du canal. Elle consiste à multiplier la trame reçue par le conjugué complexe des composantes de la matrice de canal.

b. L'alignement des trames ayant été affectés par des délais différents dus aux chemins multiples avec la trame accusant le plus grand délai afin de circonscrire les bornes du processus de détection aux bons intervalles de transmission est l'étape suivante. La Figure 18 illustre les deux étapes décrites ci-dessus . Sur celle-ci, le conjugué

complexe du coefficient de canal du trajet i considéré est représenté par $e^{-j\varphi_i}$ et l'alignement des trames par z^{ni}.

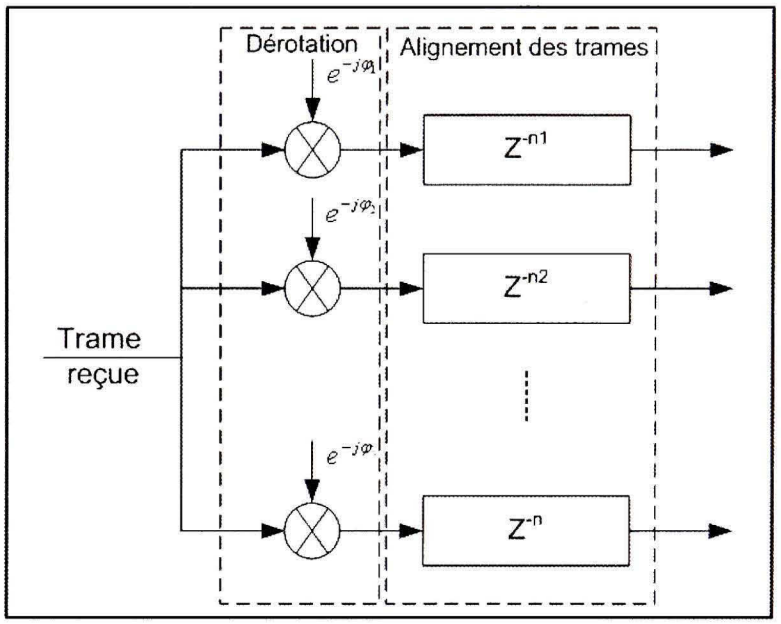


Figure 2.18 Dérotation et alignement des trames sur plusieurs doigts du récepteur.

c. Le désembrouillage : Sur la Figure 19, $C^*_{SC}(t-\tau_i)$ représente de code de désembrouillage de la station mobile désirée pendant la liaison montante. L'astérisque (*) indique la valeur conjuguée complexe dudit code et τ_i, le délai associé au trajet considéré. Ainsi, pour désembrouiller les trames alignées, il suffit de multiplier la trame de chaque trajet par le conjugué complexe du code d'embrouillage employé à l'émetteur. Celui-ci doit lui-aussi être synchronisé avec les délais qui tiennent compte de l'alignement des trames.

d. Le désétalement prend place lorsque survient le moment de multiplier de signal désembrouillé par les codes d'étalement (C_I et C_Q) utilisés au transmetteur mais affectés du délai $t-\tau_i$ induit par la propagation multi-trajets. Cette opération s'effectue sur chacune des voies I et Q en parallèle et en concordance avec les délais ajustés τ_i (Figure 20).

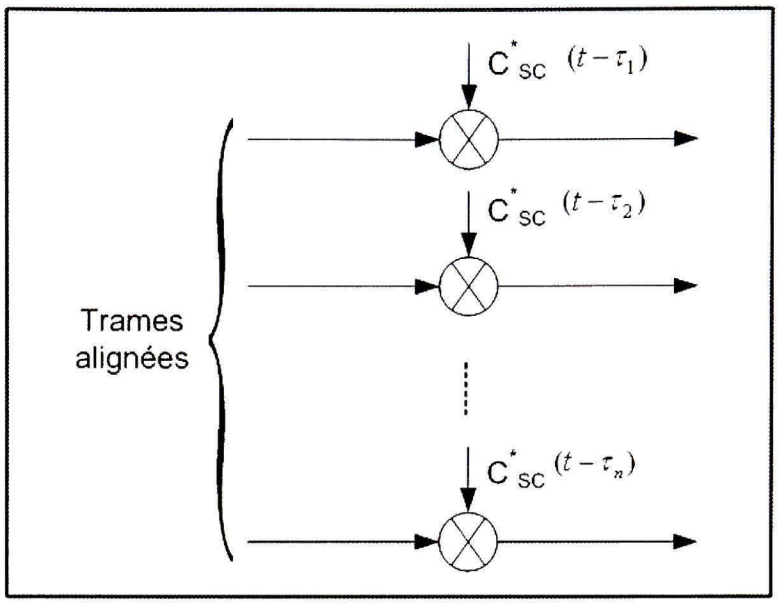


Figure 2.19 Désembrouillage sur un doigt du récepteur.

e. L'intégration est l'étape qui consiste à additionner les composantes du signal associé à chaque trajet sur un intervalle de la longueur du facteur d'étalement, soit de la voie I pour les données, soit de la voie Q pour les informations de contrôle. La trame résultante est normalisée par le facteur d'étalement correspondant tel qu'illustré à la Figure 20 .

f. La combinaison linéaire optimale consiste à combiner les différents multitrajets de chacune des voies, en un seul signal tout en tenant compte du poids de chaque chemin sur l'estimation globale. Ce poids est fonction de l'énergie totale des chemins, ΣE_i , et de celle du trajet dont on veut déterminer le poids, E_i . Il se note $\frac{E_i}{\Sigma E_i}$ comme sur la Figure 21.

g. La quantification dure autorise la prise de décision sur la valeur binaire du bit transmis. Dans ce cas, un résultat positif de l'opération de combinaison linéaire optimale conduit à une valeur 0 et à une valeur 1 dans le cas contraire.

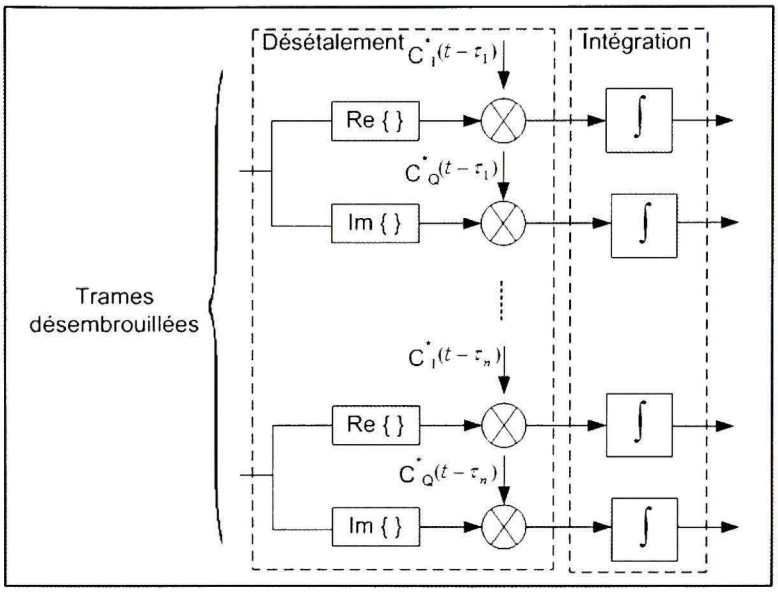


Figure 2.20 Désétalement et intégration.

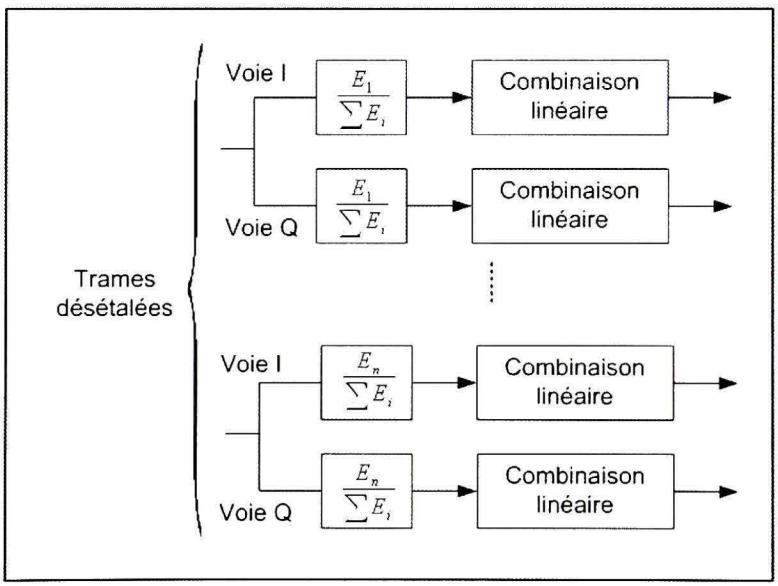


Figure 2.21 Combinaison linéaire optimale.

Les performances de la Figure 22 esquissent le taux d'erreur binaire du récepteur Rake établies selon la couche physique WCDMA du standard UMTS pour quatre doigts.

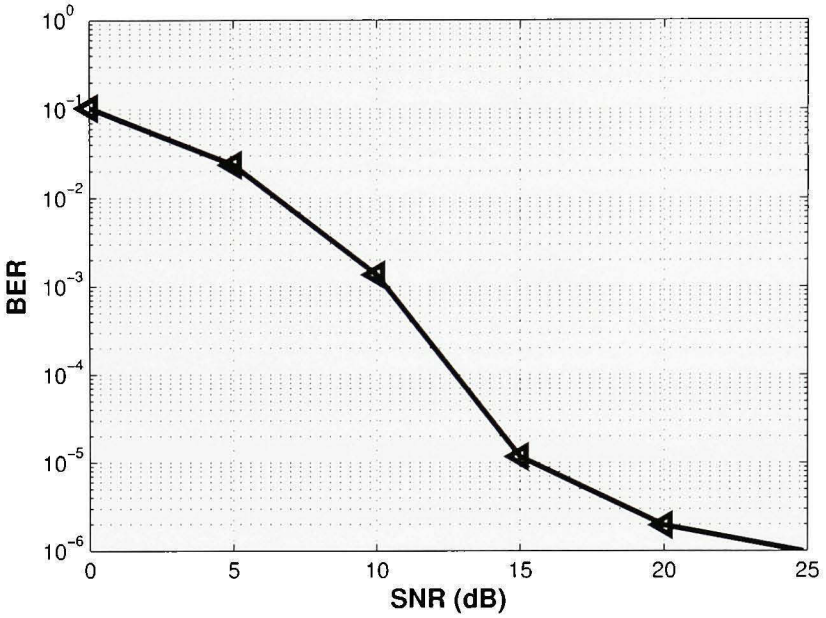


Figure 2.22 Taux d'erreurs binaire du récepteur Rake établies selon la couche physique WCDMA du standard UMTS.

2.5 La transformée rapide de Fourier

Joseph Fourier (1768-1830) est le mathématicien français qui est à l'origine de la transformée de Fourier. Le principe qui sous-tend sa théorie démontre que tout signal continu peut se décomposer en une somme de sinusoïdes. Cette décomposition est réversible et possède d'autres propriétés comme la conservation de l'énergie, la linéarité...

L'analyse fréquentielle des signaux numériques est à la base du traitement numérique du signal. Un signal numérique dans le temps peut être transformé dans le domaine fréquentiel au moyen de la transformée en z ou de la transformée de Fourier (Proakis et Manolakis, 1996). La transformée continue de Fourier s'applique aux fonctions de carré intégrable et les représente par une intégrale pondérée d'exponentielles complexes. Les équations 2.12 et 2.13 , où $x(t)$ est une fonction continue dans le temps et $X(f)$ la fonction équivalente en fréquence, sont respectivement les expressions générales de la transformée de Fourier

directe et inverse :

$$x(t) = \int_{-\infty}^{+\infty} X(f) e^{j2\pi ft} dt \tag{2.12}$$

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi ft} dt \tag{2.13}$$

L'expression théorique de la transformée continue de Fourier, telle que décrite ci-haut n'est pas convenable pour implémenter des applications en pratique car elle s'étend de l'infini négatif à l'infini positif. Il faut donc introduire une analyse fréquentielle par séquences de longueur finie. Ce processus se nomme discrétisation. La discrétisation de cette fonction périodique permet d'obtenir une somme finie d'exponentielles complexes elle-même périodique et appelée transformée discrète de Fourier (Proakis et Manolakis, 1996 ; Elliot et al. 1982).

2.5.1 La transformée de Fourier discrète

La transformée de Fourier discrète (en anglais Discrete Fourier Transform (DFT)) dérive de l'échantillonnage de la fonction continue dans le temps $x(t)$ à des intervalles de taille N. La transformée discrète inverse de Fourier (en anglais inverse discrete Fourier Transform (IDFT)) effectue l'opération contraire à la DFT c'est-à-dire qu'elle convertit le spectre fréquentiel $X(k)$ dans le domaine du temps $x(n)$. Carl Friedrich Gauss (1805) en est l'auteur. L'équation 2.14 est celle de la DFT et l'équation 2.15 est celle de la IDFT (Proakis et Manolakis, 1996 ; Kuo et al., 2006) :

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad k = 0,\ 1,\ ...,\ N-1, \tag{2.14}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \qquad n = 0,\ 1,\ ...,\ N-1 \tag{2.15}$$

où les termes W_N^{kn} représentent des facteurs de phase et correspondent à des variables complexes d'amplitudes unité ayant des phases différentes. Ils se définissent par la relation 2.16 :

$$W_N^{kn} = e^{-j(\frac{2\pi}{N})kn} \tag{2.16}$$

La complexité en calculs de la DFT est de l'ordre de N^2. Ceci demeure élevé en quantité de ressources matérielles nécessaires et suggère la considération d'autres moyens pour optimiser l'utilisation de ces dernières. Le focus est mis sur l'accélération du temps de calculs.

2.5.2 La transformée rapide de Fourier

Une factorisation récursive de la formule générale de l'algorithme de la transformée de Fourier permet son optimisation. En effet, Cooley et Tukey introduisent en 1965 le premier algorithme de la FFT connu sous le nom de radix 2 car il exige que la taille des points à traiter soit une puissance de 2. Sa rapidité réside dans l'exploitation des propriétés des facteurs de phase présentées aux équations 2.17 et 2.18 (symétrie et périodicité) qui lui permettent de ne pas répéter les opérations inutiles et d'en simplifier d'autres.

Périodicité :

$$W_N^k = W_N^{k+iN}, \qquad k = 0,\ 1,\ ...,\ N-1 \tag{2.17}$$

Symétrie :

$$W_N^k = -W_N^{k+(N/2)} \tag{2.18}$$

Cette découverte est en fait une réinvention de l'algorithme de la DFT de Carl Friedrich Gauss. D'autres algorithmes ont vu le jour (Elliot et al., 1982) pour des tailles qui ne

sont pas nécessairement des puissances de 2 et ceux-ci, comme le premier, réduisent la complexité d'implémentation de la DFT à un ordre de $Nlog_2N$. Ils partagent par ailleurs en commun l'approche diviser pour conquérir.

L'approche diviser pour conquérir consiste à séparer récursivement la DFT de tout vecteur de taille $N = N_1 + N_2$ en des DFTs de tailles plus petites, soit N_1 et N_2. Par exemple, pour la forme radix 2, la factorisation à l'étage courant, k, fait appel à deux transformées de Fourier discrètes de l'étage précédent, $k - 1$, donc de tailles égales ($N_1 = N_2 = N/2$) et la procédure est répétée jusqu'à obtenir des vecteurs de taille unité. Le calcul des DFTs s'effectue sur ceux-ci et on en déduit ceux des étages supérieurs de manière récursive. La DFT du dernier étage représente de résultat désiré.

2.5.2.1 Algorithmes Radix 2

Cette forme est la plus répandue des algorithmes de la FFT. Pour cette famille d'algorithmes, la longueur des vecteurs d'entrée N est une puissance de deux : $N = 2^m$. La formule du radix 2 met ainsi en évidence une cellule de base appelée papillon (en anglais butterfly). Il effectue le calcul de la DFT de chaque groupe de deux points et se répète récursivement le long des étages. De plus, l'addition, la soustraction et la multiplication par un facteur de phase sont ses opérations primaires.

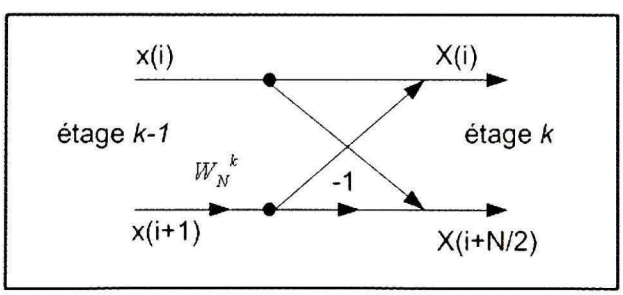


Figure 2.23 Architecture papillon DIT radix 2.

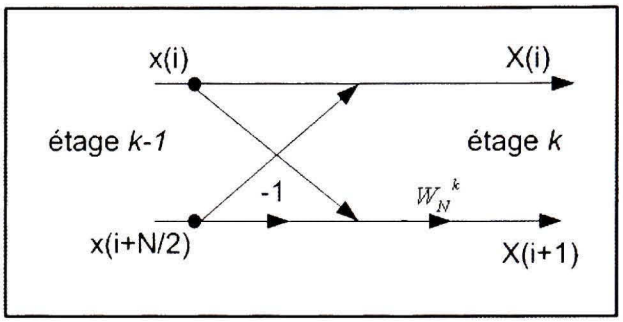


Figure 2.24 Architecture papillon DIF radix 2.

Il existe deux types de décimation selon que la multiplication par le facteur de phase ait lieu avant ou après les additions : la décimation temporelle (DIT) présentée à la Figure 23 et la décimation fréquentielle illustrée à la Figure 24 (DIF). L'autre différence notable entre les deux architectures est la séquence des points en entrée. En effet, dans le cas DIT, cette dernière suit l'ordre naturel mais produit des points en sortie suivant un ordre binaire inversé. C'est l'opération contraire pour le cas DIF. L'ordre binaire inversé (en anglais bit reversal order) transforme l'indice d'un point en un indice de valeur binaire miroir au premier. Par exemple, si un indice a la forme c3c2c1c0, l'indice équivalent suivant l'ordre inversé est c0c1c2c3. La littérature (Elliot et al., 1982) propose aussi d'autres algorithmes de la FFT : l'algorithme radix 4, l'algorithme hybride DIT/DIF... mais notre propos se circonscrit autour de l'algorithme étudié, soit le DIT radix 2.

L'étape première du DIT radix 2 nécessite la permutation des valeurs du vecteur d'entrée selon un ordre binaire inversé (Figure 25).

La seconde et dernière étape consiste à combiner les N spectres de fréquences de façon à éliminer la décomposition dans le domaine temporel de la première étape (Figure 26). Les lettres sur cette figure représentent les valeurs spectrales des signaux de la Figure 25 .

Tout le principe est résumé à la Figure 27 sous forme algorithmique et la Figure 28 présente l'exemple d'une FFT radix 2 à 8 points.

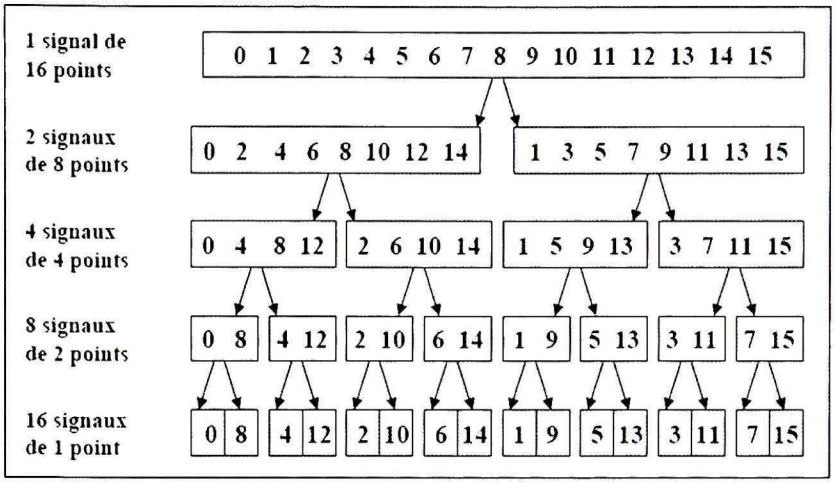


Figure 2.25 Ordre binaire inversé.

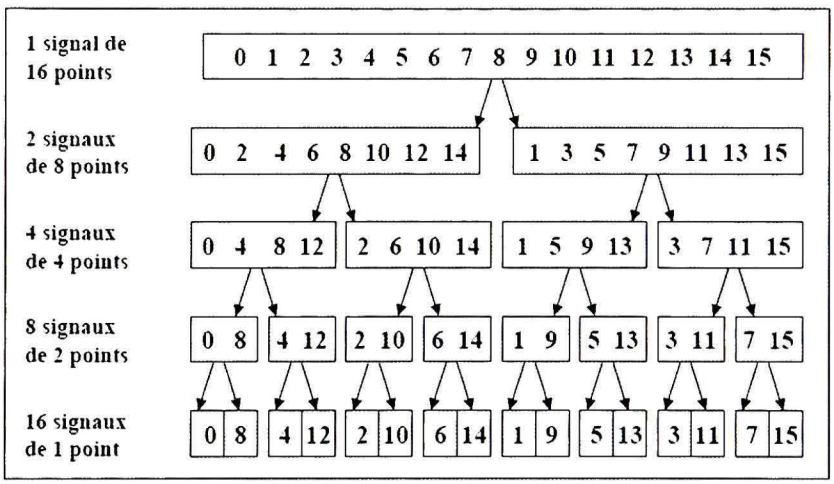


Figure 2.26 Transformation dans le domaine fréquentiel.

2.5.2.2 Inverse de la transformée rapide de Fourier

L'inverse de la FFT (IFFT) permet la détermination rapide de la IDFT. Ses étapes sont similaires à celles de la FFT à la différence de l'utilisation du conjugué du facteur de phase et de la présence du facteur d'échelle $\frac{1}{N}$. Donc le seul calcul de la FFT suffit pour

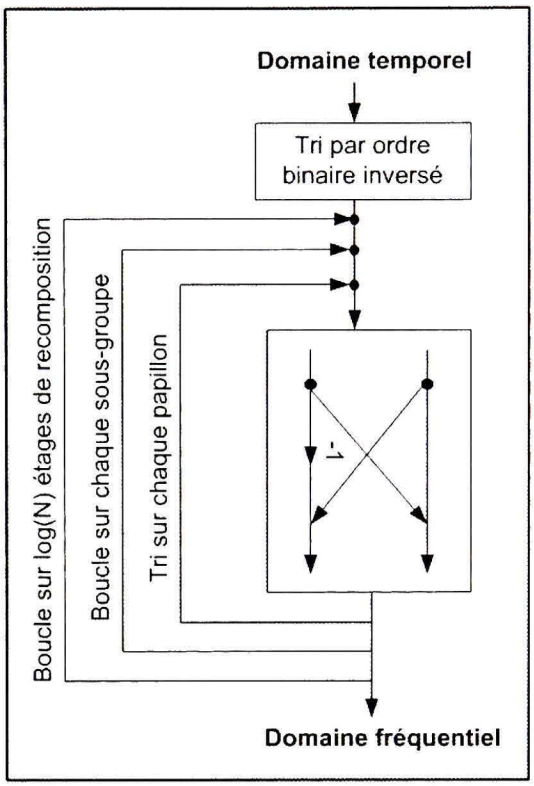


Figure 2.27 Algorithme DIT radix 2.

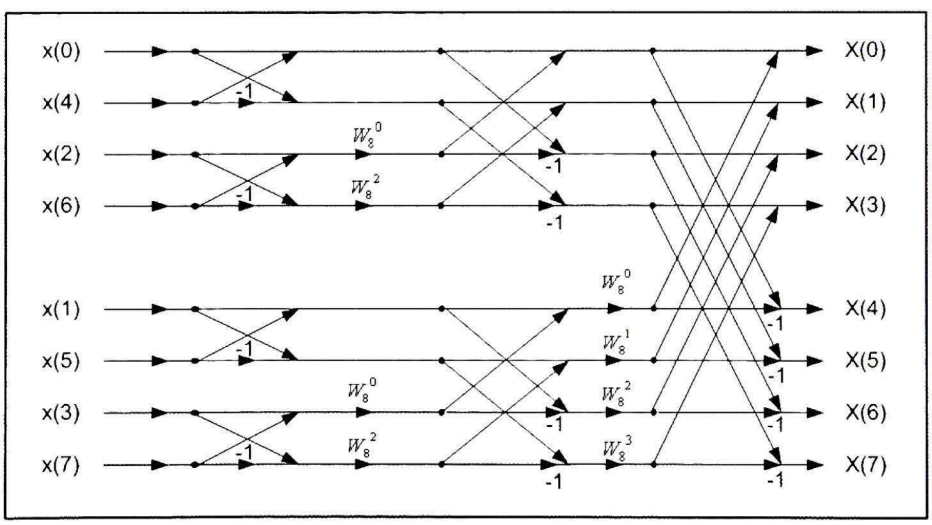


Figure 2.28 Décomposition d'une DIT radix 2 FFT à 8 points.

déduire celui de la IFFT (Kuo et al., 2006). En effet, une IFFT à N points peut être traitée comme une FFT à N points provenant d'une séquence d'entrée $X^*(k)$ comme présenté à l'équation 2.19 . Sa sortie est par la suite ajustée selon le facteur d'échelle $1/N$ pour obtenir $x^*(n)$. Le complexe conjugué s'applique à $x^*(n)$ pour déduire $x(n)$. Cette dernière étape ne s'applique pas dans le cas des signaux réels.

$$x^*(n) = \frac{1}{N} \sum_{k=0}^{N-1} X^*(k) W_N^{nk}, \quad n = 0,\ 1,\ ...,\ N-1 \tag{2.19}$$

2.6 Conclusion

Dans ce chapitre, il a été question de présenter les algorithmes qui feront l'objet d'une décomposition en primitives dans les chapitres à venir. Pour ce qui est du codage spatio-temporel par couches, nous avons choisi d'évaluer et de comparer les performances de l'architecture traditionnelle VBLAST avec celle du SQRD d'une part. Il en est ressorti qu'elles sont quasi-identiques et que la seconde architecture est celle qui offre une complexité de calculs réduite.

En outre, un brève présentation des systèmes UMTS a été introduite puisque le contexte de l'étude du récepteur Rake s'effectue dans le cadre des systèmes WCDMA obéissant aux exigences de l'interface radio terrestre de l'UMTS. Par la suite, le principe de ce récepteur a été illustré, appuyé par une description des différentes étapes de détection en supposant une connaissance parfaite du canal.

Les précédents paragraphes décrivent des variantes d'algorithmes de la FFT, soit la décimation temporelle et la décimation fréquentielle pour le cas radix 2. Il existe aussi des algorithmes employant des radix d'ordre supérieur à ceux décrits ici et d'autres qui s'emploient à faire une combinaison de radix afin d'alléger la complexité d'implémentation. L'algorithme radix 2 sous sa version décimation temporelle a été choisi pour la décomposition future en primitives à cause de sa simplicité et de sa convenance pour implé-

mentation dans les processeurs numériques du signal en général, et, dans le processeur TMS320C6711 en particulier. Ce dernier fait l'objet du prochain chapitre.

CHAPITRE 3

L'ARCHITECTURE DU PROCESSEUR

L'élaboration d'une méthodologie d'extraction de primitives fait appel d'une part à une familiarité dans la connaissance des opérations fréquemment employées dans le domaine d'intérêt. À cela s'ajoute une dextérité dans l'appréhension de l'architecture-cible dans laquelle les applications seront implémentées, utile pour faire une analogie entre les ressources du côté application et celles du côté matériel. Dans le cadre de la procédure de vérification de la méthodologie proposée, les algorithmes abordés dans le précédent chapitre font l'objet d'une implémentation matérielle au sein d'un processeur numérique de signal (en anglais Digital Signal Processor (DSP)) à virgule flottante, le TMS320C6711, de la célèbre firme américaine Texas Instruments. Ceci donne prise à une description de l'architecture générale de ce processeur, objet du présent chapitre.

Les processeurs TMS320C6x ont la particularité d'être rapides. Ils ont été conçus afin de pouvoir exécuter des centaines de millions d'instructions par seconde (MIPS) pour des applications telles le traitement d'images, la troisième génération de réseaux sans fil, les modems câbles ... Un premier volet de ce chapitre est de passer en revue la structure des processeurs de la famille 6000 suivi par une présentation de la composition de sa mémoire et d'un inventaire des modes d'adressage disponibles au sein de celle-ci. Ensuite et finalement, prendront tour à tour place un bref survol sur la représentation des nombres en format point flottant et des techniques d'optimisation avec emphase sur le pipeline logiciel.

3.1 L'architecture générale de la famille TMS320C6000

La famille TMS320 englobe les processeurs 16-32 bits à virgule fixe et à virgule flottante. En 1982, le premier processeur de cette série voit le jour sous l'appelation TMS32010 (Chassaing, 2002). Il est à virgule fixe. Aujourd'hui, la famille TMS320 se subdivise

en quatre plateformes principales (Texas Instruments, 2007) soit : la plateforme DaVinci qui regroupe le TMS320DM6000 et le TMS320DM300, les plateformes TMS320C2000, TMS320C5000 et TMS320C6000. La première plateforme est une solution de traitement numérique de signal orientée vers les applications vidéo. Elle fournit des appareils vidéo équipés de processeurs intégrés, des logiciels et des outils dans le but de simplifier les étapes de conception et d'accélérer l'innovation. La seconde est une famille de contrôleurs qui a main mise sur l'intégration des périphériques et facilite l'emploi des microcontrôleurs grâce à la puissance de traitement dont elle dispose et à son efficacité. La famille TMS320C5000 quant à elle se compose de processeurs numériques très peu dissipatifs en puissance, est dotée d'un contrôle automatique de la puissance et vise principalement les produits personnels et portatifs comme les récepteurs GPS (Global Positionning System), les appareils de musique numérique ... La série TMS320C6000 en virgule fixe offre des processeurs numériques très rapides optimisés pour la vidéo, la voix, les infrastructures sans fil, les applications en télécommunications et en traitement d'images. Ce sont les composants TMS320C62x et TMS320C64x. En virgule flottante, cette série maximise son efficacité et ses performances pour les applications audio selon le format TMS320C67x. Pour simplifier l'écriture, la série TMS320C6000 sera notée C6000 et pour le processeur TMS320C6711, la notation C6711 sera employée pour le désigner.

Les architectures VLIW (Very Long Instruction Word) sont des structures dans lesquelles plusieurs instructions sont recueillies et exécutées simultanément (Kehtarnavaz, 2004). Les processeurs C6000 sont des architectures VLIW d'ordre 8, c'est-à-dire qu'ils sont capables d'exécuter jusqu'à huit instructions de 32 bits en parallèle (Texas Instruments, 2000, 2001, 2005 et 2006). Le coeur de l'unité centrale de traitement (en anglais Central Process Unit, CPU) est constitué de 32 registres à 32 bits, de huit unités de traitement soit deux multiplieurs et six unités arithmétiques et logiques (en anglais Arithmetic and Logic Units, ALU).

La compréhension du fonctionnement des DSP se rend plus accessible par une représentation sous forme de blocs interconnectés. La Figure 1 expose l'organisation générale en blocs d'un processeur C6000.

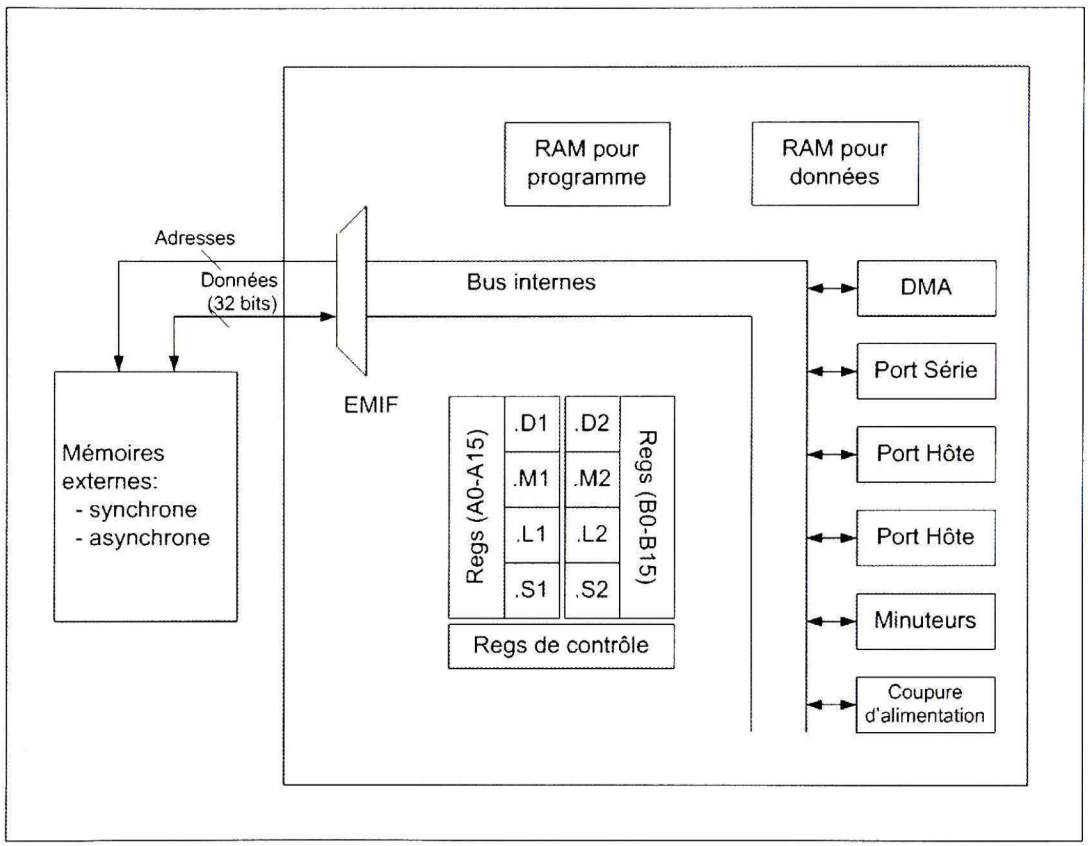


Figure 3.1 Architecture générique de la famille C6000.

La génération des DSP TMS320C67x intègre des composants à virgule flottante. Elle comprend les composants C6711, 6711B, 6711C et 6711D. Le C6711 et le C6711B peuvent exécuter jusqu'à 900 millions d'opérations à virgule flottante par seconde (MFLOPS) à une fréquence de 150 MHz et peuvent produire deux MAC (Multiply and Accumulate) par cycle pour un total de 300 MMAC (Million de MAC). Le C6711C et C6711D quant à eux peuvent exécuter jusqu'à 1200 MFLOPS à une fréquence de 200 MHz. Ces performances atteignent 1500 MFLOPS pour une fréquence de 250 MHz pour le C6711D. Ils peuvent aussi produire deux MAC par cycle pour un total de 400 MMAC.

La famille C67x utilise l'architecture VelociTI, qui est une architecture VLIW avancée. L'architecture VelociTI permet à plusieurs paquets d'exécution (en anglais Execution Packets (EP)) de s'inclure dans le même paquet de chargement. Le paquet d'exécution représente un groupe d'instructions en parallèle (Kehtarnavaz, 2004). Dans ces conditions, les instructions qui se trouvent dans le même paquet d'exécution se déplacent alors ensemble dans le même étage de pipeline. L'architecture VelociTI est déterministe, elle possède quelques restrictions sur le lieu et le moment où le chargement, l'exécution et la sauvegarde des instructions sont effectués. Les caractéristiques de cette architecture permettent de réduire la taille du code, garantissent une meilleure flexibilité des types de données et du code, et enfin, d'avoir des branchements totalement pipelinés. Le processeur C67x est composé de trois principales parties : le CPU, les périphériques, et de la mémoire. La Figure 2 illustre son architecture interne. Le coeur de son unité centrale possède :

- Une unité de chargement de programme (program fetch).
- Une unité de répartition des instructions (instruction dispatch).
- Une unité de décodage des instructions (instruction decode).
- 32 registres à 32 bits séparés en deux fois 16 registres nommés A et B.
- Deux voies pour les données (data path) contenant chacune quatre unités de traitement.
- Des registres de contrôle.
- Une logique de contrôle.

Les unités de traitement exécutent des instructions logiques, de décalage (shifting), des multiplications et des opérations sur les adresses. Tous les transferts de données entre la file de registres et la mémoire se font exclusivement par les unités d'adressage de données (« data adressing »). Les quatre chemins sont croisés, ce qui permet l'échange de données entre les deux files de registre A et B. Deux croisements entre les files de registres sont notés « 1X » ou « 2X ». Le chiffre signifie l'origine de l'échange, soit « 1 » pour le registre

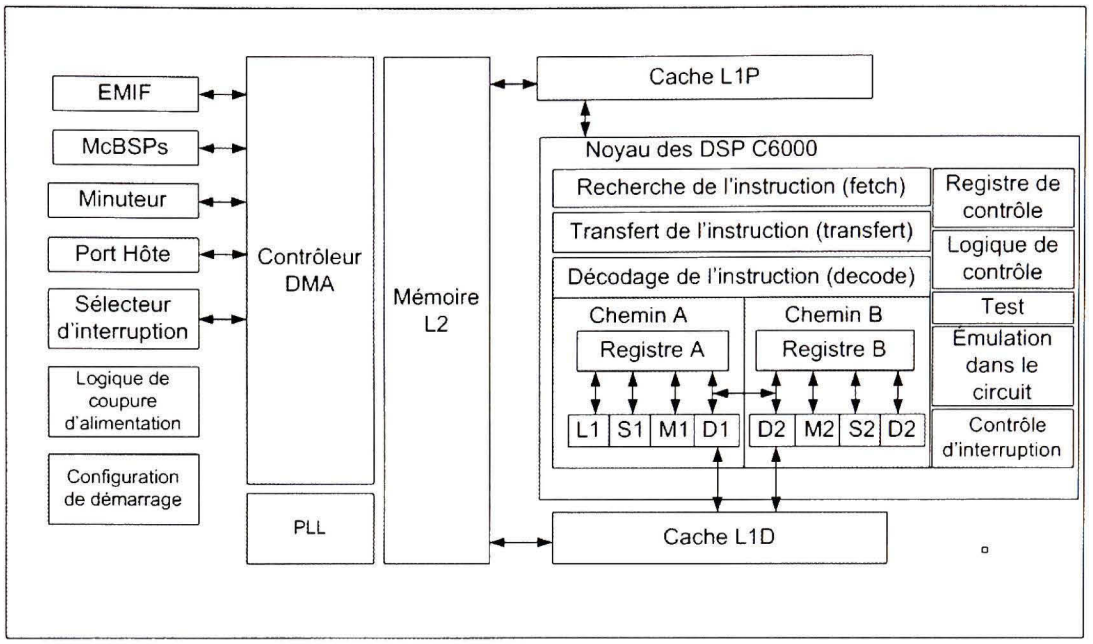


Figure 3.2 Architecture interne du processeur TMS320C6711.

A et « 2 » pour le registre B. La lettre « X » marque le croisement. Il ne peut y avoir qu'un maximum de deux croisements par cycle d'horloge. Sur le diagramme-bloc de la mémoire interne du processeur C6000, à la Figure 4, la mémoire de programme et celle des données sont séparées. La famille C6000 est donc une architecture de type Harvard. On y voit aussi apparaître :

a. Deux files de registres A et B.

b. Deux niveaux de cache. Le cache programme de niveau 1 (L1P) et le cache de données du même niveau (L1D) sont des caches de 32-Kbit. Le niveau 2 (L2) possède un espace mémoire de 512-Kbit qui est partagé entre programme et données. Le niveau L2 peut être configuré en tant que mémoire, cache ou une combinaison des deux.

Les bus internes illustrés à la Figure 3 comprennent un bus d'adresses pour les programmes de 32 bits et un bus de 256 bits pour les données. Ce dernier accommode :

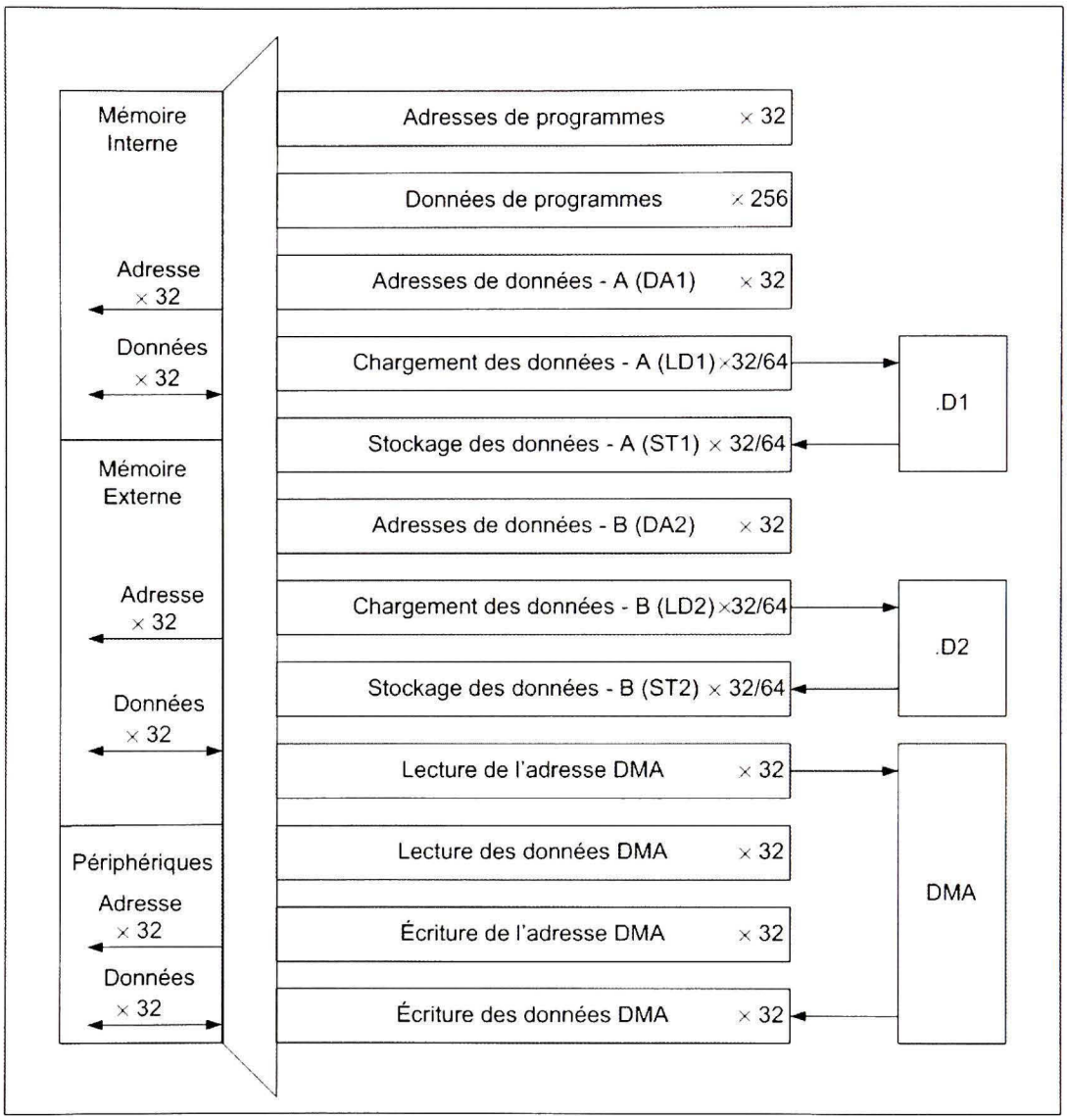


Figure 3.3 Bus internes.

– Huit unités fonctionnelles à 32 bits pouvant opérer en parallèle : .L1, .S1, .M1, .D1 pour la voie A d'une part et .L2, .S2, .M2 et .D2 pour la voie B d'autre part. Le parallélisme se détermine pendant la compilation car il n'y a pas de mécanisme matériel pour l'analyse de la dépendance des données. Parmi ces huit unités sont dénombrés quatre ALUs (Arithmetic and Logic Unit) à virgules fixe et flottante (deux unités .L et deux unités .S), deux ALUs à virgule

fixe (unités .D) et deux multiplicateurs à virgules fixe et flottante (unités .M). Chaque unité peut lire ou écrire directement dans un registre appartenant à sa voie. L'unité .M effectue des multiplications. L'unité .L opère des intructions de branchement et des opérations arithmétiques et logiques. L'unité .S permet la manipulation des bits, des opérations arithmétiques et de branchement. l'unité .D assure les opérations arithmétiques, de transferts de données, de chargement et de stockage. Finalement, l'unité .S contient des multiplicateurs à virgule fixe et flottante. L'unité .D contient des multiplicateurs à virgule fixe et flottante.

- Deux voies de chargement depuis la mémoire à 32 bits LD1 et LD2 (LD pour Load).
- Deux voies de stockage vers la mémoire à 32 bits ST1 et ST2 (ST pour Store).
- Deux chemins d'adresse de données à 32 bits DA1 et DA2 (DA pour Data Address).

c. Un contrôleur d'accès des données en mémoire (en anglais Data Memory Access, DMA) pourvoyant 32 bits pour les données et 32 bits pour le bus d'adresses.

d. Une interface avec la mémoire externe par laquelle l'accès se fait via un bus d'adresses de 20 bits et un bus de données de 32 bits.

En outre, les registres supportent des données sur 32 et 40 bits pour les calculs en virgule fixe. Les données sur 40 bits sont contenues dans deux registres : les 32 bits de poids faibles (LSB) sont stockés dans un registre pair et les huit bits de poids forts restants (MSB) sont emmagasinés dans les huit bits LSB du registre situé immédiatement au-dessus et sont de poids impairs. Le C67x peut également utiliser cette même paire de registres pour les calculs en double précision sur 64 bits. Les périphériques qui se greffent CPU sont :

- Un contrôleur d'accès mémoire direct externe (en anglais External Direct Memory Access, EDMA).

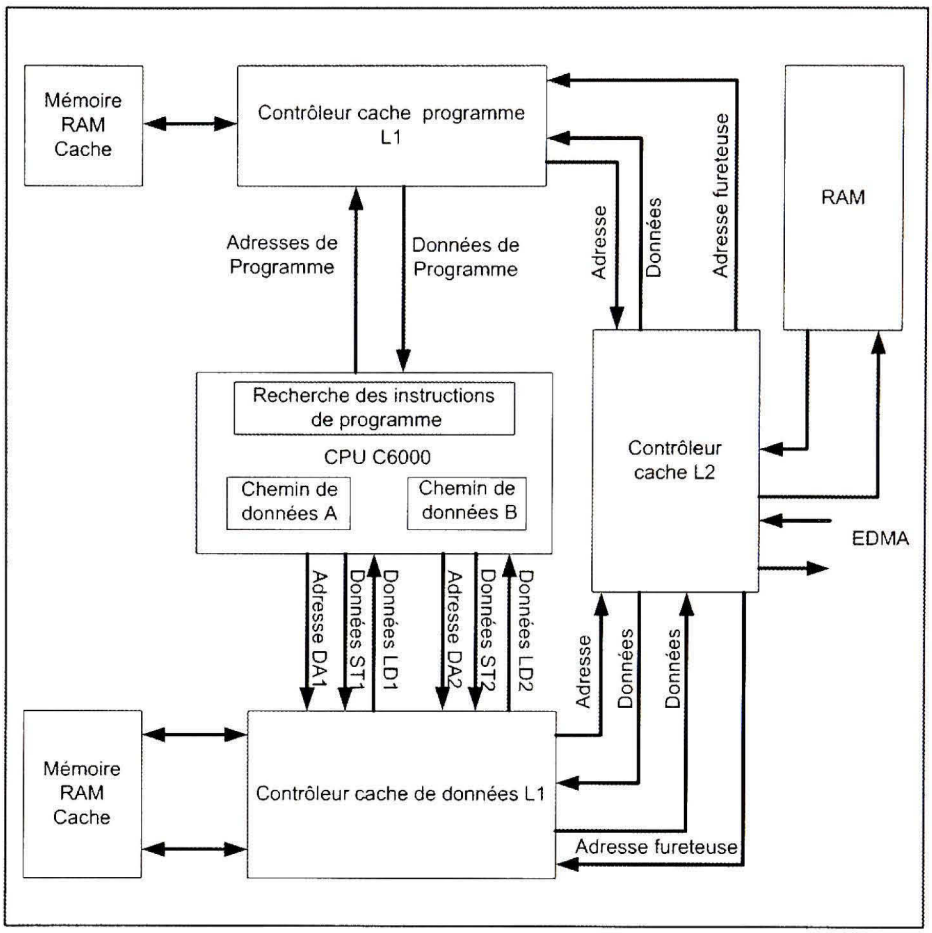


Figure 3.4 Diagramme-bloc de la mémoire interne.

- Un port parallèle de 16 bits (en anglais Host Port Interface, HPI).
- Deux bus serie (en anglais Multichannel Buffered Serial Port, McBSP).
- Deux minuteurs (« timers ») de 32 bits.
- Un générateur d'horloge par la boucle à verrouillage de phase (en anglais Phase Locked Loop, PLL).
- 32 bits d'interface mémoire externe (en anglais External Memory Interface, EMIF).

Enfin, le processeur requiert 3.3 V pour alimenter ses entrées et ses sorties et 1.8 V pour son noyau. La taille totale de la mémoire vaut 4GB et nécessite un partitionnement raffiné. Dans ce qui suit, l'expression $0x$ représente la notation hexadécimale des nombres.

3.2 La mémoire

Dans un DSP, la mémoire peut être soit statique (en anglais Static Random Access Memory, SRAM) ou dynamique (en anglais Dynamic Random Access Memory, DRAM). La première est plus rapide que la seconde mais coûte aussi plus chère puisqu'elle nécessite plus de silicium lors de sa fabrication. La mémoire peut aussi posséder des attributs de synchronisme (DRAM synchrone, SDRAM et SRAM synchrone, SBRAM) ou d'asynchronisme (SRAM) selon qu'elle dépend d'une horloge maîtresse ou pas. Par ailleurs, la mémoire dynamique exige un rafraîchissement périodique pour son bon fonctionnement. La solution impliquant l'emploi de la DRAM synchrone offre un bon compromis entre le coût et les performances.

Doté d'une mémoire totale de 4GB, le C6711 est partitionné selon un plan d'adressage qui correspond à :

- Une mémoire interne de programme (en anglais Program memory, PMEM).
- Une mémoire interne de données (en anglais Data memory, DMEM).
- Des périphériques internes.
- Des espaces de mémoire externes nommés CE0, CE1, CE2 et CE3 qui supportent à la fois les mémoires synchrones (SBRAM, SDRAM) et asynchrones (SRAM, Random Only Memory ou ROM) accessibles en bytes (huit bits), mi-mots (16 bits) ou en mots (32 bits).

Le tableau III fait cas du plan d'adressage du processeur C6711, inspiré de (Texas Instruments, 2001). Cette puce possède 16 MB de SDRAM externe associée à l'espace mémoire CE0 à l'adresse $0x80000000$ et 72 kB de mémoire interne débutant à l'adresse $0x00000000$. Par ailleurs, la présence de 128 kB de mémoire flash se voit assigner un espace mémoire qui commence à l'adresse $0x90000000$.

La mémoire interne a deux banques de sorte que deux opérations de chargement et deux opérations de stockage puissent opérer simultanément. Ceci demeure véridique tant que

Tableau III

Plan d'adressage du C6711

Attribut associé	Adresse	Taille (bytes)
RAM interne	0000 0000	64K
Réservé	0001 0000	24M
Registres de contrôle EMIF	0180 0000	32
Registres de configuration de la cache	0184 0000	4
Registres à compteurs et à adresses de la base L2	0184 4000	32
Registres à compteurs et à adresses de la base L1	0184 4020	32
Registres 'flush and clean' de L2	0184 5000	32
Registres d'attributs-mémoires CE0	0184 8200	16
Registres d'attributs-mémoires CE1	0184 8240	16
Registres d'attributs-mémoires CE2	0184 8280	16
Registres d'attributs-mémoires CE3	0184 82C0	16
Registres de contrôle HPI	0188 0000	4
Registres McBSP0	018C 0000	40
Registres McBSP1	0190 0000	40
Registres du timer 0	0194 0000	12
Registres du timer 1	0198 0000	12
Registres de sélection des interruptions	019C 0000	12
Paramètres RAM EDMA	01A0 0000	2M
Registres de contrôle EDMA	01A0 FFE0	32
Pseudo-registres QDMA	0200 0020	20
Données McBSP0	3000 0000	64M
Données McBSP1	3400 0000	64M
CE0, SDRAM	8000 0000	16M
CE1, 8 bits ROM	9000 0000	128K
CE1, 8 bits ports entrées/sorties	9008 0000	4
Carte fille CE2	A000 0000	256M
Carte fille CE3	B000 0000	256M

l'accès aux données a lieu dans des banques distinctes. Cependant, s'il se fait dans une même banque en une seule instruction, un conflit apparaît et bloque l'exécution des instructions suivantes. L'implémentation d'une application requiert de savoir si la mémoire interne est suffisante pour le traitement. Si elle l'est, il devient judicieux de l'implanter dans la mémoire interne pour éviter les délais associés aux accès mémoires externes. Par contre si tout le programme ne peut être contenu dans la mémoire interne, alors l'astuce

consiste à intégrer les portions les plus gourmandes en ressources dans la mémoire interne pour une exécution efficace. Pour les codes répétitifs, la configuration de la mémoire interne comme cache convient. Ceci diminue le rythme des accès mémoires et, par ricochet, les délais qu'ils induisent.

3.3 Les modes d'adressage

Les modes d'adressage spécifient la manière avec laquelle les accès mémoires sont effectués dans le processeur. Le rôle du registre de mode d'adressage (en anglais address mode register, AMR) est de spécifier le mode d'adressage à utiliser par chacun des huit registres A4 à A7 d'un côté et B4 à B7 de l'autre (Texas Instruments, 2000). Ces derniers peuvent opérer soit en mode linéaire ou en mode circulaire. Le registre AMR dispose d'un champ à deux bits pour que chacun des huits registres ci-dessus mentionnés sélectionne l'un des deux modes appropriés. L'adressage linéaire est l'option par défaut. Le mode indirect est le plus usuel dans cette catégorie. En ce qui concerne l'adressage circulaire, la spécification de la taille du bloc employé est nécessaire dans les registres de banques BK (en anglais bank). La Figure 5 illustre les champs de sélection ainsi que leur taille. Le tableau IV décrit la configuration adéquate au mode d'adressage désiré.

Tableau IV

Description du mode AMR

Mode	Description
00	Adressage linéaire (par défaut)
01	Adressage circulaire en utilisant BK0
10	Adressage circulaire en utilisant BK1
11	Réservé

3.3.1 L'adressage indirect

Le mode d'adressage indirect permet d'accéder à une donnée par l'intermédiaire d'un registre qui contient son adresse. Son utilité, bien que cachée, devient nécessaire pour

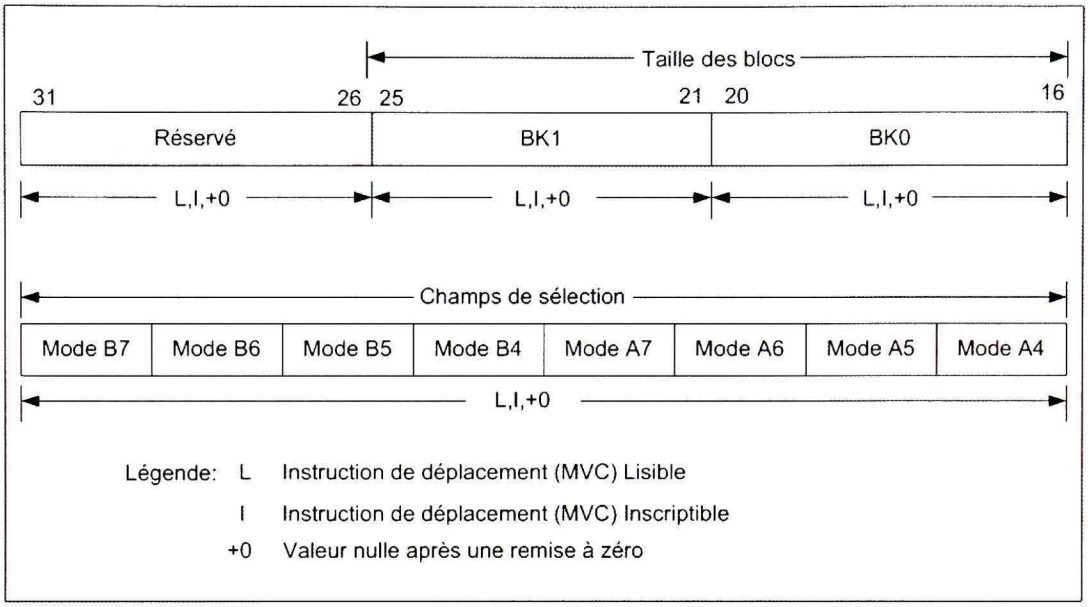


Figure 3.5 Registre de mode d'adressage.

parcourir les cases d'un tableau car il suffit d'incrémenter le registre en question de la taille d'une case pour passer d'une case à une autre ... Afin d'illustrer ce propos, considérons R comme une adresse de registre. R représente l'un des 32 registres $A0$ à $A15$ ou $B0$ à $B15$ qui pointent vers des adresses mémoires. Ces registres sont en réalité des pointeurs. La notation employée dans l'adressage indirect suppose les abréviations suivantes :

a. $*R$ est un registre contenant l'adresse de l'espace mémoire où la valeur de la donnée est stockée.

b. $*R++(d)$ suppose que $*R$ est un registre contenant l'adresse de l'espace mémoire où la valeur de la donnée est stockée. Après que l'adresse mémoire ait été utilisée, la valeur du registre R est postincrémentée de telle sorte que la nouvelle adresse s'obtient en incrémentant l'adresse courante d'une valeur d. Par défaut, la valeur de d est unitaire. À la place d'une double addition, une double soustraction $(--)$ suscite une postdécrémentation de l'adresse courante à l'adresse $R - d$.

c. $*++R(d)$ préconise une préincrémentation de d pour que l'adresse courante soit $R+d$. Un double signe moins suggère une prédécrémentation de l'adresse mémoire pour avoir une adresse courante à l'emplacement $R-d$.

d. $*+R(d)$ implique une préincrémentation de d pour avoir l'adresse courante à $R+d$, sans toutefois modifier le contenu de R. Un raisonnement semblable s'applique pour une prédécrémentation.

Dans le cas du C6711, l'instruction :

$$LDW \quad .L1 \quad *A5++[8],\ A1$$

est une application du mode d'adressage indirect. Ici, LDW est l'instruction $load$ pour un mot de 32 bits, $.L1$ est l'unité qui opère l'instruction demandée sur la voie A et $A1$ et $A5$ sont respectivement le deuxième et le sixième registre de la même voie. La valeur de d vaut huit. L'instruction ainsi écrite effectue un chargement de la valeur du registre A5 dans le registre A1 puis incrémente le pointeur de huit et inscrit la valeur de la nouvelle adresse dans le registre $A5$.

3.3.2 L'adressage circulaire

L'adressage circulaire est utilisé pour créer un tampon circulaire matériellement. Ce tampon est très prisé dans plusieurs algorithmes de traitement numérique du signal à l'exemple du filtrage numérique ou des algorithmes de corrélation qui nécessitent une mise à jour constante des données.

La famille C6000 possède un espace matériel alloué à l'adressage de type circulaire. Ce dernier peut s'utiliser avec un tampon circulaire pour mettre à jour les échantillons en effectuant une rotation des données sans dépassement. Le dépassement survient généralement lorsqu'on réalise une rotation directe sur les données. Pour l'éviter au moment où le pointeur subit une incrémentation lorsqu'il se trouve à l'endroit contenant le dernier

élément du tampon , il faut faire basculer le pointeur automatiquement au premier élément du tampon.

Deux tampons circulaires indépendants sont disponibles par le truchement des banques BK0 et BK1 dans le registre du mode d'adressage tel qu'illustré à la Figure 5 . Le code subséquent est une illustration de l'utilisation de l'adressage circulaire en employant le registre B2 pour inscrire des valeurs dans le registre du mode d'adressage.

$$\begin{array}{lll} MVK & .S2 & 0x0004,\ B2 \\ MVKLH & .S2 & 0x0005,\ B2 \\ MVC & .S2 & B2,\ AMR \end{array}$$

Les deux instructions MVK et $MVKLH$ déplacent respectivement $0x0004$ dans les 16 LSBs du registre B2 et $0x0005$ dans les 16 MSBs de B2. La première instruction inscrit la valeur un dans le troisième bit de l'AMR et annule le reste. Ceci permet la sélection du mode d'adressage circulaire (01) et du registre $A5$ en tant que pointeur du tampon circulaire. La seconde instruction met les bits 16 et 18 à l'unité et les autres 14 MSBs de l'AMR à zéro. Ceci correspond à la valeur du nombre N utilisé pour choisir la taille du tampon dans BK0 . Par exemple, pour une taille du tampon désirée à 128 dans BK0, les 16 MSBs de l'AMR doivent être initialisés de sorte que $2^{N+1} = 128$, soit $0x0006$.

L'instruction de déplacement d'une constante, MVC, est la seule instruction pour laquelle l'AMR et les autres registres de contrôle se voient octroyer un accès. Cette instruction s'exécute seulement sur la voie B en conjonction avec ses unités fonctionnelles et ses registres (Chassaing R., 2002). Une valeur à 32 bits apparaît alors dans le registre B2 et est transférée vers l'AMR via l'instruction MVC.

3.4 La représentation des nombres en point flottant

Contrairement à la représentation en point fixe, celle en point flottant ne requiert pas de facteur d'échelle. Un nombre en point flottant peut être représenté en utilisant la simple précision (en anglais single precision, SP) avec 32 bits, ou la double précision (en anglais double precision, DP) avec 64 bits.

Pour le format SP illustré à la Figure 6a, les nombres sont représentés selon : $-1^s * 2^{(exp-127)} * 1.f$. Dans cette expression s est le bit de signe (bit 31), exp est l'exposant (bits 23 à 30) et f est la mantisse encore dénommée partie fractionnelle (bits 0 à 22). Ainsi, le plus grand nombre représentable selon ce format est $3.4 * 10^{38}$ et le plus petit est $1.175 * 10^{-38}$.

Le format DP illustré à la Figure 6b offre un éventail plus large en représentation numérale car il réserve plus de bits pour l'exposant et la mantisse. Comme il se sert de 64 bits, une paire de registres s'avère nécessaire. Les nombres y sont représentés selon : $-1^s * 2^{(exp-1023)} * 1.f$. Ici, l'exposant s'étend du bit 20 au bit 30, la mantisse occupe un mot entier à 32 bits et les 20 premiers bits du second mot. Ainsi les nombres sont représentables dans ce format selon un intervalle variant de $2.2 * 10^{-308}$ à $1.7 * 10^{308}$

Les instructions se terminant par SP ou DP réfèrent respectivement au format simple et double précision. Certaines instructions en point flottant disponibles dans un processeur C67x ont une latence plus grande que les instructions en point fixe équivalentes (Texas Instruments, 2000). C'est le cas par exemple de la multiplication qui requiert un délai de traitement unitaire en point fixe (MPY) , triple en simple précision ($MPYSP$) et de neuf en double précision ($MPYDP$). Par ailleurs, une instruction SP peut être chargée dans un seul registre tandis que celle DP ne peut l'être que dans une paire de registres tels A1 :A0, A3 :A2 ... Le LSB s'inscrit dans le registre pair et le MSB dans celui impair.

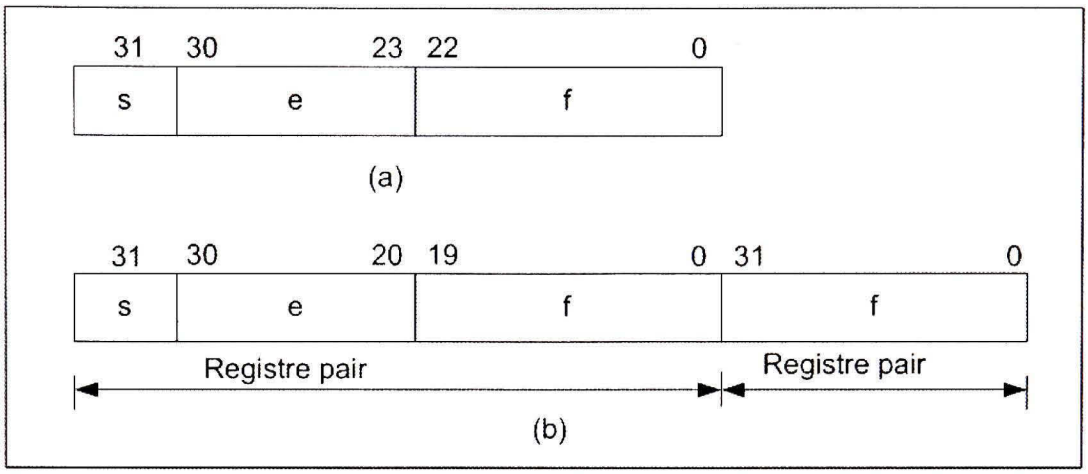


Figure 3.6 Formats de représentation des données : (a) simple précision ; (b) double précision.

La section subséquente fournit quelques détails supplémentaires pertinents sur les instructions. Dans les prochains chapitres, il sera précisément question des instructions en double précision car l'implémentation matérielle des applications est issue du logiciel Matlab® qui se sert de ce format en programmation.

3.5 Le jeu d'instructions

Le jeu d'instructions est un ensemble d'instructions servant à combler le fossé entre le langage machine et les langages de haut-niveau. La nature des instructions et leur modèle d'exécution conditionnent leur exécution. Le modèle d'exécution d'un processeur est le principe par lequel une séquence d'instructions s'exécute tout en préservant la sémantique du programme. Actuellement, la majeure partie des processeurs sont construits autour d'un pipeline. Un pipeline se compose d'étages qui effectuent une opération propre et qui sont séparés entre eux par des verrous afin d'éviter les risques d'interférences entre eux. Un étage passe le résultat de son travail à l'étage suivant en respectant un rythme cadencé par une horloge. Chaque étage du pipeline peut supporter une ou plusieurs instructions qui vont s'exécuter ou non en parallèle. Le pipeline du C6711 possède trois étages illustrés

à la Figure 7 qui sont décrits selon le format double précision ci-dessous. Ces étapes caractérisent le délai nécessaire pour qu'une instruction soit entièrement opérée (Tableau V).

a. la phase de recherche de l'instruction par accès à la mémoire-programme et d'écriture dans le registre d'instruction appelée « fetch ». Elle-même comprend quatre sous-étapes :

   - Génération de l'adresse du programme, en anglais program generate address (PG).
   - Envoi de l'adresse du programme, en anglais program address send (PS).
   - Attente de l'accès au programme, en anglais program access ready wait (PW).
   - Réception du programme chargé , en anglais program fetch packet receive (PR).

b. la phase de décodage de l'instruction et des adresses de l'opérande dite « decode » comportant deux étapes. La première est celle du transfert de l'instruction (en anglais instruction dispatch DPa) durant laquelle les paquets chargés sont transformés en paquets d'exécution. Ces derniers contiennent soit une instruction, soit deux à huit instructions en parallèle. Ils se voient ensuite assigner l'unité fonctionnelle qui leur convient. La seconde quant à elle décode les registres de départ, de destination et les voies associées (A ou B) en vue de l'exécution des instructions dans les unités fonctionnelles.

c. la phase d'exécution de l'opérande et d'écriture du résultat nommée « execute ». Elle comporte dix phases (E1-E10), soit le double des phases en simple précision. Le nombre de phases d'exécution nécessaires pour compléter une instruction dépend de son type. Il peut différer donc d'une instruction à l'autre.

Le processeur C6711 dispose d'une panoplie d'instructions (Texas Instruments, 2000) pour effectuer les opérations de traitement numérique du signal. Les plus importantes

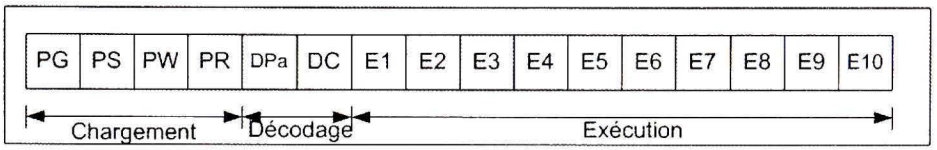


Figure 3.7 Phases du pipeline du C6711.

sont décrites ci-dessous, en relation avec les unités fonctionnelles. Sur les tableaux ci-dessous, le champ *délai* symbolise le nombre de cycles requis pour qu'une intruction soit complétée. Sur les figures présentées les abréviations ont les significations suivantes :

- *regcond* : champ à 3 bits spécifiant un registre conditionnel.
- *null* : test la présence d'une valeur nulle.
- *destreg* : registre de destination.
- *regsrc1* : registre source 1.
- *regsrc2* : registre source 2.
- *csta* : constante a.
- *cstb* : constante b.
- *cst* : constante.
- *pex* : exécution en parallèle.
- *opcode* : champ pour lequel l'opcode spécifie une instruction unique.
- *sel* : sélection de la voie A ou B pour la destination.
- *xvoie* : chemin croisé pour *regsrc2*.

3.5.1 Les instructions liées à l'unité L

La Figure 8 illustre le format des instructions pour cette unité et le tableau V résume quelques unes des opérations possibles au sein de ladite unité.

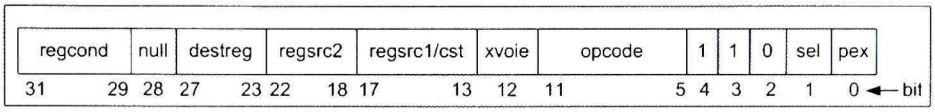


Figure 3.8 Format des instructions dans l'unité L.

Tableau V

Quelques instructions de l'unité L

Instruction	Délai	Description
ABS	1	Valeur absolue avec saturation
ADD(U)	1	Addition non saturée signée (non signée) en point fixe
ADDDP	7	Addition en précision double
ADDSP	4	Addition en précision simple
AND	1	Et logique
CMPEQ	1	Égalité en point fixe
CMPGT(U)	1	Inégalité supérieure signée (non signée) en point fixe
CMPLT(U)	1	Inégalité inférieure signée (non signée) en point fixe
DPTRUNC	4	Conversion précision double vers entier avec troncature
MV	1	Déplacement d'un registre à l'autre
NEG	1	Négation en point fixe
NOT	1	Non logique
OR	1	Ou logique
SADD	1	Addition d'entiers avec saturation
SAT	1	Saturation en point fixe
SPTRUNC	4	Conversion précision simple vers entier avec troncature
SUB(U)	1	Soustraction non saturée signée (non signée) en point fixe
SUBDP	7	Soustraction en précision double
SUBSP	4	Soustraction en précision simple
XOR	1	Ou exclusif
ZERO	1	Mise à zero

3.5.2 Les instructions liées à l'unité M

La Figure 9 présente le format des instructions de l'unité M et le tableau VI illustre certaines instructions qui peuvent y avoir lieu.

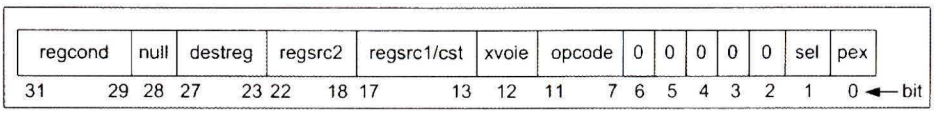


Figure 3.9 Format des instructions dans l'unité M.

Tableau VI

Quelques instructions de l'unité M

Instruction	Délai	Description
MPY(U)	2	Multiplication signée (non signée) 16 bits LSB × 16 bits LSB
MPYDP	10	Multiplication en précision double
MPYH(U)	2	Multiplication signée (non signée) 16 bits MSB × 16 bits MSB
MPYHL(U)	2	Multiplication signée (non signée) 16 bits MSB × 16 bits LSB
MPYI	9	Multiplication sur 32 bits avec résultat sur moins de 32 bits
MPYID	10	Multiplication sur 32 bits avec résultat sur 64 bits
MPYLH	2	Multiplication signée (non signée) 16 bits LSB × 16 bits MSB
MPYSP	4	Multiplication en précision simple

3.5.3 Les instructions liées à l'unité D

La Figure 10 illustre le format des instructions présentes dans l'unité D et le tableau VII, quelques unes de ses instructions.

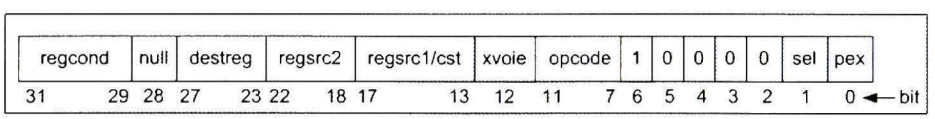


Figure 3.10 Format des instructions dans l'unité D.

3.5.4 Les instructions liées à l'unité S

La Figure 11 met en évidence le format des instructions de l'unité S et le tableau VIII fait cas des possibles instructions peuvant s'y prêter.

Tableau VII

Quelques instructions de l'unité D

Instruction	Délai	Description
ADD(U)	1	Addition non saturée signée (non signée) en point fixe
ADDAB	1	Addition en bytes
ADDAH	1	Addition sur 16 bits
ADDAW	1	Addition sur 32 bits
LDB	5	Chargement en bytes
LDH	5	Chargement sur 16 bits
LDDW	5	Chargement sur 32 bits d'un mot en précision double
LDW	5	Chargement sur 32 bits
MV	1	Déplacement d'une registre à l'autre

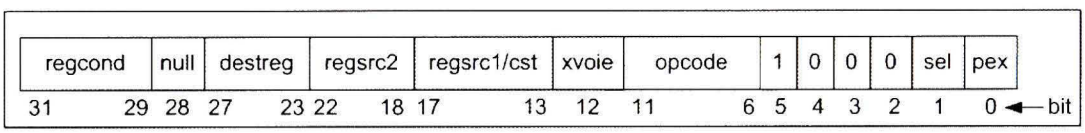


Figure 3.11 Format des instructions dans l'unité S.

3.6 Les techniques d'optimisation

De nombreuses techniques d'optimisation servent à accroître l'efficacité d'un programme implémenté en langage C sans perdre en termes de performances. Elles sont ci-dessous décrites.

3.6.1 Les instructions en parallèle

La parallélisation des instructions optimise le délai de traitement dans la mesure où elle permet à plusieurs instructions de s'exécuter simultanément c'est-à-dire en parallèle (Chassaing, 2002). Il faut alors que chacune d'elles emprunte des voies différentes pour rendre la technique possible. La Figure 12a présente un code non optimisé tandis que la Figure 12b présente l'optimisation équivalente, obtenue grâce à la parallélisation des instructions

Tableau VIII

Quelques instructions de l'unité S

Instruction	Délai	Description
ABSDP	2	Valeur absolue des nombres en précision double
ABSSP	1	Valeur absolue des nombres en précision simple
ADD	1	Addition non saturée signée en point fixe
ADDK	1	Addition signée en utilisant une constante sur 16 bits
AND	1	Et logique
B	6	Branchement
CMPEQDP	2	Égalité en précision double
CMPEQSP	1	Égalité en précision simple
CMPGTDP	2	Inégalité supérieure en précision double
CMPGTSP	1	Inégalité supérieure en précision simple
CMPLTDP	2	Inégalité inférieure en précision double
CMPLTSP	1	Inégalité inférieure en précision simple
EXT	1	Extraction et extension de signe
MV	1	Déplacement d'une registre à l'autre
MVK	1	Déplacement d'une constante de 16 bits signée et extension de signe
MVKH	1	Déplacement d'une constante de 16 bits dans les MSB d'un registre
NOT	1	Non logique
OR	1	Ou logique
RCPDP	2	Approximation réciproque en précision double (division)
RCPSP	1	Approximation réciproque en précision simple (division)
RSQRDP	2	Approximation réciproque de la racine carrée en précision double
RSQRSP	1	Approximation réciproque de la racine carrée en précision simple
SPDP	2	Conversion précision double en précision simple
SPINT	2	Conversion précision simple à entier
SET	1	Mise à un
SHL(R)	1	Décalage à gauche (droite)
SSHL	1	Décalage à gauche avec saturation
STB	1	Stockage d'une valeur en bytes
STH	1	Stockage d'une valeur sur 16 bits
STW	1	Stockage d'une valeur sur 32 bits
SUB(U)	1	Soustraction non saturée signée (non signée) en point fixe
XOR	1	Ou exclusif
ZERO	1	Mise à zero

d'initialisation des registres $A7$ et $B7$ sur les voies A et B respectivement. Cette dernière est représentée par le symbole $||$.

```
MVK    .S1   50, A1    ; Initialisation du registre A1
ZERO   .L1   A7        ; Initialisation du registre A7
ZERO   .L1   B7        ; Initialisation du registre B7
                (a)

MVK    .S1   50, A1    ; Initialisation du registre A1
|| ZERO .L1   A7        ; Initialisation du registre A7
|| ZERO .L2   B7        ; Initialisation du registre B7
                (b)
```

Figure 3.12 Optimisation par parallélisation : (a) cas non optimisé ; (b) cas optimisé.

3.6.2 La suppression des NOPs

Le terme NOP dérive de l'expression anglaise "Not Optimized Properly".Cette méthode consiste à combler des intervalles de temps pendant lesquels aucune opération ne s'exécute par des latences d'instructions plus lentes à opérer tel qu'illustré à la Figure 13 (Chassaing, 2002). L'optimisation dans ce cas réduit le nombre de NOPs de neuf à trois.

```
loop:
      LDH    .D1   *A8++, A1   ; Chargement de la valeur pointée au registre A8 dans le registre A1
|| LDH    .D2   *B9++, B2   ; Chargement de la valeur pointée au registre B9 dans le registre B2
      NOP           4         ; Pas d'opération pendant 4 cycles
      MPY    .M1x  A1,B2,A3   ; Multiplication du registre A1 par B2 et transfert du résultat dans A3
      NOP                     ; Pas d'opération pendant 1 cycle
      ADD    .L1   A3,A4,A4   ; Addition des registres A3 et A4 et transfert du résultat dans A4
      SUB           B0,1,B0   ; Décrémentation du registre B0
[B0]  B             loop      ; Branchement
      NOP           5         ; Pas d'opération pendant 5 cycles
                (a)

loop:
      LDH    .D1   *A8++, A1   ; Chargement de la valeur pointée au registre A8 dans le registre A1
|| LDH    .D2   *B9++, B2   ; Chargement de la valeur pointée au registre B9 dans le registre B2
      SUB           B0,1,B0   ; Décrémentation du registre B0
[B0]  B             loop      ; Branchement
      NOP           2         ; Pas d'opération pendant 2 cycles
      MPY    .M1x  A1,B2,A3   ; Multiplication du registre A1 par B2 et transfert du résultat dans A3
      NOP                     ; Pas d'opération pendant 1 cycle
      ADD    .L1   A3,A4,A4   ; Addition des registres A3 et A4 et transfert du résultat dans A4
                (b)
```

Figure 3.13 Optimisation par suppression des nops : (a) cas non optimisé ; (b) cas optimisé.

3.6.3 L'optimisation par déroulage de boucle

Le déroulage de boucle (J. C. Huang et T. Leng. , 1999) est une technique ancienne dont le principe, très simple, consiste à dupliquer un certain nombre de fois l'écriture du contenu d'une boucle. D'une part, l'un des objectifs est de réduire le coût des opérations ayant en

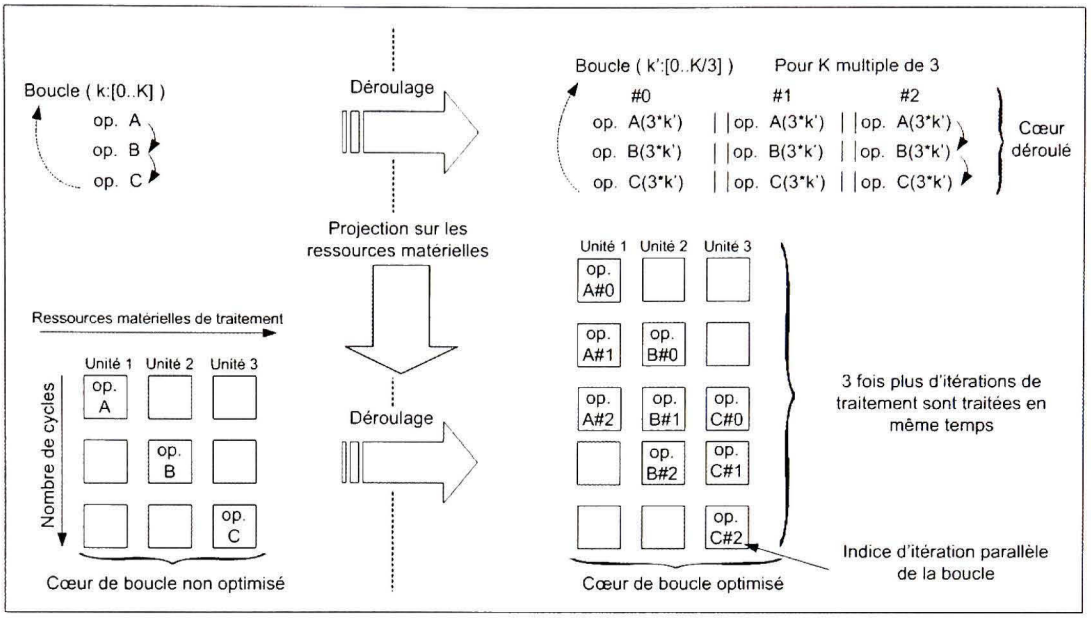


Figure 3.14 Principe du déroulage de boucle.

charge la gestion de la boucle en l'absence de contrôleur de boucle matériel par exemple la décrémentation du compteur de boucle et le branchement conditionné par cette même variable. D'autre part, l'autre est de suggérer une meilleure utilisation des ressources parallèles avec l'opportunité généralement laissée au compilateur ou au processeur de mieux entrelacer dynamiquement l'exécution temporelle des opérations du coeur de boucle déroulé. Sans cette approche explicite, l'entrelacement des opérations reste limité par la rupture du séquencement de l'exécution liée à l'opération de branchement conditionnel. La latence d'une itération de boucle peut ainsi être réduite et le nombre total d'itérations de la boucle s'en trouve amenuisé comme l'illustre la Figure 14. Sur cette figure, K représente le nombre d'itérations de boucle et *op*, une opération quelconque qui peut être une addition, une multiplication ...

3.6.4 Le pipeline logiciel

L'article de Lam introduit le pipeline logiciel (en anglais « software pipelining »), lié aux architectures VLIW (Lam, 1988). Cette technique vise l'optimisation des traitements itératifs définis au sein de boucles logicielles tout en favorisant une meilleure utilisation des ressources des processeurs. Elle permet un gain de performances très important pour la famille C6X. Dans ce cas précis, l'objectif du pipeline logiciel consiste alors à utiliser toutes les huit unités fonctionnelles mises en disposition, en un cycle d'horloge. Pour ce faire, il faut désynchroniser l'exécution des opérations de la boucle de telle sorte que plusieurs parties d'une itération de boucle évoluent en parallèle. Comme dans le cas du déroulage de boucle représenté à la Figure 14, il en résulte un entrelacement des opérations portant sur différentes itérations de la boucle initiale.

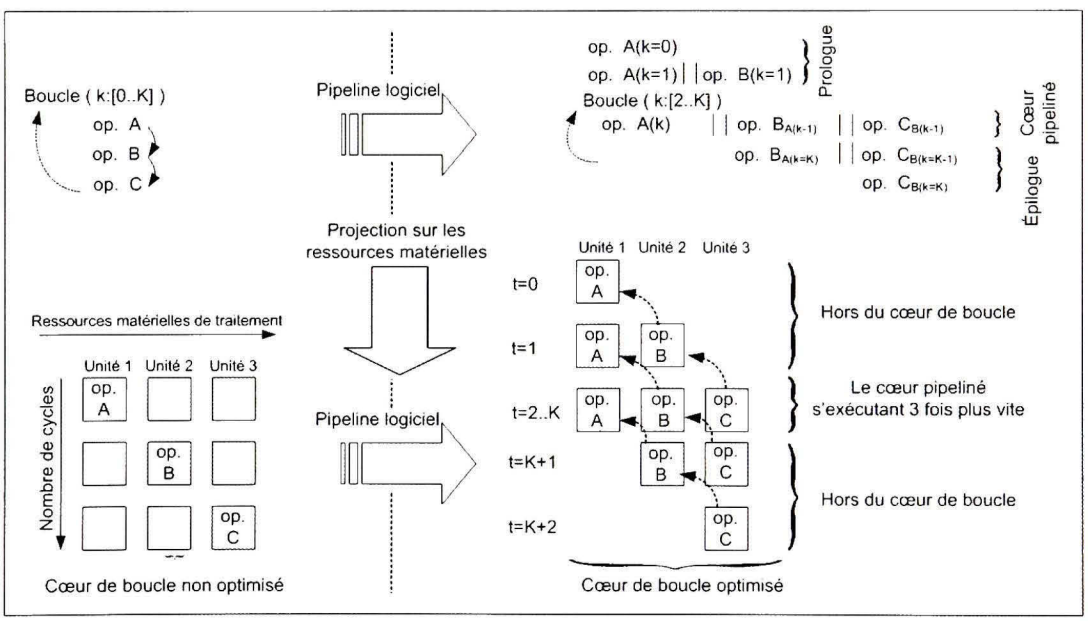


Figure 3.15 Principe du pipeline logiciel.

La description de la Figure 15 illustre le principe général du pipeline logiciel, en l'occurence la maximisation de l'utilisation temporelle des unités de calcul. Le coeur de boucle traite différentes itérations logiques en parallèle. Le traitement nécessite une étape

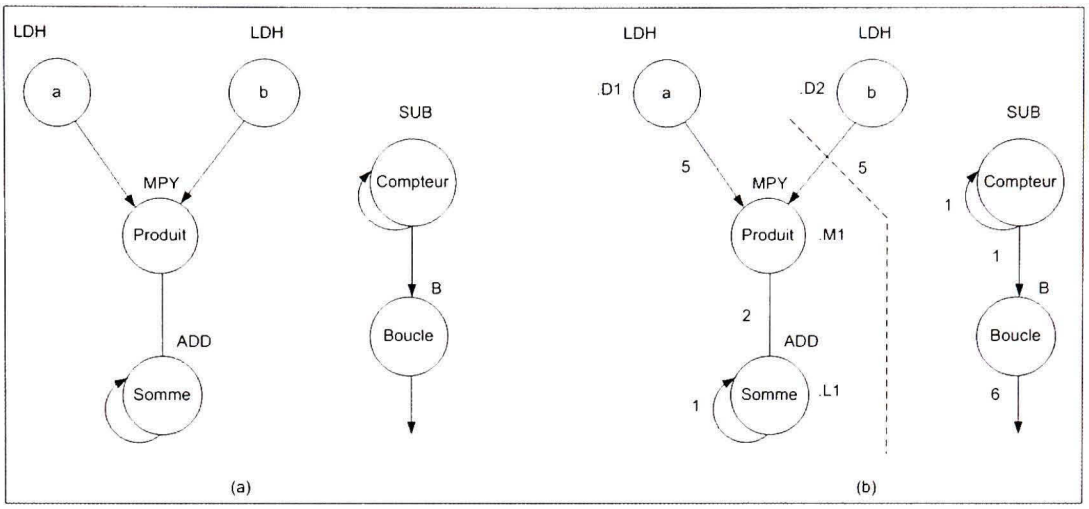


Figure 3.16 Graphe de dépendance des données (mult-add) : (a) équivalent logiciel ; (b) équivalent matériel.

d'initialisation progressive appelée prologue contenant les informations requises afin de construire une boucle. Symétriquement, une partie du code vient graduellement terminer le traitement. Il s'agit de l'épilogue qui rassemble les informations utiles pour clore les itérations. En sandwich entre le prologue et l'épilogue, se trouve le coeur de boucle pipeliné (en anglais « loop-kernel »). Dans cette boucle, toutes les instructions sont exécutées en parallèle et en un cycle. En comparaison avec le déroulage de boucle, le pipeline logiciel va plus loin dans le réordonnancement des opérations du coeur itéré dans la mesure où les étapes d'initialisation et de clôture de la boucle sont déportées à l'extérieur de celle-ci.

La méthode du pipeline logiciel revient conceptuellement à replier le graphe de dépendance des données du coeur de boucle sur lui-même. La Figure 16 est un exemple de graphe de dépendance de données pour un multiplieur-additionneur implémenté en format point fixe. Le graphe de dépendance logicielle (Figure 16a) précise les opérations constitutives de l'algorithme en situation de dépendance de données. Ce graphe permet la déduction du graphe de dépendance matérielle (Figure 16b) où il est possible de distinguer la pondération de chaque arc de dépendance avec le délai des opérations.

Cette approche permet alors d'obtenir une première estimation de la durée globale d'une itération de traitement à optimiser plus tard avec le pipeline logiciel. L'étape suivante suggère de déterminer l'intervalle minimum d'itérations (IMI) qui permet d'établir le délai minimum du coeur de boucle replié indépendamment des contraintes des ressources. Ce seuil traduit le délai minimum nécessaire au lancement d'une nouvelle itération de manière à ce que l'interdépendance temporelle des données soit satisfaite aux noeuds des cycles. Dans l'exemple, l'exécution de l'instruction d'addition est subordonnée au résultat du produit précédent. Ceci empêche de débuter un nouvel ensemble d'opérations désynchronisées tant que le délai d'exécution de l'opération de multiplication n'est pas respectée. La valeur de l'IMI se déduit alors de la durée maximale des délais présents dans le graphe. Elle s'obtient par sommation de la durée des arcs reliant les noeuds qui composent chaque itération. Ici, cette durée vaut huit et se déduit aisément de la Figure 17. La mesure de l'IMI apparaît comme une borne théorique inférieure.

La technique de l'ordonnancement (en anglais « scheduling ») supplante celle du graphe de dépendance et du calcul de l'IMI lorsque survient le moment de séquencer les opérations du graphe de dépendance replié en tenant compte des instructions du plus long chemin de la chaîne depuis le prologue jusqu'à l'épilogue. Le Tableau IX en fait cas. Sur cette table, on y distingue les huit unités fonctionnelles dont le processeur dispose. Les instructions employées sont : *ldw* pour le chargement d'un mot de 32 bits, *mpy*, *mpyh*, *add* et *sub* respectivement pour la multiplication des 16 bits de poids faibles, celle des 16 bits de poids fort, l'addition et la soustraction en point fixe et enfin, *b* pour le branchement. Également, le tableau met en évidence les sept premiers cycles constituant le prologue et le cycle 8 qui est le coeur de boucle. En fonction du nombre d'itérations N désiré, ce dernier s'étend jusqu'au cycle $N-1$ et l'épilogue survient au cycle N.

Enfin, pour éviter d'être confronté à une gestion spécifique lorsque la quantité de données traitées est inférieure au nombre d'itérations évoluant en parallèle, il est possible d'en imposer une quantité minimale. Le pipeline logiciel peut aussi inclure une optimisation de

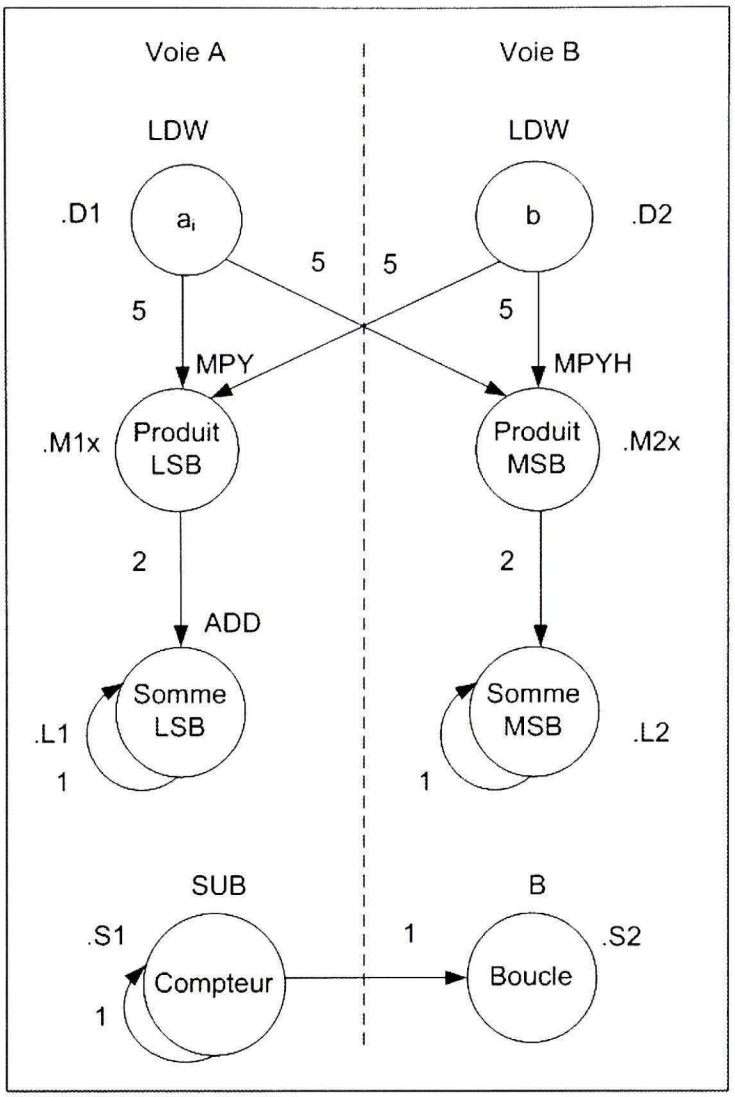


Figure 3.17 Graphe de dépendance des données replié d'un multiplieur-additionneur.

la longueur du prologue et de l'épilogue en introduisant l'exécution conditionnelle totale ou partielle des opérateurs de traitement.

3.6.4.1 L'utilisation de la table de transcodage

L'utilisation de la table de transcodage (en anglais look-up table, LUT) permet de stocker un certain nombre de calculs récurrents évalués une seule fois dans une phase d'initialisa-

Tableau IX

Ordonnancement d'un multiplieur accumulateur après pipeline logiciel pour une implémentation en point fixe dans un processeur C6x

Unités	1	2	3	4	5	6	7	8
.D1	ldw	ldw	ldw	ldw	ldw	ldw	ldw	ldw
.D2	ldw	ldw	ldw	ldw	ldw	ldw	ldw	ldw
.M1						mpy	mpy	mpy
.M2						mpyh	mpyh	mpyh
.L1								add
.L2								add
.S1		sub	sub	sub	sub	sub	sub	sub
.S2			b	b	b	b	b	b

tion. C'est le cas dans l'algorithme de la FFT, pour le calcul des facteurs de phase (Kuo et al., 2006). Un indice de la table s'associe à une valeur pré-calculée. Cette optimisation, simple à mettre en oeuvre, permet d'améliorer les performances du processeur en termes de rapidité, dans la mesure où les coûts d'accès à la table sont plus réduits que le coût des calculs correspondants exécutés dynamiquement. Cependant, cette technique présente l'inconvénient de nécessiter une quantité de mémoire supplémentaire pour le stockage des valeurs pré-calculées de la table.

3.7 Conclusion

Ce chapitre a présenté l'architecture du processeur C6711, l'organisation de sa mémoire, ses modes d'adressage, son jeu d'instructions, ses techniques d'optimisation et les interruptions dont il dispose. Il est à propos de présenter du côté application, l'équivalence des opérations matérielles du processeur. À cet effet, le reste de ce document s'attarde sur la principale contribution de ce mémoire en épiloguant sur la méthodologie développée pour extraire les primitives les plus récurrentes en télécommunications et la librairie d'opérations qui en a été déduite.

CHAPITRE 4

LA MÉTHODOLOGIE D'EXTRACTION DES PRIMITIVES

Dans le premier chapitre, il a été question de présenter les systèmes de communication point à multipoints et les systèmes de communication à antennes multiples, deux grands ensembles en télécommunications. Le second chapitre quant à lui s'évertue à discourir sur trois des applications faisant partie des grands groupes précédemment introduits. L'application ZF-SQRD inaugure le bal. Elle se situe au récepteur des stations multi-antennes. Sur les pas du ZF-SQRD, suit le récepteur Rake performant dans un contexte de propagation à trajets multiples. Ce type de récepteur s'emploie pour communiquer d'un point vers plusieurs autres. Finalement, la FFT s'inscrit en troisième place avec l'algorithme radix 2 à décimation temporelle. C'est un algorithme générique prisé dans les systèmes point à multipoints OFDMA. Le troisième chapitre met l'accent sur la plateforme d'intérêt, le TMS320C6711. Il y est présenté comme étant un processeur à virgule flottante qui possède un attrait pour des applications audio dont il maximise l'efficacité et les performances.

Le présent chapitre met en lumière l'une des principales contributions de ce mémoire, soit une méthodologie d'extraction de blocs élémentaires fréquents dans le domaine des télécommunications. Cette méthodologie préconise une compréhension bien cernée de l'architecture du processeur étudié. Celle-ci est nécessaire afin de répertorier les opérations possibles du point de vue matériel pour implémenter un algorithme donné. Ces opérations, dites primitives structurelles, se différencient légèrement d'un processeur à l'autre. C'est la raison pour laquelle la maîtrise de la plateforme d'intérêt est un atout incontournable. Parallèlement à cette maîtrise, la compréhension des algorithmes choisis du point de vue applications ne vient pas en défaut. Tout au contraire, elle sert pour bâtir une librairie d'opérations usuelles en télécommunications nommées primitives fonctionnelles. L'élaboration de cette librairie permet de créer un pont entre les primitives structurelles et celles fonctionnelles. Elle est l'élément-clé pour établir les équations de prédiction du

temps CPU nécessaire pour une bonne implémentation. Enfin, l'établissement d'une procédure de vérification de cette prédiction vient donner force à la pertinence de la librairie et couronne la fin de ce chapitre.

4.1 La compréhension de l'architecture du processeur

La première étape requise pour faire une bonne prédiction est la compréhension de l'architecture du DSP afin d'avoir une idée du type d'instructions opérées par le processeur (primitives structurelles). Cette compréhension sert ensuite de socle pour bâtir le plan d'adressage nécessaire pour les implémentations. Le TMS320C6711 est un processeur à point flottant de type VLIW (Texas Instruments, 2000) tel que présenté au chapitre 3. Le plan d'adressage décrit à la section 3.2 de ce chapitre contient les adresses des registres, des différents blocs de mémoire internes et externes (RAM ou ROM) et des données sur les bus sériels rencontrés au sein de la plateforme. En vue de l'implémentation, un plan d'adressage similaire à celui-là a été bâti. D'une part il configure la mémoire (Tableau X). Cette dernière se répartit en trois types : 4MB d'EPROM (Erasable Programmable Read Only Memory), 16 MB de SDRAM (Synchronous Dynamic Random Access Memory) et 64 KB d'IRAM (Internal Random Access Memory). D'autre part, il attribue des adresses aux sections du compilateur (Tableau XI) qui sont des espaces mémoires réservés dont les rôles sont prédéfinis afin de faciliter la répartition des différentes parties du code à implémenter dans le processeur. Il y en a plusieurs. Parmi elles, il y a celles dont la mémoire ne peut être qu'uniquement lue, celles dont la mémoire peut être lue et où il est possible d'écrire, celles dont le contenu demeure après une remise à zéro et celles dont la mémoire peut être initialisée ou non. La description de chacune d'elles suit :

- *.text* : Cette section est réservée au code.
- *.cio* : Les tampons des fonctions « stdio » définissent cette section.
- *.stack* : Les variables locales contenues dans la pile se voient allouer cette section .

- *.far* : Cette section est occupée par les variables globales statiques déclarées "éloignées" (en anglais "far").
- *.bss* : Les variables globales statiques composent cette section.
- *.switch* : Les tables pour les instructions impliquant un choix (*if*, *case* ...) sont la raison d'être de cette section.
- *.cinit* : L'initialisation des variables globales statiques s'approprie la section mentionnée ci-contre.
- *.const* : Les parties littérales des chaînes (« string ») constituent cette section.
- *.sysmem* : Cette section est réservée aux espaces mémoires des fonctions « malloc ».

Tableau X

Plan d'adressage pour configurer la mémoire du processeur

Nom	Origine	Longueur
IRAM	0000 0000	0001 0000
EPROM	0001 0000	0040 0000
SDRAM	8000 0000	0100 0000

Tableau XI

Plan d'adressage des sections du processeur

Sections de sortie	Adresse d'origine	Longueur	Type de mémoire
.sysmem	8000 0000	0000 0400	Non initialisée
.cio	8000 0400	0000 0120	Non initialisée
.stack	variable	variable	Non initialisée
.bss	variable	0000 0040	Non initialisée
.far	variable	variable	Non initialisée
.cinit	variable	variable	Initialisée
.text	variable	variable	Initialisée
.const	variable	variable	Initialisée
.switch	variable	variable	Initialisée

4.2 La compréhension de l'algorithme étudié

La compréhension de l'algorithme conduit à sa décomposition en blocs d'opérations de base nommées primitives fonctionnelles. Cette décomposition permet d'élaborer une librairie de primitives caractérisant les différentes opérations effectuées dans les algorithmes de télécommunication. On y distingue d'une part les opérations mathématiques effectuées dans l'ALU et les opérations de transferts de mémoire. Le premier groupe comporte des entrées non matricielles réelles et complexes, et, des entrées matricielles réelles et complexes. Le second quant à lui, ne fait pas de distinction entre les élements réels et matriciels. Pour chacun d'eux cependant, la granularité augmente graduellement à mesure que les opérations à accomplir s'alourdissent. Ces primitives fonctionnelles sont ensuite reliées aux primitives structurelles, inhérentes à l'architecture de la plateforme utilisée.

Une fois la décomposition des différents algorithmes en primitives fonctionnelles, un lien est établi entre ces dernières et leurs équivalences matérielles, les primitives structurelles pour en faire une librairie réutilisable pour n'importe quelle application reliée aux télécommunications. Les prochaines lignes l'exposent.

Par ailleurs, pour les applications choisies, les primitives structurelles les plus récurrentes que ce soit en virgule fixe ou en virgule flottante (simple précision ou double précision) sont l'addition, la multiplication, la négation qui est l'opération transformant un nombre positif en un nombre négatif et vice-versa, le déplacement, le chargement et le stockage des données. En ce qui concerne la notation en point fixe, l'addition, la négation, le déplacement et le stockage des données requièrent un cycle d'horloge pour leur exécution, la multiplication, deux cycles, et le chargement des données, cinq. En virgule flottante, les trois opérations impliquant des transferts mémoires, soit le déplacement, le chargement et le stockage des données, ne montrent aucune variabilité en termes de nombre de cycles par rapport au format point fixe. Par contre, l'addition voit son nombre de cycles quadrupler en précision simple et septupler en double précision par rapport au format point fixe. Pour

ce qui est de la multiplication, le temps CPU nécessaire afin de la réaliser est le même que celui de l'addition en précision simple mais vaut dix cycles en précision double. Le Tableau XII résume les équivalences entre les primitives structurelles fréquemment rencontrées dans les applications choisies et le nombre de cycles nécessaire pour les opérer (Texas Instruments, 2000) dans le cas d'une implémentation à virgule flottante. Les opérations de transfert mémoire sont indépendants du type de représentation choisi. Pour la description de ces primitives, se référer à la section 3.5.1 du chapitre 3.

Tableau XII

Équivalences entre primitives structurelles et nombre de cycles d'horloge

Opérations	Nombre de cycles
MPYSP	4
ADDSP	4
MPYDP	10
ADDDP	7
NEG	1
MV	1
LDDW, LDW,LDB, ...	5
STW,STH,STB, ...	1

Les blocs de primitives établis se séparent en trois grands groupes : les opérations mathématiques réelles, les opérations mathématiques complexes et les opérations de transfert mémoire, n'impliquant aucun calcul arithmétique. Ces blocs sont mis en relation avec les primitives structurelles équivalentes. La librairie développée n'a aucunément la prétention d'être exhaustive. Elle pourra être complétée à souhait par l'étude d'autres applications. En outre, le processus de validation des résultats présenté plus loin emploie le logiciel Matlab®. Celui-ci se sert d'une représentation des nombres en point flottant à précision double. C'est la raison pour laquelle les instructions matérielles équivalentes et les accès mémoires de ces dernières (Tableau XII) sont comptabilisés en précision double dans ce qui suit. Par ailleurs, la librairie annoncée ci-dessous présente uniquement une équivalence entre les opérations du côté application et celles du côté matériel sans tenir compte

des accès mémoires. Le nombre de cycles total prédit en tient compte dans le prochain chapitre.

Parmi les primitives fonctionnelles réelles, se distinguent :

- *L'addition* : Pour deux entrées matricielles de dimensions (X, Y) , elle équivaut à XY additions matérielles.
- *La multiplication* : On en distingue deux types. Le premier est la multiplication d'une constante par une matrice (I, J). Son équivalent matériel correspond à IJ multiplications. Le second est le produit de deux matrices (X, Y) et (Y, J) qui donne en termes de primitives structurelles XYJ multiplications et $(Y - 1)XJ$ additions.
- *La division* : La technique d'approximation réciproque permet de la réaliser. En considérant un contexte à virgule flottante avec précision double tel que précisé plus haut, au départ, l'instruction d'approximation réciproque *RCPDP*, fournit la valeur de l'exposant convenable à la représentation en point flottant avec une précision de huit bits à la mantisse. Par la suite, cet estimé peut être utilisé comme semence pour un algorithme qui calculela la reciproque avec une meilleure précision. Il s'agit de l'algorithme Newton-Rhapson dans ce cas (Liang, 2001). Celui-ci permet par ricochet l'extension de la précision de la mantisse selon l'équation 4.1 tirée de (Texas Instruments, 2000) :

$$x(n+1) = x(n) \times (2 - v \times x(n)) \tag{4.1}$$

Avec v qui représente le nombre dont la réciproque est à déterminer à la n^{ieme} itération.

Le terme $x(0)$, la semence de l'algorithme, s'obtient par l'instruction *RCPDP*. Après chaque itération, la précision des calculs double (Texas Instruments, 2000). Ainsi, par exemple, pour une itération ayant une précision de 16 bits à la mantisse, celle-ci

passe à 32 bits après la seconde itération et à la troisième, la précision est maximale à 52 bits.

La division est une opération à minimiser matériellement car onéreuse en ressources. Dans les algorithmes étudiés, ses primitives structurelles regroupent une instruction de calcul d'approximation réciproque ($RCPDP$ pour la précision double), deux multiplications et une soustraction matérielles par itération.

- *L'arg* : Cette primitive calcule le carré des normes des lignes (ou colonnes) d'un vecteur (X, Y). Matériellement, cela équivaut à XY multiplications et $Y(X-1)$ additions.
- *L'arg minimal* permet de déterminer la valeur minimale des résultats de l'arg. Elle dispose des mêmes primitives que cette dernière auxquelles se rajoutent XY comparaisons matérielles.
- *L'accumulateur (Mac)* : Il y en a de deux ordres. Le premier est un multiplieur-accumulateur d'un entier par une matrice de taille (X, Y). En relation avec le processeur, on trouve XY multiplications et additions. Par la suite, pour un multiplieur-accumulateur de deux matrices (X, Y) et (Y, J), se déduisent structurellement XYJ multiplications et additions.
- *La valeur absolue* : Elle se mesure à XY opérations ABS matériellement, pour une matrice de taille (X, Y) en entrée.
- *Le carré de la valeur absolue* : En plus des opérations matérielles de calcul de la valeur absolue, elle fait intervenir XY multiplications.
- *La racine carrée* : L'équivalent matériel de cette primitive est très semblable à celle de la division et se sert tout à son exemple de la technique d'approximation réciproque qui s'applique cette fois-ci à la détermination de la racine carrée. Elle fournit la valeur de l'exposant convenable à la représentation en point flottant avec une précision de huit bits à la mantisse. Celle-ci sert de semence à l'algorithme Newton-Rhapson (Liang, 2001) représenté par l'équation 4.2 tirée de (Texas Instruments,

2000) :

$$x(n+1) = x(n) \times (1.5 - (v/2) \times x(n) \times x(n)) \tag{4.2}$$

Avec v qui représente le nombre dont la réciproque est à déterminer à la n^{ieme} itération.

L'instruction qui initie le calcul de la racine carrée en virgule flottante est *RSQRDP* et fournit la semence $x(0)$ de l'algorithme. La précision des calculs double après chaque itération (Texas Instruments, 2000) comme dans le cas de la division.

Le calcul de racine carrée n'est pas très fréquente. Les ressources dont elle a besoin, se comparent à trois multiplications, une soustraction et une division par itération.

- *L'intégrateur* : Cette primitive somme les valeurs d'un vecteur (lignes ou colonnes) avec un pas correspondant à l'intervalle d'intégration N. Ainsi, pour un vecteur d'entrée de taille (X, Y), le vecteur de sortie équivalent est un vecteur de taille $(\frac{x}{N}, Y)$ et le calcul pour l'obtenir nécessite $Y(X-1)$ additions.
- *L'énergie d'un signal* : Son calcul se fait en effectuant le produit de la transposée d'un signal par lui-même. Pour une matrice de taille (X, Y) sont nécessaires XY multiplications, $XY-1$ additions et XY déplacements notés *MV* pour « move » en anglais.
- *La combinaison linéaire maximale* : Très prisée en contexte multi-trajets (récepteur Rake), cette opération est une étape importante lors de la détection des signaux en ceci qu'elle permet la sommation des différents trajets d'un signal affectés de leurs poids respectifs. Ces poids sont proportionnels à l'importance des évanouissements que subissent chaque trajet dans le canal. Cette sommation converge vers un seul signal précédent la prise de décision finale par quantification souple ou dure. Les opérations matérielles requises se dénombrent par $X(2Y-1)$ multiplications, $(Y-1)X$ additions et XY déplacements pour un vecteur d'entrée (X, Y) .

- *La FFT-Radix 2* : Deux considérations pour les équivalences matérielles ont été établies en fonction des résultats obtenus avec le processeur, illustrés dans le chapitre à venir. D'une part, apparaît l'éventualité excluant toute optimisation de l'algorithme, c'est-à-dire que le processeur effectue les produits des échantillons d'entrée de taille N par les facteurs de phase unitaires même si inutiles. À cette dernière s'associent matériellement $4Nlog_{10}(N)$ multiplications et additions réelles. D'autre part, lorsqu'il y a optimisation, autrement dit, lorsque le compilateur détecte la présence de facteurs de phase unitaires et agit comme tel, c'est-à-dire, qu'il n'effectue pas de multiplications surnuméraires, alors, les opérations matérielles sont réduites à $2Nlog_{10}(N)$ multiplications réelles et $3Nlog_{10}(N)$ additions réelles.

Les opérations complexes sont les suivantes :

- *L'addition* : Pour deux entrées matricielles de dimensions (X, Y) , elle équivaut à $2XY$ additions matérielles.
- *La multiplication* : On en distingue trois types. Le premier est la multiplication d'une constante par une matrice complexe (I, J). Son équivalent matériel correspond à $2IJ$ multiplications. Le second est le produit d'une matrice de réels (X, Y) par une matrice de complexes (Y, J) qui donne en termes de primitives structurelles $2XYJ$ multiplications et $2(Y-1)XJ$ additions. Enfin, le troisième est le produit de deux matrices de complexes (X, Y) et (Y, J) résultant en $4XYJ$ multiplications et $(3Y-1)XJ$ additions matérielles.
- *La division* d'une matrice de complexes (X, Y) par une constante c équivaut à la multiplication de la dite matrice par la constante $1/c$. Cette primitive fait ainsi intervenir une division réelle pour la transformation de c à $1/c$ et XY multiplications d'une constante par une matrice de complexes (voir plus haut).

- *L'arg* : Cette primitive avec en entrée un vecteur de complexes (X, Y) joue le même rôle que dans le cas d'une matrice de réels. Matériellement, elle équivaut à $2XY$ multiplications et $Y(2X - 1)$ additions.
- *L'arg minimal* permet de déterminer la valeur minimale des résultats de l'arg. Elle dispose des mêmes primitives que cette dernière auxquelles se rajoutent XY comparaisons matérielles.
- *L'accumulateur (Mac)* : Il y en a de trois ordres. Le premier est un multiplieur-accumulateur d'une valeur réelle par une matrice de taille (X, Y). En relation avec le processeur, on trouve $2XY$ multiplications et additions. Par suite, pour un multiplieur-accumulateur d'une matrice réelle (X, Y) et d'une matrice complexe (Y, J) se déduisent structurellement $2XYJ$ multiplications et additions. Finalement, pour deux matrices de complexes (X, Y) et (Y, J), s'obtiennent $4XYJ$ multiplications et $(3Y + 1)XJ$ additions.
- *La valeur absolue d'un vecteur* : Encore l'équivalent de la norme, elle se mesure à XY opérations de racine carrée, $2XY$ multiplications et XY additions matérielles, pour une matrice de taille (X, Y) en entrée.
- *Le carré de la valeur absolue d'un vecteur* se sert du même nombre d'opérations que la valeur absolue à l'exception de celles impliquant la racine carrée.
- *L'intégrateur* : Fort de la définition faite dans le cas des entrées réelles, il apparaît que son calcul nécessite $2Y(X - 1)$ additions pour un vecteur d'entrée de taille (X, Y).
- *L'énergie d'un signal* : Son calcul, pour une matrice de taille (X, Y) nécessite $2XY$ multiplications, $(2X - 1)Y$ additions et XY déplacements.
- *La combinaison linéaire maximale* : Les opérations matérielles requises se dénombrent par $2X(2Y + 1)$ multiplications, $4XY - (2X + Y)$ additions et XY déplacements pour un vecteur d'entrées complexes (X, Y).

- *Le conjugué d'un signal* : Cette opération consiste à changer le signe de la partie imaginaire de chaque composante d'un vecteur d'entrée (X, Y). Elle recourt à XY opérations de négation (NEG) dans le processeur.
- *L'hermitien* : Elle consiste à faire la transposée conjuguée d'un vecteur d'entrée (X, Y). Il lui faut pour ce faire XY multiplications et XY déplacements.
- *La FFT-Radix 2* : Sans optimisation de l'algorithme, c'est-à-dire que les produits des échantillons d'entrée de taille N par les facteurs de phase unitaires sont comptabilisés par le processeur en surnombre, $4Nlog_{10}(N)$ multiplications et additions complexes sont dénombrées, soit l'équivalent de $16Nlog_{10}(N)$ multiplications réelles et $8Nlog_{10}(N)$ additions réelles. Avec optimisation, les opérations matérielles sont réduites à $8Nlog_{10}(N)$ multiplications et $6Nlog_{10}(N)$ additions réelles .

Enfin, parmi les opérations de transfert mémoire, se dénombrent :

- *La transposée* : Cette opération transforme les lignes d'une matrice (X, Y) en colonnes et vice-versa. Elle requiert de ce fait XY déplacements.
- *La permutation* nécessite des déplacements d'une colonne (ou d'une ligne) à l'autre d'un vecteur. Elle varie en fonction des résultats escomptés.
- *La valeur minimale/maximale d'une matrice* (X, Y) se détermine en parcourant les valeurs de la dite matrice les unes après les autres jusqu'à en trouver le minimum ou le maximum par comparaison itérative. Elle nécessite donc XY comparaisons.
- *L'indice d'une valeur* retourne l'indice correspondant à une valeur donnée dans une matrice. Ce bloc dépend ainsi de la position de la valeur dans la matrice et se sert des instructions de déplacement et de comparaison.
- *Le suréchantillonnage* : Cette opération consiste à répéter N fois chacune des valeurs d'un vecteur d'entrée $(X, 1)$ ou $(1, Y)$ suivant un facteur d'échantillonnage N. Ainsi, la taille du vecteur d'arrivée devient $(XN, 1)$ ou $(1, NY)$ et s'obtient par NX ou NY déplacements.

- Le *sous-échantillonnage* est l'opération contraire au suréchantillonnage. Au lieu d'une répétition, on retire N valeurs à un vecteur d'entrée $(X, 1)$ ou $(1, Y)$ suivant un facteur d'échantillonnage N. Ainsi, la taille du vecteur d'arrivée devient $(X/N, 1)$ ou $(1, N/Y)$ et s'obtient par X/N ou Y/N déplacements.
- *L'addition modulo 2* : Elle se réalise aisément avec un ou-exclusif. Pour une entrée (X, Y), elle fait alors appel à XY déplacements.
- *La quantification dure* : Cette opération attribue une valeur binaire nulle à un nombre positif et une valeur binaire unitaire dans le cas contraire. Elle représente la dernière étape à franchir pour détecter un signal. Elle est l'équivalent matériel de XY déplacements.
- *La conversion d'une valeur décimale en valeur binaire* : Elle se sert de plusieurs opérations de déplacements qui sont fonction de la valeur du nombre décimal.
- *Le logarithme base 2* : Il se sert aussi de plusieurs opérations de déplacements qui sont fonction de la valeur du nombre dont on calcule le logarithme en base 2. Les deux primitives précédentes sont utiles par exemple pour le calcul des codes OVSF dans le CDMA.
- *La conversion unipolaire vers bipolaire* : Ce bloc convertit un signal d'entrée unipolaire (X, Y) en un signal de sortie bipolaire. Si l'entrée possède des valeurs comprises entre 0 et $M - 1$, où M est la borne supérieure M-aire, alors, la sortie correspond à des entiers compris entre $-(M - 1)$ et $M - 1$. Dans le cas binaire (M=2) qui est nôtre, les valeurs de sortie possibles sont -1 ou 1 suivant que les valeurs des entrées valent respectivement 1 ou 0. Cela conduit à XY déplacements dans le processeur.
- *Le décalage* : L'une de ses applications apparaît dans le récepteur Rake, lorsqu'on veut aligner des signaux ayant subi des évanouissements différents suivant le même délai (avance ou retard) tel que présenté à la section 2.4.2 du chapitre 2 pour parvenir à une bonne détection. Après observation du code assembleur généré pour

cette primitive, une valeur empirique des accès mémoires qui lui sont associés se déduit. En effet, le compilateur matérialise l'opération de décalage par les instructions de chargement de mot (LDDW), d'addition/soustraction en précision double (ADDDP/SUBDP), de stockage (STW), de comparaison en précision double (CMPEQDP, ...) et de conversion des valeurs en précision double vers des entiers avec troncature (DPTRUNC). Ces instructions nécessitent respectivement cinq, quatre, un, deux et quatre cycles pour s'exécuter et l'instruction de chargement intervient deux fois dans le processus. Ceci fait un total de 21 cycles pour les accès mémoires de cette primitive auxquels s'associent XY déplacements lorsqu'un vecteur d'entrée de taille (X, Y) est pris en considération.

La librairie de primitives sert de tremplin pour bâtir des équations de prédiction du temps CPU nécessaire pour implémenter une application dans le 'C6711. Les résultats obtenus par ces équations font l'objet du prochain chapitre. L'établissement d'une méthodologie de vérification s'impose afin de confirmer la véracité des prédictions.

4.3 La vérification

La méthode de profilage est celle choisie pour valider les résultats des prédictions. Cette technique consiste à mesurer ou à estimer le temps CPU nécessaire pour parcourir totalement ou en partie le code d'une application. Elle peut s'effectuer à plusieurs niveaux de granularité, depuis des instructions individuelles jusqu'aux fonctions entières (Lapsley, 1996). Pour ce faire, l'établissement d'une procédure d'automatisation des tests vient à point nommé afin de faciliter la réalisation de plus d'une centaine d'entre eux.

4.3.1 L'automatisation de la procédure de vérification

L'automatisation de la procédure d'implémentation apporte une contribution non négligeable pour savoir si oui ou non, un algorithme peut être implanté dans un processeur. Elle réduit substantiellement le temps de réalisation de plus d'une centaine de tests répéti-

tifs. Ces tests requièrent d'essayer différentes options du compilateur dans le but de choisir l'option qui, pour une application donnée, dépeint le mieux les prédictions. La pertinence de ces tests réside dans le fait que chaque option possède des niveaux d'optimisation distincts qui influent sur le nombre de cycles d'exécution total. Cela sera exprimé en détail plus loin.

L'automatisation se subdivise en quatre étapes.Le choix du logiciel Perl® s'avère judicieux à cet effet car l'utilitaire de script du simulateur de la compagnie Texas Instruments nommé Code Composer Studio® (CCS) permet l'utilisation de langages familiers pour l'automatisation des tests et la validation des applications. Dans un premier temps, il faut donc exécuter le script d'automatisation écrit en Perl (étape 1). Celui-ci va commander l'ouverture du modèle Matlab/Simulink® à implémenter dans le processeur approprié (étape 2). Une fois le modèle ouvert, il est automatiquement transféré sous forme de code C dans le processeur grâce à l'outil Real Time Workshop (RTW) de Simulink® (étape 3). Les prochains paragraphes donnent une description plus détaillée de ce processus. À la suite de ce transfert, s'entame la procédure de profilage pour collecter le temps CPU nécessaire au traitement de l'application. Le nombre de cycles se collecte au moyen d'un profileur à occurences multiples situé dans la trousse à outils d'analyse (en anglais Analysis ToolKit (ATK)) du simulateur du C6711. Ce simulateur s'attèle à imiter le comportement du processeur (Wang et Lee, 2005). En effet, CCS est un environnement de développement qui supporte les plateformes DSP TMS320C6000 et TMS320C5000 de Texas Instruments. Il supporte toutes les phases de développement à savoir la conception, le codage, le déboguage, l'analyse et l'optimisation. Ces principaux éléments sont :

- un environnement de développement qui intègre un éditeur, un débogueur, un gestionnaire de projet, et un profileur,
- un compilateur C/C++, assembleur et des outils de génération de code,
- un simulateur d'instructions,

- un noyau temps réel logiciel,
- un échange en temps réel entre l'hôte et la cible,
- des outils d'analyse et de visualisation en temps réel des données.

L'outil de profilage offre la possibilité de déduire facilement le compromis entre la taille du code et les performances. Il permet de déterminer les performances de chaque fonction sous différentes options de compilation. En employant le script d'automatisation à cet effet, le profilage des fonctions dans l'application peut être opéré en chargeant et en faisant rouler un fichier exécutable pour une durée correspondant au temps de traitement d'une trame de données au sein du processeur. L'ATK peut alors de fournir le décompte du nombre de cycles épuisés dans chaque fonction. Il résulte, après chaque exécution du script selon des paramètres de simulation préalablement choisis, un fichier de type *log* indiquant le statut de chaque appel automatique ainsi qu'une feuille de calcul Excel® (étape 4) contenant le nombre de cycles de chaque fonction profilée pour la configuration choisie. Finalement, vient la nécessité de créer une autre feuille de calcul manuellement pour assembler les résultats des fonctions désirées selon différents paramètres de simulation. Cette dernière étape pourrait aussi être automatisée.

4.3.2 La génération automatique du code C

La génération automatique du code C se fait à travers l'outil Real Time Workshop (RTW) de Simulink® selon les étapes illustrées à la Figure 1. Mais avant d'y arriver, il faut exécuter un fichier d'initialisation dans Matlab® . Celui-ci contient toutes les variables dont l'initialisation est nécessaire à l'éxécution de l'application . Survient ensuite l'exécution du programme dans Simulink®, plateforme présentant l'intérêt de favoriser une programmation proche du matériel. Les fichiers Simulink® exécutés contiennent des fonctions Matlab® embarquées dites *embedded Matlab functions* pour faciliter leur transcription en langage C. Ces fonctions embarquées, à la différence d'une fonction usuelle dans Matlab®, possède des règles de programmation très restreintes et similaire au langage

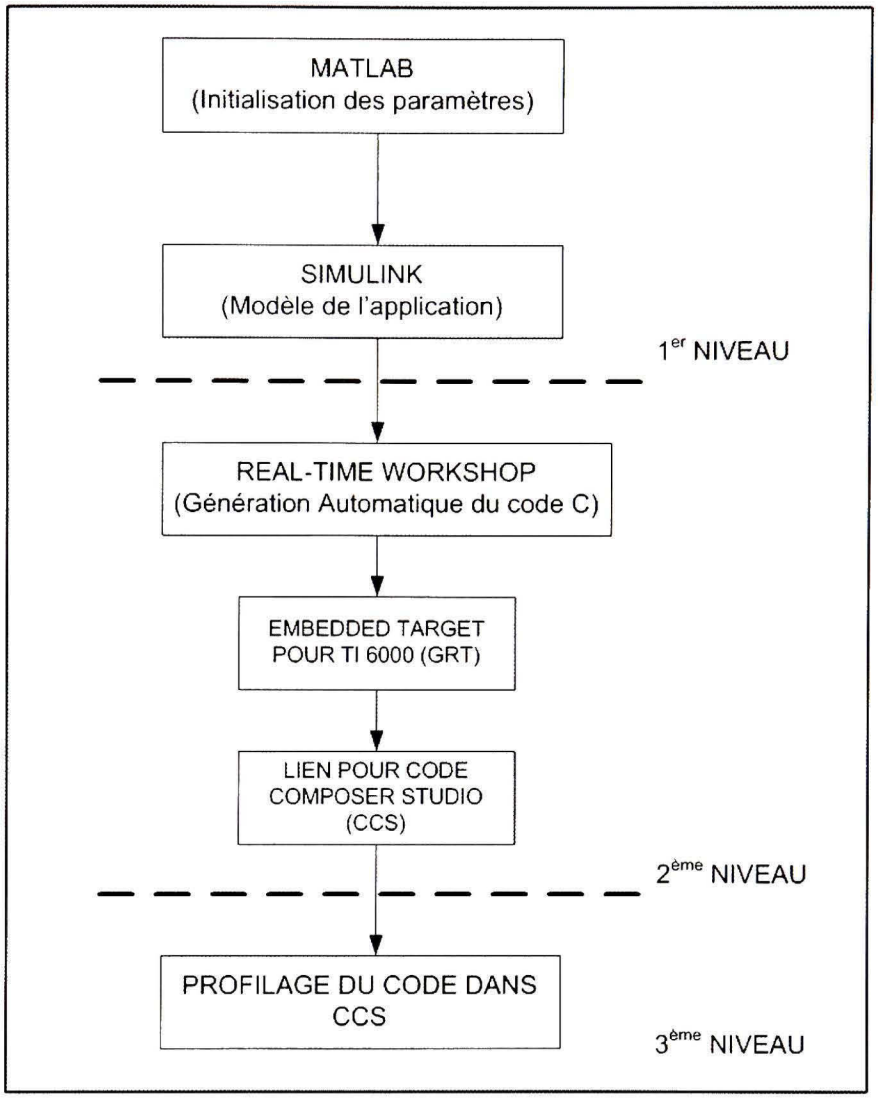


Figure 4.1 Les étapes de vérification.

C comme par exemple l'impossibilité de définir des matrices de tailles variables. Un autre choix d'implémentation aurait pu consister à bâtir ces fichiers à l'aide des blocs d'opérations disponibles dans la librairie de Simulink®. La première option l'emporte sur la seconde car cette dernière est plus longue à réaliser et produit des résultats somme toute équivalents à ceux de la première option. Ceci justifie par ricochet son choix pour simplifier la procédure de validation. Les étapes-clés afin de parvenir à une bonne génération du code sont les suivantes :

- Le choix de la plateforme d'intérêt (dans ce cas, le C6711) dans la librairie de Simulink® réservée aux processeurs embarqués pour la famille C6000 de Texas Instruments. Cette possibilité permet de spécifier les caractéristiques du plan d'adressage du processeur, indispensable pour l'implémentation. Ce plan est défini tel qu'illustré à la section 4.2.
- La configuration des paramètres de simulation dans Simulink® pour une activation en bonne et due forme de RTW. Là, il est possible de prédéfinir quelques options de simulation pour lesquelles le simulateur optera, à l'exemple du niveau d'optimisation souhaité, des conditions de génération du code dans CCS et de son niveau de transformation. Celui-ci peut aller de la simple génération du code à la construction et à l'exécution d'un projet en passant par la création d'un projet, l'archivage d'une librairie et l'unique construction d'un projet. La configuration des paramètres de simulation autorise le choix du type de processeur ciblé ainsi que du type de code C généré. Dans ce cas, le processeur d'intérêt appartient à la famille C6000 et les contraintes au niveau des licences du logiciel Matlab® à notre portée orientent le choix vers du code C générique. Ce choix, comme le démontreront les résultats dans le chapitre à venir, ne constitue pas un problème pour la validation.

4.4 Les paramètres de simulation dans le compilateur

Les options offertes par le compilateur contrôlent simultanément ce dernier et les programmes qu'il exécute. La version de CCS employée pour réaliser ce travail possède 12 catégories d'options (Texas Instruments, 2005) : les options pour le contrôle du compilateur, le débogage symbolique et le profilage, pour le changement des extensions de fichiers par défaut, pour la spécification des fichiers, pour la spécification des répertoires, spécifiques à la machine, pour l'analyse des fichiers (en anglais « parsing »), d'analyse pour le prétraitement, d'analyse pour les diagnostiques, pour le contrôle de l'optimisation, pour le contrôle de l'assembleur et pour le contrôle de liaison entre les fichiers objets et les fichiers éxécutables. La suite se focalisera cependant sur l'option dont dépend la pertinence

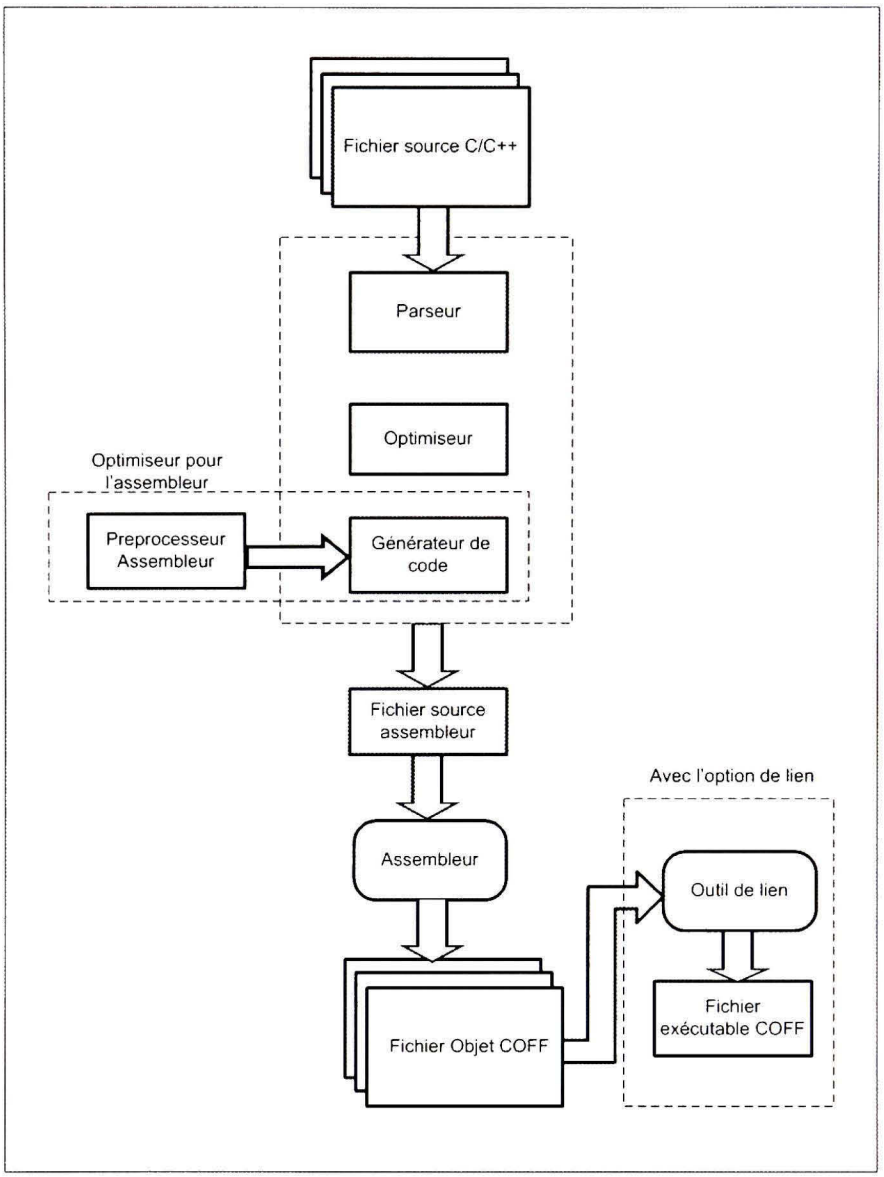


Figure 4.2 Le fonctionnement du compilateur C.

de ce travail. Il s'agit de l'option qui contrôle l'optimisation des instructions. Mais avant, il convient d'introduire de manière générale le fonctionnement du compilateur à travers une brève introduction des options qui le contrôlent.

4.4.1 Le contrôle du compilateur

Parmi les choix possibles pour cette option apparaît en premier plan la commande de l'outil d'interprétation du contenu d'un fichier. En second lieu intervient la possibilité d'activer ou de désactiver l'unité de lien qui combine ensemble les fichiers objets de format commun dit « Common Object File Format » (COFF) pour générer un fichier objet exécutable. Les fichiers de type COFF permettent de définir le plan d'adressage de la mémoire au moment du lien. Ce plan concourt à la maximisation des performances en liant le code source C/C++ avec les données dans des espaces mémoires spécifiques. Par défaut, le compilateur ne réalise pas l'étape de lien. Il faut de ce fait l'activer pour qu'il la fasse.

La figure 2 illustre le fonctionnement du compilateur et le chemin effectué par celui-ci avec et sans utilisation du lien. Elle illustre aussi le processus d'obtention des fichiers COFF. En effet, le fichier de départ C ou C++ subit une vérification grammaticale dans le parseur. Ensuite, il est optimisé et transformé en assembleur avant d'être mis en objet de type COFF.

4.4.2 L'optimisation de la compilation

Le compilateur offre plusieurs niveaux d'optimisation (Texas Instruments, 2005) catalogués en haut niveau et bas niveau. Les optimisations de haut niveau (niveaux 2 et 3) sont réalisées afin de produire du code optimal alors que celles spécifiques ou bas niveau surviennent uniquement dans le générateur de code (niveaux 0 et 1). La meilleure façon d'invoquer l'optimisation est d'utiliser le programme cl6x du compilateur en spécifiant l'option -On dans la ligne de commande de celui-ci ; n représentant le niveau d'optimisation (0, 1, 2 ou 3) qui contrôle le type et le degré d'optimisation souhaité (Texas Instruments, 2005). Ces quatre niveaux d'optimisation sont présentés ci-dessous.

- Optimisation 0 (-O0) : Ce niveau permet de contrôler la simplification du graphe de dépendance (flow-graph) et de faire des rotations de boucles. Par ailleurs, à cette

étape, l'allocation des variables aux régistres, l'élimination du code inutilisé, la simplification des expressions et, finalement, l'expansion des appels de fonctions préalablement déclarées sont possibles.

- Optimisation 1 (-O1) : Cette option accomplit toutes les actions du niveau 0. En plus, elle supprime les assignations inutilisées et élimine les expressions communes locales.
- Optimisation 2 (-O2) : C'est l'option d'optimisation par défaut du compilateur. Elle réalise toutes les optimisations de niveaux 0 et 1, et, en plus, elle offre la possibilité d'entreprendre le pipelinage logiciel, les optimisations et le déroulement de boucles. Le choix de cette option élimine les sous-expressions globales communes et les assignations globales non employées. Enfin, cette option convertit les références vectorielles des boucles pour les incrémenter sous forme de pointeur.
- Optimisation 3 (-O3) : Ce niveau d'optimisation réalise les actions des niveaux inférieurs. À cela s'ajoutent la suppression des fonctions qui ne sont jamais appelées, la simplification des fonctions qui retournent des valeurs inutilisées, le réordonnancement des déclarations des fonctions, l'identification des caractéristiques des variables au niveau fichier ...

L'emploi de l'option -On indique au compilateur la nécessité d'optimiser le code à implémenter. Plus la valeur de n est élevée, plus le compilateur doit investir des efforts pour l'optimiser. Cependant, il pourrait s'avérer pertinent de préciser au compilateur les priorités lors de l'optimisation. Pour des niveaux d'optimisation moindres (-O0 et -O1) , les priorités s'articulent autour du temps de compilation et de la facilité à déverminer. Pour des niveaux élevés (-O2 et -O3), soit la taille du code ou la vitesse de traitement de l'application peut être priorisée, l'une au détriment de l'autre. Chacune d'elles dispose de deux niveaux d'optimisation. Pour la taille du code, les deux niveaux d'optimisation sont la taille critique et la taille la plus critique. Pour la vitesse, se distinguent la vitesse critique et la vitesse la plus critique. La seconde métrique, soit la vitesse, est celle qui attire

notre attention car elle influence directement le nombre de cycles requis pour le traitement d'une application. Elle déterminera l'étude des résultats avec la prise en considération de ses deux niveaux d'optimisation. D'ailleurs, par défaut, lorsque l'on spécifie les niveaux -O2 ou -O3, le compilateur optimise en premier les performances, autrement dit la vitesse d'exécution.

Un autre acteur non moins important lors de la collecte des résultats de profilage est le pipeline logiciel introduit à la section 3.6.4 du chapitre 3. Le compilateur opère le pipeline logiciel par défaut aux niveaux 2 et 3 d'optimisation (Texas Instruments, 2005). Celui-ci affecte le code C/C++ compilé en contribuant à son optimisation. En effet, le compilateur inclut les informations relatives à cette opération dans un fichier assembleur. Ces informations apparaissent sous forme de commentaires dans le dit fichier avant chaque boucle et pour que l'optimiseur puisse y accéder lorsque l'outil de compilation est en marche (Texas Instruments, 2005). À la fin, le compilateur reçoit une rétroaction l'informant de l'issue du déroulement du pipeline dans un fichier assembleur. Si le pipeline logiciel fonctionne bien pour une boucle, une table renseignant sur le taux d'utilisation des registres à chaque cycle du coeur de boucle s'ajoute aux commentaires du pipeline logiciel dans le code assembleur généré. Le compilateur a aussi accès à l'ordonnancement des instructions à chaque itération de la boucle pipelinée. L'option du pipeline logiciel doit être désactivée pour une comparaison équitable entre les prédictions et l'implémentation sinon elle s'en trouve biaisée.

4.5 Conclusion

Ce chapitre a présenté la méthodologie développée pour l'extraction de primitives. Celle-ci débute par la compréhension de l'architecture du processeur cible. Elle contribue à bâtir un plan d'adressage qui maximise les performances et l'utilisation de la mémoire. La seconde étape est la compréhension des algorithmes choisis pour en extraire les opérations récurrentes et, ainsi déduire une librairie d'opérations usuelles en télécommunications à

granularités variables. Cette librairie n'a nullement la prétention d'être exhaustive. Enfin, une procédure de validation en majeure partie automatisée couronne le travail. Les résultats de la validation font l'objet du prochain chapitre.

CHAPITRE 5

LES RÉSULTATS

Les implémentations des différentes applications dans le processeur TMS320C6711 s'obtiennent en faisant fluctuer certaines grandeurs qui leur sont propres et en fixant d'autres. Cette décision est prise par le concepteur, selon ses besoins. Dans le cas du ZF-SQRD (Zero Forcing Sorted QR Decomposition), les paramètres considérés variables sont le nombre d'antennes à l'émetteur (Nt) et au récepteur (Nr) et les paramètres considérés fixes sont la taille des trames de données et le type de modulation. Pour le Rake, les grandeurs considérées fixées sont la taille des trames et les facteurs d'étalement des voies I et Q respectivement. La grandeur tenue pour susceptible de changement est son nombre de doigts. Les résultats de la FFT radix 2 DIT s'obtiennent en faisant varier le nombre de points en entrée entre 64 et 1024, par puissances de 2. Le choix adéquat des options de simulation est un facteur essentiel pour l'obtention de résultats conformes à ceux des prédictions car ceux-ci peuvent être grandement disparates d'une option de simulation à l'autre.

Ce chapitre s'organise comme suit. La première partie fait état des équations décrivant la complexité de l'algorithme ZF-SQRD. Ces équations permettent de déduire les prédictions du nombre de cycles respectant des critères soulignés plus loin. Tous ces résultats restent incomplets si l'analyse fait défaut. D'où sa présence pour chapoter l'étude. Par la suite, la même procédure se répète, mais cette fois-ci pour les deux applications restantes, soit le Rake et la FFT. L'analyse des résultats pour chacune des applications guidera vers l'exploration de l'impact du pipeline logiciel sur les prédictions et mettra un terme au présent chapitre.

Les résultats obtenus après implémentation tiennent compte de six modes de simulation à savoir :

a. Vitesse critique avec niveau d'optimisation 0 (- O0),

b. Vitesse critique avec niveau d'optimisation 2 (- O2) et pipeline logiciel désactivé (SP-Des.),

c. Vitesse critique avec niveau d'optimisation 2 (- O2) et pipeline logiciel activé (SP-Ac.)

d. Vitesse la plus critique avec niveau d'optimisation 0 (- O0),

e. Vitesse la plus critique avec niveau d'optimisation 2 (- O2) et pipeline logiciel désactivé (SP-Des.),

f. Vitesse la plus critique avec niveau d'optimisation 2 (- O2) et pipeline logiciel activé (SP-Ac.).

Ces options ont été choisies après maints essais car elles représentent au mieux la variation susceptible de prendre place entre les résultats qui s'éloignent les plus des prédictions (- O0) et ceux qui au contraire, s'en rapprochent de près (- O2) si les paramètres de simulation ne sont pas adéquatement choisis . Les tests préliminaires ont aussi révélé que le niveau 2 d'optimisation procure de meilleurs résultats que le niveau qui lui est directement supérieur dans le contexte qui est nôtre.

5.1 L'algorithme ZF-SQRD

Les principales métriques intervenant dans l'algorithme ZF-SQRD sont le nombre d'antennes émettrices, le nombre d'antennes réceptrices, le type de modulation et la taille des trames. La librairie des primitives présentée à la section 4.2 et le jeu d'instructions du processeur exposé à la section 3.5 sont les piliers de l'estimation de la complexité de l'algorithme ZF-SQRD. Cette complexité est polynômiale en tenant pour fixes la longueur des trames et le type de modulation (QPSK) et pour variables deux grandeurs, l'une représentant le nombre d'antennes N_t au transmetteur et l'autre,N_r, le nombre d'antennes

au récepteur. Ce choix trouve raison par le fait qu'en pratique, le type de modulation et la longueur des trames sont très souvent imposés par le standard de communication employé (UMTS, Wimax, ...), mais le nombre d'antennes que ce soit à l'émetteur ou au récepteur, dû au fait qu'il représente le nombre d'usagers et/ou le nombre d'antennes de la station de base, est susceptible de changer aléatoirement. La complexité résultante se comptabilise matériellement en termes d'opérations de multiplications (*Mult*), d'additions (*Add*), de divisions (*Div*), de racines carrées (*Sqrt*), de négation (*Neg*) et de déplacements (*Mov*). Elle se représente par les équations 5.1, 5.2, 5.3, 5.4, 5.5, 5.6 :

$$\begin{aligned} Mult &= A \times 2L \left[N_t^2 + 2 \times \left(N_t N_r + N_t + \sum_{i=1}^{N_t - 1} (N_t - i) \right) \right] \\ &\quad + A \times 2N_r \left[N_t + \left(\sum_{i=1}^{N_t} N_t - i + 1 \right) + 2 \left(\sum_{i=1}^{N_t - 1} N_t - i \right) \right] \\ &\quad + A \times 4N_r^2 \left(\sum_{i=1}^{N_t - 1} N_t - i \right), \end{aligned} \tag{5.1}$$

$$\begin{aligned} Add &= B \times L \left[(3N_r - 1) N_t + 2 (N_t - 1) + 2N_t (N_t - 1) \right] \\ &\quad + B \times \left[4L \left(\sum_{i=1}^{N_t - 1} N_t - i \right) + 2N_r (N_r + 2) \left(\sum_{i=1}^{N_t - 1} N_t - i \right) \right] \\ &\quad + B \times 2N_r \left[2 (N_r - 1) \left(\sum_{i=1}^{N_t} N_t - i + 1 \right) \right] \end{aligned} \tag{5.2}$$

$$Div = C \times (N_t + 2), \tag{5.3}$$

$$Sqrt = D \times N_t, \tag{5.4}$$

$$Neg = E \times \left[N_t N_r + N_r \left(\sum_{i=1}^{N_t - 1} N_t - i \right) \right], \tag{5.5}$$

$$\begin{aligned} Mov &= F \times \left[2 \left(N_t - 1 \right) \left(2N_t + N_r \right) + 2LN_t + N_t^2 \right] \\ &\quad + F \times \left[\left(\sum_{i=1}^{N_t} N_t - i \right) + \left(\sum_{i=1}^{N_t} N_t - i + 1 \right) \right], \end{aligned} \tag{5.6}$$

où :

- A, B, C, D, E et F font office de délais pour les accès mémoires en point flottant introduits à la section 3.5,
- L représente la longueur des trames après une modulation QPSK,
- N_t et N_r symbolisent respectivement les nombres d'antennes au transmetteur et au récepteur de la chaîne de communication.

Selon la Figure 1, il vient à remarquer que lorsque le pipeline logiciel n'est pas actif pour l'optimisation de niveau 2, alors, les prédictions dépeignent admirablement bien les nombres de cycles équivalents après implémentation. Le Tableau XIII vient appuyer ce constat en établissant la valeur moyenne du ratio implémentation/prédiction pour les six options de simulation prises en considération. Il laisse apparaître une marge d'erreur de 9% pour l'option -O2 à la vitesse critique lorsque le pipeline logiciel n'est pas fonctionnel. C'est la meilleure option pour comparer aux prédictions. Le deuxième choix de configuration des paramètres de simulation qui s'apparente aux prédictions est attribué à la vitesse la plus critique avec optimisation -O2. Celui-ci donne tout aussi bien des résultats rela-

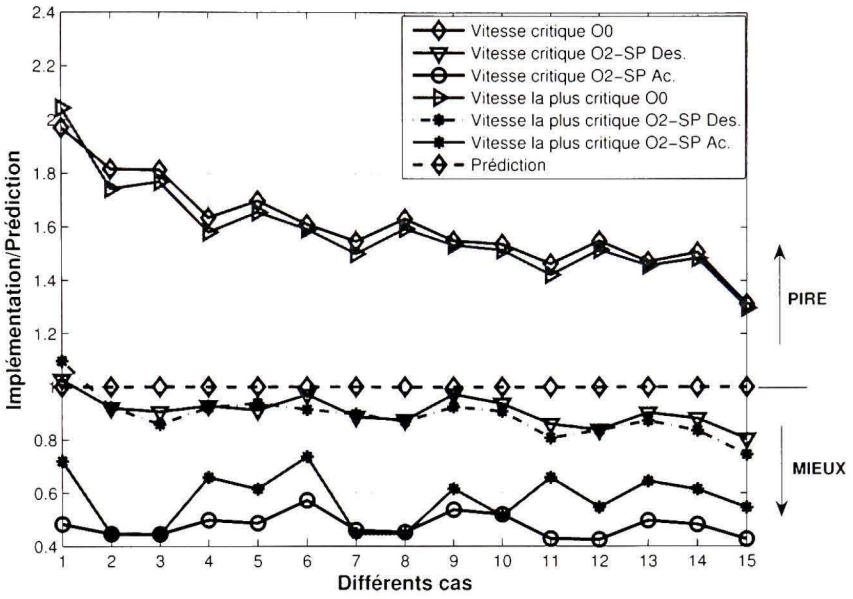


Figure 5.1 Comparaison entre la prédiction et l'implémentation pour différents cas pour des trames de longueur 122.

tivement bons, soit un ratio moyen de 89 % entre l'implémentation et la prédiction. En général, pour cet algorithme, lorsque le gain en diversité est faible (plus petit que 3), alors les prédictions sont très proches de l'implémentation pour l'option -O2. Le rapprochement du temps CPU entre la prédiction et l'implémentation pour des gains en diversité plus élevés, est au contraire moins évident quoique demeurant de mise pour le même niveau d'optimisation. La Figure 1 et le Tableau XIII justifient l'assertion selon laquelle l'option -O0 serait définitivement un choix à rejeter parce que le temps CPU obtenu après implémentation est au moins 1, 5 fois supérieur aux prédictions, quel que soit le type de vitesse considéré : critique ou la plus critique.

L'activation du pipeline dans le cadre de cet algorithme réduit de moitié l'implémentation comparativement à la prédiction pour l'option -O2 à la vitesse critique. Le Tableau XIII présente un ratio moyen implémentation/prédiction oscillant autour de 0.6 pour un choix de vitesse très critique avec l'option -O2. Ainsi, se déduit une conclusion pertinente, soit

que pour cette application, non seulement les prédictions sont fiables (option vitesse critique -O2 SP Des.), mais aussi, les résultats avec et sans pipeline sont prédictibles du fait que les seconds (résultats sans pipeline) sont les doubles des premiers (résultats avec pipeline). Finalement, il advient selon la Figure 1 que l'option « *vitesse critique* » est plus stable que l'option « *vitesse la plus critique* » précisément lorsque le pipeline est fonctionnel.

Tableau XIII

Rapport moyen du ratio implémentation/prediction pour les 15 cas considérés

Paramètres de simulation	Implémentation/Prédiction
Vitesse critique - O0	1.61
Vitesse critique - O2 - SP Désactivé	0.91
Vitesse critique - O2 - SP Activé	0.91
Vitesse la plus critique - O0	0.48
Vitesse la plus critique - O2 - SP Désactivé	0.89
Vitesse la plus critique - O2 - SP Activé	0.58

Le Tableau XIV présente à titre d'information, les constantes de normalisation pour différents nombres d'antennes à l'émetteur et au récepteur. Ces valeurs servent à rendre unitaire la courbe des prédictions de la Figure 1. Par ailleurs, l'investigation de l'impact de la diversité suggère de regrouper les résultats de la meilleure option (vitesse critique -O2) non par affinité du nombre d'antennes réceptrices comme présenté dans le Tableau XIV, mais en termes de gains en diversité semblables. Ceci permet alors pour une valeur donnée de diversité , de simplifier les équations 5.1, 5.2, 5.3, 5.4, 5.5 et 5.6 en une seule équation à deux variables N_t et N_r, les autres paramètres étant fixes pour les raisons ci-dessus énumérées. En effet, la diversité est l'une des techniques employées pour combattre les évanouissements presqu'inévitables lors d'une communication. Elle a été brièvement introduite dans la section 1.1.3. Dans le multiplexage spatial, elle vaut $N_r - N_t + 1$ comme présenté à la section 2.3. L'aboutissement à l'équation simplifiée a graduellement franchi plusieurs étapes. Tout d'abord, s'obtient, pour un gain donné en diversité, une équation

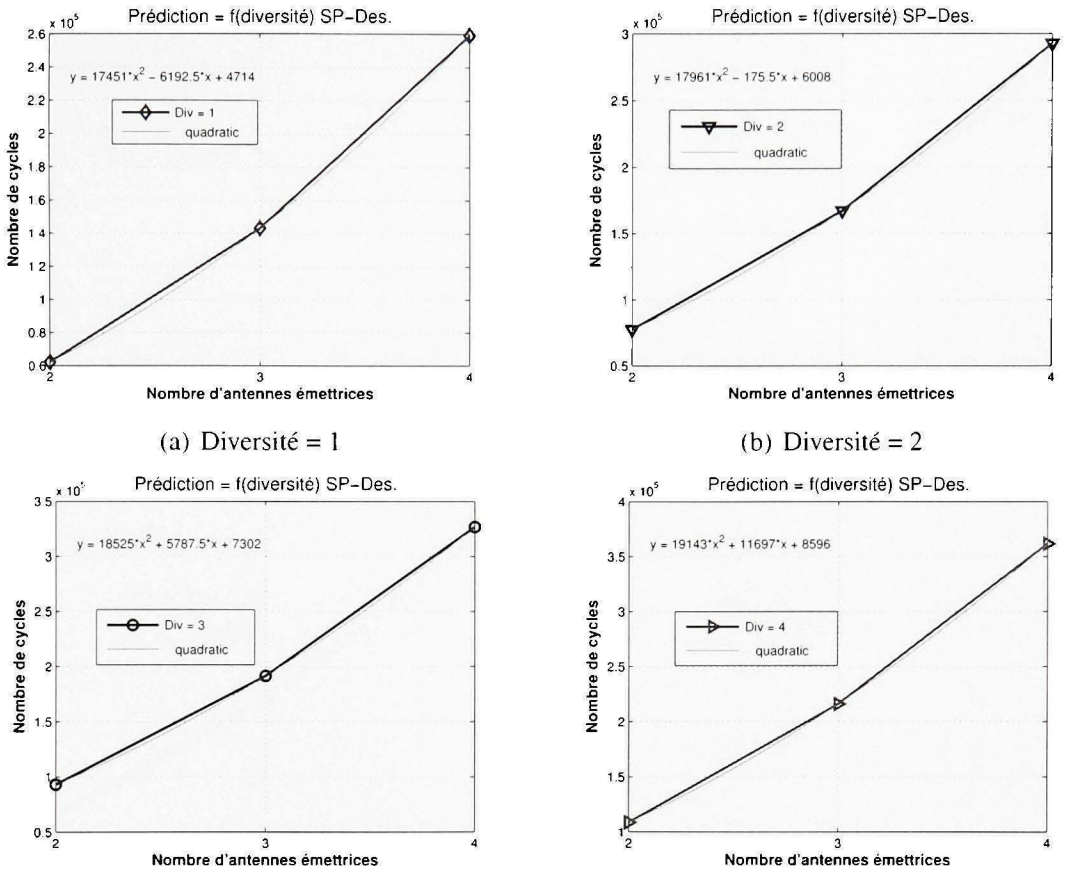
Tableau XIV

Valeurs des prédictions pour différents nombres d'antennes

Cas	(N_t, N_r)	Diversités	Prédictions
1	(2,2)	1	62135
2	(2,3)	2	77503
3	(3,3)	1	143200
4	(2,4)	3	92979
5	(3,4)	2	167135
6	(4,4)	1	259168
7	(2,5)	4	108563
8	(3,5)	3	191394
9	(4,5)	2	292690
10	(5,5)	1	412454
11	(2,6)	5	124255
12	(3,6)	4	215977
13	(4,6)	3	326860
14	(5,6)	2	456907
15	(6,6)	1	671239

quadratique construite à partir de trois points ayant pour abscisses les nombres d'antennes émettrices valant respectivement $N_t = 2$, $N_t = 3$ et $N_t = 4$ et pour ordonnées, le temps CPU qui dérive de la combinaison des équations 5.1, 5.2, 5.3, 5.4, 5.5 et 5.6 .

Cette tâche se répète pour quatre valeurs de gain en diversité (Div), soit respectivement $Div = 1$, $Div = 2$, $Div = 3$ et $Div = 4$ pour le cas où le pipeline logiciel est désactivé comme l'illustre la Figure 2 et pour celui où au contraire, il est actif comme à la Figure 3. Ensuite, la vérification que les résultats issus de ces équations sont en conformité avec les implémentations équivalentes est réalisée et prouve que c'est le cas. En effet, les prédictions diffèrent de 8% par rapport aux implémentations pour l'option SP Des. et de 4% pour l'option contraire. Finalement, pour compléter le tout, une dernière vérification est de mise. Celle qui consiste à vérifier si ces équations élaborées sur la base de trois valeurs de N_t restent valides pour des nombres d'antennes émettrices plus élevées. Les tests ont été réalisés pour N_t variant entre cinq et huit pour les quatre valeurs de diversité ci-dessus



(a) Diversité = 1

(b) Diversité = 2

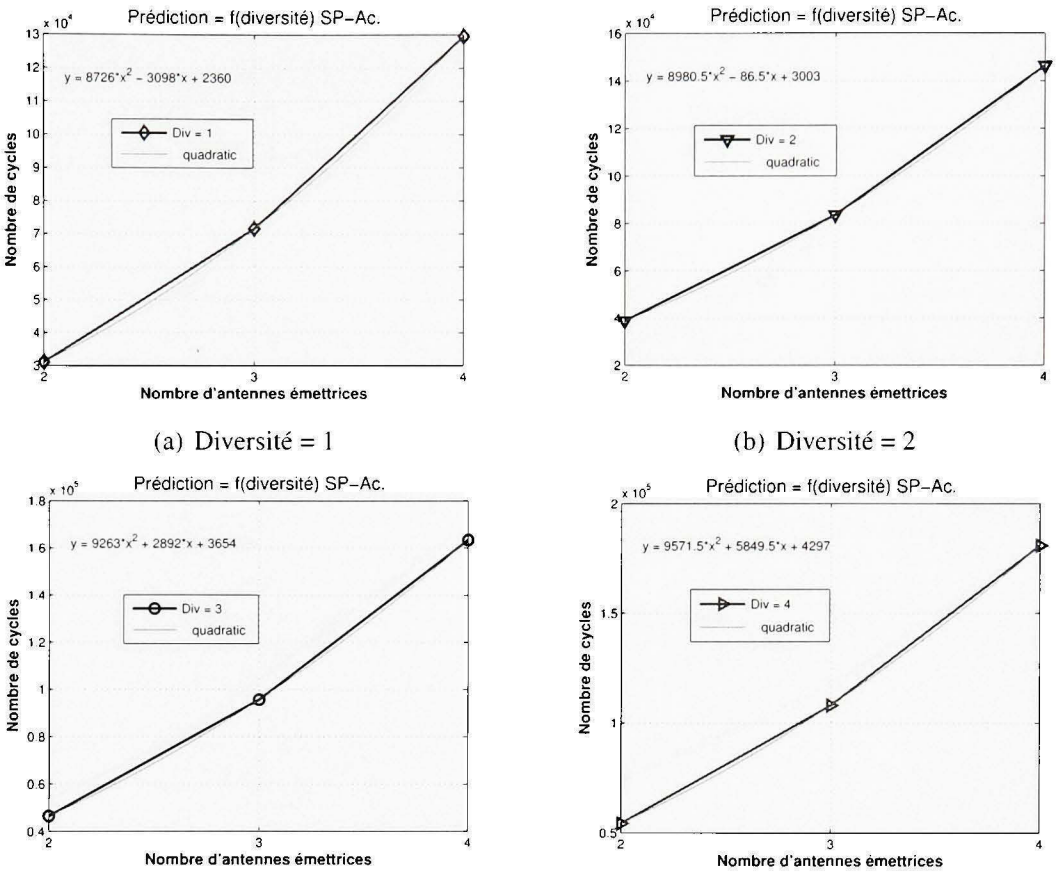
(c) Diversité = 3

(d) Diversité = 4

Figure 5.2 Équations générales décrivant la variabilité de la prédiction du nombre de cycles du ZF-SQRD lorsque le pipeline est désactivé et obtenues à partir de 3 points pour des valeurs de diversités de : (a) 1, (b) 2, (c) 3 et (d) 4

énumérées. Plus concrètement, les tests supplémentaires incluent les combinaisons suivantes du binôme (N_t, N_r) : (7,7), (8,8) pour $Div = 1$,(6,7), (7,8) pour $Div = 2$,(5,7), (6,8) pour $Div = 3$, (4,7) et (5,8) pour $Div = 4$. Ils se sont avérés concluants. Les valeurs ainsi extrapolées sont à 1% équivalentes à celles obtenues grâce aux équations 5.1, 5.2, 5.3, 5.4, 5.5 et 5.6. De plus le ratio entre l'implémentation et la prédiction pour ces valeurs sont de 0.9 pour SP Des. et 0.94 pour SP Ac..

Ce niveau de généralisation n'aurait pas pu être atteint en regroupant les résultats par affinité du nombre d'antennes réceptrices. En effet, toute extrapolation, pour un nombre



(a) Diversité = 1

(b) Diversité = 2

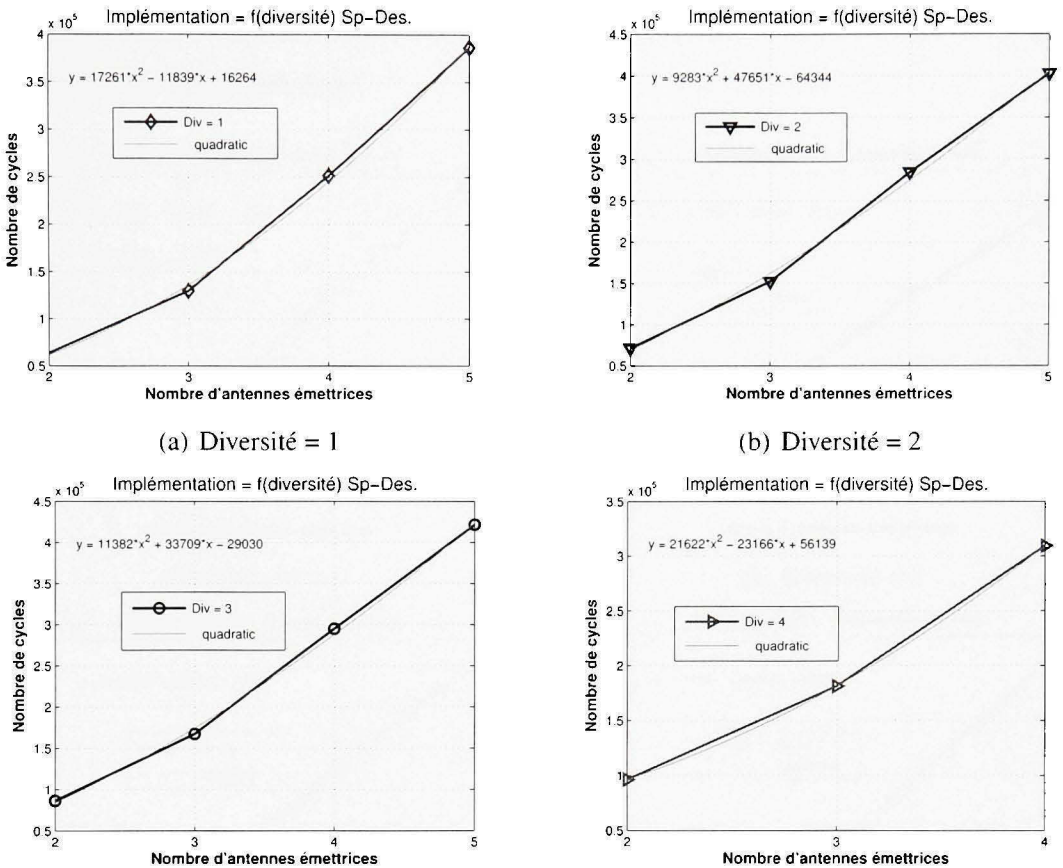
(c) Diversité = 3

(d) Diversité = 4

Figure 5.3 Équations générales décrivant la variabilité de la prédiction du nombre de cycles du ZF-SQRD lorsque le pipeline est activé et obtenues à partir de 3 points pour des valeurs de diversités de : (a) 1, (b) 2, (c) 3 et (d) 4

donné d'antennes réceptrices se serait limitée à un intervalle d'antennes émettrices plus petit ou égal à N_r. Par exemple, pour $N_r = 6$, connaissant le nombre de cycles requis pour trois combinaisons $(N_t, 6)$, on ne peut déduire que les résultats pour les deux restantes et non pour celles lorsque N_r est différent de six. Par contre, pour une diversité donnée, à partir de trois points, il est possible d'extrapoler les résultats pour n'importe quelle valeur (N_t, N_r) respectant le critère de diversité.

Une procédure analogue à celle réalisée pour les prédictions a été faite avec les implémentations et est illustrée aux Figures 4 et 5. Elle démontre qu'il est possible d'arriver à des



(a) Diversité = 1

(b) Diversité = 2

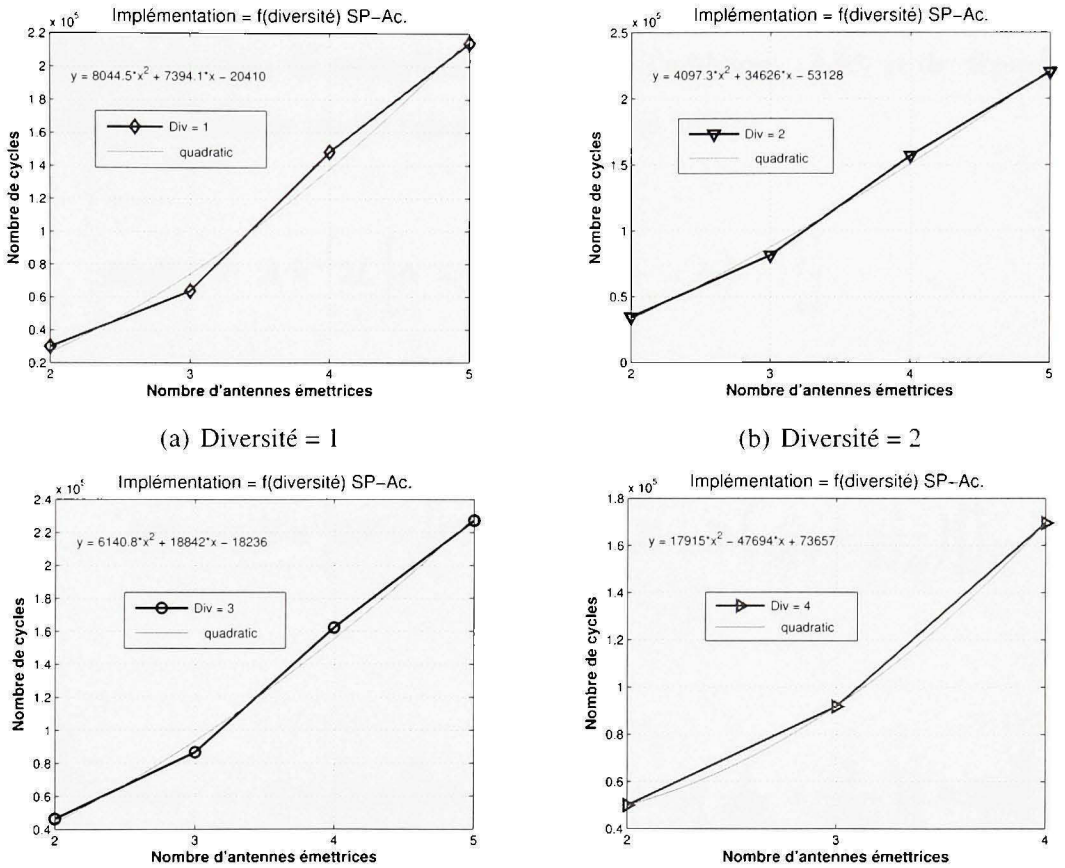
(c) Diversité = 3

(d) Diversité = 4

Figure 5.4 Équations générales décrivant la variabilité de l'implémentation du ZF-SQRD en fonction du nombre de cycles lorsque le pipeline est désactivé et obtenues à partir de 3 points pour des valeurs de diversités de : (a) 1, (b) 2, (c) 3 et de 4 points pour une valeur de diversité de (d) 4

équations fiables décrivant le comportement de l'application dans le processeur en exploitant la diversité, mais cette fois-ci basée sur quatre points pour plus de précision et ce pour des gains en diversité strictement inférieurs que quatre. Pour $Div = 4$, l'implémentation se limitant à une valeur maximale de N_r de huit, les équations obtenues respectivement après désactivation et activation du pipeline logiciel prennent appui sur trois points d'abscisses (2,5), (3,6) et (4,7). La vérification de la validité de ces équations s'est faite sur le point restant d'abscisses (5,8). En somme, les résultats des implémentations extrapolés à partir des équations de base demeurent similaires à celles obtenues réellement à 1% près.

Cette procédure n'était pas absolument nécessaire, mais elle vient bonifier celle qui a été appliquée pour les prédictions.



(a) Diversité = 1

(b) Diversité = 2

(c) Diversité = 3

(d) Diversité = 4

Figure 5.5 Équations générales décrivant la variabilité de l'implémentation du ZF-SQRD en fonction du nombre de cycles lorsque le pipeline est activé et obtenues à partir de 3 points pour des valeurs de diversités de : (a) 1, (b) 2, (c) 3 et de 4 points pour une valeur de diversité de (d) 4

5.2 Récepteur Rake

La caractérisation de l'opération de décalage par détermination empirique de la valeur associée aux accès mémoires du processeur est la pierre de touche qui rend conformes les résultats des équations de prédiction du temps CPU du récepteur Rake avec les implémentations équivalentes. Celle-ci a été présentée à la section 4.2. Les paramètres susceptibles

de varier dans l'algorithme sont la taille des trames, les facteurs d'étalement des voies de données et de contrôle respectivement et le nombre de doigts qui représentent le nombre de trajets dominants considérés. La complexité résultante se comptabilise matériellement en termes d'opérations de multiplications (*Mult*), d'additions (*Add*) et de déplacements (*Mov*). Elle se représente par les équations 5.7, 5.8 et 5.9 :

$$Mult = A \times \left[2L \left[N \times \left(\frac{1}{SF_I} + \frac{1}{SF_Q} + 5 \right) + 1 \right] \right], \tag{5.7}$$

$$Add = B \times \left[2N \times \left[(3L-1) + L\,(N-1) \left(\frac{1}{SF_I} + \frac{1}{SF_Q} \right) \right] \right], \tag{5.8}$$

$$Mov = C \times \left[L \times \left[3N + 14 + (N+1) \left(\frac{1}{SF_I} + \frac{1}{SF_Q} \right) \right] + 250 \right], \tag{5.9}$$

où :

- A et B sont les délais représentant les accès mémoires en point flottant introduits à la section 3.5,
- C est la constante empirique représentant les accès mémoires en point flottant pour une opération de décalage et introduite à la section 4.2,
- L représente la longueur des trames en bribes (*chips*) à l'entrée du récepteur,
- SF_I et SF_Q symbolisent respectivement les facteurs d'étalement des voies de données (voie I) et de contrôle (voie Q),
- N est le nombre de doigts du récepteur.

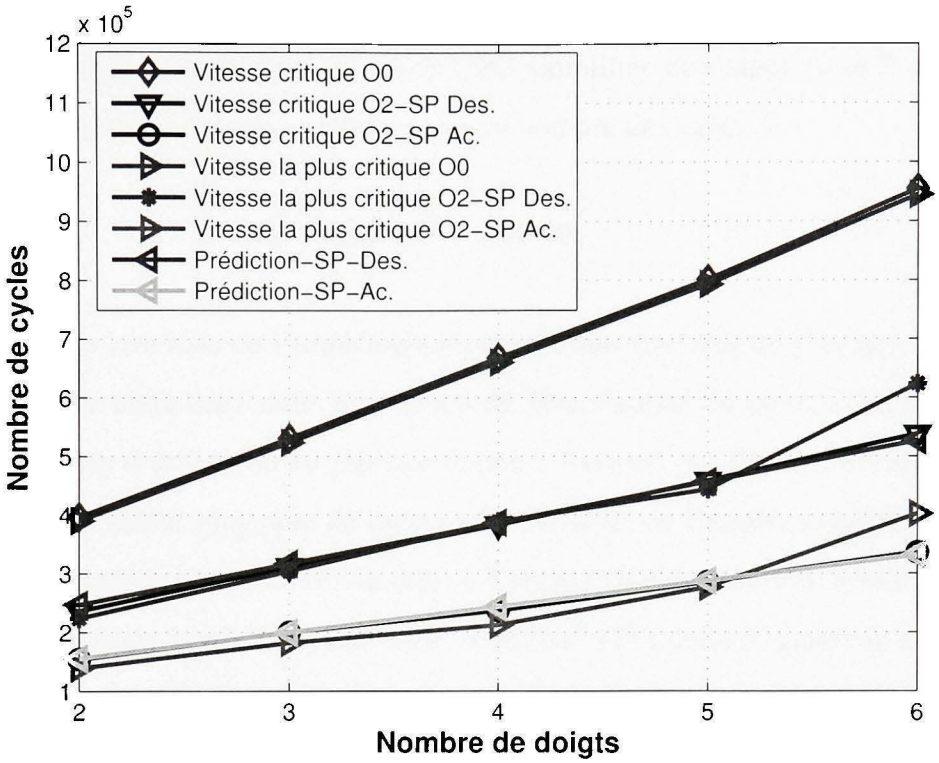


Figure 5.6 Comparaisons entre la prédiction et l'implémentation pour six options de simulation.

Les équations 5.7, 5.8 et 5.9, générales mais complexes sont simplifiées par la prise en considération d'une seule grandeur variable, le nombre de doigts du récepteur, à l'image du caractère fluctuant du nombre de trajets pendant une communication. Les autres grandeurs dont dépend l'algorithme sont généralement fixées par le standard de communication d'intérêt, en l'occurence l'UMTS dans ce cas. La Figure 6 compare les prédictions avec les résultats de l'implémentation obtenus pour six options de simulation en fixant L à 320, SF_I à 4 et SF_Q à 32. Cette figure laisse entrevoir la stabilité des différentes options considérées. Cependant, comme dans le cas du ZF-SQRD, l'option « vitesse critique -O2 SP Des. » est le meilleur choix en termes de stabilité et de conformité avec les prédictions que le pipeline logiciel soit ou non activé. En effet, en considérant un nombre de doigts variant entre deux et six, l'implémentation et la prédiction ne diffèrent que par 1% pour

SP Des. et SP Ac.. La Figure 6 est d'un grand apport pour simplifier les équations 5.7, 5.8 et 5.9. L'équation de prédiction du temps CPU simplifiée de l'algorithme Rake est alors donnée par l'équation linéaire 5.10 fonction du nombre de doigts N :

$$y = 69906N + 106990 \tag{5.10}$$

Par ailleurs, les résultats de l'implémentation prouvent une fois de plus qu'il existe une relation linéaire entre ceux issus de l'option de désactivation du pipeline logiciel et ceux issus de l'option d'activation du pipeline logiciel. Ceci est par ricochet transposable aux prédictions. La valeur moyenne du rapport des résultats de l'implémentation impliquant SP-Des. avec ceux impliquant SP-Ac. vaut 0.6 et est très comparable à la valeur 0.5 obtenue dans les mêmes conditions pour le ZF-SQRD. Il suffit donc d'ajouter le facteur 0.6 à l'équation 5.10 pour déduire l'équation des prédictions si le pipeline est actif.

5.3 L'algorithme Radix 2 DIT FFT

La librairie de primitives de la section 4.2 favorise une distinction entre la quantité de ressources matérielles requises pour implémenter une FFT en comptabilisant les multiplications par des facteurs de phase unitaires comparativement à celle les ignorant. En effet, une implémentation de l'algorithme en précision double, donc par génération automatique du code C par l'outil RTW, suggère au compilateur CCS de tenir compte de ces multiplications surnuméraires. Ainsi, en se plaçant dans le cas où les trames d'entrée sont réelles, la complexité matérielle de l'implémentation s'élève à $4Nlog_{10}(N)$ multiplications et additions réelles.

Les résultats de ces prédictions sont illustrés à la Figure 7. Le Tableau XV quant à lui présente les valeurs moyennes du ratio implémentation/prédiction pour les six options considérées. Comme pour les deux précédentes applications, l'option à vitesse critique -O2 avec SP désactivé sort gagnante en termes de conformité avec les prédictions (le rapport implémentation/prédiction vaut 1.15) et le niveau -O0 d'optimisation est le grand perdant

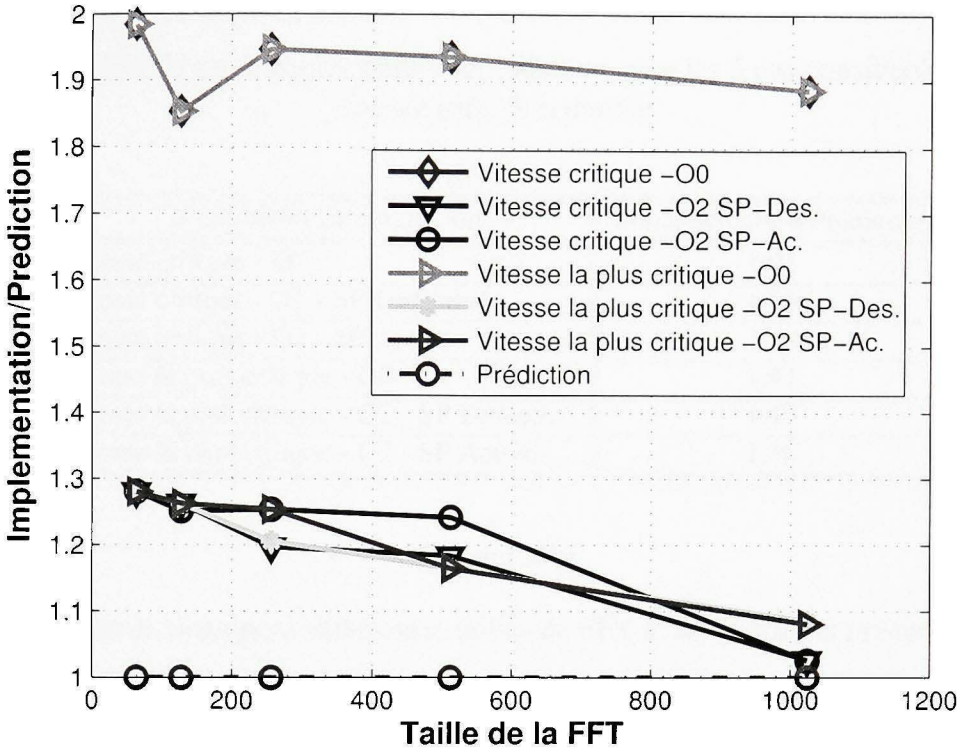


Figure 5.7 Comparaisons entre la prédiction et l'implémentation pour 5 tailles de FFT en précision double.

pour un niveau de vitesse critique et le plus critique. L'option *vitesse la plus critique -O2* fournit des résultats similaires à l'option de *vitesse critique* du même niveau d'optimisation. Force est de constater finalement que l'activation du pipeline logiciel ne change pas les résultats de l'implémentation. Elle ne fait pas d'optimisation. Ceci pourrait s'expliquer par le fait que l'algorithme en soi maximise l'utilisation des ressources matérielles du processeur.

Le Tableau XVI présente à titre d'information, les constantes de normalisation pour différentes tailles de FFT. Ces valeurs servent à rendre unitaire la courbe des prédictions de la Figure 7.

Tableau XV

Rapport moyen du ratio implémentation/prediction pour les 5 cas considérés selon le format précision double

Paramètres de simulation	Implémentation/Prédiction
Vitesse critique - O0	1.91
Vitesse critique - O2 - SP Désactivé	1.15
Vitesse critique - O2 - SP Activé	1.17
Vitesse la plus critique - O0	1.91
Vitesse la plus critique - O2 - SP Désactivé	1.17
Vitesse la plus critique - O2 - SP Activé	1.18

Tableau XVI

Valeurs des prédictions pour différentes tailles de FFT selon le format précision double

Taille	Prédictions
64	26112
128	60928
256	313344
512	696320
1024	1531904

Afin de vérifier que l'optimisation est bel et bien inexistante même après sélection du pipeline logiciel, une implémentation de l'algorithme en précision simple, codée manuellement dans le compilateur CCS a été réalisée. Par les résultats qu'elle fournit, cette implémentation est optimisée par rapport à la première car elle ne prend pas en considération les multiplications par les facteurs de phase unitaires. Ainsi, en se plaçant dans le cas où les trames d'entrée sont réelles, la complexité matérielle de l'implémentation s'élève à $2Nlog_{10}(N)$ multiplications réelles et à $3Nlog_{10}(N)$ additions réelles. Les résultats de ces prédictions sont illustrés à la Figure 8. La Table XVII expose les valeurs moyennes du ratio implémentation/prédiction pour les six options considérées. Le résultat précédent s'en trouve confirmé. L'activation du pipeline logiciel ne modifie pas les résultats de l'im-

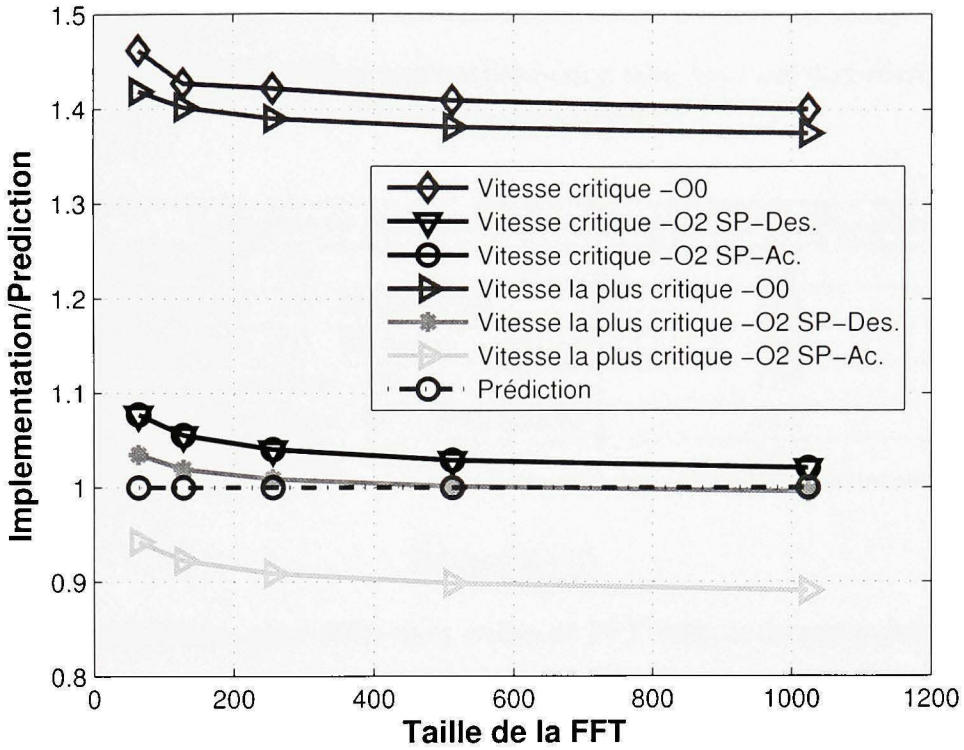


Figure 5.8 Comparaisons entre la prédiction et l'implémentation pour 5 tailles différentes de FFT en précision simple.

plémentation. De plus, comme pour les deux précédentes applications, l'option à vitesse critique -O2 avec SP désactivé est la meilleure.

Le Tableau XVIII présente à titre d'information, les constantes de normalisation pour différentes tailles de FFT. Ces valeurs servent à rendre unitaire la courbe des prédictions de la Figure 8.

5.4 L'impact du pipeline logiciel

L'étude de l'impact du pipeline logiciel aide à mieux comprendre et à établir, si lien il y a, une relation entre les résultats de profilage recueillis d'une part lorsque le pipeline logiciel est sélectionné et d'autre part lorsqu'il ne l'est pas. Le compilateur de Texas Ins-

Tableau XVII

Rapport moyen du ratio implémentation/prediction pour les 5 cas considérés selon le format précision simple

Paramètres de simulation	Implémentation/Prédiction
Vitesse critique - O0	1.42
Vitesse critique - O2 - SP Désactivé	1.04
Vitesse critique - O2 - SP Activé	1.04
Vitesse la plus critique - O0	1.39
Vitesse la plus critique - O2 - SP Désactivé	1.01
Vitesse la plus critique - O2 - SP Activé	0.91

Tableau XVIII

Valeurs des prédictions pour différentes tailles de FFT selon le format précision simple

Taille	Prédictions
64	7680
128	17920
256	92160
512	204800
1024	450560

truments, Code Composer Studio, emploie le pipeline logiciel pour établir un équilibre entre la taille du code implémenté et ses performances. Le profilage de chaque section des trois algorithmes faisant l'objet de ce travail met en lumière la compréhension de cet impact. Celui-ci a en effet permis de faire la distinction entre deux types d'optimisation : celle réalisée pour les opérations de transfert-mémoire et celle s'inscrivant dans le cadre des opérations mathématiques. La Figure 9 illustre un exemple d'opération de transfert-mémoire (décalage) et la Figure 10, un exemple d'opération mathématique.

Le passage à la loupe par profilage de ces deux types d'opérations mène à la conclusion selon laquelle, pour les transferts-mémoires, le gain en optimisation après activation du pipeline est de 10%. Ceci sous-tend que l'optimisation pour ce type d'opération est moindre.

```
for n= 1: LenSampling*SF_data
    for m= 1:SF_DPDCH
        CodeI(ind) = CodeI1(m);
        ind = ind+1;
    end
end
```

Figure 5.9 Exemple de code pour les opérations de transfert-mémoire.

```
for ii = 1:NumFingers
        for jj = 1:LenSampling
            OutScr(jj,ii) = SCcode(jj)*output(jj,ii);
            OutSpreadI(jj,ii) = CodeI(jj)*real(OutScr(jj,ii));
            OutSpreadQ(jj,ii) = CodeQ(jj)*imag(OutScr(jj,ii));
        end
end
```

Figure 5.10 Exemple de code pour les opérations mathématiques.

Au contraire, pour les opérations mathématiques, le même gain est aussi élevé que 70%. Ainsi, pour une application donnée, le rapport entre les résultats obtenus sur activation du pipeline logiciel avec ceux obtenus sur désactivation de ladite option oscille entre 0.3 et 0.9. Le ratio un pour deux (section 5.1) de l'algorithme ZF-SQRD et celui de 0.6 pour le récepteur Rake (section 5.2) s'expliquent par une présence relativement équilibrée de ces deux types d'opérations dans chacun des algorithmes. Pour ce qui est du cas de la FFT, une explication de l'absence totale d'optimisation pour cette application lorsque le pipeline logiciel est sélectionné pourrait provenir du fait que la version de l'algorithme employée, l'architecture papillon, est déjà optimisée en soi pour une implémentation matérielle en

terme d'ordonnancement des calculs et ne laisse par conséquent qu'une contribution bien mince à l'amélioration par le compilateur.

5.5 Conclusion

En résumé, les prédictions concordent avec les implémentations pour le niveau 2 d'optimisation (-O2) avec *vitesse critique* et pipeline logiciel absent. Cette dernière offre aussi une stabilité relativement bonne lors de l'obtention des résultats de l'implémentation. L'option nommée *vitesse la plus critique - O2 SP-Des.* procure des résultats qui se rapprochent de ceux de l'option *vitesse critique - O2 SP-Des.* mais ils sont moins stables que cette dernière. Le niveau d'optimisation 0 n'est pas une option souhaitable car elle fournit des résultats éloignés des prédictions que la vitesse soit critique ou supérieure. Finalement, le pipeline logiciel introduit un gain d'optimisation dont il faut tenir compte lors des prédictions. Celui-ci se situe entre 10% et 70% respectivement pour des opérations de transfert-mémoire et mathématiques. Pour une application donnée, l'estimation de la valeur de ce gain peut se faire en affectant un poids au nombre d'opérations de transfert-mémoire et au nombre d'opérations mathématiques présentes dans l'application et en faisant une combinaison linéaire de ces poids affectés des nombres d'opérations respectifs. La FFT est un exemple d'application dont l'algorithme est optimisé pour une implémentation matérielle et que le pipeline logiciel affecte dans une moindre mesure.

CONCLUSION

Ce mémoire a été consacré à l'élaboration d'une méthodologie d'extraction d'opérations usuelles en télécommunications résultant en une librairie de primitives. Trois applications constituent le socle de ce travail. Elle font partie de deux grands groupes de systèmes de communication : les systèmes MIMO d'une part et les liaisons point à multipoints d'autre part. Ces systèmes font l'objet du premier chapitre. Après un survol du principe des systèmes MIMO, un rappel sur les notions de base en communications numériques notamment les canaux de communication, gaussien et de Rayleigh suit. Dans la même lancée, les techniques de diversité sont présentées ainsi que la notion de capacité. L'étude de la capacité d'un système MIMO en présence d'une source gaussienne permet d'observer sa croissance linéaire en fonction du nombre d'émetteurs. Un autre point développé concerne les techniques d'accès multiples telles que le CDMA et l'OFDMA. Ces techniques favorisent l'optimisation de l'utilisation des resources du système entre plusieurs utilisateurs.

Si l'objectif de la troisième génération de systèmes de radio-mobiles est la flexibilité des services, la quatrième génération vise quant à elle des efficacités spectrales élevées. Les systèmes MIMO constituent une solution envisageable. Des études théoriques ont démontré que le paramètre à optimiser pour ces systèmes est la diversité au récepteur. Dans la première partie du chapitre consacré aux trois applications étudiées, il a été question de présenter l'algorithme ZF-SQRD et les raisons de son choix après comparaison de ses performances avec celles des algorithmes DBLAST et ZF-VBLAST. En effet, l'algorithme DBLAST atteint la capacité de Shannon au prix d'une complexité de décodage accrue due à la diversité spatiale introduite par la diagonalisation des trames au transmetteur. Par ailleurs, l'algorithme ZF-VBLAST, basé sur le principe d'annulation successive d'interférences, permet de diminuer cette complexité mais avec une diversité sous-optimale. La complexité du ZF-VBLAST dépend uniquement du nombre d'antennes émettrices (N_t) et réceptrices (N_r). Néanmoins, pour N_t fixe, lorsque N_r croît, l'écart de performances

entre les deux détecteurs décroît. C'est aussi le cas pour l'algorithme ZF-SQRD, qui, en plus nécessite moins de ressources matérielles que le ZF-VBLAST puisqu'il minimise les opérations de divisions et les inversions de matrices. D'où son choix. La seconde partie du chapitre s'étend sur le récepteur Rake appliqué dans les systèmes mobiles de troisième génération dans un contexte UMTS. Ce dernier offre une large gamme de débits. Le Rake tire profit de la propagation multitrajets pour être performant. Son principe consiste à faire une dérotation des trames d'entrée, à les désaligner, à les désembrouiller, à les désétaler, puis en employant la méthode de combinaison linéaire optimale et la quantification, à produire une estimation du signal de départ. Enfin, la troisième application est la FFT dont la rapidité s'explique par l'exploitation des propriétés de périodicité et de symétrie exhibées par ses facteurs de phase et par l'utilisation de l'approche diviser pour conquérir. La forme radix 2 de la FFT, dont la longueur des vecteurs d'entrée est une puissance de deux, a été présentée de manière explicite.

Le troisième chapitre s'attarde sur la structure du processeur utilisé pour élaborer les primitives struturelles. Il débute par une présentation générale de l'architecture des processeurs de la famille TMS320C6000 qui sont des processeurs VLIW. La mémoire totale du processeur C6711 qui vaut 4 GB est ensuite présentée. Elle a établi une distinction entre la mémoire interne des données et celle des programmes. Les modes d'adressage disponibles au sein du processeur sont l'adressage indirect et l'adressage circulaire. Comme le C6711 est un processeur à virgule flottante, alors une présentation de la représentation des nombres en point flottant n'a pas été pas de trop. Celle-ci peut se faire en précision simple ou en précision double. La précision double est le format utilisé par Matlab® dans ses calculs. D'où son intérêt dans notre étude. La description du jeu d'instructions a permis de connaître les primitives structurelles du C6711 et les accès-mémoires qui y sont associés. Ces primitives sont reliées aux opérations usuelles du côté application. Enfin, les techniques d'optimisation ont clôturé le chapitre. La technique de pipeline logiciel est celle qui affecte la précision des résultats de l'implémentation. D'autres méthodes sont aussi

présentées, dont l'optimisation par boucle de déroulage qui réduit le coût des opérations en dupliquant plusieurs fois le contenu d'une boucle.

La première partie du quatrième chapitre propose la méthodologie d'extraction employée afin d'aboutir à une librairie de primitives. Cette méthodologie nécessite à la fois la compréhension de l'application ciblée et celle de l'architecture de la plateforme d'intérêt d'où la librairie qui en a découlé. Une procédure de validation en partie automatisée, s'en est suivie en dernier lieu. Celle-ci conforte la pertinence des équivalences au sein de la librairie.

Finalement, le cinquième et dernier chapitre fait part des résultats de validation recueillis après implémentation et les compare aux équations de prédiction dérivées à partir de la librairie de primitives. Les résultats dépendent du choix des options de simulation. Celles-ci prennent en considération le niveau d'optimisation à atteindre par le compilateur. L'option qui s'est avérée meilleure est le niveau 2 d'optimisation. Un autre facteur de poids dont il faut tenir compte dans l'obtention des résultats est la présence ou l'absence du pipeline logiciel. Il favorise une réduction du résultat escompté de 10% dans le pire cas à 70% dans le meilleur. En général, la réduction tourne autour de 50%.

Au terme de ces travaux, il apparaît que les systèmes MIMO et la transformée rapide de Fourier ont un avenir prometteur dans les systèmes de communication numérique de quatrième génération, en l'occurrence le Wimax. Ceci en dépit du fait que les études réalisées supposent d'une part une connaissance parfaite du canal et des délais de la propagation multitrajets, et, d'autre part, une présence négligeable des interférences intersymboles, les rendant optimistes par rapport à la réalité.

Il en ressort des conclusions ci-dessus qu'une avenue possible à ce projet est l'amélioration de la flexibilité et des performances des systèmes multiprocesseurs. Celle-ci passe par l'élaboration d'une stratégie de développement des designs dérivés. En effet, la prédiction d'une métrique telle que le nombre de cycles, permet de savoir, sans avoir à implémenter

un design, si les capacités de la puce sélectionnée sont supérieures ou non à celles de l'application et ainsi, d'en identifier le goulot d'étranglement s'il y en a un. Si les capacités de la puce favorisent une implémentation, alors il serait possible de modifier le design de départ pour déduire avec aisance le comportement des designs dérivés dans le processeur.

Une autre portée de ce projet pourrait intervenir dans le partionnement optimal des tâches au sein d'une plateforme à plusieurs puces. L'intérêt de se servir d'un système à plusieurs processeurs réside dans la volonté de minimiser le temps général de traitement d'une application, la consommation de puissance, l'utilisation de la mémoire ... et de maximiser les performances. Avant d'être applicables dans un contexte plus général impliquant plusieurs processeurs, ces objectifs nécessitent entre autres une bonne estimation du temps CPU nécessaire à une portion quelconque de l'application pour être exécutée dans un seul processeur. L'estimation fiable de la métrique *temps d'exécution* devient ainsi un point d'ancrage pour un partitionnement rentable.

BIBLIOGRAPHIE

Alamouti, S. M. 1998. « A simple transmit diversity technique for wireless communications ». Selected Areas in Communications, IEEE Journal on, vol. 16, no 8, p. 1451-1458.

Benedetto, Sergio, et Ezio Biglieri. 1999. Principles of digital transmission : with wireless applications. Coll. « Information technology : transmission, processing, and storage ». New York : Kluwer Academic/Plenum Press, xvii, 855 p. p.

Biglieri, Ezio. 2007. MIMO wireless communications. Cambridge : Cambridge University Press, xvii, 323 p. p.

Chassaing, Rulph. 2002. DSP applications using C and the TMS320C6x DSK. Coll. « Topics in digital signal processing ». New York : J. Wiley, xix, 335 p. p.

Chuah, Chen-Nee, Joseph M. Kahn et David Tse. 1998. « Capacity of multi-antenna array systems in indoor wireless environment ». Conference Record / IEEE Global Telecommunications Conference, vol. 4, p. 1894-1899.

Da-Shan, Shiu, G. J. Foschini, M. J. Gans et J. M. Kahn. 2000. « Fading correlation and its effect on the capacity of multielement antenna systems ». Communications, IEEE Transactions on, vol. 48, no 3, p. 502-513.

Driessen, P. F., et G. J. Foschini. 1999. « On the capacity formula for multiple input-multiple output wireless channels : a geometric interpretation ». IEEE Transactions on Communications, vol. 47, no 2, p. 173-176.

Elliott, Douglas F., et K. Ramamohan Rao. 1982. Fast transforms : algorithms, analyses, applications. New York : Academic Press, xxii, 488 p. p.

ETSI. 2004. « Universal Mobile Telecommunications System (UMTS) ; Physical layer - general description (3GPP TS 25.201 version 6.1.0 Release 6) ».

ETSI. 2005. « Third Generation Partnership Project Technical Specification Group Radio Access Network Working Group 1 : Spreading and Modulation ». Sophia Antipolis, FRANCE :

Farrokhi, F. R., A. Lozano, G. J. Foschini et R. A. Valenzuela. 2002. « Spectral efficiency of FDMA/TDMA wireless systems with transmit and receive antenna arrays ». Wireless Communications, IEEE Transactions on, vol. 1, no 4, p. 591-599.

Foschini, G. J., et M. J. Gans. 1998. « On limits of wireless communications in a fading environment when using multiple antennas ». Wireless Personal Communications, vol. 6, no 3, p. 311-35.

Gerard J.Foschini. 1996. « Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas ». Bell Labs Technical Journal, vol. 1, no 2, p. 41-59.

Golden, G. D., C. J. Foschini, R. A. Valenzuela et P. W. Wolniansky. 1999. « Detection algorithm and initial laboratory results using V-BLAST space-time communication architecture ». Electronics Letters, vol. 35, no 1, p. 14-16.

Golub, Gene H., et Charles F. Van Loan. 1996. Matrix computations, 3rd. Baltimore : Johns Hopkins University Press, xxvii, 694 p. p.

Haiyan, J. , A. Nilsson, E. Tell et D. Liu. 2006. « MIPS cost estimation for OFDM-VBLAST systems ». In 2006 IEEE Wireless Communications and Networking Conference. (3-6 April 2006). p. 822-826. IEEE.

Huang, H., H. Viswanathan et G. J. Foschini. 2002. « Multiple antennas in cellular CDMA systems : transmission, detection, and spectral efficiency ». Wireless Communications, IEEE Transactions on, vol. 1, no 3, p. 383-392.

Huang, J. C., et T. Leng. 1999. « Generalized loop-unrolling : a method for program speedup ». In Application-Specific Systems and Software Engineering and Technology, 1999. ASSET '99. Proceedings. 1999 IEEE Symposium on. p. 244-248.

Hughes , C. J., P. Kaul, S. V. Adve, R. Jain, C. Park et J. Srinivasan. 2001. « Variability in the execution of multimedia applications and implications for architecture ». In Proceedings of 28th Annual International Symposium on Computer Architecture. p. 254-265.

Kehtarnavaz, Nasser. 2005. Real-time digital signal processing based on the TMS320C6000. Amsterdam ; Boston : Elsevier ; Newnes, xii, 306 p. p.

Kruger, Reinhold, et Heinz Mellein. 2004. UMTS : Introduction and Measurements. Munchen : Rohde et Schwartz, 275 p.

Kuo, Sen M., Bob H. Lee et Wenshun Tian. 2006. Real-time digital signal processing : implementations and applications, 2nd. Chichester, England ; Hoboken, NJ : John Wiley, xvii, 646 p.

Lam, M. 1988. « Software pipelining : an effective scheduling technique for VLIW machines ». In SIGPLAN '88 Conference on Programming Language Design and Implementation, 22-24 June 1988. Vol. 23, p. 318-28.

Lapsley, P. , et G. Blalock. 1996. « How to estimate DSP processor performance ». Spectrum, IEEE, vol. 33, no 7, p. 74-78.

Liang, Zheng, Xu-Bang Shen et Zuo-Hui Peng. 2001. « The application of redundant encoding in iterative implementation of division and square root ». In ASIC, 2001. Proceedings. 4th International Conference on. p. 603-606.

Liu, Z., G. B. Giannakis, S. Zhou et B. Muquet. 2001. « Space - Time coding for broadband wireless communications ». Wireless Communications and Mobile Computing, vol. 1, no 1, p. 35-53.

Marzetta, T. L., et B. M. Hochwald. 1998. « Fundamental limitations on multiple-antenna wireless links in Rayleigh fading ». In., p. 310.

Marzetta, Th L., et B. M. Hochwald. 1999. « Capacity of a mobile multiple-antenna communication link in Rayleigh flat fading ». IEEE Transactions on Information Theory, vol. 45, no 1, p. 139-157.

Moore, E.H. 1920. « On the reciprocal of the general algebraic matrix ». Bulletin of the American Mathematical Society, no 26, p. 394-395.

Naguib, A. F., N. Seshadri et A. R. Calderbank. 2000. « Increasing data rate over wireless channels ». Signal Processing Magazine, IEEE, vol. 17, no 3, p. 76-92.

Papadimitriou, P. D., N. A. I. Livanos et E. Zigouris. 1997. « An integrated environment for real-time implementation of DSP algorithms ». In. Vol. 2, p. 1015-1018 vol.2.

Penrose, Roger. 1955. « A genaralized inverse for matrices ». Proceedings of the Cambridge Philosophical Society, no 51, p. 406-413.

Price, R., et Jr P. E. Green. 1958. « Communication technique for multipath channels ». Institute of Radio Engineers – Proceedings, vol. 46, no 3, p. 555-570.

Proakis, John G. 1995. Digital communications, 3rd. Coll. « McGraw-Hill series in electrical and computer engineering ». New York, N.Y. : McGraw-Hill, xxi, 928 p.

Proakis, John G., et Dimitris G. Manolakis. 1996. Digital signal processing : principles, algorithms, and applications, 3rd. Upper Saddle River, NJ. : Prentice Hall, xv, 968, [48] p. p.

Sellathurai, M., et S. Haykin. 2000. « TURBO-BLAST for high-speed wireless communications TURBO-BLAST for high-speed wireless communications ». In Wireless Communications and Networking Conference, 2000. WCNC. 2000 IEEE. Vol. 1, p. 315-320 vol.1.

Shannon, C. E. 1948. « Mathematical theory of communication ». Bell System Technical Journal, vol. 27, no 3-4.

Skinner, D., et W. Kramer. 2005. « Understanding the causes of performance variability in HPC workloads ». In Proceedings of the IEEE International Workload Characterization Symposium. (6-8 octobre 2005). p. 137-149.

Telatar, Emre. 1999. « Capacity of multi-antenna Gaussian channels ». European Transactions on Telecommunications, vol. 10, no 6, p. 585-595.

Texas Instruments. 2007, 05 Octobre. http ://focus.ti.com/paramsearch/docs

Texas Instruments 2000. « TMS320C6000 CPU and Instruction Set ». SPRU189F. Dallas (Texas) : Soyink.

Texas Instruments. 2005. « TMS320C6000 Optimizing Compiler v 6.0 Beta ». SPRU187G. Dallas (Texas) : Soyink.

Texas Instruments. 2001. « TMS320C6000 Peripherals Reference Guide ». SPRU190D. Dallas (Texas) : Soyink.

Texas Instruments. 2006. « TMS320C6000 Assembly Language Tools v 6.0 Beta ». SPRU186P. Dallas (Texas) : Soyink.

Texas Instruments. 2006. « TMS320C6000 Programmer's Guide ». SPRU198I. Dallas (Texas) : Soyink.

Vucetic, Branka, et Jinhong Yuan. 2003. Space-time coding. West Sussex, England ; Hoboken, NJ : Wiley, 302 p.

Wang, Vincent, et Ki-Soo Lee. 2005. « Automated Regression Tests and Measurements with the CCStudio Scripting Utility ». Texas Instruments.

Winters, J. H., J. Salz et R. D. Gitlin. 1994. « The impact of antenna diversity on the capacity of wireless communication systems ». Communications, IEEE Transactions on, vol. 42, no 234, p. 1740-1751.

Wolniansky, P. W., G. J. Foschini, G. D. Golden et R. A. Valenzuela. 1998. « V-BLAST : an architecture for realizing very high data rates over the rich-scattering wireless channel ». In Conference Proceedings of the International Symposium on Signals, Systems and Electronics. p. 295-300. Pisa, Italy : IEEE, Piscataway, NJ, USA.

Wozencraft, John M., et Irwin Mark Jacobs. 1965. Principles of communication engineering. New York : Wiley, xv, 720 p. p.

Yaghoobi, H. 2004. « Scalable OFDMA physical layer in IEEE 802.16 WirelessMAN ». Intel Technology Journal, no 3.