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
SÉBASTIEN JOMPHE

RÉALISATION ET VALIDATION EN VHDL D'UN RÉCEPTEUR DE STATION DE BASE À MODULATION EN ÉTALEMENT DE SPECTRE

MONTRÉAL, LE 20 DÉCEMBRE 2005

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Jean Belzile, directeur de mémoire
Département de génie électrique à l'École de technologie supérieure

M. Sofiène Affes, codirecteur
Institut national de la recherche scientifique, matériaux, énergie et télécommunications

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

M. Alex Stéphenne, membre du jury
Institut national de la recherche scientifique, matériaux, énergie et télécommunications

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 24 NOVEMBRE 2005

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

RÉALISATION ET VALIDATION EN VHDL D'UN RÉCEPTEUR DE STATION DE BASE À MODULATION EN ÉTALEMENT DE SPECTRE

SÉBASTIEN JOMPHE

SOMMAIRE ABRÉGÉ

Différents types de récepteurs RAKE sont généralement employés pour combattre les effets dispersifs d'un canal de téléphonie cellulaire, mais les hauts débits et le contexte multi-usagers que proposent les standards actuellement développés entraînent invariablement des pertes de performances. Le récepteur STAR, « Spatio-Temporal Array-Receiver », permet de mitiger ces dégradations, au prix d'une complexité algorithmique supérieure.

Ce mémoire propose donc une architecture microélectronique et un protocole de réalisation « codesign » qui ont permis, pour la première fois, la réalisation sur FPGA du récepteur STAR. La validation en temps réel, sur lien sans fil, démontre que les performances ainsi obtenues s'approchent de celles du modèle théorique. Enfin, une station de base employant cette technologie est considérée, sur un seul FPGA, suggérant une réduction considérable de taille comparativement aux solutions RAKE/DSP contemporaines.

RÉALISATION ET VALIDATION EN VHDL D'UN RÉCEPTEUR DE STATION DE BASE À MODULATION EN ÉTALEMENT DE SPECTRE

Sébastien Jomphe

SOMMAIRE

Les communications en étalement spectral sont utilisées à grand déploiement depuis l'adoption du standard cellulaire civil IS-95A en Amérique du nord. Pour augmenter la capacité des réseaux sans fil et permettre la diffusion de contenu numérique multimédia, le standard 3GPP a été développé. Il spécifie une bande passante supérieure, ce qui entraîne des débits de données plus élevés. Ce type de transmission impose cependant des contraintes sur le récepteur RAKE classique et exige sa révision pour diminuer les dégradations inter-symboles, inter-usagers et multi-trajets.

L'objet de ce mémoire consiste à réaliser le récepteur STAR (« Spatio-Temporal Array-Receiver ») sur une plate-forme matérielle, en temps réel. Pour y arriver, l'algorithme STAR est d'abord analysé et quantifié. Une architecture répartie entre matériel et logiciel, ou « codesign », est ensuite établie, et les modules résultants sont finalement réalisés en langage VHDL et C, respectivement.

Les résultats démontrent qu'il est techniquement possible de réaliser STAR suivant une architecture « codesign ». Le modèle unifié de simulation prouve la faisabilité d'utiliser un seul et même ensemble de vecteurs de test pour valider le modèle de référence, sa version quantifiée, la simulation VHDL et le prototype final. La dégradation des performances causée par certaines hypothèses simplificatrices s'avère insignifiante après comparaison des résultats. Enfin, la taille physique du récepteur est évaluée de façon précise. Elle est alors comparée à une analyse sur le RAKE faite par un manufacturier de composants électroniques, et indique que STAR a le potentiel d'être utilisé comme récepteur complet et intégré sur FPGA pour une station de base cellulaire de troisième génération.

VHDL IMPLEMENTATION AND VALIDATION OF A SPREAD SPECTRUM BASE STATION RECEIVER

Sébastien Jomphe

ABSTRACT

Spread spectrum communications have gained widespread acceptance in north America ever since the release of the IS-95 cellular standard. The growing demand for digital multimedia content has stressed the need to expand wireless network capacity. As such, third generation cellular standards such as 3GPP plan wider bandwidths, thus higher binary throughput, which tend to strain the conventional RAKE receiver to the point where ISI, MUD and multipath enhancements are mandatory for successful reception.

This thesis aims at implementing the computationally-intensive STAR algorithm (Spatio-Temporal Array-Receiver) as a hardware system to meet real-time 3G communications requirements. To this end, the algorithm is first analyzed and quantized. A suitable hardware/software codesign architecture is then set forth, after which the hardware and software building blocks are respectively coded in VHDL and C.

The end-results demonstrate that the suggested architecture can handle the complex STAR algorithm. A set of test vectors borrowed from the original golden reference model is used to test each development stage (quantization, VHDL simulation and FPGA prototype) and clearly shows that performance degradations caused by hardware considerations and design decisions are negligible. Furthermore, through precise area utilization measurements and comparison with a manufacturer's own RAKE receiver assessment, STAR is shown to be a viable alternative as an integrated, multi-user, FPGA-based 3G base station receiver.

REMERCIEMENTS

Je tiens à exprimer toute ma reconnaissance au Pr. Belzile de l'École de technologie supérieure ainsi qu'au Pr. Affes de l'Institut national de recherche scientifique pour m'avoir soutenu et guidé tout au long de cet ambitieux projet de maîtrise.

Je remercie M. Karim Cheikhrouhou, Ph.D. à l'Institut national de la recherche scientifique, du support qu'il m'a accordé pour la compréhension de l'algorithme, ainsi que lors de l'adaptation de ses modèles de simulation originaux.

Je remercie également Mme Marie-Eve Grandmaison, de l'École de technologie supérieure, qui a contribué son module R2^{2}PC et l'a adapté pour les bienfaits de ce projet.

Finalement, je remercie M. Jean-Claude Thibault, d'ISR Technologies, pour sa collaboration indéfectible lors de la réalisation du prototype final, dévoilé publiquement à la démonstration industrielle d'envergure PROMPT-Québec d'avril 2005 (voir annexe 4).

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET SIGLES

ADC	« Analog to Digital Converter » (convertisseur analogique à numérique)
BER	« Bit Error Rate » (taux d'erreur binaire)
dB	Décibel
EQM	Erreur quadratique moyenne
FPGA	« Field Programmable Gate Array » (réseau programmable de logique universelle)
DAC	« Digital to Analog Converter » (convertisseur numérique à analogique)
SNR	« Signal-to-Noise Ratio » (rapport signal-à-bruit)
Mcps	« Mega chips per second » (méga bribes / s)
Msps	Méga symboles / s
PCM	« Post-Correlation Model » (modèle post-corrélation)
PM	« Path Management » (gestion des trajets)
RX	Récepteur, réception
SP	« Symbol Path » (décodage des symboles)
STAR	« Spatio-Temporal Array Receiver »
STRF	« Structure Fitting » (régularisation)
TR	« Tracking Ratio » (taux de poursuite)
TX	Transmetteur, transmission

INTRODUCTION

Les communications en spectre étalé, développées peu après la seconde guerre mondiale, offraient aux militaires de l'époque une protection contre le brouillage et l'espionnage. À l'ère du téléphone cellulaire où le spectre radiofréquence (RF) est partagé par plusieurs modulations concurrentes, l'étalement spectral intéresse plutôt par sa robustesse aux dégradations encourues par le canal et son extension naturelle à l'accès multi-usagers. Bien que le récepteur RAKE demeure le standard de facto pour la réception d'un tel signal, sa taille et ses performances le rendent peu attrayant pour les liens à haut débit et les systèmes multi-antennes qui font leur apparition.

C'est dans ce contexte que s'inscrit le récepteur spatio-temporel STAR (« Spatio-Temporal Array-Receiver ») développé par Affes et Mermelstein (1997). Le principe de réception de STAR est innovateur et lui confère à la fois les fonctionnalités d'un récepteur à part entière et celles d'un d'instrument de mesure du canal. Des évaluations théoriques suggèrent qu'il a la capacité de déclasser les solutions basées sur le RAKE. L'introduction de réseaux d'antennes réceptrices et la détection multi-usagers font partie intégrante de STAR. Il revêt cependant une complexité arithmétique généralement supérieure à celle d'un récepteur RAKE classique.

Le mandat est donc, dans un premier temps, de confirmer les estimations de complexité algorithmique déjà publiées et de vérifier la faisabilité de réaliser STAR dans un circuit logique programmable FPGA à l'aide des hypothèses simplificatrices de Cheikhrouhou (2001). En second lieu, l'algorithme doit être quantifié, sectionné et adapté à une architecture matérielle adéquate qui doit par ailleurs être définie. En troisième et dernier lieu, le récepteur doit être réalisé sur la plate-forme choisie et fonctionner en temps réel afin d'en valider les performances théoriques dans des situations de transmission réelles.

Le mémoire est divisé en six chapitres. Le premier propose un survol des principes de modulation à étalement spectral et des dégradations causées par le canal de transmission.

Certaines notions et techniques d'architecture de systèmes microélectroniques sont aussi abordées. Le second présente la théorie de base ainsi que les équations constituant l'algorithme STAR. Le chapitre 3 effectue ensuite l'analyse de complexité de l'algorithme et suggère une architecture codesign pour réaliser efficacement le récepteur. La quantification des opérandes en notation à virgule fixe y est aussi effectuée. Enfin, les chapitres 4 et 5 donnent, à tour de rôle, les réalisations en langage VHDL et C des modules arithmétiques sur FPGA et microprocesseur embarqué, respectivement. Enfin, le sixième et dernier chapitre évalue les performances du prototype STAR ainsi que l'impact de certaines hypothèses simplificatrices. La taille physique finale est aussi évaluée afin de préciser dans quelle mesure la solution proposée est viable en tant que récepteur de station de base.

CHAPITRE 1

L'ÉTALEMENT SPECTRAL

1.1 Introduction

Ce premier chapitre offre une mise en contexte des sujets abordés dans ce mémoire. Il vise en premier lieu à familiariser le lecteur avec les techniques de communications à étalement spectral civiles utilisées en téléphonie cellulaire. Ce premier sujet est abordé en deux étapes. Dans un premier temps, on situe l'étalement spectral parmi les grandes familles de modulation et l'on définit le canal de transmission et les détériorations qu'il entraîne. En second lieu, les différents types de récepteur sont présentés, portant une attention particulière sur le RAKE. Diverses améliorations pour combattre les effets néfastes du canal sont ensuite passées en revue. Des courbes de performances viennent clore cette revue et serviront de barème de comparaison pour le récepteur STAR.

Le second volet du chapitre présente d'abord les techniques de conception matérielle actuellement employées dans la réalisation de systèmes de traitement du signal. Les solutions de réseau de logique programmable universelle (« field-programmable gate array » ou FPGA), de processeur numérique du signal (« digital signal processor » ou DSP) et de système-sur-puce (« system on chip » ou SoC) sont évoquées avec divers exemples où chacune se démarque. La section suivante explique les enjeux et avantages d'une conception dite « codesign » pour la réalisation d'un algorithme complexe et hétérogène. Le chapitre se termine par un tour d'horizon des pratiques de conception de systèmes de calcul parallèles.

1.2 Communications sans fil

Le nombre de téléphones cellulaires dans le monde a connu une hausse fulgurante au cours des dix dernières années. En 1991, le nombre d'appareils actifs, de technologie

analogique, était de moins de 20 millions selon Ojanperä (1998). Plunkett Research (2005) recensait plus de 190 millions de téléphones cellulaires actifs (essentiellement numériques) uniquement aux États-Unis en 2004, ce qui représente un taux de pénétration de 61%, alors que la Grande-Bretagne, la Suède et les Pays-Bas obtiennent un taux de 100% pour la même année. De plus, un faible pourcentage (6%) des foyers avaient déjà mis un terme à leur abonnement au téléphone classique (« landline ») en 2004, n'utilisant plus que les services sans fil. La firme Plunkett Research note une hausse appréciable de cette tendance pour 2005.

On estime enfin qu'en 2007, près de 1,2 milliards d'abonnés cellulaires auront accès à des services numériques basés sur Internet par le truchement d'appareils sans fil. Cette croissance anticipée entraîne la multiplication des fournisseurs de services, des stations de base, et met en relief la nécessité de développer des solutions aux problèmes de capacité, de débit et d'accès multiple aux réseaux.

1.2.1 Transmission

1.2.1.1 Classes de modulation

La modulation à étalement spectral (« Code Division Multiple Access » ou CDMA) permet de se prémunir contre certains effets destructifs du canal puisqu'elle exploite à la fois les dimensions spatiale et temporelle du spectre. Contrairement aux transmissions par multiplexage temporel (« Time Domain Multiple Access » ou TDMA) ou fréquentiel (« Frequency Domain Multiple Access » ou FDMA) qui sont respectivement sensibles à l'évanouissement fréquentiel et temporel, la modulation CDMA utilise les deux dimensions pour accroître la diversité de transmission et ainsi améliorer ses performances. La figure 1 donne une représentation graphique des modulations TDMA, FDMA et CDMA.

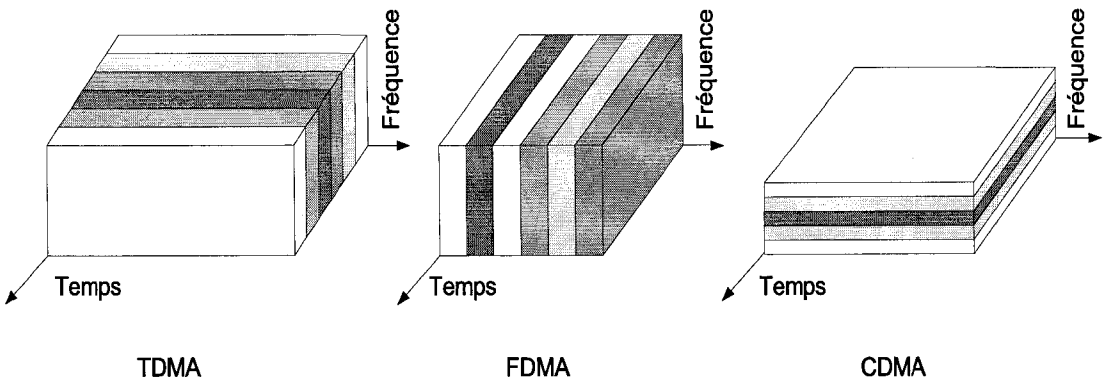


Figure 1 Trois grandes classes de modulation

Proakis (1995) raisonne ainsi l'utilité d'une diversité accrue: si la probabilité d'un évanouissement est p sur un seul canal de transmission, la probabilité $\tilde{p}$ que n canaux indépendants transportant la même information subissent au même moment un évanouissement irrécupérable se voit réduite à

$$\tilde{p} = p^n . \tag{1.1}$$

Différentes variantes de CDMA existent. La modulation DS-CDMA (« direct sequence CDMA ») utilise des séquences pseudo-aléatoires (« pseudo-noise » ou PN) quasi-orthogonales pour étaler les messages binaires à transmettre. La technique de saut de fréquence FH-CDMA (« frequency-hopping CDMA ») utilise plutôt ces mêmes séquences pour faire rapidement varier la fréquence centrale de la porteuse du message binaire. Les systèmes cellulaires de troisième génération utilisent la variante DS-CDMA (3GPP, 2000), où la séquence PN multiplie un message binaire, tel que démontré à la figure 2.

Le message binaire est constitué de bits, alors que la séquence étalée comporte des bribes (« chips » en anglais). Le débit du message original est défini par sa période T_S , alors que celui du message étalé est défini par la période de bribe T_C, fixée par convention, et de laquelle on peut déduire le facteur d'étalement L tel que

$$L=\frac{T_S}{T_C}\,, \tag{1.2}$$

de manière à ce que L bribes remplacent dans la séquence étalée un symbole de durée T_S provenant du message binaire. La spécification 3GPP fixe le taux de bribes à 3,84 x 10^6 bribes / s. La période d'une bribe est donc:

$$T_C=\frac{1}{3{,}84\,10^6\ chips/s}=260\text{ns}. \tag{1.3}$$

Remplaçant l'équation 1.3 dans 1.2, on déduit une série des débits permis par le standard 3GPP. Ces taux sont calculés au tableau I.

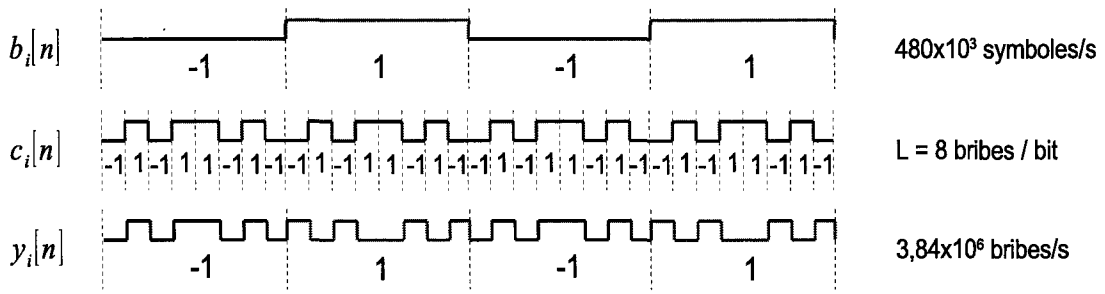


Figure 2 Étalement d'un message binaire par une séquence PN

Tableau I

Taux de symboles pour les facteurs d'étalement permis

Facteur d'étalement L	Taux de symbole (s^{-1})	Période (10^{-6} s)
4	960 000	1.04
8	480 000	2.08
16	240 000	4.17
32	120 000	8.33
64	60 000	16.67
128	30 000	33.33
256	15 000	66.67

D'un point de vue fréquentiel, la bande passante du message binaire $b_i[n]$ s'accroît d'un facteur L jusqu'à la bande passante du canal. L'utilisation d'un filtre de mise en forme à cosinus surélevé de facteur α (0,22 pour WCDMA) augmente la largeur BW (« bandwidth ») du spectre à

$$BW = (1+\alpha)3{,}84 \times 10^6 = 4{,}68\,MHz. \qquad (1.4)$$

Le standard alloue en réalité une plage de 5MHz pour la transmission, incluant des bandes de garde de part et d'autre du spectre pour empêcher l'interférence inter-canal (« inter-channel interference » ou ICI). Cette largeur de bande confère à cette modulation le nom de CDMA à large bande (« wideband CDMA » ou WCDMA), en opposition à la bande de 1,25 MHz utilisée par le standard nord-américain IS-95 (TIA, 1995).

Le processus d'étalement est illustré du point de vue fréquentiel à la figure 3. La première illustration (a) montre le contenu fréquentiel du message d'origine, (b) montre l'effet de l'étalement spectral, (c) simule la dégradation de la transmission par bruit additif, et (d) montre les effets bénins du bruit suite au processus de réception.

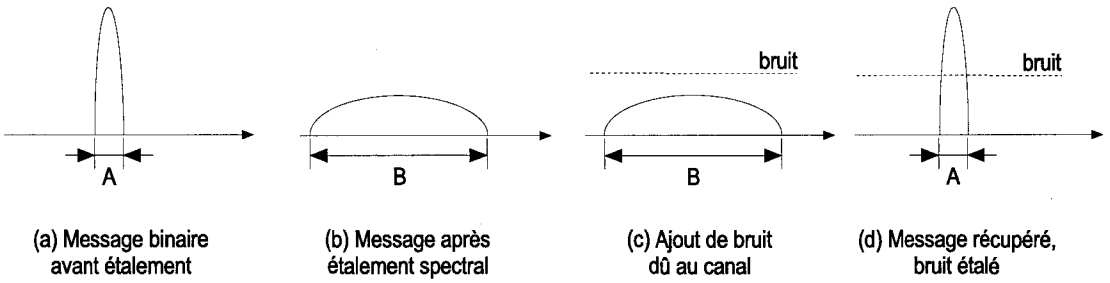


Figure 3 Largeur de bande du signal WCDMA à différentes étapes

1.2.2 Caractéristiques du canal de transmission

Contrairement aux communications filaires où le canal de transmission est fixe, la transmission hertzienne offre un canal aux caractéristiques évasives. Les paramètres du canal dépendent de la propagation dans l'environnement, de la répartition des usagers, des

fréquences et de la largeur de bande en jeu, du type de modulation employé, et des interactions incertaines entre tous ces facteurs. Généralement, des modèles statistiques sont utilisés pour représenter le canal lors du développement d'un système ou d'une norme de transmission. Parfois même, des campagnes de mesures permettent de valider et standardiser ces modèles, par exemple l'initiative européenne COST-207 (Molisch, 2001). Ces modèles confirment tous qu'aux fréquences et bandes passantes utilisées, le canal comporte un nombre important de multi-trajets dont le profil de puissance et la répartition temporelle varient selon plusieurs facteurs.

Plusieurs caractéristiques et paramètres régissent le comportement du canal. Ceux dont l'impact sur la modulation WCDMA est majeur sont énumérés ci-après.

1.2.2.1 Multi-trajets

Les multi-trajets (en anglais, « multipath ») sont une conséquence de la géométrie du canal de transmission. Selon la disposition des obstacles naturels (feuillage, collines, plans d'eau) et d'origine humaine (édifices), le front d'onde émis par un transmetteur atteint le récepteur en plusieurs répliques aux amplitudes, phases et délais différents. Leur sommation est alors constructive ou destructive. La figure 4 montre l'effet d'un canal multi-trajets sur l'enveloppe temporelle d'un signal.

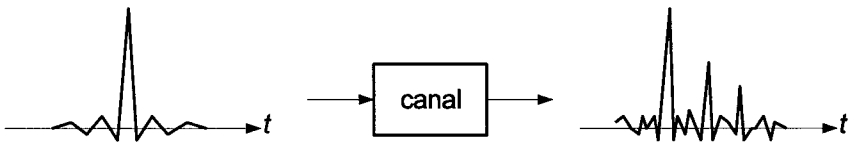


Figure 4 Effet du canal multi-trajets sur l'enveloppe temporelle

Le nombre détectable de ces répliques dépend des paramètres de transmission, notamment de la bande passante et du taux d'étalement employé. Il peut y avoir présence d'un trajet prédominant dans le cas d'une communication en ligne de mire (« line of sight » ou LOS) rendant la détection plus facile; c'est le cas du modèle de Rice. Lorsque la ligne de mire

est inexistante et chaque trajet est indépendant et quelconque (« non line of sight » ou NLOS); c'est alors le modèle de Rayleigh qui décrit mieux le canal. Ce dernier ajoute en plus un étalement de spectre dû à l'effet Doppler, mais non-corrélé entre chaque trajet. Le canal de Rice est donc généralement utilisé pour modéliser une transmission de banlieue, alors que celui de Rayleigh simule mieux un environnement urbain.

1.2.2.2 Évanouissement du canal

Les évanouissements du canal (« fading » en anglais) peuvent avoir plusieurs causes dont la principale est reliée à l'arrivée périodiquement destructive des multi-trajets. Ils affectent essentiellement la puissance perçue au récepteur de plusieurs décibels sur des bandes de fréquence variables. Les évanouissements peuvent être rapides ou lents par rapport à la vitesse d'adaptation du récepteur (ce dernier cause alors une erreur permanente (« bias ») dans son estimation). La force d'un évanouissement se traduit par des aspérités (« ditches ») plus ou moins profondes dans le spectre, tel qu'illustré à la figure 5.

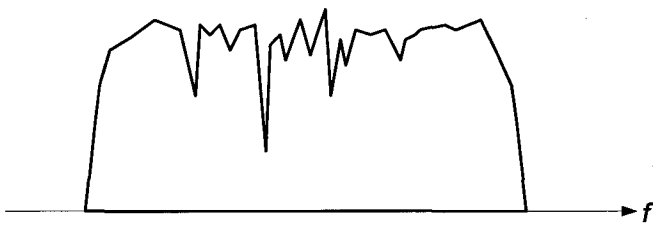


Figure 5 Aspérités du spectre dues à l'évanouissement

1.2.2.3 Dispersion temporelle (« delay spread »)

La dispersion temporelle mesure le délai entre l'arrivée du premier et du dernier trajet radio au récepteur. Elle dépend de la configuration géométrique instantanée du canal et sa borne supérieure peut atteindre quelques microsecondes en milieu urbain, voire même près de 20 µs en milieu campagnard selon Molisch (2001).

1.2.2.4 Bande de cohérence

La bande de cohérence (« coherence bandwidth ») donne une indication sur l'étendue fréquentielle d'un évanouissement (voir figure 5). Lorsque la bande passante de la transmission BW_M est inférieure à la bande passante cohérente BW_C, le canal inflige un évanouissement uniforme sur BW_C (« flat fading » en anglais). Si BW_M est supérieure à BW_C, on dit plutôt qu'il y a évanouissement sélectif en fréquence (« frequency selective fading ») et le spectre transmis est affecté de manière irrégulière. Cette situation permet cependant d'exploiter la diversité de fréquence résultante pour améliorer la communication. C'est la situation qu'exploite WCDMA. La bande passante cohérente est inversement proportionnelle à l'étendue temporelle.

1.2.2.5 Dispersion Doppler

La dispersion Doppler (« Doppler spread ») mesure la largeur du spectre de la porteuse non modulée. On la modélise de différentes manières dont les plus courantes sont le spectre plat et le spectre de Jake (en forme de U) qui sont issus de la probabilité statistique qu'a la porteuse d'occuper une fréquence en particulier de cette étendue (probabilité uniforme dans le premier cas, en U dans le second). Ce paramètre compte parmi les effets dispersifs du canal puisqu'il affecte la forme du signal reçu. Concrètement, il entraîne un déplacement fréquentiel du spectre CDMA, causant une variation de la durée T_C d'une bribe de façon différente d'un instant au suivant, augmentant la probabilité de subir des erreurs de synchronisation.

1.2.2.6 Temps de cohérence du canal

Le temps de cohérence du canal renseigne sur la durée pendant laquelle les paramètres du canal demeurent essentiellement stables. Le temps de cohérence est donc réciproque à l'étendue Doppler du spectre (Ojanperä, 1998). Primordial pour les communications à multiplexage temporel où des séquences d'entraînement (« training sequences ») estiment

périodiquement le canal, ce paramètre peut être exploité pour abaisser le taux d'estimation des récepteurs autodidactes, ou de manière équivalente, abaisser le nombre de symboles pilotes pour les autres systèmes.

1.2.2.7 Éblouissement

Particulièrement néfaste aux systèmes à étalement spectral, l'éblouissement (« near-far effect ») survient lorsque la transmission d'un usager, telle que perçue par la station de base, occulte les autres usagers. Cette situation survient lorsqu'un usager à proximité du récepteur transmet à une puissance trop élevée. Le contrôle de puissance en boucle fermée entre station de base et usagers permet d'amoindrir cet effet néfaste de l'accès multiple (en plus de combattre l'évanouissement tel que mentionné à la section 1.2.2.2).

1.2.2.8 Accès multiple

Les interférences dues à l'accès multiple (« multiple access interference » ou MAI), proviennent de la compétition entre les utilisateurs pour utiliser la même plage spectrale au même instant. Pour la transmission CDMA, les sources principales sont l'accès asynchrone, l'éblouissement et les codes d'étalement non orthogonaux. Chaque utilisateur du système devient ainsi une source d'interférence corrélée pour tous les autres. Certains récepteurs parviennent à réduire l'effet du MAI à l'aide de techniques de combinaison adaptées; d'autres le traitent à tort comme un simple bruit additif.

1.2.2.9 Interférence inter-symbole

Ce type de dégradation (« inter-symbol interference » ou ISI) découle de l'arrivée imparfaite et décalée des multi-trajets au récepteur. Lorsque l'étalement temporel du canal dépasse la période d'un symbole, il y a débordement d'un symbole sur le suivant. Ce type d'interférence affecte davantage les transmissions à faible étalement où la bande passante du message approche celle de la transmission.

1.2.2.10 Interférence inter-canal

Toutes les classes de modulation sont affectées par la trop grande proximité de deux bandes fréquentielles destinées à des canaux différents, d'où l'interférence inter-canal (« inter-channel interference » ou ICI). Des bandes de garde situées de part et d'autre du spectre utile sont généralement spécifiées pour minimiser ce type d'interférence.

1.2.3 Récepteur RAKE

Pour combattre les dégradations dues à la nature dispersive du canal multi-trajets d'une transmission DS-CDMA, Price et Green (1958) ont suggéré le récepteur RAKE qui tire simplement son nom de la forme du schéma qui rappelle celle d'un râteau. Son schéma de base est donné à la figure 6.

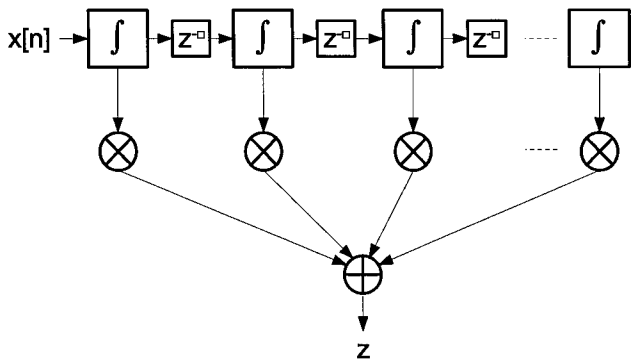


Figure 6 Récepteur RAKE

Le récepteur RAKE est un combinateur qui effectue la sommation pondérée de n filtres adaptés (« rake fingers » ou *doigts* de râteau) selon des gains ajustables. Ces filtres adaptés sont en réalité des corrélateurs indépendants qui recouvrent la séquence étalée reçue $x[n]$ à différents délais $d\tau_D$, eux aussi ajustables et dont la précision dépend du taux de suréchantillonnage utilisé en entrée. Le processus exploite simplement la diversité temporelle du canal pour combattre l'effet des multi-trajets. La valeur de corrélation recouvrée des n trajets perçus est améliorée par rapport au cas dégénéré $n = 1$, aussi appelé le « single arm », ou SA (Molisch, 2001).

Le choix du nombre de doigts fait souvent appel à une connaissance empirique des conditions de canal. Toujours selon Molisch, un nombre trop faible de doigts empêche le recouvrement du maximum de puissance, alors qu'un nombre de doigts supérieur au nombre de trajets résolubles N_r introduit du bruit dans le processus de décision. L'ajustement du paramètre τ_D pour chaque doigt s'effectue à l'aide de corrélations avancée et retardée d'un cran (« early-late gate ») à chaque symbole, pour chaque doigt.

1.2.3.1 Accès multi-usagers

Un récepteur RAKE est limité par le niveau d'interférence (« interference-limited »). En pratique, cela signifie que chaque nouvel usager dans le système augmente le niveau d'interférence pour ses voisins et dégrade la capacité globale du système. Pour pallier cet effet, des détecteurs multi-usagers (« multi-user detector » ou MUD) sont utilisés. Le MUD optimal, cependant, a une complexité trop élevée puisqu'elle croît de façon exponentielle avec le nombre d'usagers du système (Verdú, 1986). La recherche actuelle se penche donc sur des solutions sous-optimales à complexité réduite pour effectuer la détection multi-usagers.

Deux classes de MUD sous-optimaux sont ainsi introduites: l'annulation d'interférence (« interference cancellation »), dont les détecteurs parallèles (« parallel interference cancellation » ou PIC) et séquentiels groupés (« groupwise sequential interference cancellation » ou GSIC), ainsi que la détection conjointe, dont font partie les égaliseurs ZF-BLE (« zero-forcing block linear equalizer »), MMSE-BLE (« minimum mean-square error block linear equalizer ») et leur version à rétroaction ZF-DFBE (« zero-forcing decision feedback equalizer ») et MMSE-DFBE (« minimum mean-square error decision feedback equalizer). Prasad (1996) propose un survol de ces quatre catégories.

La détection conjointe, certes moins extensible au point de vue système, permet de réduire les coûts de complexité puisque les K utilisateurs sont captés conjointement par un seul et

unique détecteur qui connaît l'interaction de la MAI et de l'ISI de tous et chacun. Il s'agit donc d'une solution plausible pour un récepteur de station de base.

1.2.3.2 Performances

Le critère de performance universel pour départager les récepteurs est la probabilité d'erreur binaire (« bit error probability » ou BEP), ou de manière équivalente, le taux d'erreur binaire (« bit error rate » ou BER). À des fins de comparaisons subséquentes dans ce document, les courbes de BER de plusieurs algorithmes de détection conjointe, sans considération pour le codage, sont présentées à la figure 7 pour une et deux antennes (Prasad, 1996, fig. 11.36).

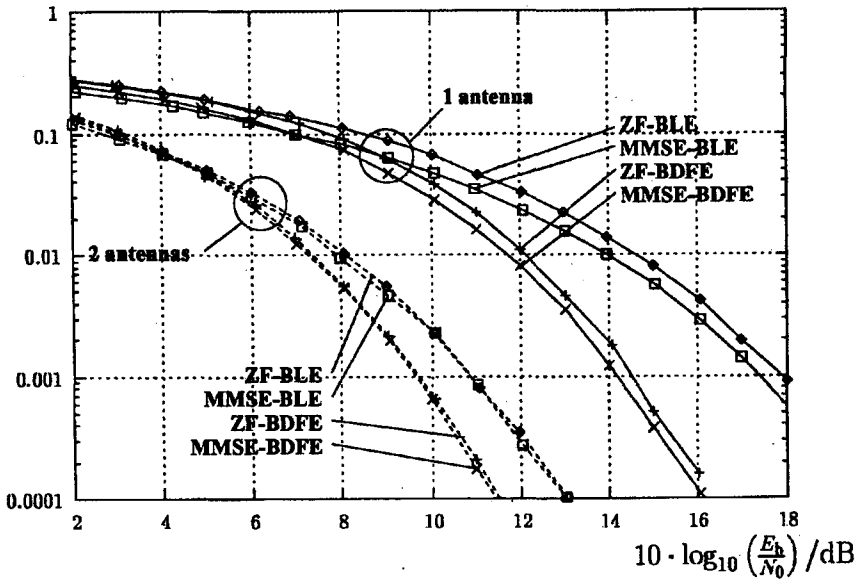


Figure 7 Performances de réception par détection conjointe (Prasad, 1996)

Les comparaisons entre les autres types de détecteurs ainsi que l'impact du nombre d'usagers sur leurs performances sont omis puisqu'ils dépassent le cadre de ce mémoire.

Les performances d'un récepteur sont aussi affectées par les décisions prises lors de sa réalisation matérielle. Ainsi, les courbes illustrées à la figure 7 témoignent de résultats

théoriques non affectés par les imperfections du matériel. Ces imperfections pourraient provenir des choix de topologies matérielles, ou encore de la quantification de l'algorithme. Ces deux critères dépendent eux-mêmes de la plate-forme matérielle exploitée par le système. La section suivante résume quelques solutions disponibles pour accueillir ce type de système.

1.3 Architectures matérielles

L'architecture de systèmes microélectroniques revêt parfois l'apparence d'une science inexacte. La réalisation d'un système électronique et/ou informatique n'est jamais qu'une somme de compromis et d'hypothèses simplificatrices. Plusieurs facteurs, tant techniques, scientifiques que marketing, servent d'arguments lors de la sélection d'une architecture matérielle. Cette section explore donc certaines alternatives qui s'offrent au concepteur et souligne les avantages et inconvénients de chacune.

1.3.1 Microprocesseur (CPU)

La puissance de calcul actuelle des systèmes à microprocesseur permet la résolution de problèmes arithmétiques de grande complexité à coût modique. Les classes de processeurs Pentium et Athlon à 64 bits cadencées à 3 GHz offrent, en théorie, une puissance de calcul chiffrée en giga-opérations par secondes, et une bande passante de plusieurs gigabits/s avec leur mémoire.

Puisque le microprocesseur est un outil générique, plusieurs instructions « inutiles » doivent accompagner chaque instruction « utile » à un calcul spécifique. Parmi les instructions *inutiles*, on compte les accès à la mémoire, la mise à jour de compteurs, les tâches du système d'exploitation, la gestion de protocoles de bus, etc. Les instructions utiles sont généralement de nature arithmétique, et directement reliées à un problème à résoudre: additions, multiplications, divisions, etc.

De plus, la répartition d'un système à microprocesseurs en plusieurs microcircuits distincts entraîne une latence dans le système puisque la communication entre les différents organes de calcul doit être conforme à certains protocoles ou à des médiums de transmission contraignants. Par exemple, le débit entre deux composants d'un circuit imprimé (« printed circuit board » ou PCB), est limité par la bande passante des traces de cuivre ainsi que leur nombre et longueur. Même pour un circuit de haute qualité, le débit final utile n'approche en rien celui de l'unité arithmétique d'un microprocesseur.

Tous ces facteurs font du microprocesseur un choix évident pour les applications génériques et le travail de simulation, mais entraînent souvent son rejet pour les tâches à forte complexité arithmétique asservies à des éléments en temps réel.

1.3.2 Processeur de traitement du signal (DSP)

Pour investir le marché du traitement numérique du signal en temps réel, certains manufacturiers, dont Analog Devices et Texas Instruments, ont développé une classe de processeurs spécialisés (« digital signal processors » ou DSP) qui offrent des antémémoires mieux adaptées aux instructions vectorielles souvent utiles au traitement du signal et que l'on retrouve sur ce type de processeur. Bien que les DSP soient soumis aux mêmes contraintes physiques que les systèmes à microprocesseurs (circuit imprimé, protocole de communications, etc.), les DSP sont généralement dotés d'un logiciel dédié ou d'un système d'exploitation minimal libéré de tâches de gestion, minimisant ainsi le nombre d'opération inutiles.

Le parallélisme est omniprésent dans ce type de processeur; on compte généralement plusieurs unités arithmétiques pipelinées, câblées en parallèle, et capables d'exécuter une même instruction sur un lot de données (« single instruction, multiple data » ou SIMD), voire même un lot d'instructions sur un lot de données (« multiple instructions, multiple data » ou MIMD).

Les DSP bénéficient d'une faible consommation énergétique, d'un coût inférieur, d'outils de développement logiciel matures et d'une architecture parallèle, mais dépendent toujours d'une structure sur circuit intégré et d'un jeu d'instructions fixé par le manufacturier.

1.3.3 Logique programmable (FPGA)

La hausse de performances, au cours des années 1990, des dispositifs électroniques programmables de logique universelle (« field-programmable logic array » ou FPGA), a permis un accroissement de complexité des systèmes logiques et embarqués qui sont disponibles aujourd'hui. Permettant dans une certaine mesure de regrouper sur le même silicium une quantité importante d'opérations logiques qui nécessitaient autrefois une grande quantité de puces distinctes disposées sur un circuit imprimé, le FPGA est aujourd'hui devenu un incontournable.

Constitués d'une maille d'éléments logiques programmables simples, à laquelle se rattachent des ressources classiques tels que des processeurs embarqués, des mémoires dédiées, des multiplicateurs rapides et des interfaces d'entrée/sortie spécialisées, les FPGAs se positionnent sur le marché comme une solution de choix pour le prototypage de systèmes de traitement numérique du signal. Leur flexibilité entraîne cependant un coût unitaire plus élevé que celui d'un microprocesseur ou DSP contemporain.

1.3.4 « System on Chip »

La tendance actuelle vise l'intégration complète des fonctions logiques (mémoire, processeur, bus) et des portions analogiques (interfaces de communications, convertisseurs, amplificateurs) d'un système microélectronique sur un seul et même substrat de silicium. Ce rassemblement est rendu nécessaire par l'augmentation de complexité des systèmes et l'augmentation de bande passante qui s'ensuit. Les progrès des techniques de fabrication microélectroniques permettent désormais d'effectuer une telle intégration tout en atteignant des taux de réussite (« yield ») acceptables. L'intégration des

interfaces optiques et radiofréquence, lesquelles nécessitent d'autres types de semiconducteurs comme le Ga_xAs_{1-x} ou $AlGa_xAs_{1-x}$, des semiconducteurs dont la maille cristalline est incompatible avec celle du silicium, demeure toutefois un obstacle majeur. À défaut d'utiliser des composants distincts pour réaliser ces fonction comme c'est toujours le cas aujourd'hui, de nouvelles techniques d'épitaxie, comme celle proposée par Pei (2002), laissent présager une solution sous peu à ce problème, ouvrant la voie à une intégration complète des systèmes de télécommunications.

Accompagnée d'interfaces programmables et d'une librairie de modules paramétrables, la technique du SoC permet d'augmenter de manière considérable la bande passante entre les composantes logiques d'un système. Par exemple, un bus de mémoire de 1024 bits, difficilement réalisable sur circuit imprimé, serait envisageable sur SoC.

Ce type de système facilite donc le parallélisme massif. Il ouvre d'ailleurs de nouvelles avenues de recherche concernant la distribution et l'échange de données entre les différents modules d'un SoC. Puisque les fréquences d'opération en jeu augmentent, la distance potentielle entre les modules s'accroît et les conducteurs les reliant se comportent davantage comme des lignes de transmission, l'hypothèse de transmission immédiate des signaux logiques n'est plus valide dans le paradigme SoC. On entrevoit donc la réalisation, sur une puce, de réseaux de communications de différentes topologies. On parle alors de réseau sur puce (« network on chip » ou NoC) (Jantsch, 2004).

1.3.5 Codesign

Le codesign matériel/logiciel (« hardware/software codesign ») est une technique de réalisation de systèmes embarqués qui suscite l'intérêt des concepteurs de matériel et de logiciel depuis les années 1990. Cette technique de conception permet de planifier la séparation des tâches arithmétiques et logiques d'un système entre des composantes matérielles et logicielles, dans le but de satisfaire une spécification fonctionnelle donnée et de répondre à des critères de conception comme le fonctionnement en temps réel

(Wolf, 1994). La conception codesign d'un système doit cependant être scrupuleusement évaluée en tout premier lieu puisqu'elle peut n'engendrer, dans certains cas, que de faibles gains de performances, accompagnés d'un coût pourtant élevé (Pospiech, 2003).

Edwards (1996) définit trois types de conceptions codesign: biaisée vers le logiciel, biaisée vers le matériel, et non-biaisée. Le premier type assume un système constitué d'algorithmes logiciels qui doit subir une accélération ponctuelle afin de répondre à une spécification fonctionnelle, souvent reliée à la nature temps-réel de l'application cible. La conception biaisée vers le matériel concerne plutôt la migration, vers le logiciel, de portions matérielles d'un système pour lequel un degré de liberté supplémentaire est requis (système adaptatif), ou qui comporte des opérations de contrôle plus avantageusement réalisées en instructions de microprocesseurs qu'en portes logiques (Gupta, 1993). La dernière catégorie, non-biaisée, présume une conception à un niveau d'abstraction suffisamment élevé pour qu'une description fonctionnelle des différentes composantes du système soit écrite. Cette description est par la suite analysée manuellement ou automatiquement afin de déterminer une dichotomie optimale en termes de bande passante et d'utilisation des ressources de part et d'autre. Pospiech (2003) rappelle que peu importe la séparation choisie, la plus grande partie de l'effort est essentiellement investie dans la réalisation d'interfaces optimales entre le matériel et le logiciel.

Le partage des parties à proéminence logicielle ou matérielle d'un algorithme répond à quelques règles approximatives. Selon Edwards (1996), les boucles arithmétiques imbriquées de deux à cinq niveaux sont généralement candidates à une réalisation matérielle. Au même titre, les opérations utilisant des ensembles de données de 10 à 10^5 éléments peuvent trouver avantage à être gérées par une logique dédiée.

Un dernier aspect du codesign matériel/logiciel est la réutilisation des objets créés. Des processeurs modulaires, suffisamment paramétrables et dotés d'interfaces communes peuvent être consignés dans une librairie d'objets génériques qui permet, lors de designs

subséquents, de diminuer le temps de conception d'un système. Fonctionnalité éprouvée et comportement testé permettent en outre de réduire le temps nécessaire au déverminage d'un tel système (Pospiech, 2003).

Un corollaire prometteur de la modularité inhérente à un système codesign est la possibilité d'en modifier de manière dynamique certaines portions (« dynamically reconfigurable logic » ou DRL), permettant l'utilisation potentielle de plusieurs gros processeurs dédiés dans un seul et unique système qui, en temps normal, ne pourrait tous les contenir, faute de ressources (Noguera, 2002).

1.3.6 Parallélisme

L'accélération d'un algorithme peut aussi s'effectuer par la résolution parallèle de plusieurs segments d'un même problème. Vai (2001) indique que tout problème n'est pas parallélisable, et ceux qui le sont ne le sont généralement pas entièrement. On parle alors de fraction parallélisable du problème. Hormis l'épreuve de reformuler le problème de manière plus parallèle, d'autres critères doivent être considérés avant de considérer un tel fonctionnement. En premier lieu, la présence d'interdépendances entre les différentes sections d'un problème peut causer l'ajout de délais dûs à un processeur parallèle en attente des résultats d'un voisin. En second lieu, les communications entre la ou les mémoire(s) et les processeurs parallèles augmentent la complexité de l'appropriation et le stockage des données. Enfin, le plus faible déséquilibre dans les sémaphores d'accès entre les processeurs en compétition pour une mémoire centrale unique peut congestionner le système au point de subir une dégradation des performances globales en comparaison au système séquentiel original.

Certains thèmes propres à l'architecture de systèmes parallèles sont énumérés ci-après afin de souligner l'importance d'une bonne planification *a priori* d'un tel système.

1.3.6.1 Modèles de mémoire

Le branchement entre les unités de mémoire et les processeurs parallèles joue un rôle clé dans les performances d'un système parallèle. La figure 8 montre trois répartitions possibles de la mémoire selon Hennesy (1996): (a) mémoire centrale partagée, (b) mémoire distribuée privée et (c) mémoire distribuée partagée.

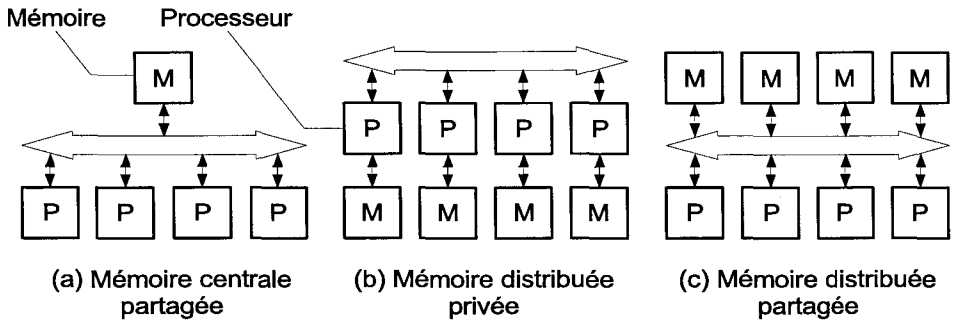


Figure 8 Répartition de la mémoire dans un système parallèle

Le premier modèle concentre la totalité des données dans une mémoire unifiée, accessible équitablement par tous les processeurs. Ce modèle garantit la disponibilité totale et immédiate de toutes les données d'un problème, mais il est prône à la congestion lors de l'accès à la mémoire. Le second modèle, où chaque processeur dispose de sa propre zone mémoire, permet d'augmenter la bande passante entre processeur et mémoire, mais entraîne la segmentation des données du problème. Un protocole d'étalement et de récupération des données (« scatter gather ») doit être mis en place pour distribuer des sections du problème provenant d'un niveau hiérarchique supérieur, mais rajoute un premier délai pour la répartition et un second pour récupérer les résultats. De plus, si un processeur devait accéder à une donnée stockée dans la mémoire privée d'un voisin, l'opération hiérarchique de « scatter gather » devrait recommencer et entraînerait des délais supplémentaires.

Le troisième modèle amoindrit ce problème en donnant à chaque processeur un accès dédié à une mémoire distribuée, sans toutefois lui en accorder l'exclusivité. Le schéma

d'adressage pour ce modèle requiert un amalgame de plages d'adresses et l'introduction de sémaphores de partage de la mémoire suivant les requêtes distantes.

1.3.6.2 Granularité des opérations

Afin de minimiser la surcharge des communications inter-processeurs, le problème d'origine doit être segmenté en sections de tailles appropriées. On qualifie de fine (« fine grain ») la granularité du problème lorsque ce dernier est segmenté en petites unités rapidement solutionnées et nécessitant peu d'interactions avec la mémoire. Ce type de séparation entraîne donc une plus grande surcharge de communication dans un court laps de temps. À l'opposé, la granularité est grossière (« coarse grain ») lorsque le problème est segmenté en unités plus grandes, solutionnées en un temps supérieur et requérant de longs, mais moins fréquents, accès à la mémoire. Grama (2003) note que « la granularité devrait être choisie de sorte que le coût d'effectuer le travail dépasse celui de le transférer. »

En général, la granularité d'un système parallèle reflète la longueur des rétroactions présentes dans l'algorithme puisqu'il est préférable d'encapsuler dans une unité logique tous les accès mémoire requis pour solutionner cette parcelle du problème initial.

1.3.6.3 Pipeline

L'alignement des opérations en pipeline est considéré comme un type de parallélisme temporel (Vai, 2001). Le pipelinage consiste à séparer en étapes séquentielles un calcul arithmétique non parallélisable au sens propre. Sectionner le problème de manière à ce que les étapes successives de résolution (les étages voisins du pipeline) soient indépendantes permet de faire fonctionner en même temps tous les étages du pipeline. Un pipeline à cinq étages peut, selon ce schéma, résoudre cinq instances d'un problème en même temps. Cette organisation entraîne l'introduction d'un délai proportionnel au nombre

d'étages, mais fournit globalement un résultat par cycle de pipeline, soit cinq fois plus rapidement qu'une architecture non-pipelinée équivalente.

Puisque chaque étage du pipeline remplit vraisemblablement une tâche différente, le temps de traitement diffère d'un étage au suivant. Cette situation mène invariablement à une utilisation sous-optimale des ressources. La solution triviale consiste à imposer à tous les étages le rythme du plus lent (« static scheduling »). D'autres solutions permettent à un étage libre d'accepter les données du précédent sitôt prêtes (« greedy scheduling »). Le problème de sous-utilisation du pipeline se réduit alors à un problème de gestion de file. Il peut être solutionné par la duplication ou la réutilisation des étages rapides (Vai, 2001). Quelle que soit la topologie de pipeline utilisée, la planification de la séquence de traitement est primordiale pour obtenir un système optimal.

1.4 Conclusion

Ce chapitre a donné un aperçu de l'état actuel de la recherche dans les domaines que touche ce projet. La modulation à étalement spectral a d'abord été définie et comparée aux autres grandes familles de modulation. Les effets néfastes du canal radio ainsi que les paramètres d'intérêts en regard à la transmission CDMA ont été présentés. Une brève revue du récepteur RAKE ainsi que des techniques de détection multi-usagers a été effectuée, accompagnée par une évaluation de leurs performances en termes de taux d'erreur binaire non-codée. Ces performances permettront une comparaison avec la solution présentée dans les chapitres subséquents.

La seconde partie a exploré les avenues disponibles pour la réalisation d'un algorithme de traitement numérique du signal, et des différents scénarios de sectionnement d'un problème arithmétique. Les techniques de conception codesign et la planification d'architectures parallèles ont été démontrées et seront mises à profit lors de la réalisation du récepteur STAR dont il est question au chapitre suivant.

CHAPITRE 2

LE RÉCEPTEUR STAR

2.1 Introduction

Après le survol au chapitre précédent des récepteurs actuels dans les communications à étalement spectral, la solution RAKE peut sembler la seule viable. S'il est vrai qu'il fait toujours l'objet de développements, ce récepteur est néanmoins empreint de limitations inhérentes face à certains types d'interférence. De plus, la structure répétitive en fonction du nombre de « doigts » est un facteur limitatif au chapitre des performances et de l'utilisation efficace du silicium. Le récepteur STAR (Affes, 1997) (Affes, 1998) (Cheikhrouhou, 2001) offre plutôt un ensemble de ressources de calcul qui est assigné dynamiquement entre plusieurs usagers.

Ce chapitre passe en revue les fondements mathématiques du récepteur STAR. Suivant la présentation du modèle post-corrélation sur lequel il fonde son analyse du canal, toutes les équations de l'algorithme sont examinées et mises en relation avec son paradigme de réception. Le survol théorique se conclut par la présentation du schéma-bloc global du récepteur.

En second lieu, certains avantages du récepteur STAR mis en relief par ce projet sont présentés en plus amples détails. En particulier, la précision de l'identification des délais sans recours au suréchantillonnage et la nature adaptative de l'algorithme font l'objet d'une explication plus poussée.

2.2 Paradigme de réception de STAR

La figure 9 présente un schéma fonctionnel du paradigme de réception de STAR. On le représente en trois modules: réception des symboles (« Symbol Path » ou SP),

régularisation des observations du canal (« Structure Fitting » ou STRF), et gestionnaire de multi-trajets (« Path Management » ou PM). Le premier module effectue le décodage proprement dit des symboles captés. Il utilise la matrice $\hat{\boldsymbol{H}}$ comme entrée au combinateur de symboles, laquelle lui est fournie périodiquement par un module de régularisation. C'est ce dernier qui renferme l'algorithme de régularisation qui permet d'obtenir, à partir d'une observation $\tilde{\boldsymbol{H}}$, un estimé de canal plus précis. Enfin, un dernier module de gestion est chargé des décisions quant au profil multi-trajets servant à contraindre $\hat{\boldsymbol{H}}$.

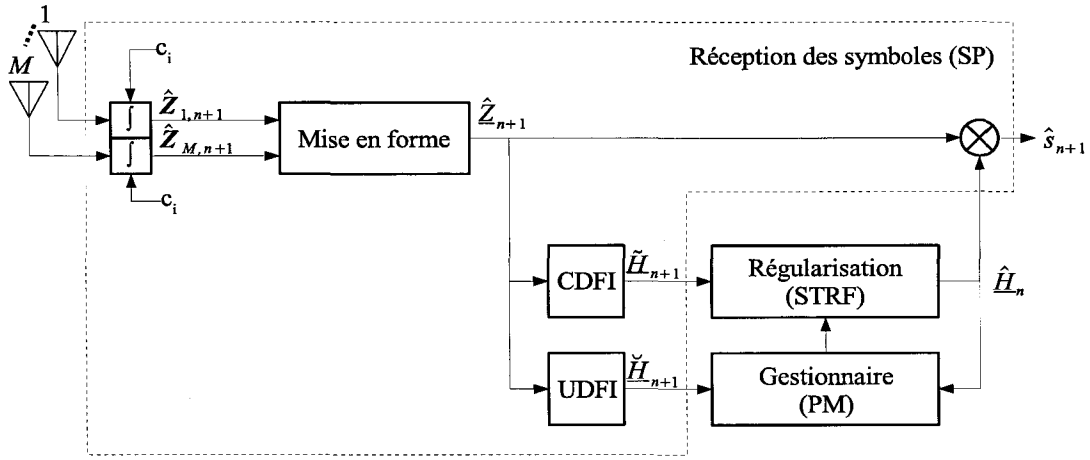


Figure 9 Organigramme général du récepteur STAR

Le combinateur du module SP s'alimente des symboles récupérés (corrélés) se trouvant dans le modèle post-corrélation (PCM), une seconde particularité du récepteur STAR.

2.2.1 Modèle post-corrélation (PCM)

Posant $b(t)$ comme message binaire transmis de période T_S, $c(t)$ comme son code d'étalement de période T_C (et de longueur $LT_C = T_S$), on définit le signal reçu comme

$$X(t) = \psi(t) \sum_{p=1}^{P} G_p(t) \varepsilon_p(t) b(t - \tau_p(t)) c(t - \tau_p(t)) + I(t), \qquad (2.1)$$

où P est le nombre de multi-trajets perçus par le récepteur, $I(t)$ est un terme représentant toutes les sources d'interférences additives du système, $\tau_p \in [0, T_s]$ sont les délais associés aux multi-trajets, $\psi^2(t)$ est la puissance totale reçue et ε_p^2 est la fraction de puissance associée au trajet p de sorte que

$$\sum_{p=1}^{P} \varepsilon_p^2(t) = 1, \tag{2.2}$$

et $G_p(t)$ est la matrice complexe des coefficients du canal. STAR estime indirectement $G_p(t)$ à l'aide d'observations continues sur le sous-espace de transmission défini par la séquence d'étalement c_i. Cet estimateur, $\hat{\boldsymbol{H}}$, est contraint par un processus de régularisation qui s'appuie sur l'observation périodique du canal effectuée par les modules CDFI et UDFI (section 2.2.3).

La première tâche du récepteur consiste à effectuer une corrélation entre le signal reçu et le code d'étalement adéquat. On la définit dans le domaine analogique comme:

$$Z(t') = \frac{1}{T_S} \int_0^{T_s} X(t+t') c^*(t)\, dt. \tag{2.3}$$

Puisqu'elle est en fait engendrée par une séquence de bribes, $Z(t')$ est discrétisée en échantillons distants de $T_C = 260$ ns, ce qui crée le vecteur $\boldsymbol{Z}[n]$. Suivant le facteur d'étalement, on regroupe les éléments de $\boldsymbol{Z}[n]$ en trames de L échantillons pour obtenir:

$$\boldsymbol{Z}_n = \left[Z_n(0) \quad Z_n(T_c) \quad \cdots \quad Z_n\left((L-1)T_c\right) \right]^T. \tag{2.4}$$

Pour traiter le cas d'un récepteur à plusieurs antennes, on introduit le paramètre M qui représente le nombre d'antennes au récepteur. Affes (1997) définit le modèle post-corrélation (« post-correlation model » ou PCM) sous la forme d'une matrice de

dimensions L par M. Chaque colonne provient d'une antenne différente et fait l'objet d'un traitement distinct jusqu'à ce point-ci.

Pour démontrer le contenu spatio-temporel de la matrice post-corrélation, on écrit

$$\begin{aligned} \boldsymbol{Z}_n &= b_n \psi_n \sum_{p=1}^{P} G_{p,n} \varepsilon_{p,n} \boldsymbol{D}_{p,n}^T + \boldsymbol{N}_n \\ &= b_n \psi_n \boldsymbol{G}_n \boldsymbol{Y}_n \boldsymbol{D}_n^T + \boldsymbol{N}_n \\ &= b_n \psi_n \boldsymbol{J}_n \boldsymbol{D}_n^T + \boldsymbol{N}_n, \end{aligned} \tag{2.5}$$

où N_n est le penchant de $I(t)$ dans un plan spatio-temporel. $G_{p,n}$ représente la pondération en énergie et en phase des P trajets sur les M antennes. $\boldsymbol{D}_{p,n}$, pour sa part, est un support temporel constitué de P colonnes dans lesquelles on observe la réponse impulsionnelle retardée de τ_p. La composante de bruit $\boldsymbol{N}_n$ illustre ici l'apport de l'interférence inter-symbole (« inter-symbol interference » ou ISI), de la dégradation due aux autres usagers du système, et du bruit thermique (Cheikhrouhou, 2001). Dans cette réalisation minimale du récepteur STAR, on considère ce bruit comme additif et non-corrélé, aussi bien dans le temps que dans l'espace.

De l'équation 2.5, on peut considérer le terme $\boldsymbol{J}_n \boldsymbol{D}_n^T$ comme la matrice spatio-temporelle des coefficients du canal de transmission. L'équation 2.5 se récrit alors

$$\boldsymbol{Z}_n = b_n \psi_n \boldsymbol{H}_n + \boldsymbol{N}_n, \tag{2.6}$$

où $\boldsymbol{Z}_n$ est le vecteur d'observation post-corrélation et $\boldsymbol{H}_n$ est la matrice des coefficients du canal de transmission vectoriel. Puisque le vecteur d'observation est une mesure prise par le récepteur et que $\hat{s}_n$ est l'élément à trouver, le travail consiste à fournir une estimation de $\boldsymbol{H}_n$, posons $\hat{\boldsymbol{H}}_n$, qui représente le plus fidèlement possible les conditions du canal.

Afin d'alléger les manipulations mathématiques et pour se conformer à la notation de (Affes, 1997) (Affes, 1998) (Cheikhrouhou, 2001), on transforme $\mathbf{Z}_n$, de taille $M \times L$, en vecteur colonne $\underline{Z}_n$ de taille $LM \times 1$ qui représente essentiellement la concaténation verticale des éléments de $\mathbf{Z}_n$ provenant des M antennes:

$$\underline{Z}_n = \left[\mathbf{Z}_{n,1}^T \;\; \mathbf{Z}_{n,2}^T \;\; \cdots \;\; \mathbf{Z}_{n,M}^T \right]^T . \tag{2.7}$$

Le même stratagème s'applique à $\boldsymbol{H}_n$, de taille $M \times L$, qui devient $\underline{H}_n$ de taille $ML \times 1$.

En supposant la connaissance parfaite de $\underline{H}_n$, il est possible, par le biais d'un module de combinaison, de déterminer l'amplitude du signal transmis. Affes (1997) suggère l'utilisation d'un simple module de combinaison à rapport maximal (« maximum ratio combiner » ou MRC). Utilisant l'estimation spatio-temporelle du canal $\hat{\underline{H}}_n$, on obtient

$$\hat{s}_n = \frac{\hat{\underline{H}}_n^H \underline{Z}_n}{M}, \tag{2.8}$$

où $\hat{s}_n$ est le symbole reçu (ou « décision douce ») et $\left| \cdot \right|^H$ est la transformée hermitienne de la matrice. Dans le cas de $\hat{\underline{H}}_n$, l'hermitienne équivaut à $\hat{\underline{H}}_n$ transposée où chaque élément prend sa valeur conjuguée.

L'exactitude de $\hat{s}_n$ dépend de l'exactitude de $\hat{\underline{H}}_n$. Ce dernier doit représenter de manière précise et instantanée la structure du canal. La contribution de STAR se situe à deux niveaux: l'estimation en continu des paramètres du canal à partir du PCM, et l'utilisation de ces paramètres pour synthétiser l'estimateur $\hat{\underline{H}}_n$.

2.2.2 Estimation du bit transmis

L'organigramme fonctionnel de STAR est montré à la figure 9 dans le cas d'un seul usager. Chaque antenne branchée au système nécessite la présence de son propre

corrélateur (intégrateur). Deux chemins en émergent: le premier, qui mène $\underline{Z}_n$ au module de combinaison, est la réalisation exacte de l'équation 2.6. Le résultat, $\hat{s}_n$, est une estimation finement quantifiée du bit reçu. La règle de décision qui le suivrait n'est qu'un signum qui interprète le signe de $\hat{s}_n$ de manière à ce que l'estimation du bit reçu, $\hat{b}_n$, n'ait que deux niveaux possibles,

$$\hat{b}_n = \begin{cases} 1 & \text{si } \mathrm{Re}\{\hat{s}_n\} < 0 \\ -1 & \text{si } \mathrm{Re}\{\hat{s}_n\} \geq 0 \end{cases}, \tag{2.9}$$

dans le cas d'une modulation antipodale. D'autres types de modulations commanderaient différents estimateurs. Le second trajet illustré à la figure 9 montre le processus analyse/synthèse qui caractérise STAR et par lequel $\underline{\hat{H}}_n$ est régularisé depuis une série de paramètres observés.

2.2.3 Évaluation de la structure spatio-temporelle du canal

Bien qu'une matrice spatio-temporelle synthétique soit utilisée lors de l'extraction du bit $\hat{s}_n$ de la trame n, deux matrices d'observation contenant les même informations sont maintenues à partir d'un algorithme de DFI (« Decision Feedback Indicator ») (Affes, 1998). La première, $\underline{\breve{H}}_n$, a pour objectif la poursuite de la structure spatio-temporelle du canal avec $\hat{b}_n$ pour seule rétroaction. Elle est définie par

$$\underline{\breve{H}}_{n+1} = \underline{\breve{H}}_n + \zeta\left(\underline{Z}_n - \underline{\breve{H}}_n \hat{b}_n\right)\hat{b}_n^* \tag{2.10}$$

où ζ est un pas d'adaptation réglant l'inertie du système. La seconde, $\underline{\tilde{H}}_n$, est définie par

$$\underline{\tilde{H}}_{n+1} = \underline{\hat{H}}_n + \mu\left(\underline{Z}_n - \underline{\hat{H}}_n \hat{s}_n\right)\hat{s}_n^*, \tag{2.11}$$

où μ est un autre pas d'adaptation réglant la rapidité avec laquelle le système considère la nouvelle structure du canal. Alors que l'équation 2.10 définit la structure du canal avec son propre état précédent, l'équation 2.11 substitue sa valeur précédente $\underline{\tilde{H}}_n$ par la matrice spatio-temporelle synthétisée du canal, $\underline{\hat{H}}_n$, aux mêmes fins. Ceci lui garantit une stabilité accrue puisqu'elle contraint la structure du canal dans un état sain et empêche l'accumulation d'erreurs (bruit, biais, etc.). Elle constitue d'ailleurs la base du traitement subséquent. Le rôle de $\underline{\breve{H}}_n$ est précisé à la section 2.2.8.

Les formes typiques de $\left|\underline{\tilde{H}}_n\right|$ et $\left|\underline{\hat{H}}_n\right|$, pour un récepteur à une seule antenne, sont données à la figure 10. Elles renseignent à la fois sur la relation temporelle entre les multi-trajets perçus ainsi que sur leur puissance relative. La première, $\left|\underline{\tilde{H}}_n\right|$, contient l'apport de la trame *n* du PCM, et sert de point de départ à l'analyse du canal. La seconde, $\left|\underline{\hat{H}}_n\right|$, est une version estimée qui provient de la projection d'une réponse impulsionnelle idéale par les paramètres de l'analyse, $\hat{\tau}$ et $\hat{\boldsymbol{J}}$, eux-mêmes extraits de $\left|\underline{\tilde{H}}_n\right|$.

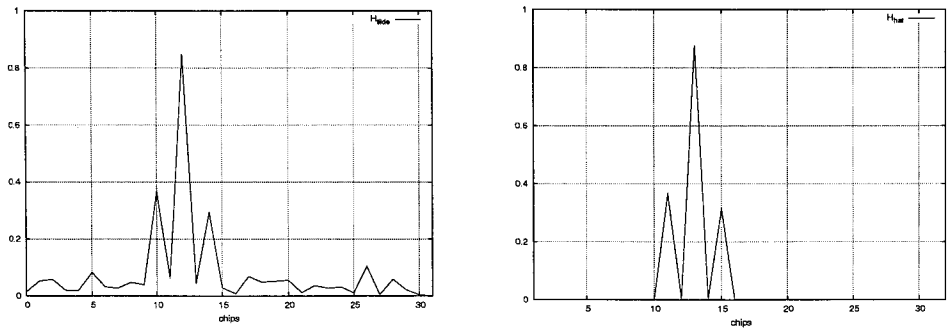


Figure 10 Forme typique de $\left|\underline{\tilde{H}}\right|$ et $\left|\underline{\hat{H}}\right|$

2.2.4 Séparation spatio-temporelle

Les manipulations effectuées sur $\underline{\tilde{H}}_n$ appartiennent à deux catégories: l'analyse spatiale, et l'analyse temporelle. L'opération de séparation spatio-temporelle,

$$\hat{\boldsymbol{J}}_{n+1}=\left(\hat{\boldsymbol{D}}_n^T\hat{\boldsymbol{D}}_n\right)^{-1}\hat{\boldsymbol{D}}_n^T\tilde{\boldsymbol{H}}_{n+1}^T, \tag{2.12}$$

engendre la matrice de pondération spatiale $\hat{\boldsymbol{J}}_{n+1}$ à partir du support temporel $\hat{\boldsymbol{D}}_n$ calculé à l'itération précédente, et de l'observation du canal, $\underline{\tilde{H}}_{n+1}$. En contrepartie, $\hat{\boldsymbol{D}}_{n+1}$ est basé sur ce $\hat{\boldsymbol{J}}_{n+1}$. Leur résolution est donc indissociable.

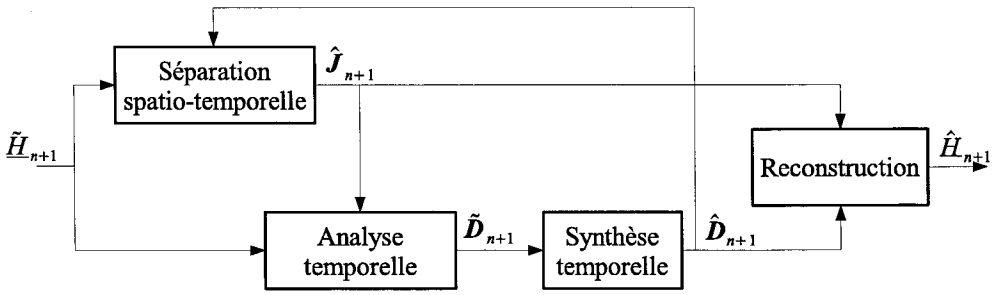


Figure 11 Dualité du traitement spatio-temporel dans STAR

La figure 11 montre la dualité des deux opérandes et leur rôle dans l'algorithme d'identification du canal. La section supérieure montre l'utilisation sans modification des informations spatiales, alors que le chemin inférieur illustre la manipulation analyse/synthèse menant à la reconstruction spatio-temporelle.

2.2.5 Estimation du support temporel

La première étape de l'analyse temporelle est la mise-à-jour de la matrice $\hat{\boldsymbol{D}}_{n+1}$ à l'aide d'une procédure de type LMS,

$$\tilde{\boldsymbol{D}}_{n+1}=\hat{\boldsymbol{D}}_n+\frac{\xi}{M}\left(\tilde{\boldsymbol{H}}_{n+1}^T-\hat{\boldsymbol{D}}_n\hat{\boldsymbol{J}}_{n+1}^T\right)\hat{\boldsymbol{J}}_{n+1}^*, \tag{2.13}$$

où ξ est un pas d'adaptation. Ceci entraîne à une estimation renouvelée de la forme temporelle du canal. Le pas d'adaptation ξ doit être choisi de manière à laisser poindre de

nouveaux multi-trajets sans toutefois introduire de manière excessive du bruit risquant de dégrader les performances.

2.2.6 Identification des délais

La procédure de poursuite des P trajets consiste à évaluer la différence de phase dans le domaine fréquentiel de chaque colonne de $\tilde{\boldsymbol{D}}_{n+1}$ et de $\hat{\boldsymbol{D}}_n$ (notées $\tilde{\mathcal{D}}_{n+1}$ et $\hat{\mathcal{D}}_n$), c'est-à-dire

$$\begin{aligned} \tilde{\mathcal{D}}_{n+1,p} &= FFT\left(\tilde{\boldsymbol{D}}_{n+1,p}\right) \\ \hat{\mathcal{D}}_{n,p} &= FFT\left(\hat{\boldsymbol{D}}_{n,p}\right) \end{aligned} \qquad p \in [0 \cdots P-1] , \tag{2.14}$$

où chaque colonne de $\tilde{\boldsymbol{D}}_{n+1}$ et de $\hat{\boldsymbol{D}}_n$ subit une transformée rapide de Fourier (TFR) sur L points.

L'estimation du déplacement des délais s'effectue dans le domaine fréquentiel par une estimation de changement de phase. On pose l'hypothèse que ce déplacement est inférieur à ½ T_C entre deux observations, ce qui correspond à un déphasage maximal de $\pm\pi$. Cette hypothèse est valide tant que le rythme d'analyse/synthèse demeure suffisamment rapide pour évaluer un délai avant qu'il ne dérive trop. La procédure de poursuite développée par Affes (1997) anticipe un déphasage quasi-linéaire. On peut alors utiliser le résidu de la multiplication entre $\tilde{\mathcal{D}}^*_{n+1}$ et $\hat{\mathcal{D}}_n$,

$$\delta\hat{\varphi} \;=\; \text{phase}\left(\hat{\mathcal{D}}_n \cdot \tilde{\mathcal{D}}^*_{n+1}\right) , \tag{2.15}$$

où le vecteur $\delta\hat{\varphi}$ de longueur L_Δ aura la forme générale d'une droite dont la pente correspond à la différence de phase recherchée. La figure 12 présente une forme possible de ce vecteur.

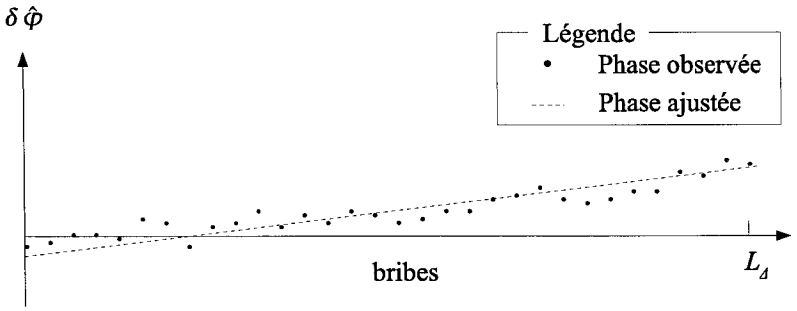


Figure 12 Régression linéaire sur la différence de phase

Par régression linéaire, on peut estimer la droite rejoignant le mieux les éléments du vecteur $\delta\hat{\varphi}$. D'après l'équation générale de la droite,

$$y = ax_i + b \ , \tag{2.16}$$

où les x_i correspondent aux éléments de $\delta\hat{\varphi}$, la pente a correspondrait à la phase recherchée. Utilisant le développement des moindres carrés, on trouve

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} n & \sum_{i=1}^{L_\Delta} \delta\hat{\varphi}_i \\ \sum_{i=1}^{L_\Delta} \delta\hat{\varphi}_i & \sum_{i=1}^{L_\Delta} i^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{L_\Delta} \delta\hat{\varphi}_i \\ \sum_{i=1}^{L_\Delta} i\,\delta\hat{\varphi}_i\, y_i \end{bmatrix} , \tag{2.17}$$

où l'inverse de la matrice 2 x 2 est donné par

$$\frac{1}{L_\Delta \sum_{i=1}^{L_\Delta} i^2 - \left(\sum_{i=1}^{L_\Delta} i\right)^2} \begin{bmatrix} \sum_{i=1}^{L_\Delta} \delta\hat{\varphi}_i \sum_{i=1}^{L_\Delta} i^2 - \sum_{i=1}^{L_\Delta} i \sum_{i=1}^{L_\Delta} i\,\delta\hat{\varphi}_i \\ n\sum_{i=1}^{L_\Delta} i\,\delta\hat{\varphi}_i - \sum_{i=1}^{L_\Delta} i \sum_{i=1}^{L_\Delta} \delta\hat{\varphi}_i \end{bmatrix} . \tag{2.18}$$

Remplaçant l'équation 2.14 dans 2.13 et ne conservant que la solution pour a, on obtient

$$a=\frac{1}{L_\Delta\sum_{i=1}^{L_\Delta} i^2-\left(\sum_{i=1}^{L_\Delta} i\right)^2}\left(n\sum_{i=1}^{L_\Delta} i\,\delta\hat{\varphi}_i-\sum_{i=1}^{L_\Delta} i\sum_{i=1}^{L_\Delta}\delta\hat{\varphi}_i\right), \tag{2.19}$$

où a est la différence de phase recherchée. Elle est transformée en fraction de bribes en multipliant par le ratio approprié, c'est-à-dire

$$\delta\hat{\tau}=\frac{L_\Delta}{2\pi}a\ , \tag{2.20}$$

d'où l'équation 2.21,

$$\hat{\tau}_{p,n+1}=\hat{\tau}_{p,n}+\frac{L}{2\pi}\left(\frac{n\sum_{i=1}^{L_\Delta} i\,\delta\hat{\varphi}_i-\sum_{i=1}^{L_\Delta} i\sum_{i=1}^{L_\Delta}\delta\hat{\varphi}_i}{L_\Delta\sum_{i=1}^{L_\Delta} i^2-\left(\sum_{i=1}^{L_\Delta} i\right)^2}\right), \tag{2.21}$$

qui permet la mise à jour de $\hat{\tau}$.

Les nouvelles valeurs $\hat{\tau}_{n+1,p}$ déclenchent la procédure de synthèse temporelle. Puisqu'elles représentent le point central de chacune des P instances de la réponse impulsionnelle de référence, $h_{RRC}(t)$, les valeurs de $\hat{\tau}_{n+1,p}$ à elles seules permettent de synthétiser la matrice de support temporel $\hat{\boldsymbol{D}}_{n+1}$, nettement moins bruitée que $\tilde{\boldsymbol{D}}_{n+1}$. La matrice de support temporel est définie par

$$\hat{D}_{n+1}=\left[\ h_{RRC}[\hat{\tau}_{n+1,0}]^T \quad h_{RRC}[\hat{\tau}_{n+1,1}]^T \quad \cdots \quad h_{RRC}[\hat{\tau}_{n+1,P-1}]^T\ \right], \tag{2.22}$$

où chaque $h_{RRC}^T(\hat{\tau}_p)$ est une version retardée par $\hat{\tau}_p$ par rapport au gabarit $h_{RRC}(t)$.

2.2.7 Reconstruction

En dernier lieu, la représentation spatio-temporelle du canal est reconstruite à l'aide du nouveau support temporel de synthèse et des ratios de puissances observés. Puisque les colonnes de $\hat{\boldsymbol{D}}_{n+1}$ sont chacune d'énergie unitaire, on amplifie directement par le coefficient correspondant de $\hat{\boldsymbol{J}}_{n+1}$ trouvé en (2.12):

$$\hat{\boldsymbol{H}}_{n+1}=\hat{\boldsymbol{J}}_{n+1}\hat{\boldsymbol{D}}_{n+1}^{T} \ . \tag{2.23}$$

Afin de correspondre à la forme vectorielle du PCM, une simple mise en forme est finalement requise:

$$\underline{\hat{H}}_{n+1}=\left[\hat{\boldsymbol{H}}_{1,l} \quad \hat{\boldsymbol{H}}_{2,l} \quad \cdots \quad \hat{\boldsymbol{H}}_{M,l}\right]^{T} \ . \tag{2.24}$$

C'est ce vecteur qu'utilise l'équation 2.8 pour la récupération des bits transmis.

2.2.8 Gestion des trajets

Bien que la procédure de régularisation suive l'évolution des P trajets perçus, la nature dispersive du canal laisse présager l'apparition et la disparition des multi-trajets. Ceci implique une variation du paramètre P, et donc un changement des dimensions de la plupart des opérandes énumérées jusqu'à maintenant.

La détection de nouveaux trajets s'effectue par l'analyse périodique de $\breve{\boldsymbol{H}}$, la matrice spatio-temporelle issue de la procédure DFI non contrainte (équation 2.10).

Alors que la réponse synthétisée ne représente que les P trajets perçus, $\breve{\boldsymbol{H}}$ est entièrement libre de laisser poindre de nouveaux trajets, si faibles et si instables soient-ils. La présence suffisamment longue dans $\breve{\boldsymbol{H}}$ d'un pic absent de $\hat{\boldsymbol{H}}$ peut entraîner une modification du paramètre P. La détection de ces changements s'effectue suivant le calcul de la norme M-

dimensionnelle (sur les M antennes) à chaque délai entier n T_C permis sur la plage de L_Δ bribes. Cette norme se calcule par

$$\begin{aligned}\left|\underline{\breve{h}}_l\right|^2 &= \sum_{m=1}^{M}\left|\breve{h}_{m,l}\right|^2 \\ &= \sum_{m=1}^{M}\left(\mathrm{Re}\left\{\breve{h}_{m,l}\right\}\right)^2+\left(\mathrm{Im}\left\{\breve{h}_{m,l}\right\}\right)^2,\end{aligned} \tag{2.25}$$

où les $\breve{h}_l$ sont les éléments de $\underline{\breve{H}}$. À l'aide d'un seuil de détection v_A, on peut déterminer l'apparition d'un trajet à la position l dans le canal si l'inéquation

$$\left|\breve{h}_l\right| \geq v_A \tag{2.26}$$

est vérifiée. Un nombre n_A de détections consécutives pour le même délai l est exigé afin de réduire les risques d'une mauvaise décision. Une seule observation négative entraînerait la remise à zéro de la séquence. Vraisemblablement, ce mécanisme ne permet la détection de nouveaux trajets qu'à des délais correspondant à des multiples entiers de T_C. Bien qu'imprécise, cette procédure ne demeure cependant qu'un simple point d'entrée dans l'algorithme de poursuite; c'est à ce dernier qu'incombe ensuite la tâche de corriger rapidement la position du nouveau trajet RF.

La disparition de trajets se produit de façon similaire. La norme M-dimensionnelle de chacune des P colonnes de la matrice $\hat{\boldsymbol{J}}$ indique la puissance relative des trajets perçus en tout moment par STAR. On la définit par:

$$\begin{aligned}\left|\hat{j}_p\right|^2 &= \sum_{m=1}^{M}\left|\hat{j}_{m,p}\right|^2 \\ &= \sum_{m=1}^{M}\left(\mathrm{Re}\left\{\hat{j}_{m,p}\right\}\right)^2+\left(\mathrm{Im}\left\{\hat{j}_{m,p}\right\}\right)^2.\end{aligned} \tag{2.27}$$

La comparaison s'effectue avec un seuil v_V choisi préalablement de sorte que

$$\left|\hat{J}_p\right| \leqslant v_V \tag{2.28}$$

soit vérifiée durant n_V observations consécutives entraîne le retrait du trajet p.

À noter que les seuils de puissance v_A et v_V ainsi que les seuils d'observations consécutives n_A et n_V sont arbitraires et déterminés par essais expérimentaux. Ils dépendent essentiellement des conditions ponctuelles du canal de transmission, ce qui pousse à rendre ces seuils adaptatifs et/ou ajustables par un algorithme de plus haut niveau connaissant l'état du système de transmission au sens large et capable de guider le récepteur à cet égard.

2.2.9 Extraction de la puissance

Le fonctionnement d'une station de base cellulaire dans le standard 3G impose aux équipements mobiles (les téléphones cellulaires des usagers) un contrôle de la puissance de transmission, suivant les consignes périodiques de la station de base. Cette procédure a pour but d'enrayer le phénomène d'éblouissement (« near-far effect ») qui survient lorsqu'un usager à proximité de la station de base submerge les signaux d'usagers lointains. Concrètement, on exige des utilisateurs à proximité de diminuer leur puissance de transmission au seuil minimum permettant le maintien du lien radio, et à ceux qui sont éloignés de l'augmenter. Dans un système équilibré, tous et chacun devraient ainsi être perceptibles par la station de base avec une puissance reçue comparable.

Chaque bit $\hat{s}_n$ provenant du module de combinaison fait ainsi l'objet d'un calcul de puissance,

$$\tilde{\psi}_{n+1}^2 = \mathrm{Re}\left\{\hat{s}_{n+1}\right\}^2 + \mathrm{Im}\left\{\hat{s}_{n+1}\right\}^2 , \tag{2.29}$$

qui est introduit dans un filtre à moyenne variable,

$$\hat{\psi}_{n+1}^2 = \alpha\tilde{\psi}_{n+1}^2 + (1-\alpha)\hat{\psi}_n^2 , \tag{2.30}$$

où le facteur α est choisi selon les conditions du canal, mais prend généralement une valeur de l'ordre de 0,03. L'asservissement en puissance est simplement définie par

$$\begin{aligned} \hat{\psi}^2_{n+1} > 1 &\rightarrow \text{augmenter la puissance de 0,5 dB et} \\ \hat{\psi}^2_{n+1} \leqslant 1 &\rightarrow \text{diminuer la puissance de 0,5 dB,} \end{aligned} \tag{2.31}$$

où la puissance unitaire est assimilée à la commande de diminution sans réelles conséquences puisque ce cas est, au mieux, rarissime. De plus, représenter la commande par un nombre binaire (1 ou 0) oblige une décision de part ou d'autre, sans possibilité de zone neutre. Cette information initiale provenant du récepteur est introduite dans l'algorithme de contrôle de la puissance, tel que défini par la spécification 3GPP. La commande d'asservissement résultante est transmise au taux de 1500 Hz aux équipements mobiles, ce qui correspond à une période de 666,66 µs, soit 80 symboles lorsque $L = 32$. La figure 13 illustre l'assemblage des opérations décrites dans cette section.

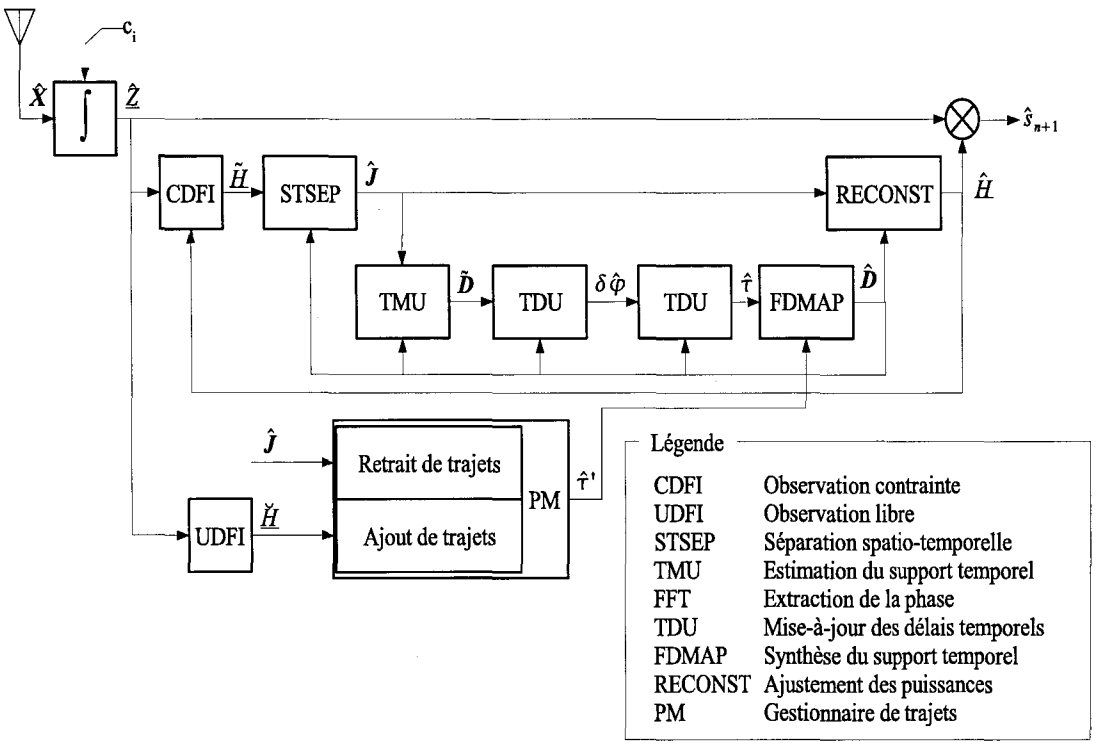


Figure 13 Schéma des opérations principales dans STAR

2.3 Avantages

Contrairement aux différentes saveurs de récepteurs RAKE (Price, 1958) (Bottomley, 2000) (Molisch, 2001) (Khalaj, 1994), STAR exploite la structure post-corrélation du signal, de forme connue et simple à répliquer. Les variables exploitées lors de la régularisation, $\hat{\boldsymbol{J}}$, $\tilde{\boldsymbol{D}}$ et $\underline{\tilde{H}}$, sont déduites exclusivement du PCM. Ceci permet de se prémunir contre les effets destructifs du canal (pré-corrélation).

2.3.1 Identification fréquentielle des délais à partir du PCM

Les récepteurs CDMA actuels déterminent la position des multi-trajets lors de la corrélation avec la séquence d'étalement c_i. Ils sont donc en proie au facteur de suréchantillonnage du convertisseur analogique/numérique (CAN) utilisé. Pour un taux de suréchantillonnage de 8 par exemple (donc $30{,}72 \times 10^6$ éch./s), on peut espérer une précision de $0{,}125\,T_C$ (ou de manière équivalente, à une phase de 45°) puisque le récepteur RAKE n'effectue pour chaque délai connu que trois corrélations ponctuelles: la première à $\hat{\tau}_p$, et les deux autres à $\tau_p - 0{,}125\,T_C$ et $\tau_p + 0{,}125\,T_C$ en guise de recherche (mécanisme « early-late gate »).

STAR effectue plutôt la mise-à-jour des trajets en employant une technique de poursuite spatiale telle que définie par Affes (1996) et conçue à l'origine pour la localisation les sources émettrices (« Direction of Arrival » ou DOA). Elle permet de localiser l'emplacement des délais avec une précision de l'ordre de $0{,}001\,T_C$, sans avoir recours aux techniques de suréchantilonnage, ce qui se traduit par une erreur quadratique moyenne (e.q.m.) inférieure à -20 dB. En pratique, la détermination des $\hat{\tau}_p$ dans le domaine fréquentiel permet d'atteindre cette précision à peu de frais en termes de complexité de calcul. En fonction des erreurs cumulées dans les opérandes $\hat{\boldsymbol{D}}$ et $\tilde{\boldsymbol{D}}$ et de la précision avec laquelle on parvient à exprimer la fonction tangente inverse lors du calcul de la régression linéaire (section 2.2.6), il est possible d'atteindre un niveau de précision arbitraire atteignant $2^{-10}\,T_C$ (soit environ $10^{-3}\,T_C$ ou une phase de 0,35°). Bien que l'on ne puisse en

pratique distinguer deux trajets séparés de moins de T_C, le positionnement de trajets distincts avec une haute précision de l'ordre de $10^{-3} T_C$ permet de rehausser le niveau de performance global puisqu'il minimise les erreurs de synchronisation (Molisch, 2001).

2.3.2 Nature adaptative de l'algorithme

Le canal choisi pour développer le récepteur doit représenter le plus fidèlement possible l'environnement où il sera déployé. Pour cette raison, le modèle de Rayleigh est sélectionné et paramétré de manière à donner un profil de transmission multi-trajets plausible en milieu urbain, sans lien direct (« line of sight » ou LOS) avec l'émetteur, avec des ratios de puissance égaux. Les effets d'évanouissement sont aussi reproduits selon une période ajustable. L'étalement Doppler, paramétrable, affecte aussi de manière indépendante les trajets choisis. Tous ces paramètres exigent de STAR un certain rythme d'adaptation, et c'est justement cette capacité à suivre rapidement et précisément l'évolution du canal qui rend STAR attrayant.

Le premier et plus important élément adaptatif est la gestion du nombre de trajets. Au nombre de P, les trajets apparaissent et disparaissent selon le rythme d'observation du canal par les régions STRF et PM. Les seuils de puissance v_A et v_V ainsi que les seuils n_A et n_V des compteurs à hystérésis sont fixés à priori, mais constituent des paramètres de choix pour un asservissement faisant appel à des statistiques de plus haut niveau (« high order statistics » ou HOS). De manière semblable, les pas d'adaptation ζ, μ et ξ des procédures LMS sont candidats à un asservissement par une procédure de niveau hiérarchique supérieur (Jomphe, 2005).

Le taux de régularisation du canal est lui aussi sujet à des modifications. Un environnement hostile commande assurément des ajustements plus rapides, mais une transmission puissante et stationnaire peut se satisfaire d'ajustements moins fréquents. Dans un récepteur où certaines parties de l'algorithme seraient dupliquées pour permettre le fonctionnement multi-usagers, l'allocation adaptative des ressources devrait s'appuyer

sur la qualité de transmission de chaque équipement mobile. Un algorithme d'aiguillage (ou « scheduling ») adaptatif trouverait sans contredits un usage dans cette situation.

2.4 Conclusion

Ce chapitre a effectué un survol des aspects théoriques du récepteur STAR. On y a d'abord expliqué son paradigme de réception basé sur une procédure d'analyse et de synthèse qui repose sur son modèle de données post-corrélation (PCM). Ses avantages ont ensuite été précisés, et sa nature adaptative, démontrée. Ces informations serviront de base au développement de l'architecture au chapitre suivant, pour ensuite être traduites en structures logiques aux chapitres 4 et 5.

CHAPITRE 3

ARCHITECTURE MATÉRIELLE

3.1 Introduction

Jusqu'à présent, seul l'aspect théorique du fonctionnement de STAR a été abordé. Le chapitre précédent démontrait que l'ensemble des opérations mathématiques de cet algorithme ne ressemblait en rien à ce qui constitue un récepteur RAKE classique. Cet ensemble d'opérations suggère un classement en trois groupes: corrélation/combinaison, analyse/synthèse et gestion des trajets. Cette observation est la base de ce chapitre qui aura pour thème principal l'élaboration d'une architecture globale pour le récepteur. La présence de rétroactions dans ces groupes sera elle aussi mise à profit pour parvenir à une architecture convenable.

Le chapitre se divise en quatre sections. La première aborde le problème de la plus longue rétroaction de l'algorithme et propose en guise de solution une architecture pipelinée capable, entre autres, de gérer plusieurs utilisateurs. Elle identifie ensuite formellement trois « domaines de calcul » à partir des observations du chapitre précédent. La section 3.3 présente ensuite une évaluation détaillée de la puissance de calcul requise pour chaque opération mathématique par unité de temps. Une analyse considérant deux taux de calcul réduits démontrera ensuite l'avantage d'un fonctionnement déséquilibré de ces trois domaines. La section 3.4, quant à elle, traite de la quantification de l'algorithme et propose une évaluation détaillée de deux opérandes critiques, $\hat{\tau}$ et $\boldsymbol{H}$.

En dernier lieu, la section 3.5 réunit les résultats du chapitre pour appuyer le choix d'une architecture modulaire et codesign. On y explique l'impact des taux de traitement réduits sur la bande passante entre les domaines de calcul, et les avantages reliés à la réalisation en micrologiciel du module de gestion des trajets. On retrouve aussi un modèle de gestion

pour un contrôleur centralisé capable d'agir en tant que mandataire des niveaux hiérarchiques supérieurs auprès des unités de calcul rapides de bas niveau.

3.2 Portrait global

La figure 13 (section 2.2.9) présentait un schéma-bloc global du fonctionnement de STAR où l'on constatait un asservissement en boucle fermée. Quelques rétroactions sont en effet présentes, mais celle qui asservit la section analyse/synthèse est la plus contraignante étant donnée la latence qu'elle impose à la régularisation répétée de la structure du canal. Le noyau de calcul se situe donc entre les équations 2.12 et 2.24, inclusivement.

Puisque ces équations doivent obligatoirement s'enchaîner, il est impossible de concevoir une architecture où elles résoudraient en parallèle le même problème. Il est néanmoins possible de les évaluer en même temps en autant qu'elles traitent, à un instant précis, des données provenant de différentes instances du même problème.

3.2.1 Architecture pipelinée

Considérant l'hypothèse d'un seul et unique récepteur STAR capable de traiter K utilisateurs, la figure 14 montre une assignation possible des K problèmes à l'ensemble des W équations de l'algorithme. Le problème de l'utilisateur k est traité en séquence par W entités de calcul, chacune adaptée à la solution d'une équation spécifique. L'entité de calcul w traite sans cesse la même partie du problème, en séquence pour les utilisateurs 1 à K. Cette architecture est dite « pipelinée ». Les différentes variables du problème sont propagées d'un étage du pipeline au suivant selon une cadence prédéfinie (section 4.5.12). La période de chaque séquence est déterminée par le plus long des calculs à effectuer dans l'ensemble actuel des K problèmes répartis sur les W processeurs. Il s'agit donc d'un pipeline à cadence statique.

Ceci diffère d'un pipeline classique puisque l'opération effectuée par un étage w n'est pas, d'un problème k à l'autre, de même complexité étant donnée la taille changeante des

opérandes à traiter. Bien qu'elle permette une certaine forme de parallélisme global et entraîne une augmentation considérable du nombre de problèmes résolus dans un même laps de temps, l'architecture pipelinée entraîne un délai dans la résolution de chaque problème *k* considéré individuellement.

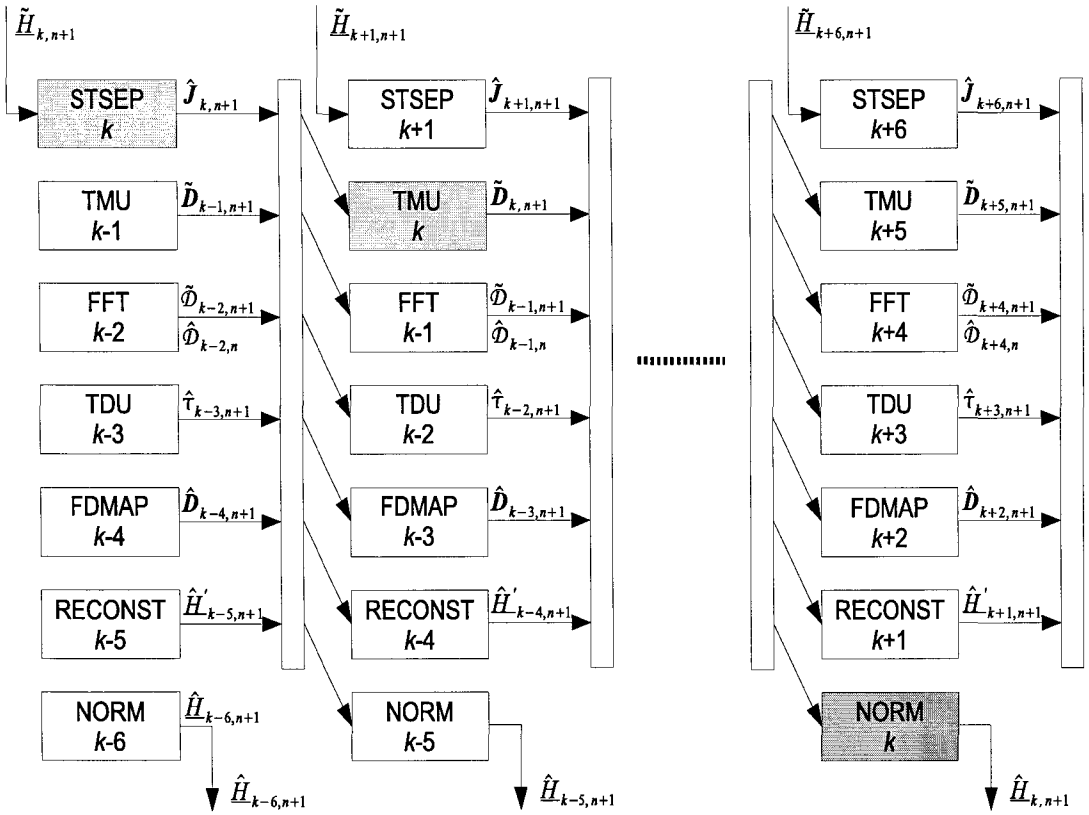


Figure 14 Architecture pipelinée à $W = 7$ étages

3.2.2 Séparation des domaines de calcul

Les équations de STAR peuvent se regrouper en trois grandes catégories. Ces associations permettent d'isoler des régions de l'algorithme qui opèrent à différents rythmes et faciliteront l'expansion du système grâce à la modularité qu'elles entraînent.

3.2.2.1 Décodage des symboles, ou « Symbol Path » (SP)

Le taux auquel le récepteur reçoit les nouveaux symboles est fixé par le taux de bribes et le facteur d'étalement L. La corrélation et la combinaison doivent en tout temps soutenir ce rythme. En référence à la figure 5, le chemin supérieur doit impérativement être complété en moins de LT_C, ou 8,33 µs lorsque $L = 32$.

3.2.2.2 Analyse et synthèse du canal, ou « Structure Fitting » (STRF)

Le mécanisme d'analyse et de synthèse du canal constitue un second ensemble appelé STRF (« Structure Fitting »). Il regroupe les opérations de séparation spatio-temporelle, de mise à jour du support temporel, de transformée de Fourier, de détection de phase, de synthèse de la réponse temporelle et de reconstruction spatio-temporelle. Essentiellement, la région STRF se présente entièrement sous forme pipelinée.

3.2.2.3 Gestion des trajets, ou « Path Management » (PM)

Une dernière région effectue le calcul des équations 2.25 et 2.27 et comporte les mécanismes d'hystérésis qui accomplissent la surveillance des multi-trajets dans le canal. Puisque les calculs accomplis par ce sous-système sont davantage à caractère logique et sont les premiers candidats à l'innovation, il est particulièrement intéressant de les isoler afin de permettre leur migration vers un microprocesseur.

3.2.2.4 Fréquence de traitement

Dans le cas idéal, l'arrivée de chaque nouveau symbole devrait déclencher la mise à jour de l'estimation de la matrice de propagation spatio-temporelle $\underline{\hat{H}}_{n+1}$, c'est-à-dire que la section STRF devrait, elle aussi, se compléter en moins de LT_C secondes. Cette contrainte est difficilement atteignable étant donné la quantité importante de calculs par seconde qu'elle entraîne.

Cheikhrouhou (2001) propose une solution tangible à ce problème en diminuant le taux de mise à jour de $\underline{\hat{H}}_{n+1}$ par un facteur n_{STRF} (noté « n_{ID} » dans l'article). Cette observation provient de la stationnarité du canal WCDMA typique que l'on observe sur un horizon de quelques dizaines de microsecondes, tout en concédant que cette période varie selon l'environnement. Posant 80 µs comme valeur de départ et considérant toujours un facteur d'étalement L variable, on calcule le nombre de symboles pouvant être reçus avec le même $\underline{\hat{H}}_{n+1}$ sans dégrader les performances de façon indue par

$$n_{STRF} = \left\lceil \frac{3{,}84\,10^6\, bribes/s \cdot 80\,\mu s}{L} \right\rceil , \tag{3.1}$$

où n_{STRF} vaut 10 lorsque L=32.

Permettre $n_{STRF} > 1$ permet d'allouer davantage de temps à la résolution des équations 2.12 à 2.23, donc de réduire le nombre de calculs par secondes pour un canal en particulier. Le résultat de l'équation 3.1 constitue la limite supérieure pour n_{STRF}, soit le plus faible taux de rafraîchissement permis. L'architecture retenue imposera, pour sa part, une valeur inférieure sur n_{STRF}, soit le taux le plus rapide atteignable. Dans un système réel, le taux de rafraîchissement de $\underline{\hat{H}}_{n+1}$ se situera entre ces deux bornes.

La troisième et dernière observation concerne la surveillance des trajets naissants et des trajets en extinction. Ces événements surviennent lorsque la topologie de l'environnement de propagation évolue, lorsque l'effet d'évanouissement (« *fading* ») survient, ou lorsque le phénomène d'éblouissement (« near-far effect ») se manifeste. Prasad (1996) note que ces événements surviennent plutôt rarement par rapport au débit du message transmis. Les simulations effectuées par Cheikhrouhou (2001) suggèrent qu'une période de 10 n_{STRF} symboles peut s'écouler entre deux évaluations successives des équations 2.25 à 2.29 sans toutefois dégrader les performances de réception. De plus, le mécanisme d'hystérésis imposé par n_A et n_V réduit davantage le nombre de transactions sur le nombre P, minimisant les arrêts ponctuels du pipeline STRF requis par un tel changement (section

5.3.2.1). À nouveau, les valeurs exactes de ces paramètres ainsi que les seuils de puissance v_A et v_V associés dépendent de l'état ponctuel du canal de transmission.

En résumé, on retient que la section SP doit être évaluée au taux de symbole T_S, la section STRF à $n_{STRF}\ T_S$ et la section PM à environ $n_{PM} = 10\ n_{STRF}\ T_S$.

3.3 Puissance de calcul

Avant d'établir une réalisation tangible, il importe d'effectuer le décompte des opérations mathématiques requises par l'algorithme, et d'établir à quel rythme elles doivent être effectuées. Cette information, quoiqu'insuffisante, permet d'inférer le nombre de ressources matérielles nécessaires.

La puissance de calcul requise par un algorithme s'exprime en nombre d'opérations par secondes (« ops »), où une opération est une multiplication ou une addition de deux scalaires réels, effectuée en notation à virgule fixe (section 3.4).

3.3.1 Nature des calculs

Toutes les équations présentées au chapitre 2 peuvent être traduites en opérations élémentaires appartenant à cinq catégories: multiplication matricielle, corrélation, transformée de Fourier, régression linéaire et normalisation. On doit faire, pour chaque catégorie, le décompte des multiplications et additions réelles et ce, en fonction de la taille de leurs opérandes.

La suite du texte utilise le terme *opérande* pour désigner une matrice servant d'argument à une fonction matricielle. La *taille* de cette opérande fait référence aux dimensions en nombre de rangées et de colonnes de cette matrice. Les *éléments* sont les composantes de la matrice, et leur *quantification* désigne le nombre de bits utilisés pour les représenter en notation binaire à virgule fixe.

3.3.1.1 Rappel: opérations complexes

Bien qu'on puisse considérer la multiplication complexe comme une fonction de deux arguments, on l'évoque ici dans sa forme de fonction à quatre arguments réels afin de chiffrer correctement la puissance de calcul du système. Le développement est trivial:

$$\begin{aligned} \mathrm{Re}\{a\cdot b\} &= \mathrm{Re}\{a\}\cdot\mathrm{Re}\{b\}-\mathrm{Im}\{a\}\cdot\mathrm{Im}\{b\}\,\text{et} \\ \mathrm{Im}\{a\cdot b\} &= \mathrm{Re}\{a\}\cdot\mathrm{Im}\{b\}+\mathrm{Im}\{a\}\cdot\mathrm{Re}\{b\}. \end{aligned} \tag{3.2}$$

On généralise ensuite comme suit:

$$1\,\text{mult. complexe} = 4\,\text{mult. réelles} + 2\,\text{sommations réelles}. \tag{3.3}$$

L'addition complexe se résout simplement par deux additions réelles,

$$\begin{aligned} \mathrm{Re}\{a+b\} &= \mathrm{Re}\{a\}+\mathrm{Re}\{b\}\,\text{et} \\ \mathrm{Im}\{a+b\} &= \mathrm{Im}\{a\}+\mathrm{Im}\{b\}, \end{aligned} \tag{3.4}$$

d'où

$$1\,\text{addition complexe} = 2\,\text{additions réelles}. \tag{3.5}$$

3.3.1.2 Corrélation

L'équation 2.3 est la pierre angulaire du récepteur à étalement spectral. Celui utilisé par STAR doit cependant fonctionner en continu pour former le modèle post-corrélation.

La corrélation se présente comme une intégration temporelle entre deux séquences quelconques. On la considère dans le cadre exclusif d'un récepteur WCDMA à modulation bipolaire où elle combine un vecteur de réception (complexe) et un code d'étalement (réel). La première séquence est de durée infinie mais subdivisée en trames de L éléments. Le code d'étalement, de taille L, doit se répéter périodiquement à chaque symbole entrant (dans le cas d'un code court, ou « short code »). La corrélation n'étant qu'une somme de produits, on la définit ponctuellement comme

$$z_m[n] = \sum_{i=0}^{L-1} y_m[n-i] \cdot c[i], \quad m \in [0 \ldots M-1] \tag{3.6}$$

où M est le nombre d'antennes reliées au récepteur. Suivant le facteur d'étalement L qui a cours au moment de la corrélation, on compte

$$\begin{array}{ll} ML^2 & \text{multiplications réelles et} \\ ML(L-1) & \text{additions réelles,} \end{array} \tag{3.7}$$

puisque l'opération s'effectue entre séquences complexe et réelle. La nature de la séquence d'étalement offre la possibilité d'abroger les multiplications. Puisque les éléments du code c sont bipolaires (c.-à-d. -1 et 1), l'équation 3.6 est avantageusement reformulée comme

$$z_n = \sum_{i=0}^{L-1} \pm y_{n-i}, \tag{3.8}$$

qui se résume à une sommation où le choix d'additionner ou de soustraire la bribe i est dicté par la polarité de l'élément c_i de la séquence d'étalement. Le chapitre 4 montre que la réalisation matérielle de ce type de sommation est triviale et exempte de toute ressource multiplicative. Le nombre d'opérations chute à

$$ML(L-1) \quad \text{additions réelles.} \tag{3.9}$$

L'équation 3.8 doit en principe se répéter L fois par antenne pour constituer le PCM sur lequel sont basés tous les calculs subséquents. À chaque trame de L bribes entrantes doit correspondre une trame post-corrélation de L bribes. Plus simplement, à chaque nouvelle bribe entrante, la relation 3.8 est calculée. Le nombre d'itérations croît donc en fonction de L^2, ce qui rend la corrélation en continu prohibitive pour des facteurs d'étalement plus importants. Paradoxalement, cette situation correspond à la réception de taux binaires moins élevés.

Cheikhrouhou (2001) apporte un élément de réponse à ce problème: une fenêtre de traitement réduite sur le PCM. Cette simplification découle du fait qu'en pratique, l'envergure des multi-trajets est bornée, et que ces derniers diminuent en puissance au fur et à mesure qu'ils s'éloignent du trajet principal. On pose donc l'hypothèse que tous les échos utiles au récepteur se retrouvent dans une plage de L_Δ=32 bribes, correspondant à un délai de

$$\frac{L_\Delta}{3{,}84\,10^6\,MHz} = 8{,}33\,\mu\text{s}. \tag{3.10}$$

Si l'on connaît en tout temps la position optimale de cette fenêtre de traitement réduite, l'équation 3.8 doit être calculée seulement L_Δ fois lorsque L>32. Si toutefois L<32, 3.8 est calculée L fois. Pour des facteurs d'étalement supérieurs à 32, la charge du corrélateur se trouve donc nettement réduite pour atteindre

$$ML_\Delta(L-1) \quad \text{additions réelles}. \tag{3.11}$$

3.3.1.3 Decision Feedback Identifiers (DFI), contraint et libre

Les équations 2.10 et 2.11 définissent l'algorithme DFI, en versions contrainte et libre, duquel sont issues respectivement $\underline{\tilde{H}}_{n+1}$ et $\underline{\breve{H}}_{n+1}$, toutes deux de taille $ML_\Delta \times 1$. Les paramètres ζ et μ sont des constantes réelles, et les symboles décodés $\hat{s}$ et $\hat{b}$ sont réels dans le cas d'une modulation bipolaire. Les calculs du DFI se divisent en trois sous-étapes. Premièrement, la multiplication de $\underline{Z}_{n+1}$ avec le symbole décodé entraîne

$$2\text{ML}_\Delta \text{ multiplications réelles}. \tag{3.12}$$

Ce premier résultat intermédiaire est soustrait à $\underline{\hat{H}}_n$, ce qui nécessite

$$2\text{ML}_\Delta \text{ additions réelles}. \tag{3.13}$$

Ce second résultat intermédiaire est ensuite multiplié par un coefficient réel, ce qui nécessite

$$2ML_{\Delta} \text{ multiplications réelles}, \tag{3.14}$$

pour être finalement soustrait à $\underline{\hat{H}}_n$ d'où

$$2ML_{\Delta} \text{ additions réelles}. \tag{3.15}$$

L'opération complète de DFI nécessite donc, en regroupant 3.12 à 3.15,

$$\begin{array}{ll} 4ML_{\Delta} & \text{multiplications réelles et} \\ 4ML_{\Delta} & \text{additions réelles}. \end{array} \tag{3.16}$$

3.3.1.4 Multiplication matricielle

Le développement qui suit illustre le cas général d'une multiplication entre deux matrices complexes de dimensions quelconques. Admettant $\boldsymbol{A}$ et $\boldsymbol{B}$, deux matrices de dimensions $E \times F$ et $F \times G$, respectivement, et la matrice $\boldsymbol{R}$, de dimensions $E \times G$, le produit matriciel $\boldsymbol{R} = \boldsymbol{AB}$ se définit comme

$$r_{i,j} = \sum_{k=1}^{k=F} a_{i,k} \cdot b_{k,j} \qquad \text{où} \qquad \begin{array}{l} i \in \{1 \ldots E\} \text{ et} \\ j \in \{1 \ldots G\}, \end{array} \tag{3.17}$$

où $r_{i,j}$, $a_{i,k}$ et $b_{k,j}$ sont les éléments complexes des matrices $\boldsymbol{R}$, $\boldsymbol{A}$ et $\boldsymbol{B}$, respectivement. Chaque $r_{i,j}$ est déterminé par F multiplications complexes et F - 1 additions complexes. Puisque $\boldsymbol{R}$ comporte EG éléments, le nombre total d'opérations complexes est

$$\begin{array}{ll} (EG)F & \text{multiplications complexes}, \\ (EG)(F-1) & \text{additions complexes}. \end{array} \tag{3.18}$$

Afin de comparer ces résultats au reste de l'algorithme, on remplace 3.3 et 3.5 dans 3.18:

$$\begin{array}{ll} 4EFG & \text{multiplications réelles,} \\ 2EFG+2EG(F-1)=2EG(2F-1) & \text{additions réelles.} \end{array} \tag{3.19}$$

Ce décompte d'opérations est exhaustif et ne tire profit d'aucune hypothèse quant au contenu des matrices. On rencontre, dans plusieurs domaines d'application, des matrices aux propriétés spécifiques permettant d'alléger considérablement la multiplication. Le cas de matrices supérieures ou inférieures, symétriques ou diagonales sont de bons exemples (Vai, 2001). L'algorithme STAR ne peut se prévaloir de ces artifices puisque l'on considère en général que ses matrices sont quelconques.

Le cas de l'addition complexe fait intervenir trois matrices de dimensions quelconques mais égales. Posons $\boldsymbol{A}$ et $\boldsymbol{B}$ deux matrices de dimensions $E \times F$, et la matrice $\boldsymbol{R}$ de dimensions $E \times F$ en guise de résultat. L'addition $\boldsymbol{R} = \boldsymbol{A} + \boldsymbol{B}$ est définie par

$$r_{i,j} = a_{i,k} + b_{k,j} \qquad \text{où} \quad \begin{array}{l} i \in \{1 \ldots E\} \\ j \in \{1 \ldots F\}, \end{array} \tag{3.20}$$

où $r_{i,j}$, $a_{i,k}$ et $b_{k,j}$ sont les éléments complexes des matrices $\boldsymbol{R}$, $\boldsymbol{A}$ et $\boldsymbol{B}$, respectivement. Ce calcul requiert

$$EF \quad \text{additions complexes} \tag{3.21}$$

ou, en remplaçant 3.5 dans 3.21,

$$2EF \quad \text{additions réelles.} \tag{3.22}$$

3.3.1.5 Combinaison

L'équation 2.14 multiplie les matrices $\hat{\underline{Z}}_{n+1}$ et $\hat{\underline{H}}_n$ toutes deux de dimensions $ML_A \times 1$. Le résultat, $\hat{s}_{n+1}$, est un scalaire. Remplaçant les dimensions appropriées dans l'équation 3.19, le nombre de calculs est:

$$\begin{matrix} 4\mathrm{ML}_{\Delta} & \text{multiplications réelles et} \\ 2\left(2\mathrm{ML}_{\Delta}-1\right) & \text{additions réelles.} \end{matrix} \tag{3.23}$$

3.3.1.6 Calcul de la puissance reçue

Puisque ce calcul est simple, il est réalisé dans la partie matérielle. À l'instar de l'opération de surveillance des trajets, on préfère conserver la valeur de la puissance au carré pour éviter une coûteuse racine carrée. Puisque cette information n'est utilisée qu'à un niveau hiérarchique supérieur, hors du récepteur, on peut considérer que la racine carrée finale peut y être effectuée, si nécessaire. D'autre part, l'équation 2.30 comporte un facteur de lissage α ainsi que sa réciproque 1-α. On considère donc la mise au carré de $\hat{s}_{n+1}$ (une multiplication réelle), sa multiplication avec α (une seconde multiplication réelle), la multiplication de Ψ_n^2 avec 1-α (une troisième multiplication réelle) et la sommation finale (une addition réelle). Le nombre d'opérations est

$$\begin{matrix} 3 & \text{multiplications réelles et} \\ 1 & \text{addition réelle.} \end{matrix} \tag{3.24}$$

3.3.1.7 Séparation spatio-temporelle

On multiplie les matrices $\hat{\boldsymbol{D}}_n^T$ et $\hat{\boldsymbol{H}}_{n+1}^T$ de dimensions $P \times L_{\Delta}$ et $L_{\Delta} \times M$, respectivement, pour trouver $\hat{\boldsymbol{J}}_{n+1}$ de taille $M \times P$, conformément à l'équation 2.12. Remplaçant les dimensions dans 3.19, on trouve:

$$\begin{matrix} 4\mathrm{ML}_{\Delta} P & \text{multiplications réelles,} \\ 2\mathrm{MP}\left(2\mathrm{L}_{\Delta}-1\right) & \text{additions réelles.} \end{matrix} \tag{3.25}$$

3.3.1.8 Mise à jour de la matrice de support temporel (« TMU »)

L'équation 2.13 contient en réalité plusieurs opérations distinctes. La première est un produit entre les matrices réelle $\hat{\boldsymbol{D}}_n$ et complexe $\hat{\boldsymbol{J}}_{n+1}^T$ de dimensions $L_\Delta \times P$ et $P \times M$. Remplaçant dans 3.19, on obtient:

$$\begin{array}{ll} 2\mathrm{MPL}_\Delta & \text{multiplications réelles et} \\ 2\mathrm{ML}_\Delta(2\mathrm{P}-1) & \text{additions réelles.} \end{array} \tag{3.26}$$

Ce résultat intermédiaire de dimensions $M \times L_\Delta$ est ensuite soustrait à $\hat{\boldsymbol{H}}_n$. De 3.26, on tire:

$$2\mathrm{ML}_\Delta \qquad \text{additions réelles.} \tag{3.27}$$

Ce nouveau résultat complexe intermédiaire de dimensions $M \times L_\Delta$ doit multiplier $\hat{\boldsymbol{J}}_{n+1}^*$ de dimensions $M \times P$. Utilisant toujours 3.19, on trouve:

$$\begin{array}{ll} 4\mathrm{MPL}_\Delta & \text{multiplications réelles et} \\ 2\mathrm{PL}_\Delta(2\mathrm{M}-1) & \text{additions réelles.} \end{array} \tag{3.28}$$

Finalement, cette dernière opérande intermédiaire de dimensions $P \times L_\Delta$ est additionnée à $\hat{\boldsymbol{D}}_n$, réelle. Reprenant 3.22, on trouve

$$2\mathrm{PL}_\Delta \qquad \text{additions réelles,} \tag{3.29}$$

où l'on comptabilise l'addition de la partie imaginaire inexistante de $\hat{\boldsymbol{D}}_n$ afin de refléter le caractère matériel de l'opération. La mise en commun de 3.26 à 3.29 indique qu'au total, l'opération de mise à jour de la matrice de support temporel $\tilde{\boldsymbol{D}}_{n+1}$ requiert

$$\begin{matrix} 6\mathrm{MPL}_{\Delta} & \text{multiplications réelles et} \\ 8\mathrm{MPL}_{\Delta} & \text{additions réelles.} \end{matrix} \tag{3.30}$$

3.3.1.9 Reconstruction

La réunion des dimensions spatiale et temporelle à l'issue de la région STRF est en réalité une multiplication matricielle entre $\hat{\boldsymbol{D}}_{n+1}$ et $\hat{\boldsymbol{J}}_{n+1}$ de dimensions $L_{\Delta} \times P$ et $P \times M$. Puisque $\hat{\boldsymbol{D}}_{n+1}$ est entièrement réelle, le nombre de multiplications réelles diminue de moitié, et les additions des produits partiels disparaissent. On obtient:

$$\begin{matrix} 2\mathrm{MPL}_{\Delta} & \text{multiplications réelles,} \\ 2\mathrm{ML}_{\Delta}(2\mathrm{P}-1) & \text{additions réelles.} \end{matrix} \tag{3.31}$$

3.3.1.10 Transformée de Fourier

La détection de phase faite à l'équation 2.21 est effectuée indépendamment pour chacune des P colonnes de L_{Δ} éléments complexes de $\tilde{\boldsymbol{D}}_{n+1}$ et de $\hat{\boldsymbol{D}}_{n}$. La puissance de calcul requise pour une transformée de Fourier dépend de l'algorithme choisi. Il est largement accepté qu'un tel calcul doit être effectué par transformée de Fourier rapide (TFR), avec la restriction supplémentaire d'utiliser des séquences dont la taille est une puissance de deux. La nature des codes d'étalement utilisés répond à cette contrainte. À chaque algorithme de TFR correspond un nombre d'opérations différent étant donné leur fonctionnement interne. Puisque le présent travail ne vise pas la réalisation d'une TFR, une version optimisée, la R2^{2}PC, est empruntée à Grandmaison (2005) et sa complexité est conforme à 3.33. Dans le cas d'une TFR de type Radix-2 sur N points, il faut compter

$$\left(\frac{N}{2}\right)\log_2 N \tag{3.32}$$

multiplications complexes. Le nombre d'additions complexes est identique. Puisque $2P$ transformées sont nécessaires, le nombre d'opérations réelles devient

$$\begin{aligned} &2\mathrm{P}\cdot 4\cdot\left(\frac{N}{2}\right)\log_2 N \quad \text{multiplications réelles et} \\ &2\mathrm{P}\cdot 4\cdot\left(\frac{N}{2}\right)\log_2 N \quad \text{additions réelles.} \end{aligned} \tag{3.33}$$

3.3.1.11 Régression linéaire et détection de phase

La détection de phase s'effectue sur le résultat d'une multiplication point à point entre les deux vecteurs complexes $\tilde{\mathcal{D}}^{*}_{n+1}$ et $\hat{\mathcal{D}}_{n+1}$ de dimensions $P \times L_{\Delta}$. Le nombre d'opérations est simplement PL_{Δ} multiplications complexes. Remplaçant dans 3.19, on trouve:

$$\begin{aligned} &4\mathrm{PL}_{\Delta} \quad \text{multiplications réelles et} \\ &2\mathrm{PL}_{\Delta} \quad \text{additions réelles.} \end{aligned} \tag{3.34}$$

Le passage de $\tilde{\mathcal{D}}^{*}_{n+1}\cdot\hat{\mathcal{D}}_{n}$ au vecteur $\delta\hat{\varphi}_i$ est effectué par tangente inverse mais ne nécessite aucun calcul puisqu'il est effectué à l'aide d'une table de valeurs pré-calculées.

L'interprétation de l'équation 2.21 est plus complexe. Si l'on considère le coefficient L_{Δ} comme une constante, le dénominateur est lui aussi constant et assimilé au coefficient $L/2\pi$. Le même raisonnement s'applique au coefficient de la seconde sommation au numérateur qui n'est fonction que de L_{Δ}, donc lui aussi constant. Le calcul se réduit à

$$A\,\frac{L}{2\pi}\left(n\sum_{i=1}^{L_{\Delta}} i\,\delta\hat{\varphi}_i - C\sum_{i=1}^{L_{\Delta}}\delta\hat{\varphi}_i\right), \tag{3.35}$$

où A remplace le dénominateur de l'équation 2.21. Le vecteur $\delta\hat{\varphi}_i$ étant formé d'éléments réels (des angles), on dénombre

$$\begin{array}{ll} L_{\Delta} & \text{multiplications réelles et} \\ 2(L_{\Delta}-1) & \text{additions réelles.} \end{array} \tag{3.36}$$

La puissance de calcul de la détection de phase réunit 3.34, 3.36, ainsi que 3 multiplications et une addition réelle, d'où le total,

$$\begin{array}{ll} 4\mathrm{PL}_{\Delta}+L_{\Delta}+3 & \text{multiplications réelles et} \\ 2\mathrm{PL}_{\Delta}+2\mathrm{L}_{\Delta}-1 & \text{additions réelles.} \end{array} \tag{3.37}$$

3.3.1.12 Normalisation

On choisit d'ajouter une limite à l'opérande $\hat{\boldsymbol{H}}_{n+1}$ afin de minimiser les fluctuations qu'elle subit suivant une mauvaise identification des trajets ou des ratios de puissance de ceux-ci.

La normalisation s'effectue en deux étapes. La première consiste à calculer le carré de la norme Frobenius de la matrice. Pour $\hat{\boldsymbol{H}}_{n+1}$, de dimensions $M \times L_{\Delta}$, on la définit comme

$$\left\|\hat{\boldsymbol{H}}'\right\|_F^2 = \frac{1}{ML_{\Delta}} \sum_{m=1}^{M} \sum_{l=1}^{L_{\Delta}} \left|\hat{h}'_{m,l}\right|^2, \tag{3.38}$$

où la norme de chaque élément complexe $\hat{h}'_{m,l}$ est

$$\left|\hat{h}'_{m,l}\right|^2 = \mathrm{Re}\left\{\hat{h}'_{m,l}\right\}^2 + \mathrm{Im}\left\{\hat{h}'_{m,l}\right\}^2. \tag{3.39}$$

Le nombre d'opérations requises pour effectuer ce calcul est de

$$\begin{array}{ll} 2\mathrm{ML}_{\Delta}+1 & \text{multiplications réelles et} \\ 2\mathrm{ML}_{\Delta}-1 & \text{additions réelles,} \end{array} \tag{3.40}$$

incluant la multiplication par le coefficient $1 / ML_{\Delta}$. La seconde étape consiste à diviser chaque élément de la matrice par la norme trouvée:

$$\hat{h}_{m,l} = \frac{\hat{h}'_{m,l}}{\left\|\hat{\boldsymbol{H}}'\right\|}. \tag{3.41}$$

Lors d'une réalisation matérielle, on assimile la division à une multiplication fractionnaire. On tabule simplement l'inverse d'un coefficient en fonction de lui-même. Le coefficient est alors utilisé comme l'adresse d'une table, et son inverse est simplement stocké à cette adresse.

Le dénominateur de l'équation 3.41 ainsi calculé, on le multiplie à tous les éléments de la matrice, ajoutant $2ML_{\Delta}$ multiplications réelles. Le nombre total d'opérations requises par la normalisation se chiffre donc à

$$\begin{array}{ll} 4\mathrm{ML}_{\Delta}+1 & \text{multiplications réelles et} \\ 2\mathrm{ML}_{\Delta}-1 & \text{additions réelles.} \end{array} \tag{3.42}$$

3.3.1.13 Surveillance des trajets naissants

La détection de trajets naissants dans le canal de transmission s'effectue par comparaison entre la norme de $\breve{\boldsymbol{H}}_{n+1}$ et un seuil prédéterminé (équation 2.26). Les dimensions de $\breve{\boldsymbol{H}}_{n+1}$ sont $M \times L_{\Delta}$ alors que le seuil est un scalaire. On doit établir la puissance associée à chacune des bribes, sur les M antennes, donc calculer L_{Δ} normes M-dimensionnelles. On préfère conserver la valeur au carré de la norme et la comparer au seuil ν_A exprimé lui aussi au carré. L'équation devient alors

$$\breve{h}_l^2 = \frac{1}{M}\sum_{m=1}^{M} \mathrm{Re}\left\{\hat{H}_{m,l}\right\}^2 + \mathrm{Im}\left\{\hat{H}_{m,l}\right\}^2, \tag{3.43}$$

ce qui donne lieu à

$$\begin{array}{ll} 2\mathrm{ML}_{\Delta}+1 & \text{multiplications réelles et} \\ L_{\Delta}(M-1) & \text{additions réelles,} \end{array} \tag{3.44}$$

où la multiplication réelle supplémentaire est due au coefficient $1 / M$. La comparaison entre chaque $\breve{h}_l^2$ s'effectue par soustraction avec v_A. La polarité du résultat indique alors si la norme est supérieure ou inférieure au seuil. Il faut donc compter

$$L_\Delta \quad \text{additions réelles} \tag{3.45}$$

pour la comparaison. En passant sous silence les additions et soustractions qui constituent la mécanique des compteurs à hystérésis, le total d'opérations requises est

$$\begin{array}{ll} 2ML_\Delta+1 & \text{multiplications réelles et} \\ ML_\Delta & \text{additions réelles.} \end{array} \tag{3.46}$$

3.3.1.14 Surveillance des trajets courants

La surveillance de la matrice $\hat{\boldsymbol{J}}_{n+1}$ suffit à déterminer si la puissance des trajets observés par STAR est toujours adéquate pour être constructive dans le processus de combinaison. L'opération mathématique est analogue à 3.43, à la seule différence qu'on extrait plutôt la norme M-dimensionnelle de chacune des P colonnes de $\hat{\boldsymbol{J}}_{n+1}$ par

$$\hat{j}_p^2=\frac{1}{M}\sum_{m=1}^{M}\mathrm{Re}\left\{\hat{J}_{m,p}\right\}^2+\mathrm{Im}\left\{\hat{J}_{m,p}\right\}^2, \tag{3.47}$$

que l'on réalise en

$$\begin{array}{ll} 2MP+1 & \text{multiplications réelles et} \\ P(M-1) & \text{additions réelles.} \end{array} \tag{3.48}$$

En ajoutant la contribution des P comparaisons avec le seuil v_V, le total devient

$$\begin{array}{ll} 2MP+1 & \text{multiplications réelles et} \\ PM & \text{additions réelles.} \end{array} \tag{3.49}$$

3.3.2 Puissance de calcul en fonction des paramètres

Le nombre total d'opérations mathématiques est fonction des paramètres système M, L, L_Δ et P. Le rythme auquel elles sont effectuées dépend de L et de n_{STRF}. La mise en contexte des relations 3.46 à 3.49 avec des valeurs tangibles permet d'extraire la complexité relative de chacun des éléments de calcul de STAR et d'envisager la complexité globale du récepteur. Ces résultats viennent préciser les valeurs publiées par Jomphe (2004a).

Le tableau II résume les opérations de la région SP de STAR et indique leur importance relative en nombre d'additions et de multiplications réelles. Afin de mieux les situer au sein de l'algorithme, le résultat de chaque calcul est rapporté dans la colonne « résultat ». Le tableau III, pour sa part, résume le nombre de calculs réels de la région STRF de STAR. Finalement, le tableau IV indique les mêmes données pour les mécanismes de surveillance d'apparition et d'extinction des multi-trajets.

Tableau II

Nombre d'opérations par symbole pour la région SP

Opération	Résultat	Additions réelles	Multiplications réelles
Corrélation (3.10)	$\hat{\underline{Z}}_{n+1}$	$ML_\Delta(L\text{-}1)$	-
DFI contraint (3.9)	$\tilde{\underline{H}}_{n+1}$	$4ML_\Delta$	$4ML_\Delta$
DFI libre (3.9)	$\breve{\underline{H}}_{n+1}$	$4ML_\Delta$	$4ML_\Delta$
Combinaison (3.2)	$\hat{s}_{n+1}$	$2(ML_\Delta\text{-}1)$	$4ML_\Delta$
Puissance ()	$\hat{\Psi}^2_{n+1}$	1	3

Tableau III

Nombre d'opérations par symbole pour la région STRF

Opération	Résultat	Add. réelles	Mult. réelles
Séparation spatio-temporelle	$\hat{\boldsymbol{J}}_{n+1}$	$2MP(L_{\Delta}-1)$	$4MPL_{\Delta}$
Mise-à-jour temporelle	$\tilde{\boldsymbol{D}}_{n+1}$	$2ML_{\Delta}(3P-1)+2L_{\Delta}(M+P)$	$6MPL_{\Delta}$
Transformée de Fourier	$\tilde{\mathcal{D}}_{n+1}, \hat{\mathcal{D}}_{n}$	$(L_{\Delta}/4)\log_2(L_{\Delta})$	$(L_{\Delta}/4)\log_2(L_{\Delta})$
Détection de phase	$\hat{\tau}_{n+1}$	$2(L_{\Delta}-1)+1$	$L_{\Delta}+3$
Reconstruction	$\hat{\boldsymbol{H}}'_{n+1}$	$2ML_{\Delta}(P-1)$	$2MPL_{\Delta}$
Normalisation	$\hat{\boldsymbol{H}}_{n+1}$	$2(ML_{\Delta}-1)$	$4ML_{\Delta}$

Tableau IV

Nombre d'opérations par symbole pour la région PM

Opération	Résultat	Additions réelles	Multiplications réelles
Nouveaux trajets	N/A	ML_{Δ}	$2ML_{\Delta}+1$
Trajets actuels	N/A	MP	$2MP+1$

Ces relations permettent de dresser un portrait de l'importance de chacun des blocs logiques de STAR dans le contexte d'un système concret répondant à la norme 3GPP. Le tableau V rapporte le nombre exact d'additions et de multiplications réelles nécessaires à la réception d'un symbole et ce, pour chaque facteur d'étalement L valide.

Puisque le taux de symboles par seconde est directement relié à L, on parvient à établir, pour chaque situation, le nombre total d'opérations par secondes du système. Pour produire les données du tableau V, on considère un système à $M = 1$ antenne où sont évalués $P = 3$ multi-trajets et où la fenêtre de traitement réduite n'existe pas, c'est-à-dire $L = L_{\Delta}$ dans tous les cas, et l'estimation du canal s'effectue à tous les dix symboles (n_{STRF}=10).

Tableau V

Complexité algorithmique sans fenêtre de traitement réduite
($M = 1$, $P = 3$, $n_{STRF} = 10$, $L_\Delta = L$)

Facteur d'étalement	Additions réelles	Multiplications réelles	Total (10^6 ops/s)
$L = 256$	107 513	41 743	1 187,08
$L = 128$	35 833	19 343	686,37
$L = 64$	13 049	8 911	431,44
$L = 32$	5 113	4 079	299,44
$L = 16$	2 105	1 855	228,98
$L = 8$	889	839	189,43
$L = 4$	377	379	165,63

Comme prévu, le nombre de calculs requis croît avec le facteur d'étalement L. La complexité du traitement mathématique est inversement proportionnelle au débit reçu, une situation contre-intuitive. C'est le corrélateur qui en est responsable puisque le volume de calculs qu'il doit effectuer croît en fonction de L^2.

Le tableau VI reprend les mêmes hypothèses qu'au tableau V, à la seule différence qu'on y introduit la fenêtre de traitement réduite, c'est-à-dire $L_\Delta = \min(32, L)$.

Tableau VI

Complexité algorithmique avec fenêtre de traitement réduite
($M = 1$, $P = 3$, $n_{STRF} = 10$)

Facteur d'étalement	Fenêtre réduite	Additions réelles	Multiplications réelles	Total (10^6 ops/s)
$L = 256$	$L_\Delta = 32$	12 281	4 079	144,95
$L = 128$	$L_\Delta = 32$	8 185	4 079	167,02
$L = 64$	$L_\Delta = 32$	6 137	4 079	211,16
$L = 32$	$L_\Delta = 32$	5 113	4 079	299,44
$L = 16$	$L_\Delta = 16$	2 105	1 855	228,98
$L = 8$	$L_\Delta = 8$	889	839	189,43
$L = 4$	$L_\Delta = 4$	377	379	165,63

Les chiffres du tableau VI indiquent clairement l'effet du paramètre L_Δ sur la quantité de calculs. La présence de 4079 multiplications réelles pour les facteurs d'étalement $L = 256$ à $L = 32$ témoignent du fait que le corrélateur n'effectue aucune multiplication. Il doit néanmoins toujours effectuer la corrélation sur des séquences de longueur L, mais seulement L_Δ fois. La colonne du total d'opérations révèle que le cas le plus exigeant est celui où $L = 32$. Il s'agit du premier cas pour lequel la fenêtre de traitement réduite n'a plus d'impact.

La figure 15 montre l'impact de la fenêtre de traitement réduite et indique le cas $L = 32$ comme la situation la plus critique. Elle corrobore les résultats de Cheikhrouhou (2001) en se basant sur un nombre de calculs qui reflète la véritable architecture matérielle.

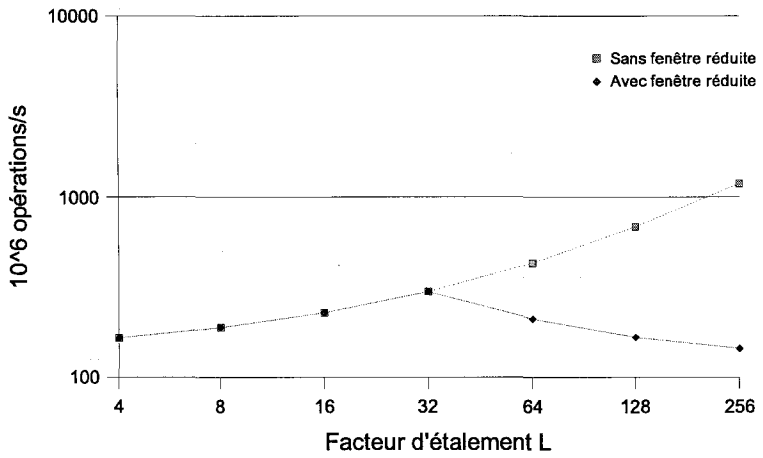


Figure 15 Effet du paramètre L_Δ sur la puissance de calcul totale

Le tableau VII explore l'impact du paramètre n_{STRF} sur la quantité totale de calculs pour l'estimation réduite du canal. Les données sont basées sur un facteur d'étalement $L = 32$ pour étudier le pire cas de figure. On considère un taux d'analyse du canal n_{STRF} variant sur une échelle quasi-logarithmique et le nombre d'opérations par seconde est indiqué pour chacune des trois sections de STAR.

Tableau VII

Nombre d'opération/s par usager en fonction de n_{STRF}
($M = 1$, $P = 3$, $n_{STRF} = 10$, $L = L_{\Delta} = 32$)

n_{STRF}	Mops/s SP	Mops/s STRF	Mops/s PM	Mops/s Total
1	211.44	878.76	1.28	1091.48
2	211.44	439.38	0.64	651.46
10	211.44	87.88	0.12	299.44
20	211.44	43.94	0.06	255.44
100	211.44	8.79	0.01	220.24

La région SP devant obligatoirement fonctionner au taux de symbole entrant, sa participation au nombre total d'opérations par seconde ne change pas en fonction de n_{STRF} et reste fixe à 211,44 Mops/s. Cette fraction du calcul global ne peut, en aucun cas, être affectée par la valeur de n_{STRF}. Dans le cas où $n_{STRF} = 1$, on constate que la région SP compte pour 20 % de la puissance totale de calcul, alors qu'avec $n_{STRF} = 10$, son apport atteint 80 %. L'augmentation de n_{STRF} dans les cas subséquents n'offre que des gains marginaux sur la puissance de calcul globale requise. Cela signifie qu'il y a peu d'avantages au chapitre de l'optimisation matérielle à pousser n_{STRF} au-delà de 10 puisqu'à

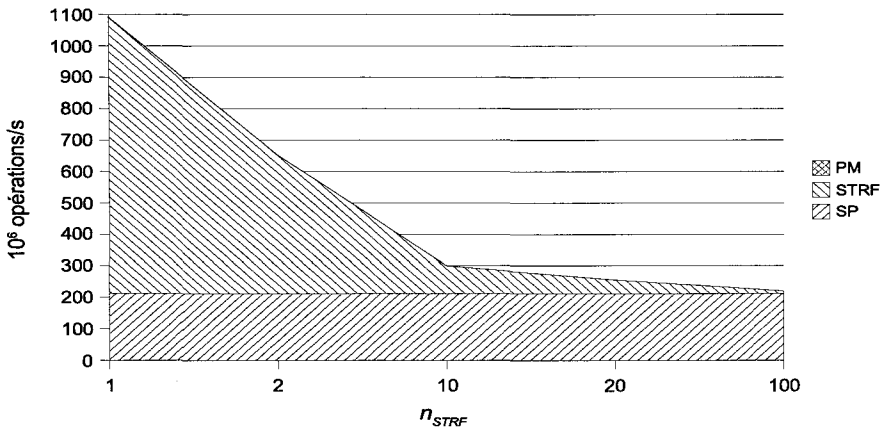


Figure 16 Puissance de calcul en fonction du paramètre n_{STRF}

ce stade, c'est le SP qui requiert le plus grand nombre de calculs. À l'inverse, réduire la contribution de la section STRF jusqu'à n_{STRF}= 10 n'affecte pas de façon substantielle les performances de réception et se révèle une optimisation intéressante. La contribution de la région PM demeure toujours négligeable, peu importe les paramètres du système.

Les résultats du tableau sont montrés à la figure 16 et témoignent de l'importance relative de chaque section de STAR en fonction du paramètre n_{STRF}. La contribution du PM est invisible étant donnée sa faible contribution au coût de complexité global.

3.4 Quantification

La représentation des nombres binaires peut s'effectuer en virgule flottante ou en virgule fixe. La représentation à virgule flottante, selon la norme IEEE-754, permet d'effectuer des calculs avec précision sur des opérandes d'un ordre de grandeur quelconque. La représentation est faite en format scientifique. Les calculs effectués sur de tels nombres requièrent la manipulation simultanée de la mantisse, de l'exposant et du signe pour chaque opération.

L'autre alternative est la représentation des nombres en virgule fixe. On doit alors connaître la plage dynamique de chaque variable pour assigner une quantification adéquate à la fois en amplitude (valeur maximale) et en précision (représentation de variations fines). Des excursions hors de ces plages dynamiques entraînent les conditions de débordement (« overflow ») ou de soupassement (« underflow ») et ont le potentiel d'amener le système dans un état indéterminé. La perte de flexibilité est compensée par la diminution de la complexité des opérations arithmétiques: une simple addition de deux nombres binaires en complément 2 permet de résoudre tous les cas d'addition et de multiplication.

Cheikhrouhou (2003) présente une étude inédite rapportant le comportement statistique de certaines opérandes cruciales de la version logicielle de STAR. Le tableau VIII rapporte ses conclusions.

Tableau VIII

Valeur moyenne et variance des opérandes

Opérande	E(min)	var(min)	E(max)
$\underline{Y}_n$	0,4271	0,0503	42,4713
$\underline{Z}_n$	0,1097	0,0033	2,1508
$\hat{H}_n$	0,0033	0,1017	2,1467
$\underline{\breve{H}}_n$	0,0192	$1{,}0106 \times 10^{-4}$	0,8128
$\hat{\boldsymbol{J}}_n$	0,2174	0,0106	0,8883
$\hat{\boldsymbol{D}}_n$	0,0014	$6{,}7449 \times 10^{-7}$	0,8956
$\delta\hat{\varphi}_{p,l,n}$	$5{,}5439 \times 10^{-4}$	$3{,}1950 \times 10^{-7}$	0,2082

3.4.1 Calcul à virgule fixe

La notation binaire à virgule fixe permet de représenter des nombres rationnels. La précision de ces fractions dépend du nombre de bits qui leur sont assignés et est implicitement une puissance de deux (négative ou positive). La notion de virgule est entièrement abstraite; on pourrait, avec 10 bits, représenter des nombres entiers de -512 à 511 en notation complément à 2, ou définir ce nombre comme des cent-vingt-huitièmes et

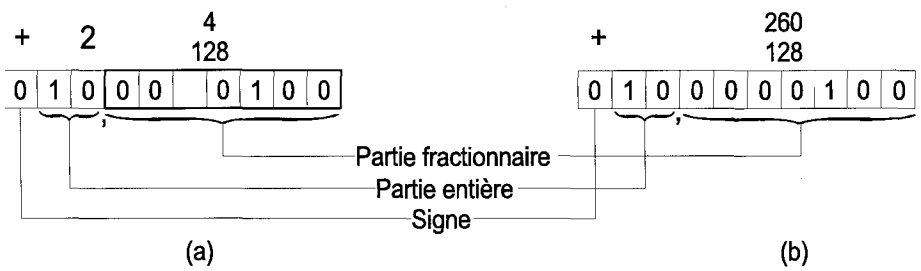


Figure 17 Représentation d'un nombre en notation à virgule fixe

alors représenter des nombres rationnels de -512 / 128 à 511 / 128 en pas de 1 / 128. On pourrait tout aussi bien représenter des quarts de -512 / 4 à 511 / 4 en incréments de 1 / 4. La figure 17 illustre le premier exemple avec les deux interprétations.

3.4.1.1 Impact des opérations arithmétiques sur la taille des opérandes

Une addition entre deux opérandes à virgule fixe répond aux mêmes règles que l'addition rationnelle, c'est-à-dire que le dénominateur des deux opérandes doit être identique. Pour adapter les deux opérandes, on définit d'abord la quantification désirée pour le résultat. On tronque ou augmente la taille des deux opérandes afin d'ajuster leur dénominateur.

La figure 18 montre le cas où l'on désire additionner une opérande exprimée en fractions de 2^{-7} à une seconde exprimée en 2^{-10}. Le résultat est requis en fractions de 2^{-8}. L'opérande de gauche est étendue d'un bit de poids faible, celle de droite est arrondie à quatre bits de poids faible et étendue d'un bit de poids fort.

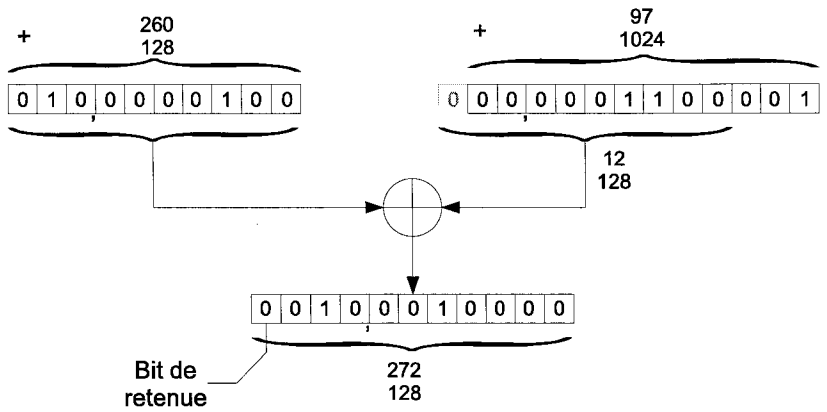


Figure 18 Addition en virgule fixe avec deux dénominateurs incompatibles

La multiplication suppose l'apparition de bits fractionnaires de poids plus faibles que n'en contiennent chacune des opérandes. Puisque ces bits pourraient ou non être requis au résultat, on les conserve lors du calcul et on arrondit plutôt le résultat pour se conformer à

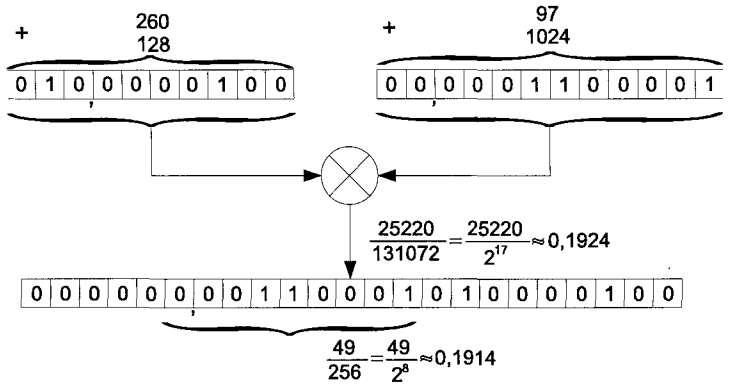


Figure 19 Multiplication en virgule fixe

la consigne. La figure 19 montre le cas d'une multiplication entre deux opérandes exprimées en 2^{-7} et 2^{-10} et la troncation subséquente pour l'obtention d'un résultat en 2^{-8}.

3.4.2 Quantification des matrices spatio-temporelles *H*

Les M rangées de $\underline{\hat{H}}_{n+1}$ doivent représenter la totalité de l'énergie à recouvrer sur les M antennes. Affes (1998) note que chaque rangée conserve une norme équivalant à $\sqrt{M}$, garantie par l'étape de normalisation, d'où la valeur maximale à représenter qui est unitaire. Le tableau VIII rapporte plutôt une valeur maximale de 2,8 étant donné l'absence de normalisation dans l'expression théorique de STAR. La partie entière doit néanmoins empêcher tout débordement et se voit attribuer 3 bits (incluant le bit de signe).

La colonne E(min) donne 0,0033 comme moyenne de toutes les valeurs minimales de $\underline{\hat{H}}_{n+1}$ observées à chaque nouveau symbole. Ceci indique qu'au moins 8 bits de fraction sont requis pour représenter $2^{-8} = 0{,}0039$. La variance associée est de 0,1017, alors qu'elle est plutôt $1{,}0106\ 10^{-4}$ dans le cas de $\underline{\breve{H}}_{n+1}$, ce qui indique une granularité potentielle dix fois plus petite que ce qu'offre une quantification sur 8 bits. On voudra plutôt représenter les éléments de $\underline{\breve{H}}_{n+1}$ en incréments successifs de $0{,}0039 / 10 \cong 0{,}0039 / 2^3$, nécessitant

trois bits de fraction supplémentaires. Afin de simplifier le système, on attribue une quantification uniformisée à toutes les opérandes $\boldsymbol{H}$, soit 3 bits pour la partie entière et 11 bits pour la partie fractionnaire, soit 14 bits au total.

3.4.3 Quantification de τ

L'opérande $\hat{\tau}_{n,p}$ sert à positionner chaque pic temporel d'un canal multi-trajets par rapport à l'ensemble du symbole n. Il est entendu que le nombre de pics ne dépassera pas $P_{MAX}= 5$ dans le cadre de ce premier effort de prototypage.

Les valeurs possibles de $\hat{\tau}_{n,p}$ sont positives et bornées entre 0 et $L_{\Delta}- \varepsilon$. Sachant que L_{Δ} est en tout temps inférieur ou égal à 32, on assigne 5 bits à la partie entière de $\hat{\tau}_{n,p}$. Le traitement de la partie fractionnaire dépend d'autres considérations du système. On sait, par exemple, qu'entre deux symboles consécutifs, le profil multi-trajets ne devrait changer que très peu. Ce changement est évidemment tributaire du taux de mise à jour n_{STRF} de la région STRF.

Affes (1996) rapporte un positionnement précis à $0{,}001T_C$ avec ce type d'identification. Si la partie fractionnaire est représentée sur 10 bits, la précision est contrainte à $1 / 1024T_C$. Puisque n_{STRF} est toujours supérieur à 3, on pourrait retrancher un certain nombre de bits pour refléter l'accroissement de la variation de $\hat{\tau}_{n,p}$ entre deux mises à jour.

3.4.4 Quantification de $\underline{X}, \underline{Y}, \hat{\boldsymbol{D}}, \hat{\boldsymbol{J}}$ et $\delta\hat{\varphi}$

Le tableau IX montre la quantification retenue pour chaque opérande mentionnée au tableau VIII. Le raisonnement est similaire à celui déjà présenté.

Ce choix de quantification est de nature préliminaire, et reflète les conclusions du rapport de Cheikhrouhou (2003). Il ne tient pas compte, par exemple, du bruit qu'ajouterait la quantification sur différents éléments du calcul dans STAR. À noter, cependant, que la

précision de 10 bits fractionnaires retenue pour $\hat{\tau}_{n+1}$ correspond de très près à la résolution temporelle maximale potentielle que peut atteindre STAR, soit 1/1000 T_C. Dans tous les autres cas, le nombre de bits est choisi pour dépasser légèrement la quantification que suggère l'analyse de Cheikhrouhou (2003).

Tableau IX

Quantification en nombre de bits des opérandes

Opérande	Partie entière	Partie fractionnaire	Total
$\underline{Y}_n$	6	3	9
$\underline{Z}_n$	3	8	11
$\hat{H}_n$	3	11	14
$\breve{\underline{H}}_n$	3	11	14
$\hat{\boldsymbol{J}}_n$	3	8	11
$\hat{\boldsymbol{D}}_n$	3	10	13
$\delta\hat{\varphi}_{p,l,n}$	1	12	13
$\hat{\tau}_{n+1}$	5	10	15

3.5 Choix de l'architecture

La séparation qui naît des discussions précédentes permet d'entrevoir une architecture modulaire au niveau système. Particulièrement, sachant que les régions définies (SP, STRF et PM) ont des capacités de calcul et des latences particulières, on peut présumer une station de base à 24 usagers, équipée du récepteur STAR, dans laquelle on retrouverait certain nombre de sections SP, effectuant le travail de décodage des messages, et quelques sections STRF qui seraient partagées entre tous les SP. Étant donné sa faible complexité mathématique, la région PM pourrait subvenir aux besoins de tous les STRF. La figure 20 illustre ce scénario.

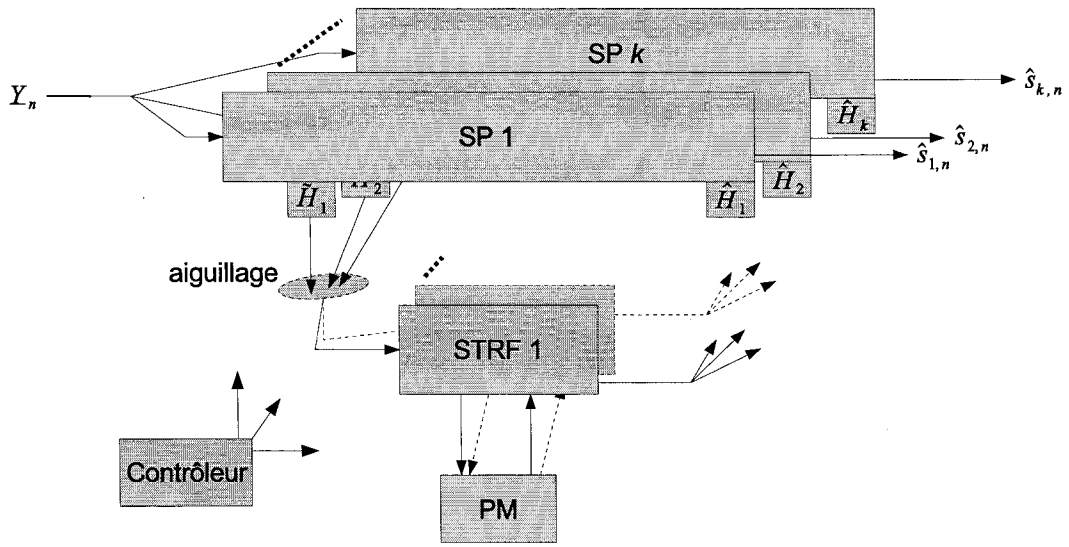


Figure 20 Topologie d'une station de base multi-usagers

Les données qui sont échangées entre les régions PM et STRF sont restreintes aux seules matrices $\tilde{H}_n$, $\breve{H}_n$ et $\hat{\tau}_n$ en entrée, et $\hat{\tau}_n'$ en sortie. Ces opérandes sont parmi les plus petitesdans le système, et on déduit que la bande passante nécessaire entre la région STRF et la région PM est ainsi minimisée.

Fonctions statistiques, décisions logiques et faible bande passante sont les raisons qui favorisent l'option codesign pour réaliser cette région. En plus de permettre un contrôle dynamique des algorithmes de contrôle par l'usager, le passage au logiciel permet de soulager l'implémentation matérielle d'une forte quantité de logique nécessaire à la réalisation d'opérations logiques pourtant simples pour un microprocesseur classique.

3.5.1 Utilisation de codesign

La taille des matrices $\tilde{H}_n$ et $\breve{H}_n$ dépend des paramètres L_Δ et M. Pour un système typique,

$$2\mathrm{L}_\Delta M \cdot \frac{3.84\times10^6}{L} \cdot \frac{1}{10} \times NB\ USAGERS \tag{3.50}$$

éléments doivent être transférés de STRF vers PM à chaque itération de calculs de gestion des trajets. Un système à $M = 4$ antennes ayant un facteur d'étalement de 256 nécessiterait une bande passante de $3{,}84 \times 10^5$ opérandes complexes/s. En direction inverse, dans le pire cas où des changements surviendraient au taux maximal (déterminé par les paramètres n_A et n_V, voir sections 3.3.1.13 et 3.3.1.14), la bande passante requise pour la mise à jour de $\hat{J}$ et $\hat{\tau}$ et se chiffrerait à

$$(MP+P)\cdot\frac{3.84\times10^6}{L}\cdot\frac{1}{10\cdot min(n_A,n_V)}\times NB\ USAGERS\,, \tag{3.51}$$

soit, inférieure à 2250 opérandes complexes/s.

3.5.2 Mise en ordre du traitement

La souplesse de traitement permise par les modules arithmétiques programmables entraîne une complexité supplémentaire au fonctionnement du système. À chaque instant où un nouvel ensemble de données est transmis de l'étage n à $n + 1$ du pipeline, ses dimensions doivent l'accompagner de quelque façon pour configurer correctement le module arithmétique du niveau $n + 1$. Ce problème se répète à chaque étage du pipeline, et les dimensions des opérandes matricielles doivent être manipulées en tenant compte du changement des paramètres M, L et P.

Le scénario est d'autant plus complexe lorsque se côtoient plusieurs usagers dans un pipeline STRF. Chacun fournit un débit constant d'observations de canal, lesquelles sont insérées en ordre à l'entrée du pipeline. Les dimensions de ces opérandes doivent être connues du STRF qui doit être configuré en conséquence à chaque nouvel ensemble de données sous peine d'anomalie fonctionnelle.

Deux solutions sont envisageables: la centralisation des opérations de configuration dans un contrôleur dédié, ou l'ajout d'une sorte d'en-tête à chaque bloc de données, annonçant sa taille au module arithmétique.

La seconde solution donne un net avantage au niveau du parallélisme. Inutile, dans cette optique, de prévoir un grand schéma de temporisation du système ni de prévoir l'allocation des ressources matérielles nécessaires pour effectuer les calculs. Cependant, chacun des processeurs doit pouvoir générer en sortie la taille de l'opérande qu'il vient de produire. De plus, il lui incombe d'indiquer le type de stockage utilisé par l'opérande (row-major ou column-major). Finalement, en guise de sécurité, il convient d'ajouter au lot un type quelconque d'étiquette permettant aux opérandes d'être aiguillées correctement dans un système vraisemblablement du type client-serveur.

Ces tâches doivent être assurées par chaque module afin qu'il soit réellement autonome. Les répercussions sur l'utilisation de ressources matérielles risquent alors d'être néfastes.

La première solution, au contraire, exige la réalisation d'un processeur dédié à la synchronisation de toutes les opérations du récepteur. Il s'agit d'un module tout à fait adapté à sa tâche, donc peu évolutif, mais vraisemblablement moins volumineux. C'est le rôle du module SCU (« STAR Control Unit »). Le rôle premier de ce contrôleur est de gérer les régions de traitement et leurs interactions. En plus de connaître les dimensions M, L et P, il devra en calculer les nouvelles combinaisons au fur et à mesure que celles-ci évoluent. Par exemple, pour une opérande de dimension $M \times P$, le nombre d'éléments MP doit être calculé lors du changement d'une des valeurs M ou P.

Étant donné son rôle central, le contrôleur SCU doit répondre à des requêtes externes lui donnant ordre de modifier certains paramètres. Des sémaphores de plusieurs types sont requises pour établir une communication avec différents acteurs; elles seront explicitées au chapitre 4.

3.6 Conclusion

Ce chapitre révèle les bases de l'architecture qu'utilisera le récepteur STAR. Il reprend l'expression théorique de STAR donnée au chapitre précédent et l'analyse, équation par

équation, afin de déterminer la puissance globale de calcul requise. Ces résultats sont ensuite ajustés à l'aide de la fenêtre de traitement réduite, L_{Δ}, et des taux de traitement réduits, n_{STRF} et n_{PM}. On constate, par groupement des opérations mathématiques, que la gestion des multi-trajets contribue très peu à la complexité globale et peut être réalisée en retrait du récepteur. On souligne aussi que la modularité des régions de traitement facilite leur duplication pour supporter des configurations de récepteur multi-antennes et multi-usagers. La quantification des opérandes est ensuite considérée, reprenant les résultats d'une analyse statistique inédite du modèle de référence. La discussion se termine en évoquant différents modèles de gestion plausibles pour coordonner le fonctionnement de STAR. Un contrôleur centralisé est préféré à un modèle distribué, plus exigeant sur l'aiguillage des données.

Cette architecture est le fondement de la réalisation matérielle du récepteur. Étant donnée sa grande envergure, deux chapitres s'y intéresseront. Les calculs mathématiques, prenant jusqu'à maintenant la forme d'étages génériques de pipeline, seront développés en détails au chapitre 4. Le chapitre 5 enchaînera avec l'aspect codesign du système et la réalisation en langage C du gestionnaire de trajet.

CHAPITRE 4

RÉALISATION MATÉRIELLE

4.1 Introduction

La structure générale du récepteur STAR étant fixée et les requis en termes de puissance de calcul, connus, les modules matériels peuvent être réalisés pour répondre aux spécifications propres à leur rôle dans le pipeline de régularisation multi-usagers.

Ce chapitre est dédié aux détails techniques de la réalisation matérielle de ces modules. Il débute par une présentation de la technologie retenue, ainsi que des plate-formes de développement disponibles pour le récepteur et ses composantes périphériques. La section 4.3 explique ensuite la répartition, dans ces plate-formes, des trois domaines de calcul définis au chapitre 3.

Le travail de conception débute réellement à la section 4.4 avec la définition des bus de données et de contrôle du pipeline à étages programmables. Ses étages arithmétiques sont définis, quant à eux, à la section 4.5. Bien qu'ils soient tous présentés en détails en termes de quantification, registres et fonctionnement, on porte une attention particulière au module de multiplication matricielle dans le but d'effectuer au chapitre 6 une analyse de dimensionnement plus poussée. Le chapitre se termine par la présentation du contrôleur interne de STAR qui effectue la reconfiguration périodique des étages du pipeline, et agissant comme mandataire des niveaux hiérarchiques supérieurs.

4.2 Technologie

La réalisation du prototype exige que le livrable ait un minimum de relation avec la plate-forme de développement sous-jacente. C'est pour cette raison qu'un maximum de fonctionnalités se retrouvent à l'intérieur d'un FPGA (« Field Programmable Gate Array »,

ou réseau programmable de logique universelle). Cette technologie est idéale pour le prototypage d'architectures parallèles comme celle élaborée au chapitre 3.

Même si les plate-formes cibles utilisent la technologie Xilinx, on préfère produire un code source VHDL générique utilisant l'inférence de ressources physiques afin de laisser au logiciel de synthèse le soin de choisir les ressources dédiées de manière optimale, peu importe le fournisseur de FPGA retenu. Certains indices sont néanmoins soufflés à l'outil de synthèse afin de le guider pour certaines sections cruciales du système. En adhérant à cette philosophie, le code VHDL peut être réutilisé avec d'autres technologies.

4.2.1 Plate-forme de développement

L'effort de réalisation de STAR outrepasse le prototype décrit ci-après. Deux plate-formes de développement sont déjà identifiées, soulignant l'importance de produire un code source VHDL au niveau d'abstraction élevé.

4.2.2 Plate-forme d'acquisition ICS daqPC

La société ICS propose un PC (désormais appelé « hôte ») contenant un module de numérisation et de pré-traitement du signal ainsi qu'un module de développement FPGA générique de la société Transtech. Ce dernier est doté d'un FPGA Virtex II 6000 de Xilinx, d'interfaces FPDP et PCI ainsi que de 256 Mbits de mémoire dynamique. Le module d'acquisition, pour sa part, est équipé de quatre convertisseurs analogique à numérique (CAN) de 105 Msps, et d'interfaces PCI et FPDP. Une carte émettrice identique dotée de convertisseurs numérique à analogique (CNA) est aussi disponible.

4.2.3 Plate-forme de radio logicielle (SDMP) d'ISR Technologies

Le partenaire industriel, ISR Technologies, propose une plate-forme de radio logicielle propriétaire dotée d'un FPGA, d'un CAN et d'un CNA ainsi que les ressources

périphériques nécessaires à son fonctionnement. Une librairie de modules VHDL couvre déjà tous les aspects du fonctionnement de cette radio.

ISR Technologies a développé son expertise avec le Virtex II 6000 et le Virtex II Pro 40. Ce dernier intègre à sa maille logique deux microprocesseurs PowerPC 405 d'IBM, le rendant particulièrement attrayant pour le développement de systèmes en codesign. En outre, des modules de surveillance et de déverminage sont déjà disponibles.

4.3 Répartition des sous-systèmes de STAR

Le chapitre 3 explicitait le partitionnement de STAR en trois régions ayant chacune des besoins particuliers. Les sections suivantes établissent une stratégie d'allocation des ressources physiques à ces trois régions.

4.3.1 Logique programmable

La région SP de l'algorithme (section 3.2.2.1) est asservie à la cadence d'arrivée T_S des symboles. Aucun dépassement (« overflow ») ni soupassement (« underflow ») n'y est admissible. La section 3.3.2 mentionne que le nombre de calculs nécessaires à la réception s'accroît en fonction de L_Δ, ce qui, malgré la fenêtre de traitement réduite, devient rapidement contraignant dans un contexte multi-usagers. On y souligne cependant la possibilité de paralléliser ces calculs.

De façon similaire, la région pipelinée STRF commande un volume important de calculs par unité de temps, malgré la réduction de complexité imputable à la fenêtre de traitement réduite L_Δ (section 3.3.1.2) et au taux de traitement réduit n_{STRF} (section 3.2.2.4).

La séparation logique de ces deux régions permet de réaliser un récepteur multi-usagers. On peut les considérer comme des boîtes noires que l'on peut assembler en nombre quelconque dans un FPGA afin de combler les besoins en bande passante combinée et en

performances de réception. Le parallélisme massif et le nombre variable de ces régions suggèrent une réalisation dans la maille de logique programmable du FPGA.

4.3.2 Micrologiciel

La région PM (« Path Management », section 3.2.2.3) participe en proportion infime à la complexité totale de STAR et ne fonctionne qu'à un faible taux comparé aux autres régions. De plus, un seul PM peut subvenir aux besoins de plusieurs STRF. Considérant sa nature logique (en opposition à arithmétique), le PM répond à tous les critères d'une réalisation logicielle. Pour le concrétiser, on utilise un microprocesseur 32 bits Microblaze.

Ce microprocesseur est entièrement constitué des composantes de la maille logique du FPGA. Il correspond au modèle Harvard (données et instructions disposées dans deux mémoires séparées et concurrentes) et est doté d'une unité de traitement arithmétique interne pipelinée utilisant, au choix, un multiplicateur dédié du FPGA. Il est peu efficace en termes de consommation énergétique et en utilisation de la surface, mais sa polyvalence et sa petite taille en font un candidat idéal lorsque la plate-forme n'est pas munie d'un microprocesseur dédié. Son appartenance à la zone de logique programmable du FPGA lui confère une latence minimale que ne pourrait atteindre l'hôte.

Les outils de développement qui l'accompagnent permettent de rédiger le micrologiciel du Microblaze en langage C. Le niveau d'abstraction qu'offre cette méthodologie est suffisamment élevé pour permettre la réutilisation quasi-totale du même programme dans un microprocesseur dédié PowerPC 405. C'est la situation à laquelle STAR devra faire face lors de son passage à la plate-forme SDMP d'ISR Technologies.

4.3.3 Contrôleur

Le fonctionnement des trois régions de traitement est régi par un module de contrôle spécialisé qui gère le flot de données à l'intérieur du récepteur. Il doit configurer les

différents modules pour le traitement successif des différents usagers du récepteur et effectue la séquence de démarrage. Puisque tout délai dans ces actions entraînerait une dégradation des performances de réception, le contrôleur doit bénéficier d'un accès privilégié aux régions SP (« symbol path », section 3.2.2.1) et STRF (« structure fitting », à la section 3.2.2.2). Le Microblaze étant isolé de ces dernières par un bus de communication (section 5.2), le contrôleur est son mandataire au sein de la maille de logique programmable du FPGA.

4.4 Interfaces normalisées

La normalisation des branchements entre les étages de pipeline du récepteur tente de promouvoir la réutilisation des modules. Jomphe (2004b) donne un aperçu de ces interfaces minimales qui n'ont nullement la prétention d'être optimales, mais dont la simplicité se traduit par une faible utilisation de ressources logiques et un temps d'accès minimal. L'objectif est double: permettre la modification rapide du pipeline afin de réaliser en peu de temps de nouvelles versions de l'algorithme, et assurer une configuration générique des blocs de traitement, peu importe leur nombre et leur fonction.

On définit donc deux parties à cette interface: une première permet de relier la sortie d'un module à une mémoire et cette mémoire à un module subséquent (interface de données), et une seconde permet l'acheminement des commandes de configuration (interface de unidirectionnelle de configuration MBUS).

4.4.1 Interfaces de données

La structure en pipeline suppose la disponibilité entière et immédiate de toutes les opérandes (jusqu'à trois) d'une opération avant qu'elle ne débute. C'est à l'aide de mémoires dédiées (« BlockRAM ») que les opérandes sont stockées à l'intérieur de la maille logique du FPGA. De taille et de profondeur programmables suivant l'outil de synthèse, ces mémoires sont dotées de ports d'accès doubles; chacun peut manipuler de

façon indépendante le même ensemble de cases mémoire. L'usage typique ne requiert qu'un port d'écriture (le port A) et un port de lecture (port B), éliminant tout recours à l'arbitrage de bus et toute situation de contention en écriture (« write after write »). Ce type d'accès en particulier est disponible à la fois chez Xilinx et Altera; il ne porte donc pas atteinte à la consigne générale d'abstraction de la technologie du FPGA.

Le code source VHDL fixe *a priori* la taille des bus d'adresse et de données de ces mémoires par les variables BRAM_A_W et BRAM_D_W. Cette dernière fixe la taille combinée des parties réelle et imaginaire qui sont en réalité concaténées dans une seule case mémoire (la partie imaginaire utilisant les bits de poids fort). En pratique, la mémoire aura plutôt une largeur du double de BRAM_D_W bits. Ces tailles ne sont réellement traduites en ressources physiques qu'une fois l'outil de synthèse exécuté.

L'accès en lecture de ces mémoires est immédiat: l'adresse est livrée au bus d'adresse à l'instant 0 et les données apparaissent sur les bus DRe et DIm au début de la période d'horloge suivante. L'accès en écriture est similaire mais son délai n'est d'aucune conséquence puisque la transaction est unidirectionnelle: l'adresse et les données DRe et DIm sont acheminées vers la mémoire en même temps qu'un signal d'écriture We, sans vérification ultérieure.

La figure 21 montre un étage typique de pipeline où l'on retrouve un bloc de traitement quelconque ainsi que trois opérandes en entrée et une mémoire où est stocké le résultat.

4.4.1.1 MBUS

La configuration des modules arithmétiques s'effectue par l'entremise d'un bus adresses/données unidirectionnel où l'on retrouve un maître et plusieurs esclaves. Les bus d'adresse et de données sont, respectivement, de taille IMBUS_A_W et IMBUS_D_W, deux variables globales du code source VHDL. Dans sa forme actuelle, MBUS ne requiert que 4 bits d'adresse, conférant à chaque esclave une plage mémoire de 16 registres.

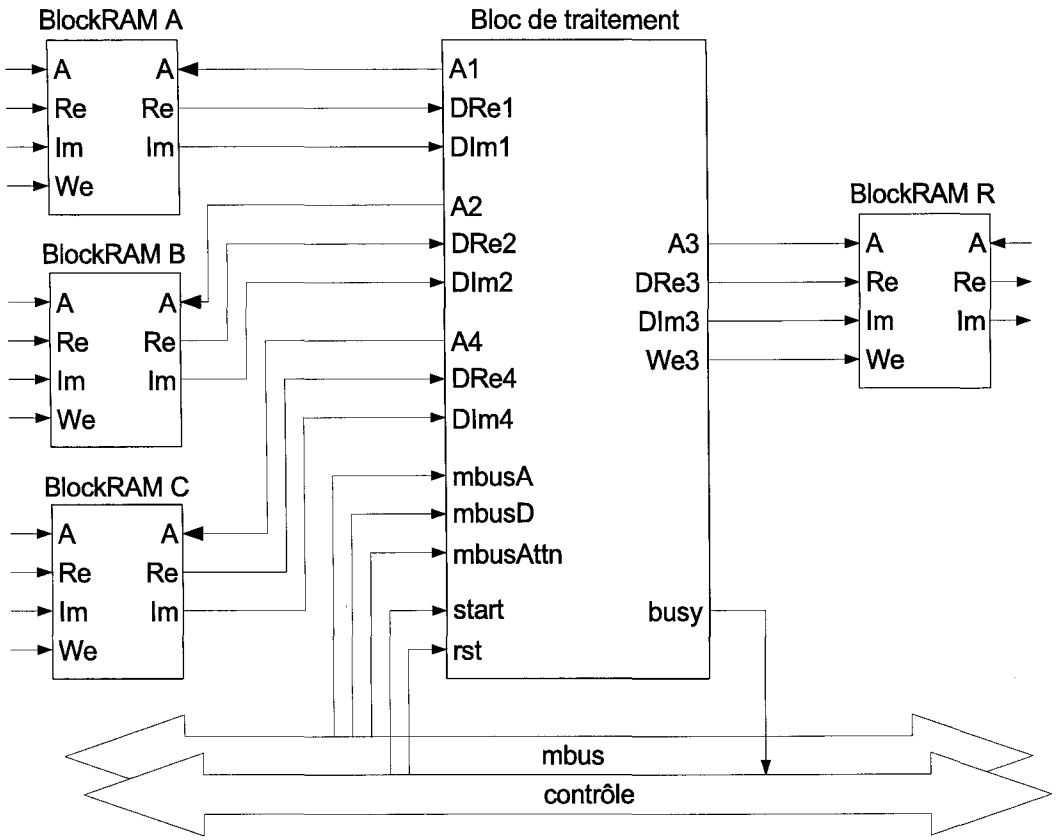


Figure 21 Branchement d'un étage de pipeline

La différentiation entre les esclaves est faite par l'assertion d'un signal mbusAttn dédié à chacun. Cette technique est clairement moins évolutive qu'un décodage par plages d'adresses, mais convient à cette étape préliminaire du prototype où le nombre de modules est encore minimal et où le paradigme de contrôle global est appelé à changer.

La configuration d'un module débute après que le maître ait confirmé son inactivité par son signal *busy*. Les consignes de configuration peuvent alors se succéder au rythme d'une par cycle d'horloge et ce, indépendamment du module auquel elles sont adressées. Aucune confirmation n'est émise par le module configuré, mais seule une consigne de démarrage pourrait corrompre une salve de configuration. Afin d'éliminer cette possibilité, le maître MBUS est conçu pour gérer les signaux de démarrage par états mutuellement exclusifs.

Toute contention est donc éliminée et la configuration unidirectionnelle, moins lourde à réaliser, demeure sans risques.

4.5 Modules de base

Les sous-sections suivantes présentent chacune la réalisation physique d'un module de calcul dédié à la résolution des problèmes développés à la section 3.3.1.

4.5.1 Multiplicateur pipeliné basicMac

L'élément fondamental du récepteur est le multiplicateur vectoriel pipeliné pour opérandes complexes basicMac. Même s'il a précédemment été question de décomposer toutes les opérations en opérandes réelles, l'avantage d'utiliser une architecture matérielle adaptée est justement d'effectuer ces calculs en parallèle. Avec basicMac, les produits partiels de la multiplication complexe sont complétés par quatre multiplicateurs dédiés en un seul cycle d'horloge.

4.5.1.1 Architecture

Présents dans toutes les plate-formes considérées, les multiplicateurs dédiés permettent la multiplication de deux nombres binaires de 18 bits en représentation signée. On sait, de manière empirique, qu'une telle multiplication requiert 5 ns dans un FPGA Virtex II de catégorie « -5 », soit la moitié de la période d'horloge de 10 ns choisie. Il est donc possible d'effectuer les deux additions subséquentes, longues elles aussi de 5 ns, dans la même période d'horloge. Essentiellement, ceci permet de solutionner l'équation 3.19 en un seul cycle d'horloge.

Puisque basicMac doit effectuer des multiplications matricielles, il est opportun de lui adjoindre un accumulateur permettant de solutionner l'équation 3.17. Ayant déjà épuisé le budget de 10 ns à l'étage 1, ce nouvel additionneur doit être rajouté dans un second étage. On peut dès lors définir un basicMac à l'architecture pipelinée, permettant au premier

étage (produits partiels) de calculer un nouveau résultat alors que le second étage cumule les résultats des cycles d'horloge précédents.

Puisque l'additionneur de l'étage 2 parvient à résoudre l'addition en 5 ns, il est possible de lui ajouter un second additionneur. Ce nouvel élément se charge de l'addition des éléments d'une matrice ***C*** (section 3.3.1.4). Cet ajout augmente évidemment la taille de basicMac et, de surcroît, ne trouve utilité que pour la mise à jour de la matrice de support temporel (section 3.3.1.8). Dans toutes les autres situations, les signaux d'entrée supplémentaires sont simplement branchés à la masse (zéro logique) et sont ensuite retirés par l'outil de synthèse. Cette stratégie n'est donc jamais sous-optimale et apporte le bénéfice d'une addition supplémentaire permettant de résoudre le cas générique ***AB + C*** à l'aide d'un seul et même module. La figure 22 montre la forme retenue pour basicMac.

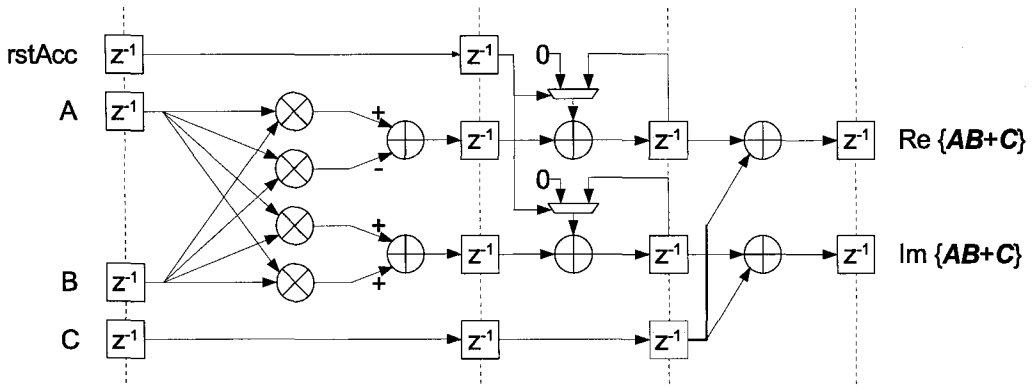


Figure 22 Topologie du sous-module basicMac

4.5.1.2 Fonctionnement en pipeline

Le sous-module basicMac n'effectue en soit qu'un produit vectoriel; l'acheminement des données depuis et vers l'extérieur doit être pris en charge par un module qui l'accompagne. C'est à celui-ci que revient la tâche d'établir la séquence d'adresses capable de récupérer chaque élément de ***A***, ***B*** et ***C*** dans l'ordre adéquat et de saisir les éléments du résultat aux

bons instants. Il doit connaître l'ordre de stockage des éléments, la taille des opérandes, ainsi que la latence qu'introduit le pipeline interne du basicMac.

En soit, basicMac fonctionne uniquement lorsque son signal d'activation *CE* (« chip enable », non montré) est actif. Sa remise à zéro s'effectue par l'entremise du signal *rstAcc* qui accompagne chaque « vague » de nouvelles données et qui annule le contenu du sommateur du second étage lorsque leurs produits partiels y parviennent. Ceci permet le calcul ininterrompu de plusieurs produits vectoriels (plusieurs éléments de ***R***).

4.5.1.3 Gestion de la virgule

Puisque le travail s'effectue sur des opérandes à virgule fixe, les différents étages du module basicMac doivent gérer les translations de virgule. Ces déterminations sont faites avant synthèse à l'aide de paramètres génériques (« GENERIC ») dans le code source VHDL. Les paramètres en jeux sont présentés au tableau X.

Tableau X

Paramètres pré-synthèse du sous-module basicMac

Paramètre	Fonction
OP_A_W	Nombre de bits de la partie entière de l'opérande ***A***
OP_A_W_FRAC	Nombre de bits de la partie fractionnaire de l'opérande ***A***
OP_B_W	Nombre de bits de la partie entière de l'opérande ***B***
OP_B_W_FRAC	Nombre de bits de la partie fractionnaire de l'opérande ***B***
OP_C_W	Nombre de bits de la partie entière de l'opérande ***C***
OP_C_W_FRAC	Nombre de bits de la partie fractionnaire de l'opérande ***C***
RESULT_W	Nombre de bits de la partie entière du résultat
RESULT_W_FRAC	Nombre de bits de la partie fractionnaire du résultat
MUL_TO_ACC	Nombre de bits de poids faible à concaténer aux produits partiels
DEL_TO_ACC	Nombre de bits de poids faible à concaténer à l'opérande ***C***
INT_OVER	Nombre de bits de poids fort à concaténer à l'accumulateur

En plus de la taille de chaque opérande (y compris le résultat), trois paramètres assurent la versatilité de basicMac. Les produits partiels entre ***A*** et ***B*** engendrent un résultat complet; aucune troncation n'est effectuée. S'ils étaient de taille OP_A_W + OP_A_W_FRAC et OP_B_W + OP_B_W_FRAC, respectivement, les bascules à l'entrée de l'étage 2 auront comme taille la somme de ces 4 paramètres. La virgule occupera alors la position OP_A_W_FRAC + OP_B_W_FRAC.

Au niveau de l'accumulateur, deux ajustements sont possibles. En premier lieu, le dépassement peut être évité en allouant davantage de bits entiers que n'en nécessitent les produits partiels. Le paramètre INT_OVER indique ce nombre de bits. Sa valeur exacte dépend du contexte dans lequel est utilisé basicMac.

Le second ajustement concerne la position de la virgule des produits partiels comparé à celle de l'opérande ***C***. La partie fractionnaire de l'accumulateur doit convenir à la fois à OP_C_W_FRAC et à la partie fractionnaire des produits de l'étage 1. En clair, on pose

$$\begin{aligned} \text{DEL\_TO\_ACC} &= \min(0, \mathit{MUL\_FRAC} - \mathit{OP\_C\_W\_FRAC}) \text{ et} \\ \text{MUL\_TO\_ACC} &= \min(0, \mathit{OP\_C\_W\_FRAC} - \mathit{MUL\_FRAC}), \end{aligned} \tag{4.1}$$

où MUL_FRAC est la somme de OP_A_W_FRAC et OP_B_W_FRAC. DEL_TO_ACC et MUL_TO_ACC sont mutuellement exclusifs; en aucun cas ne devraient-ils avoir tous deux une valeur non-nulle.

Finalement, le résultat peut récupérer une plage quelconque de l'accumulateur par le truchement des paramètres RESULT_W et RESULT_W_FRAC.

4.5.2 Multiplicateur matriciel matrixMult

L'opération la plus courante (et la plus exigeante en complexité selon la section 3.3.2) est la multiplication matricielle. Le module matrixMult la concrétise et offre la liberté d'utiliser des opérandes aux dimensions et aux types de stockage différents.

Le module matrixMult pilote le fonctionnement de basicMac au sens établi à la section 4.5.1.2. La reconstruction des séquences d'adresses s'effectue par une machine à états finis doublée d'une pile de registres. La nature adaptative de STAR exige de chaque instance de matrixMult la capacité de traiter des opérandes aux dimensions variables d'une itération à la suivante. Ceci entraîne un changement dans la séquence d'adresses que génère la machine à états à partir de ses registres. C'est essentiellement la raison pour laquelle l'interface MBUS est présente. Les registres qui lui sont accessibles sont détaillés au tableau XVI de l'annexe 1. La topologie du module matrixMult est illustrée à la figure 23.

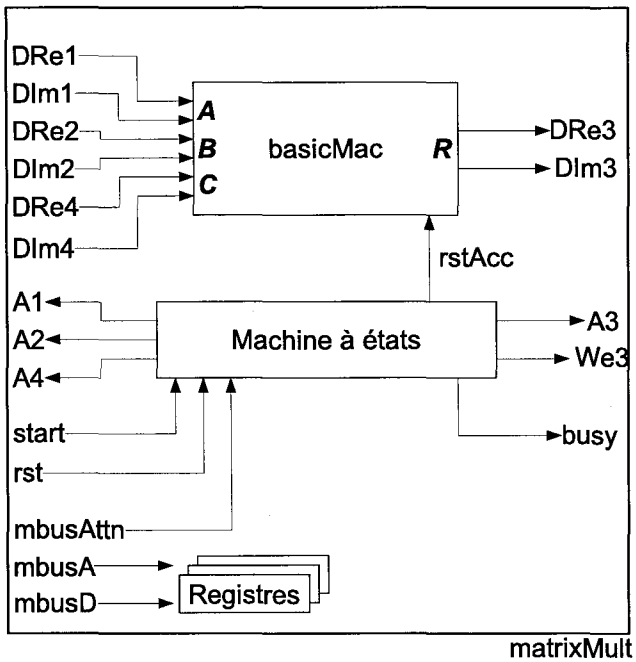


Figure 23 Topologie du module matrixMult

4.5.2.1 Ordonnancement des opérandes

L'ordre dans lequel sont stockés les éléments de chaque matrice affecte le déroulement du produit matriciel. On en distingue deux types: « row-major » et « column-major ».

Le stockage « row-major » consiste à placer dans des cases mémoire voisines chacun des éléments d'une matrice $\boldsymbol{A}$ de dimensions $H \times W$ ($\boldsymbol{A}_{H,W}$) en débutant par l'élément (1,1) et en parcourant chaque ligne de gauche à droite, ligne après ligne. L'agencement des éléments en mémoire serait alors le suivant:

$$a_{1,1} \quad a_{1,2} \quad \ldots \quad a_{1,W} \quad a_{2,1} \quad a_{2,2} \quad \ldots \quad a_{2,W} \quad a_{3,1} \quad \ldots \quad a_{H,W}.$$

L'ordonnancement « column-major » réunit plutôt les éléments de la matrice colonne par colonne. On lit alors chaque colonne de la matrice de haut en bas, colonne après colonne, depuis la position (1,1). On aurait alors la séquence d'éléments suivante dans les cases mémoire voisines:

$$a_{1,1} \quad a_{2,1} \quad \ldots \quad a_{H,1} \quad a_{1,2} \quad a_{2,2} \quad \ldots \quad a_{H,2} \quad \ldots \quad a_{1,3} \quad \ldots \quad a_{H,W}.$$

Les deux types de stockage sont complémentaires; une matrice stockée en « row major » lue comme s'il s'agissait d'une « column major » devient sa propre transposée.

Le choix de l'un ou l'autre type de stockage peut sembler inconséquent, mais le type de calcul effectué sur les opérandes peut conduire à l'un ou l'autre, en fonction des ordonnancements précédents. Le produit scalaire et le produit point à point, par exemple, n'affectent pas l'ordre de stockage. Le produit matriciel faisant intervenir une transposée, cependant, mène à des situations plus complexes.

4.5.2.2 Jeux de compteurs

Pour effectuer un produit matriciel où les opérandes à l'entrée sont d'un ordonnancement quelconque, la machine à états doit disposer de plusieurs registres lui permettant de faire évoluer les adresses des opérandes $\boldsymbol{A}$, $\boldsymbol{B}$ et $\boldsymbol{C}$ dans un ordre adéquat, selon l'ordonnancement de chacune. Ces registres sont en fait des valeurs d'arrêt et d'incrément pour différents compteurs présents dans la machine à états.

Le compteur regCountOuter gère le déroulement du calcul de la matrice $\boldsymbol{R}$. MatrixMult détecte la fin de la multiplication matricielle une fois sa valeur ayant atteint celle du

registre regStopOuter. C'est donc le nombre d'éléments de la matrice ***R*** qui doit figurer dans ce registre.

Le compteur regCountInner, quant à lui, accompagne chacun des produits vectoriels internes requis pour le calcul de chaque élément de ***R***. Lorsque sa valeur atteint celle figurant dans le registre regStopInner, matrixMult termine le calcul de l'élément regCountOuter de ***R*** et débute le calcul de l'élément suivant à l'aide du sous-module basicMac interne. Cette opération est la représentation physique de l'équation 3.17, une somme sur x éléments où x est la dimension commune aux deux matrices ***A*** et ***B***. La valeur que doit prendre regStopInner est donc x, corrigé par la latence de basicMac.

L'ordre dans lequel sont récupérés les éléments de ***A*** et ***B*** est plus complexe. Les registres regBram1Incr et regBram2Incr servent à déterminer l'espacement entre les éléments consécutifs des rangées ou des colonnes de ***A*** et ***B***, respectivement, selon leur ordonnancement. Si, par exemple, ***A*** est stockée en « row-major » et que l'on désire récupérer les éléments d'une rangée l'un après l'autre, on pose regBram1Incr = 1, indiquant que l'adresse mémoire doit augmenter de 1 pour chaque élément consécutif de ***A***. Si, par contre, on voulait récupérer les éléments de la première colonne de ***A*** (toujours stockée en « row-major »), on devrait incrémenter l'adresse d'un facteur plus élevé, correspondant à la largeur de la matrice ***A***. On devrait alors insérer dans regBram1Incr la largeur de ***A***. Les constatations sont les mêmes pour l'opérande ***B*** mais concernent plutôt le registre regBram2Incr.

De plus, lorsque survient la fin d'un produit vectoriel, matrixMult doit repositionner l'adresse mémoire de ***A*** à la rangée (ou colonne) suivante, et celle de ***B*** à la même rangée (ou colonne). On doit donc lui signifier quelle distance, en cases mémoire, sépare la prochaine rangée (colonne) du début de la précédente.

Pour un stockage « row-major », deux rangées débutent à un incrément équivalent à la largeur de la matrice, alors que deux colonnes débutent plutôt à un incrément unitaire. La

figure 24 illustre le lien entre ces agencements de dimensions matricielles et les adresses mémoire.

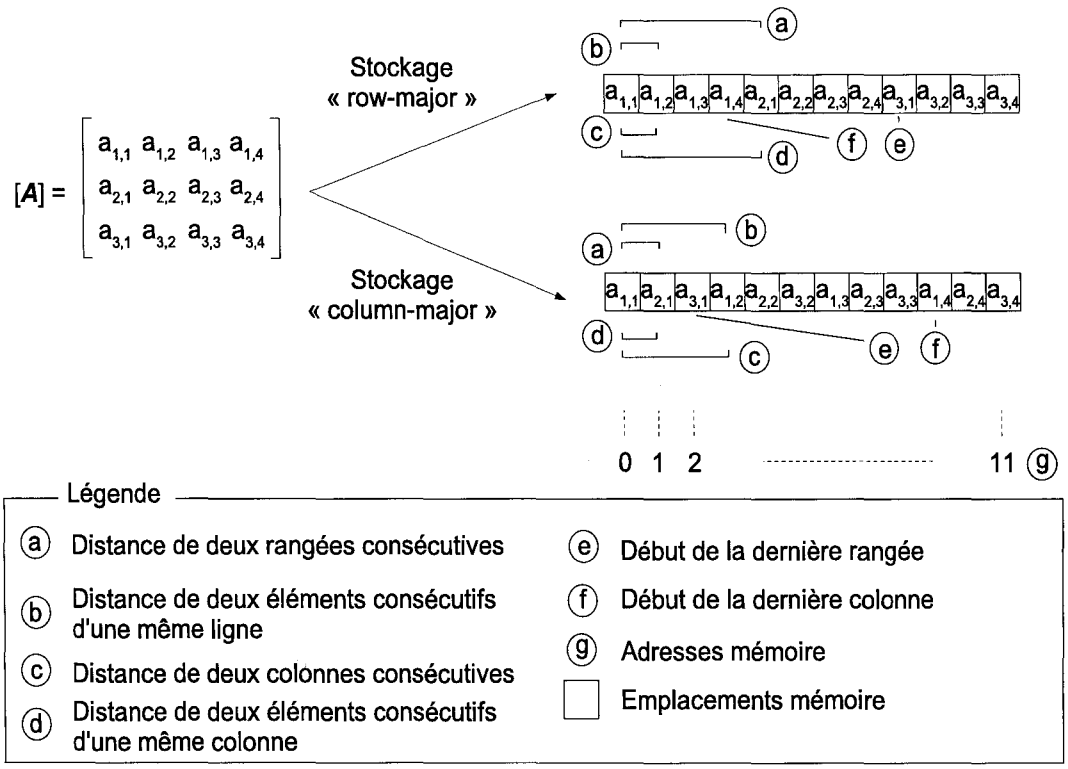


Figure 24 Adressage pour différents types de stockage matriciel

Finalement, matrixMult détermine les états limites qui indiquent un changement de ligne (colonne) dans ***R***. Ceci se traduit en pratique par l'incrément d'une rangée (colonne) de ***A*** alors que la rangée (colonne) de ***B*** retrouve sa position initiale. Ce séquencement d'adresses permet de balayer les deux matrices pour effectuer la multiplication. Les registres regBram1RstMask et regBram2RstMask permettent d'identifier ces conditions limites et de redémarrer l'adressage à l'endroit approprié.

4.5.2.3 Coefficients multiplicatifs

La multiplication de ***AB*** par un scalaire réel est effectuée suivant les valeurs figurant dans les registres regCoeff0 et regCoeff1. Pour effectuer une multiplication réelle sur le produit

vectoriel engendrant chaque élément de ***R***, on règle ces deux registres à la même valeur. Le dédoublement permet de multiplier les parties réelle et imaginaire par un scalaire différent afin d'introduire un déséquilibre des deux dimensions. La conjugaison du résultat, en multipliant par 1 et -1, en est un exemple. La quantification de regCoeff0 et regCoeff1 est la même que celle du résultat ***R***.

4.5.3 Multiplicateur scalaire CONJMULT

Pour satisfaire le cas où une multiplication par un scalaire ne peut être effectuée au détour d'une multiplication matricielle, le module CONJMULT peut répondre à la tâche. Exempt d'un basicMac, il offre une économie de ressources matérielles comparé au module matrixMult et présente une latence réduite puisqu'aucun produit vectoriel n'est calculé. La latence de CONJMULT est directement liée au nombre d'éléments de son opérande d'entrée.

Le nombre de registres est donc considérablement réduit. Hormis l'adresse de base contenue dans regBram1Base, seules les parties réelle (regCoeff0) et imaginaire (regCoeff1) du coefficient sont requises, en plus du nombre d'éléments dans l'opérande en entrée (regLength). La quantification du coefficient est ajustée à l'aide de paramètres pré-synthèse et s'effectue de manière analogue à celle de basicMac. Une configuration illégale serait détectée par l'outil de synthèse.

4.5.3.1 Topologie

CONJMULT est constitué de manière tout à fait analogue à matrixMult (figure 29) mis à part l'absence de basicMac et des ports B et C. Les registres de configuration sont identiques à ceux de matrixMult (tableau XVI de l'annexe 1).

4.5.4 Corrélateur continu, DESP

La mise à jour du PCM nécessite le calcul de L_Δ corrélations de L éléments chacune. Au rythme d'une opération par cycle d'horloge, le temps total requis dépasse T_S, la période d'un symbole entrant, lorsque $L \geqslant 32$. Cette situation est inadmissible et commande l'élaboration d'un modèle de calcul parallèle.

La topologie choisie pour le corrélateur se subdivise en trois sections: un registre à décalage agile manipulant le(s) code(s) d'étalement, un engin d'intégration modulaire et parallèle, ainsi qu'un arbitre régissant le flot de données des deux autres sections et répondant à l'interface normalisée des modules pipelinés du récepteur STAR.

4.5.4.1 Fonctionnement

On suppose en premier lieu la présence d'un registre à décalage unique, mais d'un ou plusieurs sous-blocs intégrateurs. La taille du registre à décalage est suffisante afin de contenir le code utilisé. La transmission BPSK (ou DBPSK) en phase (branche I seulement) requiert un code d'étalement binaire de longueur L. Puisque les facteurs d'étalement sont variables, on prévoit un registre de taille 256 où le dernier élément est dit « élément de code actuel ».

Le couplage du code de canal (« channelization code ») au code d'étalement (« scrambling code ») entraîne deux cas. Lorsque le code d'étalement est long (« long scrambling code »), c'est-à-dire constitué d'une séquence pseudo-aléatoire périodique de longueur largement plus élevée que L, la séquence combinée doit être régénérée *in situ* à l'aide du code de canal connu et d'un registre à rétroactions multiples de topologie connue. Lorsque le code d'étalement est court (de longueur 256), la séquence combinée est pré-calculée et insérée dans le registre à décalage. Cette dernière alternative est beaucoup plus simple à réaliser étant donné l'absence de synchronisation globale du prototype; c'est donc elle qui est retenue en première itération.

La partie supérieure de la figure 25 illustre la structure d'un code d'étalement de longueur 256 avec un code de canal de longueur 32. La séquence de 256 éléments est périodique et contenue dans le registre à décalage. La synchronisation de la séquence reçue avec le code peut donc s'étendre durant huit symboles.

La corrélation de la séquence de bribes reçue avec le contenu du registre à décalage selon la relation 2.9 implique que les mêmes 32 éléments du code doivent être combinés aux bribes i à i+62, que les 32 éléments de code suivants soient combinés avec les bribes i + 32 à i+94, et ainsi de suite. Ce mécanisme est schématisé à la figure 25. Les formes ondulées montrent l'envergure d'une multiplication point à point entre une section du code et une section de bribes, et les formes triangulaires inférieures correspondantes dénotent la sommation et le stockage à l'index montré du PCM. La combinaison d'un triangle et d'une forme ondulée représente donc un produit vectoriel sur L éléments.

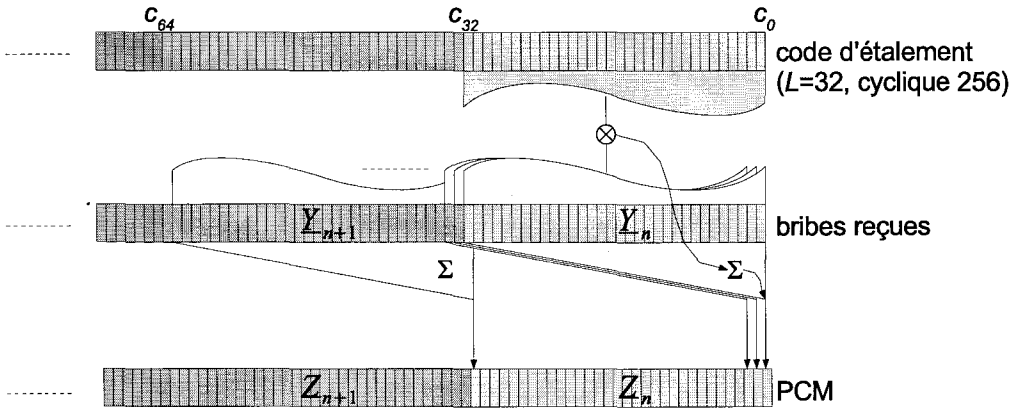


Figure 25 Corrélation par blocs sur un code d'étalement court

La résolution instantanée d'un PCM nécessiterait un corrélateur entièrement parallèle qui puisse accéder instantanément à un bloc de L bribes, impliquant un registre de taille L par 11 (selon la quantification de $\hat{\underline{Z}}$ établie à la section 3.4), un modèle trop coûteux en ressources matérielles. On doit donc rechercher un compromis entre un modèle de calcul entièrement parallèle et une version purement séquentielle. Comme le code d'étalement est disponible à raison d'un élément par cycle d'horloge, il est possible de calculer

quelques éléments du PCM en parallèle, chacun à l'aide d'un sous-module d'intégration. Chacun de ces sous-modules peut alors calculer un élément du PCM de manière autonome par addition séquentielle.

On pose le nombre de ces sous-blocs à k, chacun capable de calculer un élément du PCM en L cycles d'horloge. À cette latence s'ajoute (k-1) cycles d'horloges pour remplir le pipeline d'intégrateurs et (k-1) cycles pour le vider. Afin de respecter la contrainte temporelle, l'inéquation

$$\frac{L_\Delta}{k}\left(L+2\left(k-1\right)\right)\leqslant\frac{L\cdot f_{CK}}{f_{CHIP}} \tag{4.2}$$

doit être respectée, où f_{CHIP} est 3,84 Mcps, f_{CK} est fixée à 100 MHz et L et L_Δ varient selon l'état du système. En résolvant pour les valeurs permises de L, on trouve que $k=2$ étages sont nécessaires. Le schéma du corrélateur correspondant est présenté à la figure 26.

La machine à états finis permet de contrôler l'insertion de bribes dans le pipeline à k étages et l'extraction des éléments du PCM. Elle effectue aussi la rotation des éléments de code suivant l'évolution du calcul. Elle permet trois modes de fonctionnement du registre à décalage agile que l'on retrouve à la figure 27. Le premier mode, dit « mode d'initialisation », permet à une source externe d'insérer séquentiellement un nouveau code combiné dans le registre à décalage. Ce mode est utilisé lors de l'initialisation du corrélateur ou simplement lorsque le code de l'utilisateur doit changer. Il est aussi prévu pour l'insertion en continu d'un code d'étalement long.

Le second mode d'opération prévaut lors de la création des L_Δ - k premiers éléments du PCM pour tout symbole donné. Ce mode est dit « mode de recyclage » puisque seuls les L premiers éléments du code sont en rotation tandis que les 256-L derniers éléments demeurent en place. Puisque L éléments du code sont requis pour le calcul d'un élément du PCM, ils retrouvent leur position originale une fois les k éléments du pipeline calculés.

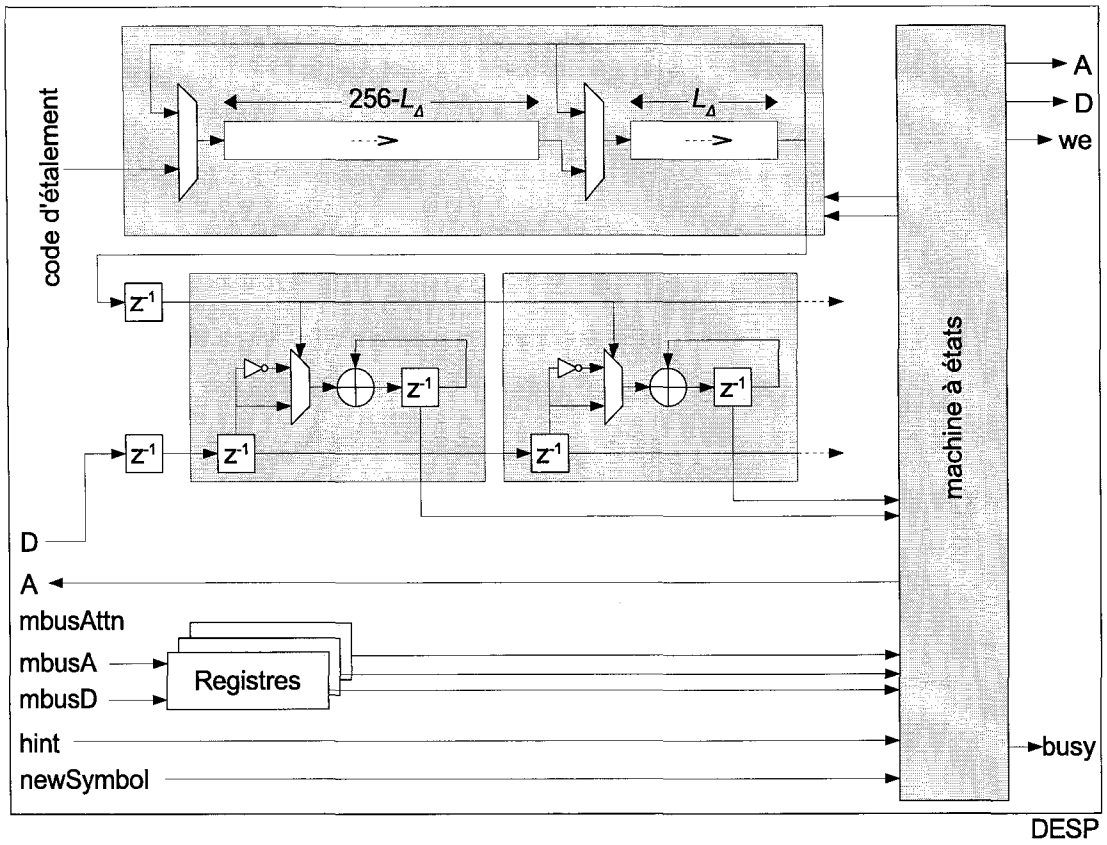


Figure 26 Schéma du corrélateur

La position de ré-insertion du code varie selon le facteur d'étalement *L*, et est ajustable de manière dynamique. Afin d'alléger la figure, seul un cas y est représenté.

Deux compteurs imbriqués sont utilisés pour régler le fonctionnement de la machine à états: un premier, regCountInner, suit l'évolution des sommations internes pour chaque élément du PCM et sert de boucle itérative interne. Le second, regCountOuter, compte le nombre d'éléments du PCM déjà calculés. Il constitue la boucle externe.

Les événements déclencheurs dans la machine à états surviennent par comparaison entre ces deux compteurs et différents registres programmables par MBUS. Le démarrage s'effectue avec une valeur négative dans regCountInner qui permet le remplissage du pipeline à *k* étages. L'extraction des *k* éléments de PCM se fait lorsque ce même compteur

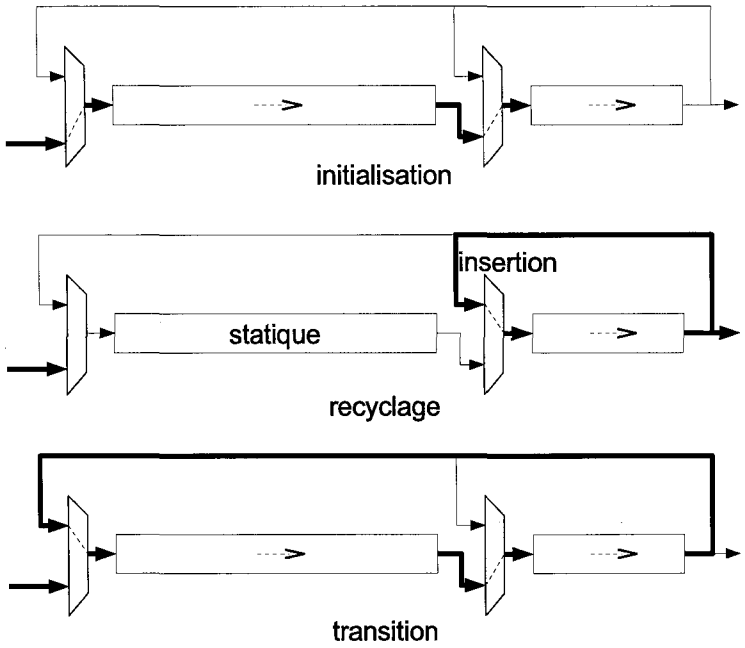


Figure 27 Évolution du code d'étalement

atteint la valeur regInnerStop. La rotation des éléments de code dans le registre à décalage ne s'effectue que lorsque regCountInner est positif et inférieur à regStopInner.

La boucle externe permet de déterminer les adresses des bribes entrantes et des éléments du PCM dans les mémoires correspondantes. Elle permet aussi de commuter les modes d'opération du registre à décalage agile. Lorsque regCountOuter atteint la valeur contenue dans regRecycle (soit $L_{\Delta} - k - 1$), la machine à états fait basculer le registre à décalage en mode transition. Lorsqu'il atteint regStopOuter, elle retrouve le mode de recyclage, mais interrompt la rotation des éléments de codes jusqu'à l'arrivée de la prochaine salve de bribes. Le détail des registres est donné au tableau XVII de l'annexe 1.

4.5.5 Decision Feedback Identifier

Les blocs d'identification contraint et libre du canal, CDFI et UDFI, respectivement, réalisent trois étapes distinctes d'un produit point à point entre le PCM, $\underline{Z}_{n+1}$, et le vecteur

d'observation spatio-temporel correspondant, $\underline{\hat{H}}_n$ ou $\underline{\breve{H}}_n$. L'utilisation de basicMac serait injustifiée puisque aucune sommation interne n'est requise. Le module DFI est donc équipé d'un pipeline de calcul spécialisé.

La section 3.3.1.3 décompose l'opération de DFI en quatre étapes: deux multiplications et deux additions intercalées. Ces étapes sont illustrées à la figure 28.

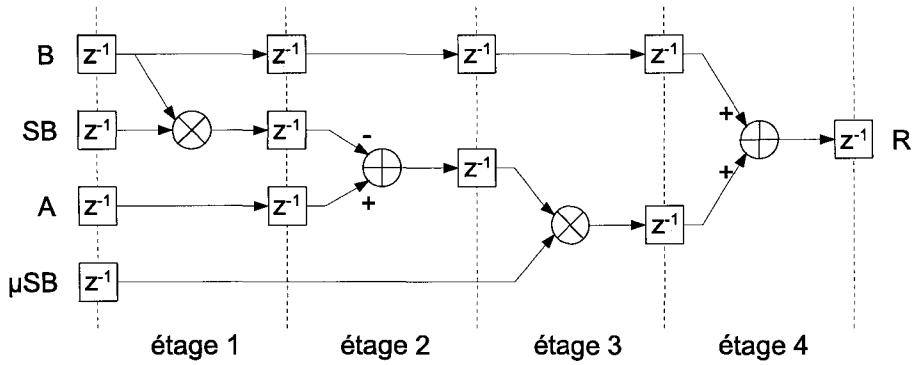


Figure 28 Pipeline interne du DFI

4.5.5.1 Interfaces

Les ports d'entrée et de sortie du module DFI sont semblables à ceux du bloc matrixMult (figure 23). On ajoute cependant un port d'entrée nommé SBVMxx dont les différents signaux émulent le comportement d'un registre à écriture seule. Accessible uniquement lorsque le module se trouve au repos, ce registre contient la valeur du dernier symbole décodé par le combinateur (sections 2.2.2 et 3.3.1.5) et dicte la polarité de la prochaine correction qu'appliquera le DFI.

4.5.5.2 Fonctionnement interne

Tout comme le module matrixMult (section 4.5.2), le fonctionnement du module DFI repose sur des compteurs. Puisque l'opération vectorielle s'effectue point à point, le type

de stockage n'a aucune incidence en autant que chaque opérande utilise le même. La seule information nécessaire est le nombre d'éléments des deux opérandes en entrée, identique à celui du résultat.

Afin de conserver la trace du pipeline interne du DFI (figure 28), le compteur d'adresse qui guide la mémoire des résultats (port 3) démarre à une adresse négative, correspondant à la profondeur du pipeline. Ce compteur comporte un bit de poids fort supplémentaire qui est simplement utilisé comme signal de masquage pour le signal d'écriture we3. Effectué par simple logique combinatoire, ce masquage simplifie la machine à états du DFI qui est ainsi exempte de vérifier les conditions d'écriture dans la mémoire 3. On chiffre l'économie de ressources à ~ 120 LUTs, et la fréquence d'opération maximale passe ainsi de moins de 100 MHz à près de 180 MHz.

4.5.5.3 Contexte d'utilisation

Les équations 2.10 et 2.11, quoique similaires, laissent entendre que l'une des opérandes à l'entrée du DFI est, dans un cas, la même que le résultat précédent, et dans l'autre cas, une opérande différente. Comme le démontre la figure 29 (a), une seule et unique mémoire à double port est utilisée pour contenir $\boldsymbol{B}$ et $\boldsymbol{R}$ ($\breve{\underline{H}}_n$ et $\breve{\underline{H}}_{n+1}$) dans le cas UDFI. Le case du CDFI voit plutôt son opérande $\boldsymbol{B}$ rapportée dans une forme contrainte par le STRF, comme l'indique la figure 29 (b). Les registres MBUS du module DFI sont énumérés dans le tableau XVIII de l'annexe 1.

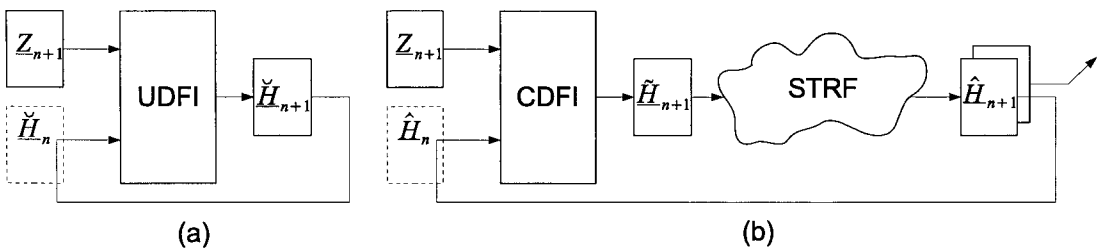


Figure 29 Contexte d'utilisation du DFI (a) libre et (b) contraint

4.5.6 Module de transformée de Fourier, FFTW

Le module FFTW fait appel au module R2²PC développé par Grandmaison (2005). En plus d'être optimisé en surface et en vitesse de calcul, il permet la reconfiguration dynamique des tailles de transformées. Ce module doit appliquer l'équation 2.14 sur $\hat{\boldsymbol{D}}_n$ et $\tilde{\boldsymbol{D}}_{n+1}$ qui sont de taille P par L_Δ. Puisque cette dernière dimension peut prendre indifféremment les valeurs 4, 8, 16 et 32, l'agilité qu'offre la R2²PC de s'y adapter répond admirablement bien à la situation. Cependant, le module R2²PC est livré avec des interfaces destinées à un usage général, et son intégration dans le pipeline STRF nécessite la conception d'une couche d'adaptation.

4.5.6.1 Particularités de la R2²PC

Les transactions de données avec le module R2²PC sont exemptes d'adressage puisque l'accès s'effectue par un lien sériel. L'insertion des éléments 1 à n du vecteur s'effectue au rythme d'un par cycle d'horloge. Le premier élément est accompagné d'un signal de contrôle qui évolue parallèlement dans le bloc et indique un premier résultat en sortie.

Le résultat, lui aussi livré sous forme sérielle, parvient en ordre à bits renversés (« bit-reversed ») dû à la nature de la TFR, comme le montre la figure 30.

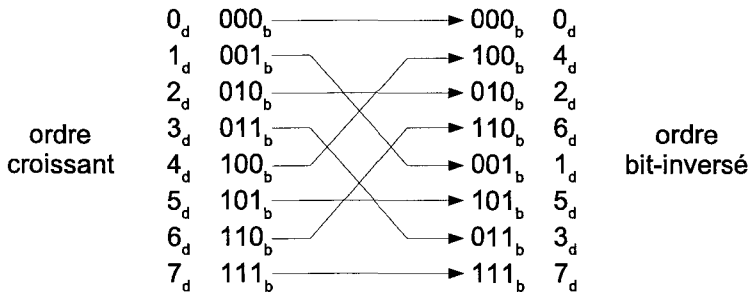


Figure 30 Sortie du module FFTW en ordre « bit-inversé »

En revanche, le module R2²PC peut traiter des données entrant en ordre bit-inversé et produire un résultat en ordre. Pour le récepteur STAR, l'ordre des éléments à la sortie importe peu puisqu'ils sont destinés à une sommation pondérée. On choisit donc de conserver les éléments en ordre à l'entrée, et en bit-inversé à la sortie.

L'architecture interne de ce module est pipelinée. Elle occasionne une latence qui dépend de la taille du vecteur à traiter; le cas d'une transformée de taille $n = 32$ se traduit par une latence de 9 + 32 cycles d'horloge. Tout comme le module basicMac, les vecteurs d'entrée peuvent s'enchaîner sans pause s'ils sont accompagnés du signal de démarrage startIn. La vidange du pipeline entre chaque TFR est donc inutile.

La quantification des opérandes s'effectue dans le code source avant la synthèse. Elle est symétrique pour les parties réelle et imaginaire. La quantification du résultat dépend de l'architecture interne de la R2²PC, mais assure qu'aucun débordement n'est possible. Ceci diminue cependant la précision du résultat dans le cas courant où le signal d'entrée ressemble davantage à du bruit qu'à un signal localisé en fréquence (par exemple, un sinus). Pour le récepteur STAR, la quantification en entrée est de 10 bits, et celle en sortie, de 12 bits, desquels seuls les 6 bits de poids faible sont utilisés.

4.5.6.2 Couche hiérarchique supérieure, FFTW

L'accès aux données provenant de l'étage de pipeline précédent s'effectue de manière analogue aux autres modules du récepteur. Différents registres, de compteurs et de masques d'arrêt permettent au module FFTW de retrouver en mémoire les L_Δ éléments consécutifs des P réponses à l'impulsion de $\tilde{\boldsymbol{D}}_{n+1}$ et $\hat{\boldsymbol{D}}_n$. La figure montre la topologie du bloc FFTW.

De surcroît, les P vecteurs de $\tilde{\boldsymbol{D}}_{n+1}$ et $\hat{\boldsymbol{D}}_n$ sont toujours stockés « row-major » en mémoire. On peut donc considérer qu'il n'y a en fait qu'un seul vecteur de PL_Δ éléments sur lequel

effectuer P transformées consécutives de longueur L_Δ. Nul besoin, donc, pour la couche d'adaptation, de faire une distinction sur le début et la fin de chacun des P vecteurs.

La présence de quatre ports permet simplement le calcul d'une TFR à partir de deux sources. Les multiplexeurs sélectionnent les ports 1 et 3 lors du calcul de $\tilde{\boldsymbol{D}}_{n+1}$, ou les ports 2 et 4 pour $\hat{\boldsymbol{D}}_n$.

On prévoit les registres regLength pour contenir le nombre d'éléments à traiter, et regFFTSize, pour conserver le mot de configuration à transmettre au module R2²PC pour la taille de la transformée. À l'instar des autres modules pipelinés, le registre regBram1Base contient l'adresse de base des vecteurs à traiter. On considère cette adresse identique pour les quatre mémoires: $\tilde{D}_{n+1}$, $\hat{D}_n$, et leurs transformées correspondantes, $\tilde{\mathcal{D}}_{n+1}$ et $\hat{\mathcal{D}}_n$. Le tableau XIX de l'annexe 1 donne le détail de ces registres.

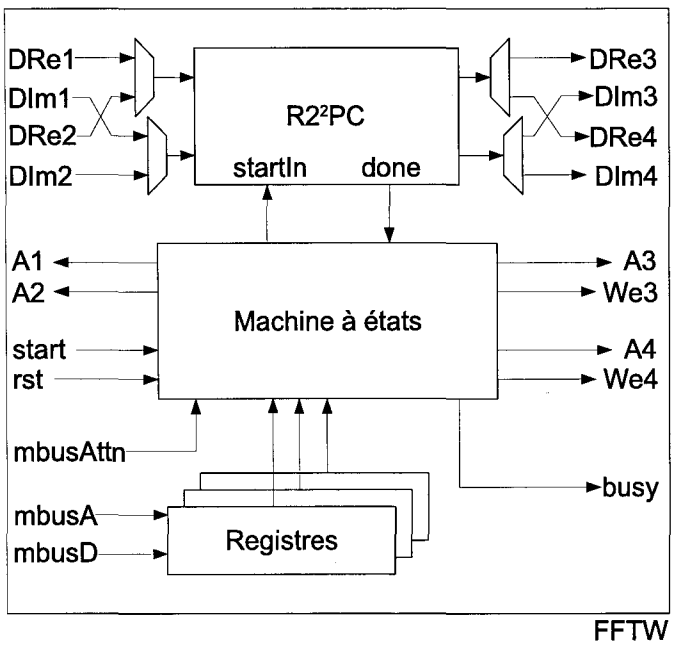


Figure 31 Topologie du bloc FFTW

4.5.7 Détection du délai, TDU

La détection du délai de chaque multi-trajet est à certains égards différente des autres étages du pipeline. Le module TDU ne sait calculer que le déplacement relatif de chaque multi-trajet par rapport à sa position précédente; il ignore leur position absolue. On préfère d'ailleurs lui interdire cette agilité et en déléguer la responsabilité à un acteur externe capable de positionner la fenêtre de traitement réduite. Les interfaces du module sont du type usuel, c'est-à-dire deux ports d'entrée pour mémoires ($\hat{\mathcal{D}}_n$ et $\tilde{\mathcal{D}}_{n+1}$) et un troisième pour le vecteur $\hat{\tau}_n$. Un port de sortie transporte le vecteur $\hat{\tau}_{n+1}$.

4.5.7.1 Fonctionnement

La régression linéaire (sections 2.2.6 et 3.3.1.11) est effectuée par l'agencement d'un pipeline et d'une machine à états finis. La manipulation d'éléments complexes ne survient qu'au premier étage du pipeline pour la multiplication de $\hat{\mathcal{D}}_n$ et $\tilde{\mathcal{D}}^*_{n+1}$. Les opérations subséquentes n'utilisent que des éléments réels.

La conversion cartésienne à polaire constitue le second étage du pipeline. Elle emploie une quantification asymétrique pour conserver la précision de 0,001 T_C dans la région d'opération. En considérant le cercle trigonométrique à la figure , on constate que la phase résiduelle de la multiplication de $\hat{\mathcal{D}}_n$ et $\tilde{\mathcal{D}}^*_{n+1}$ demeure, lorsque l'algorithme converge, dans un cône centré à 0°. L'amplitude de ce cône dépend du taux de fonctionnement du STRF, lequel dépend en contrepartie de n_{STRF}.

De surcroît, la partie imaginaire a, dans cette région, un effet prédominant sur l'angle correspondant. On peut donc, pour préciser davantage la position cartésienne dans la région ombragée de la figure , tronquer quelques bits de poids faible à la partie réelle tout en gardant intacts ceux de la partie imaginaire. Ceci engendre le quadrillage asymétrique

représenté sur le cercle. La motivation première de cette troncation est la minimisation de la taille de la mémoire de correspondance pour $\tan^{-1}(\cdot)$, ROM_ATAN.

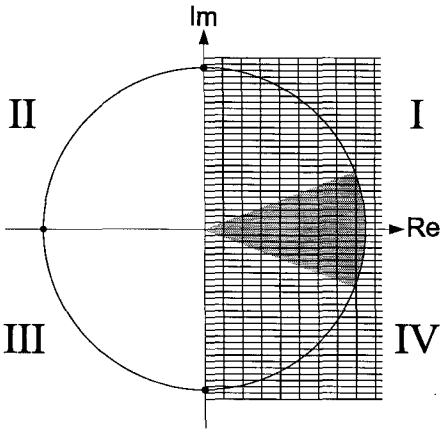


Figure 32 Quantification asymétrique de la conversion tangente inverse

Le cas des quadrants II et III n'est pas couvert, sinon que par les points $\pm \pi/2$ et π. Ceci donne lieu à une correction brusque dans la régression linéaire, incompatible avec un positionnement précis. Puisque cette région correspond à un trajet qui serait vraisemblablement mal estimé, on n'y accorde que peu d'importance; l'algorithme PM (section 3.3.1.14) le fera éventuellement disparaître.

L'étape suivante réutilise le vecteur $\delta\hat{\varphi}_{n+1}$ comme entrée de deux sommateurs: l'un pondéré, l'autre simple. La pondération des éléments $\delta\hat{\varphi}_{n+1}$ se fait avec leur index (moins un), mais puisque leur ordre est en bit-inversé (section 4.5.6), il est nécessaire d'effectuer une assignation bit à bit permettant d'inverser artificiellement les bits d'un compteur à croissance unitaire monotone. L'index bit-inversé est alors utilisé directement par le sommateur pondéré.

Les étapes subséquentes de la machine à états sont illustrées à la figure 33. Elles se résument à des multiplications de scalaires réels et par les facteurs A, B et C qui correspondent à la nomenclature employée à la section 3.3.1.11.

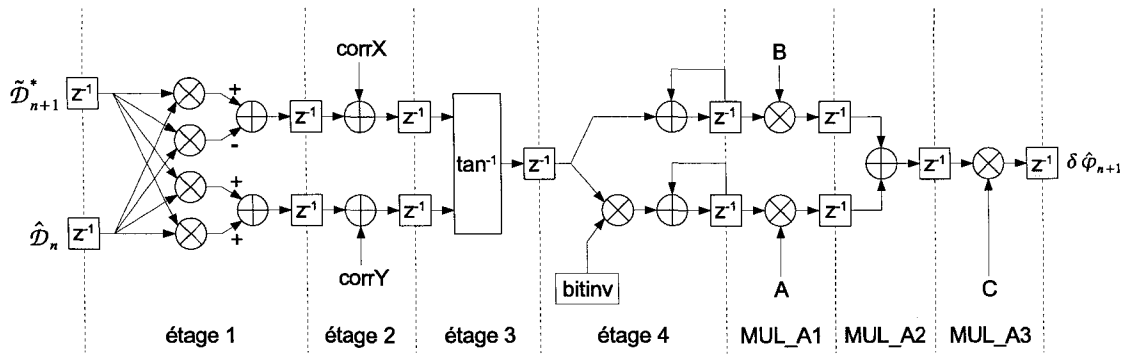


Figure 33 Déroulement du calcul dans le module TDU

Le calcul évoqué ci-haut se répète à P reprises et chaque $\Delta\hat{\tau}_{p,n+1}$ calculé est ajouté à la valeur de $\hat{\tau}_{p,n}$ correspondante provenant du port mémoire 4. Les registres MBUS du module TDU sont précisés dans le tableau XX, à l'annexe 1.

4.5.8 Module de récupération d'horloge de données, DCDT

L'absence d'asservissement des horloges dans le récepteur entraîne une dérive du PCM avec l'écoulement du temps. Les sources d'incertitude sont multiples, mais l'horloge de données du transmetteur et les variations de la fréquence porteuse sont généralement les plus importantes.

Le symptôme d'une dérive trop importante des horloges est le glissement groupé des délais $\hat{\tau}_{n+1}$ vers l'une ou l'autre des extrémités de la fenêtre de traitement réduite. On préfère conserver la totalité des coefficients non-nuls des P trajets connus à l'intérieur de cette fenêtre afin de récupérer un maximum de puissance du signal transmis.

4.5.8.1 Fonctionnement

Le module DCDT ("Data Clock Drift Tracker", ou module suiveur de dérive d'horloge de données) introduit le concept de barrières haute et basse (« highGate » et « lowGate ») entre lesquelles les P délais sont confinés. Puisque $h_{RRC,p}(t-\hat{\tau}_p)$ est constituée de 16

coefficients (section 4.5.9), l'index de sa position centrale doit être compris entre 8 et 23 inclusivement. Ces deux valeurs sont les barrières basse et haute.

Cette version préliminaire du module ne surveille que le premier trajet (p = 1). Lorsque sa position s'écarte de la plage fixée par les barrières, une correction est effectuée pour le centrer sur une cible choisie. La correction est communiquée au module de contrôle SCU (section 4.5.12) qui, en retour, détermine le moment opportun où en faire part au corrélateur afin que celui-ci puisse compenser le délai de la fenêtre du PCM de manière adéquate. En parallèle, les $\hat{\tau}_{n+1}$ sont corrigés pour refléter cette compensation.

Les deux barrières ainsi que la cible sont ajustables dynamiquement. L'effet typique du module DCDT sur l'évolution des délais $\hat{\tau}_{n+1}$ est illustré à la figure 34 (un seul délai est illustré).

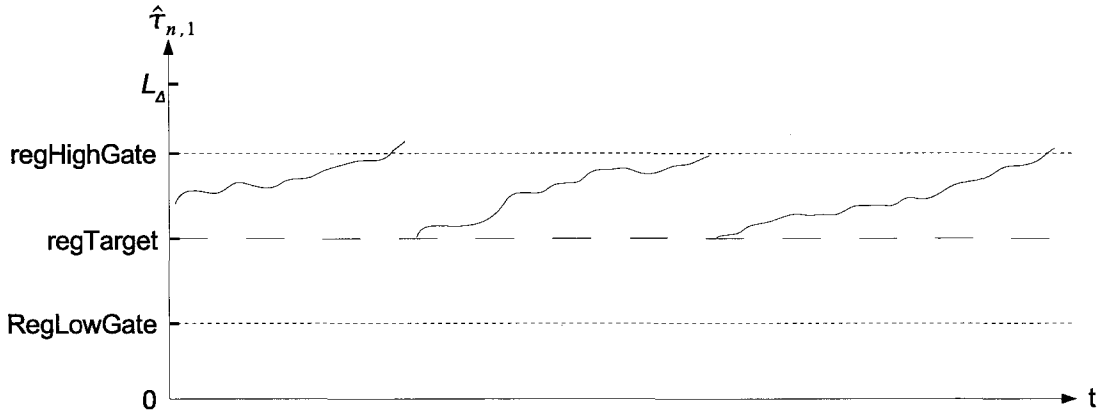


Figure 34 Ajustement de la fenêtre de traitement par DCDT

4.5.8.2 Topologie

Le module DCDT s'insère entre le port de sortie du module TDU et la mémoire BRAMT contenant les délais $\hat{\tau}_{n+1}$. En plus des ports de configuration MBUS et des signaux de contrôle et d'état habituels, il comporte un port de données unidirectionnel en entrée qui le

relie au module TDU et un port de sortie bidirectionnel en sortie relié à BRAMT. DCDT est aussi doté d'un port de sortie supplémentaire, hintD, pour communiquer la correction de la fenêtre de traitement réduite vers le module de contrôle SCU. La figure 35 montre la topologie du module DCDT ainsi que son contexte d'utilisation.

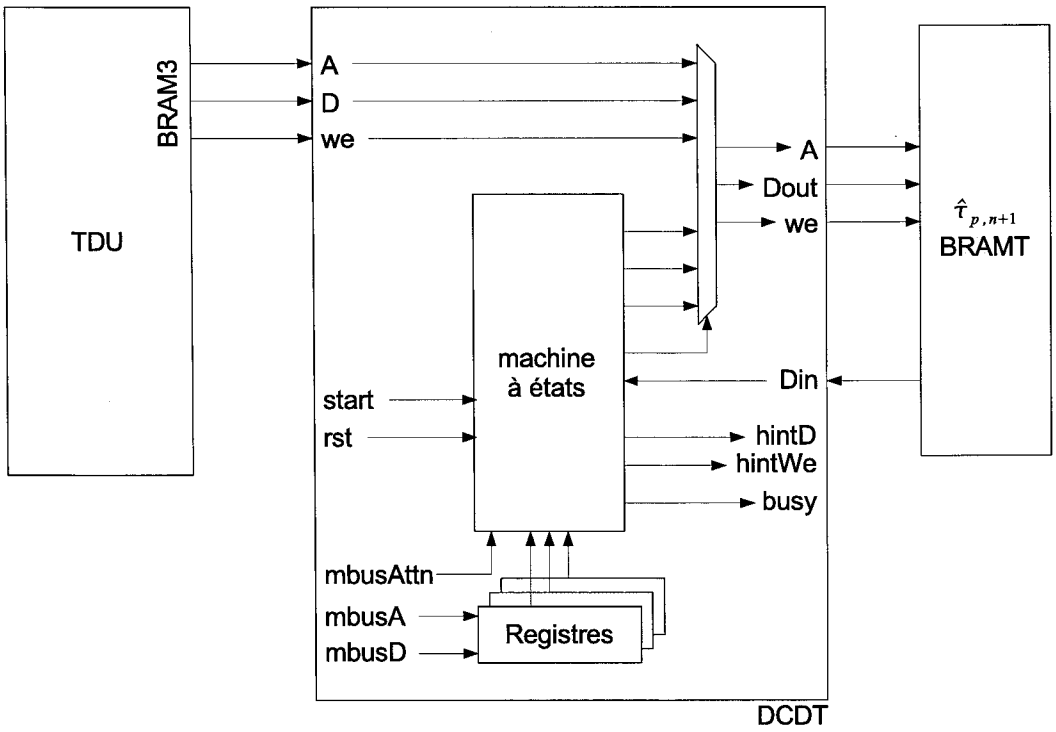


Figure 35 Topologie du module DCDT et contexte d'utilisation

En attente, DCDT transmet toute valeur écrite à son port d'entrée vers BRAMT. Il épie néanmoins chaque transaction et conserve la valeur des délais minimal et maximal en transit. Une fois déclenché, la machine à états amène DCDT à prendre la décision de corriger ou non la fenêtre de traitement en comparant les minima et maxima retenus aux barrières regLowGate et regHighGate.

Lorsqu'une correction est nécessaire, DCDT récupère de BRAMT les délais tout juste traités, leur additionne la correction, et les récrit au même endroit. La fin de l'opération entraîne la remise à zéro de tous les registres de surveillance (adresse mémoire, décompte,

minimum, maximum). La latence totale dépend du nombre de trajets à corriger, mais ne dépasse jamais quelques cycles d'horloge.

Lorsque aucune correction n'est requise ou que le registre regDCDBypass proscrit ce type de correction, la latence du module DCDT est nulle. Les registres MBUS de DCDT sont présentés au tableau XXI de l'annexe 1.

4.5.9 Module de synthèse de la réponse temporelle, FDMAP

La synthèse de la réponse du canal débute par la mise en forme temporelle. Le module FDMAP ("Fractional Delay MAPper") sert essentiellement à positionner une version retardée et échantillonnée à T_C de la réponse impulsionnelle $h_{RRC}(t)$ sur la fenêtre de traitement réduite. De plus, seuls les 16 coefficients centraux de $h_{RRC}(t)$ sont retenus; les autres sont considérés trop faibles pour avoir un impact. Le processus de projection de $h_{RRC}(t)$ s'effectue pour les P trajets consignés dans le vecteur $\hat{\tau}$.

4.5.9.1 Fonctionnement

L'utilisation de $\hat{\tau}$ pour positionner les réponses impulsionnelles $h_{RRC,p}(t-\hat{\tau}_p)$ s'effectue par correspondance dans une table de coefficients pré-calculés. La difficulté du problème réside non pas dans le pré-calcul de ces valeurs mais plutôt dans leur disposition en mémoire de façon à alléger le mécanisme d'adressage.

Deux sources de données sont définies pour le module FDMAP: une première, BRAMT, contient le vecteur $\hat{\tau}$ des positions centrales (délais) des réponses impulsionnelles de l'itération courante, et une seconde, ROMRRC, renferme les coefficients des $h_{RRC,p}(t-\hat{\tau}_p)$ pour tous les délais fractionnaires permis. Puisque la quantification de $\hat{\tau}$ entraîne une précision de 1 / 1024 T_C, 1024 versions différentes des 16 coefficients non-nuls de $h_{RRC}(t)$ sont stockées en mémoire morte. La partie entière des délais n'est en fait qu'un décalage entier de $h_{RRC,p}(t-\hat{\tau}_p)$ et est facilement mise en application.

La quantification des coefficients de $h_{RRC}(t)$ est identique à celle de $\hat{\boldsymbol{D}}$ qui est établie à la section 3.4, soit 11 bits, permettant de représenter les valeurs de -1024 à 1023 (correspondent à la plage -1,0 à 1,0 - 1 / 1024). La valeur 1,0 est absente de cette plage, alors qu'elle est nécessaire pour représenter correctement $h_{RRC}(t)$. La solution la plus simple consiste à stocker en réalité $-h_{RRC}(t)$ en mémoire et à récupérer la véritable forme lors de l'étape de reconstruction (section 2.2.7) en utilisant -1 comme coefficient dans regCoeff0 et regCoeff1 du module RECONST, retrouvant simplement $-(-h_{RRC}(t))$.

4.5.9.2 Adressage

Les valeurs binaires des délais $\hat{\tau}$ sont directement utilisées comme adresses pour la mémoire morte ROMRRC qui contient $h_{RRC}(t)$. Prenant la partie fractionnaire d'un $\hat{\tau}$ comme décalage mémoire, on définit $L_\Delta/2$ zones mémoires contenant chacune les 1024 valeurs possibles de la bribe correspondante. La figure 36 illustre l'agencement des coefficients de $h_{RRC}(t)$ dans ROMRRC.

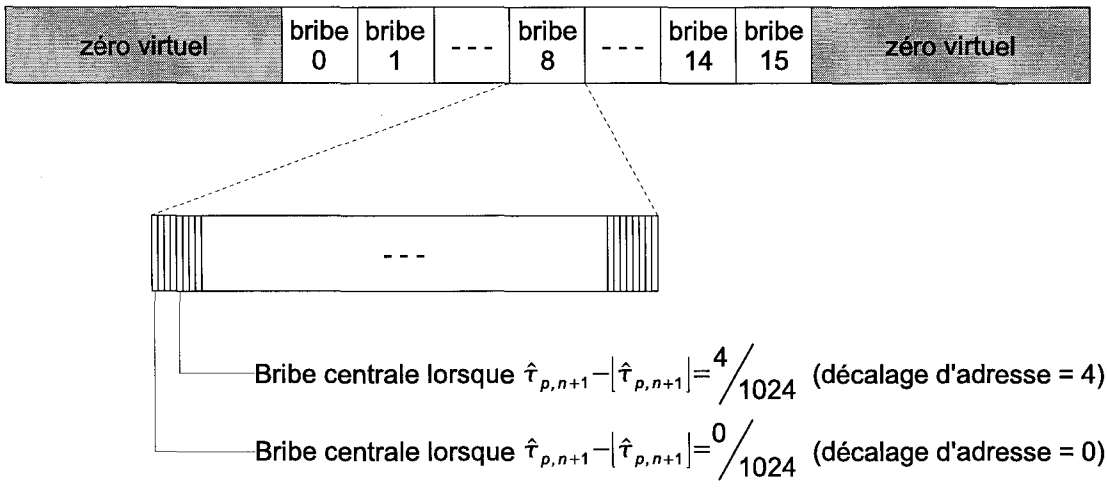


Figure 36 Organisation des coefficients de ROMRRC

La partie entière de $\hat{\tau}$ sert alors à positionner le centre de $h_{RRC}(t)$ dans la colonne p de la matrice $\hat{\boldsymbol{D}}_{n+1}$.

Pour dimensionner les mémoires mortes et découvrir le coût matériel d'une résolution fine de $\hat{\tau}$, on considère les précisions de 1 / 32 T_C à 1 / 1024 T_C en puissance de deux. Il existe toujours 16 zones (ou coefficients non-nuls) peu importe la précision choisie, donc $\log_2(16)=4$ bits d'adresse sont toujours requis; il s'agit des bits de poids fort de l'adresse de ROMRRC. La partie fractionnaire de $\hat{\tau}$ y est concaténée pour ajouter un décalage à l'intérieur de cette zone. La récupération des $L_\Delta/2$ coefficients d'une $h_{RRC,p}(t-\hat{\tau}_p)$ en particulier requiert simplement d'incrémenter la zone mémoire à chaque coefficient tout en conservant la portion "fractionnaire" intacte.

Selon les scénarios identifiés au tableau XI, une précision de 1 / 1024 T_C nécessite l'utilisation de 11 mémoires dédiées, soit moins de 6% de celles disponibles dans la plate-forme cible (section 4.2.1)

Tableau XI

Dimensions de ROMRRC en fonction de la précision de T_C

Précision	Nb. bits fractionnaires	Nb bits total	Organisation (profondeur x largeur)	Nb. Block RAM
1/32 T_C	5	9	1 024 x 18 bits	1
1/64 T_C	6	10	1 024 x 18 bits	1
1/128 T_C	7	11	2 048 x 9 bits	2
1/256 T_C	8	12	4 096 x 4 bits	3
1/512 T_C	9	13	8 192 x 2 bits	6
1/1024 T_C	10	14	16 384 x 1 bit	11
1/2048 T_C	11	15	16 384 x 1 bit	22

Il est possible de ne stocker que les coefficients pour une plage de délais $[0 \cdots 0{,}5\,T_C-\varepsilon]$ moyennant une quantité supérieure de logique programmable capable d'extraire la symétrie de $h_{RRC,p}(t-\hat{\tau}_p)$. Le faible coût en ressources mémoire justifie plutôt d'opter pour la solution générale (et sous-optimale) décrite plus haut.

La partie entière de $\hat{\tau}_{p,n+1}$ indique l'endroit où doit se retrouver le coefficient central de $h_{RRC,p}(t-\hat{\tau}_p)$ dans $\hat{\boldsymbol{D}}_{p,n+1}$ (le coefficient central provient de la zone 8 de ROMRRC). Pour obtenir une correspondance entre la première adresse de $\hat{\boldsymbol{D}}_{p,n+1}$ et la zone de ROMRRC, on soustrait $\lfloor \hat{\tau}_{p,n+1} \rfloor$ à la constante 8. Ceci entraîne une adresse invalide (c.-à-d. inférieure à zéro) lorsque que $\hat{\tau}_{p,n+1} > 8$. De plus, puisque ROMRRC ne contient que 16 zones, toute valeur de $\hat{\tau}_{p,n+1} < 23$ entraîne des adresses dépassant la plage allouée. Ces cas d'adressage sont illustrés à la figure 37.

Ces adresses invalides sont interceptées et occasionnent un coefficient nul (« zéro virtuel » sur la figure 37). Ce processus permet de balayer les L_Δ adresses de la colonne p de $\hat{\boldsymbol{D}}_{n+1}$ et de détruire tout coefficient étranger qui s'y trouve.

4.5.9.3 Topologie

Comme les autres modules pipelinés, FDMAP comporte deux ports d'entrée et un port de sortie pour données avec interfaces pour mémoires dédiées. Il réunit en outre un port de configuration MBUS et des signaux de contrôle et d'état communs à tous les blocs du récepteur. Puisqu'il n'est composé que d'une machine à états et d'une pile de registres, la topologie de FDMAP n'est pas illustrée.

En plus des adresses de base habituelles (regBram1Base et regBram3Base), seuls le facteur d'étalement (regRstMaskInnerCount), le nombre de trajets P (regBram1RstMask) et la taille d'une zone de coefficient au sens de la figure 37 (regBram4Incr) sont requis.

Un compteur interne permet de copier le nombre adéquat de coefficients de ROMRRC pour chaque trajet p et indique la fin d'une copie lorsqu'il atteint regRstMaskInnerCount, lequel doit correspondre à $L_\Delta - 1$. Tous les registres MBUS du module FDMAP sont détaillés au tableau XXII de l'annexe 1.

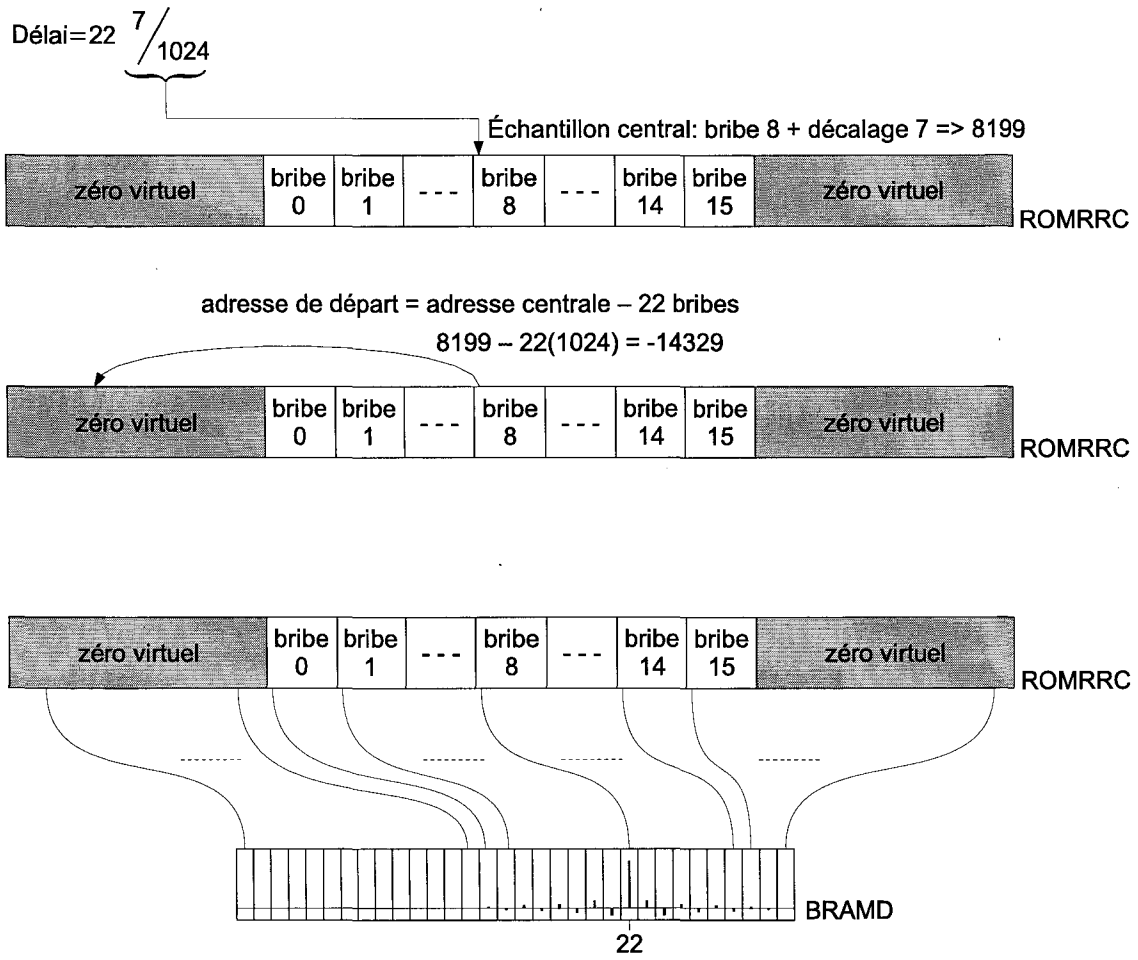


Figure 37 Correspondance entre ROMRRC et BRAMD

Le code source VHDL de la mémoire morte ROMRRC fait abstraction de l'organisation en mémoire et laisse à l'outil de synthèse le soin de trouver une configuration optimale. Suivant les technologies Xilinx et Altera, les mémoires dédiées peuvent compter jusqu'à trois ports d'accès indépendants, permettant de partager la même ROMRRC entre trois modules FDMAP distincts, et par extension, entre trois pipelines STRF.

4.5.10 Module combiné de normalisation

En conformité avec la section 3.3.1.12, la normalisation de $\underline{\hat{H}}$ tient un rôle de garde-fou dans le pipeline STRF.

4.5.10.1 Topologie

Le module de normalisation est constitué de deux sous-modules ainsi que d'une table de correspondance entre norme au carré ($\sum x^2$) et coefficient de normalisation. Illustré à la figure 38, le premier sous-module, normSnoop, est passif et se branche au port de sortie d'un bloc de traitement quelconque. Il effectue la sommation des carrés de chaque élément transitant du port 1 au port 3. Une fois la séquence d'écriture terminée, normSnoop transmet la norme (au carré) par le port sqrtA (littéralement « square root address »), laquelle est utilisée comme adresse mémoire pour la table de correspondance ROMPWR, où est stockée, à cette adresse, le facteur de normalisation correspondant.

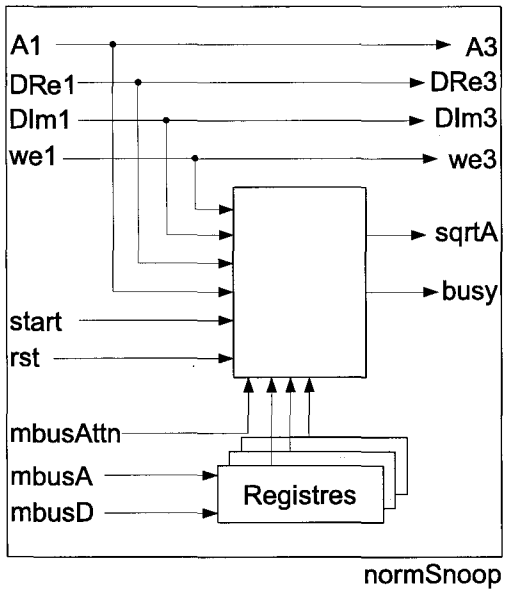


Figure 38 Topologie du sous-module normSnoop

Le second sous-module, normApply, effectue un produit scalaire entre le vecteur analysé et le coefficient provenant de ROMPWR. Le calcul est pris en charge par un module de type CONJMULT (section 4.5.3) auquel on juxtapose un élément logique chargé de la programmation autonome du coefficient extrait de la table. Normalement un conduit passif pour les commandes MBUS, cet élément injecte le coefficient de normalisation ainsi que l'adresse MBUS correspondante dans CONJMULT dès que le signal de démarrage lui

parvient du module de contrôle SCU (4.5.12). Le schéma interne de normApply est rapporté à la figure39.

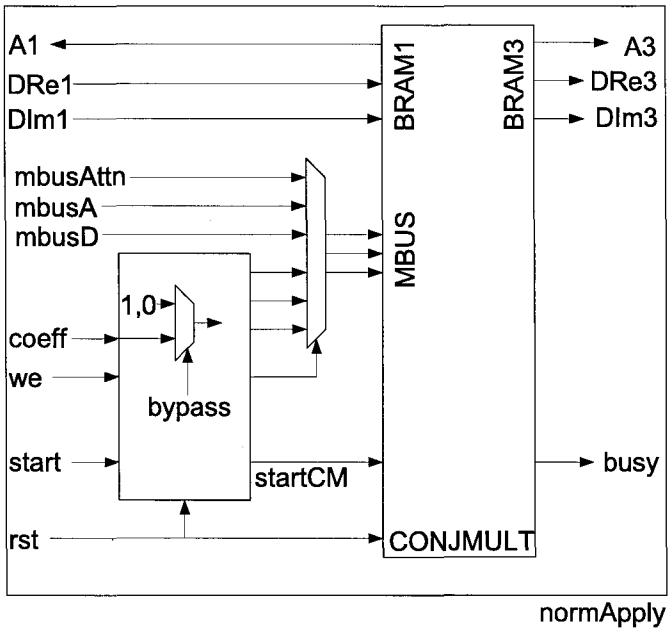


Figure 39 Topologie du sous-modules normApply

Ce module doit impérativement pouvoir être mis hors service de façon dynamique puisqu'il est théoriquement superflu (section 3.3.1.12). On ne peut cependant le laisser inerte car les données doivent le traverser pour atteindre le port 3. La solution consiste simplement à forcer la valeur 1,0 comme coefficient de normalisation. Ce contrôle s'effectue par l'entremise d'un registre booléen (« bypass » sur la figure 39) accessible par MBUS. Le détail des registres MBUS est donné au tableau XXIII de l'annexe 1.

4.5.10.2 Interfaces

Le module passif normSnoop est doté d'un port d'entrée (port no. 1) similaire à celui d'une mémoire. Son port d'adresse sqrtA, branché à ROMPWR, et l'absence d'un port MBUS, le différencie des autres modules. La mémoire de correspondance ROMPWR, quant à elle, n'a que deux ports: l'un permettant l'adressage, et l'autre, de récupérer le facteur de

normalisation. Le module actif, normApply, comporte toutes les interfaces typiques des modules pipelinés (voir matrixMult, section 4.5.2).

4.5.10.3 Contexte d'utilisation

Les branchements typiques du module de normalisation combiné sont illustrés à la figure 40. On retrouve, à l'extrême gauche, le bloc de traitement précédent, et à l'extrême droite, la mémoire dédiée qui accueillera le vecteur normalisé, $\underline{\hat{H}}_{n+1}$.

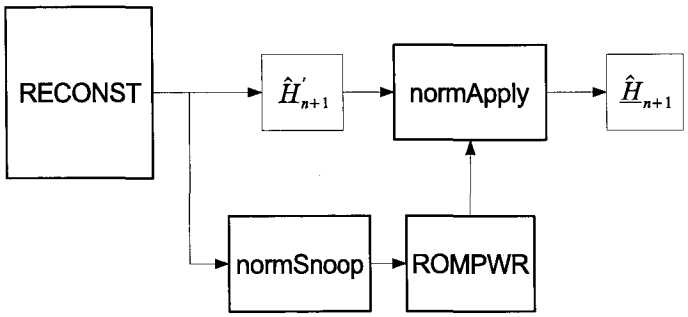


Figure 40 Contexte d'utilisation du module combiné de normalisation

4.5.11 Module d'estimation de la puissance

L'inclusion du module d'estimation de puissance (« PwrEst ») à l'intérieur du FPGA provient de la nécessité d'évaluer l'équation 2.30 à chaque nouveau symbole. Bien qu'un microprocesseur embarqué saurait s'en acquitter, les nombreux usagers de la station de base l'encombrerait rapidement. De plus, très peu de ressources sont requises, et l'algorithme ne changera vraisemblablement pas. Pour ces raisons, une réalisation en logique programmable est souhaitable. PwrEst a pour seule mission d'appliquer un facteur de lissage α à une somme conservée en mémoire tampon (section 3.3.1.6). Cette manipulation est répétée à chaque nouveau symbole.

4.5.11.1 Topologie

Les ressources utilisées par le module PwrEst se résument à deux multiplicateurs dédiés et moins d'une centaine de LUTs. Puisque l'opération arithmétique ne fait intervenir que deux scalaires, l'utilisation de registres constitués de LUTs est privilégiée. En plus de libérer des ressources dédiées, ce modèle évite la mécanique d'adressage requis par d'autres modules. On équipe donc PwrEst d'un port d'entrée de données qui pourra recueillir $\hat{s}_{n+1}$ et d'un port de sortie qui montre en tout temps la dernière valeur de $\hat{\Psi}^2_{n+1}$.

Un registre de configuration contient la valeur de α exprimée en fractions de 2^{-7}. Un second registre inaccessible contient 1-α mais demeure inaccessible via MBUS puisqu'il est calculé *in situ* à partir de α. La figure 41 donne le détail du fonctionnement du module PwrEst. Elle regroupe à la fois les deux étages d'un petit pipeline arithmétique ainsi que l'état ADD_2 de la machine à états finis qui récupère et utilise le résultat du second étage du pipeline.

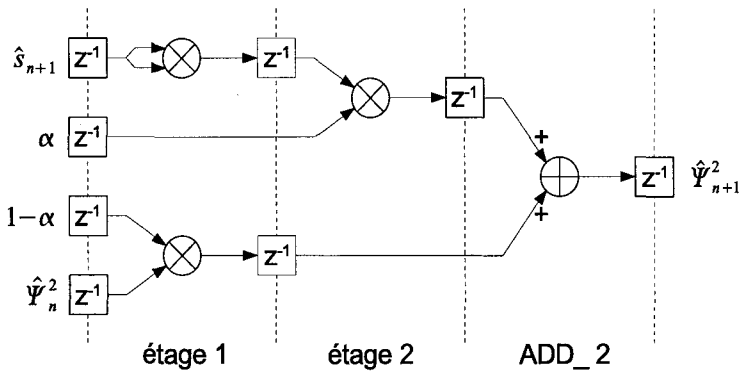


Figure 41 Topologie du bloc PwrEst

4.5.11.2 Registres

Le registre unique, regAlpha, est accessible à l'adresse 0_d par MBUS. Son contenu est multiplié par 2^{-7} et constitue le facteur de lissage α.

4.5.12 Module de contrôle, SCU

La séquence d'entraînement des modules des régions STRF et SP est prise en charge par le module de contrôle de STAR, (« STAR Control Unit » ou SCU). Ce module assure l'insertion de nouveaux paramètres de fonctionnement en séquence dans les différents modules du récepteur et permet les transactions de données entre la région PM et les différentes mémoires imbriquées dans SP et STRF. On le réalise en logique programmable afin de minimiser la latence des échanges avec ses modules subordonnés.

4.5.12.1 Fonctionnement

SCU est constitué de trois machines à états. Une première, SM_OMBUS, est le portail de configuration du récepteur STAR pour le monde extérieur. Répondant au même protocole que le MBUS interne, SM_OMBUS reçoit du port OMBUS externe (« Outside MBUS ») les paramètres systèmes tels que μ , ζ , M, P et L et les consigne dans les registres appropriés. La liste exhaustive de ces registres est donnée au tableau XXIV de l'annexe 1. Seule SM_OMBUS a droit d'écriture dans ces registres et peut les initialiser lors de la remise à zéro du système.

La seconde machine à états, SM_STRF, gère exclusivement la séquence de fonctionnement des modules de la région STRF. À cette fin, elle doit en effectuer la configuration, vérifier la disponibilité et commander le démarrage de chacun des étages. L'organigramme de la figure 42 démontre la séquence typique de configuration et de démarrage d'un étage k du pipeline STRF. La configuration s'effectue par le truchement du bus STRF_MBUS qui relie SM_STRF à tous les modules appartenant à STRF. Seule SM_STRF a droit d'écriture sur ce bus, ce qui élimine toute possibilité de contention.

Il existe une contrepartie physique à la séparation du pipeline STRF. L'étape de synthèse de la réponse spatio-temporelle, qui débute avec le module FDMAP (section 4.5.9), récupère les délais $\hat{\tau}_{p,n+1}$ qui peuvent provenir du module précédent, TDU, ou de la

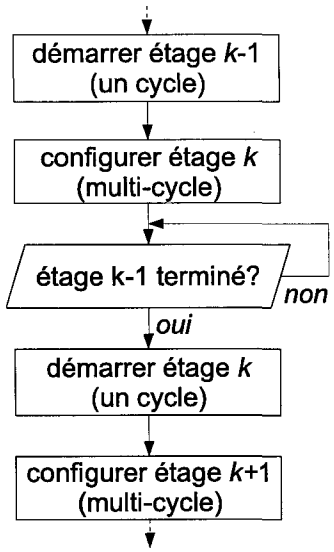


Figure 42 Séquence de configuration du pipeline

région PM externe. Afin de permettre l'injection de nouveaux délais par le PM, l'opération du pipeline peut être interrompue avant le démarrage de TDU, et le contrôle peut passer directement à FDMAP si de nouvelles données en provenance du PM sont effectivement présentes. La sémaphore requise pour cet échange est précisée à la section 5.3.2.1. Un compte à rebours est lancé lors de cette période d'échange afin d'annuler le transfert s'il ne réussit pas en 1024 cycles d'horloge.

La troisième machine à états, SM_SP, gère le fonctionnement des modules de la région SP. Par sa conception, le temps de traitement d'un symbole est inférieur à T_S. On emploie donc un signal externe, newSymbol, pour signaler à SM_SP qu'une nouvelle itération doit débuter. Son paradigme de configuration est identique à celui présenté à la figure 42, à la différence que c'est SP_MBUS qui relie SM_SP aux modules concernés.

Chaque événement newSymbol déclenche la séquence SP constituée de la corrélation (module DESP), l'identification du canal (modules CDFI et UDFI) et la combinaison (module RECONST, de type matrixMult). Une fois l'étape d'identification terminée, le compteur regSymbol est incrémenté et comparé à regSymbolToCID. Lorsqu'il y a égalité,

SM_SP lance une requête à SM_STRF pour débuter une itération d'analyse/synthèse à l'aide de la matrice $\underline{\tilde{H}}$ nouvellement calculée, et redémarre regSymbol à zéro. Si toutefois SM_STRF se trouvait occupée (cidBusy à l'état haut), regSymbol conserverait sa valeur afin d'être réévalué à l'affirmative lors du symbole suivant.

Ce mécanisme d'essais successifs permet d'atteindre deux objectifs: il traduit la fonction du paramètre n_{STRF} (section 3.2.2.4), ou de fonctionner au taux maximal soutenu par SM_STRF. Ce type d'opération sans blocage (« non-blocking ») permettra l'ajout de plusieurs PM et STRF et l'insertion d'un module d'aiguillage entre chacun.

4.5.12.2 Topologie

La figure 43 illustre les différents éléments du module SCU.

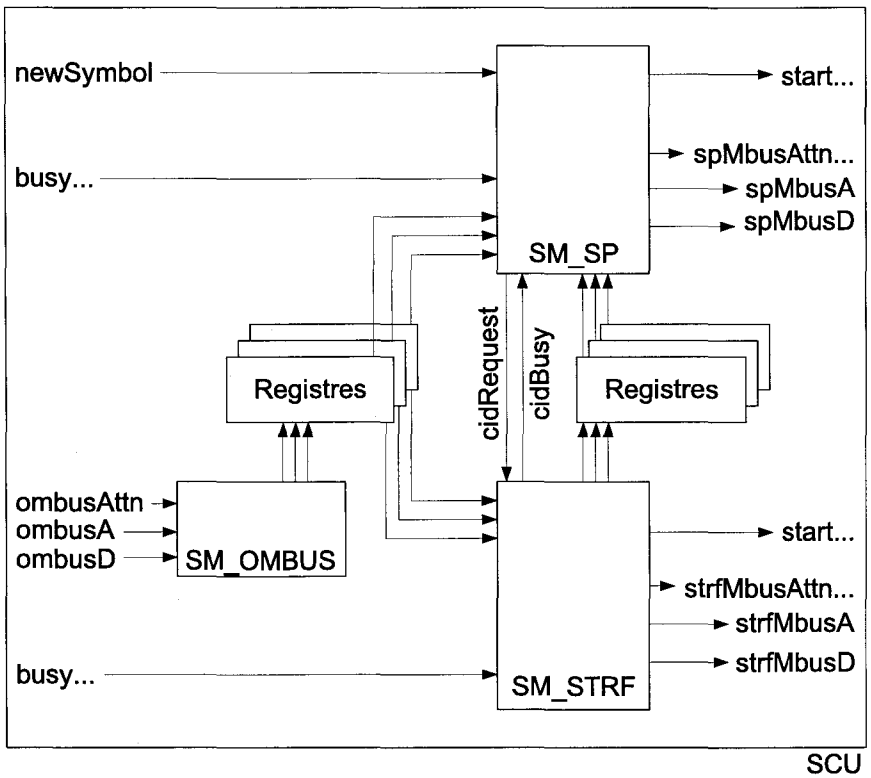


Figure 43 Topologie du module de contrôle SCU

Seuls les signaux MBUS des différentes régions et les signaux de contrôles de tous les modules subordonnés sont reliés à SCU; aucune interface mémoire n'est utilisée. Les registres de SCU concernant l'interface OMBUS sont énumérés au tableau XXIV de l'annexe 1.

4.6 Conclusion

Ce chapitre présente la première moitié de la réalisation matérielle du récepteur STAR. Les plate-formes de développement ont tout d'abord été présentées. Une séparation définitive des domaines de calculs dans différentes zones physiques a ensuite été effectuée: les régions SP et STRF dont ce chapitre fait l'objet ont été confinées à la maille logique du FPGA, alors que le gestionnaire de trajets est réservé à une réalisation par micrologiciel dont traite le chapitre 5.

Les détails techniques de la réalisation ont ensuite été donnés en deux parties. Tout d'abord, l'interface de communication entre les modules arithmétiques a été définie de manière à permettre un branchement direct aux mémoires dédiées du FPGA. Une interface de contrôle partagée par tous les modules arithmétiques a ensuite été élaborée, facilitant la configuration répétitive des étages du pipeline par un contrôleur spécialisé dont la dernière section explique les tâches.

Le chapitre suivant reprend la discussion à ce niveau puisque ce contrôleur est le lien entre les régions SP et STRF ainsi que le reste de l'algorithme STAR, qui existe en tant que code logiciel dans un microprocesseur embarqué, d'où l'emploi du terme « codesign ».

CHAPITRE 5

INTÉGRATION CODESIGN ET LOGICIELLE

5.1 Introduction

Alors que la plus grande part de la puissance de calcul requise par STAR est utilisée pour le décodage de symboles (« symbol path » ou SP) et la régularisation de la structure du canal (« structure fitting » ou STRF), tel que mentionné aux chapitres précédents, la gestion des trajets est plus favorablement réalisée en tant que logiciel pour diverses raisons qui sont données au chapitre 2.

Puisque ce microprocesseur embarqué appartient à un niveau hiérarchique supérieur et doit s'acquitter des tâches de communications avec d'autres parties du récepteur, le chapitre dresse tout d'abord un portrait d'ensemble du récepteur et des modules requis pour assurer son fonctionnement dans la plate-forme de développement choisie. La première section traite donc de l'assemblage des modules définis au chapitre 4 et des différents éléments du système.

La section 5.3 donne les spécifications des échanges de données entre les différents niveaux hiérarchiques du système. En particulier, on y retrouve les sémaphores utilisées par le microprocesseur embarqué pour dialoguer avec le contrôleur SCU qui pilote directement le pipeline interne de STAR. Ensuite, la section 5.4 donne le détail de la réalisation de l'algorithme de gestion des trajets tel que défini aux chapitres 2 et 3. On y trouve en outre un organigramme détaillé expliquant son fonctionnement ainsi que les étapes de démarrage du système. Le dernier niveau hiérarchique, l'ordinateur hôte de la plate-forme de développement, est brièvement présenté à la section 5.5 où l'on mentionne les tâches dont il doit s'acquitter ainsi que les possibilités de test qu'il offre.

Partant de ce point, la validation du récepteur, jusqu'ici ignorée, est enfin abordée à la section 5.6, qui traite de cosimulation et détaille l'architecture unifiée de simulation qui permettra, au chapitre 6, de mesurer les dégradations infligées au récepteur STAR à toutes les étapes du développement.

Finalement, la section 5.7 présente une interface de surveillance et de contrôle du récepteur STAR qui se substitue avantageusement à Matlab au niveau de la fluidité de l'affichage et du contrôle intuitif des paramètres-clés de l'algorithme STAR. C'est cette interface qui sera réutilisée lors de démonstrations publiques (annexe 4).

5.2 Assemblage

Les régions de décodage des symboles (« symbol path » ou SP) et d'analyse/synthèse du canal (« structure-fitting » ou STRF) définies au chapitre précédent sont regroupées dans un même bloc à l'extrême droite de la figure 44. Elles échangent des données avec le reste du système à l'aide de mémoires dédiées et de registres unidirectionnels, eux-mêmes reliés à un module d'adaptation (« Intellectual Property Interface » ou IPIF). Ce dernier sert de pont entre l'ensemble des modules fournis par ISR Technologies et un bus standardisé (« On-Chip Peripheral Bus » ou OPB). Ce type de bus est répandu dans l'industrie et plusieurs éditeurs commerciaux offrent des modules qui s'y branchent.

On compte, parmi ces modules, le microprocesseur Microblaze, ainsi qu'un pont d'adaptation entre les bus OPB et PCI, offert par le fabricant de la plate-forme de développement utilisée. Le système est complété par un PC hôte qui effectue le contrôle de haut niveau et permet l'interaction avec le récepteur.

5.3 Sémaphores

Les échanges de données entre les différentes régions du système doivent se conformer à différents protocoles répandus et supportés. Ces protocoles sont essentiellement des conduits virtuels par lesquels transitent des informations d'état et des sémaphores de

contrôle. Ces sémaphores encapsulées ne répondent, quant à elles, à aucune convention particulière, étant donné leur lien direct avec l'architecture du récepteur STAR. Elles sont documentées dans les sections suivantes.

5.3.1 Communications entre hôte et Microblaze

Ce lien dépend entièrement de la plate-forme de développement choisie. Il est donc primordial qu'il demeure modulaire vis-à-vis le reste du système, afin de minimiser l'effort d'adaptation lors du passage à d'autres environnements. Dans le cas de la plate-forme daqPC d'ICS, l'échange de données s'effectue à travers une série de registres que peuvent atteindre à la fois l'hôte, par son bus PCI-X, et le Microblaze, à travers le module OPB/PCI, tous deux illustrés à la figure 44.

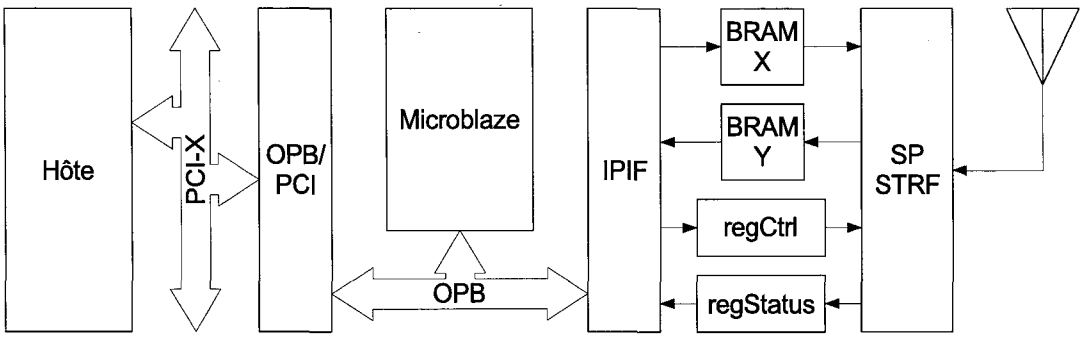


Figure 44 Communications entre l'hôte, le Microblaze et les zones pipelinées

De part et d'autre, ces registres apparaissent comme une plage mémoire constituée d'éléments de 32 bits. Afin de simplifier les transactions, cette plage est scindée en deux zones d'échange unidirectionnelles (une première de l'hôte au Microblaze, et une seconde en sens inverse), chacune bénéficiant de transferts rapides (« direct memory acces » ou DMA) sur le bus PCI-X. Aux fins de cueillette de données statistiques et d'affichage en temps réel, l'hôte lit périodiquement les plages mémoire correspondant aux opérandes concernées.

Bien que le bus PCI-X bénéficie d'une bande passante de 200 méga-octets/s en pointe, la latence que lui confère le système d'exploitation élimine toute possibilité d'asservissement en temps réel du récepteur. L'ajustement de paramètres globaux, cependant, s'y prête fort bien. Acheminés de manière intermittente (au rythme de l'humain qui l'utilise), ces paramètres parviennent au récepteur de façon asynchrone, par écriture dans les registres appropriés. Le Microblaze, mandataire de l'hôte dans ce type de transaction, récupère les paramètres périodiquement pour les communiquer au module SCU au moment opportun.

Aux yeux du Microblaze, cette plage mémoire est représentée par un pointeur à une table d'éléments de 32 bits. Comme son micrologiciel, rédigé en en langage C, pilote un système physique, l'adresse de ce pointeur est forcée à la première adresse physique de la plage mémoire, conformément à la spécification préalablement donnée à l'outil d'assemblage du système embarqué (« Xilinx Embedded Development Kit » ou EDK).

Du point de vue du micrologiciel, l'accès aux différentes zones mémoire est identique à la manipulation de variables. En pratique cependant, l'accès est plus lent que celui de variables en mémoire principale puisque la transaction doit franchir le bus OPB et le module OPB/IPIF ou OPB/PCI.

Le Microblaze accède lui aussi à ces registres de manière asynchrone; autant la lecture que l'écriture s'effectuent lorsque le microprogramme le nécessite, et non pas par demande/acquiescement avec l'hôte.

5.3.2 Communication entre Microblaze et SCU

5.3.2.1 Liens de contrôle

La figure 44 montre deux types de liens entre le Microblaze et les régions de STAR faites de logique programmable. Le premier type est identique aux liens MBUS internes: un registre unidirectionnel, accessible via OPB/IPIF (regCtrl), pilote les signaux d'adresse et de données OMBUS, qui atteignent la machine à états SM_OMBUS du module SCU

(section 4.5.12). Un lien inverse, regStatus, permet de recouvrer certaines informations d'état ainsi que les sémaphores de communication inter-régions.

Le second type de lien donne un accès privilégié à différentes mémoires dédiées internes des régions SP et STRF. Grâce à une interface de tables de mémoire développée par ISR, il est possible d'imbriquer les adresses des mémoires internes aux plages mémoire du bus OPB. Afin d'éliminer les problèmes de contention et minimiser le temps d'accès, ces liens sont unidirectionnels, ce qui entraîne le dédoublement des ressources mémoire. Les cas de lecture et d'écriture sont représentés à la figure 45.

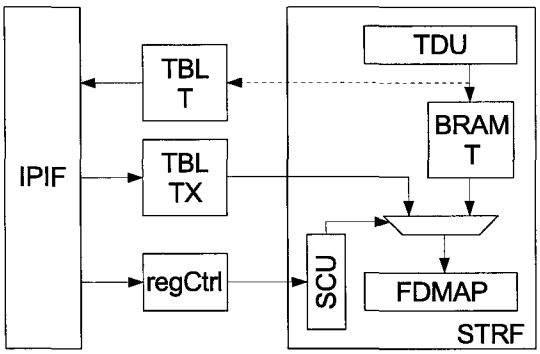


Figure 45 Échange de données entre le Microblaze et les mémoires internes

La lecture s'effectue par extraction (en pointillé à la figure 45) des signaux d'écriture internes afin d'alimenter en continu une table mémoire dédiée. L'écriture, au contraire, nécessite l'interruption momentanée du pipeline de la zone STRF. Les opérandes $\hat{\boldsymbol{j}}$ et $\hat{\tau}$, qui sont les seules à être remplacées dans STRF par la région PM, doivent être lues par le Microblaze (par exemple, TBL T), modifiées par son microprogramme, et récrites dans un laps de temps minimal (dans TBL TX).

À cette fin, le Microblaze envoie une requête au contrôleur SCU par l'entremise d'un état haut sur le bit *fetching* du registre *regCtrl*. Asynchrone, cette requête peut atteindre SM_STRF à tout moment, exigeant du Microblaze qu'il attende la première disponibilité de la machine à états, signalée par l'état haut du bit *rdyFetch* par le module SCU. Dès lors,

le pipeline STRF est interrompu et le plein contrôle en écriture des mémoires BRAMJ et BRAMT est cédé au Microblaze. Cette séquence empêche les modules STSEP et TDU d'écraser les données fraîchement écrites par la région PM.

Une fois la transaction terminée, le Microblaze émet de nouveaux paramètres *M*, *P* et *L*, qui reflètent le nouveau contenu des mémoires modifiées. Aussitôt la transaction terminée, le bit *fetching* retrouve l'état 0, permettant au module SCU de commander la remise en marche du pipeline et de répandre les nouveaux paramètres au fur et à mesure que les opérandes de nouvelles dimensions atteignent les différents étages du pipeline. La transaction est illustrée à la figure 46.

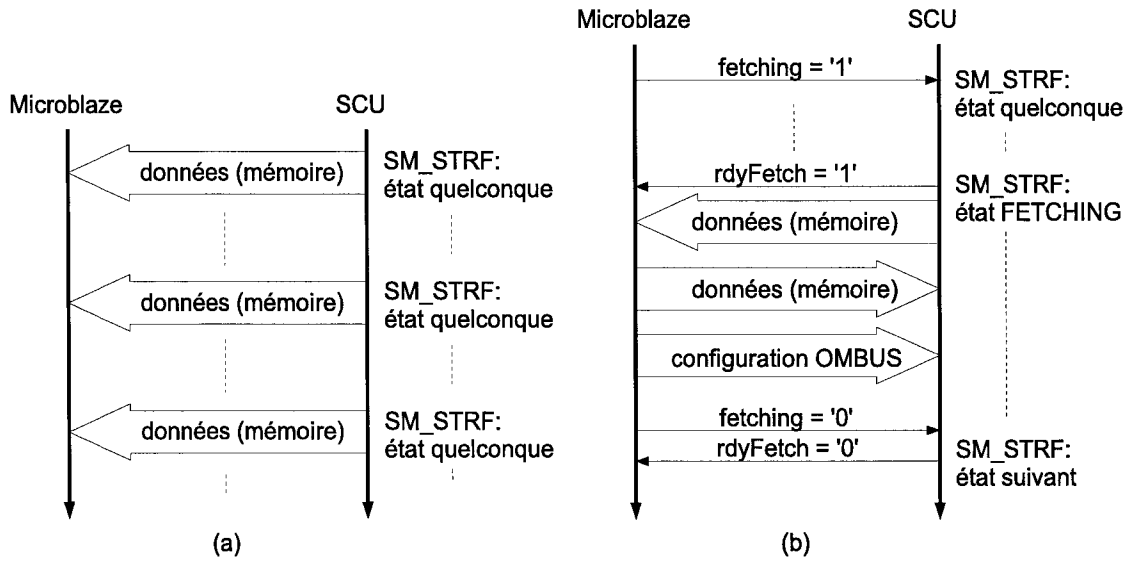


Figure 46 Sémaphores de communication (a) en lecture et (b) en écriture

5.4 Surveillance des trajets par micrologiciel

Étant donné la nature de ses algorithmes, la région PM est avantageusement réalisée à l'intérieur du micrologiciel du Microblaze. Exploitant le taux de fonctionnement réduit n_{PM}, l'interaction entre PM et STRF est considérablement réduite mais ne dégrade que de façon minime les performances de réception.

5.4.1 Organigramme logiciel

Le microprogramme qu'exécute le Microblaze est constitué d'une séquence de démarrage pour le récepteur STAR, suivie d'une boucle principale infinie. La surveillance des trajets est réalisée par la fonction *analyze*, exécutée dans la boucle principale à intervalles programmables par n_{PM}.

5.4.1.1 Démarrage de STAR

La séquence d'initialisation de STAR remplit trois tâches: elle place les modules du récepteur dans un état connu, transmet le code combiné d'étalement et de chiffrage au corrélateur, et initialise les opérandes $\hat{\boldsymbol{J}}$ et $\hat{\tau}$ à des valeurs par défaut. Les sémaphores utilisées lors du démarrage sont les mêmes qu'en fonctionnement normal, à la différence qu'elles sont entrecoupées de consignes de remise à zéro globales (« RÀZ »). La réception de nouveaux symboles est elle aussi désactivée en masquant le signal newSymbol. La figure 47 montre l'organigramme des différentes étapes de la séquence d'initialisation.

5.4.1.2 Boucle principale

La boucle où pénètre le Microblaze une fois la séquence d'amorçage terminée enchaîne les appels à la fonction *analyze* qui compare $\breve{\underline{H}}$ et $\hat{\underline{H}}$ et effectue les changements au pipeline STRF du récepteur lorsque requis. Alors qu'un évènement de comparaison (une itération de la fonction *analyze*) n'interrompt pas le fonctionnement du pipeline STRF (c.-à-d. la comparaison est « non-blocking »), la mise à jour de $\hat{\boldsymbol{J}}$ et $\hat{\tau}$ doit impérativement l'arrêter. Cette interruption doit être brève, sous peine d'affecter les performances du récepteur qui ne peut, durant cette intervention, effectuer les mises à jour de $\hat{\underline{H}}$. Le décodage des symboles par la région SP continue normalement avec la dernière matrice $\hat{\underline{H}}$ synthétisée.

Les paramètres en provenance de l'hôte sont ré-évalués à chaque itération de la boucle principale et sont communiqués au Microblaze au besoin. Ces interventions n'interrompent

pas le processus de réception puisque le module de contrôle SCU injecte ces paramètres au moment opportun, dans sa séquence d'opération normale.

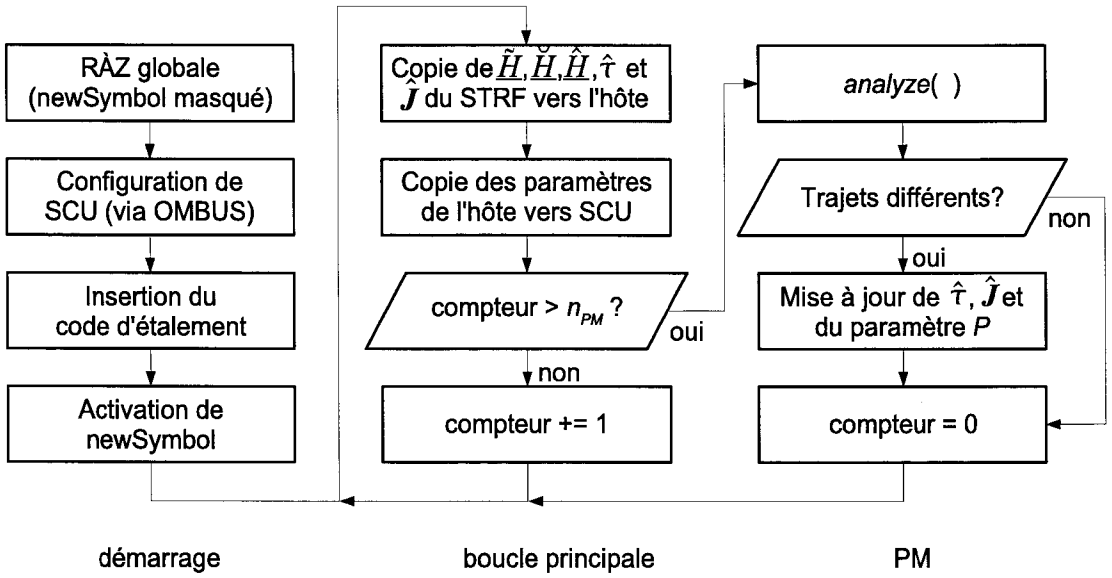


Figure 47 Organigramme du microprogramme

5.4.1.3 Fonction « analyze »

La fonction *analyze*, qui met en oeuvre les algorithme de la région PM (équations 2.25 à 2.28), est constituée de quatre étapes distinctes. En accord avec l'organigramme de la figure 47, la fonction *analyze* est appelée périodiquement, suivant le nombre de symboles reçus, et selon le paramètre n_{PM}, ajustable dynamiquement. Généralement, la fonction *analyze* est appelée à chaque 100 symboles reçus lorsque $L = 32$.

Sa première tâche consiste à évaluer les normes M-dimensionnelles des P colonnes de $\hat{\boldsymbol{J}}$ afin de les comparer au seuil de disparition *vanThresh* (qui représente υ_V), programmable dynamiquement. Un ensemble de P_{MAX} compteurs conservent le nombre d'itérations consécutives où les P trajets connus ont une puissance inférieure au seuil voulu. Une seul escapade au-dessus de ce seuil entraîne la remise à zéro du compteur et la prolongation de la présence du trajet.

La seconde étape reprend le même procédé sur le vecteur $\underline{\breve{H}}$, et compare les résultats au seuil d'apparition *appThresh* (représentant υ_A). Cette fois, c'est un ensemble de L_Δ compteurs qui conservent la trace des trajets dépassant le seuil lors d'itérations consécutives de la fonction *analyze*. Il suffit d'une seule évaluation négative pour réinitialiser le compteur correspondant. Fait à noter, seuls les trajets distants de plus d'une bribe des trajets déjà connus sont considérés.

À ce stade-ci, le microprogramme peut décider si des changements doivent être communiqués au module SCU; *analyze* se termine lorsque c'est inutile. Par contre, lorsqu'une modification au profil multi-trajets est requise, le microprogramme informe le pipeline STRF de cesser le traitement afin d'obtenir le contrôle des mémoires concernées. Cet enchaînement de signaux est illustré à la figure 46(b).

Dès cet instant moment, la troisième étape de la fonction *analyze* s'enclenche. Après avoir relu les toutes dernières valeurs de $\hat{\tau}$ dans BRAMT et de $\hat{\boldsymbol{J}}$ dans BRAMJ, le Microblaze recopie, dans BRAMTX et BRAMJX, respectivement, les données des trajets n'ayant pas été éliminés à la première étape. Du coup, la quantité d'emplacements libres (P_{MAX} - P) risque d'augmenter. Ces places fraîchement libérées sont alors disponibles pour l'ajout de trajets naissants.

La quatrième et dernière étape initialise, pour chacun des nouveaux trajets, la puissance et la position dans BRAMTX et BRAMJX, jusqu'à concurrence de (P_{MAX} - P) nouveaux trajets. Les positions des nouveaux trajets sont entières (i.e. portion fractionnaire nulle) et correspondent à l'index de $\underline{\breve{H}}$ où ceux-ci ont été découverts. Leur ratio de puissance est simplement (ré)initialisé à 1 / P et retrouvera, dès la prochaine itération du pipeline STRF, sa position optimale.

Une fois les modifications terminées, la nouvelle valeur du paramètre P est transmise au module SCU par l'entremise de l'OMBUS, et le traitement du pipeline STRF peut reprendre.

5.5 Système hôte

Le récepteur, même muni d'un microprocesseur, n'est cependant pas autonome. Certains paramètres ou blocs de données lui parviennent du monde extérieur, et la réponse en temps réel n'est alors pas requise. Ces opérations sont donc confiées à un niveau hiérarchique supérieur générique, puissant, mais où la réponse en temps réel est impossible: un logiciel de contrôle sur l'hôte et son système d'exploitation. Par l'entremise du bus PCI-X, ce logiciel assure le bon fonctionnement du récepteur dans son ensemble: modification de paramètres système, échange de données d'état, ainsi que toute autre tâche de supervision périphérique.

La mise en marche de la plate-forme est aussi effectuée par l'hôte: l'initialisation des CAN, la syntonisation des modules de sous-échantillonnage et le démarrage du Microblaze sont tous pris en charge par l'hôte. Évidemment, ces procédures sont intimement liées au matériel utilisé, et sont impropres à la réutilisation logicielle.

De plus, le lien entre les cartes d'acquisition et de transmission est physiquement établi par le bus PCI-X de l'hôte et régi par ce dernier. Puisque la station de base génère (et requiert) des taux binaires relativement faibles, ce lien tout-usage suffit largement à entretenir la communication.

Les informations de contrôle de puissance, disponibles en parallèle au message binaire reçu, doivent s'intégrer aux données à transmettre; elles doivent parvenir à la carte de transmission qui se chargera de les intégrer au message transmis en conformité avec la norme 3G. Elles peuvent par ailleurs être utilisées pour ajuster la puissance de transmission d'un appareil dédié par lien filaire sériel. Des tests ont démontré la faisabilité d'ajuster ainsi l'amplitude de transmission d'un appareil ESG 4439 d'Agilent. Étant donné le faible débit sériel supporté par l'appareil, le taux maximal d'ajustement de la puissance de transmission atteignable par cette méthode est environ le dixième du taux prévu par la norme 3G (160 fois par seconde par lien sériel versus 1500 fois par seconde selon la

norme). Nonobstant les limitations de l'appareil, cette technique offre une alternative intéressante au déploiement d'un lien radio bidirectionnel complet.

Une dernière application plausible pour l'hôte touche les aspects de recherche et de cosimulation. Pour plusieurs applications, des couches d'abstraction du matériel peuvent établir un lien entre les différents points de données du récepteur et les logiciels de développement, utilisés habituellement l'un à la suite de l'autre. Ces aspects font l'objet des sections qui suivent.

5.6 Cosimulation

La référence originale du récepteur consiste en un ensemble de scripts rédigés sous Matlab et ayant été éprouvés à l'aide d'enregistrements de canaux cellulaires réels. Ce modèle, appelé *référence d'or* (« golden reference »), repose cependant sur l'hypothèse que le l'évaluation de la structure du canal peut s'effectuer dans un temps inférieur à la période de réception d'un symbole, T_S. Cette vision est optimiste; elle équivaut à offrir un système où $n_{STRF} = 1$. De plus, ce script repose sur un traitement entièrement effectué en virgule flottante. Tel qu'expliqué aux chapitres précédents, ces deux items entraînent des contraintes irréalistes au niveau matériel.

Afin d'utiliser le modèle Matlab existant à des fins de comparaison avec le système développé, la référence d'or doit subir deux modifications majeures. D'abord, on en quantifie les variables d'état selon les tailles définies à la section 3.4. Ensuite, le modèle fonctionnel est scindé en deux modules, transmission et réception, afin de pouvoir contrôler la taille des salves de transmission et refléter plus fidèlement le comportement matériel du récepteur.

Pour concrétiser la seconde modification, des fichiers de sauvegarde temporaires conservent plusieurs variables d'état de la simulation Matlab, ainsi que le contenu des vecteurs de transmission et de réception. Ces fichiers permettent en premier lieu l'échange

entre les deux nouvelles sections du modèle scindé, et facilitent l'interaction avec d'autres modèles. Ces fichiers augmentent rapidement de taille, et leur maniement devient lourd au cours d'un long épisode de simulation. La longueur de la simulation est, à la limite, bornée par la taille mémoire du système sur lequel elle se déroule. En pratique, on préfère se limiter à moins de 20 000 symboles, ce qui correspond en réalité à une fraction de seconde d'une communication cellulaire de facteur d'étalement $L = 32$. Le temps requis pour la simulation est, dans ce cas, de l'ordre d'une dizaine de minutes sur un PC de développement (Pentium 4 cadencé à 2,0GHz avec technologie Hyperthreading, 512Mo de mémoire vive RAMBUS 400MHz, système d'exploitation Linux 2.4.21 et simulateur Octave 2.1.49 compilé avec les optimisations pour Pentium 4 de gcc 3.2.3).

La convergence de l'algorithme scindé, malgré la quantification et l'abaissement du taux d'évaluation du canal, demeure ainsi intacte, et constitue la nouvelle « référence quantifiée. »

5.6.1 Modèle VHDL

Les régions STRF et SP sont disponibles sous forme de code source VHDL. Pour les accompagner, un modèle virtuel du Microblaze est réalisé à même le banc de test (« testbench ») VHDL englobant. Ce dernier effectue toutes les tâches que l'on retrouve à la figure 47 (l'algorithme *analyze*, entre autres, est en réalité exécuté en instructions VHDL non synthétisables), et substitue les échantillons prélevés en temps normal d'un CAN par des vecteurs stockés sur disque, provenant des fichiers d'état temporaires mentionnés plus haut. Pour compléter la boucle de simulation, le banc de test consigne les valeurs de plusieurs variables d'état (les séquences d'écriture des mémoires dédiées) dans des fichiers similaires à ceux du modèle de référence.

5.6.2 Lien avec le simulateur

La relation entre le modèle de référence piloté par Matlab et le simulateur ModelSim requiert un gestionnaire externe qui puisse démarrer les outils de développement et en récupérer les résultats afin de constituer une boucle de rétroaction à l'algorithme. Un schéma-bloc de ce système apparaît à la figure 48.

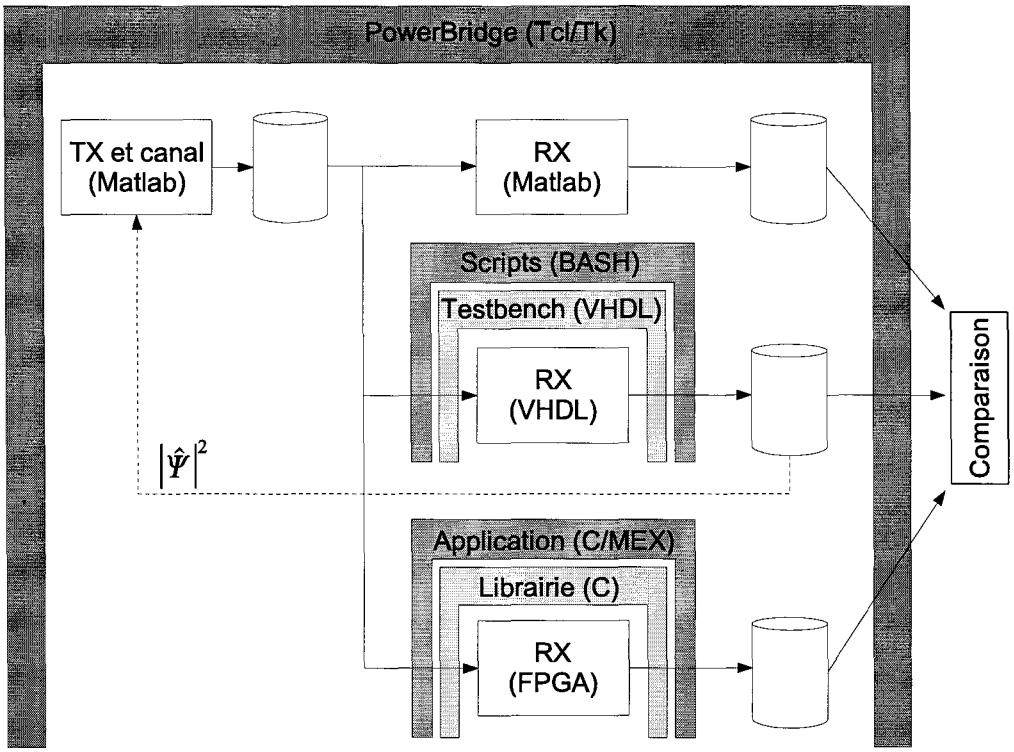


Figure 48 Schéma-bloc du système de cosimulation

L'élément de contrôle global, PowerBridge, est un script rédigé en langage Tcl/Tk (Tcl, 2005) et muni d'une interface graphique permettant le contrôle de certains paramètres de simulation (voir annexe 2). Le choix de ce langage est naturel puisque l'interface graphique de ModelSim l'utilise; les interactions s'en trouvent simplifiées par l'utilisation de sémaphores de communication logicielles qui dépassent le cadre de ce texte. PowerBridge peut ainsi prendre le contrôle total du logiciel de simulation ModelSim. Il devient facile, dès lors, de démarrer et d'interrompre la simulation, de consulter l'état de

certains signaux ou variables du modèle VHDL simulé, et même d'en changer le contenu. De plus, le moteur Tcl/Tk offre toute la maniabilité d'un langage de programmation classique, dont la possibilité de faire appel au système d'exploitation ainsi qu'à d'autres logiciels.

La collaboration entre PowerBridge et les différents outils de conception permet le fonctionnement parallèle de deux modèles du récepteur, tel qu'illustré à la figure 48. Certaines tâches secondaires sont automatisées par l'entremise de scripts (exécutés par l'interpréteur BASH) mais n'effectuent pour la plupart que de simples manipulations de fichiers (concaténation, mise en forme, tri).

La rétroaction vers le transmetteur peut s'effectuer à partir de l'un ou l'autre des deux modèles; on choisit d'utiliser les résultats du modèle VHDL. Une simulation typique s'effectue au taux d'étalement $L = 32$ (correspondant à une transmission de 120 kilobits/s) avec rétroaction en puissance à tous les 75 symboles (donc 1600 fois par secondes). La plate-forme de simulation est constituée du même PC de développement décrit plus haut. Les logiciels Modelsim et Tcl/Tk sont respectivement les versions 5.7f et 8.5.3. Dans ces circonstances, une simulation de 20 000 bits (ou 640 000 bribes) requiert tout près de 30 heures. Par profilage logiciel, il a été déterminé que près de 20% de ce temps était consumé pour la manipulation des fichiers d'état, probablement à cause de leur taille qui peut atteindre plusieurs dizaines de méga-octets.

5.6.3 Simulation avec matériel

L'inclusion de la version physique (FPGA) dans le même cadre de simulation est possible grâce à la modification des couches d'adaptation et la permutation du modèle du Microblaze par sa contrepartie réelle. Le lien avec le logiciel de simulation est scindé en deux couches logicielles: la première, propre à l'hôte, est fournie par le constructeur de la carte FPGA et permet un accès de bas niveau aux différents registres sur le bus PCI, alors que la seconde, réalisée à l'interne, établit le lien entre cette librairie et le logiciel Matlab.

Cette seconde couche est développée en langage C et est prévue pour deux types de compilateurs différents: un premier capable de créer des exécutables autonomes, et un second provenant du logiciel Matlab, permettant la création de fichiers MEX, exécutables uniquement par Matlab, mais permettant un accès direct à son espace de travail (« workspace »).

L'utilisation de fichiers MEX est le choix par excellence pour concrétiser le modèle de la figure 48. La mise en oeuvre de cet empilement logiciel montre cependant que le temps de simulation s'écoule en majeure partie lors des transactions qu'effectue le programme MEX avec l'espace de travail de Matlab. Bien que les échanges de données soit regroupés en salves, ce délai empêche une réelle cosimulation avec matériel en temps réel. Pour des simulations de courte durée, cependant, cette solution est idéale. La réception et le traitement de 20 000 symboles, dans ce cas, nécessitent environ 30 secondes.

5.7 Interface de visualisation interactive

Le même logiciel de contrôle, dans sa version autonome, permet d'atteindre un taux d'extraction de données de loin supérieur. L'utilisation d'un engin graphique tridimensionnel de type OpenGL (accélération graphique matérielle assurée par la carte graphique de l'hôte) assure une visualisation fluide des données à certains endroits choisis du pipeline STRF. Cet affichage empêche cependant l'utilisation de vecteurs de tests générés; il est plutôt destiné à l'observation qualitative du récepteur dans le cadre d'un fonctionnement en temps réel. La figure 49 montre cette interface graphique d'affichage et de contrôle. L'annexe 4 relate un événement public où elle fût employée.

5.7.1 Points de visualisation

Le premier élément d'intérêt sur la figure 49 est la nappe tridimensionnelle bleue qui représente, dans ce cas-ci, la valeur absolue de la réalisation de $\underline{\hat{H}}_f$ à $\underline{\hat{H}}_{f-50}$, où f représente un instant d'observation par le Microblaze. En premier plan, une série de

bâtonnets blancs imitent un graphique d'impulsions (*stem* sous Matlab), et montre avec clarté les valeurs absolues des L_{Δ} éléments de $\underline{\hat{H}}_f$. Ils permettent en outre d'observer et de comprendre le déplacement de $h_{RRC}(t)$ entre ces L_{Δ} échantillons.

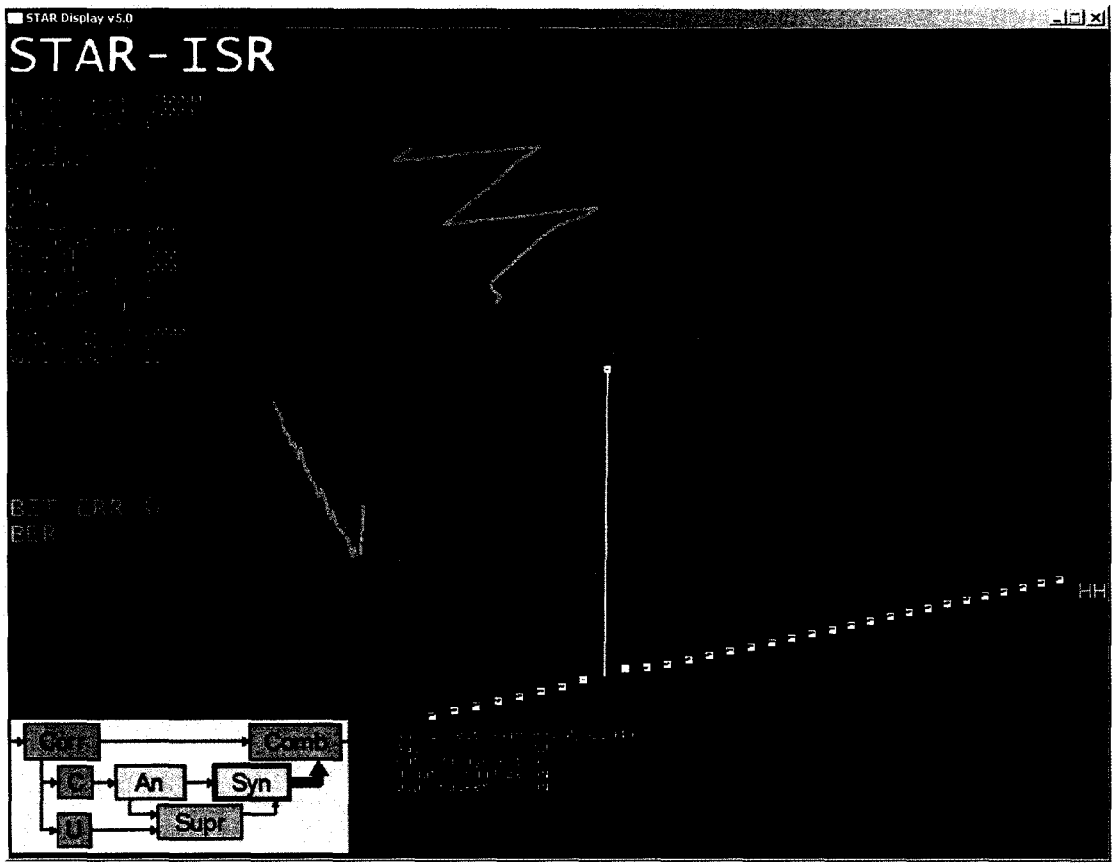


Figure 49 Interface de visualisation interactive pour STAR

Le plan supérieur représente l'état du vecteur $\hat{\tau}_f$ à $\hat{\tau}_{f-50}$ pour les P trajets connus. Puisque le nombre de trajets varie constamment, les lignes bleue, verte, rouge, jaune et mauve qui représentent les trajets $P = [0 \cdots P_{MAX}]$ apparaissent et disparaissent. Sur la surface translucide apparaît un quadrillé gris qui représente le temps dans un axe et les index de bribes dans l'autre. À noter que deux lignes de couleur saumon indiquent la position des limites haute (regHighGate) et basse (regLowGate) du module DCDT; on voit d'ailleurs à

la figure 49 que l'ensemble des trajets sont repositionnés au centre de la grille dès que le premier trajet ($P = 0$, en bleu) atteint l'une des limites.

Le plan vertical à la gauche de la nappe donne la norme de chaque colonne de $\hat{\boldsymbol{J}}_f$ à $\hat{\boldsymbol{J}}_{f-50}$. Les couleurs des différents traits correspondent au même code employé par $\hat{\tau}_f$ sur le plan supérieur.

Cet agencement tridimensionnel des plans autour de la nappe permet de visualiser que $\underline{\hat{H}}_f$ n'est bel et bien qu'une synthèse de $\hat{\boldsymbol{J}}_f$ et $\hat{\tau}_f$. On peut s'en convaincre en déplaçant le point de vue au zénith, d'où l'on peut observer la superposition des traces du plan supérieur sur les arêtes de la nappe tridimensionnelle. Il en va de même pour les traces du plan $\hat{\boldsymbol{J}}_f$ qui se superposent aux sommets de $\underline{\hat{H}}_f$ lorsqu'on déplace le point de vue vers la gauche.

Le schéma dans le coin inférieur gauche de l'application donne une représentation simplifiée du récepteur STAR dans laquelle le module duquel on observe la sortie sur la nappe tridimensionnelle est entouré d'un cadre bleu.

Enfin, le taux d'erreur binaire, tel que déterminé par un appareil dédié (Fireberd 6000A), est rapporté en direct à l'écran (« BIT ERR 0 »). Le port parallèle de cet appareil est simplement activé et relié au PC qui interprète les messages normalement destinés à une imprimante et affiche périodiquement les résultats. L'appareil de mesure fait cette détermination en analysant le train binaire reçu qui consiste en une séquence pseudo-aléatoire périodique PN-23 entachée du bruit de transmission et des imperfections du récepteur.

5.7.1.1 Taux de rafraîchissement

La vitesse à laquelle peuvent défiler les différentes données à l'écran dépend à la fois du taux de lecture maximal atteignable entre les mémoires internes et l'hôte, des

manipulations mathématiques faites par le logiciel de visualisation, et du traitement graphique effectué par le processeur de l'adaptateur graphique du PC. On estime cette vitesse en nombre d'images par seconde (« frames per second » ou *fps*). Un profilage sommaire du code montre que la majorité du temps de traitement est consommé par la carte graphique. Les performances obtenues sur la plate-forme de développement ICS (deux Pentium IV Xeon, 2Mo cache, 2,5Go mémoire vive DDR266, PCI-X/66MHz) équipée d'une carte graphique ATI Radeon 7000 sur bus PCI sont de 24 fps à une résolution de 1024x768 à 32 bits par pixel (nécessaires pour l'effet de transparence). Le passage à un autre PC comparable en puissance de calcul et en mémoire mais doté d'un adaptateur graphique nVidia Quattro Pro IV sur bus AGP 8x a permis d'atteindre une performance de 75 fps, ce qui indique que le lien limitant est actuellement l'adaptateur graphique et non le logiciel, le Microblaze ou le bus PCI-X.

Au taux de 75 fps, on extrait une nouvelle tranche de la nappe tridimensionnelle (soit 32 échantillons complexes), ainsi que les vecteurs $\hat{\tau}$ et $\hat{\boldsymbol{J}}$, pour un total de 42 échantillons, à tous les 1/75e de seconde. Pour bénéficier des transferts rapides DMA, on doit néanmoins lire 32 entiers de 32 bits par transaction, ce qui totalise 96 entiers de 32 bits, ou 384 octets. En termes de bande passante, on déduit un taux de transfert de 28800 octets/s pour satisfaire l'adaptateur graphique le plus rapide. Ce taux est néanmoins si bas que même un lien Ethernet de faible débit pourrait le soutenir, voire même à travers un réseau IP, ce qui rend possible la surveillance à distance du récepteur avec les mêmes performances de visualisation. Du point de vue du récepteur, à 120 000 symboles/s, on parvient à visualiser une donnée à chaque 1600 symboles.

5.7.2 Points de contrôle

Le second volet du logiciel permet la manipulation de certains paramètres du récepteur. Par l'entremise du menu disposé à gauche sur la figure 49, il est possible de modifier en direct les niveaux d'apparition et de disparition des trajets (υ_A et υ_V), leur niveau d'hystérésis (n_A et n_V), les paramètres n_{STRF} et n_{PM}, les pas d'adaptation des modules CDFI

et UDFI (μ et ζ), les limites haute et basse et la cible du module DCDT (regHighGate et regLowGate), et les paramètres d'un amplificateur à gain asservi (« Automatic Gain Control » ou AGC) qui précède STAR dans la plate-forme de développement pour simuler le contrôle de puissance en boucle fermée dans cette première version du prototype.

L'interface offre en outre la possibilité de mettre en service ou hors-circuit la normalisation de $\hat{\boldsymbol{H}}$, le module DCDT, l'AGC et la région PM dans son ensemble.

5.8 Conclusion

Ce chapitre a couvert la seconde et dernière partie de la réalisation du récepteur STAR. En guise de mise en contexte, un schéma de l'assemblage du récepteur en relation avec la plate-forme de développement ICS telle que présentée au chapitre 4 a été présenté à la première section. Pour formaliser ces liens, les sémaphores de communications entre les différents modules de ce système global ont été établies, mettant l'emphase sur les signaux de contrôle entre le Microblaze et le contrôleur SCU, dont la réalisation a été présentée au chapitre 4. Le micrologiciel qui anime le Microblaze a ensuite fait l'objet d'un examen détaillé, à la fois pour sa tâche de gestion des trajets décrite aux chapitres 2 et 3, mais aussi pour son rôle dans le démarrage du système.

La validation du récepteur a ensuite été abordée en démontrant une architecture unifiée de test qui est utilisée pour les différents stades de la conception, et qui permet le suivi de la dégradation des performances depuis le modèle de référence jusqu'au prototype sur FPGA. Cette couche d'abstraction a ensuite permis le développement d'une interface de visualisation, soulignant la capacité de STAR a effectuer des tâches de mesure de canal. Cette dernière a été mise à l'épreuve devant public, tel que rapporté à l'annexe 4.

Alors que la réalisation du récepteur se conclue ici, ses performances doivent encore être validées en fonction des résultats du modèle de référence. Le chapitre 6 porte sur la

validation du prototype à l'aide de différents indicateurs de performance, et offre une analyse d'utilisation des ressources physiques par le récepteur.

CHAPITRE 6

RÉSULTATS

6.1 Introduction

L'élaboration du récepteur STAR ne couvre qu'une partie du travail de prototypage. Les chapitres précédents ont porté respectivement sur le contenu théorique de l'algorithme, l'architecture du récepteur, et la réalisation matérielle et logicielle. La seconde partie du chapitre 5 présentait les mécanismes prévus pour valider les performances fonctionnelles du récepteur.

Le chapitre 6 couvre deux grands thèmes. Le premier concerne la validation, à l'aide de trois critères documentés, des performances du récepteur. Le modèle de référence, le modèle quantifié et le prototype final sont mis à l'épreuve à tour de rôle avec des conditions idéales afin d'évaluer la dégradation des performances due au passage au monde matériel. Une fois cet effet mesuré, la rapidité de fonctionnement du prototype est mise à profit pour étudier l'impact des taux de traitement réduits n_{STRF} et n_{PM} sur les mêmes indicateurs de performance.

Le second thème du chapitre touche l'analyse physique du récepteur. En particulier, l'effet de la quantification sur la taille du module matrixMult et de son sous-module basicMac est examiné pour plusieurs degrés de quantification. Enfin, une analyse de la performance de l'architecture, en termes d'opération/s par ressource physique, indique les modules susceptibles d'être révisés lors de développements futurs. Ces résultats permettent d'entrevoir le récepteur dans un contexte multi-usagers et multi-antennes.

Les résultats présentés dans ce chapitre proviennent de l'analyse de données brutes produites à l'aide du prototype sur FPGA du récepteur STAR. Puisqu'il fonctionne à un taux correspondant au temps réel (0,2 s traitée en 0,2 s), ce prototype aura permis le traitement d'une importante quantité de données, beaucoup plus que ne l'aurait permis le

modèle de simulation (0,2 s traitée en 30 h) dans un temps acceptable. Il s'agit du même prototype présenté devant public dans le cadre de la démonstration PROMPT-Québec (annexe 4).

6.2 Performances du récepteur

La série de tests présentés dans cette section fait abstraction de certains facteurs qui influencent la réception du signal, tels que la synchronisation au transmetteur, la saturation du convertisseur analogique/numérique à l'entrée, etc. Une première série de tests a pour but de mesurer l'impact de la quantification des opérandes, et une seconde, la fidélité du prototype sur FPGA au modèle de référence.

Trois critères servent à mesurer la performance à chaque étape. Il s'agit du taux d'erreur binaire non-codé (« uncoded bit error rate » ou BER non-codé), de l'erreur quadratique moyenne (ou EQM) entre les P trajets perçus $\hat{\tau}_{n,p}$ et leur position réelle $\tau_{n,p}$, et le ratio de temps (« tracking ratio » ou TR) où ces trajets sont effectivement perçus. Cette évaluation correspond à la méthodologie utilisée par Affes (1997) et Cheikhrouhou (2001).

6.2.1 Méthodologie

Le travail qui mène à la réalisation du prototype peut être divisé en quatre étapes: l'évaluation du modèle de référence (« golden reference »), la détermination d'un niveau de quantification, la réalisation et la simulation en VHDL, et l'intégration codesign sur FPGA de toutes les composantes du récepteur.

Ces quatre stades de développement ont chacun leurs particularités lors du processus de simulation. Certains aspects se prêtent mal à une comparaison directe; par exemple, le gestionnaire de trajets subit des modifications à chaque étape et est difficilement évaluable. C'est pourquoi seuls les résultats issus de deux variables, soit $\hat{\tau}$ et $\hat{s}$, sont considérés dans ce rapport (l'EQM et le TR sont issus de la première, et le BER, de la

seconde). On parvient ainsi à mesurer la dégradation des performances au fur et à mesure que l'on progresse vers le prototype. La comparaison des opérandes intermédiaires, plus utile au processus de déverminage, se prête mal à de longues comparaisons.

Pour mesurer la dégradation imputable à la quantification des opérandes, le script de référence est modifié de manière à ce que chaque variable soit arrondie proportionnellement à la quantification choisie pour sa contrepartie dans le prototype. En pratique, suivant chaque opération mathématique, une opérande subit

$$var_q = \frac{\text{round}\left(var \cdot 2^{quant}\right)}{2^{quant}}, \tag{6.1}$$

où q représente le nombre de bits alloué à la partie fractionnaire de var_q , et round( ) est l'opérateur d'arrondissement. Cette procédure ne parvient en fait qu'à imiter une portion des effets de la quantification. Les fonctions Matlab utilisées par ce test effectuent toujours les calculs par virgule flottante à l'interne; chaque étage de la transformée rapide de Fourier, par exemple, emploie des coefficients de papillons (« twiddle factors ») exprimés en virgule flottante, ce qui empêche dans une certaine mesure les débordements internes. Bien que cette simulation ne reproduise pas exactement le comportement ultérieur du modèle VHDL, elle permet néanmoins de jauger en première instance l'effet de la quantification sur le comportement global du récepteur. Elle ne peut garantir, à elle seule, l'élimination de tous les problèmes de dépassement et soupassement.

Tel que mentionné à la section 5.6, les simulations de la référence et du modèle quantifié (scripts Matlab) sur un PC de développement ne requièrent que quelques minutes pour 30 000 symboles, alors que la simulation du modèle VHDL sur ce même PC nécessite près de 30 heures pour 20 000 symboles. Le test du prototype sur FPGA, cependant, permet des simulations beaucoup plus rapides. Puisque le modèle VHDL simulé et sa réalisation sur FPGA sont en principe identiques, on préfère effectuer les tests de validation avec le prototype uniquement.

Pour assurer la cohérence des résultats, le test de chaque modèle (référence, quantifié, prototype) s'effectue sur une plage de rapports signal-à-bruit (SNR) variant de 1 à 10 dB. Chacun des dix tests utilise le même message binaire et la même réalisation du canal de Rayleigh (matrice de propagation du canal). Les trajets présents dans le canal sont au nombre de trois ($P = 3$), le facteur d'étalement demeure $L = L_{\Delta} = 32$, et le nombre d'antennes est fixe à $M = 1$. Cette réalisation du canal est identique à celle utilisée par Affes (1998) où l'étalement Doppler est de 10 Hz, la vitesse véhiculaire est de 5 km/h, la porteuse est fixée à 1,9 GHz et ou les évanouissement sur les trois trajets surviennent à une cadence aléatoire (mais dans la zone générale de $T_{FADING} \approx 1$ ms) et sont indépendants les uns des autres.

Cette démarche de comparaison serait invalide pour mesurer les performances du récepteur en absolu puisque l'élément stochastique du test est absent. Puisque cette évaluation cherche plutôt à jauger la dégradation des performances due à la réalisation sur FPGA, la réutilisation de vecteurs de tests identiques est valable. On obtient donc une évaluation relative des performances du prototype.

De plus, les variables intermédiaires du récepteur (modèle quantifié et prototype) sont initialisées aux valeurs de leur contrepartie du modèle de référence, tout juste après la période initiale d'acquisition des trajets. On s'assure ainsi de démarrer chaque modèle dans un état identique, et donc d'observer les divergences dans la poursuite des trajets plutôt que les performances de détection (« acquisition »).

Des trois critères d'évaluation employés, le taux d'erreur binaire nécessite une attention particulière. Nonobstant l'évaluation relative (et non absolue) des performances, les simulations doivent s'échelonner sur un nombre suffisant de symboles pour que la valeur de BER non-codé observée soit statistiquement acceptable. Le nombre de symboles transmis doit être supérieur d'au moins un ordre de grandeur à l'inverse du taux d'erreur anticipé. Par exemple, une valeur de BER de 10^{-3} est statistiquement acceptable lorsqu'elle provient d'une simulation de 10^{4} symboles. Puisque le taux d'erreur anticipé peut glisser

sous 10^{-3} selon Affes (1996), on prévoit des simulations légèrement plus longues, soit 3×10^4 symboles.

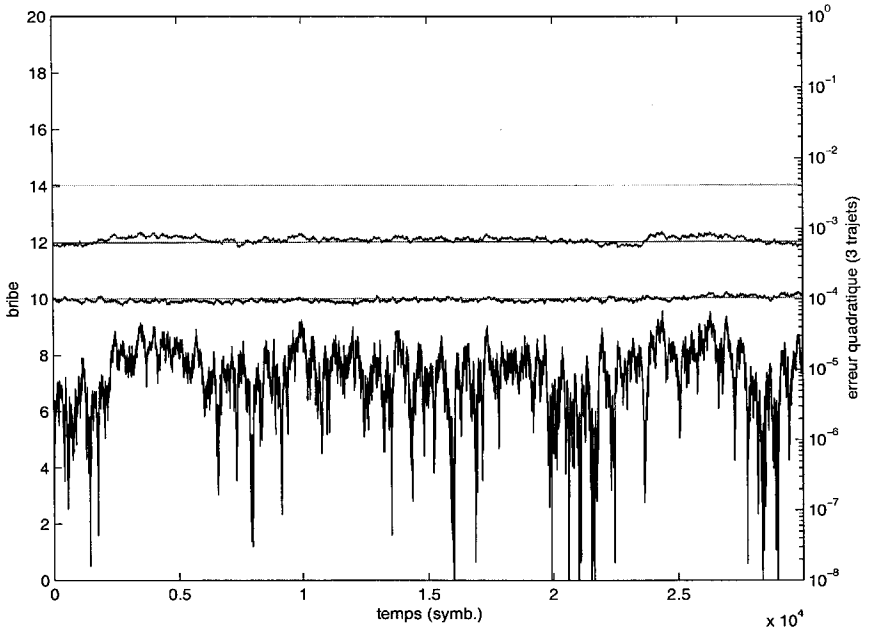
6.2.2 Impact de la quantification

La première salve de tests vise à mesurer l'impact de la quantification sur les performances du récepteur. Les figures 50, 51 et 52 présentent les résultats de tests effectués sur le modèle de référence, le modèle quantifié et le prototype matériel, respectivement, à (a) SNR = 1 dB et (b) SNR = 10 dB. Les performances de poursuite sont illustrées par la superposition des trois trajets perçus aux bribes 10, 12 et 14 (ordonnée de gauche) et de leur position réelle (progression linéaire). La courbe inférieure donne l'erreur quadratique ponctuelle de l'estimation en considérant les trois trajets,

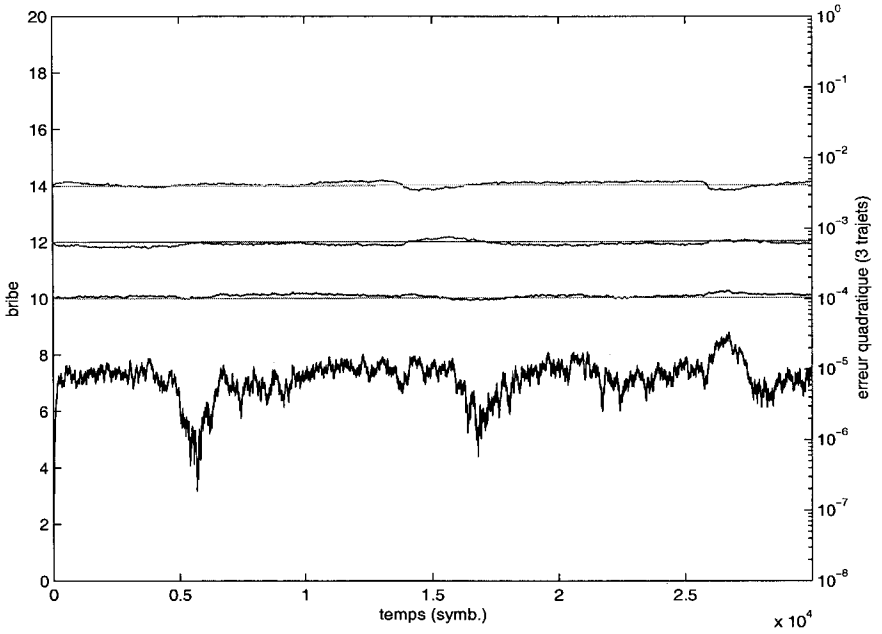
$$e.q.[n] = \frac{1}{P_n} \sum_{p=1}^{P_n} \left| \hat{\tau}_{p,n} - \tau_{p,n} \right|^2, \tag{6.2}$$

conformément à la procédure utilisée par les articles publiés sur STAR (Affes, 1997). Cette valeur est exprimée en décibels et rattachée à l'ordonnée de droite. Ces simulations sont effectuées dans le cas idéal où chaque symbole entraîne l'estimation du canal (n_{STRF}=1) et déclenche la gestion des trajets (n_{PM} = 1). Les paramètres utilisés sont μ = 8/128, ζ = 6/128, ξ = 4/16, υ_A = 0,4, υ_V = 0,3, n_A = 80 et n_V = 80.

L'EQM résultante pour le modèle de référence est de -49,50 dB (SNR = 1 dB) et -50,20 dB (SNR = 10 dB). Elle se situe à des niveaux similaires pour le modèle quantifié, soit -44,56 et -50,96 dB, respectivement. Au niveau de la poursuite des trajets, la quantification n'entraîne donc qu'une dégradation de 5 dB dans le pire cas (SNR = 1dB). Les valeurs d'erreurs quadratiques sont cependant réparties dans la plage -50 dB à -56 dB pour le modèle de référence, et -45 dB à -55 dB pour le modèle quantifié, et semblent non-corrélées au niveau SNR.

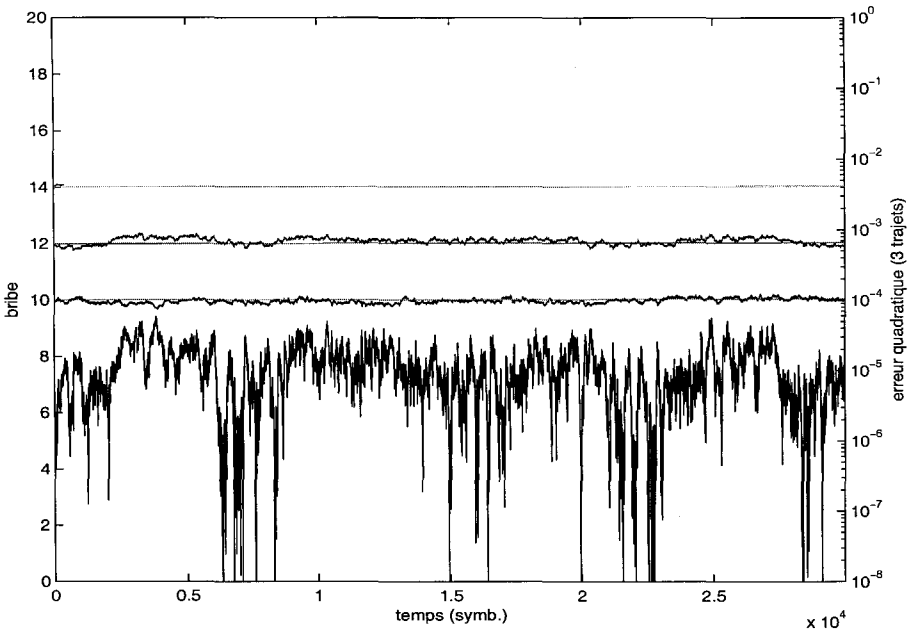


(a) SNR = 1 dB

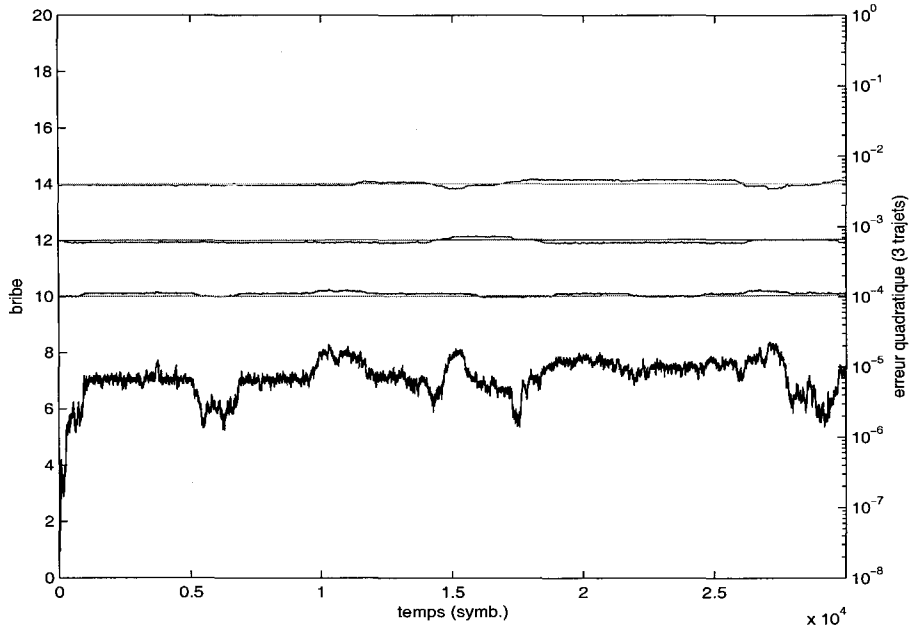


(b) SNR = 10 dB

Figure 50 Trajets perçus et réels, et e.q. du modèle de référence

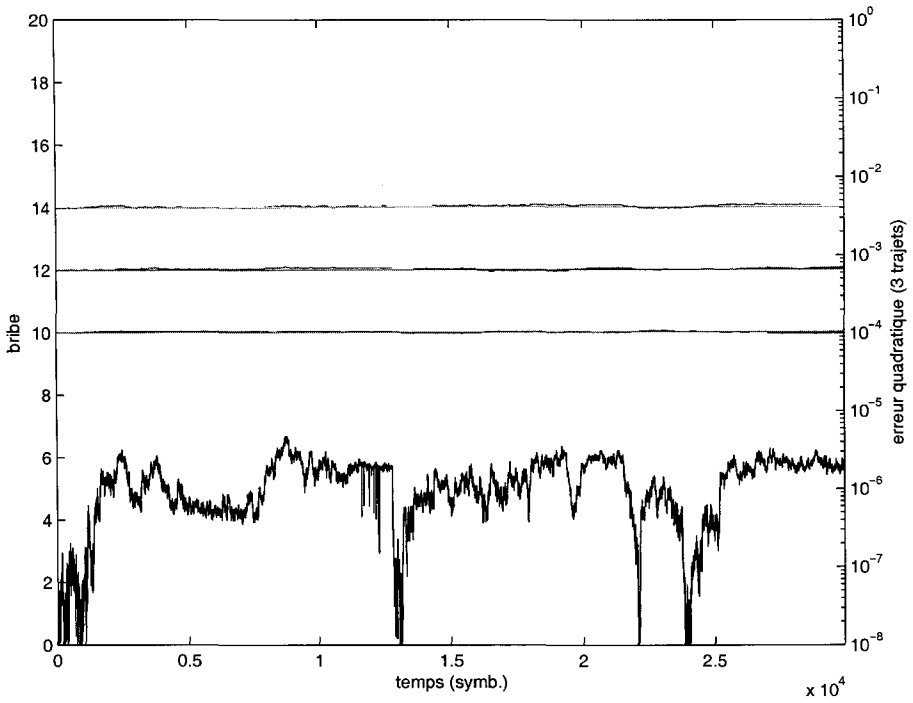


(a) SNR = 1 dB

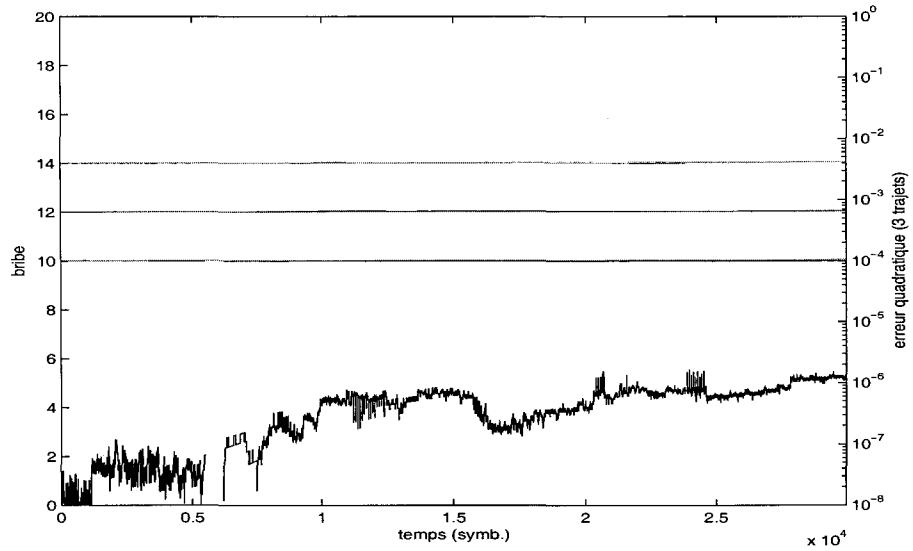


(b) SNR = 10 dB

Figure 51 Trajets perçus et réels, et e.q. du modèle quantifié



(a) SNR = 1 dB



(b) SNR = 10 dB

Figure 52 Trajets perçus et réels, et e.q. du prototype sur FPGA

Dans le cas du prototype sur FPGA, la plage d'erreurs quadratiques se situe plutôt entre -58 dB et -64 dB, pour les mêmes conditions, ce qui constitue une augmentation des performances. Bien qu'elle ne soit que modeste, cette amélioration est probablement attribuable à la sensibilité réduite du prototype aux bruits introduits par le canal; la régression linéaire effectuée par le module TDU (section) est tributaire de tables pré-numérisées, contrairement aux modèles de référence et quantifié. Cette perte de précision à la détection de phase se traduit par une correction moins rapide sur $\hat{\tau}$, et en contrepartie, une meilleure EQM dans le cas de ce canal en particulier, où les trajets évoluent lentement et linéairement. Il est probable qu'un mauvais canal urbain comme le COST-207 « bad urban » (COST, 1989) entraînerait plutôt une augmentation de l'erreur quadratique dans le cas du prototype FPGA.

Le taux de poursuite du modèle de référence est de 0,66 (SNR = 1 dB) et 1,00 (SNR = 10 dB), alors qu'il est de 0,59 (SNR = 1 dB) et 1,00 (SNR = 10 dB) pour le modèle quantifié. Conséquence de l'EQM citée plus haut, les résultats du prototype sont différents: 0,95 (SNR = 1 dB) et 0,80 (SNR = 10 dB).

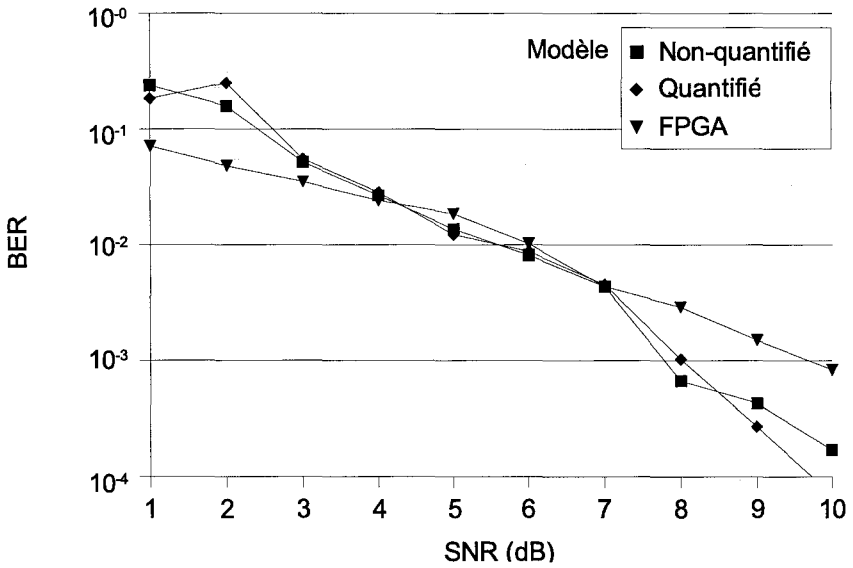


Figure 53 BER sans codage des modèles de référence, quantifié, et du prototype

Le critère de BER dépeint un portrait mixte. La figure 53 indique clairement que les modèles de référence et quantifié sont à toutes fins pratiques identiques en taux d'erreur binaire. Cependant, le prototype FPGA offre de meilleurs résultats à SNR faibles, et de moins bonnes performances à SNR plus élevés.

On remarque au premier examen de la figure 53 que la courbe intitulée « FPGA » représentant la version entièrement quantifiée de STAR ne correspond pas exactement à la version quantifiée de l'algorithme exécutée sous Matlab. Cette divergence peut s'expliquer en partie par l'impossibilité de quantifier les étapes internes de l'exécution de la transformée de Fourier rapide de Matlab, qui s'exécutera en arithmétique à point flottant. Un autre point de divergence potentiel point critique est l'emploi d'une quantification non-linéaire sur l'inverse tangente du prototype. Une inversion matricielle qui se réduit approximativement à la matrice identité dans la résolution de $\hat{\boldsymbol{J}}_{n+1}$ est aussi retirée de la version FPGA, alors qu'elle est conservée sur les deux version Matlab. Enfin, le module PM, bien que sa fonctionnalité soit semblable, diffère légèrement dans sa réalisation et sa latence d'opération dans sa forme finale au sein du microprocesseur Microblaze. Tous ces facteurs entrent en jeux aux extrémités des courbes de la figure 53, soit dans les zones d'utilisation très bruitée et très peu bruitée, respectivement.

6.2.3 Impact de n_{STRF} et n_{PM}

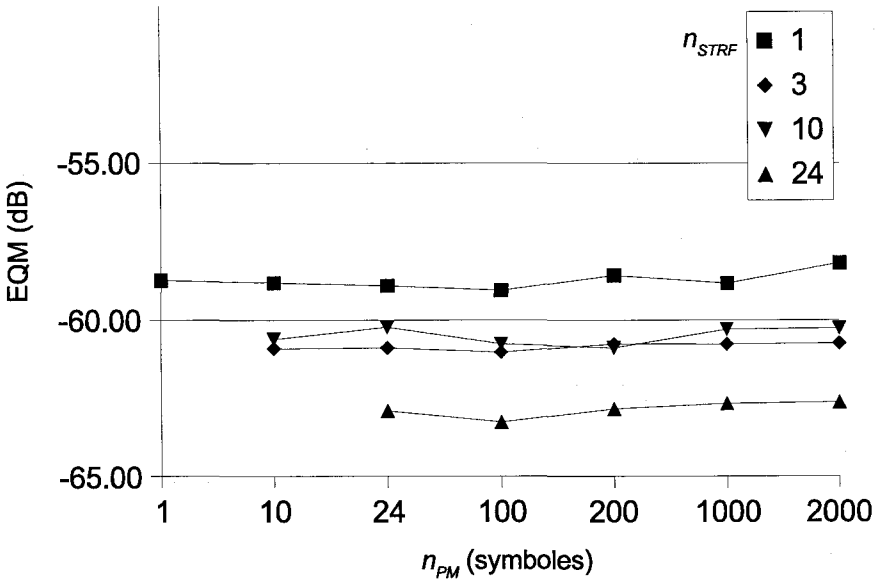
La validité du prototype par rapport au modèle de référence original étant démontrée, on passe à la mesure, à l'aide du prototype, de l'impact des facteurs de traitement réduit n_{STRF} et n_{PM} sur les trois critères d'évaluation. Afin d'illustrer leur effet, les figures de cette section illustrent à tour de rôle la variation d'un critère en fonction du paramètre n_{PM}, et ce, sur quatre courbes représentant $n_{STRF} = 1, 3, 10$ et 24. À noter que le paramètre n_{PM} est ici relatif aux symboles et non à n_{STRF}; $n_{PM} = 1$ implique un déclenchement du gestionnaire de trajet à tous les symboles, et non pas à tous les n_{STRF} symboles. Pour cette raison, les points où n_{PM} aurait été inférieur à n_{STRF} sont absents (il est inutile d'utiliser le gestionnaire de trajet plus souvent que les trajets ne sont mis à jour).

6.2.3.1 Erreur quadratique

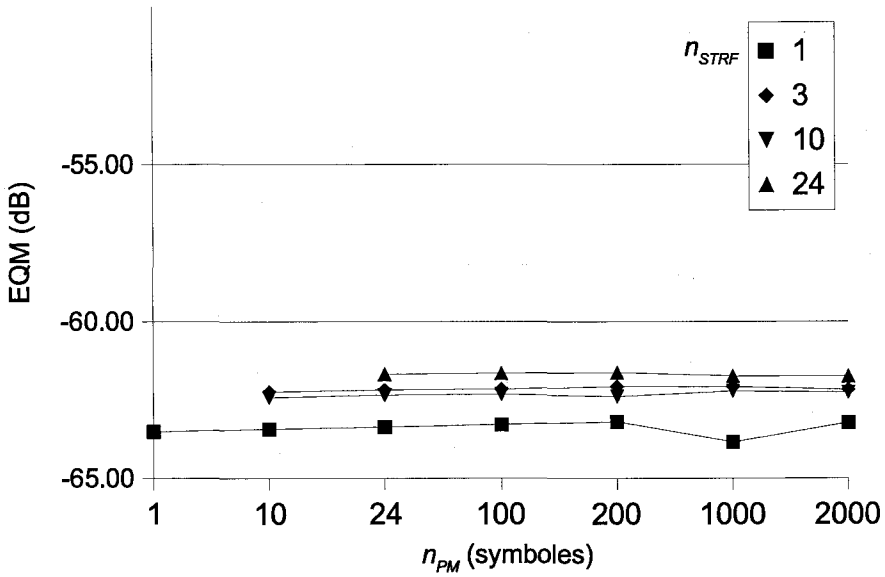
La figure 54 montre l'effet des paramètres n_{PM} et n_{STRF} sur l'EQM de l'estimation des trajets pour les cas (a) SNR = 1 dB et (b) SNR = 10 dB. Une inspection visuelle confirme que n_{PM} n'entraîne qu'une très légère dégradation de l'EQM (moins d'un décibel) dans la plage couvrant $n_{PM} = 1$ à $n_{PM} = 2000$. On peut donc considérer son effet comme négligeable. L'impact de n_{STRF} est cependant plus tangible, et tel qu'escompté, l'EQM augmente avec n_{STRF} croissant dans le cas SNR = 10 dB. Elle demeure cependant meilleure que -58 dB, ce qui constitue toujours une amélioration face aux modèles de référence et quantifié à la section 6.2.2. L'ordre des courbes (n_{STRF}) de la figure 54 (a) (SNR = 1 dB) est cependant inverse à l'ordre attendu. Il s'agit d'une situation où le niveau de bruit permet vraisemblablement aux estimations plus rares ($n_{STRF} = 24$) d'obtenir de meilleures performances pour ce canal, puisque le faible taux de mise à jour de $\hat{\tau}$ ne permet pas au récepteur de se laisser induire en erreur par le bruit blanc additif. On observe visuellement ce phénomène en comparant les figures 50 et 51 à la figure 52.

6.2.3.2 Taux de poursuite

L'effet de n_{PM} et n_{STRF} sur le taux de poursuite est illustré à la figure 55 pour des niveaux de SNR 1 dB (a) et 10 dB (b). De façon générale, les courbes démontrent un minimum près de 0,80 pour des taux n_{PM} variant entre 24 et 200. Les performances doivent, en principe, diminuer lorsque n_{PM} augmente de manière significative, mais la figure 55 indique une augmentation du taux de poursuite lorsque $n_{PM} > 200$. Cet artefact découle en fait de la lenteur du gestionnaire de trajet à enregistrer les modifications du profil multi-trajets du canal. Lorsque $n_{PM} = 2000$, le délai entre deux observations est de 16,66 ms, beaucoup plus que le temps de cohérence du canal, peu importe les paramètres du modèle de Rayleigh employé. Les résultats de la figure 55 témoignent simplement de l'incapacité du récepteur, à des taux n_{PM} trop faibles, de percevoir les événements d'ajout et de retrait des trajets. Incidemment, les trajets connus demeurent dans le système même lors d'évanouissements, ce qui laisse faussement croire à un meilleur taux de poursuite, et

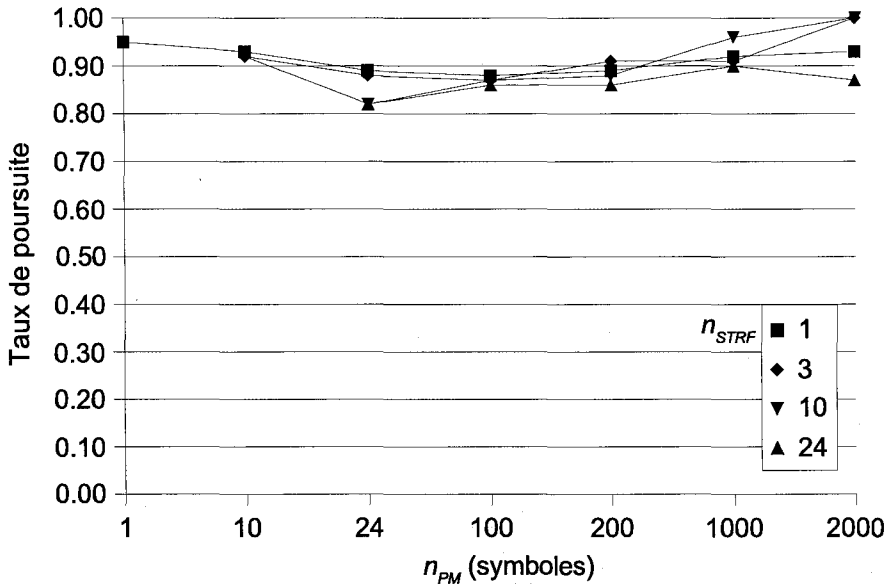


(a) SNR = 1 dB

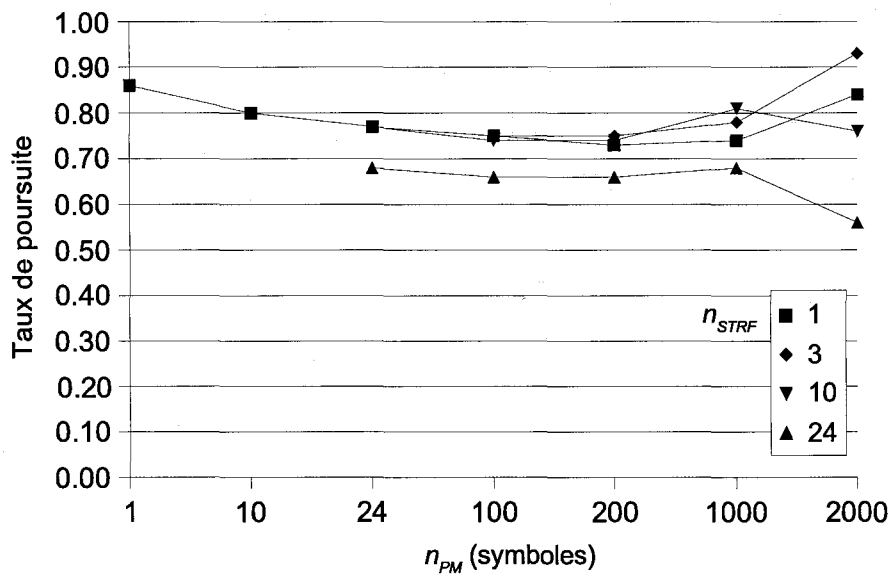


(b) SNR = 10 dB

Figure 54 EQM en fonction du taux de traitement réduit



(a) SNR = 1 dB



(b) SNR = 10 dB

Figure 55 Taux de poursuite en fonction du taux de traitement réduit

explique le rehaussement du taux de poursuite pour $n_{PM} > 200$. Cette valeur correspond approximativement aux occurrences d'évanouissements majeurs sur le canal simulé, soit dans le domaine de la milliseconde, tel que mentionné à la section 6.2.1.

6.2.3.3 Taux d'erreur binaire

Cette dernière section présente les courbes de BER du prototype en fonction du paramètre n_{STRF} seulement. L'impact du paramètre n_{PM}, qui est pourtant mesurable sur l'EQM et le taux de poursuite, est nul sur le BER. Sur l'horizon de 30 000 symboles, un bit en erreur survient à de rares occasions, trop peu pour être considéré dans la plage d'intérêt ($> 10^{-3}$). La figure 56 montre le BER en fonction du ratio SNR pour les taux d'analyse/synthèse réduits de 1, 3, 10 et 24 symboles / itération.

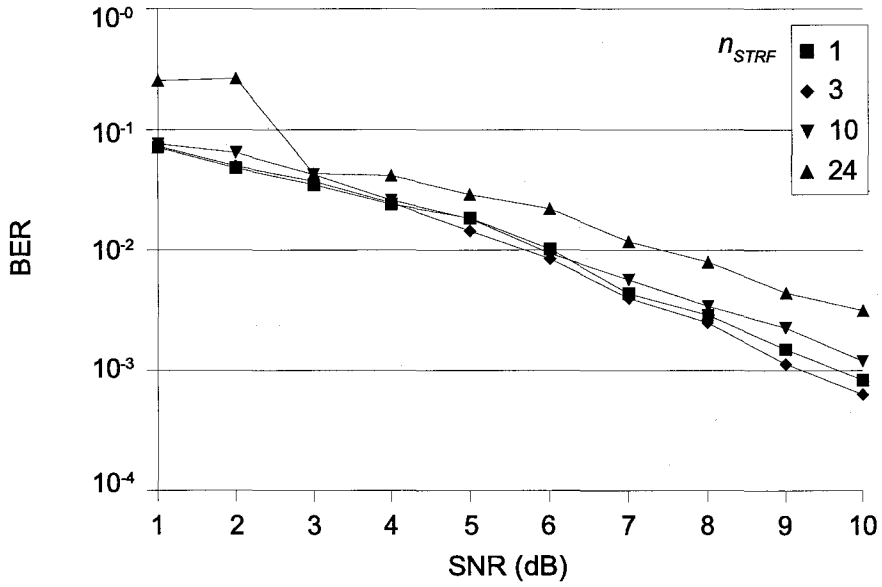


Figure 56 BER du prototype en fonction de n_{STRF}

Le cas idéal $n_{STRF} = 1$ est surclassé par le cas $n_{STRF} = 3$ pour des SNR élevés. Autrement, le cas $n_{STRF} = 10$ démontre une dégradation de 1 dB à un BER équivalent pour un SNR de

10 dB, alors que $n_{STRF} = 24$ montre plutôt une dégradation de 2 dB à ce niveau de puissance. Ces différences s'estompent lorsque le SNR est inférieur à 7 dB.

L'effet du taux de traitement réduit est donc observable par une légère diminution des performances du récepteur. Les efforts d'intégration devraient cependant mettre l'accent sur la possibilité d'intégrer des régions STRF supplémentaires, de manière à restreindre n_{STRF} à des valeurs voisines de 10. Les performances de réception seront donc, dans une certaine proportion, liées à la taille globale du récepteur, et en particulier, au nombre de régions STRF.

6.3 Taille physique du récepteur

Les constats issus de la section précédente laissent entendre que malgré les hypothèses simplificatrices, il est judicieux de réaliser le récepteur STAR sous forme modulaire avec plus d'une région STRF. Cette section tente de mesurer avec précision l'utilisation des ressources du FPGA par les différents modules de STAR de manière à estimer la taille globale d'un récepteur complet.

6.3.1 Méthodologie

Les résultats rapportés ici impliquent d'importantes séquences de compilation, de synthèse et de placement et routage. Pour les modules de petite taille, soit moins de 500 SLICEs (la SLICE est une entité logique du FPGA souvent utilisée pour comparer la taille de modules commerciaux), on se permet de faire une évaluation précise des ressources en exécutant, successivement, la suite complète d'outils de conception en ne changeant qu'un seul paramètre par itération. Les combinaisons possibles pour couvrir tous les cas augmentent de façon exponentielle en fonction du nombre de paramètres à évaluer.

Ce type d'évaluation passe donc par l'automatisation des séquences de test. À l'aide d'un script sous l'interpréteur de commande BASH, et avec la collaboration des outils Synplify et Xilinx ISE qui supportent le traitement par lots (« batch processing »), on parvient à

réduire les étapes de synthèse et de placement et routage à de simples appels de fonction que l'on aura disposés au centre de p boucles « for » contrôlant chacun des p paramètres à explorer.

À chaque boucle, certains fichiers de configuration sont modifiés pour refléter les nouveaux paramètres, à la suite de quoi les outils sont lancés et leurs fichiers de résultats sont examinés pour en extraire les valeurs d'intérêt. On y retrouve, entre autres, le nombre de SLICEs, de mémoires « Block RAM », et de multiplicateurs dédiés nécessaires. Cette méthode permet de démarrer automatiquement une quantité importante de tests sans intervention humaine.

6.3.2 Module basicMac et ses dérivées

Puisqu'ils prennent en charge la majeure partie de la complexité de l'algorithme, les modules basicMac et matrixMult sont examinés conjointement.

6.3.2.1 Sous-module basicMac

L'utilisation de ressources par basicMac est prévisible en fonction de ses paramètres de quantification. Les multiplicateurs dédiés sont toujours au nombre de quatre, reflétant le parallélisme des produits partiels. Le nombre de SLICEs dépend, lui, de la quantification des opérandes. Chaque sommateur et registre est constitué d'un nombre de SLICEs qui reflète sa taille en bits. Afin de jauger l'effet de la quantification sur la taille de basicMac, on varie la taille du résultat $\boldsymbol{R}$ de 9 à 15 bits (abscisse), et la taille de l'opérande $\boldsymbol{C}$ (légende). On ne considère pas la taille des opérandes $\boldsymbol{A}$ et $\boldsymbol{B}$ puisqu'elles n'ont aucune incidence sur la taille du sous-module, en autant qu'elle soient inférieures à 18 bits (correspondant à la taille fixe des ports d'entrée des multiplicateurs dédiés).

La figure 57 montre les courbes résultant des rondes de synthèse et de placement et routage. Le nombre de SLICEs varie de façon linéaire avec l'abscisse, mais révèle un

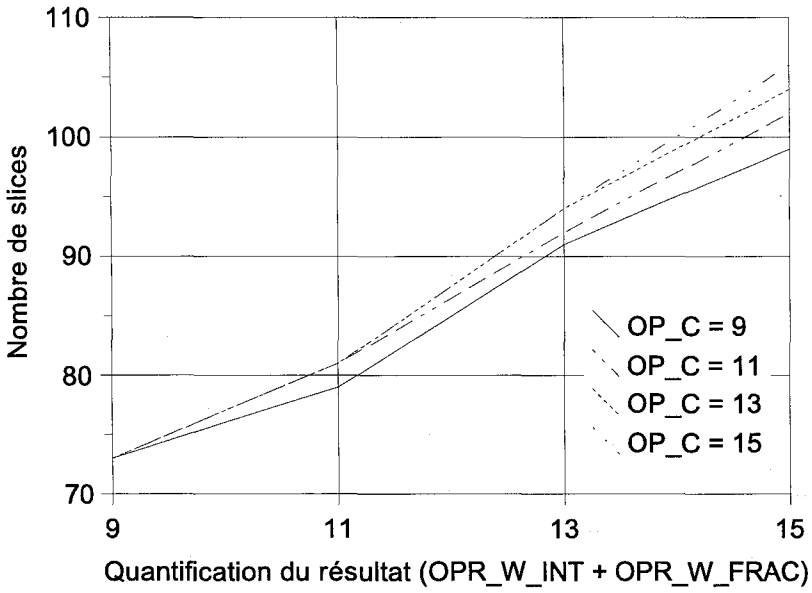


Figure 57 Taille de basicMac selon la quantification des opérandes

plancher d'environ 70 SLICEs qui englobe les éléments de contrôle. On peut donc conclure que la taille des opérandes a une incidence sur le reste de la taille de basicMac.

MatrixMult est soumis aux mêmes tests que basicMac. Variant les tailles de ***C*** et ***R*** selon les mêmes conventions, on obtient une courbe similaire, présentée à la figure 58. L'augmentation des ressources matérielles est quasi linéaire et dépend essentiellement des registres de stockage d'éléments, eux-mêmes tributaires de la quantification des opérandes ***A***, ***B***, ***C*** et ***R***.

Le nombre de SLICEs pour un résultat sur 9 bits est différent que celui indiqué à la figure 57, et découle d'un regroupement de ressources fait par le logiciel Synplify. On note toujours une variation d'environ 30 SLICEs entre les cas OP_C = 9 et OP_C = 15 lorsque le résultat est exprimé sur 15 bits, ce qui corrobore les résultats de la section 6.3.2.1. Jomphe (2004b) présente une étude abrégée combinant les résultats des figures 57 et 58.

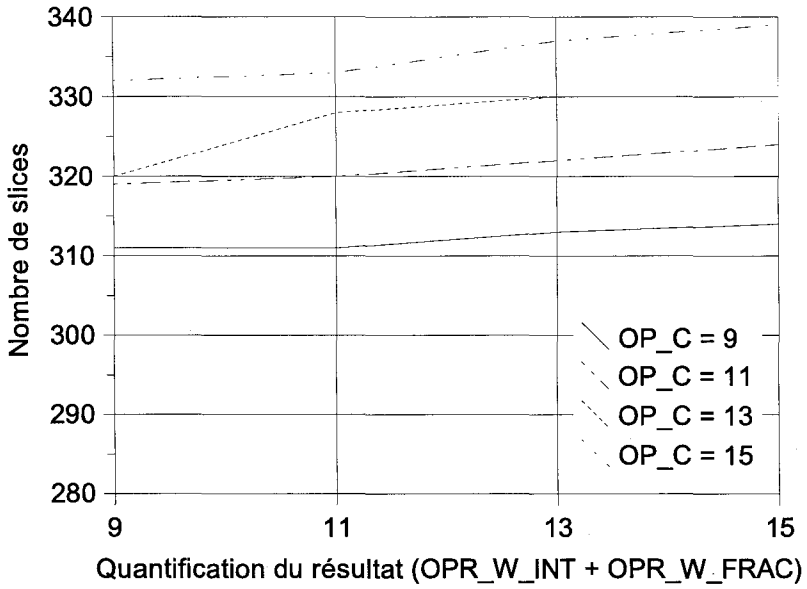


Figure 58 Taille de matrixMult (incluant basicMac)

6.3.3 Modules spécialisés

L'étude détaillée de la taille des autres modules est de moindre importance puisque ces derniers ne sont utilisés qu'à un endroit unique, à un seul niveau de quantification. Pour les niveaux de quantification calculés à la section 3.4, les tailles en SLICEs ainsi que le nombre de multiplicateurs dédiés requis par chaque module sont rapportés au tableau XII. Ces résultats sont adaptés des nombres en LUTs (sous-unité d'une SLICE) rapportés par Jomphe (2005).

6.3.4 Ensemble du système

Le tableau XIII présente la taille des différentes régions du prototype, en nombre de SLICEs, de mémoires et de multiplicateurs dédiés. La catégorie « infrastructure interne » renferme le module SCU ainsi qu'une certaine quantité de logique utilisée pour l'aiguillage des données dans le pipeline STRF (multiplexeurs, signaux de contrôle). La portion « assemblage » concerne plutôt les ressources nécessaires au fonctionnement du

récepteur sur la plate-forme ICS: bus OPB, modules fournis par ISR Technologies, interface au bus PCI, etc.

Tableau XII

Taille en SLICEs des modules du récepteur STAR

Module	Multiplicateurs	SLICEs
DESP	-	320
CDFI	2	498
UDFI	2	498
MRC	4	315
PWREST	3	106
STSEP	4	315
KAPPA	2	88
LAMBDA	4	320
FFTW	8	776
TDU	5	255
DCDT	-	101
FDMAP	-	99
RECONST	4	314
NORMSNOOP	2	63
NORMAPPLY	2	99
SCU	1	759
Total	43	4 926

KAPPA et LAMBDA constituent ensemble le module TMU (section)

Tableau XIII

Utilisation des ressources entre différentes régions de STAR

Région	Multiplicateurs	Block RAM	Slices
SP	11	9	1 737
STRF	31	19	2 430
Infrastructure interne	3	-	1 023
Assemblage (PM, IPIF, OPB, etc.)	7	54	4 239
Total	52	82	9 429

6.3.4.1 Station de base 3G

La station de base cellulaire 3G nécessite la prise en charge de 24 usagers. Sans perte de généralité, on assume la réception de 24 codes d'étalement différents, bien qu'une partie du corrélateur doit être dupliquée pour décoder chaque canal supplémentaire d'un usager.

Suivant les performances de réception mesurées à la section précédente, on alloue trois régions STRF pour supporter chacune 8 usagers. La section SP est elle aussi dupliquée trois fois, à la différence que chacune contient alors 8 corrélateurs. Étant donné la faible charge de calcul, le Microblaze ainsi que toute l'infrastructure interne demeurent les mêmes que dans le cas d'un usager unique. Le tableau XIV donne l'utilisation des ressources physiques dans le cas d'une station de base 3G complète à une antenne ($M = 1$).

On peut noter le nombre de mémoires dédiées des trois régions STRF qui est inférieur au triple du même nombre au tableau XIII et qui témoigne du partage entre deux STRF des mémoires dédiées utilisées en mémoires mortes.

Tableau XIV

Utilisation des ressources, station de base 3G à 24 usagers

Région	Multiplicateurs	Block RAM	Slices
SP	11	16	3 977
STRF	93	54	7 290
Infrastructure interne	3	-	1 023
Assemblage (PM, IPIF, OPB, etc.)	7	54	4 239
Total	114	134	16 529

6.3.5 Performance de l'architecture

On peut établir un lien entre l'utilisation des ressources matérielles et la quantité d'opérations par seconde qu'un module peut effectuer. La figure 59 montre à cet égard que

la majorité des modules de STAR, considérés selon ce critère, sont groupés autour d'une droite. La zone supérieure à cette droite renferme des modules plus performants que la moyenne, et la zone inférieure, des modules moins performants. Deux modules se démarquent: d'une part, le corrélateur, responsable d'un grande proportion des calculs arithmétiques, se compose de très peu de SLICEs, et s'acquitte pourtant d'une grande proportion des calculs. D'autre part, le Microblaze figure au bas du palmarès avec une charge de calcul très faible pour une utilisation de près de 1200 SLICEs. À noter aussi que le module de contrôle SCU (section 4.5.12) se retrouverait dans la même région que le Microblaze. Il a été omis puisque son apport concerne la gestion et non le calcul. Le module FFT, lui, a été inclus pour démontrer que malgré sa forte consommation de ressources logiques, se retrouve dans la moyenne.

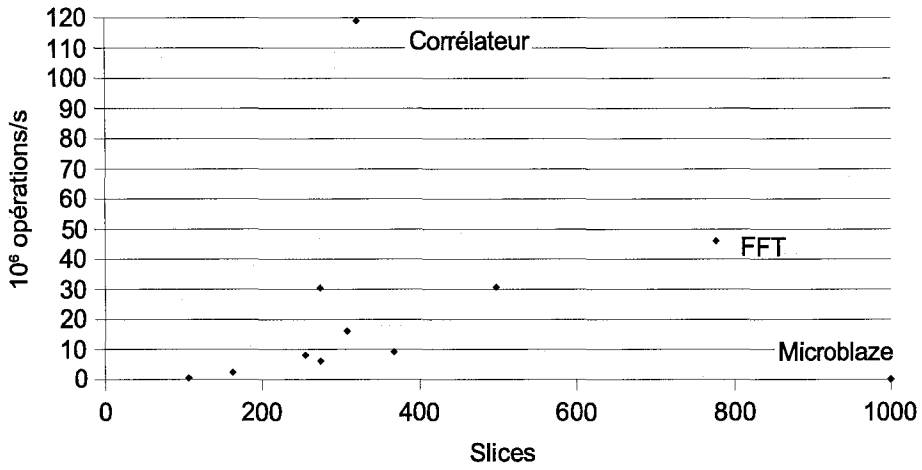


Figure 59 Puissance de calcul versus ressources logique

On doit cependant considérer que les modules utilisent en majorité des multiplicateurs dédiés, alors que le Microblaze n'en possède qu'un seul qu'il doit partager entre toutes ses tâches. La figure 60 montre le nombre d'opérations par seconde effectuées par un module en fonction du nombre de multiplicateurs dédiés qu'il comporte.

Évidemment, les modules à 4 multiplicateurs sont plus nombreux puisqu'il s'agit de matrixMult, responsables de près de 50% de la puissance de calcul totale de STAR.

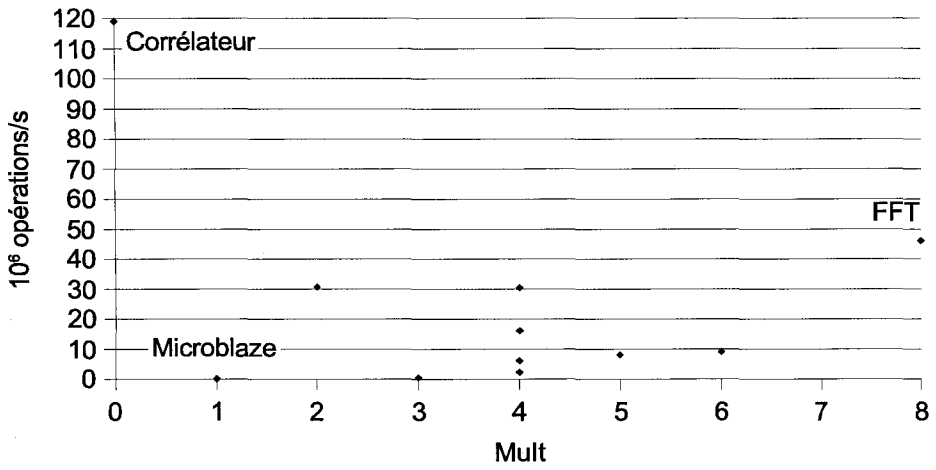


Figure 60 Puissance de calcul versus multiplicateurs dédiés

On peut déduire de ces observations une mesure d'efficacité d'utilisation du FPGA. Connaissant le nombre de SLICEs utilisées et la quantité de calculs effectués, on établit le ratio d'opérations par seconde en fonction de la taille en SLICEs comme

$$\frac{240 \text{ x } 10^6 \, ops/s}{8155 \, \text{SLICEs}} = 29\,430 \frac{ops/s}{\text{SLICE}} . \tag{6.3}$$

Le nombre de 8155 SLICEs considère le total du tableau XIII soulagé de la contribution de l'assemblage externe. Si l'on ne considère que les sections SP et STRF, qui constituent le véritable noyau de l'algorithme, on trouve plutôt

$$\frac{240 \text{ x } 10^6 \, ops/s}{4167 \, \text{SLICEs}} = 57\,595 \frac{ops/s}{\text{SLICE}} , \tag{6.4}$$

pour le cas d'un usager unique. En opposition à l'équation 6.2, ceci indique bien que près de la moitié des ressources sert en réalité à accomplir des tâches de support dans le

système. De surcroît, puisque la section STRF peut supporter trois usagers au taux maximal de $n_{STRF} = 3$ (ou huit au taux de $n_{STRF} = 8$), on peut envisager un système minimal doté de trois SP et d'un seul STRF, et, négligeant le reste, on obtiendrait

$$\frac{720 \times 10^6 \, ops/s}{7641 \, \text{SLICEs}} = 94\,228 \frac{ops/s}{\text{SLICE}}. \tag{6.5}$$

Plus le partage des tâches de support par les ressources arithmétiques actives est important, plus la surcharge sera minime vis-à-vis l'équation 6.1. Bien que ces résultats dépendent hautement de l'architecture des modules, ils permettent, dans ce cas-ci, et à haut niveau, de dimensionner un récepteur en fonction de la puissance de calcul requise.

6.3.6 Comparaison: récepteur RAKE sur DSP

La société TI offre un design de référence d'un récepteur de type RAKE pour sa famille de DSP à virgule fixe TMS32062xx. L'analyse des performances de cette solution (TI, 2000) permet de mieux situer la taille physique du récepteur STAR telle que décrite à la section précédente. Cet exercice effectue la comparaison de deux algorithmes différents, évoluant sur deux plate-formes différentes, et n'est donc valable que pour souligner les différences fondamentales entre un récepteur RAKE et la solution STAR.

6.3.6.1 Puissance de calcul

Texas Instruments propose un récepteur RAKE doté de deux méthodes de recherche de trajets: une rapide (« quick »), qui considère l'emplacement déjà connu de six trajets, et une complète (« full »), qui n'utilise aucune information préalable et tente de découvrir l'emplacement de six (6) trajets potentiels. Ces deux procédures, qui sont l'équivalent de l'analyse/synthèse de STAR, s'enchaînent à toutes les 10ms. La première remplaçant la seconde uniquement lorsque le récepteur n'arrive plus à recouvrer une transmission intelligible.

Plusieurs assomptions sont effectuées pour trouver un dénominateur commun au deux récepteurs. D'abord, l'algorithme de TI est conçu pour la réception de six trajets, alors que les résultats de STAR n'en utilisent que trois ($P = 3$). Un facteur de 0,5 est donc appliqué à la puissance de calcul de l'algorithme de recherche de TI. Ensuite, le facteur d'étalement utilisé par TI, soit $L = 256$, diffère d'un facteur 8,0 avec la situation dans laquelle STAR a été vérifié. Puisque le débit des symboles à $L = 32$ est huit fois plus élevé, le facteur de correspondance avec STAR est simplement 1,0 pour la section « Receive 3 fingers » chez TI.

On remarque en dernier lieu que la recherche des trajets de STAR ne correspond ni à la recherche complète, ni à la recherche abrégée que propose TI. Afin de conserver l'objectivité de la comparaison, on considère que la recherche à $n_{STRF} = 10$ de STAR est suffisante à maintenir les performances présentées au chapitre 5, et que la recherche « complète » ou « abrégée » (dont le ratio dépend des conditions du canal et n'est pas estimé par TI) effectuée à toutes les 10 ms suffit à maintenir les performances voulues pour la réalisation RAKE WCDMA sur DSP.

À titre indicatif, le rythme d'analyse/synthèse de STAR devrait être positionné à $n_{STRF} = 83$ (lorsque $L = 32$) pour rejoindre celui du RAKE, ou de manière équivalente, celui du RAKE devrait être ajusté à $T = 83\mu s$ pour atteindre le taux de STAR.

TI évalue les performances de cette réalisation en nombre de cycles de processeurs, ce qui se traduit en vitesse d'exécution requise (en MHz) et en nombre de DSP requis. Bien que les résultats soient donnés pour un facteur d'étalement $L = 256$, TI souligne que « pour des facteurs d'étalement plus faibles [...] le nombre d'opérations par seconde augmente. » Puisque cet effet n'est pas davantage documenté, il n'est pas considéré dans la comparaison à $L = 32$ qui suit.

Le tableau XV reprend les valeurs du tableau 4 de (TI, 2000) et les oppose aux données de puissance de calcul de STAR, tirées du tableau VII de la section 3.3.2, où les taux de traitement réduits sont de $n_{STRF} = 10$ et $n_{PM} = 100$.

Tableau XV

Complexité d'un récepteur RAKE sur DSP versus STAR sur FPGA

Récepteur	Section	Cycles	Période	Total cycles / s
RAKE	« Full Search »	1 010 278	10 ms	$101{,}0 \times 10^6$
	« Receive 3 fingers »	18 125	625 µs	$29{,}0 \times 10^6$
	Total (1 usager)	s/o	s/o	$130{,}0 \times 10^6$
	Total (24 usagers)	s/o	s/o	$3{,}12 \times 10^9$
STAR	SP ($L = 32$, $M = 1$)	1761	8,33 µs	$211{,}44 \times 10^6$
	STRF ($P = 3$, $L = 32$, $M = 1$)	7294	83 µs	$87{,}88 \times 10^6$
	PM	99	830 µs	$0{,}12 \times 10^6$
	Total (1 usager)	s/o	s/o	$299{,}44 \times 10^6$
	Total (24 usagers)	s/o	s/o	$7{,}19 \times 10^9$

Le total de cycles/s pour l'ensemble des 24 usagers est supérieur à 3×10^9 cycles d'horloge dans le cas de la solution RAKE, ce qui nécessiterait un DSP cadencé à 3 GHz pour réaliser la réception en temps réel, dans le cas unique ou chaque cycle du DSP serait utilisé à la résolution du problème. Il est généralement acquis, pour un processeur, qu'une instruction sur dix est « utile », alors que les neuf autres sont employées pour les déplacements de données, l'évolution du programme et tout autre contrôle logique. Puisqu'il est impossible d'obtenir cette information sans effectuer le profilage du code, on s'en remet à TI qui affirme de manière qualitative qu' « un système réaliste nécessiterait au moins un DSP par usager » (TI, 2000).

Nonobstant la puissance de calcul de $7{,}19 \times 10^9$ ops/s, le récepteur STAR sur FPGA bénéficie d'un modèle de calcul parallèle et du partage massif des ressources matérielles développés aux chapitres 3 et 4. Pour une cadence d'opération de 100 MHz, il est

vraisemblablement possible, quoique non réalisé, d'intégrer la version à 24 usagers de STAR à l'intérieur d'un Virtex2 6000 -5 (constitué de près de 19600 SLICES) selon les données du tableau XIV.

Une ambiguïté subsiste suite à cet exercice de comparaison puisque les prémisses de fonctionnement d'un DSP sont différentes de celles d'un système adapté sur FPGA. Les chiffres présentés au tableau XV démontrent uniquement que les deux algorithmes présentés engendrent une complexité algorithmique de l'ordre de 10^9 ops/s, alors que leurs performances de réception sont différentes.

6.3.6.2 Consommation électrique

Au chapitre de la consommation électrique, la solution basée sur une trentaine de DSP se verrait attribuer des composantes de la famille TMS320C62xx, qui offre des cadences d'opération allant jusqu'à 300 MHz. Dans ces conditions, chaque DSP consomme en moyenne 1,1 W pour une « activité typique », et la consommation totale du système dépasserait les 30W. Cet estimé ne comptabilise pas les mémoires vives qui sont associées à un tel montage, alors qu'il est généralement acquis que la consommation électrique des mémoires vives dans un système de traitement de signal est voisine de celle du processeur employé.

En comparaison, le FPGA de modèle Virtex 2 6000 de grade « -5 » occupé à 75% par le récepteur STAR, selon la configuration donnée au tableau XIV, requerrait une puissance électrique de l'ordre de 5,0 W, d'après l'estimation du logiciel XPower (version 6.3.03i) de Xilinx. Puisque toutes les mémoires nécessaires au systèmes sont situées dans le FPGA, aucune mémoire vive externe ne serait requise, rendant cette consommation électrique d'une situation réelle.

6.4 Conclusion

Ce chapitre a permis de mesurer la dégradation des performances du récepteur STAR au fil des étapes de réalisation du prototype. L'effet de la quantification et du passage au récepteur physique ont pu être évalués par trois critères de performance, soit le taux d'erreur binaire (BER), l'erreur quadratique de l'estimateur temporel des multi-trajets, et le ratio poursuite des multi-trajets. Les résultats ont démontré que la quantification telle que définie au chapitre 3 n'avait que très peu d'incidence sur les performances, mais que le passage au prototype sur FPGA entraînait une certaine dégradation des performances. En comparaison avec les récepteurs documentés dans la littérature, la solution STAR permet d'atteindre avec une seule antenne ($M = 1$) les taux d'erreur qui en nécessitent deux ($M = 2$) avec d'autres solutions du type RAKE.

L'effet des hypothèses simplificatrices a aussi été évalué à l'aide des mêmes critères. Il a été démontré que le taux de gestion de trajets n_{PM} n'a aucune incidence sur le BER, mais que le taux d'analyse/synthèse n_{STRF} pouvait entraîner une dégradation allant jusqu'à 2 dB par rapport au cas idéal. Malgré ces dégradations, les performances demeurent supérieures aux autres solutions. Les hypothèses simplificatrices sont donc validées.

La taille physique du système final a ensuite fait l'objet d'une analyse détaillée. L'impact de la quantification des opérandes sur la taille précise du module matrixMult et du sous-module basicMac a été mesuré. Le résultat révèle la contribution importante de l'infrastructure du système par rapport aux ressources utiles aux manipulations arithmétiques. La taille globale du récepteur est ensuite détaillée, permettant de dresser un portrait plausible d'un récepteur STAR modulaire contenant plusieurs régions SP, STRF, et PM.

Enfin, l'optimalité de l'architecture a été évaluée en utilisant le ratio opérations/s par SLICE comme indicateur de performance. Il est alors apparu que les tâches de support comptaient pour près de la moitié de la puissance de calcul globale dans le cas d'un usager

unique, mais que leur importance se diluait dans le cas multi-usagers et multi-antennes auquel est destiné le récepteur STAR. En comparaison, un récepteur RAKE réalisé sur DSP est évalué, et démontre que le récepteur STAR sur FPGA obtiendrait vraisemblablement une consommation électrique inférieure dans le cas multi-usagers.

CONCLUSION

Le travail dont faisait l'objet ce mémoire visait la réalisation du récepteur STAR sur une plate-forme matérielle afin de valider ses performances en temps réel. Il devait, en particulier, vérifier certaines hypothèses suggérées par des publications antérieures portant sur STAR (Affes, 1996) (Affes, 1997) (Cheikhrouhou, 2001). Il devait aussi répondre aux interrogations soulevées par les choix architecturaux préliminaires au développement d'un prototype.

En premier lieu, la complexité arithmétique de STAR a été vérifiée en termes d'opérations par seconde, confirmant globalement les chiffres avancés par Cheikhrouhou (2001). La quantification de l'algorithme a ensuite été effectuée suivant une étude statistique des variables concernées. Par la suite, l'architecture globale du récepteur physique a été élaborée, considérant des taux de traitement réduits et une construction modulaire permettant le passage futur aux modes multi-antennes et multi-usagers. Une topologie de système codesign a été suggérée, reconnaissant l'asymétrie de complexité entre les différentes régions de l'algorithme. Quatrièmement, les sections matérielles et logicielles du récepteur ont été réalisées en langage VHDL et C, respectivement. Ensuite, les dégradations causées par les étapes de prototypage ont été mesurées par plusieurs rondes successives de tests comparatifs entre le modèle de référence, le modèle quantifié, et le prototype final. Enfin, une mesure de la dimension du récepteur a été effectuée à partir de l'analyse de la taille de certains modules plus utilisés.

La faisabilité du récepteur STAR en système physique est donc démontrée par l'aboutissement de ce travail, qui a été l'objet d'un dévoilement devant public au colloque PROMPT-Québec d'avril 2005 (annexe 4). Le paradigme de réception analyse/synthèse est donc validé, et indique que l'architecture modulaire présentée dans ce document est valable.

L'impact de la quantification, telle que déduite du comportement statistique des variables de l'algorithme, entraîne une dégradation négligeable du taux d'erreur binaire par rapport

au modèle de référence. Le prototype sur FPGA montre pour sa part une amélioration de 2 dB pour des rapports signal-à-bruit (SNR) faibles, et une dégradation de 2 dB à SNR élevé. Ce comportement suggère une sensibilité plus faible que prévue du module de régularisation du récepteur. L'erreur quadratique, indicateur de performance de la poursuite des trajets, montre cependant une augmentation des performances par le prototype FPGA de l'ordre de 5 dB (EQM de -60 dB). Le taux de poursuite demeure semblable et ne permet pas de tirer de conclusions définitives. On conclue donc que la quantification actuelle entraîne certaines pertes, qui sont cependant insuffisantes pour retirer à STAR l'avantage qu'il détient sur les récepteurs RAKE conventionnels.

L'effet des hypothèses simplificatrices est mesuré en fonction du BER, de l'erreur quadratique, et du taux de poursuite. On montre que le taux de fonctionnement du gestionnaire de trajets, n_{PM}, n'a aucune influence sur le BER, alors qu'il entraîne une dégradation très légère sur l'erreur quadratique du positionnement des trajets. Le taux de poursuite, lui, se détériore suivant la diminution de n_{PM}.

Le taux d'analyse/synthèse, n_{STRF}, montre un effet marqué sur le BER. Le taux proposé de 1:24 entraîne des pertes de près de 2 dB à des niveaux de SNR élevés. L'erreur quadratique subit une faible dégradation (moins de 5 dB), et le taux de poursuite demeure essentiellement identique. Les hypothèses simplificatrices, qui stipulent que des taux de traitement inférieurs n'ont que peu d'impact sur les performances, sont donc validées.

La taille physique du récepteur est évaluée précisément et révèle que le cas minimal de trois usagers à un taux d'analyse idéal peut être contenu à l'intérieur de 12903 SLICEs, 74 multiplicateurs et 100 mémoires dédiées. Suivant les indicateurs de dimension, il est plausible d'envisager la réalisation d'un récepteur STAR à 24 usagers dans la plate-forme actuelle, basée sur le FPGA Virtex 2 6000.

ANNEXE 1

Définition des registres

Cette annexe donne une liste exhaustive des registres de configuration de tous les modules programmables des régions STRF (« structure fitting ») et SP (« symbol path ») de STAR. Chaque nouveau lot de données doit nécessairement être accompagné d'informations quant aux dimensions des matrices et de l'usager à traiter (via les adresses de départ).

Ces registres sont accessibles par un bus unidirectionnel centralisé (MBUS) que contrôle le module SCU (« STAR Control Unit »). Ce bus est constitué de 12 bits de données et 4 bits d'adresse. Le décodage des adresses est remplacé par un signal d'activation privé (« chip enable » ou CE) pour chaque module.

Tableau XVI

Registres MBUS du processeur matrixMult

Registre	Adresse	Description
regCoeff0	0_d	Coefficient multiplicatif de la partie réelle
regCoeff1	1_d	Coefficient multiplicatif de la partie imaginaire
regBram1Base	2_d	Adresse de départ de l'opérande ***A***
regBram2Base	3_d	Adresse de départ de l'opérande ***B***
regBram3Base	4_d	Adresse de départ du résultat (***R***)
regLength	5_d	Nombre d'éléments du résultat
regRstMaskInnerCount	6_d	Nombre d'éléments d'une somme interne
RegBram1RstMask	7_d	Adresse du dernier élément de la dernière rangée (colonne) de ***A***
RegBram2RstMask	8_d	Adresse du dernier élément de la dernière rangée (colonne) de ***B***
regBram1Incr	9_d	Espacement entre deux éléments consécutifs de ***A***
regBram2Incr	10_d	Espacement entre deux éléments consécutifs de ***B***
RegBram1LoopIncr	11_d	Espacement entre le début de deux rangées (colonnes) de ***A***
RegBram2LoopIncr	12_d	Espacement entre le début de deux rangées (colonnes) de ***B***
regBram4Base	13_d	Adresse de départ de l'opérande ***C***

Tableau XVII

Registres MBUS du corrélateur DESP

Registre	Adresse	Description
regBram1Base	2_d	Adresse de départ des bribes reçues
regBram3Base	4_d	Adresse de départ des éléments de Z_{n+1}
regStopInner	8_d	Valeur d'arrêt du compteur interne de corrélation
regStopOuter	16_d	Valeur d'arrêt du compteur externe de corrélation
regInjectSpoke	14_d	Masque d'adresse. Le registre à décalage passe en mode « injection » lorsque regCountOuter atteint cette valeur

Tableau XVIII

Registres MBUS du processeur DFI

Registre	Adresse	Description
regMu	1_d	Pas d'adaptation du LMS, exprimé en fractions de 2^{-7}
regBram1Base	2_d	Adresse de départ de l'opérande $\boldsymbol{A}$
regBram3Base	4_d	Adresse de départ des opérandes $\boldsymbol{B}$ et $\boldsymbol{R}$
regRstMask	6_d	Nombre d'éléments (moins 1) des opérandes $\boldsymbol{A}$, $\boldsymbol{B}$ et $\boldsymbol{R}$

Tableau XIX

Registres MBUS du processeur FFTW

Registre	Adresse	Description
regBram1Base	2_d	Adresse de départ des éléments
RegStopMask	9_d	Adresse du dernier élément à injecter
RegFFTSize	14_d	Taille de la transformée (encodage « one-hot »): $0001_b \rightarrow$ 4 points, jusqu'à... $1000_b \rightarrow$ ... 32 points

Tableau XX

Registres MBUS du processeur TDU

Registre	Adresse	Description
regBram1Base	2_d	Adresse de départ des éléments de $\hat{\mathcal{D}}_n$
regBram2Base	3_d	Adresse de départ des éléments de $\tilde{\mathcal{D}}_{n+1}$
regBram3Base	4_d	Adresse de départ du résultat
regLength	6_d	Nombre de trajets P
regRstMaskInnerCount	7_d	Longueur des vecteurs $\mathcal{D}$
regBram1LoopIncr	12_d	Espacement entre le début de deux vecteurs de $\hat{\mathcal{D}}_n$
regBram2LoopIncr	13_d	Espacement entre le début de deux vecteurs de $\tilde{\mathcal{D}}_{n+1}$
RegSprGainMask	15_d	Masque binaire des bits actifs suivant L_Δ

Tableau XXI

Registres MBUS du module DCDT

Registre	Adresse	Description
regTarget	8_d	Bribe vers laquelle le délai $\hat{\tau}_1$ est propulsé s'il y a correction
regLowGate	9_d	Bribe inférieure des P délais
regHighGate	10_d	Bribe supérieure des P délais
regBypassDCD	11_d	Booléen permettant de court-circuiter le module DCDT

Tableau XXII

Registres MBUS du processeur FDMAP

Registre	Adresse	Description
regBram1Base	2_d	Adresse de départ des éléments de $\hat{\tau}_{n+1}$
regBram3Base	4_d	Adresse de départ des éléments de $\hat{D}_{n+1}$
regRstMaskInnerCount	6_d	Nombre de coefficients à traiter (correspond à $L_\Delta - 1$)
regBram1RstMask	7_d	Dernière adresse (relative) de $\hat{\tau}_{n+1}$ (correspond à P-1)

Tableau XXIII

Registres MBUS du module actif de combinaison normApply

Registre	Adresse	Description
regM	0_d	Correction pour la normalisation (nombre d'antennes)
regBram3Base	8_d	Adresse de départ de $\underline{\hat{H}}_{n+1}$

Le module SCU se démarque des autres puisqu'il est le maître du bus MBUS interne du récepteur STAR. Néanmoins, de par son rôle de mandataire des niveaux hiérarchiques supérieurs, il doit recevoir des paramètres globaux et autres informations de contrôle. Ces transactions sont réalisées par l'entremise d'un second MBUS duquel SCU est l'unique cible. Cet MBUS, appelé OMBUS (« Outside MBUS ») utilise 32 bits de données, mais toujours 4 bits d'adresse. Ces tailles sont ajustables pour des développements futurs.

Tableau XXIV

Registres MBUS du module de contrôle SCU

Registre	Adresse	Description
regMu	1_d	Pas d'adaptation LMS pour le module CDFI, quantifié en fractions de 2^{-7}
regEta	2_d	Pas d'adaptation LMS pour le module UDFI, quantifié en fractions de 2^{-7}
regXiPlus	3_d	Pas d'adaptation LMS pour le module LAMBDA (de type matrixMult), quantifié en fractions de 2^{-4}
regXiMinus	4_d	Inverse de regXiPlus (regXiPlus × -1)
regAlpha	5_d	Facteur de lissage pour le module PWREST, quantifié en 2^{-8}
normHHBypass / regBypassDCD	6_d	Bit 0: 0 → active la normalisation de $\underline{\hat{H}}$ Bit 0: 1→ court-circuite la normalisation de $\underline{\hat{H}}$ Bit 4: 0 → active la correction DCDT Bit 4: 1→ court-circuite la correction DCDT

Tableau XXIV (suite)

Registre	Adresse	Description
regTarget	7_d	Indice de la bribe où positionner le trajet $\hat{\tau}_1$
regLowGate /	8_d	Bits 5-0: limite inférieure pour la correction DCDT
regHighGate		Bits 11-6: limite supérieure pour la correction DCDT
regParametersPP	9_d	Paramètres suivant une modification par la région PM
		Bits 8-0: Taux d'étalement L
		Bits 11-9: Nombre d'antennes M
		Bits 14-12: Nombre de trajets P
		Bits 16-15: Réservé
		Bits 25-17: Taille de la fenêtre L_Δ
regCPUYIncr	13_d	Incrément entre deux lectures de la mémoire de bribes entrantes (BRAMY) (correspond à L)
regCPUZIncr	14_d	Incrément entre deux écritures dans la mémoire du PCM (BRAMZ) ($0 \rightarrow Z_{n+1}$ écrase Z_n)
symbolToCID	15_d	Latence d'utilisation du pipeline STRF (correspond au paramètre n_{STRF})

ANNEXE 2

Interface de simulation PowerBridge

L'infrastructure de simulation développée au chapitre 5 utilise le logiciel PowerBridge, un outil développé pour permettre l'échange d'informations entre le moteur de simulation du code VHDL, ModelSim, et l'interpréteur de scripts Octave (http://www.octave.org et http://octave.sourceforge.net). Octave est un logiciel permettant l'exécution de scripts Matlab, mais publié en tant que logiciel libre. Il facilite en outre l'interaction avec les autres langages scriptés sous Linux, tel que l'interpréteur BASH et le langage Tcl/Tk. Puisque l'interface graphique du logiciel ModelSim est elle aussi réalisée en Tcl/Tk, la sélection de ce langage est toute indiquée.

PowerBridge agit à titre d'entremetteur entre ModelSim et le modèle de référence de STAR exécuté par Octave. La fenêtre de contrôle montrée à la figure 61 permet d'ajuster les paramètres du récepteur STAR facilement et de contrôler le déroulement des simulations. Puisque l'interface illustrée utilise une nomenclature différente de celle employée dans ce document, le tableau XXV donne la correspondance entre les termes.

Tableau XXV

Correspondance des termes de simulation

Figure	Document
SymbolToCID	n_{STRF}
Feedback Rate	n_{PM}
Appearing Path Threshold	appThresh, υ_A
Appearing Path Hysterisis	appCount, n_A
Vanishing Path Threshold	vanThresh, υ_V
Vanishing Path Hysterisis	VanThresh, n_V

Les paramètres supplémentaires concernent le mécanisme de passage de paramètres entre les différents logiciels. PowerBridge démarre ModelSim pour une période de « Atom Run Length » unités de temps (ici 10 µs). Au terme de cette période, Powerbridge vérifie la nécessité de générer un nouveau lot de symboles en entrée (signal en spectre étalé, en bande de base). Lorsque requis, il récupère la valeur de contrôle de puissance dont le taux

est contrôlé par le paramètre « Feedback Rate », et la quantification, par le paramètre « Quantization ». Il peut alors redémarrer le modèle de référence sous Octave, dont l'état intermédiaire est sauvegardé, avec cette nouvelle information de contrôle de puissance. « Feedback Rate » symboles sont alors produits et insérés dans un fichier qui alimente le banc de test utilisé par ModelSim. Powerbridge reprend alors simplement la simulation pour une nouvelle période de « Atom Run Length » secondes.

Comme le montre la figure 61, une visualisation de certaines variables intermédiaires de STAR peut être effectuée périodiquement. Le cas illustré montre l'évolution, sur dix symboles, de la matrice $\underline{\hat{H}}$ avec un taux d'étalement de 32 bribes / symbole.

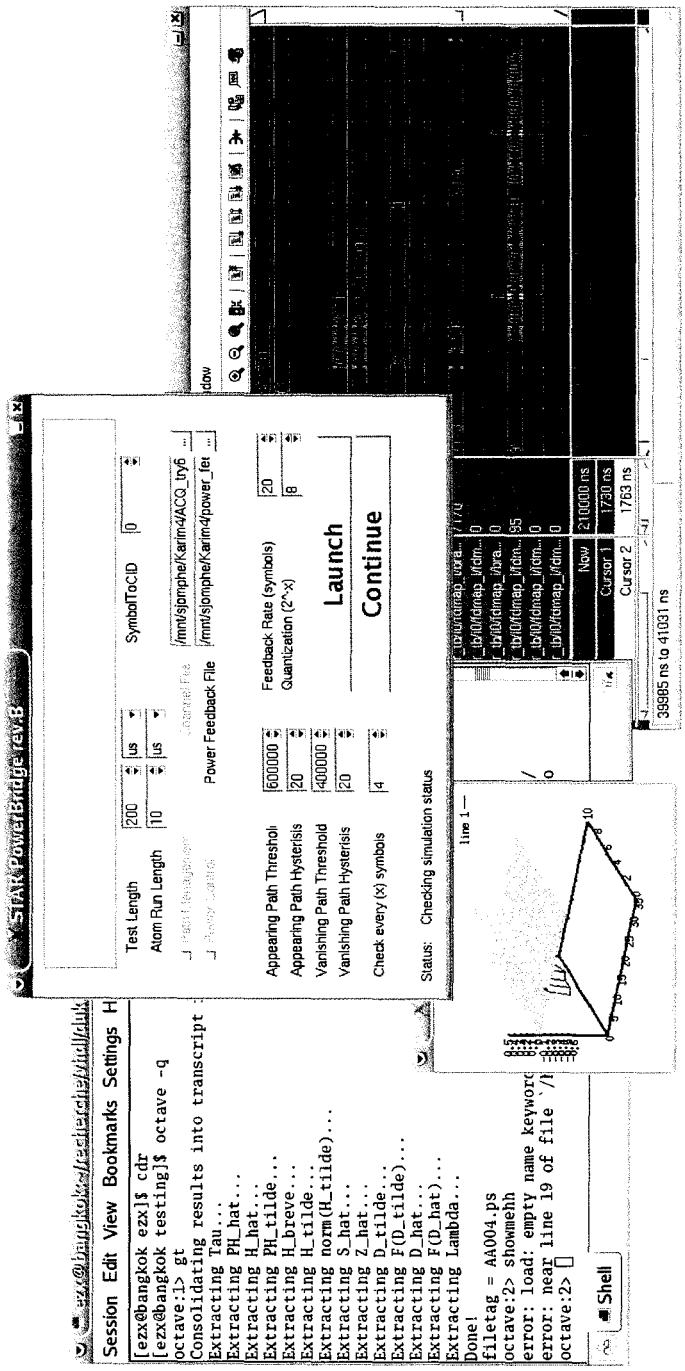


Figure 61 Interface de contrôle PowerBridge pilotant Octave et ModelSim

ANNEXE 3

Publications issues du projet

TABLE I
Computational Burden for STAR

Block	Complexity n_{ID}=1 (Mops)	Complexity n_{ID}=10 (Mops)	Location
Constrained DFI	66	6.6	FPGA
Unconstrained DFI	66	6.6	FPGA
Space-time Separation	97	9.7	FPGA
Temporal Matrix Upd.	320	32.0	FPGA
FFT on $\tilde{D}$	246	24.6	FPGA
Time-Delay Update	42	4.2	FPGA
Impulse Generation	102	10.2	FPGA
FFT on $\hat{D}$	246	24.6	FPGA
Reconstruction	82	8.2	FPGA
Difference Extraction	195	19.5	CPU/stats.
Localization Spectrum	54	5.4	CPU/stats.
Appearing Path Decision[a]	20	0.2	CPU/ctrl
Vanishing Path Decision[a]	2	negl.	CPU/ctrl
Combining	65.3	65.3	FPGA
Total	**1603**	**219.1**	**mixed**
Total (FPGA)	**1332**	**192.0**	**FPGA**
Total (CPU)	**271**	**25.1**	**CPU**

Table 1: Computational burden for the proposed codesign architecture. Light gray denotes blocks running once every n_{ID}=10 symbol. Dark grey denotes $n_{IDCPU(CTRL)}$ =100. Results are shown for M=2 antennas, P=3 tracked paths and L=32. White blocks operate at symbol rate.

[a] Part of "STAR Control in Fig.1.

$\hat{H}_{n+1}$ [1]. The unconstrained channel estimate, $\breve{H}_{n+1}$, though coarse and less stable, is free to track new appearing paths. A comparison between the two is carried out every 100 symbols or so. A significant difference between the two suggests the detection of appearing paths and triggers an update of the number of tracked multipaths, P, which resets the constrained DFI output $\tilde{H}_{n+1}$ to $\breve{H}_{n+1}$ (see Fig. 2). The reader is encouraged to refer to [1],[2] for an in-depth explanation of STAR.

2.2 Codesign Approach

There are three parameters to consider when partitioning an algorithm between hardware and software, as in Fig. 1. The first is the computational load (in 10^6 ops./sec, or *Mops*) of each module. The second is the bandwidth required by closely coupled blocks to exchange data. The third is the branch complexity of decision blocks.

Modules with a high computational load are natural candidates for a purely hardware implementation. Those with a high branch complexity are more appropriately realized in a CPU. The separation must however insure that closely coupled modules remain within the same implementation domain, otherwise the computational savings could be outweighed by the bandwidth requirements.

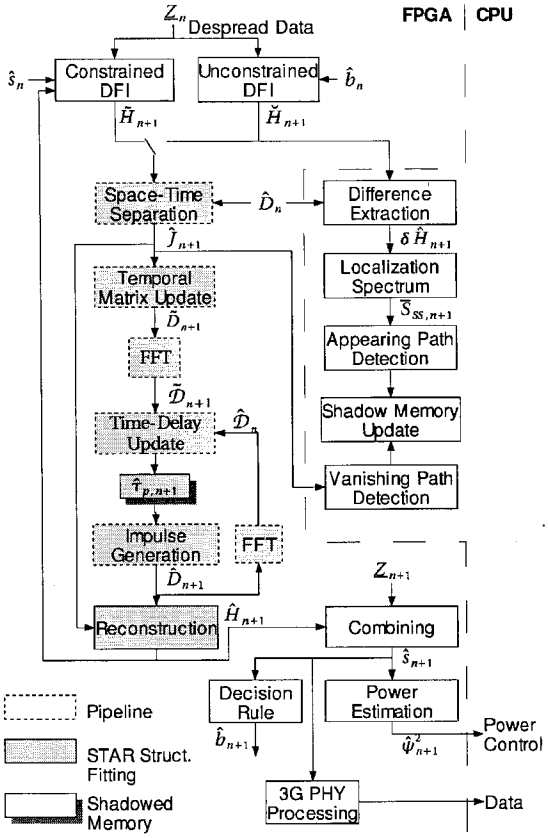


Figure 2: Pipeline Architecture of STAR

3. ARCHITECTURE

3.1 Computational Burden

Modifications to the original STAR algorithm have been carried out in [2], lowering the computational burden by 1 to 2 orders of magnitude.

Ideally, as in [1], channel estimation should take place every symbol. That is, the structure-fitting subsystem (grey blocks in Fig. 2 and Table 1) should be excited by every incoming symbol, requiring a total processing power of nearly 1.6 Gops. However, [2] makes the key assumption that the channel parameters are nearly time-invariant for the duration of a few symbols. Updating the

channel parameters every n_{ID}=10 symbols still preserves the channel tracking performance but dramatically reduces the required processing power. Table 1 shows that activating the structure-fitting pipeline every n_{ID}=10 symbols and re-examining the number of tracked paths every $n_{IDCPU(CTRL)}$=100 symbols requires a reasonable complexity of 219 Mops per radio channel.

Another key element of [2] is the introduction of a reduced processing window of length L_Δ. That is, the despreading operation is carried out with the entire spreading code over $2L$-1 chips, but only the first L_Δ elements of the cross-correlation vector are considered. When tracked properly, the main power components should not escape this reduced window. Experimental results in [2] suggest that a value of L_Δ=32 is sufficient. All H and D operands are dimensioned along L_Δ, so a lower value has a direct impact on matrix multiplications and FFT operations. With spreading factors smaller than 32, L_Δ takes the value of L.

3.2 Partitioning the algorithm

Implementing STAR as a software/hardware codesign architecture implies separating the highly regular computations from the control algorithms.

3.2.1. Core Computations. From the description of STAR [2] and our discussion about n_{ID}, we establish a frontier in the STAR block of Fig. 2 between the structure-fitting and the path management modules. Used every n_{ID} symbols, the structure-fitting pipeline consists mainly of matrix multiplications and sums. Though the dimensions of the operands can vary (eg. with changing P), an upper limit of M=2, L_Δ=32 and P_{MAX}=5 has been impose for implementation within an FPGA.

3.2.2. Decision Blocks. We can further partition STAR by recognizing that radio paths do not appear and vanish instantaneously. Therefore, the decision to add or remove them from the tracking process need not be made on a symbol-by-symbol basis. In fact, STAR only triggers such a change after $n_A = n_V \approx 100 = n_{IDCPU(CTRL)}$ iterations of continuous detection of an appearing or vanishing path, respectively. This translates to an update rate of only a few hundred Hz, hence moving these blocks to the CPU is natural. Moreover, since a given decision need not be taken at a specific moment, the underlying statistics can suffer some lag and thus, can be moved into the CPU where they will be updated every $n_{IDCPU(STAT)} \approx 10$ iterations.

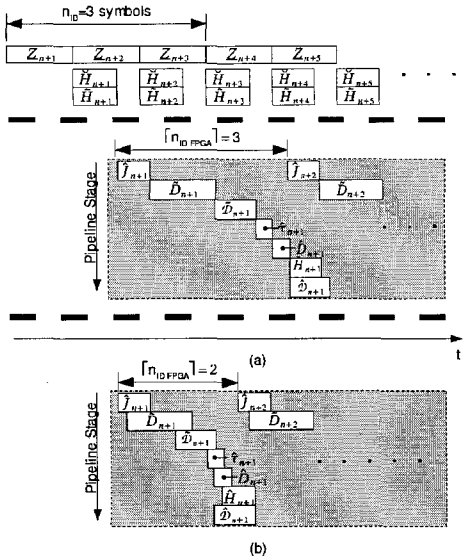


Figure 3: Fastest possible n_{ID} for STAR at L=32 for (a) the unchained pipeline and (b) the chained pipeline.

3.2.3. Inter-process Communications. The preferred mechanism to exchange data between the FPGA and the CPU is through shadow memory. In the current implementation, when the CPU detects a change in the number of paths, it rebuilds a coherent set of matrices in shadow memory and the FPGA is instructed to swap active and shadow memories at its next structure-fitting iteration. The CPU then reclaims the formerly active memory as shadow memory for a subsequent update. Since both CPU and FPGA can access the same physical memory resource on their own clock domain, the exchange from CPU to FPGA is a single data word: a new memory base address. This is known as the mailbox principle.

3.3 Pipeline Architecture

We have identified the structure-fitting pipeline in Fig. 2 as the critical path in the system. Our efforts are directed at reducing the length of its longest feedback.

A number of feedbacks are required from iteration n to n+1. This precludes the instantiation of a true pipeline structure. Specifically, the longest feedback begins with the temporal propagation matrix $\hat{D}_n$ and branches to the next iteration into the spatial propagation matrix $\hat{J}_{n+1}$. This dependency has not been resolved and we are forced to accept that the pipeline will be mostly empty. This situation can however be harnessed in two ways.

A first interesting concept is to feed many independent sets of data to the structure-fitting pipeline in an interleaved fashion. With each set representing the radio channel of a given end-user, we can effectively share a single structure-fitting pipeline between many users. A second way is to chain the pipeline, as described in the next section.

3.4 Chaining the Pipeline

Chaining allows the processor at stage k+1 access the output of stage k before actual completion. Since data exchange between neighboring modules is achieved through distributed shared memory, we must grant two separate processes access to the same memory resource at the same time. The preferred method is to use distributed dual-port memory resources. In doing so, we must however make sure that the input rate of stage $k+1$ does not exceed the output rate of stage k or a WAR ("Write-After-Read") data hazard could occur. The FFT core [3] is a good candidate for chaining because its I/Os are both serial and of the same rate.

Considerable care is required when designing a chained pipeline since different processors may require their input data in row-major and others may not (eg. for transpose matrices). This affects the actual chaining indent that each processor can gain.

3.5 Modified MAC

The bulk of the operations contained in the structure-fitting chain consists of matrix-MUL-matrix-ADD-matrix-type operations. Since all operands are dense matrices, no simplification can take place, so a specialized multiply-accumulate (MAC) architecture is introduced.

As shown in Fig. 4, a third input has been added to a traditional MAC structure in order to facilitate the matrix-times-vector-plus-vector operation which is frequent in STAR. Two stages (one per clock cycle) have been defined in the MAC, which considerably reduces the overall pipeline delay.

In comparison, a sequential processing unit requires 4 multiplications and 6 additions (2 real and 2 complex) to complete a partial result, for a total of 10 operations. While our MAC requires two cycles to produce a result, its pipelined architecture permits the chaining of operations. As the number of elements in the input vectors increases, the speedup over a purely sequential processor approaches 10.

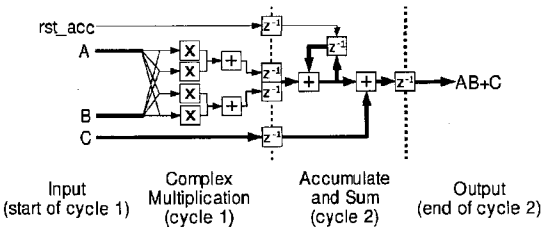


Figure 4: Two-cycle, complex operand MAC structure for the STAR pipeline. Bold lines denote complex numbers. Thin lines are either real or imaginary parts.

4. CONCLUSION

Through the use of a software/hardware codesign approach, we have shown how the Spatio-Temporal Array-Receiver (STAR) can be implemented using available technology. This receiver outperforms the 2D-RAKE in a WCDMA base station context.

By sorting the highly repetitive calculations from the branching operations, we have worked out a codesign architecture. Through algorithmic modifications that further emphasize the difference between software and hardware, we have reduced the latency of the structure-fitting pipeline thereby increasing the idle time of the pipeline.

We have shown how the structure-fitting pipeline can be further optimized to allow interleaved processing of multiple users or, through chaining, allow faster and more accurate processing of a single user.

Finally, a modified MAC allowed us to gain a speedup factor of nearly 10 relative to a sequential architecture.

Acknowledgments

Work supported by NSERC, ISR Technologies inc. and a Canada Research Chair on High Speed Wireless Communications.

References

[1] S. Affes and P. Mermelstein, "A New Receiver Structure for Asynchronous CDMA: STAR—The Spatio-Temporal Array-Receiver," IEEE J. Select. Areas Commun., vol. 16, pp. 1411-1422, Oct. 1998

[2] K. Cheikhrouhou, S. Affes and P. Mermelstein, "Impact of Synchronization on Performance of Enhanced Array-Receivers in Wideband CDMA Networks,", IEEE J. Select Areas Commun., vol. 19, pp. 2462-2476, Dec. 2001

[3] M.-E. Grandmaison, J. Belzile, C. Thibeault and F. Gagnon, "Reconfigurable and Efficient FFT/IFFT Architecture," accepted for publication at CCECE, Niagara Falls, May 2004

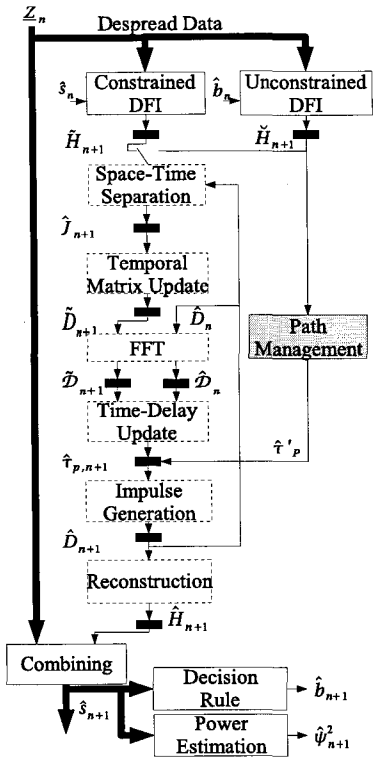


Figure 2. Pipeline structure of STAR. In FPGA: white solid-line blocks form the main data path (T_{MAIN}= 1 symbol), white dotted-line blocks make up the structure-fitting pipeline (T_{SF} ~ 3 symbols). In CPU: the gray block performs adaptive path management (T_{CPU}~ 100 symbols). Black boxes are dual-port RAM elements.

When allocating these functions between FPGA and CPU, we must ensure that closely coupled modules remain within the same implementation domain. Ignoring this would create a high bandwidth requirement through the FPGA/CPU boundary which would in turn outweigh the benefits of the codesign scheme.

Further splitting the structure-fitting portion of STAR into basic operations results in the pipelined model of Fig. 2. Given the serial nature of the algorithm, most of the pipe remains idle during a structure-fitting pass (see [3]), which enables the sharing of one SF between multiple users.

The building blocks required are matrix multipliers (MM), a dynamically-adjustable length FFT processor, a custom regression module capable of tracking sub-sample delays and a fractional-delay filter (FDFIR) for accurate temporal impulse generation.

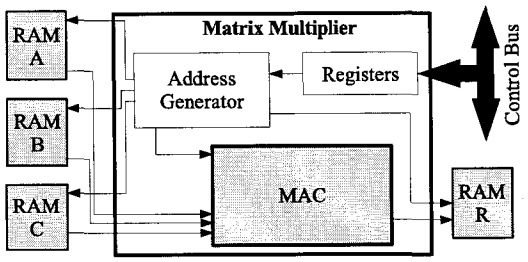


Figure 3. Matrix Multiplier architecture.

When allocating these functions between FPGA and in Figs. 1 and 2). This allows in some instances a module to process a data set even before the preceding stage has completed its task. In other situations, it lets the CPU update shadow memory areas while processing continues uninterrupted in the FPGA.

Given the worst-case dimensions of the operands, only a quarter of each RAM is required to hold the active data set of one user. With two data sets per user (shadow and current), we can fit two users within the available RAM ressources.

4. Implementation

In order to promote design reuse, we keep the number of different module types to a minimum. As such, we need to generalize the modules so they can accomplish different tasks, being careful not to over-design.

The FPGA part of the algorithm shown in Fig. 2 is handled by five different types of modules. While each type has a different core computation, they all exhibit the same interface. For example, the matrix multiplier (Fig. 3) utilizes an embedded specialized multiply-accumulate (MAC) unit to carry out the sequential multiplication, and has the necessary signals to interface to 4 RAMs and a STAR-wide unidirectional setup bus. The MM by itself simply feeds the right sequence of data from RAMs A, B and C to the MAC and stores the results in RAM R.

All operands depicted on Fig. 2 are matrices using fixed-width binary representation for every element. This width must be specified prior to synthesis for every module. However, each module contains a set of registers that keep track of the dimensions L, M and P (Fig. 1) of the matrices. These values must be dynamically updated by the CPU to reflect the current state of channel identification. With a few more dynamic settings, the MM can perform vectorial product, dot product or norm on its three operands at no further hardware cost.

The correct value for these registers is gathered from the CPU by the global STAR Control Unit (SCU) through a set of mailbox registers. This method facilitates the

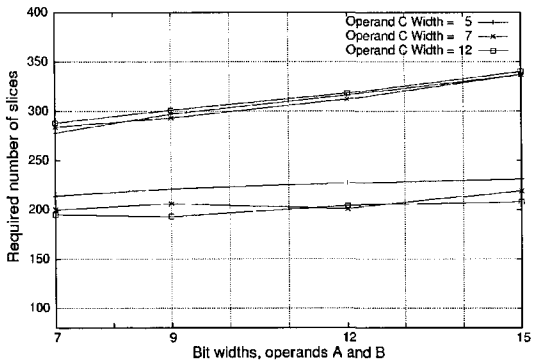


Figure 4: Complexity of modules in number of slices for the complete Matrix Multiplier (upper set) and the Matrix Multiplier without its embedded MAC.

crossing of the FPGA/CPU boundary. Following the precise order of pipeline operations, the SCU reprograms the modules in sequence through a simple unidirectional bus and can effectively bootstrap a whole new identification scheme if needed.

5. Synthesis Results

Design reuse gives us the ability to extrapolate the results of synthesis and place-and-route of a few modules to predict the global size of the STAR receiver.

Taking the MM as an example, we demonstrate that the selection of pre-synthesis generics has a linear impact on the core computation submodule (the MAC in this case), but nearly no impact on the wrapper itself. The upper set of curves in Fig. 4 shows the place-and-route results of several combinations of bit widths for the two multiplicative inputs (operands A and B), and the additive input (each curve in the set) in terms of number of slices. For bit widths varying from 7 to 15, the slice usage varies from ~275 to ~340. We should point out that the width of the result has minor impact on the slice utilization since it only discards unwanted bits which are computed anyway.

The lower set of Fig. 4 shows the size of the MM under the same conditions, omitting the MAC submodule. The nearly-constant number of slices throughout the different bit widths shows the complexity of the MM wrapper at around 225 slices, depending on the size of operand C. We should note however that these figures take into account the presence of operand C in the MAC, whereas it is seldom encountered in STAR, and hence, discarded by the synthesis process, leading to smaller synthesized modules.

The frequency of operation was targeted at 100MHz for a -5 grade XC2VP40 FPGA and was attained without much effort.

TABLE I
NUMBER OF SLICES PER MODULE

Block	Complexity (slices)	Number required	Total
DFI	203	2	406
Matrix Multiplier	309	5	1545
FFT Processor	800	1	800
Time-Delay Regr.	198	1	198
Fract. Delay Filter	213	1	213
Despreader	498	1	498
STAR Control Unit	595	1	595
Total	---	**12**	**4255**

Table 1 gives a by-module breakdown of the complexity of STAR for plausible bit widths. Without benefiting from the above-mentioned simplifications, the total number of slices required still shows that STAR canbe instantiated within our target platform.

6. Conclusion

In this article we have shown that the STAR algorithm can be implemented in a 3G Base Station within a software-defined radio context.

Through algorithm partitioning we have established a natural frontier between hardware- and software-specific portions of STAR, leading to a codesign implementation.

Furthermore, we have explained the dataflow between modules and over the FPGA/CPU boundary as well as the proposed architecture for constituting modules. Design reuse has been promoted in that many generalized modules are used for many functions.

Finally, we have given figures on the hardware usage both by module and globally, proving that STAR can indeed be implemented cost-effectively in off-the-shelf FPGAs.

REFERENCES

[1] S. Affes and P. Mermelstein, "A New Receiver Structure for Asynchronous CDMA: STAR—The Spatio-Temporal Array-Receiver," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1411-1422, Oct. 1998

[2] K. Cheikhrouhou, S. Affes and P. Mermelstein, "Impact of Synchronization on Performance of Enhanced Array-Receivers in Wideband CDMA Networks,", *IEEE J. Select Areas* Commun., vol. 19, pp. 2462-2476, Dec. 2001

[3] S. Jomphe, J. Belzile, S. Affes and K Cheikhrouhou, "Codesign Implementation of a 3G WCDMA Base Station Receiver", *IEEE Canadian Conference on Electrical and Computer Engineering, Niagara Falls, to appear, May 2-5 2004.*

past state of the channel structure, symbol recovery statistics and the PCM of incoming observation vectors. They are defined as:

$$\breve{H}_{n+1}=\breve{H}_n+\zeta\left(Z_n-\breve{H}_n\hat{b}_n\right)\hat{b}_n^* \quad (3a)$$

$$\tilde{H}_{n+1}=\hat{H}_n+\mu\left(Z_n-\hat{H}_n\hat{s}_n\right)\hat{s}_n^*, \quad (3b)$$

where μ and ζ are adaptation step-sizes and $\hat{b}$ is a hard-quantized version of $\hat{s}$. $\hat{H}_n$ is a synthetic, noiseless version of $\tilde{H}_n$ from the structure fitting subsystem (STRF) which prevents the CDFI from going astray.

A second point of interest is the path management unit (PM). Its only role is to monitor the power level of known paths as well as unit time-delays and assesses the necessity to either lock onto emerging paths or release fading paths. This decision relies on a hysteresis mechanism to prevent false detections.

Crossing into the STRF, we first extract the spatial dimension from $\tilde{H}$:

$$\hat{J}_{n+1}=\tilde{H}_{n+1}\hat{D}_n^T, \quad (4)$$

where $\hat{J}_{n+1}$ is an M-by-P spatial propagation matrix comprising the MP channel fading coefficients arising from the P paths and M antennas, and $\hat{D}_n$ is a synthesized P-by-L temporal support matrix for each of the P paths from the previous structure-fitting iteration. An updated time estimate is then extracted by LMS-fitting $\hat{D}_n$ onto the new spatio-temporal observation $\tilde{H}_{n+1}$:

$$\tilde{D}_{n+1}=\hat{D}_n+\frac{\xi}{M}\left(\tilde{H}_{n+1}^T-\hat{D}_n\hat{J}_{n+1}^T\right)\hat{J}_{n+1}^* \quad (5)$$

where ξ is an adaptation step-size and $\tilde{D}_{n+1}$ is the new temporal support estimate. We then determine the new optimal center-position of the chip impulse response for each of the P paths. This procedure is omitted for lack of space but can be found in [3]. The resulting temporal delays $\hat{\tau}_p$ have a precision of 0.001 T_c [3], far better than is possible with oversampling in RAKE-type receivers. The new synthetic temporal support matrix $\hat{D}_{n+1}$ is then populated with P replicas of a delayed chip impulse response. The STRF then recombines the processed spatio-temporal components as:

$$\hat{H}_{n+1}=\hat{J}_{n+1}\hat{D}_{n+1}^T. \quad (6)$$

We name this receiver paradigm the "analysis/synthesis" approach [5]. Key channel parameters such as $\hat{\tau}$ and $\hat{J}$ are computed prior to the synthesis operation, and can be used for channel characterization. Further enhancements such as carrier frequency offset recovery (CFOR) [5] integrate seamlessly within this algorithm with only minor processing of $\hat{J}$.

III. Performance Analysis and Verification

We proceed to show simulation results which outline the robustness of STAR in terms of time-delay tracking and

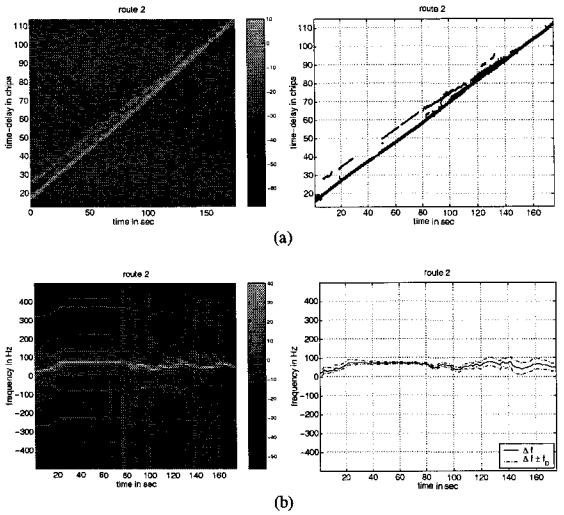


Fig. 2. Performance analysis of STAR along test route 2 in Laval near Montreal. (a) shows impulse response contour and corresponding extracted time-delay values. (b) shows power density spectrum and corresponding extracted CFO (Δf) and Doppler spread ($\pm f_D$).

CFOR, namely. We used a recording of a wideband CDMA channel from test route 2 [5]. These recordings employ a base rate spreading factor of 256, a carrier frequency of 1.9825 GHz, a chip rate of 4.096 Mcps and a power control of ± 0.25dB at a rate of 1600Hz. Further details can be found in [5].

Time-delay synchronization is the most crucial of all aspects of the receiver. Ref. [6] shows that time drifts significantly degrade the performance of enhanced WCDMA receivers. Relying on its analysis/synthesis paradigm, STAR can monitor the precise time-evolution of multipath components in a realistic manner. Fig. 2-a shows the time-delay impulse response contour measured from the recording alongside the corresponding time-delays extracted by STAR. The algorithm was able to maintain tracking for 100%, 97% and 60% of the entire recording time, respectively, for paths of power levels 0dB, -4.3dB and -8.0 dB.

CFOR also plays a major role. Ref. [4] allows a 0.10 ppm mismatch between transmitter and receiver carrier frequencies on the uplink but significant losses in SNR have been shown to happen for even smaller discrepancies [3].

The CFOR algorithm of STAR instantaneously estimates and compensates such imperfections and reduces the SNR losses accordingly. To verify this, we show in Fig.2-b the power spectral density of the first tracked multipath from test route 2 along with the extracted carrier frequency offset. Also depicted is the maximum Doppler spread. CFO extraction relies entirely on the PCM and compensation requires no explicit hardware in the RF chain.

IV. Dataflow and Building Blocks

We have previously published a framework from which STAR could be implemented [7]. Having recognized the duality of repetitive and logical (high branch count) operations, we have chosen a codesign approach. To this end, we have split STAR into three computational domains and kept cross-boundary bandwidth requirements at a minimum.

A) Algorithm Partitioning

Because a live receiver must handle incoming symbols at a fixed rate, the time allotted to (1) and (2) is finite. Intuitively, each received Z_{n+1} should be combined with a matched $\hat{H}_{n+1}$, but the amount of computations involved in synthesizing an optimal $\hat{H}_{n+1}$ from $\tilde{H}_{n+1}$ requires more time than is available.

Meanwhile, [6] shows that relaxing this one-to-one constraint has very little impact in terms of time-synchronization and received BER. It is suggested that updating $\hat{H}$ every n_{ID}=10 symbols is acceptable when L=32. This amounts to structure-fitting the channel once every 10 symbols, or every 83μs. This unties the timing requirements of the STRF from those of the SP and allows us to confine them to separate clock domains, exchanging $\breve{H}_{n+1}$ and $\hat{H}_{n+1}$ periodically.

To implement the PM subsystem, power level monitoring is required. By monitoring $\hat{J}$ and $\breve{H}_{n+1}$, the PM can assess the need to drop vanishing paths or lock onto emerging ones. Preliminary benchmarks suggest that the PM should run at a rate of approximately $10 \cdot n_{ID}$. Because this processing occurs at a comparatively low rate and is mainly composed of comparisons and branching, the PM is better suited to the software realm. Fig. 3 shows the three domains and how they interconnect.

B) Resource Reuse

STAR is ultimately meant as a multi-user receiver. It follows, then, that every spreading code in use will require its own SP, or more precisely its own despreader, in order to keep up with the incoming symbol rate. Sharing of other resources, such as DFIs, combiner, power estimator and STRF is, however, possible.

Structure-fitting $\tilde{H}_{n+1}$ into $\hat{H}_{n+1}$ is a sequential process that involves multiple intermediary variables. As such, this task can be segmented and carried out by specialized nanoprocessors separated by distributed memory resources acting as data conduits. The structure fitting is basically dependent on a feedback of $\hat{D}$ between iterations n and n+1 (from m to f in Fig. 3). This translates into a lower bound on the value of n_{ID} that the hardware can offer for any one user (n_{ID}=3).

This pipeline structure is an obvious overdesign for a single user, but seeing as the algorithm is well segmented and globally sequential, interleaved processing of multiple disjoint data sets is possible, and can recover the otherwise idle processor time and thus increase data throughput. We can further exploit this, since only n_{ID}=10 is required, by simply interleaving 10 different users in the STRF while only incurring a 3-symbol latency to each one. Should the upper bound on n_{ID} in [6] be proved too strict, we could potentially relax it further and handle every user in a single STRF

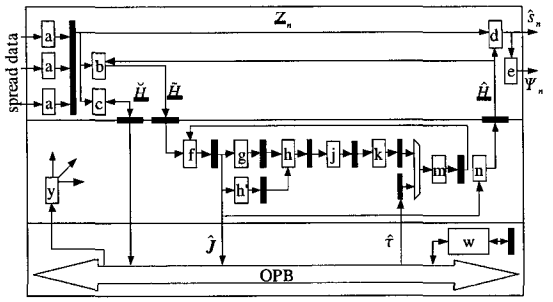


Fig. 3. Three-domain architecture of STAR. The top section shows the symbol path (SP), the middle section is the structure fitting pipeline (STRF) and the bottom section shows the path management subsystem (PM). Black rectangles are block RAM resources. Refer to ID column in Table I for legend.

The required nanoprocessors belong to five categories: despreader, matrix multiplier, FFT processor, linear regression fitter and norm. The different computations require these ressources to be programmable both in terms of data bus bit widths, and operand sizes to suit dynamic operand sizes. The STAR Control Unit ("y" in Fig. 3) dynamically handles these configuration steps through a unidirectional setup bus.

D) Codesign

The PM unit is shown in the the lower part of Fig. 3 and runs within a Microblaze 32-bit soft microprocessor connected to an industry-standard on-chip peripheral bus (OPB). Sampling power levels is done by passive monitoring of STRF memory locations. Once it triggers a path removal or arrival for a user, the PM raises a flag to the STAR control unit so that the STRF pipeline will discard data from the time-delay update processor (Fig. 3-k) for this user and fetch new time-delays ($\hat{\tau}'$) from the PM to the fractional-delay impulse mapper (Fig. 3-m). The fetching operation only takes $8(MP+P)$ clock cycles to complete, much less than any other nanoprocessor would.

The software nature of the path management algorithms also presents the possibility of swapping them in a live system depending on the transmission environment, further enhancing the flexibility of STAR.

E) Data Passing

Two problems arise from inter-domain and intra-pipeline data-passing. First, we must consider the need for the input data from stage q of the pipeline to remain stable (and available) while stage q-1 produces other results. This is handled by using dual-port distributed block RAM (BRAM) resources (two independent read ports, one write port) and allocating a different memory address space to each possible data set (each user). The STAR Control Unit reprograms this information into each nanoprocessor at each pipeline hop.

TABLE I
HARDWARE RESOURCE USAGE FOR STAR (M=1)

Ressource Name	ID	LUTs	MULTs	BRAMs
Symbol Path				
Despreader	a	456	-	-
Constrained DFI	b	566	2	-
Unconstrained DFI	c	566	2	-
Combiner	d	365	4	-
Power Estimator	e	62	3	1
Channel Structure Fitting				
Space-Time Separation	f	382	4	-
Conjugate and Multiply	g	69	2	-
Time Matrix Update (1/2)	h'	417	4	-
Time Matrix Update (2/2)	h	401	4	-
Fast Fourier Transform	j	1039	8	-
Time-delay Update	k	338	5	-
Fractional Impulse Mapper	m	101	-	3
Reconstruction	n	376	4	1
Path Management				
Microblaze and glue logic	w	5929	3	32
Misc				
STAR Control Unit	y	896	1	-
Pipeline framework	-	528	2	36
3G PHY				
Viterbi decoder		~18000	-	4
Cyclic Redundancy Checker		~200	-	-
Interleavers		~100	-	1
Total		~30791	48	78

Inter-domain data-passing as shown in Fig. 3 draws from the same conclusions. A further concern arises from the asynchronous nature of the STRF pipeline with regards to the SP. The exact time required for structure-fitting varies on a number of dynamic factors and cannot be determined with complete accuracy beforehand, so simultaneous access to the same $\hat{H}_n$ can occur from both STRF and SP.

Three factors make this irrelevant. First, the BRAM is programmed with a vendor-specific "read after write" attribute which ensures that no erroneous data will be read should both ports access the same location simultaneously. Second, by architectural choice, the rate at which $\hat{H}_n$ is read by the SP always exceeds that at which it can be written by the STRF, so that there can be no continuous read/write contentions. Third, by the nature of the algorithm, the overall shape of $\hat{H}_{n+1}$ will never evolve by more than the adaptation step size μ (3b) compared to that of $\hat{H}_n$ from one iteration to the next. An overlap could not catastrophically affect the system.

V. HARDWARE RESOURCE UTILIZATION

Our prototype uses a Xilinx Virtex2 6000 FPGA containing in excess of 67,000 look-up tables (LUTs), 144 dedicated block multipliers (MULTs) and 144 18-kilobit block RAM (BRAMs). Table I gives a breakdown of the elements in STAR in terms of FPGA resources. Considering the amount of resources needed by the despreader unit compared to those of the STRF pipeline, the multi-user, single-STRF scenario seems promising. The multiple antenna case only incurs supplementary resources for added despreaders, and slightly increases the STRF cycle time for each user.

Using a Viterbi decoder from the Xilinx LogiCore™ library, parametrized to suit requirements in [4], we find that a dual-rate, single-channel unit would require 18,000 LUTs and 4 BRAMs and could be reused by each user. We further estimate that the remainder of the PHY layer which is comprised of interleavers and a CRC decoder would easily fit in the remaining resources (in excess of 35,000 LUTs). The global resource requirement for a 24-user base station as defined in [4], with a 10% overhead in glue logic would still fit in the afore-mentioned FPGA with a reasonable packing factor of 60%.

VI. CONCLUSION

We have presented the Spatio-Temporal Array-Receiver and shown how it exploits a new receiver paradigm which translates into increased channel identification performance compared to existing array receivers. The global analysis/synthesis approach results in a far more accurate spatio-temporal identification of the channel, thereby dramatically increasing the performance and inherently making STAR a live channel characterization tool. Of particular interest are time-delay extraction and carrier frequency offset recovery which have been addressed in Section IV with real-world channel measurements.

We then outlined the hardware framework needed to realize STAR, and have discussed specifics of its implementation such as partitioning, pipelining, resource reuse and codesign approach. Finally, we have given tangible resource utilization figures along with a breakdown of a potential STAR-enabled 3G base station receiver contained within a single FPGA.

REFERENCES

[1] R. Price and P.E. Green, "A Communication Technique for Multipath Channels", *Proc. IRE*, vol. 46, 1958, pp. 555-570.

[2] G.E. Bottomley, T. Ottoson, and Y.E. Wang, *"A Generalized RAKE Receiver for Interference Suppression", IEEE J. Select. Areas Commun.*, vol. 18, no. 8, 2000, pp.1536-1545.

[3] S. Affes, and P. Mermelstein, "A New Receiver Structure for Asynchronous CDMA: STAR-- The Spatio-0Temporal Array-Receiver", *IEEE J. Select. Areas Commun.*, vol. 16, no. 8, pp.1411-1422, Oct. 1998

[4] 3rd Generation Partnership Project (3GPP), Technical Specification Group (TSG), Radio Access Network (RAN), and Working Group (WG4), "UE Radio Transmission and Reception (FDD), ", TS 25.101, V3.3.0, 2000.

[5] K. Cheikhrouhou et al., "Design Verification and Performance Evaluation of an Enhanced Wideband CDMA Receiver using Channel Measurements", to appear in EURASIP-JASP, 2nd Quarter 2005.

[6] K. Cheikhrouhou, S. Affes, and P. Mermelstein, "Impact of Synchronization on Performance of Enhanced Array-Receivers in Wideband CDMA Networks", *IEEE J. Select. Areas Commun.*, vol. 19, no. 12, December 2001, pp. 2462-2476.

[7] S. Jomphe, J. Belzile, S. Affes, and K. Cheikhrouhou, "Codesign Implementation of a 3G CWCDMA Base Station Receiver", in *Proc. IEEE CCECE'04*, Niagara Falls, Canada, May 2004, pp. 1191-1194.

ANNEXE 4

Démonstration industrielle PROMPT-Québec

Le prototype du récepteur STAR a bénéficié d'une réalisation matérielle complète dans le cadre d'un programme de partenariat stratégique de PROMPT-Québec. Cet organisme subventionnaire a pour mission de favoriser le développement des technologies de télécommunications au Québec, afin de constituer un portfolio consolidé de technologies exploitables. Il se veut une vitrine technologique où partenaires universitaires et indiustriels collaborent dans le cadre de projets soumis à l'organisme et retenus selon l'avis d'un comité formé d'experts externes.

Le volet sans fil de la ronde de projets 2003-2005 réunissait cinq sous-projets suggérés par cinq groupes réunissant chacun un chercheur principal, un partenaire industriel, une équipe de professionnels et plusieurs étudiants, dans des ratios variant selon le projet. Les cinq projets retenus devaient réaliser, dans leur domaine respectif, tous les éléments requis pour fabriquer une chaîne de réception RF avec des technologies de pointe:

- antenne double bande, Tayeb Denidni, Institut national de la recherche scientifique;
- filtre accordable constitué de matériaux ferro-électriques, Ke Wu, École Polytechnique de Montréal;
- convertisseur analogique à numérique auto-calibré à haut débit et faible bruit, Mohamed Sawan, École Polytechnique de Montréal;
- récepteur WCDMA novateur (STAR), Sofiène Affes, Institut national de la recherche scientifique;
- récepteur QPSK avec multi-égaliseur, François Gagnon, École de technologie supérieure.

Une démonstration publique était planifiée pour mettre en valeur le fruit de chaque projet, et a adopté la forme d'un colloque qui s'est tenu le 5 avril 2005 à Montréal. Cet évènement survenait donc au dix-huitième mois de chaque projet, mis à part celui du récepteur STAR (dont fait l'objet de mémoire) qui s'est joint à la cohorte avec six mois de retard. Le projet du professeur Affes a néanmoins participé à part entière à cette démonstration avec

seulement douze mois de travail accomplis. Le schéma simplifié du transmetteur et du récepteur présentés au colloque est illustré à la figure 62.

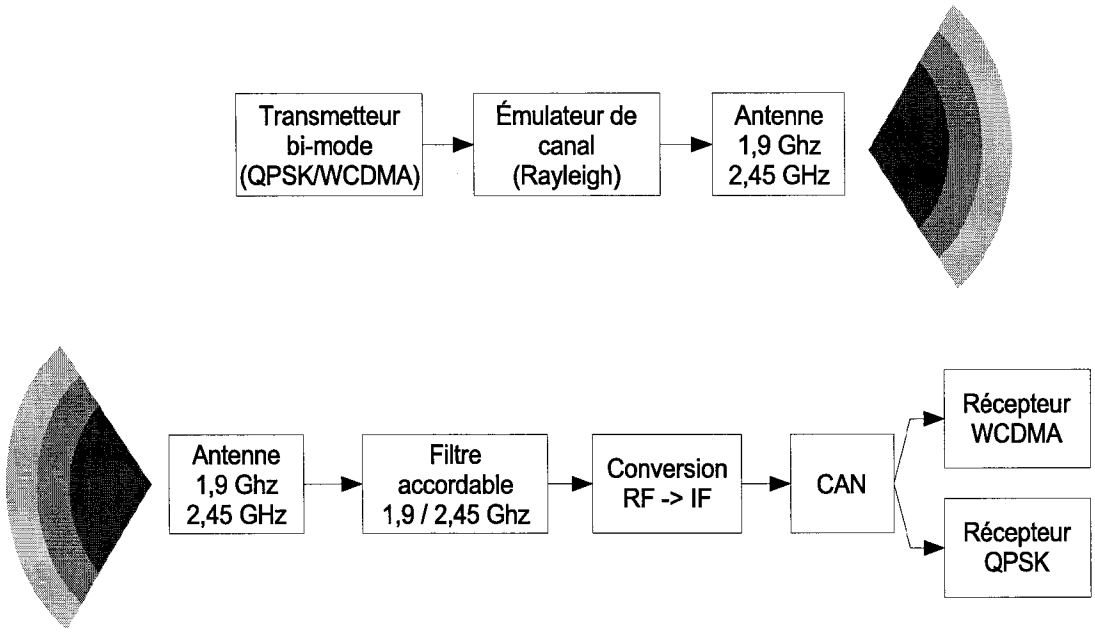


Figure 62 Schéma des chaînes TX et RX présentées au colloque PROMPT (S. Jomphe, Colloque PROMPT-Québec 2005)

La portion du transmetteur comporte deux types de modulations indépendantes l'une de l'autre, et intègre un émulateur de canal qui permet de simuler le comportement d'un canal de Rayleigh. Cette astuce permet d'obtenir à la fois les effets d'un véritable canal dispersif, tout en permettant au transmetteur et au récepteur d'être voisins l'un de l'autre pour des fins de présentation. L'antenne émettrice n'est en effet séparée de l'antenne réceptrice que d'environ cinq mètres, ce qui entraîne néanmoins la captation de bruits RF ambiants, d'interférences provenant des appareils cellulaires dans la salle, et des évanouissements causés par le passage du public entre les deux antennes. La figure 63 donne un aperçu du montage global avec le transmetteur en avant plan à gauche, et les cinq sous-projets constituant le récepteur à l'extrême droite. La figure 64 détaille la section du récepteur.

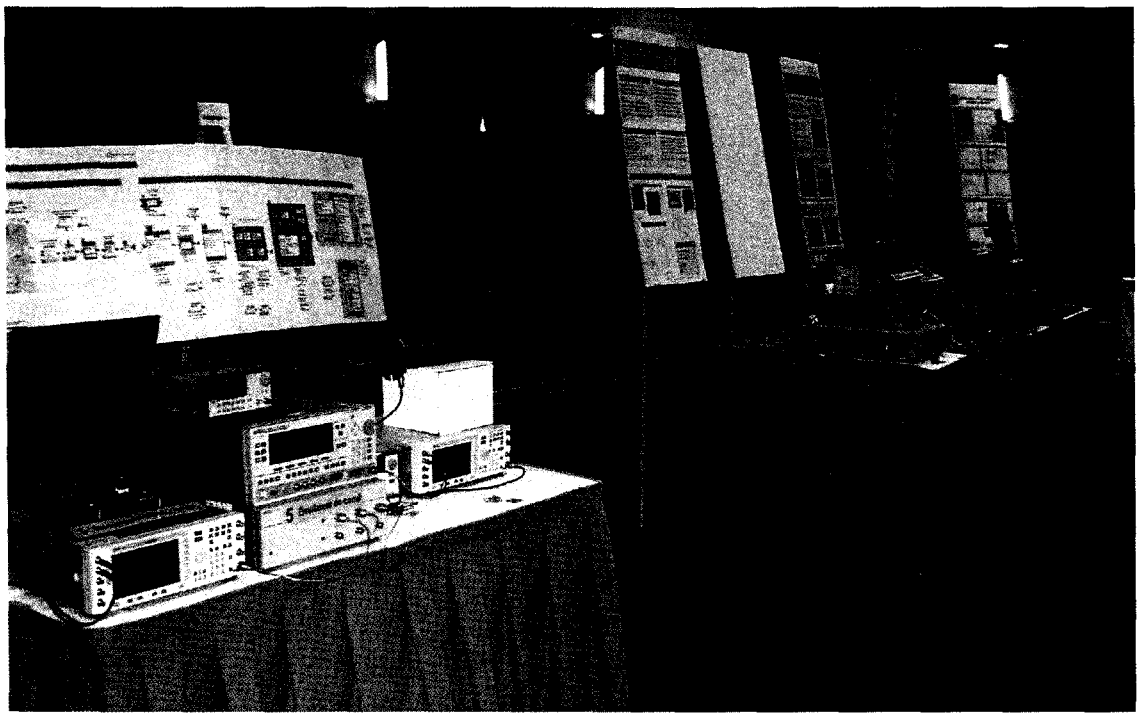


Figure 63 Transmetteur (gauche) et récepteur (droite) complets
(S. Jomphe, colloque PROMPT-Québec 2005)

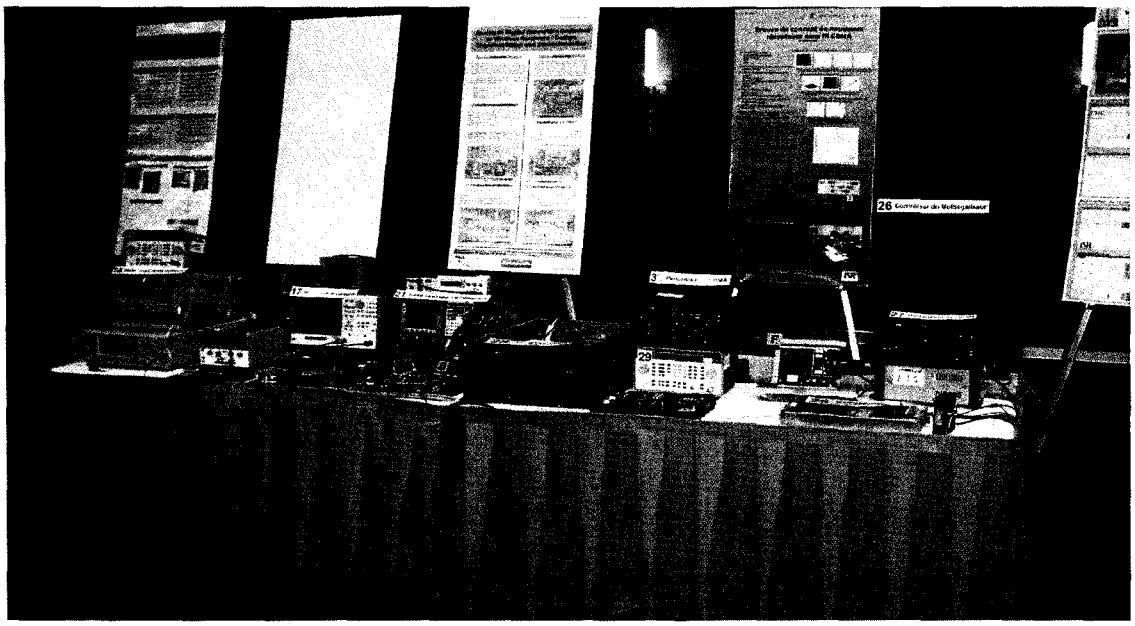


Figure 64 Chaîne de réception bi-mode réunissant cinq sous-projets
(S. Jomphe, colloque PROMPT-Québec 2005)

Les figures 65 à 67 sont des gros plans des sous projets depuis l'antenne réceptrice jusqu'au récepteur STAR dans la plate-forme ICS (telle que décrite au chapitre 5).

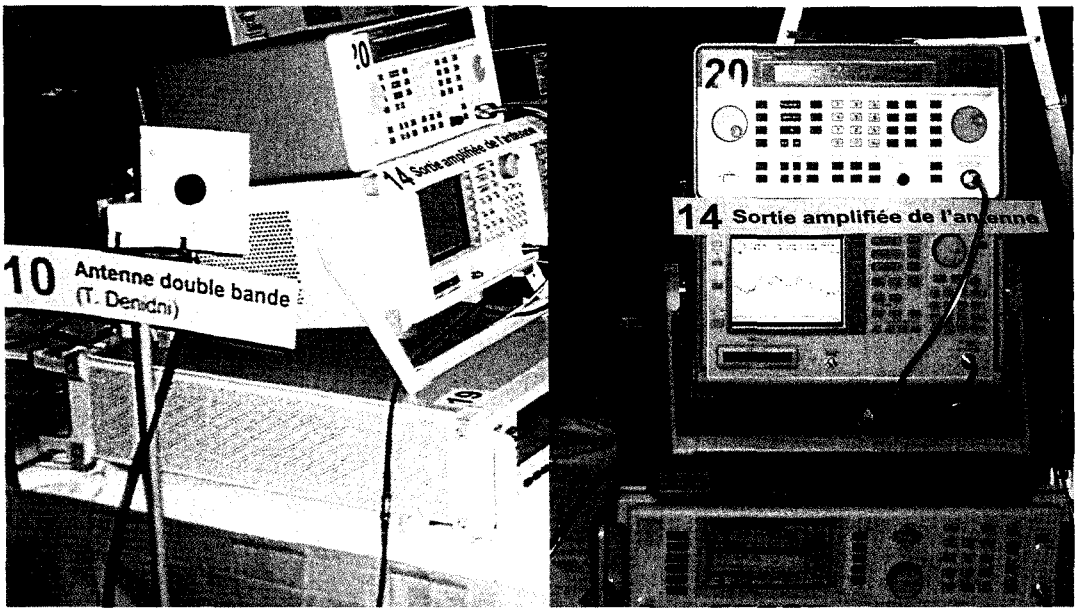


Figure 65 Antenne double bande (no. 10) et spectre amplifié à la sortie de l'antenne (S. Jomphe, colloque PROMPT-Québec 2005)

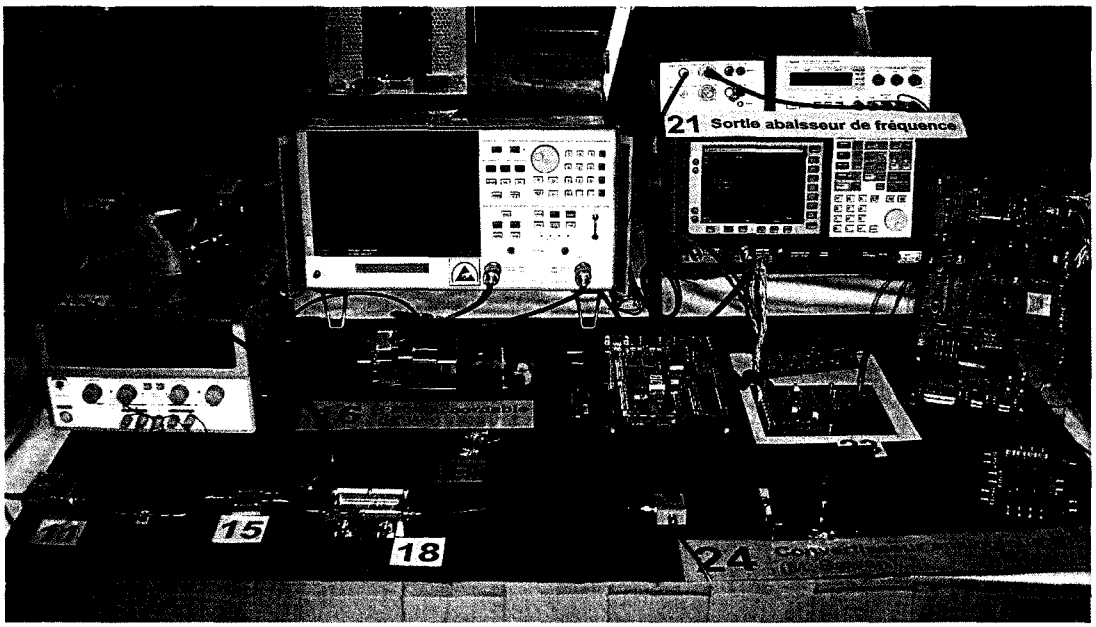


Figure 66 Filtre accordable (no. 16) et convertisseur analogique/numérique (no.24) (S. Jomphe, colloque PROMPT-Québec 2005)

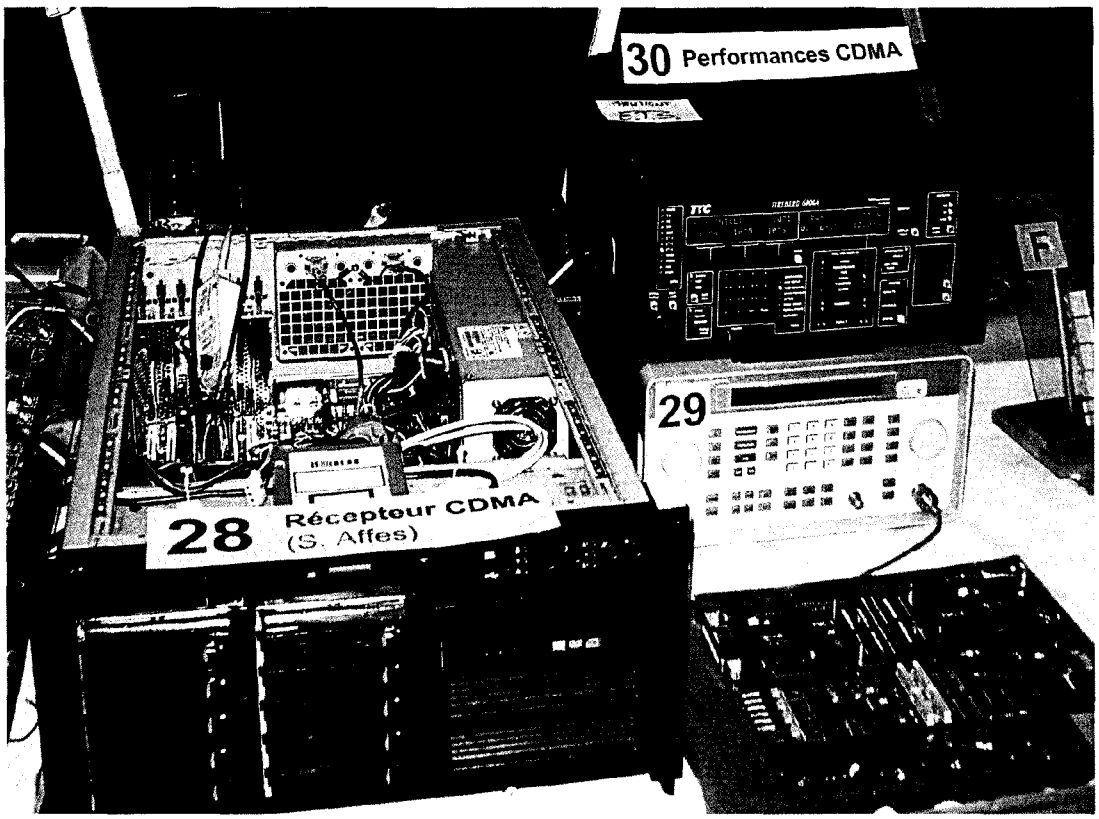


Figure 67 Récepteur STAR sur plate-forme ICS (no. 28)
(S. Jomphe, colloque PROMPT-Québec 2005)

La figure 68 montre le professeur Affes expliquant les rouages du récepteur à l'aide du logiciel STAR Display 5.0, développé par l'auteur et présenté au chapitre 5.

Figure 68 Pr. Affes démontrant le logiciel STAR Display 5.0
(S. Jomphe, colloque PROMPT-Québec 2005)

L'inclusion du récepteur STAR dans le projet d'envergure sans fil de PROMPT-Québec aura permis la réalisation du système et sa validation dans un contexte d'utilisation réel. Ultimement, ce passage de la simulation logicielle vers la réalisation matérielle aura permis de générer une quantité importante de données qui se traduisent, au chapitre 6, par un grand nombre d'analyses révélant l'impact de plusieurs paramètres du système.

Bien que cette étape outrepasse le mandat accordé, l'exercice de réalisation complète du système aura prouvé la validité de l'architecture microélectronique et du modèle de contrôle multi-niveaux présentés de manière théorique aux chapitres 3 à 5.

BIBLIOGRAPHIE

Affes, S., Gazor, S., Grenier, Y., (1996). An Algorithm for Multisource Beamforming and Multitarget Tracking. *IEEE Transactions on Signal Processing, vol. 44, no. 6,* pp. 1512 – 1522.

Affes, S., Mermelstein, P., (1997). Spatio-Temporal Array-Receiver for Multipath Tracking in Cellular CDMA. *Proceedings of the IEEE International Conference on Communications, vol. 3,* pp. 1340 – 1345.

Affes, S., Mermelstein, P., (1998). A New Receiver Structure for Asynchronous CDMA: STAR – The Spatio-Temporal Array-Receiver. *IEEE Journal on Selected Areas in Communications, vol. 16, no. 8,* pp. 1411 – 1422.

Cheikhrouhou, K., Affes, S., Mermelstein, P., (2001). Impact of Synchronization on Performance of Enhanced Array-Receivers in Wideband CDMA. *IEEE Journal on Selected Areas in Communications, vol. 19, no. 12,* pp. 2462 – 2476.

Cheikhrouhou, K., (2003). *Plages dynamiques des matrices et vecteurs de STAR.* Inédit.

Commision of the European Communities, (1989). *Digital Land Mobile Radio Communications (COST-207).* Office for Official Publications of the European Communities : Luxembourg.

Communauté des développeurs Tcl, *Tcl Developer Site*, [En ligne]. http://www.tcl.tk (Page consultée le 26 juin 2005).

Edwards, M.D., Forrest, J., (1996). Software Acceleration using Programmable Hardware Devices. *IEEE Proceedings on Computers and Digital Techniques, vol. 143, no. 1,* pp. 55 – 63.

Grandmaison, M.-E., (2005). *Conception d'un module reconfigurable de FFT.* Mémoire de maîtrise. Montréal : École de technologie supérieure.

Grant, P.M.; Thompson, J.S.; Mulgrew, B., (1998). Adaptive Arrays for Narrowband CDMA Base Stations. *IEEE Journal on Electronics and Communication Engineering, vol. 10, no. 4,* pp. 156 - 166.

Gupta, R.K., De Micheli, G., (1993). Hardware-Software Cosynthesis for Digital Systems. I*EEE Journal on Design and Test of Computers, vol. 10, no. 3,* pp. 29 – 41.

Hennesy, J.L., Patterson, D.A. (1996). *Computer Architecture: A Quantitative Approach (2e éd.).* San Francisco : Morgan Kauffman.

Jantsch, A., (2004). *Network on Chip – A Novel Architecture Template for Integrated Telecommunication Systems.* [En ligne]. http://www.imit.kth.se/info/FOFU/NOC (Page consultée le 29 juin 2005).

Jomphe, S., Belzile, J., Affes, S., Cheikhrouhou, K. (2004a). Codesign Implementation of a 3G Base Station Receiver. *IEEE Proceedings of the Canadian Conference on Electrical and Computer Engineering, vol. 2,* pp. 1191-1194.

Jomphe, S., Belzile, J., Affes, S., Cheikhrouhou, K. (2004b). Codesign Implementation of STAR in a 3G Base Station Receiver. *Proceedings of the 22nd Biennial Symposium on Communications,* pp. 319 - 321.

Jomphe, S. et al., (2005). Area-Efficient Advanced Multiuser WCDMA Receiver. *IEEE Proceedings of the International Northeast Workshop on Circuits and Systems, vol. 3,* pp. 291 - 294.

Khalaj, B.H., Paulraj, A., Kailath, T., (1994). 2D RAKE Receivers for CDMA Cellular Systems. *IEEE Proceedings of the Global Telecommunications Conference, vol. 1,* pp. 400 - 404.

Molisch, A.F., (2001). *Wideband Wireless Digital Communications.* Upper Saddle River, New Jersey : Prentice-Hall.

Noguera, J., Badia, R.M., (2002). HW/SW Codesign Techniques for Dynamically Reconfigurable Architectures. *IEEE Transactions on Very Large Scale Integrations (VLSI) Systems, vol. 10, no. 4,* pp. 400 - 404.

Ojanperä, T., Prasad, R., (1998). *Wideband CDMA for Third Generation Mobile Communications*, Boston : Artech House.

Pei, C.W., Héroux, B., Sweet., J. et al., (2002). High Quality GaAs grown on Si-on-Isulator Compliant Substrates. *Journal of Vacuum Science & Technology – Microelectronics and Nanometer Structures, vol. 20, no. 3,* pp. 1196 – 1199.

Plunkett Research, (2005). *Major Trends Affecting the Wireless Industry.* http://www.plunkettresearch.com/wireless/wireless_trends.htm (Page consultée le 27 juin 2005).

Pospiech, F., Olsen, S., (2003). Embedded software in the SoC world. How HdS helps to face the HW and SW design challenge [hardware dependent software], *Proceedings of the IEEE Custom Integrated Circuits Conference, 21-24 sept. 2003,* pp. 653 – 658.

Prasad, R., (1996). *CDMA for Wireless Personal Communications*, Boston : Artech House.

Price, R., Green, P.E., (1958). A Communications Technique for Multipath Channels. *IRE Proceedings, vol. 46,* pp. 555 – 570.

Proakis, J.G., (1995). *Digital Communications* (3e éd.). États-Unis : McGraw-Hill.

Texas Instruments, (2000). *Implementation of a WCDMA Rake Receiver on a TMS320C62x DSP Device* (Application Report SPRA680). [En ligne]. http://www-s.ti.com/sc/psheets/spra680/spra680.pdf (Page consultée le 11 juillet 2005).

TIA (Telecommunications Industry Association), (1995), *Mobile Station-Base Station Compatibility Standard for Wideband Spread Spectrum Cellular Systems* (TIA/EIA/IS-95-A), États-Unis : TIA.

TWGRAN (Technical Working Group for Radio Access Networks), (2000). *UE Radio Transmission and Reception (FDD)*, (v.3.3.0), 3rd Generation Partnership Project (3GPP), France : 3GPP.

Vai, M.M. (2001). *VLSI Design*, Boca Raton : CRC Press.

Verdú, S., (1986). Optimum Multiuser Asymptotic Efficiency, *IEEE Transactions on Communications, vol. 34*, no. 9, pp. 890 – 897.

Wolf, W.H., (1994). Hardware-software Codesign of Embedded Systems [and prolog], *Proceedings of the IEEE*, vol. 82, no. 7, pp. 967 – 989.