الجـــــــــــــــــــــمـهـوريـــــــة الجـــــــــــــــزائـريـة الديمـــــــقـراطيـة الـشـــــــــعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التـــــــــــــــــــــــــــــعـليـم العـــــــالي و البـــــــــــحث العـــــلـمـــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـــــــــــــــامعة أبي بكـــر بـلـقـايد – تـلمســـــــــــــــان –

Université Aboubakr Belkaïd – Tlemcen –

Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Réseaux et Télécommunications

Par: CHEBOUROU Mohammed & REMAOUN Ahmed Ilyas

Sujet

Modélisation d'un filtre numérique multi-cadence à l'aide d'un DSP TMS320VC5416

Soutenu publiquement, le 24/06/2018, devant le jury composé de :

M. M. BOUSSAHLA	MCB	Univ. Tlemcen	Président
M. S.M.H IRID	MCB	Univ. Tlemcen	Examinateur
M. M. HADJILA	MCB	Univ. Tlemcen	Encadrant
M. F. DERRAZ	MCB	Univ. Tlemcen	Encadrant

Année Universitaire 2017-2018

REMERCIMENTS

Nous remercions Les professeures M.HADJILA et Monsieur F.DERRAZ.

Nous remercions aussi tous nos collègues, en particulier B.RAHMI, I.BENMOKRANE, M.GAAD et tous les autres qui nous ont inspiré de nouvelles idées durant la préparation de ce mémoire.

Nous tenons à remercier tous nos amis (es) pour leur soutien moral dans les moments difficiles.

Nous remercions chaleureusement nos parents, nos frères et sœurs et tous les membres de nos familles qui n'ont cessé de nous encourager.

`

Table des matières

REMERCIMENTS 2

Listes des Figures 5

Liste des tableaux : 8

Liste des abréviations : 9

Résume : 11

Introduction Générale 12

Chapitre I : Rappels sur les concepts du filtrage numérique 14

I.1. Introduction : 15
I.2. Gabarit des filtres : 17
I.3. Classification des filtres numériques : 18
I.4. Choix entre un filtre RIF et IIR : 18
I.5. Filtre à réponse impulsionnelle finie RIF : 19
I.5.1. Réalisation des filtres RIF : 19
I.5.2. Propriétés des filtres RIF : 20
I.5.3. Synthèse des filtres RIF : 20
II.6. conclusion : 22

Chapitre II : Conception des filtres multi-cadence 23

II.1. Introduction : 24
II.2. Décimation : 24
II.2.1. Filtre de décimation : 25
II.2.2. Identité de la décimation : 25
II.3. Interpolation : 27
II.3.1. Filtre d'interpolation : 28
II.3.2. Identités de l'interpolation : 28
II.4. Décomposition polyphasé : 30
II.5. Les bancs des filtres : 32
II.5.1. Bancs de filtre de deux canaux : 33
II.5.2. Banc de filtres multi-cadence à M canaux : 35
II.5.2. Bancs de filtre à mémoire quadrature QMF : 35
II.6. Réalisation de filtre RIF : 36
II.6.1. Structure directe pour les décimateurs RIF : 36
II.6 .2. Structures directes pour les interpolateurs RIF : 37
II.6.3. Structures de réalisations polyphasées pour filtres RIF : 38
II.7. Conclusion : 41

Chapitre III : Processeur de traitement du signal DSP 42

III.1. Introduction : 43
III.2. Présentation d'un DSP : 43
III.3. Système de traitement numérique du signal à base de DSP : 44
III.4. Les caractéristiques d'un processeur DSP : 44
III.5. Applications d'un processeur DSP: 45
Figure III.3 : Domain d'applications d'un DSP 45

III.6. Principales distinctions entre un DSP et un microprocesseur classique : 45
III.7. Mesure de vitesse de calcul pure : 46
III.8. Les type des DSP : 47
III.8.1. Les DSP à virgule fixe : 47
III.8.2. Les DSP à virgule flottante : 49
III.10. Architecture du processeur de signaux numériques : 50
III.10.1. Harvard : 50
III.11. La famille TMS320 de Texas Instruments : 51
III.11.1. La gamme DSP TMS320VC5416 de Texas Instruments : 52
III .12. Conclusion 58
Chapitre IV : Résultats d'implémentation des filtres 59
IV. 1. Introduction : 60
IV.2. Réalisation des filtres numérique : 60
IV.3. Première partie : Réalisation des filtres à réponse impulsionnelle finie (RIF) sous MATLAB. 60
IV.3. 1. Analyse d'un filtre numérique RIF passe-bas du second ordre : 60
IV.3.2. Synthèse d'un filtre RIF : 61
IV.3.3. Décimation et interpolation : 62
IV.4. Deuxième partie : réalisation d'un filtre numérique multi-cadence à l'aide d'une carte DSP TMS320VC5416 sous MATLAB. 64
IV.4.1. Réalisation d'un filtre RIF : 68
IV.4.2. Réalisation d'un filtre RIF de décimation polyphasique : 72
IV.4.3. Réalisation d'un filtre RIF d'interpolation polyphasique : 72
IV.4.4. Implémentation d'un filtre multi-cadence dans une carte Raspberry :) 73
IV.4.5. La conception de base d'un filtre de décimation : 74
IV.4.6. Exécution sur la carte Raspberry : 79
IV.4.7. Lancement de l'exécution avec SIMULINK : 80
IV.4.8. Les résultats à l'exécution du filtre de décimation : 81
IV.5. Conclusion : 82
Conclusion générale 83
Bibliographie : 84
Résume : 85

Listes des Figures

Figure I.1 : La structure générale d'un filtre numérique.

Figure I.2 : Représentation sous forme de fonction de transfert en z.

Figure I.3 : Les différentes représentations sous forme de fonctions de transfert en z.

Figure I.4 : Classification des filtres.

Figure I.5 : Gabarit fréquentiel linéaire.

Figure I.6 : Gabarits des filtres.

Figure I.7 : Réalisation directe de type 1 d'un filtre à réponse impulsionnelle finie.

Figure I.8 : Synthèse de filtre : méthode des fenêtres.

Figure I.9 : la fenêtre Hemming dans le domaine fréquentiel

Figure II.1 : Représentation de la décimation

Figure II.2 : Effets de sous-échantillonnage sur le spectre.

Figure II.3 : Filtre de décimation.

Figure II.4 : Première identité de décimation

Figure II.5 : Deuxième identité de décimation.

Figure II.6 : Troisième identité de décimation.

Figure II.7 : Interpolation d'un signal.

Figure II.8: Effets de sur-échantillonage sur le spectre.

Figure II.9 : Filtre d'interpolation.

Figure II.10 : Première identité d'interpolation.

Figure II.11 : Deuxième identité d'interpolation.

Figure II.12 : Troisième identité d'interpolation.

Figure II.13 : Structure de filtre polyphasé (M=4).

Figure II.14 : Filtres polyphasé de décimation.

Figure II.15 : Filtres polyphasé d'interpolation.

Figure II.16 : (a) Schéma fonctionnel d'un banc de filtres ; (b) Un schéma fonctionnel généralisé d'un système transmultiplexeur.

Figure II.17 : Banc de filtres à 2 canaux multi cadence.

Figure II.18 : Banc de filtres à M canaux multi cadence.

Figure II.19 : Décimateur à facteur de M avec filtre RIF.

Figure II.20 : Implémentations directes d'un décimateur à facteur de M : (a) structure en cascade. b) Structure efficace directe.

Figure II.21 : Implémentations directes d'un interpolateur du facteur L: (a) structure en cascade. (b) Structure efficace directe.

Figure II.22 : Réalisation polyphasée du décimateur RIF : (a) Filtre en cascade. b) Décimateur poly-phasique efficace

Figure II.23 : Réalisation polyphasée d'interpolateur RIF : (a) Filtre en cascade. b) Décimateur poly-phasique efficace.

Figure III.1 : Structure MAC de DSP.

Figure III.2 : Schéma de principe d'un DSP.

Figure III.3 : Domain d'applications d'un DSP

Figure III.4 : représentation de virgule fixe.

Figure III.5 : Exemple de codage fixe avec addition.

Figure III.6 : Exemple de codage fixe avec multiplication.

Figure III.7 : représentation de point flottant.

Figure III.8 : Architecture de Harvard.

Figure III.9 : L'´évaluation des DSP de la famille TMS320

Figure III.10 : schéma fonctionnel du matériel interne 54x

Figure III.11 : TMS320VC5416 Schéma fonctionnel en bloc

Figure III.12 : Carte mémoire de programme et de donnée.

Figure III.13 : Carte mémoire de programme étendue.

Figure IV.1. Résultat d'Analyse d'un filtre numérique RIF passe-bas de second ordre

Figure IV.2. Résultat de Synthèse d'un filtre RIF par la méthode de fenêtrage de Hamming.

Figure IV.3. Résultat de la décimation

Figure IV.4. Résultat d'interpolation de Y2 (t)

Figure IV.5. Première étape de l'installation de support package

FigureIV.6. deuxième étape de l'installation de support package

Figure IV.7. Troisième étape de l'installation de support package

Figure IV.8. Quatrième étape de l'installation de support package

Figure IV.9. Cinquième étape de l'installation du support package.

Figure IV.10. L'interface de CCS

Figure IV.11. Exemple de filtre RIF

Figure IV.12. Signal d'entrée au filtre RIF

Figure IV.13. Résultat d'exemple RIF

Figure IV.14. Exemple de filtre RIF à décimation

Figure IV.15. Spectre du signal source filtre RIF à décimation

Figure IV.16. Spectre de signal à la sortie d'un filtre RIF à décimation

Figure IV.17. Exemple de filtre RIF d'interpolation

Figure IV.18. Signal à l'entrée du filtre RIF d'interpolation

Figure IV.19. Signal à la sortie du filtre RIF d'interpolation

Figure IV.20. Exemple de filtre RIF polyphasique de décimation

Figure IV.21. Spectre à l'entrée et à la sortie du filtre RIF de décimation

Figure IV.22. Exemple de filtre RIF polyphasique d'interpolation

Figure IV.23. Spectre du signal à l'entrée et à la sortie du filtre RIF polyphasique d'interpolation

Figure IV.24. Schéma Simulink basique d'un filtre de Décimation

Figure IV.25. Les paramètres de bloc Sine Wave.

Figure IV.26. Paramètres du bloc ADC

Figure IV.27. Paramètres du bloc Integer to Bit Converter

Figure IV.28. Paramètres du bloc Data Type Convetrsion

Figure IV.29. Contrôleur de gain automatique

Figure IV.30. Paramètres du Décodeur de Viterbi

Figure IV.31. La sortie Raspberry Pi

Figure IV.32. Paramètres du Bloc Matrix Concatenate

Figure IV.33. Interconnexion entre ordinateur et Raspberry Pi

Figure IV.34. Exécution sur Raspberry

Figure IV.35. Fenêtre qui permet d'introduire les données sur la carte

Figure IV.36. Barre de gestion sur Simulink

Figure IV.37. Résultats de l'exécution d'un filtre de Décimation sur Raspberry

Figure IV.38. Composants du bloc Subsystem

Figure IV.39. Spectre du signal avant Décimation

Figure IV.40. Spectre du signal après Décimation

Liste des tableaux :

Tableau I.1 : Le choix entre un filtre RIF et RII.

Tableau III.1 : Les unités les plus courantes de mesures de performance des DSP.

Tableau III.2 : Comparaison enter virgule fixe et virgule flottante.

Tableau III.3 : La disposition standard de la mémoire morte sur puce.

Liste des abréviations :

RII	**R**éponse **I**mpulsionnelle **I**nfinie
RIF	**R**éponse **I**mpulsionnelle **f**inie
FB	**B**anc de **F**iltre
DSP	**D**ensité **S**pectrale de **P**uissance ou **D**igital **S**ignal **P**rocessing
CCS	**C**ode **C**omposer **S**tudio ; logiciel de développement pour DSP avec IDE
MAC	**M**ultiply **a**nd **A**ccumulat
CAN	**C**onvertisseur **A**nalogique **N**umérique
CNA	**C**onvertisseur **N**umérique **A**nalogique
A /N	**A**nalogique/Numérique
N /A	**N**umérique/**A**nalogique
RAM	**R**andom-**A**ccess **M**emory
CPU	**C**entral **P**rocessing **U**nit
DMA	**D**irect **M**emory **A**ccess
DRAM	**D**ynamic **R**andom **A**ccess **M**emory
SRAM	**S**tatic **R**andom **A**ccess **M**emory
McBSP	**M**ulti-**c**hannel **B**uffered **S**erial **P**ort
TI	**T**exas **I**nstument
E/S	**E**ntries/**S**orties
LTI	Linéaire à variation temporelle
TMUX	Transmultiplexeur
TFD	Transformée de Fourrier direct
Fe	Fréquence d'échantillonnage
Te	Période d'échantillonnage
Bp	Bande passante
QMF	Bancs de Filtres à Miroirs quadrature
ALD	Distorsion d'Aliasing
PHD	Distorsion de Phase

AMD	Distorsion d'Amplitude
PR	Reconstruction Parfaite
GPS	**G**lobal **P**ositioning **S**ystem
PLL	**P**hase-**l**ocked **l**oop
GPIO	**G**eneral **P**urpose **I**nput/**O**utput
VLSI	**V**ery-**L**arge-**S**cale **I**ntegration
ALU	**U**nité de **L**ogique **A**rithmétique
EFR	Enhanced Full Rate

Résume :

L'objectif principal de ce mémoire est l'étude et l'implémentation d'un filtre multi-cadence. Un filtre multi-cadence est un filtre numérique qui change la fréquence d'échantillonnage au cours de la chaine de traitement afin qu'elle soit la plus adaptée au traitement à réaliser, en d'autres termes, il modifie le contenu spectral du signal d'entrée en atténuant ou éliminant certaines composantes spectrales non désirées.

Les principaux opérateurs utilisés en traitement numérique du signal multi-cadence sont la décimation (réduction d'un facteur M de la fréquence d'échantillonnage) et l'interpolation (augmentation d'un facteur M de la fréquence d'échantillonnage) ou une combinaison des deux.

Mots clés : Filtrage, Filtre RIF, Filtre RII, DSP, Simulink Model base design

Abstract :

The aim of this document is to study and implement of a multi-cadence filter. A multi-cadence filter is a digital filter that changes the sampling frequency during the processing sequence so enhance the performance. In addition, this modifies the spectral content of the input signal by attenuating or eliminating certain unwanted spectral components.

The main operators used in multi-rate digital signal processing are decimation (reduction by a factor M of the sampling frequency) and interpolation (increase by a factor M of the sampling frequency) or a combination of both.

Keywords: Filtrage, Filtre RIF, Filtre RII, DSP, Simulink Model base design.

الملخص:

الهدف الرئيسي من هذه الرسالة هو دراسة وتنفيذ مرشح متعدد المستويات. المرشح متعدد المعدلات هو مرشح رقمي يغير تردد أخذ العينات أثناء سلسلة المعالجة بحيث يكون هو الأكثر ملاءمة للمعالجة التي يتعين القيام بها ، وبعبارة أخرى ، فإنه يعدل المحتوى الطيفي للإشارة. الإدخال عن طريق تخفيف أو إزالة بعض المكونات الطيفية غير المرغوب فيها.
من تردد M العوامل الرئيسية المستخدمة في معالجة الإشارات الرقمية متعددة المعدلات هي الاستهلال (تخفيض عامل من تردد الاعتيان) أو توليفة على حد سواء M الاعتيان) والاستكمال الداخلي (زيادة عامل.

كلمات البحث: تصفية ، مرشح RIF ، RII ،DSP تصميم قاعدة ، Simulink النموذجي

Introduction Générale

L'une des fonctions principales en traitement numériques du signal est le filtrage. Le filtrage numérique consiste à sélectionner ou supprimer une ou plusieurs composantes fréquentielle. Il est utilisé pour supprimer la partie du signal non désirée qu'on appelle bruit pour ne préserver que l'information pertinente. Nous considérons dans ce mémoire que le spectre du signal est différent de celui du bruit, d'où la notion de déterminisme dans l'approche de conception. La linéarité prévoit que le théorème de superposition est applicable et l'invariance des coefficients du filtre suppose que les signaux en question soient de nature stationnaire. Nous distinguons quatre catégories de filtres : Passe-bas, Passe-haut, Passe-bande et Coupe-bande.

Le développement spectaculaire des circuits intégrés, de plus en plus complexes et puissants, permet une croissance importante des applications en télécommunication sans fil, notamment, la radio mobile, les radars, la vidéo conférence numérique... Ces nouvelles applications nécessitent, généralement, de transmettre une quantité importante d'information à haute vitesse. De plus, l'utilisation de hautes fréquences d'échantillonnage permet de remplacer plusieurs composants analogiques qui sont coûteux et peu flexibles car l'unité de traitement numérique est rapprochée de l'antenne. Ceci a pour but d'augmenter la flexibilité au niveau de la programmation et de la configuration *hardware* pour pouvoir supporter plusieurs standards de communication. Cependant, le traitement numérique de ces signaux en haute fréquence s'avère complexe car les dispositifs de traitement numérique des signaux disponibles ne peuvent assurer une haute fréquence d'échantillonnage de l'ordre des gigahertz (GHz). Un autre défi pour les communications sans fil est la réduction de la puissance consommée. Une des solutions est de réduire la tension d'alimentation. Cette solution engendre une augmentation des délais ce qui limite à son tour la fréquence d'opération. Une solution efficace à ce problème est le traitement numérique multi-cadence.
En effet, la technique de filtrage multi-cadence a été introduite dans le but de réduire la vitesse de calcul dans les filtres numériques. Cette technique inclut les opérations de décimation et d'interpolation qui permettent de modifier la fréquence d'échantillonnage. L'introduction du filtrage multi-cadence a suscité l'intérêt de plusieurs chercheurs tels que R.E. Crochiere et L.R. Rabiner qui ont été les premiers à présenter une conception optimale en terme de minimisation du nombre de multiplicateurs dans les filtres numériques à Réponse Impulsionnelle Finie (RIF) décimateurs multi-étage. Ils ont apporté des contributions au plus haut niveau dans le domaine de filtrage numérique en général et le filtrage multi-cadence en particulier. Cette approche consiste à réduire le nombre de multiplicateurs dans les filtres RIF.

Notre travail a pour objectif de décrire ce domaine important du traitement de signal (filtrage numérique). Pour ce faire, nous avons organisé notre travail en quatre chapitres :
Nous introduisons dans le premier chapitre une étude particulière sur le filtrage numérique tels que les représentations, les structures, les différents types de filtres (RIF, RII), la réalisation des filtres, etc.
Le second chapitre consiste à présenter une étude particulière sur le système de filtrage multi-cadence.
Le troisième chapitre est consacré à la description de la carte TMS320VC5416 à base de DSP et qui

sera utilisée pour implémenter le filtre numérique.
Des résultats sont également présentés au niveau du quatrième chapitre. Nous terminons ce document par une conclusion de ce travail.

Chapitre I : Rappels sur les concepts du filtrage numérique

I.1. Introduction :

Les filtres numériques sont, pour les signaux échantillonnés, les équivalents des filtres analogiques pour les signaux continus. En raison du développement des circuits intégrés rapides, les filtres numériques deviennent plus intéressants que les filtres analogiques en apportant de nombreux avantages : précision, fiabilité, stabilité, adaptabilité et facilité de commande.

Un filtre numérique est un algorithme de calcul qui fait correspondre à une suite d'échantillons $x(n)$ une autre suite d'échantillons $y(n)$. Dans le cas le plus général, l'échantillon $Y(n)$ s'écrit [1] :

$$Y(n) = b_0.x(n) + b_1.x(n-1) + ... + b_q.x(n-q) - a_1.y(n-1) + a_2.y(n-2) + ... + a_p.y(n-p) \quad \text{(I.1)}$$

En utilisant le symbole de sommation, l'équation peut être réécrite de la façon suivante :

$$Y(n) = \sum_{k=0}^{N} b_k.x[n-k] - \sum_{k=1}^{M} a_k.y[n-K] \quad \text{(I.2)}$$

Cet algorithme conduit naturellement à la structure générale d'un filtre numérique [1].

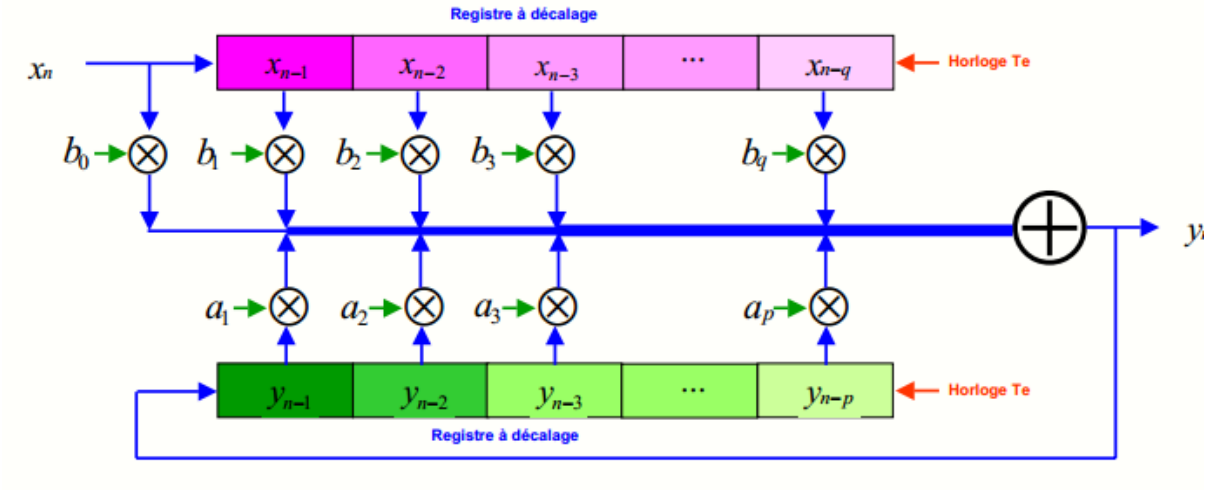


Figure I.1. La structure générale d'un filtre numérique

A une séquence d'échantillons d'un signal d'entrée à temps discret $x(n)$, un filtre numérique, défini par sa réponse impulsionnelle h(n) ou par sa fonction de transfert en z H(z), répond par une séquence d'échantillons d'un signal de sortie $y(n)$.

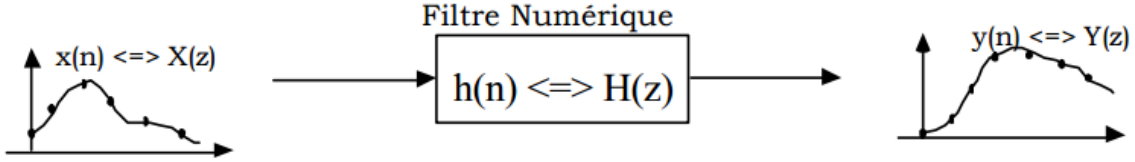


Figure I.2. Représentation sous forme de fonction de transfert en z

Soient X(z) et Y(z) les transformées en z des séquences d'entrée et de sortie. H(z) est la fonction de transfert du filtre, elle est alors définie par [2] **:**

$$H(z)=\frac{y(z)}{x(z)} \qquad \text{(I.3)}$$

La transformée en z permet de trouver la fonction de transfert d'un système discret.

$$H(z)=\frac{y(z)}{x(z)}=\frac{b_0+b_1z^{-1}+b_2z^{-2}+...+b_m\,z^{-m}}{1+a_1z^{-1}+a_2z^{-2}+...+a_nz^{-n}} \qquad \text{(I.4)}$$

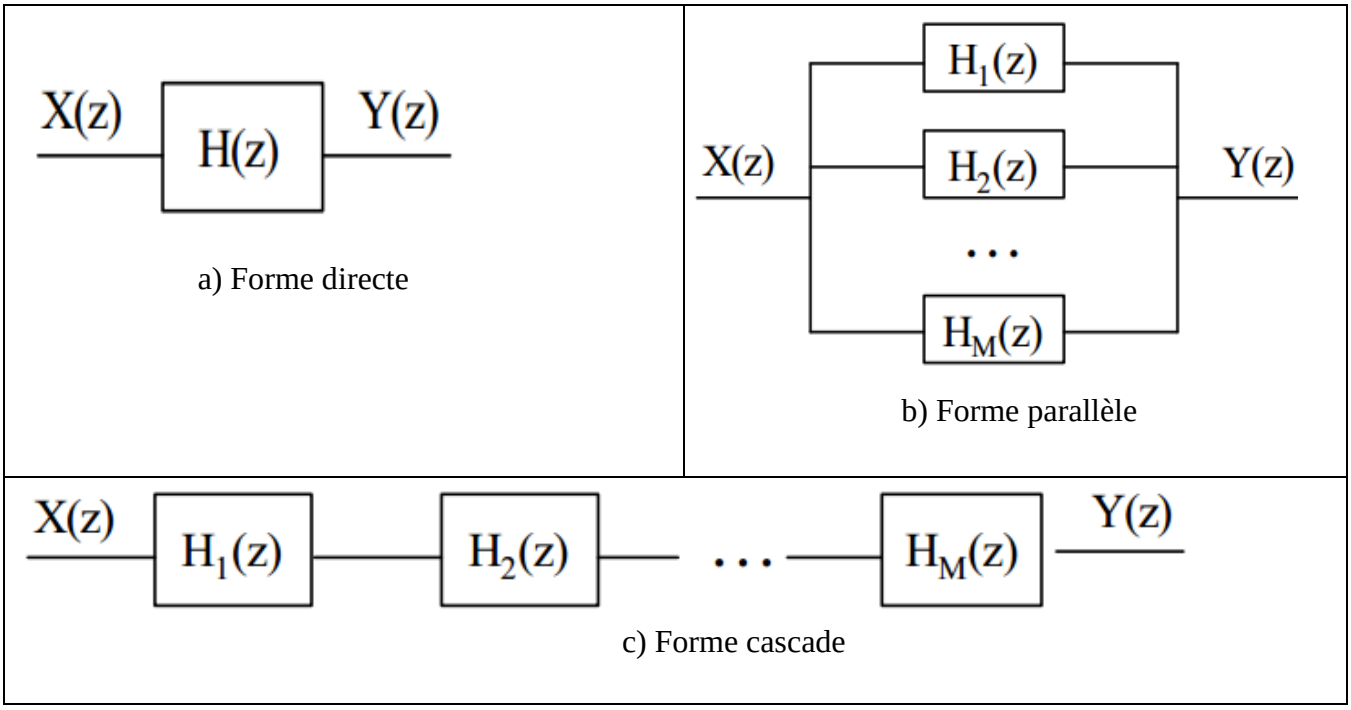


Figure I.3. Les différentes Représentations sous forme de fonctions de transfert en z.

Avant qu'un filtre numérique soit conçu et implanté, nous avons besoin de définir ses spécifications. Un filtre doit laisser passer certaines fréquences, alors qu'il doit en atténuer (voire éliminer) d'autres. Nous devons donc pouvoir représenter ses contraintes. Il y a quatre filtres de bases [2] :

- Les filtres passe-bas laissent passer les fréquences inférieures à une fréquence de coupure f_c et bloquent celles qui lui sont supérieures (voir figure (**I**.4. a)).
- Les filtres passe-haut bloquent les fréquences inférieures à une fréquence de coupure f_c et laissent passer celles qui lui sont supérieures (voir figure (**I**.4. b)).
- Les filtres passe-bande laissent passer les fréquences autour d'une fréquence centrale f_0 (ou comprises entre f_1 et f_2) et bloquent les autres (voir figure (**I**.4. c)).
- Les filtres réjecteur-de-bande bloquent les fréquences autour d'une fréquence centrale f_0 (ou comprises entre f_1 et f_2) et laissent passer les autres (Figure (**I**.4. d)).

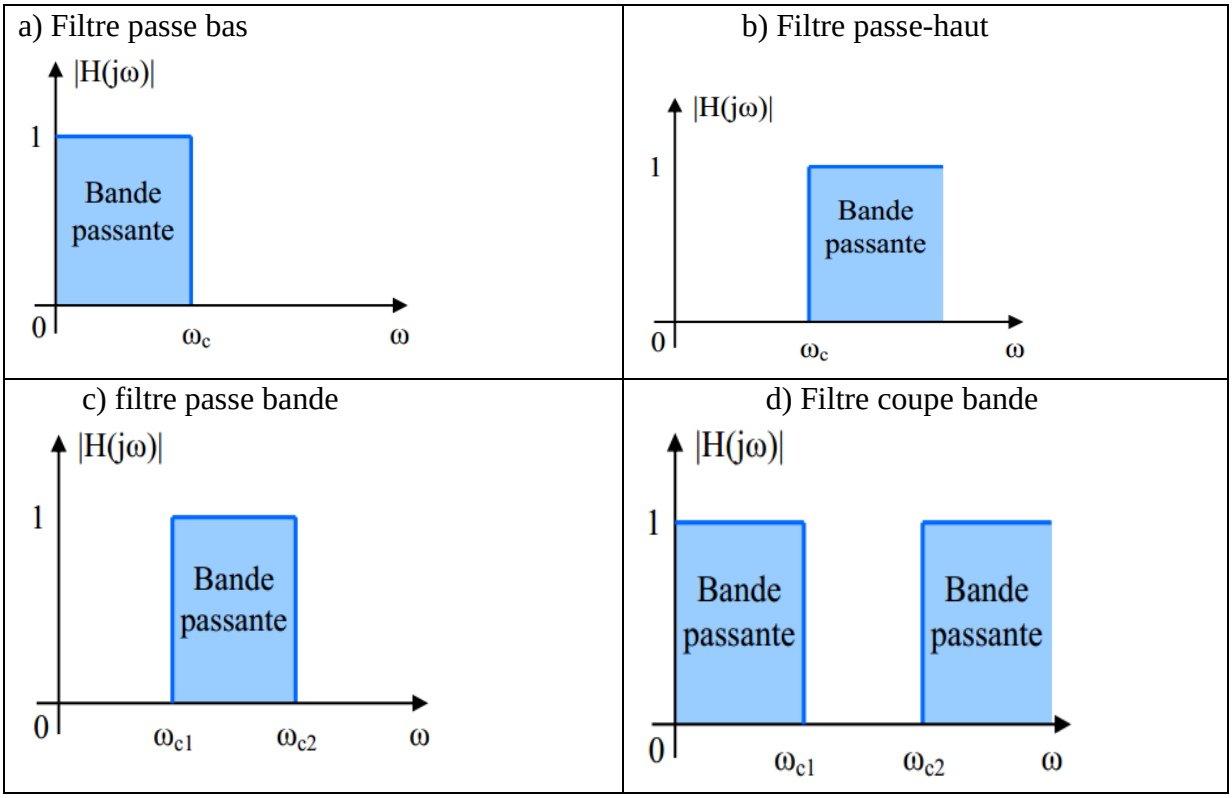


Figure I.4. Classification des filtres

I.2. Gabarit des filtres :

On peut définir des filtres passe bas, passe haut, passe bande, coupe bande à l'aide de gabarit dans le domaine fréquentiel.

Comme le filtre idéal n'est pas réalisable, on spécifie un filtre à l'aide d'un gabarit qui donne les tolérances dans les différentes bandes de fréquence.

On distingue trois bandes de fréquence différentes : la bande de transition **($f_p \leq f \leq f_a$)**, la bande passante **($0 \leq f \leq f_p$)**, et la bande atténuée **($f_a \leq f \leq f_e/2$)**

(Voir figure (**I**.5)) [3],

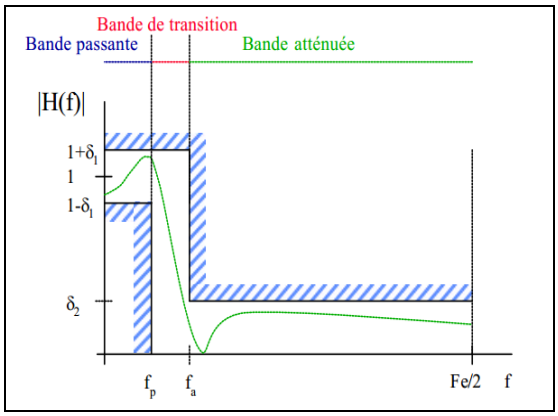


Figure I.5. Gabarit fréquentiel linéaire

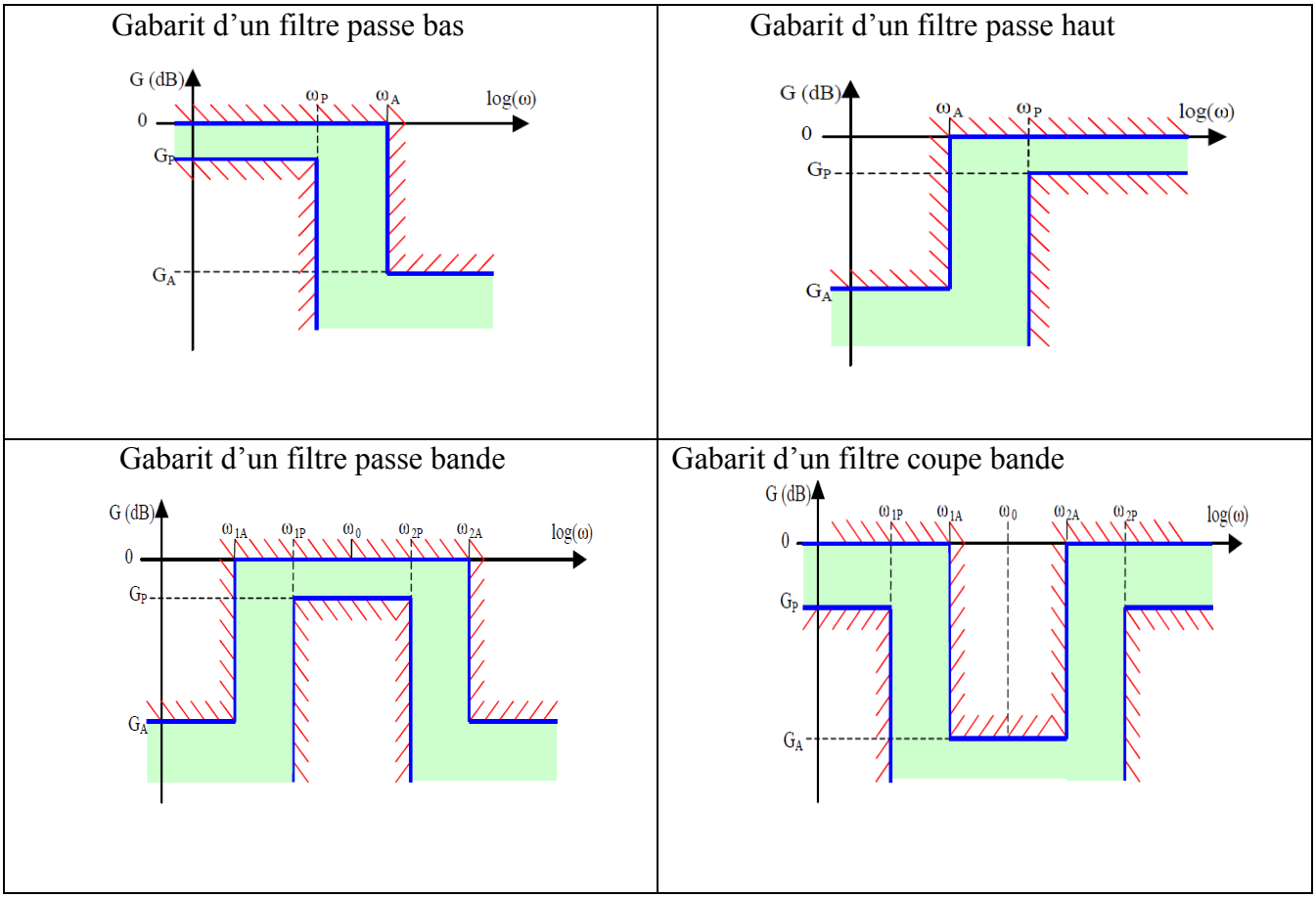


Figure I.6. Gabarits des filtres

I.3. Classification des filtres numériques :

Les filtres numériques peuvent être classés selon plusieurs critères :

1. la longueur de la réponse impulsionnelle implique deux types de filtres RII et RIF.
2. le type de représentation, ou de structure, implique deux types de filtres récursifs et non récursifs [4].

I.4. Choix entre un filtre RIF et IIR :

Le choix entre un filtre RIF et IIR dépend :

- Des performances recherchées ;
- De l'application ;
- De la vitesse du processeur ;
- De la mémoire RAM disponible.

Tableau I.1 : Le choix entre un filtre RIF et RII.

Critère	R.I.F	R.I.I
Maîtrise de la phase	Oui	Non
Complexité	Très faible	Faible
Stabilité	Calcul possible par TFD Toujours	Risque de problème en cas de précision de calcul insuffisante
Nombre de coefficients nécessaires	Moyen	Faible
Précision nécessaire pour les calculs	Moyenne	Assez grande
Adapté à la multi-cadence	Oui	Non

I.5. Filtre à réponse impulsionnelle finie RIF :

Les filtres RIF ne peuvent pas être dérivés des filtres analogiques. Ils sont cependant très largement utilisés car ils possèdent des propriétés uniques (phase linéaire, stabilité, flexibilité).

De façon générale, le filtre à réponse impulsionnelle finie est décrit par la combinaison linéaire suivante où $x[i]\,1\leq i\leq n$ représente les valeurs du signal d'entrée et $y[i]\,1\leq i\leq n$ les valeurs du signal de sortie [2].

$$y(n) = b_0\, x[n] + b_1.x[n-1] + b_2.x[n-2] + \ldots + b_N .x[n-N] \qquad (I.5)$$

En utilisant le symbole de sommation, l'équation peut être réécrite de la façon suivante [3] :

$$y(n) = \sum_{k=0}^{N} b_k.x[n-k] \qquad (I.6)$$

La fonction de transfert en z est donnée par [3] :

$$H(z) = \sum_{k=0}^{N-1} b_k.z^{-k} \qquad (I.7)$$

I.5.1. Réalisation des filtres RIF :

Les filtres numériques peuvent être réalisés à l'aide de trois éléments ou opérations de base. Soit l'élément gain, l'élément de sommation et le retard unitaire. Ces éléments sont suffisants pour réaliser tous les filtres numériques linéaires possibles [4].

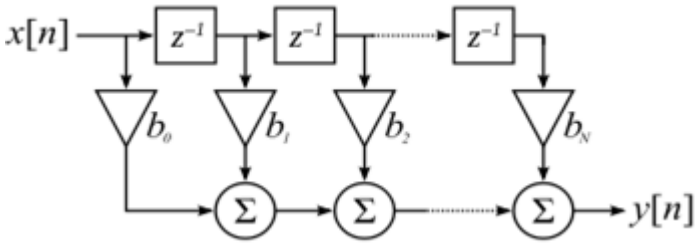


Figure I.7. Réalisation directe de type 1 d'un filtre à réponse impulsionnelle finie

I.5.2. Propriétés des filtres RIF [5] :

- Les RIF sont toujours stables (système tout zéros).
- Une plus grande facilité d'implantation dans un système numérique de traitement.
- Une phase qui peut être exactement linéaire, par conséquent un temps de propagation de groupe constant et une absence de distorsion harmonique dans le signal.
- A sélectivité équivalente, ils sont toujours plus coûteux (en temps de calcul) que leur équivalent RII.

I.5.3. Synthèse des filtres RIF :

Un filtre RIF est défini par sa fonction de transfert en z :

$$H(z) = \sum_{k=0}^{N-1} b_k . z^{-k} \qquad (\mathbf{I}.8)$$

A la différence des filtres RII, les filtres RIF ne sont réalisables que dans le domaine discret. Par conséquent, leurs méthodes de synthèse ne sont pas dérivées des filtres analogiques. On distingue trois principales méthodes, dont deux seront détaillées dans les sections suivantes [3] :

1. la méthode des fenêtres ;
2. la méthode de l'échantillonnage en fréquence ;
3. Méthodes d'optimisation.

I.5.4.1). Méthode des fenêtres :

La réponse impulsionnelle $h_d(n)$ associée à un filtre désiré donner par sa réponse en fréquence $h_d(f)$ peut être obtenue soit par un calcul de transformée de Fourier inverse à temps discret selon l'équation (1.9), soit par un calcul de transformée de Fourier inverse selon l'équation (1.10) suivi d'un échantillonnage de la fonction $h_d(t)$ obtenue [4]:

$$h_d(t) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \bar{h}_d(f) e^{j2\pi ft} df \qquad (\mathbf{I}.9)$$

$$h_d(n) = h_d(nT_e) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \bar{h}_d(f)\, e^{j2\pi fnT_e}\, df \qquad (\mathbf{I}.10)$$

Te et $Fe = 1/Te$ étant respectivement la période et la fréquence d'échantillonnage. Cette réponse impulsionnelle $h_d(n)$ étant généralement de support infini, la méthode de fenêtrage consiste à effectuer une troncature de la séquence $h_d(n)$ en la multipliant par une fenêtre appropriée w(n) de taille finie, dite fonction fenêtre, pour obtenir un filtre RIF dont la réponse impulsionnelle est donnée par :

$$h(n) = h_d(n).w(n) \qquad (\mathbf{I}.11)$$

La fenêtre intuitive que nous pouvons appliquer pour tronquer la réponse impulsionnelle est la fenêtre rectangulaire :

$$w(n) = \begin{cases} 1 \Rightarrow n = 0,1,...,N \\ 0 \Rightarrow ailleurs \end{cases} \qquad (\mathbf{I}.12)$$

Ainsi, par une simple troncature des coefficients de $h_d(n)$ nous ne retenons que les N +1 premiers échantillons de $h_d(n)$ pour obtenir un filtre RIF h(n) d'ordre N :

$$h(n) = \begin{cases} h_d(n), \Rightarrow n = 0,1,...,N \\ 0 \Rightarrow ailleurs \end{cases} \qquad (\mathbf{I}.13)$$

Dans le domaine fréquentiel, cette opération de fenêtrage se traduit par une convolution du spectre $h_d(f)$ du filtre désiré avec le spectre $W(f)$ de la fenêtre utilisée. Ainsi, le spectre du filtre synthétisé est donné par la relation :

$$\bar{h}(f) = \bar{h}_d(f) \times w(f) \qquad (\mathbf{I}.14)$$

Avec $h_d(f)$et $W(f)$ étant périodiques de période F_e,

$$\bar{h}(f) = \frac{1}{F_e} \int_{-\frac{F_e}{2}}^{\frac{F_e}{2}} \bar{h}_d(\theta) w(f-\theta) d\theta \qquad (\mathbf{I}.15)$$

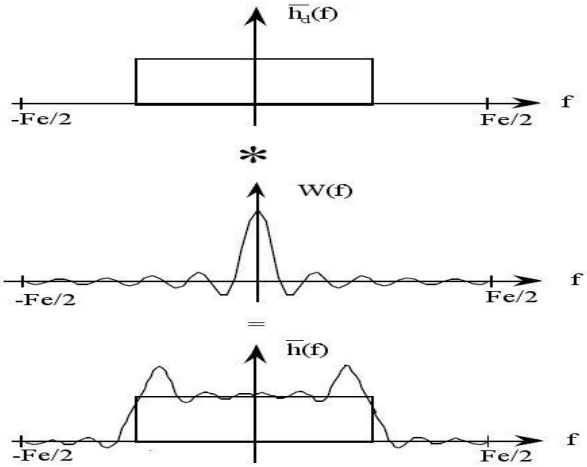


Figure I.8. Synthèse de filtre : méthode des fenêtres

Fenêtre de Hemming [5] :

$$y(n) = 0,54 - 0,46 * \cos\left(\frac{2\pi n}{N-1}\right) \qquad (\mathbf{I}.16)$$

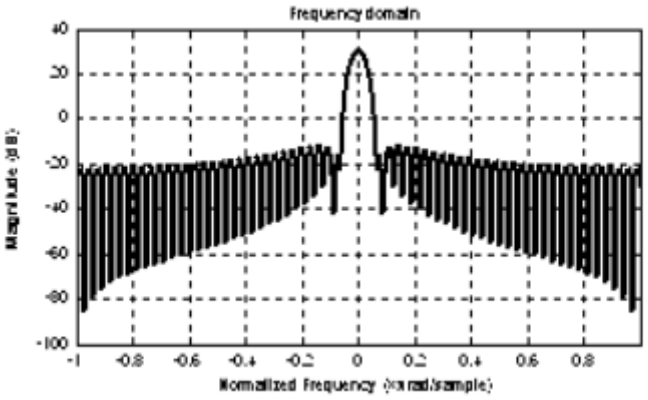


Figure I.9. La fenêtre Hemming dans le domaine fréquentiel

II.6. conclusion :

Avec l'amélioration rapide de la technologie informatique, le traitement du signal numérique est devenu plus important. Par conséquent, le problème de conception de filtres numériques a reçu beaucoup d'attention. Ces filtres ont de nombreuses applications importantes, par exemple, dans les systèmes radar, sonar, de traitement du signal et du traitement d'images.

Chapitre II : Conception des filtres multi-cadence

II.1. Introduction :

La conversion de la fréquence d'échantillonnage signifie qu'un signal discret est converti en un autre signal discret avec une fréquence d'échantillonnage différente. Deux signaux discrets avec des fréquences d'échantillonnage différentes peuvent être utilisés pour transmettre la même information. Par exemple, un signal continu $x_t(t)$ limité en bande passante peut être représenté par deux signaux discrets différents $x(n)$ et $y(n)$ obtenus par échantillonnage uniforme du signal original $x_c(t)$ avec deux fréquences d'échantillonnage différentes F_e et F_e' [6].

$$x(n) = x_c(n.T) \Rightarrow \text{avec} \Rightarrow y(n) = x(n.T) \qquad (\mathbf{II}.1)$$

Où $T= 1/F_e$ et $T' =1/ F_e'$ sont les intervalles d'échantillonnage correspondants. Lorsque les fréquences d'échantillonnage F_e et F_e' sont choisies de telle sorte que chacune d'entre elles dépasse au moins deux fois la fréquence la plus élevée du spectre de $x_c(t)$, le signal original $x_c(t)$ peut être reconstitué à partir de $x(n)$ ou $y(n)$. Par conséquent, les deux signaux fonctionnant à deux fréquences d'échantillonnage différentes sont porteurs de la même information. En utilisant les opérations en temps discret, le signal $x(n)$ peut être converti en $y(n)$, ou vice versa, avec des distorsions de signal minimales.

Les opérations de base dans le processus de modification de la fréquence d'échantillonnage sont la diminution de la fréquence d'échantillonnage (décimation) et l'augmentation de la fréquence d'échantillonnage (interpolation). L'utilisation de deux opérateurs permet d'effectuer la modification de la fréquence d'échantillonnage [6].

II.2. Décimation :

L'opération de sous-échantillonnage avec un facteur de sous-échantillonnage M, où M est un entier positif, est une opération consistant à éliminer **M-1** échantillons successifs sur M échantillons de la séquence d'origine [7]. Soit un signal continu $x_c(t)$ échantillonné à la période T et représenté par la séquence $\boldsymbol{x(n) = xc(n.T)}$, on posera donc le résultat de la décimation de $x(n)$ d'un facteur **M** par la relation suivante : $y(m) = x(m.M)$ (**II**.2)

La notion graphique utilisée pour la décimation est **M** ↓ (voir la figure II.1). Si la fréquence d'échantillonnage en entrée du décimateur est **Fe**, alors celle de sortie vaudra **Fe'=Fe /M**, la période d'échantillonnage en sortie est [2] :

$$T'=M*T \qquad (II.3)$$

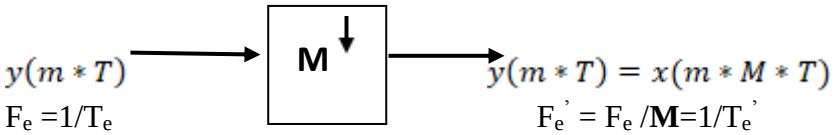


Figure II.1. Représentation de la décimation

Il y a une relation entre le spectre de $x(n)$ et $y(n)$; cette relation est :

$$y(e^{jw}) = \frac{1}{M}\sum_{k=0}^{M-1}(X.e^{\frac{j(w-2\pi k)}{M}}) avec : Z = e^{jw} \qquad (\mathbf{II}.4)$$

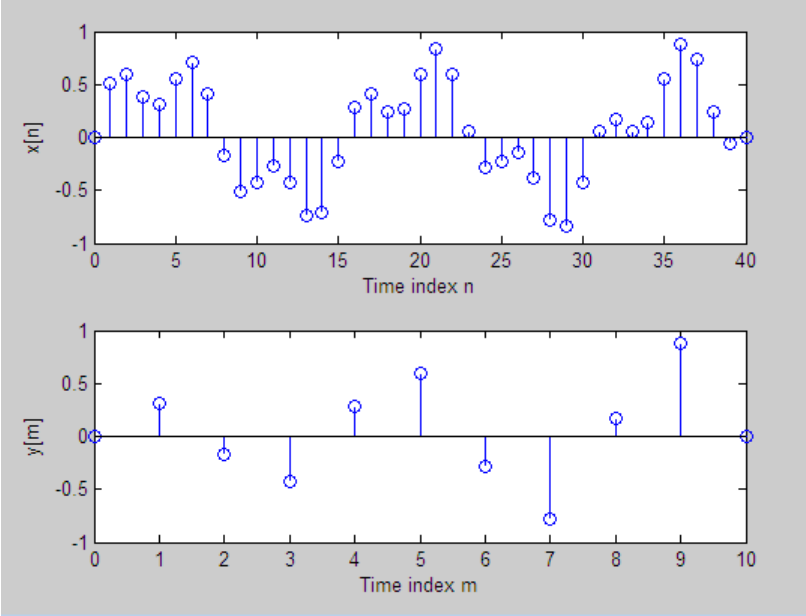


Figure II.2. Effets de sous-échantillonnage sur le spectre.

II.2.1. Filtre de décimation :

Avant de procéder au sous-échantillonnage avec le facteur **M**, il faut donc limiter la bande passante du signal original à π/M, dont le but est d'éviter l'alias [8]. Cela signifie que la décimation du facteur de **M** doit être mise en œuvre en deux étapes [6] :

1. Limitation de bande passante du signal original à $\frac{Bp}{M}$.
2. Échantillonnage descendant par le facteur de **M**.

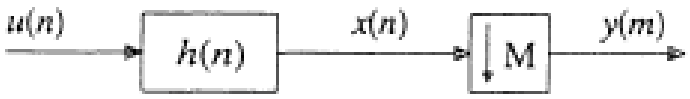


Figure II.3. Filtre de décimation.

II.2.2. Identité de la décimation :

Il y a 3 identités de base liées au traitement du signal à plusieurs débits. Lorsqu'elles sont correctement utilisées, elles améliorent l'efficacité des systèmes à taux multiples [9].

a. <u>Première identité :</u>

La première identité est indiquée à la figure II.4. La mise à l'échelle des signaux dans les branches, leur addition au niveau du nœud et sous- échantillonnage équivalent à un sous- échantillonnage des signaux avant la mise à l'échelle et l'addition [9].

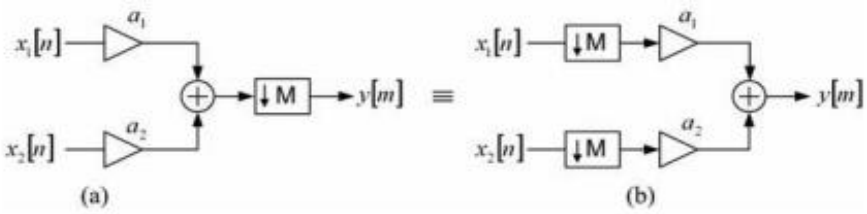


Figure II.4. Première identité de décimation

b. Deuxième identité :

La deuxième identité indique que le délai par **M** suivi d'un sous-échantillonneur par **M** équivaut au sous-échantillonneur par M suivi d'un délai par un (voir Figure **II**.5) [9].

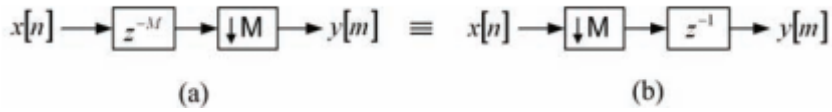


Figure II.5. Deuxième identité de décimation.

c. Troisième identité :

Une propriété remarquable de la décimation est que la sortie d'un système composé par un filtre suivi par un décimateur est la même que celle d'un décimateur suivi d'un filtre [7] (voir figure II.6).

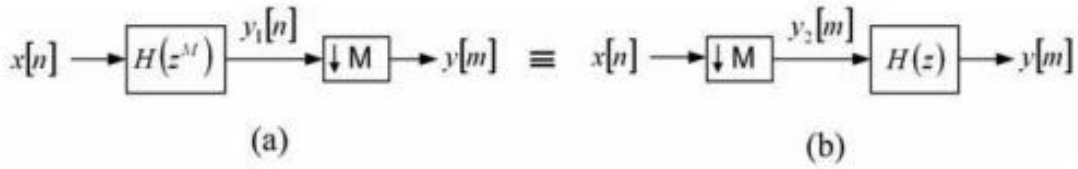


Figure II.6. Troisième identité de décimation.

Sur la partie gauche de la (figure II.5), une entrée x(n) est filtrée par un filtre de la fonction de transfert H(z). La transformée en z est donnée par [9] :

$$Y_1(z) = x(z).\mathrm{H}(z^M) \qquad (\mathbf{II}.5)$$

La sortie de décimation est donnée par [1]:

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} u.(e^{j(w-2pk)/\mathrm{M}}) = \frac{1}{M}\sum_{k=0}^{M-1} X.(e^{j(w-2pk)/\mathrm{M}}).\mathrm{H}(e^{j(w-2pk)/M}) = \frac{1}{M}\sum_{k=0}^{M-1} X.(e^{j(w-2pk)/\mathrm{M}}).H(z) \quad (\mathbf{II}.6)$$

Sur la partie droite de cette figure II.5, l'entré x(n) est d'abord décimée, la transformée en z de la sortie U(n) du décimateur est donc [9] :

$$Y_2(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(e^{j(w-2pk)/M}) \qquad (\mathbf{II}.7)$$

Puis la sortie du décimateur est filtrée par un filtre de fonction de transfert H(z). La transformée en z de la sortie y(n) est donc :

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(e^{\frac{j(w-2pk)}{M}}).H(z) \qquad (\mathbf{II}.8)$$

II.3. Interpolation :

On définit par élévateur de fréquence, le système décrit à la figure II.6 consistant en l'ajout de L - 1 zéros entre deux échantillons successifs de la séquence d'entrée x(n) définie par [7] :

$$u(n) = \begin{cases} (\frac{n}{L}), \text{Þ } n = m*L \\ 0 \text{Þ } ailleurs \end{cases} \qquad (II.9)$$

La notation graphique utilisée est ↑ L (voir figure II.7). Si la fréquence d'échantillonnage en entrée de l'élévateur de fréquence est Fe, alors celle de sortie vaudra Fe' = Fe.L. La période d'échantillonnage en sortie vaudra quant à elle T ′ = T/L [2].

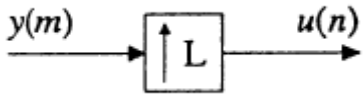


Figure II.7. Interpolation d'un signal

La transformée en z du signal sur-échantillonné est donnée par :

$$Y(z) = \sum y(m).z^{-m} \qquad (II.10)$$

$$U(z) = \sum u(n).z^{-n} \qquad (II.11)$$

$$U(z) = y(z^L) \qquad (II.12)$$

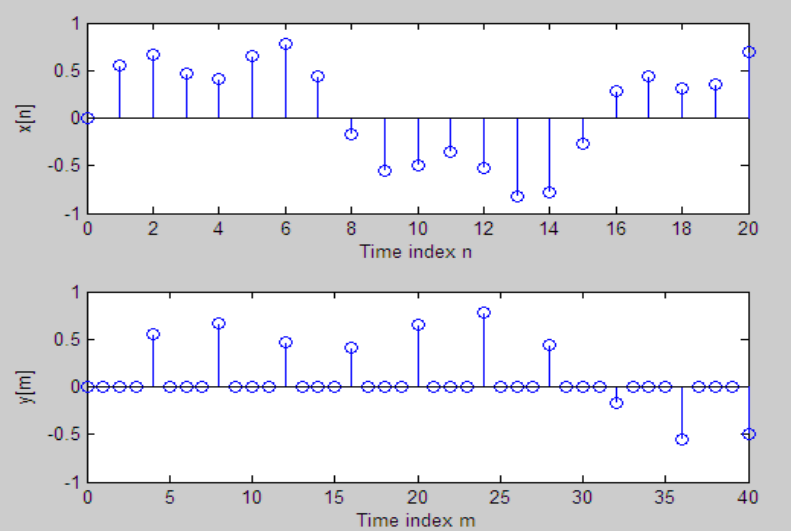


Figure II.8: Effets de sur-échantillonage sur le spectre.

II.3.1. Filtre d'interpolation :

En général, un filtre numérique anti-repliement est placé après un interpolateur (voir figure II.9). Cela signifie que l'interpolation factorielle doit être réalisée en deux étapes :

1. Échantillonnage du signal d'origine en insérant des échantillons de valeur zéro L-1 entre deux échantillons consécutifs de type échantillons [9].

2. Suppression des images **L-1** du spectre du signal sur-échantillonné [2].

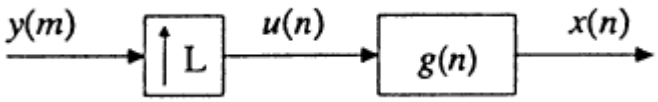


Figure II.9. Filtre d'interpolation

II.3.2. Identités de l'interpolation :

Il y a 3 identités de base liées au traitement du signal à plusieurs débits. Lorsqu'elles sont correctement utilisées, elles améliorent l'efficacité des systèmes à taux multiples.

a. Première identité :

Elle est indiquée à la figure II.10. Le sur-échantillonnage avant la ramification et la mise à l'échelle équivaut à la ramification et la mise à l'échelle avant le sur-échantillonnage [9].

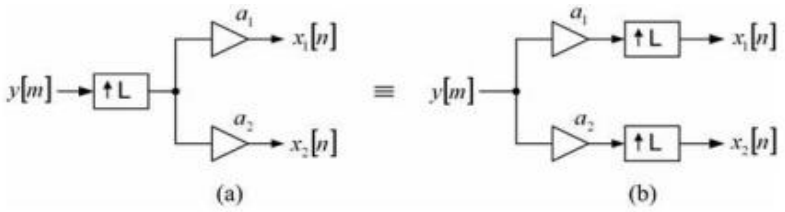


Figure II.10. Première identité d'interpolation

b. Deuxième identité :

Elle indique que le signal retardé par 1 et échantillonné par **L** est équivalent au signal sur-échantillonné par **L** et retardé par 1 (voir figure II.11) [9].

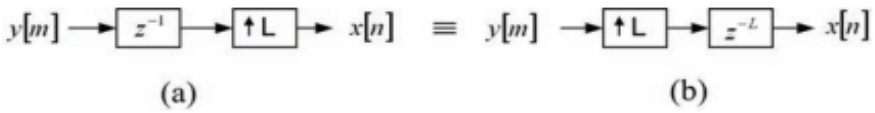


Figure II.11 : Deuxième identité d'interpolation

c. Troisième identité :

Une propriété remarquable d'interpolation est que la sortie d'un système composé par un filtre suivi par une interpolation est la même que celle d'interpolation suivie d'un filtre (voir figure II.12) [7].

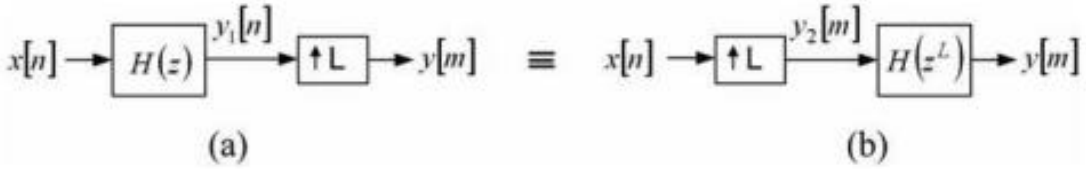


Figure II.12. Troisième identité d'interpolation

Pour les signaux de la Figure II.12 (a), nous avons :

$$Y_1(z) = H(z).X(z) \qquad (\mathbf{II}.13)$$

$$Y(z) = Y_1(z^L) = H(z^L).X(z) \qquad (\mathbf{II}.14)$$

Pour les signaux de la Figure II.12 (b), nous avons :

$$Y_2(z) = X(z^L) \qquad (\mathbf{II}.15)$$

$$Y(Z) = Y_2(z^L).H(z^L) = X(z^L).H(z^L) \qquad (\mathbf{II}.16)$$

II.4. Décomposition polyphasé :

La décomposition polyphasée d'une réponse à l'impulsion permet de réaliser une fonction de transfert équivalente avec un système dont la plupart des éléments fonctionnent à une cadence réduite [6]. Cependant, les structures polyphasées introduites ici sont d'une importance fondamentale pour les bancs de filtres que nous examinerons plus loin [2].

Toute réponse à l'impulsion peut être décomposée comme suit [10] :

$$H(z) = ... + h(0) + h(M).z^{-M} + h(2M).z^{-2M} + ... + h(M-1)z^{-(M-1)} + h(2M+1).z^{-(2M-1)} + ... \quad \text{(II.17)}$$

Qui devient, en mettant des délais en commun :

$$\begin{aligned} H(z) &= \left[... + h(0) + h(M)z^{-M} + h(2M)z^{-2M} + ... \right] \\ &+ z^{-1}\left[... + h(1) + h(M+1)z^{-M} + h(M+2)z^{-(M+1)} + ... \right] + \\ &z^{-(M-1)}\left[... + h(M-1) + h(2M+1)z^{-M} + ... \right] \end{aligned} \quad \text{(II.18)}$$

Donc :

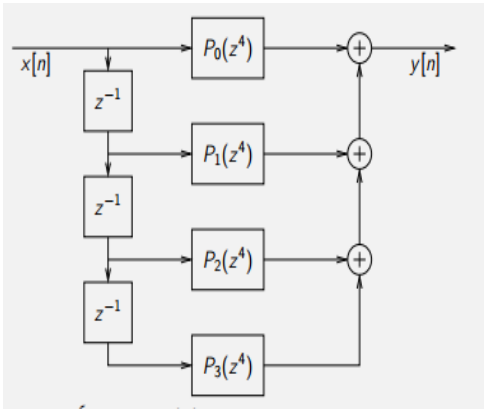
$$\begin{aligned} H(z) &= \sum_{k=0}^{M-1} z^{-k}\left[... + h(-M+K)z^{M} + h(k) + h(M+K)z^{-M} + h(2M+K)z^{-2M} \right] \\ H(z) &= \sum_{k=0}^{M-1} z^{-k} P_k(z^M) \end{aligned} \quad \text{(II.19)}$$

Où

$$P_k(z) = \sum_{n=0}^{\infty} h[n.M + k]z^{-n} \quad \text{(II.20)}$$

- Chaque ligne de la décomposition polyphasée constitue une réponse à un taux *M* fois plus faible, déphasée d'un échantillon par rapport à la ligne précédente.
- Il s'agit de la décomposition polyphasée en **M** composantes. Les $P_k(z)$ constituent les composantes polyphasées de H (z) [10].

(a) : Structure direct



(b) : Structure transposée

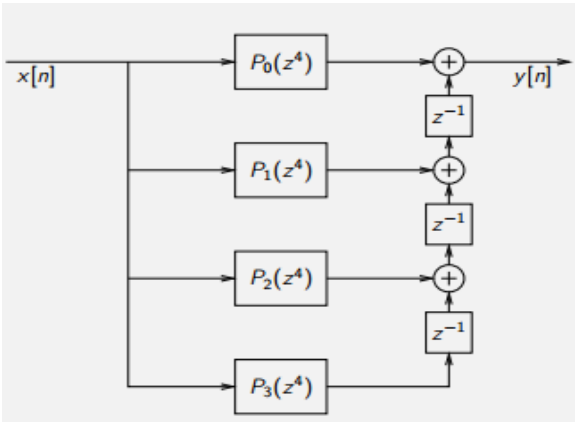


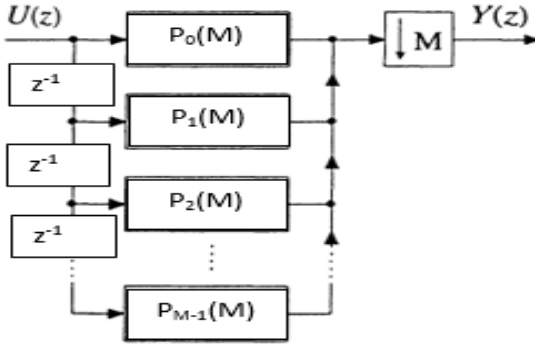
Figure II.13. Structure de filtre polyphasé (M=4)

Pour $M = 4$, la décomposition polyphasé prend la forme figure II.13 :

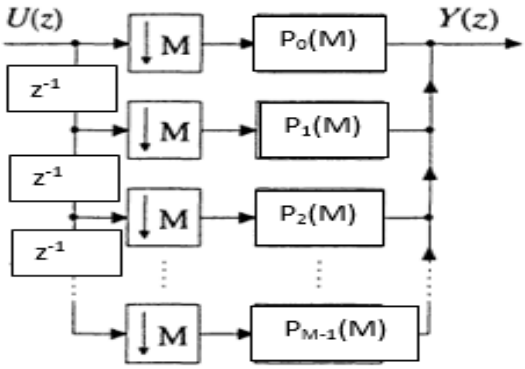
$$Y(z) = H(z).X(z) \qquad \text{(II.21)}$$

$$Y(z) = P_0\left(z^4\right)X\left(z\right) + z^{-1}P_1\left(z^4\right)X\left(z\right) + z^{-2}P^2\left(z^4\right)X\left(z\right) + z^{-3}P_3\left(z^4\right)X\left(z\right) \qquad \text{(II.21.a)}$$

$$\mathrm{Y(z)} = P_0\left(z^4\right)X\left(z\right) + z^{-1}\left[P_1\left(z^4\right)X\left(z\right) + z^{-1}\left[P_2\left(z^4\right)X\left(z\right) + z^{-1}P_3\left(z^4\right)X\left(z\right)\right]\right] \qquad \text{(II.21.b)}$$

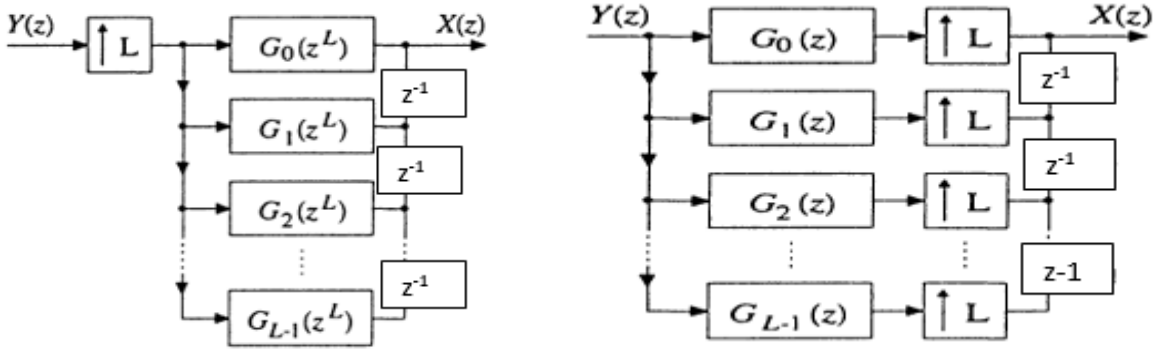


(a) Structure direct



(b) Structure transposée

Figure II.14. Filtres polyphasés de décimation



(a) Structure direct. (b) Structure transposée.

Figure 2.15. Filtres polyphasés d'interpolation

$$G(z) = G_L Z^L \qquad \text{(II.22)}$$

II.5. Les bancs des filtres :

Les bancs de filtres se composent d'un système linéaire invariant (LTI) et d'opérations de sous-échantillonnage et sur-échantillonnage. Les systèmes multi-cadences sont classés en deux types selon leur mode de fonctionnement. Le premier type correspond à une structure de bancs de filtres dans laquelle le signal d'entrée original est subdivisé en différentes bandes ou canaux, selon les exigences d'une application donnée, tandis que le second type correspond à une structure de transmultiplexeur (TMUX), dans laquelle plusieurs signaux sont combinés et transférés par le même canal dans un système de communication. La figure II.16 présente un schéma fonctionnel généralisé de ces structures. Dans la Figure II.16 (a), le signal d'entrée est décomposé en différentes sous-bandes à l'aide d'un banc de filtres ; ici, un signal d'entrée est analysé, de sorte que ces filtres sont également appelés filtres d'analyse, et du côté du récepteur, ces sous-bandes sont synthétisées en un signal reconstruit, de sorte qu'on les appelle filtres de synthèse. En raison de cette opération, cette structure est également connue sous le nom de système d'analyse/synthèse multi-cadences. Dans la Figure II.16b, plusieurs signaux d'entrée sont filtrés par un ensemble de filtres, également appelés filtres de synthèse et combinés en un signal composite, qui est à nouveau filtré en plusieurs signaux côté récepteur par un banc de filtres, également appelés filtres d'analyse [11].

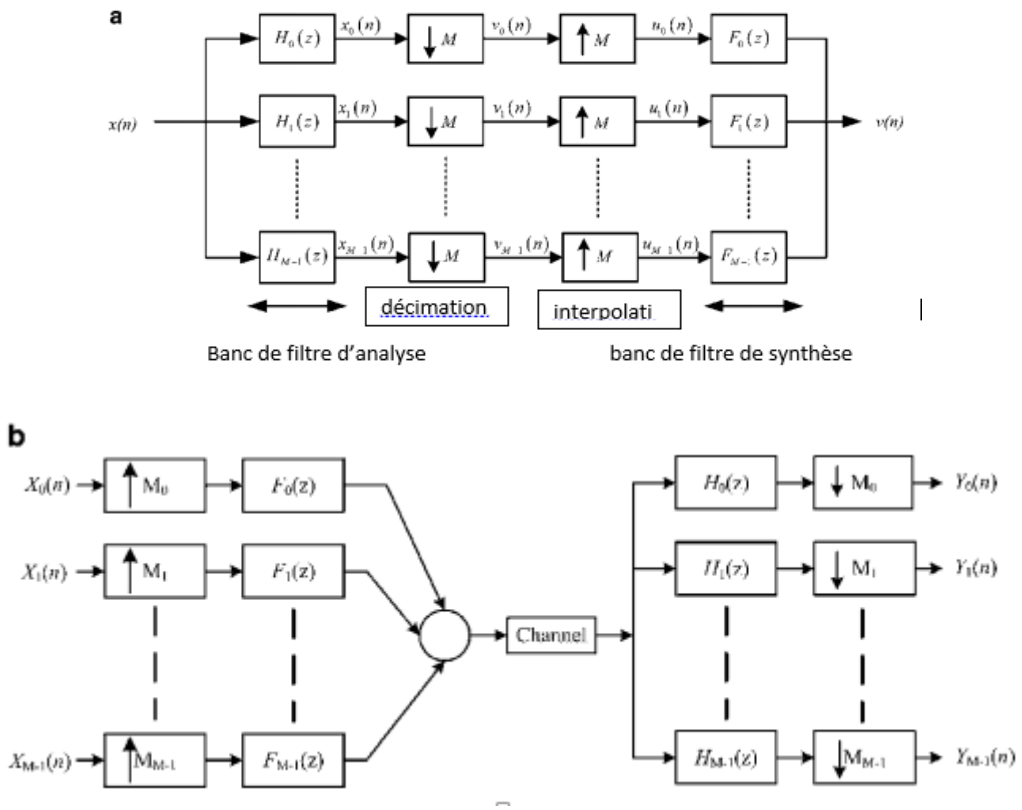


Figure II.16. (a) Schéma fonctionnel d'un banc de filtres ; (b) Un schéma fonctionnel généralisé d'un système transmultiplexeur

Le banc de filtre BF numérique est l'amalgame de différents filtres passe-bande avec une entrée partagée et une sortie additionnée, qui sont principalement utilisés pour la synthèse et l'examen de spectres différents du signal. Par conséquent, les BF regroupent les BF d'analyse et les BF de synthèse, comme le montre la figure II.16.

L'analyse BF se compose de sous-filtres, appelés filtres d'analyse. Les filtres d'analyse sont utilisés pour diviser le signal d'entrée en sous-bandes dissemblables dans le domaine fréquentiel. Chaque sous-bande comprend une certaine part de fréquence du signal original. De même, la synthèse BF comprend les sous-filtres appelés filtres de synthèse, qui combinent les signaux de sous-bande et génèrent le signal ou reconstruisent le signal. Ces BF peuvent être classés en deux types : les BF à deux canaux et les BF à canal M, en fonction du nombre de canaux utilisés pour la séparation des signaux [11].

II.5.1. Bancs de filtre de deux canaux :

En général, le BF à deux canaux divise un signal d'entrée en deux sous-bandes et se compose d'un filtre d'analyse et de synthèse ainsi que d'une unité de traitement entre les BF. Ainsi, le processus élémentaire d'un BF à deux canaux comporte deux étapes séquentielles en l'absence d'unité de traitement : le fonctionnement des bancs d'analyse et des bancs de synthèse [12].

Un schéma fonctionnel général de BF à deux canaux est illustré à la Figure II.17. A partir de l'analyse des relations d'entrée et de sortie, la sortie du BF à deux canaux est écrite comme suit :

$$X_0(z) = X(z).H_0(z) \quad avec \quad X_1(z) = X(z).H_1(z) \qquad (\mathbf{II}.23)$$

Le sous-échantillonnage avec M = 2,

$$V_0(z) = \frac{1}{2} X(z^{(1/2)}).H_0(z^{(1/2)}) + \frac{1}{2} X(-z^{(1/2)}).H_0(-z^{(1/2)}) \qquad (\mathbf{II}.24)$$

$$V_1(z) = \frac{1}{2} X(z^{(1/2)}).H_1(z^{(1/2)}) + \frac{1}{2} X(-z^{(1/2)}).H_1(-z^{(1/2)}) \qquad (\mathbf{II}.25)$$

Le sur-échantillonnage avec M = 2 ;

$$Y(z) = G_0(z).V_0(z^2) + G_1(z).V_1(z^2) \qquad (\mathbf{II}.26)$$

Donc :

$$Y(z) = \frac{1}{2}\left[G_0(z).H_0(z) + G_1(z).H_1(z)\right]X(z) + \frac{1}{2}\left[G_0(z).H_0(-z) + G_1(z).H_1(-z)\right]X(-z) \qquad (\mathbf{II}.27)$$

$$Y(z) = F_0(z).X(z) + F_1(z).X(-z)$$

Avec :

$$F_1(z) = \frac{1}{2}\left[G_0(z).H_0(-z) + G_1(z).H_1(-z)\right] \qquad (\mathbf{II}.28)$$

Et :

$$F_0(z) = \frac{1}{2}\left[G_0(z).H_0(z) + G_1(z).H_1(z)\right] \qquad (\mathbf{II}.29)$$

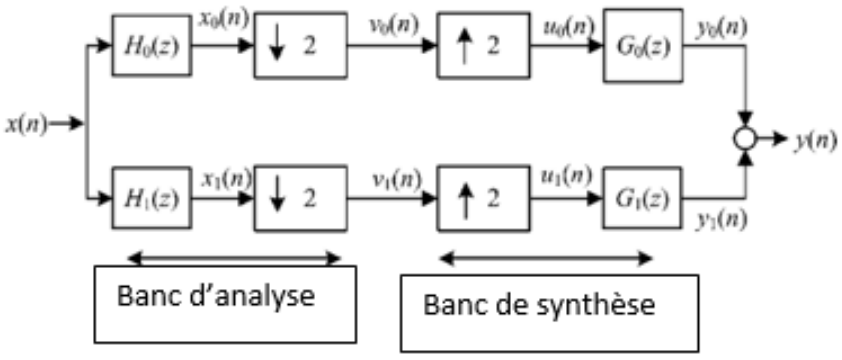


Figure II.17. Banc de filtres à 2 canaux multi cadence

II.5.2. Banc de filtres multi-cadence à M canaux :

Le canal M BF divise le signal en M nombre de sous-bandes. En fonction de la largeur de bande des canaux, les BF du canal M sont classées en deux catégories : les BF uniformes et les BF non uniformes [11] (figure II.18).

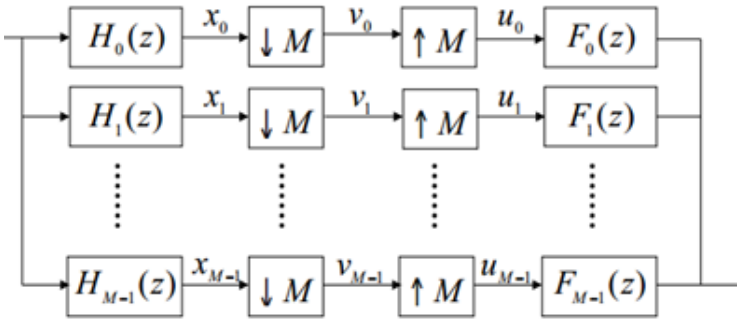


Figure II.18. Banc de filtres à M canaux multi-cadence

II.5.2. Bancs de filtre à mémoire quadrature QMF :

Le banc de filtre à miroirs quadrature à deux canaux (QMF) est une structure de filtres numériques à débits multiples qui se compose de deux décimateurs dans la section d'analyse du signal et de deux interpolateurs dans la section de synthèse du signal. Ces bancs de filtres ont d'abord été utilisés dans le codage de sous-bandes, où le signal est divisé en deux ou plusieurs sous-bandes dans le domaine des fréquences, de sorte que chaque signal de sous-bandes peut être traité de manière indépendante. Les bancs de QMF trouvent des applications dans divers domaines, tels que, codage d'image, systèmes de modulation multi porteuse, analyse spectrale bidimensionnelle à court terme, systèmes d'antenne, progrès de la théorie de l'échantillonnage, ...

La figure II.19 montre les sections d'analyse et de synthèse d'un banc QMF à deux canaux. Le signal d'entrée discret $x(n)$est divisé en deux signaux de sous-bande comprenant largeur de bande égale, en utilisant les filtres d'analyse passe-bas et passe-haut respectivement H_1 (z) et H_2 (z). Les signaux de sous-bande sont décimés par un facteur de deux pour obtenir une compression du signal ou réduire la complexité du traitement. Les sorties des filtres de synthèse sont combinées pour obtenir le signal reconstruit $x(n)$. Le signal reconstruit $x(n)$souffre de distorsion d'aliasing (ALD), de distorsion de phase (PHD) et de distorsion d'amplitude (AMD) du fait que les filtres d'analyse et de synthèse ne sont pas idéaux. Par conséquent, l'accent principal de la plupart des chercheurs, en concevant le prototype de filtre pour le banc de QMF à deux canaux, a été sur l'élimination ou la minimisation de ces trois erreurs afin d'obtenir un système de reconstruction parfaite(PR) [11].

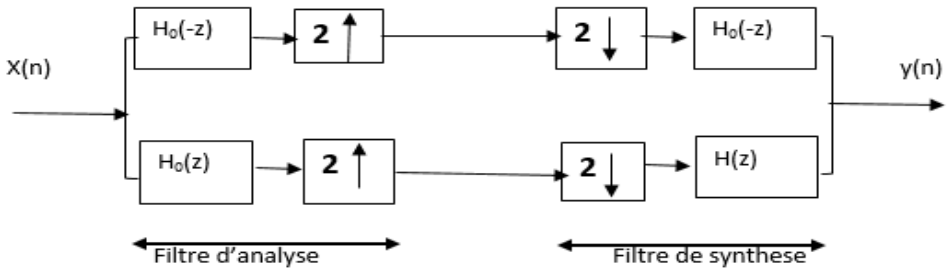


Figure II.19. Filtre à mémoire quadrature QMF

II.6. Réalisation de filtre RIF :

La nature non récursive des filtres RIF offre la possibilité de créer des schémas d'implémentation qui améliorent significativement l'efficacité globale des décimateurs et interpolateurs RIF.

II.6.1. Structure directe pour les décimateurs RIF :

Considérons le décimateur du facteur de **M** de la figure II.20, qui utilise un filtre RIF avec la réponse impulsionnelle $h[n]$. Les relations d'entrées-sorties du domaine temporel pour le filtre sont exprimées par la relation [6] :

$$v(n) = \sum_{k=0}^{N-1} h(k).x(n-k) \qquad (\mathbf{II}.30)$$

Le signal décimé $y[m]$ est obtenu après application de l'opération de décimation sur le signal à l'état dégradé $v[n]$ est :

$$y(m) = v(m*M) \qquad (\mathbf{II}.31)$$

Pour l'évaluation du signal décimé $y[m]$, nous n'utilisons que chaque échantillon M^n du signal décimé. Par conséquent, il est suffisant de ne calculer que tous les échantillons de $v[n]$, c'est-à-dire :

$$v(m*M) = \sum_{k=0}^{N-1} h(k).x(m*M-k) \qquad (\mathbf{II}.32)$$

A partir des équations (**II**.30), (**II.**31) et (**II**.32) :

$$y(m) = \sum_{k=0}^{N-1} h(k).x(m*M-k) \qquad (\mathbf{II}.33)$$

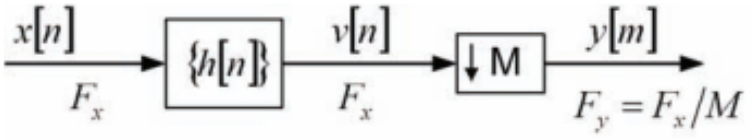
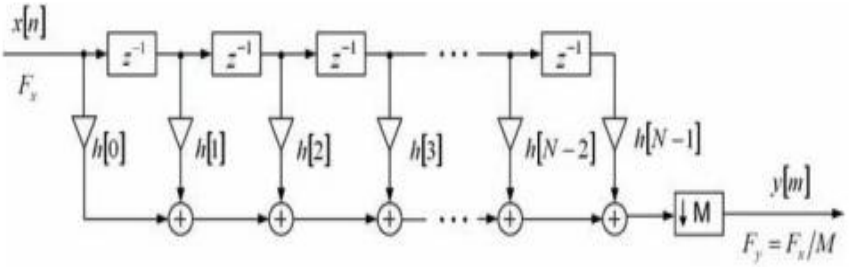
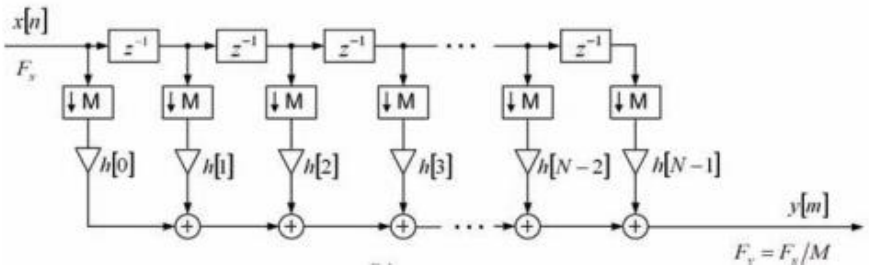


Figure II.20. Décimateur à facteur de M avec filtre RIF

Deux structures de mise en œuvre directe pour un facteur de décimateur **M** sont illustrées dans les figures II.21 (a) et II.21 (b). La figure II.21 (a) présente la cascade d'un filtre RIF et d'un down-sampler, telle que définie par les équations (**II**.30) et (**II**.31) comme l'indique la figure II.20. La structure efficace qui met en œuvre l'équation (**II**.33) est illustrée à la figure II.21 (b). Formellement, la structure de la figure II.21 (b) peut aussi être dérivée de la structure de la figure II.21 (a) en appliquant la première identité de décimation. Dans la mise en œuvre conventionnelle de la figure II.21 (a), le nombre de multiplications par échantillon d'entrée dans le décimateur est égal à la longueur du filtre RIF N. La structure de mise en œuvre efficace de la figure II.21 (b) réduit le nombre de multiplications par échantillon d'entrée à N/M. Cette propriété améliore significativement l'efficacité du filtre RIF à plusieurs vitesses.



(a)



(b)

Figure II.21. Implémentations directes d'un décimateur à facteur de **M** : (a) structure en cascade. b) Structure efficace directe

II.6 .2. Structures directes pour les interpolateurs RIF :

Considérer un interpolateur de facteur **L** et d'un RIF de la réponse impulsionnelle {h[n]} comme indiqué à la (figure II.22) [6].

Le signal à faible débit $\{x[n]\}$ est sur-échantillonné par **L** et ensuite échauffé par le filtre RIF. Notre objectif est d'introduire l'opération d'échantillonnage ascendant dans la structure du filtre pour augmenter la fréquence d'échantillonnage. Nous observons que à l'entrée du filtre le signal $v[m]$ est le résultat de l'opération de sur-échantillonnage effectuée sur le signal d'entrée $x[n]$.

$$v(m)=\begin{cases} x\left(\frac{m}{L}\right), & m=0,\pm L,\pm 2L,\ldots.. \\ 0 & si,non \end{cases} \qquad (\mathbf{II}.34)$$

Nous constatons que tout échantillon d'entrée L^n dans le filtre est valorisé non nul. Dans la structure de filtre RIF conventionnelle, les échantillons d'entrée **L**-1 sur **L** sont multipliés par les coefficients de filtre sans contribuer aux valeurs des échantillons de sortie. Ceci est illustré à la figure II.22 (a) pour le formulaire de transposition de la mise en œuvre directe.

Étant donné qu'il n'est pas nécessaire de multiplier les coefficients de filtre par les échantillons à valeur nulle, nous pouvons effectuer des multiplications à la fréquence d'échantillonnage du signal d'entrée, puis sur-échantillonné par L des signaux multipliés comme le montre la figure II.22 (b). Cette structure d'implémentation réduit le nombre de multiplications par échantillon de sortie de N (longueur du filtre) à N/L.

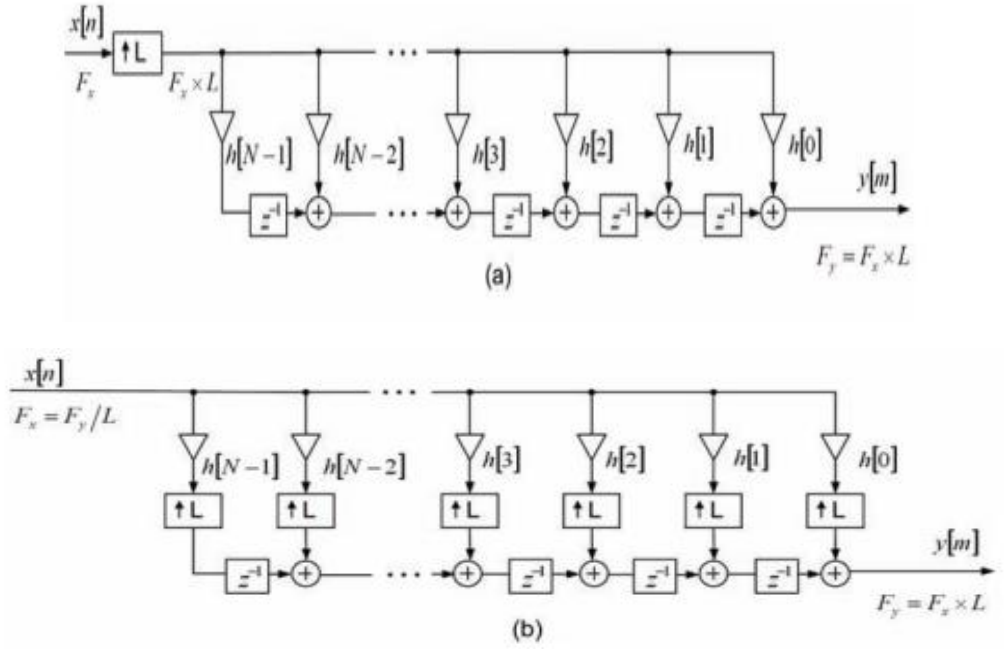


Figure II.22. Implémentations directes d'un interpolateur du facteur **L** : (a) structure en cascade. (b) Structure efficace directe

II.6.3. Structures de réalisations polyphasées pour filtres RIF :

Un filtre RIF d'ordre supérieur peut être réalisé dans une structure parallèle basée sur la décomposition poly-phasique de la fonction de transfert. La fonction de transfert de filtre RIF est décomposée en M fonctions de transfert d'ordre inférieur, appelées composants polyphasés, qui sont ensuite additionnées pour composer la fonction de transfert d'origine. Par souci de simplicité, nous montrons ensuite la décomposition d'un système RIF H(z) en deux composants polyphasés $E_0(z)$ et $E_1(z)$. Exprimons la fonction de transfert de filtre RIF H(z) sous forme développée [6] :

$$H(z)=h(0)+h(1)z^{(-1)}+h(2)z^{(-2)}+h(3)z^{(-3)}+...+h(N-2)z^{-(N-2)}+h(N-1)z^{-(z-1)} \qquad (\mathbf{II}.35)$$

La fonction de transfert ci-dessus peut être exprimée sous la forme d'une somme de deux termes. Nous formons le premier terme à partir des coefficients indexés pairs et le deuxième terme à partir des coefficients indexés impairs. Sans perte de généralité, on peut supposer que N est un nombre impair, et exprimer l'équation (II.35) comme suit :

$$\begin{aligned} H(z)&=h(0)+h(2)z^{-2}+h(4)z^{-4}+...+h(N-3)z^{-(N-3)}+h(N-1)z^{-(N-1)} \\ &+h(1)z^{-1}+h(3)z^{-3}+h(5)z^{-5}+h(N-4)z^{-(N-4)}+h(N-2)z^{-(N-2)} \end{aligned} \qquad (\mathbf{II}.36)$$

Donc :

$$\begin{aligned} H(z)&=h(0)+h(2)z^{-2}+h(4)z^{-4}+...+h(N-3)z^{-(N-3)}+h(N-1)z^{-(N-1)} \\ &+z^{-1}[h(1)+h(3)z^{-2}+h(5)z^{-4}+h(N-4)z^{-(N-4)}+h(N-2)z^{-(N-2)}] \end{aligned} \qquad (\mathbf{II}.37)$$

En utilisant la notation :

$$E_0(z)=h(0)+h(2)z^{-2}+h(4)z^{-4}+...+h(N-3)z^{\frac{-(N-3)}{2}}+h(N-1)z^{\frac{-(N-1)}{2}}) \qquad (\mathbf{II}.38)$$

$$E_1(z)=h(1)+h(3)z^{-2}+h(5)z^{-4}+h(N-4)z^{-(N-4)}+h(N-2)z^{-(N-2)}) \qquad (\mathbf{II}.39)$$

Donc la fonction de transfert (**II**.36) est égale à la somme des composants polyphasés E_1 (z) et E_0 (z) :

$$H(z)=E_0(z^2)+z^{-1}E_1(z^2) \qquad (\mathbf{II}.40)$$

Dans un cas général, une fonction de transfert H(z) de longueur N peut être décomposée en M branches polyphasées $E_0(z)$, $E_1(z)$,..., $E_{M-1}(z)$, donc H(z) peut être exprimée sous la forme suivante :

$$H(z)=\sum_{k=0}^{M-1} z^{(-k)}E_k(z^M) \qquad (\mathbf{II}.41)$$

$$Avec \Rightarrow E_k(z)=\sum_{n=0}^{\frac{N}{M}} h(M*n+k)z^{(-n)}, \Rightarrow 0\le k\le M-1 \qquad (\mathbf{II}.42)$$

Un filtre RIF peut être mis en œuvre en tant que connexion parallèle de composants polyphasés **M (L)** qui sont ajoutés ensemble à la sortie. Les composants polyphasés sont parfois appelés sous-filtres poly-phasiques ou branches polyphasées. Un composant polyphasé est généralement mis en œuvre sous forme directe. La (figure II.23 (a)) montre une représentation cascade d'un filtre RIF implémenté comme connexion parallèle de branches polyphasées M et d'un décimateur de facteur de M. Dans ce cas, les opérations arithmétiques dans les branches polyphasées doivent être effectuées à la fréquence d'échantillonnage en entrée. Au lieu d'effectuer un sous-échantillonnage à la sortie du filtre, on peut déplacer l'opération de sous-échantillonnage dans les dérivations polyphasées avant les additionneurs de sortie. Cette modification offre la possibilité d'appliquer la troisième identité et de parvenir à une mise en œuvre efficace [6].

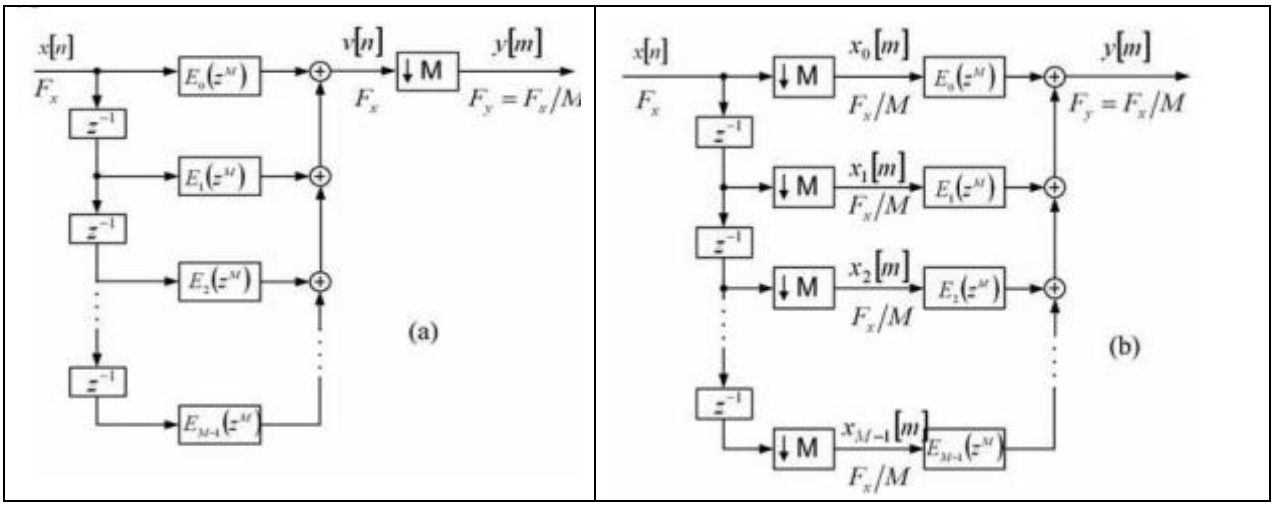


Figure II.23. Réalisation polyphasée du décimateur RIF : (a) Filtre en cascade. b) Décimateur poly-phasique efficace

Les séquences d'entrée dans les branches polyphasées de la (figure II.23 (b)), x_0(m), x_1(m), x_2(m),..., x_{M-1}(m), sont des versions retardées et sous-échantillonnées du signal d'entrée $x(n)$. On peut dire que la séquence particulière x_k(m) est obtenue lorsque le sous échantillonnage par M la séquence x (n) est appliqué, avec le décalage de phase k, k = 0,1,..., M- 1. Ainsi, à partir de la séquence causale x (n)= { $x(0)$, x (1), x (2),..., x (M-1), x (M), x (M+1),...}, il est simple pour extraire les séquences M.

Avec :

$$\begin{aligned} x_0(m) &= x(0); x(M); x(2M), \ \ldots \\ x_1(m) &= x(-1); x(M-1); x(2M-1), \ \ldots \\ x_2(m) &= x(-2); x(M-2); x(2M-2), \ \ldots \\ x_{M-1}(n) &= x(-M+1); x(1); x(M+1); x(2M+1), \ \ldots \end{aligned} \qquad (\mathbf{II}.43)$$

La structure polyphasée parallèle composée de composants polyphasés en **L** peut être utilisée pour fournir une implémentation efficace d'un interpolateur à facteur de **L**. La figure II.26 (a) montre l'interpolateur composé d'un sur-échantillonneur facteur de **L** et d'un filtre RIF réalisé sous la forme polyphasée. On remarque que la réalisation de la figure II.24 (a) est une transposition de la réalisation utilisée pour le décimateur de la figure II.23 (a).

Dans la structure interpolatrice de la (figure II.24 (a)), l'échantillonnage ascendant précède le filtrage, et selon cette structure, le filtrage dans les composants poly-phasiques est effectué à la fréquence d'échantillonnage la plus élevée. En utilisant la troisième Identité d'interpolation, la structure de la (figure II.24 (a)) peut être modifiée en fonction de la structure la plus efficace présentée à la (figure II.24 (b)). Les positions des échantillonneurs et des composants polyphasés sont inter-changées et le filtrage dans les branches polyphasées doit être effectué à la fréquence d'échantillonnage la plus basse. Les signaux filtrés $u0(n), u1(n), u2(n), \ldots, uM-1(n)$ sont sur-échantillonnés par **L** et transmis à la chaîne

d'additionneurs et de retards. Les échantillons doivent passer par des retards au taux d'échantillonnage de sortie et enfin donner le signal de sortie $y(m)$ [6].

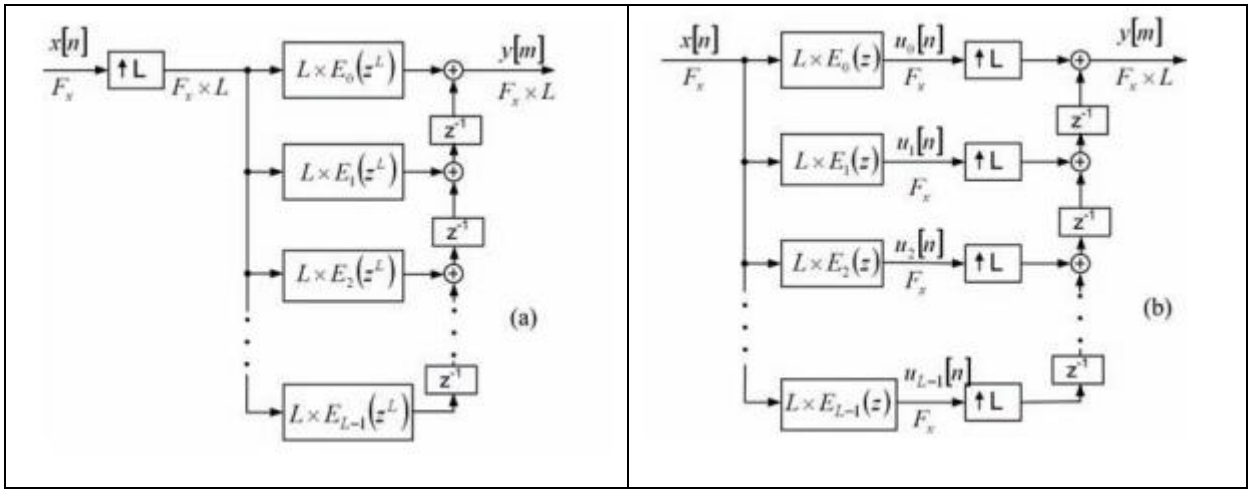


Figure II.23. Réalisation polyphasée d'interpolateur RIF : (a) Filtre en cascade. b) Décimateur poly-phasique efficace.

II.7. Conclusion :

La conception d'un filtre numérique est conditionnée par un choix approprié de la représentation du signal, notamment les méthodes de calcul des coefficients des filtres, ainsi que la structure de réalisation. Chaque élément doit être choisi adéquatement afin d'avoir les performances requises essentiellement pour le filtrage à haute vitesse.

Chapitre III : Processeur de traitement du signal DSP

III.1. Introduction :

Depuis plusieurs décennies les techniques de traitement numériques ont beaucoup évolués. Nous allons nous intéresser aux processeurs de traitements du signal, plus communément désignés par l'acronyme Anglais DSP (Digital Signal Processor). Les DSP ont été initialement développés pour des applications de radars militaires et de télécommunications cryptées dans les années 70 [12]. En 1978, un nouveau DSP a été introduit pour la synthèse de la voix pour des applications très grand public. Il aura fallu attendre 15 ans pour que les DSP deviennent des composants incontournables pour diverses applications notamment dans les domaines télécommunications, santé…; Un DSP est composé essentiellement de processeurs. Il est caractérisé par le fait qu'il intègre un ensemble de fonctions spéciales. Ces fonctions sont destinées à le rendre performant dans le domaine du traitement numérique du signal. Comme un microprocesseur classique, un DSP dispose de la mémoire (RAM, ROM) et des périphériques d'entrées-sorties.

III.2. Présentation d'un DSP :

Un processeur de traitement numérique du signal (Digital Signal Processor) est un composant électronique programmable [12].

DSP de Texas Instruments est un processeur spécialisé pour effectuer en grandes vitesse les opérations divers de traitement de signal tel que le filtrage, la transformation et l'analyse spectrale en temps réel. L'architecture en DSP est optimale pour effectuer des calculs complexes en un cycle d'horloge, mais aussi pour accéder très facilement à un grand nombre d'entrées-sorties (numériques ou analogiques). La fonction principale utilisée dans le DSP est la fonction MAC (Multiply and Accumulate) (voir figure III.1). Les DSP sont utilisés dans la plupart des applications en temps réel. On les trouve dans les modems, les téléphones mobiles, les appareils multimédia, les récepteurs GPS, etc… .

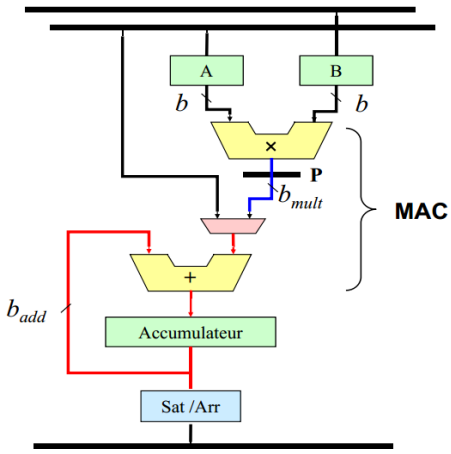


Figure III.1 : Structure MAC de DSP

III.3. Système de traitement numérique du signal à base de DSP :

Le traitement numérique du signal (DSP) implique la manipulation de signaux numériques afin d'en extraire des informations utiles. Bien qu'un nombre croissant de traitements de signaux soient effectués dans le domaine numérique, il reste nécessaire de s'interfacer avec le monde analogique dans lequel nous vivons. Les convertisseurs de données analogique-numérique (A/N) et numérique-analogique (N/A) sont les dispositifs qui rendent cette interface possible. La figure III.2 illustre les principales composantes d'un système DSP, composé de convertisseur A/N, et convertisseur N/A [13].

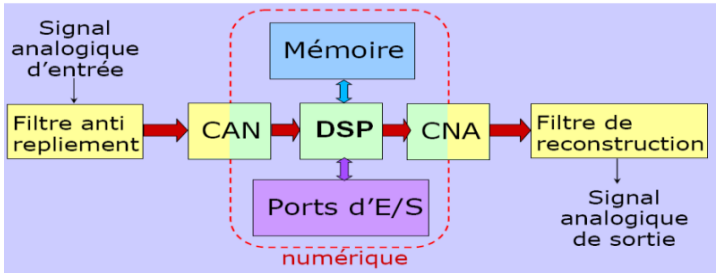


Figure III.2 : Schéma de principe d'un DSP

Il y a de nombreuses raisons pour lesquelles nous souhaitons traiter un signal analogique de manière numérique en le convertissant en un signal numérique, le coup de traitement analogique est élevé, et limité en fréquence d'échantillonnage d'où la limitation en gamme de fréquence. Enfin le traitement numérique amené à l'algorithmique permet d'avoir une grande souplesse pour la réalisation des taches de filtrage, la modularité des DSP permet leurs utilisations pour plusieurs taches à la fois [13]. Remarquons que l'utilisation des DSP n'est pas affectée par les variations de température.

III.4. Les caractéristiques d'un processeur DSP :

Les processeurs DSP partagent certaines caractéristiques communes avec les processeurs classiques. Cependant ils se distinguent par les caractéristiques suivantes [14] :

1. Ils sont optimisés pour faire face à la répétition ou au bouclage d'opérations courantes dans les algorithmes de traitement du signal. Relativement parlant, les jeux d'instructions des DSP sont réduits et optimisés pour les opérations de traitement du signal, telles que la multiplication et l'accumulation à cycle unique.
2. Les DSP permettent un adressage indirect et circulaire. Ce sont des mécanismes d'adressage efficaces pour la mise en œuvre de nombreux algorithmes de traitement des signaux.
3. Les DSP possèdent des périphériques appropriés qui permettent une interface entrée/sortie (E/S) efficace avec d'autres périphériques (HPI, DMA, PLL, RS232…).
4. Accès mémoire multiple par cycle machine. En d'autres termes, ces processeurs ont une bande passante relativement élevée entre leurs unités centrales de traitement (CPU) et la mémoire.

5. Les DSP peuvent être combinés avec d'autres composants dans le même boîtier. Par exemple, un ou plusieurs DSP peuvent être combinés avec un microprocesseur classique et des convertisseurs CAN et CNA. Ce type d'assemblage (circuits intégrés dédiés) permet de réduire les coûts dans des fabrications de grande série. Les fonctions de traitement de signal peuvent également être réalisées à l'aide de FPGA, qui peut incorporer des « cœurs DSP » (en général des MAC).

III.5. Applications d'un processeur DSP [14] :

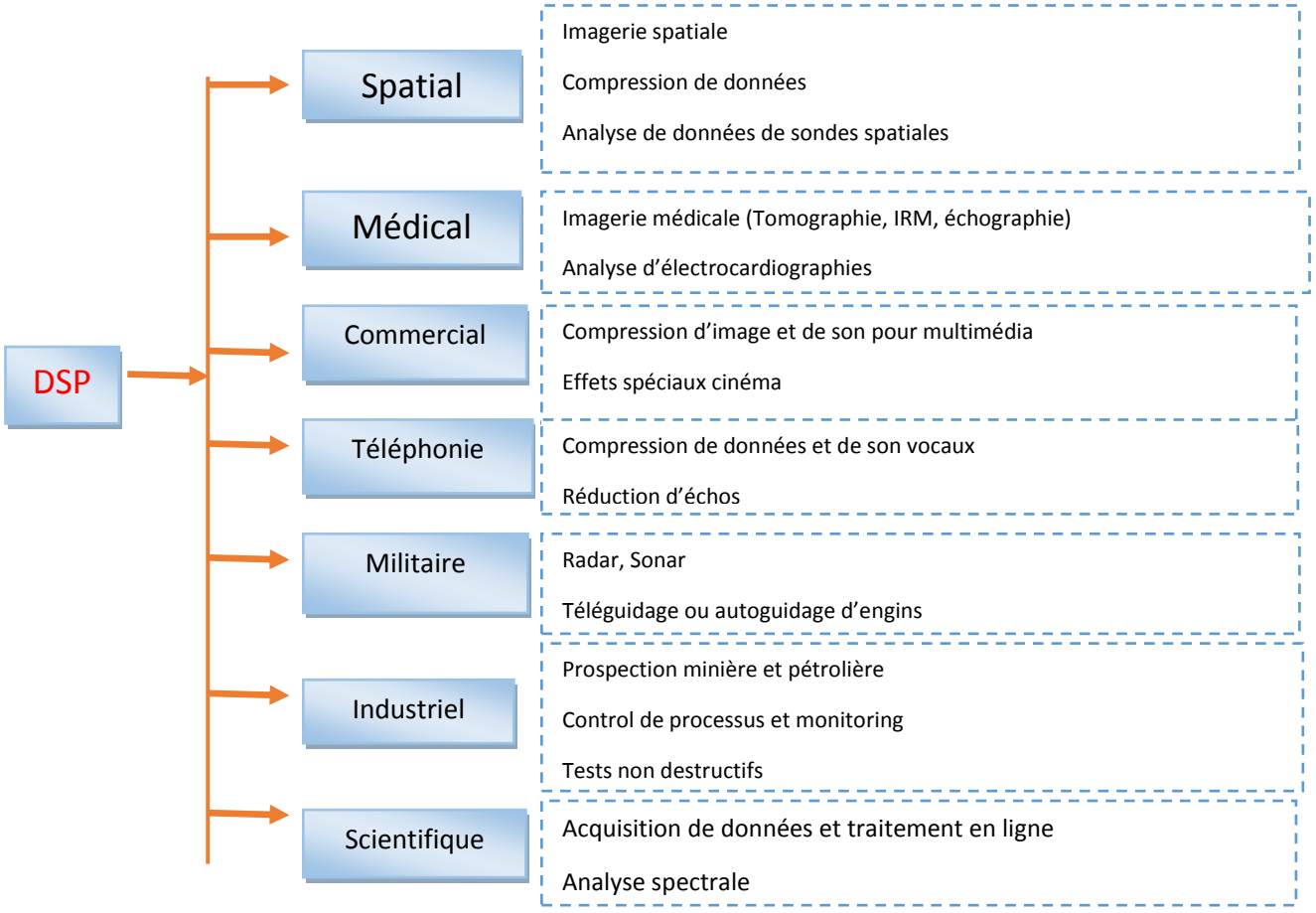


Figure III.3 : Domain d'applications d'un DSP

III.6. Principales distinctions entre un DSP et un microprocesseur classique :

Les principales distinctions entre un DSP et un microprocesseur classique sont :

- Contrairement aux microprocesseurs classiques, un DSP réalise le MAC en un cycle d'horloge grâce à un circuit multiplieur.
- Un DSP réalise plusieurs accès mémoire en un seul cycle, grâce à l'architecture Harvard (à comparer avec l'architecture Von Neumann des microprocesseurs classiques).
- Contrairement aux microprocesseurs classiques, la plupart des DSP n'ont pas besoin de consommer des cycles d'horloge pour tester la valeur du compteur de boucle. Ceci est effectué par un circuit on-chip.

- Les DSP bénéficient des modes d'adressage adaptés aux algorithmes de traitement du signal (circulaire, inversion de bits, …).
- Dans les microprocesseurs classiques, l'utilisation de la mémoire cache et la prédiction des branchements sont effectuées par des circuits logiques, et peuvent changer d'un programme à l'autre. Ceci rend impossible la prédiction du temps d'exécution d'un programme, ce qui est primordial pour une application en temps réel.
- La plupart des DSP sont équipés des interfaces entrée/sortie numériques (GPIO, McBSP, HPI..).

III.7. Mesure de vitesse de calcul pure [13] :

La méthode classique pour évaluer les performances d'un DSP est de se baser sur sa vitesse d'exécution. Encore faut-il trouver une bonne définition de ce qu'est la vitesse d'exécution, ce qui n'est pas forcément simple.

Cette méthode de base consiste donc à compter le nombre d'instructions effectuées par seconde. Un obstacle apparaît alors, car une instruction ne signifie pas forcément la même chose d'une famille de DSP à l'autre. Le tableau 1 résume les principales définitions en usage.

Tableau III.1. : Les unités les plus courantes de mesures des performances des DSP

Acronyme Anglais	Définition
MFLOPS	**M**illion **F**loating-**P**oint **O**perations **P**er **S**econd. Mesure le nombre d'opérations à virgule flottante (multiplications, additions, soustractions, etc.) que le DSP à virgule flottante peut réaliser en une seconde.
MOPS	**M**illion **O**perations **P**er **S**econd. Mesure le nombre total d'opérations que le DSP peut effectuer en une seconde. Par opérations, il faut comprendre non seulement le traitement des données, mais également les accès DMA, les transferts de données, les opérations d'E/S, etc. Cette définition mesure donc les performances globales d'un DSP plutôt que ses seules capacités de calcul.
MIPS	**M**illion **I**nstructions **P**er **S**econd. Mesure le nombre de codes machines (instructions) que le DSP peut effectuer en une seconde. Bien que cette mesure s'applique à tous les types de DSP, le MFLOPS est préféré dans le cas d'un DSP à virgule flottante.
MBPS	**M**éga-**o**ctets **P**er **S**econd. Mesure la largeur de bande d'un bus particulier ou d'un dispositif d'E/S, c'est à dire son taux de transfert.

III.8. Les type des DSP :

Le traitement du signal numérique peut être divisé en deux catégories, à virgule fixe et à virgule flottante. Il s'agit du format utilisé pour stocker et manipuler les nombres à l'intérieur des dispositifs [14].

III.8.1. Les DSP à virgule fixe :

Les données en virgule fixe sont composées d'une partie fractionnaire et d'une partie entière pour lesquelles le nombre de bits alloués reste figé au cours du traitement. L'exposant associe à chaque donnée est implicite et fixe. La figure III.4 représente une donnée en virgule fixe composée d'un bit de signe et de *b-1* bits repartis en *m* bits pour la partie entière et *n* bits pour la partie fractionnaire [2].

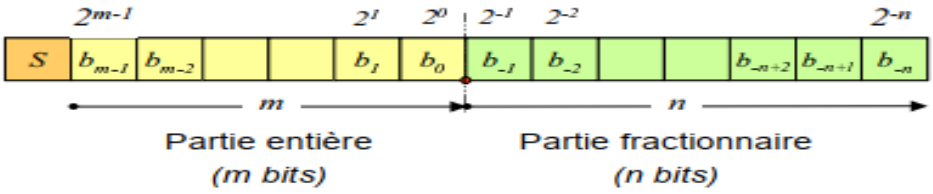


Figure III.4 : Représentation des données en virgule fixe.

Le format d'une donnée en virgule fixe est défini par la longueur de sa partie entière et de sa partie fractionnaire et de la représentation choisie.

a. Codage en virgule fixe complément à deux :

La représentation en code complément a 2 de la donnée *x* en virgule fixe est égale à :

$$x = -2^m S + \sum_{i=-n}^{m-1} b_i 2^i \qquad \text{(III.1)}$$

Le domaine de définition de ce code n'est pas symétrique par rapport à l'origine, il est composé de $\mathbf{2^{b-1}-1}$ valeurs positives et de $\mathbf{2^{b-1}}$ valeurs négatives :

$$\mathbf{D= [-2^m ,2^m-2^{-n}]}$$

Précision du codage Pas de quantification : $q=2^{-n}$

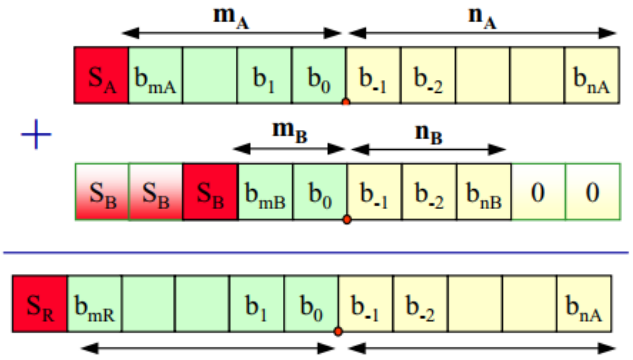
Doivent être identiques pour les deux opérandes. Si cette condition n'est pas respectée il est nécessaire de modifier le format des opérandes afin d'obtenir un format identique (b_c,m_c,n_c). Le format commun garantissant l'absence de perte d'information est le suivant [2] :

$$m_R = max(m_A, m_B)$$
$$n_R = max(n_A, n_B) \qquad \textbf{(III.2)}$$
$$b_R = m_R + n_R + 1$$

Additionnement en virgule fixe :

L'addition de deux opérandes *A* et *B* nécessite qu'elles possèdent un format commun. Le type de représentation, la longueur de la partie entière et la longueur de la partie fractionnaire pour les données ayant un format différent du format commun, il est nécessaire d'étendre le nombre de bits des parties entières et fractionnaires en suivant ces règles :

- partie fractionnaire : les (n_A,n_B) bits supplémentaires sont mis à 0.
- Partie entière : extension du bit de signe 1. Dans le cas du complément à 2 les (m_A,m_B) nouveaux bits prennent la valeur du bit de signe.

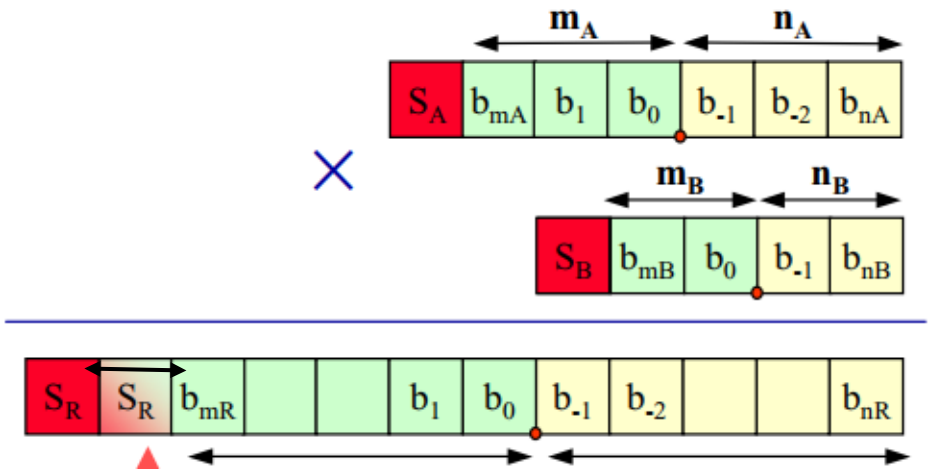


$m_R = \max(m_A, m_B) + 1 \quad n_R = \max(n_A, n_B)$

FigureIII.5 : Exemple de codage avec additionnement.

Multiplication :

Pour une multiplication, les deux opérandes doivent posséder la même représentation mais le nombre de bits réservés pour chaque partie peut être différent. Néanmoins, il est nécessaire avant d'effectuer l'opération, d'étendre le bit de signe. La multiplication de deux nombres en virgule fixe entraine le doublement du bit de signe, celui-ci peut être éliminé automatiquement à l'aide d'un décalage à gauche. Pour un code en complément à 2 nous pouvons considérer que ce bit de signe redondant appartient à la partie entière [2].

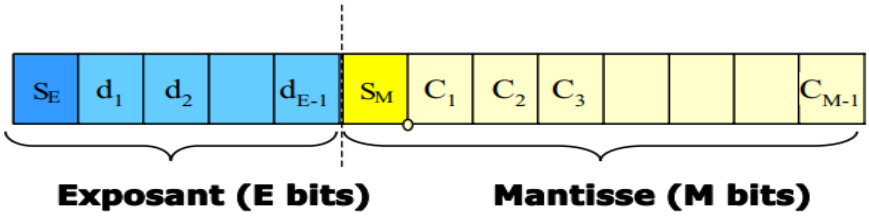


Les 2 bits de signe $M_R=mA+m_B$ $n_R=n_a+n_B$

FigureIII.6. Exemple de codage avec multiplication.

III.8.2. Les DSP à virgule flottante :

L'exposant associé à la donnée est codé au sein de celle-ci. Les données sont composées de deux parties : Exposant, Mantisse [2].
Sa valeur est comprise dans l'intervalle : [-1 ;-1/2] Ụ [1/2 ; 1]



FigureIII.7 : Présentation des données en virgule flottante.

$$x = 2^{u}(-1)^{S_E}\left(\frac{1}{2}+\sum_{i=1}^{M-1} C_i 2^{i-1}\right) \qquad \text{(III.3)}$$

Avec :

$$u = (-1)^{S_E}\sum_{i=1}^{E-1} d_i 2^{i} \qquad \text{(III.4)}$$

Tableau III.2 : Comparaison entre virgule fixe et virgule flottante [15].

Virgule fixe	virgule flottante
- Opérateurs plus simples, processeur plus rapide. - Largeur des données stockées en mémoire : 16 bits. - Efficacité énergétique plus importante, consommation moins importante. - Processeur moins cher (surface du circuit moins importante). - Applications grand public DSP virgule fixe : 95% des vente.	- Largeur des données stockées en mémoire : 32 bits (IEEE 754). - Opérateurs complexes (gestion de la mantisse et de l'exposant). - Temps de développement faible.

III.10. Architecture du processeur de signaux numériques :

Tous les DSP sont constitués de plusieurs modules fondamentaux : un noyau de traitement du signal numérique pour effectuer des opérations mathématiques, une mémoire pour stocker les données et les instructions de programmation, et éventuellement un produit à signaux mixtes pour converser entre les mondes analogique et numérique.

En tant que machine de programme stockée, le processeur doit savoir quoi faire à chaque cycle d'horloge. Typiquement, un DSP récupère une instruction et quelques données de la mémoire, opère sur ceux-ci, et puis retourne les données manipulées au stockage. La façon de procéder n'est pas là même pour tous les processeurs. Deux architectures différentes peuvent être identifiées [14] : (Von Neumann et Harvard). L'application DSP, en plus de la configuration mémoire et périphérique, régit généralement le type d'architecture employée**.**

III.10.1. Harvard :

Lorsque la vitesse d'exécution d'un programme est importante, il est préférable d'utiliser la structure de HARVARD. Cette structure se distingue de l'architecture Von Neumann uniquement par le fait que les mémoires programmes et données sont séparées. L'accès à chacune des deux mémoires se fait via un chemin distinct. Cette organisation permet de transférer une instruction et des données simultanément, ce qui améliore les performances [15]. Ainsi en un seul cycle, le processeur pourra lire l'instruction et la donnée.

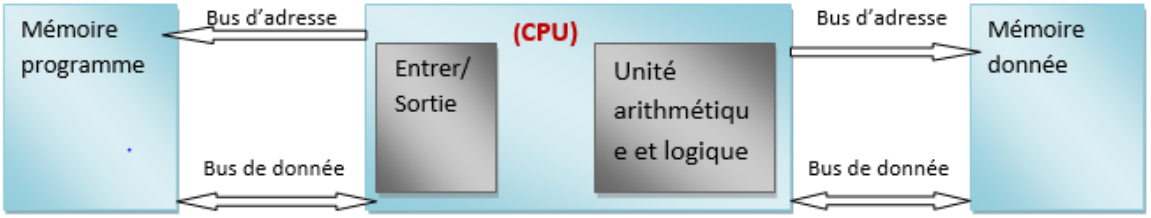


Figure III.8 : Architecture de Harvard

III.11. La famille TMS320 de Texas Instruments :

La famille TMS320 se compose de processeurs DSP 16 bits à virgule fixe, 32 bits à virgule flottante et multiprocesseur 64 bits à puce unique. Ces processeurs ont la flexibilité opérationnelle des contrôleurs à grande vitesse et la capacité numérique des processeurs matriciels. Combinant ces deux qualités, les processeurs TMS320 sont des solutions de rechange peux coûteuses aux processeurs VLSI et aux processeurs à tranches binaires multi-puces fabriqués sur mesure. Les caractéristiques suivantes font de cette famille le choix idéal pour une large gamme d'usinage applications [15] :

- Jeu d'instructions flexible ;
- Flexibilité opérationnelle inhérente ;
- Performances à grande vitesse ;
- Architecture innovante et parallèle ;
- Rapport coût-efficacité.

En 1982, Texas Instruments a lancé le TMS32010, le premier DSP à virgule fixe de la famille TMS320. Le TMS32010 est devenu le modèle pour les générations futures de TMS320.

Aujourd'hui, la famille TMS320 (voir figure III-9) se compose de ces générations : les générations de points fixes C1x, C2x, C20x, C24x, C5x, C5x, C54x et C62x 16 bits ; les générations C4x, C3x et C67x 32 bits à virgule flottante ; et les générations de multiprocesseurs C8x 64 bits. Ces générations sont toutes complétées par des produits à signaux mixtes tels que les convertisseurs de données. Chaque génération d'appareils TMS320 a une structure similaire, un processeur combiné à une variété de mémoires sur puce et de configurations périphériques. De nouvelles combinaisons de mémoire à puce et d'options périphériques sont utilisées pour créer des dispositifs spin-off qui répondent à un large éventail de besoins sur le marché mondial de l'électronique. Lorsque la mémoire et les périphériques sont intégrés dans un processeur, le coût global du système est considérablement réduit et l'espace de carte est économisé [16].

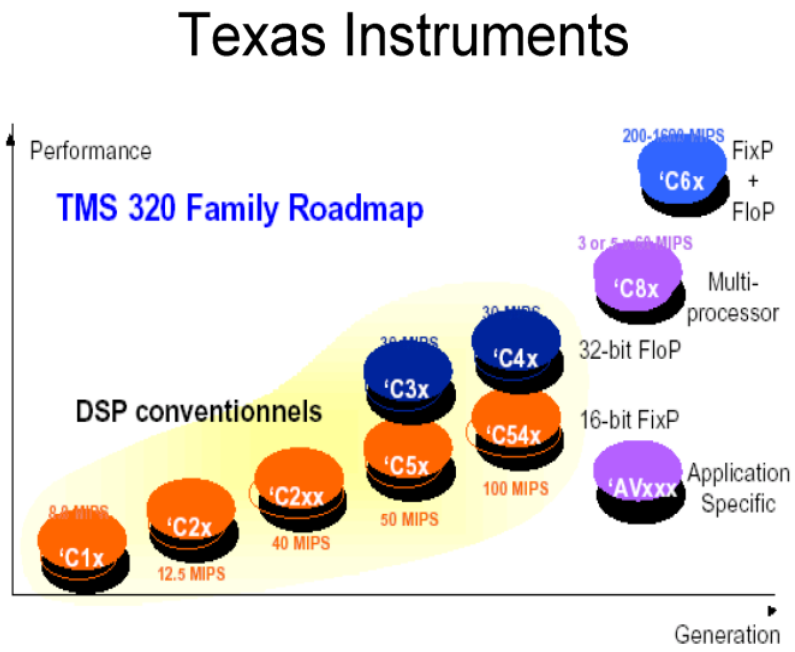


Figure III.9 : L'évaluation des DSP de la famille TMS320

III.11.1. La gamme DSP TMS320VC5416 de Texas Instruments :

a. Définition :

Le processeur DSP à point fixe TMS320VC5416 est basé sur une architecture Harvard modifiée de pointe qui possède un bus de mémoire de programme et trois bus de mémoire de données. Ce processeur fournit une unité de logique arithmétique (ALU) avec un haut degré de parallélisme, une logique matérielle spécifique à l'application, une mémoire sur puce et des périphériques supplémentaires sur puce. La flexibilité et la rapidité opérationnelles de ce DSP reposent sur un ensemble d'instructions hautement spécialisées [16].

Des espaces de programme et de données séparés permettent l'accès simultané aux instructions et aux données du programme, ce qui permet un degré élevé de parallélisme. Deux opérations de lecture et une opération d'écriture peuvent être exécutées en un seul cycle. Les instructions avec stockage parallèle et les instructions spécifiques à l'application peuvent pleinement utiliser cette architecture. En outre, les données peuvent être transférées entre les espaces de données et de programme. Ce parallélisme prend en charge un ensemble puissant d'opérations arithmétiques, logiques et de manipulation de bits qui peuvent toutes être exécutées en un seul cycle machine. L'appareil comprend également des mécanismes de contrôle pour gérer les interruptions, les opérations répétées et les appels de fonction.

Le schéma fonctionnel comprend les blocs principaux et la structure du bus dans les appareils 54x.

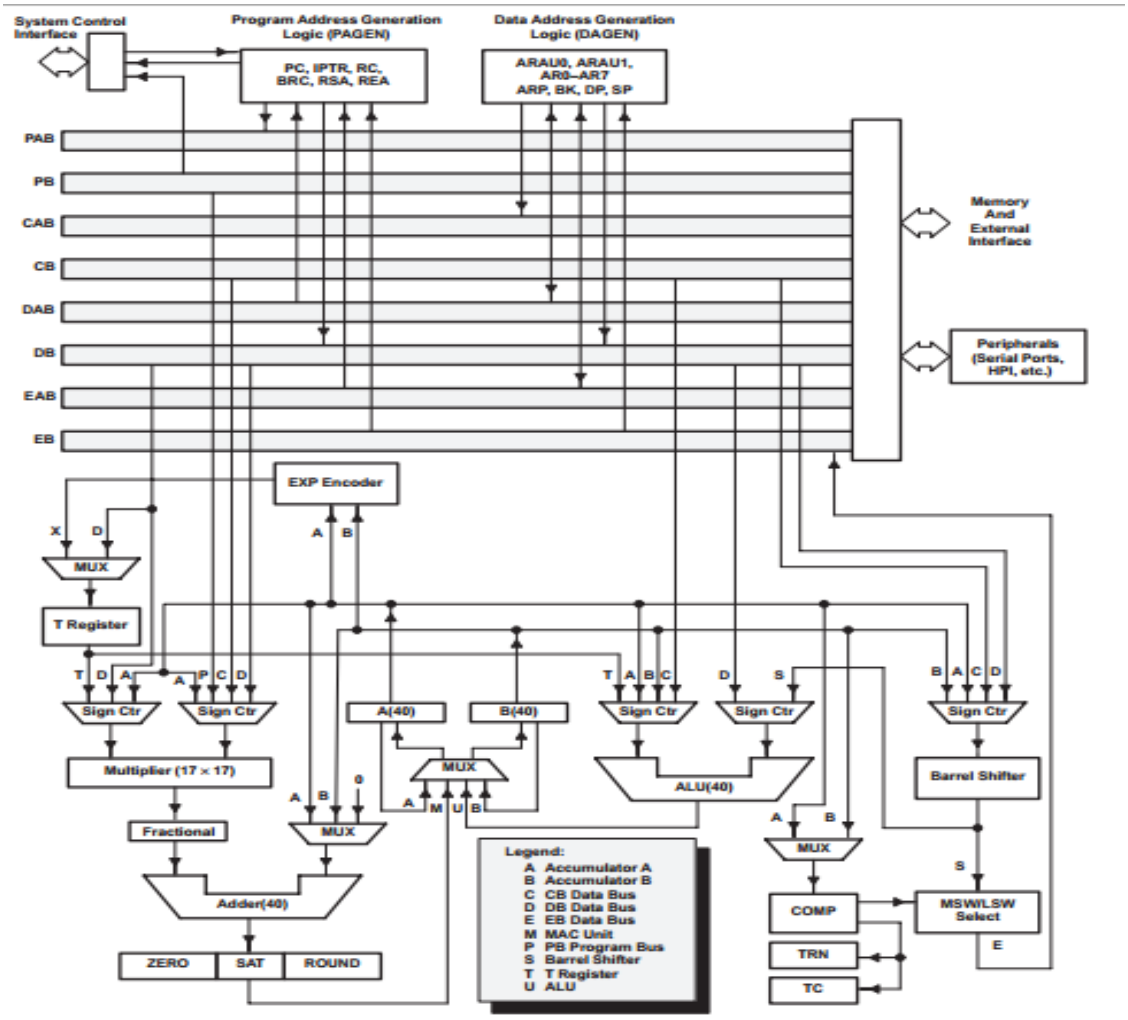


Figure III.10 : Schéma fonctionnel du matériel interne 54x

L'architecture des périphériques 54x est construite autour de huit bus 16 bits majeurs : Un bus de lecture de programme (PB), qui porte le code d'instruction et les opérandes immédiates de la mémoire programme. Deux bus de lecture de données (CB, DB) et un bus d'écriture de données (EB), qui s'interconnectent à différents éléments, tels que CPU, les mémoires programmes, périphériques (GPIO, PLL, McBSP) et mémoire de données. Les bus CB et DB portent les opérandes lus dans la mémoire de données. Le bus EB transporte les données à écrire en mémoire. Il y'a quatre bus d'adresses (PAB, CAB, DAB et EAB), qui portent les adresses nécessaires à l'instruction d'exécution [17].

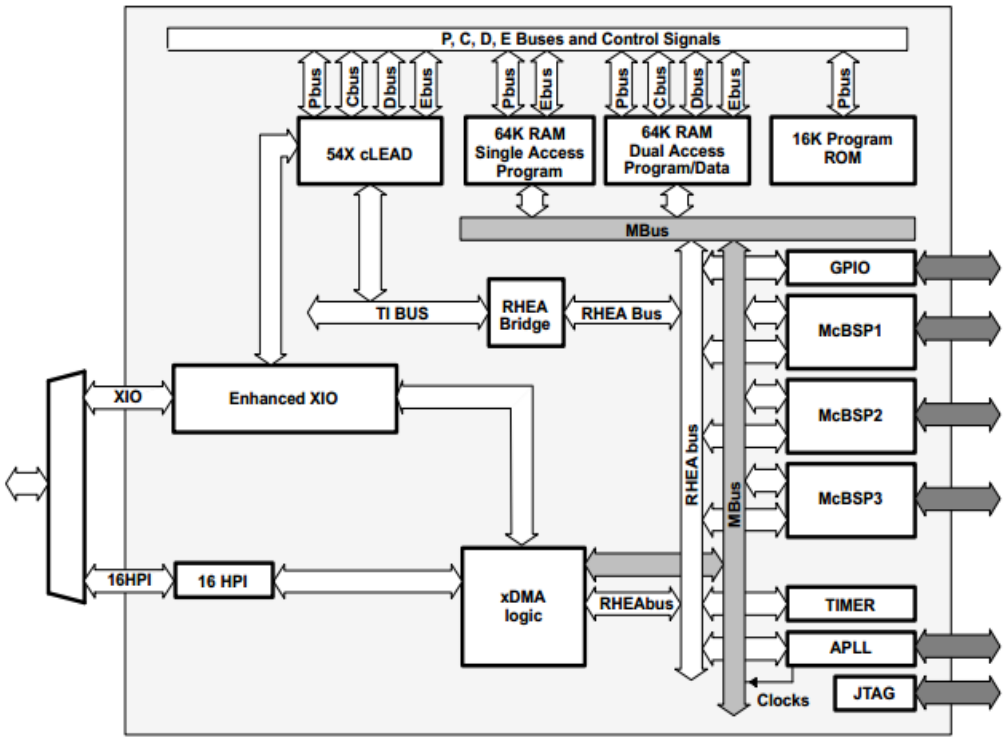


Figure III.11 : Schéma fonctionnel en bloc TMS320VC5416.

b. Mémoire :

Le dispositif fournit des mémoires ROM et RAM sur puce pour faciliter la performance et l'intégration du système [18].

b.1. Mémoire de données :

L'espace mémoire de données peut contenir jusqu'à 64K de mots de 16 bits. L'appareil accède automatiquement à la mémoire RAM lors de l'adressage à l'intérieur de ses limites. Lorsqu'une adresse est générée en dehors des limites de la RAM, l'appareil génère automatiquement un accès externe [18].

b.2. ROM sur puce avec chargeur de Bootloader [16] :

Le périphérique dispose d'une ROM masquable sur puce de 16k mots × 16 bits pouvant être mappée dans le programme espace mémoire. La disposition standard de la mémoire morte sur puce est indiquée dans le tableau **III.3**.

Tableau III.3. La disposition standard de la mémoire morte.

PLAGE D'ADRESSE	DESCRIPTION
C000h-D4FFh	Tables ROM pour le codec vocal GSM EFR
D500h-F7FFh	Réservé
F800h-FBFFh	Bootloader
FC00h-FCFFh	Table d'expansion de la loi
FD00h-FDFFh	A- Table d'expansion de la loi
FE00h-FEFFh	Table de consultation sinusoïdale
FF00h-FF7Fh	Réservé(1)
FF80h-FFFFh	Table des vecteurs d'interruption

b.3. Mémoire RAM :

Le dispositif contient 64 K mots × 16 bits de RAM à double accès sur puce (DARAM) et 64 K mots × 16 bits de RAM à accès unique sur puce (SARAM). Le DARAM est composé de huit blocs de 8K mots chacun. Chaque bloc du DARAM peut supporter deux lectures dans un cycle, ou une lecture et une écriture dans un cycle. Quatre blocs de DARAM sont situés dans la plage d'adresses 0080h-7FFFh dans l'espace de données, et peuvent être mappés dans l'espace programme/données en réglant le bit OVLY sur un. Les quatre autres blocs de DARAM sont situés dans la plage d'adresses 18000h-1FFFFh dans l'espace programme. Le DARAM situé dans la plage d'adresses 18000h-1FFFFh dans l'espace programme peut être mappé dans l'espace de données en réglant le bit DROM sur un. Le SARAM est composé de huit blocs de 8K mots chacun. Chacun de ces huit blocs est une mémoire à accès unique. Par exemple, un mot d'instruction peut être extrait d'un bloc SARAM dans le même cycle qu'un mot de données écrit dans un autre bloc SARAM. Le SARAM est situé dans la plage d'adresses 28000h-2FFFFh, et 38000h-3FFFFh dans l'espace programme [18].

b.4. Carte mémoire

La carte de mémoire de programme et de données est illustrée à la figure III.12. Plages d'adresses pour DARAM sur puce sont comme suit :

DARAM0 : 0080h-1FFFh

DARAM1 : 2000h-3FFFh -

DARAM2 : 4000h-5FFFh

DARAM3 : 6000h-7FFFh

DARAM4 : 8000h-9FFFh

DARAM5 : A000h-BFFFh

DARAM6 : C000h-DFFFh

DARAM7 : E000h-FFFFh

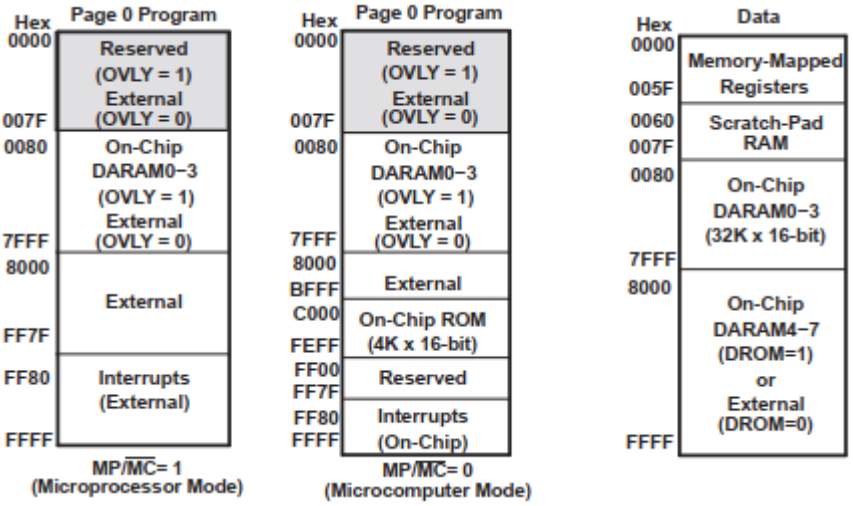


Figure III.12. Carte de mémoire de programme et de données

La carte mémoire étendue du programme est illustrée à la figure III.13. Les plages d'adresses pour DARAM sur puce des données sauvegardées sont [18] :

DARAM4 : 018000h-019FFFh

DARAM5 : 01A000h-01BFFFh

DARAM6 : 01C000h-01DFFFh

DARAM7 : 01E000h-01FFFFh

Les plages d'adresses pour le SARAM sur puce dans la mémoire programme sont :

SARAM0 : 028000h-029FFFh

SARAM1 : 02A000h-02BFFh

SARAM2 : 02C000h-02DFFh

SARAM3 : 02E000h-02FFFFh

SARAM4 : 038000h-039FFFh

SARAM5 : 03A000h-03BFFFh

SARAM6 : 03C000h-03DFFFh

SARAM7 : 03E000h-03FFFFh

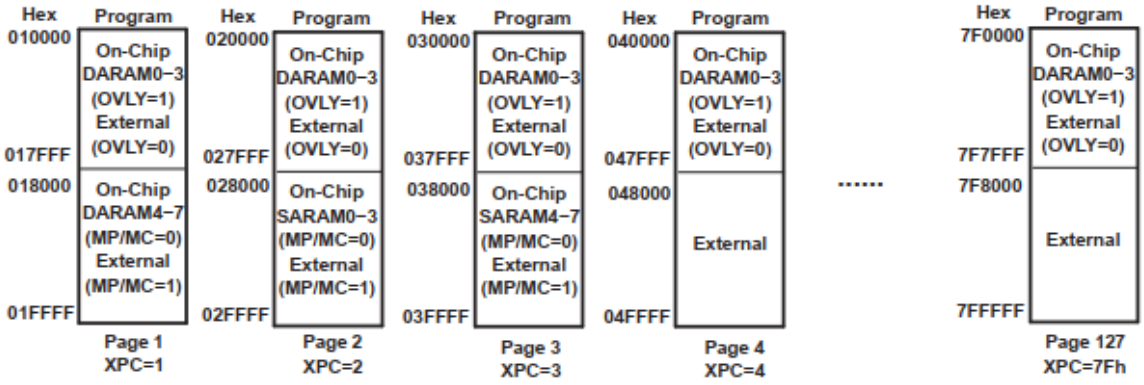


Figure III.13. Carte de la mémoire programme étendue

b. Périphériques sur TMS320VC5416 :

L'appareil possède les périphériques suivants [18] :

- Générateur d'état d'attente programmable par logiciel.
- Commutation de banque programmable
- Une interface hôte-port (HPI8/16)
- Trois ports série tamponnés multicanaux (McBSPs)
- Une minuterie matérielle
- Un générateur d'horloge avec une boucle à verrouillage de phase multiple (PLL).
- Interface parallèle externe améliorée (XIO2) un cntrôleur (DMA)

c. Quelques caractéristiques du TMS320VC5416 [18] :

- Architecture multi-bus avancé avec trois bus de mémoire de données 16 bits séparés et un seul bus mémoire de programme.
- Unité arithmétique et logique de 40 bits (ALU) comprenant un sélecteur de vitesses de 40 bits et deux accumulateurs indépendants de 40 bits.
- Multiplicateur parallèle de 17 × 17 bits couplé à un Additionneur dédié de 40 bits pour une opération multiplicateur/accumulateur à cycle unique (MAC) sans pipeline.
- Mode d'adressage étendu pour un espace de programmation externe adressable de 8M × 16 bits maximum.
- 6.25-ns est le temps d'exécution des instructions en virgule fixe à cycle unique (160 MIPS).
- Tension d'alimentation du cœur 1,6 V (160 MIPS)
- Tension d'alimentation du cœur 1,5 V (120 MIPS)

III .12. Conclusion

Dans ce chapitre, on a présenté le processeur de signal numérique. Le DSP « Digital Signal Processor », est un processeur optimisé pour exécuter des applications de traitement numérique du signal (filtrage, extraction de signaux, etc.) le plus rapidement possible. D'un point de vue économique, on peut être assuré d'une croissance importante du marché des DSP eu égard les équipements dans lesquels ils sont présents, (les téléphones mobiles, la télévision haute définition (TVHD), etc.).

Chapitre IV : Résultats d’implémentation des filtres

IV. 1. Introduction :

Avant d'entamer l'implémentation du filtre RIF sur le DSP, nous avons proposé quelques programmes sous MATLAB et ceci dans le but de bien comprendre les étapes citées dans le chapitre précédent. En effet, le premier exemple est un programme d'analyse d'un filtre numérique RIF passe-bas de second ordre. Le deuxième programme est dédié à l'étude de la synthèse d'un filtre RIF par la méthode de fenêtrage de Hamming, on a rajouté à cela deux programmes consacrés à l'interpolation et la décimation.

La deuxième partie de ce chapitre est consacrée à la réalisation d'un filtre numérique multi-cadence sous MATLAB/SIMULINK. Toutes les étapes et les résultats sont aussi bien définis.

IV.2. Réalisation des filtres numérique :

Ce travail est destiné à synthétiser des filtres RIF et RII sous MATLAB. Un filtre numérique est un élément qui effectue un filtrage à l'aide d'une succession d'opérations mathématiques sur un signal discret. Il y a deux grandes familles de filtres numériques : RII et RIF. On synthétise le filtre RIF par la méthode des fenêtres et le résultat est donné sur MATLAB.

IV.3. Première partie : Réalisation des filtres à réponse impulsionnelle finie (RIF) sous MATLAB.

IV.3. 1. Analyse d'un filtre numérique RIF passe-bas du second ordre :

Un filtre RIF passe-bas d'ordre 2 de bande de transition minimale, ayant pour limites de la bande passante : f_p =1 kHz et f_s = 3 kHz. La fréquence d'échantillonnage fe=10 kHz.

On présente la fonction de transfert en amplitude et en phase, la réponse impulsionnelle et les zéros du filtre.

```
close all;
clear all;
L = 256;
N = 32; n=0:N-1;
delta = [1; zeros(N-1,1)]; %Impulsion
step = ones(N,1); %Echelon unité; %Impulsion
% 1.2 Analyse d'un filtre numérique RIF passe-bas du second ordre
b11 = [0 -0.0309396 -0.0390182 0.0766059 0.288307 0.4 0.288307 0.0766059 -
0.0390182 -0.0309396 0];
[h11,w] = freqz(b11,1,L);
figure; plot(w,20*log10(abs(h11)),'g'); title('Log Magnitude Filtre RIF 2nd ordre');
axis([0 pi -60 2]); hold on;
Fe = 10000;%fréquence d'échantillonnage
Fp = 1000; NFp = round(L*Fp/(Fe/2)); %Bp inferieure
Fs = 3000; NFs = round(L*Fs/(Fe/2)); %Bp supérieure
Delta1 = 1; Delta2 = -20;
gabl = [-Delta1*ones(NFp,1); -5000*ones(L-NFp,1)];%gabarit du filtre
```

```
gabh = [Delta1*ones(NFs,1); Delta2*ones(L-NFs,1)];%gabarit du filtre
plot(w,gabh,'r');plot(w,gabl,'r'); %dessin des gabarits des filtres
figure; plot(w,unwrap(angle(h11)),'g'); title('Phase Filtre RIF 2nd ordre');
grid;
h11 = filter(b11, 1, delta);
figure;  stem(n,h11,'g'); %représentation des échantillons
figure; zplane(b11,1); %représentation des pôles et des zéros
```

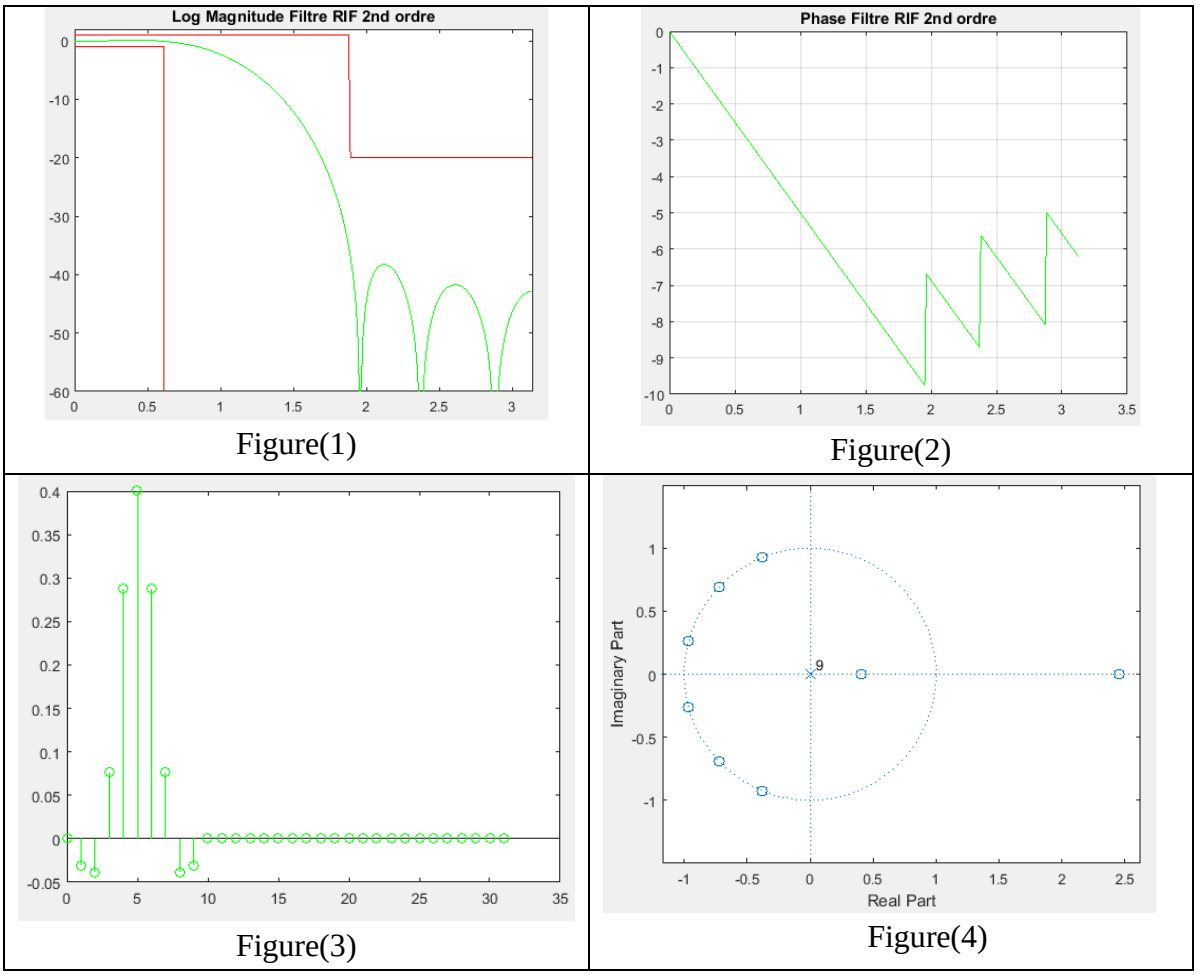


Figure IV.1. Résultat d'Analyse d'un filtre numérique RIF passe-bas de second ordre

IV.3.2. Synthèse d'un filtre RIF :

En utilisant un fenêtrage de Hamming, avec une fréquence d'échantillonnage fe=8 kHz et fc=1kHz.

```
% 2.2 Synthèse d'un filtre RIF
 %Réponse impulsionnelle théorique : h(n) = Wc/pi sinc(nWc/pi)
Fe = 8000; Fc = 1000; Wc = 2*pi*Fc/Fe;
% Méthode du fenêtrage
N = 33; % Longueur du filtre RIF, doit être impaire pour la suite
```

```
alpha = (N-1)/2; n=0:N-1;
b = 4; %RI du filtre idéal
bh = b.*hamming(N)';
[h,w] = freqz(bh,1,L); figure; plot(w,20*log10(abs(h)),'b');
title('Log Magnitude Filtre RIF fenetrage'); axis([0 pi -60 2]); grid
[hh,w] = freqz(bh,1,L);
hold on; plot(w,20*log10(abs(hh)),'g');
figure; stem(n,bh,'b'); hold on; stem(n,bh,'g'); grid
```

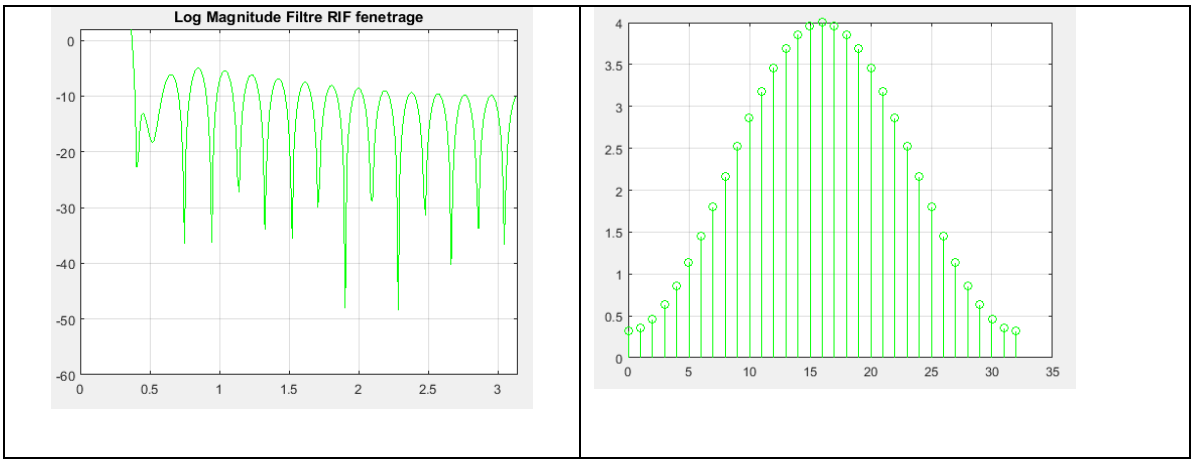


Figure IV.2. Résultat de Synthèse d'un filtre RIF par la méthode de fenêtrage de Hamming.

IV.3.3. Décimation et interpolation :

a. <u>Décimation :</u>

Le programme suivant fait la décimation de 3 fonctions, et les résultats de l'opération apparaissent dans la figure IV.3.

```
% 3.1 Décimation
N = 192;
n = 0:1:N-1;
x = sin(2*pi*0.02*n)
figure; subplot(4,1,1); stem(x); grid
subplot(4,1,2); plot(x)
y1 = x(1:2:N); subplot(4,1,2); stem(y1); grid
y2 = x(1:4:N); subplot(4,1,3); stem(y2); grid
y3 = x(1:6:N); subplot(4,1,4); stem(y3); grid
y1 = decimate(x,2); subplot(4,1,2); stem(y1)
y2 = decimate(x,4); subplot(4,1,3); stem(y2)
y3 = decimate(x,6); subplot(4,1,4); stem(y3)
X = fft(x,300);
Y1 = fft(y1,300);
Y2 = fft(y2,300);
Y3 = fft(y3,300);
figure; subplot(4,1,1); plot(abs(X)); grid
```

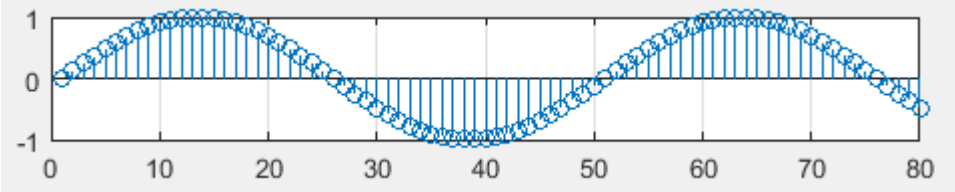
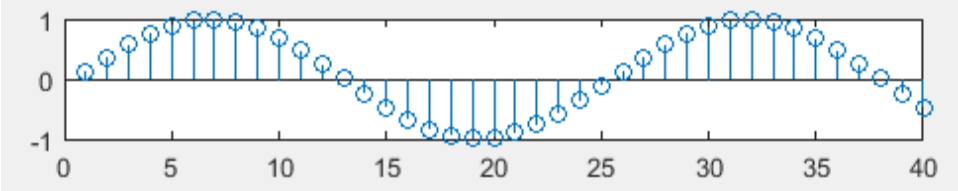
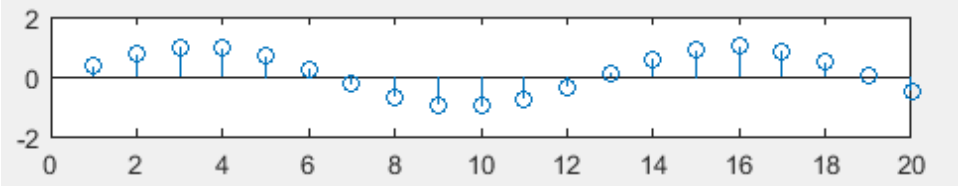
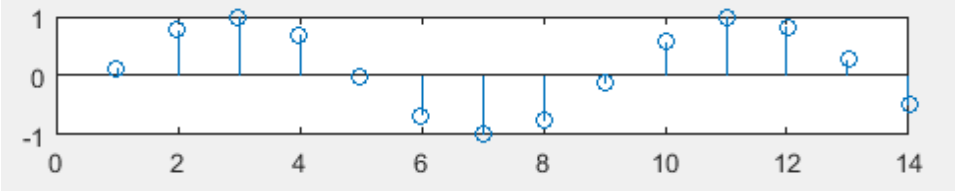
X(t)	
$Y_1(t)$ (facture de décimation =2)	
$Y_2(t)$ (facture de décimation =4)	
$Y_3(t)$ (facture de décimation =6)	

Figure IV.3. Résultat de la décimation

b. Interpolation :

On a étudié l'opération d'interpolation pour le $Y_2(t)$ avec un facteur d'interpolation L=4.

```
% 6.2 Interpolation
L=4;
x = y2; %sinus décimé par 4
y = zeros(N, 1);
y(1:L:N) = x;
figure; stem(y); grid
```

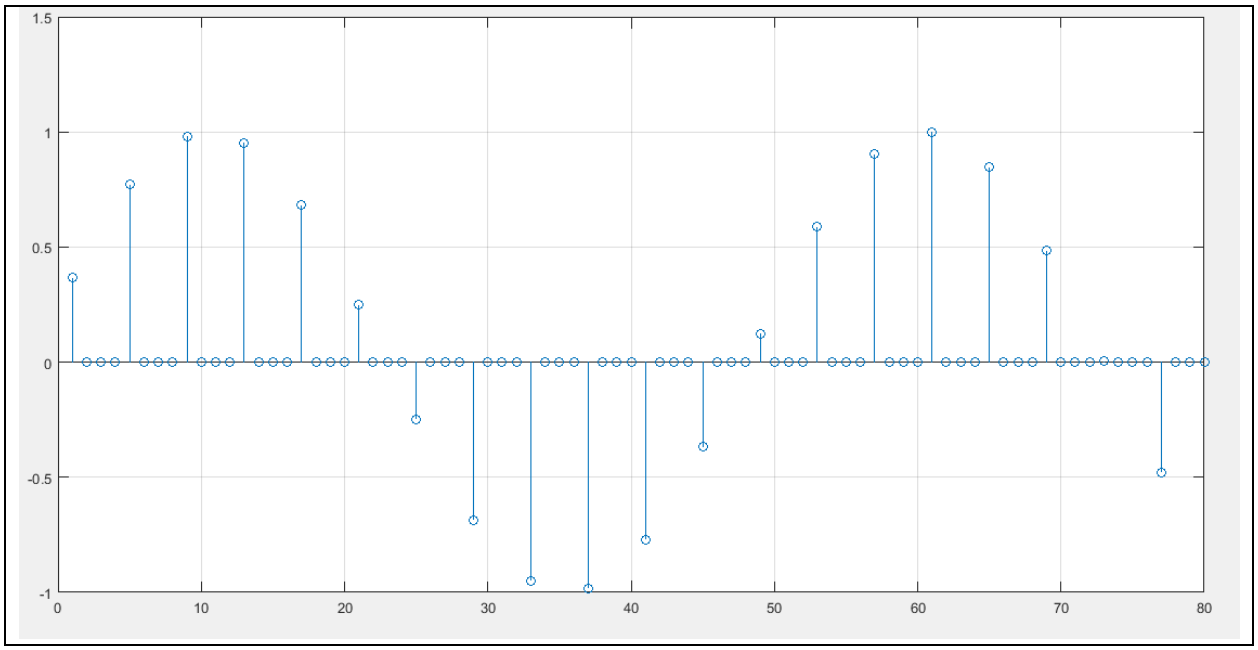


Figure IV.4. Résultat d'interpolation de $Y_2(t)$

IV.4. Deuxième partie : réalisation d'un filtre numérique multi-cadence à l'aide d'une carte DSP TMS320VC5416 sous MATLAB.

Il est possible d'utiliser MATLAB pour communiquer avec les cartes DSP et leurs périphériques en utilisant MATLAB Support Package de Texas instruments. Ainsi on peut programmer ces cartes pour exécuter nos algorithmes à l'aide de Simulink Support Package de DSP Hardware. Le package de support génère du code à partir de notre modèle Simulink en un clic d'un bouton qui s'exécute ensuite sur la carte DSP.

Et pour effectuer ses opérations on passe par plusieurs étapes :

1. Pour obtenir le support package, premièrement on ouvre MATLAB et on clique sur le bouton Add-Ons qui se situe en haut à droite et on choisit « Get hardware Support Packages ».

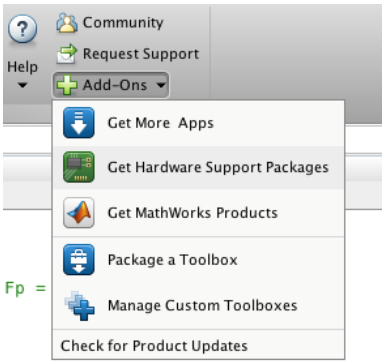
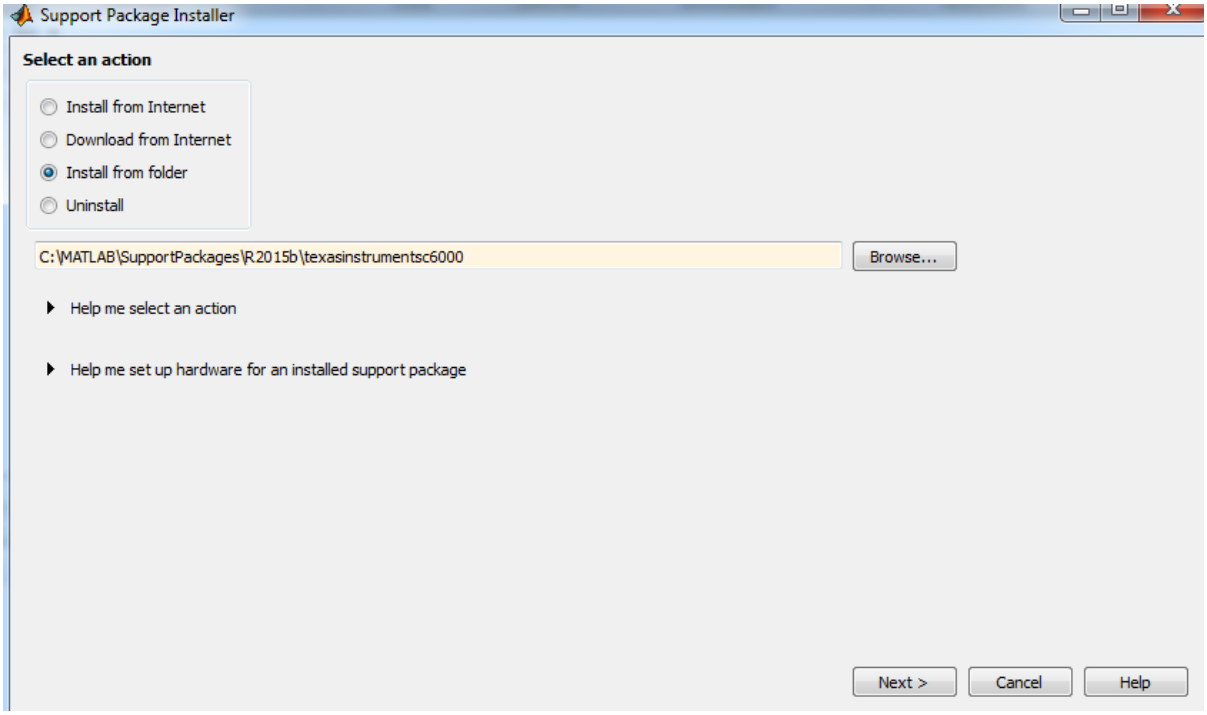


Figure IV.5. Première étape de l'installation de support package

2. Pour installer le support package de **TI**, soit on installe directement à partir d'internet en choisissant « Install From Internet » et une fois le téléchargement terminé il faudra l'installer directement, soit on l'installe à partir d'un dossier déjà existant en choisissant « Install From Folder ».

Dans notre cas, on utilise l'option « Install From Folder » et puis on clique « next » après avoir choisi le dossier dans lequel se trouve notre fichier d'installation.



FigureIV.6. Deuxième étape de l'installation de support package

En cliquant sur <<Next>>, la fenêtre qui suit apparait :

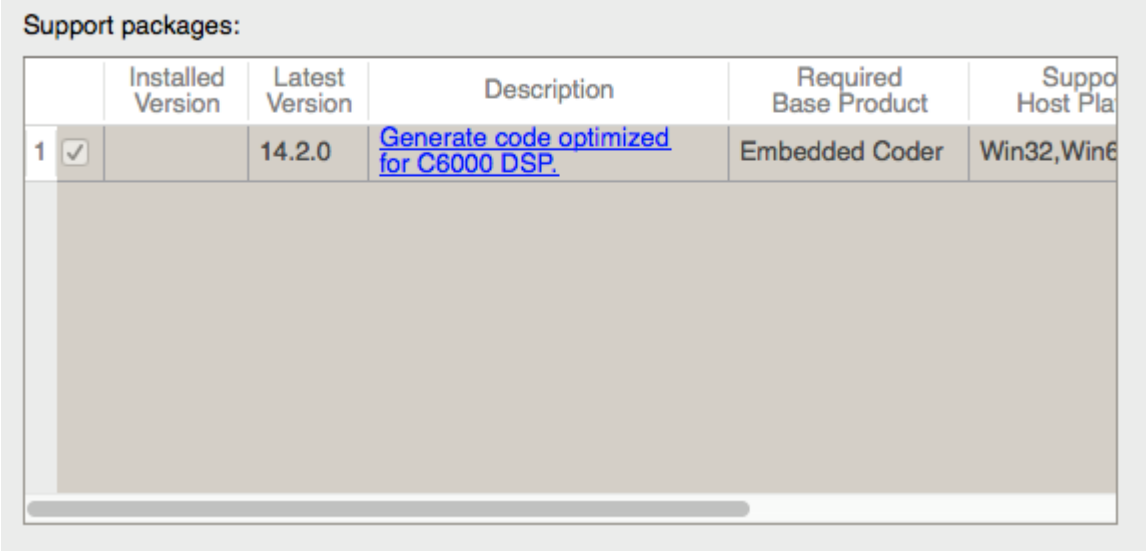


Figure IV.7. Troisième étape de l'installation de support package

On clique sur <<Next>> une seconde fois pour passer aux étapes suivantes :

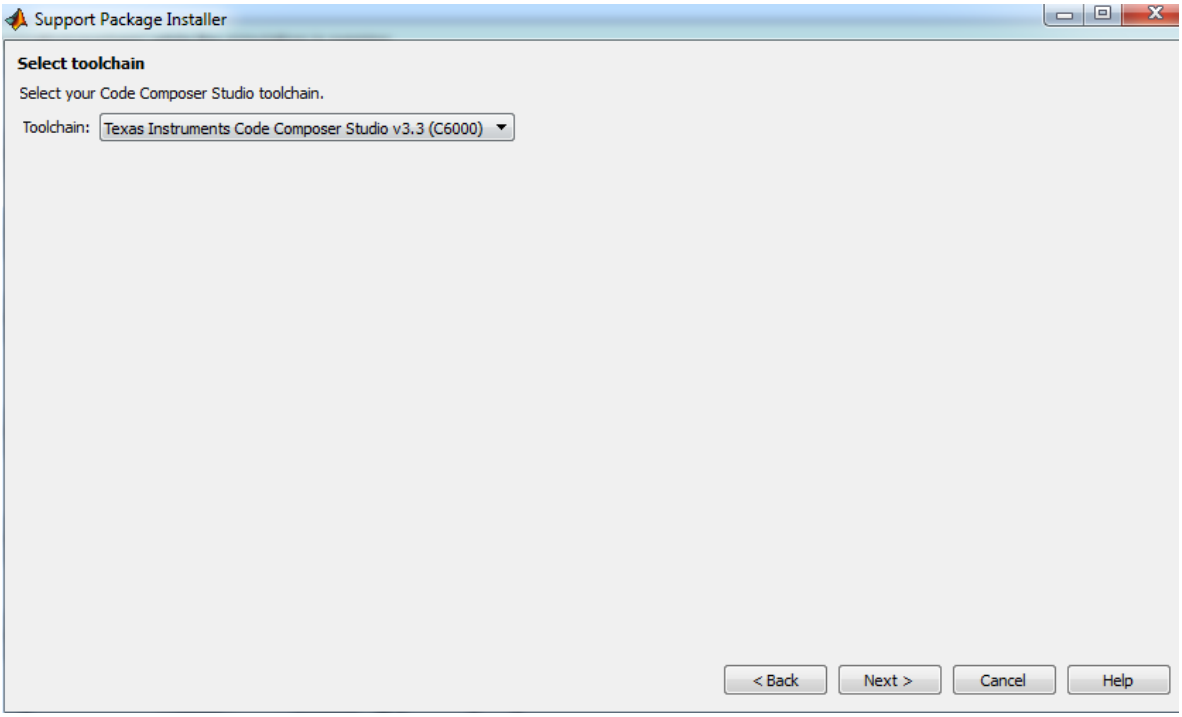


Figure IV.8. Quatrième étape de l'installation de support package

Dans l'étape qui suit, on nous demande d'installer CCS pour poursuivre l'installation.

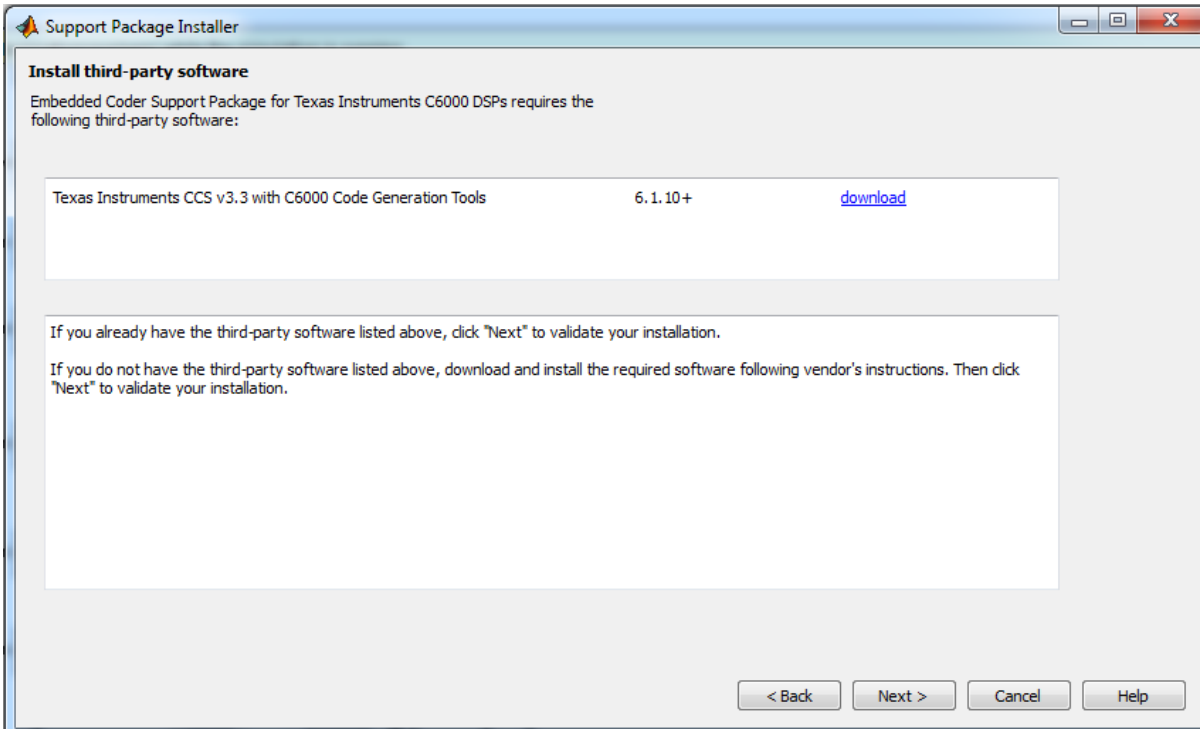


Figure IV.9. Cinquième étape de l'installation du support package.

L'installation du CCS est nécessaire pour terminer l'installation du package de **TI**.

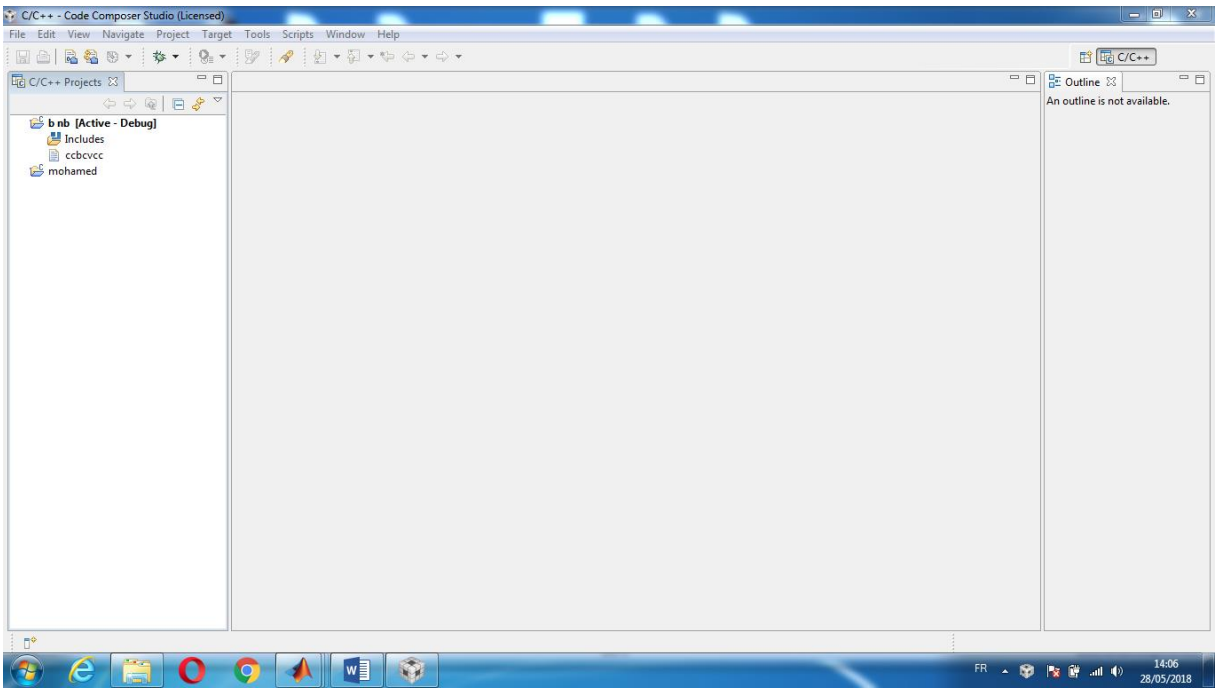


Figure IV.10. L'interface de CCS

IV.4.1. Réalisation d'un filtre RIF :

Afin de réaliser un filtre RIF, on a besoin de trois blocs essentiels qui sont définis par un gain, un retard et une sommation (voir Figure ci-dessous).

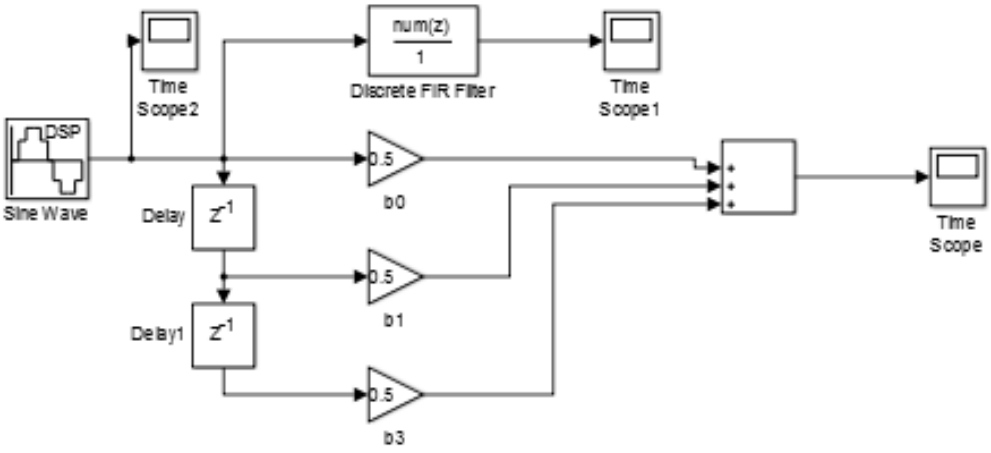


Figure IV.11. Exemple de filtre RIF

Le signal à l'entrée de notre simulation donne les résultats suivants :

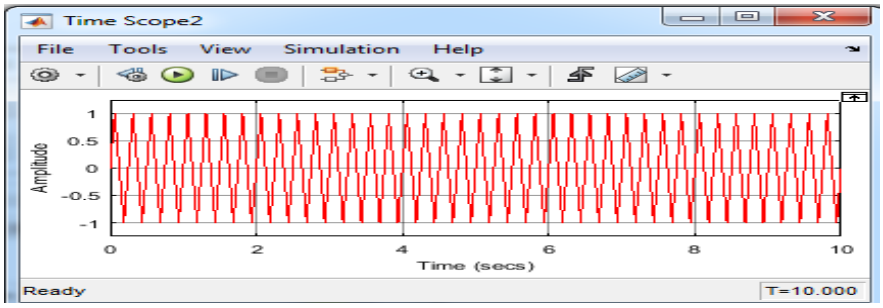


Figure IV.12. Signal d'entrée au filtre RIF

D'autre part, on observe le signal à la sortie du filtre RIF, que ce soit le filtre qu'on trouve à la bibliothèque sur Simulink ou bien le filtre qu'on a conçu nous-mêmes. Les résultats obtenus sont identiques.

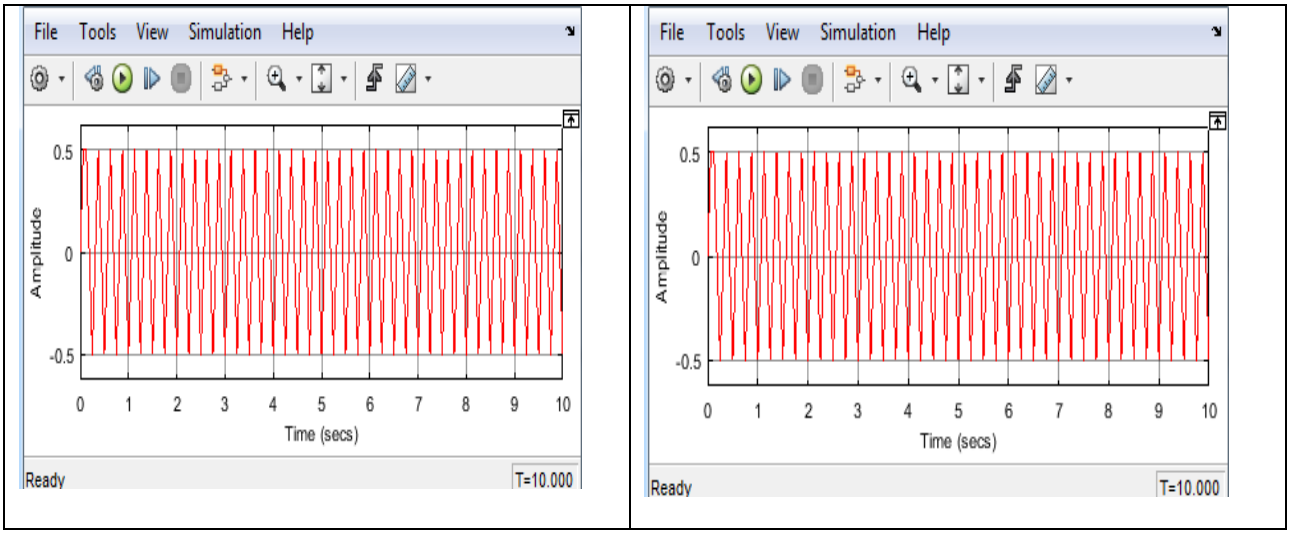


Figure IV.13. Résultat d'exemple RIF

On ajoute au filtre RIF déjà réalisé précédemment des blocs Downsimplink (lien descendant) pour obtenir un filtre RIF de décimation.

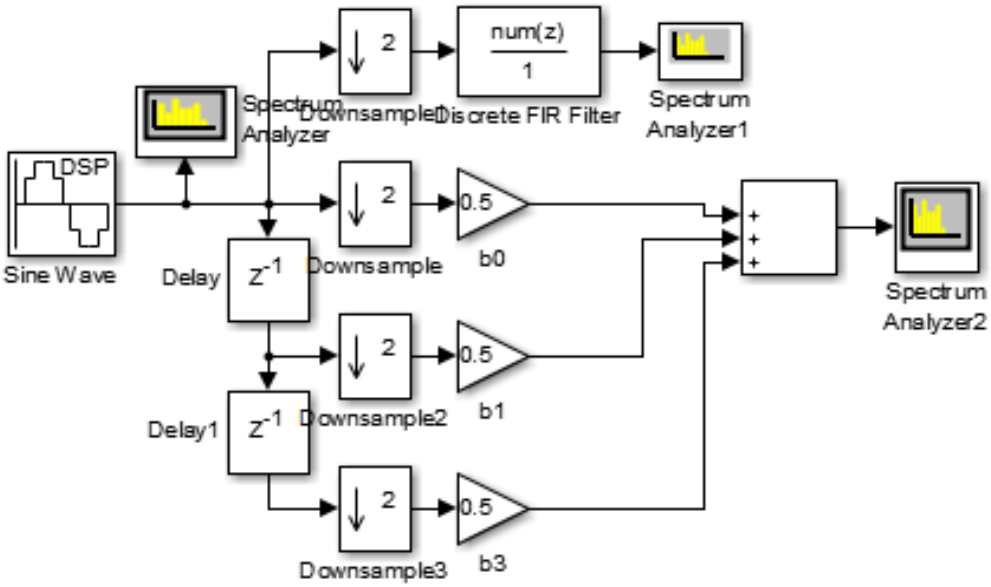


Figure IV.14. Exemple de filtre RIF à décimation

Les spectres obtenus par le schéma sont donnés par les figures IV-15 et IV-16 :

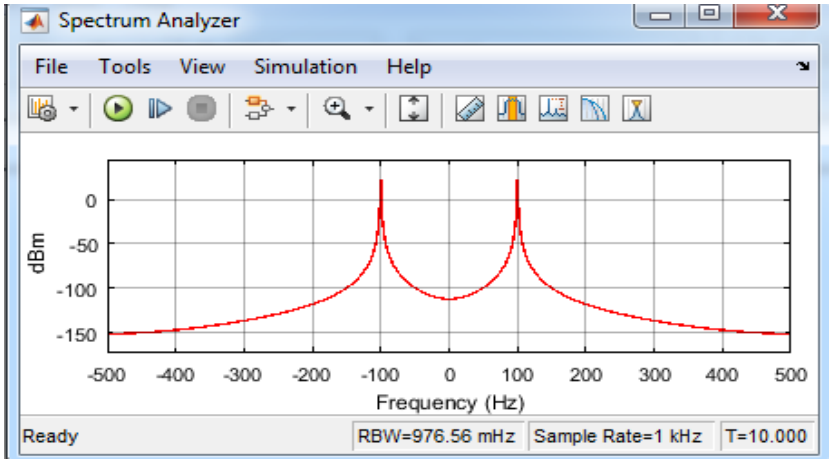


Figure IV.15. Spectre du signal source filtre RIF à décimation

La fréquence d'échantillonnage est égale à 1kHz à l'entrée comme il est bien indiqué dans le schéma précédent.

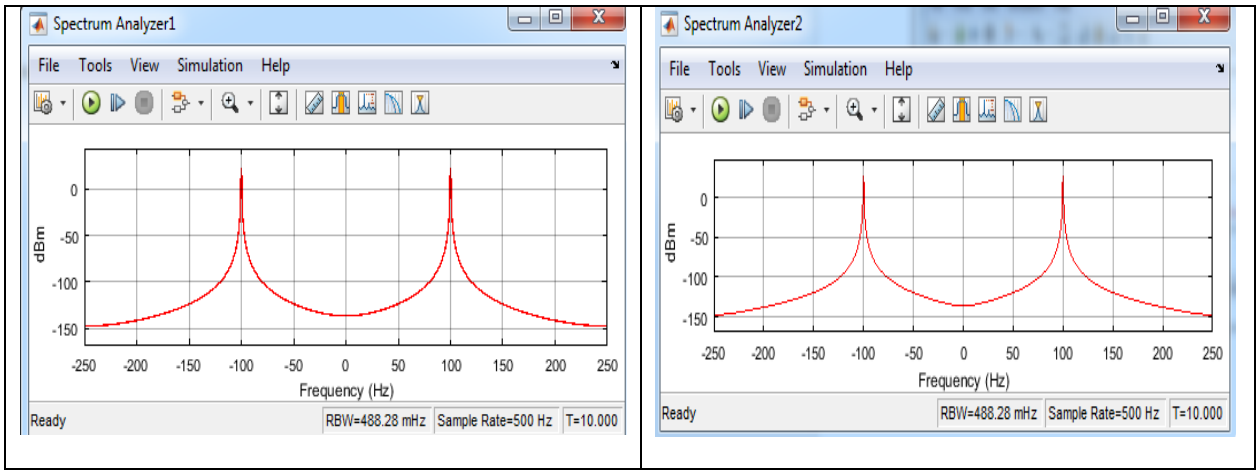


Figure IV.16. Spectre du signal à la sortie d'un filtre RIF à décimation

A la sortie et après le sous-échantillonnage avec un facteur de décimation M=2, on obtient une bande de fréquence de 500Hz, soit le un demi de la fréquence d'échantillonnage à l'entrée.

Pour obtenir un filtre RIF d'interpolation on ajoute au filtre RIF déjà réalisé précédemment des blocs Upsample (lien montant).

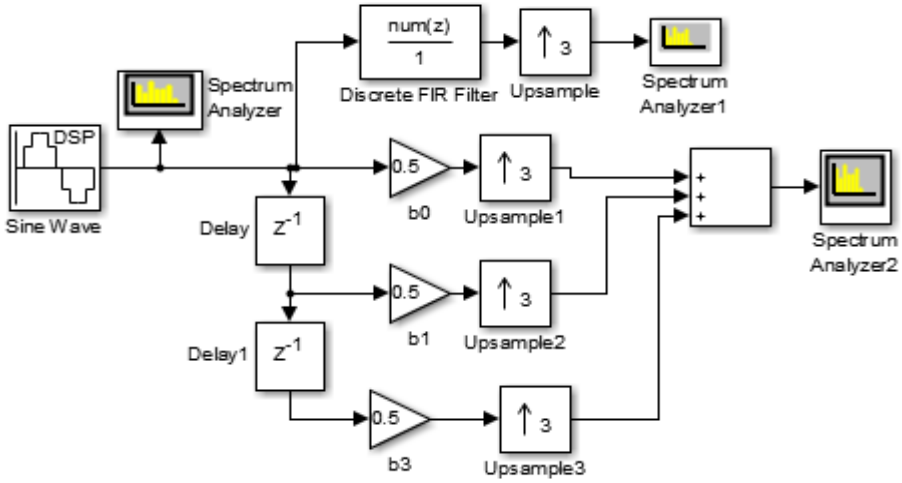


Figure IV.17. Exemple de filtre RIF d'interpolation

Les spectres obtenus par le schéma sont donnés par les figures IV-18 et IV-19 :

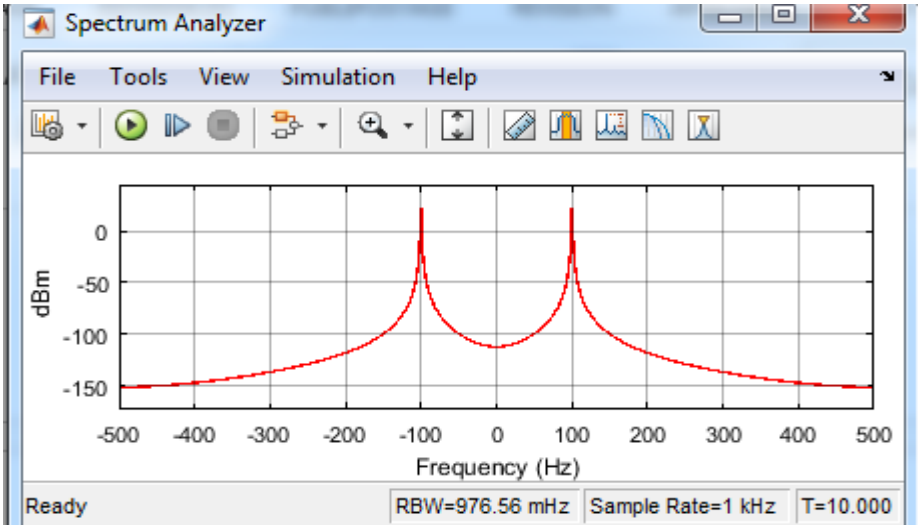


Figure IV.18. Signal à l'entrée du filtre RIF d'interpolation

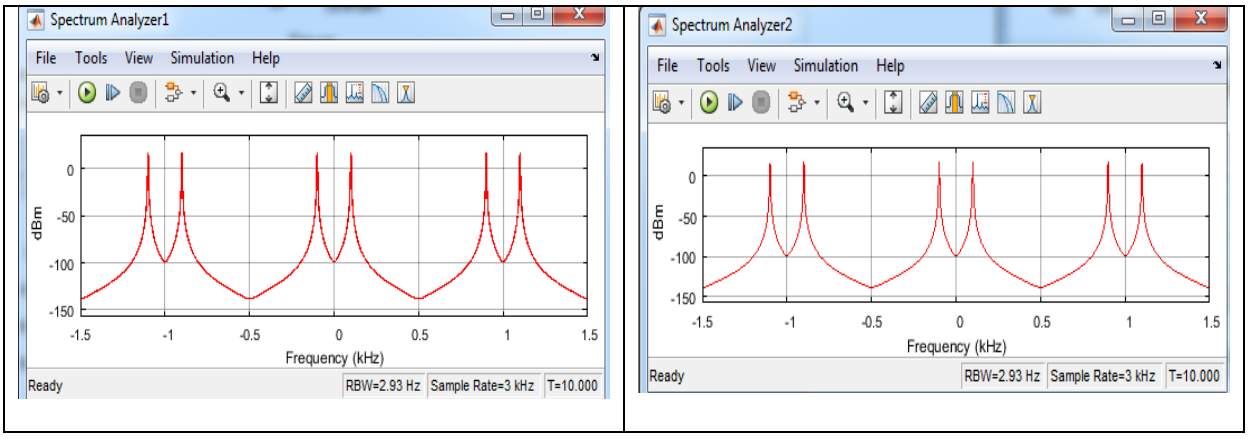


Figure IV.19. Signal à la sortie du filtre RIF d'interpolation

Dans ce cas et après l'opération de sur-échantillonnage en appliquant un facteur d'interpolation L=3, la fréquence d'échantillonnage du signal de sortie est égale trois fois celle du signal d'entrée.

IV.4.2. Réalisation d'un filtre RIF de décimation polyphasique :

Pour ce fait, on a besoin des retards, des Downsimplink , des filtres RIF et d'un sommateur. Le shéma correspondant est le suivant :

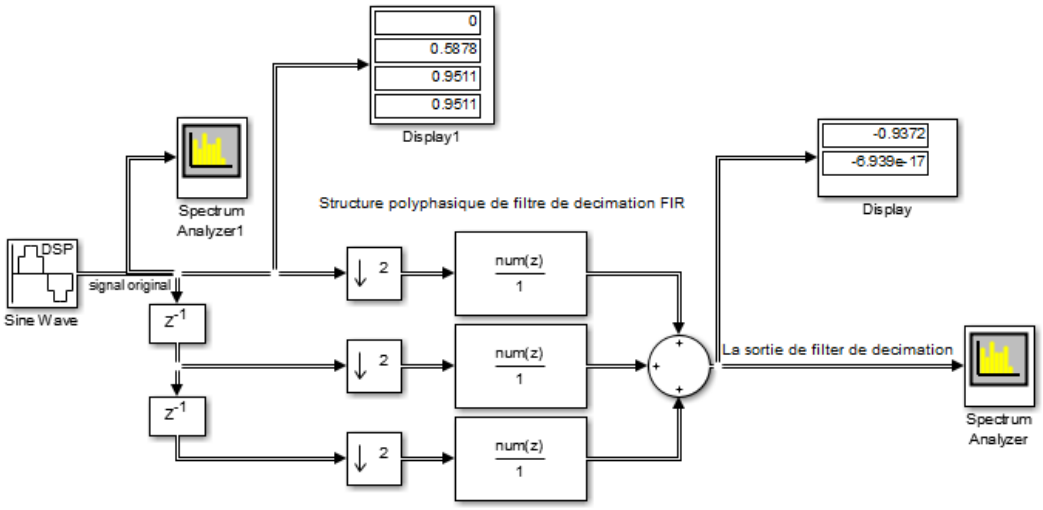


Figure IV.20. Exemple de filtre RIF polyphasique de décimation.

Les spectres obtenus par le schéma sont donnés par la figure IV-21 :

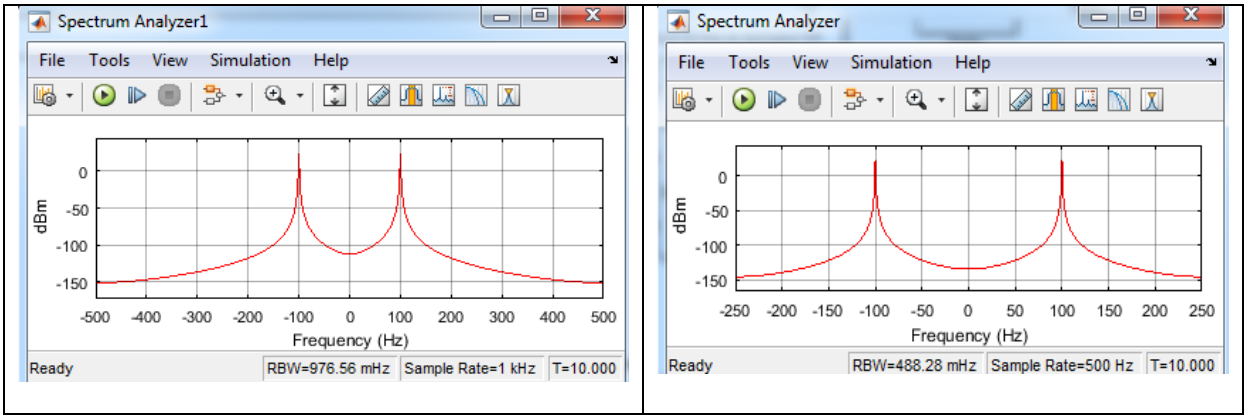


Figure IV.21. Spectre à l'entrée et à la sortie du filtre RIF de décimation.

On remarque que la fréquence d'échantillonnage à la sortie est égale à la moitié de celle appliquée à l'entrée du filtre.

IV.4.3. Réalisation d'un filtre RIF d'interpolation polyphasique :

Pour ce fait, on a besoin des retards, des Upsimplink , des filtres RIF et des sommations. Le shéma correspondant est le suivant :

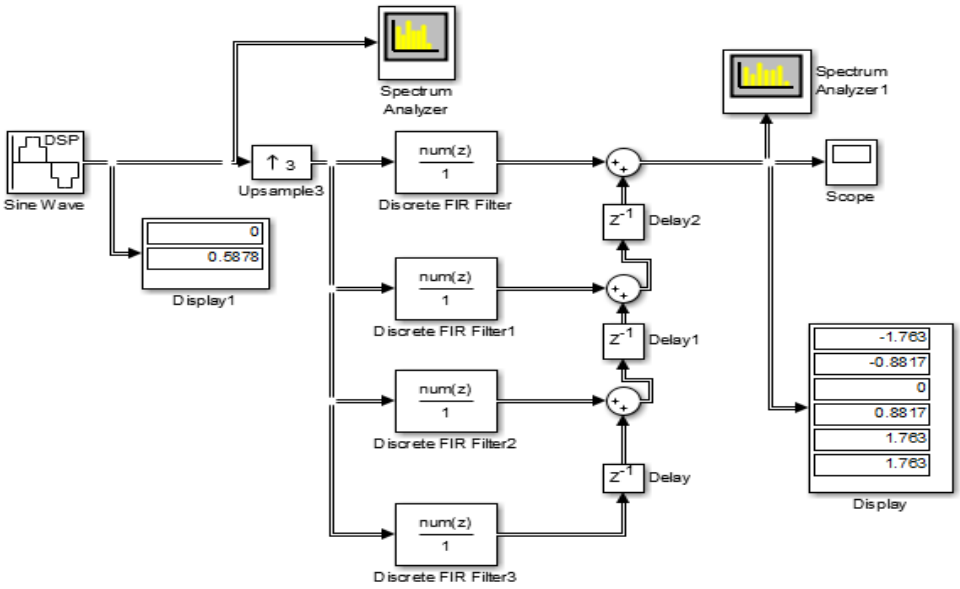


Figure IV.22. Exemple de filtre RIF polyphasique d'interpolation.

Les spectres obtenus par le schéma sont donnés par la figure IV-23 :

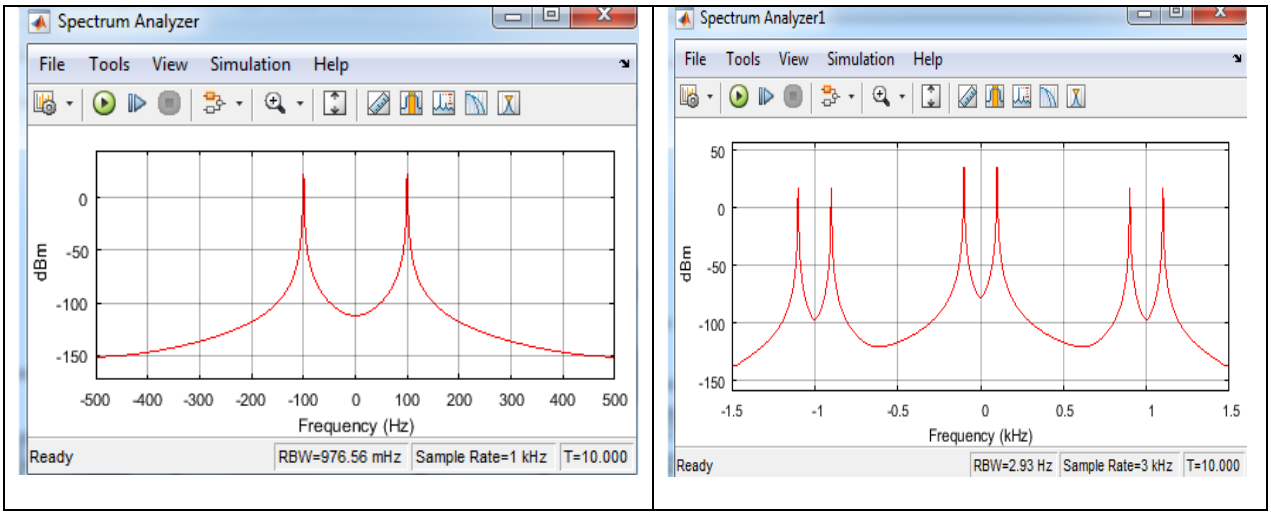


Figure IV.23. Spectre du signal à l'entrée et à la sortie du filtre RIF polyphasique d'interpolation

La fréquence d'échantillonnage à la sortie de ce filtre est égale trois fois celle de l'entrée avec un facteur d'interpolation L=3.

IV.4.4. Implémentation d'un filtre multi-cadence dans une carte Raspberry : (pour installer le supporte package de raspberry Pi on va appliquer les mêmes étapes de l'installation de supporte package de DSP mais en a pas besoin de l'installation de CCS)

Les composants que nous avons utilisés sont :

- Raspberry Pi B ;
- Carte mémoire micro SD ;

- Câble réseau ;
- Câble micro-USB.

IV.4.5. La conception de base d'un filtre de décimation :

L'ordre de placement des différents blocs composant ce schéma est montré dans la figure IV.24. Ce modèle comme on peut le remarquer dans la figure IV-24 contient deux macro-blocs (contrôle de gain automatique, Raspberry Pi sortie), un sous-système (3 sous échantillonnage, 2 retards et 3 filtres) et 6 blocs (sine wave, ADC, integer to bit, autocorrelation LPC, Data Type Conversion et décodeur de Viterbi).

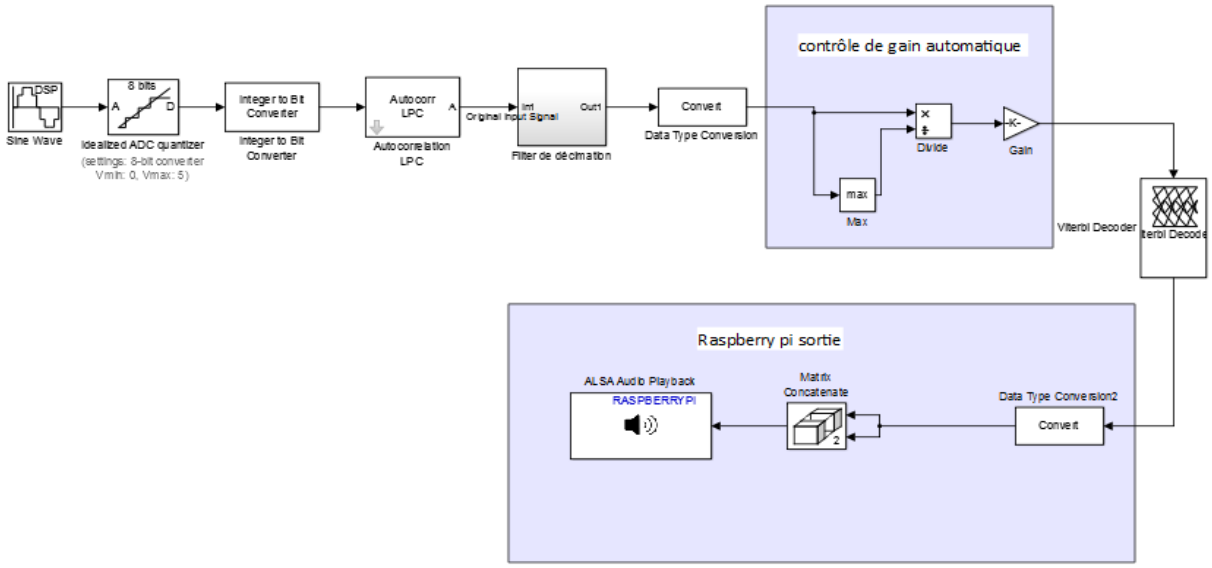


Figure IV.24. Schéma Simulink basique d'un filtre de Décimation

a. Bloc sine wave :

Le bloc « Sine Wave » génère un signal sinusoïdal complexe ou réel multicanal, avec amplitude, fréquence et phase indépendantes dans chaque canal de sortie. Un signal sinusoïdal réel est généré lorsque le paramètre de complexité de sortie est réglé sur Réel, et est défini par une expression de type (voir la figure **IV.25)** :

$$y = A\sin(2\pi ft + \phi) \qquad \text{(IV.1)}$$

Où vous spécifiez A dans le paramètre Amplitude, f en hertz dans le paramètre Fréquence, et ϕ en radians dans le paramètre Décalage de phase. Un signal exponentiel complexe est généré lorsque le paramètre Output complexité est réglé sur complexe, et est défini par une expression de type :

$$y = Ae^{j(2\pi ft+\phi)} = A\left[\cos(2\pi ft+\phi) + j\sin(2\pi ft+\phi)\right] \qquad \text{(IV.2)}$$

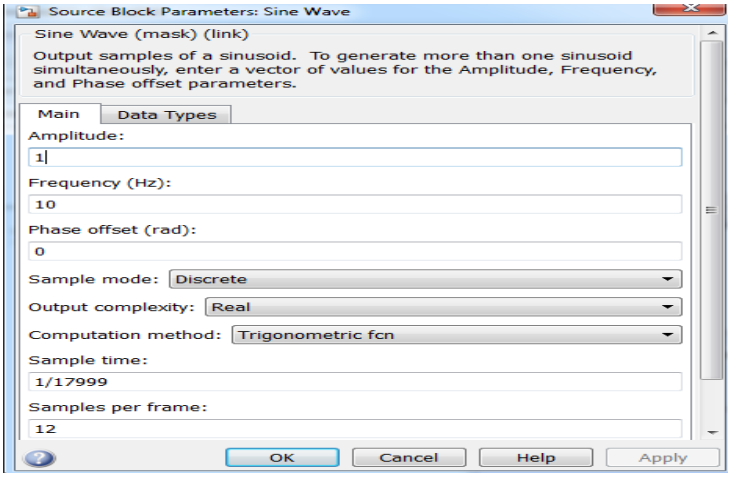


Figure IV.25. Les paramètres de bloc Sine Wave.

b. Bloc ADC :

Le bloc « ADC » est un quantificateur idéal pour un convertisseur analogique-numérique linéaire.

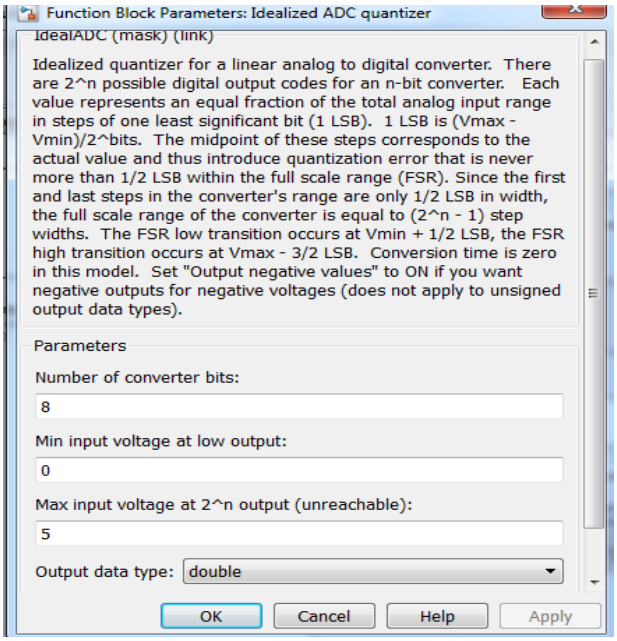


Figure IV.26. Paramètres du bloc ADC

c. Bloc integer to bit converter :

Le bloc « integer to bit » est un bloc qui va convertir un vecteur d'entrées de valeurs entières à un vecteur en binaire.

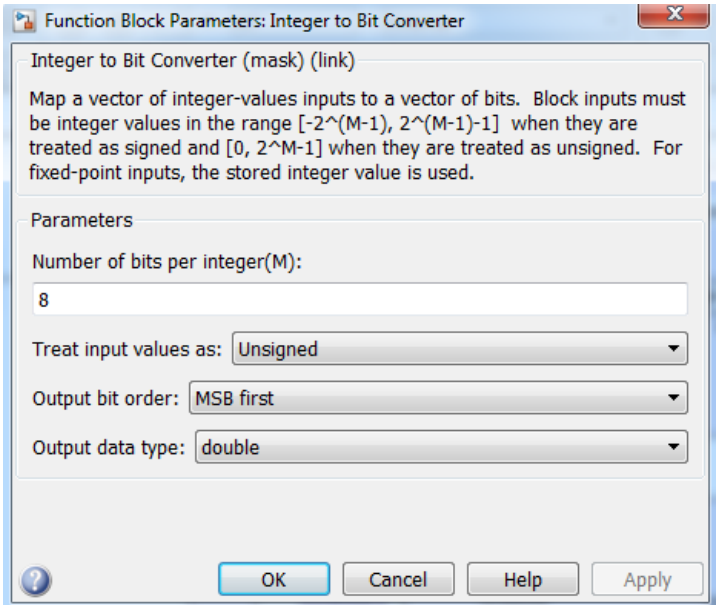


Figure IV.27. Parametres du bloc Integer to Bit Converter

d. Bloc data type converter :

Il est responsable de convertir l'entrée au type de données et à l'échelle de la sortie. La conversion a deux objectifs, l'un d'entre eux est de faire en sorte que les valeurs réelles de l'entrée et de la sortie soient égales. L'autre objectif est de faire en sorte que les valeurs entières stockées de l'entrée et de la sortie soient égales. Les débordements et les erreurs de quantification peuvent empêcher l'objectif d'être pleinement atteint.

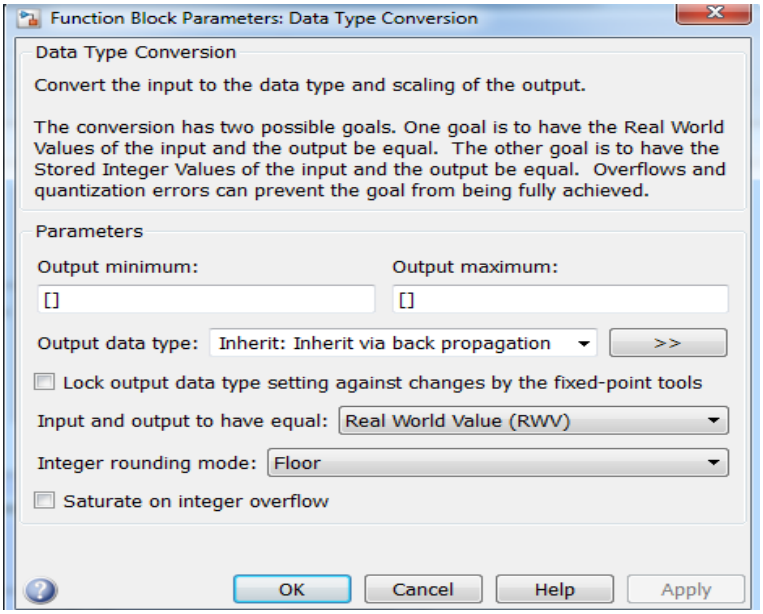


Figure IV.28. Paramètres du bloc Data Type Conversion

d. La macro bloc contrôle du gain automatique :

Le contrôleur de gain automatique adapte la plage dynamique du signal à l'intensité du signal. La structure de macro-bloc est définie dans la figure IV-29 et est basée sur les 3 blocs (Max, Diviser et le gain).

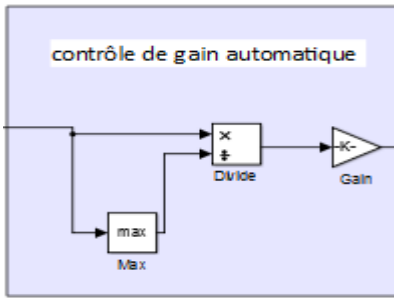


Figure IV.29. Contrôleur de gain automatique

Pour chaque trame d'entrée, le contrôleur de gain automatique effectue une normalisation, en divisant tous les échantillons de chaque trame par leurs valeurs maximales.

e. Bloc Viterbi decoder :

Le décodeur de Viterbi décode les symboles d'entrée pour produire des symboles de sortie binaires. Ce bloc peut traiter plusieurs symboles à la fois pour des performances plus rapides.

Ce bloc est capable de générer des séquences dont la longueur varie pendant la simulation. Pour plus d'informations sur les séquences dont la longueur ou les signaux de taille varient, on se référencie de la section base des signaux de taille variable dans la documentation Simulink.

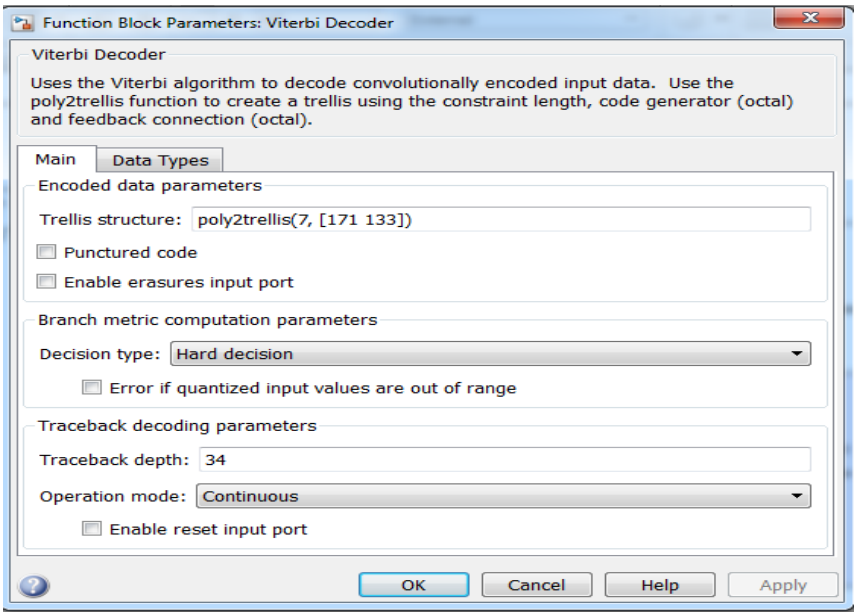


Figure IV.30. Paramètres du Décodeur de Viterbi

f. La sortie Raspberry Pi :

Le macro bloc de sortie Raspberry est le port de sortie du signal. Ce macro bloc se compose de 3 blocs : Data type conversion, Matrix concatenate et ALSA Audio Playback.

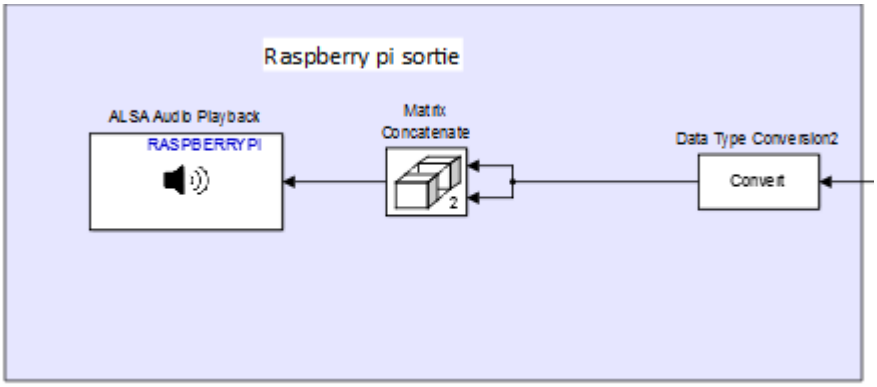


Figure IV.31. La sortie Raspberry Pi

f.1. Bloc Matrix Concatenate :

Le bloc matrix concatenate permet de concaténer plusieurs signaux en un seul de sortie. Le signal de sortie peut être un vecteur ou un tableau multidimensionnel. Dans ce projet, ce bloc est utilisé en mode multidimensionnel.

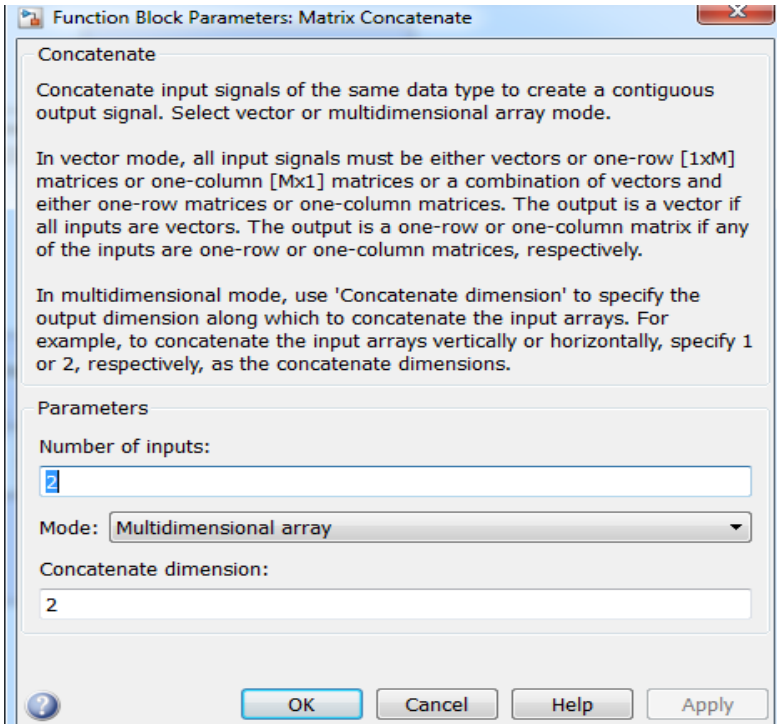


Figure IV.32. Paramètres du Bloc Matrix Concatenate

f.2. ALSA Audio Playback :

Le bloc ALSA Audio Playback représente la sortie analogique de notre carte Raspberry Pi, il est responsable de la conversion du signal analogique en numérique avant de le transférer vers la carte son pour la lecture. Ce bloc est dépendant du pilote audio ALSA du Raspberry Pi, qui gère tous les périphériques audio connectés au Raspberry Pi. Le signal d'entrée doit avoir une dimension de [N*2].

La sélection du taux d'échantillonnage se fait dans le champ de la fréquence d'échantillonnage audio. Dans ce modèle et tous les modèles suivants, un taux d'échantillonnage de 4000 échantillons correspondant à la valeur la plus élevé a été utilisé.

L'identificateur de la carte son doit être défini dans le champ « Device name field » : l'entrée est définie par la syntaxe « plughw: card,device », où les deux paramètres carte et périphériques peuvent être facilement obtenus en utilisant la commande aplay -1 dans shell sur Linux. Les résultats de tous les appareils connectés au Raspberry et de leurs paramètres vont être aussi tôt imprimés sur notre écran.

IV.4.6. Exécution sur la carte Raspberry :

Après avoir terminer la conception de notre schéma sur SIMULINK et après l'avoir testé, on va l'implémenter sur la carte, et pour cela on va connecter la carte Raspberry Pi à notre ordinateur comme le montre la figure qui suit, tout en fournissant à Simulink les informations liées à notre carte afin qu'il puisse la reconnaitre.

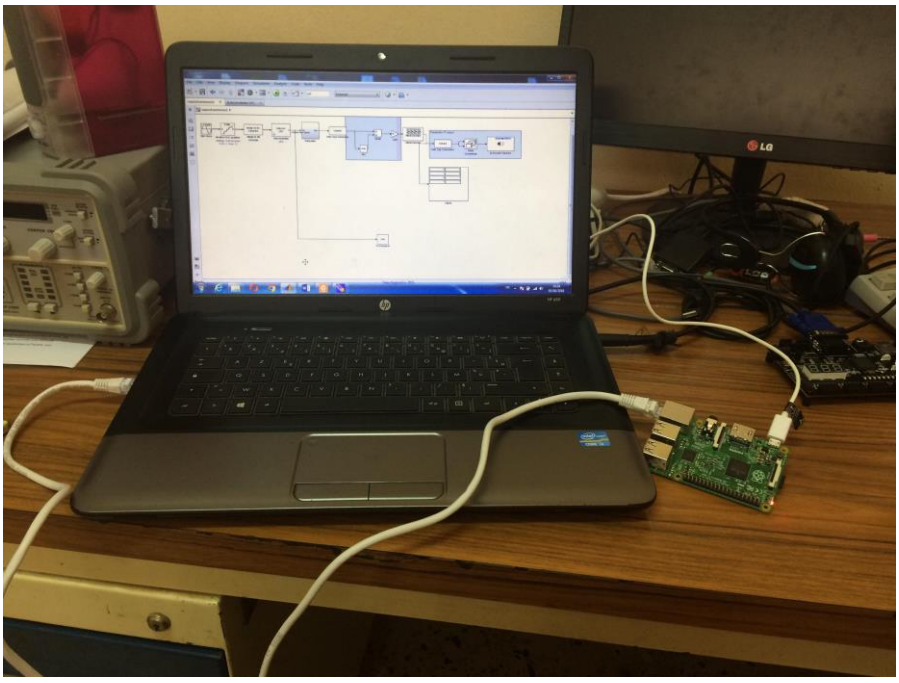


Figure IV.33. Interconnexion entre ordinateur et Raspberry Pi

Pour que Simulink sache qu'on va utiliser une carte Raspberry, on sélectionne « Tools/Run on Target Hardware/Option » puis on choisit Raspberry Pi (voir la figure **IV.34**).

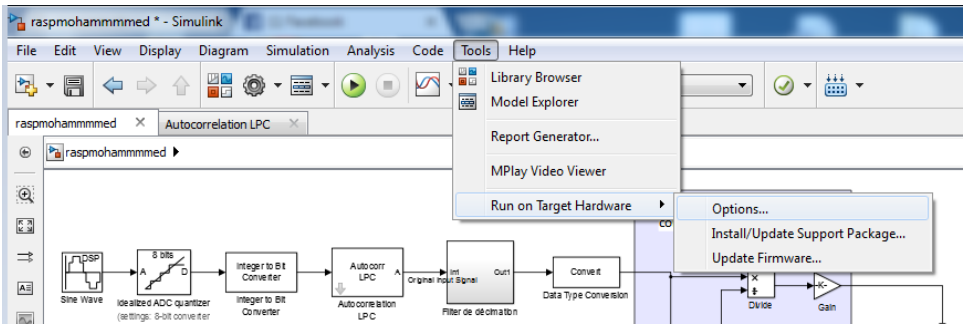


Figure IV.34. Exécution sur Raspberry Pi

Dans la fenêtre qui apparaît (voir figure **IV.35**), on introduit tous les paramètres concernant le nom de l'appareil, le nom d'utilisateur, le mot de passe et le répertoire Build.

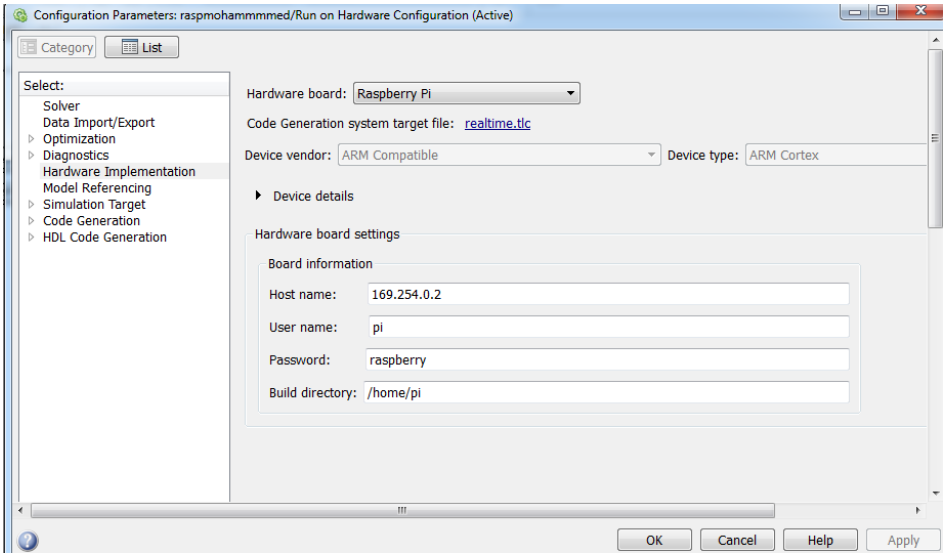


Figure IV.35. Fenêtre d'introduction des données sur la carte

IV.4.7. Lancement de l'exécution avec SIMULINK :

Simulink permet l'exécution des programmes sur la carte Raspberry Pi. Le mode appliqué est le mode externe, ce qui nous permet de maintenir la connexion entre Simulink et la carte pour une bonne gestion (choix d'amplitude, arrêter l'exécution,...) (Voir la figure **IV.36**).

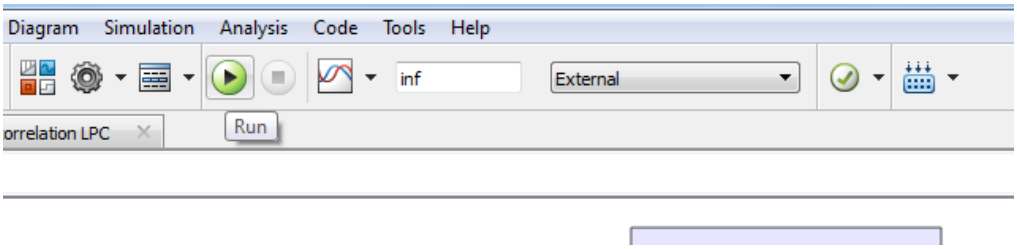


Figure IV.36. Barre de gestion sur Simulink

IV.4.8. Les résultats à l'exécution du filtre de décimation :

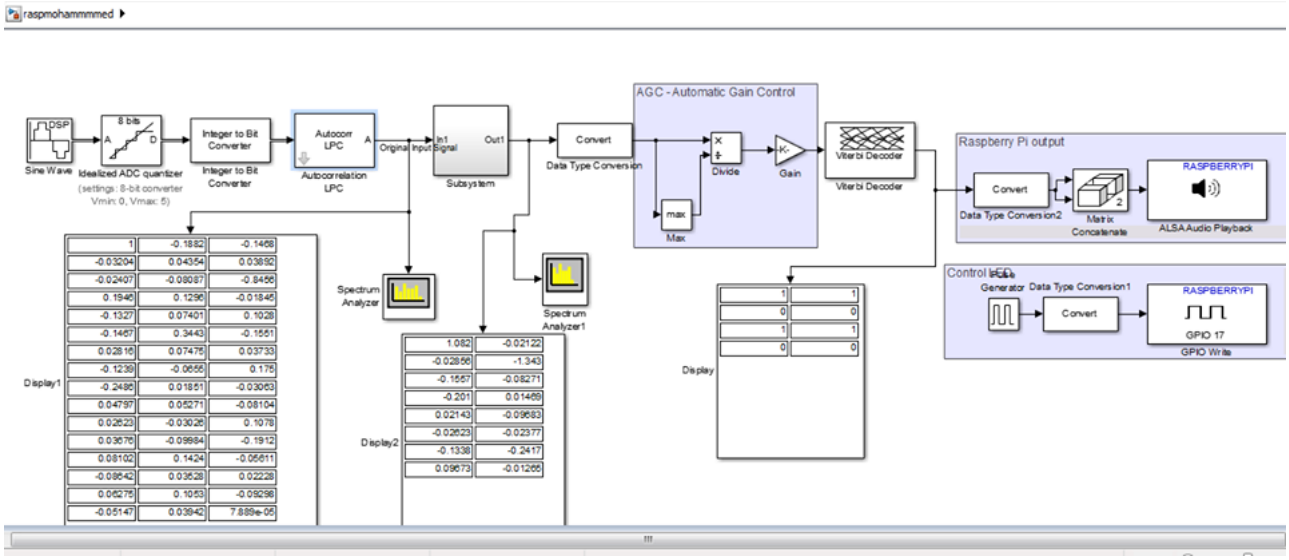


Figure IV.37. Résultats de l'exécution d'un filtre de Décimation sur Raspberry

On remarque qu'à l'entrée du bloc Subsystem (voir figure IV.37) qui contient un filtre RIF de décimation avec un coefficient de 3 comme il est montré dans la figure IV.38, il y'a 48 valeurs, et à la sortie il a été divisé pour qu'il en reste que 16 valeurs.

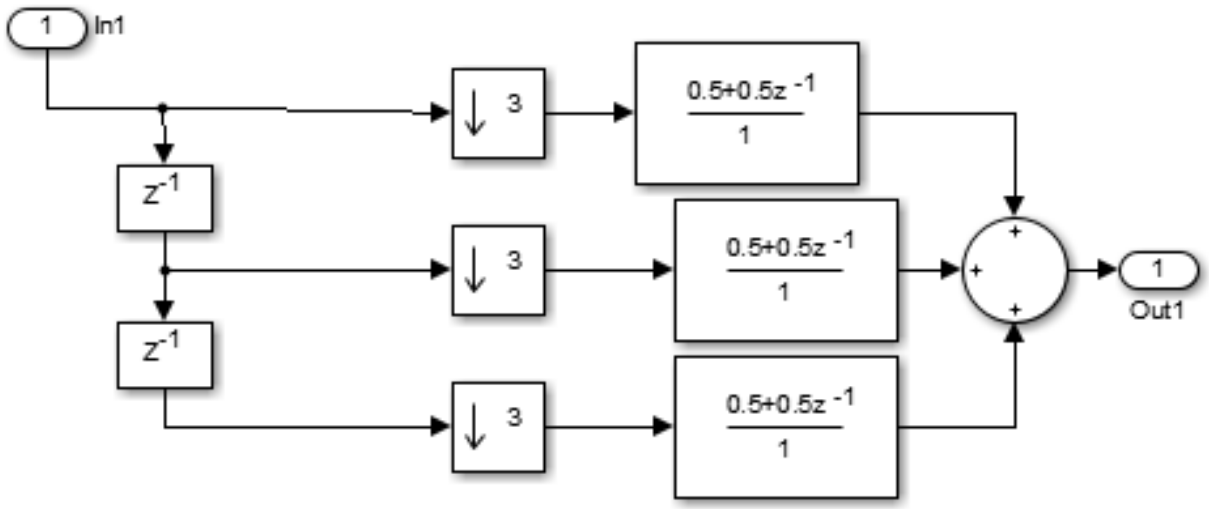


Figure IV.38. Composants du bloc Subsystem

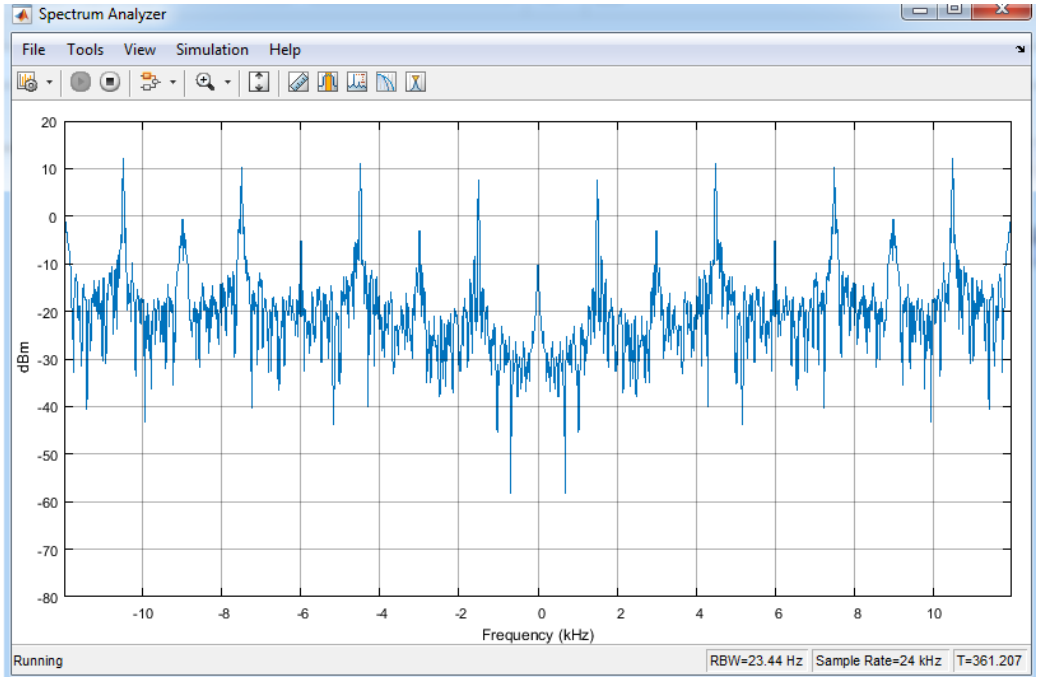


Figure IV.39. Spectre du signal avant décimation

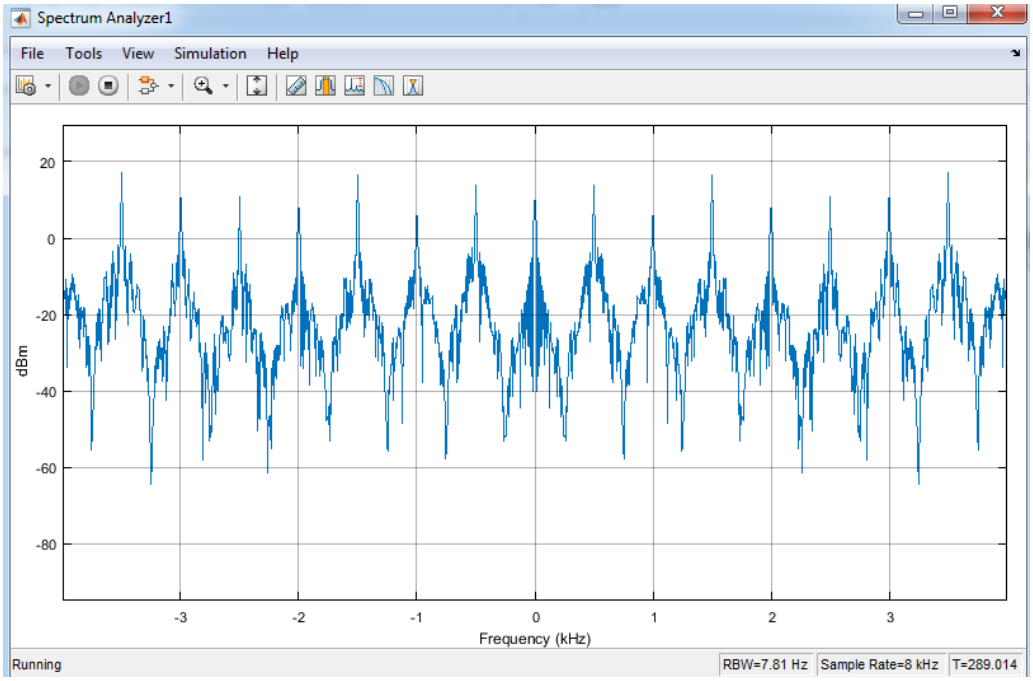


Figure IV.40. Spectre du signal après décimation

IV.5. Conclusion :

Dans ce chapitre, on a étudié la réalisation des filtres RIF sous Matlab et sous Simulink, et on a défini toutes les étapes pour arriver à l'exécution de notre programme.

Vu que la carte DSP n'est pas disponible, on a voulu la remplacer par une carte FPGA, mais les packages ne sont compatibles, on s'est donc contenté d'implémenter notre programme dans un Raspberry Pi, mais tant que le programme fonctionne à la perfection le principe reste le même

Conclusion générale

Dans ce mémoire, nous sommes intéressés à la conception d'une classe des filtres numériques. En effet, nous avons commencé par une étude générale en ce propos.

Par conséquent, le problème de conception de filtres numériques a reçu beaucoup d'attention. C'est pour cette raison que des systèmes qui utilisent plusieurs fréquences d'échantillonnage différentes, plus connus sous le nom de systèmes multi-cadences ont été mis au point. Ces systèmes, comparés aux systèmes traditionnels, offrent de meilleures performances à un coût réduit. Ce réajustement de fréquence peut se faire de deux façons : la première façon est de passer d'une grande fréquence d'échantillonnage à une fréquence d'échantillonnage beaucoup plus petite (décimation) à l'aide d'un filtre à décimation. La deuxième façon, consiste à transformer une petite fréquence d'échantillonnage en une beaucoup plus grande (interpolation) à l'aide d'un filtre à interpolation. Nous avons étudié une méthode de conception de filtre multi-cadence d'une façon détaillée, c'est celle des filtres RIF, avec des structures assez régulières. Ces structures ne permettent jamais d'aller vers des fréquences d'échantillonnages élevés. Ainsi, pour accélérer la vitesse de calcul de ces filtres, la décomposition polyphasée est utilisée. Cette technique fonctionne bien si le facteur de décimation est petit. Tout processus de décimation utilise un filtre passe bas avec une fréquence de coupure égale à π/M pour limiter la largeur de bande du signal. Ainsi, si le facteur de décimation est grand, alors la largeur de bande du filtre est étroite. Sa bande de transition est préférablement étroite aussi. Ceci nécessite un filtre RIF de grand ordre ce qui engendre à son tour un grand nombre de multiplications à effectuer.

Dans le troisième chapitre nous constatons que les DSP ne représentent qu'une partie du traitement numérique du signal, mais son rôle reste essentiel par son rôle important dans la chaîne de traitement.

Dans le quatrième chapitre nous avons fait l'exemple de quelques programmes sur Matlab pour mieux comprendre le principe du filtrage multi-cadence, et nous avons présenté les résultats après la réalisation et l'exécution de notre programme sur Simulink.

Bibliographie :

[1] : Jean Phylipe _ cours. https://fr.scribd.com/doc/271141551/Filtres-numeriqu

[2] : OLIVIER Sentiys_le 29september 2014.

[3] : OLIVIER Sentiys_le 28mais2014.

[4] : ALI DAHER, Application de la théorie des nombres à la conception optimale et à l'implémentation de très faible complexité des filtres numériques, 8 Décembre 2009

[5] : Mr Benchenief Abderrezak _ jain 2009.

[6] : Ljijana Milic,Multifiltring for digital signal prossissing,2009.

[7] : OLIVVIER VENARD, Algorithmes pourr le traitement du signal, 2009-2010

[8] : P.P.VAIDYANATHAN, SENIOR, MEMBER, IEEE, Juin 1993.

[9] : N .J .Fleige Degital Signal processing (multirate, systems, Filter, Banks, Wavlet).

[10] : Christian S.GARGOUR,Marcel GABREA et Venkat RAMACHANDRAN, traitement numéri.que des signaux 3eme édition Presses de l'Uneversité de Québec.

[11] : Gordana Javanovic Dolecek Editor, Advances in multirate systems.

[12] : Mme S. Hanafi. Université des sciences et des technologies d'Oran (MB), Laboratoire signaux et image.

[13] : www.technologuepro.com .

[14] : http://www.dspguide.com/ch28/3.htm

[15] : REAL-TIME DIGITAL SIGNAL PROCESSING based on the TMS320C6000, NASSER kehternavaz, with Labotory contrinbution by NAMJIN.

[16] : TMS320 DSP Development Support Reference Guide, May 1998.

[17] : TMS320C54x, TMS320LC54x, TMS320VC54x FIXED-POINT DIGITAL SIGNAL PROCESSORS, FEBRUARY 1996 – REVISED DECEMBER 1999.

[18] : Data Manual,March 1999–Revised January 2005.

Résume :

L'objectif principal de ce mémoire est l'étude et l'implémentation d'un filtre multi-cadence. Un filtre multi-cadence est un filtre numérique qui change la fréquence d'échantillonnage au cours de la chaine de traitement afin qu'elle soit la plus adaptée au traitement à réaliser, en d'autres termes, il modifie le contenu spectral du signal d'entrée en atténuant ou éliminant certaines composantes spectrales non désirées.

Les principaux opérateurs utilisés en traitement numérique du signal multi-cadence sont la décimation (réduction d'un facteur M de la fréquence d'échantillonnage) et l'interpolation (augmentation d'un facteur M de la fréquence d'échantillonnage) ou une combinaison des deux.

Mots clés : Filtrage, Filtre RIF, Filtre RII, DSP, Simulink Model base design

Abstract :

The aim of this document is to study and implement of a multi-cadence filter. A multi-cadence filter is a digital filter that changes the sampling frequency during the processing sequence so enhance the performance. In addition, this modifies the spectral content of the input signal by attenuating or eliminating certain unwanted spectral components.

The main operators used in multi-rate digital signal processing are decimation (reduction by a factor M of the sampling frequency) and interpolation (increase by a factor M of the sampling frequency) or a combination of both.

Keywords: Filtrage, Filtre RIF, Filtre RII, DSP, Simulink Model base design.

الملخص:

الهدف الرئيسي من هذه الرسالة هو دراسة وتنفيذ مرشح متعدد المستويات. المرشح متعدد المعدلات هو مرشح رقمي يغير تردد أخذ العينات أثناء سلسلة المعالجة بحيث يكون هو الأكثر ملاءمة للمعالجة التي يتعين القيام بها ، وبعبارة أخرى ، فإنه يعدل المحتوى الطيفي للإشارة. الإدخال عن طريق تخفيف أو إزالة بعض المكونات الطيفية غير المرغوب فيها.
من تردد M العوامل الرئيسية المستخدمة في معالجة الإشارات الرقمية متعددة المعدلات هي الاستهلال (تخفيض عامل من تردد الاعتيان) أو توليفة على حد سواء M الاعتيان) والاستكمال الداخلي (زيادة عامل.

كلمات البحث: تصفية ، مرشح RIF ، RII ،DSP تصميم قاعدة ، Simulink النموذجي