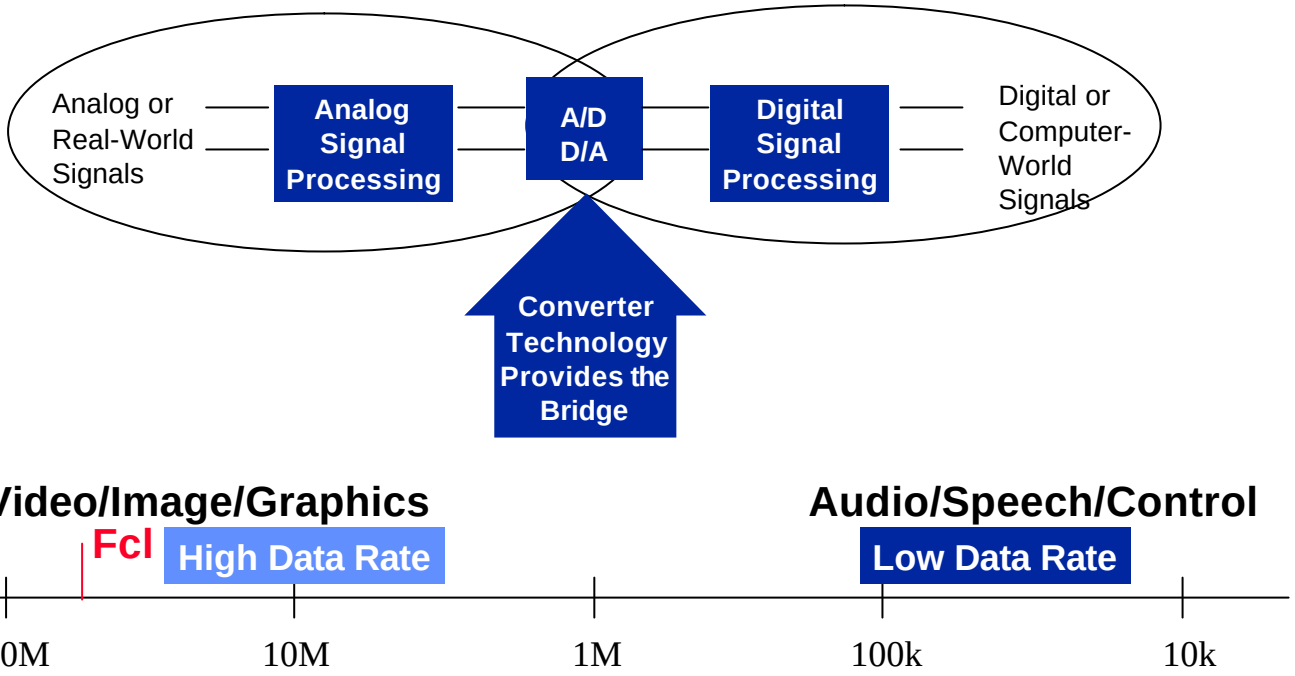
EII2

Traitement Numérique du Signal

Olivier SENTIEYS
sentieys@enssat.fr
http://lasti.enssat.fr/GroupeArchi/enseignements/Tns

Notes :

Traitement Numérique du Signal
(Digital Signal Processing)



[ICE97] 2

Notes :

Digital signal processing (DSP) is a segment of the IC industry where advanced digital and analog technologies merge. The typical function of the DSP device is to perform real-time processing of a digitized analog signal, changing that signal using arithmetic algorithms, and then passing the signal on. The process is very math intensive and quite complicated. In fact, finding competent DSP designers and programmers is often a challenge for many DSP manufacturers.

Plan du cours

I. Introduction

1. Introduction, problématique, caractéristiques, solutions architecturales
2. Types de signaux
3. Applications typiques de TNS
4. Chaîne de traitement et problèmes temps réel

II. Analyse des filtres numériques

1. Spécification, classification, représentation
2. Analyse fréquentielle
3. Structures des filtres RII et RIF

III. Transformées en TNS

1. TFD, convolution linéaire
2. TFR : Transformée de Fourier Rapide

Notes :

Plan du cours (suite)

IV. Quantification

1. Formats de codage
2. Quantification
3. Effets de la quantification en TNS

V. Synthèse des filtres numériques RII

1. Invariance Impulsionnelle
2. Transformation Bilinéaire

VI. Synthèse des filtres numériques RIF

1. Introduction
2. Filtres à Phase Linéaire
3. Méthode du Fenêtrage
4. Echantillonnage en Fréquence

Notes :

Plan du cours (fin)

VII. Analyse spectrale

1. Effets de la troncature
2. Caractéristiques des fenêtres
3. Influence sur l'analyse

VIII. Systèmes multi-cadences

1. Définition
2. Décimation
3. Interpolation

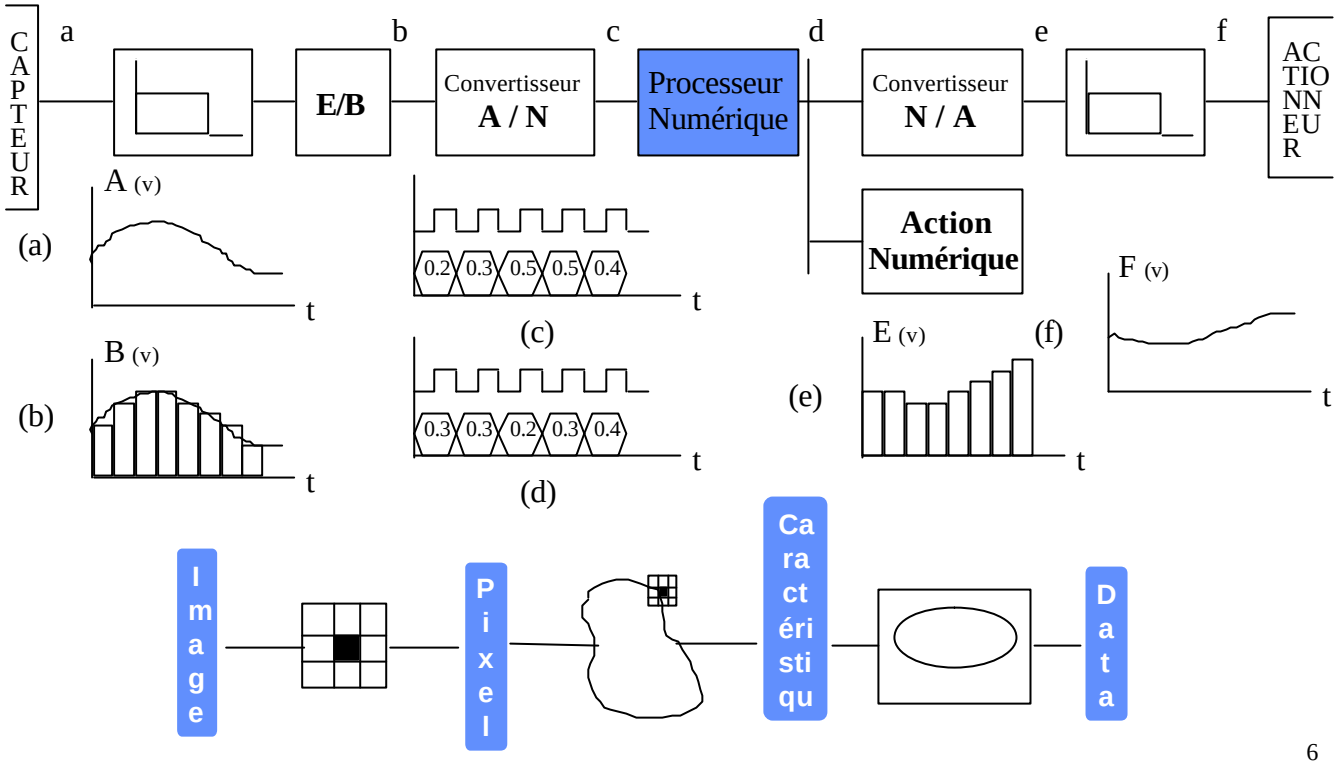
IX. Processeurs de Traitement du Signal

1. Architecture de base
2. Principales familles
3. Problèmes d'implantation en virgule fixe

Notes :

I.1 Introduction

Chaîne de TNS



Notes :

I.1 Traitement Numérique du Signal

- **Avantages**
  - Pas de dérive : température, vieillissement, valeur des composants
  - Précision : garantie par le nombre de bits
  - Souplesse : plusieurs tâches simultanées possibles
  - Prédiction : simulation sur ordinateur
  - Prototypes : changements par modifications du logiciel
  - Performances : pas de distorsion de phase, filtrage adaptatif
  - Intégration : progrès des systèmes VLSI (DSP, ASIC, ...)
- **Inconvénients**
  - Coût : élevé pour des réalisations simples
  - Vitesse : bande passante large = vitesse de calcul élevé
  - Complexité : réalisation à la fois matérielle et logiciel

Notes :

Quelles applications ?

- **A la Maison**
  - Télévision à la demande, Télévision Satellite, Jeu Vidéo et Réalité Virtuelle, Electroménager, Réseaux, ...
  - DVD, HDTV, CD, DAB

- **Au Bureau**
  - Vidéoconférence, Fax, Modems, Pagers, ...
  - Réseaux rapides, Sans-fil
  - ATM, ISDN, PBX, ADSL
- **Sur la route**
  - Téléphone cellulaires, Commande vocale, Radar et Sonar, GPS et traceur de route, Fax/Modems sans-fil, Automobile, ...

Systèmes de l'âge de l'information
= fusion entre
Calculateur- Télécommunications- Consommateur

Notes :

Glossaire du TNS

- **Saisie**, acquisition, conversion (A/N, N/A), codage
- **Filtrage**, FIR, IIR, convolution rapide, filtres spéciaux
- **Représentation**, modélisation, analyse spectrale, transformées
- **Compression**, approximation, extrapolation, codage de source, réduction de débit
- **Modulation**, codage de canal, protection contre les erreurs, cryptage, garantique
- **Détection**, réception optimale, démodulation, décodage, correction
- **Estimation**, paramétrique, estimation d'onde ou d'état, filtrage, prédiction, lissage
- **Analyse de système**, modèles de canaux, milieux de propagation
- **Amélioration**, réduction de bruit, annulation d'écho, compensation, égalisation
- **Déconvolution**, imagerie, résolution, détection de source, restauration
- **Classification**, reconnaissance, signatures
- **Apprentissage**, estimation séquentielle, adaptation, poursuite
- **Analyse temps fréquence**, non stationnarité, estimation de délais, de phase
- **Traitement multi-fréquences**, décimation, interpolation, filtrages en sous bandes
- **Arithmétique bit fini**, quantification, dépassement, virgule fixe, flottante, bruits de calcul, sensibilité des coefficients.
- **Architecture des systèmes**, DSP, ASIC, mémoire

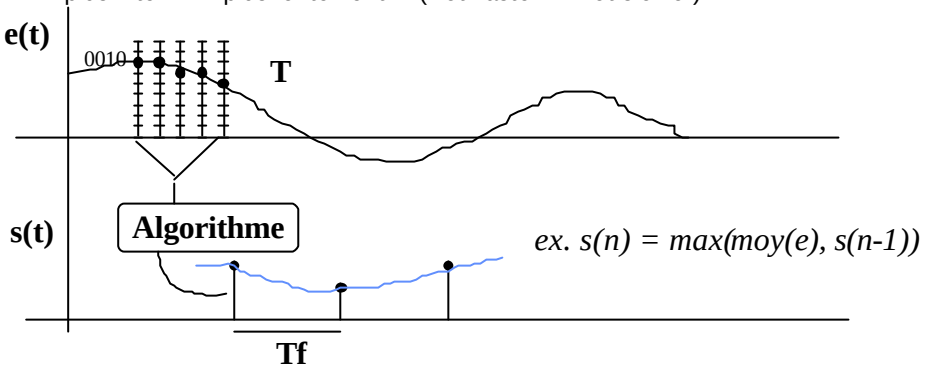
Notes :

Domaines d'application

- **Communication** homme machine, synthèse, transformation texte-parole et inverse, reconnaissance de parole, identification et vérification du locuteur
- **Télécommunications**, codage et restauration de la parole, courrier vocal, télécopie, audionumérique (CD, DAB), TV numérique, compression et transmission d'images, cryptage et protection, transmission de données, télé informatique, annulation d'écho, codage à débit réduit, télé et visioconférence, téléphonie cellulaire, ...
- **Défense**, systèmes d'armes, surveillance, guidage, navigation
- **Biophysique**, génie biomédical, EEG, ECG, radiographie, tomographie, scintigraphie, gammagraphie, échographie, aide aux handicapés, ...
- **Acoustique**, aérienne, sous-marine, sonar, ultrasons, nuisances
- **Géophysique**, sismique, de surface, océanographique, télédétection
- **Electromagnétisme**, radar, radionavigation, optique, astrophysique
- **Automobile** injection électronique, ABS, positionnement global, commande d'assiette adaptative
- **Musique**, numérique, MIDI, échantillonneurs (samplers), synthétiseurs, mélangeurs, réverbération et écho, effets spéciaux, filtrage, enregistrement (DAT)
- **Instrumentation**, capteurs, métrologie, analyse spectrale, génération de signaux, analyses de transitoires, DPLL
- **Graphisme et imagerie**, rotation 3D, vision, reconnaissance de formes, restauration d'images, stations de travail, animation, cartographie

Notes :

- **Systèmes DSP exécutent algorithmes temps réel sur des signaux numériques**
  - **Signaux numériques :** quantité mesurable, échantillonnée, quantifiée, encodée, vecteur multi-dimensionnel
  - **Flot de Données**
  - **Temps Réel**
    - Temps de l'exécution de l'algorithme **Tex** guidé par acquisition I/O
    - Période d'échantillonnage **T**
    - Période des sorties **Tf** (frame period) > **Tex**
    - Ni plus vite … ni plus lentement (Not Faster … not slower)



Notes :

I.1 Introduction

3. Caractéristiques des algorithmes

- **Quantité importante de données**
  - scalaires, vectorielles, matricielles, multidimensionnelles
  - opérations I/O intensives par DMA
- **Charge de calcul importante**
  - multiplications-accumulations (convolution)
  - multiplications-additions (FFT, DCT, adaptation, distances,...)
- **Virgule Fixe ou Flottante**
  - problèmes liés à la quantification !!!
- **Calculs d'adressage complexes**
  - bit-reverse ou similaire (FFT), vieillissement des données, ...
- **Boucles de traitement courtes**
  - les instructions peuvent être placées en interne
  - pas besoin de grande taille de cache (données ou instructions)

Notes :

Fonctions typiques de TS

- **Convolution, filtrage**

  y = y + x.h : MAC

- **Adaptation**

  $y_n = y_{n-1} + x.h$: MAD

- **FFT, multiplication complexe**

  xr=xr.wr-xi.wi; xi=xr.wi+xi.wr

- **Viterbi**

  a1 = x1 + x2; a2 = y1 + y2; y = (a1>a2) ? a1 : a2 : ACS

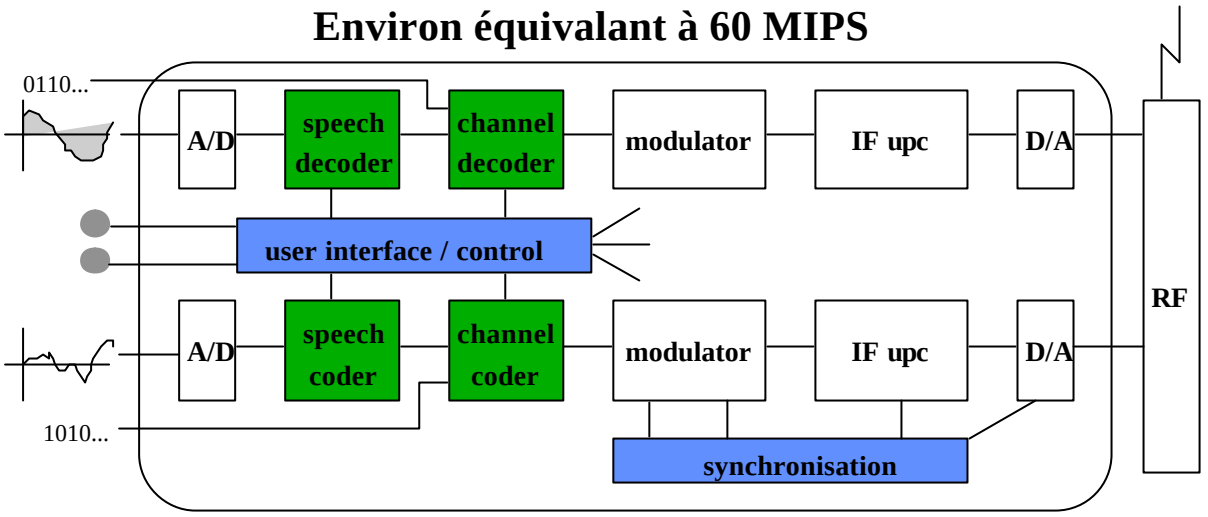
Notes :

4. Solutions architecturales

- **Processeurs programmables du commerce**
  - Processeurs généraux ***RISC, VLIW, Superscalaire***
  - Processeurs de Traitement du Signal (DSP) ***Conventionnel, VLIW***
  - Processeurs Multimédia
  - Microcontrôleurs
- **Processeurs programmables “maison” (ASIP)**
  - De type DSP ou µCtrl
- **Processeurs et logique reconfigurables**
  - FPGA enfouis, Processeur reconfigurable
- **Coprocesseurs ASIC**

Notes :

Exemple : GSM

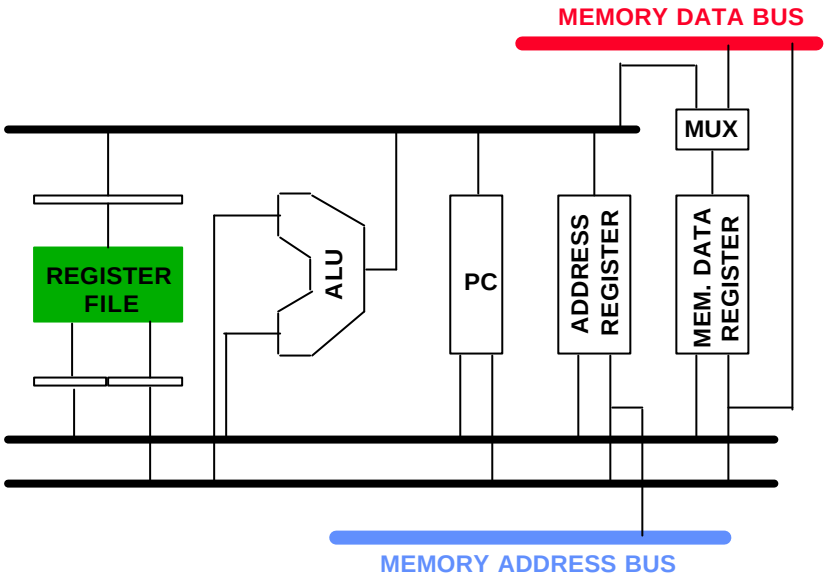


Coprocessor, Hardware
DSP, ASIP, ASP
µController

Notes :

Architecture Von Neumann

- **Les processeurs RISC**



Code example:
multiply & accumulate
*r2 = r2 + #imm * r1*

```
mov #imm,r3
mul r1,r3
add r3,r2
```

⇒ 3 instructions, >3 cycles d'horloge

Notes :

A typical data path of a RISC architecture is shown. It includes a register file with source and destination latches, an ALU (arithmetic and logic unit) and a program counter (PC). A RISC processor may also have additional registers for data and instruction addressing or other control realted functions.

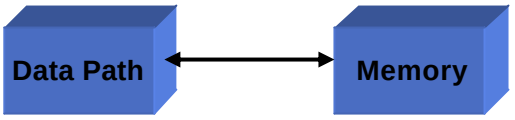
Most RISC designs use the same ALU to compute both algebraic operations and memory addresses for load and store operations. The justification for such a design is that because during load and store operations the ALU is not busy, such an implementation does not cause any performance penalty.

[Bhaskaran 95]

FIR sur machine Von Neumann

- **Problèmes**
  - Bande passante avec la mémoire
  - Code pour le contrôle et la gestion de l'adressage
  - Multiplication lente

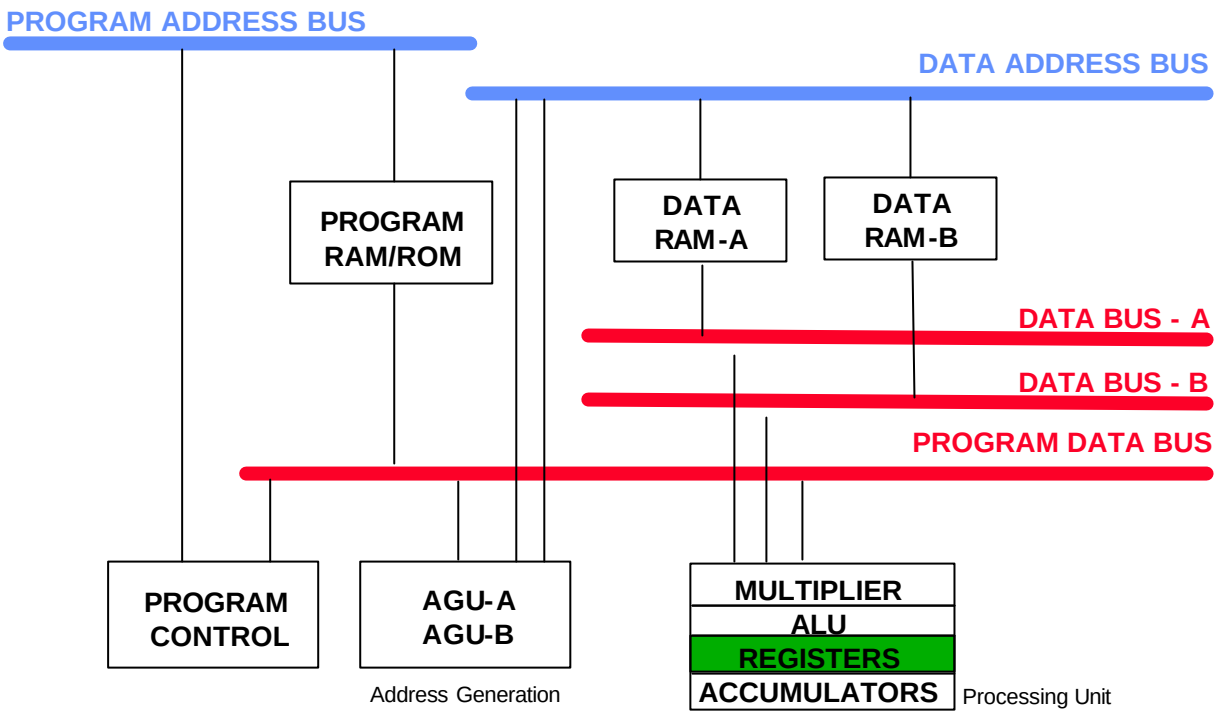
```
loop:
   mov *r0,x0
   mov *r1,x1
   mpy x0,y0,a
   add a,b
   mov y0,*r2
   inc r0
   inc r1
   inc r2
   dec ctr
   tst ctr
   jnz loop
```



Exécution en 15 à 20 cycles

Notes :

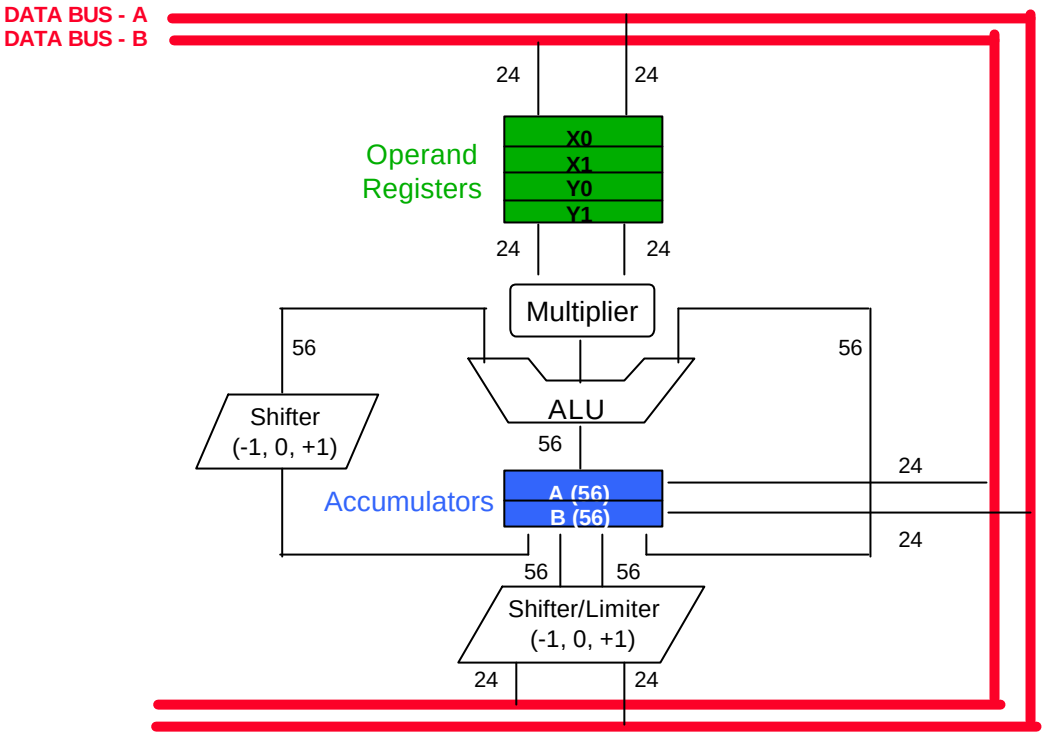
Architecture Harvard



Notes :

The figure shows the block diagram of a typical DSP core. It has a Harvard architecture, i.e., separate data and instruction busses and separate instruction and data memories, a processing unit, two data memories wirh their own address generation units (AGUs), a program controller, and program memory. The processing unit includes a parallel multiplier, an ALU, accumulators, and registers.

Architecture Harvard



Notes :

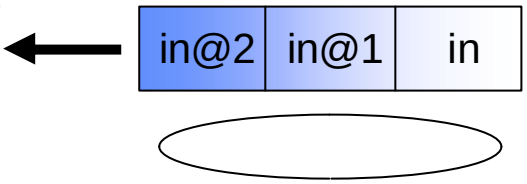
The MAC operation is useful in DSP calculations such as the dot vector product used for digital filtering. The dot product, derived by summing the products of vector element pairs, is efficiently calculated with repeated MAC operations. To achieve a single-cycle MAC, DSP processors integrate a multiplier and accumulator into the main data path of the processor (see figure). In addition to allow series of MAC ops to proceed without the possibilkity of arithmetic overflow, DSPs generally provide extra bits in the accumulator t accommodate the bit growth of the result.

A second feature shared by DSP processors is the ability to complete several accesses to memory in a single instruction cycle. This allows the processor to fetch an instruction while simultaneously fetching operands and/or storing the result of a previous instruction to memory. For example, in calculating the vector dot product o~ the sample vector and coefficient vector for an FIR filtep, almost DSP processors are able to perform a MAC while simultaneously loading the data sample and coefficient for the next MAC. In general, such single-cycle multiple memory accesses are subject to many restrictions. Typically, all but one of the memory locations accessed must reside on-chip, and multiple memory accesses can only take place with certain instructions. To support simultaneous access of multiple memory locations, DSP processors provide multiple on-chip busses, multi-ported on-chip memories, and in some cases multiple independent memory banks.

A third feature often used to speed arithmetic processing on DSP processors is one or more dedicated address generation units. Once the appropriate addressing registers have been configured, the address generation unit operates in the background (.e, without using the main data path of the processor), fonning the addresses required for operand accesses in parallel with the execution of arithmetic instructions. In contrast, genera1-pur~ose processors often requirc extra cycles to generate the addresses needed to load operands. DSP processor address generation units typically support a selection ol addressing modes tailored to DSP applications. The most common of these is register-indirect adressing with post-increment, which is used in situations where a repetitive computation is performed on a series of data stored sequentially in memory. Modulo addressing is often supported, to simplify the u,,~e of circular buffers Some processors also support bit-reversed addressing, which eases the task of interpreting the results of certain fast Fourier transform (FFT) al gorithms.

Architecture Harvard

- **Bus et mémoires données/instructions séparées**
- **Unité de traitement de type mpy-acc**
- **Registres distribués (1 RISC register file)**
  - Chaque module (ALU) possède ses propres registres locaux
- **Génération adresses efficaces (AGUs)**
  - Modes d'adressage spéciaux : auto incr-decr, circular buffering (delay line) ...



Notes :

Because many DSP algorithms involve performing repetitive computations, most DSP processors provide special support for efficient looping. Often, a special loop or repeat instruction is provide which allows the programmer to implement a for-next loop without expending any instruction cycles for updating and testing the loop counter or branching back to the top of the loop.

Finally, to allow low-cost, high-performance input and output, most DSP processors incorporate One Or more serial or parallel I/O interfaces, and specialized I/O handling mechanisms such as low-overhead interrupts and direct memory access (DMA) to allow data transfers to proceed with little or no intervention from the rest of the processor.

In some cases, system designers may prefer to use a general-purpose processor over a DSP processor. Although general-purpose processor architectures often require several instructions to perform operations done with just one DSP processor instruction, general-purpose processors sometimes compensate by running at extremely fast clock speeds. If the designer needs to perform non-DSP processing, then using a using a general-purpose processor for both DSP and non-DSP processing could reduce the system parts count and lower costs versus using a separate DSP processor and general-purpose microprocessor. Furthermore, some popular general-purpose processors feature a tremendous selection of application development tools.

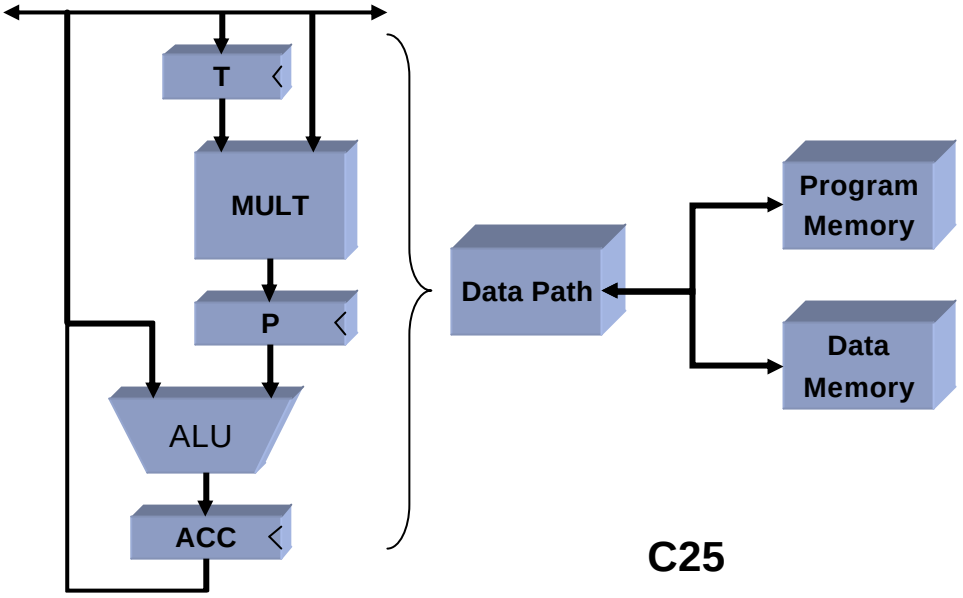
On the other hand, because general-purpose processor architectures lack eatures that simplify DSP programming, software development is sometimes more tedious than on DSP processors and can result in awkward code that's difficult to maintain. Moreover, if general-purpose processors are used only for signal processing, they are rarely cost-effective compared to DSP chips designed specifically for the task. Thus, at least in the short run, we believe that system designers will continue to use traditional DSP processors for the majority of DSP intensive applications.

FIR sur DSP conventionnel

- **Jeu d'instructions complexe**

```
loop:
   acc=acc+P
// P=T*PM(*r0+)
// T=DM(*r1+)
   jnz loop
```

Exécution en 1 cycle



Notes :

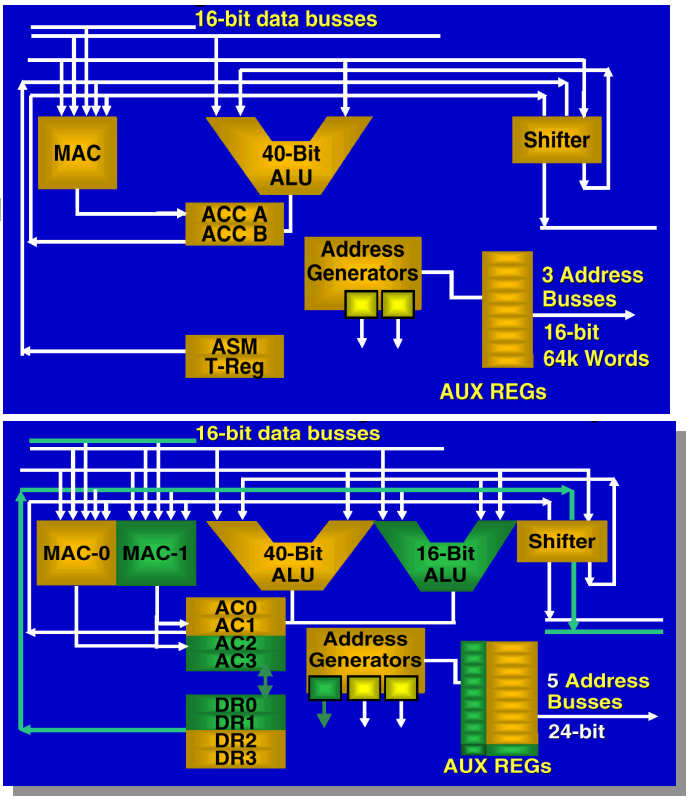
Nouvelles architectures DSP

- **DSPs conventionnels améliorés**
  - UT multiples, SIMD, coprocesseurs
  - Lucent DSP16xxx, ADI ADSP-2116x, TI C55x
- **DSPs VLIW**
  - TI C6xxx, Infineon Carmel
- **DSPs superscalaires**
  - ZSP 164xx
- **Processeurs généralistes ou hybrides**
  - GPP+SIMD, µC/DSP
  - PowerPC/AltiVec, Pentium/MMX
  - Infineon TriCore, SHx, ARM Piccolo, STx, SHx

Notes :

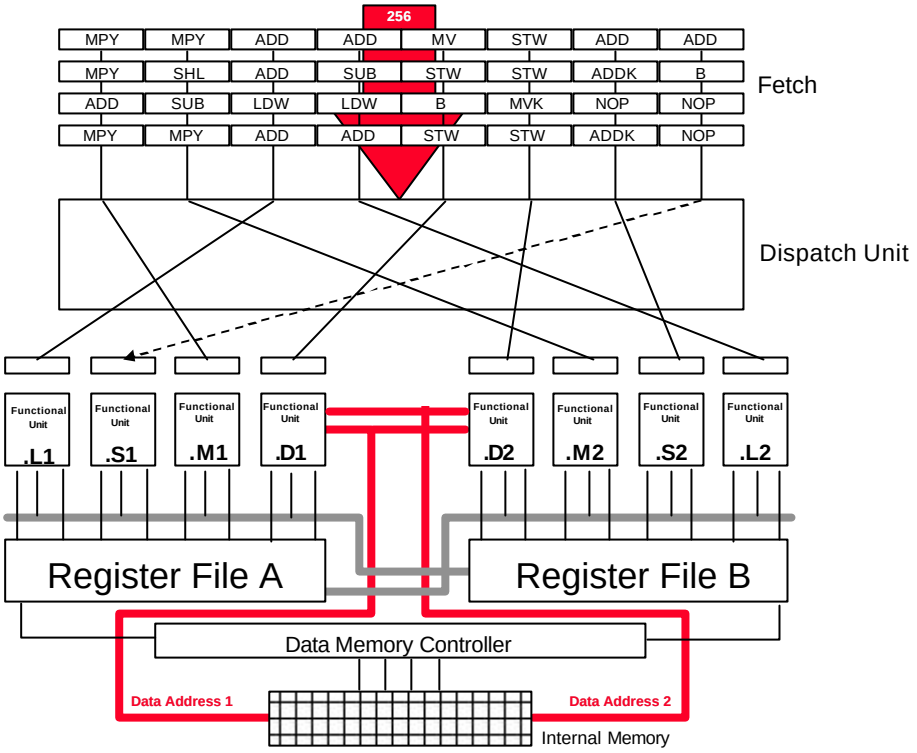
C5x Architecture

- **C54x**
  - 40-160 MIPS
  - 1000-3000 MIPS/W
  - 17x17b multiplier, 40b ALU, 40b add ACS unit
  - 60% of cellular handsets
  - $5 (C5402 100MIPS) - $75
- **C55x**
  - Dual MAC, 160 KW SRAM
  - 400 MIPS
  - 20 MIPS/mW
  - 0.05 mW/MIPS



Notes :

VLIW DSP C6x Architecture



Notes :

TI' VLIW DSP

	C62x Fixed-Point	C64x Fixed-Point	C67x Floating-Point
MHz	**150-300**	**300-600**	**100-225**
MIPS/MFLOPS	**1200-2400 MIPS**	**2400-4800 MIPS**	**600-1350 MFLOPS**
8-bit MMACS	**300-600**	**1200-2400**	**200-550**
16-bit MMACS	**300-600**	**2400-4800**	**200-550**
Broadband Communications	General	Special-Purpose instructions	General
Imaging	General	Special-Purpose instructions	General

Notes :

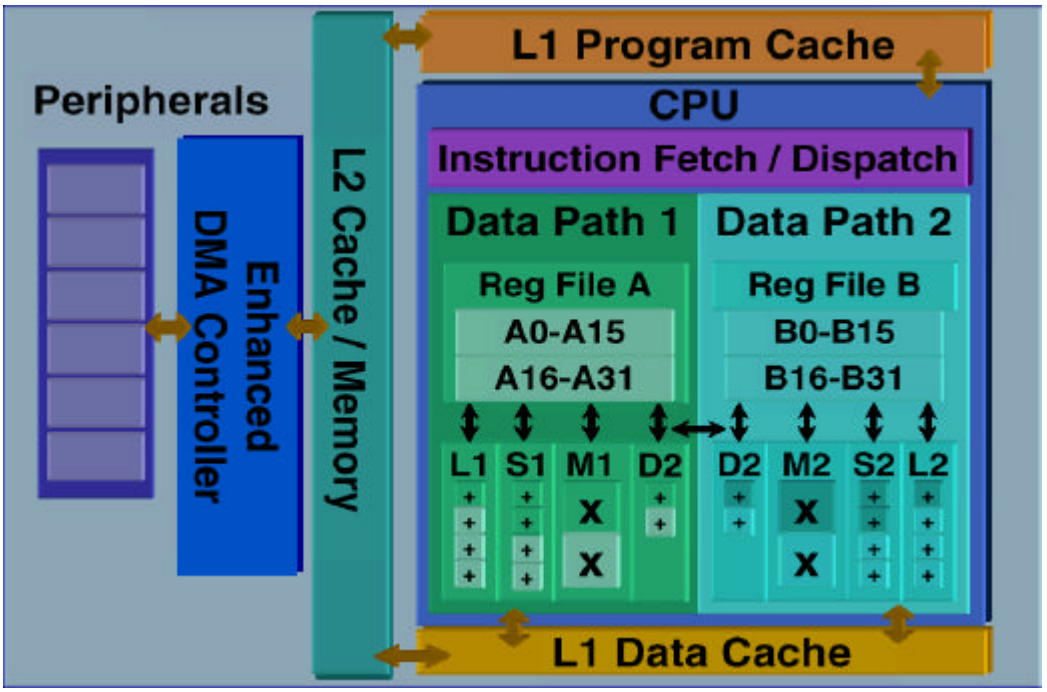
C64x

- **C64x**
  - Jusqu'à 1.1 GHz, 9 GOPS
  - Six ALUs (32-/40-Bit), une 32-Bit, deux 16-Bit, ou quatre 8-Bit opérations arithmétiques par cycle
  - Deux multiplieurs, quatre 16x16-Bit ou huit 8x8-Bit multiplications par cycle
  - Coprocesseurs VCP (Viterbi) et TCP (Turbo)

  - 'C6411: 300 MHz, $39, 1.0 V, 250mW, 2400 MIPS, 1200 MMACS

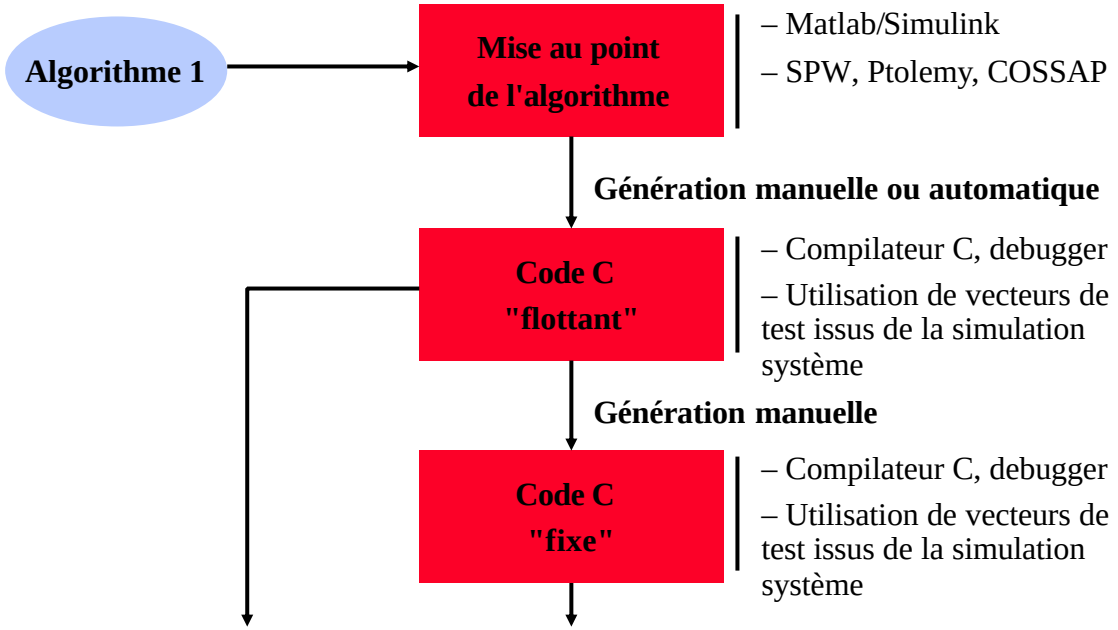
Notes :

C64x


Peripherals
Enhanced DMA Controller
L2 Cache / Memory
L1 Program Cache
CPU
Instruction Fetch / Dispatch
Data Path 1
Data Path 2
Reg File A
A0-A15
A16-A31
Reg File B
B0-B15
B16-B31
L1 S1 M1 D2 D2 M2 S2 L2
L1 Data Cache

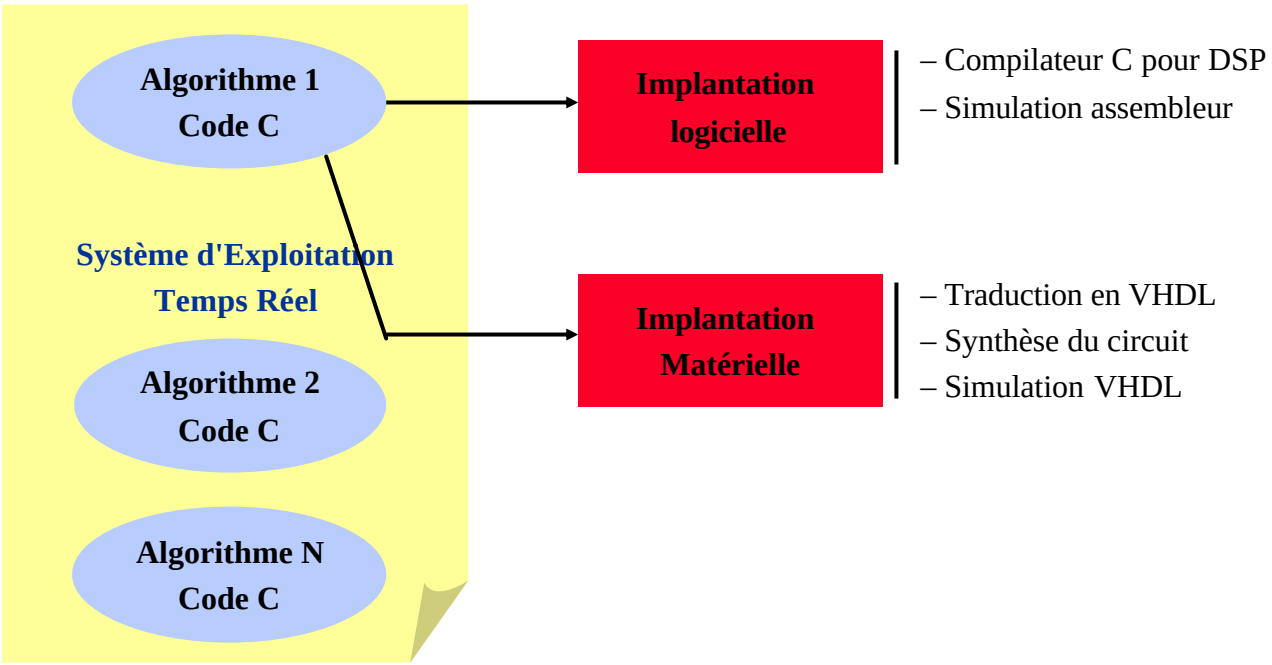
Notes :

- **Développement d'applications de TNS**



Notes :

- **Développement d'applications de TNS**



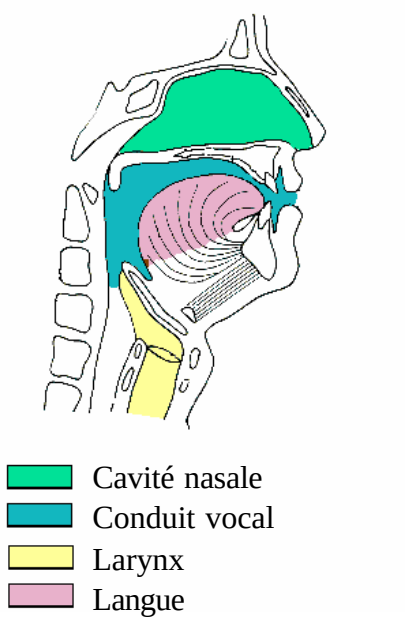
Notes :

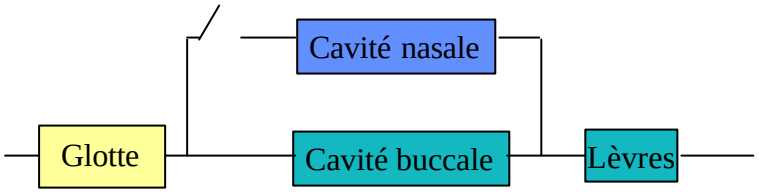
- **Signaux médicaux**
  - EEG, ECG, Images IRM, Images scanner, ...
- **Signaux sismiques**
- **Données**
  - Statistiques, Bourse, ...
- **Signal de parole**
- **Sons**
- **Images**
- **...**

Notes :

Signal de Parole

- **Processus de phonation**
  - Génération d'une énergie ventilatoire (poumons+trachée)
  - Vibration des cordes vocales
  - Réalisation d'un dispositif articulatoire (conduit vocal)



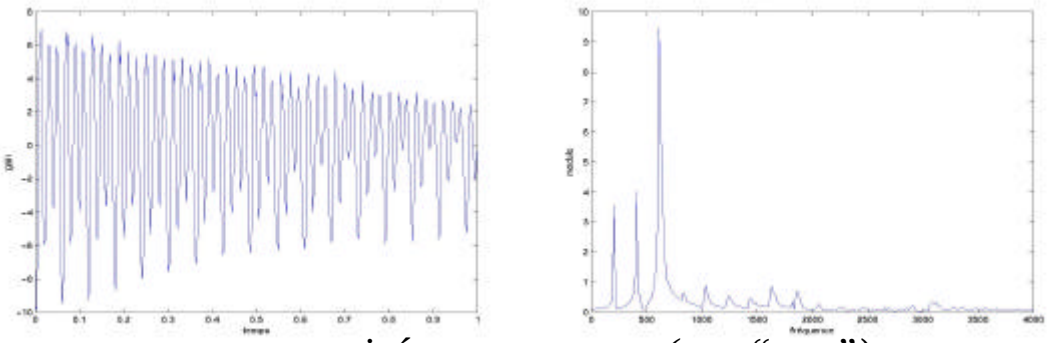


Modèle source filtre

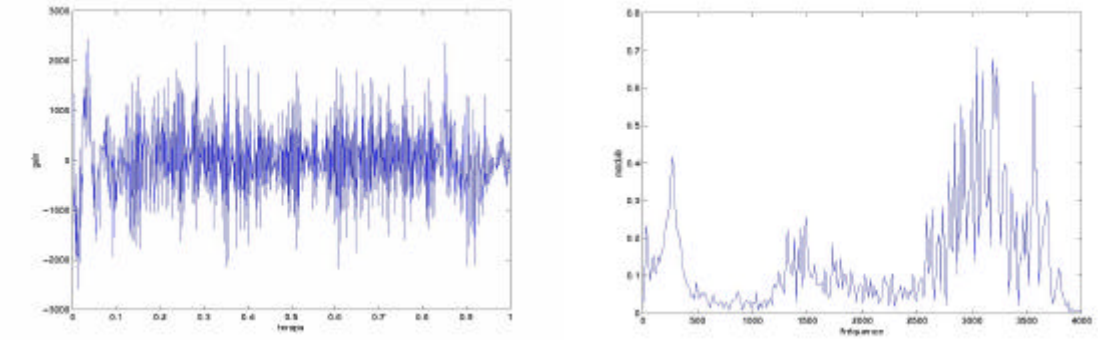
Notes :

Le signal de parole est réalisé au moyen d'un appareil phonatoire non initialement prévu pour cela. Il faut tout d'abord une énergie ventilatoire pour être l'initiateur du mouvement d'air qui crée les ondes acoustiques et qui crée par la même occasion le mouvement oscillatoire des cordes vocales ou au contraire qui les écarte pour créer du bruit; c'est le rôle de la **soufflerie**. Elle est composée des poumons qui sont le générateur d'air et du conduit trachéo-bronchique. Ce souffle passe le **larynx** (où siégent les cordes vocales) pour former l'onde glottique. Un dispositif écarte ou rapproche les cordes vocales selon qu'elles doivent vibrer ou non pour obtenir le son. Si les cordes vocales sont rapprochées, elles vibrent et donnent un son dit voisé (80% de la phonation), sinon on obtient un son dit non voisé. Ensuite, le pharynx, la langue et les parois se modifient et forment un filtre possédant une certaine fonction de transfert qui modifie l'onde glottique par convolution, ce qui donne, après rayonnement au niveau des lèvres, le signal de parole .

Signal de parole



un son voisé et son spectre (son “ eu ”)



un son non voisé et son spectre (son “ ch ”)

Notes :

On peut distinguer la nature du son (voisé ou non) par son allure (temporelle et fréquentielle). Un son voisé possède un signal pseudo périodique : son spectre contient des harmoniques (énergie présente dans les différentes fréquences). Le premier harmonique, qui reflète la fréquence de vibration des cordes vocales est appelé la **fréquence fondamentale (F0)** ou en anglais le **pitch** (de 80 à 100 Hz chez l'homme, de 175 à 300 Hz pour la femme et de 200 à 600 Hz chez l'enfant). L'évolution de la fréquence fondamentale détermine la **mélodie** de la parole [MAL,01].

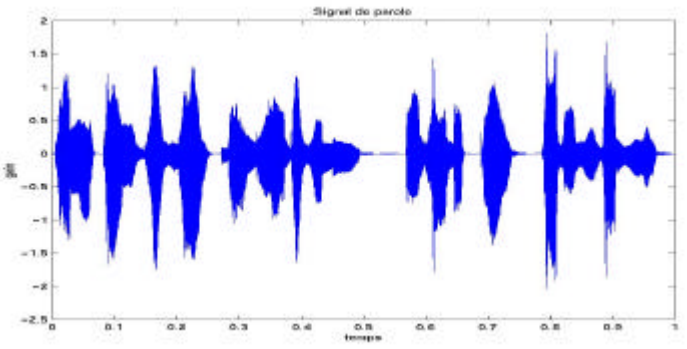
Pour un son non voisé, on obtient un signal qui ressemble à un bruit possédant beaucoup de hautes fréquences : le fait de filtrer le signal par un passe-bas pour respecter le théorème de Shannon fait qu'il devient plus difficile de distinguer des sons chuintants comme “ ch ”et “ sh ” car leurs différences se situent justement dans les hautes fréquences.

Les spectres des figures 2.3 et 2.4 sont obtenus par le passage de l'onde glottique dans la fonction de transfert du conduit vocal. On définit un **formant** comme le maximum de la fonction de transfert du conduit vocal, mais les maxima de la fonction de transfert sont excités par les composantes spectrales du signal glottique. Donc, les maxima du spectre de la figure 2.3 correspondent à peu près aux formants.

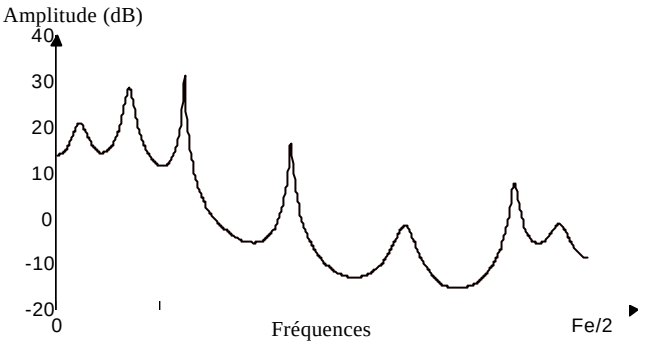
Pour cataloguer et référencer les sons afin de les étudier, on utilise une base de donnée qui possède tous les sons “ simples ” : les éléments de cette base sont appelés **phonèmes**. Un phonème se définit rigoureusement comme étant la plus petite unité susceptible de changer un mot en un autre : par exemple, [k] de “ car ” et [p] de “ par ” sont des phonèmes. Enfin, la mélodie de la parole liée à la durée et l'intensité des syllabes sont les éléments qui caractérisent la **prosodie**, qui sert entre autre à la compréhension syntaxique de la phrase (augmenter de façon significative la valeur du pitch à la fin d'une phrase interrogative).

Signal de parole

- **Signal temporel**
  - Aspects statistiques
  - Variabilité intra-individuelle
  - Variabilité inter-individuelle
  - Masquage temporel
  - Prosodie

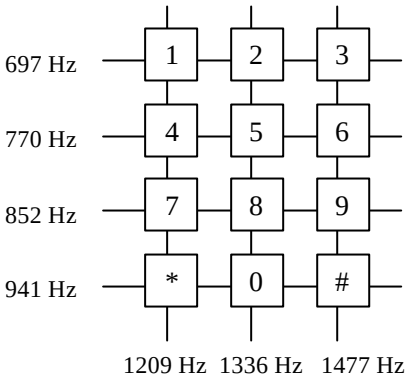


- **Signal fréquentiel**
  - Structure formantique
    - Fondamental (pitch)
    - Harmoniques
  - Masquage fréquentiel



Notes :

- **Télécommunications : détection de tonalité**



Fréquences de tonalité du clavier numérique

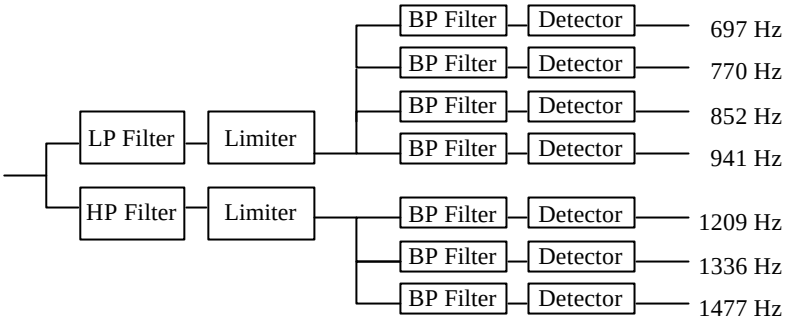
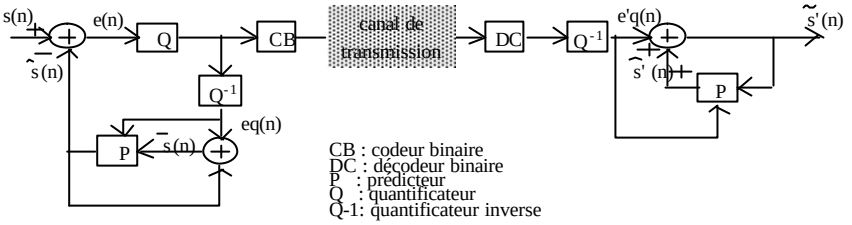


Schéma de détection de tonalité

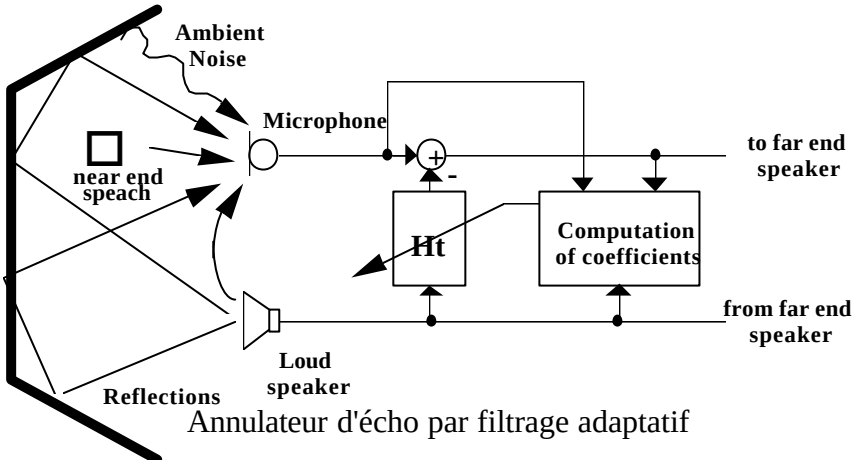
Notes :

- **Voie MIC (PCM)**
  - parole échantillonnée à 8 kHz en bande limitée à 300-3400Hz sur 8 bits par une loi logarithmique (Rec. G711 du CCITT)
  - débit normalisé de 64 kbit/s par voie numérique (MIC)
- **Codage de la parole**
  - le codage permet : soit d'augmenter le nombre de signaux par voie (multiplexage temporel), soit d'élargir la bande codée (7kHz pour audio et visioconférence)
  - MICDA : 32 kbit/s sans dégradation audible



Notes :

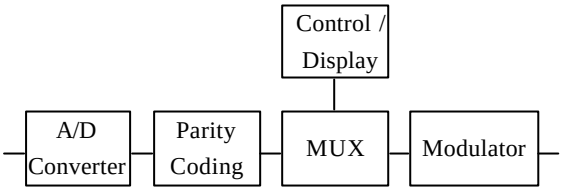
- **Annulation d'écho**
  - Réseau téléphonique utilisant les satellites géostationnaires (540ms)
  - Téléphone main libre en voiture (écho + bruit)
  - Téléconfèrence
    - réponse impulsionnelle de la salle
    - Effets : écho, Larsen, réverbération
    - problème de déconvolution



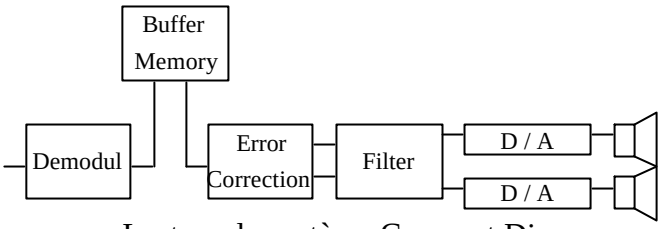
Annulateur d'écho par filtrage adaptatif

Notes :

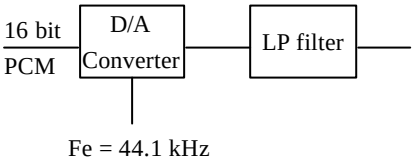
- **Compact Disc Audio**
  - Echantillonnage à 44,1 kHz sur 16 bits des deux voies : 1,41 Mbit/s
  - Information + correction d'erreurs, contrôle et affichage : 4,32 Mbit/s
  - 90 dB de rapport Signal à Bruit et de séparation stéréo (contre 60 et 30 dB)



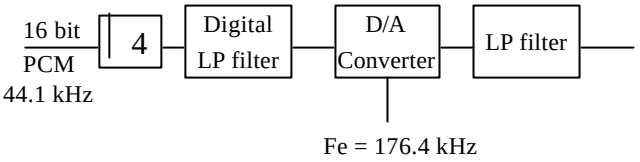
Encodeur du système Compact Disc



Lecture du système Compact Disc



Conversion N/A classique



En pratique : Suréchantillonnage

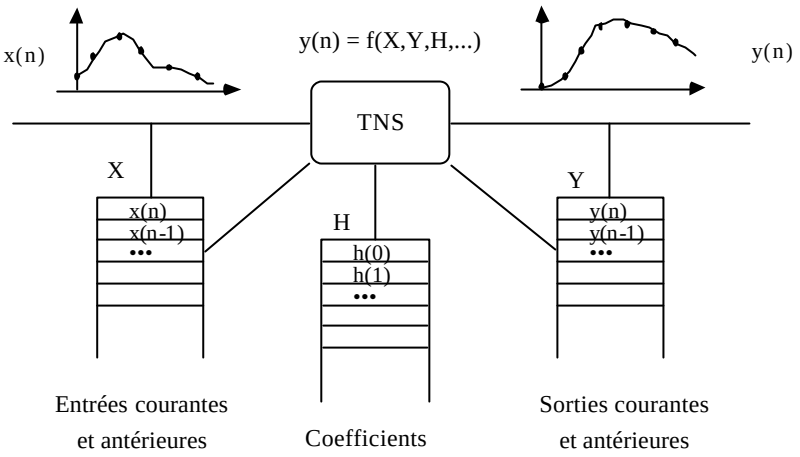
Notes :

• Traitement par ligne

– Flot de données ininterrompu

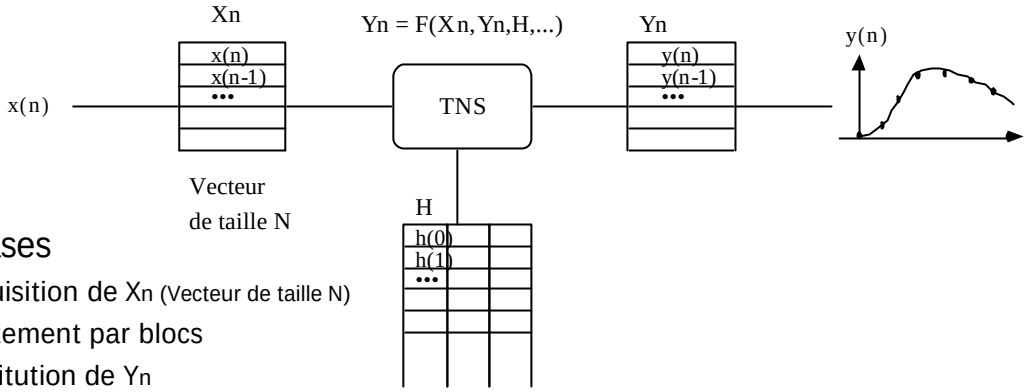
– En entrée : flot d'échantillons sur N bit à N.Fe bit/s par entrée

– En sortie : flot d'échantillons sur N' bit à N'.F'e bit/s par sortie



Notes :

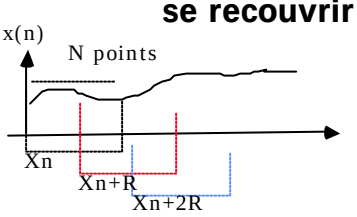
• Traitement par blocs



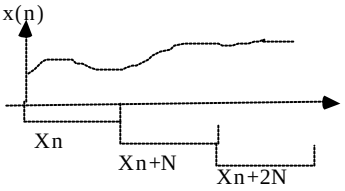
– 3 phases
  Acquisition de X_n (Vecteur de taille N)
  Traitement par blocs
  Restitution de Y_n

– Les phases d'acquisition d'un traitement au suivant peuvent soit

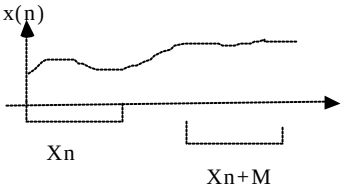
se recouvrir



être jointives



être disjointes



Notes :

Cadence des calculs

- **Nv : nombre de voies à traiter en parallèle**
- **Nop : quantité d'opérations élémentaires nécessaires au TNS**
- **Te : périodicité du calcul**
- **On en déduit la puissance de calcul de la machine :**
  - Pc = Nv.Nop / Te (en MIPS, MOPS ou MFlOps)
  - Pc = 2.B.Nv.Nop ou B est la bande du signal

Notes :

Exemple : Convolution

Traitement ligne

$$y(n) = \sum_{k=0}^{N-1} h(k).x(n-k)$$

- Complexité pour un point y(n) : ??
- Architecture réalisant une multiplication accumulation en 1 cycle

Traitement blocs avec recouvrement de N-R échantillons

$$Y_n = H \cdot X_n$$

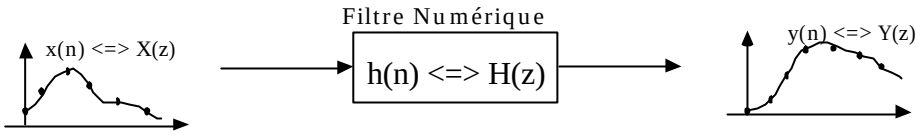
$$\begin{bmatrix} y(n) \\ y(n-1) \\ \cdots \\ y(n-N+1) \end{bmatrix} = \begin{bmatrix} h(0) & h(1) & \cdots & h(N-1) \\ h(N-1) & h(0) & h(1) & \cdot \\ \cdots & & & \cdot \\ h(1) & \cdots & h(N-1) & h(0) \end{bmatrix} \cdot \begin{bmatrix} x(n) \\ x(n-1) \\ \cdots \\ x(n-N+1) \end{bmatrix}$$

- Complexité pour un vecteur Yn :
- Même Architecture

Notes :

Définition

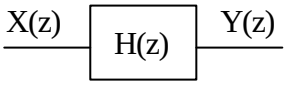
– Système Linéaire Discret (SLD) modifiant la représentation temporelle et fréquentielle de signaux



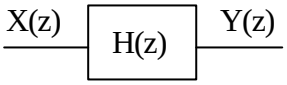
1. Introduction

Un filtre numérique peut être représenté par :

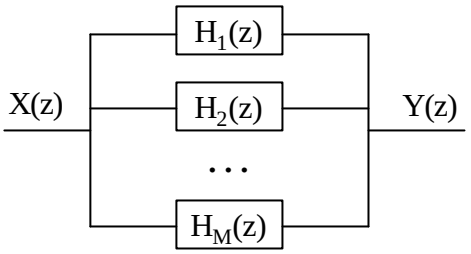
– une fonction de transfert en z : H(z) = Y(z) / X(z)



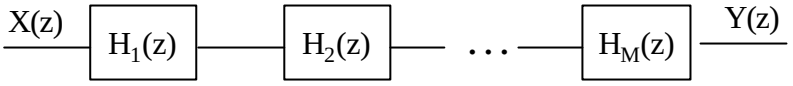
Notes :



a) Forme directe



b) Forme parallèle

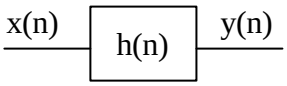


c) Forme cascade

– une réponse impulsionnelle (finie ou infinie) (RIF ou RII)

$$y(nT) = \sum_{k=0}^{\infty} x(kT)h[(n-k)T]$$

$$y(n) = \sum_{k=0}^{\infty} x(k)h(n-k)$$



Notes :

si x(n) = δ(n) alors y(n) = h(n)

– une équation aux différences (récursive ou non récursive)

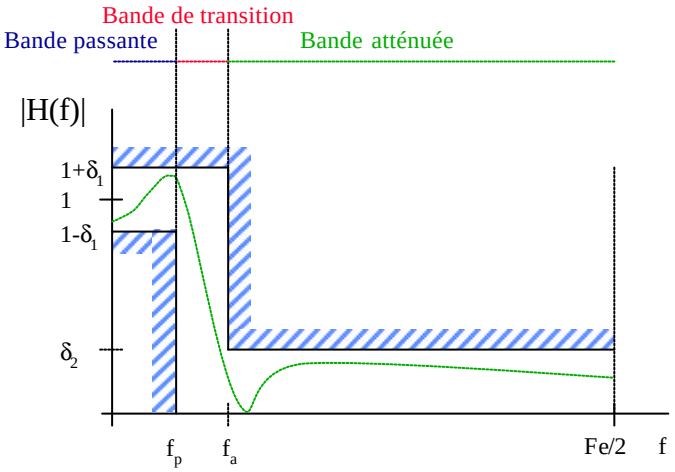
$$y(n) = \sum_{i=0}^{M} b_i.x(n-i) - \sum_{i=1}^{N} a_i.y(n-i)$$

Notes :

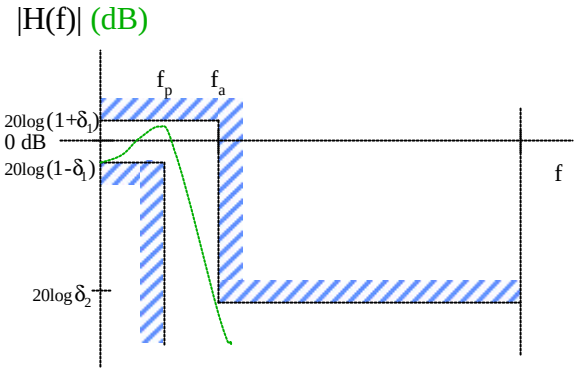
2. Spécification d'un filtre numérique

– Gabarit fréquentiel

Passe-Bas (ou Passe-Haut) défini par sa sélectivité, son ondulation en BP et son atténuation en BA



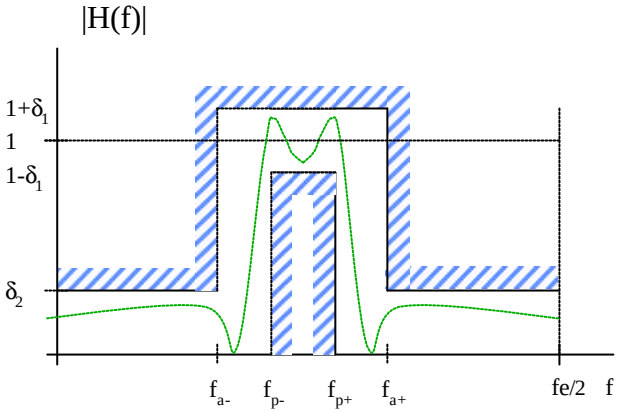
a) Gabarit fréquentiel linéaire



b) Gabarit fréquentiel en dB

Notes :

Passe-Bande (ou Réjecteur-de-Bande) défini par sa fréquence centrale, sa sélectivité, son ondulation en BP et son atténuation en BA



Notes :

3 Classification des filtres numériques

Un filtre numérique peut être classé selon :

– la durée de sa réponse impulsionnelle

finie : les filtres **RIF** ont leur réponse impulsionnelle à support fini
i.e. $h(n) = 0$ pour $n<0$ et $n>N$

infinie : les filtres **RII** ont leur réponse impulsionnelle à support infini
i.e. $h(n) \neq 0 \ \forall n$

– le type de représentation temporelles

récursifs : la sortie y(n) dépend de l'entrée courante, des entrées précédentes et des sorties précédentes

non récursifs : la sortie y(n) ne dépend que de l'entrée courante et des entrées précédentes

Notes :

3.1 Filtres numériques non récursifs

(ou transversaux)

$$y(n) = \sum_{i=0}^{M} b_i \, x(n-i)$$

$$Y(z) = H(z)\, X(z)$$

$$\Rightarrow H(z) = \sum_{i=0}^{M} b_i \, z^{-i} = \sum_{n=0}^{M} h(n)\, z^{-n}$$

$$\Rightarrow h(n) = \sum_{i=0}^{M} b_i \, d(n-i)$$

Les coefficients b_n du filtre sont les valeurs de la RI ($h(n) = b_n$). Ceci montre qu'un filtre non récursif est à **Réponse Impulsionnelle Finie (RIF)**.

M est appelée la longueur du filtre.

Notes :

• Principales propriétés

– Les RIF sont toujours stables (pas de pôles)
– Les RIF peuvent avoir une caractéristique de phase linéaire
  • Retard constant en fréquence (temps de propagation de groupe)
  • Pas de distorsion harmonique
  • Symétrie de la RI
– A sélectivité équivalente, ils sont toujours plus coûteux (en temps de calcul) que leur équivalent RII

Notes :

3.2 Filtres numériques récursifs

$$y(n) = \sum_{i=0}^{M} b_i\, x(n-i) - \sum_{i=1}^{N} a_i\, y(n-i)$$

$$\Rightarrow H(z) = \frac{\sum_{i=0}^{M} b_i\, z^{-i}}{1 + \sum_{i=1}^{N} a_i\, z^{-i}} = \frac{N(z)}{D(z)}$$

En pratique on a N=M, N est appelée l'ordre du filtre.

Notes :

– Si N(z) n'est pas divisible par D(z) (cas général), on a un nombre infini de termes dans la division polynomiale.

$$H(z) = \sum_{i=0}^{\infty} c_i \, z^{-i} = \sum_{n=0}^{\infty} h(n) \, z^{-n}$$

Les coefficients c_n sont les valeurs de la RI ($h(n) = c_n$). Ceci montre qu'un filtre récursif est, dans le cas général, à **Réponse Impulsionnelle Infinie (RII)**.

– Si N(z) est divisible par D(z) (cas particulier), on a un nombre fini de termes dans la division polynomiale. Dans ce cas, le filtre est RIF.

- **Exemple : filtre moyenneur**

Notes :

– Si N(z)=1 : filtre tout-pôle
– Si D(z)=1 : filtre RIF

$$H(z) = \frac{\sum_{i=0}^{M} b_i \, z^{-i}}{1 + \sum_{i=1}^{N} a_i \, z^{-i}} = \frac{N(z)}{D(z)}$$

• Principales propriétés

– Les RII peuvent être instables : structure à base de pôles et de zéros

$$H(z) = b_0 \, z^{N-M} \frac{\prod_{i=1}^{M} (z - z_i)}{\prod_{i=1}^{N} (z - p_i)}$$

– Bande de transition faible
– Synthèse par réutilisation des méthodes analogiques
– Instabilité numérique due au rebouclage : forme cascade plus stable

Notes :

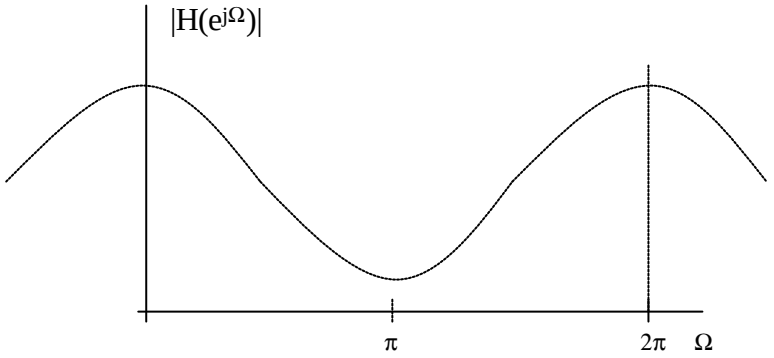
4. Analyse fréquentielle

L'analyse fréquentielle est l'étude du module, de la phase et du temps de propagation de groupe du filtre H.

$$H(e^{j\Omega}) = H(z)\Big/_{z=e^{j\Omega}}$$

Ω est la pulsation relative : Ω = ωT = 2πfT

La fonction de transfert en fréquence $H(e^{j\Omega})$ est périodique de période 2π.

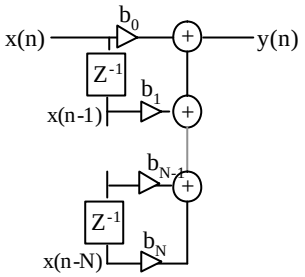


Notes :

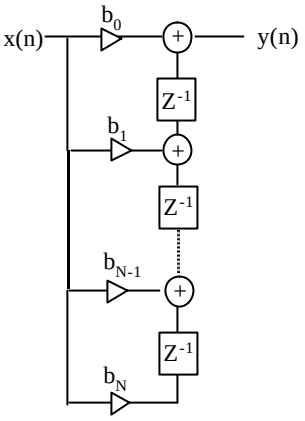
5. Structures de réalisation

– Filtres RIF

$$y(n) = \sum_{i=0}^{N} b_i \, x(n-i)$$



a) Structure directe



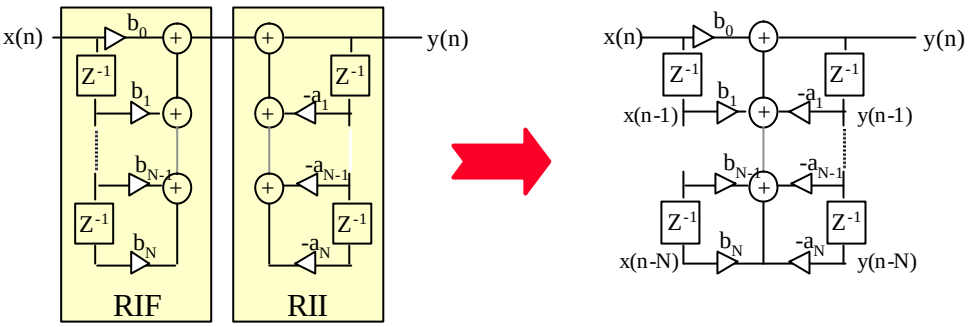
b) Structure transposée

Notes :

– Filtres RII

$$y(n) = \sum_{i=0}^{N} b_i\, x(n-i) - \sum_{i=1}^{N} a_i\, y(n-i)$$

$$\Rightarrow H(z) = \frac{N(z)}{D(z)} = N(z) \times \frac{1}{D(z)} = \sum_{i=0}^{N} b_i\, z^{-i} \times \frac{1}{1 + \sum_{i=1}^{N} a_i\, z^{-i}}$$



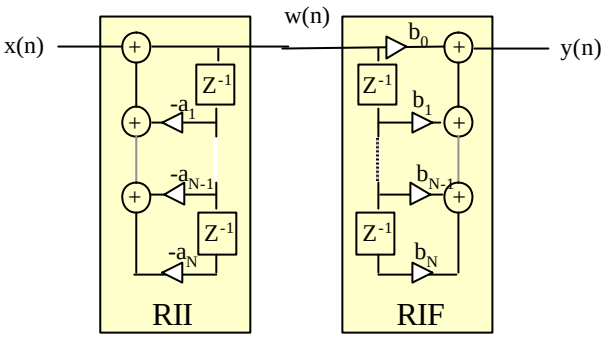
a) Structure directe

Notes :

– Filtres RII

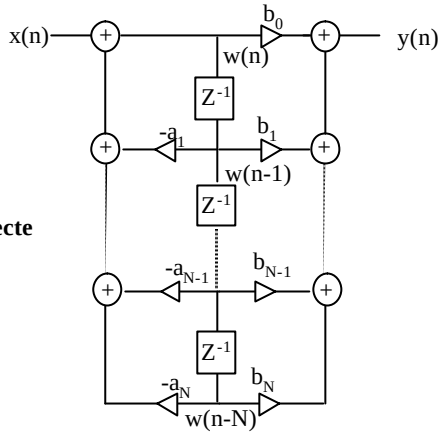
$$H(z)=\frac{1}{D(z)}\times N(z)=\frac{1}{1+\sum_{i=1}^{N} a_i\, z^{-i}}\times\sum_{i=0}^{N} b_i\, z^{-i}$$

$$\begin{cases} W(z)=\dfrac{1}{D(z)}.X(z) \\ Y(z)=N(z).W(z) \end{cases} \qquad \begin{cases} w(n)=x(n)-\sum_{i=1}^{N} a_i\, w(n-i) \\ y(n)=\sum_{i=0}^{N} b_i\, w(n-i) \end{cases}$$

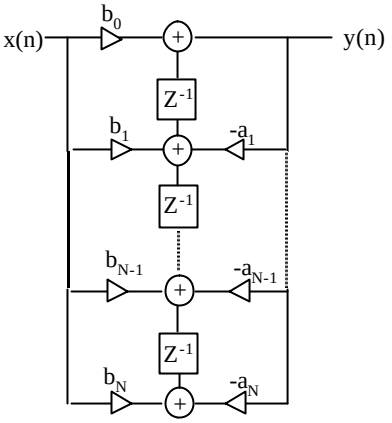


Notes :

– Filtres RII



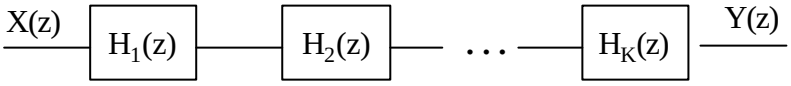
a) Structure canonique directe



b) Structure canonique transposée

– Forme cascade de filtres du second ordre

$$H(z)=\prod_{i=1}^{\frac{N+1}{2}} Hi(z)=\prod_{i=1}^{\frac{N+1}{2}} \frac{b_{i,0}+b_{i,1}z^{-1}+b_{i,2}z^{-2}}{1+a_{i,1}z^{-1}+a_{i,2}z^{-2}}$$



Notes :

- **TFSD : Transformée de Fourier d'un Signal Discret**

$$X(e^{jwT}) = \sum_{n=-\infty}^{\infty} x(nT) e^{-jnwT}$$

Non périodique

$$\begin{cases} X(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} x(nT)\, e^{-jn\Omega} \\ x(nT) = \frac{1}{2p} \int_{-p}^{p} X(e^{j\Omega}) e^{jn\Omega} d\Omega \end{cases}$$

- **Propriétés**
  - Linéarité
  - Décalage en temps/fréquence
  - Produit de convolution en temps/fréquence
  - Théorème de Parseval
  - Transformées de fonctions réelles

Notes :

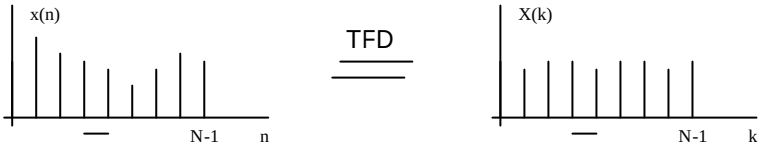
• TFD

– En pratique, on prend seulement un nombre fini d'échantillons de x(nT). On ne peut donc obtenir qu'un nombre fini d'échantillons fréquentiels de $X(e^{j\Omega})$.

$$\begin{cases} X(k) = \sum_{n=0}^{N-1} x(n) e^{-2jp\frac{kn}{N}} & k = 0..N-1 \\ x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{+2jp\frac{kn}{N}} & n = 0..N-1 \end{cases}$$

x(n) est considéré comme périodique de période N, x(n) = x(n+qN)

X(k) est donc également périodique de période N, X(k) = X(k+qN)



Pas temporel : T=1/Fe

Pas fréquentiel : 1/NT=Fe/N

Notes :

- **Propriétés**
  - Linéarité
  - Décalage en temps/fréquence
  - Produit de convolution en temps/fréquence
  - Théorème de Parseval
  - Transformées de fonctions réelles

Notes :

III Transformées en TNS

2 Transformée de Fourier Discrète

- Définition

TFD
X(k) $\Leftrightarrow$ x(n)

$$\begin{cases} X(k) = \sum_{n=0}^{N-1} x(n) e^{-2jp\frac{kn}{N}} & k = 0..N-1 \\ x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{+2jp\frac{kn}{N}} & n = 0..N-1 \end{cases}$$

x(n) et X(k) sont, dans le cas général, des nombres complexes.

- Forme Matricielle

$$\begin{bmatrix} X(0) \\ X(1) \\ . \\ . \\ X(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & . & . & 1 \\ 1 & W_N^1 & W_N^2 & . & W_N^{N-1} \\ . & W_N^2 & W_N^4 & . & W_N^{2(N-1)} \\ . & . & . & . & . \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & . & W_N^{(N-1)^2} \end{bmatrix} \times \begin{bmatrix} x(0) \\ x(1) \\ . \\ . \\ x(N-1) \end{bmatrix}$$

$\underline{X} = \underline{W} . \underline{x}$ (2)

$$W_N = e^{\frac{-2jp}{N}}$$

$$W_N^{nk} = e^{\frac{-2jp}{N}kn}$$

Notes :

Propriétés des $W_N^n = e^{-2j\pi \frac{n}{N}}$

$$W_N^{k(N-n)} = (W_N^{kn})^* \quad (3.1)$$

$$W_N^{kn} = W_N^{k(n+N)} = W_N^{(k+N)n} \quad (3.2) \quad \text{Périodicité}$$

$$W_N^{n+N/2} = -W_N^n \quad (3.3) \quad \text{Symétrie}$$

$$W_N^{2kn} = W_{N/2}^{kn} \quad (3.4)$$

Complexité de calcul

La TFD revient à calculer un produit matrice-vecteur où chaque élément est de type complexe. La complexité de calcul de la TFD est de N^2 multiplications, et de N(N-1) additions sur des nombres complexes. Ceci revient à une complexité de $4N^2$ multiplications réelles et N(4N-2) additions réelles. Cet algorithme se comporte donc en $O(N^2)$, mais ne possède pas de problèmes d'adressage car les x(n) et les Wi sont rangés dans l'ordre en mémoire.

– En 1965, Cooley et Tuckey [COOLEY 65] ont publié un algorithme applicable quand N est le produit de 2 ou plusieurs entiers dont la complexité est en $O(N\log_2 N)$

Notes :

• TFR (FFT) partagée dans le temps (DIT)

$$X(k)=\sum_{npair} x(n).W_N^{nk}+\sum_{nimpair} x(n).W_N^{nk}$$

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

En exploitant la propriété 3.4, on obtient :

$$X(k)=\sum_{n=0}^{N/2-1} x(2n).W_{N/2}^{nk}+W_N^k.\sum_{n=0}^{N/2-1} x(2n+1).W_{N/2}^{nk} \qquad (4)$$

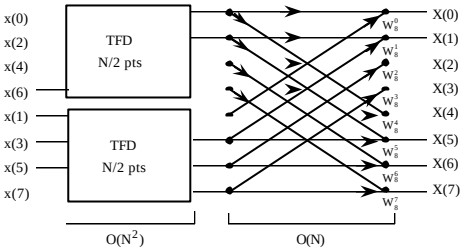
$$X(k)=G(k)+W_N^k.H(k) \qquad k=0,1,\ldots,N-1$$

où G(k): TFD sur N/2 points d'indices pairs,
H(k): TFD sur N/2 points d'indices impairs.

$$X(k+\frac{N}{2})=\sum_{n=0}^{N/2-1} x(2n).W_{N/2}^{n(k+N/2)}+W_N^{k+N/2}.\sum_{n=0}^{N/2-1} x(2n+1).W_{N/2}^{n(k+N/2)}$$

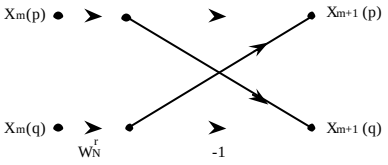
$$X(k+\frac{N}{2})=G(k)-W_N^k.H(k)$$

Notes :



2 TFD de taille N/2 **N/2 papillons**

Papillon DIT



$$X_{m+1}(p) = X_m(p) + W_N^r X_m(q)$$
$$X_{m+1}(q) = X_m(p) - W_N^r X_m(q) \quad (6)$$

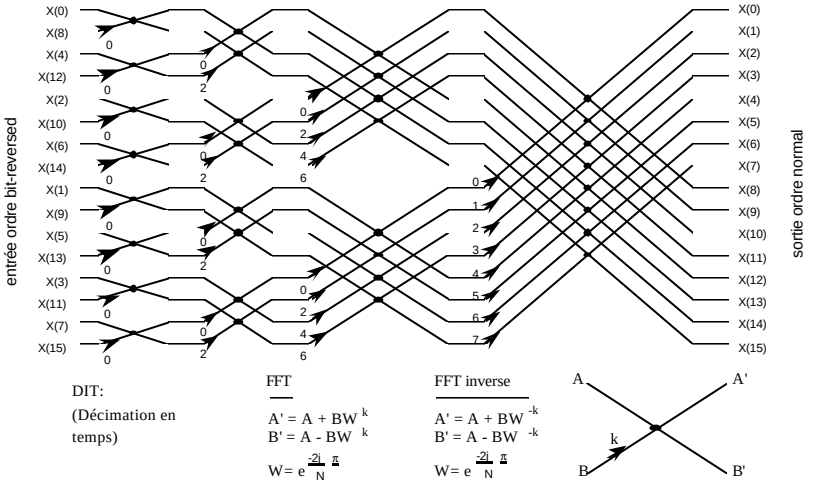
Complexité d'un papillon : 1 multiplication complexe, 2 additions/soustractions complexes

Notes :

$\frac{N}{2}\log_2 N$ multiplications de nombres complexes,
$N \log_2 N$ additions/soustractions de nombres complexes, ou,

$2\,N \log_2 N$ multiplications de nombre réels,
$3\,N \log_2 N$ additions/soustractions de nombre réels.

FFT DIT RADIX-2 en place sur 16 points



DIT:
(Décimation en temps)

FFT

$A' = A + BW^{k}$
$B' = A - BW^{k}$
$W = e^{\frac{-2j\pi}{N}}$

FFT inverse

$A' = A + BW^{-k}$
$B' = A - BW^{-k}$
$W = e^{\frac{-2j\pi}{N}}$

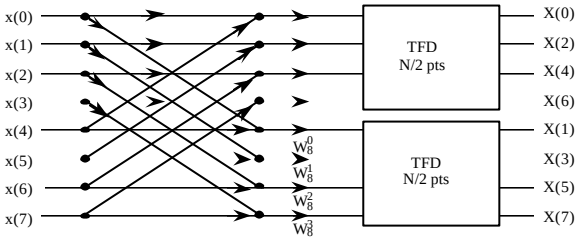
Notes :

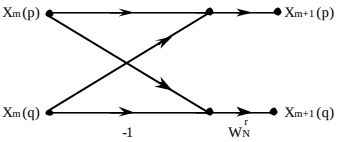
• TFR (FFT) partagée dans les fréquences (DIF)

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

$$X(k) = \sum_{n=0}^{N/2-1} x(n).W_N^{nk} + W_N^{k.N/2}.\sum_{n=0}^{N/2-1} x(n+\frac{N}{2}).W_N^{nk} \qquad (7)$$

$$X(k) = \sum_{n=0}^{N/2-1} [\, x(n) + (-1)^k.x(n+\frac{N}{2})]\, W_N^{nk}$$





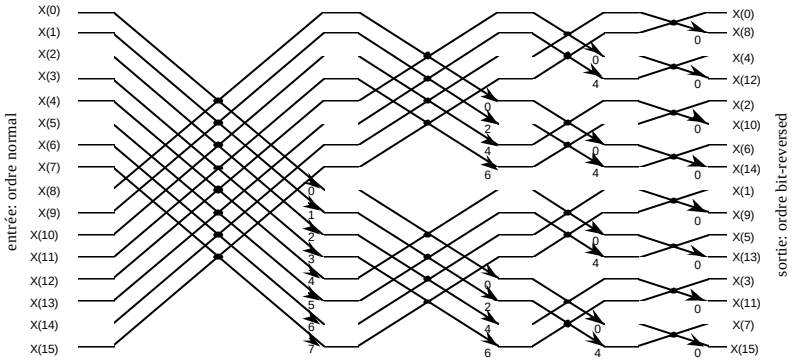
$$X_{m+1}(p) = X_m(p) + X_m(q)$$

$$X_{m+1}(q) = [X_m(p) - X_m(q)]\, W_N^r \qquad (8)$$

Notes :

••• TFR DIF

FFT DIF RADIX-2 en place sur 16 points



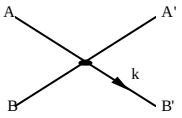
DIF:
(Décimation en fréquence)

FFT

$A' = A + B$
$B' = (A-B)\, W^{k}$

FFT inverse

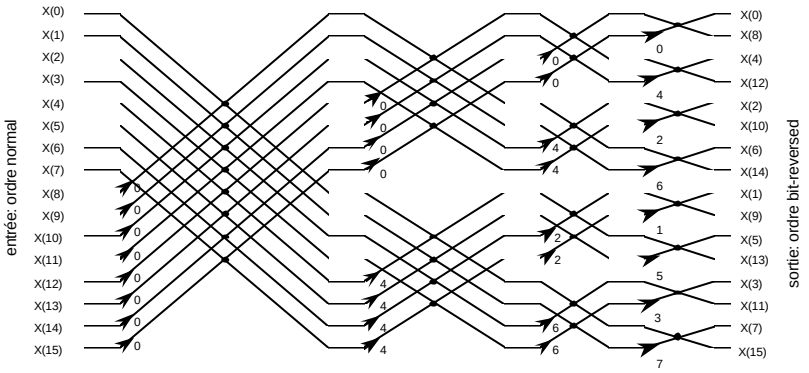
$A' = A + B$
$B' = (A-B)\, W^{-k}$



Notes :

Transformée de Fourier Rapide

FFT DIT RADIX-2 en place sur 16 points



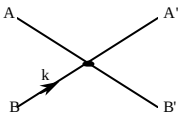
DIT:
(Décimation en temps)

FFT

$A' = A + BW^{k}$
$B' = A - BW^{k}$

FFT inverse

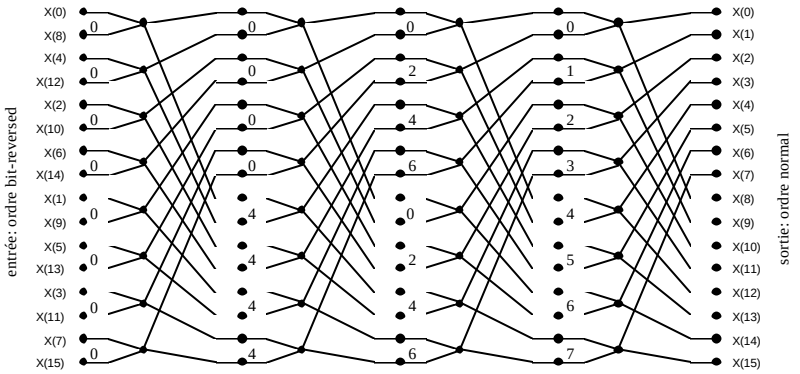
$A' = A + BW^{-k}$
$B' = A - BW^{-k}$



Notes :

Transformée de Fourier Rapide

FFT GEOMETRIE CONSTANTE sur 16 points



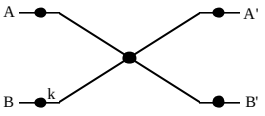
Géométrie Constante

FFT

$A' = A + B \cdot W^{k}$
$B' = A - B \cdot W^{k}$

FFT inverse

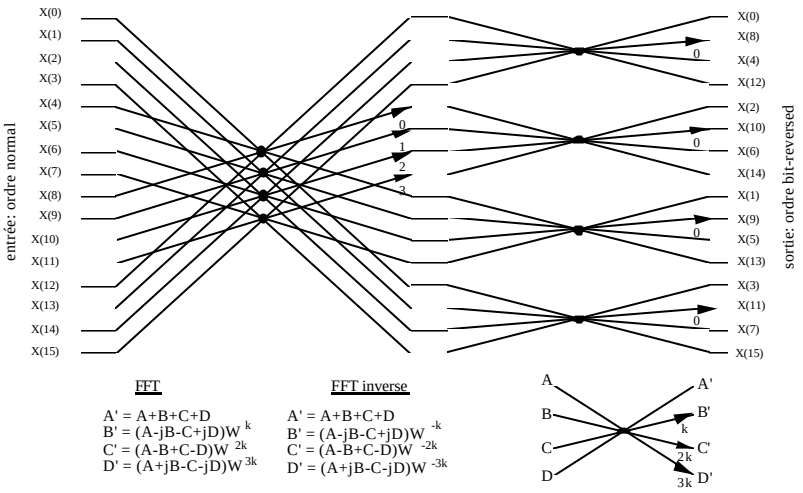
$A' = A + B \cdot W^{-k}$
$B' = A - B \cdot W^{-k}$



Notes :

Transformée de Fourier Rapide

FFT DIF RADIX-4 en place sur 16 points



FFT

A' = A+B+C+D
B' = (A-jB-C+jD)W ^{k}
C' = (A-B+C-D)W ^{2k}
D' = (A+jB-C-jD)W ^{3k}

FFT inverse

A' = A+B+C+D
B' = (A-jB-C+jD)W $^{-k}$
C' = (A-B+C-D)W $^{-2k}$
D' = (A+jB-C-jD)W $^{-3k}$

Notes :

III.4 Convolution et corrélation

1 Définitions

- **Corrélation**
  - Soit x_1 et x_2, 2 signaux de durée finie [0 … N-1], la corrélation est :

$$y(n) = \sum_{i=0}^{N-1} x_1(i)\, x_2(i+n)$$

- **Convolution linéaire**
  - Soit x et h, 2 signaux de durée finie respectivement N et M, la convolution est définie par :

$$y(n) = (x * h)(n)$$

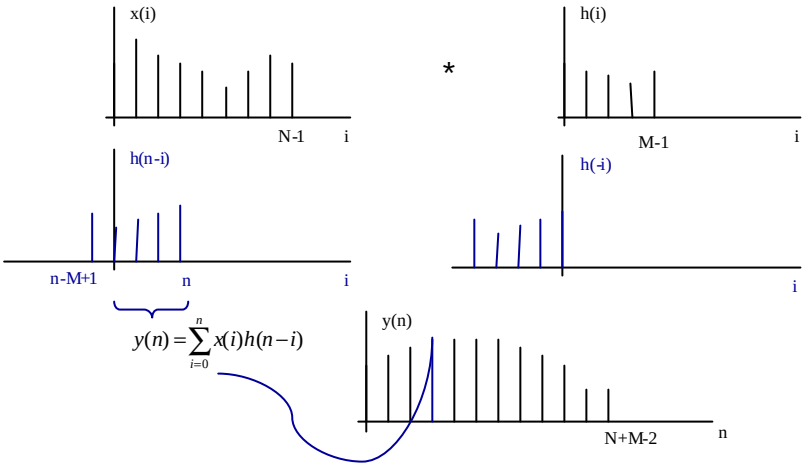
$$y(n) = \sum_{i=0}^{\infty} x(i)\, h(n-i) = \sum_{i=0}^{\infty} h(i)\, x(n-i)$$

Le signal y(n) est de durée [0 … N+M-2]

Notes :

- **Exemple de convolution** $y(n) = \sum_{i=0}^{\infty} x(i)\, h(n-i)$

N > M



Notes :

- **Propriétés**
  - Y(z) = H(z) X(z) (TZ)
  - Y(k) ≠ H(k) X(k) (TFD)
- **Vue matricielle de la convolution**

$$\begin{bmatrix} y(0) \\ y(1) \\ . \\ . \\ y(N+M-2) \end{bmatrix} = \begin{bmatrix} h(0) & 0 & . & . & 0 \\ . & h(0) & . & . & 0 \\ h(M-1) & . & . & . & 0 \\ 0 & . & . & . & h(0) \\ 0 & 0 & . & . & h(M-1) \end{bmatrix} \times \begin{bmatrix} x(0) \\ y(1) \\ . \\ . \\ x(N-1) \end{bmatrix}$$

➔ **O(N²)**

Notes :

• Convolution circulaire

– Soit x et h, 2 signaux périodiques de période N, la convolution circulaire est définie par :

$$y(n) = \sum_{i=0}^{N-1} x(i)\, h(n-i)$$

$$y(n) = x(n) \circledast h(n)$$

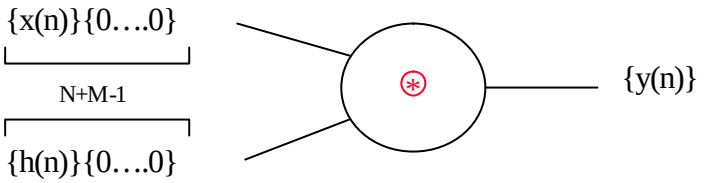
Le signal y(n) est de période N

h(n-i) est évalué modulo N

TFD : Y(k) = H(k).X(k)

Notes :

- **Convolution circulaire**
  - On passe de la convolution circulaire à la convolution linéaire en remplissant de zéros chaque séquence jusqu'à M+N-1

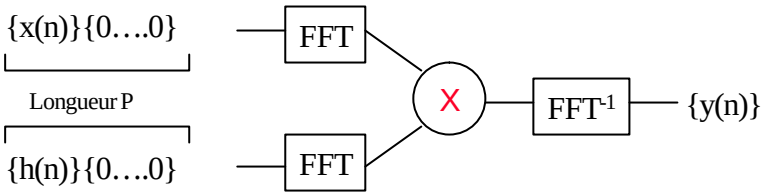


Notes :

• Convolution rapide

– Passer dans le domaine de Fourier par une TFD : la convolution se transforme en produit

– Utiliser la FFT sur P points pour accélérer les calculs



– Compléter les suites x(n) et h(n) par des zéros jusqu'à P > N+M-2, avec $P = 2^p$.

➔ $O(N\log_2 N)$

Notes :

– Problème : h(n) et x(n) doivent être de durée finie

– Application : FIR rapide

- h(n) de durée M : H(k) peut être calculé une fois pour toute
- x(n) de durée infinie

• Convolution sectionnée

– x(n), de durée infinie, est découpé en blocs x_k de taille M

$$x_k(n) = \begin{cases} x(n) & pour\ kM \le n < (k+1)M \\ O & ailleurs \end{cases}$$

$$y(n) = h(n) * \sum_{k=-\infty}^{\infty} x_k(n)$$

$$y(n) = \sum_{k=-\infty}^{\infty} y_k(n)$$

Notes :

- **Méthode OLA (Overlap Add)**
  - Blocs x_k de taille M
  - Addition des recouvrements entre les y_k

- **Méthode OLS (Overlap Save)**
  - Blocs x_k de taille N+M avec recouvrement
  - Troncature des y_k sur M points, addition entre les y_k

Notes :

IV Quantification

Introduction : pourquoi la quantification ?

1. Format de codage

Entiers, Virgule fixe

2. Modèle de quantification

Caractéristiques de quantification
Modèle de bruit, Caractéristiques de dépassement

3. Bruit de conversion

Filtrage d'un bruit

4. Limitation des chemins de données

5. Effets en TNS

Filtrage RIF, RII, cycles limites, quantification des coefficients

Notes :

- **Choix d'un format de codage**
  - Format des données: entiers, relatif, réel,...
  - Précision: Acquisition, Calculs
  - Dynamique: Acquisition, Calculs
  - Coût des opérateurs matériels
- **Codage des entiers positifs** $D = \sum_{i=0}^{N-1} d_i \, 2^i$
- **Codage des entiers relatifs**
  - Codage signe valeur absolue $D = -1^{d_{N-1}} \sum_{i=0}^{N-2} d_i \, 2^i$

    Un bit de la représentation est affecté au signe

    exemple: D = 65 = ***0****1000001; D = -65 =* ***1****1000001*

    Additionneur différent pour les additions de nombres, suivant leurs signes

Notes :

• Codage des entiers relatifs

– Codage en complément à la base

- Codage cyclique modulo 2^N

$$D = -d_{N-1}2^{N-1} + \sum_{i=0}^{N-2} d_i 2^i$$

- On obtient un nombre négatif en complémentant sa valeur absolue puis en additionnant 1.
  Exemple : D = 65 = 01000001; D = -65 = $\overline{01000001}$ + 1 = 10111111
  ou D = -65 => -128 + 65 = -10000000 + 01000001 = 10111111
- Si la somme de plusieurs termes appartient au domaine de codage, les opérations sont correctes, même si un résultat intermédiaire sort du domaine de codage.
  Exemple : (((+2+7)-6)-3) = 0
  en codage signé: = ((-1-6)-3) = (-7-3) = -2
  en complément à 2: = ((-7-6)-3) = (+3-3) = 0

– Additionneur unique quels que soient les signes de nombres

Notes :

• Nombres réels

– Codage en virgule fixe cadrée à gauche $D = -d_{N-1} + 2^{-(N-1)} \sum_{i=0}^{N-2} d_i \, 2^i$

Le facteur d'échelle est implicite.

Codage de nombre compris entre -1 et 1

Codage d'un nombre négatif similaire à celui d'un nombre entier relatif.

Addition : débordement, extension de signe

Multiplication : bxb -> 2b bits, extension du signe sur 2b bits, doublement du signe

Conservation du domaine de codage : pour la multiplication

- en virgule fixe cadrée à gauche si A et B appartiennent au domaine de codage
  alors Z = A.B appartient au domaine de codage.

– Codage en virgule flottante

$D=M.2^E$

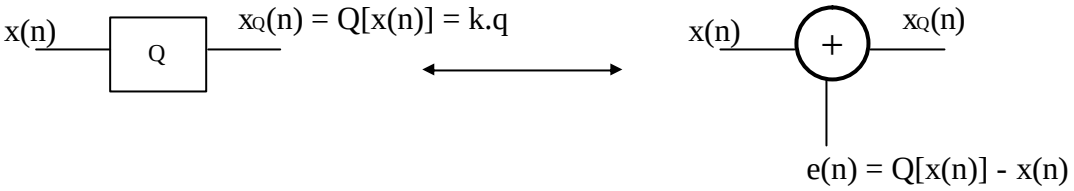
L'exposant E est explicite.

Mantisse M: nombre entier relatif, ou nombre réel.

Exposant E: nombre entier relatif.

Notes :

• Modèle



– Definition : Approximation de chaque valeur d'un signal x(n) par un multiple entier du pas de quantification élémentaire q.
– e(n) est l'erreur de quantification -> Modèle bruit-blanc additif

• Sources de bruit

– Bruit de conversion A/N
– Limitation des chemins de données de l'architecture cible

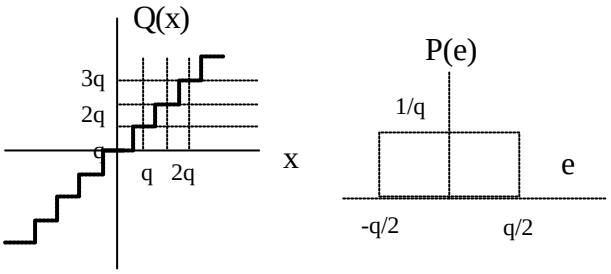
Multiplication => Quantification

Addition => Débordement

Notes :

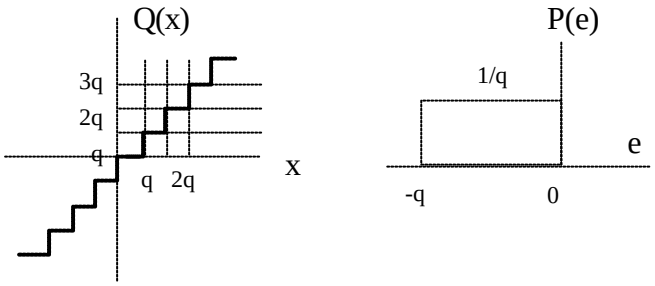
(a) Arrondi

$Q(x) = k.q$ si $(k-0.5).q \leq x < (k+0.5).q$



(b) Troncature

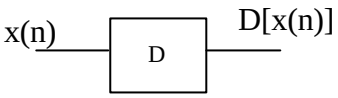
$Q(x) = k.q$ si $k.q \leq x < (k+1).q$



Etude statistique

- {e(n)} est une séquence d'un processus aléatoire stationnaire
- {e(n)} est décorrelée de {x(n)}
- {e(n)} est un bruit blanc additif
- la distribution de probabilité de {e(n)} est uniforme sur l'intervalle de quantification
- ergodicité : moyennes temporelles = moyennes statistiques
- moyenne m_e
- variance σ_{e^2} = Puissance du bruit
  variance = $q^2/12$

Notes :

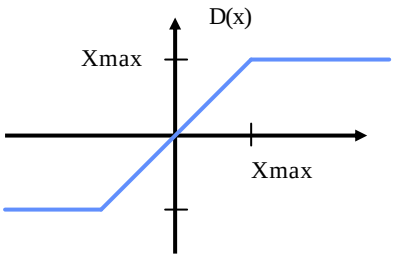


– Valeurs de x(n) lorsqu'il sort de la dynamique de codage

Saturation

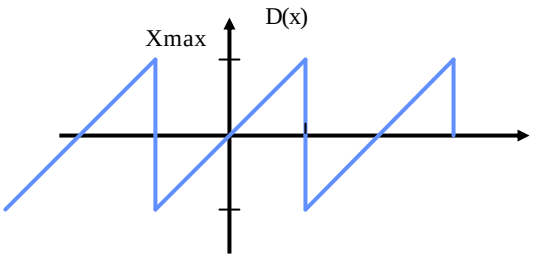
– Complexe

– Moins d'effets indésirables



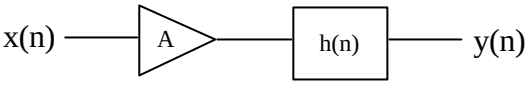
Modulaire

– Effets indésirables



Notes :

– Afin d'éviter le dépassement, on diminue l'amplitude avant ou pendant le traitement par un facteur d'échelle A < 1 (*scaling*).



- *A* peut être combiné avec les valeurs des coefficients
- *A* puissance de 2 (en pratique)

• Critères

– Critère du pire-cas

Pas de dépassement tant que |x(n)|<Xmax

– Critère de puissance

Pas de dépassement tant que Px<Pmax

– Critère du sinus

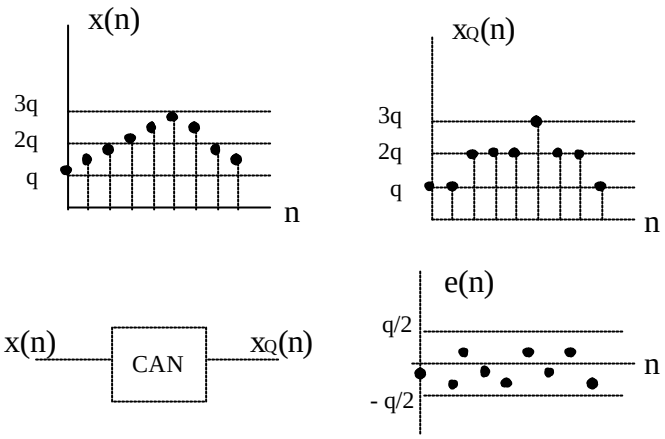
Pas de dépassement tant que |x(n)|<Xmax, avec x(n) sinusoïdal.

Notes :

IV Quantification

3 Bruit de conversion A/N

Quantification en conversion A/N

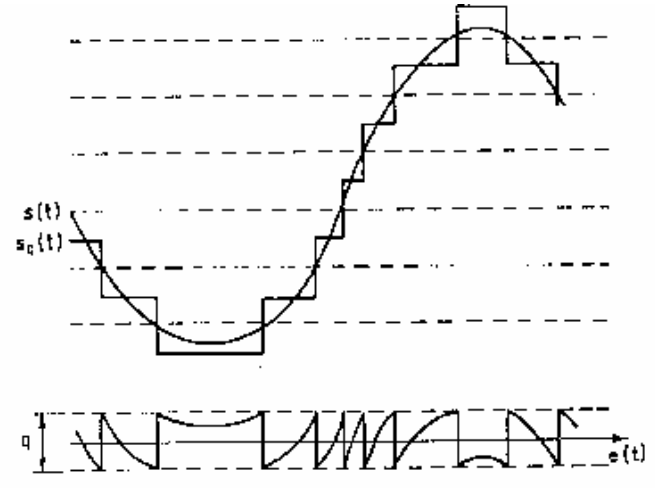


$x_Q(n) = Q[x(n)]$

$e(n) = x_Q(n) - x(n)$

$|e(n)| \leq q/2$

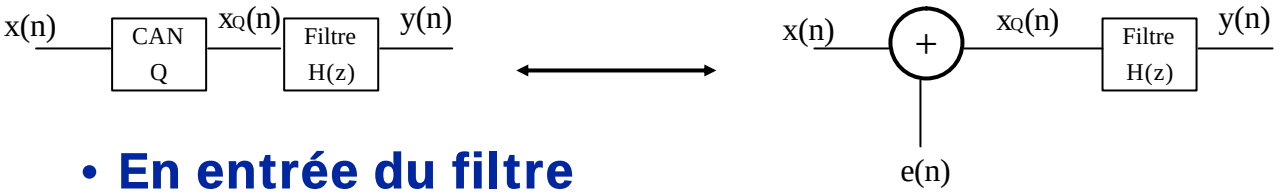
Quantification d'une sinusoïde



Notes :

Filtrage d'un bruit

- **Exemple : filtrage du bruit de conversion**



- **En entrée du filtre**
  - Signal x(n) + Bruit de conversion e(n)

$$s_e^2 = \frac{q^2}{12}, s_x^2 \; puissance du signal d'entrée$$

$$RSB = \frac{s_x^2}{s_e^2} = \frac{s_x^2}{q^2/12} = 12.2^{2(b-1)} s_x^2$$

$$RSB_{dB} = 10\log RSB = 6.02b + 4.77 + 10\log s_x^2$$

Le RSB augmente de 6dB par bit ajouté

Notes :

- **Limitation des chemins de données de l'architecture cible**
  - Multiplication => Quantification
  - Addition => Débordement

0,1101	0,8125	0,1101	0,8125
+ 0,1001	+ 0,5625	x 0,1001	x 0,5625
01,0110	1,375	00,0111 0101	0,4570 3125

Notes :

V. Synthèse des filtres RII

1. Introduction
2. Rappels sur la synthèse des filtres analogiques
3. Invariance impulsionnelle
4. Transformation bilinéaire

VI. Synthèse des filtres RIF

1. Introduction
2. Filtre à phase linéaire
3. Méthode du fenêtrage
4. Méthode de l 'échantillonnage fréquentiel

Notes :

V Synthèse des filtres RII

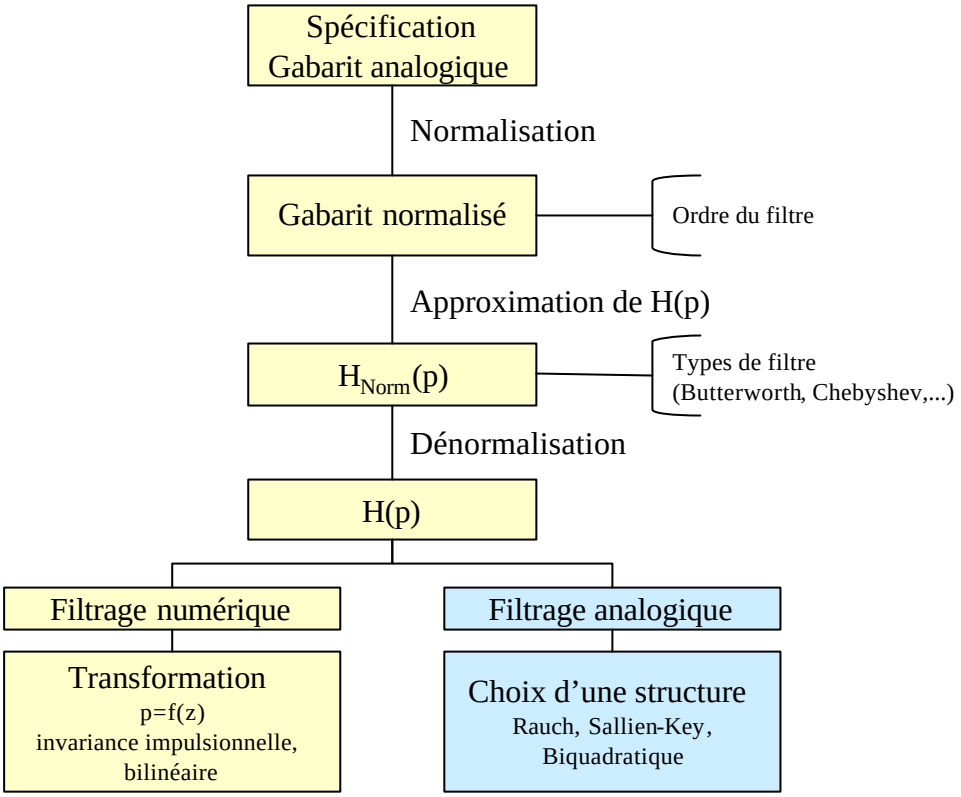
1 Introduction

- **Recherche de H(z) correspondant aux spécifications (gabarit)**
  - Transposition des méthodes de synthèse applicables aux filtres analogiques, puis transformation de H(p) vers H(z)
    - Invariance impulsionnelle
    - Transformation bilinéaire
  - Synthèse directe en z
  - Méthodes d'optimisation : minimiser un critère d'erreur entre courbe réelle et courbe idéale

Notes :

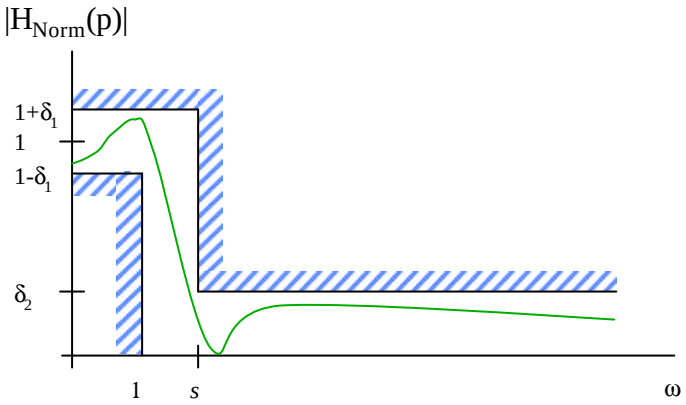
V Synthèse des filtres RII

2 Synthèse de filtre analogique

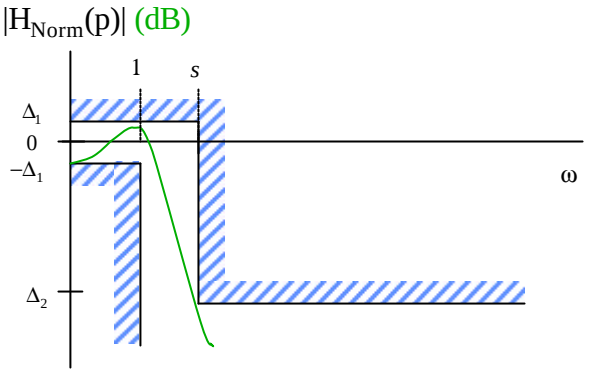


Notes :

- **Normalisation**
  - Calcul de la sélectivité s



a) Gabarit prototype linéaire



b) Gabarit prototype en dB

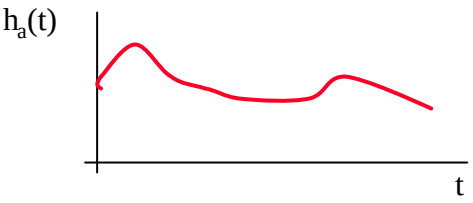
Notes :

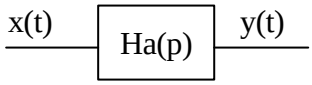
- **Ordre du filtre et fonction de transfert normalisée**
  - Butterworth, Chebyschev, Elliptique, Bessel, Legendre, ...
  - $H_{NORM}(p_N)$
- **Dénormalisation**
  - Passe-bas : $p_N = p / \omega_c$
  - Passe-haut : $p_N = \omega_c / p$
  - Passe-bande : $p_N = 1/B\ (p / \omega_0 + \omega_0 / p)$
- **On obtient une fonction de transfert H(p) respectant le gabarit analogique spécifié**

$\Rightarrow$ **Passage vers H(z)**

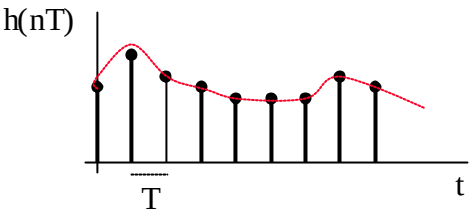
Notes :

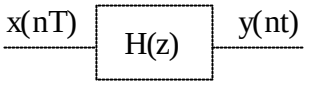
- **Le filtre numérique et le filtre analogique ont la même réponse impulsionnelle**





filtre analogique





filtre numérique

$$h(nT) = h_a(t) \Big/_{t = nT}$$

Notes :

- **Le filtre numérique et le filtre analogique ont la même réponse impulsionnelle**

$$H_a(p) \xrightarrow{L^{-1}} h_a(t) \xrightarrow{t=nT} h(nT) \xrightarrow{Tz} H(z)$$

ou formulation directe

$$H(z) = \sum_{\{pôles\ p_i\ de\ H(p)\}} Résidus\left[\frac{H_a(p)}{1-z^{-1}e^{pT}}, p_i\right]$$

– Conserve la réponse temporelle et la stabilité
– Phénomène de recouvrement de spectre du à l'échantillonnage
– Non respect de la spécification fréquentielle

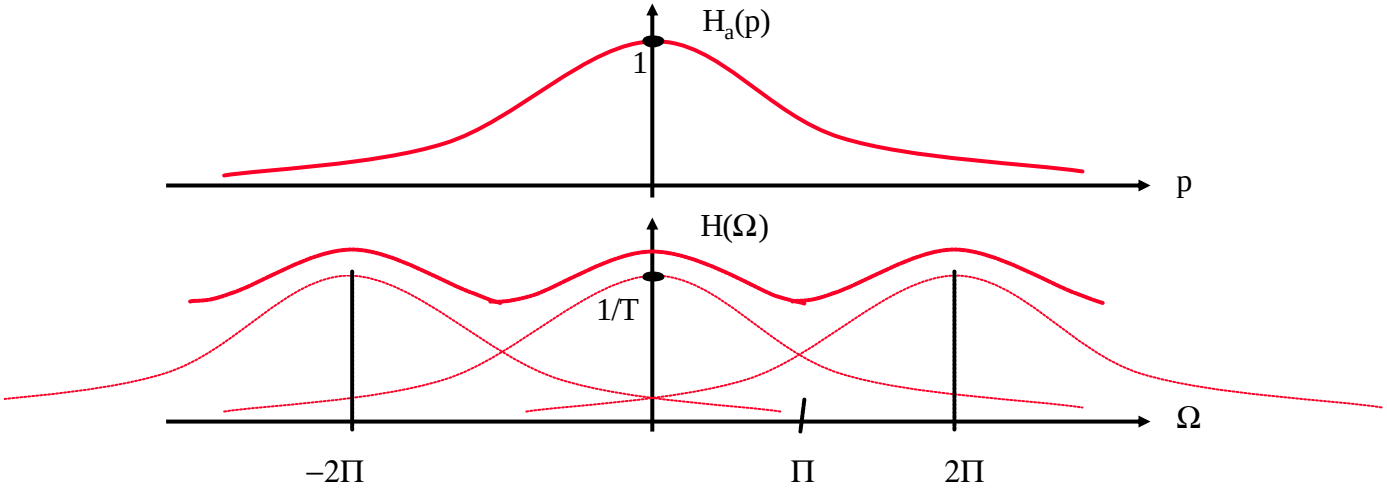
$$H(e^{j\Omega}) = \frac{1}{T}\sum_k H_a\left(jw + j\frac{2pk}{T}\right)$$

Notes :

V Synthèse des filtres RII

3 Invariance impulsionnelle

- **Réponse fréquentielle**

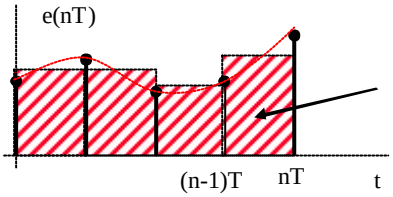


– Normalisation

(xT) ou (/H(0))

Notes :

- **Approximation d'une intégrale par la méthode des rectangles**



$$S(n) = S(n-1) + T[e(n) + e(n-1)]/2$$

Intégrateur numérique

Intégrateur analogique

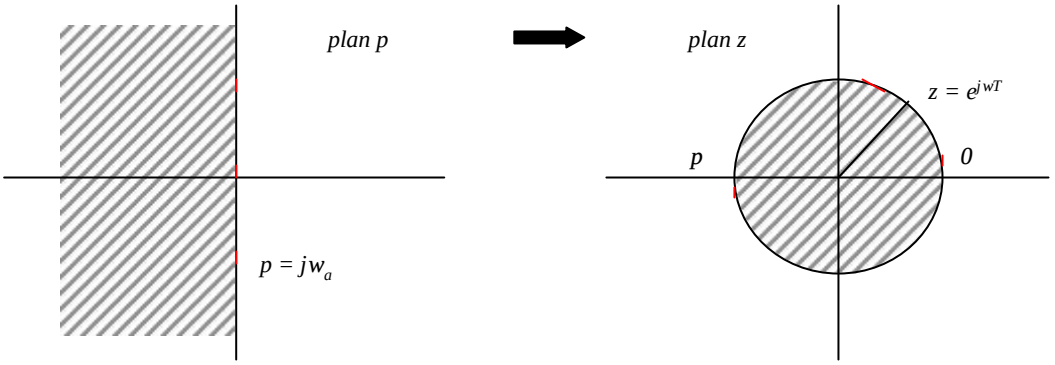
$$p = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}}$$

Notes :

V Synthèse des filtres RII

4 Transformation bilinéaire

– Conservation de la stabilité



– Relation entre fréquences numériques et analogiques

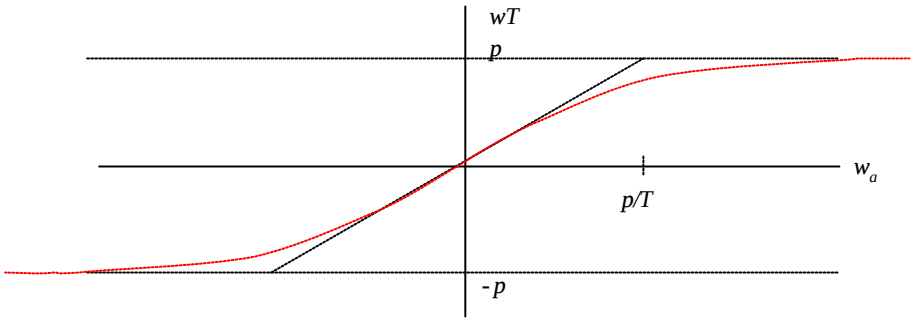
Notes :

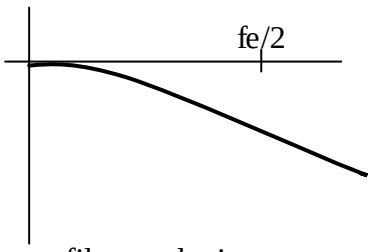
V Synthèse des filtres RII

4 Transformation bilinéaire

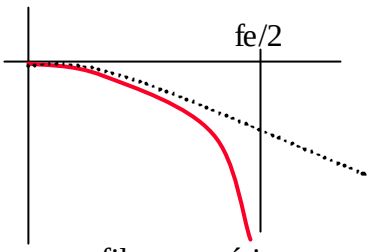
– Distorsion en fréquence connue

$$w_{analogique}\frac{T}{2} = tg\left(w_{numérique}\frac{T}{2}\right)$$





filtre analogique



filtre numérique

Notes :

• Procédure de synthèse

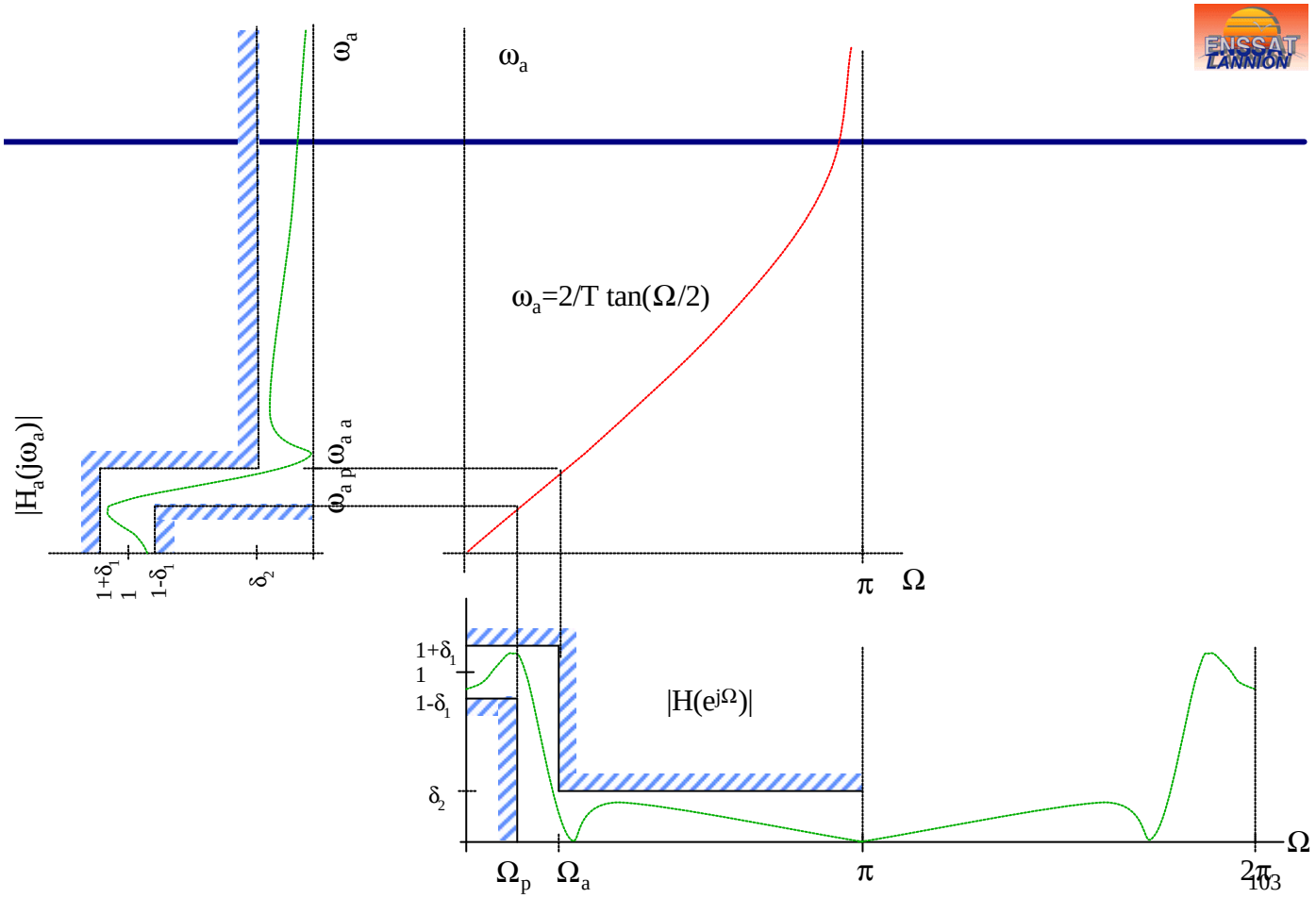
– A partir du gabarit en fréquence numérique ω_n
– Effectuer une prédistorsion en fréquence

$$w_a \frac{T}{2} = tg\left(w_n \frac{T}{2} \right)$$

– Synthèse de H(p) par méthodes du chapitre V.2
– Transformation bilinéaire

$$H(z) = H(p) \Big/ p = \frac{2}{T} \frac{1 - z^{-1}}{1 + z^{-1}}$$

Notes :


ωa
ωa
ωa=2/T tan(Ω/2)
|Ha(jωa)|
ωa p
ωa a
1+δ1
1
1-δ1
δ2
π
Ω
1+δ1
1
1-δ1
δ2
|H(ejΩ)|
Ωp
Ωa
π
2π
Ω

Notes :

VI Synthèse des filtres RIF

1 Introduction

- **Recherche de H(z) correspondant aux spécifications (gabarit)**
  - Synthèse directe en z
  - Filtres à phase linéaire ou minimale
- **3 méthodes de synthèse**
  - Méthode du fenêtrage
  - Méthode de l'échantillonnage fréquentiel
  - Méthodes d'optimisation : minimiser un critère d'erreur entre courbe réelle et courbe idéale

Notes :

VI Synthèse des filtres RIF

2 Phase linéaire

- **Filtre à phase minimale**
  - Zéros dans le cercle unité
- **Filtre à phase linéaire**

$$H(e^{j\Omega}) = A(\Omega).e^{j\,j\,(\Omega)}$$

$$avec \begin{cases} A(\Omega): pseudo - module \ \ (amplitude) \\ j\,(\Omega) = b - a\Omega \end{cases}$$

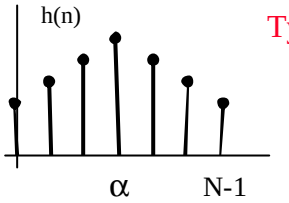
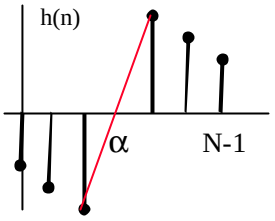
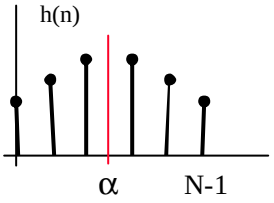
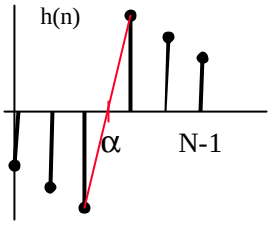
  - Condition pour avoir une phase linéaire

    Symétrie ou antisymétrie par rapport à $\alpha = (N-1)/2$

Notes :

VI Synthèse des filtres RIF

2 Phase linéaire

	réponse impulsionnelle symétrique $\beta=0$	réponse impulsionnelle antisymétrique $\beta=\pm\pi/2$
N impair α entier	Type I 	Type III 
N pair α non entier	Type II 	Type IV 

Notes :

VI Synthèse des filtres RIF

2 Phase linéaire

Type I

$$H(e^{j\Omega}) = e^{-ja\Omega}\sum_{n=0}^{a} a_n \cos(n\Omega)$$

$$a_0 = h(a),\ \ a_n = 2h(a-n),\ n = 1\ldots a$$

N impair

Tout filtre

Type III

$$H(e^{j\Omega}) = e^{j\frac{p}{2}}e^{-ja\Omega}\sum_{n=1}^{a} c_n \sin(n\Omega)$$

$$c_n = 2h(a-n),\ n = 1\ldots a$$

$$H(0) = H(p) = 0$$

Passe Bande
Dérivateur

N pair

Type II

$$H(e^{j\Omega}) = e^{-ja\Omega}\sum_{n=1}^{N/2} b_n \cos[(n-1/2)\Omega]$$

$$b_n = 2h(N/2-n),\ n = 1\ldots N/2$$

$$H(p) = 0$$

~~Passe Haut~~

Type IV

$$H(e^{j\Omega}) = e^{j\frac{p}{2}}e^{-ja\Omega}\sum_{n=1}^{N/2} d_n \sin[(n-1/2)\Omega]$$

$$d_n = 2h(N/2-n),\ n = 1\ldots N/2$$

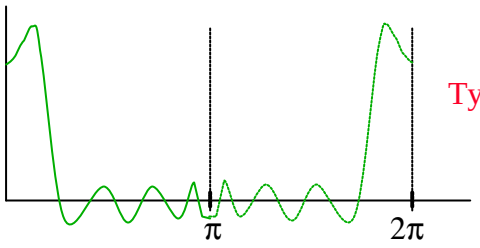
$$H(0) = 0$$

Passe Haut
Dérivateur

Notes :

VI Synthèse des filtres RIF

2 Phase linéaire

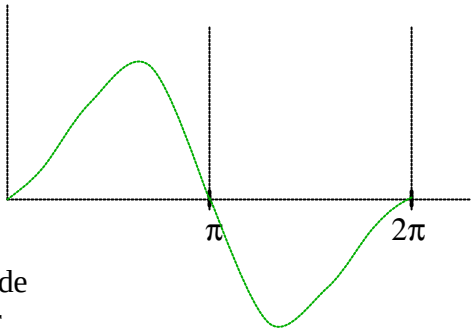


Type I

N impair

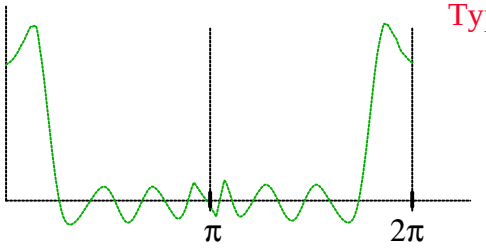
Tout filtre

Type III



Passe Bande
Dérivateur

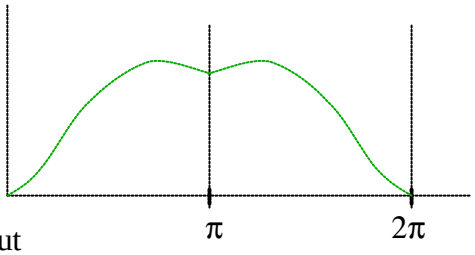
N pair



Type II

~~Passe Haut~~

Type IV



Passe Haut
Dérivateur

Notes :

VI Synthèse des filtres RIF

3 Méthode du fenêtrage

- **Développement en série de Fourier du filtre idéal**

$$H(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} h(n)\, e^{-jn\Omega}$$

$$h(n) = \frac{1}{2p}\int_{-p}^{p} H(e^{j\Omega}).e^{jn\Omega}\, d\Omega$$

  – Filtre non causal, de type RII

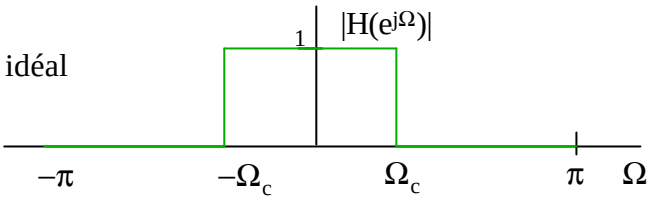
- **Passage de h(n) idéal au RIF approché par fenêtrage de h(n)**

$$\boxed{h_a(n) = h(n).w(n)}$$

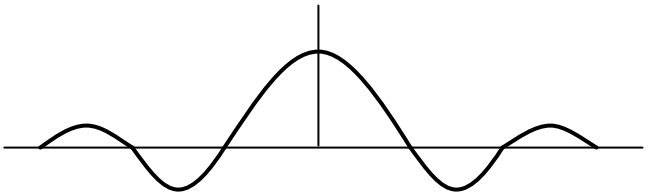
Notes :

- **Exemple : filtre passe-bas idéal**

Filtre passe-bas idéal



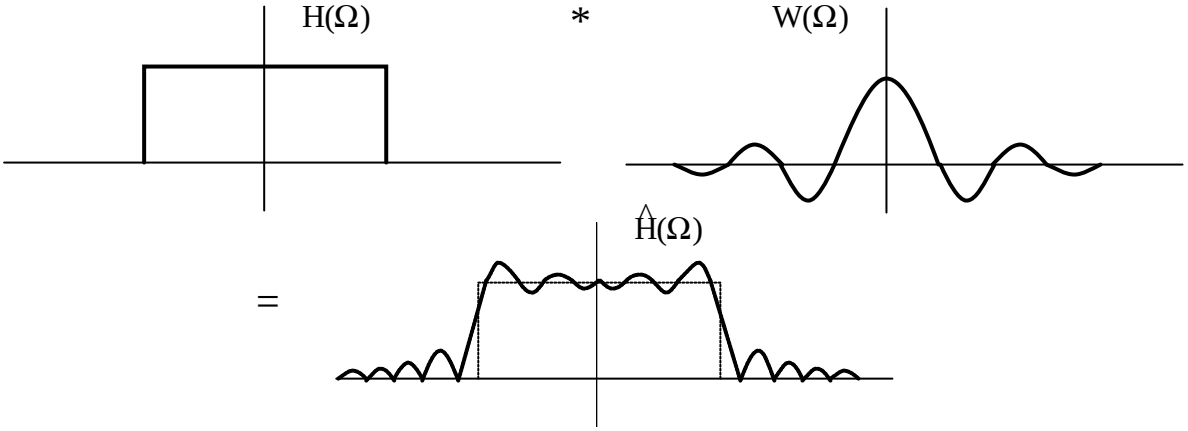
$$h(n) = \frac{\Omega_c}{p}\frac{\sin(n\Omega_c)}{n\Omega_c}$$



Notes :

- **Prise en compte d'une condition de phase linéaire par décalage de** a
- **Fenêtrage de h(n)**

$$h_a(n) = h(n).w(n) \Leftrightarrow H_a(e^{j\Omega}) = H(e^{j\Omega}) * W(e^{j\Omega})$$



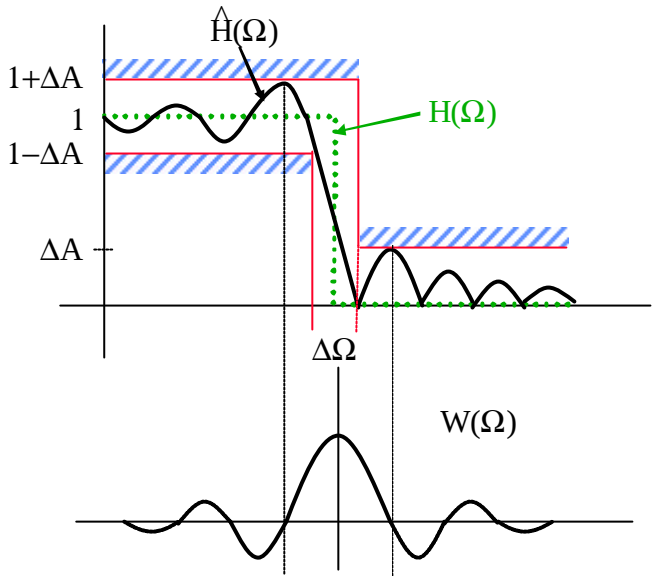
Notes :

VI Synthèse des filtres RIF

3 Méthode du fenêtrage

– Largeur de la zone de transition $\Delta\Omega \Leftrightarrow$ 1/2 largeur du lobe principal

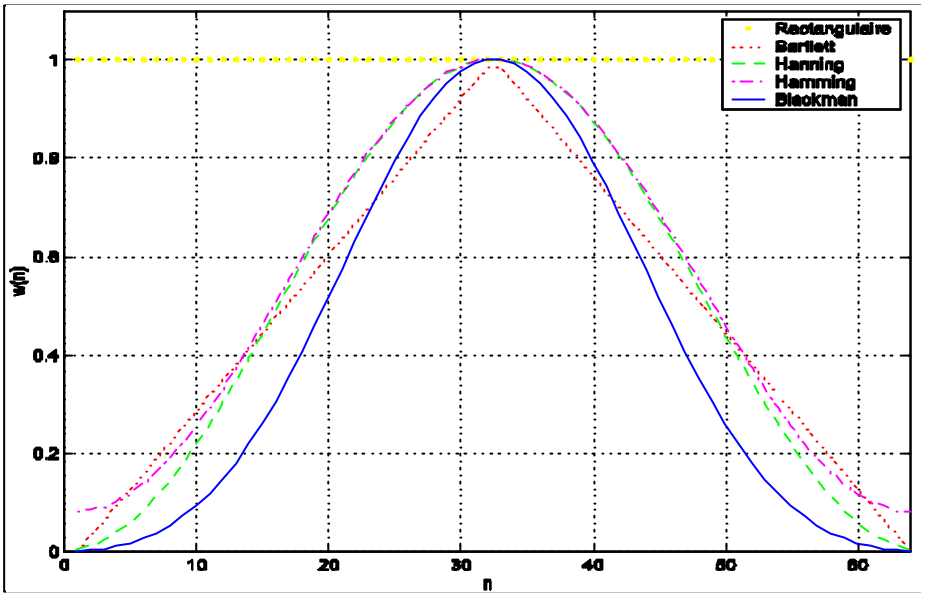
– Atténuation $\Delta A \Leftrightarrow$ amplitude du premier lobe secondaire



Notes :

3 Méthode du fenêtrage

- **Fenêtres usuelles**
  - Rectangle, Triangule, Hanning, Hamming, Blackman, Kaiser, ...
  - Réponses temporelles

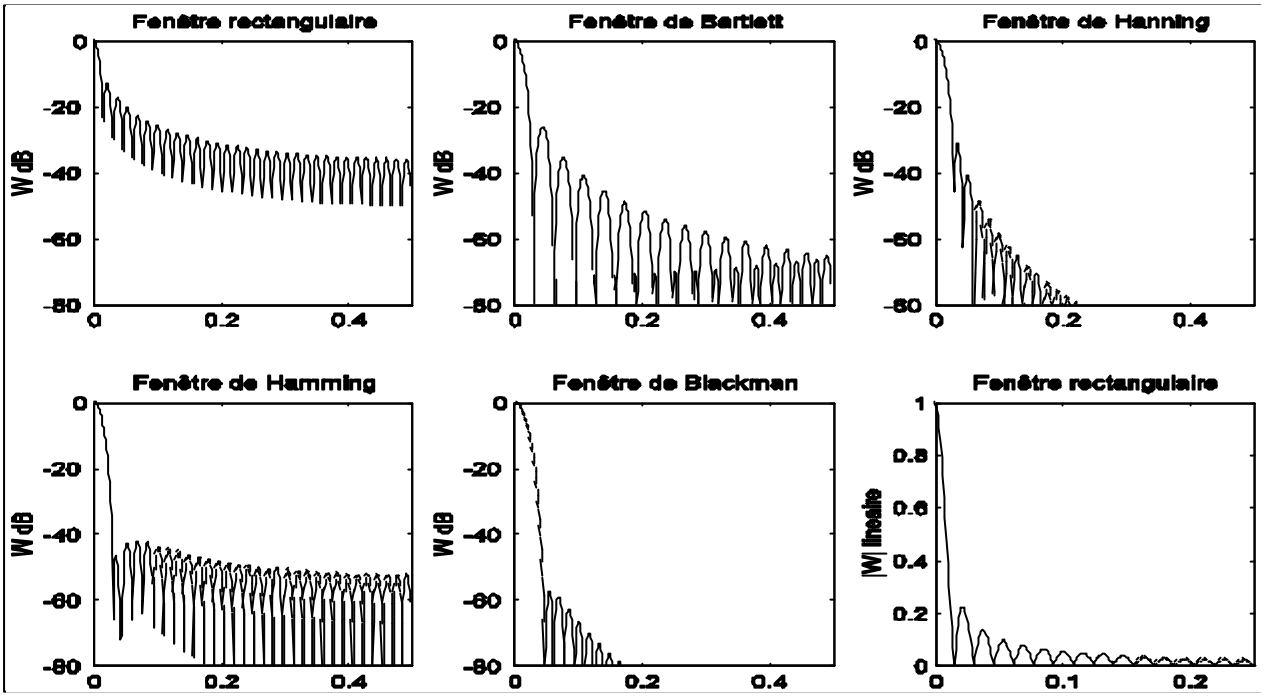


Notes :

IV.2 Synthèse des filtres RIF

3 Méthode du fenêtrage

- **Fenêtres usuelles**



Notes :

• Influence de la fenêtre

Fenêtre	Lobe secondaire	Demi largeur du lobe principal	Atténuation minimum
Rectangulaire	-13dB	$2\pi/N$	-21dB
Triangulaire	-25dB	$4\pi/N$	-25dB
Hanning	-31dB	$4\pi/N$	-44dB
Hamming	-41dB	$4\pi/N$	-53dB
Blackman	-57dB	$6\pi/N$	-74dB

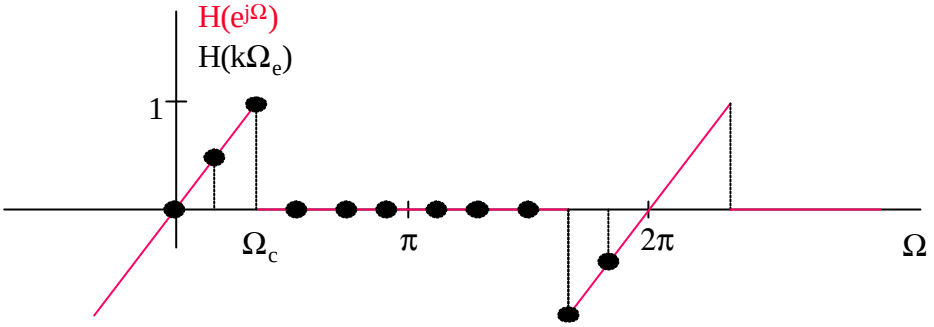
– Le type de fenêtre influe sur ΔA et $\Delta\Omega$

– Le nombre de points influe sur $\Delta\Omega$

Notes :

• Échantillonnage en fréquence

– Échantillonnage du filtre idéal



– TFD inverse de $H(k\Omega_e)$

$$h(n) = \frac{1}{N}\sum_{k=0}^{N-1} H(k\Omega_e) e^{j\frac{2p}{N}n.k}$$

– Méthode valable pour tout type de filtre

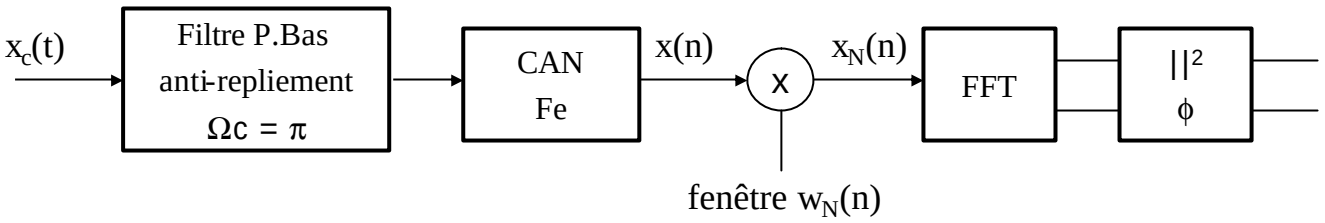
– Possibilité d'utiliser un fenêtrage

Notes :

VII Analyse spectrale

1 Définition

- **Analyse spectrale de signaux continus**
  - Etude du contenu fréquentiel (spectre) d'un signal continu xc(t)
  - Nombre limité d'échantillons du signal d'entrée pour la TFD



- **Troncature temporelle**
  - $x_N(n) = x(n) \cdot w_N(n)$ avec $w_N(n)$ fenêtrage sur N points
  - $T_0 = N.T$: horizon d'observation

Notes :

- **Troncature temporelle**
  - $x_N(n) = x(n) \,.\, w_N(n)$ avec $w_N(n)$ fenêtrage sur N points
- **Influence sur le spectre**
  - Convolution fréquentielle $X_N(e^{j\Omega}) = X(e^{j\Omega}) * W_N(e^{j\Omega})$
- **TFD du signal tronqué**

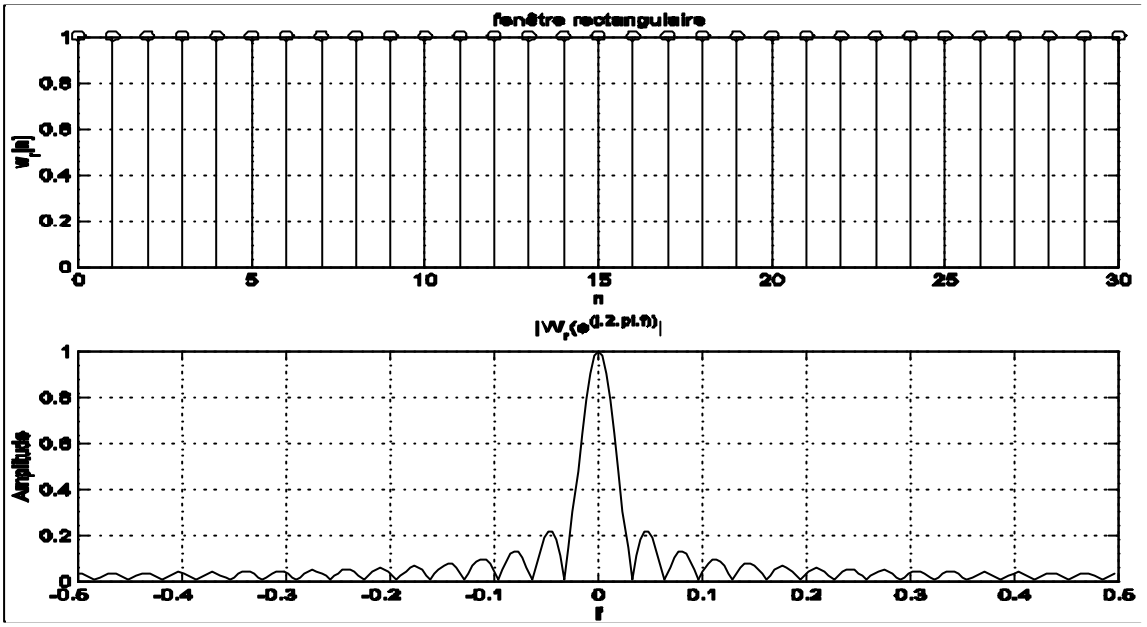
$$X_N(k) = \sum_{n=0}^{L-1} x_N(n)\, e^{-2jp\frac{kn}{L}} \qquad k = 0..L-1$$

$$L \geq N$$

$$X_N(k) = X_N(e^{j\Omega})\Big|_{\Omega = 2pk/L}$$

Notes :

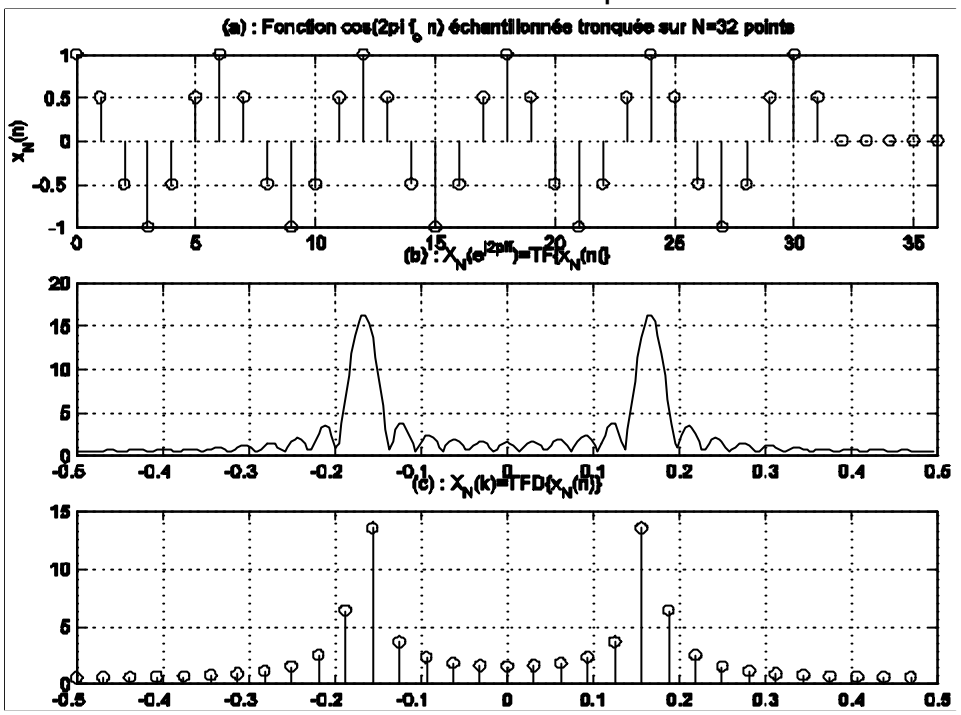
- **Exemple**
  - Fenêtre rectangulaire, N=31



Notes :

3 Influence de la fenêtre

– fonction cosinus fenêtrée sur N=32 points



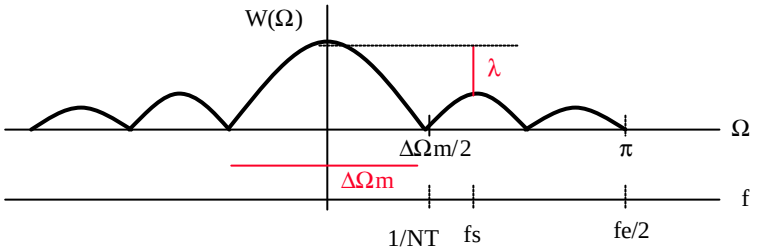
Notes :

• Influence de la fenêtre

Fenêtre	Lobe secondaire $\lambda = 20\log\|W(fs)/W(0)\|$	Largeur du lobe principal $LLP = \Delta\Omega m$
Rectangulaire	-13dB	$4\pi/N$
Triangulaire	-25dB	$8\pi/N$
Hanning	-31dB	$8\pi/N$
Hamming	-41dB	$8\pi/N$
Blackman	-57dB	$12\pi/N$

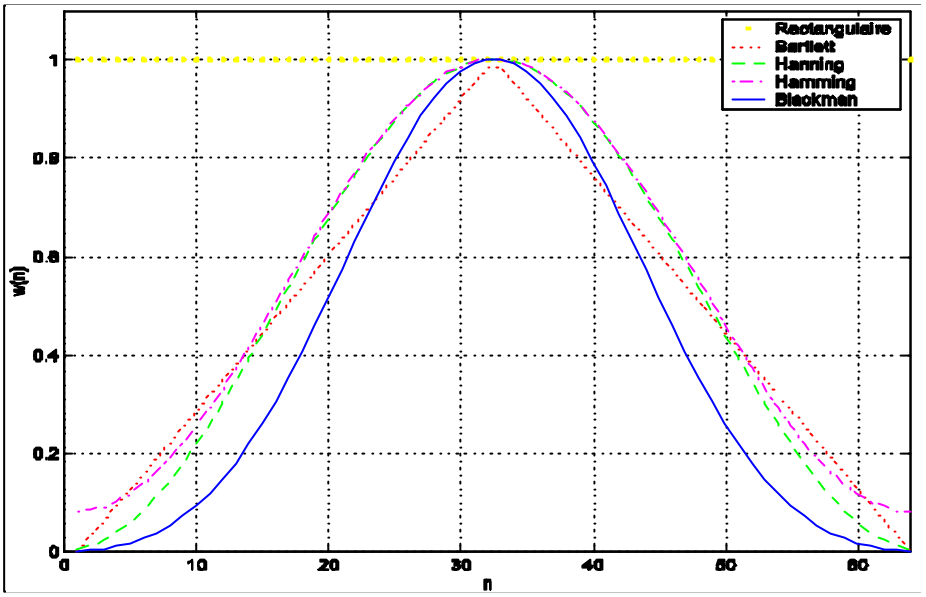
– Le type de fenêtre influe sur λ et $\Delta\Omega m$

– Le nombre de points influe sur $\Delta\Omega m$



Notes :

- **Fenêtres usuelles**
  - Rectangle, Triangule, Hanning, Hamming, Blackman, Kaiser, ...
  - Réponses temporelles

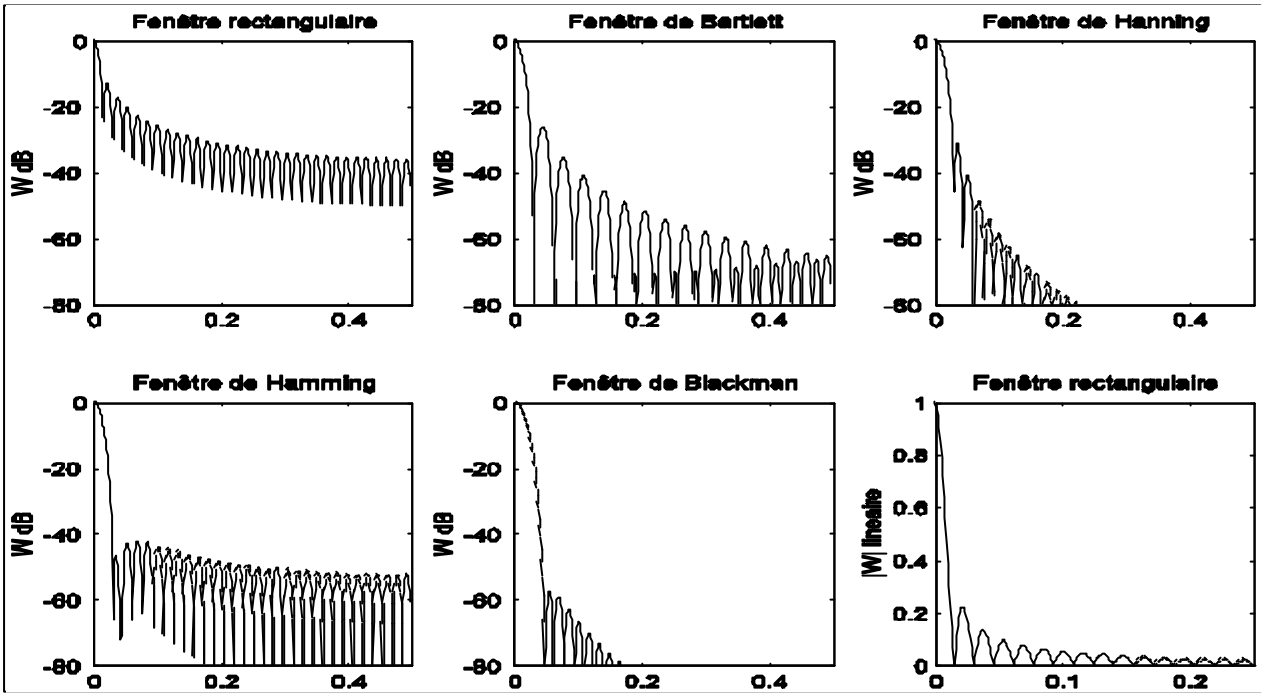


Notes :

VII Analyse spectrale

3 Influence de la fenêtre

- **Fenêtres usuelles**



Notes :

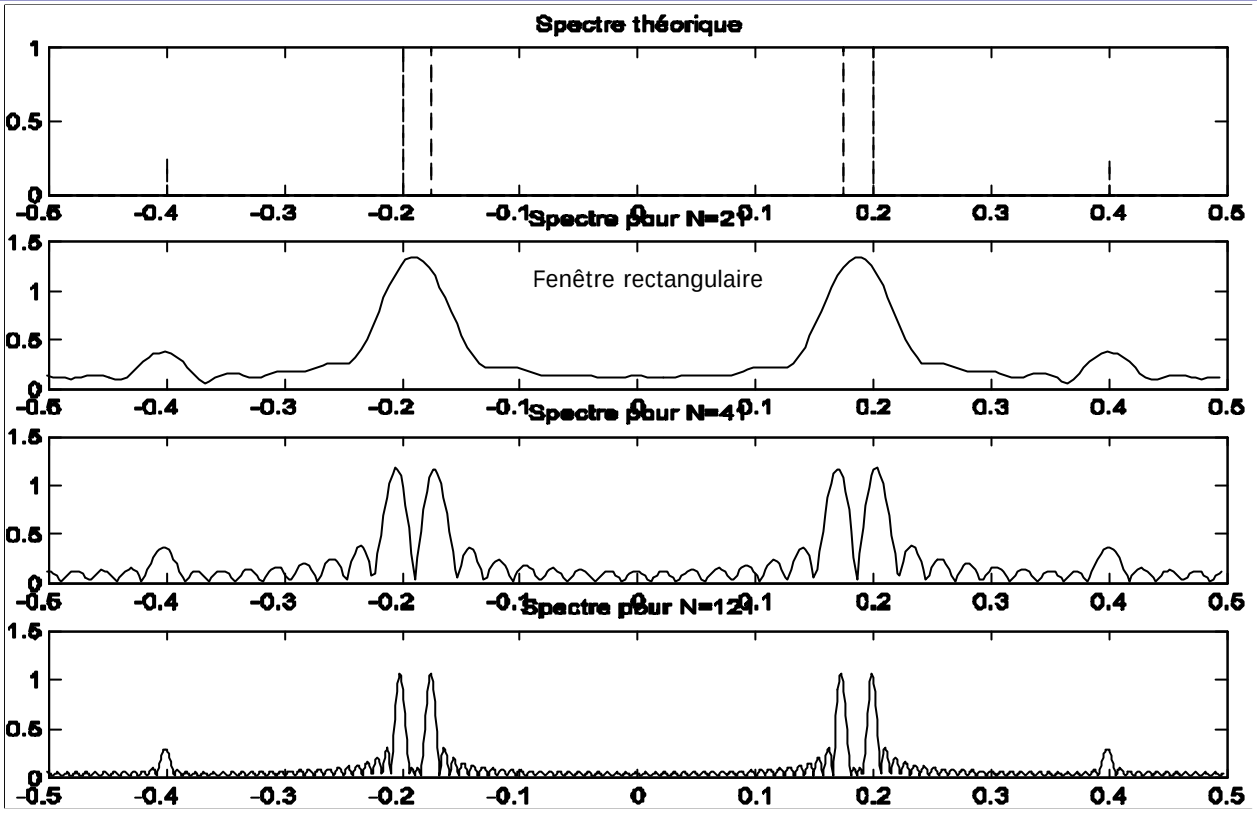
- **Finesse en fréquence**
  - Capacité de l'analyseur à détecter 2 raies proches
  - Masquage fréquentiel
  - Largeur du lobe principal : LLP = $2\Delta\Omega$
  - Dépend de N et du type de fenêtre

    Exemple sur transparent 9
- **Finesse en amplitude**
  - Capacité de l'analyseur à détecter des raies de faibles amplitudes ou masquée par une autre raie proche
  - Masquage d'amplitude ou bruit de l'analyse

    $\lambda = 20\log|W(fs)/W(0)|$
  - Dépend du type de fenêtre

    Exemple sur transparent 10

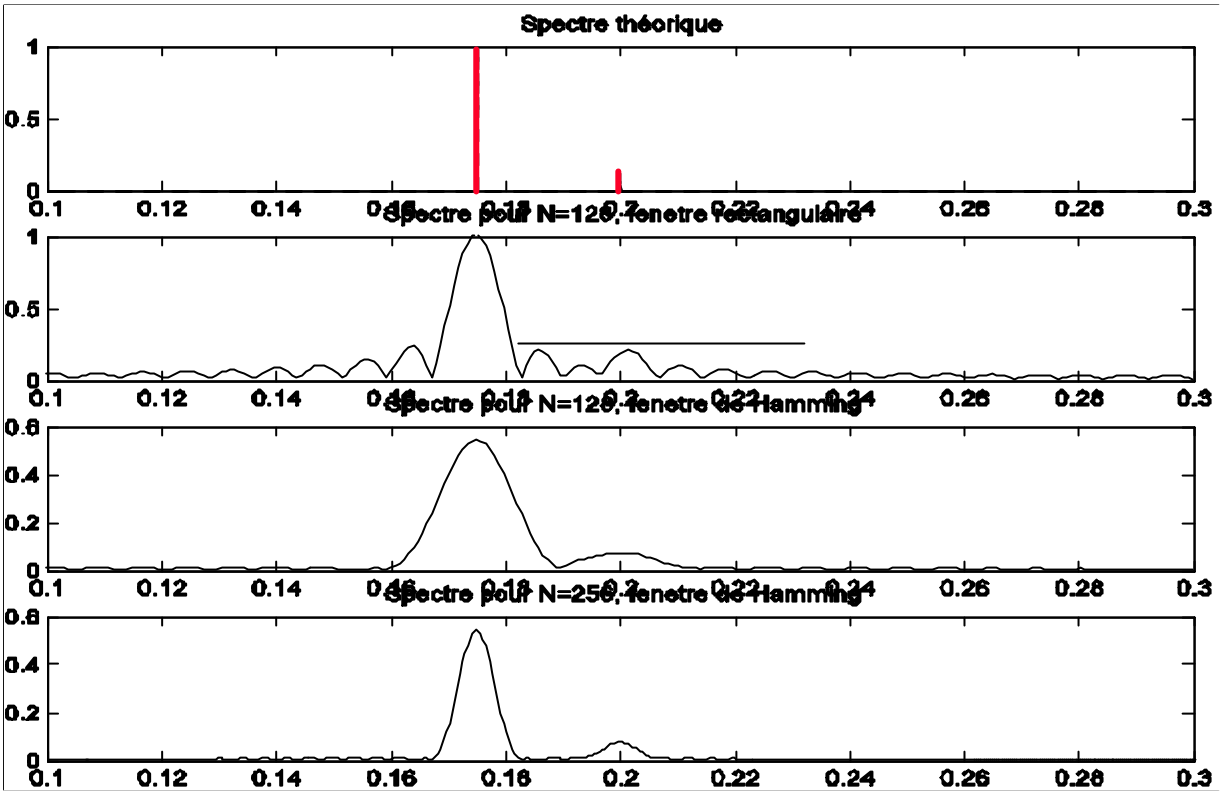
Notes :


Spectre théorique
Spectre pour N=2
Fenêtre rectangulaire
Spectre pour N=4
Spectre pour N=12

Notes :

VII Analyse spectrale

4 Paramètres de l'analyse

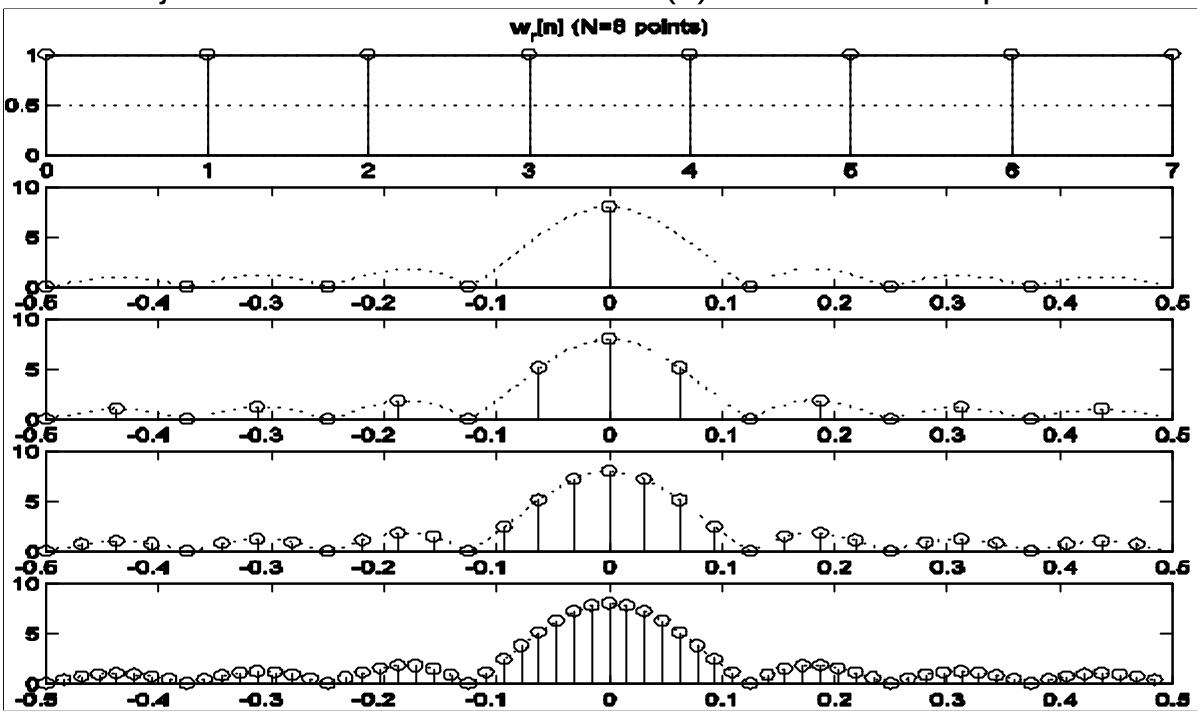


Notes :

VII Analyse spectrale

5 Zero-padding

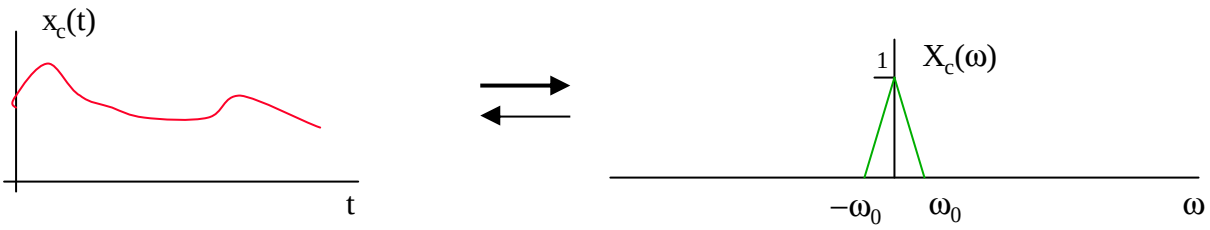
– Ajout de L-N zéros à la suite de x(n) avant TFD sur L points



Notes :

- **Systèmes multi-cadences**
  - Systèmes dans lesquels on pourra avoir plusieurs fréquences d'échantillonnage dans une même chaîne de traitement
  - Ils tirent partie de la forme spectrale d'un signal en gardant Fe toujours à sa valeur optimale

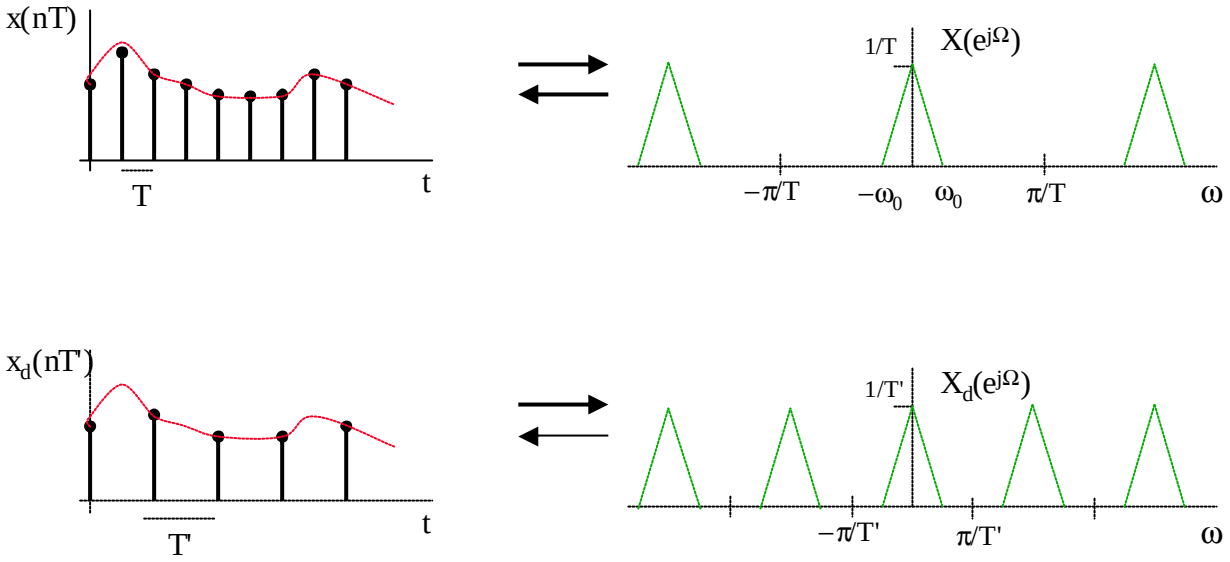
    -> Réduction de la complexité



Notes :

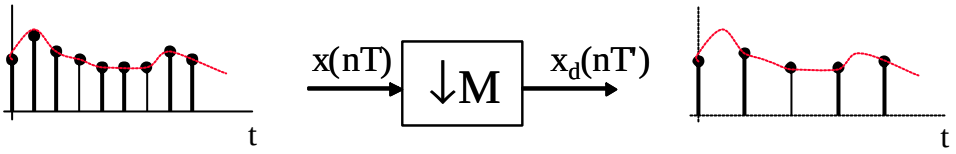
VIII Systèmes multi-cadences

1 Définition



Notes :

• Décimation d'un facteur *M*



$$T' = MT$$

$$F'e = Fe/M$$

$$x_d(n) = x(nM)$$

$$X_d(e^{j\Omega}) = \frac{1}{M}\sum_{i=0}^{M-1} X(e^{j(\Omega/M - 2pi/M)})$$

– Pour éviter le recouvrement de spectre, le signal xc(t) doit être à bande limitée et respecter le théorème de Shannon par rapport à T'

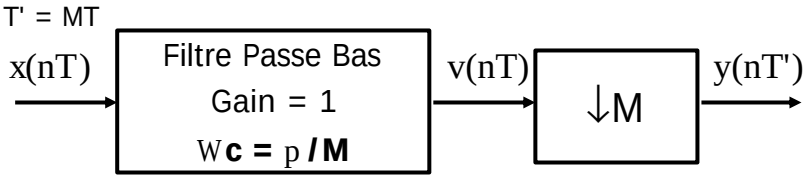
$$X_c(w) = 0 \quad pour \left|w\right| \geq w_0$$

$$et\ p/\mathrm{T}' = p/(\mathrm{MT}) \geq w_0 \ ou \ F'e = Fe/M \geq w_0$$

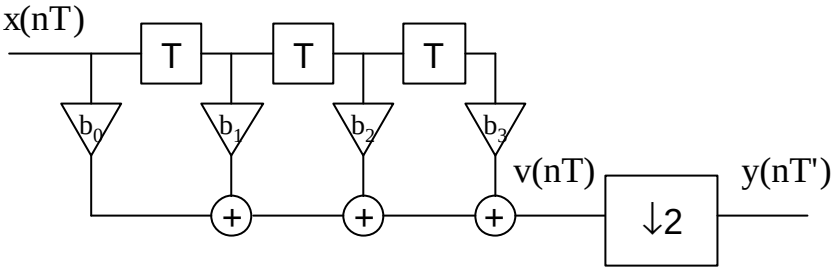
Notes :

• Filtres à décimation

– Filtre suivi d'un décimateur



• Optimisation du filtre à décimation



Notes :

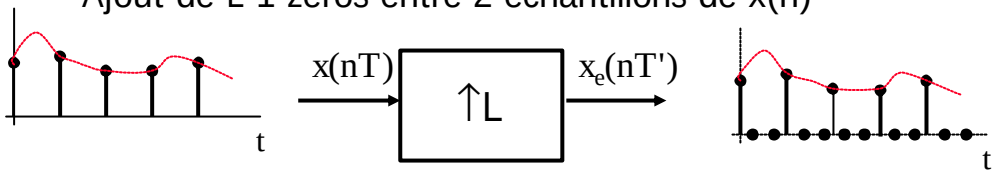
• Interpolation d'un facteur *L*

– Objectif : augmenter la fréquence d'échantillonnage d'un signal x(n) échantillonné à la période T d'un facteur L

$$x_i(n) = x(n/L) = x_c(nT'), \quad avec\ T' = T/L$$

• Elévateur de fréquence

– Ajout de L-1 zéros entre 2 échantillons de x(n)



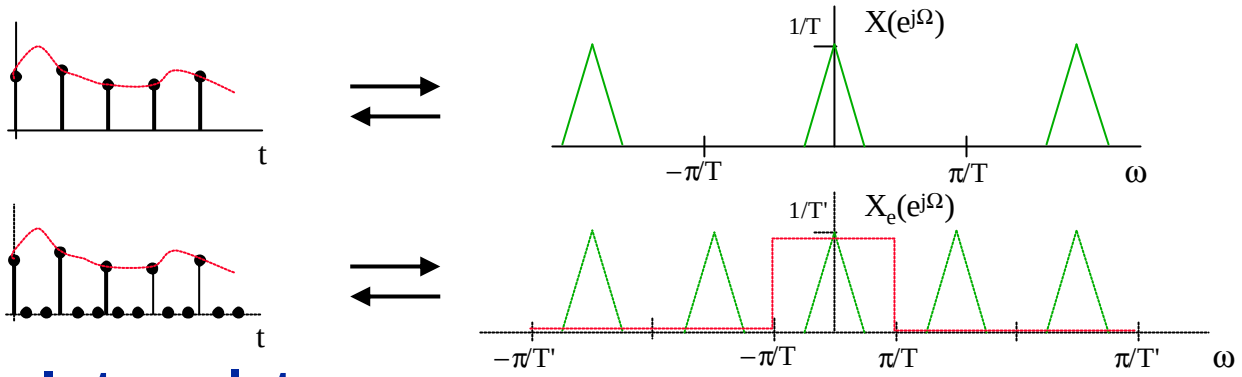
$$x_e(n) = \begin{cases} x(n/L), n = 0, \pm L, \pm 2L, \ldots \\ 0\ ailleurs \end{cases}$$

$$X_e(e^{jwT'}) = X(e^{jwT})$$

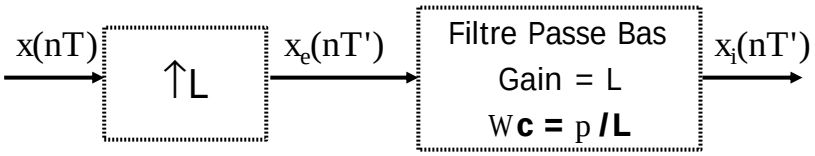
$$X_e(e^{j\Omega}) = X(e^{j\Omega L})$$

Pas d'effet sur le spectre

Notes :

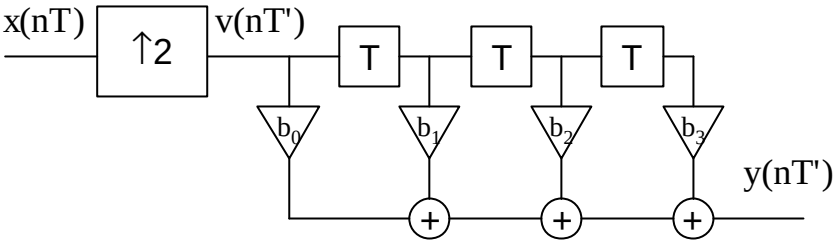


- **Interpolateur**
  - Succession d'un élévateur de fréquence et d'un filtre passe-bas idéal de gain L, de période d'échantillonnage T' et de fréquence de coupure Fc = 1/2T (i.e. Ωc = π/L).

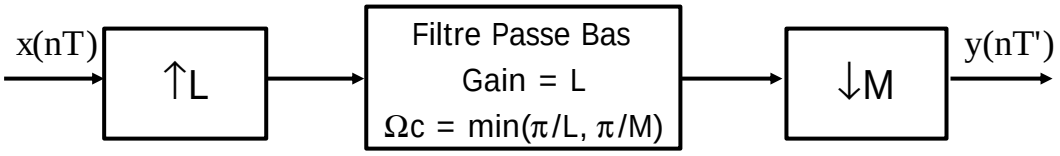


Notes :

- **Optimisation du filtre à interpolation**



- **Multiplication de Fe par un facteur rationnel R=L/M**
  - T' = T.M/L



Notes :