وزارة التعليم والبحث العلمي

MINISTERE DE L'ENSEIGNEMENT ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : D'ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

CONCEPTION ET REALISATION D'UN PROCESSEUR PIPELINE DE FFT

Proposé par :
M. BESSALAH HAMIB

Etudié par :
Melle MOUZALI FATIMA
M. RAMDA MERZAK

Dirigé par :
M. BESSALAH HAMIB

PROMOTION :

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم والبحث العلمي
Ministère de l'Enseignement et de la Recherche Scientifique

المدرسة الوطنية للعلوم الهندسية
ECOLE NATIONALE POLYTECHNIQUE D'ALGER

DEPARTEMENT ELECTRONIQUE

CENTRE DE DEVELOPPEMENT DES TECHNIQUES AVANCEES
LABORATOIRE ARCHITECTURE DES SYSTEMES

PROJET DE FIN D'ETUDES
INGENIORAT EN ELECTRONIQUE

THEME

Conception et Réalisation d'un Processeur Pipeline de F F T

Proposé et suivi par :
M. BESSALAH Hamid

Présenté par :
Melle MOUZALI Fatima
M. RAMDA Merzak

Promotion Janvier 1984

R E M E R C I E M E N T S

Nous remercions Monsieur ABELAOUI A. Directeur du centre de developpement des techniques avancées pour nous avoir accueilli au C D T A, et Madame RASNAAMA F. secrétaire du C D T A pour avoir règlé tous nos problèmes. administratifs,

Nos plus vifs remerciements à Monsieur BESSALAH H. responsable du laboratoire architecture des systèmes, dont l'aide et les conseils nous ont été particulièrement précieux, pour les connaissances que nous avons pu acquerir au cours de notre projet.

Nous remercions vivement Monsieur BOUZID M. technicien au laboratoire architecture des systèmes, pour son amical collaboration.

Que le personnel du laboratoire photogravure trouve ici l'expression de nos remerciements pour leur amicale collaboration.

Nous tenons à remercier particulièrement Madame BOUBIA M. pour le soin apporté à la dactylographie de cette these.

Que Monsieur LAAZIB trouve ici l'expression de nos remerciements pour avoir assuré le tirage de cette thèse.

Que tous ceux qui ont contribué à l'accomplissement de ce travail trouvent ici l'expression de notre reconnaissance.

S O M M A I R E

INTRODUCTION 1

1 LA TRANSFORMEE DE FOURIER DISCRETE ET SES APPLICATIONS

1.1 Introduction 3
1.2 Etude de la transforméede fourier discrete (DFT) 5
1.3 La transformée de fourier discrete bidimensionnelle (DFTB) 6
1.4 Application de la transformée de fourier discrete 7
1.4.1 Spectre de puissance et spectre de puissance croisé 7
1.4.2 La correlation 8
1.4.3 La convolution 9
1.4.4 La fonction de transfert 9
1.4.5 La coherence 10
1.5 Conclusion 11

2 LES ALGORITHMES DE FFT ET LEURS PROPRIETES

2.1 Introduction 12
2.2 Rappel sur les transformation matricielles 13
2.3 Les algorithmes de FFT 16
2.3.1 L'algorithme A 16
2.3.2 L'algorithme B 22
2.3.3 L'algorithme C 23
2.3.4 L'algorithme D 23
2.4 Conclusion 25

3 ASPECTS ALGORITHMIQUES ET STRUCTURELS DES SYSTEMES RAPIDES DE TRAITEMENT DE DONNEES

3.1 Introduction 26
3.2 Les aspets algorithmiques 26
3.3 Les aspets structurels 27
3.3.1 Organisation lineaire 28
3.3.2 Organisation de la simultaneité 28
3.3.3 Organisation de la simultaneite sur la base de la division de la mémoire 30

3.3.4. Autres types d'organisation pour augmenter la rapidité d'un système 30
3.4. Classification des systèmes de traitement de données par flots de données et flots d'instructions 33
3.4.1. Les systèmes SISD 33
3.4.2. Les systèmes MISD 35
3.4.3. Les systèmes SIMD 35
3.4.4. Les systèmes MIMD 38
3.5 Conclusion 40

4 ETUDE COMPARATIVE DES PROCESSEURS PARALLELES DE FFT

4.1 Configuration du système de traitement du signal 41
4.2 Processeur à structure parallèle 41
4.3 Processeur à structure iteratve 44
4.4 Processeur à structure pipeline 44
4.5 Comparaison des trois structures et présentation de la solution choisie . 47
4.5.1 Description du schéma synoptique 47
4.5.2 Fonctionnement du processeur 50

5 PROCESSEUR PIPELINE DE FFT

5.1 Organisation du processus de calcul de la FFT 52
5.2 L'unité de traitement 55
5.2.1 Circuit d'aiguillage des données 56
5.2.2 Unité arithmétique 60
5.2.3 Circuit d'aiguillage des résultats 67
5.3 Le commutateur 1 70
5.4 L'unité mémoire 70
5.4.1 Organisation 70
5.4.2 Fonctionnement 75
5.4.3 Realisation 75
5.4.4 Mémoire PROM 78
5.5 Commutateur 2 78
5.6 Sequenceurs d'adresses 80
5.6.1 Le séquenceur Seq 1 80
5.6.2 Le séquenceur Seq 2 83
5.6.3 Le séquence Seq 3 89
5.7 Unité de controle 89
CONCLUSION 94
BIBLIOGRAPHIE 95

I N T R O D U C T I O N

Il est incontestable que le développement de la science et de la technologie par lesméthodes de simulation et de traitement de données experimentales, l'étude de phénomènes physiques pour un contrôle de processus industriels par exemple, la conception et la réalisation de dispositifs et installations complexes, l'analyse de projets, la gestion mathématique, le calcul scientifique et bien d'autres applications, aurait été impossible sans l'évolution rapide et l'utilisation de puissants moyens d'acquisitions, de stokage, d'édition et de traitement de l'information

Les énergies nouvelles ne font pas exception car beaucoup de problèmes liés à leurs développements (spectrometrie, études des radiations traitement d'images ...) trouvent leur solution dans le traitement numérique du signal.

Le traitement du signal est une technique qui requière une grande puissance de calcul. Cette puissance qui se compte généralement en millions d'opérations par seconde est réservée aux gros ordinateurs.

Ces ordinateurs dont *l'universalité* est généralement mal adaptée et dont le coût est très élevé se trouvent remplacés par des processeurs spécialisés.

Notre objectif dans ce projet de fin d'études est la réalisation d'un processeur rapide spécialisé, orienté vers la résolution des algorithmes de la FFT.

Notre projet s'articule autour de cinq chapitres :

- Le premier chapitre est consacré à l'étude de la transformée de Fourier discrète afin de faire apparaitre les domaines d'application.

Le deuxième chapitre est consacré à l'étude des différents algorithmes de FFT et le choix d'un algorithme repondant à une structure Hardware.

- Dans le troisième chapitre nous avons étudié les aspects algorithmiques et les principes du parallelisme et de la simultaneité.

- Le quatrième chapitre a été consacré à une étude comparative des différentes architectures que peut avoir ce processeur FFT et à la présentation de la solution choisie.

- Le cinqième et dernier chapitre est reservé à la conception du processeur FFT et la réalisation de certains de ses modules.

CHAPITRE 1.

LA TRANSFORMEE DE FOURIER DISCRETE ET SES APPLICATIONS.

1.1. INTRODUCTION

De nombreux problèmes d'analyse temporelle et de traitement du signal sont liés à la transformée de Fourier et son calcul rapide. Dans la résolution de ces problèmes, la transformée de Fourier a un rôle important non seulement en tant que moyen d'analyse spectrale, mais aussi comme élément intermédiaire nécessaire pour l'obtention de certains paramètres qualitatifs tels que les densités spectrales de puissance, les densités spectrales naturelles, les corrélation, les cohérences, les réponses en fréquence, etc...

Cette remarquable méthode utilisée depuis longtemps implique une description complète du signal en fonction de deux caractéristiques : l'amplitude et la phase.

Il convient de préciser le type d'opérations qui permettent de relier entre les signaux temporel et fréquentiel d'une même information.

On distingue quatre sortes de transformations: [4]

- La transformation des signaux apériodiques à temps continu (figure1.1a)

Transformation directe : $X(f) = \int_{-\infty}^{+\infty} x(t) \cdot \exp(-j2\pi ft) \cdot dt$

Transformation inverse : $x(t) = \int_{-\infty}^{+\infty} X(f) \cdot \exp(+j2\pi ft) \cdot df$

- La transformation des signaux périodiques à temps continu(figure1.1b)

$$X(f) = \frac{1}{T_0} \int_{T_0} x(t) \cdot \exp(-j2\pi ft) \cdot dt$$

$$x(t) = \sum_{n=-\infty}^{\infty} X(n) \cdot \exp(+j2\pi n f_0 t)$$

- La transformation des signaux apériodiques à temps discret

(figure 1.1.C)
$$X(f) = \sum_{k=-\infty}^{\infty} x(k) \exp(-j2\pi fk)$$

$$x(t) = \int_{-1/2}^{1/2} X(f) \exp(+j2\pi fk) \cdot df$$

- La transformation des signaux périodiques à temps discret

(figure 1.1 d)
$$X(n) = \sum_{k} x(k) \exp(-j2\pi nk/N) \quad ; \quad n = 0; 1; \cdots; N-1$$

$$x(k) = \frac{1}{N} \sum_{n} X(n) \cdot \exp(+j2\pi nk/N) \quad ; \quad k = 0; 1; \cdots; N-1$$

Cette dernière transformation est à la base de la FFT, son étude

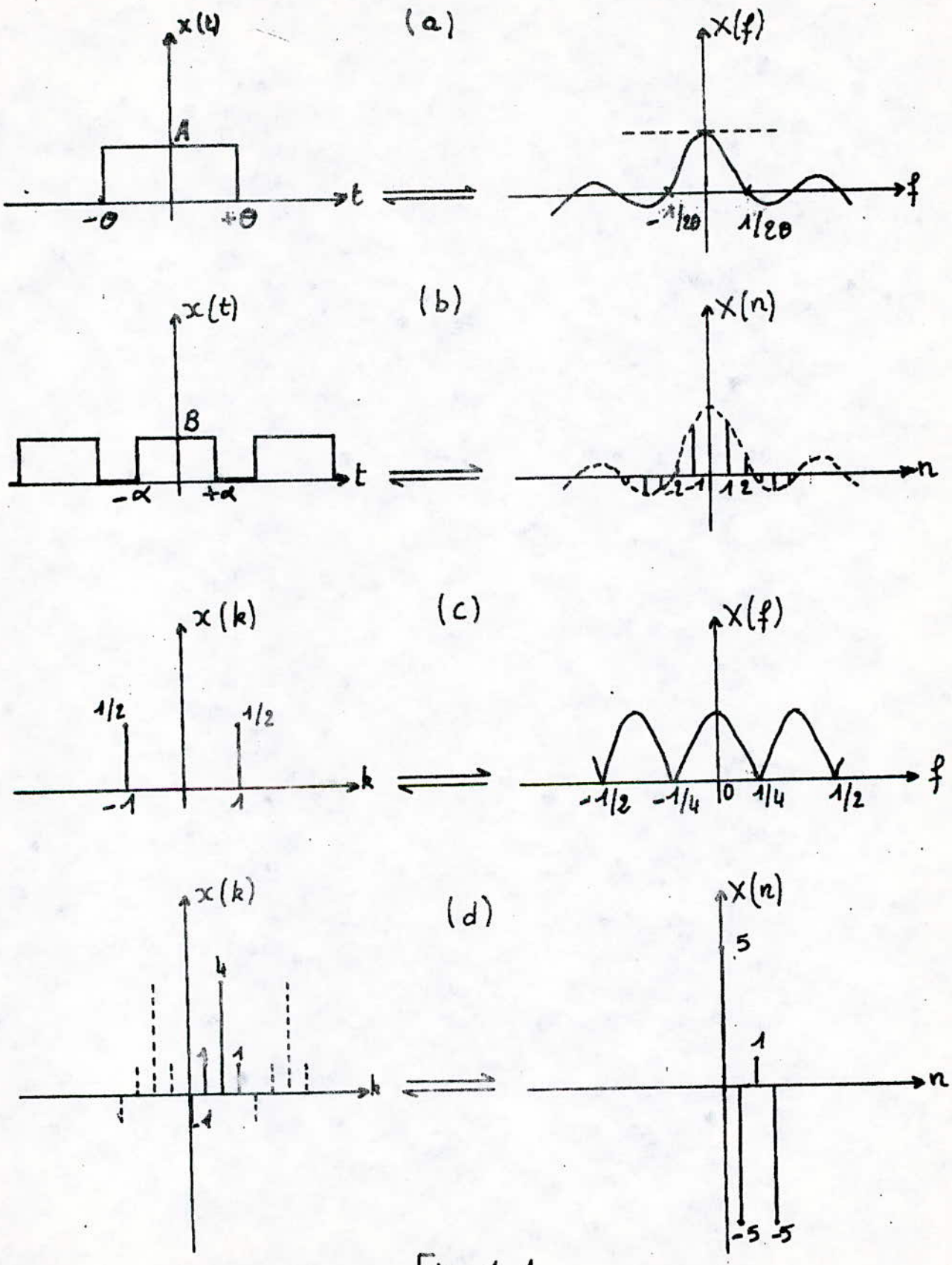

x(t)
(a)
A
t
-θ
+θ
X(f)
f
-1/2θ
1/2θ
x(t)
(b)
B
t
-α
+α
X(n)
n
-2
-1
1
2
x(k)
(c)
1/2
1/2
k
-1
1
X(f)
f
-1/2
-1/4
0
1/4
1/2
x(k)
(d)
4
1
k
-1
X(n)
5
1
n
-5
-5

Fig. 1.1

détailléesera reprise dans le paragraphe suivant.

1.2. ETUDE DE LA TRANSFORMEE DE FOURIER DISCRETE :(DFT)

La transformée de Fourier discrète permet de calculer le spectre d'une suite de valeurs ou précisement le spectre d'une séquence d'impulsions de Dirac dont le poids correspond aux valeurs numériques des éléments de la suite x(t)

La DFT possède les caractères fondamentaux suivants :

- La suite x (k) est périodique finie.
- Son spectre est également une suite périodique finie
- Les périodes de la suite x (k) et les périodes de son spectre contiennent toutes le même nombre d'échontillons.

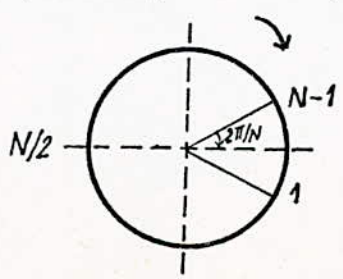
Sa relation de définition est la suivante :

$$X(n) = \sum_{n=0}^{N-1} x(k) \cdot \exp(-j 2\pi nk/N) \qquad n = 0; 1; \cdots; N-1 \qquad (1.1)$$

En posant exp ($-j2\pi/N$) = W, et en détaillant la relation (1.1), on obtient X (n) sous la forme matricielle :

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & \cdots & W^0 \\ W^0 & W^1 & \cdots & W^{N-1} \\ & & & \\ W^0 & W^{N-1} & \cdots & W^{(N-1)^2} \end{bmatrix} \cdot \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

La DFT est donc une transformation qui, à un vecteur x défini dans le domaine temporel, fait correspondre un vecteur X dans le domaine fréquentiel, l'opérateur de la transformation étant la matrice W qui est visiblement une matrice carrée du type N x N et présentant des particularités telles que les lignes et les colones de même indice ont les mêmes éléments. Ces éléments sont de la forme $W^i = \exp(-j2\pi i/N)$ tel que $W^N = 1$ ce sont donc les racines $N^{ième}$ de l'unité et leur image sont les points qui divisent le plan complexe en N parties égales.



1.3. LA TRANSFORMEE DE FOURIER DISCRETE BIDIMENSIONNELLE : (DFTB)

L'acquisition de l'information dans de nombreux domaines fait de plus en plus appel à l'emploi des images, c'est à dire des signaux à deux dimensions. Comme exemple on peut citer : les photos aériennes, les figures de la télévision, les images radiologiques, etc...

Pour certains traitements (transformation globale) de ces images on utilise la transformation de Fourier à deux dimensions dont nous allons exposer les principes.

Soit une suite à deux dimensions $\{ x(k,l) \}$ satisfaisant les préscriptions suivantes :

- x (k,l) est une fonction périodique de la variable discrète k, sa période est N.
- x (k,l) est une fonction périodique de la variable discrète l, sa période est M.

donc $x(k,l) = x(k + rN , l + sM)$; r et s entiers.

- $\{x (k,l)\}$ est représentée par une matrice du type N x M.

$$\{x(k,l)\} = \begin{bmatrix} x(0,0) & x(0,1) \cdots\cdots & x(0,M-1) \\ x(1,0) & x(1,1) \cdots\cdots & x(1,M-1) \\ \vdots & \vdots & \vdots \\ x(N-1,0) & x(N-1,1) \cdots\cdots & x(N-1,M-1) \end{bmatrix}$$

La DFT d'une telle suite satisfait ces même préscriptions.

Relation de définition de la DFTB :

$$\{X(n,m)\} = \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} x(k,l) \exp\left[-j2\pi(ml/M + nk/N)\right] \qquad (1.2)$$

posons $W_N^{nk} = \exp(-j2\pi nk/N)$ et $W_M^{ml} = \exp(-j2\pi ml/M)$

la relation (1.2) se réecrit :

$$X(n,m) = \sum_k \sum_l x(k,l) \cdot W_N^{nk} \cdot W_M^{ml} \qquad (1.2')$$

nous obtenons ainsi une forme *bilinéaire* à deux dimensions, or on sait qu'une forme *bilinéaire* peut se mettre sous la forme :

$$\sum_i \sum_j a_{ij} x_i y_j = \vec{x}^T [a] \vec{y}$$

x^T étant la transposée de x

La relation (1-2') peut donc s'écrire : $X(n,m) = \left(W_N^{nk}\right)^T \{x(k,l)\} W_M^{ml}$

d'après les proprietés de la matrice W ; $(W_N^{nk})^T = W_N^{nk}$ donc :

$$\{X(n,m)\} = (W_N^{nk}) \cdot \{x(k,\ell)\} W_M^{m\ell} \qquad (1.3)$$

L'analyse de l'expression (1.3) montre que, faire une transformée bidimensionnelle revient à faire deux transformées unidimensionnelles.

1.4 APPLICATIONSDE LA TRANSFORMEE DE FOURIER DISCRETE :

L'application la plus importante de la TFD est l'analyse spectrale dont les principales fonctions sont : le spectre de puissance, le spectre croisé la fonction de cohérence, la réponse en fréquence, etc...

1.4.1. SPECTRE DE PUISSANCE ET SPECTRE DE PUISSANCE CROISE ;

Les méthodes d'analyse spectrale [8] nous permettent d'analyser et interpreter la répartition de puissance pour un signal unique et, la répartition de la puissance mutuelle (ou croisée) pour deux signaux, en fonction de la fréquence.

Examinons brièvement les différents éléments de l'analyse spectrale de puissance et leur réalisation numérique.

Supposons que nous disposons d'un signal x (t), étant donné que nous allons l'analyser à l'aide d'un système numérique, il devra être échontillonné, cet échontillonnage doit être complété par un filtrage pour éviter les effets de deploiement qui ferait qu'à l'échontillonnage, les composantes à hautes fréquences du signal seraient indiscernables des composantes basses fréquences. Après cette opération une saisie des blocs de données (segmente les données) en blocs séquentiels. L'application d'une fenêtre au bloc de données élimine les effets d'extremité découlant de l'échontillonnage.

L'opération suivante est la transformée de Fourier qui transpose dans le domaine fréquentiel le signal vu à travers la fenêtre. Ainsi pour le $n^{ième}$ bloc, le spectre X_n (f) représentera la puissance spectrale contenue dans le bloc de données auquel la fenêtre a été appliquée. Le spectre du bloc est ensuite détecté ce qui signifie que l'on prend la somme des carrés des parties réelle et imaginaire pour calculer l'amplitude carrée du spectre du bloc $|X(f)|^2$. Finalement un processus de moyennage est entrepris pour éliminer les incertitudes statistiques.

Les opérations decrites précédement sont regroupéessur la figure 1.2

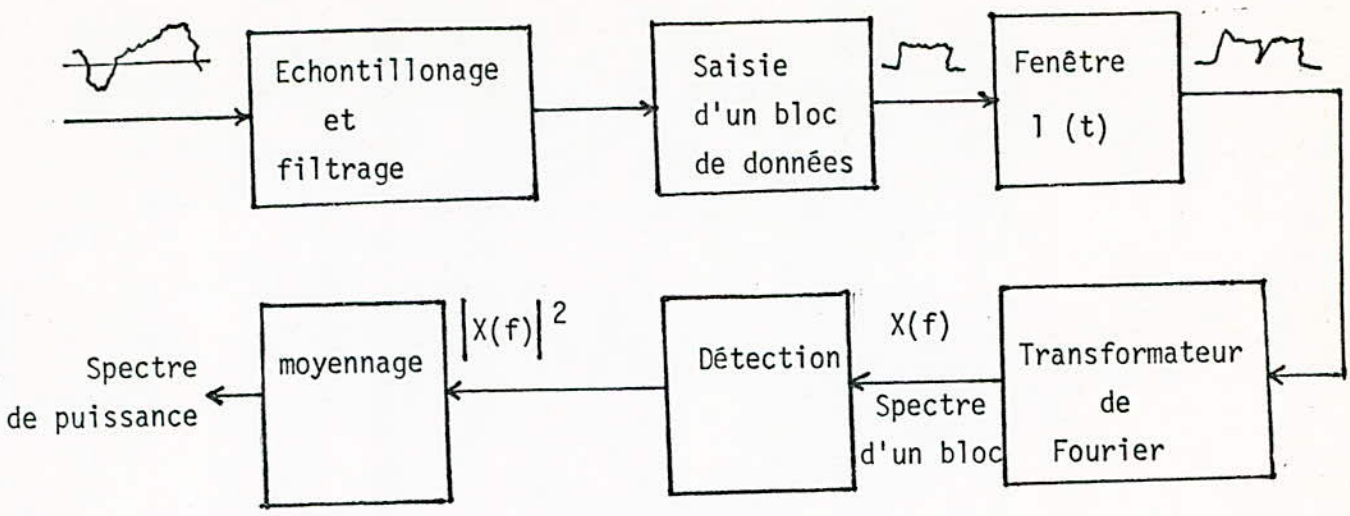


fig. 1.2.

Le spectre de puissance croisé (ou interspectre) $S_{xy}(f)$ de deux signaux x (t) et y (t) est obtenu en prenant le spectre de chacun d'eux et en effectuant le produit suivant la relation

$$S_{xy}(f) = X(f) \cdot Y^{*}(f)$$

où X (f) est le spectre de x (t) et $Y^{*}(f)$ est le spectre conjugué de y (t)

1.4.2 LA CORRELATION

L'intercorrelation entre deux signaux est une mesure de leur similarité en fonction du décalage temporel qui leur est imposé.

De même l'autocorrelation permet de comparer un signal temporel avec lui même en fonction d'un décalage τ . L'autocorrelation est très utile pour extraire un signal noyé dans un bruit.

L'intercorrelation est utilisée pour :

- Détecter les échos radar
- Détecter des objets
- Déterminer la forme d'un signal inconnu.
- Témoigner de l'éxistence possible d'anomalies fonctionnelles dans le domaine médical, en cardiologie et en encéphalographie par exemple

La relation de définition de la correlation de deux suites (signaux échontillonnés) est la suivante :

$$r_{xy}(k) = \{x(k)\} * \{y(k)\} = \sum_{j=0}^{N-1} x(j) \cdot y(j-k)$$

La transformée de Fourier de la suite $r_{xy}(k)$ est :

$$R(n) = \sum_k r_{xy}(k)\, W_N^{nk}$$

$$R(n) = \sum_k \left[\sum_j x(j) \cdot y(j-k) \right] W_N^{nk}$$

$$R(n) = \sum_k \sum_j x(j) \cdot y(j-k) \cdot W_N^{-n(j-k)} \cdot W_N^{nj}$$

$$R(n) = \left[\sum_j x(j) \cdot W_N^{nj} \right] \cdot \left[\sum_{j-k} y(j-k)\, W_N^{-n(j-k)} \right]$$

$$R(n) = X(n) \cdot Y^*(n)$$

donc $$r_{xy}(k) = F^{-1}\left[X(n) \cdot Y^*(n) \right]$$

On peut en conclure que la transformée de Fourier permet de calculer les suites de correlation en passant par leur spectre.

Les calculs ainsi conduits sont beaucoup plus rapides que l'application directe de la relation de définition.

1.4.3. LA CONVOLUTION :

L'opération de convolution est une opération qui revient très fréquement dans le traitement des signaux. Elle décrit le comportement des systèmes linéaires invariants dans le temps, entièrement caractérisés par leur réponse impulsionnelle.

La relation de définition de la convolution de deux suites est :

$$c_{xy}(k) = \sum_j x(j) \cdot y(k-j)$$

La transformées de Fourier de $c_{xy}(k)$ est :

$$C(n) = \sum_k c_{xy}(k) \cdot W_N^{nk}$$

$$C(n) = \sum_k \left[\sum_j x(j) \cdot y(k-j) \right] W_N^{nk}$$

$$C(n) = \sum_k \sum_j x(j) \cdot y(k-j) \cdot W_N^{n(k-j)} \cdot W_N^{nj}$$

$$C(n) = \left[\sum_j x(j) \cdot W_N^{nj} \right] \cdot \left[\sum_{k-j} y(k-j) \cdot W_N^{n(k-j)} \right]$$

$$C(n) = X(n) \cdot Y(n)$$

donc. $$c_{xy}(k) = F^{-1}\left[X(n) \cdot Y(n) \right]$$

1.4.4. LA FONCTION DE TRANSFERT.

La fonction de transfert est l'équivalent de la réponse impulsionnelle dans le domaine des fréquences. Elle décrit en effet le rapport éxistant entre les signaux sortie entrée d'un système, par exemple,

dans le nystagnus provoqué par stimulation vestibulaire [5], elle décrit le rapport entre le signal vestibulaire afférent et le mouvement des jeux.

La fonction de transfert d'un système est calculée à partir de l'auto et l'interspectre de deux signaux x(t) et y (t)

$$H(f) = \frac{S_{xy}(f)}{S_{xx}(f)}$$

avec $S_{xy}(f) = S_y(f) \cdot S_x^*(f)$; l'interspectre

$S_{xx}(f) = S_x(f) \cdot S_x^*(f)$; l'autospectre

1.4.5. LA COHERENCE :

La fonction de coherence carrée est un outil extrêmement util lorsque l'on veut déterminer les intéractions entre voies d'un système dynamique.

La coherence γ^2 est définie en termes de l'interspectre S_{xy} divisé mar le spectre de puissance du signal en entrée S_{xx} et le spectre de puissance du signal en sortie S_{yy}. C'est en quelque sorte une mesure normalisée du coefficient de correlation entre deux signaux x (t) et y (t) mais exprimée dans le domaine fréquentiel.

Bien qu'il n'y ait pas de correspondance entre elle, la fonction de coherence est dans le domaine des fréquences ce que la fonction d'intercorrelation est dans le domaine temporel.

La connaissance de la fonction de coherence est interéssante dans deux grands domaines : les mesures de fonction de transfert et la détermination de la causalité dans certains cas.

Sa formulation mathémqtique est la suivante :

$$\gamma_{xy}^2(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f) \cdot S_{yy}(f)}$$

1.5. CONCLUSION :

L'étude des applications de la DFT n'étant pas l'objet de ce projet, nous avons cité uniquement quelques domaines pour montrer les intérêts de la DFT.

Ces applications connues depuis des années n'ont pu voir le jour à cause du problème de la rapidité de calcul de la DFT jusqu'à 1965 lorsque cooley et Turkey lui conférèrent le caractère pratique qui lui manquait en introduisant un algorithme permettant le calcul rapide de la DFT. Quand la DFT est calculée à l'aide de tels algorithmes on dit que l'on effectue une transformation de Fourier rapide que l'on note FFT (Fast Fourier Transform) et qui fera l'objet du prochain chapitre.

CHAPITRE 2.

LES ALGORITHMES DE FFT ET LEURS PROPRIETES :

2.1. INTRODUCTION

Ces dernières années on assiste à un développement considérable des méthodes numériques de traitement du signal. Ces méthodes ont trouvé un très large champ d'applications : Sonar, radar, sysmologie, holographie, traitement de la parole et de l'image, géophysique, spectrometrie, detection des bruits, etc...

Cet important eventail d'applications est du d'une part au développement des recherches fondamentales et appliquées dans le domaine de la théorie du signal et d'autre part à l'apparition de puissants outils informatiques : circuits intégrés du type LSI et VLSI, moyens algorithmiques et structurels etc...

L'apparition de l'algorithme de la transformée rapide de Fourier (FFT) peut être considéréecomme étant un tournant de très grande importance dans le développement de traitement du signal. Malgrè l'utilisation, aujourd'hui d'autres systèmes de fonctions Orthogonales, la transformée de Fourier demeure un outil irremplaçable dans les méthodes numériques de traitement du signal. Pour confirmer ceci, on peut souligner le fait que durant la période allant de 1971 à 1981 furent remis plus de 63 brevets de decouvertes et d'innovation des méthodes de réalisation et d'application des algorithmes de la FFT dans l'analyse spectrale. Dans ces patents et brevets furent proposés plusieurs méthodes de choix de l'algorithme " optimal ". La notion d' " optimal " ne signifie guère l'éxistance d'un algorithme universel valable pour tous les cas de réalisation et d'applications, mais plutôt, il faudra comprendre le choix d'un algorithme (dans ce cas optimal) qui lors de sa réalisation, répond aux éxigences antagonistes que sont la vitesse, l'efficacité et la précision de traitement.

Pour la déduction et la représentation des algorithmes de FFT sont

utilisés trois formes symboliques :

- Le système de représentation algébrique, proposé par cooley est utilisé pour l'obtention d'équations récursives simples, qui s'averent très commodes pour la mise au point de programmes et l'étude des erreurs

- Le système graphique est très utilisé pour une représentation visuelle de l'algorithme de FFT. La première représentation graphique fût proposée par Rabiner, Gold et Rader [9] . Seulement cette représentation a ses limites qui se traduisent par :

a) Un nombre réduit d'échantillons

b) Elle est valable seulement pour les cas radix 2 ou 4

- Le système matriciel est le plus adapté lorsqu'il s'agit de mettre au point une approche globale pour le choix de l'algorithme" optimal " de la FFT et la mise au point des éléments de contrôle du processus de calcul.

2.2. RAPPEL SUR LES TRANSFORMATIONS MATRICIELLES

Les notions de produits élémentaires de KRONECKER et de matrices de transposition jouent un rôle essentiel lors de l'étude et la conception des différents algorithmes de la FFT.

Soit $E_{mi} = |\alpha(\varepsilon^i, \tau^i)|$ une matrice carrée d'ordre m_i ; $\alpha(\varepsilon^i, \tau^i)$ les éléments de cette matrice ; i l'indice de la matrice Em_i ; $\varepsilon^i, \tau^i =$ 0 ; 1 ; ... ; (m-1) les numéros de lignes et de colonnes de la matrice Em_i. Le produit élémentaire de KRONECKER des deux matrices Em_p et Em_q est déterminé par la formule :

$$Hm_p \cdot m_q = E_{m_p} \otimes E_{m_q} = |E_{m_p} \cdot \alpha(\varepsilon^q, \tau^q)| \qquad (2.1)$$

Donc lors de la formation de la matrice $Hm_p m_q$, chaque élément $\alpha(\varepsilon^q, \tau^q)$ de la matrice Em_q est multiplié par tous les éléments de la matrice Em_p . Les éléments de la matrice $H_{m_1 \cdot m_2 \cdot m_3 \ldots m_n}$ qui sont le résultat du produits de KRONECKER de la suite de matrice Em_1 , Em_2 , ... Em_n sont définis par la relation :

$$a(h, l) = \prod_{i=1}^{n} \alpha(\varepsilon^i, \tau^i) \qquad (2.2)$$

Eclaircissons le produit de KRONECKER par l'exemple suivant :

$$E_{2q} = \begin{bmatrix} \alpha_{11}^{(q)} & \alpha_{12}^{(q)} \\ \alpha_{21}^{(q)} & \alpha_{22}^{(q)} \end{bmatrix} \qquad E_{2p} = \begin{bmatrix} \alpha_{11}^{(p)} & \alpha_{12}^{(p)} \\ \alpha_{21}^{(p)} & \alpha_{22}^{(p)} \end{bmatrix}$$

Le produit matriciel est:

$$E_{2q} \cdot E_{2p} = \begin{bmatrix} \alpha_{11}^{(q)} \cdot \alpha_{11}^{(p)} + \alpha_{12}^{(q)} \cdot \alpha_{21}^{(p)} & \alpha_{11}^{(q)} \cdot \alpha_{12}^{(p)} + \alpha_{12}^{(q)} \cdot \alpha_{22}^{(p)} \\ \alpha_{21}^{(q)} \cdot \alpha_{11}^{(p)} + \alpha_{22}^{(q)} \cdot \alpha_{21}^{(p)} & \alpha_{21}^{(q)} \cdot \alpha_{12}^{(p)} + \alpha_{22}^{(q)} \cdot \alpha_{22}^{(p)} \end{bmatrix}$$

Le produit de Kronecker est:

$$E_{2q} \otimes E_{2p} = E_{2q} \otimes \begin{bmatrix} \alpha_{11}^{(p)} & \alpha_{12}^{(p)} \\ \alpha_{21}^{(p)} & \alpha_{22}^{(p)} \end{bmatrix}$$

$$E_{2q} \otimes E_{2p} = \begin{bmatrix} \begin{bmatrix} \alpha_{11}^{(q)} & \alpha_{12}^{(q)} \\ \alpha_{21}^{(q)} & \alpha_{22}^{(q)} \end{bmatrix} \cdot \alpha_{11}^{(p)} & \begin{bmatrix} \alpha_{11}^{(q)} & \alpha_{12}^{(q)} \\ \alpha_{21}^{(q)} & \alpha_{22}^{(q)} \end{bmatrix} \alpha_{12}^{(p)} \\ \begin{bmatrix} \alpha_{11}^{(q)} & \alpha_{12}^{(q)} \\ \alpha_{21}^{(q)} & \alpha_{22}^{(q)} \end{bmatrix} \cdot \alpha_{21}^{(p)} & \begin{bmatrix} \alpha_{11}^{(q)} & \alpha_{12}^{(q)} \\ \alpha_{21}^{(q)} & \alpha_{12}^{(q)} \end{bmatrix} \alpha_{22}^{(p)} \end{bmatrix}$$

$$E_{2q} \otimes E_{2p} = \begin{bmatrix} \alpha_{11}^{(q)} \cdot \alpha_{11}^{(p)} & \alpha_{12}^{(q)} \cdot \alpha_{11}^{(p)} & \alpha_{11}^{(q)} \cdot \alpha_{12}^{(p)} & \alpha_{12}^{(q)} \cdot \alpha_{12}^{(p)} \\ \alpha_{21}^{(q)} \cdot \alpha_{11}^{(p)} & \alpha_{22}^{(q)} \cdot \alpha_{11}^{(p)} & \alpha_{21}^{(q)} \cdot \alpha_{12}^{(p)} & \alpha_{22}^{(q)} \cdot \alpha_{12}^{(p)} \\ \alpha_{11}^{(q)} \cdot \alpha_{21}^{(p)} & \alpha_{12}^{(q)} \cdot \alpha_{21}^{(p)} & \alpha_{11}^{(q)} \cdot \alpha_{22}^{(p)} & \alpha_{12}^{(q)} \cdot \alpha_{22}^{(p)} \\ \alpha_{21}^{(q)} \alpha_{21}^{(p)} & \alpha_{22}^{(q)} \cdot \alpha_{21}^{(p)} & \alpha_{21}^{(q)} \cdot \alpha_{22}^{(p)} & \alpha_{12}^{(q)} \cdot \alpha_{22}^{(p)} \end{bmatrix}$$

Propriétés des produits élémentaires de KRONECKER :

$$\left(E_{m_1} + E_{m_2}\right) \otimes E_{m_3} = E_{m_1} \otimes E_{m_3} + E_{m_2} \otimes E_{m_3} \quad ; \tag{2.3}$$

$$E_{m_1} \otimes \left(E_{m_2} \otimes E_{m_3}\right) = \left(E_{m_1} \otimes E_{m_2}\right) \otimes E_{m_3} \quad ; \tag{2.4}$$

$$\left(E_{m_1} \otimes E_{m_2}\right) \cdot \left(E_{m_3} \otimes E_{m_4}\right) = E_{m_1} \cdot E_{m_3} \otimes E_{m_2} \cdot E_{m_4} \quad ; \tag{2.5}$$

$$\left(E_{m_1} \cdot E_{m_2} \cdots E_{m_n}\right) \otimes I = \left(E_{m_1} \otimes I\right) \cdot \left(E_{m_2} \otimes I\right) \cdots \left(E_{m_n} \otimes I\right) \quad ; \tag{2.6}$$

$$R_N \cdot \left(E_{N/2} \otimes I_2\right) R_N = I_2 \otimes E_{N/2} \quad ; \tag{2.7}$$

La matrice R_N de transposition qu'on appelle aussi matrice d'inversion numérique ou de désembrouillage est une matrice carrée où chaque ligne et chaque colonne ne contient qu'un seul " 1 ", les autres éléments sont des " 0 ".

Le produit d'une matrice d'inversion R_N avec une matrice E_N à droite ou à gauche conduit à la transposition des colonnes ou des lignes de E_N . Autrement dit la multiplication de E_N par R_N équivaut à l'éxécution des opérations suivantes :

1- Numéroter les lignes ou les colonnes de la matrice E_N de 0 à N-1

2- Utiliser pour la numérotation des lignes et des colonnes une représentation binaire.

3- Inverser l'ordre des chiffres pour chaque numéro de ligne ou de colonne.

Par exemple pour un nombre $h = h^{(n-1)} \cdot 2^{n-1} + \cdots + h^{(o)} \cdot 2^o$.

on aura $< h > = h^{(o)} \cdot 2^{n-1} + \cdots + h^{(n-1)} \cdot 2^o$.

ou $< h >$ est l'inversé numérique de h

Le nombre $< h >$ obtenu définit le nouveau numéro de la ligne ou de la colonne correspondante de la matrice résultante du produit E_N par R_N

De la définition de R_N on peut déduire que :

$$R_N \cdot R_N = I_N \tag{2.8}$$

$$\left(R_N\right)^T = R_N \tag{2.9}$$

Où I représente la matrice unitaire et T le symbole de la transposition des matrices :

Dans le processus des déductions des algoritmes de FFT nous utiliserons les propriétés suivantes de l'inversion numérique :

1- $\forall h \in \{0,1,\ldots,(N/2-1)\}$, $\exists \langle h \rangle$ et $\langle h \rangle$ nombre pair (2.10)

2- $\forall m \in \{1,2,\ldots\}$, $\exists N = 2^m$ et $\langle N/2 \rangle = 1$ (2.11)

3- $\forall h \in \{0,1,\ldots,N/2-1\}$, $\forall m \in \{1,2,\ldots\}$ $\exists N = 2^m$

et $\langle h + N/2 \rangle = \langle h \rangle + 1$ (2.12)

2.3. LES ALGORITHMES DE FFT.

La majorité des algorithmes de FFT peut être regroupée en quatre classes. Schématiquement, ces classes sont illustrées par la figure 2.1. [2]

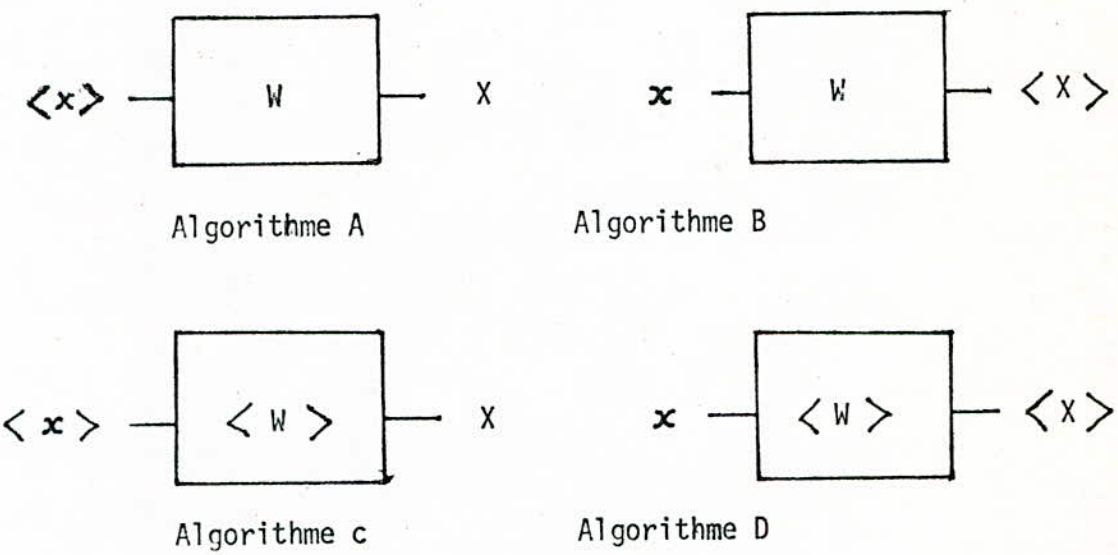


fig.2.1.

2.3.1. L'ALGORITHME A DE LA FFT :

Si nous supposons que $\mathbf{x}(k) / k \in \{0, 1, \ldots, N-1\}$ et $X(n) / n \in \{0, 1, \ldots, N-1\}$ sont respectivement les échontillons du signal à traiter et les coefficients de FOURIER et que les éléments de la matrice $|def(n, k)|$ sont définis par l'expression :

$$def(n,k) = \exp -j(2\pi/N)\cdot n\cdot k$$

la représentation matricielle de la transformée discrète de Fourier prend la forme suivante :

$$X = \frac{1}{N} \left| def(n,k) \right|_N \cdot \mathbf{x} \qquad (2.13)$$

où $\mathbf{x} = \text{col}\left(\mathbf{x}(o), \mathbf{x}(1), \ldots, \mathbf{x}(N-1)\right)$ et X = col (X(o), X(1),...,X(N-1))

En tenant compte de (2.8) la dernière expression peut s'écrire :

$$X = \frac{1}{N} \cdot \left| def(n,k) \right|_N \cdot R_N \cdot R_N \cdot x$$

$$X = \frac{1}{N} \cdot G_N(n,q) \cdot R_N \cdot x \qquad (2.14)$$

G_N (n,q) est une matrice carrée obtenue à partir d'un ordonnancement à inversion numérique des colonnes de la matrice $\left| \text{def } (n,k) \right|_N$. Elle peut être définie comme un ensemble d'éléments dont les coordonnées (n,q) peuvent satisfaire aux conditions $n \in [o \div N-1]$ et $q \in [o \div N-1]$ c'est à dire :

$$G_N(n,q) = \left\{ [n,q] / n \in [o \div N-1],\ q \in [o \div N-1];\ [n,q] = def(n, <q>) \right\} \ ; \qquad (2.15)$$

Dans le but de décomposer G_N (n,q) en matrices de plus petit ordre on écrit :

$$G_N(n,q) = \left[\begin{array}{c|c} G_{N/2}^{(1)} & G_{N/2}^{(2)} \\ \hline G_{N/2}^{(3)} & G_{N/2}^{(4)} \end{array}\right] \qquad (2.16)$$

Déterminons la relati on qui existe entre les sous matrices $G_{N/2}^{(1)}$; $G_{N/2}^{(2)}$; $G_{N/2}^{(3)}$, $G_{N/2}^{(4)}$ et la matrice $G_{N/2}$. Cette dernière peut être définie à partir de (2.15) comme :

$$G_{N/2}(n,q) = \left\{ [n,q] / n \in [o \div N/2 -1],\ q \in [o \div N/2 -1],\ [n,q] = def(n, <q>) \right\}$$

avec $[n,q] = def(n, <q>) = exp -j2\pi n \cdot <q> / (N/2)$; (2.17)

D'après l'expression (2.16), la sous matrice $G_{N/2}^{(1)}$ peut être écrite de la façon suivante :

$$G_{N/2}^{(1)} = \left\{ [n,q]^{(1)} / n \in [o \div N/2 -1],\ q \in [o \div N/2 -1],\ [n,q]^{(1)} = def(n, <q>) \right\} \qquad (2.18)$$

et par conséquent :

$$G_{N/2}^{(1)} = G_{N/2} \qquad (2.19)$$

La sous matrice $G_{N/2}^{(2)}$ peut être définie de la façon suivante :

$$G_{N/2}^{(2)} = \left\{ [n,q]^{(2)} / n \in \left[o \div \frac{N}{2} - 1\right]; q \in \left[\frac{N}{2} \div N-1\right], [n,q] = def(n, <q>) \right\} \qquad (2.20)$$

ou de la façon suivante :

$$G_{N/2}^{(2)} = \left\{ [n, q+N/2]^{(2)} / n \in [o \div N/2 -1],\ q \in [o \div N/2 -1];\ [n, q+N/2]^{(2)} = def(n, <q+N/2>) \right\} \qquad (2.21)$$

Si on tient compte des propriétés de l'inversion numérique (2.10), (2.11) et (2.12) ainsi que de (2.17).

$$[n, q+N/2]^{(2)} = def(n, <q+N/2>) = def(n, <q>) \cdot def(n,1) \qquad (2.22)$$

Pour faciliter la compréhension, nous supposerons que def (n,1)= w^n

où suivant (2.17) :

$$w = \exp -j2\pi / (N/2)$$

Introduisons la matrice diagonale $F_1 = (W^\circ, w^1, w^2, \ldots, w^{N/2-1})$ et utilisons l'écriture symbolique n à la place de w^n, alors $F_1 = (0, 1, 2, \ldots, N/2-1)$. En tenant compte des relations introduites on a:

$$G^{(2)}_{N/2} = G_{N/2} \cdot F_1 \qquad (2.23)$$

De (2.16) on déduit que :

$$G^{(3)}_{\frac{N}{2}} = \left\{ [n,q]^{(3)} / n \in \left[\tfrac{N}{2} \div N-1\right], q \in \left[0 \div \tfrac{N}{2} - 1\right], [n,q]^{(3)} = def\,(n, \langle q \rangle) \right\}$$

ou bien

$$G^{(3)}_{N/2} = \left\{ \left[n+\tfrac{N}{2}, q\right]^{(3)} / n \in \left[0 \div \tfrac{N}{2}-1\right], q \in \left[0 \div \tfrac{N}{2}-1\right], \left[n+\tfrac{N}{2}, q\right]^{(3)} = def\left(n+\tfrac{N}{2}, \langle q \rangle\right) \right\}$$

Si on tient compte du fait que def (N/2 , $\langle q \rangle$) = 1 (car $\forall q \in \left[0, \frac{N}{2}-1\right]$, $\langle q \rangle$ nombre pair), on aura alors :

$$\left[n+\tfrac{N}{2}, q\right]^{(3)} = def\,(n, \langle q \rangle) \qquad (2.24)$$

En conséquence :

$$G^{(3)}_{\frac{N}{2}} = G_{\frac{N}{2}} \qquad (2.25$$

La sous matrice $G^{(4)}_{\frac{N}{2}}$ peut être définie à partir de (2.17) de la façon suivante :

$$G^{(4)}_{N/2} = \left\{ [n,q]^{(4)} / n \in \left[\tfrac{N}{2} \div N-1\right], q \in \left[N/2 \div N-1\right], [n,q]^{(4)} = def\,(n,q) \right\}$$

ou de la façon suivante :

$$G^{(4)}_{N/2} = \left\{ \left[n+\tfrac{N}{2}, q+\tfrac{N}{2}\right] / n \in \left[0 \div \tfrac{N}{2}-1\right], q \in \left[0 \div \tfrac{N}{2}-1\right]; \left[n+\tfrac{N}{2}, q+\tfrac{N}{2}\right] = def\left[n+\tfrac{N}{2}, \langle q+\tfrac{N}{2} \rangle\right] \right\}$$

Les valeurs $\left[n+\frac{N}{2}, \langle q+\frac{N}{2} \rangle\right]^{(4)}$ des éléments $\left(n+\frac{N}{2}, q+\frac{N}{2}\right)^{(4)}$ de la matrice $G^{(4)}_{N/2}$ sont déterminées par la relation :

$$\left[n+\tfrac{N}{2}, q+\tfrac{N}{2}\right]^{(4)} = -\,def\,(n, \langle q \rangle) \cdot def\,(n,1)$$

En conséquence si on tient compte de (2.23), on aura :

$$G^{(4)}_{N/2} = -\,G_{N/2} \cdot F_1 \qquad (2.26)$$

D'où la matrice $G_N(n,q)$ à partir de (2.17, (2.19), (2.23), (2.25) et (2.26).

$$G_N(n,q) = \left|\begin{array}{c|c} G_{\frac{N}{2}} & G_{\frac{N}{2}} \cdot F_1 \\ \hline G_{\frac{N}{2}} & -G_{\frac{N}{2}} F_1 \end{array}\right|$$

Cette matrice peut être factorisée de la façon suivante :

$$\left|\begin{array}{c|c} G_{\frac{N}{2}} & G_{\frac{N}{2}} \cdot F_1 \\ \hline G_{\frac{N}{2}} & -G_{\frac{N}{2}} \cdot F_1 \end{array}\right| \equiv \left|\begin{array}{c|c} I_{N/2} & I_{N/2} \\ \hline I_{N/2} & -I_{\frac{N}{2}} \end{array}\right| \cdot \left|\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right|$$

Déterminons les valeurs A, B, C et D. Pour cela on doit effectuer l'opération de multiplication des deux matrices de droite.

On obtient alors :

$$\left|\begin{array}{c|c} G_{\frac{N}{2}} & G_{\frac{N}{2}} \cdot F_1 \\ \hline G_{\frac{N}{2}} & -G_{\frac{N}{2}} \cdot F_1 \end{array}\right| \equiv \left|\begin{array}{c|c} A+C & B+D \\ \hline A-C & B-D \end{array}\right|$$

Effectuant une identification par blocs et déterminons les valeurs de A, B, C et D.

$$\begin{aligned} A + C &= G_{\frac{N}{2}} \\ B + D &= G_{\frac{N}{2}} \cdot F_1 \\ A - C &= G_{N/2} \\ B - D &= -G_{\frac{N}{2}} \cdot F_1 \end{aligned} \Longrightarrow \begin{cases} A = G_{\frac{N}{2}} \\ C = |0| \\ B = |0| \\ D = G_{\frac{N}{2}} \cdot F_1 \end{cases}$$

D'où :

$$G_N(n,q) = \left|\begin{array}{c|c} I_{N/2} & I_{N/2} \\ \hline I_{N/2} & -I_{N/2} \end{array}\right| \cdot \left|\begin{array}{c|c} G_{N/2} & 0 \\ \hline 0 & G_{\frac{N}{2}} \cdot F_1 \end{array}\right|$$

La logique de raisonnement nous amène à décomposer la deuxième matrice en deux matrices dont l'une est de la forme :

$$\left|\begin{array}{c|c} G_{\frac{N}{2}} & 0 \\ \hline 0 & G_{\frac{N}{2}} \end{array}\right|$$

donc l'autre doit être de la forme suivante :

$$\left|\begin{array}{c|c} I_{N/2} & 0 \\ \hline 0 & F_1 \end{array}\right|$$

d'où $$G_N(n,q) = \left|\begin{array}{c|c} I_{N/2} & I_{N/2} \\ \hline I_{N/2} & -I_{N/2} \end{array}\right| \cdot \left|\begin{array}{c|c} I_{N/2} & 0 \\ \hline 0 & F_1 \end{array}\right| \cdot \left|\begin{array}{c|c} G_{N/2} & 0 \\ \hline 0 & G_{N/2} \end{array}\right|$$

si à travers D_N on définit la matrice quasidiagonale $(I_{N/2}, F_1)$ et on utilise l'écriture symbolique de KRONECKER on aura :

$$G_N(n,q) = \left(I_{N/2} \otimes G_2\right) \cdot D_N \cdot \left(G_{\frac{N}{2}} \otimes I_2\right) \qquad (2.27)$$

où $$G_2 = \left|\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right| \quad \text{et} \quad I_2 = \left|\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right|$$

Dans (2.27) $G_N(n,q)$ est exprimée à travers $G_{N/2}$. Si on refait la même procédure pour $G_{N/2}$ on aura :

$$G_{N/2}(n,q) = \left(I_{N/4} \otimes G_2\right) \cdot D_{N/2} \cdot \left(G_{\frac{N}{4}} \otimes I_2\right) \qquad (2.28)$$

Si dans (2.27), nous remplaçons $G_{N/2}$ par (2.28) on aura :

$$G_N(n,q) = \left(I_{N/2} \otimes G_2\right) \cdot D_N \cdot \left\{\left[\left(I_{\frac{N}{4}} \otimes G_2\right) \cdot D_{\frac{N}{2}} \cdot \left(G_{\frac{N}{4}} \otimes I_2\right)\right] \otimes I_2\right\}$$

Si on utilise la propriété de distribution (2.6), on aura :

$$G_N(n,q) = \left(I_{\frac{N}{2}} \otimes G_2\right) \cdot D_N \left\{\left[\left(I_{\frac{N}{4}} \otimes G_2\right) \otimes I_2\right]\left[D_{\frac{N}{2}} \otimes I_2\right]\left[\left(G_{\frac{N}{4}} \otimes I_2\right) \otimes I_2\right]\right\}$$

ou :

$$G_N(n,q) = \left(I_{\frac{N}{2}} \otimes G_2\right) \cdot D_N \cdot \left(I_{\frac{N}{4}} \otimes G_2 \otimes I_2\right)\left(D_{\frac{N}{2}} \otimes I_2\right)\left(G_{\frac{N}{4}} \otimes I_4\right)$$

Si on exprime $G_{N/4}$ à travers $G_{N/8}$, ensuite répéter cette opération tant que G_N ne soit exprimée complètement par G_2, on obtient alors pour $j = 3, 4, 5, \ldots, m-1$

$$G_N = \left(I_{\frac{N}{2}} \otimes G_2\right) D_N \cdots \left(I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}}\right)\left(D_{2^j} \otimes I_{2^{m-j}}\right) \cdots \left(I_2 \otimes G_2 \otimes I_{2^{m-2}}\right) \cdot \left(D_{2^2} \otimes I_{2^{m-2}}\right)\left(G_2 \otimes I_{2^{m-1}}\right)$$

(2.29)

où

$$D_{2^{m-j}} = \text{diag}\left(I_{2^{m-j-1}} , F_{2^j} \right)$$

$$F_{2^j} = \text{diag}\left(0, 2^j, 2.2^j, 3.2^j, \ldots, 2^{m-j}.2^j\right)$$

$$N = 2^m$$

C'est ainsi que l'expression (2.13), en tenant compte de (2.14) et (2.29) peut être écrite de la façon suivante :

$$X = \frac{1}{N}\left(I_{2^{m-1}} \otimes G_2\right).D_{2^m} \ldots\left(I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}}\right)\left(D_{2^j} \otimes I_{2^{m-j}}\right)\ldots$$

$$\ldots \left(I_2 \otimes G_2 \otimes I_{2^{m-2}}\right)\left(D_{2^2} \otimes I_{2^{m-2}}\right)\left(G_2 \otimes I_{2^{m-1}}\right)R.x \qquad (2.30)$$

Cette expression contient trois types de matrices

a) Les matrices de sommation algébrique qui ont pour forme $\left(I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}}\right)$. Dans chaque ligne de ces matrices on trouve seulement deux éléments différents de zéro. Ces matrices sont représentées par S(j).

b) Les matrices M(j) des coefficients de rotation composées d'éléments différents de zéro sur la diagonale. Ces coefficients déterminent l'opération de multiplication dans les algorithmes de FFT.

c) La matrice de l'inversion numérique R_N

Dans chaque itération (j) sont effectuées les opérations qui sont données par les matrices M(j) et S(j). Dans ce cas la réalisation des algorithmes de FFT s'effectue en $m = \log_2 N$ itérations en plus d'une itération pour effectuer l'inversion numérique

Les particularités de cet algorithme sont les suivants :

1) Les données d'entrée (échontillons) ; c'est à dire les éléments du vecteur R x ont un ordonnancement inversé.

2) Dans chaque itération l'opération de multiplication précède l'opération de sommation.

3) Lors de la première opération nous n'avons pas de multiplication à effectuer.

4) Les coefficients de Fourier et les coefficients de rotation ont

un ordonnancement direct.

Pour N = 16 ; m = 4

$$X = \frac{1}{16}(I_8 \otimes G_2) D_{16} (I_4 \otimes G_2 \otimes I_2)(D_8 \otimes I_2)(I_2 \otimes G_2 \otimes I_4)(D_4 \otimes I_4)(G_2 \otimes I_8) R.x \quad (2.30a)$$

2.3.2. L'ALGORITHME B DE LA FFT.

Comme l'illustre la figure 2.1 , cet algorithme a les particularités suivantes :

1- Les données d'entrée ont un ordonnancement direct.

2- Les coefficients de rotation ont eux aussi un ordonnancement direct

3- Les données de sorties (les coefficients de Fourier) sont inversées

Cet algorithme peut être déduit de deux façons :

i) Reprendre la même démarche que celle utilisée lors de la déduction de l'algorithme A. Seulement cette fois il s'agira de multiplier la matrice $|def(n,k)|_N$ à gauche tout en tenant compte de (2.8) c'est à dire

$$X = \frac{1}{N} R_N \cdot R_N |def(n,k)| . x \quad (2.31)$$

si on pose $R_N |def(n,k)|_N = G_N(h,k)$

on aura ; $$X = \frac{1}{N} \cdot R_N \cdot G_N (h.K)\ x \quad (2.32)$$

Où G_N (h,K) est une matrice carrée obtenue par un ordonnancement inversé des lignes de La matrice $|def(n,k)|$. Elle peut être définie comme un ensemble d'éléments dont les coordonnées sont (h,K) et les valeurs $[h,K] = def(\langle h \rangle, K) = def(n;K)$

En décomposant $G_N(h,K)$ en sous matrices d'ordre 2, l'équation (2.32) pourra s'écrire :

$$X = \frac{1}{N} \cdot R_N (G_2 \otimes I_{2^{m-1}})(D_{2^2} \otimes I_{2^{m-2}})(I_2 \otimes G_2 \otimes I_{2^{m-2}}) \ldots$$
$$\ldots (D_{2^j} \otimes I_{2^{m-j}})(I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}}) \ldots D_{2^m}(I_{2^{m-1}} \times G_2).X \quad (2.33)$$

De cette expression on peut déduire que l'algorithme de traitement contient les trois types de matrices R_N , S(j) et M(j) et qu'à la dernière

itération les opérations de multiplication sont énixistantes

ii) La deuxième approche consiste à faire la transposition de l'algorithme A tout en utilisant les propriétés des produits de KRONECKER de deux matrices.

2.3.3. L'ALGORITHME C DE LA FFT.

Ces algorithmes diffèrent des précédents par la distribution inversée des données d'entrée et des coefficients de rotation, alors que les coefficients de Fourier ont un ordonnancement direct.

En tenant compte de (2.7), nous remplaçons dans l'expression (2.33) les matrices S(j) par leurs équivalentes qu'on pourra définir de la façon suivante :

$$I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}} = R_N \left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}} \right) R_N$$

$$G_2 \otimes I_{2^{m-1}} = R_N \left(I_{2^{m-1}} \otimes G_2 \right) R_N \cdot$$

$$I_{2^{m-1}} \otimes G_2 = R_N \left(G_2 \otimes I_{2^{m-1}} \right) R_N$$

Dans ce cas on aura :

$$X = \frac{1}{N} R_N \left[R_N \left(I_{2^{m-1}} \otimes G_2 \right) R_N \right] \left(D_{2^2} \otimes I_{2^{m-2}} \right)$$
$$\left[R_N \left(I_{2^{m-2}} \otimes G_2 \otimes I_2 \right) R_N \right] \cdots \left(D_{2^j} \otimes I_{2^{m-j}} \right)$$
$$\left[R_N \left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}} \right) R_N \right] \cdots D_{2^m} \left[R_N \left(G_2 \otimes I_{2^{m-1}} \right) R_N \right] X$$

La matrice M(j) des coefficients de rotation est quasi-diagonale. Sa matrice inverse peut être déduite par un ordonnancement inversé des lignes et des colonnes, ce qui peut être réalisé par le produit de M(j) avec R_N à gauche et à droite. Si on tient compte de ce fait la dernière expression s'écrit comme suit :

$$X = \frac{1}{N} \left(I_{2^{m-1}} \otimes G_2 \right) \left[R_N \left(D_{2^2} \otimes I_{2^{m-2}} \right) R_N \right] \cdots \left[G_2 \otimes I_{2^{m-1}} \right] R_N X$$

2.3.4. LES ALGORITHMES D DE LA FFT.

Cette classe d'algorithmes peut être déduite de la même manière que l'algorithme C, seulement les propriétés de transposition seront appliquées aux matrices de sommation algébrique de l'algorithme A. Dans ce cas on aura :

$$\left(I_{2^{m-1}} \otimes G_{2^m}\right) = R_N \left(G_{2^m} \otimes I_{2^{m-1}}\right) R_N$$

$$I_{2^{j-1}} \otimes G_2 \otimes I_{2^{m-j}} = R_N \left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}}\right) R_N$$

et $$G_2 \otimes I_{2^{m-1}} = R_N \left(I_{2^{m-1}} \otimes G_2\right) R_N$$

L'expression (2.33) de l'algorithme A s'écrira sous la forme :

$$X = \frac{1}{N} \left[R_N \left(G_2 \otimes I_{2^{m-1}}\right) R_N\right] D_{2^m} \cdots\cdots \left[R_N \left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}}\right) R_N\right] \left(D_{2^j} \otimes I_{2^{m-j}}\right) \cdots \left[R_N \left(I_{2^{m-2}} \otimes G_2 \otimes I_2\right) R_N\right] \left(D_{2^2} \otimes I_{2^{m-2}}\right) \left[R_N \left(I_{2^{m-1}} \otimes G_2\right) R_N\right] R_N \cdot x$$

En utilisant les propriétés d'associativité des produits matricielles et la propriété (2.8) de KRONECKER, l'expression ci-dessus peut être ecrite de la façon suivante :

$$X = \frac{1}{N} R_N \left\{\left(G_2 \otimes I_{2^{m-j}}\right)\left[R_N \cdot D_{2^m} \cdot R_N\right]\left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}}\right) \left[R_N \left(D_{2^j} \otimes I_{2^{m-j}}\right) R_N\right] \cdots\cdots \left(I_{2^{m-2}} \otimes G_2 \otimes I_2\right)\left[R_N \left(D_{2^2} \otimes I_{2^{m-2}}\right) R \cdots\cdots \left(I_{2^{m-1}} \otimes G_2\right)\right\} x$$

En multipliant les deux nembres de cette égalité par la matrice R_N, on obtien l'expression, reflétant l'algorithme D.

$$R_N X = \langle X \rangle = \frac{1}{N} \left\{\left(G_2 \otimes I_{2^{m-1}}\right)\left[R_N D_{2^m} R_N\right]\left(I_{2^{m-j}} \otimes G_2 \otimes I_{2^{j-1}}\right) \left[R_N \left(D_{2^j} \otimes I_{2^{m-j}}\right) R_N\right] \cdots \left(I_{2^{m-2}} \otimes G_2 \otimes I_2\right)\left[R_N \left(D_{2^2} \otimes I_{2^{m-2}}\right) R_N\right] \left[I_{2^{m-1}} \otimes G_2\right]\right\} x$$

L'analyse de cette expression fait ressortir les particularité suivante de l'algorithme D.

1- L'ordonnancement des échontillons (x) est direct

2- Dans toutes les itérations l'opération de multiplication précède celle de l'addition (decimation dans le temps)

3- La première itération est composée que d'additions.

4- Les coefficients de FOURIER et de rotation ont un ordonnancement inversé.

CONCLUSION :

L'analyse comparative de ces quatre classes d'algorithme du point de vue vitesse d'éxécution, précision du resultat, régularité de la structure a montré que :

1- Tous les algorithmes (A, B, C, D) s'éffectuent en $\log_2 N$ itérations. Chaque itération comprend $\frac{N}{2}$ opérations de base.

2- Lors de l'utilisation de l'algorithme A en virgule fixe le rapport bruit/signal pour un signal d'entrée sous forme de bruit blanc et pour une normalisation automatique (décalage à droite d'1 bit à chaque itération) est défini par l'expression.

$$\frac{\sigma^2 \text{ de l'erreur}}{\sigma^2 \text{ du signal}} = \left(\frac{m}{6}\right) 2^{-2B}$$

σ^2 - erreur quadratique

B - le nombre de bits de représentation des nombres en virgule fixe.

Cette formule permet d'obtenir une évaluation approximative de de l'erreur, les mesures expérimentales [2] conformé l'accroissement lineaire de l'erreur par rapport au nombre d'itérations.

2- Les algorithmes A et D sont constitués d'opérations de base symétriques, ce qui facilite leur réalisation dans des processeurs spécialisés à structure régulière.

4- Les algorithmes B et C sont plus adaptés à une réalisation programmée ou micro programmée

CHAPITRE 3.

ASPECTS ALGORITHMIQUES ET STRUCTURELS DES SYSTEMES RAPIDES DE TRAITEMENT DE DONNEES.

3.1. INTRODUCTION :

Dans differents cas peut se poser le problème de devoir traiter des signaux en temps réels, par exemple pour effectuer des transformées de Fourier, des convolutions et des correlations sur des paquets consécutifs d'échantillons d'un signal.

Pour de telles applications, l'emploi d'un ordinateur universel en temps réel (" One line ") ne peut être envisagé que dans des cas très particuliers où la fréquence d'entrée des échantillons est très faible (cas des signaux sismiquespar exemple). En général, les ordinateurs ne disposent que d'une unité centrale et ne satisfont pas les exigences en vitesse requises par le traitement des signaux, qui veulent que le temps employé pour élaborer une seule ou un groupe de grandeurs soit plus petit (au moins en moyenne) que la période avec laquelle les données se présentent à l'entrée du système. Cette exigence en vitesse se traduit par la présence, dans les machines spécialisées pour le traitement en temps réel, de plusieurs unités arithmetiques identiques effectuant des blocs en parallele. Dé telles structures sont rencontrées dans les processeurs FFT, dans les systèmes de filtrage digital, dans les corrélateurs, etc...

La réalisation d'un système ayant une architecture permettant un traitement en temps réel nécessite une étude des aspects algoritmiques et structurels des systèmes rapides de traitement de données.

3.2. LES ASPECTS ALGORITHMIQUES :

Pour ce qui est des considérations algorithmiques, on peut définir dans une première approche trois types de parallelisme :

a) Le parallélisme " implicite " fonctionnel, par exemple au niveau d'opérateur d'une Structure MISD.

b) Le parallélisme " explicite ", qui correspond à l'ennoncé algorithmique :

Pour tout x Faire (x)

c) Le parallélisme " caché " dans un algorithme séquentiel.

Soit un programme W :

$$W = \emptyset_1 , \emptyset_2 ; \quad , \emptyset_i , \quad , \emptyset_n$$

les fragments $\emptyset_i$ ($\emptyset_i$ modifiant l'esemble des objets OM_i et lisant l'ensembles des objets OL_i) sont exécutables simultanément s'ils verifient les conditions de BERNSTEIN [1] du nom de celui qui les a utilisés le premier pour décrire les instructions indépendantes dans un programme FORTRAN. Les conditions sont :

$$\begin{aligned} &\text{Intersection} (OL_i , OM_j) = \text{vide} \\ &\text{Intersection} (OM_i , OL_j) = \text{vide} \qquad \forall i \neq j \\ &\text{Intersection} (OM_i , OM_j) = \text{vide} \end{aligned}$$

Cette approche de classification du parallélisme est faite sur la base des contraintes liées aux méthodes arithmétiques traditionnelles implantées sur les systèmes de traitement de données sans tenir compte de certains aspects d'éxécution des processus réels à caractères iteratif et séquentiel telle que la résolution des équations differentielles, intégrales, le calcul des transformées orthogonales, etc...

Les algorithmes de réalisation de processus iteratifs et de calcul des expressions algébriques à plusieurs opérandes peuvent avoir une structure soit séquentielle, soit parallelo-séquentielle. Dans le premier cas, l'algorithme serait une suite de fragments operationnels indécomposables sur lesquels est réalisée l'opération de superposition. Dans le deuxième cas l'algorithme renferme des branches paralleles.

L'éxécution de ce genre d'algorithmes par des méthodes arithmétiques classiques ne peut s'effectuer en un temps inferieur à la somme des temps de traitement des fragments séquentiels, car dans ces cas précis, il est impossible d'utiliser l'idée répandue de la décomposition des algorithmes en fragments indépendants du point de vu interaction.

3.3. LES ASPECTS STRUCTURELS :

L'évolution des aspects structurels nous a conduit à faire une

étude du parallelisme et de la simultaneité dans les systèmes de traitement de l'information.

3.3.1. ORGANISATION LINEAIRE.

C'est la plus simple de toutes les organisations, elle est caractérisée par le fait que l'unité mémoire (UM), l'unité de commande (UC) et l'unité de traitement (UT) travaillent en régime séquentiel et il n'y a pas de recouvrement dans leur fonctionnement. La figure 3.1 nous donne une représentation simplifiée d'une telle organisation. Pour comprendre le fonctionnement de ce système on se reportera au diagramme de la figure 3.2. Ce diagramme comporte le temps en abscisse et trois niveaux logiques en ordonnée, représentant :

- Le processus en mémoire.
- Le processus en unité de commande.
- Le processus en unité de traitement.

sur ce diagramme ont peut voir que l'organisation linéaire crée des temps morts à toutes les unités et surtout à l'unité de traitement. Le pourcentage entre ces temps varie suivant le type de pragramme exécuté. Ceci entraine une utilisation non éfficace des ressources que ce soit Hardware ou Software, d'où la nécéssité de l'organisation de la simultaneité dans le fonctionnement du système.

Nous allons nous interesser à cette notion du point de vu Hardware seulement, il ne sera pas difficile de faire le parallele avec le Software. En Software on dira qu'un ordinateur est capable de simultaneité s'il peut éxécuter deux instructions langage machine interne en même temps, sans que la durée d'éxécution d'aucune de ces deux instructions ne soit augmentée.

3.3.2. ORGANISATION DE LA SIMULTANEITE :

L'organisation de la simultaneité est fondée sur l'idée de synchronisation dans le temps des processus qui se produisent dans l'unité mémoire l'unité contrôle et l'unité de traitement. Cette forme d'organisation est rencontrée dans la littérature sous le nom de parallelisme local ou vertical.

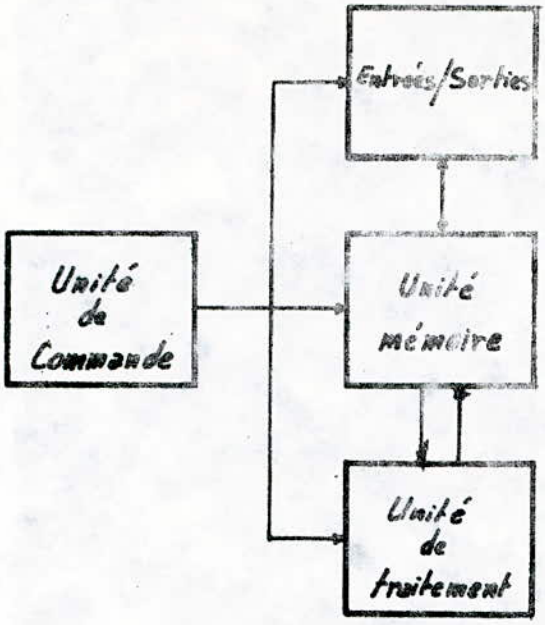


fig. 3.1 Synoptique d'un ordinateur à organisation linéaire.

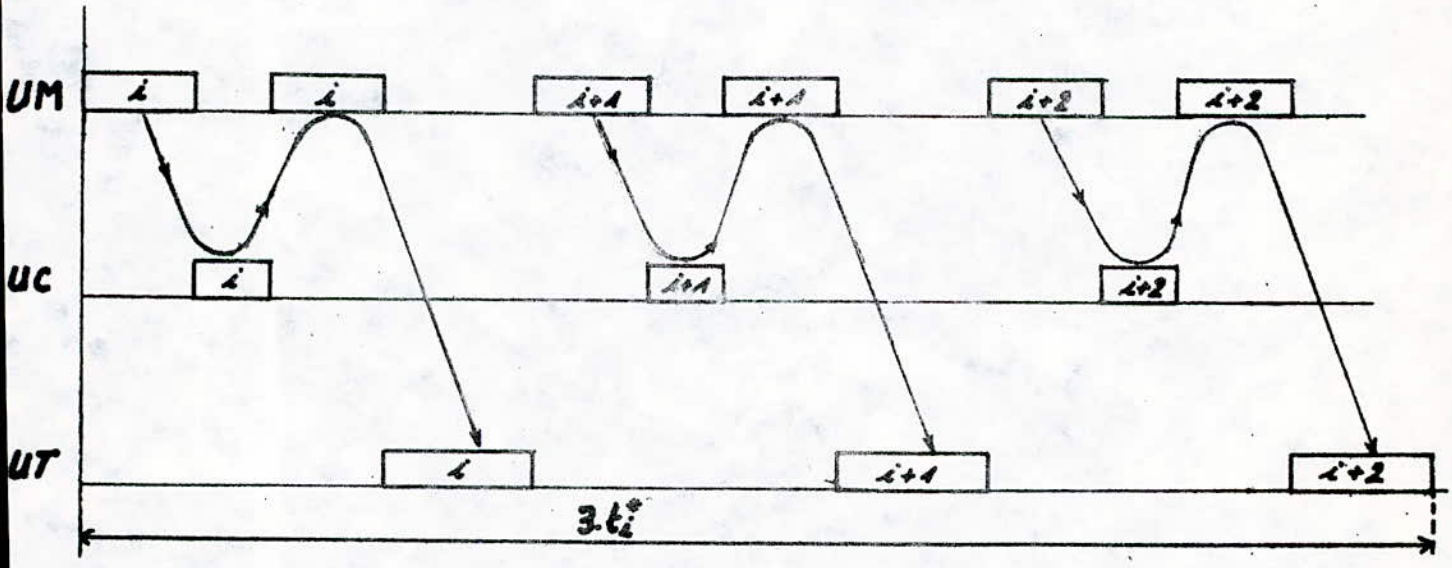


Fig. 3.2 Diagramme fonctionnel de l'organisation linéaire d'un ordinateur.

* t_i : Temps d'exécution d'une instruction.

Un cas simple de cette forme d'organisation est représentée dans la figure 3.3. et le diagramme illustrant son fonctionnement à la figure 3.4, où ont peut voir qu'à certains moments, les processus (opérations) liés à l'instruction (i - 1) et le processus de lecture de l'instruction (i) sont réalisés au même temps. Il est évident que la réalisation d'une telle organisation entraine une diminution du cycle machine, mais ceci se paye par une augmentation de la partie Hardware, qui servira à stocker les instructions préparées préalablement.

Dans ce cas si une opération de branchement conditionnel (IF) se présente, elle constituera un inconvénient et ceci nous oblige à revenir temporairement à l'organisation linéaire car il est impossible d'entamer le processus (i + 1) avant d'obtenir le résultat du processus (i)

3.3.3. ORGANISATION DE LA SIMULTANEITE SUR LA BASE DE LA DIVISION DE LA MEMOIRE :

Cette forme de simultaneité qu'on peut rencontrer dans l'évolution du traitement parallèle consiste à diviser l'unité mémoire en blocs fonctionnels : une mémoire pour stocker les opérandes (UMO) et une autre pour stocker les instructions (UMI). Ceci permet d'avoir accés aux opérandes et aux instructions en même temps, ce qui entraine une diminution considérable du cycle mémoire. Une telle organisation est décrite par le figure 3.5. dont son fonctionnement est illustré par le diagramme de la figure 3.6.

3.3.4. AUTRES TYPES D'ORGANISATION POUR AUGMENTER LA RAPIDITE D'UN SYSTEME :

Dans le but de rendre encore plus rapide le fonctionnement des systèmes, furent crées ce qu'on appelle les systèmes confluents, dans lesquels sont réalisés les recouvrements partiels des cycles d'éxécution d'instructions (Over Lapping). Les conditions de confluences sont réalisées grâce au dédoublement des unités les moins rapides et le fonctionnement en parallèle de toutes les unités.

La figure 3.7 nous donne le schéma synoptique d'un tel système à six blocs mémoires dont chacune à un temps d'accés T_{ma}, d'où le temps d'accés moyen $(T_{ma})_{moy.}$ sera :

$$(T_{ma})_{moy.} = \frac{1}{6} T_{ma}$$

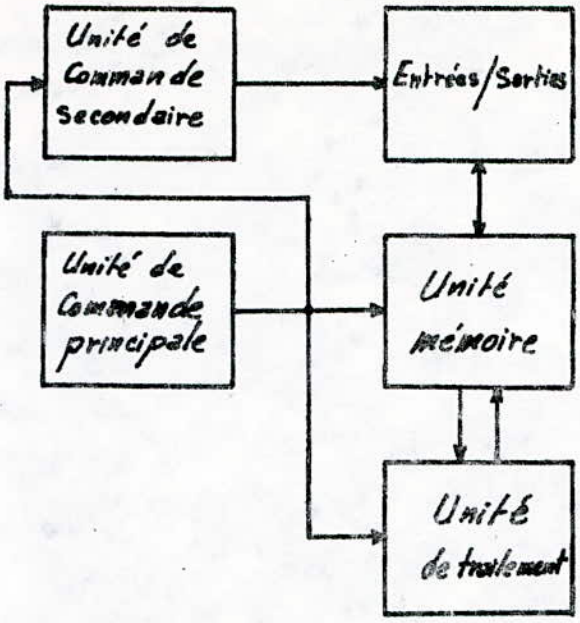


fig. 3.3 Synoptique d'un ordinateur à organisation simultanée.

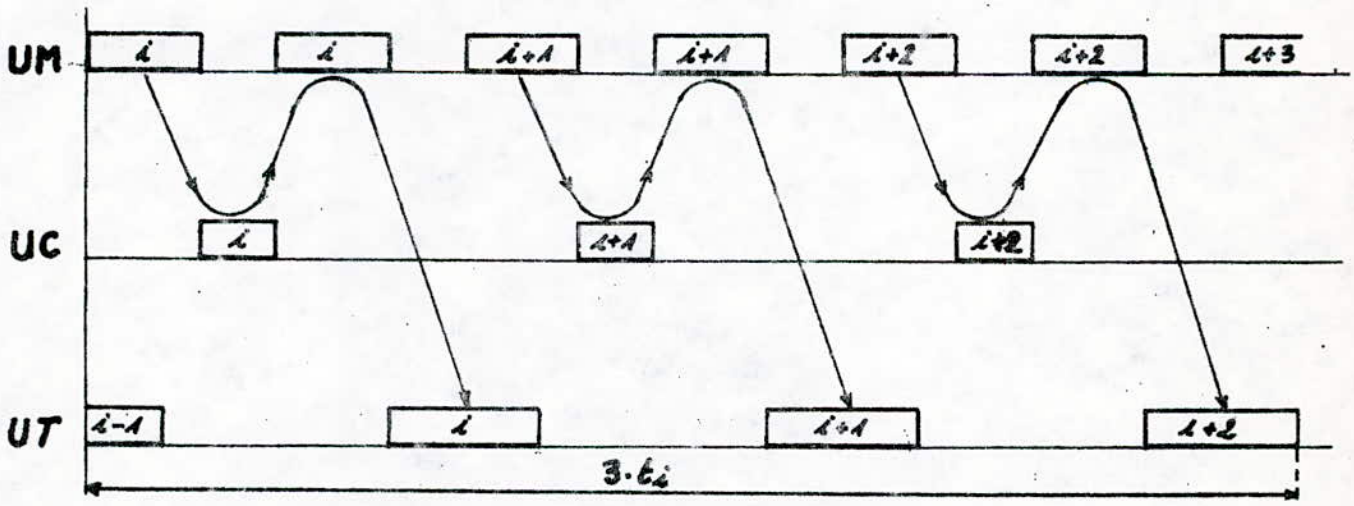


fig. 3.4 Diagramme fonctionnel de l'organisation de la simultanéité.

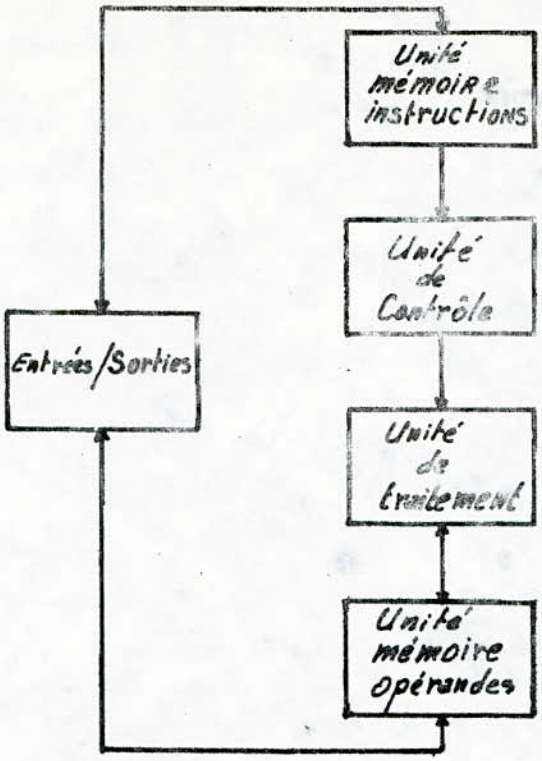


Fig. 3.5 Synoptique d'un ordinateur à mémoire d'instructions et mémoire d'opérandes.

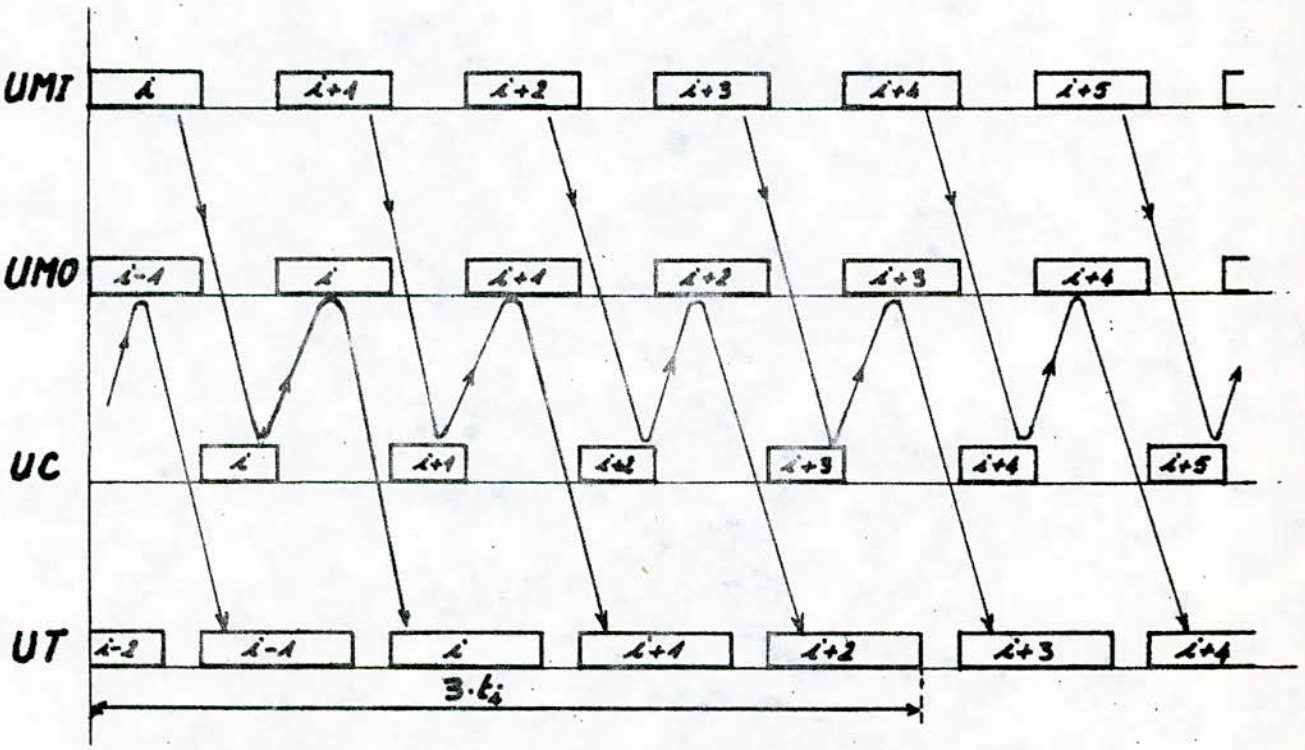


Fig. 3.6 Diagra . fonctionnel de l'organisation de la simultanéité sur la base de la division de la mémoire.

Le diagramme (figure 3.8) illustre le fonctionnement de ce système

Ce genre de solution est utilisé lorsque l'unité de traitement est très rapide ou bien doit être alimentée en opérandes à une fréquence très élevée ; c'est le cas des systèmes pipelines.

Tous les moyens cités d'amélioration de l'organisation linéaire sont fondés sur le recouvrement des cycles des différents processus, liés à l'éxécution de l'instruction. Ces différents processus ont été regroupé par FLYNN [3] suivant une classification qui sera éxaminée dans le paragraphe ci-dessous.

3.4. CLASSIFICATION DES SYSTEMES DE TRAITEMENT DE DONNEES PAR FLOTS DE DONNEES ET FLOTS D'INSTRUCTIONS. :

Actuellement il existe plusieurs approches de classification des systèmes de traitement de données dont la plus répandue reste le classification de FLYNN qui définit l'actuvuté d'un ordinateur comme l'intersection de deux flots : les instructions et les données. Chacun de ces flots peut être simple ou multiple, ce qui définit quatre types génériques de systèmes informatiques.

- Les systèmes SISD (single instructions flow, single data flow) à simples flots d'instructions et simples flots de données.
- Les systèmes SIMD (single instructions flow, multiple data flow.) où le même flot d'instructions est éxécuté sur plusieurs ensembles de données.
- Les systèmes MISD (multiples instructions flow, single data flow.) où le même ensemble de données sert d'opérandes à différents flots d'instructions s'éxécutant simultanément.

- Les systèmes MIMD (multiple instructions flow, multiple data flow.) sont caractérisés par le fait que différents flots d'instructions s'éxécutent simultanément chacun prenant ses opérandes dans un ensemble de données préalablement choisi.

3.4.1. LES SYSTEMES SISD :

Comme nous pouvons le constater, l'organisation linéaire est caractérisée par un simple flots de données et un simple flots d'instructions.

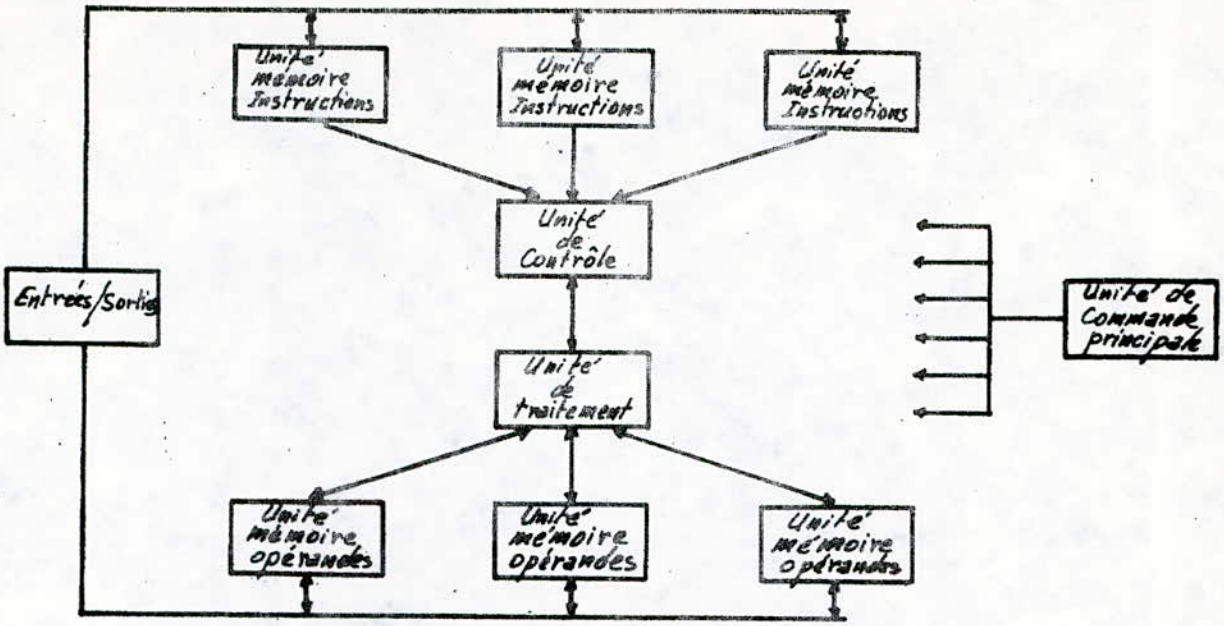


Fig. 3.7 Synoptique d'un système confluent.

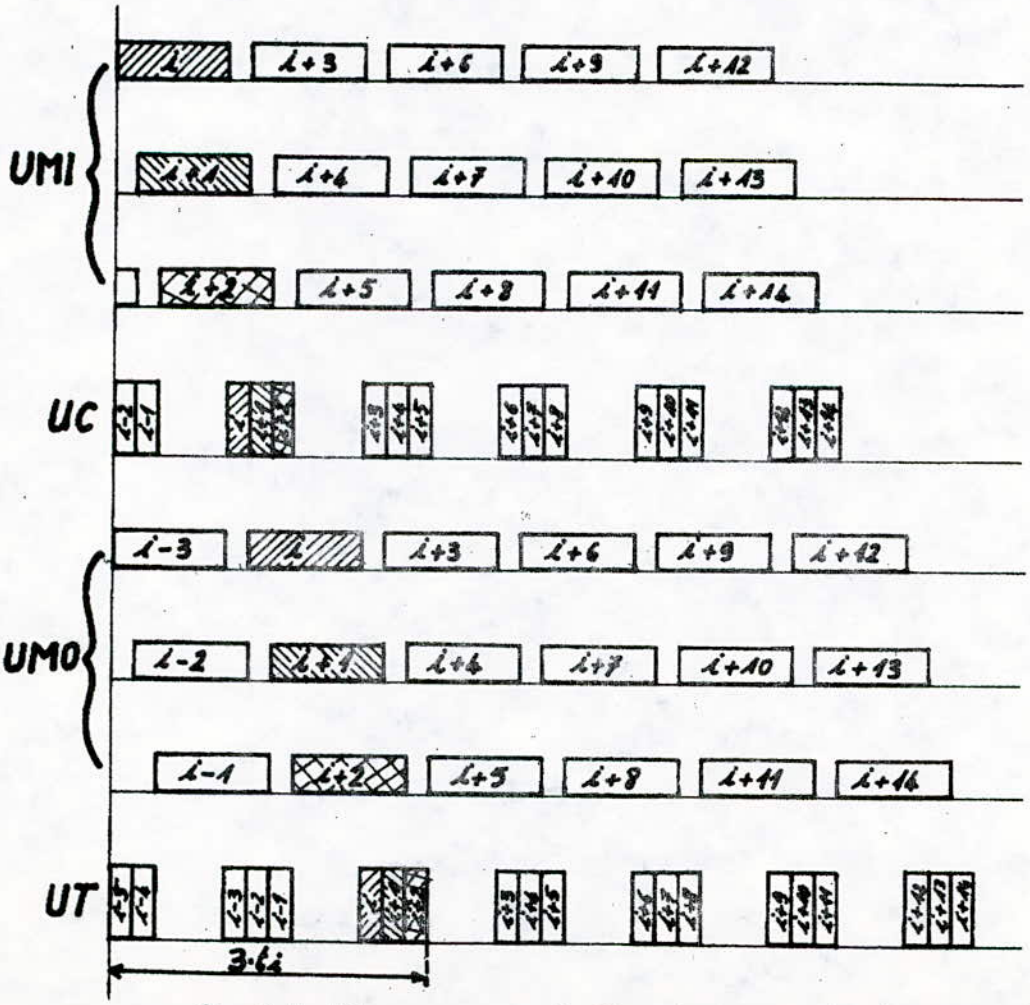


Fig. 3.8 Diagramme de fonctionnement d'un système confluent.

Dans ces systèmes, en un cycle de calcul, il n'y a qu'une seule instruction et qu'un seul groupe d'opérandes qui sont traités. Donc à ce niveau, on ne peut guère parler de parallelisme dans le traitement d'un ensemble de données. Le traitement en parallele des données a fait son apparition avec les systemes de la classe MISD.

3.4.2. LES SYSTEMES MISD :

On rencontre rarement des algorithmes où plusieurs instructions s'éxécutent simultanément sur les mêmes données, et par conséquent cette structure est très peu répandue. La figure 3.9 illustre une telle structure

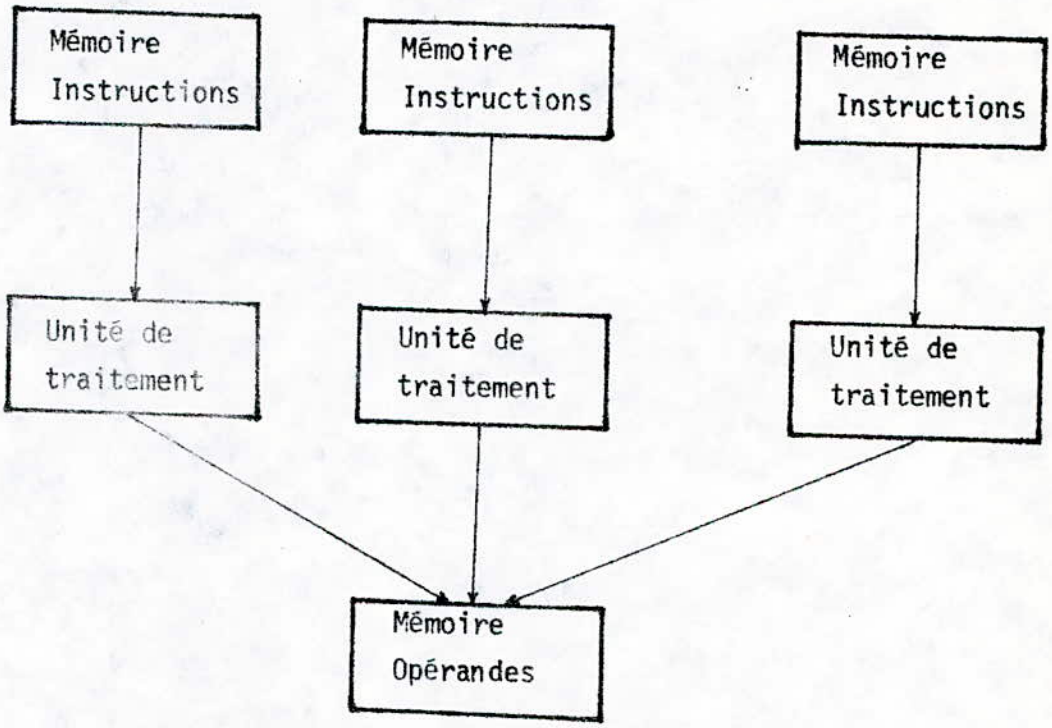


Fig. 3.9.

3.4.3. LES SYSTEMES SIMD.

A cette classe de systèmes appartient les processus du type SOLOMON et ILLIAC IV de la NASA. Aujourd'hui, on distingue trois approches de conception de ces systèmes :

- La première approche consiste à décomposer les algorithmes de

calcul d'un certain nombre de problèmes en un ensemble de processus pouvant être traités indépendemment l'un de l'autre pour ensuite les répartir suivant des unités de calcul technologiquement identiques.

- La deuxième approche consiste à représenter le système de traitement comme un ensemble d'unités de traitement fonctionnant en parallele et reliées de telle sorte à ce que les résultats de l'unité (i) soient utilisés comme données d'entrée par l'unité (i + 1) chaque unité réalisant la même fonction. Ce sont les structures pipelines. Comme exemple on peut citer les systèmes STAR 100, CRAY1 et MU5 de CIC.

La figure 3.10 représente un système à structure pipeline où chaque niveau (i) est composé d'un circuit combinatoire et de deux registres (figure 3.11).

Bien que le temps total nécessaire pour effectuer toute une opération parait relativement grand, la présence des registres de decouplage dans la ligne d'opération formée par les n niveaux, permet d'introduire une nouvelle donnée après chaque période Δt (où Δt est le temps de traitement d'un niveau i) avant même que le traitement des précedentes ne soit terminé. Le premier résultat sera donc obtenu après une durée $n.\Delta t$ et les suivants règulièrement à des intervalles de temps Δt. Le temps T de N traitement sera donc :

$$T = n.\Delta t + (N-1). \Delta t.$$

La structure de la mémoire centrale de ce système est représentée par la figure 3.12. Nous avons m blocs mémoires, dont le temps d'accés à chacun d'eux est t_a, organisés de manière à ce que le temps d'accés à la mémoire sera réduit à Δt. Il est clair que ce temps Δt sera réduit d'autant plus qu'on ajoute des blocs mémoires. Pour une exploitation rationnelle on prendra m tel que $\Delta t = t_a / m$; (m entier).L'accés à cette mémoire sera donc aux instants t_1 , $t_1 + \Delta t$, $t_1 + 2\Delta t$, $t_1 + 3\Delta t$, ...

Ainsi, la structure pipeline permet d'augmenter la fréquence de travail sans devoir multiplier le matériel par rapport à la structure matricielle.

- La troisième approche nous conduit aux processus associatifs. En fait ce n'est qu'une version des processus matriciels avec la seule

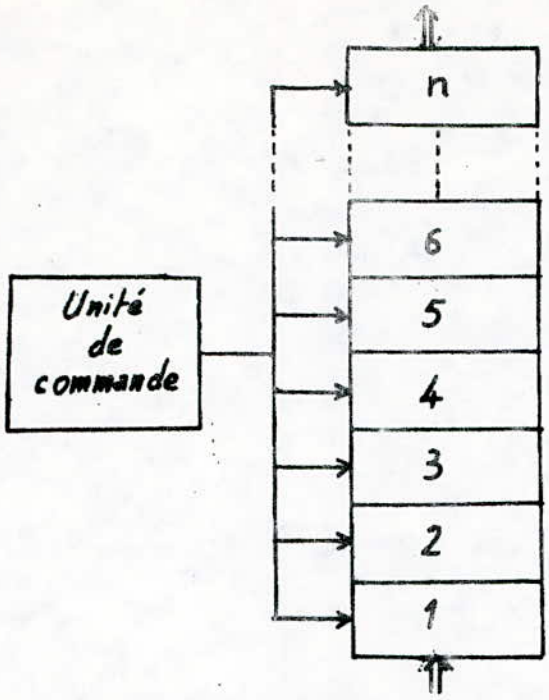


fig. 3.10 Système à structure pipeline.

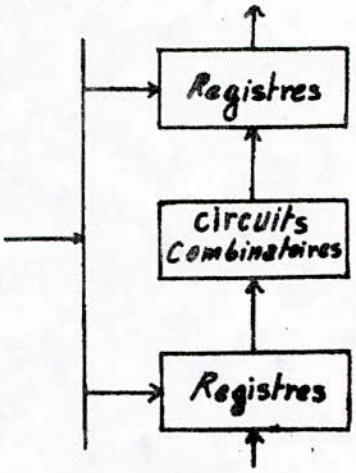


fig. 3.11 Schéma d'un niveau du pipeline.

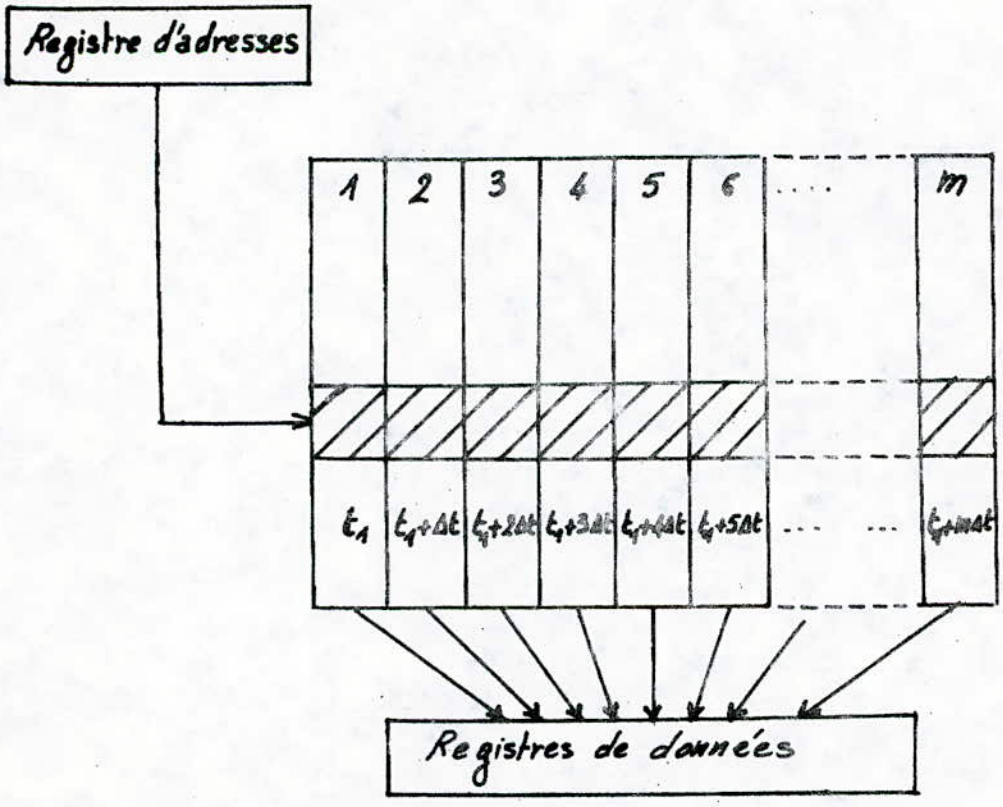


fig. 3.12 Organisation de la mémoire centrale d'un système à structure pipeline.

différence, que dans ce cas, une logique multimodèle est utilisée pour faire le choix des unités opérationnelles devant passer à un régime actif. les structures matricielles et associatives sont illustrées par le schéma de la figure 3.13. On voit bien ici, que chaque processeur élémentaire est commandé par le processeur principal dont sa mémoire appelée " mémoire centrale " est schématisée à la figure 3.14. Tous les blocs mémoires sont commandés au même instant t_1 , et leur contenu est stocké dans des registres avant d'être destiné au traitement dans le processeur matriciel et / ou à sa commande.

Ce genre de système n'est pas exempts d'inconvénients dont les plus évidents sont :

1°) Incompatibilité entre les dimensions et la structure du vecteur logique qui doit être traité et la capacité du réseau des unités opérationnelles qui doivent assurer le traitement.

2°) La liaison entre les unités opérationnelles et leurs intéractions lors du processus de calcul.

3°) Dégradation de la productivité crée par le IF de branchement comme le montre la figure 3.4 , plus le nombre de IF est important, moins les processeurs parallèles seront utilisés.

L'accroissement de l'éfficacité et de la vitesse de traitement passent par la résolution de tous ces problèmes , d'où l'apparition des systèmes multiprocesseurs du type MIMD.

3.4.4. LES SYSTEMES MIMD :

Dans ce type de système, on découpe le programme informatique en tâches différentes et chacune de celles-ci est affectée à un seul processeur. Actuellement on rencontre trois types de systèmes MIMD :

- Système maitre esclave : Les programmes moniteurs sont réalisés par le processeur maitre, dans ce cas son arrêt entraine automatiquement la perte des capacités du système et le temps mort des processeurs esclaves décroit avec la rapidité du processeur maitre.

- Système où l'éxécution des opérations est répartie au sein de plusieurs processeurs, dont chacun a son propre système d'exploitation et

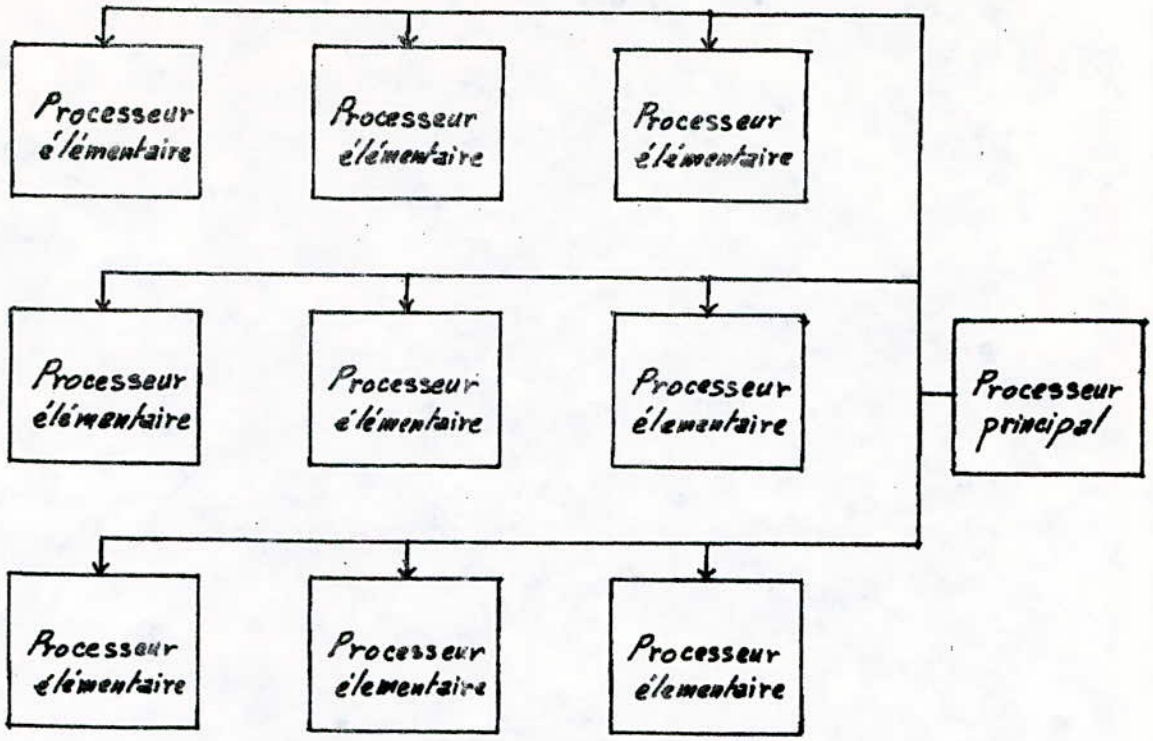


Fig. 3.13 Structure d'un processeur matriciel.

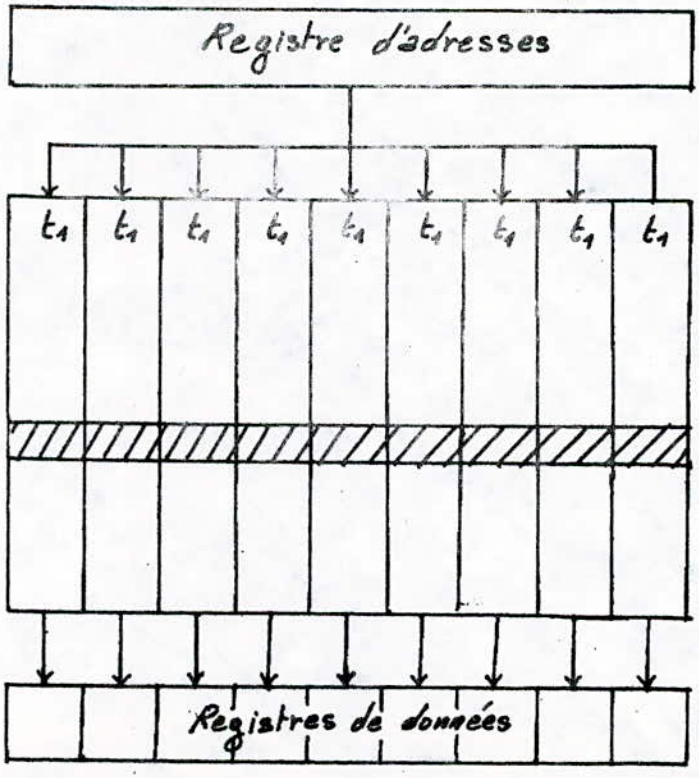


Fig. 3.14 Organisation de la mémoire centrale d'un processeur matriciel.

par conséquent l'arrêt de l'un n'entraine pas l'arrêt du système.

- Système à traitement homogène ou symétrique. Chaque processeur dans ce cas remplit la fonction de processeur principal.

Ce mode d'organisation (MIMD) est beaucoup plus dificile à gérer. Il faut réguler des opérations de manière differenciée d'où la difficulté de coordination entre tâches dépendantes les unes des autres.

La figure 3.15 représente la structure d'un système MIMD où l'on voit bien que différents flots d'instructions s'éxécutent simultanément sur différents flots d'opéraudes θi.

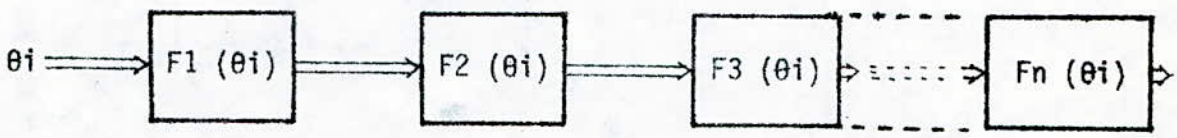


Fig. 3.15

3.5. CONCLUSION :

Comme l'illustre la relation (2.30) les algorithmes de FFT sont caractérisés par un grand volume d'opérations papillon à traiter et un très faible nombre d'entrées-sorties. L'ensemble des opérations papillon peut être assimilé à un simple flot d'instructions qui agit sur plusieurs flots de données que sont les valeurs échontillonnées et les données intermédiaires.

Cette remarque a fait porter notre choix sur les systèmes SIMD qui se présentent comme étant les plus adaptés à la réalisation du processeur FFT.

CHAPITRE 4.

ETUDE COMPARATIVE DES PROCESSEURS PARALLELES DE FFT

Avant de procéder au choix d'une structure du processeur FFT, il serait bon de le situer dans le système de traitement du signal en cours de réalisation dans le laboratoire architecture des systèmes :

4.1. CONFIGURATION DU SYSTEME DE TRAITEMENT DU SIGNAL

La fig. 4.1 illustre cette configuration. On se limitera ici à citer les différentes liaisons du processeur FFT avec les autres organes du système

L'entrée du processeur sera reliée au coupleur parallele qui lui fournira séquentiellement des blocs de données, chaque bloc étant un signal échontillonné. Sa sortie est reliée à un module mémoire ou seront stockés les résultats de la FFT et disponibles à subir certains traitements (voir applications au paragraphe 1.4). Ce processeur sera geré par le micro-ordinateur hôte.

4.2 PROCESSEUR A STRUCTURE PARALLELE

Comme le montre la figure 4.2, ce processeur est constitué d'une série de latches Lij et d'unités de traitement UTij ($i = 1 \div \log_2 N$ et $j = 1 \div N/2$). Chaque unité de traitement réalise une opération papillon dont les valeurs des coefficients w_N sont implantées sur les UTij correspondantes.

Les données arrivent aux latches d'entrée L_{1j} , les opérations papillons de la première itération sont effectuées simultanément par les unités de traitement UT_{1j}. Les sorties de ces UT_{1j} sont directement reliées aux entrées des latches L_{2j} de façon à ce que chacun reçoit les valeurs qui serviront à effectuer les opérations papillons de la deuxième itération. Ce processus se répète de la même manière dans les étages suivants jusqu'à la dernière itération où le résultat sera disponible aux sorties des UTij ($i = 9$ dans le cas où N = 512.)

Dans ce cas, la FFT sera réalisée en un temps Tp :

$$T_p = (t_p + t_l) \cdot \log_2 N \qquad (4.1.)$$

Où t_p est le temps de traitement d'une opération papillon et t_l le temps de mémorisation des données et résultats intermédiaires dans un latche

Pour N = 512 on aura $T_p = 9 (t_p + t_l)$ (4.2)

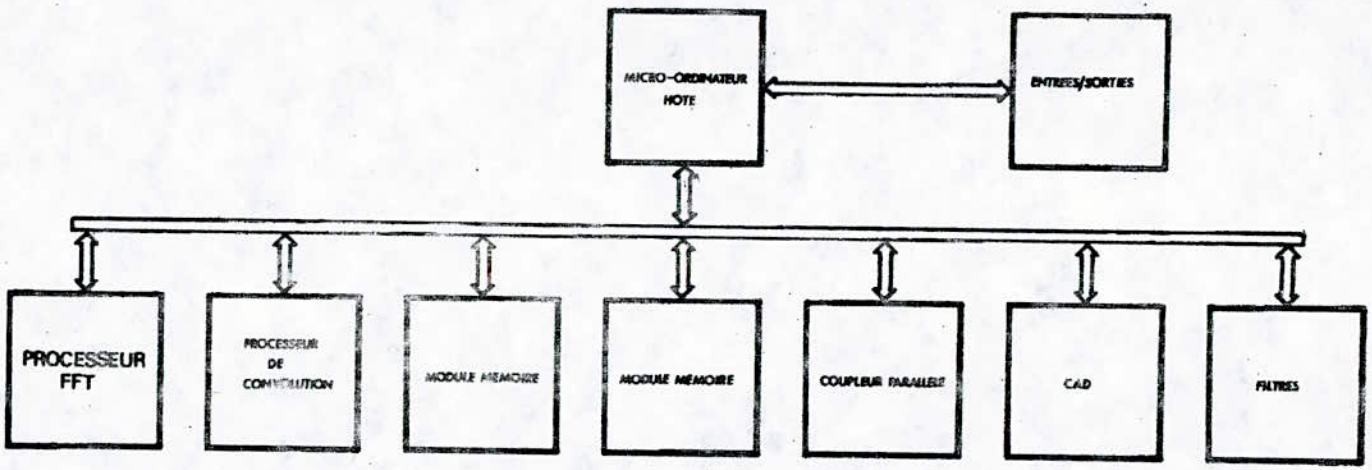


FIG 4-1 CONFIGURATION DU SYSTEME DE TRAITEMENT DU SIGNAL

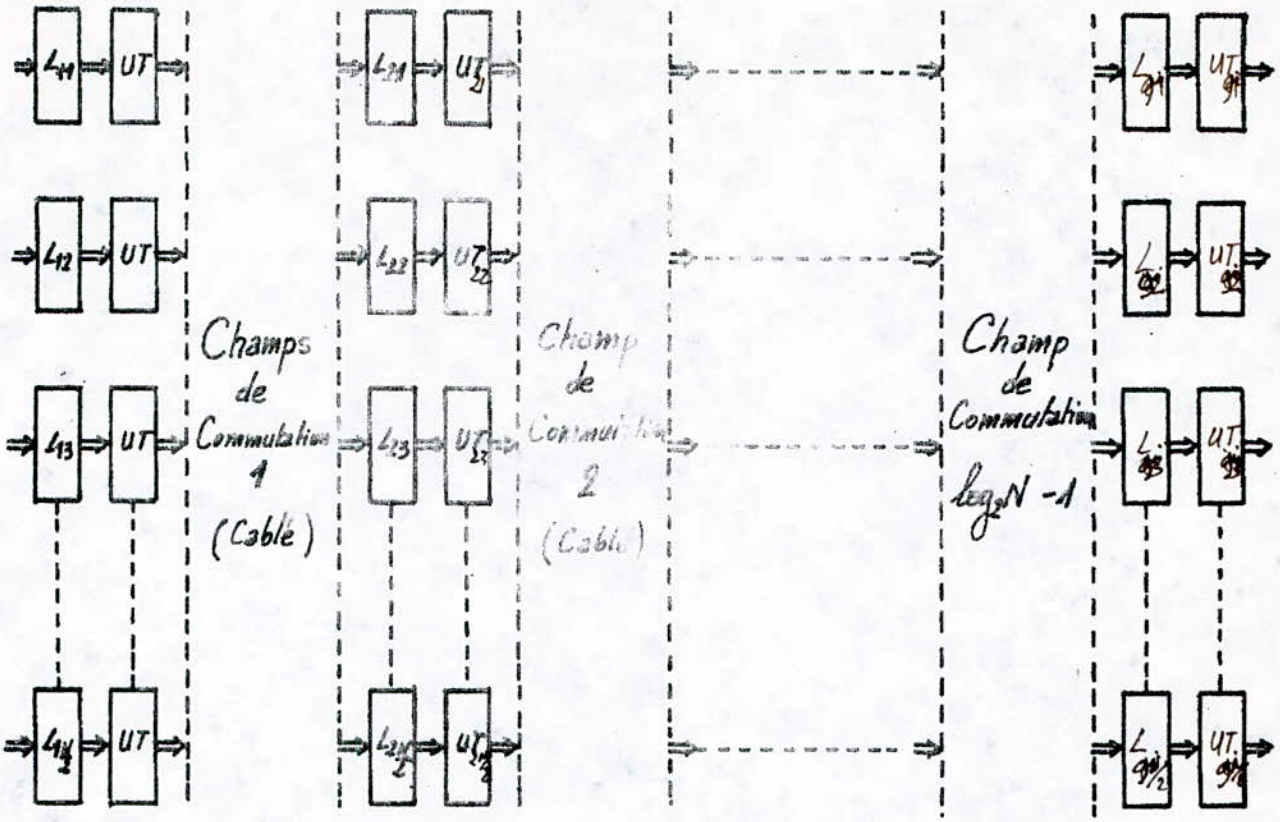


Fig. 4.2 Structure parallele d'un processeur FFT.

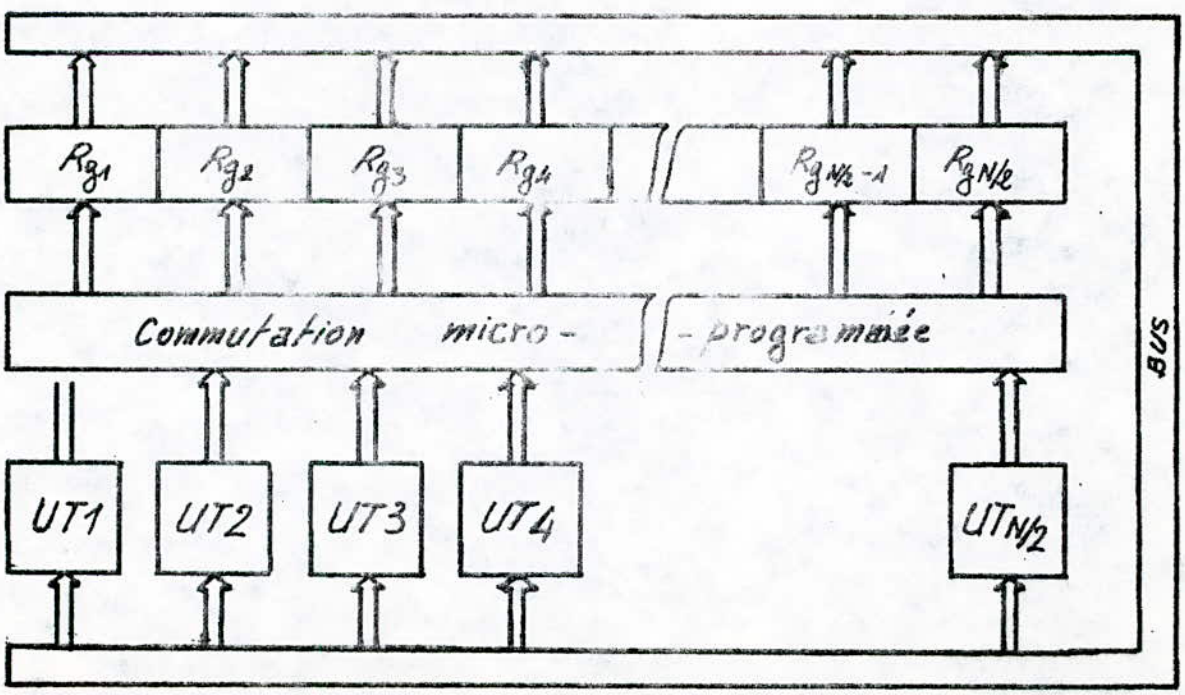


Fig. 4.3 Structure itérative d'un processeur FFT.

Avec une telle structure on pourra donc effectuer des FFT en des temps de l'ordre quelques dizaines de micro-secondes, Néanmoins cette structure présente beaucoup d'inconvenients dont les plus importants sont :

- Nécessité de $\frac{N}{2}\log_2 N$ unitésde traitement
- Beaucoup de cablage, ce qui diminue la fiabilité du système
- Encombrement
- Coût très éleyé,

4,3 PROCESSEUR A STRUCTURE ITERATIVE

Cette structure est une amélioration de la précédente, Comme le montre la figure 4,3, le nombre d'unités de traitement est réduit à N/2,

Les données arrivent en parallele aux unités de traitement, chacune effectue une opération papillon, les résultats de toutes ces unités qui travaillent en parallele, subissent une commutation micro programmée, et sont stockés dans les registres Rg de façon à trouver dans chaque Rg_j, les termes nécessaires pour effectuer la prochaine opération papillon par l'unité de traitement UTj , Ce processus se répète pour chaque iteration. Notons que dans chaque unité de traitement sont figés les coefficients w_N utilisés dans différentes itérations,

Le temps de traitement de la FFT est T_I

$$T_I = (t_I + t_c + t_R).\log_2 N \qquad (4.3)$$

où t_I est le temps de calcul d'une opération papillon, t_c le temps de commutation et t_R le temps de mémorisation des résultats intermédiaires dans les registres,

Pour N = 512 ; $T_I = 9(t_I + t_c + t_R)$ (4.4)

Avec cette structure on pourra aussi effectuer des FFT en des temps de l'ordre de quelques dizaines de micro-secondes, Néanmoins cette structure présente pratiquement les mêmes inconvénients que la précedente, bien qu'il y'a une sensible amélioration HARDWARE,

4.4. PROCESSEUR A STRUCTURE PIPELINE :

Si dans les structures précedentes (4.2 et 4.3) on peut effectuer la FFT en utilisant indiféremment des algorithmes A, B, C ou D rien qu'en jouant sur les valeurs des coefficients w_N implantées dans les unités de traitement et les commutations, dans cette structure (figure 4.4) on ne

peut utiliser que l'algorithme B ou D. Mais il ne sera pas difficile de concevoir une strucutre analogue pour pouvoir utiliser les algorithmes A ouC.

Les données x(k) ; k = 0, 1, ..., N-1 ; arrivant à l'entrée sont démultiplexées, les N/2 premières valeurs sont stockées dans la RAM 1, les valeurs suivantes sont dirigées séquentiellement vers l'unité de traitement UT1 qui recevra en même temps les valeurs stockées dans la RAM 1 et les coefficients w_N dont les adresses sont fournies par le séquenceur 1.

La sortie de l'UT1 est reliée à deux démultiplexeurs DX2 et DX3 qui orienteront réspectivement les valeurs d'indice $\left(0 \div \frac{N}{2} - 1\right)$ vers la RAM 2 et $\left(\frac{N}{2} \div \frac{3N}{4} - 1\right)$ vers la RAM 3. Les autres sorties des deux démultiplexeurs qui attaquent le commutateur 1 sont sélectionnées et les RAM 1 et RAM 2 mises en état de lecture. Le commutateur 1 selectionne alors alternativement les données qui arrivent de la RAM 2 et DX2 d'une part et celles qui parviennent (avec un retard) de la RAM 3 et DX3 d'autre part et par conséquent l'UT2 reçoit séquentiellement des données pour traiter la deuxième iteration. Le séquenceur2 fournira à l'UT2 les adresses des coefficients w_N adéquats. Il n'est pas difficile de comprendre le fonctionnement des étages suivants étant donnée que dans cette structure on ne fait que suivre pas à pas l'algorithme (représentation graphique).

La figure 4.4 illustre les trois premièrs étages d'un processeur FFT à structure pipeline. Notons qu'il faudra $\log_2 N$ étages pour effectuer une FFT de N points.

Dans ce cas le temps de traitement de la FFT est T_a :

$$T_a = (t_a + t_t)\frac{N}{2} \qquad (4.5)$$

où t_a est le temps d'accés à la mémoire et t_t le temps de traitement d'une opération papillon.

Pour N = 512 ; $T_a = 256\ (t_a + t_t)$. (4.6)

Bien que cette structure présente une amélioration Hardware nettement superieure aux précedentes, elle n'est pas exempt d'inconvénients dont le plus important est la difficulté de commande de ce processeur ; le vecteur de commande doit se déplacer au même rithme que le processus de calcul dans les différents étages. De même le nombre d'unités de traitement ($\log_2 N$) demeure excessif.

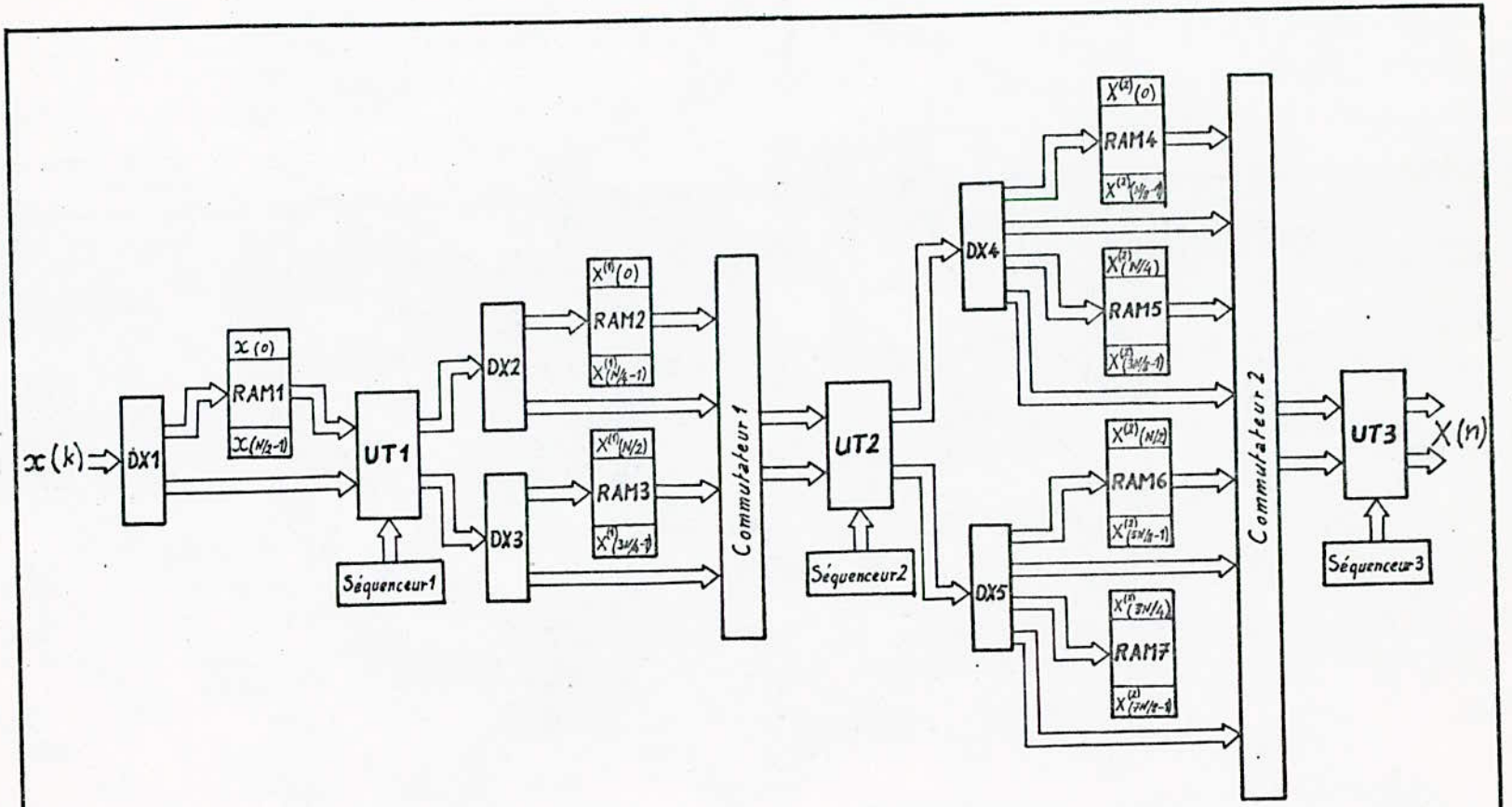


FIG. 4.4. Schéma synoptique d'un processeur FFT à structure pipeline.

4.5 COMPARAISON DES TROIS STRUCTURES ET PRESENTATION DE LA SOLUTION CHOISIE

Notre objectif est de réaliser un processeur FFT qui travaillera au même rithme que le coupleur parallele en voie de réalisation. Le temps estimé pour délivrer une suite de valeurs numériques d'un signal échontillonné est de 500 micro-secondes. Considerons ce paramètre comme hypothèse pour déterminer le choix de la structure la plus économique et la plus fiable

Le tableau 4.1 nous donne le nombre d'unités de traitement et la capacité mémoire nécessaire pour réaliser le processeur FFT ainsi que le temps approximatif (estimé) pour effectuer une opération papillon.

Le tableau 4.2 fait apparaitre les inconvénients et les avantages des differentes structures.

Les structures citées ci-dessus sont basées sur le principe de la décomposition de l'algorithme en plusieurs fragments identiques, ce qui necéssite un traitement en parallele donc un nombre d'unités (de traitement) important.

Notre choix est porté sur une structure à une seule unité de traitement très puissante basée sur le principe de la décomposition de l'opération papillon. Ceci nous permet d'utiliser les avantages du parallelisme et du pipeline.

De même l'algorithme A a été choisi pour réaliser ce processeur à cause des avantages cités dans le chapitre 2 à savoir :

- La symetrie de l'opération papillon
- L'erreur de troncature est minimale.

4.5.1 DESCRIPTION DU SCHEMA SYNOPTIQUE :

Le schéma synoptique de la fig. 4.5 illustre le fonctionnement du processeur FFT. Il est essentiellement constitué de

- Deux commutateurs destinés comme leur nom l'indique à acheminer les données entre l'unité de traitement et les mémoires dans les deux sens
- D'une unité mémoire composée de quatre blocs mémoires RAM de capacité globale 8K - Octets. Les blocs 1 et 3 recevant alternativement les données et les résultats intermédiaires. Le bloc 2 contiendra uniquement les résultats intermédiaires et travaillera alternativement en intermittance

Structure	En fonction de N		Pour N=512		Vitesse de calcul d'une opération papillon
	Capacité mémoire	Nombre d'UT	Capacité mémoire	Nombre d'UT	
Parallele	$3N \log_2 N$	$(N/2)\log_2 N$	13,5K octets	2404	50 µS.
Itérative	4N	N/2	3K octets	256	40 µS.
Pipeline	$2N(2+\log_2 N)$	$\log_2 N$	11K octets	9	1,5 µS.

Tableau 4.1

Structure	Avantages	Inconvénients
Parallele	- UT de traitement de faible puissance donc utilisation de composants de faibles performances pour les réaliser . - Synchronisation simple	- Nombre d'UT très important - Grande capacité mémoire - Encombrement - Coupleur n'arrive pas à suivre le processeur.
Itérative	- UT de traitement de faible puissance donc utilisation de composants de faibles performances pour les réaliser - Capacité mémoire réduite	- Synchronisation difficile - ~~Grande capacité mémoire~~ - Encombrement - Coupleur n'arrive pas à suivre le procésseur.
Pipeline	- Nombre d'UT réduit - Encombrement.	- Grande capacité mémoire - UT puissante ⇒ utilisation de composants de grande performances - Difficulté de commande - Compleur n'arrive pas à suivre le processeur.

Tableau 4.2

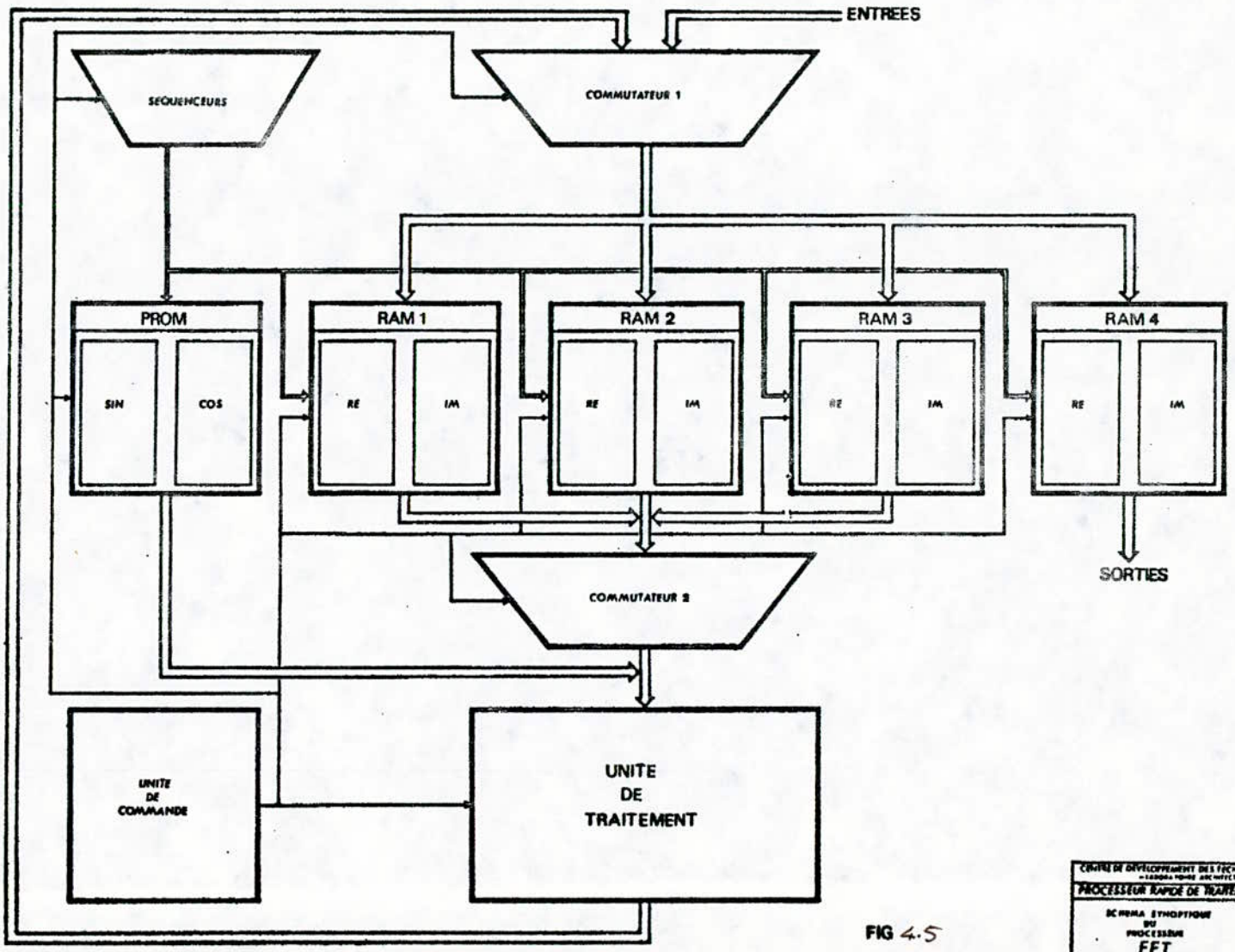


FIG 4.5

CENTRE DE DEVELOPPEMENT DES TECHNIQUES AVANCEES (C.D.T.A)
PROCESSEUR RAPIDE DE TRAITEMENT DU SIGNAL.
SCHEMA SYNOPTIQUE DU PROCESSEUR FFT
M. BESSALAH H.
M. RAMDA M.
Mme MOUZALI F.
PROJET DE FIN D'ETUDES
JANVIER 94

avec les blocs 1 et 3. Le quatrième bloc recevra uniquement les résultats finals.

Chaque bloc est divisé en deux sous blocs de 512 x 16 bits l'un contiendra la partie réelle, l'autre la partie imaginaire.

- Deux mémoires PROM de 256 octets chacune, l'une contient les valeurs de sinus et l'autre celles des cosinus

- D'un séquenceur délivrant les adresses aux différents blocs mémoires.

- D'une unité de traitement très rapide.

- D'une unité de commande permettant l'initialisation du système et la commande des différentes unités.

4.5.2 FONCTIONNEMENT DU PROCESSEUR

Les données numériques (512 valeurs d'un signal x_1 (t) échontillonné) arrivent à l'entrée du processeur et sont acheminées vers la RAM 1 grâce au commutateur 1. L'unité de commande déclenche le séquenceur qui fournit

les adresses en binaire réfléchi, ce qui permet de réaliser l'opération désembrouillage à l'écriture. Après un instant T1, les 512 valeurs sont inscrites dans la RAM 1, un deuxième bloc de données (d'un signal x_2 (t) échontillonné) arrive et l'U.C selectionne le bus commutateur 1 - RAM 3 et l'opération précedente se répète. Au même temps que les données du deuxième signal échontillonné s'inscrivent dans la RAM 3, l' UC met la RAM 1 et la PROM contenant les sinus et les cosinus en état de lecture. L'unité de traitement reçoit alors séquentiellement les données provenant de la RAM 1 et les valeurs de sinus et de cosinus adequats et effectue ainsi les opérations papillon de la première itération dont les résultats vont se loger à la RAM 2 et constituent des données pour la deuxième itération. A partir de ces données la deuxième itération est lancée et ses résultats vont s'inscire dans la RAM 1 et représentent ainsi des données d'entrée pour la troisième itération . Ce processus se répète entre RAM 1 et RAM 2 jusqu'à la huitième itération dont les résultats seront à la RAM 1. A la neuvième itération, les données proviennent à l'UT de la RAM 1 à travers le commutateur 2 et les résultats seront inscrits à la RAM 4. A la fin de cette dernière itération, les RAM 1 et RAM 2 sont disponibles, le deuxième bloc de données est entièrement stocké dans la RAM 3 puisque le temps de calcul de la FFT et de stockage d'un bloc de données est le même,

alors la RAM 1 recevra un troisième bloc de données (d'un autre signal échontillonné) et le cycle précédent recommence avec cette fois-ci la RAM 2 et la RAM 3 travaillant en intermittance. Dans ce cas le résultat de la huitième itération sera à la RAM 3. Il est clair qu'il faudra vider la RAM 4 durant le traitement des itérations afin de pouvoir écrire dans celle-ci ce deuxième résultat (FFT du deuxième signal).

Ce processus de calcul va se répéter indéfiniment jusqu'au traitement de tous les signaux.

On voit bien que le premier résultat est obtenu après un temps $2T_1$, le deuxième après $2T_1 + T_1$ et le $n^{ième}$ après $(n + 1) T_1$.

CHAPITRE 5.

PROCESSEUR PIPELINE DE FFT.

5.1. ORGANISATION DU PROCESSUS DE CALCUL DE LA FFT.

L'algorithme A de FFT (2.30) que l'on a adopté peut être représenté sous forme graphique et ce dans le but de faire refléter le processus de calcul.

La figure 5.1 illustre le diagramme de FFT sur 16 points. L'éxamen de cette figure nous montre que la partie essentielle de ce diagramme est constituée par des motifs appelés itérations qui relient le signal initial à ses états intermédiaires, Les coefficients de Fourier sont obtenus à la dernière itération. On reconnait là une géometrie identique de toutes les étapes de traitement composées d'unités algorithmétiques, semblables qu'on appelle " opération papillon " .

Les conventions du " papillon " (figure 5.2) sont les suivantes :

- Le cercle vide (0) définit une opération d'addition-soustraction dans laquelle la somme apparait en haut et la différence en bas.

- La flèche (⟶) décrit une multiplication avec une valeur située au dessus de la flèche.

- a et b deux noeudx sources

- c = a + bw et d = a - bw deux noeuds puits

Dans le cas de 16 points, le premier papillon de la première itération a pour noeuds sources les points x (0) et x (8),et pour les noeuds puits X_1 (o) et X_1 (8) reliés aux noeuds sources par la relation suivante :

$$X_1(0) = x(0) + W^{\circ} x(8)$$
$$X_1(8) = x(0) - W^{\circ} x(8) \qquad (5-1)$$

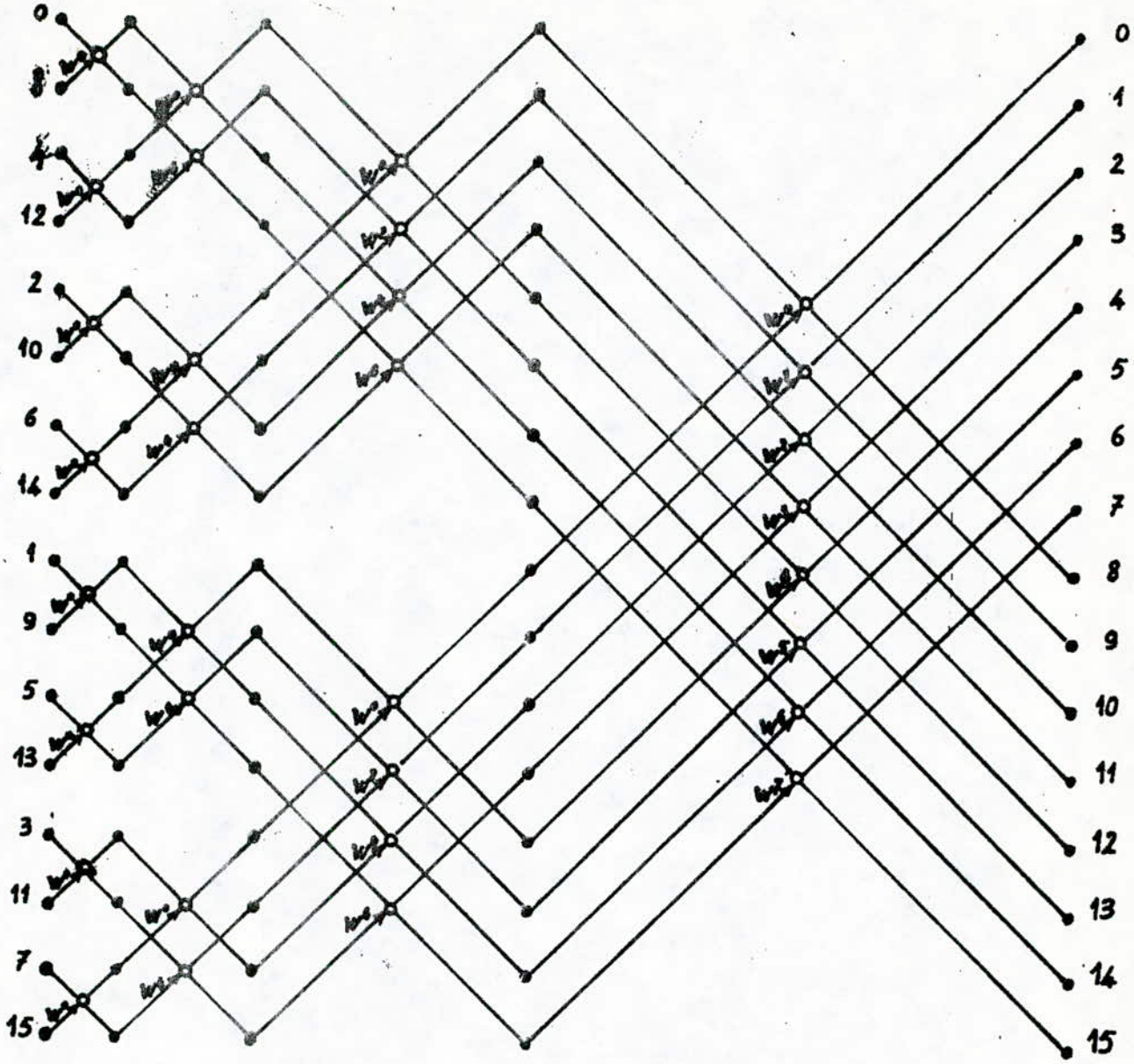


Fig. Diagramme de FFT sur 16 points, radix 2
Entrée désembrouillée - Sortie ordonnée.

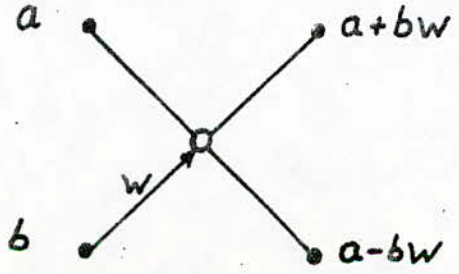


Fig. - Opération papillon.

Pour le deuxième papillon :

$$X_1(4) = x(4) + W^4\, x(12)$$
$$X_1(12) = x(4) - W^4\, x(12) \qquad (5-2)$$

Le premier papillon de la deuxième itération a pour noeuds sources les points $X_1(o)$ et $X_1(4)$ et pour noeuds puits les points $X_2(o)$ et $X_2(4)$ reliés aux noeuds puits par la relation

$$X_2(0) = X_1(0) + W^0\, X_1(4)$$
$$X_2(4) = X_1(0) - W^0\, X_1(4) \qquad (5-3)$$

Dans le cas de N points et pour $0 \leqslant K \leqslant N/2 - 1$ et d'après les expressions (5.1) , (5.2) et (5.3) on peut déterminer les expressions liant chaque état intermédiaire $(j+1)$ avec l'état j

$$X_{j+1}(k) = X_j(k) + W^p\, X_j\left(k + \frac{N}{2}\right)$$
$$X_{j+1}\left(k + \frac{N}{2}\right) = X_j(k) - W^p\, X_j\left(k + \frac{N}{2}\right) \qquad (5-4)$$

En posant $k = K0$ et $k + \frac{N}{2} = K1$ le système (5.4) devient :

$$X_{j+1}(K0) = X_j(K0) + W^p\, X_j(K1)$$
$$X_{j+1}(K1) = X_j(K0) - W^p\, X_j(K1) \qquad (5-5)$$

Puisque les X_j sont des valeurs complexes on utilise la formule d' EULER de décomposition en parties réelle et imaginaire :

$$W^p = \exp\left(-j\,\frac{2\pi}{N}\,p\right) = \cos\theta - j\sin\theta \qquad \text{avec } \theta = \frac{2\pi}{N}\,p$$

On aura pour chaque papillon les expressions suivantes :

$$
\begin{aligned}
RE_{j+1}(K0) &= RE_j(K0) + \left[RE_j(K1)\cdot\cos\theta + IM_j(K1)\cdot\sin\theta\right] \\
RE_{j+1}(K1) &= RE_j(K0) - \left[RE_j(K1)\cdot\cos\theta + IM_j(K1)\cdot\sin\theta\right] \\
IM_{j+1}(K0) &= IM_j(K0) + \left[IM_j(K1)\cdot\cos\theta - RE_j(K1)\cdot\sin\theta\right] \\
IM_{j+1}(K1) &= IM_j(K0) - \left[IM_j(K1)\cdot\cos\theta - RE_j(K1)\cdot\sin\theta\right]
\end{aligned}
\qquad (5.6)
$$

où

$$K0 = k_{m-1}\ k_{m-2} \ldots \overset{k_j}{0} \ldots k_0$$

$$K1 = k_{m-1}\ k_{m-2} \ldots \underset{k_j}{1} \ldots k_0 .$$

5.2 L'UNITE DE TRAITEMENT

Afin de limiter le nombre d'entrées sorties de l'unité de traitement et d'obtenir une très grande rapidité de calcul, nous avons opté pour une organisation pipeline de traitement.

La figure 5.3 représente le schéma synoptique de l'unité de traitement. Elle est composée de :

- Un circuit d'aiguillage des données
- Une unité arithmetique (UA)
- Un circuit d'aiguillage des résultats.

L'acheminement pipeline des données est assuré par un circuit d'aiguillage d'entrée. Les résultats transitent par un circuit d'aiguillage de sortie qui assure la synchronisation entre l'UT et la mémoire RAM.

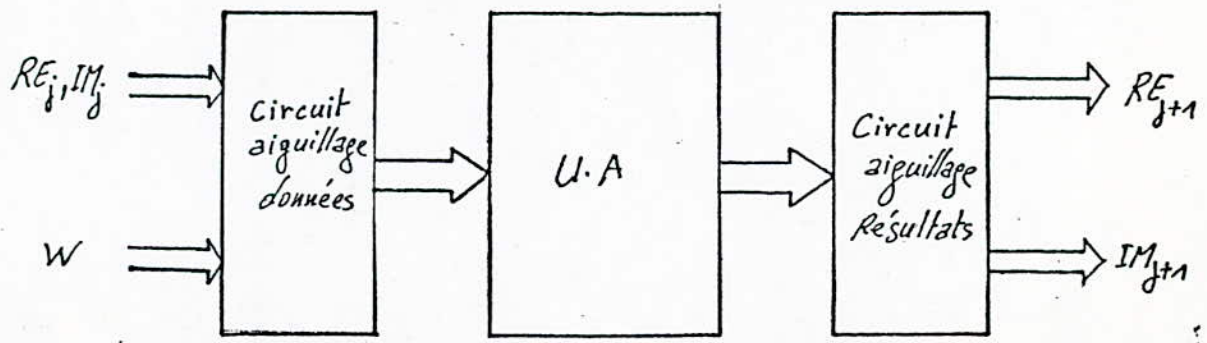


Fig.5.3.

5.2.1. CIRCUIT D'AIGUILLAGE DES DONNEES.

Les groupes de données $[RE_j(K0)\ ,\ IM_j(K0)]$ et $[RE_j(K1), IM_j(K1)]$, $\cos\theta$ et $\sin\theta$ arrivent séquentiellement à l'entrée du circuit d'aiguillage toutes les 80 nS.

A l'instant t_1, RE_j (Ko) et IM_j (Ko) sont respectivement mémorisées dans L5 et L6. A l'instant (t_1 + 80 nS), RE_j (K1), IM_j (K1), $\cos\theta$ et $\sin\theta$ sont respectivement stockés dans les latches L1, L2, L1S, et L3C. Chacune de ces deux opérations se répète chaque 160 ns.

Le fonctionnement de ce circuit est illustré par le schéma fonctionnel (figure 5.4), le diagramme espace-temps (Fig 5-5) et le chronogramme (annexe).

L'aiguillage des données nous permet donc d'obtenir les groupes d'opérandes $[RE_j(K1), \cos\theta]$; $[IM_j(K1), \sin\theta]$, $[IM_j(K1), \cos\theta]$ et $[RE_j(K1), \sin\theta]$ toutes les 160 nS avec une période de 40 nS.

Le circuit d'aiguillage fut réalisé à base de latches du type SN74LS 373 qui présentent les avantages suivants :

- Faible consommation
- Nombre de commandes limité
- Entrées paralleles sorties paralleles
- Encombrement réduit (latches de 8 bits et on travailleavec 16 bits)

et de portes OR du type SN74S32 qui présentent l'avantage d'avoir un faible temps de propagation (5 nS)

Le brochage du latche SN74LS373 et sa table de fonction sont représentés page 60

Le schéma élèctrique de principe est illustré par la figure 5.5a.

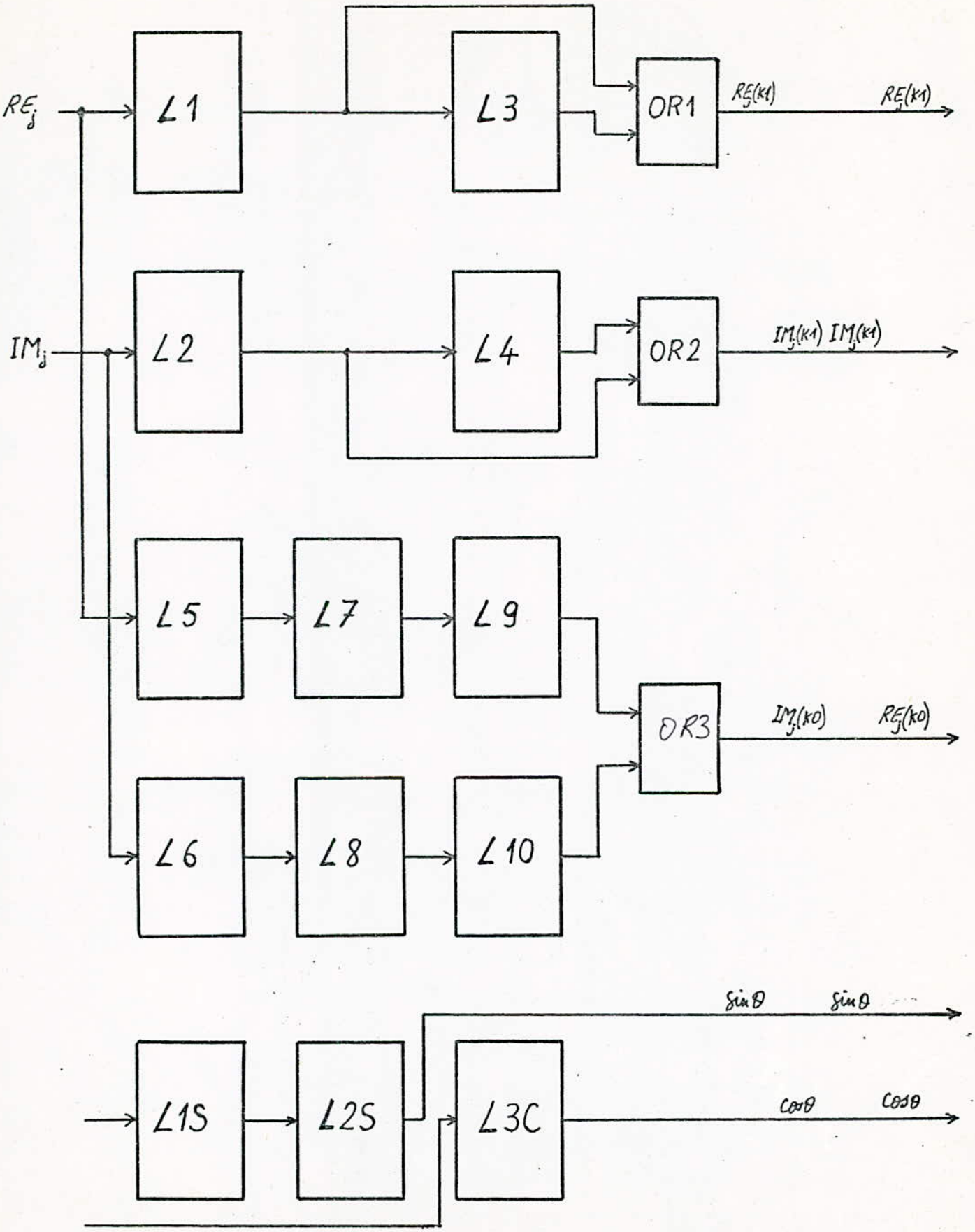


Fig. 5.4. Schéma fonctionnel du circuit d'aiguillage des données.

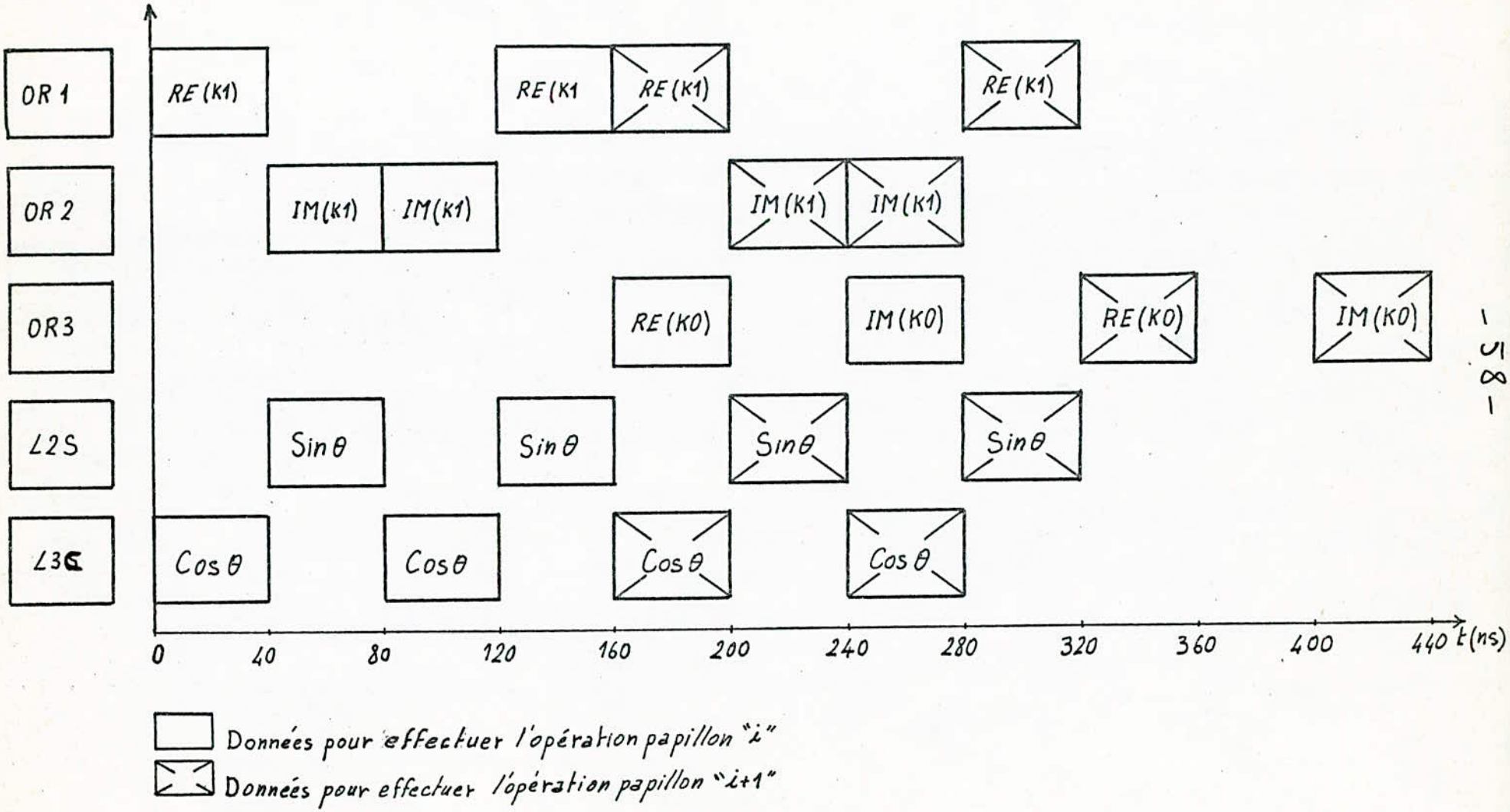


Fig. 5.5. Diagramme éspace-temps des sorties du circuit d'aiguillage des données.

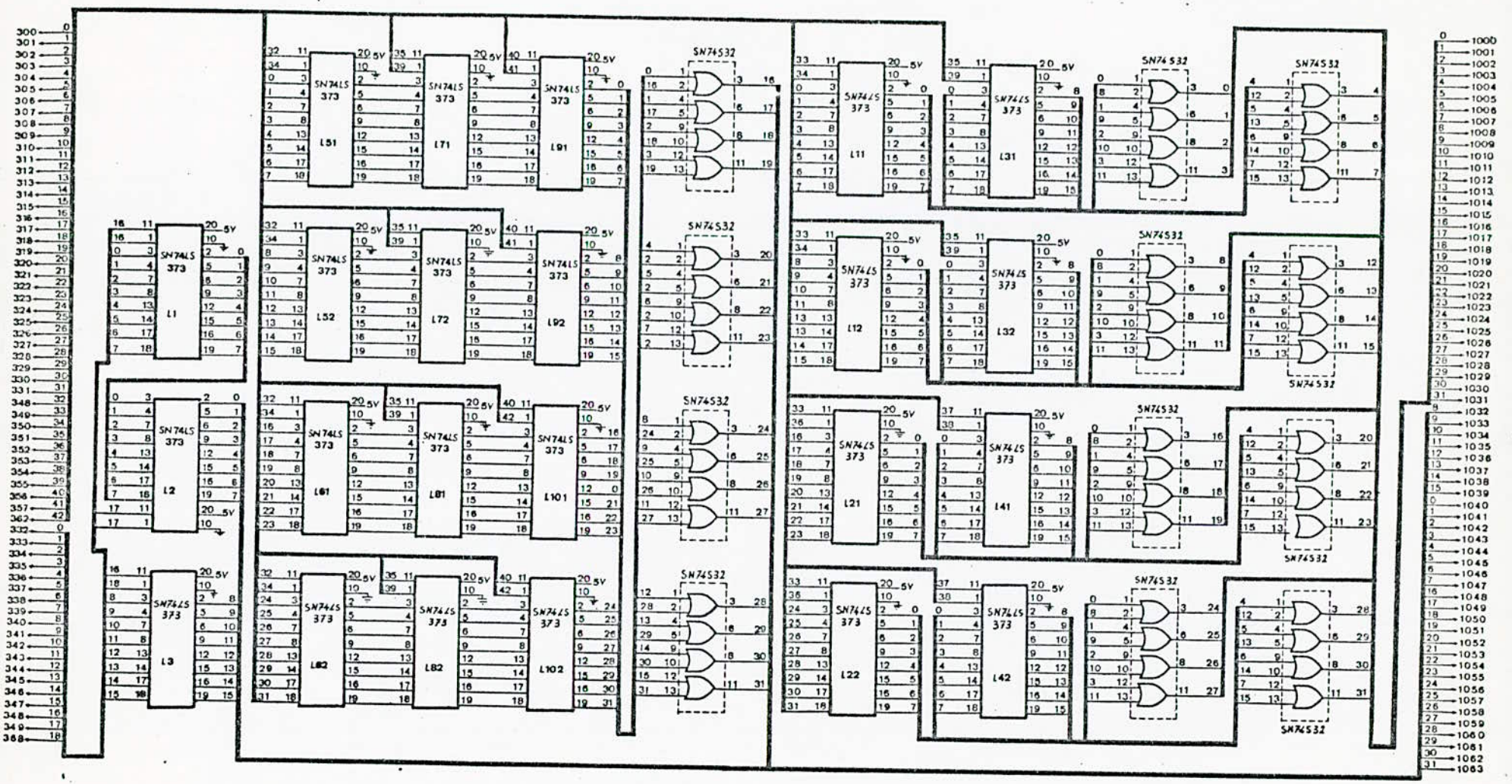
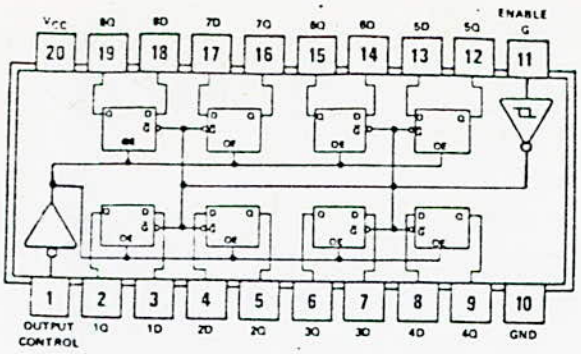


Fig. 5.5a SCHEMA ELECTRIQUE DU CIRCUIT D'AIGUILLAGE DES DONNEES.



SN74LS373

'LS373, 'S373
FUNCTION TABLE

OUTPUT CONTROL	ENABLE G	D	OUTPUT
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Q_0 : Etat antérieur
Z : Haute impédance

La mémorisation s'effectue lorsque la commande ENABLE est à un niveau haut " H " et la lecture lorsque la commande OUTPUT CONTROL est à un niveau bas " L ".

5.2.2. UNITE ARITHMETIQUE

5.2.2.1. Fonctionnement

La figure 5.6 illustre le schéma fonctionnel de l'unité arithmétique L'unité arithmétique reçoit les données suivant le diagramme (figure 5.5). Les produits $RE_j(K1).\cos\theta$, $IM_j(K1).\sin\theta$, $IM_j(K1).\cos\theta$ et $RE_j(K1).\sin\theta$ sont respectivement obtenus aux sorties des multiplieurs M1, M2, M3, et M4 avec un décalage temporel de 40 nS l'un par rapport à l'autre. La multiplication s'effectue en 160 nS puisque chaque multiplieur est alimenté toutes les 160 nS.

La valeur RE_j (Ko) provenant du circuit d'aiguillage, et le produit $RE_j(K1).\cos\theta$ sont obtenus en même temps et se présentent aux entrées de l'additionneur U1 et du soustracteur U3. Les opérations

$$S = RE_j(K0) + RE_j(K1).\cos\theta$$

$$D = RE_j(K0) - RE_j(K1).\cos\theta$$

s'effectuent en 40 nS. Ces deux sommes algébriques et le produit $IM_j(K1).\sin\theta$ seront obtenus en même temps et se présentent aux entrées des additionneurs soustracteurs U2 et U4 où les opérations :

$$RE_{j+1}(K0) = S + IM_j(K1).\sin\theta$$
$$RE_{j+1}(K1) = D - IM_j(K1).\sin\theta \qquad (5-7)$$

s'effectuent en 40 nS.

A cet instant, le résultat $IM_j(K1).\cos\theta$ et la valeur $IM_j(Ko)$ provenant respectivement de M3 et du circuit d'aiguillage se présentent aux entrées de U1 et U3 où les opérations suivantes sont éxécutées :

$$S' = IM_j(K0) + IM_j(K1)\cos\theta$$
$$D' = IM_j(K0) - IM_j(K1)\cos\theta$$

Ces deux résultats et le produit $RE_j(K1).\sin\theta$ sont obtenus en même temps et se présentent aux entrées de U2 et U4 qui sont commandés respectivement en soustracteur et additionneur. Après 40 nS, on obtient les résultats :

$$IM_{j+1}(K0) = S' - RE_j(K1).\cos\theta \qquad (5-8)$$
$$IM_{j+1}(K1) = D' + RE_j(K1).\cos\theta$$

Ce cycle se répète jusqu'à la fin du traitement et ainsi on obtient alternativement, chaque 80 nS, les résultats (5,7) et (5,8)

Le diagramme espace-temps de la figure 5,7 illustre l'enchainement des opérations dans l'unité arithmétique.

5,2,2,2, Choix et présentation du multiplieur

Contrairement aux additions et soustractions qui sont réalisées dans des temps très courts, la multiplication nécessite de developper des algorithmes plus ou moins complexes et performants pour obtenir le résultat. Le recours à un opérateur cablé peut s'averer extrêmement encombrant, Des

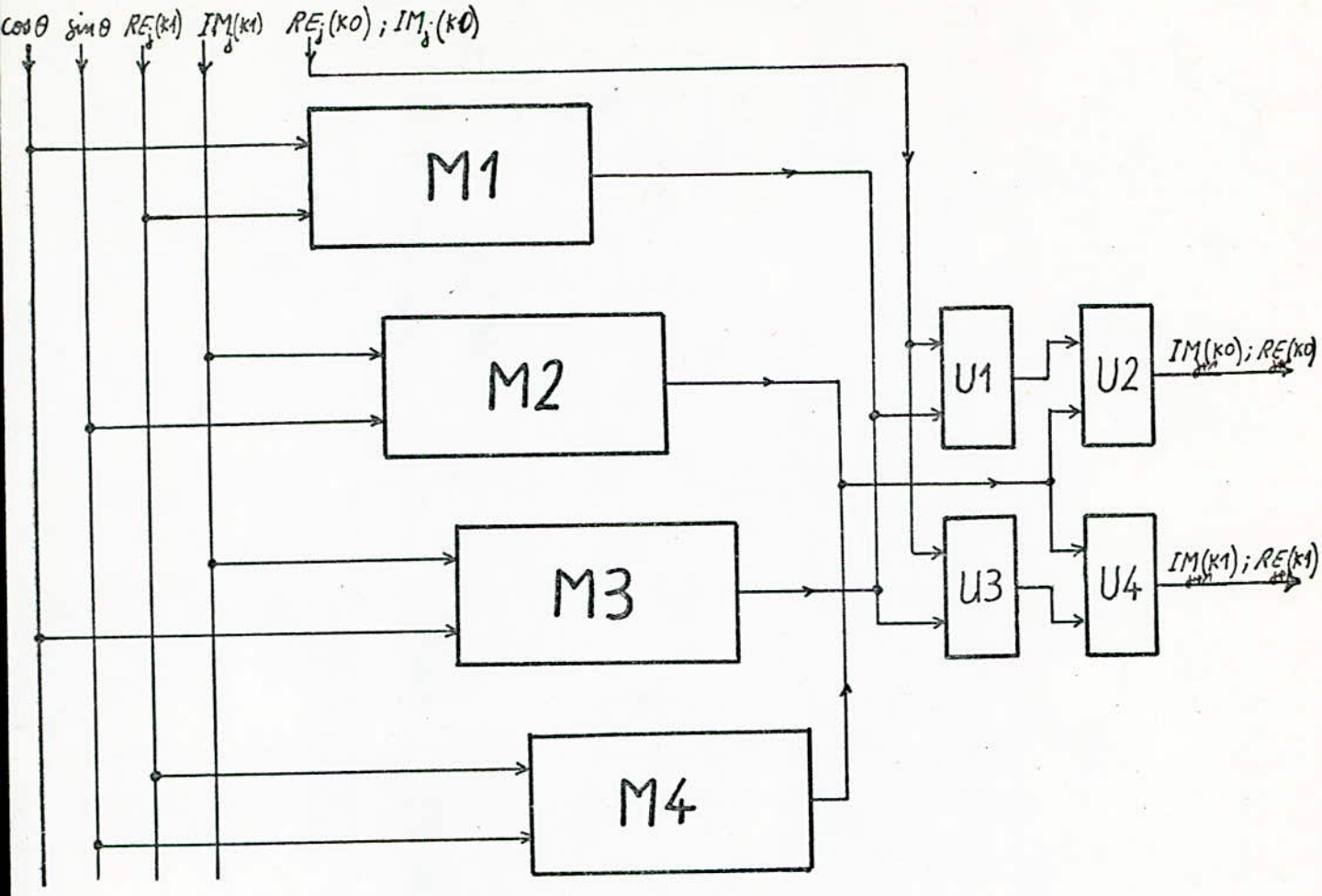


Fig. 5.6. Schéma fonctionnel de l'unité arithmétique.

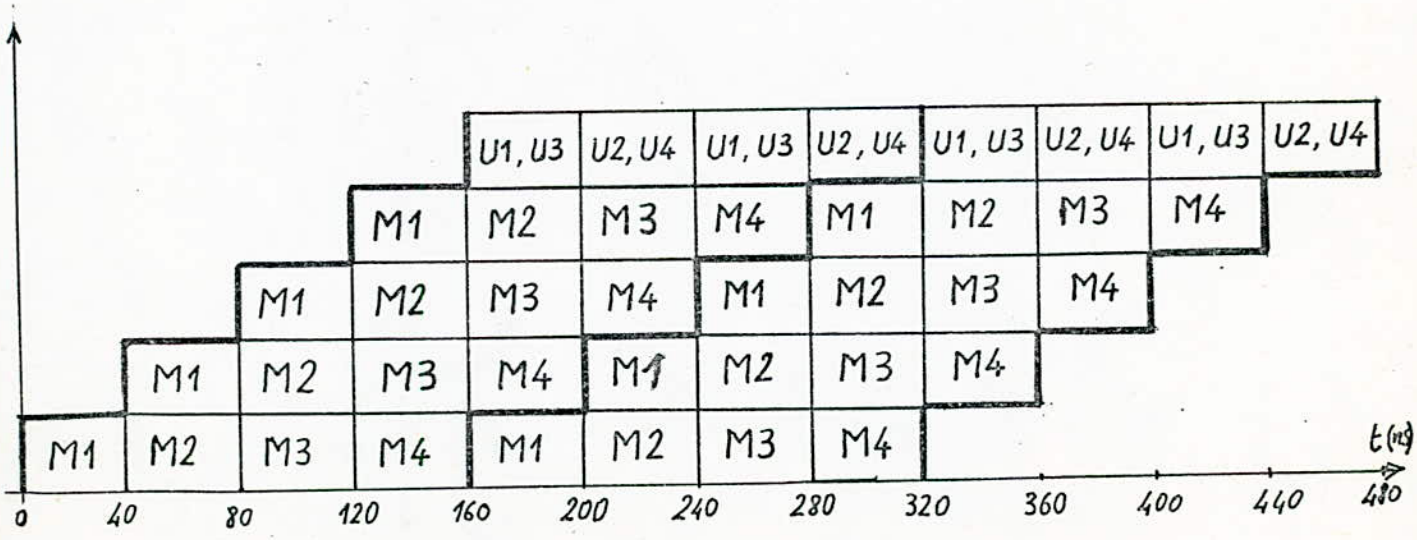


Fig. 5.7. Diagramme éspace-temps de l'unité arithmétique.

progrés significatifs ont été accomplis ces cinq dernières années et ont abouti à des circuits intégrés monolitiques spécialisés aux performances élevées.

Ces multiplieurs utilisent des technologies bipolaires et la structure arrays pour obtenir la vitesse maximale. Dans cette structure les produits partiels sont générés simultanément. Ces composants sont généralement cascadables pour permettre une extension cablée du format de travail.

D'autres multiplieurs dits séquentiels existent sur le marché et sont plus lents puisqu'ils génèrent séquentiellement les produits partiels.

Pour notre application, le choix du multiplieur est basé sur les critères suivants :

- Représentation en complèment à 2
- Multiplieurs 16 x 16 bits
- Très grande vitesse de traitement (max 160 ns)
- Registres tampons d'E/S
- Faible consommation
- Encombrement minimal
- Faible coût

Le tableau 5.1 illustre quelques caractéristiques des principaux multiplieurs 16 x 16 bits

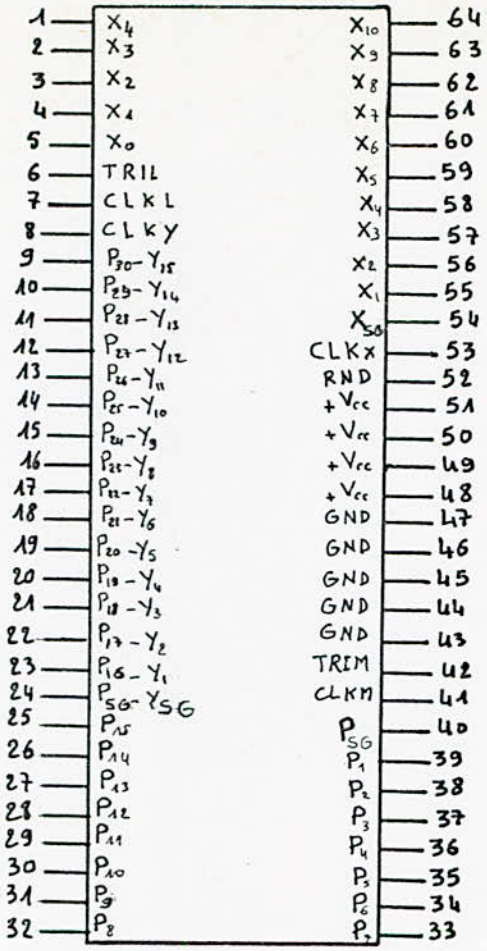
L'optimisation très poussée des performances des multiplieurs des series MPY et TDC (de TRW) a porté notre choix sur le multiplieur MPY - 16HJ dont le schéma de brochage et son diagramme temps sont représentés à la figure 5.8. Ses caractéristiques sont les suivantes :

- Multiplieur parallele 16 x 16 bits, sortie sur 32 bits
- Double précision
- Les entrées Y et les sorties LSP sont multiplexées.

TABLEAU 5.1. [6]

Multiplieur de base Employé	Nombre de boitiers	Vitesse typ	Consommation typ	Consommation Max	Représentatin	Observation
AMD 25LS2516	2	800 ns	2,3 W	-	compl à 2	Séquentiel
AMD 25S05	32	100 à 150 ns	-	30 w	compl.à 2	
AMD 25S558	* 4 + 13	80 ns	-	18 w	binaire pur ou compl à 2	Extension du format par batterie d'additionneur
MMI 67516	1	1µs	1,6 w	-	compl à 2	multiplieur/ diviseurs (séquentiel)
MMI 67558	4 * + 13	135 ns	-	18 w	-	Extension du format pat batterie d'additionneur
SN 74S274	16 * + 29	120 ns	-	30 w	binaire pur	extension du format avec SN 74S27
MPY-16 HJ (TRW)	1	100 ns	3	4	binaire pur ou compl à 2	-Registres d'E/S -Extension aisée du format avec les additionneur -Boitier pourvu d'un radiateur
MPY-16K (TRW)	1	50 ns	-	4	binaire pur ou compl à 2	
TDC 1010J (TRW)	1	115 ns	3,4	4,5	binaire pur ou compl à 2	Multiplieur avec accumulateur orienté vers le traitement du signal

* En ce qui concerne le nombre de boitier, le premier chiffre indique le nombre de multiplieurs de base. Le second indique le nombre de circuits MSI standards (additionneurs, générateurs de retenue anticipée, registres à décalages,..) qu'il faut ajouter pour la sommation partielle.



MPY-16HJ.

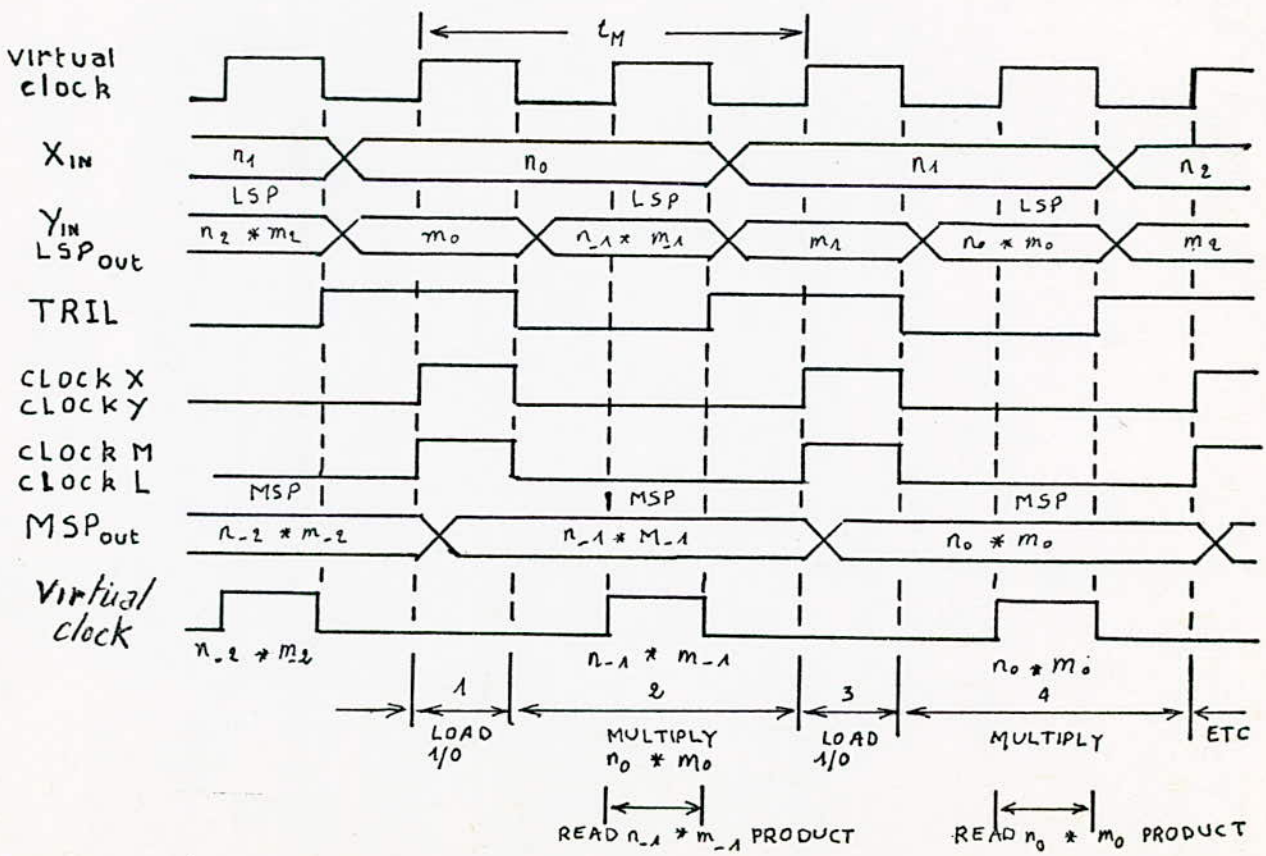


Fig. 5.8

- Registres d'entrées sorties inclus
- Entrées-sorties compatibles TTL
- Sorties trois états
- Alimentation unique + 5V
- Entrées et sorties sont en complèment à 2 ou binaire pur.

Le MPY-16 HJ a quatre entrées d'horloge, une pour chaque registre (d'entrées et de sorties). Les entrées ou les sorties sont validées lorsque les"CLK" sont à un niveau bas.

TRIM et TRIL sont les commandes respectives des trois états de MSP et LSP. La sortie est en haute impédance lorsque TRIM ou TRIL est à un niveau haut. Elle est validée lorsque TRIM ou TRIL est au niveau bas.

La commande RND sert à arrondir le résultat s'il est tranqué. Lorsque RND est à un niveau haut, un "1" est ajouté au bit le plus significatif des LSP (c'est à dire 2^{-16}), donc le multiplieur arrondie le résultat MSP.

5.2.2.3. Choix et présentation de l'additionneur/soustracteur.

Le temps d'éxécution d'une addition ou d'une soustraction sur 16 bits est de 40 ns. Donc le choix de l'additionneur-soustracteur sera basé sur les critères suivants:

- Opérations sur 16 bits
- Représentation en complèment à 2
- Le temps d'exécution 40 ns
- Facilité de commande
- Faible consommation

Pour effectuer cette opération, nous disposons du circuit SN74LS181 qui est une unité arithmétique et logique (UAL) capable d'éxécuter 16 opérations arithmétiques binaires de 2 mots de 4 bits et 16

opérations logiques. Les différentes opérations à effectuer sont commandées par les entrées (S0 S1 S2 S3). Les retenues internes sont permises en appliquant un niveau bas de tension sur l'entrée (M) du mode contrôle dans le cas des opérations arithmétiques.

Le brochage et la table de fonction de ce circuit sont données par la figure 5.8a et le tableau 5.2

La table de fonction montre que si la commande de l'addition est donnée par la configuration (S0 S1 S2 S3) celle de la soustraction sera donnée par ($\overline{S0}$ $\overline{S1}$ $\overline{S2}$ $\overline{S3}$).

Les opérations d'addition / soustraction en format 16 bits nécéssitent 4 circuits SN74LS181 traitant chacun des mots de 4 bits.

Le temps de réponse d'un circuit SN74LS181 est de 24 ns le temps de réponse de 4 circuits couplés augmente de (3 x 16) ns. La durée 16 ns étant le temps de propagation de la retenue.

La conjonction de 4 UAL avec le circuit générateur de retenues anticipées SN74182 diminue le temps de réponse et le rend égale à 44 ns.

Le brochage du circuit SN74182 est représenté à la figure 5.8b et la conjonction des circuits SN74LS181 avec le circuit SN74 182 à la figure 5.9.

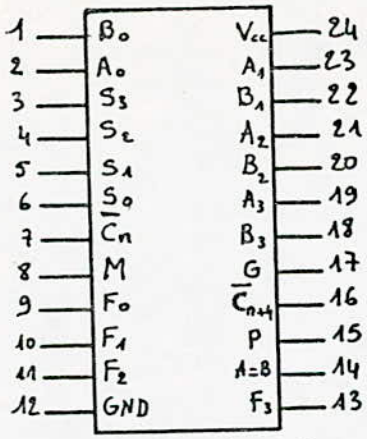
Le schéma éléctrique de principe est représenté par la figure 5.10.

5.2.3. CIRCUIT D'AIGUILLAGE DES RESULTATS.

Comme nous l'avons vu (5.2.2.1.) on obtient, à la sortie de l'UA, alternativement les groupes de valeurs $[RE_{j+1}(K0), RE_{j+1}(K1)]$ et $[IM_{j+1}(K0), IM_{j+1}(K1)]$ toutes les 160 ns.

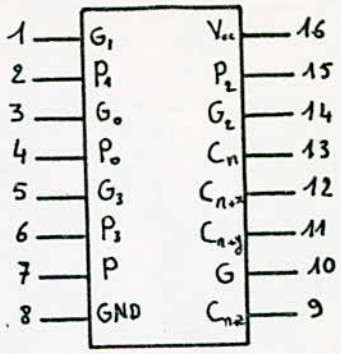
Afin de pouvoir lire ou écrire dans la même position mémoire la partie réelle et la partie imaginaire, nous avons mis au point un circuit d'aiguillage et de retard, Ce circuit nous permet d'obtenir à sa sortie

(a)



SN74LS181

(b)



SN74182

Fig. 5.8

SELECTION				ACTIVE-HIGH DATA		
				M = H	M = L; ARITHMETIC OPERATIONS	
S3	S2	S1	S0	LOGIC FUNCTIONS	$\overline{C}_n$ = H (no carry)	$\overline{C}_n$ = L (with carry)
L	L	L	L	F = $\overline{A}$	F = A	F = A PLUS 1
L	L	L	H	F = $\overline{A+B}$	F = A + B	F = (A + B) PLUS 1
L	L	H	L	F = $\overline{A}$B	F = A + $\overline{B}$	F = (A + $\overline{B}$) PLUS 1
L	L	H	H	F = 0	F = MINUS 1 (2's COMPL)	F = ZERO
L	H	L	L	F = $\overline{AB}$	F = A PLUS A$\overline{B}$	F = A PLUS A$\overline{B}$ PLUS 1
L	H	L	H	F = $\overline{B}$	F = (A + B) PLUS A$\overline{B}$	F = (A + B) PLUS A$\overline{B}$ PLUS 1
L	H	H	L	F = A ⊕ B	F = A MINUS B MINUS 1	F = A MINUS B
L	H	H	H	F = A$\overline{B}$	F = A$\overline{B}$ MINUS 1	F = A$\overline{B}$
H	L	L	L	F = $\overline{A}$ + B	F = A PLUS AB	F = A PLUS AB PLUS 1
H	L	L	H	F = $\overline{A \oplus B}$	F = A PLUS B	F = A PLUS B PLUS 1
H	L	H	L	F = B	F = (A + $\overline{B}$) PLUS AB	F = (A + $\overline{B}$) PLUS AB PLUS 1
H	L	H	H	F = AB	F = AB MINUS 1	F = AB
H	H	L	L	F = 1	F = A PLUS A*	F = A PLUS A PLUS 1
H	H	L	H	F = A + $\overline{B}$	F = (A + B) PLUS A	F = (A + B) PLUS A PLUS 1
H	H	H	L	F = A + B	F = (A + $\overline{B}$) PLUS A	F = (A + $\overline{B}$) PLUS A PLUS 1
H	H	H	H	F = A	F = A MINUS 1	F = A

Tableau 5.2

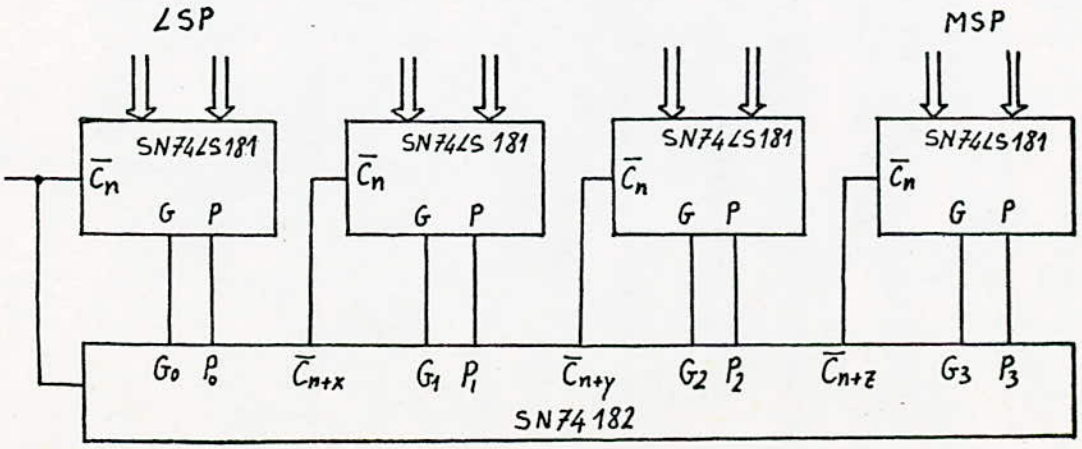


Fig. 5.9

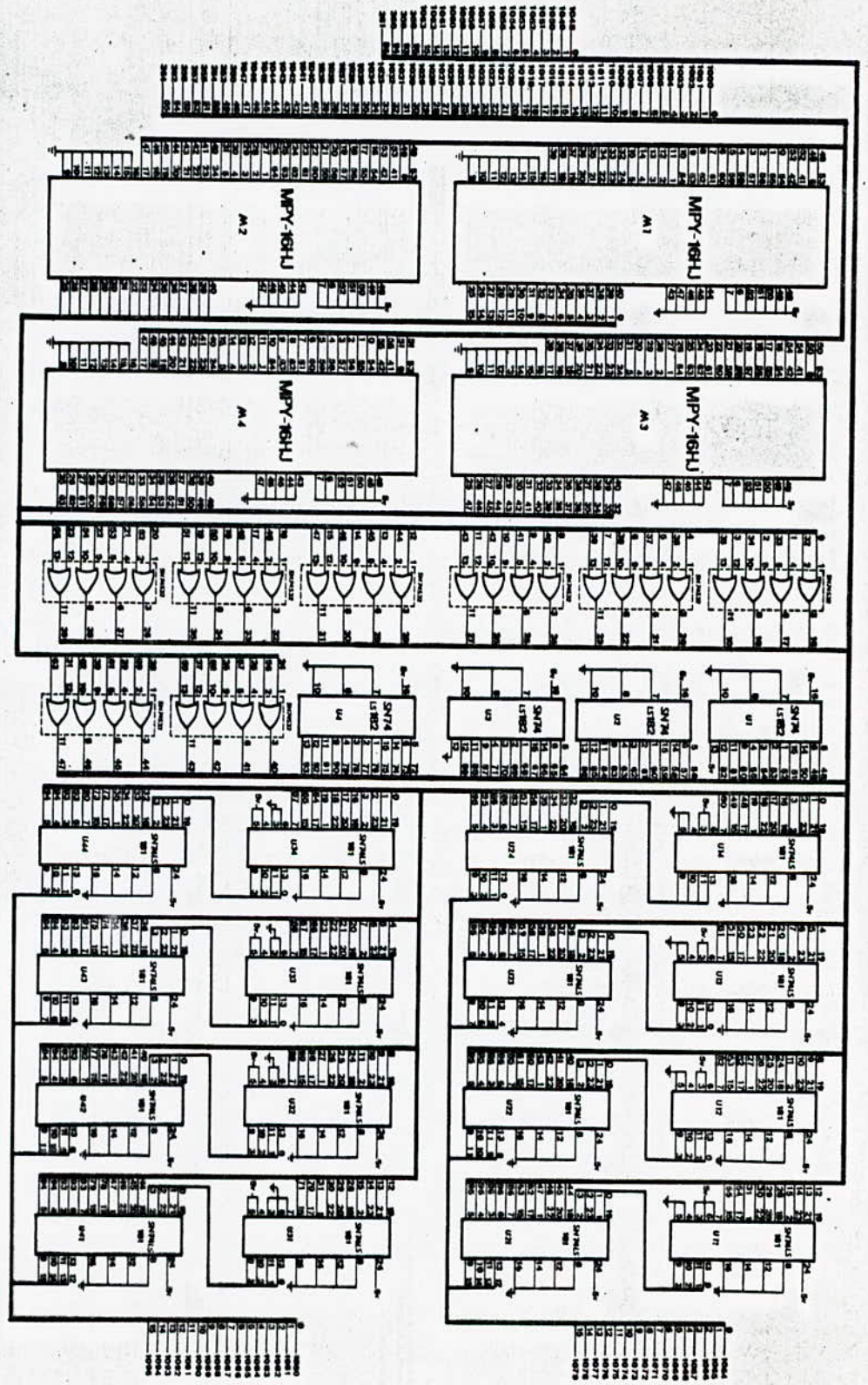


Fig.5.10 .schéma de l'U.A.

alternativement les groupes de valeurs $[RE_{j+1}(K0), IM_{j+1}(K0)]$ et $[RE_{j+1}(K1), IM_{j+1}(K1)]$ toutes les 160 nS.

La figure 5.11 et le chronogramme illustre le fonctionnement de ce circuit. Son schéma électrique de principe est représenté à la figure 5.12 Il a été réalisé à base de circuits SN74LS373 et SN74S32 pour les raisons citées en (5.2.1.).

5.3. LE COMMUTATEUR 1

Le commutateur 1 est constitué d'un démultipléxeur DX1 de mots de 16 bits dont le rôle est d'acheminer les données vers la RAM1 et la RAM3, d'un démultiplexeur de mots de 32 bits à 4 sorties qui achemine les résultats intermédiaires vers les mémoires RAM1, RAM2 et RAM3 et les résultats finals vers la RAM4. Son schéma synoptique est représenté à la figure 5.13

DX1 est réalisé par 16 démultiplexeurs (figure 5.14) d'un bit à 2 sorties et DX2 par 32 démultipléxeurs (figure 5.15) à 4 sorties. Les tables de verité de DX1 et DX2 sont respectivement représentées aux tableaux 5.3 et 5.4

Le schéma éléctrique de principe est illustré par la figure 5.16 son fonctionnement par le tableau 5.3 et 5.4 et le chronogramme.

5.4 L'UNITE MEMOIRE.

5.4.1 Organisation

L'unité mémoire a été décrite au paragraphe (4.5.1). Le choix de la mémoire dépend de la vitesse de traitement. Etant donnée que l'écriture ou la lecture s'effectue toutes les 80 ns nous avons choisi une mémoire dynamique rapide, la MCM93412DC dont le temps d'accés est de 45 ns et organisée en 256 x 4 bits.

Chaque bloc mémoire est composé de 16 boitiers reliés par un bus d'adresses commun. Ceci nous permet d'avoir accés au sous blocs R_K et au

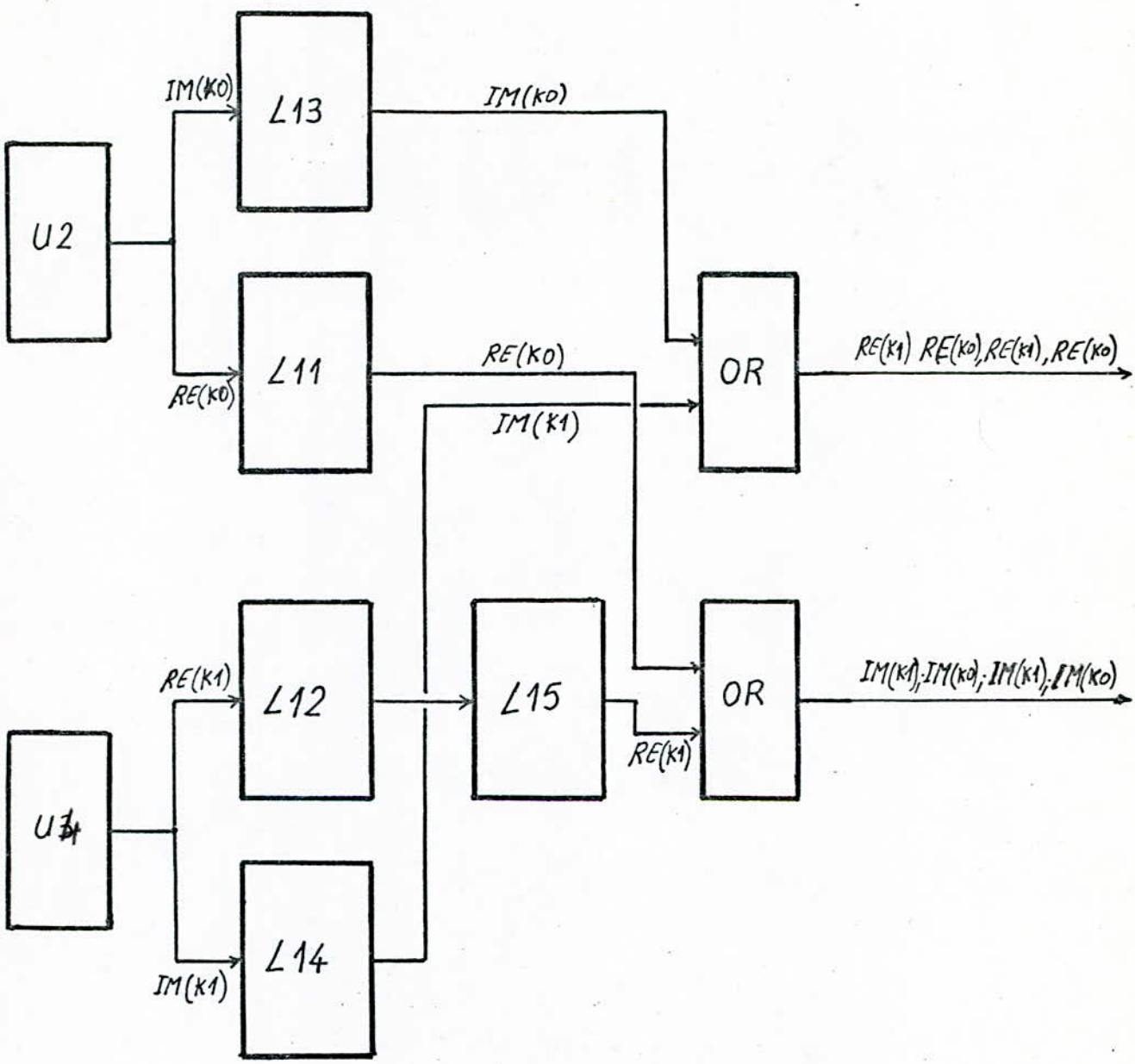


Fig. 5.11. Schéma fonctionnel du circuit d'aiguillage des résultats.

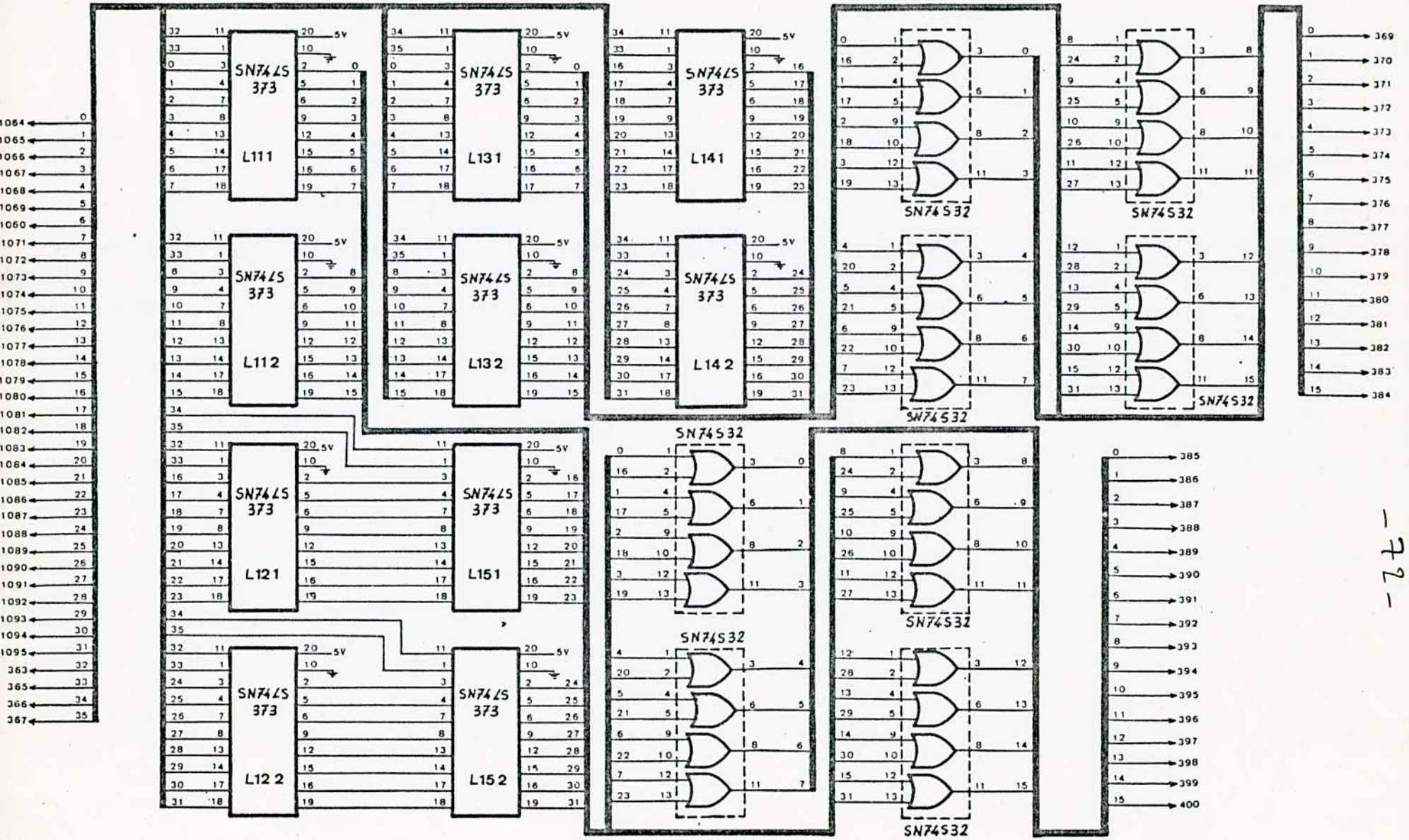


Fig.5.12 - SCHEMA ELECTRIQUE DU CIRCUIT D'AIGUILLAGE DES RESULTATS.

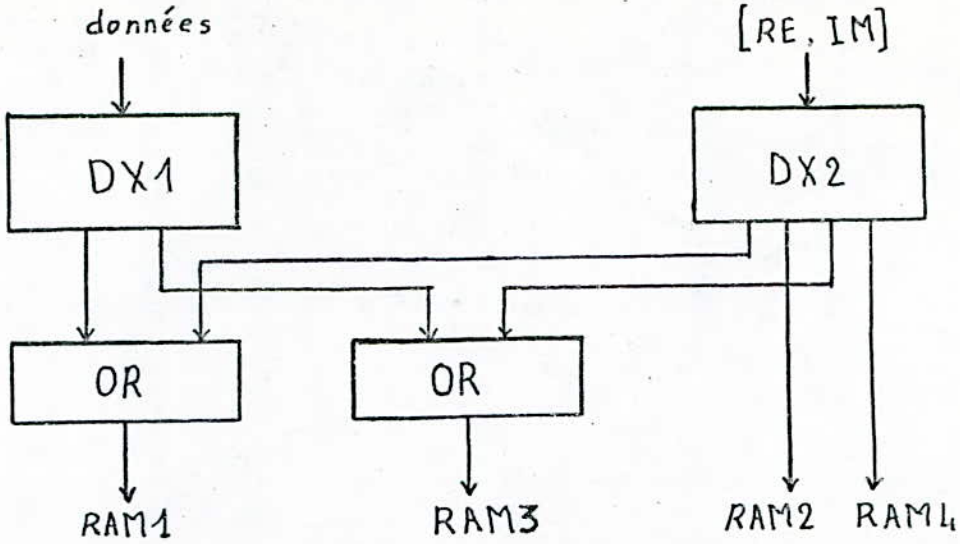


Fig. 5.13. Schéma synoptique du commutateur 1.

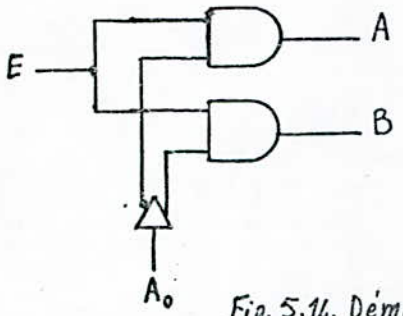


Fig. 5.14. Démultiplexeur à 2 sorties.

E	A_0	A	B
0	X	0	0
1	0	1	0
1	1	0	1

Tableau 5.3.

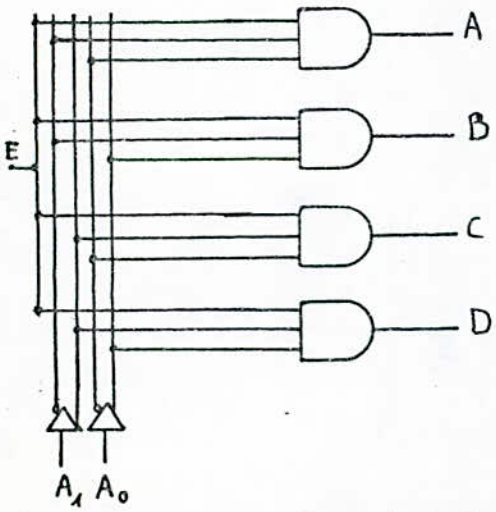


Fig. 5.15. Démultiplexeur à 4 sorties.

E	A_1	A_0	A	B	C	D
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Tableau 5.4

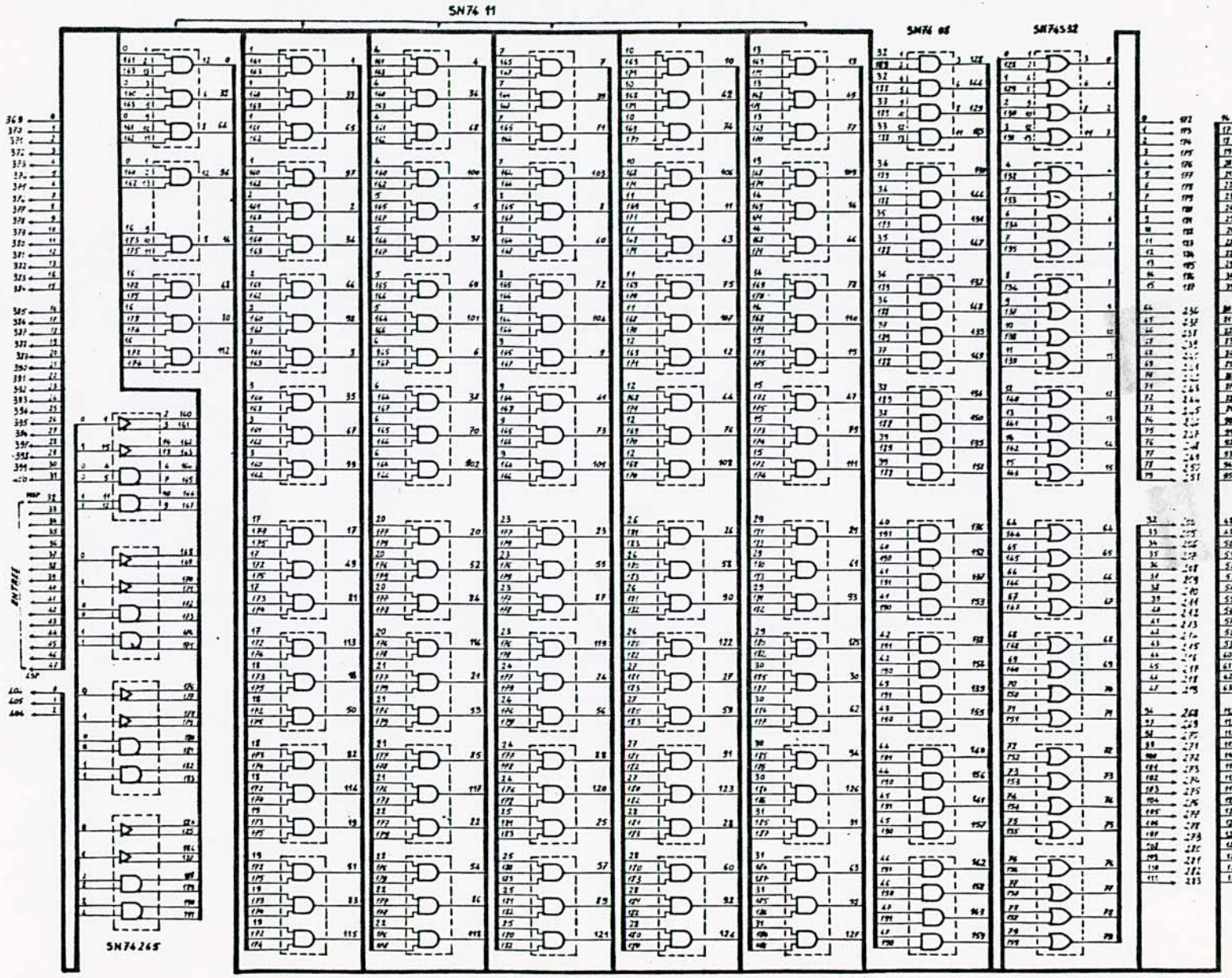


Fig. 5.16. SCHEMA ELECTRIQUE DU COMMUTATEUR 1

sous bloc I_K , qui contiennent respectivement la partie réelle et la partie imaginaire, en même temps.

Le schéma électrique de principe est représenté à la figure 5.17

Le brochage du circuit MCM93412DC est représenté en annexe

5.4.2. Fonctionnement

Initialement la RAM1 reçoit les données du coupleur et ce pendant 500 μS. Une fois ceci fait, pendant que la RAM3 reçoit un deuxième bloc de données, les memoires RAM1 et RAM2 travaillent en intermittance ; durant une itération pendant que l'une fournit les données à l'UT, l'autre reçoit les résultats intermédiaires. Durant l'itération suivante les rôles s'inversent et le cycle se répète jusqu'à la huitième itération. A la neuvième itération, les données sont fournies par la RAM1 et les résultats finals sont inscrits à la RAM4

A cet instant, le deuxième bloc de données est entièrement inscrit à la RAM3. Le cycle precédent se répète entre les différents blocs memoires. Il est clair (comme nous l'avons déjà cité au paragraphe 5.4.2.) qu'il faudrait exploiter les résultats contenus dans la RAM4 (les transférer dans un module mémoire du système de traitement du signal (figure 4.1.) durant le temps de traitement des huits itérations afin qu'elle soit disponible à recevoir le prochain résultat.

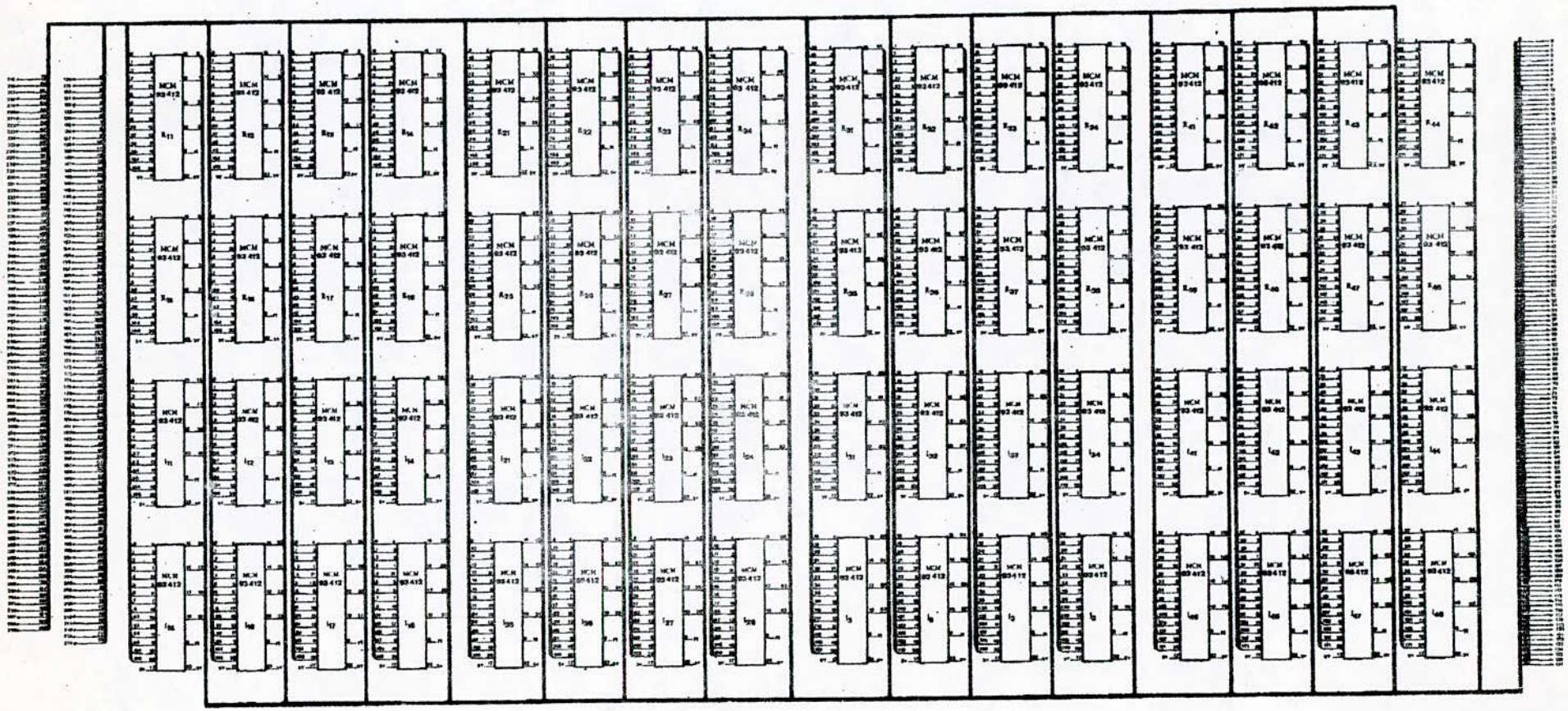
5.4.3. Réalisation.

La réalisation a été effectuée à l'aide de mémoires MCM93415DC de capacité 1K x 1bit. La figure 5.17a illustre le schéma electrique de principe de deux blocs mémoires (512 x 24 bits chacun).

La MCM93415DC est caractérisée par :

- Un très faible temps d'accés : 35 nS
- Des entrées - sorties séparées.
- Une consommation de 0,5 mW / bit

Son brochage est représenté en annexe.



CENTRE DE DEVELOPPEMENT DES TECHNOLOGIES AVANCÉES (C.D.T.A.)
- LABORATOIRE ARCHITECTURE DES SYSTÈMES -

PROCESSEUR RAPIDE DE TRAITEMENT DU SIGNAL

SCHEMA ELECTRIQUE DE L'UNITE MEMOIRE

M. BESSALAH
M. RAMDA
Mme MOUZAL

PROJET DE FIN D'ETUDES

JANVIER 8

Fig 5.17

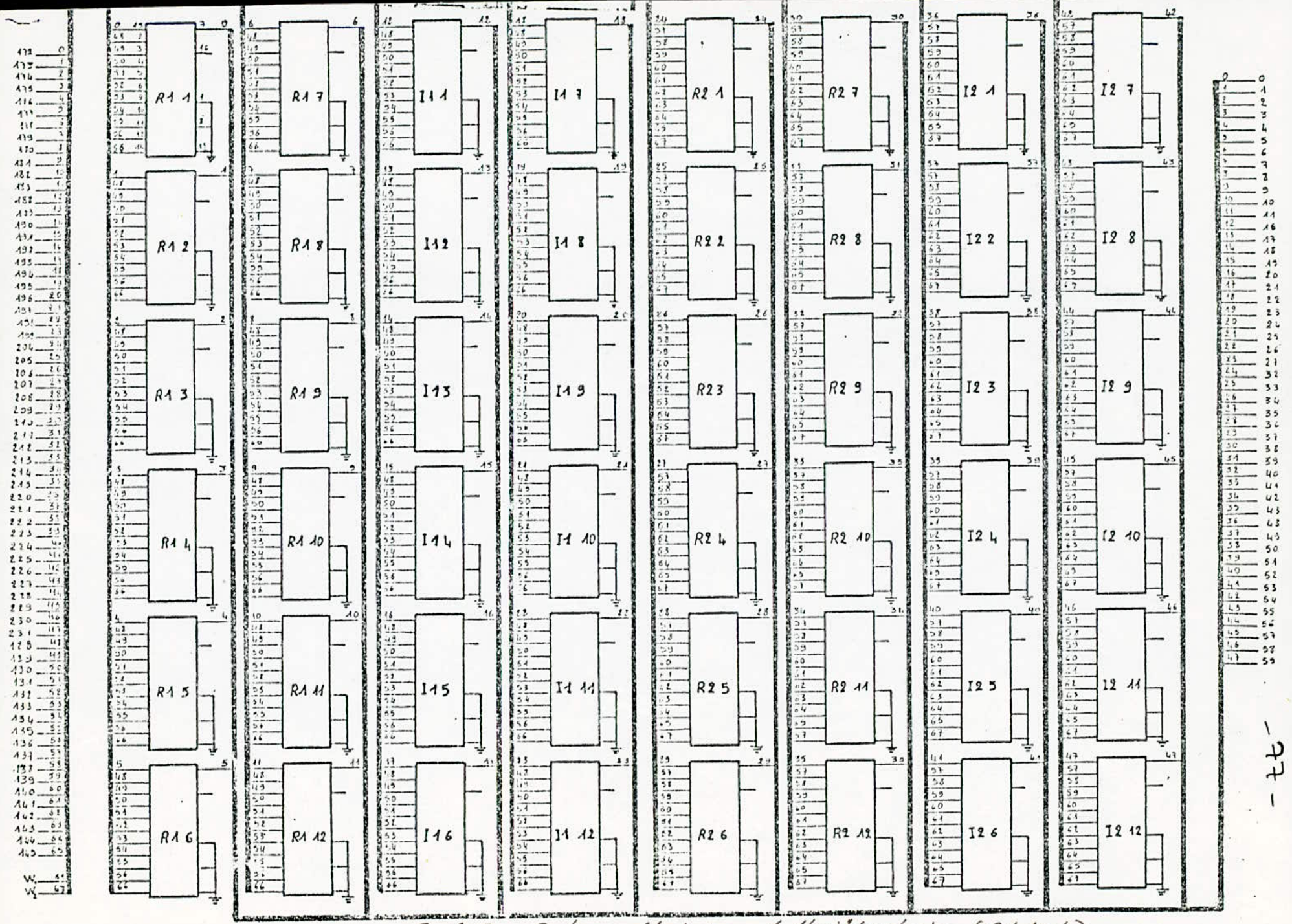


Fig. 5.17a Schema électrique de l'unité mémoire (Réalisée.)

SCHEMA ELECTRIQUE DE L'UNITÉ MEMOIRE

5.4.4. MEMOIRE PROM.

Deux mémoires PROM sont utilisées pour stocker les valeurs des coefficients de rotations W_N.

Le schéma electrique est représenté à la figure 5.17b

5.5. LE COMMUTATEUR 2

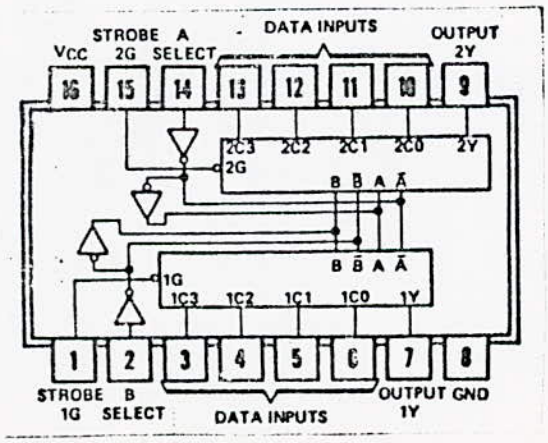
Le rôle du commutateur 2 est d'acheminer les données (sorties des blocs mémoires RAM1, RAM2, et RAM3) vers l'entrée de l'unité de traitement.

Ce commutateur est composé de 16 modules multiplexeurs de 4 entrées de 2 bits . Chaque module traite un bit de la partie réelle et un bit de la partie imaginaire.

Pour notre réalisation, nous avons utilisé le multiplexeur SN74LS153 qui a les caractéristiques suivantes :

- Faible consommation
- Selectionne un mot de 2 bits parmi 4 mots (chaque adresse permet de selectionner en sortie (1Y et 2Y).

Son brochage est sa table de vérité son illustré ci-dessous: son schéma electrique est représenté à la figure 5.17c.



SN74LS153.

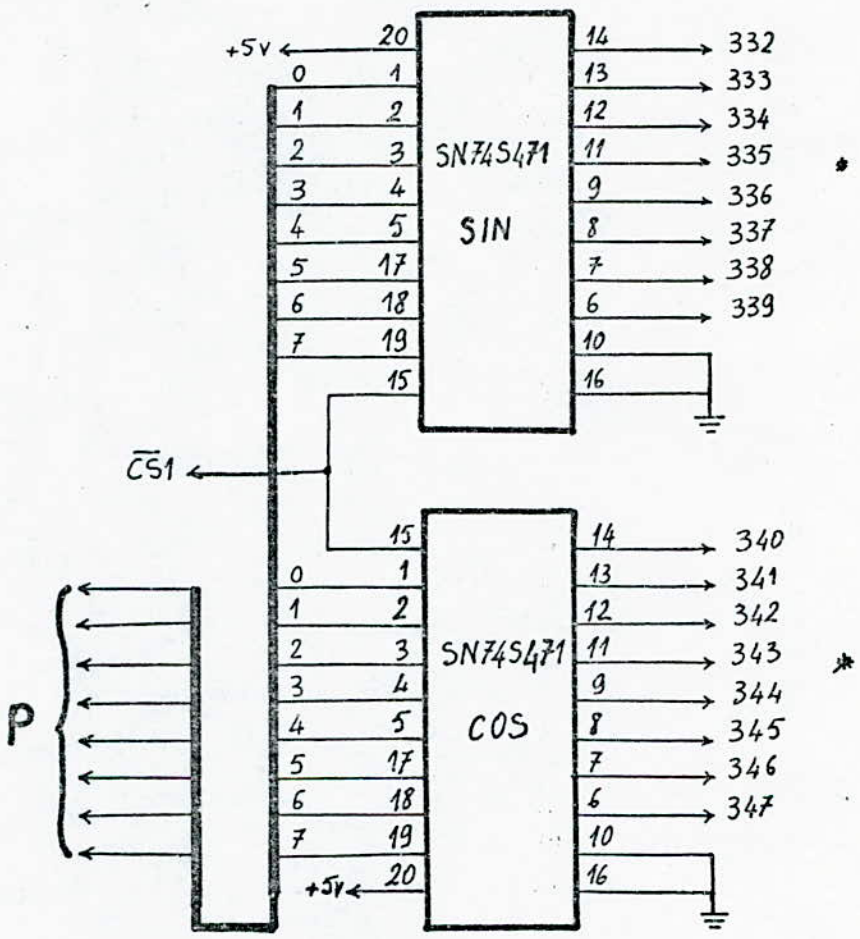


Fig. 5.17b_ Schéma électrique de la mémoire PROM.

(Voir brochage en annexe.)

* Sorties reliées au circuit d'aiguillage de données (Fig. 5.5a)

Table de verité du SN74LS153

SELECT INPUTS		DATA INPUTS				STROBE	DATA INPUTS				STROBE	OUTPUT	OUTPUT
B	A	1C0	1C1	1C2	1C3	1G	2C0	2C1	2C2	2C3	2G	1Y	2Y
X	X	X	X	X	X	H	X	X	X	X	H	L	L
L	L	L	X	X	X	L	L	X	X	X	L	L	L
L	L	H	X	X	X	L	H	X	X	X	L	H	H
L	H	X	L	X	X	L	X	L	X	X	L	L	L
L	H	X	H	X	X	L	X	H	X	X	L	H	H
H	L	X	X	L	X	L	X	X	L	X	L	L	L
H	L	X	X	H	X	L	X	X	H	X	L	H	H
H	H	X	X	X	L	L	X	X	X	L	L	L	L
H	H	X	X	X	H	L	X	X	X	H	L	H	H

5.6 SEQUENCEURS D'ADRESSES

La génération des adresses de lecture et d'écriture des opérandes (données initiales, intermédiaires, finales et coefficients) est effectuée par différents séquenceurs que l'on désigne par Seq 1, Seq 2 et Seq 3.

Le séquenceur Seq 1 formera les adresses d'écriture des données initiales issues du CAD ou des coupleurs paralleles.

Le séquenceur Seq 2 doit générer les adresses de lecture et d'écriture des valeurs intermédiaires des coefficients de Fourier y compris la lecture des données initiales et l'écriture des résultats finals.

Le séquenceur Seq 3 formera les adresses des coefficients de rotations.

5.6.1 LE SEQUENCEUR Seq 1

5.6.1.1. Fonctionnement

Les données d'échantillonnage des signaux sont issues des CAD ou des coupleurs paralleles suivant un ordonnancement direct. Elles doivent être logées en mémoire selon les adresses fournies par le Seq 1 dont le fonctionnement peut être déduit de l'algorithme A et plus particulièrement de la matrice R d'inversion numérique.

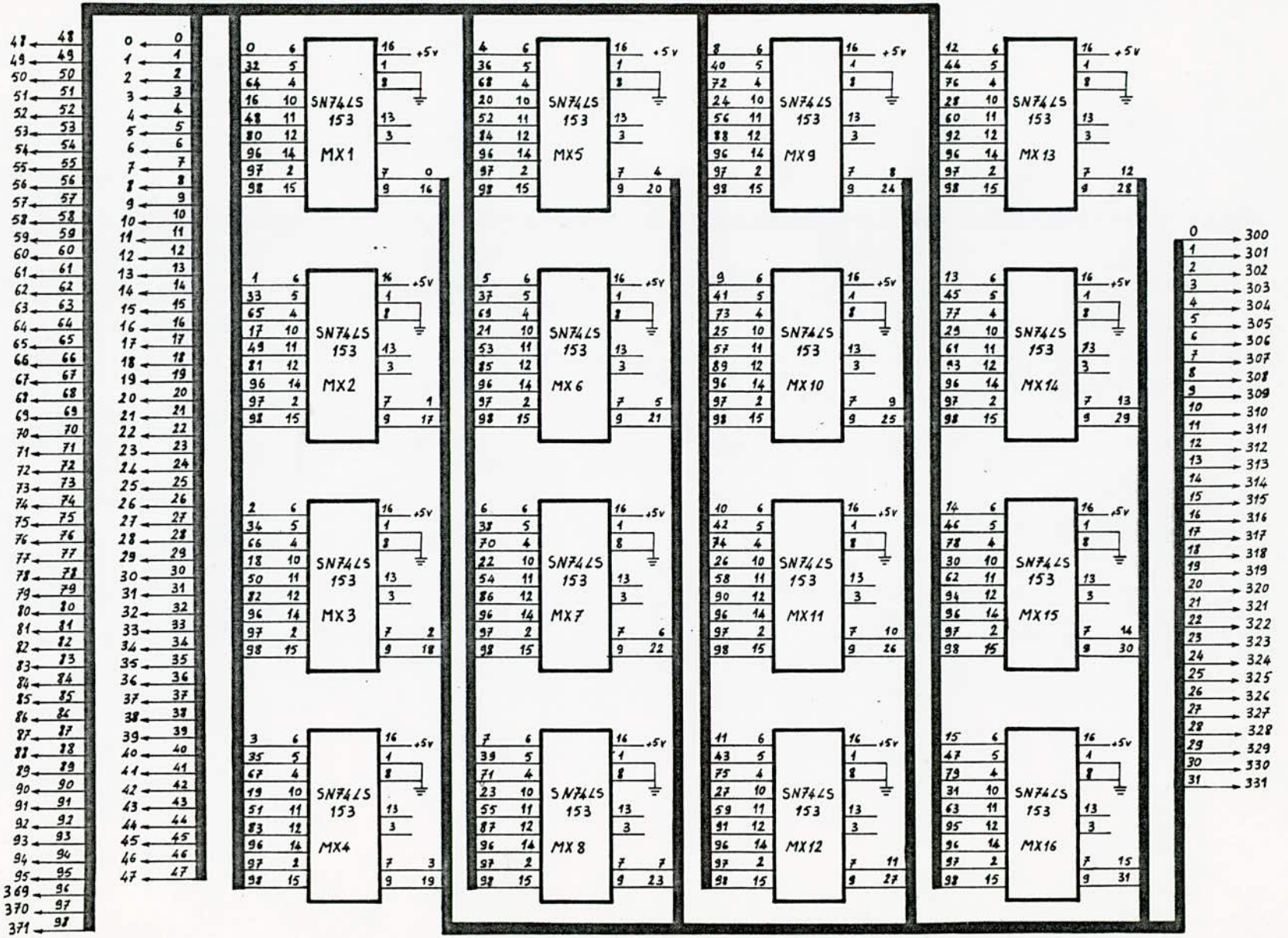
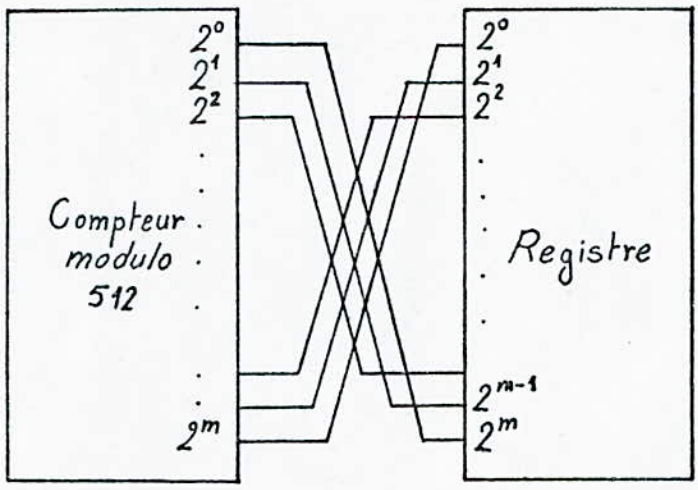


Fig.5.17c-SCHEMA ELECTRIQUE DU COMMUTATEUR 2.

Cette matrice permet d'effectuer l'opération de desembrouillage du vecteur x suivant l'organigramme figure 5.18

5.6.1.2. Structure du Seq 1

La réalisation du séquenceur peut être faite à l'aide d'un compteur modulo 512 et d'un registre de 9 bits. Les sorties MSB et LSB du compteur sont connectées successivement avec les entrées LSB et MSB du registre suivant le schéma ci-dessous :



Le séquenceur Seq 1 travaille à une fréquence $\log_2 N$ fois plus faible que la fréquence fondamentale, qui est celle des séquenceurs Seq 2 et Seq 3.

Dans le but de faciliter l'apprehension de l'algorithme de fonctionnement des séquenceurs référons nous à la relation (2.30a) et la figure (5.1) où le nombre d'échantillons est fixé à N = 16.

Dans ce cas précis, l'opération de désembrouillage est illustrée par le tableau 5.5, conçu sur la base de la méthode du binaire réfléchi.

Comme le montre la figure 5.1, le stockage des données d'entrée éxige 16 cellules mémoires numérotées respectivement mo, m1, ..., m15

La première étape de traitement s'acheve par le transfert des données x (n), n = o, 1, ... , 15 dans les cellules mémoires suivant l'opération R.x de la manière suivante :

mo ⟵	x(o)	m8 ⟵	x(1)
m1 ⟵	x(8)	m9 ⟵	x(9)
m2 ⟵	x(4)	m10 ⟵	x(5)
m3 ⟵	x(12)	m11 ⟵	x(13)
m4 ⟵	x(2)	m12 ⟵	x(3)
m5 ⟵	x(10)	m13 ⟵	x(11)
m6 ⟵	x(6)	m14 ⟵	x(7)
m7 ⟵	x(14)	m15 ⟵	x(15)

5.6.2. SEQUENCEUR Seq 2

5.6.2.1. Fonctionnement

Le fonctionnement du séquenceur Seq 2 est simulé par la matrice S(j) de l'algorithme A. Effectivement la distribution des " 1 " de la matrice S(j) diffère d'une itération à une autre et permet de définir les positions des éléments des vecteurs x, X_1 ,....., X_{m-1} engagés dans la même opération papillon.

La première itération de l'algorithme FFT (deuxième étape de traitement) qui revient à l'éxécution de l'opération ($G_2 \otimes I_8$) R.x consiste à effectuer huit opérations papillons sur huit pairs d'éléments du vecteur x désembrouillé. Comme le montre l'expression $S(1) = G_2 \otimes I_8$ les opérandes de la paire papillon sont situés à un pas $2^\circ = 1$ l'un de l'autre.

Le fait que M(1) soit égale à la matrice unité I_{16} signifie que lors de la première itération, aucune multiplication n'est effectuée. En conséquent pour cette itération les opérandes des opérations papillons seront extraits des cellules mémoires (mo, m1), (m2, m3), (m4, m5), (m6, m7), (m8, m9), (m10, m11) , (m12, m13) , (m14, m15)

Les resultats RE_1 (Ko), RE_1 (K1), IM_1 (KO), IM_1 (K1) seront transmis dans les cellules mémoires ayant les mêmes adresses que les opérandes correspondants.

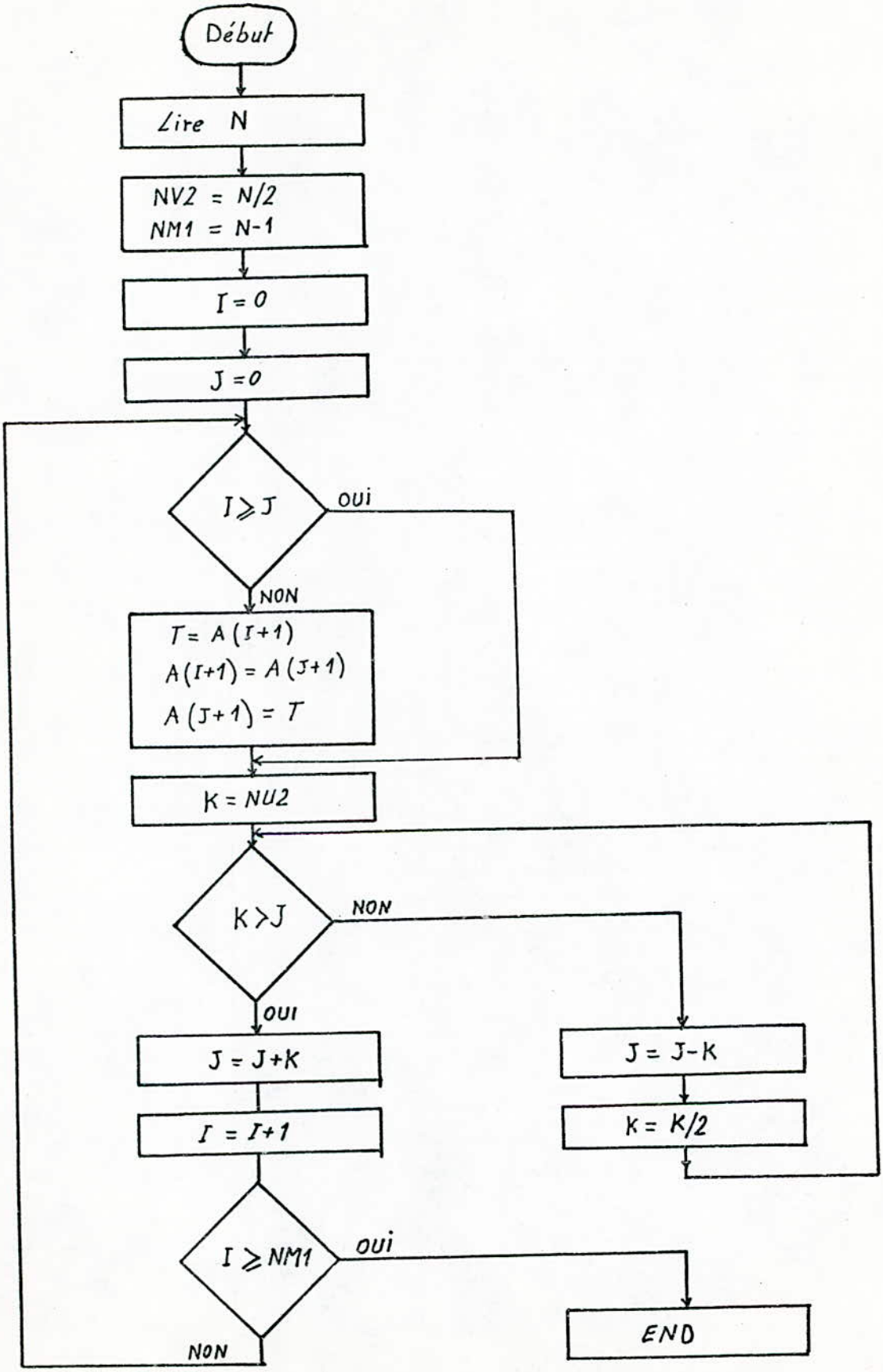

Début
Lire N
NV2 = N/2
NM1 = N-1
I = 0
J = 0
I ⩾ J
OUI
NON
T = A (I+1)
A (I+1) = A (J+1)
A (J+1) = T
K = NU2
K > J
NON
OUI
J = J+K
I = I+1
J = J-K
K = K/2
I ⩾ NM1
OUI
NON
END

Fig. 5.18.

Indice	Valeur binaire	Binaire réfléchi	Valeur
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

Tableau.5.5. Illustration de l'opération désembrouillage.

Ceci signifie que l'unité d'adressage des opérandes et des résultats seront identiques du point de vue fonctionnement à un temps T d'initialisation près, où T équivaut à l'inertie de l'unité de traitement

Après l'achèvement de la première itération est entamée la deuxième itération qui consiste en l'éxécution de l'opération $(I_2 \otimes G_2 \otimes I_4)(D_4 \otimes I_4)$; (fig. 5.1).

Suivant la matrice $(I_2 \otimes G_2 \otimes I_4)$ les opérandes des papillons sont disposés à partir de m0 avec un pas de $2^1 = 2$.

La matrice $(D_4 \otimes I_4)$ des coefficients de rotation témoigne de l'éxistence de la multiplication dans cette itération. En conséquent, pour la deuxième itération les données des papillons sont lues à partir des cellules memoires suivantes :

$(m0, m2, w^0), (m1, m3, w^4), (m4, m6, w^0), (m5, m7, w^4)$

$(m8, m10, w^0), (m9, m11, w^4), (m12, m14, w^0), (m13, m15, w^4)$

Les autres itérations s'accomplissent de la même manière généralement à l'itération j sont utilisées des données, situées à un pas 2^{J-1} l'un de l'autre ainsi que 2^{J-1} coefficients de rotation dont les valeurs sont les suivantes :

$$W_N^0 , W_N^{N/2^j} , W^{2.N/2^j} , \ldots\ldots , W^{N(2^{j-1}-1)/2^j}$$

Pour la quatrième itération de l'exemple traité les données des opérations papillons sont lues à partir des cellules mémoires suivantes :

$(m0, m8, w^0), (m1, m9, w^1), (m2, m10, w^2), (m3, m11, w^3)$

$(m4, m12, w^4), (m5, m13, w^5), (m6, m14, w^6), (m7, m15, w^7)$

Ce qui correspond aussi bien à l'expression (2.30a) et à la représentation graphique illustrée par la figure 5.1

Le tableau 5.6 représente l'organisation des adresses des données

et des coefficients de rotation pour le cas N= 16

N° de l'itération	Représentation binaire $k \in \{0,1,\ldots,7\}$	Inversion numérique de k	Permutation du bit j avec LSB -K0-	Iversion du bit j -K1-	Adresse de w -P-
000	0000	0000	0000	0001	000
	0001	1000	1000	1001	000
	0010	0100	0100	0101	000
1°	0011	1100	1100	1101	000
itération	0100	0010	0010	0011	000
	0101	1010	1010	1011	000
j=1	0110	0110	0110	0111	000
	0111	1110	1110	1111	000
001	0000	0000	0000	0010	000
	0001	1000	1000	1010	000
	0010	0100	0100	0110	000
2°	0011	1100	1100	1110	000
itération	0100	0010	0001	0011	100
	0101	1010	1001	1011	100
j=2	0110	0110	0101	0111	100
	0111	1110	1101	1111	100
010	0000	0000	0000	0100	000
	0001	1000	1000	1100	000
	0010	0100	0001	0101	010
3°	0011	1100	1001	1101	010
itération	0100	0010	0010	0110	100
	0101	1010	1010	1110	100
j=3	0110	0110	0011	0111	110
	0111	1110	1011	1111	110
011	0000	0000	0000	1000	000
	0001	1000	0001	1001	001
	0010	0100	0100	1100	100
4°	0011	1100	0101	1101	101
itération	0100	0010	0010	1010	010
	0101	1010	0011	1011	011
j=4	0110	0110	0110	1110	110
	0111	1110	0111	1111	111

Tableau.5.6. Illustration de l'organisation des adresses des données et des coefficients de rotation pour N=16.

L'étude du fonctionnement des séquenceurs que nous venons d'effectuer permet de faire les remarques suivantes :

i) Les séquenceurs doivent générer simultanément les adresses KO, K1 et P (P adresses de lecture des coefficients de rotation) des opérandes de l'opération papillon correspondante.

ii) Les adresses KO et K1 sont utilisées aussi bien pour la lecture des opérandes que pour l'écriture des résultats.

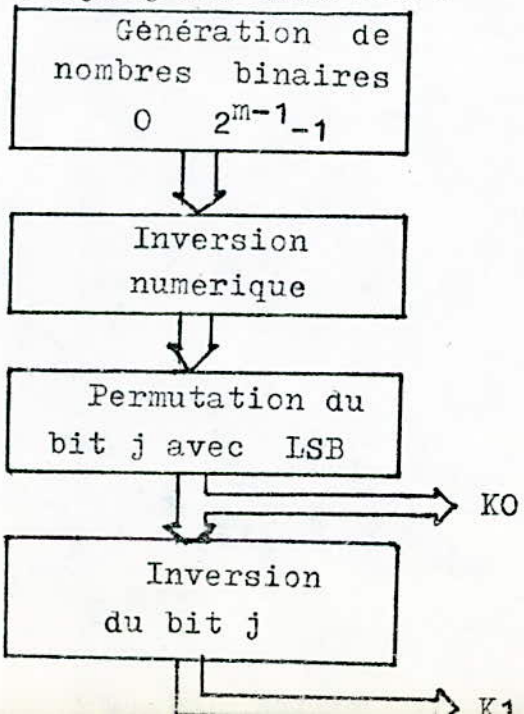
iii) Le mode d'adressage est immédiat

5.6.2.2. Structure du Seq 2.

La réalisation de séquenceur Seq 2 peut être effectuée de diverses manières : par l'utilisation d'unités arithmétiques et logiques microprogrammées, de mémoires mortes où seront stockées les differentes adresses, etc ...

Nous avons opté pour une rélisation des séquenceurs dont le temps de réponse est compatible avec les éxigences de vitesse posées par l'unité de traitement. Effectivement toutes les 80 nS cette dernière doit être alimentée en opérandes lues des mémoires RAM.

A partir du tableau 5.6, on peut remarquer que la formation des adresses KO et K1 est effectuée en plusieurs séquences que nous présenterons sous forme d'organigramme comme suit :



La réalisation de ces séquences est illustrée par le schéma fonctionnel de la figure 5.19

La génération des nombres de 0 à 255 est effectuée par un compteur modulo 256, dont les sorties sont reliées à un registre d'inversion de tel sorte que le MSB du compteur est connecté au LSB du registre et vice versa. Cette façon de connecter des sorties du compteur avec les entrées du registre permet de réaliser la séquence d'inversion numérique.

A l'aide d'un décodeur et d'un multiplexeur, le compteur modulo 9 des itérations realise les séquences de permutation du bit j avec le LSB. Les adresses K0 et K1 sont obtenues en parallele par l'intermédiaire de portes logiques.

5.6.3 Le séquenceur Seq 3.

Comme le montre le tableau 5.6, les adresses P des coefficients de rotations peuvent être obtenues à partir de K0 ou de K1. Il suffit de faire pour la première itération 2^{m-1} décalages à gauche. Pour la deuxième itération 2^{m-2} décalages, pour la $J^{ième}$ itération 2^{m-j} décalages et prendre dans tous les cas 2^{m-1} bits LSP comme adresse des coefficients de rotation.

Le séquenceur d'adresse P peut être réalisé à base de registres à décalage seulement dans ce cas le temps nécéssaire à la formation des adresses sera différent d'une itération à une autre. Ceci nous a fait opté pour un circuit plus complexe mais invariant aux itérations du point de vue temps de génération des adresses.

Le schéma fonctionnel de ce circuit est représenté à la figure 5.20

5.7 UNITE DE CONTROLE

On se limitera à la conception de l'unité de controle de l'UT

Cette unité est constituée essentiellement de l'horloge et du distributeur d'horloge.

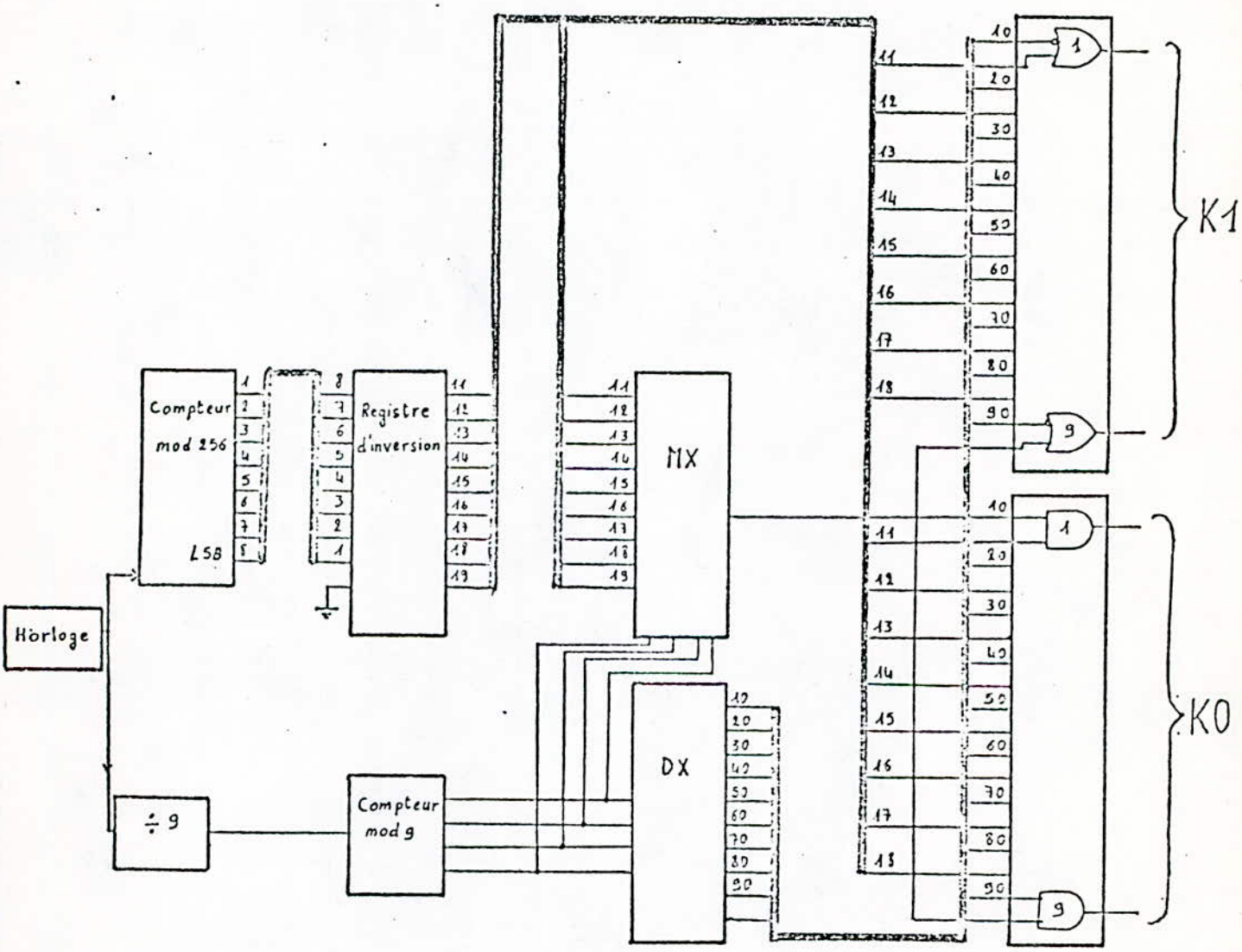

Compteur
mod 256
LSB
Registre
d'inversion
MX
DX
Horloge
÷ 9
Compteur
mod 9
K1
K0

Fig. 5.19

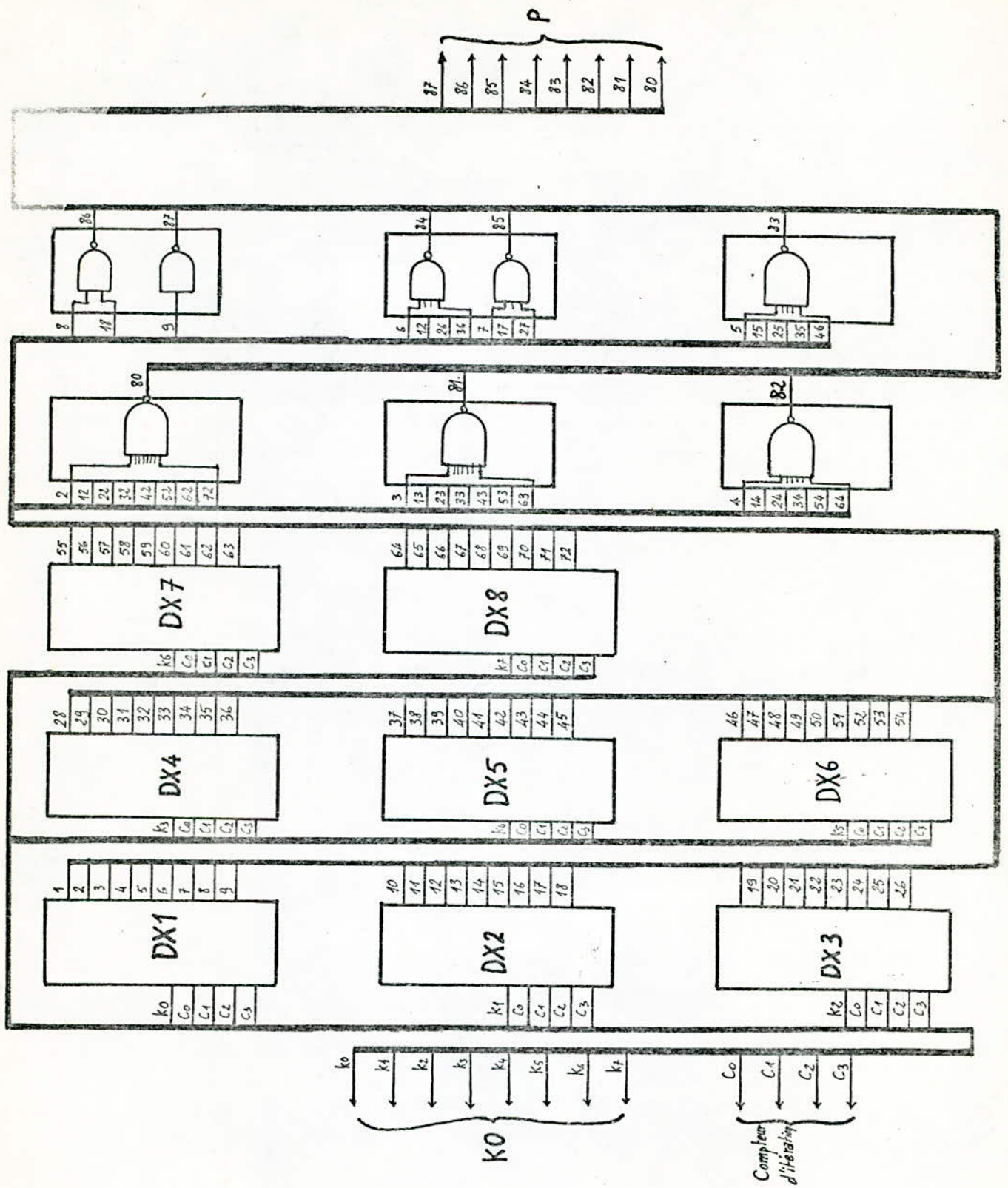

P
87
86
85
84
83
82
81
80
DX1
DX2
DX3
DX4
DX5
DX6
DX7
DX8
K0
Compteur d'itération

Fig. 5.20

5.7.1. HORLOGE

Elle doit générer des signaux à une fréquence de 25 MHZ. Sa réalisation a été faite à base de circuit SN74S124 son calcul est effectué comme suit :

$$f = \frac{5.10^{-4}}{C}$$ avec C en Farads et f en Hertz.

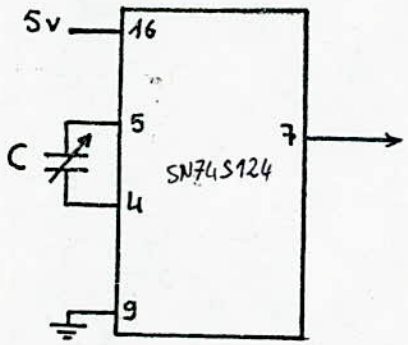
$$C = \frac{5.10^{-4}}{f}$$

$f = 25.10^{6}$ Hz donc $C = 20$ pF

5.7.2. DISTRIBUTEUR D'HORLOGE

Son rôle est de générer les signaux indispensables à la commande des éléments de l'UT et qui sont illustrés par le chronogramme.

Les circuits générant ces signaux composés d'un compteur modulo 4 de 2 portes NOR et OR, d'un driver à deux sorties complèmentaires et de 3 pairs de registres placés en cascades. Aux sorties de ces registres on obtient les signaux de commandes correspondants .



Horloge

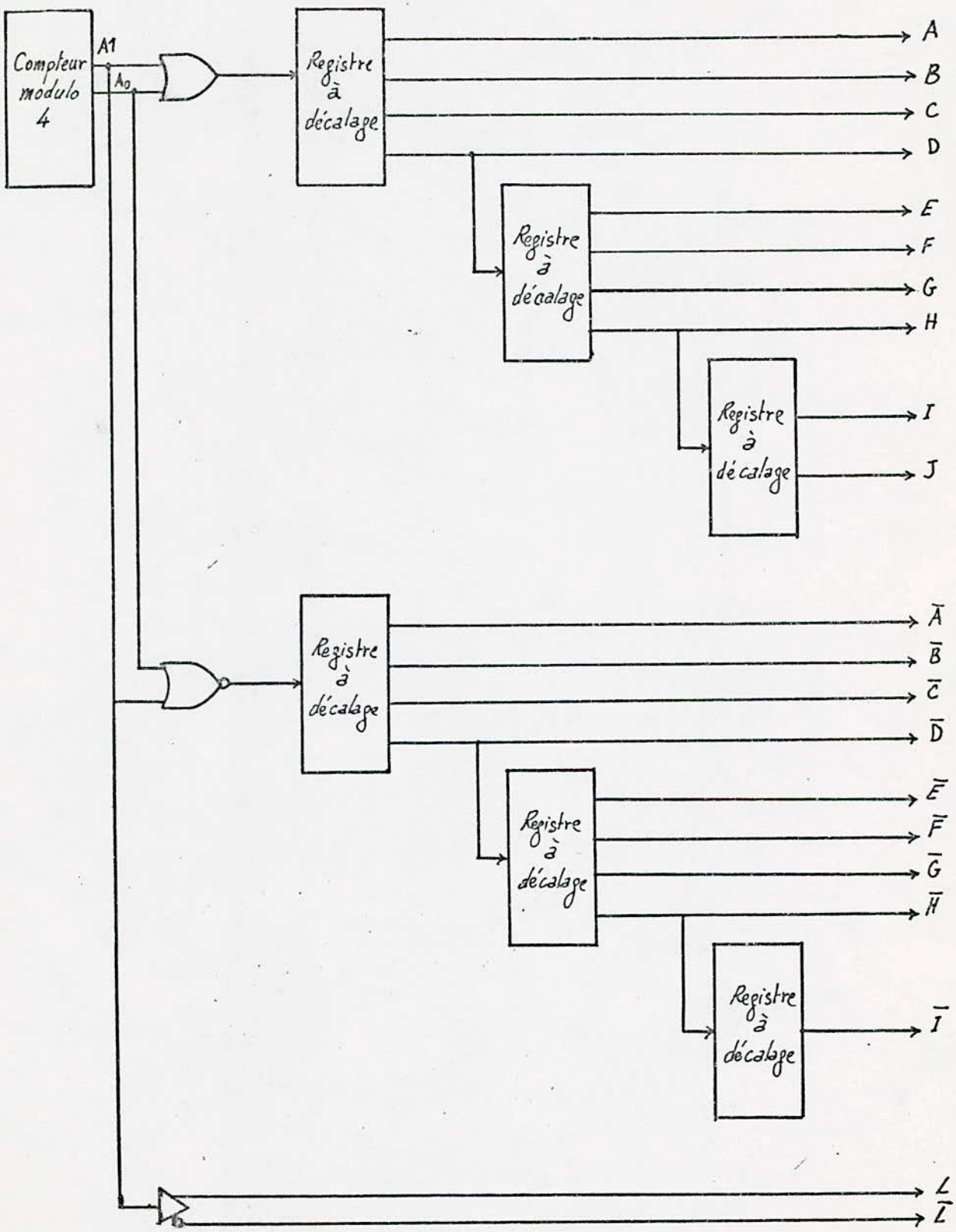


Fig. 5.21 Schéma fonctionnel de l'unité de controle.

C O N C L U S I O N

L'étude de la transformée de Fourier et des domaines de son application a montré que les algorithmes de FFT (Fast Fourier Transform) demeurent d'actualité, leur nombre ne fait que croitre, et ce malgré l'apparition de nouvelles transformées orthogonales.

Dans le but de faire le choix d'un algorithme de FFT compatible avec l'objectif qu'on s'était fixé, nous avons éxaminé de différents types d'algorithmes de la FFT suivant une classification donnée. Pour des raisons de symetrie et de précision nous avons opté pour la classe d'algorithme A.

L'étude des aspects algorithmiques et structurels des systèmes de traitement de données nous a permis de constater que l'éxécution de la FFT par des systèmes informatiques de la classe SISD (single flow instruction single flow data) ne pouvait satisfaire aux éxigences de rapidité posées par certaines applications de traitement du signal.

Après l'analyse des différentes structures paralleles et pipelines des processeurs FFT, nous avons opté pour une structure du type SIMD où sont inclus les avantages de traitements paralleles et pipelines. Tout en permettant d'atteindre de grandes vitesses (moins de 500 micro-secondes pour le traitement de 512 échontillons), une telle structure est économiquement avantageuse du fait de l'utilisation d'une seule unité de traitement

Le principe pipeline et la structure matricielle ont permis de réduire considérablement le nombre d'entrées / sorties de l'unité de traitement, ce qui permettera la réalisation sous forme d'un cristal par l'utilisation de la technologie des circuits prédiffusés

L'objectif du projet de fin d'étude qui consistait en la conception et la réalisation d'un processeur FFT a été partiellement atteint du fait que nos efforts ont été porté sur l'étude, la conception et la réalisation de l'unité de traitement, la carte mémoire et la carte commutateurs.

Ce travail s'achevera par la réalisation de l'unité de controle dont la conception a été largement entamée.

B I B L I O G R A P H I E

[1] BERNSTEIN A.J. " Analysis of programms for parallel processing " IEE Transaction on electronics computers, EC 15 , 5 October 1966

[2] BESSALAH H. "Conception et étude d'algorithmes récursifs des transformées rapides digitales orthogonales et des méthodes de leur réalisation sur des systèmes informatiques à structure pipeline multidimensionelle" Thèse de doctorat (PHD), KIEV 1981.

[3] FLYNN M.J " Same computer organisations and effectivness " IEEE Transactions on computers C21, 3, September 1972.

[4] LIFERMANN J. "Les méthodes rapides de transformation du signal : Fourier, Walsh, Hadamard, Haar . " Edition Masson

[5] MANSOURIAN B. " Introduction à l'information médicale" édition Masson

[6] MAURICE B. "Multiplieurs numériques Revues Mini et mocros N° 136, 139 et 188.

[7] MAZARE G. " Structures multi-microprocesseurs, Problème du parallelisme Définition et évaluation d'un système particulier" Thèse d'état, Grenoble 1978.

[8] PETER W. " Analyse spectrale : un outil previlégié pour l'analyse du signal. " Revue Mesure - Régulation - Automatisme Juin - Juillet 1982.

[9] RABINER L. et GOLD B. " Theory and application of digital signal processing " Edition Prentice Hall.

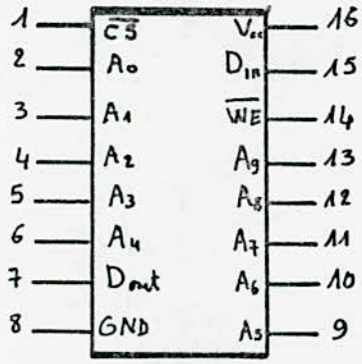
[10] The TTL Data Book for Desig Engineers. Texas instrument.

[11] Notice technique the TRW

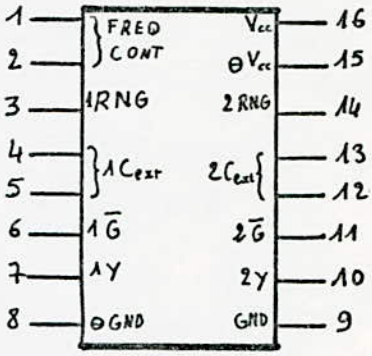
[12] Notice technique de MC 93415 de Motorola.

A N N E X E B

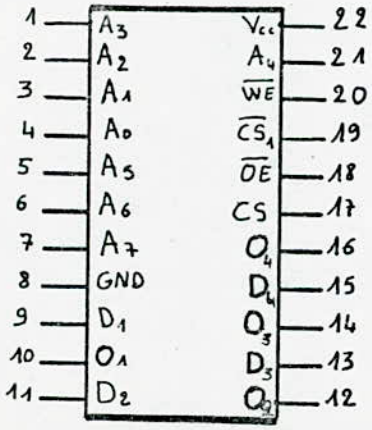
Brochage des circuits intégrés



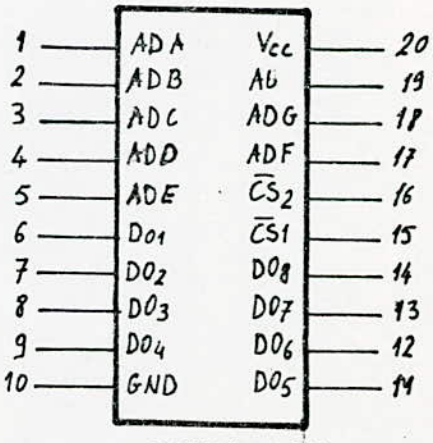
MCM93415DC



SN74S124



MCM93412DC



SN74S471