الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

11/89

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ÉCOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT **ELECTRONIQUE**

PROJET DE FIN D'ETUDES

SUJET

ETUDE ET REALISATION D'UN EMETTEUR MICRO-PROGRAMMÉ POUR MODEM V.32

Proposé par : *Mr. J. GORALSKI*

Etudié par : *S. MESSAOUD*
A. AMARARENE

Dirigé par : *Mr. J. GORALSKI*

PROMOTION : **JUIN 89**

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ÉTUDE ET RÉALISATION D'UN ÉMETTEUR MICROPROGRAMMÉ POUR MODEM V32 DU CCITT

- JUIN 89 -

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

DEDICACES

Je dedie ce travail :

A mes parents , pour les sacrifices consentis afin d'assurer mon avenir .
A mes freres et soeurs .
A mes amis .

SAID

je dedie cet ouvrage :

A mes parents , pour les privations consenties afin d'assurer mon education .
A mes freres et soeurs
A tous mes proches .
A mes amis .

AMER

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

REMERCIEMENTS

On ne saurait présenter la présente étude sans mentionner les personnes qui de prés ou de loin , materiellement ou moralement ont contribué à son aboutissement , aussi nous adressons nos remerciments :

-A tous les professeurs de l'ECOLE NATIONALE POLYTHECHNIQUE qui ont contribué à notre formation , en particulier , Monsieur J . GORALSKI qui a dirigé cette étude et dont les précieux conseils nous ont permet de découvrir cette science moderne qu'est la téléinformatique .

-A monsieur BENDDIR pour sa précieuse aide .

-A tous ceux qui nous ont aidés , de quelque maniére que ce soit à l'élaboration de ce travail , ils trouveront ici l'expression de notre profonde reconnaissance .

Nous tenons également à remercier tous les membres du jury qui ont bien voulu donner leur appréciation de la présente étude .

Sans oublier de presenter nos vifs remerciements pour HAMID et MOHAMED Zahrir pour l'aide qu'ils nous ont apportee.

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

SOMMAIRE

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

INTRODUCTION

Le développement considérable de l'informatique et des télécommunications a donnè naissance à une nouvelle discipline en plein essor : la téléinformatique. cette discipline éssentiellement axée sur un usage de plus en plus répandu d'ordinateurs interconnectés a entrainé la nécessité de développer des systèmes de transmission de données à des débits de plus en plus élevés, c'est ainsi que sur le réseau téléphonique commuté initialement conçu pour la transmission de la parole, des liaisons à 1200bit/s et 2400bit/s puis à 4800bit/s et dernièrement à 9600bit/s ont vu le jour.

Dans cette course éffrénée vers des débits binaires de plus en plus élevès et dans le souci de préserver ou même améliorer les pérformances de transmission en duplex , le codage et la modulation ainsi que les autres caractéristiques de l'information transmise se sont vus imposés par le CCITT (Comité Consultatif International pour la Télégraphie et la Téléphonie) une série d'éxigences techniques qui sont répertoriées dans des recommandations regroupées dans le fasicule "V" relatives aux modems de transmission de données sur le réseau téléphonique .

Parmi ces recommandations du CCITT , on trouve l'avis V32 qui definit une famille de modems dont les principales caractéristiques sont :

-opération en mode duplex sur ligne 2 fils du réseau téléphonique commuté ou spécialisè.

-fonctionnement selon le principe de l'annulation d'écho auto adaptative
-transmission de données synchrones , les debits utilisés étant une combinaison quelconque de débits parmi 9600 bit/s , 4800 bit/s et 2400bit/s (en cours d'étude)
-interfonctionnement automatique avec les autres modems de la famille et les modems V26 ter
-possibilité de transmission de données arythmiques aux mêmes debits nominaux (en cours d'étude)
-modulation d'amplitude en quadrature (QAM) , la fréquence de la porteuse est 1800 Hz et la vitesse de modulation (rapidité) 2400 bauds
-mode de transmission : synchrone ou arythmique (option).

le présent ouvrage traite de l'élaboration d'un modulateur V32 microprogrammé réalisant 3 fonctions éssentielles:
-le brouillage.
-le codage : differentiel et convolutif.
-la modulation qui intègre le filtrage .

L'élaboration d'un programme en accord avec les spécificités de l'avis V32 et la nécéssité de réaliser diverses opérations allant du brouillage jusqu'à la récupération des signaux sous forme d'échantillons a orienté notre choix sur le microprocesseur TMS 320-10 spécialisé dans le traitement du signal et dont la rapidité (200 ns par cycle) et la disponibilité d'instructions sous forme hard (multiplication en 1 cycle) nous permet une certaine aisance dans l'élaboration du programme .

A. PRESENTATION DU MODEM V32

A.1. Généralites

Pour jouer pleinement leur rôle de moyen de communication , les dispositifs utilisés pour la transmission de données dans les differents pays doivent être compatibles entre eux , une standardisation internationale est donc nécéssaire.
Cette tâche a été confiée à la fin des années 50 au CCITT qui s'est attelé à la standardisation d'une gamme de modems de manière à faciliter la mise en oeuvre de liaisons de données internationales.

Au fur et à mesure que la technologie progresse , de nouvelles possibilités et de nouveaux besoins apparaissent qui nécessittent la définition de nouveaux matériels de transmission , c'est pourquoi la liste des modems standardisés est loin d'être close.

Un modem se caractérise au moyen de deux paramètres :
-le (ou les) débit d'information transporté.
-le (ou les) support de transmission utilisable .

Un ensemble de débits préferentiels a été standardisé (avis V5 et V6) , les plus couramment utilisés sont :
-Pour la transmission asynchrone 300 , 600 et 1200 bit/s
-Pour la transmission synchrone 600 , 1200 , 2400 , 4800 , 9600 , 19200 , 48000 , 56000 , 64000 , 72000 , 127000 et 144000 bit/s.

Les supports de transmission considérés sont éssentiellement :
-le réseau téléphonique général commuté.
-la ligne téléphonique spécialisée définie par l'avis M 1020.

La famille de modems V32 est universelle , car elle permet la transmission de signaux synchrones ou arythmiques sur le réseau téléphonique commuté ou ligne spécialisée 2 fils , et ce dans une large gamme de débits compatibles .

La transmission bidirectionnelle simultanée sur 2 fils ou duplex constitue l'un des axes d'évolution récents les plus importants dans le domaine des modems .

L'avis V32 recommande un mode de transmission synchrone avec fonctionnement selon le principe de l'annulation d'écho auto adaptative .

A.2 Organisation d'un système télé_informatique

L'échange de méssages de données entre un terminal A et un terminal B éloignés l'un de l'autre , est réalisé á l'aide de modems (ou ETCD : Equipement de terminaison de circuit de données) , et ce par l'intermediaire du réseau de télécommunication.

Chacun des terminaux A et B (fig 1) est en général source et collecteur , mais dans la présente étude , on considerera la transmission de méssages de données dans un seul sens : du terminal source vers le terminal collecteur

La figure ci-aprés (fig 1) représente les differents éléments fonctionnels d'un système de transmission Télé-informatique :

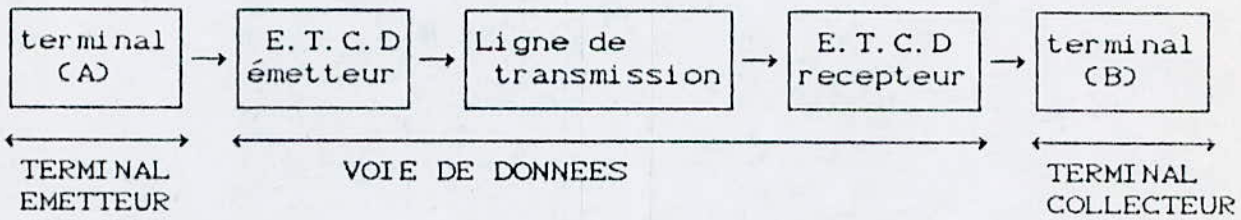


FIG 1 . Transmission de données d'un terminal source vers un terminal collecteur

Dans la présente étude , on s'interessera nottament au fonctionnement de l'ETCD (ou modem) émetteur.
En plus du terminal et du modem , éxiste la ligne de transmission qui permet une liaison physique des emplacements géographiques où se trouvent situés les terminaux A et B .

Les lignes de transmission sont caracterisées ,sous cartaines conditions , par un filtrage linéaire et l'addition d'un bruit .

A.2.1 ETCD (MODEM) émetteur. [2]

Le signal issu de l'émetteur doit être adapté aux contraintes imposées par le canal de transmission , contraintes ,au premier rang desquelles figure la nécessité de n'occuper que la bande de fréquence permise .

La conception de l'émetteur doit également prendre en compte les problèmes de fonctionnement du récepteur . Pour répondre à ces contraintes , l'émetteur réalise deux opérations représentées schématiquement sur les figures 2.

-La première opération est une transfomation du méssage numérique en une suite d'impulsions électriques.

-La seconde opération est une modulation qui transforme le signal éléctrique précédent en un signal éléctrique mieux adapté à un canal de transmission passe-bande .

- *bloc codeur :*

L'opération de codage peut se décomposer en deux parties :

a) une transformation du message numérique issu de la source en un message numérique codé ,selon une correspondance entre l'alphabet de la source et un autre alphabet , qui est l'alphabet du code.
La suite de données binaires ainsi obtenue n'a pas toujours les proprietés désirables , pour que la transmission et la réception puissent s'effectuer d'une manière correcte , on introduit donc une opération supplémentaire de précodage applée brouillage (ou brassage) dont la fonction sera détaillée ulterieurement.

b) Une transformation du méssage numérique codé en un train numérique d'impulsions , selon une correspondance entre l'alphabet du code et une famille finie de formes d'impulsion électriques , de façon à transcrire le méssage numérique dans le domaine analogique.
La densité spectrale de puissance du signal ainsi obtenue , qui représente la répartition en fréquence de sa puissance , est généralement concentrée au tour de la fréquence zéro ,ce qui éxplique son utilisation , en tant que signal émis , sur les canaux du type passe-bas . On parle alors de transmission numérique en bande de base.

- *bloc modulateur:*

Le modulateur modifie le signal numérique en faisant intervenir de façon plus ou moins éxplicite , une fréquence particulière dite fréquence porteuse .

La modulation apparait comme une fonction complémentaire du codage , et produit un signal de caractèristique spectrale appropriée à la transmission sur un canal de type passe-bande .
On parle alors de transmission numérique sur fréquence porteuse .

- *bloc filtrage:*

Un élément supplémentaire , le filtrage apparaît sur la figure 2 , soit avec une caractéristique passe-bas pour un signal numérique en bande de base , soit avec une caractéristique passe-bande pour un signal numérique modulé.

Le filtrage assure la mise en forme définitive du signal avant l'émission , et ce compte tenu du codage et de la modulation utilisée et des contraintes du canal .

Le filtrage doit répondre au double but :
-assurer le minimum de distorsion au signal qui va être traité par le récepteur .
-limiter l'occupation des fréquences à la seule bande permise.

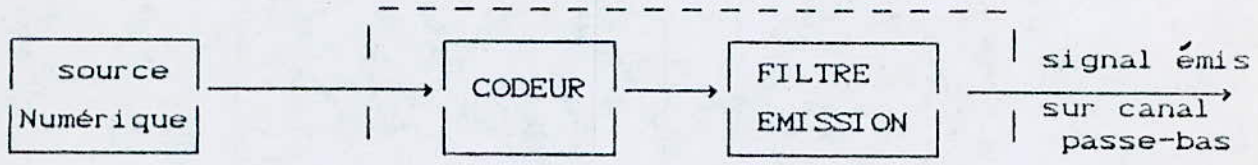


FIGURE 2-a transmission en bande de base

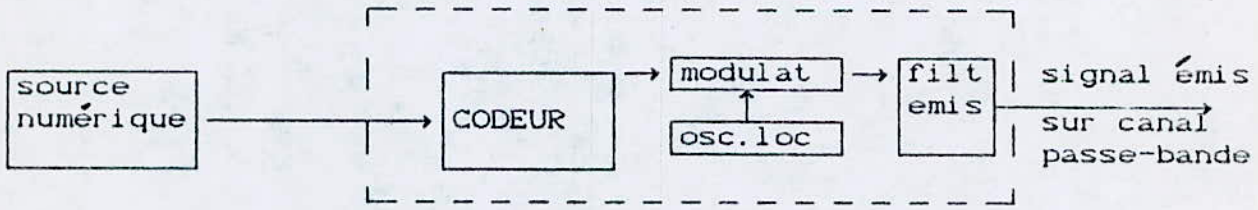


FIGURE 2-b transmission en bande transposée.

FIGURE 2 . EMETTEUR.

A.2.2 Supports de transmission de données

Pour la transmission de données , on utilise differents supports.

a) circuit téléphonique du réseau public commuté

La bande passante nominale de ce support est comprise entre 300 et 3400 Hz.

Le réseau commuté est utilisé couramment pour la transmission de données jusqu'à des débits de 4800 bit/s avec des taux d'erreurs raionnables.

Les débits superieurs à 4800 bit/s sont envisageables , au prix d'une augmentation du taux d'erreurs.

b. liaisons specialisées téléphoniques

Les liaisons spécialisées sont des lignes louées à l'administration et établies d'une façon permanente entre deux installations d'abonnés .
Elles peuvent être de type " 2 fils " ou " 4 fils " , de qualité " normale " ou de qualité " supérieure "
Les lignes de qualité supérieure sont obligatoirement [illegible]

Les lignes spécialisées ont l'avantage par rapport aux lignes du réseau commuté , de présenter des caractéristiques mieux garanties et stables au cours du temps .
Les lignes de qualité normale permettent de transmettre dans de bonnes conditions des débits de 4800 bit/s voir 9600 bit/s .

Les lignes de qualité superieure permettent en général , d'améliorer les performances à 9600 bit/s et d'atteindre des débits superieurs : 12000 , 14400 , bit/s, parfois même d'avantage.

En ce qui concerne les défauts occasionnés aux signaux par les lignes de transmission , il y a en plus de l'affaiblissement et des déformations , des signaux perturbateurs d'origines diverses dont :

-*le bruit blanc* :notion plus mathématique que physique qui reflète une réalité complexe , ce signal aléatoire présente une puissance constante dans les domaines fréquentiel et temporel

-*le bruit impulsif* :à la différence du bruit blanc , il présente une puissance moyenne faible , mais se trouve concentrée dans des intervalles de temps brefs où elle atteint des valeurs trés élevées , les sources de bruit impulsif sont nottament les organes de commutation.

En plus de ces notions on définit :[1]

-*gigue de phase* :certains organes de transposition introduisent une variation de phase périodique liée à l'alimentation par le secteur ,cette variation est appelée gigue de phase qu'on caractérise par une éxcursion crête à crête de la phase .

-*décalage de fréquence* : les opérations de multiplexage et de démultiplexage des signaux téléphoniques étant réalisées à partir d'oscillateurs locaux dont les fréquences peuvent présenter un décalage de quelque Hz , il en résulte une dissymétrie entre les opérations de transposition , d'émission et de réception et un décalage de fréquence résiduel de l'ensemble du signal pouvant atteindre quelques Hz.

-*échos* : les circuits de transmission étant composés de tronçons hétérogènes et d'équipements divers entre lesquels l'adaptation n'est pas toujours parfaite , il en résulte des réflexions partielles des signaux.

A.3 Description des signaux

A.3.1 Modes de transmission

Il existe essentiellement deux modes de transmission : synchrone et asynchrone.

a. signaux synchrones (ou isochrones)

Un signal numérique est synchrone (ou isochrone) si les intervalles de temps alloués à chaque symbole sont égaux et coincident avec les périodes successives d'un signal périodique appelé " base de temps " ou " horloge ", le signal synchrone le plus usuel est le signal binaire représenté par la fig 3.

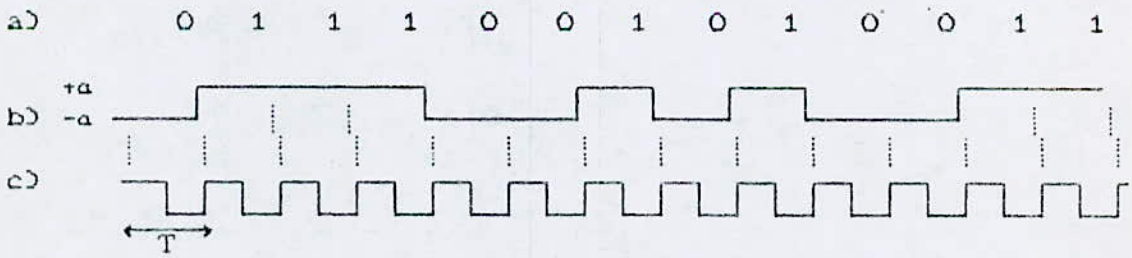


FIG 3 *Signal binaire synchrone.*

a) Séquence binaire.
b) Signal binaire.
c) Horloge associée.

Le signal d'horloge est indispensable à l'interprétation du signal de données , et ce en échantillonnant aux instants qui coincident avec les fronts descendants du signal d'horloge.

Le plus petit intervalle de temps entre 2 transitions successives du signal de données *Te* est appelé " intervalle élémentaire " .

On appelle alors " rapidité de transmission " R le nombre d'intervalles élémentaires par seconde ,elle s'éxprime en bauds

$$R = 1/Te \text{ bauds} .$$

Le " débit binaire " est le nombre d'éléments binaires ou bits transmis par seconde .
Soit un signal numérique synchrone caractérisé par une rapidité R et le nombre m d'éléments de l'alphabet de symboles.

m représente également le nombre de niveaux d'amplitude que peut prendre ce signal.
Le débit maximal transporté Dmax est $Dmax = R.Log_2 m$ (bit/s).

b) *signaux asynchrones*

On distingue deux catégories de signaux asynchrones :
-Les signaux anisochrones.
-Les signaux arythmiques.

Les signaux anisochrones sont des signaux dont le nombre d'états est fini , mais pour lesquels la durée de ces états n'est pas multiple de l'intervalle élémentaire Te .

Dans le cas général l'intervalle entre 2 transitions successives peut être quelconque pourvu qu'il soit supérieur ou égal à Te .

En pratique , les liaisons de transmission de données asynchrones sont le plus souvent utilisées pour transmettre des suites de caractères séparés par des intervalles de temps variables.

Chaque caractère se présente sous la forme d'un petit bloc d'information synchrone , ce type de transmission est appelé " transmission arythmique ".

c) *Mesure de qualité*

Dans le cas de la liaison synchrone qui intéressee l'avis V32 , on caractérise la qualité de la liaison par son taux d'érreurs .

On appelle taux d'érreurs sur les bits , le rapport τ_b entre le nombre de bits reçus faux et le nombre de bits transmis pendant un intervalle de temps donné .

Si la durée de la mesure est suffisante, τ_b est une éstimation de la probabilité d'érreur par bit d'information .

La mesure des taux d'érreurs est réalisée en émettant une séquence de bits d'éssai , et en comparant la séquence reçue avec la séquence émise .

Idéalement , la séquence d'éssai est une suite aléatoire dont le nombre de bits est grand devant l'inverse de la probabilité d'érreurs .

Etant donné que les informations transmises sont en général structurées en blocs et protegées contre les erreurs par des codes detecteurs d'erreurs agissant au niveau du bloc .

Dans ces conditions , qu'il y ait une ou plusieures erreurs à l'interieur du bloc , le résultat est le même , le bloc tout entier est rejeté et doit être répeté ,on définit donc le " taux d'erreurs sur les blocs " qui est la probabilité pour qu'un bloc comporte au moins une érreur.

Le CCITT a standardisé une séquence d'essai décrite dans l'avis V52 qui est une suite binaire pseudo-aléatoire de période 511 bits .

Compte tenu du taux d'érreurs habituellement rencontré sur les liaisons téléphoniques , la longueur des séquences d'essai utilisée sur ces supports est de l'ordre de 10^6 bits au moins soit d'environ 20 000 blocs de 511 bits .

Le taux d'erreurs sur les bits varie entre 10^{-4} et 10^{-7} selon les lignes et le débit binaire éssentiellement .

SIGNAUX DANS LE CAS DE L'AVIS V32

Dans le cas de l'avis V32 , on a une transmission en duplex qui concerne la transmission de données sur un circuit dans les 2 sens à la fois et ce sur ligne 2 fils du réseau téléphonique commuté ou spécialisé .

Cette transmission s'opère sur des données synchrones , les débits utilisés étant une combinaison quelconque parmi 9600 bit/s et 4800 bit/s.

L'option 2400 bit/s étant en cours d'étude, n'a pas encore été spécifié par le CCITT .
A ceci s'ajoute la possibilité de transmission de données arythmiques aux mêmes débits nominaux , option aussi à l'état d'étude.

A.3.2 Annulation d'écho ,[1]

Parmi les principales caractéristiques de l'avis V32 , on trouve le fonctionnement selon le principe de l'annulation d'écho auto-adaptative dont nous allons éxposer succintement le principe .

La méthode d'annulation d'écho consiste à superposer les signaux des 2 sens de transmission dans la même bande de fréquence.

La séparation des signaux est réalisée dans les équipements d'éxtrémité au moyen de coupleurs 2 fils / 4 fils apparentés à un transformateur différentiel .

A la difference des 2 autres méthodes de transmission que sont le multiplexage fréquentiel et temporel , la transmission bidirectionnelle avec annulation d'écho , ne nécéssite par rapport à la transmission unidirectinnelle ni augmentation de largeur de bande , ni accroissement de la compléxité du signal .

Les performances sont conditionnées uniquement par l'éfficacité des coupleurs d'éxtremité .

Le schéma général du modem avec annulation d'écho est donné par la figure 4

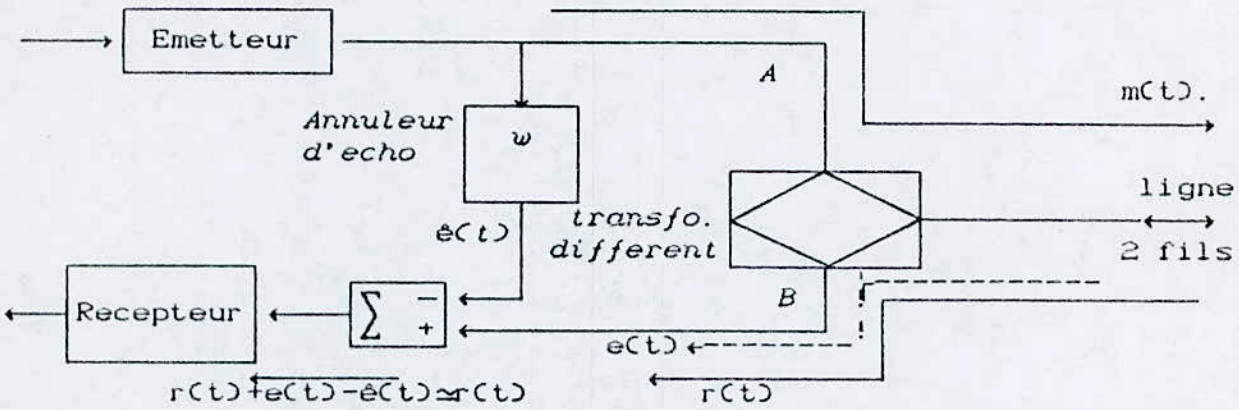


FIG 4. FILTRE ANNULEUR D'ÉCHO.

La sortie de l'émetteur et l'entrée du récepteur sont couplées à la ligne 2 fils par un transformateur différentiel.

Le signal émis est m(t) , le signal reçu est r(t) . Si le transformateur différentiel était parfaitement équilibré , seul le signal r(t) parviendrait à l'entrée du récépteur et la transmission bilatérale simultanée pourrait être éffectuée avec les mêmes performances que sur une ligne 4 fils .

En pratique l'équilibrage ne peut être qu'approximatif et le récépteur reçoit en plus du signal r(t) un signal perturbateur e(t) comportant 2 composantes :

-*Un écho local* : image déformée et atténuée du signal m(t) , due à l'imperfection d'équilibrage local du coupleur .

- *des échos lointains* :crées par des réflexions du signal émis en différents points de la ligne de transmission où éxistent des désadaptations d'impédance .

Le signal perturbateur est supérieur au signal reçu d'environ 25 á 30 dB , ce qui rend toute transmission impossible .

Le niveau de e(t) peut être ramené à une valeur acceptable en appliquant la technique de l'annulation d'écho auto-adaptative .

Le principe de l'annulation d'écho repose sur l'hypothèse d'une relation linéaire entre le signal m(t) et le signal d'écho e(t).

Si tel est le cas , le signal e(t) se déduit de m(t) par une opération de convolution

$$e(t)=m(t)* w(t).$$

w(t) peut être considéré comme la réponse impulsionnelle d'un filtre " chemin d'écho " dont l'entrée et la sortie sont réspectivement les points A et B de la figure 4 .

Si w(t) est connu , on peut construire un filtre annuleur d'écho w , de réponse impulsionnelle w(t) , et , en appliquant à l'entrée de ce filtre m(t) , on obtient à sa sortie un signal é(t) identique à e(t) , il suffit alors de soustraire é(t) de l'ensemble des signaux reçus r(t)+e(t) pour ne conserver à l'entrée du récepteur que le signal utile r(t) comme specifié sur la figure 4 .

En pratique , la réponse impulsionnelle du chemin d'écho n'est pas connue à l'avance , le filtre annuleur d'écho est donc auto adaptatif .

Cela impose que le filtre w soit numérique et qu'on lui applique des échantillons du signal m(t) .

A.4 *Modulation [1];[3]*

La plupart des supports de transmission ,nottament la ligne téléphonique , ne permettent pas la transmission directe d'un signal numérique en bande de base .

Cette limitation est due éssentiellement au phénomene de décalage en fréquence qui se traduit par une perte ou un éxcés de symboles . D'autre part , il est nécessaire de faire coincider la bande passante occupée par le signal transmis avec la bande passante du support de transmission .

Les deux problèmes peuvent être résolus simultanément en modulant une sinusoide porteuse de fréquence convenable par le signal à transmettre .

En transmission numérique , differentes techniques de modulation sont utilisées dont la plus importante est la modulation linéaire qui est basée sur une variation combinée de l'amplitude et de la phase .

Le signal modulé m(t) s'écrit alors :

$$m(t)=A(t).\cos[\omega_0 t+\phi(t)] \quad \text{.....(4-1)}$$

A.4.1 *Modulation d'amplitude à double bande (MDB).*

Dans ce type de modulation , c'est l'amplitude A(t) qui est reliée au méssage de données a(t) selon la formule :

$$A(t)=K+X(t)$$

Où le méssage X(t) se déduit par un filtre de mise en forme du méssage a(t) . la modulation est dite avec ou sans porteuse selon que K=0 ou K≠0 .

Le schéma du modulateur est donné par la figure(5) :

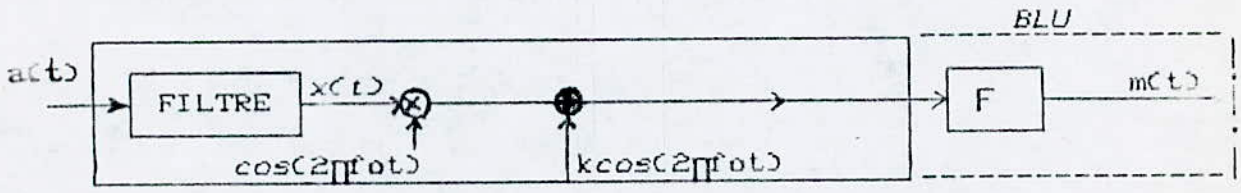


FIG.5 MODULATION MDB et BLU

La largeur de bande du signal est double de celle du méssage . Ce phénomene constitue le défaut majeur de la MDB . c'est pourquoi , dans la pratique , on utilise plutôt une autre modulation d'amplitude.

A.4.2 Modulation à bande latérale unique (BLU).

La modulation BLU est destinée à supprimer l'une des deux bandes latérales du spectre du signal modulé en MDB , pour cela on place simplement , à la sortie du modulateur MDB , un filtre idéal *F* dans la bande que l'on désire conserver , par éxemple la bande [fo-B,fo] .

Dans ce cas le spectre du signal BLU correspond à la partie gauche du spectre du signal MDB . Ceci est indiqué sur les figures 5 et 7.

La mise en oeuvre d'une modulation BLU nécéssite que le filtrage de sortie *F* présente un front trés raide au voisinage de la fréquence porteuse fo , ce qui entraîne une grande sensibilité vis-a-vis de l'instant d'échantillonnage c'est là un des inconvenients de la BLU .

En outre cette coupure brutale éxige pratiquement que le méssage en bande de base ne comporte pas une puissance significative .

A.4.3.1 Modulation d'amplitude à 2 porteuses en quadrature (MAQ)

Considerons le signal modulé :

$$m(t)=A(t)\cdot Cos[\omega_o t+\phi(t)]$$

m(t) peut s'écrire sous la forme :

$$m(t)=[A(t).Cos\phi(t)].Cos\omega t-[A(t).Sin\phi(t)].Sin\omega t$$

$$m(t)=X_p(t).Cos\omega t-X_q(t).Sin\omega t \quad \text{..... (4-2)}$$

où :
$$\begin{cases} X_p(t)=A(t).Cos\phi(t) \\ X_q(t)=A(t).Sin\phi(t) \end{cases}$$

$X_p(t)$ et $X_q(t)$ sont les coordonnées rectangulaires du

du point M extremité du vecteur $\overrightarrow{OM}$ représentant le signal m(t) [4-1] , qui s'écrit sous la forme :

$$m(t) = X_p(t).Cos\omega t + X_q(t)Cos(\omega t+\pi/2) \quad \ldots\ldots\ldots\ldots (4-3)$$

Cette éxpression montre que le signal modulé en phase et en amplitude m(t) a deux composantes en quadrature :

-La première correspond à la modulation en amplitude de la porteuse $Cos\omega t$ par $X_p(t)$.

-La deuxième correspond à la modulation en amplitude d'une porteuse en quadrature $Cos(\omega t+\pi/2)$ par le signal $X_q(t)$

Comme il éxiste un nombre limité de couples

de valeurs $\rho(KT)$ et $\phi(KT)$, les valeurs $X_p(KT)$ et $X_q(KT)$ également en nombre limité , $X_p(t)$ et $X_q(t)$ sont donc des signaux numériques .

Ce type de modulation est appelé " modulation d'amplitude de deux porteuses en quadrature " , en abrégé " MAQ " .

Les deux composantes en quadrature occupent la même bande de fréquence .

La largeur de bande du signal m(t) est donc la même que celle de l'une quelconque de ces composantes (FIG 7) . et la bande passante B du canal de transmission doit satisfaire l'inégalité $R \leq B$.

R étant la vitesse de modulation du signal numérique à transmettre .

A.4.3.2 Mode de réalisation .

La réalisation est basée sur l'éxpression (4-3) représentant la modulation de 2 porteuses en quadrature.

A chacun des points M_i du diagramme spatial , on associe un couple de coordonnées x_i , y_i .

Le schéma du modulateur est alors :

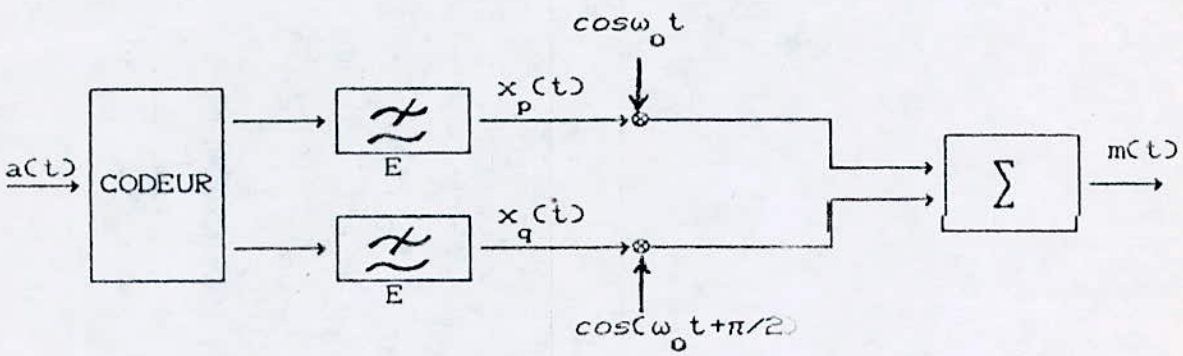


FIG 6 Modulation de 2 porteuse en quadrature

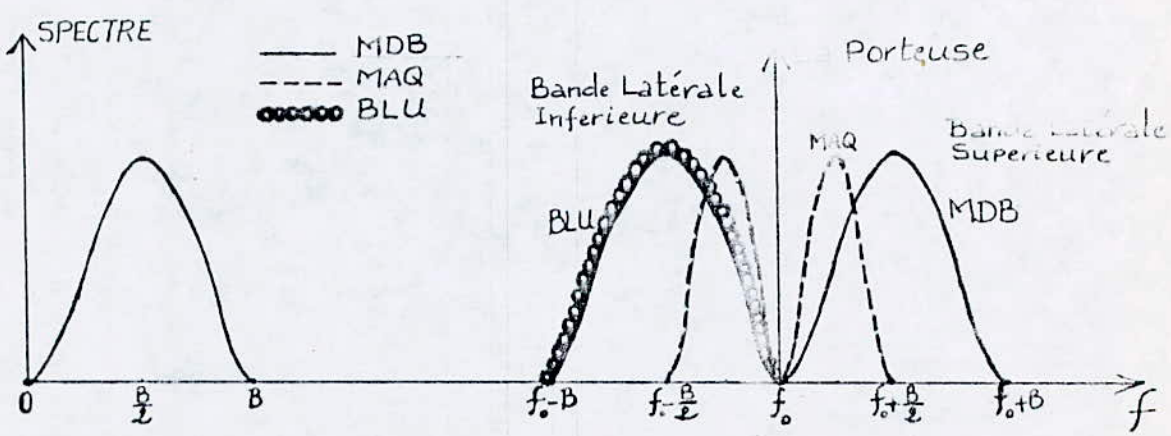
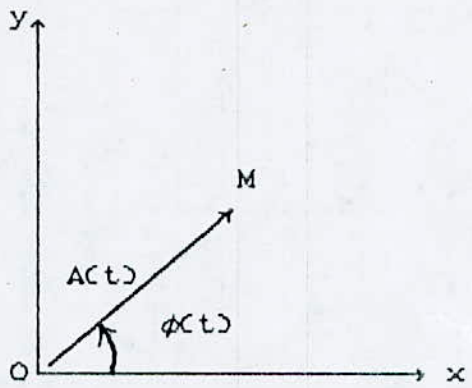


FIG 7 Spectre des signaux modulés en amplitude

4.3.3 Diagramme spatial : CONSTELLATION :

Dans cette représentation de *FRESNEL* , au signal m(t) on associe un vecteur $\overrightarrow{OM}$ dans un référentiel cartésien xOy . $\overrightarrow{OM}$ a pour module A(t) et pour argument $\phi(t)$.



Dans le cas d'une transmission numérique A(t) et $\phi(t)$ sont des signaux numériques ou des signaux numériques filtrés .

Le lieu géometrique du point M aux instants d'échantillonnage est une constéllation de points que l'on désigne par le terme " diagramme spatial ".

A.4.4 Choix d'une méthode de modulation:

Le choix d'une méhode de modulation dépend de plusieurs paramètres :
- débit numérique à transmettre .
- bande passante du support .
- perturbations introduites par le support (bruit , distorsion ...).
- coût maximal envisageable .

Le diagramme spatial permet une évaluation approchée mais rapide des diverses méthodes de modulation linéaires vis a vis du bruit et de la gigue de phase.

A.4.4.1 Comportement en présence du bruit.

La probabilité d'érreur en présence du bruit dépend à la fois du type de modulation , des caractéristiques du support de transmission , de celles des filtres contenus dans l'émetteur et le récepteur et de la nature du bruit .

On ne considérera dans ce qui suit que l'influence du type de modulation , défini par son diagramme spatial , le bruit étant supposé blanc et gaussien et les filtres conformes au 1er critère de *NYQUIST*.

Dans la représentation vectorielle du signal , celui ci est représenté à l'instant t , par un vecrteur $\overrightarrow{OM}$ de longueur égale à l'amplitude instantanée du signal .

On peut assimiler le bruit à chaque instant à un vecteur $\overrightarrow{MM'}$ de longueur égale à l'amplitude intantanée du bruit (fig 8) .

Le signal reçu a l'instant t est le vecteur $\overrightarrow{OM'}$

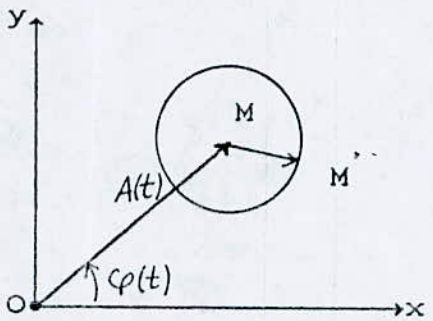


FIG 8 Bruit additif

Toutes les orientations du vecteur MM' sont équiprobables , le point M' à la probabilité p de se trouver à l'interieur d'un cercle de rayon MM'.

Considérons l'éxemple d'une modulation MAQ à 8 états , il éxiste plusieures façons possibles de disposer les 8 points dans le plan , l'une d'elles est donnée par la figure 9.

Pour avoir une puissance moyenne egale á A^2 , on prendra $\ell = A\sqrt{2/11}$, dans ces conditions , la distance minimale entre 2 points est égale à $2A\sqrt{2/11}$.

Une améioration supplémentaire peut être amenée en abondonnant la contrainte d'un rapport simple entre les coordonnées des differents points

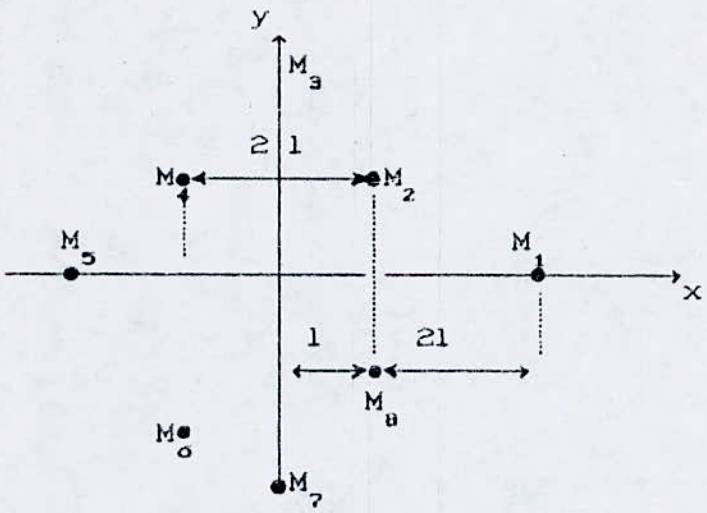


FIG 9 MODULATION MAQ à 8 etats .

Une modulation MAQ à 16 états permet de transmettre un débit binaire superieur sans augmenter la bande passante du support .

A.4.4.2 Comportement en présence de la gigue de phase.

L'éffet de la gigue de phase sur le signal reçu est schématisé par le diagramme de la figure 10.
Le vecteur $\overrightarrow{OM}$ représentatif du signal subit une rotation périodique autour de l'origine d'amplitude crête a crête ϑ .
En pratique la fréquence de la gigue est le plus souvent égale au superieure a 50 Hz .
La combinaison du bruit et de la gigue de phase a pour éffet d'entourer les points du diagramme spatial de zones de dispersion

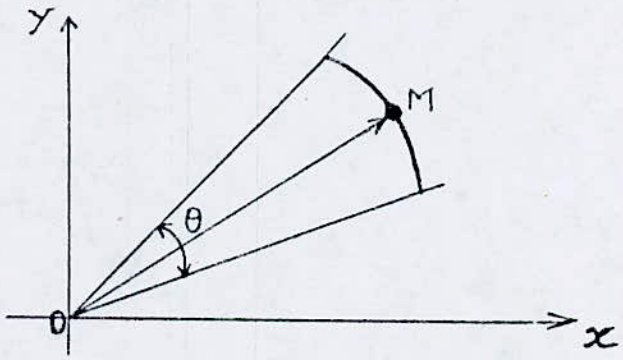


FIG 10 Effet de la gigue de phase

La recherche du meilleur diagramme spatial passe par la prise en compte de ces 2 phénomenes : à savoir bruit et gigue de phase.

A.4.4.5 Diagramme spatial pour l'avis V32.

Pour l'avis V32 le type de modulation utilisé est la modulation MAQ .La fréquence de la porteuse est 1800 Hz et la rapidité 2400 bauds .

A 4800 bits/s le diagramme spatial est identique à celui d'un signal modulé en phase à 4 états . La loi du codage differentiel est donnée par le tableau de la figure 11.

Débit	Saut de phase
0 0	$+90^{o}$
0 1	$+0^{o}$
1 1	$+270^{o}$
1 0	$+180^{o}$

FIG 11 LOI DE CODAGE DIFFERENTIELLE POUR 4800 BIT/S.

A 9600 bit/s , deux méthodes de codage sont prévues . la première ne comporte pas de redondance et conduit à un diagramme spatial à 16 états donné par la figure 1/V32 (page 29)

La deuxième utilise le codage convolutionnel en treillis . le diagramme spatial correspandant est donné par la figure 3/V32.

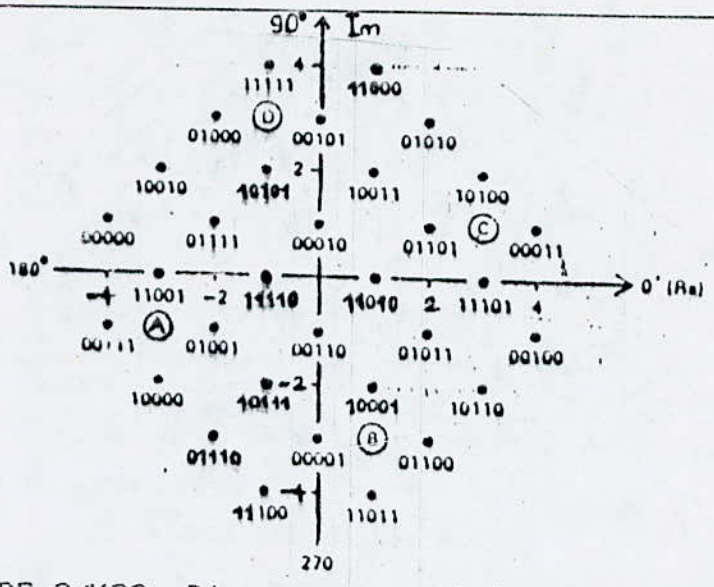


FIGURE 3/V32 Diagramme spatial pour le codage en treillis à 9600 bit/s et pour les états ABCD utilises à 4800 bit/s

A.5 Brouillage [2]

D'une part , les circuits de récupération de rythme employés dans une transmission numérique ne peuvent fonctionner correctement que si le méssage modulant comporte des transitions assez nombreuses .
Certaines sources risquent de fournir de longues séquences sans transition , et il est alors nécéssaire de leur faire subir un traitement avant de les transmettre .

D'autre part , une séquence périodique courte conduit à un signal modulé comportant des raies à fort niveau qui jouent le rôle de perturbateur vis-à-vis d'autres transmissions.
Afin d'éviter de telles perturbations , il nécéssaire d'augmenter la période des séquences courtes qui risquent de se présenter .

Le brouilleur est un dispositif logique , utilisé à l'émission , qui modifie les symboles fournis par une source de telle sorte que les séquences périodiques courtes soient remplacées par des séquences de grande période comportant de nombreuses transitions.
Le débrouilleur est utilisé à la réception et a pour rôle de reconstituer la séquence d'origine à partir du signal démodulé .

A.5.1 Brouilleur - débrouilleur synchrone

Le brouilleur réalise l'addition bit a bit modulo 2 , du train numérique représentant le méssage à transmettre a_n, et d'une séquence pseudo-aléatoire (p_n).

Les éléments binaires brouillés sont $b_n = a_n \oplus p_n$

Le débrouilleur est identique au brouilleur , il réalise l'addition bit à bit , modulo 2 , du train numérique démodulé b_n, et d'une séquence pseudo-aléatoire identique à celle utilisée a l'émission .

Les générateurs pseudo-aléatoires du brouilleur et du débrouilleur doivent être identiques et délivrer les mêmes séquences avec la même phase . Il est donc nécessaire de prévoire un dispositif de synchronisation pour les faire fonctionner en phase .

A.5.2 Brouilleur-debrouilleur de base , autosynchronisant.

Le schéma de base d'un brouilleur autosynchronisant est dérivé de celui d'un générateur de séquence pseudo-aléatoire. Le brouilleur est constitué d'un registre à décalage de m bascules d'additionneurs modulo -2 . Les connexions sont définies par un polynôme primitif , h(x) , de degré m :polynôme générateur .

$$h(x) = \sum_{j=0}^{m} h_j \cdot x^j \qquad (h_j = 0 \text{ ou } 1)$$

L'inconvénient de l'opèration brouillage-débrouillage autosynchronisant est une multiplication du taux d'erreurs sur lers éléments binaires .

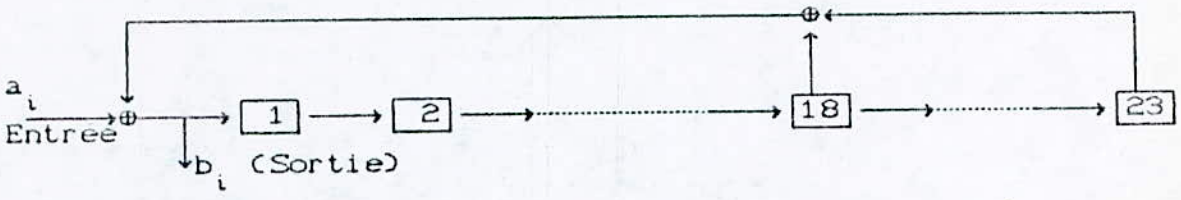
On a interêt à utiliser des polynômes ayants peu de coefficients non nuls, cette multiplication du taux d'érreur devient minimum pour un polynôme primitive à trois coefficients (il faut au moins un coefficient non nul en plus de ho et hm)

Ce type de brouilleur-débroullieur est autosynchronisant la probabilité de transition dans une séquance brouillée est trés voisine de 1/2

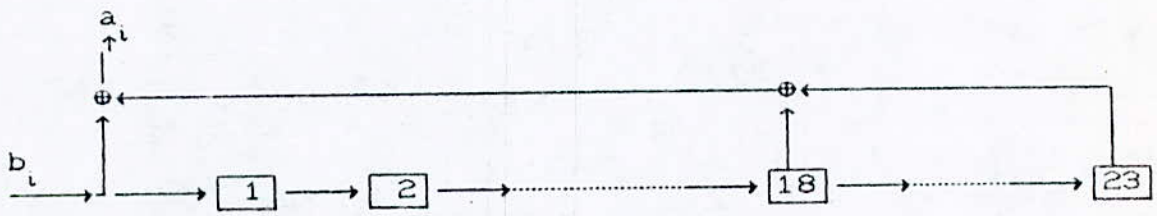
A.5.3 Brouilleurs du modem V32

Dans le cas du modem V32 , le brouilleur possede 23 étages , il est different selon le sens de transmission .
Les polynômes générateurs sont :
- Pour l'émetteur appelant $1 + x^{18} + x^{23}$ (brouilleur 1)
- Pour l'émetteur appelé $1 + x^{5} + x^{23}$ (brouilleur 2)

Pour élaborer une émission de données , en active le brouilleur en mode appelant (brouilleur 1) .
Pour le mode appelant on a la structure brouilleur-débrouilleur suivante



Brouilleur



Débrouilleur

FIG 12 Brouilleur et Débrouilleur autosynchronisant du modem V.32

L'utilisation de deux brouilleurs différents est justifiée par le fait qu'étant donné le principe de l'annulation d'écho auto-adaprtative repose sur l'indépendance statistique des signaux transmis dans les deux sens , cependant dans certaines configurations , les deux peuvent être correlés .
Pour garantir tous les cas une décorrélation suffisantes on utilise des brouilleurs différents dans les deux sens de transmission .
Le modem appelant utilise le brouilleur 1 dans le sens émission et le brouilleur 2 dans le sens récepteur .

A.6 Codage : [4],[3],[1]

Le méssage numérique à coder est une suite de variables aléatoires $\langle i_k \rangle$. Ces variables prennent leurs valeurs dans l'alphabet à q eléments .

Les symboles sont délivrés par la source à un rythme de période T_s , ce qui correspond à un débit symbole $1/T_s$ et un débit binaire de $(1/T_s).\log_2(q)$.

On peut considérer que le codage se compose :

a) d'une règle de codage proprement dite qui permet d'associer à la suite $\langle i_k \rangle$ une suite de variables aléatoires $\langle c_k \rangle$, prenant leursvaleurs dans un alphabet de sortie a M valeurs $c = \langle c_0, c_1, \ldots\ldots, c_{M-1} \rangle$.

Cette règle de codage est décrite soit à l'aide d'un dictionnaire de mots de code , soit à l'aide d'un graphe ou diagramme d'états décrivant les transitions entre les états c_j en fonction des symboles associés à $\langle i_k \rangle$.

b) d'une mise en forme éléctrique ,faite par une correspondance bi-univoque entre les c_j et les impulsions $g_j(t)$.

La valeur C_j de c_k est transmise à l'instant KT par l'impulsion $g_j(t-KT)$ d'énergie finie prenant des valeurs dans $\mathbb{R}$ (ou $\mathbb{C}$) .

L'impulsion associée à la variable aléatoire c_k est notée $c_k(t)$.

Le codage fait donc correspondre au méssage le signal $c(t)$:

$$c(t) = \sum_{k=-\infty}^{+\infty} C_k(t)$$

La grandeur $1/T$ qui mesure le rythme de transmission des impulsions codées est habituellement dénomée vitesse de modulation .

La vitesse de modulation est augmentée (resp. diminuée) par rapport au débit symbole si $T < T_s$ (resp. $T > T_s$) et elle est conservée si $T = T_s$.

Le signal codé est dit redondant si le débit binaire maximal que l'on peut transmettre avec un ensemble de M signaux de durée T , $(1/T).log_2(M)$ est supérieur au débit binaire effectivement transmis $(1/T_s).log_2(q)$ soit :

$$(1/T_s).log_2(q) < (1/T).log_2(M)$$

La redondance peut donc être obtenue avec :

- $M = q$ et $T_s > T$ ou $1/T > 1/T_s$ ce qui correspond à une augmentation de la vitesse numérique .
- $M > q$ et $T_s = T$,ce qui correspond à un méssage binaire codé à l'aide d'un signal à plus de deux nivaux .

A noter que l'on fait souvent subir au signal avant le codage une transformation linéaire dite précodage , dans l'intéret apparait dans la procédure de décodage d'un code lineaire .Un cas particulièrement important est celui de codage par transition, encore désigné par differentiel.

Dans l'élaboration du modem V32 ,interviennent éssentiellement 3 types de codage :

codage differentiel , non redondant et redondant .

A.6.1 *Codage differentiel :*

Le codage differentiel est un traitement éffectue sur le signal numérique avant la modulation de telle sorte que l'information ne soit pas transmise par la phase , mais par les sauts de phase de la porteuse .

Cette méthode permet de résoudre le problème de l'ambiguité de phase (coté reception)au prix d'un codage supplémentaire et d'une légère augmentation du taux d'erreur .

Cette ambiguité de phase n'est pas due à un défaut de conception des circuits de récuperation desporteuses ,mais est intrinsèquement liée à la nature même du signal modulé .

En effet , pour la plupart des modulations linéaires , la constellation des points représentatifs des signaux émispossibles dans l'éspace de signaux est identique à elle-même par rotation d'un angle $2\pi/N$ autour de l'origine (dans notre cas N=4) .

Il est impossible à un observateur ne connaissant pas le méssage transmisdedéterminer éxactement la phase de la porteuse utilisée.

Il ne peut la determiner qu'à des multiples prés de $2\pi/N$.

a) Dans le modem V32 : est prévu pour le débit 4800 bit/s , un diagramme spatial identique à celui d'un signal modulé en phase à 4 états .
La loi du codage est differentielle, elle est donnée par la fig 13 .

Entrées		Sor Prec		Saut de phase	Sorties		Codage Q-aire pour 4800 bit/s
Q_{1n}	Q_{2n}	Y_{1n-1}	Y_{2n-1}		Y_{1n}	Y_{2n}	
0	0	0	0	+90°	0	1	B
0	0	0	1		1	1	C
0	0	1	0		0	0	A
0	0	1	1		1	0	D
0	1	0	0	0°	0	0	A
0	1	0	1		0	1	B
0	1	1	0		1	0	D
0	1	1	1		1	1	C
1	0	0	0	+180°	1	1	C
1	0	0	1		1	0	D
1	0	1	0		0	1	B
1	0	1	1		0	0	A
1	1	0	0	+270°	1	0	D
1	1	0	1		0	0	A
1	1	1	0		1	1	C
1	1	1	1		0	1	B

Fig-13

TABLE 1/V32 Codage differentiel pour 4800 bit/s et 9600 bit/s non redondant .

Pour le modem V32 , les données brouillées sont subdivisées en groupes de 2 bits consecutifs .
Ces bits notés Q_{1n} et Q_{2n} où l'indice n désigne le numéro de la séquence , sont codés differentiellement en Y_{1n} et Y_{2n} , et ce en accord avec la table 1/V32 .
La figure 1/V32 montre les points utilisés pour le débit4800 bit/s entourés d'un cercle.

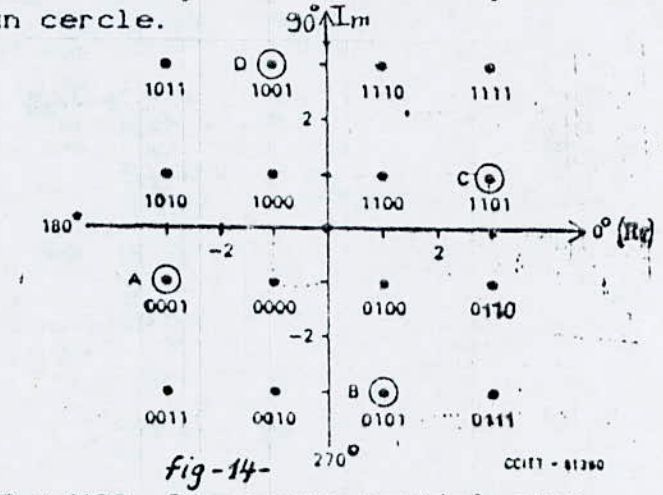


fig-14-

FIGURE 1/V32 Diagramme spatial pour 9600 bit/s non redondant et pour les états ABCD utilisés à 4800 bit/s

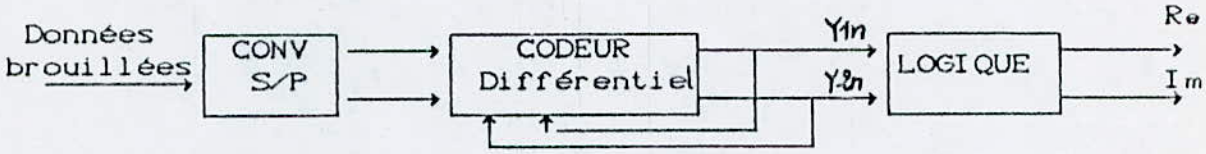


FIG 15 bloc codage pour 4800 bit/s

b) Pour le débit 9600 bit/s avec un codage non-redondant .
Dans cette configuration , le train binaire brouillé est divisé en groupe de 4 bits (quadribits) .
Les deux premiers bits Q_{1n} et Q_{2n} sont codés différentiellement en Y_{1n} et Y_{2n} , et ce en accord avec la table 1/V32 .

Les bits Y_{1n}, Y_{2n} , Q_{3n} , Q_{4n} sont alors transformés conformément à la figure 1/V32 et la table 3/V32

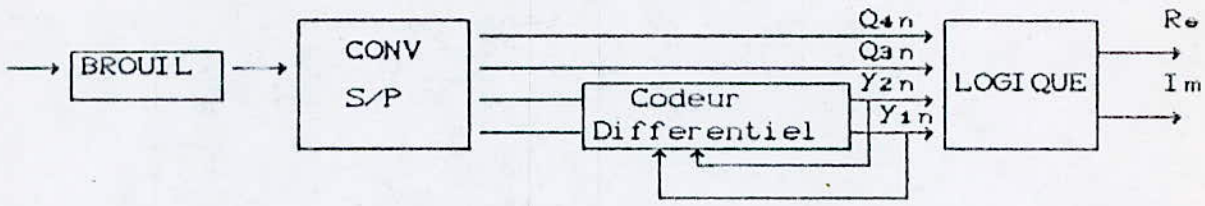


FIG 16 - BLOC CODAGE POUR 9600 BIT/S .
AVEC UN CODE NON REDONDANT

C Codage convolutionnel [4]

L'organisation général d'un codeur convolutionnel est la suivante .

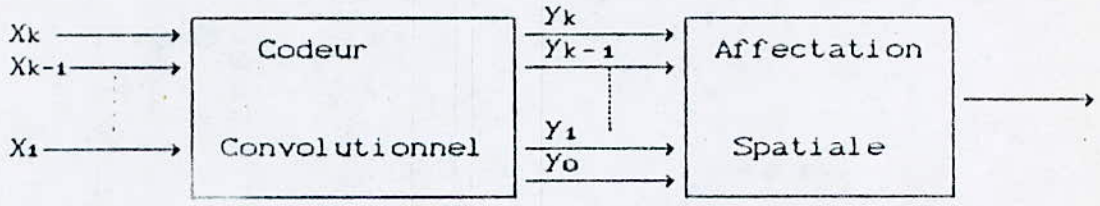


FIG 17 - CODEUR CONVOLUTIONNEL

Le groupe de K bits non codés est appliqué en parallèle à un codeur convolutionnel qui délivre un groupe de K+1 bits .
Une loi d'application spatiale fait correspandre aux K+1 bits un
point du diagramme spatial parmi 2^{k+1} dans le plan complexe .

Le fonctionnement du codeur convolutionnel peut être décrit par un diagramme en treillis .
Le groupe de bits génerés à l'instant nT par le codeur dépends des K bits présentés à l'entrée et de l'état du codeur à cet instant .

Le décodage est effectué dans le récepteur au moyen de l'algorithme de *viterbi* , en utilisant le même diagramme en treillis .

Le codage convolutionnel tend à être de plus en plus utilisé dans la transmission de hauts débits d'information sur canal téléphonique .

Un codage en treillis approprié permet un gain en rapport signal sur bruit de l'ordre de 8 à 6 dB .

Dans le codage en treillis utilisé pour le débit 9600 bit/s du modem V32 , le train binaire brouillé à transmettre est divisé en groupe de 4 bits consécutifs comme indiqué sur la figure 2/V32 , les deux premiers bits Q_{1n} et Q_{2n} sont premierement codés differentiellement en Y_{1n} et Y_{2n} en accord avec la table 2/V32 .

Les deux bits différentiellement codés Y_{1n} et Y_{2n} sont ensuite utilisés comme entrées d'un codeur convolutionnel systématique qui génère un bit redondant Y_{0n} .

Ce bit redondant et les 4 bits informationnels Y_{1n} , Y_{2n} , Q_{3n} et Q_{4n} sont alors transformés en accord avec le diagramme spatial de la figure 3/V32 et la table 3/V32

Entrées		Sorties précédentes		Sorties	
Q_{1n}	Q_{2n}	Y_{1n-1}	Y_{2n-1}	Y_{1n}	Y_{2n}
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	0	0
0	1	1	0	1	1
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	1	0	0
1	1	0	0	1	1
1	1	0	1	1	0
1	1	1	0	0	0
1	1	1	1	0	1

fig-18-

TABLE 2/V32

Codage differentiel utilisé dans l'alternative du codage en treillis à 9600 bit/s

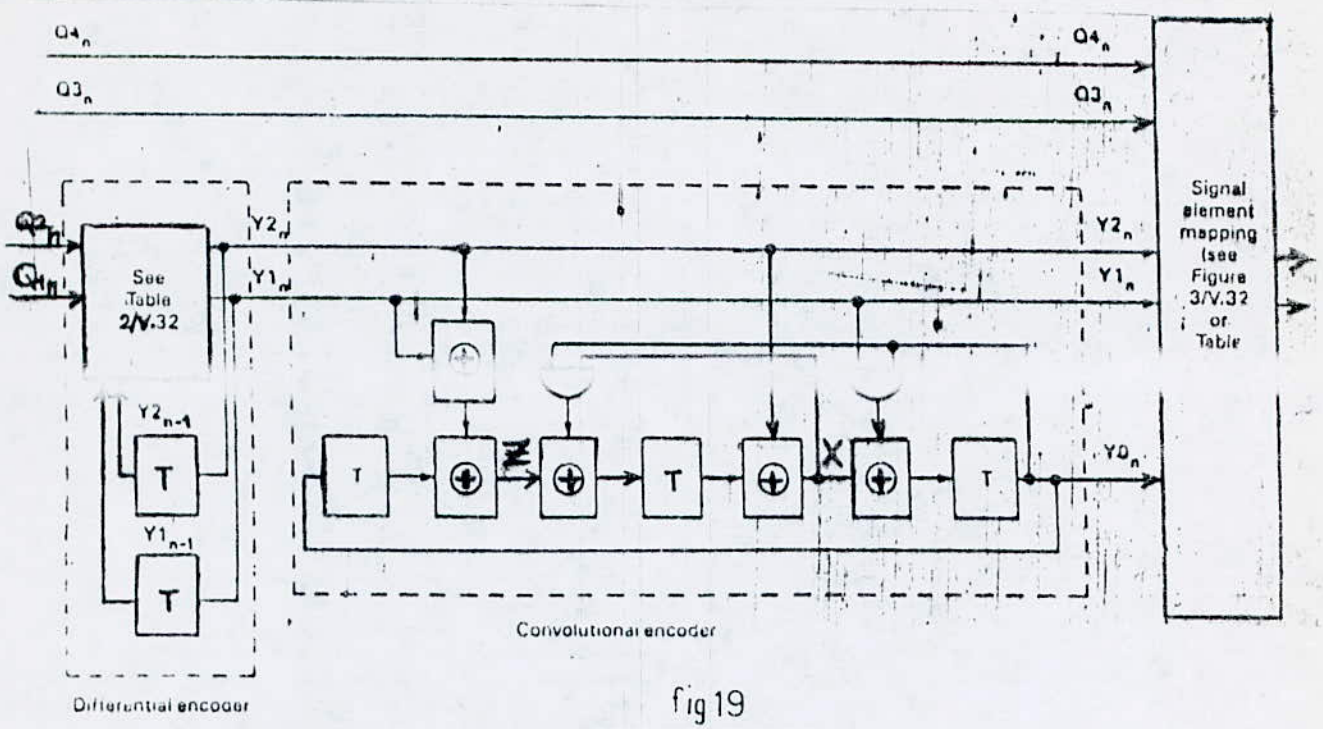


fig 19

FIGURE 2/V32

Codage en treillis à 9600 bit/s .

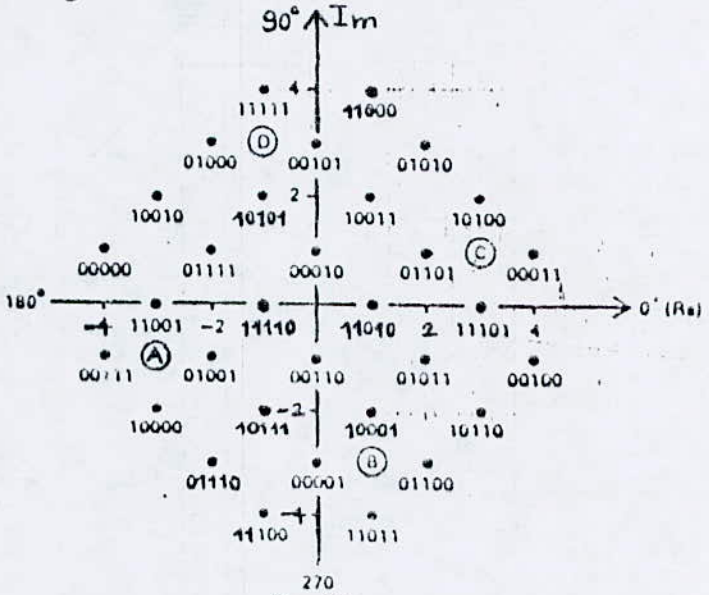


fig 20

FIGURE 3/V32

Diagramme spatial pour le codage en treillis à 9600 bit/s et pour les états ABCD utilisés à 4800 bit/s

Entrées codées					Codage non-redondant		Codage en treillis	
(Y0)	Y1	Y2	Q3	Q4	Re	Im	Re	Im
0	0	0	0	0	-1	-1	-4	1
	0	0	0	1	-3	-1	0	-3
	0	0	1	0	-1	-3	0	1
	0	1	0	0	1	-1	4	-1
	0	1	0	1	1	-3	0	3
	0	1	1	0	3	-1	0	-1
	0	1	1	1	3	-3	-4	-1
	1	0	0	0	-1	1	-2	3
	1	0	0	1	-1	3	-2	-1
	1	0	1	0	-3	1	2	3
	1	0	1	1	-3	3	2	-1
	1	1	0	0	1	1	2	-3
	1	1	0	1	3	1	2	1
	1	1	1	0	1	3	-2	-3
	1	1	1	1	3	3	-2	1
	0	0	0	0			-3	-2
	0	0	0	1			1	-2
	0	0	1	0			-3	2
	0	0	1	1			1	2
	0	1	0	0			3	2
	0	1	0	1			-1	2
	0	1	1	0			3	-2
	0	1	1	1			-1	-2
	1	0	0	0			1	4
	1	0	0	1			-3	0
	1	0	1	0			1	0
	1	0	1	1			1	-4
	1	1	0	0			-1	-4
	1	1	0	1			3	0
	1	1	1	0			-1	0
	1	1	1	1			-1	4

fig 21

TABLE 3/V32

les deux alternatives du codage pour 9600 bit/s

A.7 -*filtrage d'émission* : [5] , [6] , [1]

Un signal de données ayant un spectre infini doit être limité en largeur de bande de fréquence à l'aide d'un filtre approprié avant d'empreinter un support (canal) dont la bande passante est limitée .

Or lorsqu'on réduit le [illegible] entraine un elargissement de l'impulsion .
Des impulsions appartenant à des signaux differents peuvent interférer .

Ce phénomène est connu sous le nom d'interférence intersymboles ou interférences longitudinales .

Ainsi le filtrage est une fonction importante , qui doit assurer à la fois :

a) limiter l'occupation des fréquences *i.e* largeur du spectre du signal de données à transmettre , à la bande passante - du canal de transmissions utilisé.

b) Assurer le minimum de distorsions du signal qui va être traité par le récepteur . Ceci consiste en l'annulation de l'interférence intersymbole :

b-1 : Aux instants de prise de décision afin d'extraire du signal reçu (à la réception) par échantillonnage des valeurs successives des symboles transmis.
b-2 : Aux instants correspondants aux transitions du signal reçu afin d'avoir un rythme d'échantillonnage synchronisé en fréquence et en phase avec le signal reçu .

Pour que le filtre d'émission assure les deux conditions précédantes (*b-1 et b-2*) , il doit satisfaire à la fois le premier et le second critères de *NYQUIST* (REF [1]) .

Ainsi la fonction de transfert du filtre (voir la démonstration dans la référence1) doit satisfaire à la fois :

$$\left.\begin{array}{l} H(f) + H(1/T - f) = T \\ H(f) - H(1/T - f) = T\cos\Pi fT \end{array}\right\} \longrightarrow H(f) = T/2\,(1 + \cos\Pi fT)$$

La courbe représentative de H(f) (cosinusoide surelevée).
‖H(f)‖ est le module de H(f).

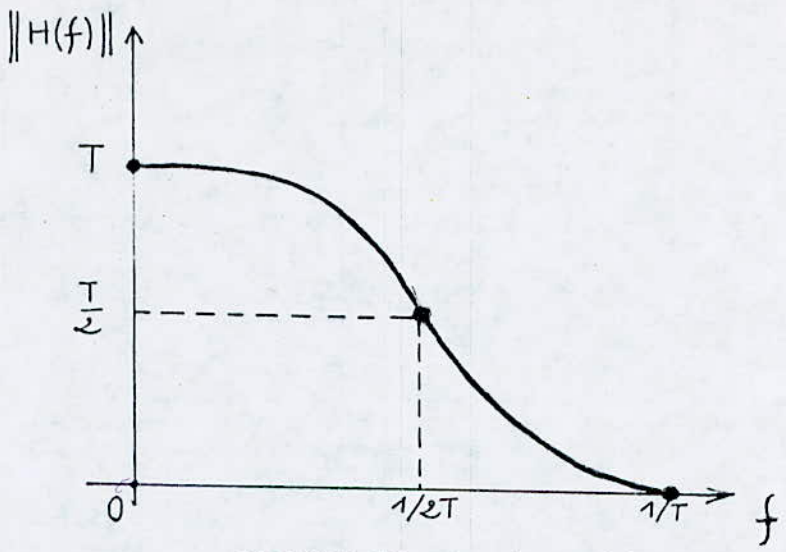


FIGURE 22 Cosinusoide

La réponse impulsionnelle h(t) du filtre , passe par tous les points entourés d'un cercle .

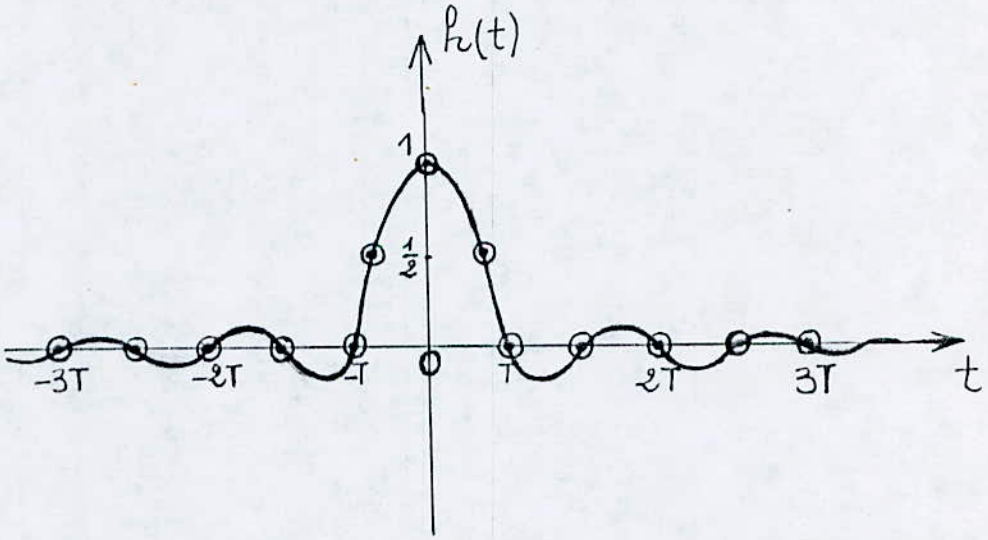


FIG 23 Réponse impulsionnelle du filtre de Nyquist (1 er et 2 em critère)

La courbe en cosinusoide surélevé est celle du filtre rectangulaire de fréquence de coupure $f_c = 1 / 2T$ constituent les deux extrêmes d'une famille de courbes de gain trés utilisée en pratique .

en posant $\alpha = \dfrac{f_c - 1/2T}{1/2T}$.

α est appelé " coéfficient d'arrondi "

Ces filtres sont définis de la façon suivante :

$$\|H(f)\| = \begin{cases} T & \text{pour } 0 \le f < (1-\alpha)/2T \\ T/2\,[\,1 + \cos\pi T/\alpha\,(f - (1-\alpha)/2T\,]\,] & \text{pour } (1-\alpha)/2T \le f < (1+ \\ 0 & \text{pour } f \ge (1+\alpha)/2T \end{cases}$$

La réponse impulsionnelle de ces filtres est de la forme :

$$h(t) = \frac{\sin(\pi t/T)}{\pi t/T} \cdot \frac{\cos(\alpha\pi t/T)}{1-(2\alpha t/T)^2}$$

La figure 24 donne l'allure de la courbe gain et de la réponse impulsionnelle pour les valeurs de α : 0 , 0.5 , 1.

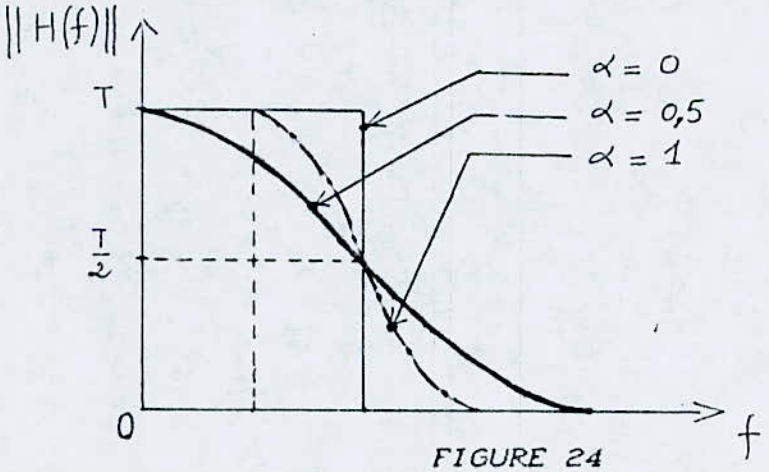


FIGURE 24

Soient x(t) et y(t) réspectivement l'entrée et la sortie du flitre .

$$y(t) = x(t) * h(t) = \int_{-\infty}^{+\infty} x(\tau).h(t-\tau).d\tau$$

Le filtre considéré précedemment est du type passe-bas ainsi pour l'adapter a la bande passante du canal utilisé ie la ligne téléphonique il faut translater la caracteristique du filtre H(f) de fo (fréquence porteuse) .

Ainsi la réponse impulsionnelle du filtre s'écrira sous la forme :

$$h(t) = \frac{\sin(\Pi t/T)}{\Pi t/T} \cdot \frac{\cos(\alpha \Pi t/T)}{1-(2\alpha t/T)^2} \cdot \cos(\omega_0 t)$$

Comme le calcul analytique conduit à des opérations difficiles à élaborer , on a opté pour le calcul numérique , vu la facilité de son élaboration par l'outil informatique .

L'operation de convolution s'écrira alors :

$$y(n) = \sum_{m=-\infty}^{+\infty} x(m) \cdot h(n-m) = \sum_{m=-\infty}^{+\infty} h(m) \cdot x(n-m)$$

La causalité du filtre réel impose $h(n-m) = 0$ pour $n-m < 0$.

D'ou :

$$y(n) = \sum_{m=-\infty}^{n} x(m) \cdot h(n-m)$$

Comme la notion de causalité peut être appliquée sur les signaux, alors la réponse d'un système lineaire invariant causal à une excitation causale est de la forme :

$$y(n) = \sum_{m=0}^{n} x(m) \cdot h(n-m) = \sum_{m=0}^{n} h(m) \cdot x(n-m)$$

Au cours de notre calcul , on a opté pour la forme

$$y(n) = \sum_{m=0}^{n} x(m) \cdot h(n-m)$$

$n = KT_e$ ou , $K = 1 , 2 , 3 \ldots\ldots , N$.

Le signal à transmettre avant d'être filtré peut s'écrire sous la forme :

$$x(t) = \sum_n x_i (t-nT)$$

A la sortie du filtre $y(t) = x(t) * h(t)$

$$y(t) = \left[\sum_n x_i(t-nT)\right] * h(t)$$

$$= \sum_n x_i(t-nT) * h(t) = \sum_n y_i(t-nT)$$

où $\quad y_i(t-nT) = x_i(t-nT) * h(t)$

Ceci montre l'independance de $y(t)$ de l'origine des temps choisie .

Comme les valeur des x_i sont limitées en nombre .

1) 4 valeurs de x_i pour le débit 4800 bit/s .
2) 16 valeurs de x_i pour le débit 9600 bit/s pour le codage non redondant .
3) 32 valeurs de x_i pour le débit 9600 bit/s pour le codage redondant .

Alors il est préférable de calculer les échantillons de y_i correspondants aux valeurs de x_i .

Ce calcul sera effectué une seule fois et les resultats (valeurs de y_i) seront sauvegardés définitivement dans un éspace mémoire qui leur est reservé dans la ROM .

Une fois les échantillons traités (modulés et filtrés) sont stockés , il suffit alors uniquement de connaître la séquence codée de données qui décidera de la position mémoire à activer ainsi que de l'échantillon traité à présenter à la sortie l'émetteur simulé .

T_e : est la période d'échantillonnage $T_e = \frac{1}{f_e}$

$m = i.T_p \qquad i = 0 , 1 , 2 .....$

Tp est un paramètre servant uniquement au calcul du produit de convolution , dont la valeur doit être trés inferieur devant celle de Te (les resultats sont d'autant précis que Tp << Te).

Le nombre d'échantillons N à prendre sera déterminé à partir du spectre de puissance défini par l'avis V32 (fig 25) ..

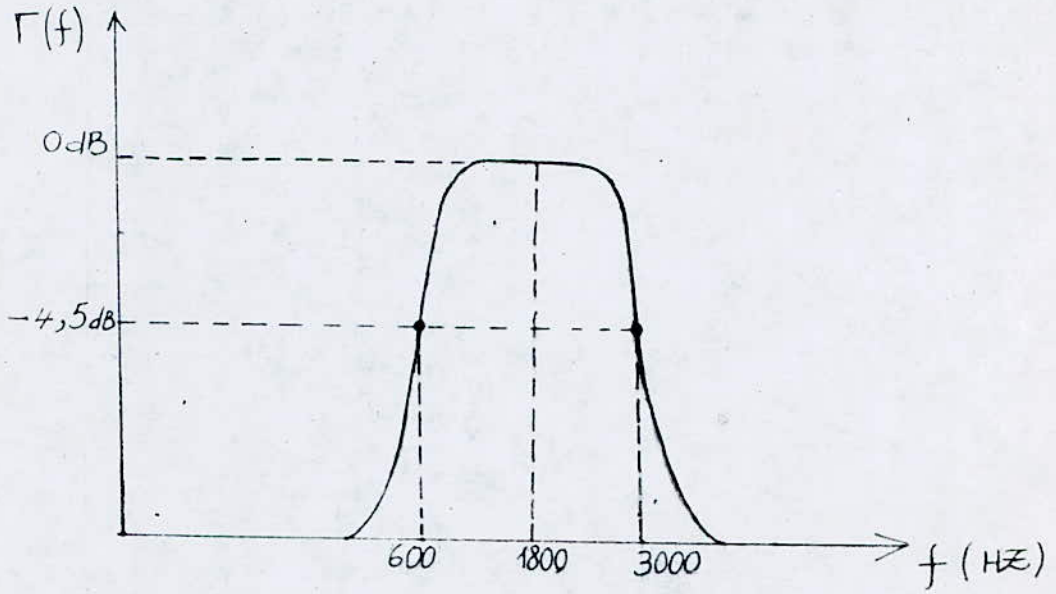


FIGURE 25

Soit Te/Tp = L

Comme $n \leq n \Rightarrow i\ Tp \leq KTe \Rightarrow i \leq LK$.

$$y(KTe) = \sum_{i=0}^{L.K} x(iTp) \cdot h(KTe - iTp)$$

A noter que le traitement ci dessus (i.e calcul des y(KTe)) est indépendant de l'origine des temps ,mais dépend plus précisement de la période d'échantillonnage Te et de la valeur du rapidité Tp choisie .

A.8 *Calcul du spectre du signal émis [6] , [1]*

Aprés codage et modulation , le signal à émettre à pour éxpression .

$$x(t) = \sum [a_k \cos(\omega_0 t) + b_k \sin(\omega_0 t)] \cdot \text{rect}[(t-KT)/T]$$

avec $T = 1/\nu_m = 1/2400$ s ; $f_0 = \omega_0/2\pi = 1800$ Hz

rect(t) désignant la fonction rectangulaire .

a_k et b_k appartiennent à l'ensemble des m valeurs specifiées des signaux numériques à plusieurs niveaux . *(avis V.32)*

On pose $x_k(t) = a_k \cos(\omega_0 t) + b_k \sin(\omega_0 t)$

La fonction d'autocorrélation d'un signal aléatoire de la forme

$\sum_k a_k . \text{rect}[(t-KT)/T]$ a pour éxpression :

$$R_{a_k}(\tau) = \sigma_x^2 \, \text{tri}(\tau/T) + \mu_x^2$$

μ_x est l'éspérance mathématique des a_k

σ_x^2 est la variance des a_k

tri(t) représente la fonction triangle .

La fonction d'autocorrélation est la même pour le signal

$\sum_k b_k . \text{rect}[(t-KT)/T]$

Dans le cas de l'avis V32 et pour les 3 differents codages anterieurement énumérés on a :

$$\mu_x = \sum_{n=1}^{m} a_k . \text{prob}(a_k)$$

D'où en supposant les a_k et b_k équiprobables dans chacun des codages c'est à dire :

$\forall k \quad p(a_k) = p(b_k) = 1/N$

N = 32 pour le codage redondant à 9600 bit/s
N = 16 pour le codage non redondant à 9600 bit/s
N = 4 pour le codage differentiel à 4800 bit/s

On obtient alors $\mu_x(a_k) = \mu_x(b_k) = 0$

$$\sigma_x^2 = E(x^2) - E^2(x) = E(x^2) \quad \text{car } E^2(x) = 0$$

$$\sigma_x^2 = E(a_k^2) = \sum_{k=1}^{n} a_k^2 . \text{ prob}(a_k)$$

La densité spéctrale de puissance est la transformée de Fourier de la fonction d'auto-corrélation (théorème de Wiener Kintchin) .

$$\phi_{a_k}(f) = \sigma_x^2 \, sinc^2(Tf) = \sigma_x^2 . \frac{\sin^2(\pi Tf)}{(\pi Tf)^2} = \phi_{b_k}(f)$$

La densité spéctrale du signal numérique avant la modulation est la même pour le signal en a_k et b_k .

Aprés modulation on a l'expréssion générale de la densité spectrale

$$\phi_x(f) = \frac{1}{4} \left[\phi_{a_k}(f+f_o) + \phi_{a_k}(f-f_o) \right]$$

$$\phi_x(f) = \frac{\sigma_x^2}{4} \left[sinc^2(T(f+f_o)) + sinc^2(T(f-f_o)) \right]$$

Aprés filtrage le signal $y(t) = x(t) * h(t)$
ou $h(t)$ est la réponse impulsionnelle du filtre

$$h(t) = F^{-1}\left[H(j.2\pi f) \right]$$

d'où $\phi_y(f) = \phi_x(f) . | H(j.2\pi f) |^2$

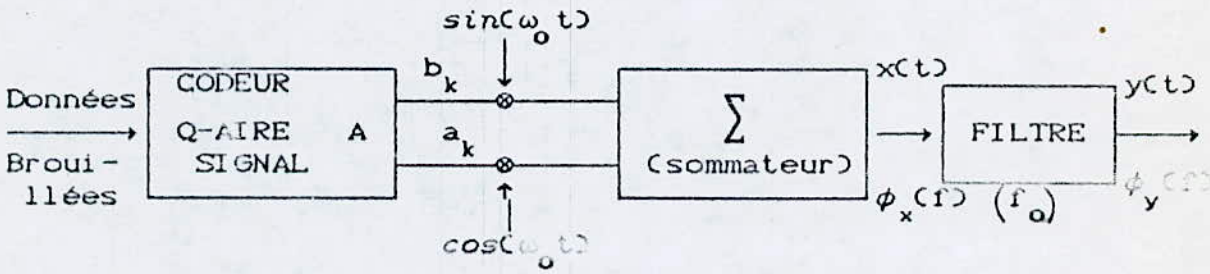


FIG 26 Cheminement du signal émis

A+8.1 *Cas du débit 4800 bit/s*

a_k et b_k prennent des valeurs parmi l'ensemble :

$\{ -3 , -1 , +1 , +3 \}$.

$p(-3) = p(-1) = p(1) = p(3) = 1/4 \Rightarrow \sigma_x^2 = 1/4(9+1+9+1) = 5$

$$\phi_x(f) = \frac{5}{4} \left[sinc^2(T(f+f_o)) + sinc^2(T(f-f_o)) \right]$$

A+8.2 *Cas du débit 9600 bit/s*

$$\sigma_x^2 = \sum_{i=1}^{n} p(a_i) \cdot [a_i]^2$$

Les probabilités affectées aux a_k telles qu'elles apparaissent dans la table de l'avis V32 (fig 21) sont egales aux probabilites des b_k

- Codage non redondant

$a_k \in \{ -3 , -1 , +1 , +3 \}$

$p(a_k=3) = p(b_k=3) = 4/16=1/4$ $\quad p(a_k=1) = p(b_k=1) = 4/16=1/4$

$p(a_k=-1) = p(b_k=-1) = 4/16=1/4$ $\quad p(a_k=-3) = p(b_k=-3) = 4/16=1/4$

On trouve $\sigma_x^2 = 5$

$$\phi_x(f) = \frac{5}{4} \left[sinc^2(T(f+f_o)) + sinc^2(T(f-f_o)) \right]$$

- Codage redondant

$a_k \in \{ -4 , -3 , -2 , -1 , 0 , 1 , 2 , 3 , 4 \}$

$P(-4) = 2/32$	$P(-3) = 3/32$	$P(-2) = 4/32$
$P(-1) = 5/32$	$P(0) = 4/32$	$P(+1) = 5/32$
$P(+2) = 4/32$	$P(+3) = 3/32$	$P(+4) = 2/32$

D'ou $\sigma_x^2 = E(x^2) = 5$

$$\phi_x(f) = \frac{5}{4} \left[sinc^2(T(f+f_o)) + sinc^2(T(f-f_o)) \right]$$

On obtient donc la même densité spéctrale du signal à transmettre pour les trois type de codage .

A.9 *La fonction modem-terminal : interface numérique*

L'interface entre modem et terminal à pour rôle de permettre la gestion par le terminal du déroulement d'une communication. Dans le cas général, la communication comprend quatre phases :- établissement d'un circuit entre les deux correspondants.
- initialisation de la transmission .
- transmission .
- libération de la liaison .

L'établissement d'un circuit consiste à effectuer les opérations qui permettent de créer un chemin pour la transmission des données .

La phase d'initialisation à pour but de conditionner les équipements pour qu'ils soient en mesure de transmettre les informations d'une façon correcte .

la phase de trasmission est la période au cours de laquelle les informations utiles sont effectivement transmises .

La phase de libération termine la communication une fois la tansmission achevée et libère le circuit .

L'enchainement de ces differentes opérationsnécessite un dialogue entre le terminal et le modem qui éxécute et rend compte .

Ce dialogue s'éffectue à travers l'interface numérique .

La standardisation de l'interface porte sur trois groupes de caractéristiques .
-les caracteristiques fonctionnelles .
-les caractéristiques électriques .
-les caractéristiques mécaniques .

On s'interesse éssentiellement à l'aspect fonctionnel qui recouvre la définition des signaux échanges et leur fonctions .

Le CCITT a défini deux types d'interfaces fonctionnels respectivement décrits dans les avis V24 et X24 .

L'interface V24 est à la fois la plus ancienne et la plus utilisée .

L'avis V24 définit une ligne de démarcation entre terminal et modem , chaque type de commande ou de signalisation est materialisée au niveau de l'interface par un circuit physique distinct .

Les circuits se répartissent en deux groupes :
- Le groupe de la serie 100 pour utilisation générale
- le groupe de la serie 200 reservé a l'appel automatique

La serie 100 comprend une quarantaine de circuits , la serie 200 en comprend 12 .

Tous ces circuits vehiculent des informations à 2 états
état 1 = repos , état 0 = travail

A.10 *Procédure de communication*

La communication à distance posait deux problèmes :

a) Il n'éxiste qu'une seule voie physique (la ligne) pour transmettre en série aussi bien les données que les informations de contrôle .
b) Le moyen de communication (la ligne) n'est pas fiable et la réception (des informations de contrôle et des données) n'est pas spontanée .

Ainsi pour pallier à ces deux inconvénients , un ensemble de règles ont été établi pour la reconnaissance des méssages valides , de reprise sur erreur et de réinitialisation .

Cet ensemble est appellé procédure de communication .

La fonction essentielle de la procedure est de garantir la fiabilité du passage d'information de l'émetteur au récepteur á travers un moyen de communication dont la fiabilité est insuffisante .

Cette procédure doit tenir compte des caractéristiques de la liaison de données , sa complexité dépendra des performances attendues :

- Le temps de propagation
- La nature de la liaison : unidirectionnelle , bidirectionnelle à l'alternat ou simultanée .
- Le débit binaire utilisé qui définit le temps de transfert d'un bit ou d'un caractère .
- L'utilisation de circuits commutés qui nécessitent une phase d'établissement (et la libération) avant (et aprés) le tranfert des données .

La procédure va donc prendre en charge :
- Le transfert de l'information ,ceci suppose une identification de la source , et du destinataire .
- La structuration des données et des séquences de commande
- La supervision de la liaison , ceci se traduit par l'enchainement des commandes des réponses, des accusés de réception aux messages reçus .
- Les reprises en cas d'érreur ; lorsqu'un méssage est erroné , il est réemis . Un nombre maximal de réessais est prévu .
- La temporisation (time-out) ; pour éviter de bloquer les échanges , on attend une réponse pendant un temps déterminé à l'avance .
- La gestion des équipements spécifiques aux transmissions (modem , contrôleur , multiplexeur) .

La procédure doit donc connaître les caracteristiques du matériel utilisé pour éviter les incompatibilités (temps d'égalisation ,conditions de maintien de la synchronisation)

Dans l'avis V32 du CCITT , la ligne de transmission est du type bidirectionnelle simultanée (full duplex) il n ' y a pas de temps à perdre à inverser le sens de la transmission (retournement des Modems) .

En éffet , il est beaucoup plus simple de considerer qu'une transmission duplex est l'association de deux transmission simples , les deux sens peuvent être utilisé de manière asynchrone ou les coupler pour réaliser par exemple un mode convesationnel .

Les modems définis par l'avis V32 sont universels , en effet ces modems doivent à la fois pouvoir communiquer avec leurs homologues du même avis (i.e V32) et avec les modems definis par l'avis V26 ter .

B : EMETTEUR : RÉALISATION MICROPROGRAMMÉ

B.1 Diagramme général de l'émetteur

Ce diagramme est destiné à simuler les differents blocs de l'émetteur , conformément aux exigences du [illegible] (avis V32 du CCITT) .

Comme nous disposons d'un alphabet code limité (tables de l'avis V32) , pour chacun des éléments composant cet alphabet , on procedera au codage correspandant , filtrage et modulation , et ce par traitement numérique .

Une fois que le calcul des échantillons est fait ,les résulats seront sauvegardés dans la mémoire ROM du microprocesseur TMS 320-10 sous forme d'échantillons .

Ainsi pour chaque séquence binaire codée , il suffira d'activer et de présenter à la sortie du système de traitement de données (port de sortie du microprocesseur) le groupe d'échantillons correspandant .

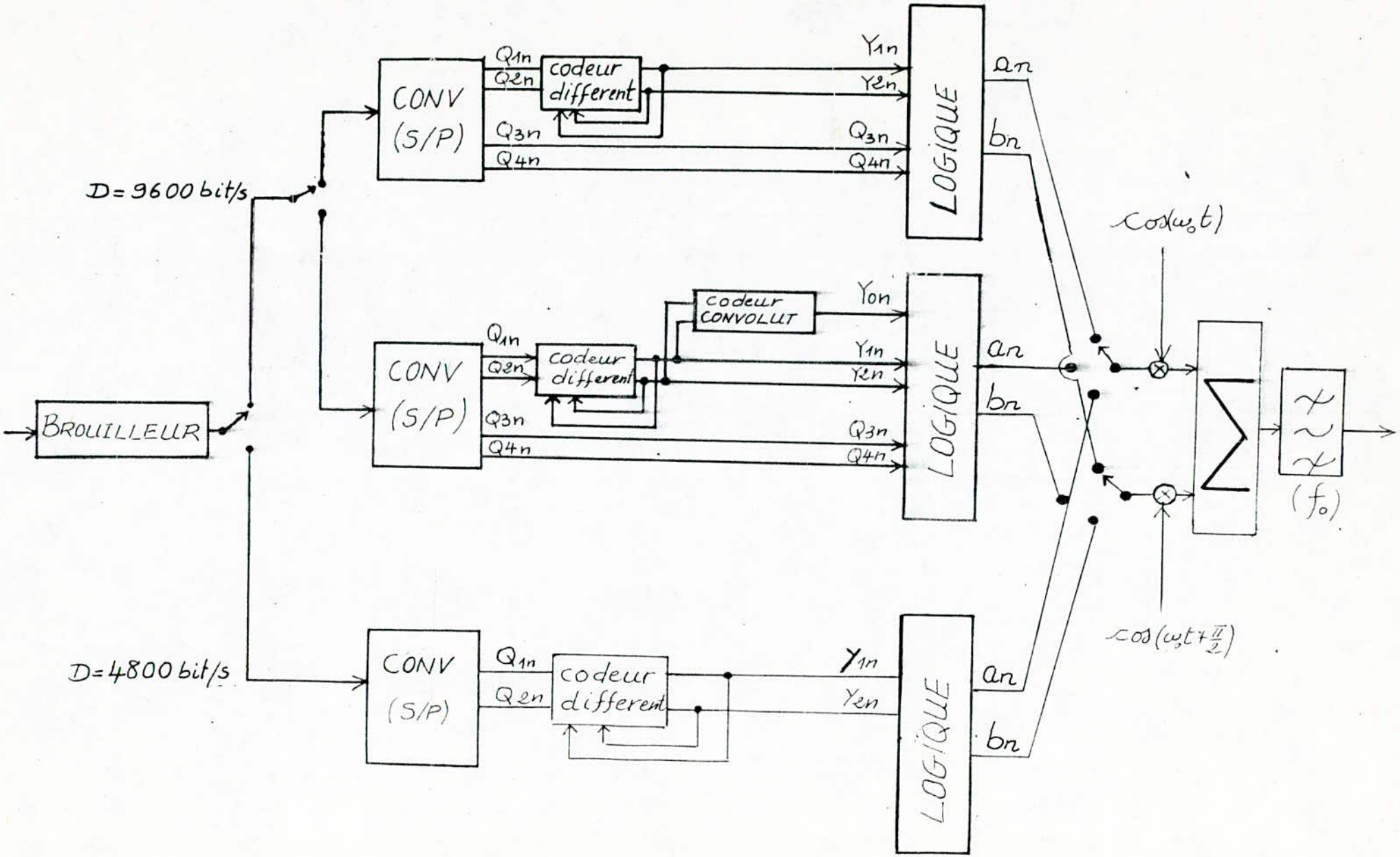
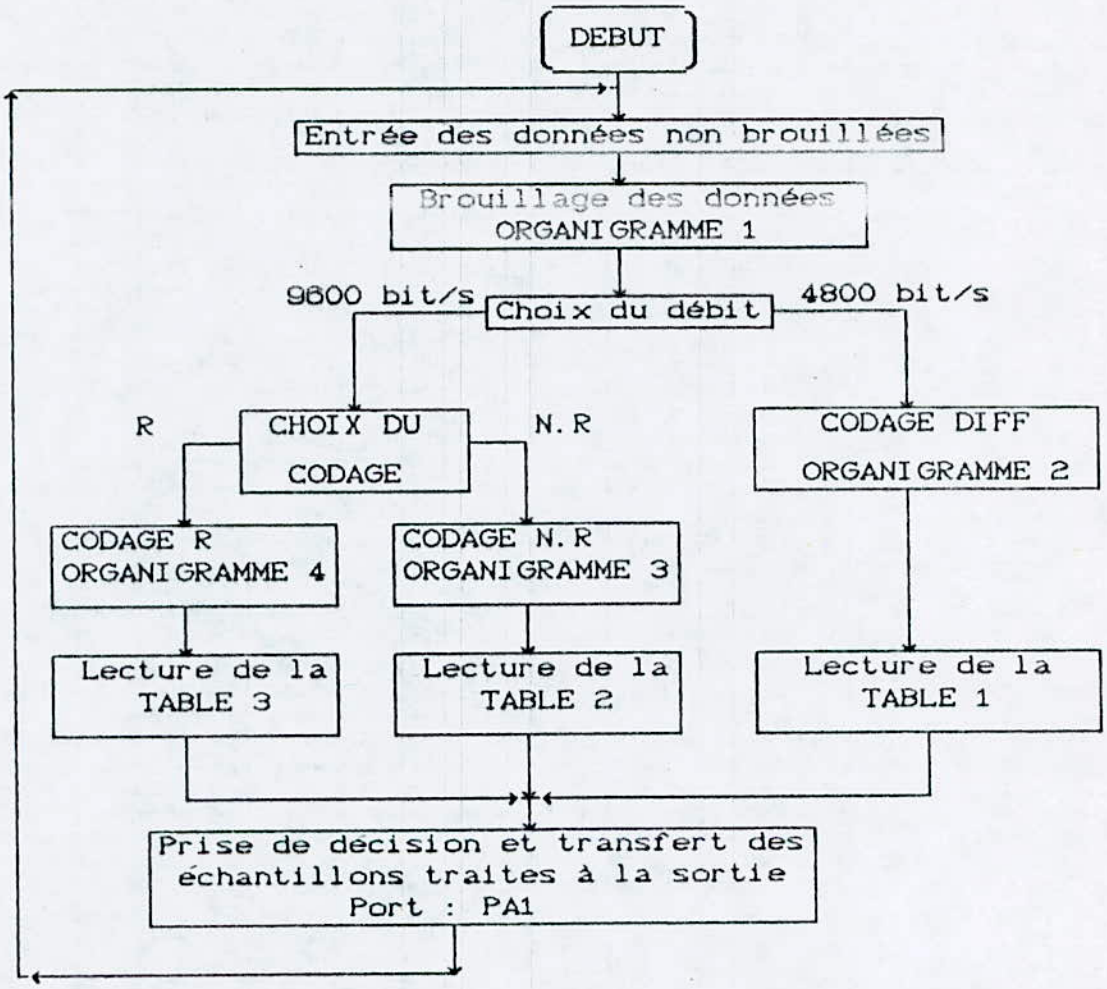


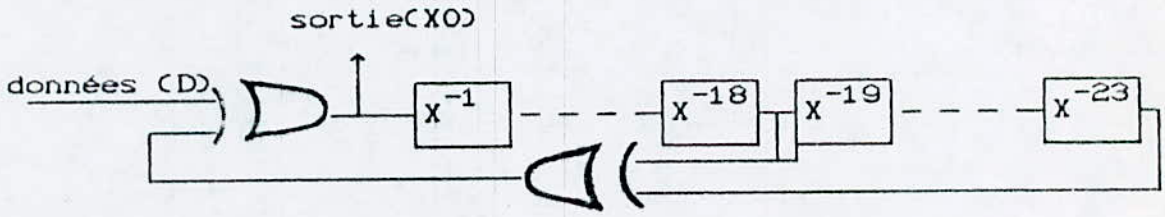
DIAGRAMME GENERAL DE L'ÉMETTEUR.

B.2 *Organigramme general :*

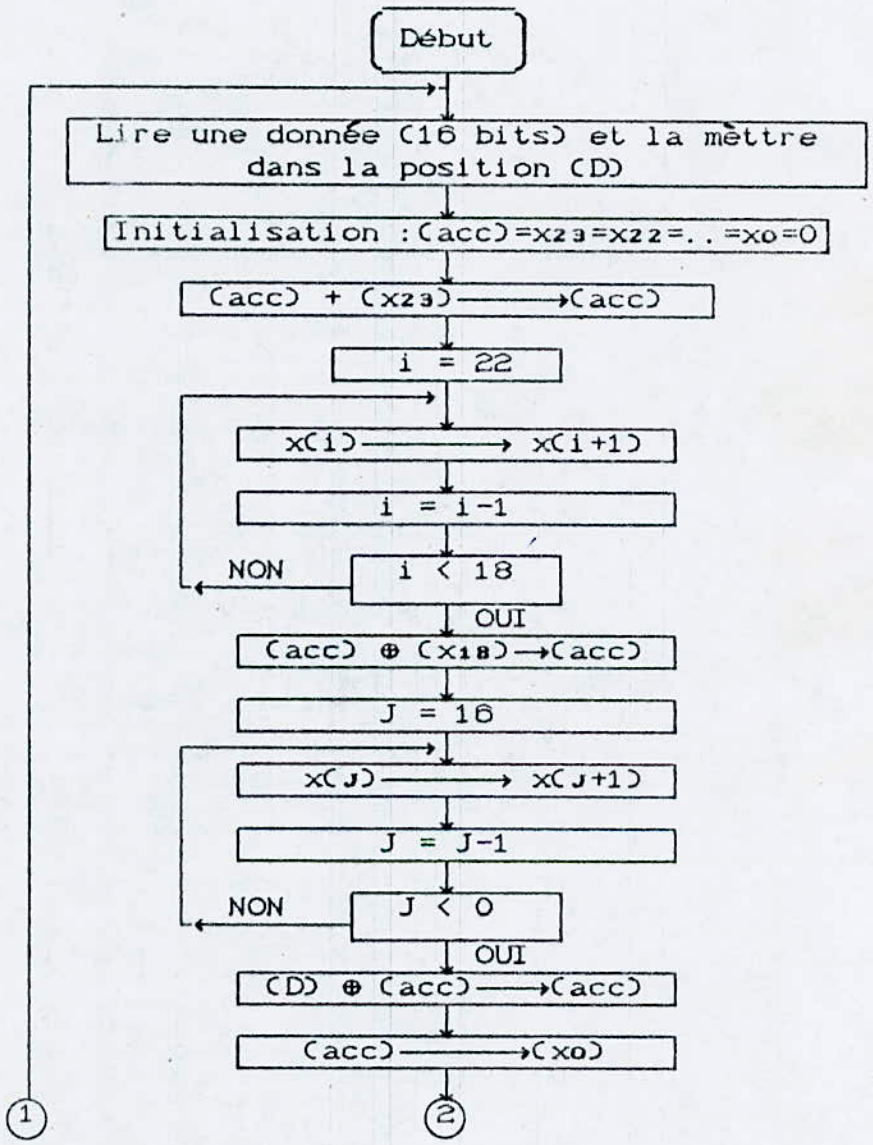


B.2.1 Brouillage des données

B.2.1.1 Le circuit brouilleur à simuler est le suivant



B.2.1-2 **L'organigramme :**



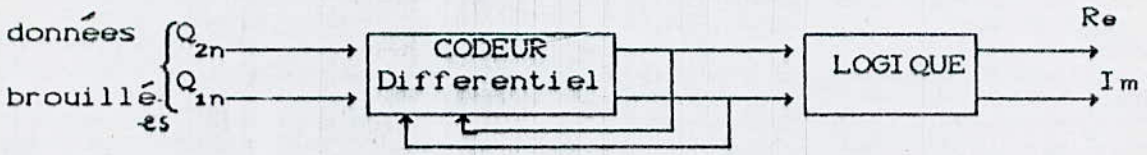
B.3.1 *Codage des données brouillées :*

B.3.1-1 **Codeur pour 4800 bit/s :**

Le schéma de ce codeur est constitué de deux blocs:

Le codeur differentiel et un bloc d'affectation logique qui associe à chaque combinaison binaire des Y_{1n} , Y_{2n} des niveaux de sortie :

Re , Im (signaux multiniveaux)



D'aprés la table de vérité du codeur differentiel , on trouve en utilisant le diagramme de KARNAUGH , les éxpressions :

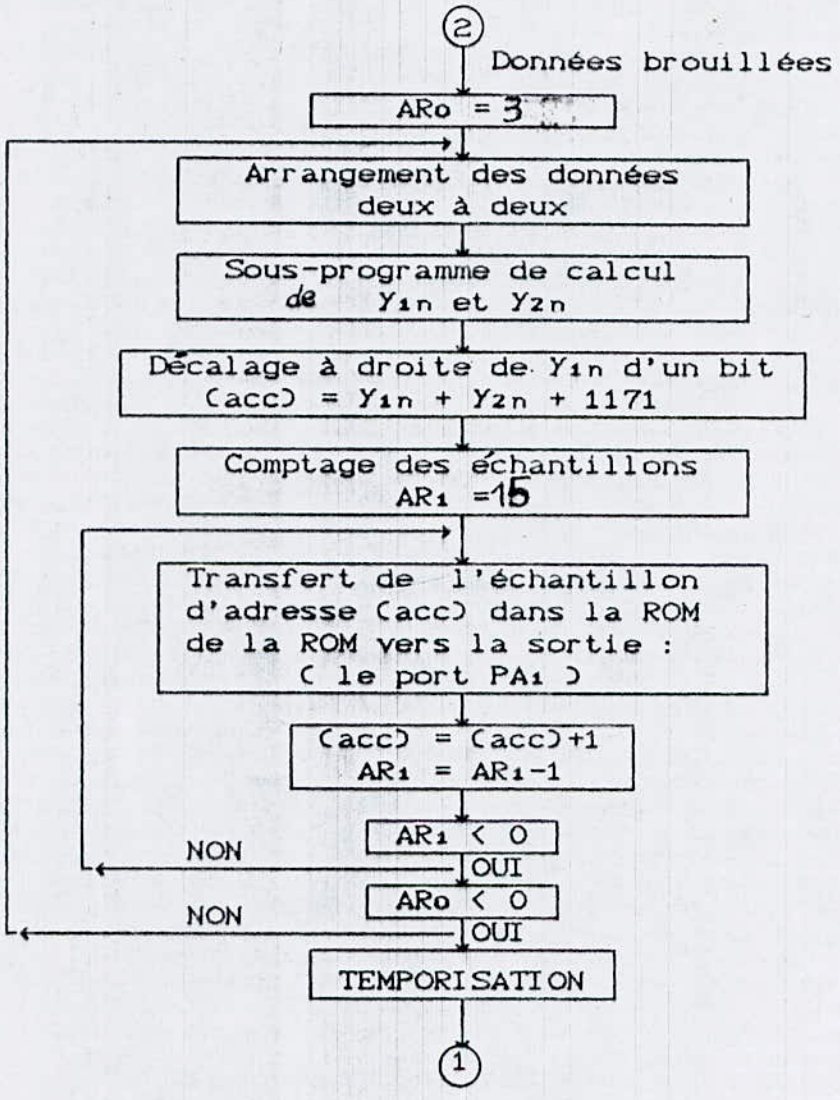
$$Y_{1n} = Q_{1n}(\overline{Q}_{2n}\overline{Y}_{1(n-1)} + Q_{2n}\overline{Y}_{2(n-1)}) + \overline{Q}_{1n}(Q_{2n}Y_{1(n-1)} + \overline{Q}_{2n}Y_{2(n-1)})$$

$$Y_{2n} = Q_{1n}(Q_{2n}Y_{1(n-1)} + \overline{Q}_{2n}\overline{Y}_{2(n-1)}) + \overline{Q}_{1n}(\overline{Q}_{2n}\overline{Y}_{1(n-1)} + Q_{2n}Y_{2(n-1)})$$

Comme le codeur pour 9600 bit/s non-redondant utilise un codeur differentiel identique à celui utilisè pour le débit 4800 bit/s , alors il est commode et avantageux de créer un sous-programme qui calcule les éxpressions de Y_{1n} et Y_{2n} aussi bien pour le 4800 bit/s que pour le 9600 bit/s N.R .

Ceci permet , en effet d'avoir un gain en éspace mémoire assez important , vu l'étroitesse de l'éspace mémoire de la ROM qui nous est alloué (0 - 1535) et le nombre important d'échantillons qu'il faut stocker définitivement dans la ROM.

ORGANIGRAMME

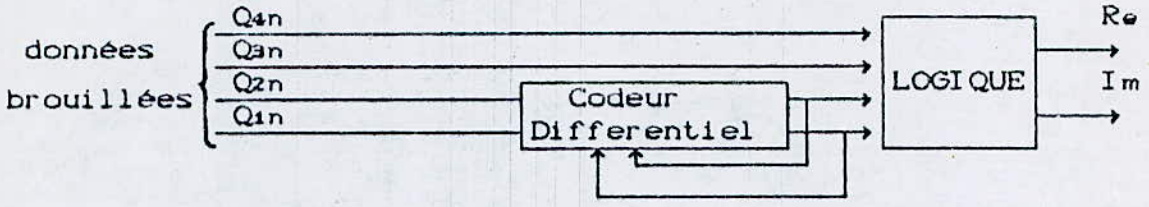


B.3.1.2 Codeur pour 9600 bit/s non redondant :

Le codeur considéré est constitué de deux parties :

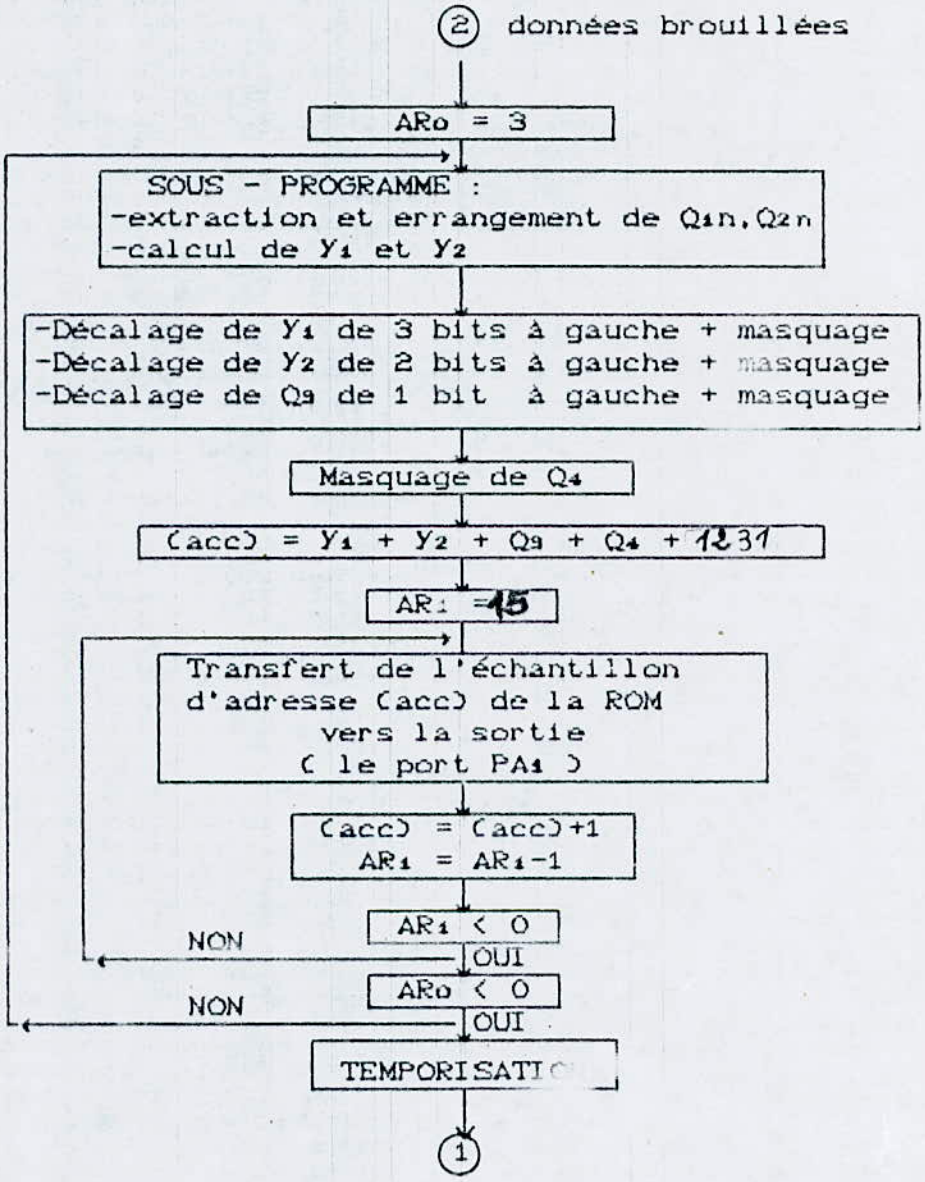
-Un codeur differentiel qui code Q_{1n} , Q_{2n} uniquement , donnant Y_{1n} , Y_{2n} .
-Une logique d'affectation qui associe à chaque combinaison binaire Y_{1n} , Y_{2n} , Q_{3n} , Q_{4n} deux sorties à plusieurs niveaux .

Le schéma bloc du codeur à simuler :



A noter que le codeur differentiel est le même que celui utilisé pour le 4800 bit/s .
Comme l'affectation logique est prise à partir d'une combinaison binaire de 4 bits , il faut reserver $2^4 \times 15$ positions mémoires de la ROM pour stocker les échantillons .

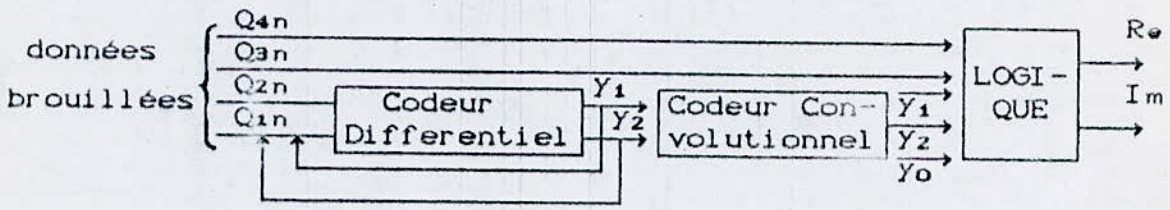
ORGANIGRAMME



B.3.1-3 *Codeur pour 9600 bit/s redondant :*

Le codeur considéré est constitué de 3 parties :
-Un codeur differentiel different des précedants .
-Un codeur convolutionnel (génération d'un bit de redondance) .
-Logique d'affectation à partir d'une combiaison binaire de 5 bits .

Le schema bloc :



D'apres la table du codeur differentiel (voir chapitre précedant) on obtient les expressions :

$$Y_{1n} = Q_{1n} \oplus Y_{1n}$$

$$Y_{2n} = Q_{1n} \cdot Y_{1(n-1)} \oplus (Q_{2n} \oplus Y_{2(n-1)})$$

D'aprés la structure du codeur convolutionnel (voir page N° 32) le bit de redondance sera calculé à partir des 3 équations suivantes :

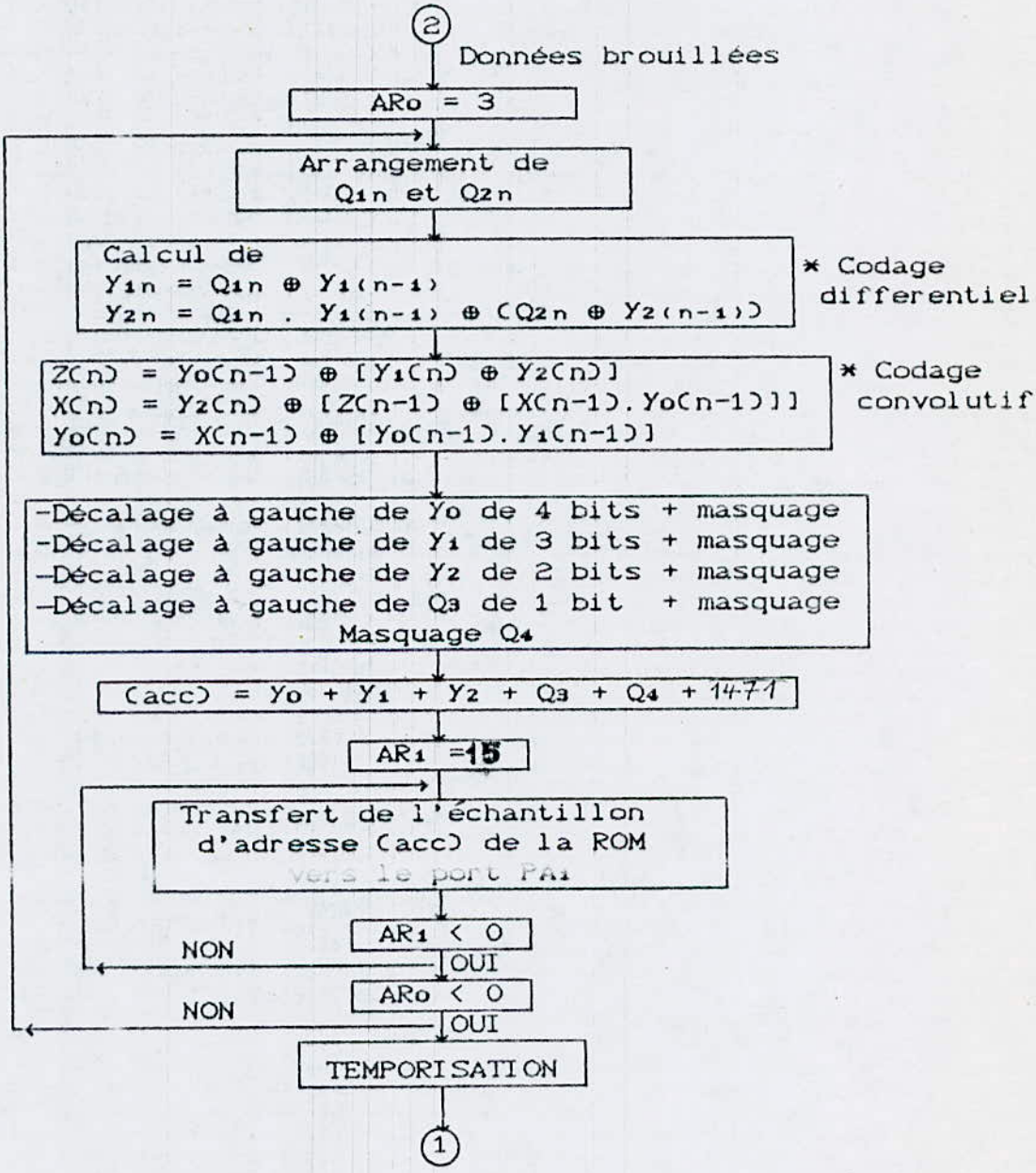
$$Z(n) = Yo(n-1) \oplus [\, Y_1(n) \oplus Y_2(n) \,]$$

$$X(n) = Y_2(n) \oplus [\, Z(n-1) \oplus [\, X(n-1) . Yo(n-1) \,] \,] .$$

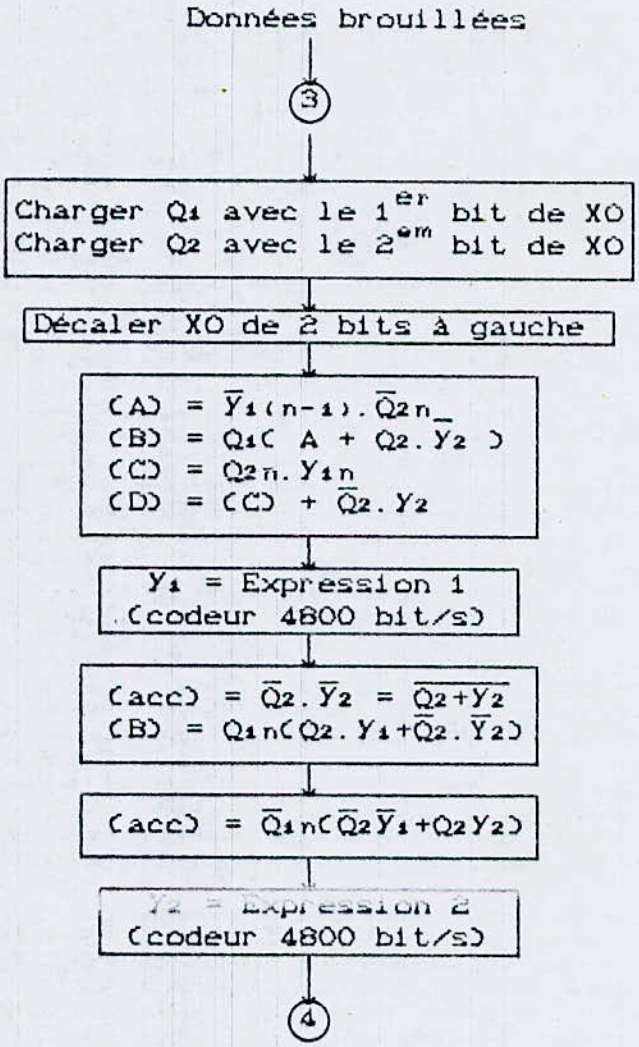
$$Yo(n) = X(n-1) \oplus [Yo(n-1) . Y_1(n-1) \,] .$$

Comme l'affectation logique est prise à partir d'une combinaison linéaire de 5 bits , alors il faut $2^5 \times 15$ positions mémoires à reserver dans la ROM pour les échantillons .

ORGANIGRAMME



ORGANIGRAMME DE CALCUL DES EXPRESSIONS DE Y_1 ET Y_2



CONCLUSION

Dans cette étude , on a realisé la simulation d'un émeteur microprogarammé pour modem V32 à base du microprocesseur TMS 320-10.

Pour un fonctionnement en temps réel de notre simulateur , il a été necessaire d'introduire dans le programme principal des temporisations , et ce à cause des performances du microprocesseur TMS 320-10 qui réalise de nombreuses opérations en un seul cycle d'horloge (200 n.s) .

Les temporisations assez longues dans notre programme peuvent être utilisés pour réaliser d'autres fonctions de communicatation , tel la génération de signaux de procédure .

Parmi les interêts de la simulation , le principale bénéfice a été de pouvoir éviter la complexité d'une réalisation pratique , tel le circuit brouilleur ; ainsi que le codeur convolutionnel trés difficile à mettre au point .

Ainsi la simulation se revèle alors être un puissant moyen d'investigation simple et éfficace qui nous a permet d'introduire des modifications dans la conception même du système , tel le traitement numérique que nous avons introduit pour remplacer les blocs de modulation et filtrage .

ANNEXE A

PRESENTATION DU TMS 320-10 [8];[9]

A-EXPOSE DES CARACTERISTIQUES:

1_Description:

Le TMS 320-10 est un processeur numérique du signal,faisant partie de la famille TMS 320 basée sur une architecture parallèle 16/32 bits.
Les principales caractéristiques du microprocesseur TMS 320-10 sont:

_Temps de cycle d'une instruction de 200 ns.
_Mémoire RAM intégrée de 288 octets.
_Multiplication 16 par 16 bits en 200 ns.
_Accumulateur et unité arithmétique et logique sur 32 bits.
_Huit voies d'entrées et huit voies de sorties.
_Fréquence d'horloge est de 5 MHz.

2_Constitution du Hardware:

Le TMS 320-10 possède quatre éléments arithmétiques de base qui sont:L'unité arithmétique et logique (ALU),l'accumulateur,le multiplieur et le décaleur.

2.1Unité arithmétique et logique (ALU):
L'ALU manipule des données sur des mots de 32 bits.En plus des opérations classiques,elle éffectue aussi des opérations logiques entre les 16 bits les moins significatifs de l'accumulateur et la valeur de la mémoire de données.

2.2_Accumulateur:
Il opère sur des mots de 32 bits.

2.3_Multiplieur:
La multiplication est réalisée à l'aide d'un multiplieur cablé (16 par 16) bits en complément à 2,elle utilise 2 registres pour le multiplicande et le résultat du produit.

2.4_Le décaleur:
Deux décaleurs à gauche sont disponibles dans la structure TMS 320:

_Un décaleur de 0 à 15 positions,permet de manipuler la donnée avant de la stocker en mémoire.
_Un autre décaleur de 0,1 ou 4 positions, permet le décalage des 16 bits de poids le plus fort de l'accumulateur avant le rangement de la donnée en RAM.

2.5_Mémoire de programme:
Le TMS 320-10 est équipé d'une mémoire programme (ROM) interne de 1536 mots éxtensibles à l'aide d'une mémoire (ROM) externe de 2560 mots.

Le TMS 320-10 opère uniquement en mode microprocesseur ou les instructions adressées de 0 à 4095 sont toutes dans la ROM externe.

2.6_Mémoire de données:
Elle est constituée de 144 mots.

2.7_Compteur programme (PC):
C'est un registre de 12 bits contenant l'adresse courante de la mémoire programme.

2.8_Registres auxiliaires 0 et 1 (ARI):
Composés de 16 bits chacun.

2.9_Registre T:
C'est un registre de 16 bits contenant le multiplicande dans une opération de multiplication.

2.10_Registre P:
C'est un registre 32 bits contenant le résultat de la multiplication.

2.11 Horloge:
Le TMS 320-10 peut utiliser son oscillateur interne ou un oscillateur externe comme horloge.
L'oscillateur interne est mis en service par la connection d'un cristal à travers X1 et X2/CLKIN (voir figure).
La fréquence CLKOUT est le quart de la fréquence CLKIN.

3_Instructions:

Le TMS 320-10 possède un jeu de 60 instructions dont la majorité sont opérées en un mot et un cycle seulement,on peut les classer dans les catégories suivantes:
_Instructions de l'accumulateur.
_Instructions des registres auxiliaires.
_Instructions des registres T et P et de la multiplication.
_Instruction de controle.
_Instructions de branchement.

4_Assembleur:

Le programme en langage assembleur,codé par le programmeur, est appelé programme source.
Avant qu'il ne soit éxécuté par le calculateur,ce programme source doit être traité par le programme macro-assembleur pour obtenir un programme en langage machine afin qu'il puisse être éxécuté par le similateur. Ce traitement est appelé assemblage.

Les directives de l'assembleur contrôllent le procédé qui permet de faire un programme en langage machine à partir du programme en langage assembleur.

B_SIMULATION:

On dispose d'une disquette contenant trois programmes logiciels principaux éxécutables:

_LE MACRO-ASSEMBLEUR XASM3.EXE: qui traduit le langage assembleur en langage machine éxécutable directement par le similateur.

_EDITEUR DE LIENS LINKER.EXE: permet à un programme d'être éxécuté par des modules séparés,et assemblés par la suite pour former un programme complet,cela permet l'utilisation des mêmes modules pour différents problèmes.

_LE SIMILATEUR SIM.EXE: permettant de simuler les opérations du TMS 320-10 et de vérifier grâce à un mode de mise au point,la bonne éxécution d'un programme, il utilise le code objet du TMS 320-10 produit par le macro-assembleur /éditeur de liens.

I_MISE EN OEUVRE DE LA SIMULATION SUR MICRO-ORDINATEUR COMPATIBLE IBM.

La mise en oeuvre de la simulation se fait par un chargement au préalable des 3 programmes de la disquettes dans le disque dur. aprés le chargement les étapes à suivre sont les suivantes :

I_1_Ecriture du programme en langage assembleur dans un fichier crée par l'instruction EDIT.

C> EDIT (nom du fichier) (.ASM) (return)

I_2 Exécution du macro-assembleur.

Ce logiciel s'éxécute comme suite:

C> XASM3 (return).

Le macro-assembleur demande ensuite les noms de fichiers source,listing et code objet à créer:

XASM3 (source file .ASM) : (nom du fichier).ASM

XASM3 (listing file .LST) : (nom de fichier) .LST

XASM3 (object file . MPO) : (nom de fichier) .MPO

La création de ces fichiers se fait systématiquement dans le disque dur ou ils seront sauvegardés.

I_3 Exécution du simulateur:

Pour éxécuter le simulateur du TMS 320-10 on utilise la commande

```
C> SIM     (return).
   ---     ---------
```

Le système doit maintenant éxécuter le similateur,toutes les positions mémoire de programme et de données sont initialisées à zéro.

Le simulateur demande ensuite le mode de traitement qui sera simulé,mode microprocesseur (MP) ou mode microcalculateur (MC).

Dans le cas du TMS 320-10,on sélectionne le mode microprocesseur (MP) en faisant entrer un zéro (0) ou la commande return.

Le chargement de la mémoire de programme par le fichier écrit en code objet et le chargement de la mémoire de données, des registres internes ainsi que l'éxécution,s'éffectue à l'aide des commandes du simulateur.

II_COMMANDES DU SIMULATEUR:

Les commandes du simulateur du TMS 320-10 éxécutent les fonctions spécifiques du simulateur,elles sont visualisées par la commande:

_DM : qui affiche le menu des commandes.

III_CHARGEMENT DES FICHIERS:

Les fichiers sont chargés dans le simulateur à l'aide de la commande L (LOAD) comme suite:

```
C> SIM    (return)
   ---    -------
          (C) Copyright Texas Instruments
              Incorporated    1984.
           SIMULATION OF THE TMS 320-10
           VERSION  #  1.6

    0_ MICROPROCESSOR MODE  (ADDR 0-1535,OFF CHIP)
    1_ MICROCALCULATEUR MODE (ADDR 0-1535,ON CHOP)

0_ (return)
-  --------
YOU ARE IN THE MICROPROCESSOR MODE (ADDR 0-1535,OFF CHIP)
```

```
ENTER COMMAND  (HELP=(return))

L   (return)
-   --------
ENTER NEW OBJECT FILE

( Nom du fichier ).MPO     (return)
--------------------       --------

          * * * *  LOADING PROGRAM "NO$IDT"  * * * *

ENTER COMMAND  (HELP=(return))
```

Le procédé de chargement de la ROM est le même que celui de la RAM, seulement on doit faire entrer la commande ROM au lieu de RAM.

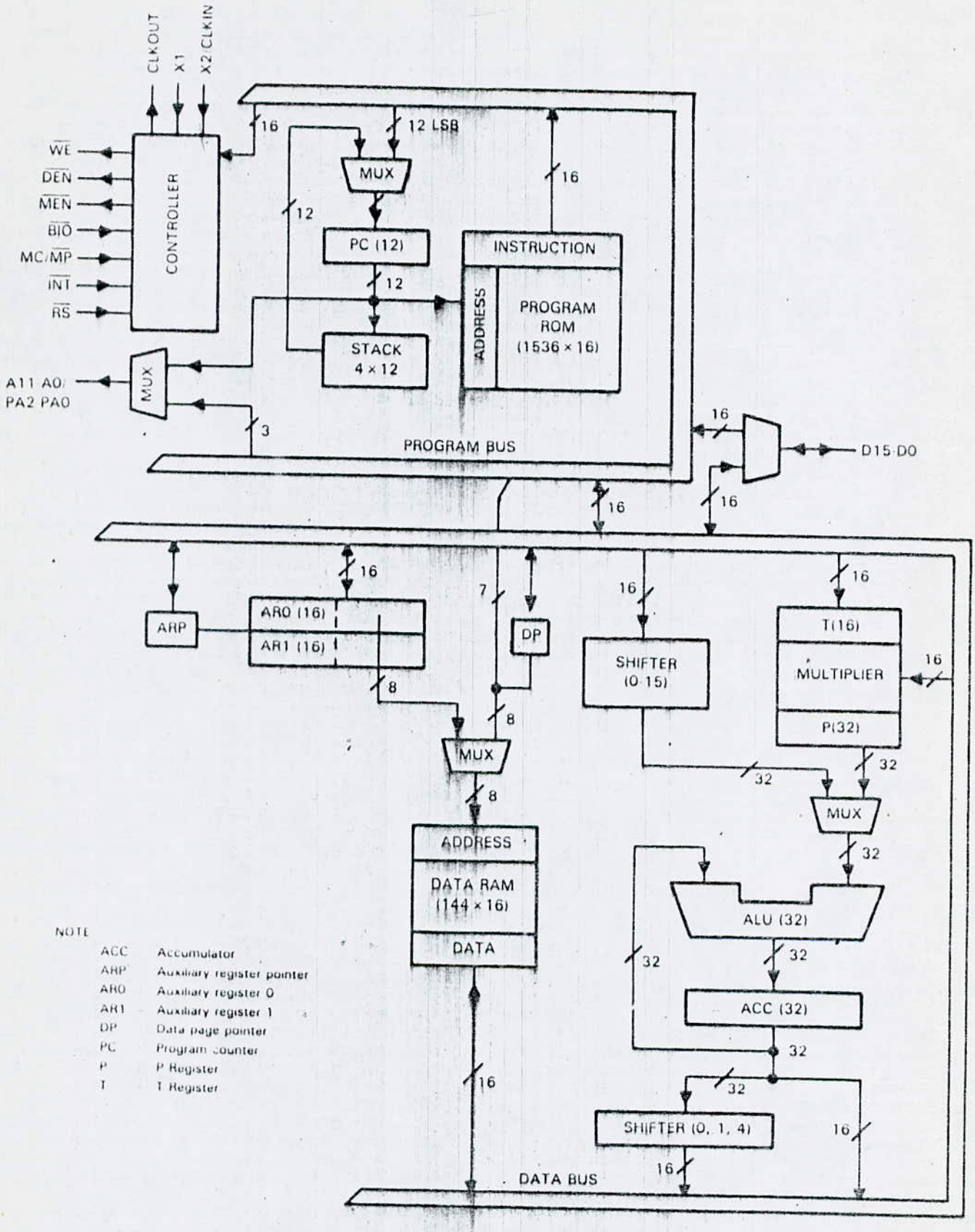
NOTE:La commande Q peut aussi etre utilisée pour quitter completement le simulateur .

Aussitot que le fichier écrit en code objet sera chargé dans le simulateur,mettre un arrêt de contrôle en utilisant la commande BIAQ. Cette commande est utilisée pour arreter l'éxécution à la dernière instruction du programme.
Aprés ceci on utilise la commande R (RUN),la procédure est la suivante :

```
ENTER COMMAND  (HELP=(return))
BIAQ   (return)
----   --------
BREAK ON INSTRUCTION ACQUISITION
ENTER THE ADDRESS (IN HEX)
3    (return)
-    --------
ENTER COMMAND  (HELP=(RETURN))
R     (return)
-     -------
```

IV_CHARGEMENT DE LA MEMOIRE DE DONNÉES:

```
ENTER COMMAND  (HELP=(return))
RAM   (return)
---   --------
ENTER STARTING ADDRESS (IN HEX)
1 = 0     "La position 1 contient la valeur 0"
5  (return)   "Modifier cette valeur par 5 (par éxemple)"
-  --------
1 = 5    (return)  "Passage à la position mémoire suivante"
         --------
2 = 0
```



NOTE

ACC	Accumulator
ARP	Auxiliary register pointer
ARO	Auxiliary register 0
AR1	Auxiliary register 1
DP	Data page pointer
PC	Program counter
P	P Register
T	T Register

FIGURE 2-1 — BLOCK DIAGRAM OF THE TMS320M10

CROSS-ASSEMBLER DIRECTIVES (CONCLUDED)

IDT assigns a name to the object module produced.

PROGRAM IDENTIFIER — **IDT**

Syntax: [<label>] ... IDT ... '<string>' ... [<comment>]

TITL supplies title to be printed in the heading of each page of the source listing.

PAGE TITLE — **TITL**

Syntax: [<label>] ... TITL ... '<string>' ... [<comment>]

LIST restores printing of the source listing.

RESTART SOURCE LISTING — **LIST**

Syntax: [<label>] ... LIST ... [<comment>]

UNL halts the source listing output until the occurrence of a LIST directive.

STOP SOURCE LISTING — **UNL**

Syntax: [<label>] ... UNL ... [<comment>]

PAGE causes the assembler to continue the source program listing on a new page. The PAGE directive is not printed in the source listing, but the line counter increments.

EJECT PAGE — **PAGE**

Syntax: [<label>] ... PAGE ... [<comment>]

DATA places one or values in one or more successive words of memory.

INITIALIZE WORD — **DATA**

Syntax: [<label>] ... DATA ... <exp> [, <exp>] ... ... [<comment>]

TEXT places one or more characters in successive words of memory.

INITIALIZE TEXT — **TEXT**

Syntax: [<label>] ... TEXT ... [-] '<string>' ... [<comment>]

EQU assigns a value to a symbol.

DEFINE ASSEMBLY-TIME CONSTANT — **EQU**

Syntax: <label> ... EQU ... <exp> ... [<comment>]

DEF makes one or more symbols available to other programs for reference.

EXTERNAL DEFINITION — **DEF**

Syntax: [<label>] ... DEF ... ... <symbol> [,<symbol>] ... ... [<comment>]

REF provides access to one or more symbols defined in other programs.

EXTERNAL REFERENCE — **REF**

Syntax: [<label>] ... REF ... <symbol> [,<symbol>] ... ... [<comment>]

SREF provides access to one or more symbols defined in other programs.

SECONDARY EXTERNAL REFERENCE — **SREF**

Syntax: [<label>] ... SREF ... <symbol> [,<symbol>] ... ... [<comment>]

LOAD is similar to REF, but the symbol does not need to be used in the module containing the LOAD. The symbol used in LOAD must be defined in some other module. LOADs are used with SREFs.

FORCE LOAD — **LOAD**

Syntax: [<label>] ... LOAD ... <symbol> [,<symbol>] ... ... [<comment>]

END terminates the assembly. The last source statement of a program is the END directive.

PROGRAM END — **END**

Syntax: [<label>] ... END ... [<symbol>] ... [<comment>]

COPY changes the source input for the assembler.

COPY SOURCE FILE — **COPY**

Syntax: [<label>] ... COPY ... <file name> ... [<comment>]

RTC HOTLINE NUMBERS

For help with the TMS32010, call the TI Regional Technology Center nearest you. The centers are staffed with applications engineers ready to answer all your questions.

Milan	0225-32451
Rieti	0746/6941
Freising (D)	08161/804407
Hannover (D)	0511/648021
Stuttgart (D)	0711/34030
Bedford	0234/223825
Stockholm	(08)-235480
Paris	(3) 946-97-12
Nice	(93) 20-01-01

TMS32010 DIGITAL SIGNAL PROCESSOR Programmer's Reference Card

PIN DESCRIPTIONS

PIN	FUNCTION	PIN	FUNCTION
1	A1/PA1	21	D5
2	A0/PA0	22	D4
3	MC/MP	23	D3
4	$\overline{RS}$	24	D2
5	$\overline{INT}$	25	D1
6	CLKOUT	26	D0
7	X1	27	A11
8	X2/CLKIN	28	A10
9	$\overline{BIO}$	29	A9
10	V_{SS}	30	V_{CC}
11	D8	31	$\overline{WE}$
12	D9	32	$\overline{DEN}$
13	D10	33	MEN
14	D11	34	A8
15	D12	35	A7
16	D13	36	A6
17	D14	37	A5
18	D15	38	A4
19	D7	39	A3
20	D6	40	A2/PA2

ASCII REFERENCE TABLE

	00	10	20	30	40	50	60	70
00	NUL	DLE	SP	0	@	P	\	p
01	SOH	DC1	!	1	A	Q	a	q
02	STX	DC2	"	2	B	R	b	r
03	ETX	DC3	#	3	C	S	c	s
04	EOT	DC4	$	4	D	T	d	t
05	ENQ	NAK	%	5	E	U	e	u
06	ACK	SYN	&	6	F	V	f	v
07	BEL	ETB	'	7	G	W	g	w
08	BS	CAN	(	8	H	X	h	x
09	HT	EM	)	9	I	Y	i	y
0A	LF	SUB	*	:	J	Z	j	z
0B	VT	ESC	+	;	K	[	k	{
0C	FF	FS	,	<	L	\	l	|
0D	CR	GS	-	=	M	]	m	}
0E	SO	RS	.	>	N	^	n	~
0F	SI	US	/	?	O	_	o	DEL

HEX-DECIMAL TABLE

HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
0000	0	000	0	00	0	0	0
1000	4,096	100	256	10	16	1	1
2000	8,192	200	512	20	32	2	2
3000	12,288	300	768	30	48	3	3
4000	16,384	400	1,024	40	64	4	4
5000	20,480	500	1,280	50	80	5	5
6000	24,576	600	1,536	60	96	6	6
7000	28,672	700	1,792	70	112	7	7
8000	32,768	800	2,048	80	128	8	8
9000	36,864	900	2,304	90	144	9	9
A000	40,960	A00	2,560	A0	160	A	10
B000	45,056	B00	2,816	B0	176	B	11
C000	49,152	C00	3,072	C0	192	C	12
D000	53,248	D00	3,328	D0	208	D	13
E000	57,344	E00	3,584	E0	224	E	14
F000	61,440	F00	3,840	F0	240	F	15

TEXAS INSTRUMENTS

SYMBOLS FOR INSTRUCTION SET SUMMARY

SYMBOL	MEANING
D	Data memory address field
I	Addressing mode bit
K	Immediate operand field
PA	3 bit port address field (PA0 through PA7 are predefined assembler symbols equal to 0 through 7, respectively)
R	1 bit operand field specifying auxiliary register
S	4 bit left shift code
X	3 bit accumulator left shift field

INSTRUCTION SET SUMMARY

MNEMONIC	DESCRIPTION	NO CYCLES	NO WORDS	OPCODE INSTRUCTION REGISTER
				15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
ABS	Absolute value of accumulator	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0
ADD	Add to accumulator with shift	1	1	0 0 0 0 ← S → I ← D →
ADDH	Add to high order accumulator bits	1	1	0 1 1 0 0 0 0 0 1 ← D →
ADDS	Add to accumulator with no sign extension	1	1	0 1 1 0 0 0 0 1 1 ← D →
AND	AND with accumulator	1	1	0 1 1 1 1 0 0 1 1 ← D →
APAC	Add P Register to accumulator	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1
B	Branch unconditionally	1	2	1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BANZ	Branch on auxiliary register not zero	2	2	1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BGEZ	Branch if accumulator ≥ 0	2	2	1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BGZ	Branch if accumulator > 0	2	2	1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BIOZ	Branch on BIO = 0	2	2	1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BLEZ	Branch if accumulator ≤ 0	2	2	1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BLZ	Branch if accumulator < 0	2	2	1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BNZ	Branch if accumulator ≠ 0	2	2	1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BV	Branch on overflow	2	2	1 1 1 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
BZ	Branch if accumulator = 0	2	2	1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
CALA	Call subroutine from accumulator	2	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 0 0
CALL	Call subroutine immediately	2	2	1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ← BRANCH ADDRESS →
DINT	Disable interrupt	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1
DMOV	Copy contents of data memory location into next location	1	1	0 1 1 0 1 0 0 1 1 ← D →
EINT	Enable interrupt	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 1 0
IN	Input data from port	2	1	0 1 0 0 0 ← PA → I ← D →
LAC	Load accumulator with shift	1	1	0 0 1 0 ← S → I ← D →
LACK	Load accumulator immediate	1	1	0 1 1 1 1 1 1 1 0 ← K →
LAR	Load auxiliary register	1	1	0 0 1 1 1 0 0 R I ← D →
LARK	Load auxiliary register immediate	1	1	0 1 1 1 0 0 0 R ← K →
LARP	Load auxiliary register pointer immediate	1	1	0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 0 K
LDP	Load data memory page pointer	1	1	0 1 1 0 1 1 1 1 1 ← D →
LDPK	Load data memory page pointer immediate	1	1	0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 K
LST	Load status register	1	1	0 1 1 1 1 0 1 1 1 ← D →
LT	Load T Register	1	1	0 1 1 0 1 0 1 0 1 ← D →
LTA	LTA combines LT and APAC into one instruction	1	1	0 1 1 0 1 1 0 0 1 ← D →
LTD	LTD combines LT, APAC and DMOV into one instruction	1	1	0 1 1 0 1 0 1 1 1 ← D →
MAR	Modify auxiliary register and pointer	1	1	0 1 1 0 1 0 0 0 1 ← D →
MPY	Multiply with T Register, store product in P Register	1	1	0 1 1 0 1 1 0 1 1 ← D →
MPYK	Multiply T Register with immediate oper and, store product in P Register	1	1	1 0 0 ← K →
NOP	No operation	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
OR	OR with accumulator	1	1	0 1 1 1 1 0 1 0 1 ← D →
OUT	Output data to port	2	1	0 1 0 0 1 ← PA → I ← D →
PAC	Load accumulator from P Register	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 0
POP	Pop stack to accumulator	2	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 0 1
PUSH	Push stack from accumulator	2	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 0 0
RET	Return from sub routine	2	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 0 1
ROVM	Reset overflow mode	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 0 1 0
SACH	Store high order accumulator bits with shift	1	1	0 1 0 1 1 ← X → I ← D →
SACL	Store low order accumulator bits	1	1	0 1 0 1 0 0 0 0 1 ← D →
SAR	Store auxiliary register	1	1	0 0 0 1 1 0 0 0 R I ← D →
SOVM	Set overflow mode	1	1	0 1 1 1 1 1 1 1 1 1 0 0 0 1 0 1 1
SPAC	Subtract P Register from accumulator	1	1	0 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0
SST	Store status register	1	1	0 1 1 1 1 0 0 0 1 ← D →
SUB	Subtract from accumulator with shift	1	1	0 0 0 1 ← S → I ← D →

(Continued)

INSTRUCTION SET SUMMARY (CONCLUDED)

MNEMONIC	DESCRIPTION	NO CYCLES	NO WORDS	OPCODE INSTRUCTION REGISTER
				15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
SUBC	Conditional subtract (for divide)	1	1	0 1 1 0 0 1 0 0 1 ← D →
SUBH	Subtract from high order accumulator bits	1	1	0 1 1 0 0 0 1 0 1 ← D →
SUBS	Subtract from accumulator with no sign extension	1	1	0 1 1 0 0 0 1 1 1 ← D →
TBLR	Table read from program memory to data RAM	3	1	0 1 1 0 0 1 1 1 1 ← D →
TBLW	Table write from data RAM to program memory	3	1	0 1 1 1 1 1 0 1 1 ← D →
XOR	Exclusive OR with accumulator	1	1	0 1 1 1 1 0 0 0 1 ← D →
ZAC	Zero accumulator	1	1	0 1 1 1 1 1 1 1 1 0 0 0 1 0 0 1
ZALH	Zero accumulator and load high-order bits	1	1	0 1 1 0 0 1 0 1 1 ← D →
ZALS	Zero accumulator and load low order bits with no sign extension	1	1	0 1 1 0 0 1 1 0 1 ← D →

CROSS-ASSEMBLER DIRECTIVES

AORG places a value in the location counter and defines the succeeding locations as absolute.

ABSOLUTE ORIGIN — AORG

Syntax: [<label>] ...AORG ...[<wd-exp>] ...[<comment>]

RORG places a value in the location counter and defines the succeeding locations as program relocatable.

RELOCATABLE ORIGIN — RORG

Syntax: [<label>] ...RORG ...[<exp>] ...[<comment>]

DORG places a value in the location counter and defines the succeeding locations as a dummy section. No object code is generated in a dummy section.

DUMMY ORIGIN — DORG

Syntax: [<label>] ...DORG ...<exp> ...[<comment>]

BSS first assigns the label, if present, and then advances the location counter by the value of the expression.

BLOCK STARTING WITH SYMBOL — BSS

Syntax: [<label>] ...BSS ...<wd-exp> ...[<comment>]

BES first advances the location counter by the value of the expression and then assigns the label, if present.

BLOCK ENDING WITH SYMBOL — BES

Syntax: [<label>] ...BES ...<wd-exp> ...[<comment>]

DSEG places a value in the location counter and defines succeeding locations as data relocatable.

DATA SEGMENT — DSEG

Syntax: [<label>] ...DSEG ...[<comment>]

DEND terminates a block of data relocatable code by placing a value in the location counter and defining succeeding locations as program relocatable.

DATA SEGMENT END — DEND

Syntax: [<label>] ...DEND ...[<comment>]

CSEG places a value in the location counter and defines succeeding locations as common relocatable (i.e., relocatable with respect to a common segment).

COMMON SEGMENT — CSEG

Syntax: [<label>] ...CSEG ...['<string>'][,<exp>] ...[<comment>]

CEND terminates the definition of a block of common relocatable code by placing a value in the location counter and defining succeeding locations as program relocatable.

COMMON SEGMENT END — CEND

Syntax: [<label>] ...CEND ...[<comment>]

PSEG places a value in the location counter and defines succeeding locations as program relocatable.

PROGRAM SEGMENT — PSEG

Syntax: [<label>] ...PSEG ...['<string>'][,][<exp>] ...[<comment>]

PEND places a value in the location counter and defines succeeding locations as program relocatable. (Since PEND properly appears only in program relocatable code, the relocation type of succeeding locations remains unchanged.)

PROGRAM SEGMENT END — PEND

Syntax: [<label>] ...PEND ...[<comment>]

OPTION selects several options for the assembler listing output.

OUTPUT OPTIONS — OPTION

Syntax: [<label>] ...OPTION ...<option>[,<option>] ... ...[<comment>]

(Continued)

ANNEXE - B -

CALCUL DU NOMBRE D'ECHANTILLONS

I_ PRINCIPE DE LA METHODE:

On borne la réponse impulsionnelle du filtre dans le temps sur un intervalle T° choisi de telle sorte que la fonction de transfert du filtre soit comprise dans un gabarit fixé par le CCITT.

Le problème à résoudre est alors de determiner le paramètre T°,dont la valeur doit etre optimale c'est à dire la plus petite possible tout en conservant une caractéristique fréquentielle H(f) dans le gabarit imposé.

La valeur de T° doit etre la plus petite possible ,dans le but d'avoir un nombre minimum d'échantillons,étant donné que le nombre d'échantillons N est determiné par

N =T°/Te avec Te=1/Fe , Fe:Fréquence d'échantillonnage

Ceci nous permet de stocker un nombre minimum d'échantillons dans un éspace mémoire le plus faible possible,et ce pour les différents types de codage.

Cette capacité mémoire sera determinée comme suite:
_Pour le débit 4800 bit/s,on a 4 fois N positions mémoire.
_ " " " 9600 " avec codage non-redondant,on a 16 fois N positions mémoire.
_ " " " 9600 " avec codage redondant ,on a 32 fois N positions mémoire.

Au total on a un nombre d'échantillons égal à M= 4N+16N+32N=52N.

La méthode la plus simple pour limiter dans le temps la réponse impulsionnelle h(t) du filtre consiste en une pondération de h(t) par une fonction rectangulaire,ce qui nous permet d'écrire :

$$\hat{h}(t) = h(t).\mathrm{rect}(t/T°)$$

Le procédé de traitement choisi est du type numérique ,ainsi h (t) échantillonnée s'écrira:

$$h_e(t) = \sum_{n=-\infty}^{+\infty} h(t).\delta(t-nTe)$$

$$\hat{h}_e(t) = \sum_{n=-\infty}^{+\infty} h(nTe).\mathrm{rect}((t-nTe)/T°)$$

La condition imposée au gabarit est:

$$-1dB \leqslant F\left[\hat{h}(t)-h(t)\right] \leqslant 1dB$$ (F:Transformée de Fourier.)

La transformée de Fourier discrète de la quantité $(\hat{h}'(t)-h(t))$ s'écrit:

$$F\left[\hat{h}'(t)-h(t)\right] = \sum_{n=-N/2}^{N/2}\left[h'(t)-h(t)\right].\exp(-j2\,nTe)$$

$$= F\left[\hat{h}'(t)\right] - F\left[h(t)\right] .$$

D'où $F[h(t)] - 1\ dB \leqslant F[\hat{h}'(t)] \leqslant F[h(t)] + 1\ dB$

La tolérance imposée est de 1dB pour la fréquence F1=3000Hz , et comme

$$F[h(t)]_{f=F1\ dB} = H(f=F1)_{dB} = -3dB$$

$$-4dB \leqslant F[\hat{h}'(t)]_{dB} \leqslant -2dB$$

Ou bien : $0.398 \leqslant F[\hat{h}'(t)] \leqslant 0.631$

D'où $0.398 \leqslant \sum_{n=-N/2}^{N/2} h(nTe).\exp(-j2\omega_1 F1 nTe) \leqslant 0.631$

Soit $A = \sum h(nTe)\exp(-j2\omega_1 F1 nTe) =$

$= \sum h(nTe).[\cos(n\omega_1 Te) - j\sin(n\omega_1 Te)]$, avec $\omega_1 = 2\pi F1$.

$= \sum h(nTe).\cos(n\omega_1 Te) - j\sum h(nTe).\sin(n\omega_1 Te)$

Soit $S1 = \sum h(nTe).\cos(n\omega_1 Te)$

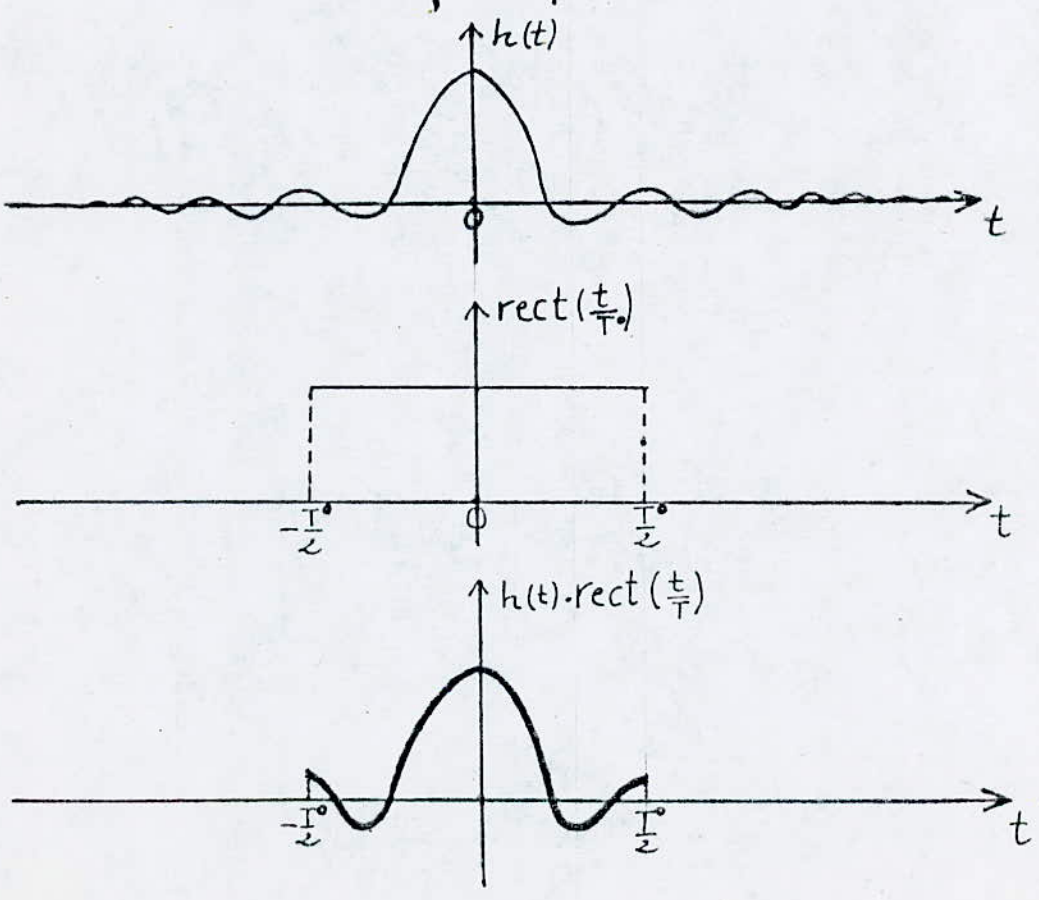
$S2 = \sum h(nTe).\sin(n\omega_1 Te)$

On pose $|A| = \sqrt{(S1)^2 + (S2)^2}$

$N = T°/Te$ d'où $T° = N.Te$.

Ainsi N sera determiné de manière à ce que:

$$0.398 \leqslant |A(N)| \leqslant 0.631$$



II_ PROGRAMME DE CALCUL DU NOMBRE D'ECHANTILLONS:

Le programme élaboré conformément à la méthode décrite,nous a permet de trouver un nombre minimum d'échantillons N=15 ,et ce pour une série de coéfficients d'arrondi dont nous avons choisi le coefficient d'arrondi optimum égal à 0.55 qui nous permet d'avoir une assez large bande de frequence pour éviter les distorsions de signaux.

```
5 DIM H(200)
6 DIM G(200)
8 INPUT "donnez la valeur de alpha R=";R
10 INPUT "donnez la valeur de N=";N
25 X(0)=0:Y(0)=0:Z(0)=0
30 S1=0:S2=0:S=0:S3=0:S4=0:S0=0
40 FE=14400:F=3000
50 L1=3.14*2400/FE:L2=6.28*1800/FE:L3=6.28*F/FE
55 H(0)=1
60 FOR I=-N/2 TO N/2
62 J=I+N/2
65 IF I=0 THEN 110
70 X=I*L1:Y=I*L2:Z=I*L3
80 H(J)=(SIN(X)*COS(Y)*COS(R*X))/(X*(1-.405696*(R*R)*(X*X)))
85 G(J)=H(J)*(.54+.46*COS(Z))
90 S1=S1+H(J)*COS(Z)
100 S2=S2+H(J)*SIN(Z)
102 S3=S3+G(J)*COS(Z)
104 S4=S4+G(J)*SIN(Z)
110 NEXT I
120 S=SQR(S1*S1+S2*S2)
122 S0=SQR(S3*S3+S4*S4)
130 IF S<.398 THEN 210
135 IF S>.631 THEN 210
140 PRINT "la valeur de N=";N;"est bonne pour la fonction REC"
141 IF S0<.398 THEN 214
142 IF S0>.631 THEN 214
150 PRINT "la valeur de N=";N;"est bonne pour la fonction HAMMING"
160 PRINT "voulez vous essayer une autre valeur? "
170 PRINT "si oui entrez 1 dans E sinon 0"
180 INPUT "E" ;E
190 IF E=1 THEN 10
200 GOTO 230
210 PRINT "la valeur de N=";N;"est mauvaise pour la fonction REC"
212 GOTO 141
214 PRINT "la valeur de N=";N;"est mauvaise pour la fonction HAMMING"
220 GOTO 160
230 END
```

```
donnez la valeur de alpha R=? 0.55
donnez la valeur de N=? 14
la valeur de N= 14 est bonne pour la fonction REC
la valeur de N= 14 est mauvaise pour la fonction HAMMING
voulez vous essayer une autre valeur?
si oui entrez 1 dans E sinon 0
```

ANNEXE - C -

CALCUL DES ÉCHANTILLONS

```
10 REM CALCUL DES COEFFICIENTS DU FILTRE
20 INPUT "LE COEFFICIENT D'ARRONDI ALPHA";R
30 INPUT "LA FREQUENCE D'ECHANTILLONNAGE";FE
40 INPUT "LE RAPPORT TE/TP";L
45 INPUT "LE NOMBRE D'ECHANTILLONS N=";N
50 M=N*L:S1=0:S2=0
60 DIM H(M+1)
70 H(1)=1
80 FOR I=0 TO M-1
90 T=(M-I)*(1/(L*FE))*2400
100 A=(3.14*T)*(1-4*(R*R)*(T*T))
110 B=(SIN(3.14*T))*(COS(R*(3.14*T)))*(COS(1.5*(3.14*T)))
120 X=M+1-I:H(X)=B/A
130 NEXT I
140 REM CALCUL DES ECHANTILLONS DE Y(T)=X(T)*H(T)
150 REM CAS DU DEBIT 4800 bit/s
160 DIM Y1(4,N+1)
165 DIM A(32)
167 DIM B(32)
170 A(1)=-3:A(2)=1:A(3)=3:A(4)=-1:B(1)=-1:B(2)=-3:B(3)=1:B(4)=3
180 FOR I=1 TO 4
190             Y1(I,1)=A(I)
200             FOR K=2 TO N
210                     FOR F=0 TO L*K
220                     T=F/(L*FE)
230                     X1=A(I)*(COS(11309*T))+B(I)*(SIN(11309*T))
240                     S1=S1+X1*H(F+1)
260                     NEXT F
270                     Y1(I,K)=S1:S1=0
280             NEXT K
290 NEXT I
300 REM CAS DU DEBIT 9600 bit/s NON-REDONDANT
310 DIM Y2(16,N+1)
320 A(1)=-1:A(2)=-3:A(3)=-1:A(4)=-3:A(5)=1:A(6)=1:A(7)=3:A(8)=3
322 A(9)=-1:A(10)=-1:A(11)=-3:A(12)=-3:A(13)=1:A(14)=3:A(15)=1
324 A(16)=3:B(1)=-1:B(2)=-1:B(3)=-3:B(4)=-3:B(5)=-1:B(6)=-3:B(7)=-1
326 B(8)=-3:B(9)=1:B(10)=3:B(11)=1:B(12)=3:B(13)=1:B(14)=1:B(15)=3
328 B(16)=3
360 FOR I=1 TO 16
370             Y2(I,1)=A(I)
380             FOR K=2 TO N
390             FOR F=0 TO K*L
400                     T=F/(L*FE)
```

```
410                         X2=A(I)*(COS(11309*T))+B(I)*(SIN(11309*T))
420                         S2=S2+X2*H(F+1)
440                         NEXT F
450                         Y2(I,K)=S2:S2=0
460               NEXT K
470 NEXT I
480 REM CAS DU DEBIT 9600 bit/s REDONDANT
490 DIM Y3(32,N+1)
500 A(1)=-4:A(2)=0:A(3)=0:A(4)=4:A(5)=4:A(6)=0:A(7)=0:A(8)=-4:A(9)=-2
502 A(10)=-2:A(11)=2:A(12)=2:A(13)=2:A(14)=2:A(15)=-2:A(16)=-2:A(17)=-3
503 A(18)=1:A(19)=-3:A(20)=1:A(21)=3:A(22)=-1:A(23)=3:A(24)=-1:A(25)=1
504 A(26)=-3:A(27)=1:A(28)=1:A(29)=-1:A(30)=3:A(31)=-1:A(32)=-1:B(1)=1
505 B(2)=-3:B(3)=1:B(4)=1:B(5)=-1:B(6)=3:B(7)=-1:B(8)=-1:B(9)=3:B(10)=-1
506 B(11)=3:B(12)=-1:B(13)=-3:B(14)=1:B(15)=-3:B(16)=1:B(17)=-2:B(18)=-2
507 B(19)=2:B(20)=2:B(21)=2:B(22)=2:B(23)=-2:B(24)=-2:B(25)=4:B(26)=0
508 B(27)=0:B(28)=-4:B(29)=-4:B(30)=0:B(31)=0:B(32)=4
540 FOR I=1 TO 32
550    Y3(I,1)=A(I)
560    FOR K=2 TO N
570          FOR F=0 TO L*K
580               T=F/(L*FE)
590               X3=A(I)*(COS(11309*T))+B(I)*(SIN(11309*T))
600               S3=S3+X3*H(F+1)
620           NEXT F
630      Y3(I,K)=S3:S3=0
640               NEXT K
650 NEXT I
670 REM AFFICHAGE DES RESULTATS
675 OPEN "O",#1,"C:"+"AMER"
680    PRINT #1,"LES VALEURS DE Y(I) AVEC D=4800 bit/s"
700 FOR J=1 TO N
710   PRINT #1,Y1(1,J),Y1(2,J),Y1(3,J),Y1(4,J)
730            NEXT J
740   PRINT #1,"    "
745    PRINT #1,"LES VALEURS DE Y(I) AVEC D=9600 bit/s N.REDONDANT"
750 FOR I=1 TO 13 STEP 4
760    FOR J=1 TO N
770    PRINT #1,Y2(I,J),Y2(I+1,J),Y2(I+2,J),Y2(I+3,J)
780    NEXT J
785  PRINT #1,"    "
790 NEXT I
795 PRINT #1,"    "
796 PRINT #1,"    "
798     PRINT #1,"LES VALEURS DE Y(I) AVEC D=9600 bit/s REDONDANT"
800 FOR I=1 TO 29 STEP 4
810     FOR J=1 TO N
820     PRINT #1,Y3(I,J),Y3(I+1,J),Y3(I+2,J),Y3(I+3,J)
830     NEXT J
835     PRINT #1,"   "
840 NEXT I
850 END
```

LES VALEURS DE Y(I) AVEC D=4800 bit/s

-3	1	3	-1
-36.65736	-7.675509	36.65736	7.675509
-38.37212	.4890282	38.37212	-.4890282
-48.43717	8.757473	48.43717	-8.757473
-55.37662	8.517061	55.37662	-8.517061
-56.14148	7.889046	56.14148	-7.889046
-55.90487	7.068068	55.90487	-7.068068
-53.37862	5.181566	53.37862	-5.181566
-50.24015	5.423495	50.24015	-5.423495
-49.51793	6.084401	49.51793	-6.084401
-49.42962	5.59074	49.42962	-5.59074
-49.21568	5.363717	49.21568	-5.363717
-49.42958	5.323698	49.42958	-5.323698
-49.53742	5.221207	49.53742	-5.221207
-49.55995	5.337287	49.55995	-5.337287

LES VALEURS DE Y(I) AVEC D=9600 bit/s N.REDONDANT

-1	-3	-1	-3
-16.19805	-36.65736	-28.13482	-48.59414
-15.25104	-38.37212	-22.63205	-45.75312
-17.62338	-48.43717	-22.05632	-52.87012
-20.44724	-55.37662	-26.41232	-61.3417
-20.87878	-56.14148	-27.37365	-62.63635
-20.94833	-55.90487	-27.88846	-62.845
-20.31514	-53.37862	-27.88192	-60.94541
-19.01136	-50.24015	-25.80528	-57.03407
-18.59029	-49.51793	-24.84323	-55.77086
-18.65369	-49.42962	-25.18517	-55.96108
-18.61352	-49.21568	-25.23843	-55.84057
-18.70708	-49.42958	-25.39878	-56.12127
-18.77071	-49.53742	-25.54547	-56.31217
-18.75651	-49.55995	-25.46612	-56.26956

1	1	3	3
4.261268	-7.675509	24.72058	12.7838
7.870035	.4890282	30.99111	23.6101
13.1904	8.757473	44.00423	39.57127
14.48215	8.517061	49.41154	43.44644
14.38391	7.889046	49.64662	43.15174
14.0082	7.068068	48.96474	42.0246
12.74835	5.181566	45.81184	38.24503
12.21742	5.423495	43.4462	36.65226
12.33734	6.084401	43.26498	37.01201
12.12222	5.59074	42.89813	36.36663
11.98862	5.363717	42.59076	35.96584
12.01539	5.323698	42.73787	36.04616
11.99596	5.221207	42.76264	35.98786
12.0469	5.337287	42.85031	36.14068

-1	-1	-3	-3
-4.261268	7.675509	-24.72058	-12.7838
-7.870035	-.4890282	-30.99111	-23.6101
-13.19042	-8.757473	-44.00423	-39.57127
-14.48215	-8.517061	-49.41154	-43.44644
-14.38391	-7.889046	-49.64662	-43.15174
-14.0082	-7.068068	-48.96474	-42.0246
-12.74835	-5.181566	-45.81184	-38.24503
-12.21742	-5.423495	-43.4462	-36.65226
-12.33734	-6.084401	-43.26498	-37.01201
-12.12222	-5.59074	-42.89813	-36.36663
-11.98862	-5.363717	-42.59076	-35.96584
-12.01539	-5.323698	-42.73787	-36.04616
-11.99596	-5.221207	-42.76264	-35.98786
-12.0469	-5.337287	-42.85031	-36.14068

1	3	1	3
16.19805	36.65736	28.13482	48.59414
15.25104	38.37212	22.63205	45.75312
17.62338	48.43717	22.05632	52.87012
20.44724	55.37662	26.41232	61.3417
20.87878	56.14148	27.37365	62.63635
20.94833	55.90487	27.88846	62.845
20.31514	53.37862	27.88192	60.94541
19.01136	50.24015	25.80528	57.03407
18.59029	49.51793	24.84323	55.77086
18.65369	49.42962	25.18517	55.96108
18.61352	49.21568	25.23843	55.84057
18.70708	49.42958	25.39878	56.12127
18.77071	49.53742	25.54547	56.31217
18.75651	49.55995	25.46612	56.26956

LES VALEURS DE Y(I) AVEC D=9600 bit/s REDONDANT

-4	0	0	4
-34.9502	-17.90517	5.968389	46.88701
-42.55164	-11.07151	3.690503	49.93265
-59.41111	-6.649425	2.216475	63.84407
-66.87621	-8.947628	2.982542	72.8413
-67.27794	-9.742298	3.247433	73.7728
-66.44297	-10.4102	3.470066	73.3831
-62.34353	-11.35017	3.783391	69.91033
-59.06055	-10.19089	3.396964	65.85449
-58.72875	-9.379411	3.12647	64.98169
-58.28605	-9.797212	3.265737	64.81751
-57.8918	-9.937351	3.312451	64.5167
-58.09908	-10.03754	3.345847	64.79076
-58.14595	-10.16213	3.387377	64.92071
-58.25199	-10.06442	3.354807	64.9616

4	0	0	-4
34.95023	17.90517	-5.968389	-46.88701
42.55164	11.[illegible]51	-3.690503	-49.93265
59.41111	6.649425	-2.216475	-63.84407
66.87621	8.[illegible]628	-2.982542	-72.8413
67.27794	9.[illegible]2298	-3.247433	-73.7728
66.44297	10.4102	-3.470066	-73.3831
62.34353	11.[illegible]017	-3.783391	-69.91033
59.06055	10.19089	-3.396964	-65.85449
58.72875	9.[illegible]9411	-3.12647	-64.98169
58.28605	9.[illegible]7212	-3.265737	-64.81751
57.8918	9.937351	-3.312451	-64.5167
58.09908	10.03754	-3.345847	-64.79076
58.14595	10.16213	-3.387377	-64.92071
58.25199	10.06442	-3.354807	-64.9616

-2	-2	2	2
-2.554147	-26.4277	38.36448	14.49093
-12.04957	-26.81158	34.19259	19.43058
-24.1643	-33.03028	37.46322	28.59732
-25.98175	-37.91193	43.87701	31.94684
-25.52039	-38.51013	45.00499	32.01526
-24.54633	-38.4266	45.36673	31.48647
-21.7133	-36.84687	44.41365	29.28009
-21.03788	-34.62573	41.41966	27.83181
-21.54821	-34.05409	40.30703	27.80116
-20.97869	-34.04163	40.57311	27.51017
-20.66479	-33.91458	40.53949	27.28969
-20.68494	-34.06831	40.76001	27.37663
-20.60455	-34.15404	40.9288	27.3793
-20.739	-34.1582	40.86783	27.44861

2	2	-2	-2
2.554147	26.4277	-38.36448	-14.49093
12.04957	26.81158	-34.19259	-19.43058
24.16437	33.03028	-37.46322	-28.59732
25.98175	37.91193	-43.87701	-31.94684
25.52039	38.51013	-45.00499	-32.01526
24.54633	38.4266	-45.36673	-31.48647
21.7133	36.84687	-44.41365	-29.28009
21.03788	34.62573	-41.41966	-27.83181
21.54821	34.05409	-40.30703	-27.80116
20.97869	34.04163	-40.57311	-27.51017
20.66479	33.91458	-40.53949	-27.28969
20.68494	34.06831	-40.76001	-27.37663
20.60455	34.15404	-40.9288	-27.3793
20.739	34.1582	-40.86783	-27.44861

-3	1	-3	1
-42.62575	-1.70712	-18.75219	22.16643
-42.06262	4.179531	-27.30061	18.94155
-50.65364	10.9739	-41.78775	19.83985
-58.35915	11.4996	-46.42899	23.42978
-59.3889	11.13648	-46.39918	24.12621
-59.37493	10.538	-45.49466	24.4184
-57.16199	8.964956	-42.02843	24.09852
-53.63706	8.82045	-40.04923	22.40831
-52.64436	9.21087	-40.13849	21.71675
-52.6953	8.85647	-39.63237	21.91942
-52.52808	8.676169	-39.27829	21.92596
-52.77537	8.669544	-39.392	22.05292
-52.92473	8.608584	-39.37525	22.15809
-52.91469	8.692093	-39.4955	22.11131

3	-1	3	-1
42.62575	1.707121	18.75219	-22.16643
42.06262	-4.179531	27.30061	-18.94155
50.65364	-10.97395	41.78775	-19.83985
58.35915	-11.4996	46.42899	-23.42978
59.3889	-11.13648	46.39918	-24.12621
59.37493	-10.53813	45.49466	-24.4184
57.16199	-8.964956	42.02843	-24.09852
53.63706	-8.820458	40.04923	-22.40831
52.64436	-9.210874	40.13849	-21.71675
52.6953	-8.856479	39.63237	-21.91942
52.52808	-8.676169	39.27829	-21.92596
52.77537	-8.669544	39.392	-22.05292
52.92473	-8.608584	39.37525	-22.15809
52.91469	-8.692093	39.4955	-22.11131

1	-3	1	1
34.10321	-30.68897	10.22966	-13.6439
26.32255	-34.68162	11.56054	-3.201477
24.2728	-46.2207	15.4069	6.540996
29.39486	-52.39408	17.46469	5.534517
30.62108	-52.89405	17.63135	4.641613
31.35853	-52.43481	17.47827	3.597999
31.66531	-49.59522	16.53175	1.398173
29.20225	-46.84317	15.61439	2.026528
27.9697	-46.39144	15.46382	2.957929
28.45091	-46.16385	15.38796	2.325001
28.55088	-45.9032	15.30108	2.051265
28.74463	-46.0837	15.36124	1.977851
28.93286	-46.15002	15.38334	1.833831
28.82094	-46.20512	15.40171	1.982481

-1	3	-1	-1
-34.10321	30.68897	-10.22966	13.6439
-26.32255	34.68162	-11.56054	3.201477
-24.2728	46.2207	-15.4069	-6.540996
-29.39486	52.39408	-17.46469	-5.534517
-30.62108	52.89405	-17.63135	-4.641613
-31.35853	52.43481	-17.47827	-3.597999
-31.66531	49.59522	-16.53175	-1.398173
-29.20225	46.84317	-15.61439	-2.026528
-27.9697	46.39144	-15.46382	-2.957929
-28.45091	46.16385	-15.38796	-2.325001
-28.55088	45.9032	-15.30108	-2.051265
-28.7446	46.0837	-15.3	
-28.93286	46.15002	-15.38334	-1.833831
-28.82094	46.20512	-15.40171	-1.982481

ANNEXE -D-

```
*************************************************************************
*************** SIMULATION D'UN EMETTEUR MICRO-PROGRAMME ****************
*************************************************************************
*faire un choix de débit(en bit/s):4800-9600(non-red)-9600(red)
*la selection de débit se fera à partir du port PA2:
*4800bit/s :PA2=0 ,9600bit/s N-R :PA2=01 ,9600bit/s R :PA2=11
*DEFINITION DES CONSTANTES:
****************************
        AORG 0
X0      EQU  0
X1      EQU  1
X2      EQU  2
X3      EQU  3
X4      EQU  4
X5      EQU  5
X6      EQU  6
X7      EQU  7
X8      EQU  8
X9      EQU  9
X10     EQU  10
X11     EQU  11
X12     EQU  12
X13     EQU  13
X14     EQU  14
X15     EQU  15
X16     EQU  16
X17     EQU  17
X18     EQU  18
X19     EQU  19
X20     EQU  20
X21     EQU  21
X22     EQU  22
X23     EQU  23
Y1      EQU  24
Y2      EQU  25
M1      EQU  26
ONE     EQU  27
D       EQU  28
D1      EQU  29
D2      EQU  30
Y0      EQU  31
Q3      EQU  32
Q4      EQU  33
A       EQU  34
X       EQU  35
Y       EQU  36
C       EQU  37
B       EQU  38
Q1      EQU  39
Q2      EQU  40
Z       EQU  41
        ZAC
        SACL Y0
        SACL X
        SACL Y
        SACL Y1
        SACL Y2
        LACK 1
        SACL ONE
*CHOIX DU TYPE DE CODAGE:
```

```
*************************
        IN D2,PA2
*****************************
**  BROUILLAGE DES DONNÉES **
*****************************
START   IN D,PA0
        LAC X23
        DMOV X22
        DMOV X21
        DMOV X20
        DMOV X19
        LT X18
        DMOV X18
        MPYK 1
        SACL M1
        ZAC
        LTD X17
        XOR  M1
        LARK AR0,16
        LARP 0
        MPYK 0
LOOP1   DMOV *
        BANZ LOOP1

        LAC D2,0            *choix du type de codage
        BZ D4800
        SUB ONE,0
        BZ D9600N
        B D9600R
*************************
** DEBIT  : 4800 bit/s  **
*************************
D4800   LARK AR0,7
LOOP2   CALL SPROG
        LAC Y1,1
        ADD Y2
        SACL Y1
        LACK 1171
        ADDS Y1

        LARK AR1,14
        LARP 1

LOOP3   TBLR D1
        OUT D1,1
        ADDS ONE
        BANZ LOOP3        *transmission des données traitées

        LARP 0
        BANZ LOOP2

TEMPO1  LARK AR1,3771     *une temporisation
        LARP 1
        BANZ TEMPO1

        B START

*************************************************
** DEBIT :9600 bit/s  avec un Code Non-Redondant **
*************************************************
D9600N  LARK AR0,3
LOOP4   CALL SPROG
```

```
        LAC X0,1
        SACH Q3
        LAC X0,2
        SACH Q4
        SACL X0

        LACK 1
        AND Q3
        SACL Q3
        LACK 1
        AND  Q4
        SACL Q4

        LAC Y1,3
        SACL A
        LAC Y2,2
        ADDS A
        SACL A
        LAC Q3,1
        ADDS A
        ADDS Q4
        SACL A
        LACK 1231       *calcul des adresses des positions mémoires à point
        ADDS A

        LARK AR1,14
        LARP 1

LOOP5   TBLR D1
        OUT D1,1
        ADDS ONE
        BANZ LOOP5      *transmission des données traitées.

        LARP 0
        BANZ LOOP4
TEMP02  LARK AR1,1857    *une temporisation
        LARP 1
        BANZ TEMP02

        B START
*********************************************
** DEBIT 9600 bit/s avec un Code Redondant **
*********************************************
D9600R  LARK AR0,3
LOOP6   LAC X0,1
        SACH Q1
        LAC X0,2
        SACH Q2
        LAC X0,3
        SACH Q3
        LAC X0,4
        SACH Q4
        SACL X0

        LAC Q1,0
        XOR Y1
        SACL A          *(A)=[Q1(n)] ou-ex [Y1(n-1)]

        LAC Y2
        XOR Q2
        SACL Y2
        LACK 1
```

```
        AND Y1
        AND Q1
        XOR Y2
        SACL Y2      *Y2(n)=[Q1(n) and Y1(n-1)]ou-ex[Q2(n) ou-ex Y2(n-1)

        LACK 1                                                          -
        AND A
        SACL Y1      *Y1(n)=[Q1(n)] ou-ex [Y1(n-1)], "réstitution et mas

        XOR Y2
        XOR YO
        SACL Z       *Z(n)= [YO(n-1)] ou-ex [Y1(n) ou-ex Y2(n)].

        XOR Y2
        SACL A
        LAC X,O
        AND YO
        XOR A
        SACL X       *X(n)=[Y2(n)] ou-ex [Z(n-1) ou-ex [X(n-1) and YO(n-1

        LAC YO,O
        AND Y1
        XOR X
        SACL YO      *YO(n)=[X(n-1)] ou-ex [YO(n-1) and Y1(n-1)]

        LACK 1
        AND Q3
        SACL Q3
        LACK 1
        AND Q4        *masquage des Q3 et Q4.
        ADD Q3,1
        ADD Y2,2
        ADD Y1,3
        ADD YO,4
        SACL A
        LACK 1471
        ADDS A

        LARK AR1,14
        LARP 1
LOOP7   TBLR D1
        OUT D1,1
        ADDS ONE
        BANZ LOOP7

        LARP 0
        BANZ LOOP6

TEMPO3  LARK AR1,1888  *temporisation
        LARP  1
        BANZ TEMPO3

        B START
        END


XO      EQU 0
Q1      EQU 39
Q2      EQU 40
Y1      EQU 24
A       EQU 34
Y2      EQU 25
```

```
B       EQU 38
C       EQU 37
D       EQU 28

*****************************************************************
* sous-programme de calcul des expressions de Y1(n) et Y2(n) *
*****************************************************************
SPROG   LAC XO,1
        SACH Q1
        LAC XO,2
        SACH Q2
        SACL XO

        LAC Q2
        OR Y1
        SACL A
        LACK 1
        XOR A
        SACL A          *(A)= [Y1(n-1) and Q2(n)]=[Y1(n-1) or Q2(2)]

        LACK 1
        XOR Y2
        AND Q2
        OR A
        AND Q1
        SACL B          *(B)=[Q1(n) and [(A) or [Q2(n) and Y2(n)]]]

        LAC Q2
        AND Y1
        SACL C          *(C)=[Q2(n) and Y1(n)]

        LACK 1
        XOR Q2
        AND Y2
        OR C
        SACL D          *(D)=(C) or [Q2(n) and Y2(n)]
        LACK 1
        XOR Q1
        OR D
        OR B
        SACL Y1
        LACK 1          *élimination de l'éxtention de signe.
        AND Y1
        SACL Y1         *Y1(n)=éxpression n°(1)

        LAC Q2
        OR Y2
        SACL B
        LACK 1
        XOR B           *(acc)=[Q2(n) and Y2(n)]=[Q2(n) or Y2(n)]
        OR C
        AND Q1
        SACL B          *(B)=Q1(n) and [[Q2(n) and Y1(n)] or [Q2(n) and Y2
        LAC Y2
        AND Q2
        OR A            *(acc)=Q1(n) and [[Q2( ) and Y1(n)] or [Y2(n) and
        SACL A
        LACK 1
        XOR Q1
        AND A

        OR B
```

```
SACL  Y2          *élimination de l extention de signe
LACK  1
AND   Y2
SACL  Y2          *Y2(n)=éxpressio  n°(2)

RET
```

BIBLIOGRAGHIE

[1]-_M STEIN - LES MODEMS pour transmission de données
MASSON 1987

[2]- J.C.BIC , D.DUPONTEIL , J.C.IMBEAU
- ELEMENTS DE COMMUNICATIONS NUMERIQUES
DUNOD 1986

[3]- C.MACCHI - LA TELEINFORMATIQUE
DUNOD 1987

[4]- A.J.VETERBI et J.K.OMURA
- PRINCIPES DES COMMUNICATIONS NUMERIQUES
DUNOD 1982

[5]- M.BELLANGER - TRAITEMENT NUMERIQUE DU SIGNAL
MASSON 1987

[6]- KUNT - TRAITEMENT NUMERIQUE DU SIGNAL

[7-] TEXAS INSTRUMENTS
- TMS 320-10 USER'S GUIDE 1983

[8]- TEXAS INSTRUMENTS
- TMS 320-10 ASSEMBLY LANGUAGE PROGRAMMER'S GUIDE

[9]- RECOMMANDATIONS V32 DU CCITT 1984