

DIXIEME COLLOQUE SUR LE TRAITEMENT DU SIGNAL ET SES APPLICATIONS



NICE du 20 au 24 MAI 1985

OUTILS DE MISE AU POINT DE MACHINES POUR LE TRAITEMENT DU SIGNAL

ETUDE D'UN CAS : REALISATION D'UN MODEM 4800 bps

Yves SOREL - Philippe WOLF

INRIA - Domaine de Voluceau - B.P. 105 - Rocquencourt - 78153 Le Chesnay Cedex

RESUME

Le besoin croissant d'outils de mise au point de machines pour le traitement temps réel du signal, nous a amené à étudier et réaliser un modem rapide 4800 bps à base de microprocesseurs μ PD7720 de NEC. Cette application, de complexité raisonnable, sature aujourd'hui, les structures monoprocesseur et rassemble une gamme assez large de fonctions typiques du domaine visé.

Nous présentons successivement, une description fonctionnelle d'un modem 4800 bps conforme à l'avis V 27ter de CCITT, les outils de mise au point utilisés, la structure matérielle de la maquette réalisée et quelques résultats significatifs de son fonctionnement. Cette réalisation s'inscrit dans le cadre général de développement d'une chaîne de CAO adaptée au traitement temps réel du signal.

La phase de simulation autant au niveau des algorithmes que des machines a accéléré de façon significative l'obtention de la maquette modem. La chaîne de CAO s'articulera donc autour de deux axes :

- du côté algorithmique, nous développons l'utilisation de langages adaptés à la description et à la simulation d'applications de traitement du signal. Le langage SIGNAL développé à l'IRISA s'est avéré très utile pour spécifier de manière concise le modem.

- du côté des processeurs, nous continuons d'utiliser le langage ISPS pour la spécification d'architectures internes. Les nouvelles applications visées, par exemple l'annulation d'écho acoustique, nécessitent la mise en oeuvre de structure multiprocesseurs.

Les objectifs principaux visés actuellement sont l'accélération des simulateurs et la mise en oeuvre de fonctions permettant de valider automatiquement le choix d'une répartition de l'algorithme entre plusieurs microprocesseurs pour satisfaire la contrainte temps réel.

SUMMARY

The need for software development tools for Signal Processors is growing up fast. Therefore we realized a 4800 bps all-digital modem on μ PD7720's NEC micro-processors.

This application saturates today's monoprocesor capabilities and exhibits typical signal processing functions.

We present a functional description of the 4800 bps modem according to the CCITT recommendations V27ter, the software development tools, the hardware prototype and some experimental results.

This work will lay the grounds for a future signal processing oriented CAO Workstation.

The simulation of the algorithm and the processors speeded up the realization of the modem. The CAO environment will center on two points :

- we develop the use of a data flow oriented real time language "SIGNAL" written at IRISA. It is suited to the expression and simulation of signal or image processing algorithms.

- we proceed with ISPS for the internal specification of hardware architectures. New applications, as acoustical echo cancellation require multiprocessor nets simulations.

Our main goal is now, the improvement of the simulator speed and the automatic partitionary of the algorithms in order to allow real time implementation on several microprocessors.



I - DESCRIPTION FONCTIONNELLE DU MODEM 4800 bps

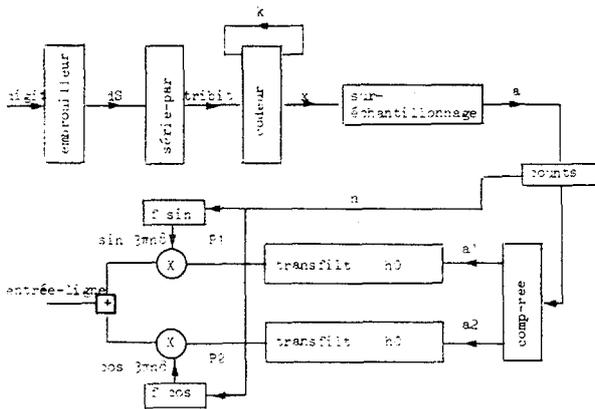
Le modem synchrone 4800 bps est composé des deux éléments principaux :

- un modulateur qui assure l'émission des données binaires de l'équipement source sur le canal de transmission. Après embrouillage et codage des bits émission a lieu la modulation de phase différentielle (PSK) octovalente, avec fonctionnement synchrone caractéristique de la transmission à 4800 bps.

- un démodulateur dont le rôle est de retrouver les bits émis à partir du signal de ligne reçu. Il identifie les caractéristiques de la ligne pour reconstituer par filtrage inverse le signal émis (rôle de l'égalisation numérique). Il retrouve le rythme d'émission des bits pour assurer le synchronisme entre émetteur et récepteur. Un désembrouilleur permet de régénérer la séquence des bits émis.

1. Description fonctionnelle du modulateur

Le schéma de principe du modulateur est [Gerwen77]



Sa forme particulière est adaptée à la traduction de l'algorithme en langage SIGNAL (voir II).

Le train de données $\{D_s(nT_d)\}_{n=0,\infty}$ (à transmettre à la vitesse $f_d = 1/T_d = 4800$ Hz après embrouillage des données $\{D_1(nT_d)\}_{n=0,\infty}$, par un dispositif décrit par l'avis V27) est divisé en groupes de trois bits consécutifs ce qui assure une vitesse de modulation $1/T_b = 1/\Delta = 1600$ Hz.

Chaque tribit est codé sous la forme d'un changement de phase K_0 (tribit) par rapport à la phase du tribit qui le précède immédiatement et constitue un élément $x(nT_b) = x_1(nT_b) + i x_2(nT_b)$ de signal à transmettre ;

$$x(nT_b) = \exp(j K_n \pi/4) \text{ où } K_n = K_{n+1} + K_0$$

est la phase absolue du symbole transmis.

Pour moduler les deux porteuses $\sin(2\pi f_p t)$ et $\cos(2\pi f_p t)$ avec $f_p = 1/T_p = 1800$ Hz fréquence porteuse, il faut filtrer les signaux $x_1(t)$ et $x_2(t)$ qui ont une largeur de bande théoriquement infinie par un filtre de mise en forme h_0 en cosinus surélevé équiréparti entre émetteur et récepteur de réponse impulsionnelle :

$$h_0(t) = \frac{1}{f_e} \frac{8f_1 t \cos B - \sin A}{e^{\pi t [1 - (8f_1 t)^2]}} \text{ avec } \begin{cases} A = 2\pi f_1 t - \pi t/\Delta \\ B = 2\pi f_1 t + \pi t/\Delta \end{cases}$$

avec $f_1 = \frac{1}{2\Delta}$ et $f_e = 9600$ Hz fréquence d'échantillonnage.

$$\begin{aligned} \text{Si l'on note } a(nT_e) &= a_1(nT_e) + ja_2(nT_e) \\ &= \begin{cases} x(n/6 T_b) & \text{pour } n = 0, \pm 6, \pm 12, \dots \\ 0 & \text{autrement} \end{cases} \end{aligned}$$

le signal complexe $p(t) = \sum_{k=-\infty}^{+\infty} a(kT_b) \times h(t - kT_b) =$

$p_1(t) + jp_2(t)$ est modulé par les porteuses en quadrature $\sin(2\pi f_p t)$ et $\cos(2\pi f_p t)$. Les valeurs échantillonnées du signal $e(t)$ émis par le modulateur sont

$$e[n] = p_1[n] \times \sin(3\pi n/8) + p_2[n] \times \cos(3\pi n/8).$$

2. Description du canal de transmission

La ligne téléphonique peut être représentée par une fonction de transfert complexe :

$$\Lambda(f) = |\Lambda(f)| e^{j\Phi(f)} \text{ où } |\Lambda(f)| \text{ représente l'affaiblissement du canal et } \Phi(f) \text{ sa phase.}$$

On modélise donc le canal de transmission par un filtre linéaire $l(t)$:

$$l(t) = \text{Re} \left(\int_{-\infty}^{+\infty} \Lambda(f) \exp(2j\pi f t) df \right)$$

De plus s'introduisent des écarts de phase qui peuvent être décrits comme un bruit multiplicatif complexe : $m(t) = \exp(-i\theta(t))$ où $\theta(t)$ est de la forme :

$$\theta(t) = \theta_0 + \theta_g(t) + \theta_d(t) \quad \theta_0 \text{ est une constante}$$

$$\text{avec : } \theta_g = \alpha \sin(2\pi f_0 t)$$

où : $f_0 = 50, 10$ Hz est un multiple de la fréquence secteur. θ_g représente la gigue de phase due à l'utilisation de câbles à courant porteur ; 2α est l'amplitude crête à crête de la gigue de phase.

$$\theta_d(t) = 2\pi f_d t$$

représente la dérive en fréquence de la ligne. Il s'y rajoute enfin un bruit gaussien, stationnaire.

3. Description fonctionnelle du démodulateur

Le récepteur réalise trois fonctions essentielles :

- récupération du rythme des symboles émis. Des solutions analogiques, bien maîtrisées, sont couramment employées mais, plus récemment, on a apporté des solutions numériques à ce problème [Godard 78].

- égalisation adaptative du canal pour supprimer l'interférence intersymbole. Cela peut être réalisé par des filtres transversaux, par des filtres récursifs en utilisant la robustesse des algorithmes de gradient stochastique [Benveniste 79].

- correction de phase pour maîtriser les effets de la dérive de fréquence et de la gigue de phase [Levy 79].

Le schéma de principe est :



1. Description en SIGNAL de l'algorithme MODEM

SIGNAL est un langage développé à l'IRISA.

Les caractéristiques générales du langage sont inspirées des principes flots de données : un programme SIGNAL décrit un réseau d'opérateurs (arithmétiques ou temporels) interconnectés à travers des ports orientés sur lesquels circulent des signaux. Ces réseaux sont construits progressivement à l'aide d'expressions sur filtres permettant d'interconnecter des ports selon un principe s'appuyant sur une identité de leurs noms. Différents opérateurs sont proposés : des opérateurs binaires permettent des interconnexions effectives selon divers modes, des opérateurs unaires agissent sur les noms de ces ports.

Nous avons décrit séparément le modulateur et le récepteur du modem 4800 bps en essayant de faire apparaître les structures des schémas I1 et I3. Le langage SIGNAL permet de transcrire scrupuleusement les flots des signaux, les parallélismes et rebouclages ainsi que les mécanismes de synchronisation des divers signaux. Un calcul d'horloge implicite est mené à partir des relations temporelles entre les divers signaux manipulés. Cela détermine l'horloge la plus fine qui constitue la référence du simulateur généré. Ce dernier dans la première version V00 du langage est constitué par du code FORTRAN exécutable.

On trouvera en Annexe 1 des extraits des deux programmes. On remarquera la concision d'écriture.

2. Simulation en ISPS de l'implantation sur microprocesseur du MODEM

Le langage ISPS a permis d'écrire des simulateurs des microprocesseurs de traitement du signal (voir [Sorel 83]) susceptibles d'accueillir les algorithmes du modem.

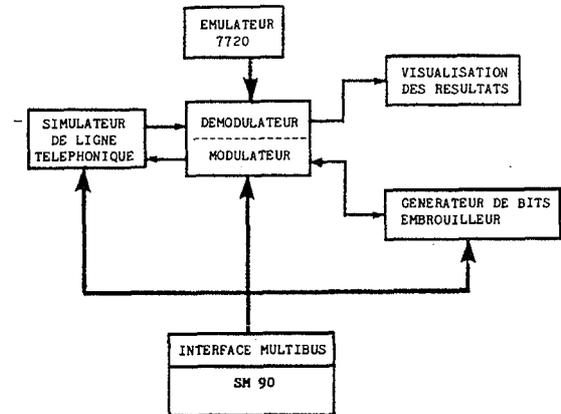
A partir de ces descriptions ISPS on peut simuler leur fonctionnement au niveau matériel. L'analyse fine des algorithmes peut être menée grâce à la prise en compte des contraintes matérielles. On peut notamment, au vu de simulations sur des processeurs distincts, choisir la structure la mieux adaptée pour l'étape ultime qui est la construction de la maquette. La simulation ISPS consiste à :

- traduire les algorithmes en langage assembleur. Cela se fait actuellement "à la main" car on peut difficilement envisager un traducteur efficace pour une machine donnée.
- mettre au point et valider le fonctionnement statique de l'algorithme et si possible le fonctionnement dynamique sur des données réalistes.

Les performances du simulateur sur un ordinateur DEC10 sont les suivantes : Pour 1658 instructions M7720 exécutées pendant un intervalle d'échantillonnage, par l'algorithme de démodulation on a un temps de simulation de 58s ce qui correspond à un temps moyen d'exécution de 28 instructions par seconde. La lenteur relative d'une telle simulation est compensée par la souplesse d'utilisation (modification d'architecture, utilisation du moniteur de mise au point). C'est aussi un puissant outil d'évaluation de performances. Des paramètres du type de ceux définis dans [Sorel 83] ont été calculés. Une simulation au niveau plus fin de la micro-machine permet de rendre compte du comportement externe des microprocesseurs. On décrit ainsi des réseaux de processeurs

III - DESCRIPTION DE LA MAQUETTE

La structure générale choisie pour la maquette et son environnement est celle de la figure ci-dessous.



La maquette électrique construite pour tester le modem 4800 bps se compose de deux cartes électroniques (voir annexe 2).

Sur la première a été implanté un simulateur de ligne téléphonique que l'on peut paramétrer par l'intermédiaire d'une interface Multibus reliée au miniordinateur SM90. Ceci permet d'introduire les 64 coefficients représentant la réponse impulsionnelle correspondant à un gabarit de ligne donné. De même on peut introduire du bruit pseudo gaussien et une modulation de phase sinusoidale à 50 ou 100 Hz représentant le phénomène de gigue introduit par le secteur. Enfin on peut simuler l'introduction d'un décalage en fréquence induit par les multiplexages successifs qui sont effectués sur les signaux transitant sur les lignes téléphoniques. Le réglage des paramètres décrits ci-dessus conduit à un modèle de ligne très précis et évolutif.

La seconde carte réalise la fonction modem proprement dite. Elle ne comprend que le minimum des fonctions de signalisation nécessaires.

La partie modulateur reçoit les bits à transmettre à 4800 Hz via l'interface Multibus. Ils sont générés par une carte construite autour d'un microprocesseur Intel 8085. La séquence des bits est programmable. On peut émettre la séquence de conditionnement pour le mode apprentissage de l'égaliseur correspondant à l'avis V27 ter. On la fait suivre d'une séquence pseudoaléatoire de très longue période pour le mode autoadaptatif de l'égaliseur. Ces bits sont introduits dans un circuit NEC 7720 sur lequel est implanté l'algorithme de modulation.

La partie démodulateur est implantée sur un émulateur du NEC 7720 afin de faciliter la mise au point des algorithmes. Il est simple de changer la longueur des filtres transversaux et récursifs de l'égaliseur et de modifier le pas du gradient puisque l'on a accès directement à la RAM et la ROM par l'intermédiaire du moniteur de l'émulateur. On peut visualiser sur oscilloscope par l'intermédiaire de deux convertisseurs X et Y, la phase retrouvée donc le triplet émis, l'erreur quadratique, ou toute autre variable. Une sortie logique Z synchronise les sorties X et Y lors des transferts sur un système d'acquisition rapide en vue de mémoriser ces résultats.

Afin de séparer les problèmes lors des tests, la maquette a été conçue de façon à pouvoir fonctionner soit en mode numérique soit en mode analogique.

Le premier mode implique une horloge de référence

unique pour tous les composants séquentiels des deux cartes. Les échanges entre modulateur ligne et démodulateur sont synchronisés par une horloge commune de fréquence égale à 9600 Hz et un multiple de cette fréquence égal à 32 x 9600 Hz.

On a pu tester plusieurs algorithmes d'égalisation pour le démodulateur, en s'affranchissant des défauts de synchronisation symbole introduits par les liaisons analogiques entre ligne et modem.

En situation réelle, le modulateur et le démodulateur sont séparés par la liaison analogique que constitue la ligne physique et leurs horloges respectives ne sont pas synchrones, ceci correspond au mode analogique. Le démodulateur gère sa propre horloge symbole par l'intermédiaire d'une boucle à verrouillage de phase. Un algorithme ajuste l'horloge symbole en estimant le déphasage à partir des symboles qu'il reçoit.

IV - RESULTATS

On trace trois sortes de courbes : une constellation représentant un nuage des points $S[k]$ correspondant aux symboles retrouvés, l'évolution de l'erreur $er[k]$, la phase estimée en sortie du correcteur de phase $\theta[k]$.

La figure 1 présente dans un cas typique pour la ligne M1020 (voir avis V27 ter du CCITT) la constellation sur 300 symboles. Les 150 premiers sont les derniers de la séquence d'apprentissage de 1074 symboles et les 150 suivants représentent une émission de bits aléatoires.

La figure 2 montre la courbe d'erreur dans les mêmes conditions (convergence de l'algorithme).

La figure 3 représente le suivi du correcteur de phase qui doit compenser la gigue de phase d'un angle crête à crête de 30° et la dérive en fréquence de 12 Hz. Les sauts correspondent au passage à la valeur $\pm\pi$ de l'angle de correction.

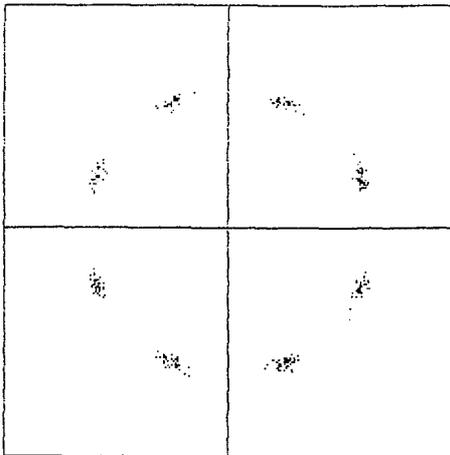


Figure 1 : Constellation
ligne 1020, S/B 25 db
gigue 15°, dérive 7 Hz

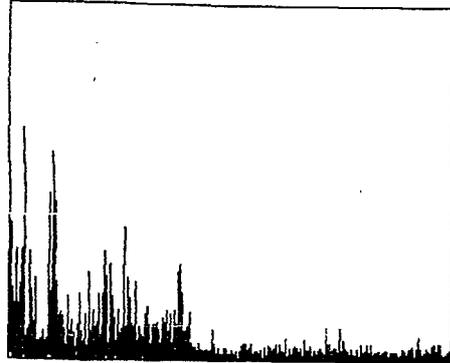


Figure 2 : Erreur
Mêmes conditions

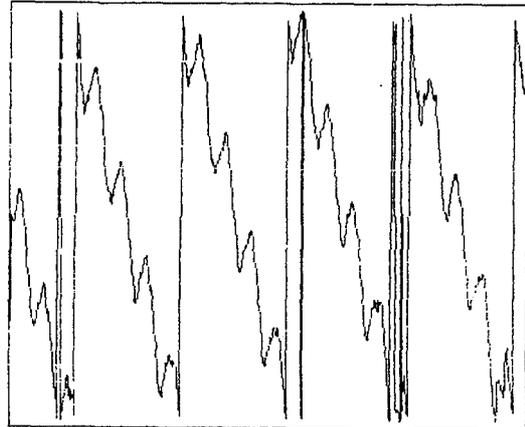


Figure 3 : Suivi du correcteur de phase, ligne 1020,
S/B 20 db, gigue 30°

CONCLUSION

Les outils de mise au point décrits précédemment ont permis d'accélérer la réalisation du modem 4800 bps. Ceci est une première approche vers un système de CAO pour les machines de traitement du signal.

BIBLIOGRAPHIE

[Benveniste 79] A. Benveniste, M. Goursat : "Robustesse des égaliseurs auto-adaptatifs et récursifs", Colloque Grets, 1979.

[Gerwen 77] P.J. Van Gerwen & al. : "Microprocessor Implementation of High-Speed Data Modems", IEEE Trans. of Com., Vol. COM-25, n° 2, Février 1977.

[Godard 78] D.N. Godard : "Passband Timing Recovery in All-Digital Modem Receiver", IEEE Trans. of Com., Vol. COM-26, n° 5, Mai 1978.

[Le Guernic 85] P. Le Guernic, A. Benveniste, P. Bournai T. Gautier : "SIGNAL : a data flow oriented language for signal processing", IRISA, Publication n° 246, Janvier 1985.

[Levy 79] M. Levy, C. Poinas : "Adaptive Phase Correctors for Data Transmission Receivers", Proc. Int. Conf. Commun., Boston, Juin 1979.



[Sorel 83] Y. Sorel, P. Wolf : "Evaluation d'architectures de microprocesseurs de traitement du signal", Colloque GRETSI, pp. 783-789, Nice 1983.

Annexe 1

```

MODUL ( ( ? logical DIGIT, FE ! real ENTREELIGNE )
  par integer L; (L)real H0
  synchro H: (FE); (DIGIT)=(2,0)H )

- ( ( EMBROUILLEUR ? ENTREE: DIGIT ! SORTIE: DS ;
  SERIE PAR ? ENTREE: DS ! SORTIE: TRIBIT ;
  CODEUR ? ENTREE: TRIBIT ! SORTIE: X @ K ;
  X1:=REAL(X) ; X2:=AIMAG(X) ) ;
  ( A has X synchro FE ; CPT6 counts A ;
  if MOD(CPT6,6)=0 then A:=A else A:=0 f1 ;
  A1:=REAL(A) ; A2:=AIMAG(A) ;
  N counts A ; N1:=MOD(N-1,16) ) ;
  ( TRANSFILT(L,H0) ? EE:A1 ! S:P1 &
  TRANSFILT(L,H0) ? EE:A2 ! S:P2 ) ;
  TETA:=(3*3.1416 e 0*N1)/8 ;
  ( COS3PIN8:=COS(TETA) & SIN3PIN8:=SIN(TETA) ) ;
  ENTREELIGNE:=(P1*SIN3PIN8)+(P2*COS3PIN8)
  ) !! ENTREELIGNE

where
logical DS;
(3)logical TRIBIT;
integer N,N1,CPT6;
real TETA,SIN3PIN8,COS3PIN8;
complex X,A;
real X1,X2,A1,A2,P1,P2

process
EMBROUILLEUR ( ( ? logical ENTREE ! logical SORTIE ) )
=..... end;

SERIE_PAR ( ( ? logical ENTREE ! (3)logical SORTIE ) )
=..... end;

CODEUR ( ( ? (3)logical ENTREE; integer K
! complex SORTIE; integer K ) )
=..... end;

TRANSFILT ( ( ? real EE ! real S )
  par integer N; (N)real H )
=..... end

end
-----
DEMODU ( ( ? real SORTIE_LIGNE ! complex A )
  par ..... )=

(BVP ? ENTREE: SORTIE_LIGNE ! SORTIE: EE ;
  if DEBUT then
  (( TRANSFILT(N,H1) ! S:D1 &
  TRANSFILT(N,H1) ! S:D2 ) ;
  DD:=CMPLX(D1,D2) ; CPT counts DD ;
  CPT6:=MOD(CPT,6) ;
  U extracts DD synchro(selects CPT6=0) ;
  if CPT>=DEB EGA then
  ( ( EGFILT(M) ? EE:U,H:H ! S:Y &
  EGFILT(N) ? EE:A,H:G ! S:Z ) ;
  ( COSTETA:=COS(TETA) &
  SINTETA:=SIN(TETA) ) ;
  G:=CMPLX(COSTETA,SINTETA) ;
  YCORR:=Y*G ; S:=YCORR-Z ;
  CPT_TRIBIT counts S ;
  if CPT_TRIBIT<=DEB AUTO then
  A:=-AP(CPT_TRIBIT) else AUTO APPRENT f1 ;
  ER:=A-S ; CONJ_YCORR:=CONJG(YCORR) ;
  EPSILON:=ER*CONJ_YCORR ;
  IM_EPSILON:=AIMAG(EPSILON) ;
  AJUSTPHASE(GAMMA1,GAMMA2,GAMMA3)?ERREUR:IM_EPSILON ;
  CONJ_G:=CONJG(G) ; ERTOUR:=-ER*CONJ_G ;
  ( AJUST(M,MU1,H0) ? EE:U,ERREUR:ERTOUR,H:H ! H:H &
  AJUST(N,MU2,GO) ? EE:A,ERREUR:ER,H:G ! H:G ) f1
  @ A,TETA,H,G ) f1 ) !! A

where
integer CPT,CPT6,CPT_TRIBIT;
real TETA init 0.0 e 0,COSTETA,SINTETA,
RE_EPSILON,IM_EPSILON,D1,D2;
complex A init (0.0 e 0,0.0 e 0),
U,Y,Z,G,YCORR,S,ER,
CONJ_YCORR,EPSILON, CONJ_G,ERTOUR,DD;
(M)complex H init H0;
(N)complex G init GO

process
.....

end
  
```

Annexe 2

Figure 1
Connexion carte-ligne / carte-modem
mode analogique

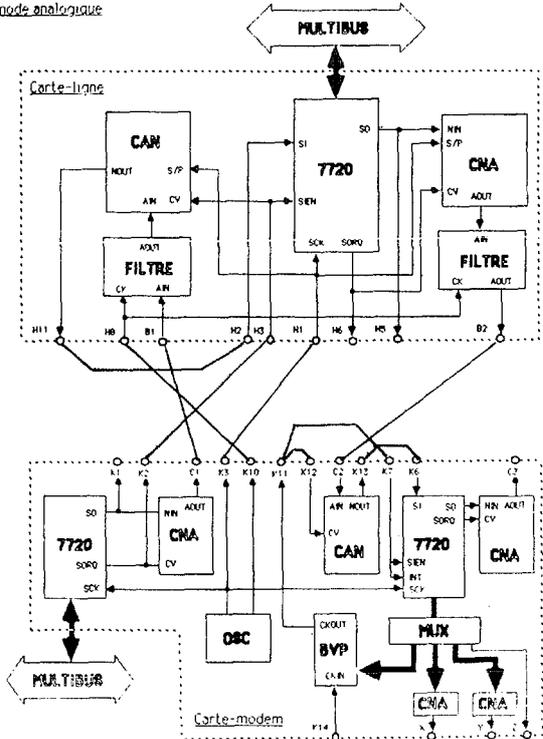


Figure 2
Connexion carte-ligne / carte-modem
mode numerique

