HUITIEME COLLOQUE SUR LE TRAITEMENT DU
SIGNAL ET SES APPLICATIONS

NICE du 1er au 5 JUIN 1981

REALISATION MICROPROGRAMMEE D'UN CODEUR EN SOUS-BANDES POUR LA TRANSMISSION DU SIGNAL DE PAROLE A 16 KBIT/S

J.P. Béraud, C. Galand, D.Esteban

Centre d'Etudes et de Recherches IBM
06610 La Gaude

RESUME

Cette démonstration par poster a pour but de présenter un système de compression numérique du signal de parole utilisant une technique de codage en sous-bandes, et fonctionnant en temps réel.

La méthode utilisée, dont le principe théorique avait été décrit dans un précédent Colloque GRETSI /1/, permet d'obtenir, après compression du taux d'information à 16 Kbit/s, un signal décodé ayant une bonne qualité de communication, ou 'sub-téléphonique'. En d'autres mots, le bruit de quantification est audible, mais reste assez faible, et ne produit pas de perte d'intelligibilité. La qualité du signal est comparable à celle obtenue par un modulateur delta fonctionnant à 32 kbit/s. Le codage en sous-bandes ne présente en outre qu'une faible complexité par rapport à des techniques assurant sensiblement les mêmes performances à ce débit, comme par exemple le codage prédictif adaptatif.

Le système présenté se compose de deux prototypes de laboratoire reliés par une ligne 4 fils. Dans le premier prototype, le signal de parole d'une des voies est échantillonné et numérisé sur 12 bits par modulation par impulsions codées (MIC). Après compression, le train de bits est envoyé sur 2 fils au deuxième prototype qui effectue la décompression et la conversion en analogique. Pour permettre d'apprécier les performances du codage, la deuxième voie est transmise directement en analogique. Les prototypes comprennent un microprocesseur à base de composants du commerce, construit autour d'une unité arithmétique et logique bipolaire Am2901 en tranche /2/.

La programmation des algorithmes a été faite entièrement en langage de haut niveau, assurant ainsi une grande flexibilité dans un environnement de laboratoire. En effet, cette option permet de tenir compte de l'évolution rapide des algorithmes de codage.

Outre la démonstration des prototypes, la présentation comprend une description détaillée des algorithmes, ainsi qu'une évaluation de la charge de calculs correspondante.

SUMMARY

This poster paper presents a real-time operating digital speech compression system, based on sub-band coding techniques.

Digital coding of speech in sub-bands has been described in a preceding GRETSI meeting /1/, and allows to encode the speech signal at 16 kbps with a communications quality. In other words, the quantization noise is audible, but does not decreases the intelligibility. The speech quality is close from that obtained through a Delta Modulator operating at 32 kbps. In addition, sub-band coding requires much less processing than conventional Adaptive Predictive Coders (APC) which provide similar performances at the same bit rate.

The proposed system includes two laboratory prototypes which communicate in full duplex mode through a four wire link. In the first prototype, the input speech signal coming from the first voice line is sampled and digitized at 8 kHz, using 12 bit linear pulse code modulation (PCM). After digital compression at 16 Kbps, the bit stream is sent on 2 wires to the second prototype, in which the decompression and the digital to analog (D/A) conversion are performed. The speech signal coming from the second voice line is transmitted in analog form to the first prototype, so as to allow the comparison between the original and decoded speech signals. The prototypes are build around a bit slice microprocessor Am2901 in bipolar technology /2/.

The speech compression algorithms have been programmed in a High Level Language (HLL). The major goal of this option was to offer a high flexibility in a laboratory environment, and therefore to take into account the fast evolution of the coding algorithms.

The poster presentation includes the real-time operation of the prototypes, as well as a detailed description of the algorithms and of the correponding processing load.

REALISATION MICROPROGRAMMEE D'UN CODEUR EN SOUS-BANDES POUR LA TRANSMISSION DU SIGNAL DE PAROLE A 16 KBIT/S

1.0 INTRODUCTION.

Cet article faisant l'objet d'une présentation par poster, il a été jugé préférable de restreindre le volume du texte au bénéfice de la publication de l'intégralité des schémas, tableaux et photographies. Pour plus de détails, on pourra se reporter à /1,2,4,5/.

2.0 MICROPROCESSEUR.

2.1 ARCHITECTURE.

La Figure 1 représente le schéma de principe du microprocesseur. La partie 'traitement', composée essentiellement de l'unité arithmétique et de la mémoire de travail, fonctionne en même temps que la partie de 'séquencement des instructions'. Toutes les instructions sont exécutées en un cycle de 250 nanosecondes. L'unité arithmétique et logique (ALU) travaille en virgule fixe sur 16 éléments binaires (e.b ou bit). Elle est composée de 4 cicuits Am2901 et comprend 17 registres. On peut, pour certaines instructions, travailler en double précision sur deux cycles.
Le format des instructions est fixe et se compose de deux champs de 8 bits chacun. Le premier champ comprend le code opération. Le deuxième champ est l'opérande qui est en fonction du code opération, soit une adresse de registre, soit une valeur immédiate, soit un déplacement par rapport au pointeur d'adresse de la mémoire de travail, soit une adresse de branchement du compteur d'instructions.
La mémoire de programme est composée de 3 K mots de mémoire morte (ROM) et de 256 mots de mémoire vive (RAM) où l'on peut charger des instructions générées dynamiquement.
La mémoire de données est adressée en indirect à l'aide d'un registre appelé le pointeur d'adresse (indice) au contenu duquel est ajouté un déplacement. On peut adresser la mémoire par le même mécanisme, mais modulo 16, ce qui permet de simplifier le transfert des données dans les opérations de filtrage numérique. Le séquencement des instructions est assuré par trois circuits Am2911.
Le fonctionnement normal d'un programme peut être interrompu pour exécuter un autre programme dit 'programme d'interruption'. Dans ce cas l'adresse courante est conservée dans un registre spécial du séquenceur. Un deuxième pointeur de la mémoire de travail est attribué à ce nouveau niveau de programme.
Le microprocesseur est monté sur une carte de 18 x 24 cm (Fig.2), et comprend 80 circuits TTL, les 4 unités arithmétiques Am2901, et les 3 séquenceurs Am2911. La puissance consommée par la carte est de l'ordre de 30 watts.

2.2 SUPPORT DE PROGRAMMATION.

La programmation du microprocesseur est facilitée par un assembleur, un langage de haut niveau, et un simulateur /2/ (Fig.3).
L'assembleur permet d'écrire des programmes en utilisant des mnémoniques (80 au total) pour définir les instructions utilisées, et des symboles alphanumériques pour définir des adresses. Il résoud les adresses et convertit les mnémoniques et ses opérandes en mots binaires.
Le langage de haut niveau est dérivé d'un langage propre à l'ordinateur IBM 370: le code généré est retranscrit pour être assimilable par l'assembleur. La traduction repose sur la similitude entre l'architecture de l'IBM 370 et celle du microprocesseur.
Le simulateur facilite la mise au point des programmes, et fonctionne de façon interactive sur IBM 370.

2.3 PROGRAMMATION DES MULTIPLICATIONS.

L'architecture ne comprennant pas de multiplieur cablé, les multiplications sont remplacées par une série d'additions et de décalages suivant la représentation canonique minimale signée des nombres binaires /3/:

- Si l'un des termes de la multiplication est connu, la séquence d'instructions correspondant à la succession des additions et des décalages est écrite à l'assemblage du programme. C'est le cas des filtres numériques à coefficients fixes dont un exemple est donné à la Figure 4.
- Si les deux termes sont inconnus à priori, comme par exemple dans une opération de filtrage adaptatif, la séquence correspondante d'instructions est générée par programme, chargée dans la partie dynamique (RAM) de la mémoire programme (Fig.1), puis exécutée pour effectuer la multiplication.

3.0 CODAGE EN SOUS-BANDES.

Le codage en sous-bandes consiste à filtrer le signal de parole par un banc de filtres passe-bande adjacents couvrant toute la bande téléphonique, puis à re-échantillonner les signaux résultants à leur fréquence de Nyquist, et finalement les quantifier séparément. La reconstitution du signal de parole se fait en ramenant les signaux dans leur bande d'origine par interpolation, filtrage par le même banc de filtre, et sommation. Les performances de ce type de codage proviennent de deux propriétés : 1/ le nombre de bits utilisés pour quantifier chaque sous-bande peut être ajusté suivant l'énergie du signal, ce qui permet de minimiser le bruit de quantification, et 2/ ce bruit reste localisé dans la sous-bande où il est produit.
Le re-échantillonnage des sous-bandes est capital, car il permet de conserver la quantité d'information constante. Or, les réponses harmoniques des filtres utilisés doivent se couper à -3 dB si l'on veut assurer une réponse globale unitaire dans toute la bande téléphonique. L'échantillonnage provoque donc des repliements de spectre d'une sous-bande à l'autre. Pour éliminer ces repliements auxquels l'oreille est très sensible, on utilise un banc de filtres miroirs en quadrature (QMF) /4/.
Considérons le schéma de principe du codeur représenté à la Figure 5. Le signal de parole est filtré, puis échantillonné à 8 kHz, codé sur 12 bits par modulation par impulsions codées, puis filtré par un banc de 8 filtres QMF. Il en résulte 8 signaux de sous-bandes re-échantillonnés à 1 kHz.
$\{ y_i(n) \quad (i=1,2,\ldots,8) \}$.

REALISATION MICROPROGRAMMEE D'UN CODEUR EN SOUS-BANDES
POUR LA TRANSMISSION DU SIGNAL DE PAROLE A 16 KBIT/S

Chaque échantillon de la sous-bande 'i' est ensuite quantifié avec ni bits; la somme des ni est constante et détermine le taux de bits global. Dans chaque sous-bande, le pas de quantification Qi est ajusté toutes les 16 ms, c'est à dire pour chaque bloc de 16 échantillons.

$$(1) \qquad Q_i = C_i / 2^{n_i - 1}$$

avec

$$C_i = \text{Max} \ (|y_i(n)|, \ n=1,\ldots,16)$$

L'allocation des ressources binaires est calculée de façon adaptative, ce qui permet d'améliorer les performances du codeur. On peut en effet montrer /5/ que l'énergie totale du bruit de quantification est minimale si les ni sont calculés par :

$$(2) \qquad n_i = \frac{1}{8} \left(N - \sum_{j=1}^{8} \log_2 C_j \right) + \log_2 C_i$$

où N représente le nombre de bits total à distribuer. La relation (2) montre que l'allocation optimale des bits ne nécessite la transmission d'aucune information supplémentaire, puisque les paramètres Ci sont de toute façon transmis pour déduire au récepteur les pas de quantification à l'aide de la relation (1). Les paramètres Ci de chaque sous-bande sont donc ajustés toutes les 16 ms, et codés sur 5 bits de façon logarithmique. Les 40 bits correspondants sont ensuite multiplexés avec les échantillons quantifiés des sous-bandes (Fig.5). L'allocation adaptative des bits (AAB) est recalculée au récepteur à partir des Ci, de la même manière qu'à l'émetteur, en utilisant la relation (2). Remarquons que d'après cette relation, ni peut devenir négatif, ou supérieur à la valeur maxima considérée (en pratique $0 \leqslant ni \leqslant 5$). En fait, un algorithme simple permet de contrôler ces écarts /5/.
Les caractéristiques du codeur sont résumées sur la Figure 6. La Figure 7 représente les réponses harmoniques des filtres QMF.

4.0 TRANSMISSION A 16 KBPS.

La programmation du codeur en sous-bandes a été effectuée en utilisant le support décrit précédemment. Le nombre d'instructions nécessaires à chacune des fonctions de l'émetteur et du récepteur est indiqué à la Figure 8. Dans chaque cas, on a mentionné la taille du programme, ainsi que le nombre d'instructions exécutées en moyenne à chaque instant d'échantillonnage (1/8 kHz). En résumé, l'ensemble du codeur nécessite environ 1500 instructions (ROM), 900 mots de mémoire vive (RAM), et le déroulement de 1,3 million d'instructions par seconde (MIPS).
Pour la transmission, l'émetteur doit mettre les données compressées à 16 Kbit/s sous la forme d'un train de bits série. Comme d'autre part le traitement du signal de parole se fait par blocs de 16 ms, le récepteur doit reconnaître la limite des blocs. Cette synchronisation est rendue possible grâce à l'insertion de bits dits de synchronisation dans chaque bloc de données. La répartition des bits dans chaque bloc est la suivante :

- Echantillons	:	208 bits
- Caractéristiques	:	40 bits
- Synchronisation	:	8 bits
- TOTAL (pour 16 ms)	:	256 bits

Le récepteur fonctionne dans un des deux modes suivants :

- o Recherche de synchronisation
  Chaque période de l'horloge à 16 kHz, correspondant à la réception en série des bits, provoque une interruption pendant laquelle le nouveau bit reçu est chargé en mémoire, et examiné par rapport aux derniers bits reçus. Si la séquence de huit bits de synchronisation n'est pas détectée, ce mode de fonctionnement est conservé à la prochaine interruption. Dans le cas contraire, le mode est commuté comme suit.

- o Fonctionnement synchrone
  Dans ce cas les interruptions sont faites à 8 kHz. A chaque interruption, un échantillon de parole reconstitué est présenté sur la sortie correspondant au convertisseur numérique analogique (D/A). Toutes les huit interruptions, un mot de 16 bits est chargé à partir du registre de conversion série/parallèle (Fig.9). Un bloc complet est donc chargé en 16 fois. Pour chaque bloc, la présence des huit bits de synchronisation est vérifiée avant d'effectuer la décompression. Si le test est négatif, le programme active le premier mode de fonctionnement.

5.0 CONCLUSION.

Dans cette présentation par poster, on a démontré la simplicité de mise en oeuvre d'un codeur en sous-bandes, qui ne présente qu'une faible complexité -1,3 MIPS- par rapport à des techniques assurant sensiblement les mêmes performances à ce débit, comme par exemple le codage prédictif adaptatif avec prédiction à long terme (environ 3 MIPS avec le même microprocesseur, et environ 2 MIPS avec l'adjonction d'un multiplieur).
Bien que l'on puisse envisager une mise en oeuvre à faible coût sur des microprocesseurs récemment commercialisés, l'approche adoptée a permis une programmation rapide des algorithmes, assurant une grande souplesse dans un environnement de laboratoire.

6.0 REFERENCES.

/1/C.Galand, D.Esteban, J.Menez, D.Mauduit, 'Système de codage du signal de parole par décomposition spectrale', Colloque GRETSI, Nice, Avril 1977
/2/J.P.Béraud, D.Esteban, C.Galand, D.Mauduit, O.Maurel, M.Orsini, 'Une nouvelle architecture de microprocesseur pour le traitement du signal de parole', Colloque GRETSI, Nice, Juin 1979
/3/A.Peled, 'On the Hardware Implementation of Digital Signal Processors', IEEE Trans. on ASSP, Vol.24, Feb. 1976.
/4/D.Esteban, C.Galand, 'Application of Quadrature Mirror Filters to Split-band Voice Coding Schemes', IEEE Proc. Int. Conf. on Acoustics, Speech and Signal Processing, Hartford, pp.191-195, May 1977
/5/C.Galand, D.Esteban, 'An efficient approach to digital signal companding for PCM transmission', Journées Internationales d'Etude sur la Transmission des Données Liège, 21-22 Novembre 1977

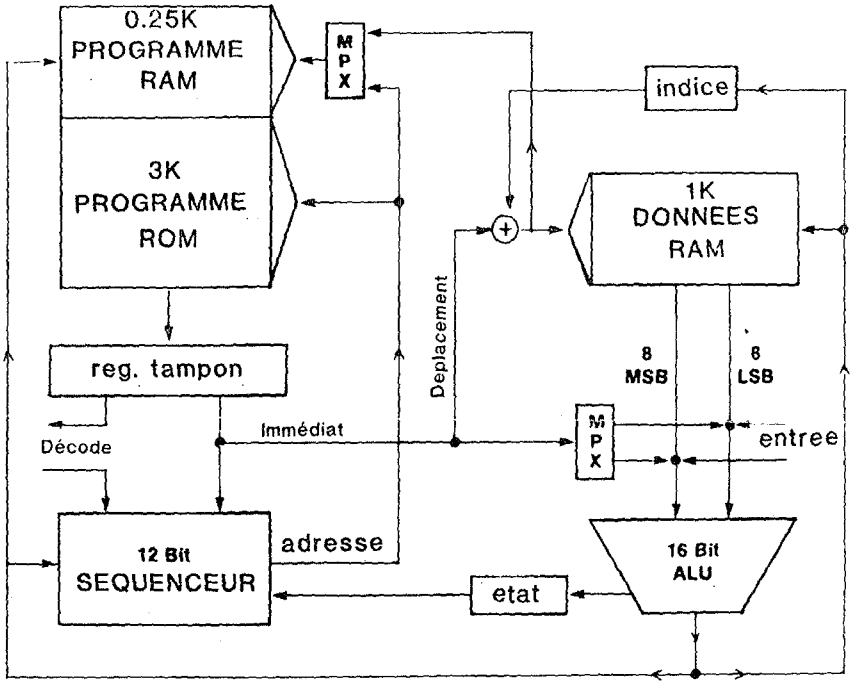

FIG 1 SCHEMA DE PRINCIPE
DU MICROPROCESSEUR

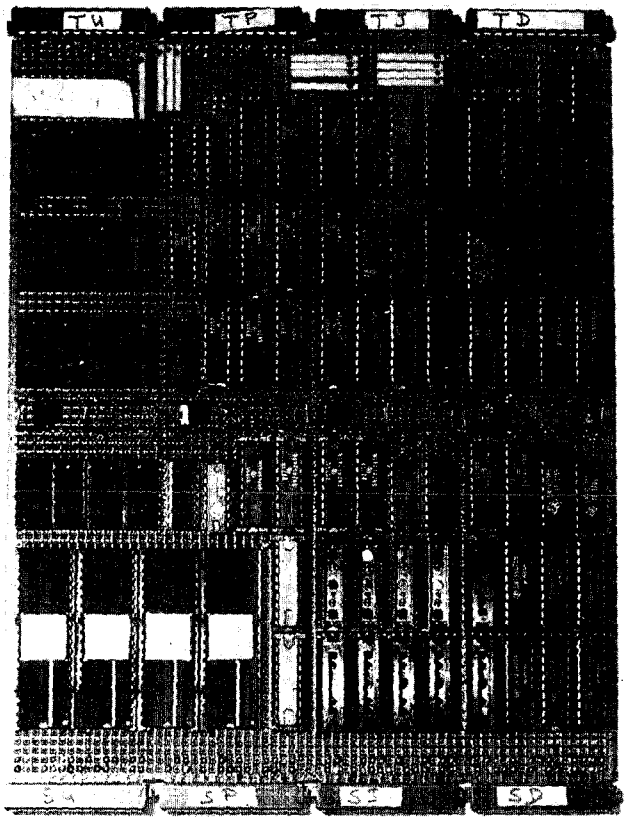

FIG 2 CARTE MICROPROCESSEUR

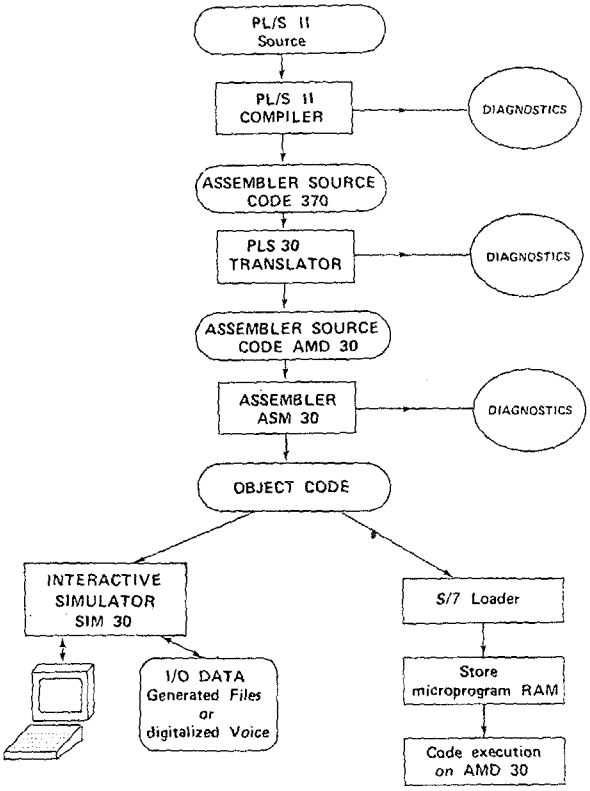

FIG 3 SUPPORT DE PROGRAMMATION

```
*   /*    1 =  0  0  0  0  1    =  MEM(1)  */
*   /*    7 =  0  1  0  0 -1    =  MEM(2)  */
*   /*   -5 =  0  0 -1  0 -1    =  MEM(3)  */
*   /*   12 =  1  0 -1  0  0    =  MEM(4)  */

*       ACC= 0 ;      /* ACC RESET  */
        SR    ACC,ACC

*       ACC=ACC +MEM(1);
        A     ACC,MEM(@PTR)

*       ACC=ACC -MEM(2);
        S     ACC,MEM+2(@PTR)

*       ACC=ACC -MEM(3);
*       SRA(ACC,2);  /* DIVIDE ACC */
        SSH   ACC,MEM+4(@PTR)
        SHR   ACC,ACC

*       ACC=ACC -MEM(3);
        S     ACC,MEM+4(@PTR)

*       ACC=ACC -MEM(4);
*       SRA(ACC,1);  /* DIVIDE ACC */
        SSH   ACC,MEM+6(@PTR)

*       ACC=ACC +MEM(2);
*       SRA(ACC,1);  /* DIVIDE ACC */
        ASH   ACC,MEM+2(@PTR)

*       ACC=ACC +MEM(4);
*       SRA(ACC,1);  /* DIVIDE ACC */
        ASH   ACC,MEM+6(@PTR)
```

FIG 4 EXEMPLE DE PROGRAMMATION DES MULTIPLICATIONS: FILTRE A COEFFICIENTS FIXES

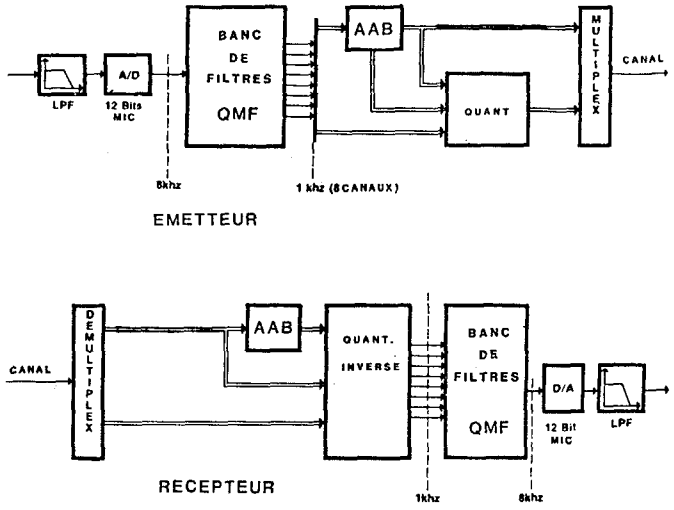

FIG 5 SCHEMA DE PRINCIPE DU CODEUR EN SOUS-BANDES

CARACTERISTIQUES DU CODEUR EN SOUS-BANDES A 16 KBPS

- FREQUENCE D'ECHANTILLONNAGE : 8 KHZ
- LONGUEUR DES BLOCS : 16 MS
- NOMBRE DE SOUS-BANDES : 8
- ORDRE DES FILTRES PASSE-BANDE : 40
- ALLOCATION ADAPTATIVE DE 13 KBITS/S
- QUANTIFICATION DES
  - ECHANTILLONS : DE 0 A 5 BITS
  - CARACTERISTIQUES : 5 BITS

FIG 6 CARACTERISTIQUES

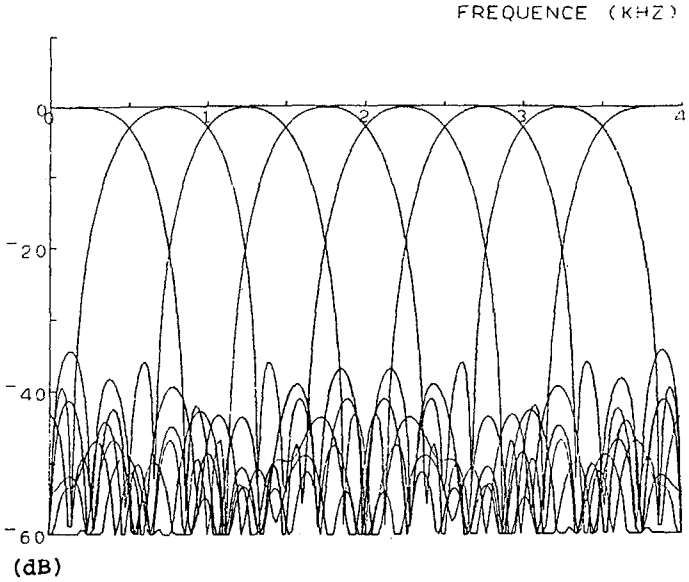

FIG 7 REPONSES HARMONIQUES DU BANC DE FILTRES Q.M.F

REALISATION MICROPROGRAMMEE D'UN CODEUR EN SOUS-BANDES
POUR LA TRANSMISSION DU SIGNAL DE PAROLE A 16 KBIT/S

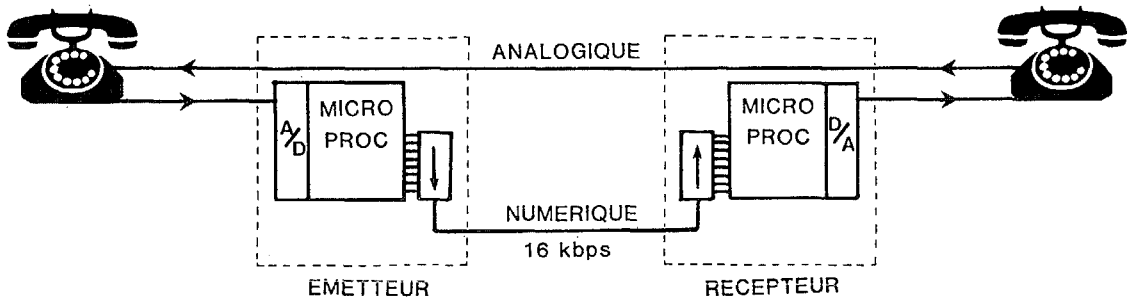

FIG 9 SCHEMA DU SYSTEME DE COMPRESSION

16 KBPS S.V.C.S : MICROPROGRAMME

	INSTRUCTIONS	
	MEMORISEES	EXECUTEES /ECHANT.
EMETTEUR		
Filtrage	340	42
Quantification	428	28
Multiplexage	60	15
Entrees/Sorties	50	4
RECEPTEUR		
Entrees/Sorties	50	4
Demultiplexage	60	15
Quantification inverse	120	15
Filtrage	340	42
TOTAL	1448	165

MEMOIRE PROGRAMME = 1448 MOTS DE 16 bits
MEMOIRE DONNEES = 900 MOTS DE 16 bits
CHARGE DE CALCUL = 1.32 MIPS

FIG 8

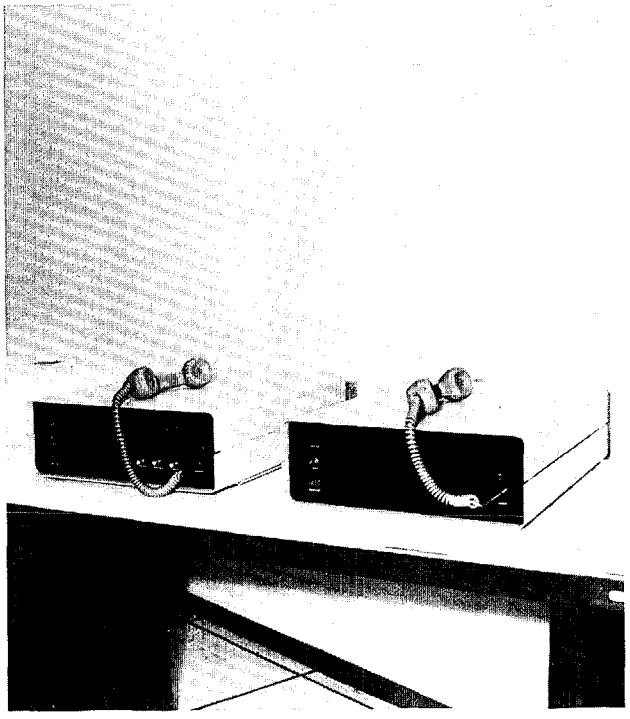

FIG 10 MAQUETTES EXPERIMENTALES PRESENTEES