THÈSE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNIQUES DE LILLE

pour obtenir le grade de

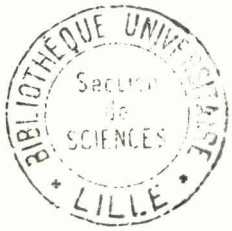
DOCTEUR - INGENIEUR

par

Abdenbi MANDAR

Ingénieur E.U.D.I.L

ETUDE THEORIQUE ET REALISATION D'UN SYSTEME AUTONOME INTERACTIF DE TRANSCRIPTION BRAILLE



Soutenue le 22 avril 1982 devant la Commission d'Examen

Membres du Jury	MM.	A. LEBRUN	Président
		J.P. DUBUS	Rapporteur
		L. AVAN	Examinateur
		L. RACZY	Examinateur
		Y. MOSCHETTO	Examinateur
		M. JACQUIN	Invité
		P. SEJEAN	Invité
		J.P. TOSSER	Invité

à MOUNA,
à AICHA,
à tous ceux qui me sont chers,

Ce travail a été effectué au sein de l'équipe de "Modélisation et Instrumentation électronique" du "Centre de Recherche Sciences des Matériaux et Techniques de Construction" (CRESMAT) de l'Université des Sciences et Techniques de Lille.

Monsieur le Professeur LEBRUN, Directeur du Laboratoire, m'a accueilli dans son Laboratoire. Qu'il me permette de lui exprimer ici ma profonde gratitude pour les encouragements qu'il m'a prodigués et pour l'honneur qu'il me fait de présider le jury de cette thèse.

Je voudrais que ce mémoire soit un témoignage de ma reconnaissance envers M. DUBUS, Professeur, qui m'a confié le thème de cette étude. Les nombreux encouragements et conseils précieux qu'il m'a apportés ont été déterminants pour la réalisation de cette recherche. Je tiens à le remercier pour son amicale collaboration ainsi que pour la confiance qu'il m'a témoignée tout au long de ce travail.

Je ne crois pas que les mots puissent suffisamment exprimer ma reconnaissance envers MM. LEBRUN et DUBUS qui m'ont toujours soutenu en des moments difficiles, aussi qu'il me soit permis de leur dire tout simplement merci.

Je suis particulièrement reconnaissant à Monsieur le Professeur RACZY, Directeur de l'Institut "Informatique, Mesures, Automatique" de l'EUDIL, pour l'honneur qu'il me fait de juger ce travail.

Que Monsieur MOSCHETTO, Directeur du CTB INSERM, Président du Pôle Régional Génie Biomédical, soit remercié de l'intérêt qu'il a toujours porté à cette étude et en acceptant de participer à mon jury.

Monsieur AVAN, Professeur au CNAM de Paris me fait le grand honneur de faire partie du jury, qu'il trouve ici l'expression de ma très grande reconnaissance.

Je tiens à remercier MM. TOSSER et LEGAI pour l'accueil qu'ils m'ont toujours réservé lors de mes visites à l'Ecole Nationale des Handicapés Visuels de Loos et qui sont les instigateurs de cette étude. Que Monsieur TOSSER soit remercié pour avoir bien voulu faire partie du jury de cette thèse.

Mes remerciements vont tout particulièrement à MM. JACQUIN et CHAZAL pour la confiance qu'ils m'ont témoignée et pour leurs précieux conseils en matière de Braille abrégé. Monsieur JACQUIN a accepté de faire partie de mon jury, qu'il en soit remercié.

Monsieur SEJEAN Proviseur de l'Ecole Nationale des Handicapés Visuels de Loos a accepté de faire partie de mon jury, je le remercie vivement.

Que Monsieur WATTRELOT, Ingénieur, Monsieur QUENEE et toute l'équipe du Laboratoire sachent que j'ai toujours apprécié leur compétence et estimé leurs conseils et leurs critiques et pris plaisir à discuter et travailler avec eux.

Madame CASTEGNIER, Secrétaire au Laboratoire, a eu le rôle ingrat, mais combien nécessaire de dactylographier cette thèse, je lui exprime mes vifs remerciements pour sa compétence et sa vigilence.

Monsieur DELVAS, Professeur de Braille à l'Ecole des Handicapés Visuels de Loos et Melle BARROIS m'ont toujours aidé chaque fois que j'ai eu besoin de leurs connaissances en Braille. Je les remercie infiniment.

Je n'oublie pas tous ceux qui de près ou de loin m'ont apporté leur aide, qu'ils en soient ici remerciés.

Je remercie enfin toute l'équipe du service imprimerie qui a assuré l'impression de ce document.

S O M M A I R E

-:-:-:-:-:-:-:-:-:-

N° page

INTRODUCTION..... 1

1ÈRE PARTIE : DÉFINITION ET ÉTUDE DES FONCTIONS TEMPS RÉEL DE LA TRANSCRIPTION

I - DEFINITION DES BESOINS DES INDIVIDUS ET DES ORGANISMES..... 3

II - RAPPELS SUR LE PRINCIPE DE L'ECRITURE EN RELIEF..... 4

II-1- LE LIVRE EN RELIEF..... 5

II-2- LE BRAILLE..... 5

II-3- LE BRAILLE INTEGRAL..... 7

II-4- LE BRAILLE ABREGE..... 8

III - DEFINITION DES FONCTIONS D'UN TRANSCRIPTEUR AUTONOME BRAILLE DE LA SAISIE AU CLAVIER DE TEXTES "NOIRS"..... 10

IV - ENUMERATION DES PARTICULARITES DE LA SYNTAXE BRAILLE INTEGRAL PAR RAPPORT A CELLE PREVUE PAR L'EMPLOI D'UN CLAVIER AZERTY..... 12

V - JUSTIFICATION DE L'INFORMATION A APPORTER A L'OPERATEUR POUR EFFECTUER LA COUPURE DES MOTS EN FIN DE LIGNE..... 17

V-1- DEFINITION DES GRANDEURS MATHEMATIQUES A INTRODUIRE..... 17

V-2- JUSTIFICATION DE LA METHODE DE COUPURE DES MOTS EN FIN DE LIGNE DE TEXTE BRAILLE..... 19

VI - DEFINITION ET DESCRIPTION DE LA VISUALISATION DE L'EDITION SUR L'ECRAN..... 21

VI-1- DESCRIPTION DU PRINCIPE DE LA GENERATION DE L'IMAGE SUR L'ECRAN.... 21

VI-2- REPRESENTATION DES TEXTES AVEC ESPACE ET INTERLIGNE VARIABLES..... 26

VI-3- DESCRIPTION DU LOGICIEL QUI PERMET DE FAIRE DEFILER UNE PARTIE DU TEXTE SUR L'ECRAN..... 28

VII - SYNOPTIQUE DES LOGICIELS DE TRADUCTION EN BRAILLE INTEGRAL.. 32

VII-1- LOGICIEL DE GESTION CLAVIER ET TRADUCTION BRAILLE INTEGRAL..... 32

VII-2- LOGICIEL DE GESTION DES TEXTES..... 35

.../...

.../...

2ÈME PARTIE : ETUDE DE LA TRANSCRIPTION EN BRAILLE ABRÉGÉ

I - ETUDE DU DICTIONNAIRE ET DE SA STRUCTURATION..... 38

I-1- APPLICATION DE LA THEORIE DES ENSEMBLES A L'ETUDE DU DICTIONNAIRE ... 38

I-2- IMPLANTATION DES DICTIONNAIRES EN MEMOIRE 49

II - DECOMPOSITION D'UN MOT EN CONTRACTIONS..... 58

II-1- NOTATIONS UTILISEES 58

II-2- POSITION DU PROBLEME ET JUSTIFICATION DE LA METHODE DE DECOUPE ADOPTEE DES MOTS 60

II-3- ENUMERATION DES DIFFERENTES REGLES RELATIVES AUX CONTRACTIONS SIMPLES. 61

II-4- CONTRACTIONS COMPOSEES..... 65

II-5- AUTRES PROBLEMES PARTICULIERS A RESOUDRE 67

II-6- STRUCTURE DES REGISTRES ET CALCUL DU TEMPS DE TRANSCRIPTION..... 68

3ÈME PARTIE : DESCRIPTION TECHNOLOGIQUE DU SYSTEME ET ARCHITECTURES D'APPLICATION DE L'AUTOMATE

I - INTRODUCTION 76

II - SYNOPTIQUE DE LA CONFIGURATION STATION AUTONOME 77

II-1- DESCRIPTION DES FONCTIONS DE L'EDITEUR 77

II-2- DESCRIPTION DES CIRCUITS DE L'EDITEUR 83

II-3- DESCRIPTION DU DUPLICATEUR..... 87

III - PROPOSITION DES DIFFERENTES UTILISATIONS POSSIBLES DU PROCEDE PERMETTANT LA COMMUNICATION VOYANT-AVEUGLE..... 92

III-1- UTILISATION INDIVIDUELLE ET TELEMATIQUE DU DISPOSITIF 92

III-2- UTILISATION DU DISPOSITIF COMME MODULE SPECIALISE 93

CONCLUSION 94

BIBLIOGRAPHIE 95

ANNEXE 100

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

L'édition de textes relief en Braille à partir de textes noirs nécessite un personnel spécialisé (transcripteurs connaissant le Braille). Le faible nombre de transcripteurs Braille disponibles ne permet pas de répondre aux besoins actuels de l'édition massive d'ouvrages en Braille.

Il est par conséquent nécessaire de chercher les moyens de transcrire des textes noirs en Braille sans nécessiter un effectif accru de personnel spécialisé. Une solution possible consiste à pratiquer la transcription automatique à partir de la saisie d'un texte noir au clavier par un personnel non spécialisé du Braille ou de l'Informatique. Cette démarche intéresse à l'origine les établissements scolaires pour résoudre les problèmes logistiques de la pédagogie, et les associations qui éditent des ouvrages pour aveugles.

De nombreux travaux de recherche mettant en oeuvre des techniques de l'automatique et de l'informatique ont été entrepris à partir de trois grandes classes de modes de saisie : la saisie de textes par reconnaissance de la forme des caractères noirs, la transcription à partir de l'édition de textes stockés sous forme de bande de photocomposition, la saisie manuelle de textes.

Dans le premier cas, le procédé de saisie s'adapte bien à l'instrumentation individuelle de lecture de textes noirs par un aveugle. Mais, dans tous les autres cas, il y a toujours à l'origine une saisie manuelle de texte.

Notre connaissance du problème posé et les techniques de la microélectronique nous ont permis en même temps que l'évolution de la technologie pendant le cours même de nos travaux, de concevoir un système interactif de transcription en temps réel de la saisie d'un texte noir en Braille. Si un tel système existe, il peut équiper tout matériel de saisie au clavier de textes noirs (machine de traitement de textes, photocomposeuse et machines à écrire futures).

Le mémoire que nous présentons décrit la démarche que nous avons adoptée pour bâtir le cahier des charges de cet automate, les algorithmes de la traduction qui conduisent à une réduction considérable de la taille mémoire du programme et les circuits microélectroniques qui équipent cet automate dont nous avons participé activement à la réalisation.

.../...

.../...

Dans une première partie, après avoir brièvement rappelé les principes de l'écriture du relief Braille d'une part, et d'autre part les besoins de l'édition du Braille à partir de la frappe au clavier d'une dactylographe qui ne connaît pas le Braille, nous décrivons les grandes lignes du cahier des charges de cet automate.

Dans une deuxième partie, nous décrivons la classification systématique des règles de traduction en Braille intégral et Braille abrégé en vue de l'élaboration des algorithmes de traduction rapide Noir-Braille.

Nous développons les concepts mathématiques de la théorie de l'informatio que nous avons mis en oeuvre pour concevoir des algorithmes de traduction rapide qui conduisent à une méthode originale de structuration et recherche de mots dans le dictionnaire et à une taille de programme minimale.

Dans une troisième partie, nous décrivons la réalisation de l'architecture d'une machine autonome et interactive de transcription en Braille intégral et abrégé.

En conclusion, à partir de l'existence même de ce module de transcription de chaînes de caractères de textes "noirs" en Braille abrégé qui gère en même temps divers périphériques, nous proposons diverses configurations possibles d'application de ce système.

PREMIERE PARTIE

DÉFINITION ET ÉTUDE DES FONCTIONS TEMPS RÉEL DE LA TRANSCRIPTION

I - DEFINITION DES BESOINS DES INDIVIDUS ET DES ORGANISMES

Nous décrivons dans ce paragraphe les besoins des individus et organismes, l'état des recherches et la place qu'occupe nos travaux dans ce domaine.

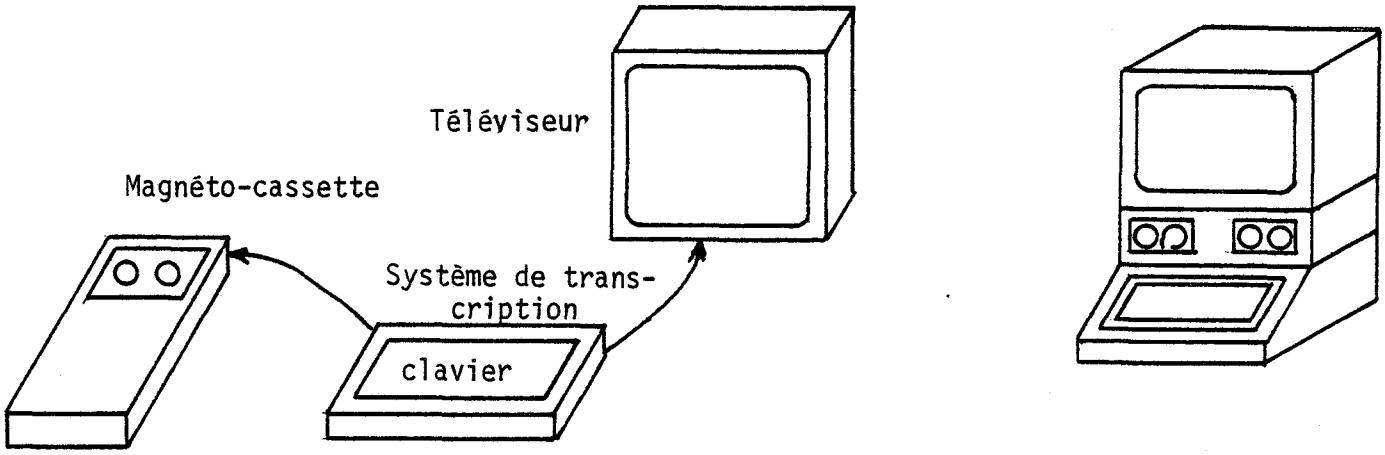
Jusqu'en 1979, la production en relief Braille intégral et surtout Braille abrégé ne se faisait que par des personnes qui connaissaient le Braille. Cette pratique ne répondait pas d'une façon satisfaisante aux besoins pédagogiques des établissements scolaires qui nécessitaient de pouvoir transcrire rapidement des notes de service ou des textes scolaires (exercices, épreuves d'examen, éléments de cours etc...). Elle ne répondait pas non plus à la production à grand débit de l'édition d'ouvrages en Braille par des associations ou des maisons d'édition en Braille.

A cette époque, des études avaient été menées par différentes approches (Courtin, Truquet etc.) pour traduire automatiquement par ordinateur des textes noirs en Braille. C'est ainsi qu'il existait dans la mémoire d'ordinateurs de divers établissements des programmes écrits en Fortran ou autres langages évolués, de traduction de textes "noirs" en Braille. Les seuls travaux publiés à notre connaissance étaient à cette époque ceux référencés [6] [7] [8]. Le problème est qu'ils nécessitaient l'utilisation d'un gros ordinateur et obligeaient les organismes ayant à éditer les textes en Braille abrégé ou intégral à s'adresser à des centres de traduction avec certes des avantages énormes par rapport aux moyens d'édition existant jusqu'à présent, mais aussi avec des inconvénients liés à l'éloignement de ces centres et de leurs utilisateurs ou à un investissement en matériel considérable.

Notre démarche a consisté à résoudre tous les problèmes inhérents à la différence de syntaxe entre Braille et "noir" au niveau du logiciel et du matériel pour constituer un automate le plus évolué et le moins contraignant possible en vue de rendre la transcription la plus transparente des textes noirs en Braille sur place sans nécessiter de centres de traduction ou de personnel spécialisé, mais aussi sans vouloir se substituer à ces centres ou à mettre en cause leur existence.

A partir des besoins qui nous avaient été formulés et de l'idée de l'instrumentation individuelle poursuivie au laboratoire, nous avons imaginé un automate interactif qui permet de traduire d'une façon transparente en Braille en temps réel la frappe au clavier d'une machine à écrire de toute personne sans nécessiter soit la connaissance du Braille, soit la connaissance d'ordres informatiques pour mettre en oeuvre la saisie.

Ainsi les établissements scolaires ou les associations peuvent produire du Braille en banalisant les personnels de saisie. Par ailleurs, un tel automate permettrait, en faisant évoluer la technologie, d'aboutir à la réalisation d'une machine miniature de faible coût. On peut, en effet, imaginer la possibilité pour toute personne de coupler cet automate à ses périphériques domestiques (téléviseur, magnétocassette) pour communiquer par écrit à un aveugle sans connaître le Braille. On réaliserait ainsi un organe de communication individuel voyant-aveugle.



Organe de communication individuelle
voyant - aveugle

Logibraille ou
système autonome

Figure 1

II - RAPPELS SUR LE PRINCIPE DE L'ECRITURE EN RELIEF

Le moyen le plus ancien qui a permis aux déficients visuels l'accès à la connaissance est la lecture du texte par un voyant [1][2]. L'aveugle n'a pas de contact avec le texte lui-même. Ce moyen est encore utilisé par des personnes atteintes tardivement de cécité, ou par des aveugles qui bénéficient d'une aide pour lire le courrier, les copies etc...

On imagine sans mal les contraintes d'une telle pratique : disponibilité du lecteur, interprétation du texte etc. de sorte qu'historiquement il a été complété par la communication écrite directe en relief. La chaîne de communication écrite directe est composée d'un organe source, soit manuel (membre preneur comme la main, le pied, la bouche par exemple), soit mécanique (machine à embosser), et un instrument graveur (poinçon), d'un canal support de l'information matérialisé à l'origine et encore souvent aujourd'hui par du papier relief et d'un récepteur constitué par un organe tactile (peau du doigt ou de la main en général).

II-1- LE LIVRE EN RELIEF

La première idée, assez intuitive d'ailleurs, consista à reproduire sur un livre des caractères en relief pouvant être lus par le toucher. Cette représentation n'était pas adaptée au sens du toucher : trop d'informations à saisir pour chaque caractère et donc lenteur de reconnaissance et fatigue.

II-2- LE BRAILLE

En 1825, Louis BRAILLE propose un système qui porte son nom. Ce système a eu quelques difficultés au début pour être adopté, mais sa logique et la facilité de la reconnaissance de la forme l'a rendu universel. Il a donné aux déficients visuels la possibilité de lire des textes à une vitesse à peu près égale à celle des voyants.

Un caractère Braille est constitué d'une matrice de six points pouvant être mis en relief (figure 2-a). La fabrication de ce relief peut se faire soit en déposant une matière sur un support papier plan (thermogravure) ou en poinçonnant un creux au verso du support papier (poinçonnement ou embossage), soit en déformant une feuille de matière thermodurcissable (thermoformage). Dans le dernier cas de fabrication du relief, qui est aussi et est encore celui utilisé manuellement à l'origine par les aveugles, l'écriture d'un texte relief en Braille se fait dans le sens inverse de la lecture du même texte. C'est-à-dire que l'on écrit en creux de droite à gauche sur une face et on lit le relief de gauche à droite sur l'autre face. Mais dans ces conditions, le caractère Braille apparaît lui-même inversé en écriture et en lecture. Il en résulte une convention de repérage numérique des points d'un caractère qui se comptent dans le sens écriture de la manière représentée figure 2 a.

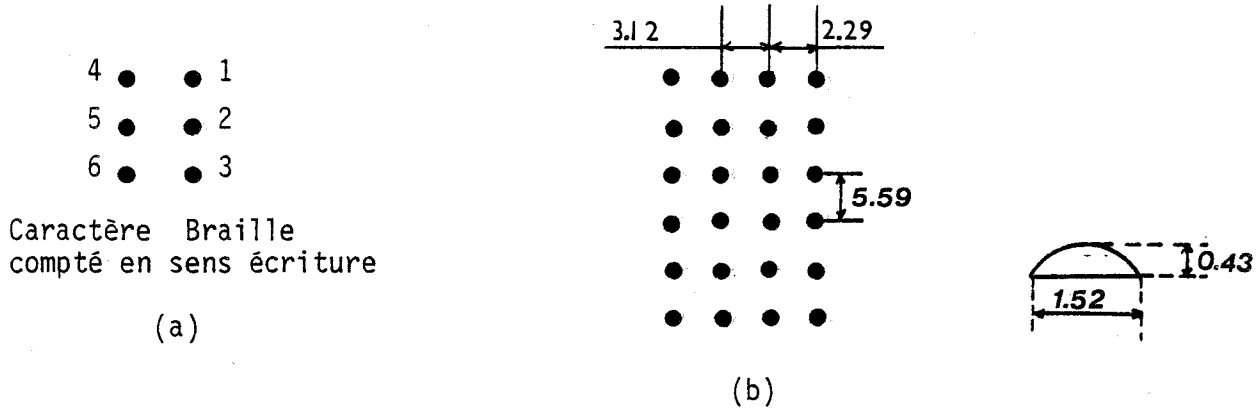


Figure 2

Le nombre de caractères possibles pouvant être généré par ce système est donc de 2^6, soit 64 caractères différents (espaces compris).

La matrice à 6 points est optimale car avec un point en moins le nombre de caractères générés par le système serait insuffisant (32) et avec un point en plus, les dimensions d'un caractère dépasseraient la zone la plus sensible du doigt : la pulpe.

Ce système est actuellement le plus adapté à la psychophysiologie des excitations tactiles qui sont du type "tout ou rien" et il ne demande qu'une faible quantité d'informations à appréhender au toucher.

Si on appelle $A(I,J) = [aij]$ la matrice de 6 points d'un caractère Braille, dont chaque élément est la probabilité d'apparition d'un point en relief égale à $p_i = 0,5$ avec $I = 3$ et $J = 2$, l'information contenue dans l'apparition d'un point en relief est donnée par :

$$H(aij) = \sum_{i=0}^{1} p_i \log \frac{1}{p_i} = 1$$

et l'information contenue dans l'apparition d'un graphème Braille est :

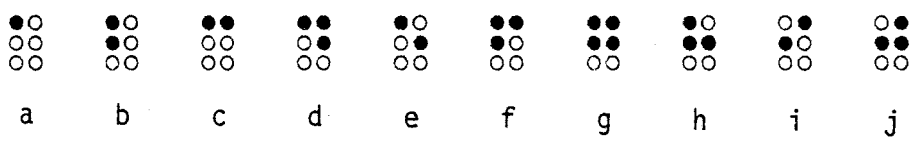
$$H(G) = \sum_{I,J} H(i,j) = 6 \text{ logons}$$

En ce qui concerne les dimensions et les caractéristiques physiques d'un caractère Braille, il n'y a pas vraiment de normalisation entre les différents pays du monde[1], cependant, une étude expérimentale menée par Meyers, Ethington et Ashcroft a montré que les dimensions données sur la figure 2(b) sont les meilleures pour la lecture.

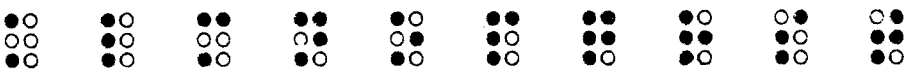
Pour une personne apprenant tardivement le Braille, il est très difficile de distinguer les caractères individuellement. La difficulté de l'apprentissage du Braille dépend essentiellement de l'âge du déficient visuel. En effet, peu de personnes ayant une vue résiduelle raisonnable utilisent le sens tactile pour lire le Braille. Un certain temps est donc nécessaire pour que le déficient visuel s'habitue à ce nouveau sens (peu familier).

Les 64 caractères de l'alphabet Braille peuvent représenter seuls les lettres alphabétiques, accentuées ou non, et les caractères de ponctuation.

Les dix premières lettres de l'alphabet sont composées par les points 1, 2, 4 et 5.

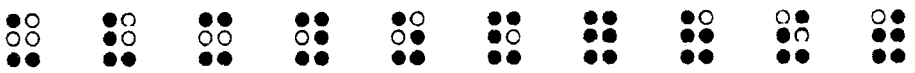


Une deuxième série de lettres est obtenue en ajoutant le point 3 à la série précédente.



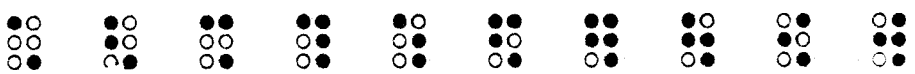
k l m n o p q r s t

Une troisième série est obtenue en ajoutant le point 6 à la série précédente.



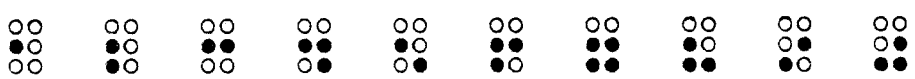
u v x y z ç é à è ù

Une quatrième série est obtenue en ajoutant le point 6 à la première série.



â ê î ô û ë ï ü œ w

La ponctuation s'indique par les premières lettres de la première série que l'on écrit en position inférieure.



, ; : . ? ! ( ) « * »

II-3- LE BRAILLE INTEGRAL

Pour écrire un texte en noir, compte tenu des signes de ponctuation, les majuscules, les minuscules, les chiffres, les signes arithmétiques et divers, il faut plus de 64 caractères distincts. Il en résulte que pour représenter tous les caractères de la syntaxe "noir", avec ceux de l'alphabet Braille, il faut utiliser des combinaisons particulières de ces derniers.

C'est ainsi que pour représenter les chiffres, on se sert des dix premiers graphèmes de l'alphabet Braille précédés d'un graphème spécifique appelé signe numérique, avec des conventions particulières pour représenter les nombres (partie entière, partie décimale) (figure 3).

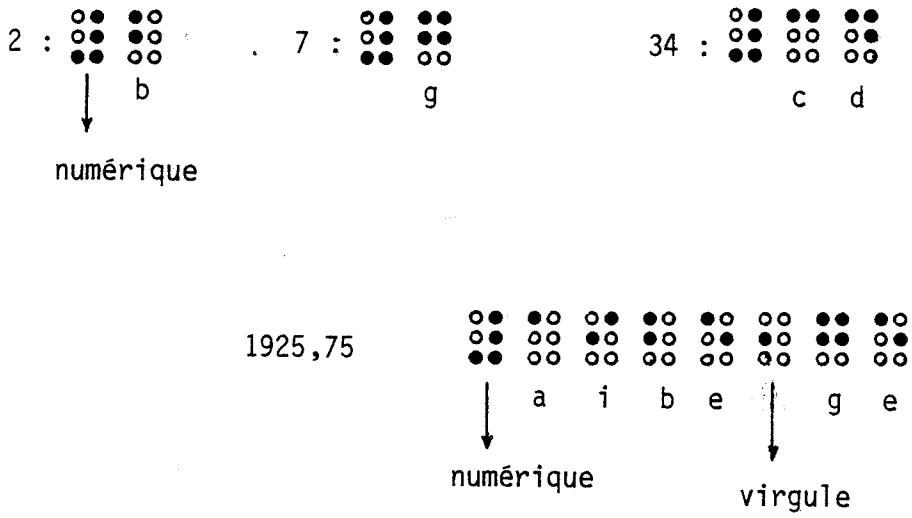


Figure 3

D'autres conventions sont utilisées pour représenter les méthodes de mise en valeur de partie de texte (par exemple, les mots soulignés, mots en italique) ou pour représenter des signes de syntaxe noir (c'est le cas de la représentation du %, ‰ , degré, numéro. Ces représentations seront décrites en détail dans le paragraphe IV qui représente les solutions choisies pour permettre la traduction automatique.

On remarque alors, compte tenu des dimensions et du faible nombre des caractères Braille, qu'un texte en Braille intégral occupera dans un document plus de place qu'un texte noir (on compte en général 3 pages de Braille intégral pour une page de texte noir). Ce procédé conduit à la génération d'ouvrages très volumineux et très lourds. Pour dévier cet inconvénient, on a cherché à réduire le nombre de caractères représentant un mot sans altérer la reconnaissance de sa signification. C'est pour atteindre cet objectif que le principe du Braille abrégé a été mis au point.

II -4- LE BRAILLE ABREGE

Le Braille abrégé a été créé par Maurice de la SIZERANNE en 1882. A un signe Braille correspond deux ou plusieurs caractères noirs. Il en résulte qu'à une page de texte noir correspond environ deux pages de Braille abrégé, d'où une diminution de volume par rapport au Braille intégral de 30 % en moyenne. Les avantages incontestables du Braille abrégé sont donc :

- rapidité de lecture et d'écriture,
- réduction de volume des ouvrages.

Il faut cependant noter que le Braille intégral est indispensable, car c'est la syntaxe utilisée jusqu'à l'âge de 12 ans (apprentissage de l'orthographe), et il est nécessaire à l'apprentissage du Braille abrégé.
D'autre part, parmi les 40.000 aveugles en France, seulement 1/3 connaissent le Braille abrégé.

Pour transcrire correctement un texte noir en Braille abrégé, il faut connaître :
- les 63 signes Braille,
- un ensemble de mots et locutions préabrégés appartenant à un dictionnaire,
- un ensemble de contractions associées à leurs règles d'utilisation,
- des règles de présentation d'un texte Braille.

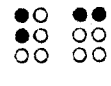
Ces ensembles sont décrits dans un document publié par l'Association Valentin Hauÿ dont l'édition la plus récente date à notre connaissance de 1964. Le contenu de cette édition nouvelle succède à "l'extension de l'abrégé orthographique français" qui date de 1924. Cette édition est le résultat des travaux d'une commission internationale qui s'est réunie en 1950 à l'American Foundation for Overseas Blind. Ce document sert à l'heure actuelle de référence en matière de transcription Braille.

L'ensemble des mots et locutions préabrégés constitue un dictionnaire qui comprend :
- 43 mots représentés par un seul caractère Braille (exemple : au, ai, bien, celui, cet etc.)
- 43 locutions représentées chacune par un seul mot Braille telles que :
  la plupart, pour ainsi dire, c'est-à-dire, parce que ...
- Environ 800 mots préabrégés et qui sont formés de "mots clés"
  exemple : avec → "ac"
  alors→ "al"
  afin → "af"

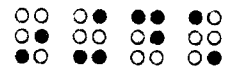
L'ensemble des contractions et les règles de leur utilisation que nous avons reproduites en Annexe est organisé pour faciliter la lecture du Braille et non en vue d'une transcription automatique.

Nous avons fait une lecture originale[5][23] de ces documents en organisant l'index en vue de la transcription automatique et confronté nos points de vue avec divers spécialistes du Braille en France qui nous ont aidé et conseillé pour éclaircir les points particuliers qui restaient obscurs.

Pour fixer les idées, les deux exemples ci-dessous montrent comment apparaissent les séquences de caractères Braille abrégé de deux mots "noirs" l'un appartenant à la liste des mots du dictionnaire, l'autre devant être décomposé en contractions.

"beaucoup" :  Appartient à la liste des mots du dictionnaire

b c

"inondation" :  Décomposé en contractions

(in)(on)(d)(ation)

III - DEFINITION DES FONCTIONS D'UN TRANSCRIPTEUR AUTONOME BRAILLE DE LA SAISIE AU CLAVIER DE TEXTES "NOIRS"

Le point de départ de cette étude est la nécessité d'éditer des textes en Braille, comme les secrétariats éditent des textes noirs sur machine à écrire. L'idée de base consiste donc à concevoir une machine type traitement de textes qui permet d'effectuer sur le texte noir la composition du texte Braille.
Il faut pour cela visualiser à la fois le texte noir saisi et les informations qui définissent la composition du texte Braille (centrage des titres, saut de lignes, saut de page, coupure des mots en fin de ligne etc.) . Le format du texte Braille est imposé par celui des ouvrages édités. Il est variable suivant les éditeurs mais les formats les plus couramment utilisés sont au nombre de trois ou quatre. On peut citer le format de l'embosseuse SAGEM (31 caractères par ligne et 28 lignes par page), le format MIXTE (29 caractères par ligne, 23 lignes par page), le format IN-OCTAVO (21 caractères par ligne et 22 lignes par page). D'autres formats peuvent être souhaités. Par ailleurs, le Braille intégral occupe en moyenne plus de place que le texte noir. En Braille intégral, la coupure des mots en fin de ligne se fait de la même façon qu'en noir. On voit donc que l'utilisateur doit être renseigné sur le moment où il doit faire la coupure du mot Braille et passer à la ligne, ou bien l'automate doit exécuter automatiquement cette coupure sans que l'utilisateur ait à s'en occuper. Dans ce dernier cas, cela implique de concevoir un logiciel de coupure automatique aux syllabes du texte Braille, et un logiciel qui effectue automatiquement la composition de ce même texte. En ce qui concerne la coupure des mots en fin de ligne, le centrage des titres, le saut de ligne, la pagination des textes noir et Braille, il n'y a pas de difficulté majeure pour concevoir ce logiciel. Par contre, sa mise au point est fastidieuse et le programme occuperait en mémoire une place au moins égale à 4 k octets environ. Enfin quelle que soit la complexité du logiciel, l'indépendance totale de la composition des textes noir et Braille, poserait à terme des difficultés considérables.

Par contre, la première méthode conduit à une solution techniquement très simple et beaucoup plus élégante. Elle consiste à visualiser le remplissage

du texte Braille sur une ligne. Ainsi, l'utilisateur peut, en observant la ligne Braille décider de la coupure des mots du texte noir au moment voulu. Cette solution, non contraignante pour l'utilisateur, a l'avantage de ne nécessiter aucun logiciel compliqué et par conséquent d'économiser au moins 4 k octets de programme. Tous les problèmes de composition de texte Braille sont résolus parce qu'on édite directement dans le format du Braille intégral et de plus, cela permet, au niveau de stockage des textes (noir, Braille intégral, Braille abrégé) édités d'effectuer une correspondance page à page.

Cette correspondance est avantageuse pour retrouver un texte en Braille abrégé car il est difficilement compréhensible pour un voyant.

Etant donné que le format du Braille intégral est variable, il faut pouvoir représenter les textes sur toute l'étendue de l'écran de visualisation quel que soit le format d'édition choisi.

Il est toujours possible d'utiliser des architectures de gestion de moniteur de visualisation comme il en existe dans tous les microordinateurs d'usage général. Mais ces architectures sont prévues pour visualiser des textes dans un format fixe et ne sont pas adaptées nécessairement à notre problème. Par exemple, avec des proportions courantes de caractères et avec un format de 30 caractères par ligne, en visualisant 16 lignes pour éditer une page de texte de 29 lignes sachant qu'il faut réserver au moins une ligne de remplissage Braille, il est nécessaire de faire défiler le texte noir alors que la ligne de remplissage Braille doit rester fixe.

Pour satisfaire toutes les contraintes imposées par les problèmes particuliers de la transcription simultanée dans une autre syntaxe que celle du texte noir, nous avons choisi la solution qui consiste à concevoir une architecture spécifique de visualisation de texte à format variable sur écran.

Ce choix, outre la souplesse qu'il confère, nous permet d'imaginer de nouvelles applications de la visualisation de textes à usage des amblyopes et est générateur de progrès dans le domaine de l'aide aux handicapés visuels.

Nous allons, dans un premier temps, décrire l'ensemble des solutions adoptées pour réaliser cet automate. Pour cela, nous allons d'abord énumérer les particularités de la syntaxe Braille intégral. Un tel problème nous amène à appliquer un raisonnement mathématique particulier. Après avoir donné quelques définitions mathématiques, nous décrirons le raisonnement qui conduit à la définition des informations qu'il faut inscrire sur l'écran pour permettre de lever des ambiguités de traduction. Nous décrirons alors l'architecture de visualisation et le logiciel mis au point pour parvenir à la réalisation de traduction automatique en Braille intégral.

Dans la deuxième partie nous décrirons en les justifiant les méthodes employées pour traduire en Braille abrégé sur microordinateur. Nous donnons les algorithmes, les organigrammes et les calculs du temps de traitement dans le cas où un mot appartient au dictionnaire et lorsqu'il doit être décomposé en contractions.

IV - ENUMERATION DES PARTICULARITES DE LA SYNTAXE BRAILLE INTEGRAL PAR RAPPORT A CELLE PREVUE PAR L'EMPLOI D'UN CLAVIER AZERTY

IV-1- DESCRIPTION DU CODAGE CLAVIER

Nous avons utilisé un clavier AZERTY, géré par l'unité centrale à travers un interface P.I.A. (voir 3ème partie)

La mémoire EPROM du clavier contient des codes correspondants à la position de chaque touche. Ce code ne dépend que de la position de la touche sans représenter la signification du caractère inscrit. Il est appelé code position de la touche. L'intérêt de ce procédé est d'affecter plusieurs codes pour l'enfoncement d'une même touche.
Exemple : le clavier AZERTY ne possède qu'une touche pour l'ouverture et la fermeture de guillemets. Mais, en Braille, ces deux touches sont codées différemment.

guillemets ouverts : 

guillemets fermés : 

Lorsque la touche "guillemets" a été enfoncée, il est possible de concevoir un logiciel de gestion du clavier pour qu'à partir de la connaissance des informations contenues dans le texte précédemment entré, il soit fourni au logiciel de transcription Braille des codes différents suivant qu'il s'agisse de l'ouverture ou de la fermeture de guillemets.

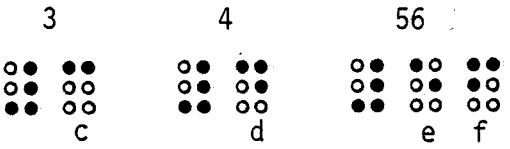
Nous allons énumérer certains problèmes qui se posent pour la transcription en Braille et qui sont liés à la fois à l'utilisation du clavier et à la visualisation du texte sur l'écran.

IV-2- LES CHIFFRES

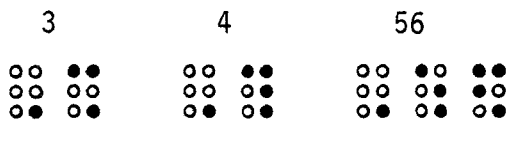
Un chiffre noir est transcrit par deux symboles Braille : un signe numérique et un signe correspondant au chiffre (l'une des dix premières lettres de l'alphabet). Mais lorsqu'un nombre est formé de plusieurs chiffres, on ne met le signe numérique que devant le premier de ces chiffres. Remarquons qu'une notation ϕ, â, ê, î, ô, û, ë, ï ü, œ est actuellement la plus employée pour écrire les chiffres des nombres.

Exemple :

ancienne notation



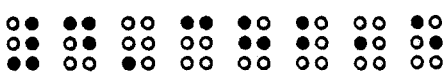
nouvelle notation



Comme en noir, la partie entière est séparée de la partie décimale par une virgule et le caractère spécifique numérique ne doit pas être généré devant la partie décimale. Par ailleurs, l'espace utilisé pour séparer les milliers est transcrit en Braille par le symbole correspondant à l'apostrophe.

Exemple :

4 382,5



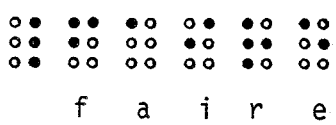
La solution consiste à charger le logiciel de transcription de ne pas provoquer le symbole spécifique numérique devant la partie décimale quand la virgule est encadrée par des chiffres. De même, l'espace qui sépare les tranches de trois chiffres doit être codé comme apostrophe lorsqu'il est encadré par des chiffres.

IV-3- LES MAJUSCULES

En Braille, il n'existe pas deux signes différents pour la même lettre noir écrite en majuscule ou en minuscule. Lorsqu'un mot est écrit en majuscule il est précédé d'un signe spécifique majuscule et tous les caractères du mot sont transcrits en minuscule.

Exemple :

"Faire"



majuscule

Une ambiguité existe cependant dans le cas où une lettre "noire" écrite en majuscule est une voyelle (celle-ci pouvant être accentuée ou pas).
Exemple : si une phrase commence par le mot "être", en "noir" on peut écrire "Etre" et le "ê" sera transcrit en Braille comme un "e",ce qui constitue une erreur de transcription. Il faut donc que l'opérateur qui n'est pas sensé connaître le Braille apporte une information complémentaire pour que la transcription soit correcte. Nous avons choisi la solution qui consiste à inscrire sur l'écran de visualisation du texte un commentaire accompagné d'un signal sonore.

Le signal sonore avertit l'opérateur de la présence d'une ambiguité de traduction. Le commentaire précise à l'opérateur la manoeuvre à réaliser pour apporter l'information manquante nécessaire à la transcription. Cette possibilité de demande d'apport d'information complémentaire par l'opérateur sera exploitée pour la transcription en Braille abrégé chaque fois qu'il sera nécessaire de lever une ambiguité.

Pour être capable de formuler le commentaire le plus clairement possible à l'opérateur, nous avons réservé deux lignes de texte visualisées à cette fonction.

IV-4- LE SOULIGNE

On utilise deux codages différents suivant que l'on veut souligner un groupe de moins de trois mots ou de plus de trois mots.

Groupe de moins de trois mots :
Chaque mot souligné doit être précédé du signe Braille : ⠐

Groupe de plus de trois mots :
Le premier mot doit être précédé des deux signes Braille ⠐⠐

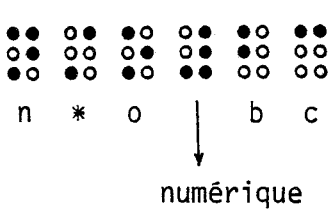
et le dernier mot du groupe doit être précédé du signe : ⠐

Là encore, nous avons choisi la solution qui consiste à utiliser deux touches différentes car cela n'altère pas le mode d'utilisation d'un clavier de machine à écrire. La visualisation sur l'écran du souligné se fait par inversion du signal vidéo des codes de caractères du ou des mots soulignés (l'inversion du signal vidéo se fait en mettant le bit 7 du code ASCII de chaque caractère à 1 ; ainsi le caractère sera visualisé en blanc sur fond noir quand le texte est visualisé en noir sur fond blanc).

IV-5- DEGRE ET NUMERO

Le symbole (°) est souvent utilisé pour désigner soit le degré, ex : 15°C, soit le numéro, ex : n° 15.

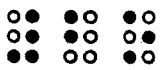
Cas du numéro :
exemple : n° 23



Le signe (°) est codé par deux signes Braille. Dans ce cas, le signe (°) suitla lettre n.

Cas du degré :

Le signe (°) suit un chiffre et il est codé par le symbole Braille correspondant à la lettre "O" .

exemple : 2°C 

Pour la transcription, il faut pouvoir distinguer les 2 significations dans un texte. Deux solutions sont possibles : concevoir un logiciel pouvant faire la distinction ou disposer de deux touches.
Dans ce cas, nous avons choisi la solution de la conception d'un logiciel qui génère l'une ou l'autre séquence de graphèmes suivant que la touche (°) est enfoncée après la frappe de la touche (n) ou après la frappe de la touche numérique.

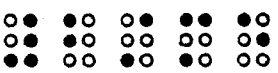
IV-6 - CAS DU Nième

Dans ce cas, comme dans celui du degré, numéro et chiffre etc. , pour transcrire un caractère en Braille intégral il faut tenir compte de son contexte c'est-à-dire des caractères qui le précèdent et de ceux qui le suivent.

exemple :

1er 

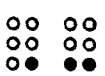
La chaîne de caractères "er" est précédée d'un chiffre.

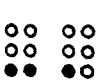
2me 

La chaîne de caractères "me" est précédée d'un chiffre.

IV-7- TIRET INITIAL , TIRET FINAL

En Braille, le tiret initial et le tiret final sont codés différemment.

Tiret initial 

Tiret final 

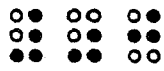
deux signes Braille pour un signe noir

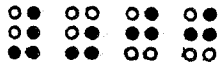
Dans la transcription, il faut donc pouvoir les distinguer dans un texte. Deux solutions sont possibles :
- concevoir un logiciel pouvant faire la distinction automatiquement. Mais, il faut connaître l'état (initial ou final) du dernier tiret entré au clavier, ce qui pose un problème lors de la reprise d'un travail.
- Disposer de deux touches sur le clavier correspondant l'un au tiret initial, l'autre au tiret final.

C'est cette deuxième solution qui a été finalement adoptée parce qu'elle ne demande pas de traitement spécial pour le cas de la reprise d'un travail ni de contraintes particulières de la part de l'utilisateur.

IV-8- CAS DU % ET ‰

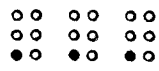
Le % se traduit par 4 signes et le ‰ par 5 signes Braille.

% 

‰ 

IV-9- POINTS DE SUSPENSION

Les trois points de suspension sont traduits en Braille par trois apostrophes :



Pour générer les trois apostrophes, deux solutions sont possibles :
- traiter les points de suspension comme une chaîne de caractères,
- disposer sur le clavier d'une touche spéciale "points de suspension ", et dans ce cas elle sera considérée comme un séparateur se traduisant par trois signes Braille.

C'est cette deuxième solution qui a été adoptée pour avoir un logiciel de transcription Braille intégral le plus simple possible.

V - JUSTIFICATION DE L'INFORMATION A APPORTER A L'OPERATEUR POUR EFFECTUER LA COUPURE DES MOTS EN FIN DE LIGNE

Pour développer cette justification, nous allons rattacher à l'information entrée au clavier quelques grandeurs mathématiques dont les propriétés conduisent à une méthode simple de coupure transparente des mots en fin de ligne du texte Braille intégral.

V-1- DEFINITION DES GRANDEURS MATHEMATIQUES A INTRODUIRE

Nous allons ici introduire quelques définitions qui seront utiles pour justifier les solutions adoptées.

V-1-1- Vocabulaire

On appellera dans ce qui suit vocabulaire noir un ensemble fini V contenant tous les caractères "noirs"

$$V = \{ \text{␣}, A, B, C, \ldots\ldots Z \ldots\ldots \}$$

et B un ensemble fini appelé vocabulaire Braille contenant tous les caractères Braille

$$B = \{ \text{⠀ ⠁ ⠂ ⠃ ⠄ ⠅ ⠆ ⠇ ⠈ ⠉} \ldots\ldots \}$$

V-1-2- Chaîne

On appellera chaîne sur V ou sur B une suite finie de symboles de V ou de B

$v_1\ v_2 \ldots\ldots v_n$ avec $v_i \in V$

$b_1\ b_2 \ldots\ldots b_n$ avec $b_i \in B$

Pratiquement, une chaîne sera représentée par un ou plusieurs mots noir ou Braille.

On désigne par V^* : l'ensemble de toutes les chaînes que l'on peut construire sur V, y compris la chaîne vide notée "$\wedge$" .

On désigne aussi l'ensemble $V^+ = V^* - \{\wedge\}$

On définit de la même façon B^* et B^+

L'ensemble V est muni d'une relation d'ordre notée "< " et nous prenons la convention :

$$v_i < v_j \Longleftrightarrow \text{code ASCII}(v_i) < \text{code ASCII}(v_j)$$

Nous définissons une loi de composition interne notée "*" et qui est la concaténation de deux ou plusieurs éléments de V^*

Soient deux chaînes $\alpha, \beta \in V^*$ telles que :

$$\alpha = v_1\ v_2\ \ldots\ v_n$$

$$\beta = u_1\ u_2\ \ldots\ u_m$$

alors $\alpha * \beta = v_1\ v_2\ \ldots\ v_n\ u_1\ u_2\ \ldots\ u_m$

Pour une simplification d'écriture, nous noterons :

$$\alpha * \beta = \alpha\beta$$

De plus, on notera la concaténation d'une chaîne λ avec tous les éléments d'un ensemble de chaînes de la façon suivante :
soit un ensemble de chaînes $E = \{ \alpha_1 , \alpha_2 , \ldots \alpha_n \}$
avec $\forall i \ \alpha_i \in V^*$ et tel que : $\alpha_i = \beta_i \lambda$

on écrira E sous la forme : $E = \{ \beta_1 \lambda, \beta_2 \lambda, \ldots \beta_n \lambda \}$

$$= \{ \beta_1 , \beta_2 , \ldots \beta_n \} \lambda$$

V-1-3- Longueur d'une chaîne

Nous appellerons $\ell(\alpha)$ la longueur d'une chaîne α . Cela consiste en une application de V^* dans N (ensemble des entiers naturels).

Les propriétés de cette application peuvent s'énoncer de la manière suivante :

Si $\alpha = \wedge$: $\ell(\alpha) = 0$

si non, si $\alpha = a_1 \phi$: $\ell(\alpha) = 1 + \ell(\phi)$

où a_1 est un caractère quelconque du vocabulaire noir V ou Braille B
ϕ est une chaîne noir ou Braille

V-2- JUSTIFICATION DE LA METHODE DE COUPURE DES MOTS EN FIN DE LIGNE DE TEXTE BRAILLE

L'étude de la transcription en Braille intégral à partir de l'édition d'un texte noir conduit à noter la transcription par l'application T_i .

Si l'on a à transcrire la chaîne $\alpha \in V^*$ où $\alpha = v_1\ v_2\ \ldots\ v_m$, on écrit $T_i(\alpha) = T_i(v_1\ v_2\ \ldots\ v_m)$.

Etant donné qu'un mot noir traduit en Braille intégral revient à concaténer la traduction de chaque lettre du mot noir, on a :

$$T_i(v_1\ v_2\ \ldots\ v_m) = T_i(v_1)\ T_i(v_2)\ \ldots\ T_i(v_m)$$

On montre qu'un caractère noir doit être traduit par un ou plusieurs caractères Braille donc :

$$\ell(T_i(v_i)\ ) \geqslant \ell(v_i) \qquad \forall\ v_i \in V$$

avec $\ell(v_i) = 1$

et donc si $\beta = T_i(\alpha)$ avec $\beta \in B^*$

on a $\ell(\beta) \geqslant \ell(\alpha)$

Si on appelle N le nombre de caractères Braille possible par ligne de texte, il s'agit de déterminer quel est le nombre de caractères de la même ligne noir .

Donc, soit $m_1\ m_2\ \ldots\ m_n$ avec $m_i \in V^*$: mots noir

tel que :

$$\ell(T_i(m_1\ m_2\ \ldots\ m_n)) \leqslant N$$

$$\ell(T_i(m_1)\ T_i(m_2)\ \ldots\ T_i(m_n)) \leqslant N$$

où $\ell(T_i(m_1)) + \ell(T_i(m_2)) + \ldots + \ell(T_i(m_n)) \leqslant N$

$$\ell(T_i(m_n)) \leqslant N - \sum_{j=1}^{n-1} \ell(T_i(m_j)) \qquad (**)$$

La longueur de la traduction du dernier mot noir m_n est donnée par

$\ell(T_i(m_n)) = \ell(T_i(v1) + \ldots. + \ell(T_i(v_m)\ )$

avec $m_n = v_1\ v_2\ \ldots\ v_m$

Pour avoir une ligne Braille complète de N caractères, il faudrait donc que :

$$\ell(T_i(m_n)) = N - \sum_{j-1}^{n-1} \ell(T_i(m_j))$$

si $$\ell(T_i(m_n)) > N - \sum_{j=1}^{n-1} \ell(T_i(m_j))$$

il faut faire une coupure du dernier mot noir m_n de manière à satisfaire l'égalité (**)

Il faut donc que l'opérateur n'ayant aucune connaissance du Braille et par conséquent aucune indication sur la longueur Braille des chaînes qu'il compose au clavier, puisse couper le dernier mot de la chaîne noir de façon à satisfaire l'égalité (**)

Tout au long de l'édition du texte noir, l'opérateur dispose d'une ligne supplémentaire composée de N carrés noirs sur blanc . Chaque caractère $v_j \in V$ introduit au clavier est traduit en Braille intégral. Connaissant v_j , $T_i(v_j)$ et donc $\ell(T_i(v_j))$, il suffit alors de concevoir le logiciel pour qu'il provoque l'effacement de $\ell(T_i(v_j))$ carrés noir sur blanc. L'opérateur sera ainsi informé à chaque instant du remplissage de la ligne Braille.

$\sum_{j=1}^{n-1} \ell(T_i(m_j))$ $\qquad$ $N - \sum_{j=1}^{n-1} \ell(T_i(m_j))$

VI - DEFINITION ET DESCRIPTION DE LA VISUALISATION DE L'EDITION SUR L'ECRAN

Nous venons de montrer les diverses informations à visualiser pour permettre une traduction transparente.

Ces solutions se résument à faire apparaître sur l'écran trois types d'informations :

1) Un commentaire de communication. Ce commentaire sert à lever certains cas d'ambiguité. Pour présenter un commentaire suffisamment clair, il faut lui réserver deux lignes de texte.

2) Une ligne de symboles de remplissage de la ligne Braille. Cette ligne sert à l'opérateur pour effectuer la coupure correcte entre les syllabes des mots du texte noir en bout de ligne du texte Braille intégral.

3) Le texte noir édité. Ce texte noir doit apparaître avec un format identique à celui de la reproduction du Braille intégral. Il doit pouvoir évoluer entre 21 et 29 caractères par ligne et entre 24 et 28 lignes par page.

Pour satisfaire les contraintes du nombre de caractères par ligne, tout en assurant une forme de caractères qui permet la lecture aisée, on ne peut visualiser plus de 16 lignes en tout, c'est-à-dire 13 lignes de texte. Etant donné qu'une page de texte Braille comporte entre 21 et 29 lignes, il faudra faire défiler les lignes de texte de la page en cours d'édition.

Dans ce paragraphe, nous décrivons le principe de l'architecture de la génération de l'image et celui du logiciel de visualisation.

VI-1- DESCRIPTION DU PRINCIPE DE LA GENERATION DE L'IMAGE SUR L'ECRAN

Ce principe est celui de la visualisation de caractères alphanumériques qui consiste à balayer l'écran par rafraîchissement d'une zone mémoire appelée mémoire écran. Cette mémoire contient les codes des caractères à visualiser et qui représentent les adresses des caractères contenues dans une mémoire générateur de caractères.

L'architecture de gestion de l'écran est construite autour d'un circuit intégré synthétiseur programmable de signaux vidéofréquence type MC 6845.

Le système fonctionne suivant le principe illustré par le schéma synoptique figure 4.

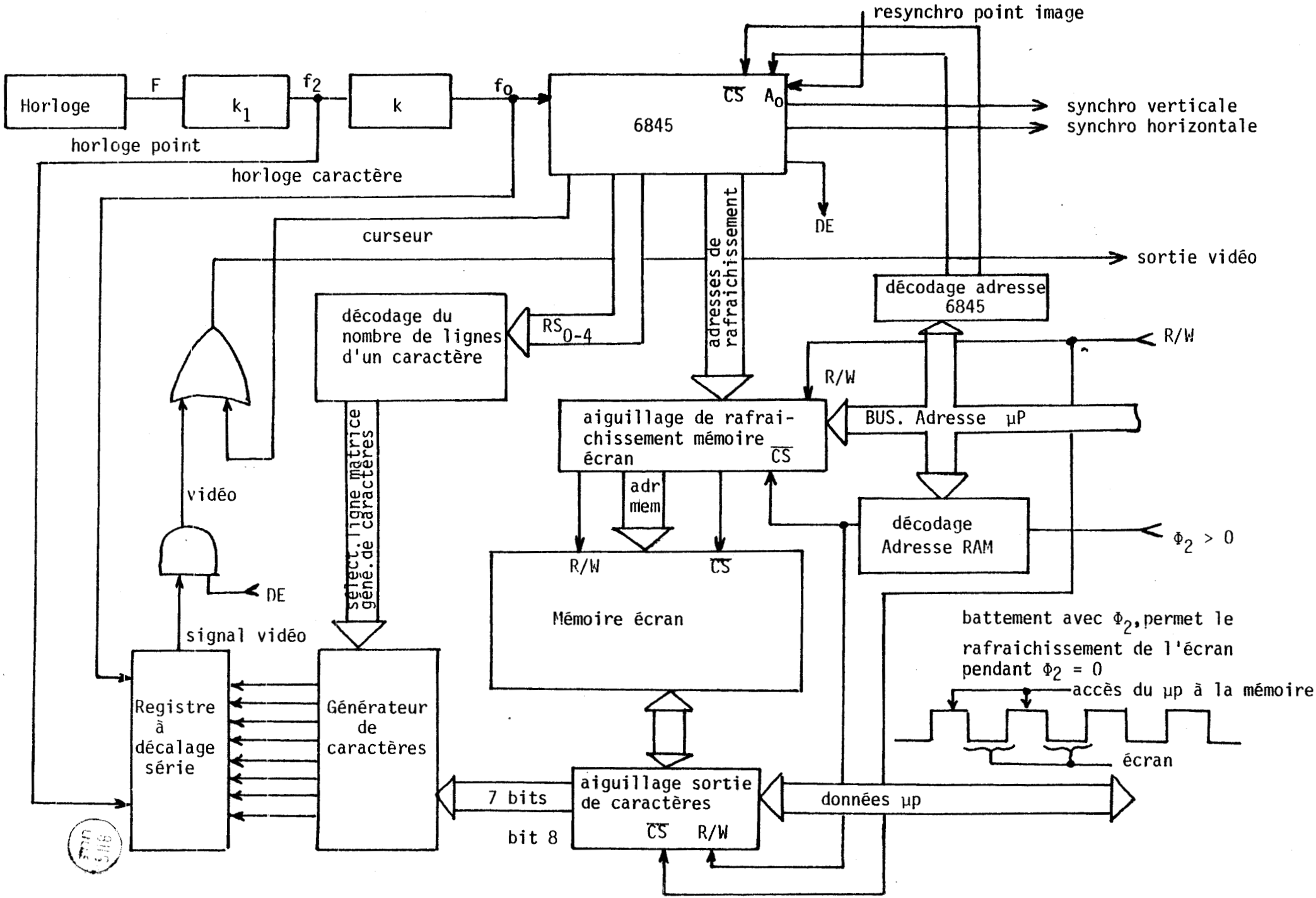

resynchro point image
Horloge
F
k_1
f_2
k
f_o
6845
$\overline{CS}$ A_o
synchro verticale
synchro horizontale
horloge point
horloge caractère
DE
curseur
adresses de rafraichissement
sortie vidéo
décodage adresse 6845
décodage du nombre de lignes d'un caractère
RS_{0-4}
R/W
R/W
aiguillage de rafraichissement mémoire écran $\overline{CS}$
BUS. Adresse µP
vidéo
sélect. ligne matrice géné. de caractères
adr mem
décodage Adresse RAM
$\Phi_2 > 0$
DE
R/W
$\overline{CS}$
Mémoire écran
signal vidéo
battement avec Φ_2, permet le rafraichissement de l'écran pendant $\Phi_2 = 0$
accès du µp à la mémoire
écran
Registre à décalage série
Générateur de caractères
7 bits
aiguillage sortie de caractères
données µp
bit 8
$\overline{CS}$ R/W

Figure 4

Le synthétiseur balaye en permanence la mémoire écran qui contient les codes ASCII des caractères visualisés. Cette mémoire écran est à double accès. Elle peut être adressée soit à partir des lignes d'adresse de rafraîchissement du synthétiseur, soit à partir des lignes d'adresse du microprocesseur qui sert à l'édition de texte et à la traduction.

Le rafraîchissement du synthétiseur assure l'écriture du texte sur l'écran ; l'adressage par le microprocesseur sert à modifier les caractères visualisés sur l'écran dont les codes sont entrés au clavier. Ainsi, chaque cellule mémoire écran sert à adresser indirectement un caractère à visualiser.

Si chaque caractère contenu dans la mémoire générateur de caractères est défini par n lignes, interligne compris, les N codes des caractères d'une même rangée seront sélectionnés n fois. Le rang de la ligne de points de chaque caractère est défini par un mot binaire apparaissant sur les lignes RS_{0-4} de 5 bits. La taille de ce bus permet donc de représenter une rangée de caractères, interligne compris, sur N = 32 lignes de trame de télévision.

La définition des matrices de caractères du générateur est donnée figure 5 et l'ensemble des matrices de l'alphabet occupe 2 k octets.

Le circuit synthétiseur de signaux vidéofréquence possède 18 registres dont les contenus désignés sur la figure 6 définissent la conformation de l'image visualisée sur l'écran.

Le principe de visualisation des caractères alphanumériques sur l'écran est classique, c'est celui qui est universellement utilisé sur les consoles. Mais, nous décrivons une utilisation originale du circuit synthétiseur de signaux vidéo fréquence qui consiste à commander ce synthétiseur avec des signaux qui produisent un effet de loupe sur du texte visualisé.

Les contenus des registres R_0 , R_1 , R_2 définissent le format d'une des lignes de balayage qui composent une rangée de caractères. L'unité binaire de leurs contenus représente le temps $T_0 = 1/f_0$ que met le point lumineux pour parcourir la largeur d'un caractère et l'espace qui le sépare du suivant.

Cette fréquence f_0 est celle du signal qui alimente normalement l'entrée horloge du contrôleur.

Si p est le nombre de points de définition de la largeur d'un caractère, f_2 la fréquence de l'horloge de commande du registre à décalage qui génère le signal vidéo-fréquence, le temps de parcours de la largeur d'un caractère T_c est donné par l'expression :

$$T_c = \frac{P}{f_2}$$

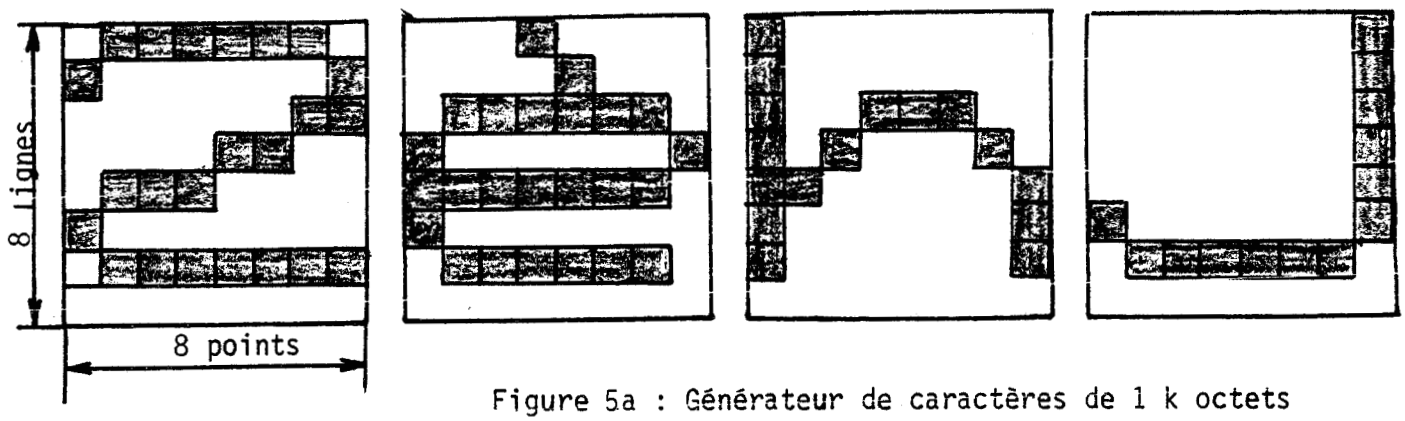


Figure 5a : Générateur de caractères de 1 k octets

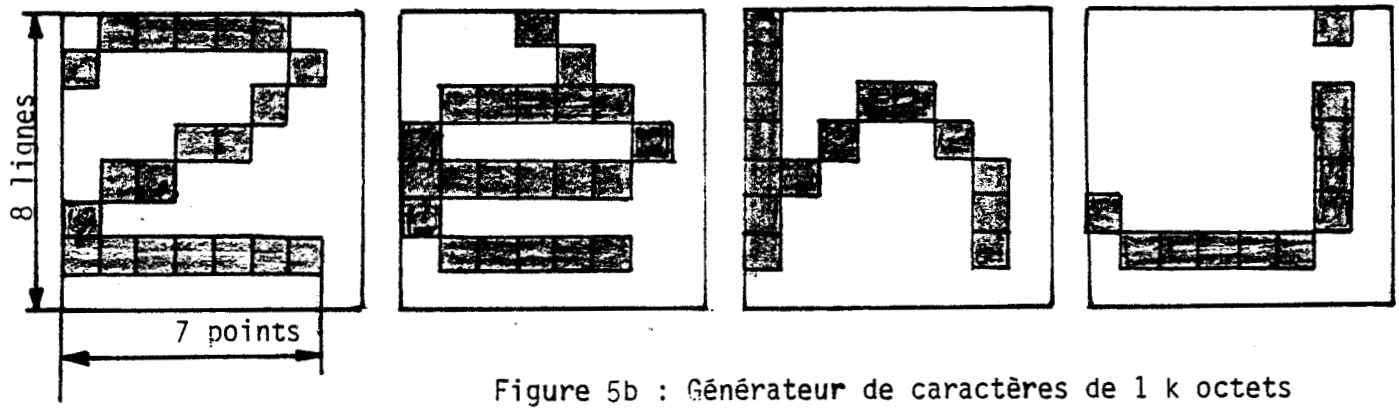


Figure 5b : Générateur de caractères de 1 k octets

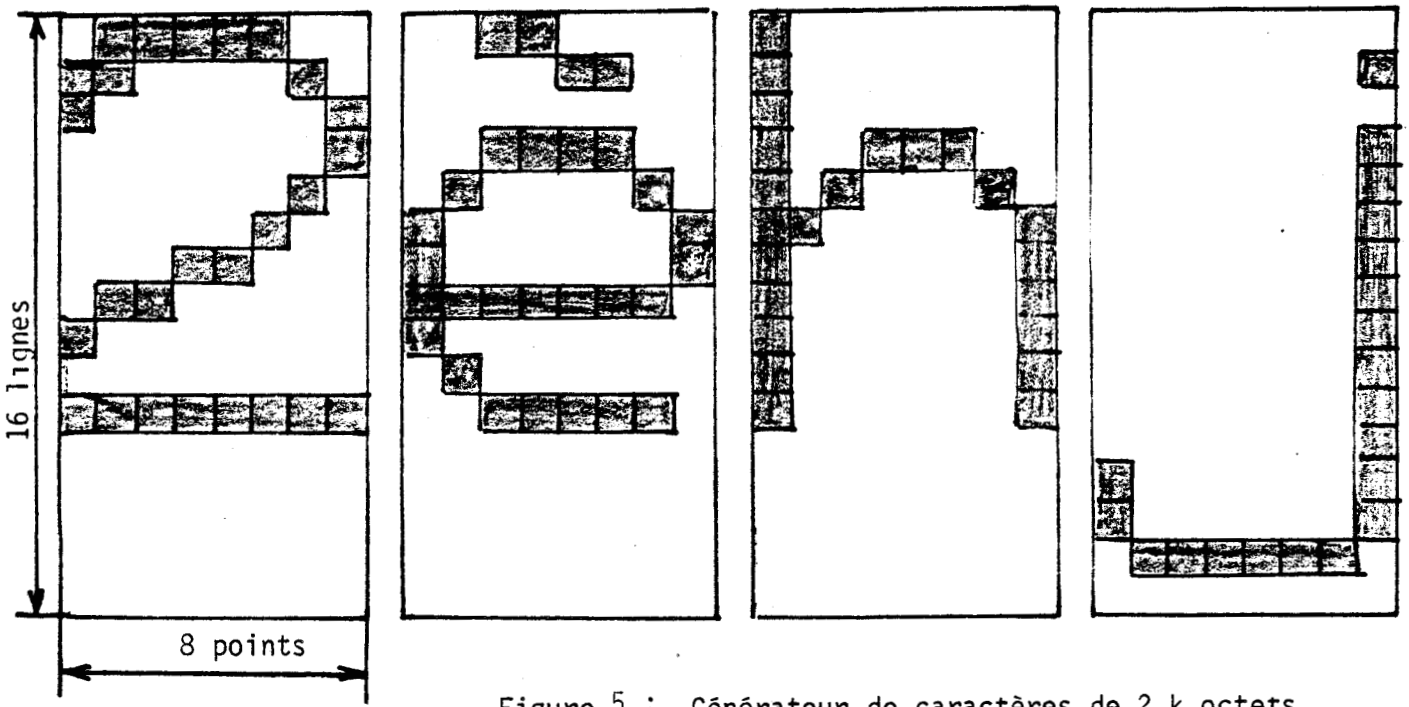


Figure 5 : Générateur de caractères de 2 k octets

N° du rang	rôle du registre	unité	accès lecture/écriture
R_0	Nbre de caractères total sur une ligne horizontale	caractères	E
R_1	Nbre de caractères visualisés sur une rangée	"	E
R_2	Position impulsion synchronisation ligne	"	E
R_3	Largeur impulsion de synchro ligne	"	E
R_4	Nbre de rangées par image	rangées	E
R_5	Nbre de lignes d'ajustement image	lignes	E
R_6	Nbre de rangées visualisées	rangées	E
R_7	Position impulsion Synchronisation vert	rangées	E
R_8	Mode interligne		E
R_9	Nombre maximum de lignes par rangée	lignes	E
$R_{10} - R_{11}$	Dimension curseur	lignes	
$R_{12} - R_{13}$	Adresse de départ de balayage mémoire	"	E
$R_{14} - R_{15}$	Position curseur	"	L/E
$R_{16} - R_{17}$	Adresse crayon lumineux	"	L

Figure 6

Si on pose $k = f_2/f_0$, l'espace sera élaboré en un temps égal à :

$$T_e = \frac{k - p}{k} \frac{1}{f_0}$$

Les deux expressions de T_c et T_e montrent que l'on peut faire varier séparément les largeurs des caractères et des espaces en modifiant séparément la fréquence f_2 et le rapport de division k. f_2 peut être obtenue à partir d'une fréquence fixe F par un diviseur de fréquence variable de rapport k_1 .
Les deux expressions ci-dessus deviennent alors :

$$T_c = \frac{p\ k_1}{F} \qquad \text{et} \qquad T_e = k_1\ (k - p)\ \frac{1}{F}$$

Les variations de T_c et T_e sont d'autant plus fines que F est élevé.

Le rapport k a deux limites qui sont celles qui font évoluer la largeur de l'espace entre une valeur nulle et la largeur d'un caractère

$$p \leqslant k \leqslant 2p$$

IV-2- REPRESENTATION DES TEXTES AVEC ESPACES ET INTERLIGNES VARIABLES

Le circuit de contrôle d'écran 6845 comporte parmi ses 18 registres programmables 6 registres qui permettent de définir le format et le cadrage du texte visualisé.

Le rôle de chacun de ces registres est décrit sur les diagrammes de la figure 7.

Chacun d'eux doit contenir un nombre entier qui représente une grandeur, dont l'unité est le nombre de caractères ou le nombre de rangées de caractères sur l'intervalle de temps qu'il définit.

Par exemple : $R_0 = T_1/T_0$ = nombre de caractères entre deux impulsions de synchronisation ligne

$R_4 + R_5$ = nombre de rangées de caractères entre deux impulsions de synchronisation image.

Le nombre de caractères visualisés par rangée (R_1) et le nombre de rangées de caractères par image (R_6) étant imposé, il s'agit de trouver la période $T_0 (T_0 = T_c + T_e)$ du signal d'horloge appliqué au contrôleur d'écran qui satisfait les contraintes de conformation des signaux.

Ces contraintes conduisent au système d'inéquations :

$(R_2 - R_1)\ T_0 > 1{,}4\ \mu s$

$(R_0 - R_2 - R_3)\ T_0 > 5{,}4\ \mu s$

$R_3\ T_0 > 4{,}8\ \mu s$

$60\ \mu s < R_0\ T_0 < 64\ \mu s$

Les solutions de ce système sont obtenues à l'aide d'un programme basé sur l'algorithme :

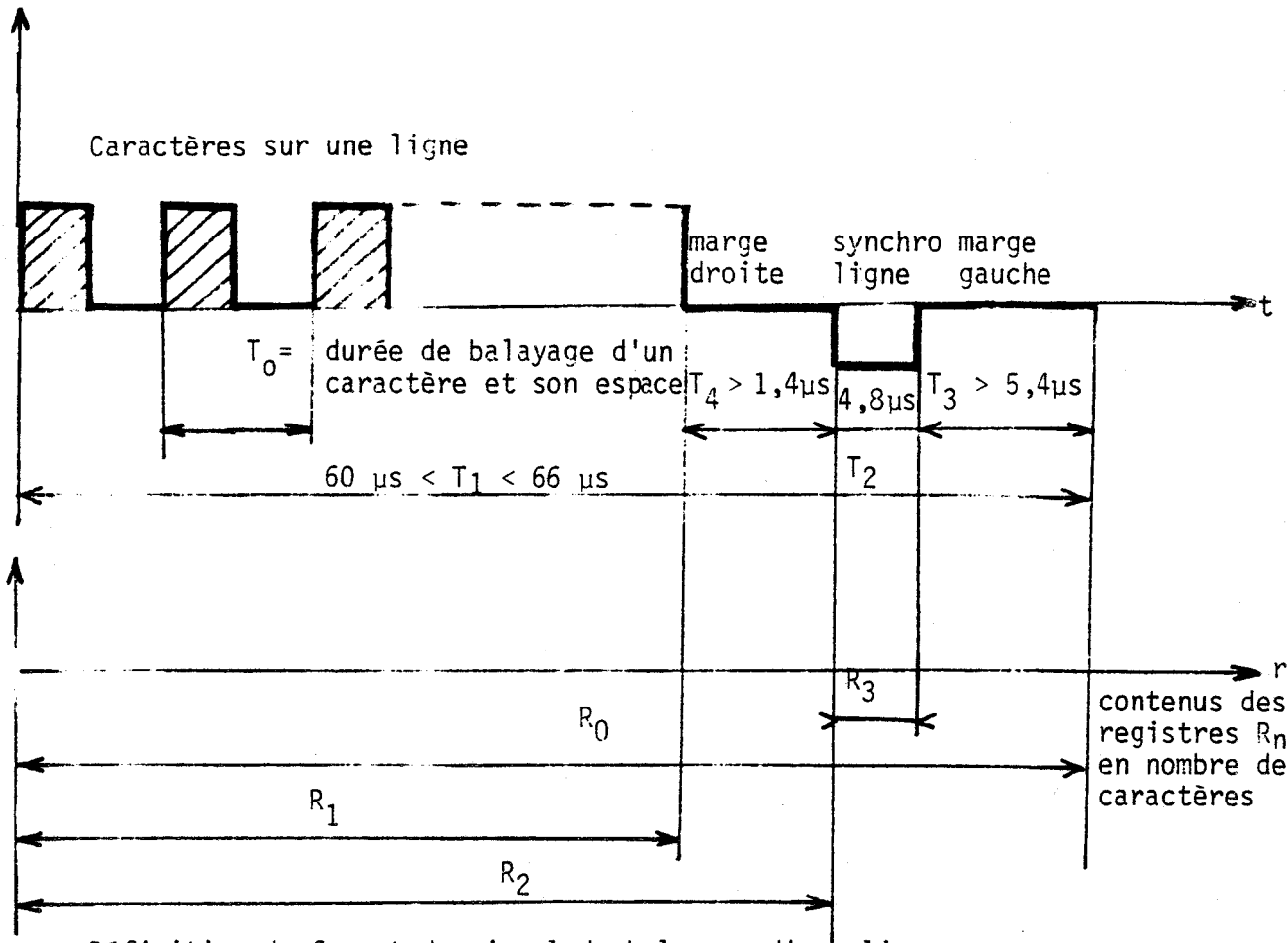
$$R0_{n+1} = R_1 + 2 + \text{Entier}\left(\frac{T_3+T_4}{T_1} R0_n\right) + \text{Entier}\left(\frac{T_2}{T_1} R0_n + 0{,}5\right)$$

On arrête l'algorithme lorsque $R0_{n+1} = R0_n$

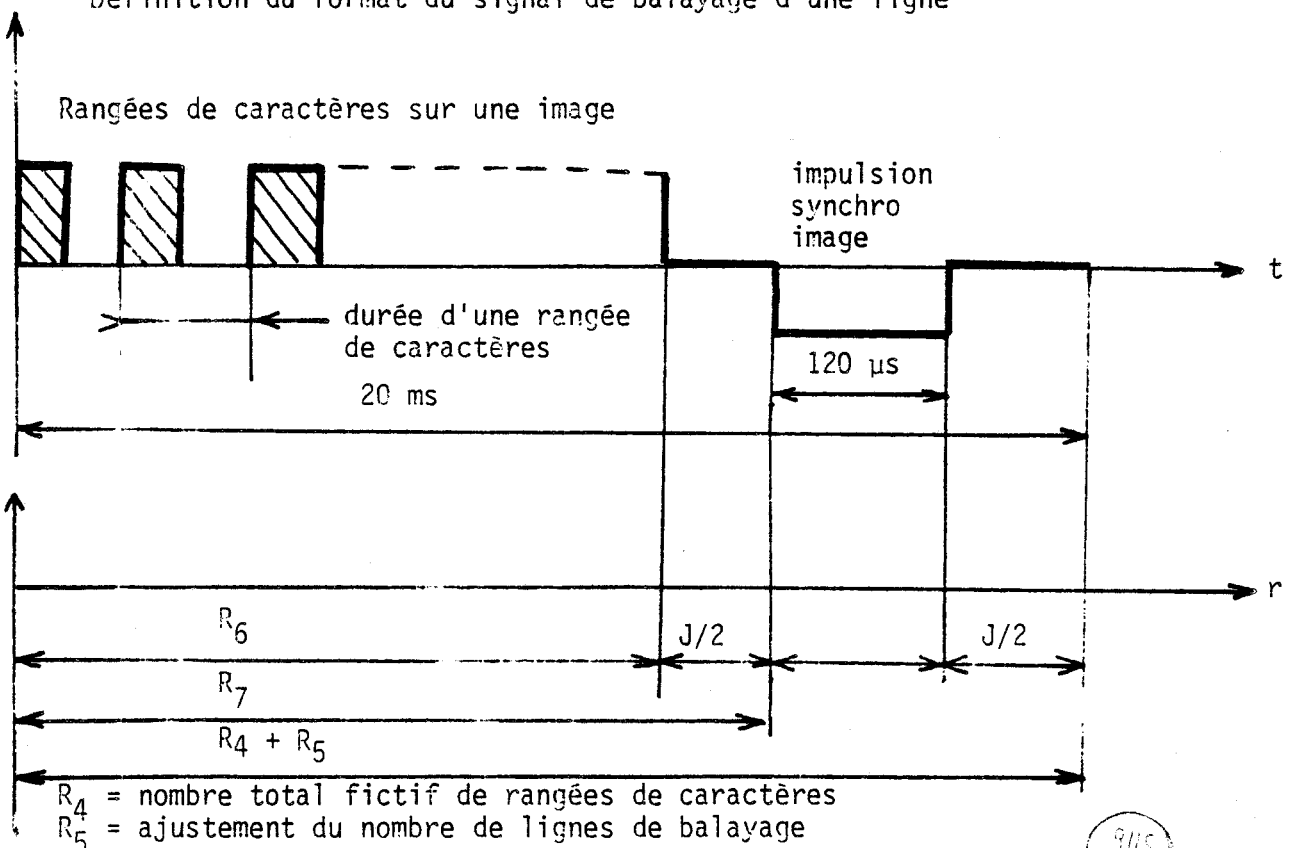
On obtient alors : $R_3 = \text{Entier}\left(\frac{T_2}{T_1} R0_n + 0{,}5\right)$

$R_2 = 1 + \text{Entier}\left(\frac{T_4}{T_1} R0_n\right) + R_1$

$T0_{n+1} = T_1/R0_n$



Définition du format du signal de balayage d'une ligne



Définition du format du signal de balayage d'une image

Figure 7

Compte tenu du caractère discontinu de la valeur de la période T_0 du signal d'horloge, le logiciel permet d'obtenir un ajustement afin de choisir le rapport de division de fréquence k_1 $k = F/f_0$ qui fournit une période T_0 la plus proche de T_{0n+1} trouvée et qui satisfait les mêmes contraintes.

Pratiquement, avec des valeurs de k et k_1 telles que : $1 \leqslant k \leqslant 16$ et $1 \leqslant k_1 \leqslant 31$ et avec une horloge de 16 MHz, on peut représenter toutes les valeurs possibles comprises entre 6 et 64 caractères visualisés par ligne dans un format s'étendant sur la plus grande largeur possible du plan de l'écran.

Les contenus des registres qui définissent les paramètres des signaux de balayage image sont obtenus ensuite facilement à l'aide des relations :

R_4 = Entier(V/N_4)
= Nombre total fictif de rangées de caractères

V = Entier$(T_6/R_0T_0 + 0,5)$
= Nombre réel de lignes de balayage

N_4 = Nombre de lignes de balayage par rangée de caractères.

$R_5 = V - R_4N_4$, R_6 = entier $\leqslant R_4$ = nombre de rangées de caractères visibles de texte sur l'image

$J = R_4 - R_6+1$ = nombre de rangées de caractères non visualisés

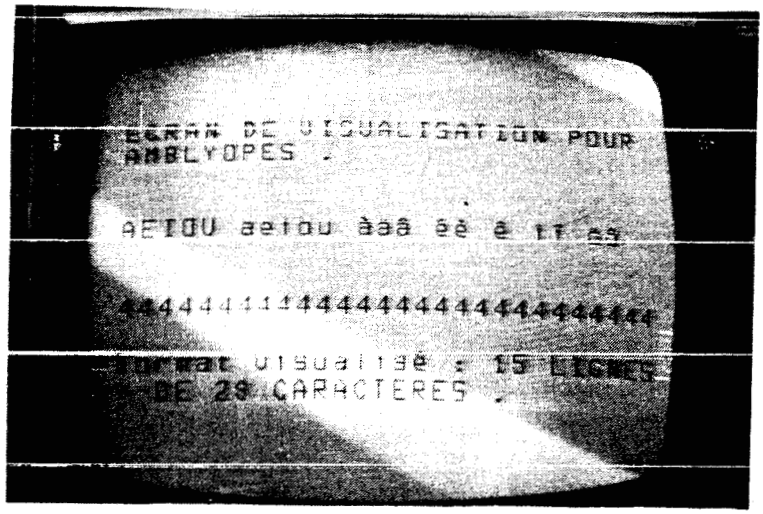
R_7 = Entier$(J/2 + R_6 - 1)$

La planche I représente le circuit de quelques caractères inscrits sur l'écran dans divers formats.

VI-3- DESCRIPTION DU LOGICIEL QUI PERMET DE FAIRE DEFILER UNE PARTIE DU TEXTE SUR L'ECRAN

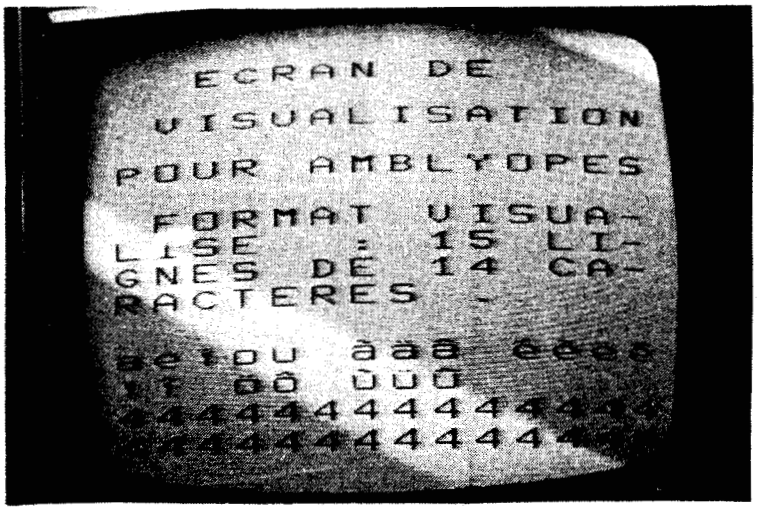
On dispose de deux mémoires : une mémoire écran de 512 octets qui contient le texte visualisé et une mémoire de sauvegarde du même texte noir de 1 k octets par page de texte.

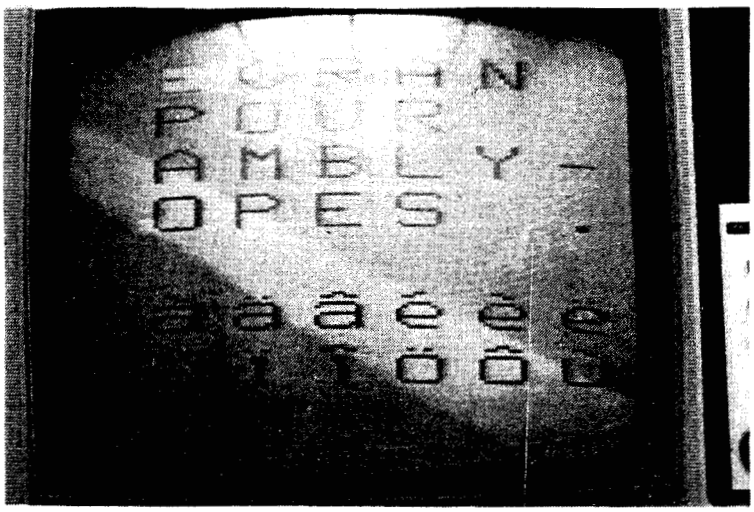
La mémoire écran sera divisée en deux parties (figure 8)
- une partie de 3 x 32 octets qui contient les deux lignes de commentaires ainsi que la ligne de remplissage Braille. Le contenu de cette partie sera fixe sur l'écran.



Format : 15 lignes de
28 caractères

Format : 15 lignes de
14 caractères





Format : 8 lignes de
6 caractères

PLANCHE 1

- La deuxième partie contient les codes des caractères du texte visualisé sur l'écran.

Lorsque la touche RC a été actionnée, tout le contenu de cette partie sera translaté de N-1 octets (N-1 étant le nombre de caractères visualisés par ligne et on réserve un octet pour le code ASCII de RC).

Cette opération se traduit par le défilement de tout le texte sur l'écran d'une ligne vers le haut ou vers le bas suivant le sens de translation du contenu de la mémoire écran (pour notre cas, c'est vers le haut) alors que les deux lignes du commentaire et la ligne de remplissage de ligne Braille restent fixes.

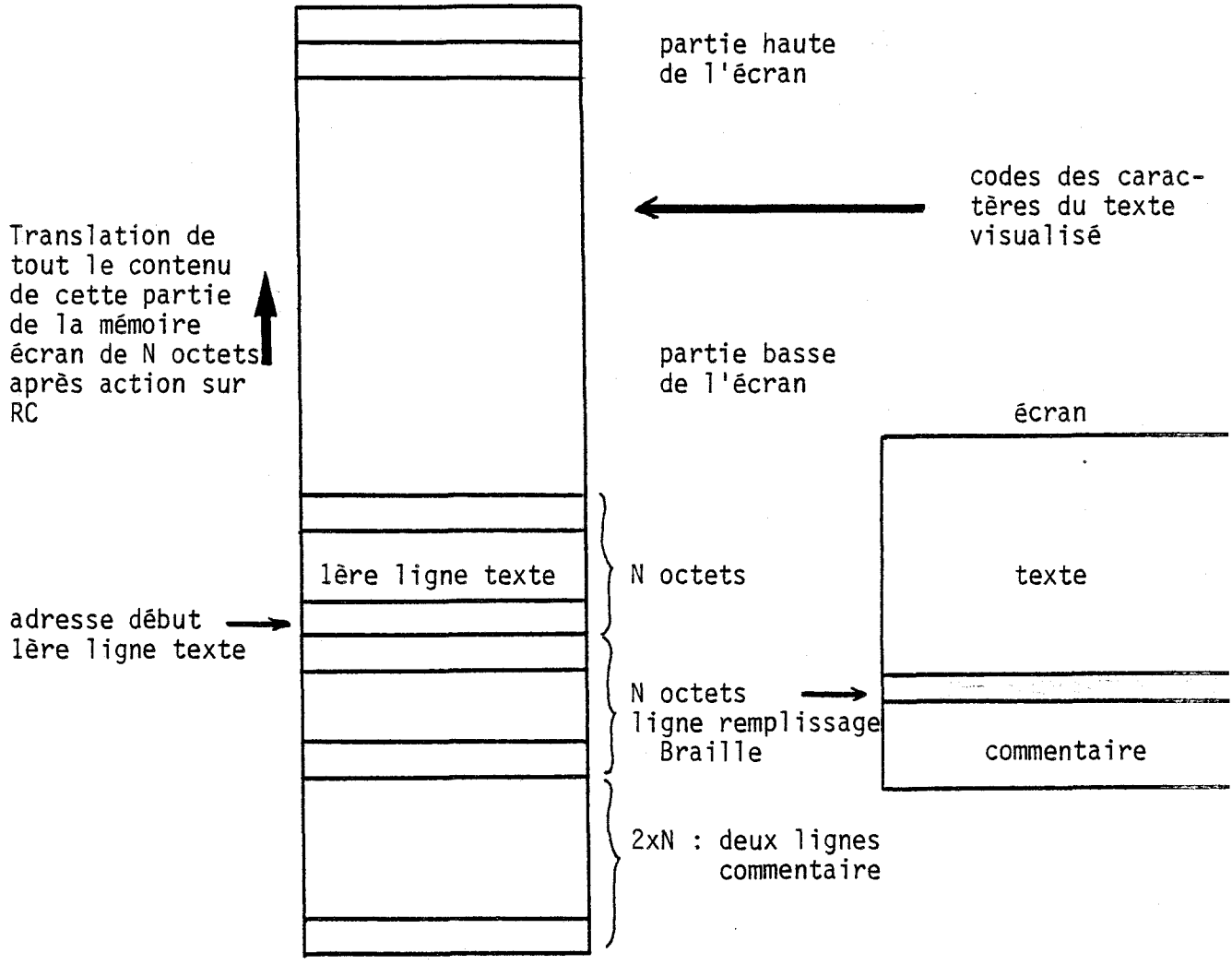


Figure 8

Après chaque translation les contenus des N octets de la première ligne seront remplacés par des "nuls" ce qui se traduit par la libération de la ligne pour l'entrée de la suite du texte.

D'une façon très succinte, le logiciel de défilement du texte sur l'écran est représenté par l'organigramme de principe de la figure 9.

Ce logiciel inscrit chaque code de caractère noir à visualiser à la fois dans la zone mémoire de N octets réservée à la première ligne de texte et à la suite des codes précédemment entrés dans la mémoire de sauvegarde de texte noir indépendante de la mémoire écran.

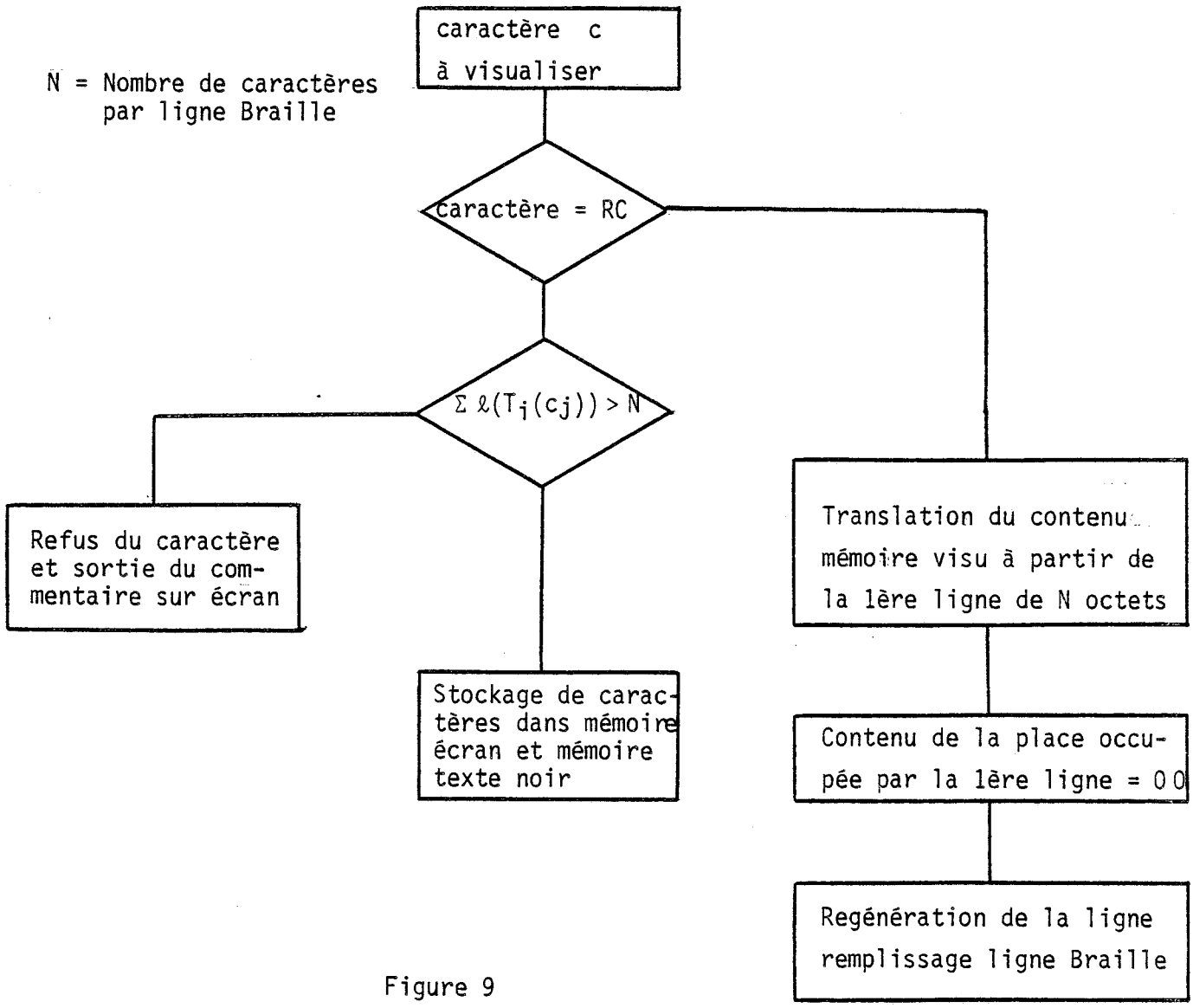


Figure 9

Une autre solution consiste a utiliser une mémoire commune de 1 k octets qui soit en même temps une mémoire écran et une mémoire de sauvegarde de texte, de cette façon on pourrait faire l'économie de 1/2 k octets.

Nous venons de montrer comment on pourrait à la fois faire varier la taille des caractères visualisés et comment faire défiler le texte sur l'écran à l'aide du circuit décrit précédemment.

Nous allons montrer dans le paragraphe suivant comment appliquer un tel outil pour aplanir les difficultés de transcription automatique en Braille d'un texte entré au clavier sans nécessiter de connaissance particulière en Braille ou en informatique.

VII - SYNOPTIQUE DES LOGICIELS DE TRADUCTION EN BRAILLE INTEGRAL

L'ensemble des logiciels est organisé autour d'un registre général et piloté par un superviseur. Ce superviseur permet,à la mise sous tension, d'initialiser l'automate, le choix de la fonction à exécuter et la distribution des tâches (ou appel de chacun des modules du logiciel).

Les deux organes d'entrée sont le clavier et deux lecteurs enregistreurs de cassette . Les organes de sortie sont les mêmes lecteurs enregistreurs de cassette , un écran de visualisation et une sortie V24 (figure 10).

VII-1- LOGICIEL DE GESTION CLAVIER ET TRADUCTION BRAILLE INTEGRAL

La figure 11 donne le synoptique de la structure de ce logiciel. La figure 12 décrit une partie de l'organigramme du codage Braille intégral des codes générés au clavier.

Ce logiciel permet de faire l'acquisition des informations venant du clavier et de les filtrer. Ce filtre consiste à faire la distinction entre les caractères de contrôle et les caractères qui font partie du texte à traduire.

Dans ce cas, le filtre les transforme pour qu'ils contiennent toute l'information nécessaire au codage Braille intégral. C'est ce logiciel qui va inscrire dans le registre général (appelé CCABSV) des informations qui permettront par la suite (c'est-à-dire après l'entrée d'un caractère quelconque au clavier) la gestion de tous les modules bâtis autour de ce registre.

Le CCABSV dont le contenu constitue le résultat du filtrage et du codage Braille intégral est composé de 8 registres élémentaires de 8 bits.

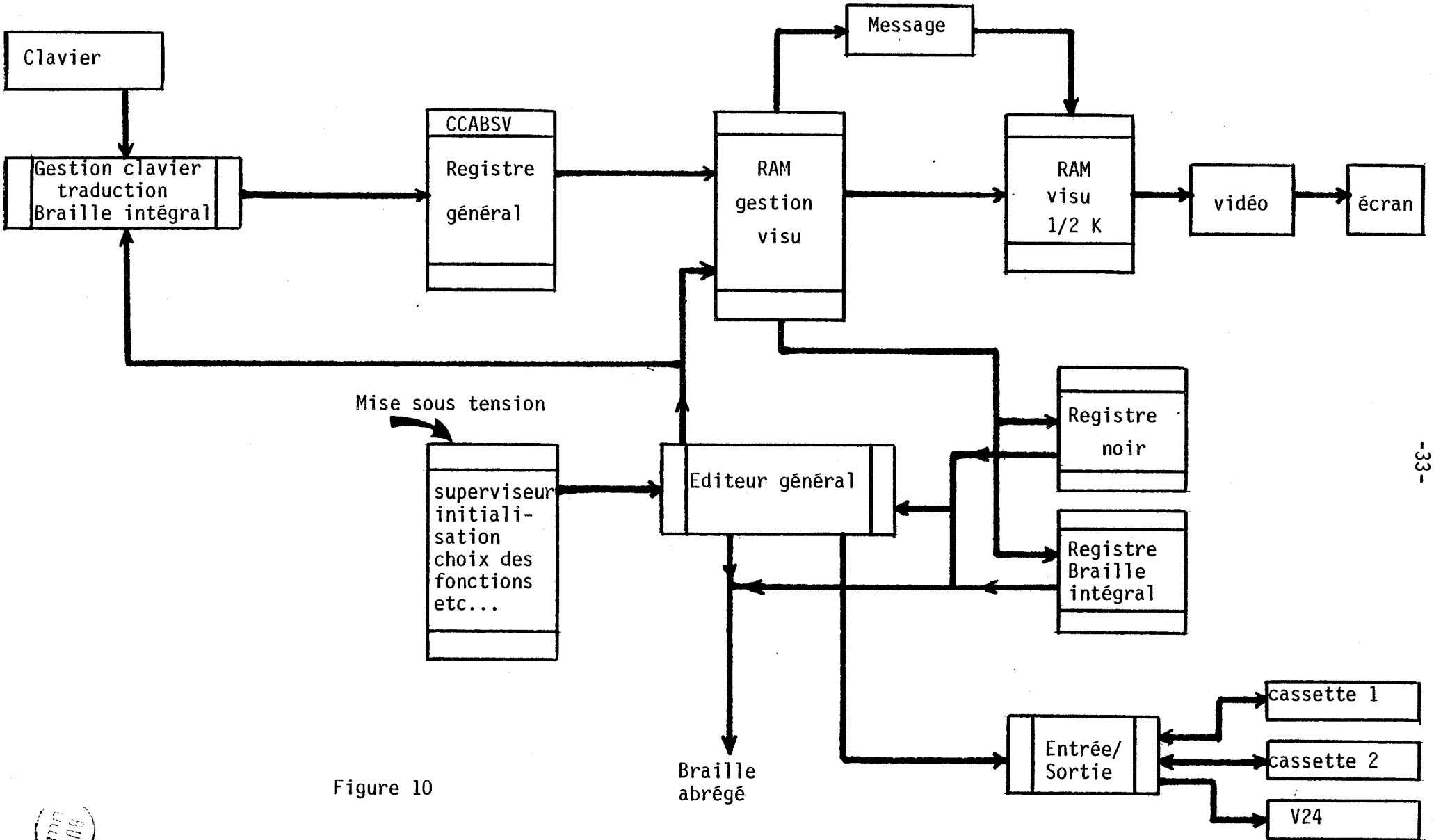

Clavier
Gestion clavier
traduction
Braille intégral
CCABSV
Registre
général
Message
RAM
gestion
visu
RAM
visu
1/2 K
vidéo
écran
Mise sous tension
superviseur
initiali-
sation
choix des
fonctions
etc...
Editeur général
Registre
noir
Registre
Braille
intégral
Braille
abrégé
Entrée/
Sortie
cassette 1
cassette 2
V24

Figure 10

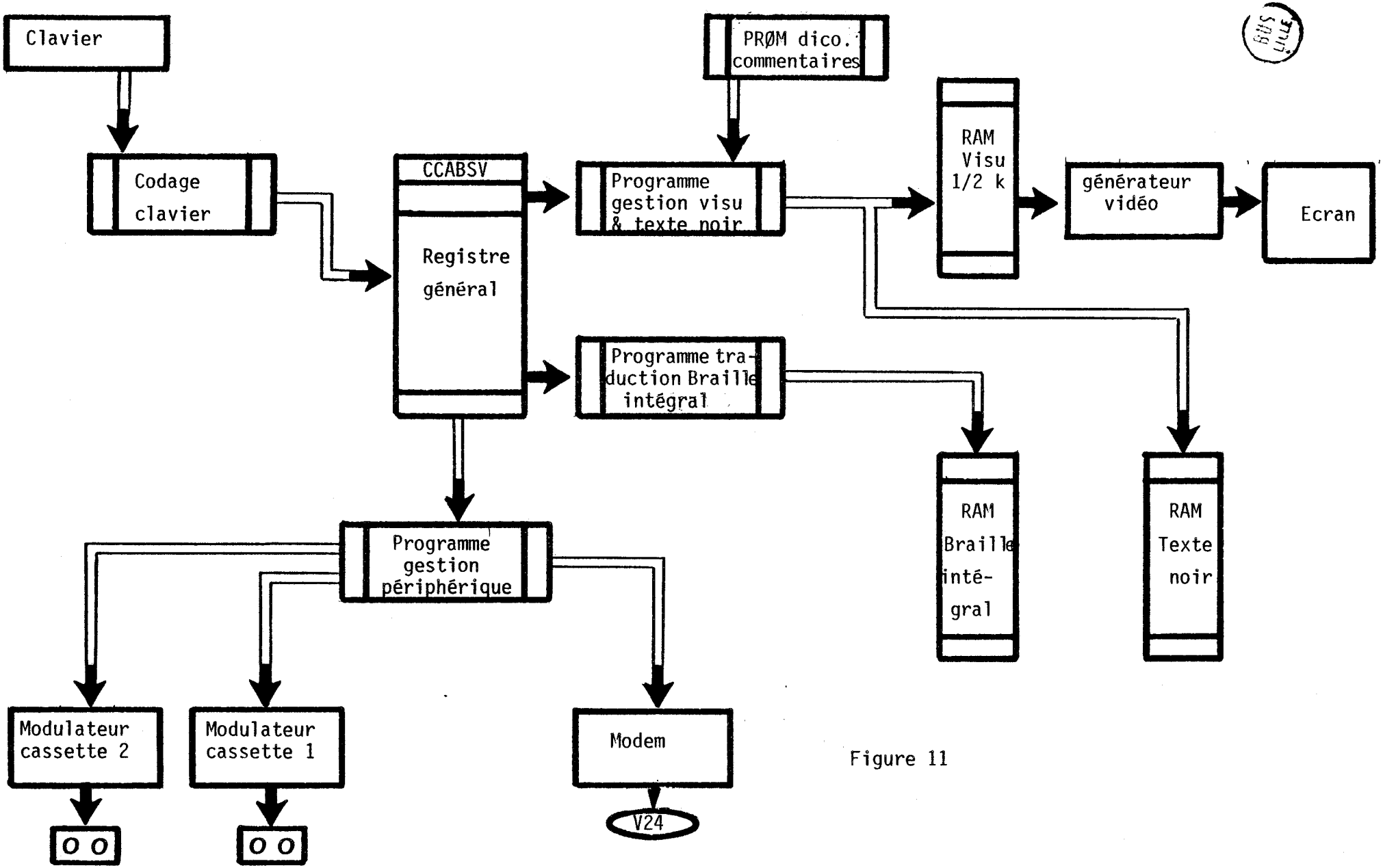

Clavier
Codage clavier
CCABSV
Registre général
PRØM dico. commentaires
Programme gestion visu & texte noir
Programme traduction Braille intégral
RAM Visu 1/2 k
générateur vidéo
Ecran
RAM Braille intégral
RAM Texte noir
Programme gestion périphérique
Modulateur cassette 2
Modulateur cassette 1
Modem
V24

Figure 11

Les contenus de ces registres sont :

CCABSV0 : Code type de caractère (caractère de texte, caractère de contrôle etc.)
CCABSV1 : ASCII du caractère entré au clavier
CCABSV2 : Nombre de caractères Braille correspondant au caractère noir entré au clavier
CCABSV3 : 1er caractère Braille
CCABSV4 : 2è " "
CCABSV5 : 3è " "
CCABSV6 : 4è " "
CCABSV7 : 5è " "

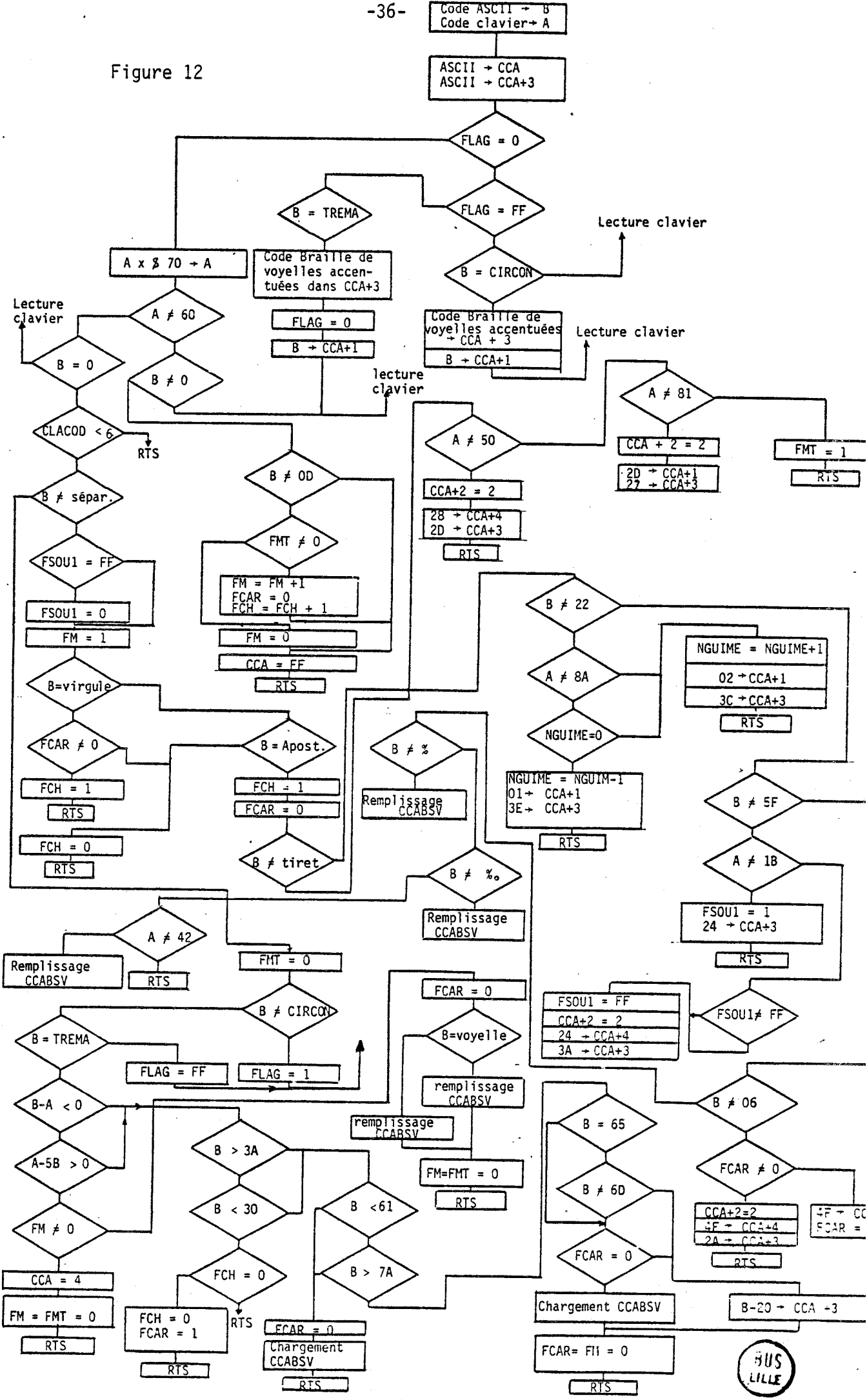
Nous avons réservé 5 registres de 8 bits pouvant contenir les codes de caractères Braille qui traduisent un caractère noir dont le code ASCII est contenu dans CCABSV1, car la plus longue séquence Braille correspondant à un caractère noir est celle du ‰ (5 caractères Braille).

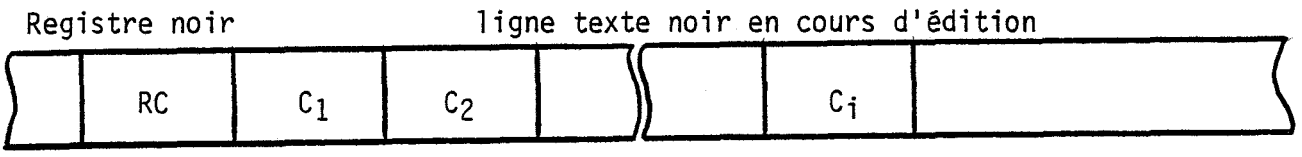
VII-2- LOGICIEL DE GESTION DES TEXTES

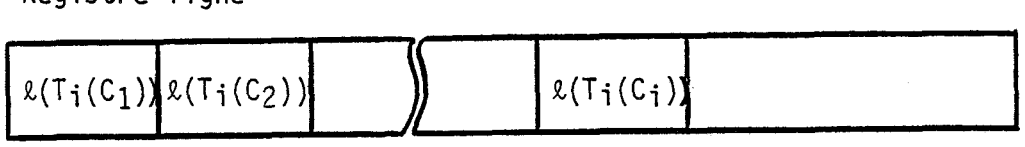
Ce logiciel permet à partir des informations contenues dans le CCABSV :
- la gestion de l'écran,
- le rangement des codes des caractères noirs et les codes des caractères Braille respectivement dans le registre texte noir (RAM 1 k octets) et dans le registre texte Braille intégral (RAM 1 k octets),
- l'inscription sur l'écran, dans certains cas en cours de saisie d'un commentaire pris parmi n commentaires contenus dans une bibliothèque (PROM 300 octets),
- le remplissage d'un registre ligne de 32 octets. Ce registre permet à partir de la connaissance du contenu de CCABSV2 (c'est-à-dire le nombre de caractères Braille traduisant un caractère noir) d'inscrire cette information dans l'octet correspondant au caractère noir considéré.

Si nous appelons c_j les caractères noirs qui composent une ligne de texte et $\ell(T_j(c_j))$ la longueur de la séquence Braille correspondant à c_j, alors on peut représenter le registre ligne sous la forme ci-dessous :

Figure 12


Code ASCII → B
Code clavier → A
ASCII → CCA
ASCII → CCA+3
FLAG = 0
FLAG = FF
B = TREMA
B = CIRCON
Lecture clavier
A x 8 70 → A
Code Braille de voyelles accentuées dans CCA+3
Code Braille de voyelles accentuées → CCA + 3
B → CCA+1
FLAG = 0
A ≠ 60
B = 0
B ≠ 0
CLACOD < 6
RTS
B ≠ sépar.
FSOU1 = FF
FSOU1 = 0
FM = 1
B=virgule
FCAR ≠ 0
FCH = 1
RTS
FCH = 0
RTS
B = Apost.
FCH = 1
FCAR = 0
B ≠ tiret
B ≠ OD
FMT ≠ 0
FM = FM +1
FCAR = 0
FCH = FCH + 1
FM = 0
CCA = FF
RTS
A ≠ 50
CCA+2 = 2
28 → CCA+4
2D → CCA+3
RTS
A ≠ 81
CCA + 2 = 2
2D → CCA+1
27 → CCA+3
FMT = 1
RTS
B ≠ 22
A ≠ 8A
NGUIME=0
NGUIME = NGUIME+1
02 → CCA+1
3C → CCA+3
RTS
NGUIME = NGUIM-1
01 → CCA+1
3E → CCA+3
RTS
B ≠ %
Remplissage CCABSV
B ≠ %o
Remplissage CCABSV
B ≠ 5F
A ≠ 1B
FSOU1 = 1
24 → CCA+3
RTS
FSOU1≠ FF
FSOU1 = FF
CCA+2 = 2
24 → CCA+4
3A → CCA+3
A ≠ 42
Remplissage CCABSV
RTS
FMT = 0
B ≠ CIRCON
B = TREMA
FLAG = FF
FLAG = 1
FCAR = 0
B=voyelle
remplissage CCABSV
remplissage CCABSV
FM=FMT = 0
RTS
B-A < 0
A-5B > 0
FM ≠ 0
CCA = 4
FM = FMT = 0
RTS
B > 3A
B < 30
FCH = 0
FCH = 0
FCAR = 1
RTS
RTS
B <61
B > 7A
FCAR = 0
Chargement CCABSV
RTS
B = 65
B ≠ 6D
FCAR = 0
Chargement CCABSV
FCAR= FM = 0
RTS
B ≠ 06
FCAR ≠ 0
CCA+2=2
4F → CCA+4
2A → CCA+3
RTS
4F → CC
FCAR =
B-20 → CCA +3
Lecture clavier
lecture clavier
BUS LILLE





Ce registre permet dans le cas de l'effacement de l'un quelconque des caractères noir de la ligne en cours d'édition de connaître combien de caractères Braille il faut effacer dans le registre Braille intégral. L'avantage de ce système est qu'il n'est pas nécessaire de faire une détraduction du texte Braille intégral dans le cas d'une correction. Il permet d'autre part de décider de la validation ou non du dernier caractère entré en gardant dans un registre quelconque le nombre courant M des caractères Braille intégral précédemment validés.

Ce nombre est donné par :

$$M = \sum_{i=1}^{n} \ell(T_i(c_i))$$

avec n : nombre de caractères noir précédemment entrés.

Ce procédé permet de corriger la ligne qui vient d'être saisie au clavier avant son acquittement par la génération d'un (RC).

Nous retiendrons ce principe pour l'étude éventuelle du cas où l'on souhaite pratiquer la correction dans toute la partie d'une page précédemment saisie. Dans ce cas, nous devrions réserver 1 K octets.

DEUXIEME PARTIE

ETUDE DE LA TRANSCRIPTION EN BRAILLE ABRÉGÉ

I - ETUDE DU DICTIONNAIRE ET DE SA STRUCTURATION

Nous avons décrit au chapitre III-4 de la première partie le dictionnaire. Dans cette partie, nous allons faire l'étude séparée de la structuration du dictionnaire et de la découpe en contractions. Dans les deux cas, nous allons d'une part tenter d'attacher quelques concepts mathématiques à la façon dont est structuré le dictionnaire pour nous permettre de justifier d'une façon rationnelle la découpe que nous avons proposée et pour nous conduire à des solutions que nous n'aurions nécessairement pas atteintes systématiquement par des raisonnements intuitifs, et d'autre part structurer notre logiciel pour faire en sorte de satisfaire les diverses nuances qui peuvent exister entre les utilisateurs sans remettre en cause la structure globale du logiciel que nous avons conçu.

I-1- APPLICATION DE LA THEORIE DES ENSEMBLES A L'ETUDE DU DICTIONNAIRE

Pour développer cette application, il nous faut tout d'abord introduire quelques définitions qui vont nous permettre de rattacher les êtres mathématiques aux données à manipuler.

I-1-1- Définitions des grandeurs mathématiques à introduire

I-1-1-a- Préfixe de deux chaînes α, β: $P(\alpha,\beta)$

Soient $\alpha, \beta \in V^*$, $P(\alpha,\beta)$ est une application de $V^* \times V^*$ dans V^*
Elle consiste à extraire le préfixe commun de deux chaînes α , β

Soient $a, b \in V$ et $\phi, \lambda, \mu \in V^*$

- si $\alpha = \lambda a$, $\beta = \lambda b$ alors $P(\alpha,\beta) = \lambda$
- si $\alpha = a\lambda$, $\beta = b\lambda$ alors $P(\alpha,\beta) = \wedge$
- si $\alpha = \phi\lambda$, $\beta = \phi\mu$ alors $P(\alpha,\beta) = \phi P(\lambda,\mu)$

exemple :

α = "indigne"

β = "inaction"

on a $P(\alpha,\beta)$ = "in"

I-1-1-b- Relation "inférieur" ("<") entre deux chaînes non vides

Soient α et β deux éléments de V^+ avec $\alpha \neq \beta$ et soient $a_1, a_2 \in V$ et $\lambda, \mu, \phi \in V^*$

- si $\alpha = \phi\, a_1\, \lambda$ et $\beta = \phi\, a_2\, \mu$ avec $a_1 \neq a_2$

  alors $\alpha < \beta$ si, et seulement si, $a_1 < a_2$ si non : $\beta < \alpha$

- si $\beta = \alpha\ a_2\ \lambda$ alors $\beta < \alpha$
- si $\alpha = \beta\ a_1\ \lambda$ alors $\alpha < \beta$

C'est une relation d'ordre stricte. En conséquence si $\neg(\alpha < \beta)$ et $\alpha \neq \beta \rightarrow \beta < \alpha$

Le symbole $\neg$ désigne "non".

La relation "<" entre deux chaines est transitive

si $\alpha < \beta$ et $\beta < \delta$ alors $\alpha < \delta$ $\forall\ \alpha, \beta, \delta \in V^*$

Soient deux chaînes $\alpha, \beta \in V^*$ telles que $\beta = P(\beta,\alpha)$
on a alors $\ell(\alpha) > \ell(\beta)$ car si $\beta = P(\beta, \alpha)$ cela implique que $\alpha = \beta\lambda$ avec $\lambda \in V^*$

On a $\ell(\alpha) = \ell(\beta) + \ell(\lambda)$

et si $\lambda \neq \wedge \rightarrow \ell(\lambda) \neq 0$ et donc $\ell(\beta) < \ell(\alpha)$
Dans ce cas, on prend $\alpha < \beta$

I-1-1-c- Suffixe de deux chaînes α, β : $S(\alpha,\beta)$

C'est une application de $V^* \times V^*$ dans V^* . Elle consiste à extraire le suffixe commun de deux chaînes α, β

Soient deux chaînes $\alpha, \beta \in V^*$ et soient a, b $\in$ V et $\phi, \lambda, \mu \in V^*$

- si $\alpha = a\lambda$ et $\beta = b\lambda$ alors $S(\alpha,\beta) = \lambda$
- si $\alpha = \lambda\phi$ et $\beta = \mu\phi$ alors $S(\alpha,\beta) = S(\lambda,\mu)\phi$
- si $\alpha = \lambda a$ et $\beta = \lambda b$ avec $b \neq a$ alors $S(\alpha,\beta) = \wedge$

exemple : soit α = "calmement"
et β = "faiblement"
on a $S(\alpha,\beta)$ = "ement"

I-1-1-d- Préfixe, suffixe et racine d'une chaîne β

Ce sont des applications de V^* dans V^*

Soient $\lambda, \alpha, \beta, \mu \in V^*$, on peut décomposer la chaîne β en trois chaînes α, λ, μ telle que $\beta = \alpha\ \mu\ \lambda$

avec : λ = Suf (β) = $S(\beta)$
α = Préf(β) = $P(\beta)$
μ = Rac(β) = $R(\beta)$

Les abréviations Suf, Préf et Rac désignent respectivement Suffixe, Préfixe et Racine.

La décomposition de la chaîne β de cette façon n'est pas arbitraire, c'est une donnée linguistique .

Exemple : soit la chaîne α = "indignement"
alors Préf(α) = "in"
Suf(α) = "ment"
Rac(α) = "digne"

Pour une simplification d'écriture, on notera Préf(α), Suf(α) et Rac(α) successivement P(α), S(α) et R(α)

I-1-2- Propriétés du procédé de transcription de chaînes préabrégées

Nous appellerons ce procédé T_1 . Il consiste à établir une relation entre les chaînes de caractères appartenant à la syntaxe noir et celles appartenant à la syntaxe Braille. T_1 est donc une application qui, à tout élément de V_1^*, fait correspondre un élément de B_1^*

avec $V_1^* \subset V^*$ et $B_1^* \subset B^*$

V^* et B^* sont les ensembles de toutes les chaînes possibles de l'alphabet noir V et de l'alphabet Braille B tandis que V_1^* et B_1^* sont des ensembles particuliers.

V_1^* est un ensemble constitué d'éléments d'usage très fréquent et qui sont préabrégés. C'est ce qui est appelé dictionnaire.

Nous pouvons donc dire que : $\forall \alpha \in V_1^*$ il existe un seul élément $\beta \in \beta_1^*$ tel que $\beta = T_1(\alpha)$

Nous pouvons dégager du procédé de transcription de chaînes préabrégées trois propriétés que nous énonçons ci-dessous :
- la première propriété est relative à la forme verbale, au féminin, au pluriel, ou au féminin pluriel
- la deuxième propriété est relative aux lois d'associations des mots du dictionnaire avec des préfixes
- la troisième propriété est relative aux lois d'association des racines de mots du dictionnaire avec des suffixes.

I-1-2-1- Propriété 1 de T_1 (forme verbale, féminin, pluriel, et féminin pluriel)

Soient les chaînes $\alpha, \beta, \lambda \in V^*$ telles que $\alpha = \beta\lambda$

avec $\lambda = S(\alpha)$ et $T_1(\beta) \in B_1^x$

On a $T_1(\alpha) = T_1(\beta\lambda) = T_1(\beta)\ T_1(\lambda)$ si et seulement si $\lambda \in D_{s_1}$

avec $D_{s_1} = \{ e, s, x, es \}$

et $T_1(D_{s_1}) = \left\{ \begin{matrix} \bullet\circ & \circ\bullet & \bullet\bullet & \bullet\circ \\ \circ\bullet & \bullet\circ & \circ\circ & \circ\bullet \\ \circ\circ & \bullet\circ & \bullet\bullet & \circ\bullet \end{matrix} \right\}$

Cette propriété peut s'énoncer de la façon suivante : pour tous les mots du dictionnaire, on peut ajouter les lettres e, s, x ou es pour former le féminin, le pluriel, le féminin pluriel ou pour avoir une forme verbale.

exemple : Soit $\beta \in V_1^*$ avec β = "bien" , $T_1(\beta) = \begin{matrix} \bullet\circ \\ \bullet\circ \\ \circ\circ \end{matrix}$

et soit à transcrire la chaîne α = "biens"

α peut être décomposée en deux chaînes β et λ avec β = "bien" et λ = "s"

Sachant que $T_1(\beta) \in B_1^*$ et $\lambda \in D_{s_1}$

on a : $T_1(\alpha) = T_1(\beta)\ T_1(\lambda)$

$= \begin{matrix} \bullet\circ & \bullet\bullet \\ \bullet\circ & \bullet\circ \\ \circ\circ & \bullet\circ \end{matrix}$

Certains cas particuliers (très rares d'ailleurs) peuvent causer des erreurs de transcription car le fait par exemple d'ajouter un "e" en terminaison d'une chaîne de V_1^* ne signifie pas que c'est son féminin ou sa forme verbale.

Exemple : β = "dans" $\in V_1^*$

pour ce cas T_1("danse") $\neq$ T_1("dans") T_1("e")

Il suffit alors d'ajouter la chaîne "danse" associée à sa transcription dans le dictionnaire.

A partir d'une chaîne $\alpha \in V^*$ donnée, on cherche :

si $\alpha = \beta\lambda$ telle que

$\beta \in V_1^*$ et $\lambda = S(\lambda) \in D_{s_1}$

alors $T_1(\alpha) = T_1(\beta)\ T_1(\lambda)$

I-1-2-2- Propriété 2 de T_1 (lois d'association des mots du dictionnaire avec des préfixes

En Braille abrégé, certains mots du dictionnaire peuvent s'associer avec un ou plusieurs préfixes pour constituer un nouveau mot (exemple : mettre et remettre)

Soient les chaînes p, β, $\alpha \in V^*$ telles que $p = P(\alpha)$ et $\beta = R(\alpha)$

avec : $P(\alpha)$ = préfixe de α
$R(\alpha)$ = racine de α

On peut écrire $\alpha = p\ \beta$

si $T_1(\beta) \in B_1^*$ on a $T_1(\alpha) = T_1(p)\ T_1(\beta)$ si, et seulement si :

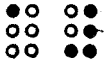
a) $p \in Dp = \{$ in, com, con, auto, ... $\}$

Dp est un sous-ensemble de V^* que nous appelons dictionnaire des préfixes

b) α garde le sens sémantique de $\beta = R(\alpha)$

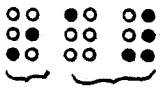
exemple 1 : α = "inaction"
on a : $R(\alpha)$ = "action"
$P(\alpha)$ = "in" $\in$ Dp

avec $R(\alpha) \in V_1^*$ donc $T_1(R(\alpha)) \in B_1^*$

$T_1(R(\alpha))$ = 

$T_1(P(\alpha))$ = 

Dans ce cas, on peut écrire : $T_1(\alpha) = T_1(P(\alpha))\ T_1(R(\alpha))$

On a $T_1(\alpha)$ = 

$T_1(P(\alpha))\ T_1(R(\alpha))$

L'exemple que nous avons pris satisfait aux conditions a et b , c'est-à-dire $p \in Dp$ et $\beta \in V_1^*$ a le même sens sémantique que β.

Exemple 2 :

α = "recette"

Nous pouvons décomposer α en deux chaînes β = "cette" avec $T_1(\beta) \in B_1^*$ et p = "re" $\in$ Dp. Ceci satisfait la condition a mais la condition b n'est pas satisfaite car "cette" et "recette" n'ont pas le même sens sémantique.

Il faut donc construire un dictionnaire qui sera constitué d'éléments de Dp qui concaténés avec ceux de V_1^* donnent $p \in Dp$ et $\beta \in V_1^*$, $\alpha = p\beta$ a le même sens sémantique que β.

Soit Dp l'ensemble des préfixes

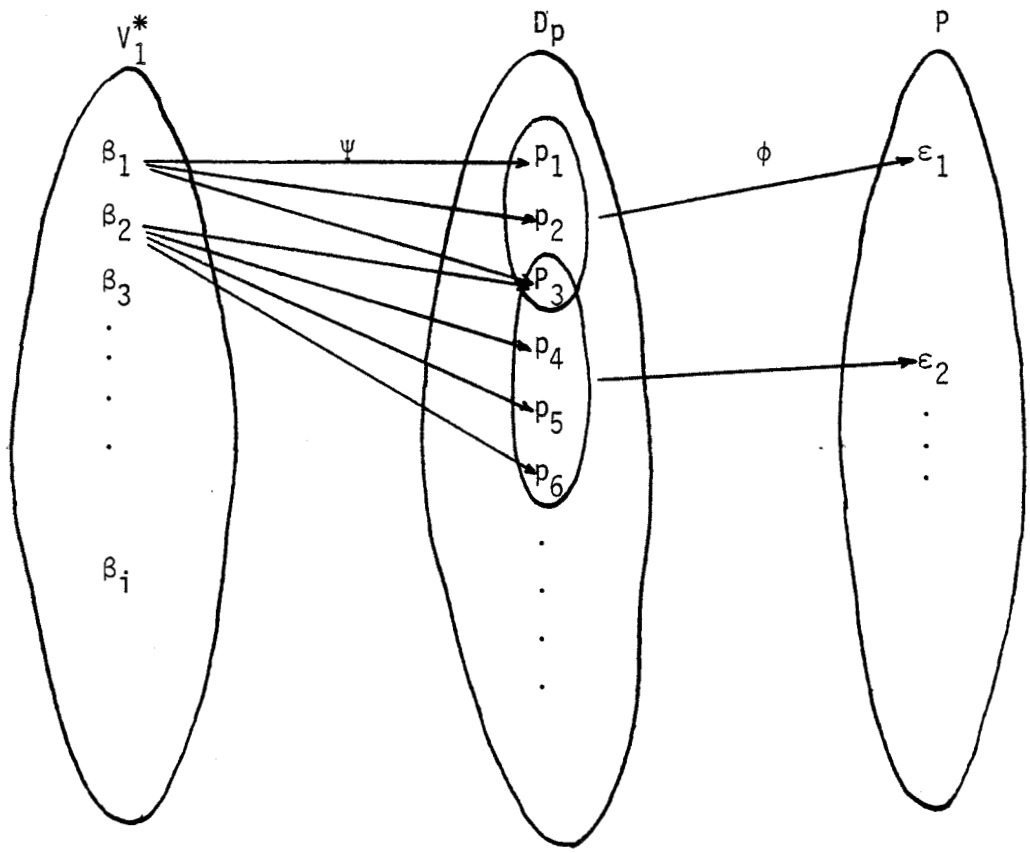
ε_i : sous ensembles de Dp avec $\bigcup_i \varepsilon_i = Dp$

et $\bigcap_i \varepsilon_i \neq \{\Phi\}$

Soit la chaîne $\beta_i \in V_1^*$ et soit $p_j \in Dp$

ε_i est l'ensemble des préfixes p_j pouvant être associés à β_i pour former $\alpha = p_j \beta_i$ avec sens sémantique de α identique à celui de β_i

β_i ne peut s'associer qu'avec $p_j \; \forall \; p_j \in \varepsilon_i$

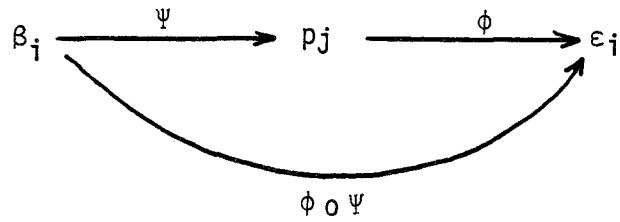


Soit Ψ une application de V_1^* dans Dp qui à tout élément $\beta_i \in V_1^*$ fait correspondre un ou plusieurs éléments $p_j \in \varepsilon_i$ qui peuvent lui être associés

Soit ϕ une application de D_p dans P qui à tout élément p_j lui fait correspondre les ensembles ε_i auxquels il appartient.
On peut alors écrire :

$\Psi(\beta_i) = p_j \quad \forall\, p_j \in D_p$ tel que si $\alpha = p_j\, \beta_i$ le sens sémantique de α est le même que celui de β_i

et $\quad \phi(p_j) = \varepsilon_i \quad \forall\, i$ tel que $p_j \in \varepsilon_i$



ϕ o $\Psi(\beta_i) = \varepsilon_i \quad \forall\, i$ tel que les préfixes pouvant s'associer à β_i sans que le sens de celui-ci change appartiennent à ε_i

On peut alors dire que :

$\forall\, p_j \in Dp$ et $\beta_i \in V_1^*$, si $\alpha = p_j\, \beta_i$

$T_1(p_j\, \beta_i) = T_1(p_j)\, T_1(\beta_i)$ si et seulement si : $\phi_o\ \Psi(\beta_i) = \phi(p_j)$

Dans l'ensemble V_1^* , chaque élément β_i sera associé à ε_i (ensemble auquel il appartient) et chaque élément p_i de D_p sera associé à tous les ε_i (ensembles auxquels il appartient).

V_1^* sera donc un ensemble de couples $(\varepsilon_i\ ,\ \beta_i)$ tel que :

$$V_1^* = \left\{ (\varepsilon_1\ ,\ \beta_1)\ ,\ (\varepsilon_3\ ,\ \beta_2)\ ,\ (\Phi,\ \beta_3)\ \ldots\ \right\}$$

Lorsqu'un élément β quelconque ne peut s'associer à aucun préfixe, le couple sera $(\Phi\ ,\ \beta)$.

De plus, $\forall$ i, k tels que $\varepsilon_k \subset \varepsilon_i$ et si $p_j\, \beta_k$ avec $p_j \in \complement_{\varepsilon_i}\varepsilon_k$ n'a pas de sens dans V^* (donc $p_j\, \beta_n$ n'existe pas), on prend $\varepsilon_k = \varepsilon_i$.

Ceci permet de réduire le nombre d'indices associés à chaque préfixe p_j . Ces indices étant tous les sous-ensembles de Dp auxquels appartient p_j .

Chaque élément de Dp étant un n-uple de la forme :

$$(p_j \ , \underbrace{\varepsilon_i \ , \varepsilon_j \ , \varepsilon_k \ \ldots\ldots \ \varepsilon_m}_{\text{n-1 élément}})$$

n dépend de chaque préfixe p_j

I-1-2-3- Algorithme de recherche d'un préfixe

Soit la chaîne $\delta = v_1 \ v_2 \ \ldots\ldots \ v_m$, on cherche le plus grand préfixe de δ qui appartient à Dp c'est-à-dire un élément p de Dp tel que $P(p, \delta) = p$ et $\forall \ q \neq p$ avec $q \in$ Dp tel que $P(q, \delta) = q$, on ait $q = P(q, p)$.

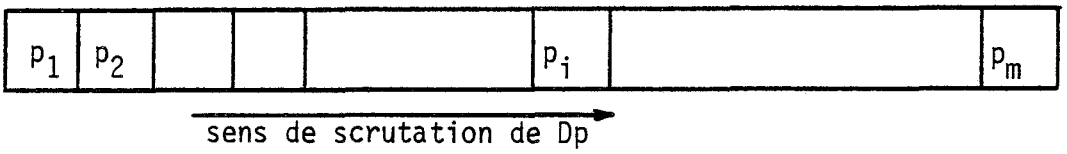
Ceci est facile à obtenir sans traitement spécial si on organise le dictionnaire des préfixes de la façon suivante :

Dp étant l'ensemble des préfixes, $Dp = \{ p_1, p_r \ldots\ldots p_m \}$ avec $\forall_i \ p_i \in V^*$

Nous avons vu précédemment au paragraphe I-1-1-b que :

$\forall \ p, q \in V^*$ si $p = P(p, q)$ alors $p > q$ et $\ell(p) < \ell(q)$

Si nous ordonnons l'ensemble Dp dans un ordre croissant des indices , c'est-à-dire $\forall_i, j$ si $j > i$, $p_j > p_i$



En scrutant séquentiellement Dp en commençant par p_1 , si on trouve un élément p_i tel que $p_i = P(p_i \ \delta)$, on est sur que c'est le préfixe de δ le plus long dans Dp, car s'il existe un autre élément $p_j \in$ Dp avec $j > i$ tel que $p_j = P(p_j, \delta)$ et puisque $p_i < p_j$, alors $p_j = P(p_j, p_i)$ et donc $\ell(p_j) < \ell(p_i)$

Exemple : Soit à transcrire la chaîne α = "inaction". Cette chaîne n'appartenant pas à V_1^* , il faut chercher un élément p de Dp tel que p = P("inaction").

Le préfixe de α le plus long dans Dp est $P(\alpha)$ = "in". Ensuite, on cherche un élément β de V_1^* tel que $P(\alpha) \ \beta = \alpha$

Dans V_1^* , on a le couple (ε_{12} , action)

Dans Dp on a le n-uple : (in,ε_{12} , ε_{13} , ε_{20})

Nous avons vu que pour que T_1 (P(α) R(α)) = T_1 (P(α)) T_1(R(α)) il faut et il suffit que : ϕ o $\Psi(\beta) = \phi(p)$

Pour notre cas, on a ϕ o $\Psi(\beta) = \phi$o Ψ ("action") = ε_{12} et $\phi(p) = \varepsilon_{12} \cup \varepsilon_{13} \cup \varepsilon_{20}$

On peut donc dire que T_1 ("inaction") = T_1("in") T_1("action")

avec T_1("in") = ⠉ et T_1("action") = ⠒

d'où T_1("inaction") = ⠛

I-1-2-4- Propriété 3 de T_1 (lois d'association des racines de mots du dictionnaire et des suffixes)

Il existe un certain nombre de suffixes d'éléments de V_1^* dont la transcription en Braille est invariable quelle que soit la chaîne à laquelle ils sont associés.

De même, il existe un grand nombre d'éléments de V_1^* ayant la même racine associée à des suffixes différents et dont la transcription en Braille est indépendante des suffixes auxquels elle est associée. C'est cette propriété qui nous a permis de réduire considérablement la taille mémoire occupée par le dictionnaire.

Si nous formons un ensemble $D_S = \{ \lambda_1 , \lambda_2 \dots \lambda_i \dots \}$ (les λ_i étant des suffixes des éléments de V_1^*) nous pouvons ne garder dans V_1^* que des racines en éléminant ainsi les redondances du dictionnaire. D_S est ce que nous appelons le dictionnaire des suffixes.

On peut formuler la première remarque comme suit :

$\forall \delta, \alpha \in V_1^*$ et $\forall \lambda_i \in D_S$ tel que $\lambda_i = S(\alpha) = S(\delta)$

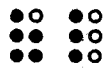
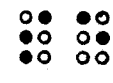
avec $T_1(\alpha) = T_1(R(\alpha)\ )\ T_1(S(\alpha)\ )$

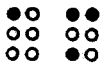
et $T_1(\delta) = T_1(R(\delta)\ )\ T_1(S(\delta)\ )$

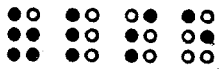
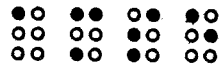
on a $T_1(S(\delta)\ ) = T_1(S(\alpha)\ )$

Exemple : soit α = "chaleureuse" avec $S(\alpha)$ = "euse" $\in D_S$

et soit δ = "amoureuse" avec $S(\delta)$ = "euse"

$T_1(R(\alpha))$ =  et $T_1(S(\alpha)) = T_1(S(\delta))$ = 

$T_1(R(\delta))$ = 

$T_1(\alpha)$ =  et $T_1(\delta)$ = 

La deuxième remarque peut être formulée de la façon suivante :

$\forall$ les chaînes $\alpha, \delta \in V_1^*$ telles que $R(\alpha) = R(\delta)$ et telles que $S(\delta) \in D_S$ et $S(\alpha) \in D_S$ avec $S(\alpha) \neq S(\delta)$

On a : $T_1(\alpha) = T_1(R(\alpha))\ T_1(S(\alpha))$

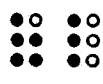
$T_1(\delta) = T_1(R(\delta))\ T_1(S(\delta))$

Dans ces conditions : $T_1(R(\alpha)) = T_1(R(\delta))$

Exemple :

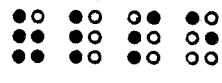
α = "chaleureuse" $S(\alpha)$ = "euse" $\in D_S$

δ = "chaleureux" $S(\delta)$ = "eux" $\in D_S$

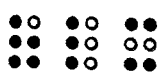
$R(\alpha) = R(\delta)$ = "chaleur" et donc $T_1(R(\alpha)) = T_1(R(\delta))$ = 

α et δ sont transcrites de la façon suivante :

$T_1(\alpha) = T_1(R(\alpha)\ T_1(S(\alpha))$

= 

$T_1(\delta) = T_1(R(\delta))\ T_1(S(\delta))$

= 

I-1-2-5- Algorithme de recherche d'une chaîne de V_1^* associée à une chaîne de D_s

Soit la chaîne $\delta = v_1\ v_2\ \ldots\ v_n$. Il s'agit d'abord de chercher la plus longue chaîne $\beta \in V_1^*$ telle que $\beta = v_1\ v_2\ \ldots\ v_m$ avec $m < n$ ($\beta = R(\delta)$) . Ensuite, vérifier si la chaîne $\lambda = v_{m+1} \ldots v_\ell$ appartient à D_s avec $\ell \leqslant n$

Si $\ell = n$ alors $T_1(\delta) = T_1(\beta)\ T_1(\lambda)$

Si $\ell < n$ il faut vérifier si la chaîne $\theta = v_{\ell+1}\ ..\ v_n$ appartient à D_{s_1} avec $\ell(\theta) \in [1,2]$

Si $\theta \in D_{s_1}$, alors $T_1(\delta) = T_1(v_1\ v_2 \ldots v_m)\ T_1(v_{m+1} \ldots v_\ell)\ T_1(v_{\ell+1} \ldots v_n)$ si non : $T_1(\delta) \notin \beta_1^*$

La propriété 3 de T_1 que nous venons de voir ne peut s'appliquer à tous les éléments de V_1^* , il faut donc tenir compte de cette particularité.

Exemple : La chaîne δ = "libéralement" sera décomposée d'après l'algorithme ci-dessus en deux chaînes β et λ telles que $\delta = \beta\ \lambda$

avec β = "libéral" $\in V_1^*$

λ = "ement" $\in D_s$

mais δ ne figure pas parmi l'ensemble des chaînes préabrégées, c'est-à-dire que l'on ne peut faire : $T_1(\delta) = T_1(\beta)\ T_1(\lambda)$

Pour distinguer les éléments de V_1^* auxquels on ne peut appliquer cet algorithme, il faut que chacun soit associé à un code de non association de suffixe. Ce code indique qu'ils ne peuvent s'associer à un élément quelconque de D_s. Mais, une étude statistique de V_1^* montre que ces éléments sont beaucoup plus nombreux que ceux qui peuvent s'associer aux éléments de D_s ; il apparait donc plus économique du point de vue occupation d'espace mémoire d'associer plutôt ces derniers à un code d'association de suffixe.

L'algorithme que nous venons de décrire sera exact si avant de vérifier que la chaîne $\lambda = v_{m+1} \ldots v_\ell$ appartient à D_s . On vérifie que $\beta = v_1\ v_2 \ldots v_m$ est associée à un "code d'association de suffixe". Si oui, on continue l'exécution de l'algorithme , si non : $T_1(\delta) \notin B_1^*$.

I-2- IMPLANTATION DES DICTIONNAIRES EN MEMOIRE

La taille mémoire nécessaire à la transcription d'un mot ou locution préabrégés est occupée par les programmes de recherche et par le dictionnaire des mots associés à leur traduction.

Dans ces conditions, une implantation simple du dictionnaire occuperait près de 13 K octets.

Lorsque l'on transcrit un texte en Braille abrégé avec un gros ordinateur, on n'a pas à se préoccuper tellement de la taille mémoire occupée par le dictionnaire. En général même, les auteurs ajoutent des mots jusqu'à atteindre plus de 2000mots[6] pour résoudre plus simplement certains cas particuliers de transcription. Dans notre cas, où nous avons à concevoir une architecture spécifique, nous cherchons à minimaliser la taille mémoire occupée par le dictionnaire, et pour cela nous avons choisi la méthode que nous exposons ci-dessous.

I-2-1- Structure du dictionnaire des racines et locutions

Parmi les différentes méthodes de représentation de données en mémoire, nous avons choisi la représentation contigue avec accès dichotomique indirect.

Cette représentation présente l'avantage d'être assez rapide et permet une mise à jour facile (ajout ou suppression de mots du dictionnaire).

Ce dictionnaire est constitué de tables. Chaque table regroupe les chaînes commençant par la même lettre. Les chaînes qui composent une même table se succèdent dans l'ordre de la valeur numérique du mot code représentant chaque lettre (ce n'est pas nécessairement l'ordre alphabétique). En effet, pour deux chaînes contigues m_1 et m_2 appartenant à la même table, avec $m_1 \in V_1^*$ et $m_2 \in V_1^*$, leur implantation en mémoire est faite de la façon suivante :

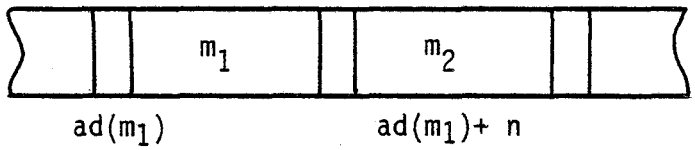
soit $m_1 = v_1\ v_2 \ldots v_\ell$

$m_2 = u_1\ u_2 \ldots u_k$

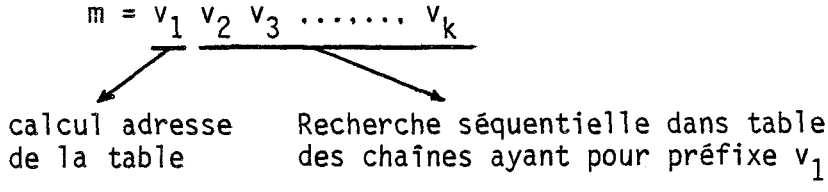
On a $v_1 = u_1$. De plus, s'il existe une chaîne λ avec $\lambda = v_1\ v_2 \ldots v_j$ telle que $\lambda = P(m_1, m_2)$ pour $j < \min(\ell,k)$ si $v_j+1 > u_j+1$ alors $m_1 > m_2$, sinon $m_1 < m_2$

Si $\min(\ell,k) = k$ et $j = k \rightarrow m_2 = P(m_2,m_1)$ on a $m_1 > m_2$

Soit $ad(m_1)$ l'adresse en mémoire de m_1 et $ad(m_2)$ l'adresse en mémoire de m_2 , on a : $ad(m_2) = ad(m_1) + n$ avec $n > 0$



Dans ces conditions, la première lettre d'un mot ou locution entrée au clavier permet de déterminer l'adresse de la table des mots commençant par cette lettre. La deuxième lettre permet de trouver par scrutation séquentielle dans la table le premier des mots ayant pour préfixe la chaîne des deux premiers caractères du mot.



Remarquons que puisque la recherche est séquentielle, toute économie d'octets du dictionnaire permet aussi de diminuer le temps de recherche.

La réduction de la taille du dictionnaire s'effectue à l'aide de deux démarches. La première consiste à supprimer la première lettre de chaque mot en mémoire. Ceci permet d'économiser environ 1 k octets. La deuxième démarche qui procure le plus d'économie est celle relative à la propriété 3 de T_1. Elle consiste à supprimer la redondance de certains mots en créant un dictionnaire des suffixes. Deux cas généraux se présentent :

- Exemple 1er cas :

Dans la table des mots commençant par la lettre "a", il existe la séquence suivante :

Chaîne $\alpha \in V_1^*$	$S(\alpha) \in D_S$	$\ell(T_1(R(\alpha)))$	$\ell(T_1(S(\alpha)))$	$\ell(T_1(\alpha))$
avantage		3		3
avantage	use	3	2	5
avantage	ux	3	1	4
avantage	usement	3	2	5

Cette séquence occuperait 84 octets en mémoire. Pour réduire la taille mémoire de cette séquence, on ne conserve que le préfixe commun ("noir" et Braille) de ces chaînes appelé <racine> et on regroupe les terminaisons dans une table de suffixes.

La recherche de l'une des quatre chaînes de cette séquence consiste à reconnaître la racine comme appartenant au dictionnaire et à juxtaposer à la traduction de la racine celle des suffixes (voir algorithme de recherche de suffixe). Ainsi réduite, cette séquence n'occupe que 11 octets dans le dictionnaire des suffixes (voir dictionnaire des suffixes en annexe) , soit au total 28 octets au lieu de 84.

- Exemple 2è cas

Chaîne $\alpha \in V_1^*$	$S(\alpha) \in D_S$	$\ell(T_1(R(\alpha)))$	$\ell(T_1(S(\alpha)))$	$\ell(T_1(\alpha))$
facile		2		2
facil	ité	2	1	3
facile	ment	2	1	3

Cette séquence occuperait 32 octets. Si on ne conserve que la racine "facile" on peut grouper dans le dictionnaire des suffixes les terminaisons "ité" et "ment".

La chaîne facilité n'étant pas une simple concaténation de "facile" et "ité", le programme doit donc tenir compte de cette particularité. Cette séquence n'occupe plus alors dans le dictionnaire que 17 octets au lieu de 32.

Globalement, l'application systématique de cette méthode permet de ramener la taille du dictionnaire compte tenu de la table des suffixes de 13 K octets à 6 K octets.

I-2-1-1- Structure de chaque élément du dictionnaire

Nous avions vu que chaque élément de V_1^* doit être associé à :
- un "indicateur d'association racine préfixe"
- un "indicateur d'association racine suffixe" dans le cas où l'élément peut être associé à un suffixe.

De plus, il faudrait disposer d'une table de tous les éléments de B_1^* et avoir un moyen de correspondance entre chaque élément $\alpha \in V_1^*$ et $T_1(\alpha) \in B_1^*$. Pour cela, le moyen le plus simple et le plus économique en espace-mémoire est d'avoir chaque élément α du dictionnaire contigue avec $T_1(\alpha) \in B_1^*$.

Codage des éléments de B et V : chaque élément de V est codé ASCII (entre $*$20 et $ 7E). Plusieurs codages des éléments de B sont utilisés suivant le périphérique de sortie (embosseuse SAGEM, PED30 etc..).

Le codage des éléments de B et V est décrit en détail en annexe .

Nous avons utilisé une représentation mixte qui tient compte des codes du périphérique générateur de relief Braille le plus utilisé et des codes des caractères "noir" (code ASCII).

Chaque élément de B a été codé entre $ A0 et $ DE, ce qui permet de distinguer les codes des caractères "noir" des codes des caractères Braille (fig.13) et de permettre par un traitement simple (soustraction d'une puissance entière de 2) de convertir les codes des caractères Braille sous forme du code du périphérique générateur de relief (embosseuse SAGEM).

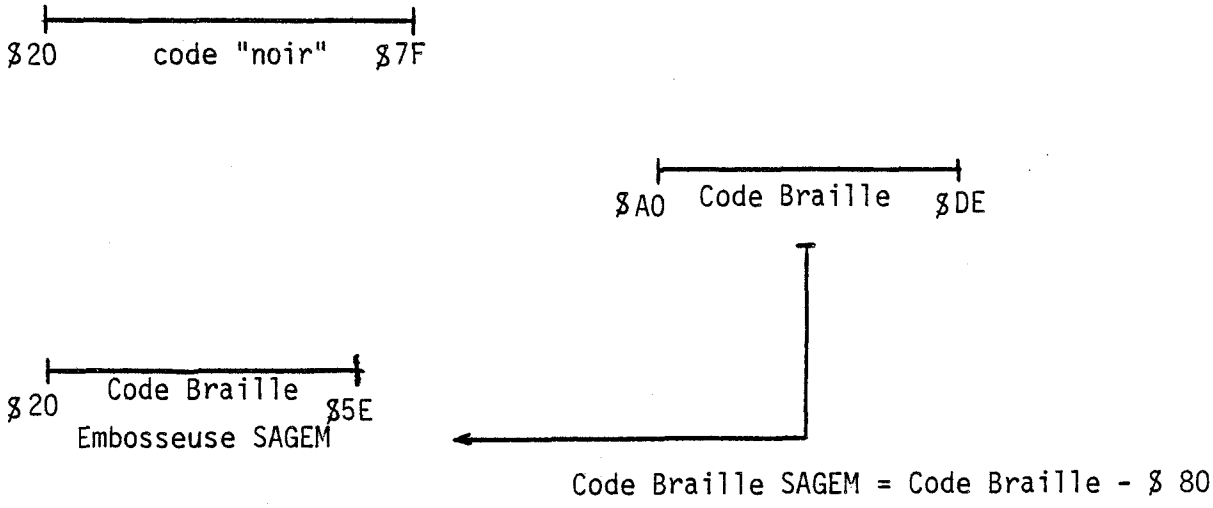


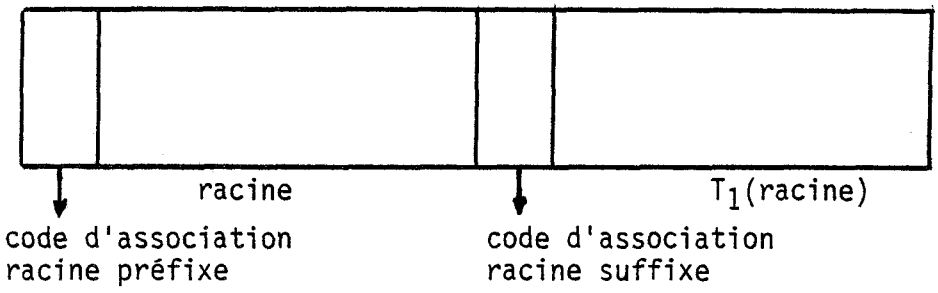
Figure 13

---

*Le signe "$" signifie valeur hexadécimale.

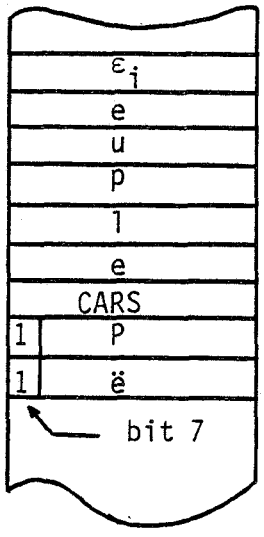
Structure des éléments du dictionnaire des racines :

La structure de chaque élément du dictionnaire des racines est de la forme :



Exemple d'un mot du dictionnaire implanté en mémoire :

Exemple du mot : "peuple" . Le mot peuple s'associe avec certains préfixes éléments de ε_i ; ces préfixes sont (re, dé, sur)



Code d'association racine préfixe

Représentation en ASCII du mot "peuple"

Code d'association racine suffixe

Code Braille de T_1("peuple")

Figure 14

Pour représenter en mémoire le ε_i , nous lui donnons une valeur arbitraire, mais inférieure à 7F pour ne pas le confondre avec les codes des caractères Braille du mot précédent (figure 14).

I-2-2- Structure des éléments du dictionnaire des préfixes

Le dictionnaire des préfixes est ce que nous avons appelé l'ensemble Dp. Nous avons vu que chaque élément p_i de cet ensemble est un n-uple de la forme :

$(p_i , \underbrace{\varepsilon_i , \varepsilon_j , \varepsilon_k \ldots \varepsilon_\ell}_{\text{n-1 éléments}})$

n dépend de chaque préfixe p_i

Les ε_i étant des sous-ensembles de Dp

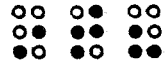
L'accès au dictionnaire des préfixes est séquentiel avec un seul point d'entrée. La sortie se fait lorsque le code ASCII du $i^{ème}$ caractère de la chaîne à transcrire est inférieur au code ASCII du $i^{ème}$ caractère d'une chaîne quelconque $p_j \in Dp$. C'est l'un des avantages de la représentation que nous avons choisie. La recherche d'un élément de ce dictionnaire est assez rapide, même si elle est séquentielle étant donné que son cardinal est assez petit.

Les caractères de chaque préfixe sont représentés par leurs codes ASCII. Deux octets contenant des "nuls",l'un permet de séparer les codes du préfixe de ceux représentant les ensembles ε_i et l'autre permet de séparer ceux-ci des codes des caractères Braille par lesquels on transcrit le préfixe considéré.

Nous avons décrit précédemment comment était ordonné Dp en vue d'une scrutation séquentielle et dans le but de la recherche du préfixe le plus long d'une chaîne donnée.

Le dictionnaire des préfixes ainsi que les sous-ensembles ε_i de Dp sont décrits en annexe

Exemple d'implantation d'un n-uple de Dp en mémoire :

préfixe p = "inter" et $T_1(p)$ = 

Les sous-ensembles de Dp auxquels appartient p sont ε_{12} et ε_{21} d'où l'implantation en mémoire (figure 15)

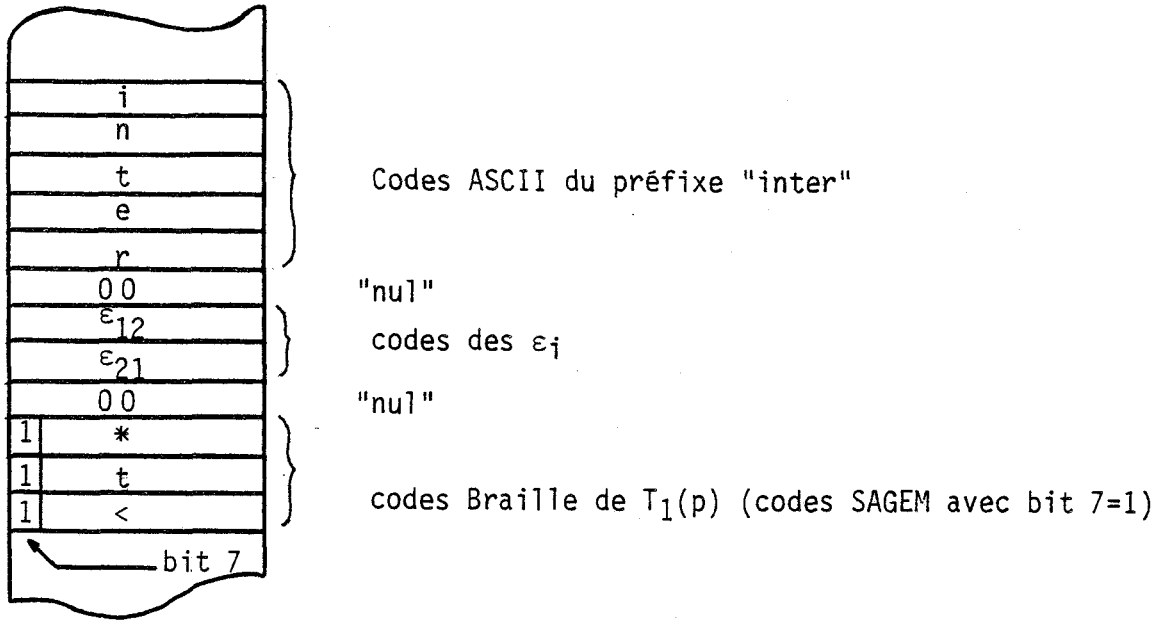


Figure 15

I-2-3- Structure des éléments du dictionnaire des suffixes

Le dictionnaire des suffixes est ce que nous avons appelé l'ensemble Ds. L'ensemble de ses éléments est donné en annexe . Il est implanté en mémoire sous forme d'une table ordonnée à accès séquentiel avec un seul point d'entrée.

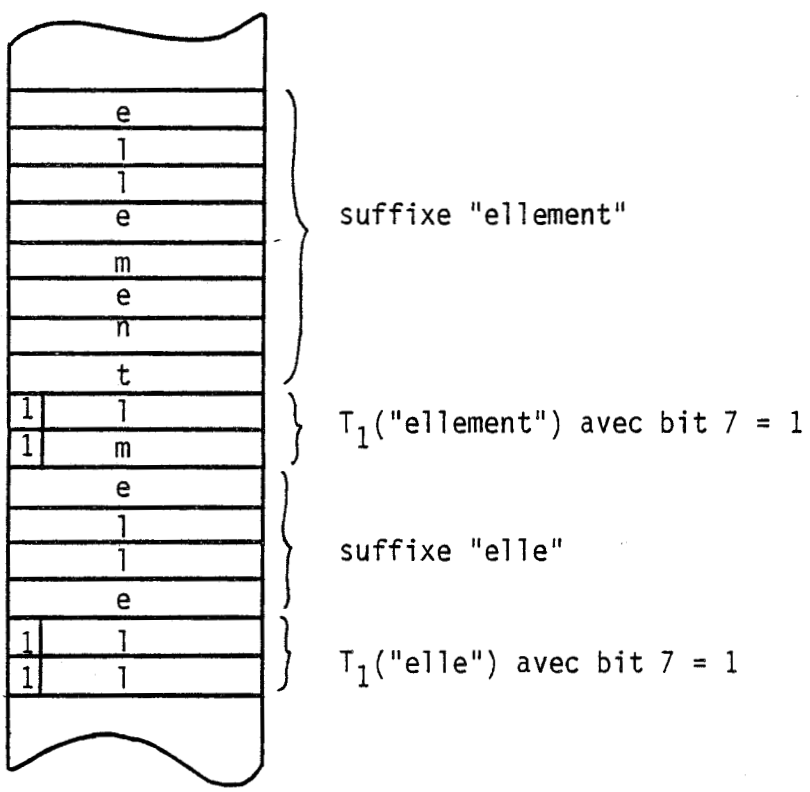
La sortie de la table s'effectue de la manière suivante : en appelant δ la chaîne dont il faut chercher le suffixe , si il existe un suffixe quelconque $S \in Ds$ tel que $S = s_1\ s_2\ \ldots\ s_m$

avec $\delta = v_1\ v_2\ \ldots\ v_n$

la sortie se fait lorsque $s_i > v_j$ avec $j = \ell(R(\delta)) + i$

Exemple d'implantation en mémoire de deux suffixes contigus

Soit à implanter en mémoire les deux suffixes "ellement" et "elle"



La recherche d'un suffixe se fait à l'aide de l'algorithme que nous avons donné au paragraphe I-1-2-5.

I-2-4- *Calcul de l'adresse des sous-tables du dictionnaire des racines*

Nous avons vu que le dictionnaire est divisé en sous-tables et chaque sous-table contient tous les mots et locutions commençant par la même lettre.

Nous disposons d'une table majeur dans laquelle nous avons rangé les adresses des sous-tables dans l'ordre alphabétique (Figure 16).

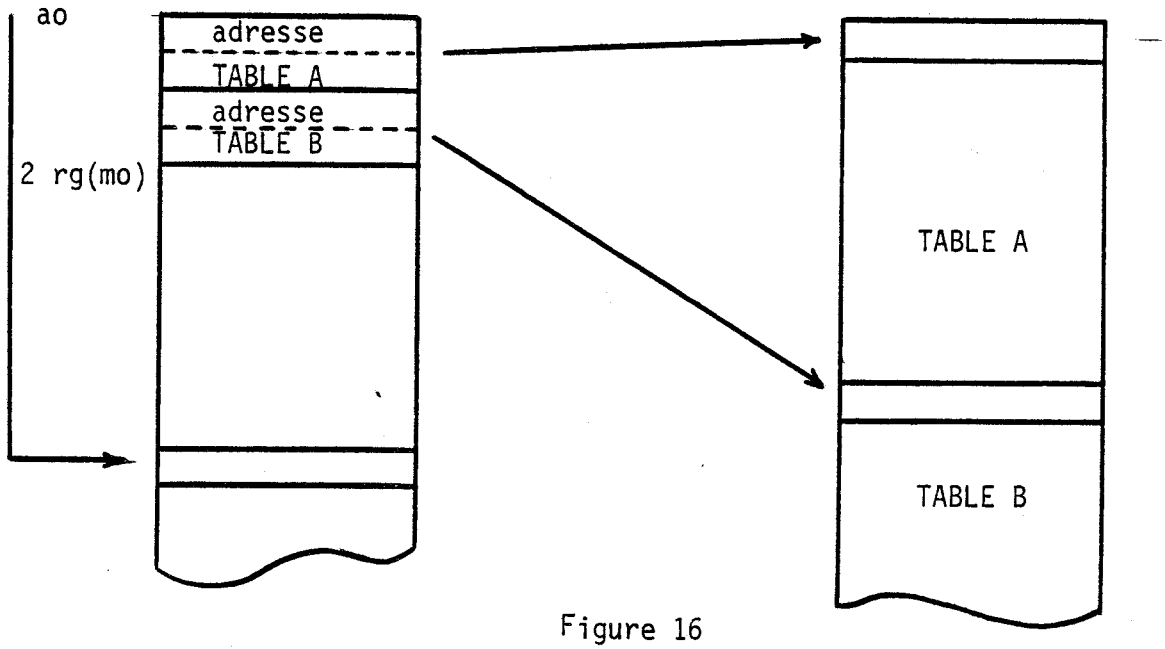


Figure 16

Si nous appelons C[x] : contenu de l'octet à l'adresse x

ao : adresse de début de la table majeur

rg(mo): rang dans l'alphabet de la première lettre du mot à transcrire

a' : adresse de la sous-table de toutes les chaînes ayant pour préfixe mo

On a :

$$a' = C\ [ao + 2rg(mo)]$$

avec $rg(mo) = ASCII(mo) - 61$

d'où $a' = C\ [ao + 2(ASCII(mo) - 61)]$

et cette expression n'est évidemment valable que pour $61 \leqslant ASCII(mo) \leqslant 7B$

Pour les mots ou locutions commençant par les lettres "œ", "à" , "ê" etc. le calcul précédent est remplacé par un test qui branche à une table d'adresse .

I-2-5- Evaluation du coût de la stratégie choisie

Le coût total d'une recherche séquentielle dans une liste de p objets de fréquence d'appel F(i) est :

$$C_T = \sum_{i=1}^{p-1} i\ F(i)$$ i = rang de l'élément dans la liste

Si l'on suppose que les p objets sont équiprobables et donc $F(i) = \frac{1}{p}\ \forall\ i$

on a : $C_T \simeq \frac{p}{2}$

Les mesures en temps de recherche d'un élément du dictionnaire ont donné (pour un microprocesseur ayant un cycle de durée 1 µs) :

C_T = 15 ms

C_{Tmax} = 40 ms

C_T = temps moyen de recherche d'un mot quelconque du dictionnaire pondéré par la longueur des tables,

C_{Tmax} = temps maximum de recherche du dernier mot de la table la plus longue (TABLE P).

I-2-6- Taille mémoire occupée

Ainsi constitué, le dictionnaire, les tables des préfixes et des suffixes occupent environ 7 K octets au lieu de 14 K octets ce qui procure une économie mémoire d'environ 50 %.

Le logiciel de transcription d'un mot préabrégé dans le dictionnaire occupe environ 1.5 K octets. L'organigramme général, le détail de ce logiciel et la signification de tous les registres utilisés sont donnés en annexe .

II - DECOMPOSITION D'UN MOT EN CONTRACTIONS

La décomposition d'un mot en contractions se fait lorsque l'on n'a pas trouvé le mot dans le dictionnaire (accompagné ou non d'un préfixe et / ou d'un suffixe).

Cette décomposition doit se faire en respectant les règles de transcription en Braille abrégé.

II-1- NOTATIONS UTILISEES

Soit δ une chaîne de V^* avec $\delta = v_1\ v_2 \ldots v_n \quad \forall i\ v_i \in V$

Nous noterons $\delta(i,j)$ la chaîne de caractères $v_i\ v_{i+1} \ldots v_j$ avec $i \leqslant j$ et $i,j \in [i,n]$

Nous noterons $\delta(j)$ ou $\delta(j,j)$ un élément $v_j \in V$ de la chaîne δ

$\delta(j,j) = \delta(j) = v_j$

On a alors $\ell(S(i,j)) = j-i+1$.

$\delta(i,j)$ est appelée contraction lorsque, faisant partie d'une chaîne plus longue, elle peut être abrégée (avec $i < j$ ou $\ell(\delta(i,j)) > 1$)

Nous noterons T le procédé de transcription par contraction. On peut alors écrire : si $\delta(i,j)$ est une contration $\ell(\delta(i,j)) > \ell\ T(\delta(i,j))$

Nous utiliserons également les notations suivantes :

<vℓ> : voyelle
<cn> : consonne
<sp> : séparateur de mots
Dc : ensemble ou dictionnaire des contractions
$Rg(\delta(i,j))$: numéro de règle à vérifier pour la contraction $\delta(i,j)$
$V(\delta(i,j))$: le procédé de vérification des règles de transcription de la contraction $\delta(i,j)$. $V(\delta(i,j))$ est une application de Dc (dictionnaire des contractions) dans [0,1]
$V(\delta(i,j))$ prend la valeur 1 lorsque les règles de transcription de $\delta(i,j)$ sont vérifiées et 0 dans le cas contraire.

II-2- POSITION DU PROBLEME ET JUSTIFICATION DE LA METHODE DE DECOUPE ADOPTEE DES MOT

Nous illustrons la méthode de décomposition d'un mot en contractions sur l'exemple du mot "cantique".

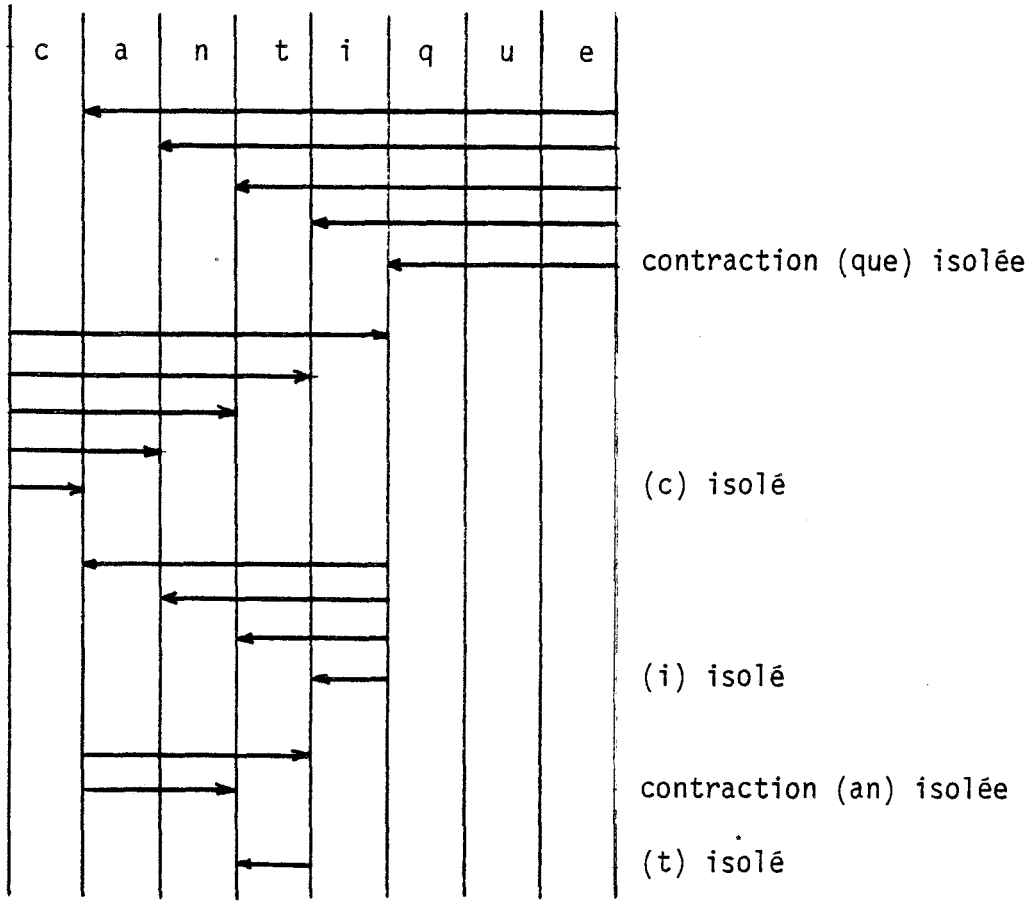
L'application des règles du Braille abrégé conduit à la décomposition suivante :

(c) -(an) - (t) - (i) - (que)

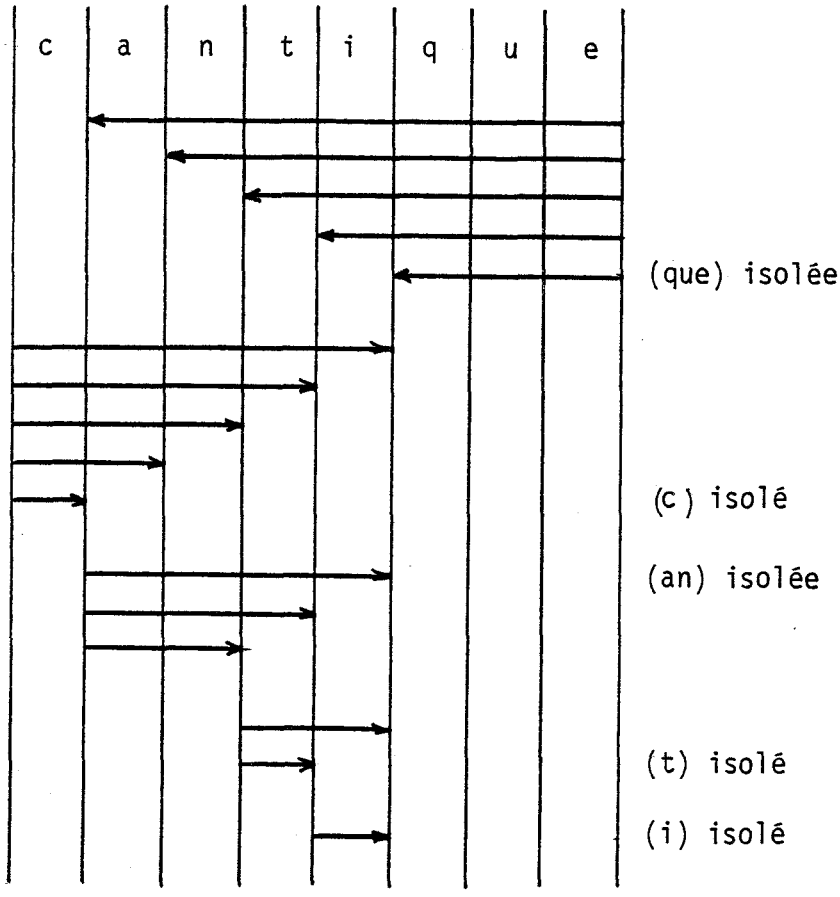
- la contraction "que" est isolée car elle se trouve en terminaison du mot
- la contraction "an" ne s'emploie pas isolément, ni en terminaison, mais devant consonne
- les lettres "c", "t", "i" sont transcrites en intégral

Les méthodes automatiques de découpe des mots commencent par l'isolement de la terminaison car celle-ci est souvent plus longue que les autres contractions du mot[6].

Un algorithme de découpe déjà utilisé[6] est celui illustré ci-dessous :



Pour des raisons de simplification du programme de découpe, nous avons choisi l'algorithme suivant.



Cet algorithme choisi impose de classer les contractions suivant 19 tables et chaque table correspond à une règle. Nous donnons ci-dessous un aperçu de ces tables.

01	02	03	04	05	06	07	08	09	11
ai	ant	bl	em	ll	ess	im	es	eur	ient

12	14	15	16	17	18	19	20	21	
er	ien	en	logi	com	gn	in	con	quin	

Ces tables sont destinées à faire le choix des groupements de lettres ou contractions en tenant compte des règles du Braille abrégé et des problèmes particuliers de traduction à résoudre.

II-3- ENUMERATION DES DIFFERENTES REGLES RELATIVES AUX CONTRACTIONS SIMPLES

II-3-1- Règle 01 (toujours)

Ce sont des chaînes de caractères qui s'abrègent toujours quel que soit leur emplacement dans le mot et quel que soit le caractère qui les précède ou qui les suit.

Pour ce type de contraction, c'est-à-dire lorsque $R_g(\delta(i,j)) = 01$ on a $V(\delta(i,j)) = 1$ quel que soit $\delta(j+1)$ et $\delta(i-1)$

exemple : "au" avec T("au") =
k

II-3-2- Règle 02 (terminaison)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées en terminaison d'un mot

exemple : "able" avec T("able") : 

Pour ce type de contraction, on a : $R_g(\delta(i,j)) = 02$

et $V(\delta(i,j)) = [\delta(j+1) = \text{<sp>}] \cup [\delta(j+1) = \text{"s"}] \cap [\delta(j+2) = \text{<sp>}]$

II-3-3- Règle 03 (devant voyelle)

Ce sont des chaînes de caractères qui ne s'abrègent que devant une voyelle.

exemple : "cl" avec T("cl") = 
î

Pour ce type de contraction, on a :

$Rg(\delta(i,j)) = 03$

et

$V(\delta(i,j)) = [\delta(j+1) = <vl>]$

II-3-4- Règle 04 (devant consonne)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées devant une consonne.

exemple : "pro" avec T("pro") = 
!

Pour ce type de contractions, on a :

$Rg(\delta(i,j)) = 04$

et

$V(\delta(i,j)) = [\delta(j+1) = <cn>]$

II-3-5- Règle 05 (entre voyelles)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées entre deux voyelles.

exemple : "ss" avec T("ss") =
è

Pour ce type de contractions, on a :

$Rg(\delta(i,j)) = 05$

et

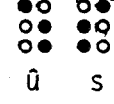
$V(\delta(i,j)) = [\delta(j+1) = <v\ell>] \cap [\delta(i-1) = <v\ell>]$

II-3-6- Règle 06 (début uniquement)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées au début d'un mot que ce soit devant consonne ou devant voylle

On a pour $Rg(\delta(i,j)) = 06$

$V(\delta(i,j)) = [\delta(i-1) = <sp>]$

exemple : "ess" avec T("ess") = 
û s

II-3-7- Règle 07 (début et devant consonne)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées au début d'un mot et devant une consonne.

On a pour $R_g(\delta(i,j)) = 07$

$$V(\delta(i,j)) = [\delta(i-1) = \langle sp \rangle] \cap [\delta(j+1) = \langle cn \rangle]$$

II-3-8- Règle 08 (début et terminaison)

Ce sont des chaînes de caractères qui ne s'abrègent que lorsqu'elles sont placées au début d'un mot que ce soit devant consonne ou devant voyelle ou en terminaison.

On a pour $R_g(\delta(i,j)) = 08$

$$V(\delta(i,j)) = [\delta(i-1) = \langle sp \rangle] \cup [\delta(j+1) = \langle sp \rangle] \cup [\delta(j+1) = \langle s \rangle] \cup [\delta(j+2) = \langle sp \rangle]$$

II-3-9- Règle 09 (devant consonne et terminaison)

Ce type de chaînes de caractères ne s'abrègent qu'en terminaison ou devant consonne lorsqu'elles sont dans le corps ou au début du mot.

On a pour $R_g(\delta(i,j)) = 09$

$$V(\delta(i,j)) = [\delta(j+1) = \langle cn \rangle] \cup [\delta(j+1) = \langle sp \rangle] \cup [\delta(j+1) = \langle s \rangle] \cap [\delta(j+2) = \langle sp \rangle]$$

II-3-10- Règle 12 (terminaison ou devant consonne, jamais au début ni devant <<)

Pour $R_g(\delta(i,j)) = 12$

$$V(\delta(i,j)) = [\delta(j+1) = \langle sp \rangle] \cap [\delta(j+1) \neq \langle \langle \rangle] \cup [\delta(j+1) = \langle s \rangle] \cap [\delta(j+2) = \langle sp \rangle] \cup [\delta(j+1) = \langle cn \rangle] \cap [\delta(i-1) \neq \langle sp \rangle]$$

II-3-11- Règle 13 (début devant voyelle)

Pour $Rg(\delta(i,j)) = 13$

on a $V(\delta(i,j)) = [\delta(i-1) = \langle sp \rangle] \cap [\delta(j+1) = \langle v\ell \rangle]$

II-3-12- Règle 14 (devant consonne et terminaison sauf devant point

Pour $Rg(\delta(i,j)) = 14$

on a $V(\delta(i,j)) = [\delta(j+1) = \langle cn \rangle] \cup [\delta(j+1) = \langle s \rangle] \cap [\delta(j+2) = \langle sp \rangle] \cup [\delta(j+1) = \langle sp \rangle] \cap [\delta(j+1) \neq \langle \bullet \rangle]$

II-3-13- Règle 15 (devant consonne et terminaison sauf devant point d'interrogatio

Pour $Rg(\delta(i,j)) = 15$

on a $V(\delta(i,j)) = [\delta(j+1) = \langle cn \rangle] \cup [\delta(j+1) = \langle s \rangle] \cap [\delta(j+2) = \langle sp \rangle]$

$\cup [\delta(j+1) = \langle sp \rangle] \cap [\delta(j+1) \neq \langle ? \rangle]$

II-3-14- Règle 16

C'est une règle particulière qui ne s'applique qu'à la chaîne "logi" qui ne peut s'abréger que lorsqu'elle fait partie de la chaîne "logiquement" ou de la chaîne "logique" mais jamais au début.

Pour $Rg(\delta(i,j)) = 16$

$V(\delta(i,j)) = [\delta(i-1) \neq \langle sp \rangle] \cap [\delta(j+1) = \langle q \rangle]$

II-3-15- Règle 17 (début et devant consonne sauf après le séparateur <—>

Pour $Rg(\delta(i,j)) = 17$

On a $V(\delta(i,j)) = [\delta(i-1) = \langle sp \rangle] \cap [\delta(i-1) \neq \langle — \rangle]$

$\cap [\delta(j+1) = \langle cn \rangle]$

II-3-16- Règle 18 (toujours sauf au début après le séparateur <(>)

Pour $Rg(\delta(i,j)) = 18$

on a $V(\delta(i,j)) = [\delta(i-1) \neq \langle sp \rangle] \cup [\delta(i-1) = \langle sp \rangle]$

$\cap [\delta(i-1) \neq \langle (\rangle]$

II-3-17- Règle 19 (toujours au début, dans le corps du mot devant consonne, en terminaison sauf devant le séparateur <*>)

Pour $Rg(\delta(i,j)) = 19$

on a $V(\delta(i,j)) = [\delta(i-1) = \langle sp \rangle] \cup [\delta(j+1) = \langle cn \rangle]$

$\cup [\delta(j+1) = \langle sp \rangle] \cap [\delta(j+1) \neq \langle * \rangle] \cup [\delta(j+1) = \langle s \rangle]$

$\cap [\delta(j+2) = \langle sp \rangle]$

II-3-18- Règle 20 (devant consonne, jamais en terminaison)

Pour $Rg(\delta(i,j)) = 20$

on a $V(\delta(i,j)) = [\delta(j+1) \neq \langle sp \rangle] \cap [\delta(j+1) = \langle cn \rangle]$

II-3-19- Règle 21 (dans le mot devant consonne, en terminaison sauf devant le séparateur <*>)

Pour $R(\delta(i,j)) = 21$

On a $V(\delta(i,j)) = [\delta(j+1) = \langle cn \rangle] \cup [\delta(j+1) = \langle sp \rangle]$

$\cap [\delta(j+1) \neq \langle * \rangle] \cup [\delta(j+1) = \langle s \rangle] \cap [\delta(j+2) = \langle sp \rangle]$

Mise à part la vérification des règles relatives à chaque contraction, il existe cependant d'autres problèmes dont quelques uns sont cités ci-dessous.

II-4- CONTRACTIONS COMPOSEES

II-4-1- Contrations type "ain"

Cette contraction est la combinaison de deux contractions qui sont "ai" et "in". Or, dans le cas du groupement "ain" dans un mot, deux solutions de découpe sont possibles :

(ai) - (n) et (a) - (in)

Chacune des découpes est utilisée suivant le son du groupement "ain" dans le mot

exemple : "plaine" → son ai → découpe (ai) - (n)
"main" → son in → découpe (a) - (in)

Une solution à ce problème consiste à structurer les tables de contractions et le programme de découpe de façon à donner la priorité à la contraction "ai" quand la chaîne "ain" est devant une voyelle.

Nous avons donc formé la contraction $\delta(i,j)$ = "ain" avec $V(\delta(i,j)) = [\delta(j+1) = \langle v\ell \rangle]$

Il en est de même pour les divers groupements suivants :

contraction : "uin" combinaison des contractions "ui" et "in"
" : "oin" " " " "oi" et "in"
" : "aim" " " " "ai" et "im"
" : "ren" " " " "re" et "en"

etc...

II-4-2- Terminaisons finissant par la lettre "e"

Ce sont des chaînes de caractères $\delta(i,j)$ ne pouvant être abrégées qu'en terminaison d'un mot avec $S(\delta(i,j))$ = "e" (exemple : "able", "que" etc...).

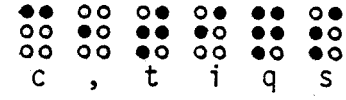
Ces chaînes de caractères peuvent être suivies de la lettre "s" pour former le pluriel des mots dont elles font partie.

Si l'on ne tient compte que des contractions définies par les règles officielles du Braille, une erreur de traduction systématique peut apparaître

exemple : cantiques → découpe → (c)-(an)-(t)-(i)-(qu)-(es)

Cette découpe est erronnée car la traduction sans erreur est :

(c)-(an)-(t)-(i)-(que)-(s)



Pour contourner cette difficulté, on ajoute pour ce cas la terminaison ("ques") dans la table des contractions, ainsi que sa traduction (deux graphèmes Braille)

Cette pratique conduit à ajouter six terminaisons à celles prévues par la règle.

II-4-3- Cas de répétition de singularités

Il faut tenir compte, après ou en cours de traduction, de singularités éventuelles qui peuvent conduire à des difficultés de compréhension de lecture de l'aveugle.

Exemple : cas du mot "drôle"

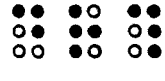
"drôle" → découpe (dr)-(ô)-(l)-(e)

Il se trouve que le caractère Braille qui traduit la contraction (dr) est celui qui représente la voyelle "ô"

Dans ce cas, il faut écrire le groupement (drô) comme en Braille intégral.

Pour éviter une analyse à postériori du mot traduit, on ajoute dans la table des contractions le groupement (drô) avec :

T("drô") =



C'est le cas également d'un certain nombre de groupements comme "êfl" etc..

II-5- AUTRES PROBLEMES PARTICULIERS A RESOUDRE

II-5-1- Terminaison type "ient"

C'est la règle n° 11. Suivant la prononciation de la terminaison "ient" (son "ien" ou "i") , la traduction est différente.

Exemples :

. "revient" → son "ien" → découpe (re)-(v)-(ien)-(t)
. "trient" → son "i" → découpe (tr)-(i)-(ent)

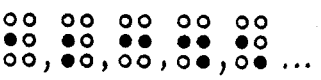
La résolution de ce problème nécessite l'analyse du reste du mot qui permet de préciser la nature du son de la terminaison.

Il subsiste néanmoins deux cas de conflit :
- un "expédient" et ils "expédient"
- il "convient" et ils "convient"

Ces deux cas ont été résolus en utilisant une communication homme-machine.

II-5-2- Mots traduits par des symbôles inférieurs

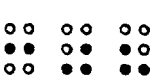
On appelle symbôle inférieur un caractère Braille qui n'est composé que d'une combinaison des points 2-3-5-6.

Exemple de symbôles inférieurs : 

Lorsqu'un mot est traduit, il faut s'assurer que les caractères Braille qui composent sa traduction ne sont pas tous inférieurs. Au quel cas, il peut y avoir pour l'aveugle une difficulté de lecture. Dans ce cas, il faut traduire la contraction la plus courte du mot en Braille intégral.

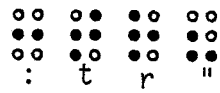
Dans l'algorithme de traduction que nous avons étudié, il est prévu cette correction automatique.

Exemple : soit à transcrire en Braille abrégé le mot "contrer". Ce mot est composé de trois contractions qui sont "con", "tr" et "er" et qui satisfont toutes aux règles de leur transcription. On a donc :

T("contrer") = 

Mais tous les symboles Braille de la transcription de ce mot sont inférieurs, ce qui est interdit en Braille abrégé. Dans ce cas, on transcrit en intégral la contraction "tr".

La transcription correcte du mot "contrer" est donc :



II-5-3- Mots devant être transcrits en intégral

La grande majorité des caractères Braille sont tels que l'ambiguité à la lecture d'un mot peut être levée par l'examen du caractère précédent ou suivant (ou les deux), en général sans connaissance de la langue française. Si, malgré l'application rigoureuse des règles relatives à chacune des contractions, il pouvait subsister un doute, le mot entier doit être transcrit en intégral précédé du signe d'intégral (point 6).

Exemples : "inouïe" qui abrégé pourrait se lire "inougre"
"aïeul" " " " " " "agreul"
etc...

Notre programme de traduction permet de tenir compte de ces particularités dans le cas où le contexte ne permet pas à la lecture de lever l'ambiguité .

C'est le cas par exemple des mots contenant un "ï" suivi d'une voyelle ou un "k" en terminaison ou suivi d'une consonne.

On constate d'après les algorithmes énumérés ci-dessus que traduire un mot par découpe en contraction nécessite, indépendamment de la manipulation d'index, un grand nombre de comparaisons de chaînes de caractères par scrutation dans les tables de contractions.

Le problème à résoudre consiste donc à déterminer une structure qui conduit à un temps minimum de traduction, tout en facilitant la correction d'éventuelles erreurs de traduction par l'apport de modifications de détail qui ne remettent pas fondamentalement en question le principe.

II-6- STRUCTURE DES REGISTRES ET CALCUL DU TEMPS DE TRANSCRIPTION

L'étude que nous avons faite a consisté à déterminer une structure (implantation de registres, structure des tables, algorithmes, mode de transfert d'information etc...) qui permet de résoudre les types de problèmes évoqués à l'aide d'un microprocesseur. L'ensemble doit conduire à un instrument autonome.

Dans ce cas, tenu de satisfaire des contraintes de temps d'exécution et d'encombrement mémoire minimum, il est nécessaire de choisir une structure de registres et algorithmes de découpe spécifiques fondamentalement différents

de ceux que l'on peut employer quand on a à sa disposition un gros ordinateur.

A l'origine, notre étude a été faite pour obtenir du Braille abrégé directement à partir de texte "noir". Mais la machine industrialisée devant transcrire ces textes "noir" en Braille intégral et en Braille abrégé, nous nous sommes rendus compte qu'il est plus simple d'obtenir le texte Braille abrégé à partir du texte Braille intégral. On évite ainsi le dédoublement de certaines fonctions telles que traitement des séparateurs, des chiffres, du souligné etc... étant donné que ces traitements ont déjà été conçus pour obtenir le texte Braille intégral.

II-6-1- Structure du dictionnaire des contractions et registre de transcription

L'ensemble des contractions est divisé en sept sous-ensembles, chaque sous-ensemble contenant des contractions de même longueur, ce qui permet une scrutation rapide lors de la recherche d'une contraction. Chaque élément de ces sous-ensembles est de la forme :

$$\left\{\delta(i,j)\ ,\ if,\ Rg(\delta(i,j))\ ,\ T(\delta(i,j))\right\}$$

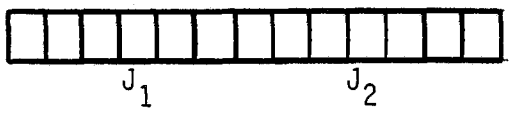
"if" est un indicateur égal à 1 lorsque $T(\delta(i,j))$ ne contient aucun symbole supérieur et égal à 0 lorsque $T(\delta(i,j))$ contient au moins un symbole supérieur .

La lecture d'un registre Π_{if} contenant le produit des if correspondant à chaque contraction permet de savoir si la transcription d'une chaîne α contient au moins un symbole supérieur.

Dans le cas contraire, la transcription de la contraction la plus courte est refaite de façon à satisfaire $\Pi_{if} = 0$.

L'ensemble des contractions étant divisé en sept tables, l'accès à chacune de ces tables se fait par l'intermédiaire d'une table majeure contenant leur adresse ainsi que le pas de saut d'un élément à l'autre de la table des contractions, ce pas étant égal à $\ell(\delta(i,j)) + \ell(T(\delta(i,j)) + 2$. Les paramètres if et $Rg(\delta(i,j))$ occupent chacun 1 octet.

Le calcul de l'adresse de chaque table se fait de la façon suivante. J_1 et J_2 étant deux index pointant le début et la fin de la contraction à identifier.



b' adresse de la table de la contraction à identifier

b_o adresse d'implantation de la table majeure

on a : $b' = C[b_o + 3((J_2 - J_1 + 1) - 2)]$

$b' = C[b_o + 3(J_2 - J_1 - 1)]$

la notation $C[x]$ signifie le contenu de la mémoire à l'adresse x.

L'ensemble de ces tables et registres est organisé autour du programme de transcription comme indiqué figure 17

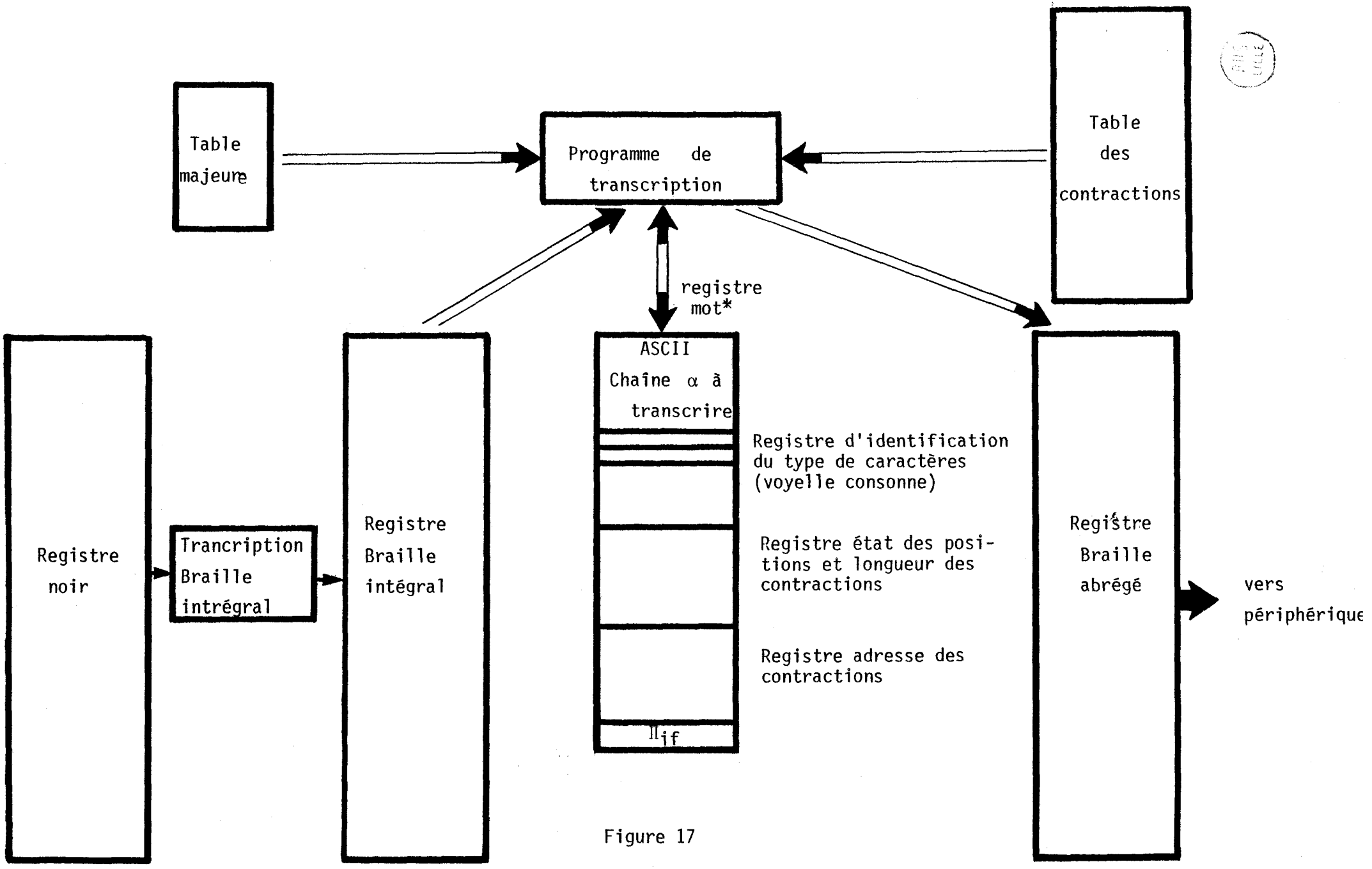
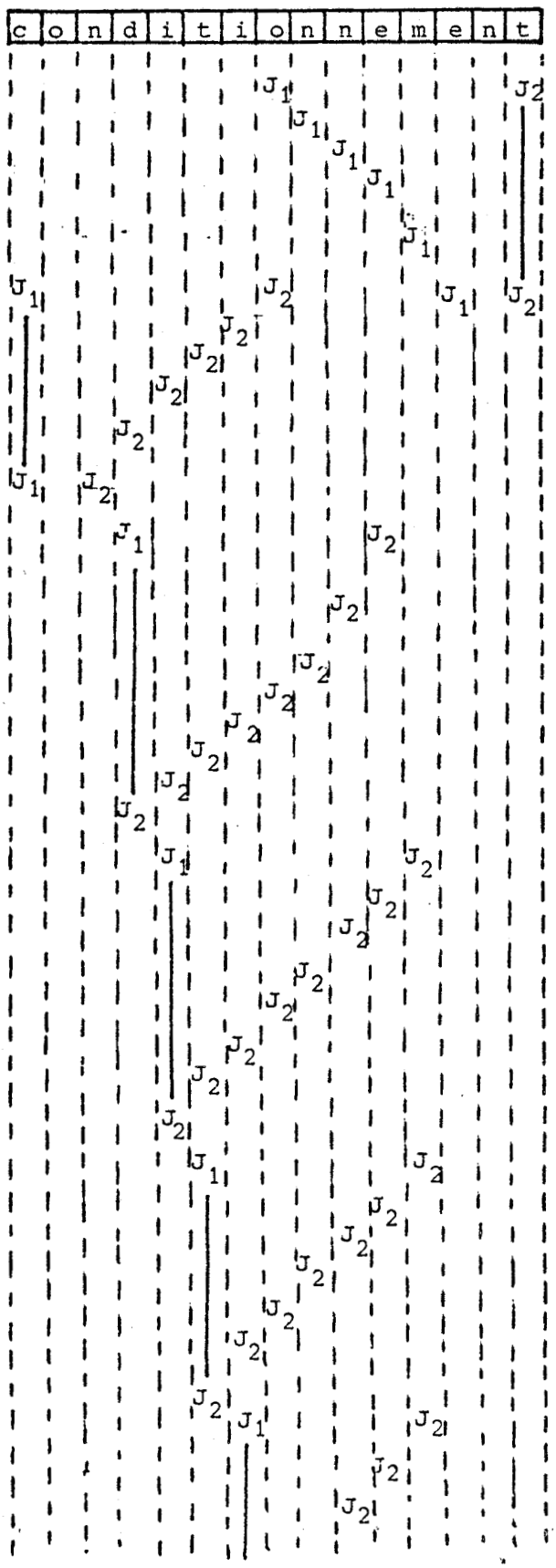


Figure 17

* Le registre mot contient toutes les informations résultant de la découpe en contractions et nécessaires à la transcription.

II-6-2- Calcul du temps de traduction

La figure ci-dessous illustre la traduction du mot "conditionnement"



ℓ : longueur du mot noir

$\ell > 8 \rightarrow J_2 = \ell - 1$, $J_1 = \ell - 8$

$J_1 - 1 \rightarrow SAVJ_1$

$SAVJ_1 > 8 \rightarrow J_1 = 0$, $J_2 = 7$

$J_2 + 1 \rightarrow SAVJ_2$

$(SAVEJ_1 - SAVEJ_2) > 7 \rightarrow J_1 = SAVEJ_2$

$J_2 = J_1 + 7$

$J_2 + 1 \rightarrow SAVJ_2$, $J_1 = SAVJ_2$

$SAVJ_1 - J_1 = 7 \rightarrow J_2 = SAVJ_1$

$J_2 + 1 \rightarrow SAVJ_2$ $J_1 = SAVJ_2$

$J_2 = SAVJ_1$

$J_2 + 1 \rightarrow SAVJ_2$

$J_1 = SAVJ_2$, $J_2 = SAVJ_1$

etc...

Finalement, nous aurons la découpe suivante :

(con)(d)(i)(t)(i)(on)(n)(e)(m)(ent)

On constate à partir de cette illustration que lorsque le mot à traduire ne comporte pas de contractions, l'algorithme conduit à la traduction du mot lettre par lettre. Or, cette traduction lettre par lettre conduit à un temps maximum de traduction du mot.

Le problème consiste à connaître le temps maximum de découpe, et comparer ce temps au débit d'entrée des caractères au clavier.

Ce temps, qui constitue un des résultats de faisabilité de la transcription Braille abrégé en temps réel peut être déterminé à l'aide du calcul suivant :
(ce temps peut être interprété comme un majorant du temps de traduction)

- Soit la plus longue contraction de longueur ℓ lettres
- Soit la plus courte contraction de longueur = 2 lettres
- Soit un mot à traduire de longueur N
  avec $N > \ell$

Nous savons que le coût C_j de recherche séquentielle dans une table de longueur N_j (j étant la longueur de contraction de la table) est donné par la relation :

$$C_j = \sum_{i=1}^{N_j-1} i \; F(i)$$

F(i) étant la fréquence d'appel des éléments de la table. Si tous ces éléments sont équiprobables, on a :

$$C_j \# \frac{N_j}{2}$$

1er coup → $C_1 = \sum_{j=2}^{\ell} C_{1j}$ il reste à traduire N-1 lettres

2è coup → $C_2 = \sum_{j=2}^{\ell} C_{2j}$ " " " " N-2 "

$k^{ème}$ coup → $C_k = \sum_{j=2}^{\ell} C_{kj}$ " " " " N-k "

$(N-\ell)^{ème}$ coup → $C_{N-\ell} = \sum_{j=2}^{\ell} C_{N-\ell\, j}$ " " " " ℓ "

$[N-(\ell-1)]^{ème}$ coup → $C_{N-\ell+1} = \sum_{j=2}^{\ell} C_{N-\ell+1\, j}$ " " " " $\ell-1$ "

Le coût des $\ell-1$ lettres à traduire $= \sum_{i=1}^{\ell-2} \sum_{j=2}^{\ell-i} C_{N-\ell+i_j}$

Le coût total $C_T = \sum_{i=1}^{N-\ell+1} \sum_{j=2}^{\ell} C_{ij} + \sum_{i=1}^{\ell-2} \sum_{j=2}^{\ell-i} C_{N-\ell+i_j}$

or $C_{ij} = C_{kj} = C_j \; \forall \; k, i$

$$\Rightarrow C_T = (N-\ell+1) \sum_{j=2}^{\ell} C_j + \sum_{i=1}^{\ell-2} \sum_{j=2}^{\ell-i} C_j$$

avec $C_j = \frac{N_j}{2}$

$$C_T = (N-\ell+1) \underbrace{\sum_{j=2}^{\ell} \frac{N_j}{2}}_{\frac{N_T}{2}} + \sum_{i=1}^{\ell-2} \sum_{j=2}^{\ell-i} \frac{N_j}{2}$$

N_T : nombre total des contractions

$$C_T = (N-\ell+1)\frac{N_T}{2} + \sum_{i=1}^{\ell-2}\sum_{j=2}^{\ell-i}\frac{N_j}{2}$$

où N : longueur du mot à traduire

ℓ : longueur de la contraction la plus longue

Si on applique ce calcul au mot le plus long de la langue française qui comporte 25 lettres, on obtient $C_T \# 1100$ unités de coût.

Nous avons déterminé une unité de coût à partir de notre programme de traduction et elle est en moyenne égale à 100 µs.

On aboutit à un temps maximum de traduction du mot le plus long (lorsque celui-ci ne comporte aucune contraction) de 100 ms.

A ce temps de traduction, il faut ajouter le temps de recherche dans le dictionnaire qui est au maximum de 40 ms.

La fréquence d'entrée des lettres au clavier est au maximum de 5 lettres par seconde, il en résulte dans ces conditions qu'un mot sera traduit entre l'instant de l'entrée de sa dernière lettre et l'instant de l'entrée du premier caractère du mot suivant.

TROISIEME PARTIE

DESCRIPTION TECHNOLOGIQUE DU SYSTÈME ET ARCHITECTURES D'APPLICATION DE L'AUTOMATE

I - INTRODUCTION

L'étude que nous avons menée a pour but de réaliser un dispositif autonome permettant à partir de la saisie au clavier l'enregistrement des textes Braille intégral, Braille abrégé et noir. Ces textes enregistrés sont destinés à être exploités, soit pour imprimer le texte noir en grands caractères, soit pour reproduire en relief les textes Braille (embossage, thermogravure, thermoformatage etc...).

Tout au long de notre étude, nous avons été partagés entre la possibilité d'utiliser des matériels commercialisés et la possibilité de concevoir nous-mêmes des architectures adaptées au problème.

Compte tenu des solutions que nous avons choisies et de la technologie que nous avons mise en oeuvre (1979) en fonction de l'analyse du problème et de la définition de l'objectif à atteindre, nous nous sommes rendus compte qu'au moment où nous avons eu besoin des matériels qui équipent à présent le Logibraille, ces matériels n'étaient pas encore commercialisés.

Nous faisons remarquer ici que cette démarche nous a permis de mettre au point une architecture de gestion de visualisation de caractères originale et qui n'existe pas sur le marché en 1982 (Euromicro).

L'analyse du problème et l'objectif à atteindre nous ont fait mettre en oeuvre une technologie très moderne pour l'époque.

De ce fait, beaucoup de fonctions microinformatiques n'existaient pas sous forme commercialisée au moment où nous en avons eu besoin. Ces conditions nous ont obligé à concevoir pour nous-mêmes la plupart des circuits qui équipent actuellement le Logibraille.

Nous restons néanmoins tout à fait attentifs aux matériels qui apparaissent sur le marché et qui pourraient remplacer la technologie actuelle.

Il faut reconnaître que si nous avions attendu le matériel idéal répondant à toutes les exigences du problème posé, nous n'aurions pas pu atteindre les performances actuelles (coût, temps de traduction, taille mémoire, adaptation aux périphériques etc...).

Ces différentes décisions permettent de posséder la connaissance parfaite de la technologie des circuits et de pouvoir l'adapter facilement à diverses configurations possibles d'utilisation de l'automate.

Dans cette troisième partie, nous décrivons en détail la technologie des circuits appliquée au cas de la configuration d'une station autonome de transcription et de duplication Braille. Cette configuration intéresse surtout les établissements ou organismes tels que les écoles et les associations.

Il est très possible d'imaginer une version miniaturisée comme nous le décrivons dans ce chapitre, mais cela ne fait pas l'objet d'une recherche mais d'un développement purement industriel.

II - SYNOPTIQUE DE LA CONFIGURATION STATION AUTONOME

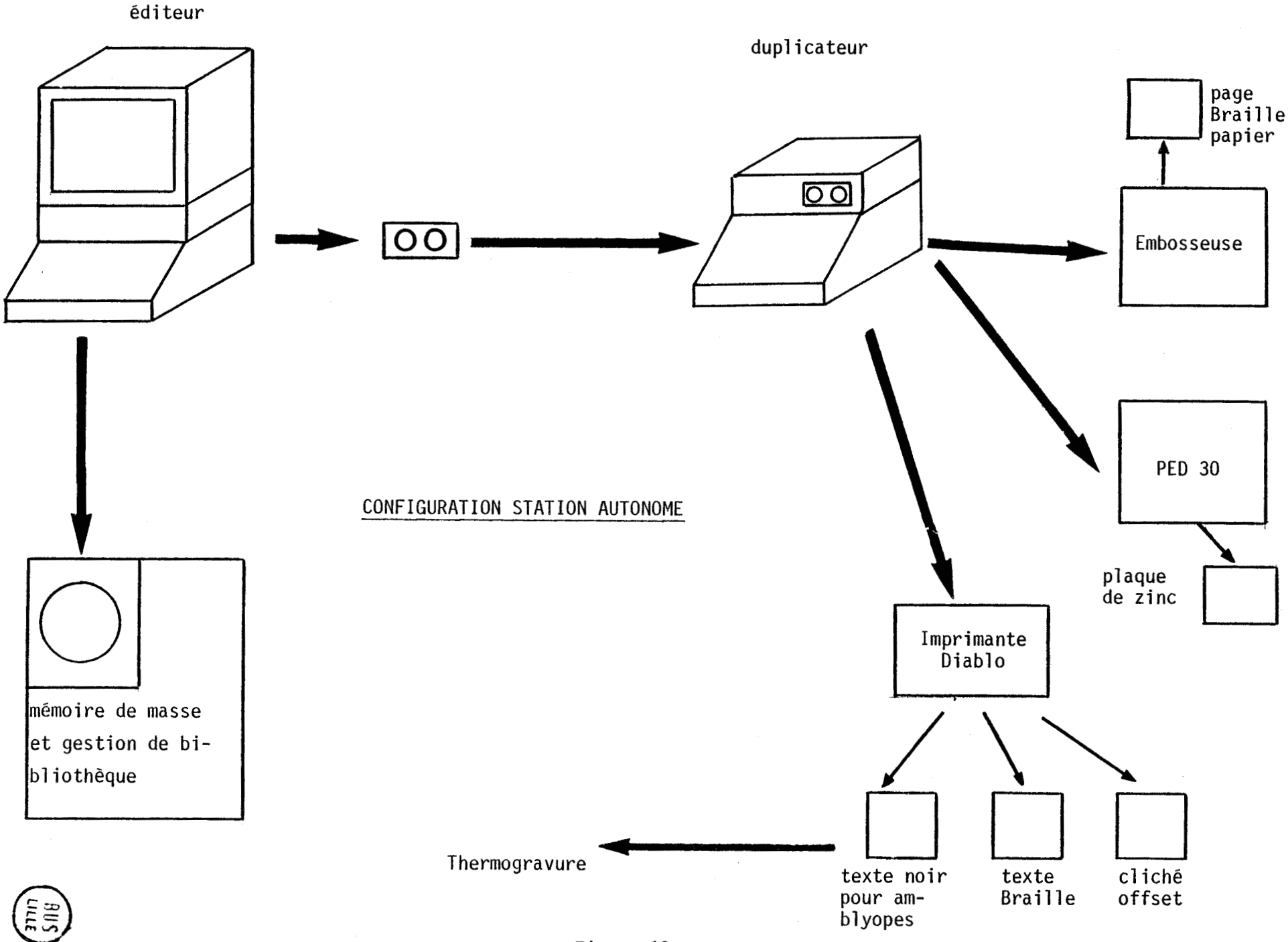
L'ensemble se compose de deux parties : l'éditeur et le duplicateur (figure 18).

Nous avons choisi la séparation physique de la fonction d'édition de celle de la duplication pour plusieurs raisons : la première raison vient des utilisateurs qui ont souhaité cette configuration leur permettant de séparer la production de la saisie (dans un but d'organisation du travail). Il est possible d'étudier un dispositif permettant de faire, parallèlement à la saisie et la transcription en Braille, la duplication (duplication des textes venant d'être édités ou édités antérieurement) moyennant un logiciel assez complexe et des dialogues homme-machine assez nombreux qui risqueraient de rendre l'utilisation du dispositif assez lourde et demanderait une spécialisation de l'opérateur.

Malgré un coût un peu plus élevé, cette structure a l'avantage d'être d'utilisation beaucoup plus simple.

II-1- DESCRIPTION DES FONCTIONS DE L'EDITEUR

A la mise sous tension, un programme d'initialisation affiche sur l'écran l'ensemble des fonctions dont l'utilisateur dispose. Chacune de ces fonctions est définie sur la figure 19 .



CONFIGURATION STATION AUTONOME

Figure 18

(E) Edition avec référence	Création de références avant édition d'une page
(C) Edition sans référence	L'édition sans référence est prévue pour reprendre sur une même cassette la suite d'un travail d'édition
(L) Lecture	Relecture sur l'écran des textes enregistrés sur cassette
(F) Format	Mise au format du texte Braille abrégé lu sur une cassette et stocké sur une autre cassette
(T) Test cassette	Enregistrement Test permettant de vérifier le bon fonctionnement de l'appareil (enregistrement et lecture du contenu d'une cassette)
(R) Test lecture	Permet de contrôler si le contenu de la cassette est correct (visualisation sur l'écran), de repositionner la cassette à l'endroit souhaité (reprise d'édition)

Figure 19

Nous décrivons brièvement chacune de ces fonctions de l'éditeur.

II-1-1- *Edition avec référence (E)*

L'organigramme général de cette fonction est représenté figure 20. Cette commande se caractérise par un dialogue initial entre l'utilisateur et la machine. Ce dialogue lui permet de choisir à un premier niveau le périphérique sur lequel seront enregistrés les textes saisis et d'effectuer les manoeuvres des platines d'enregistrement. A un deuxième niveau, l'utilisateur enregistre les références de sa saisie et choisit le format d'édition. L'enregistrement d'une page éditée s'effectue en même temps que s'enregistrent les textes de la page précédente. Cet enregistrement se pratique en mode interruptif.

La fonction d'édition sans référence se déroule de la même manière que l'édition avec référence (E). Dans ce cas, le deuxième niveau est emputé de la fonction d'enregistrement des références. Cette fonction est prévue pour permettre à l'utilisateur la reprise d'un travail.

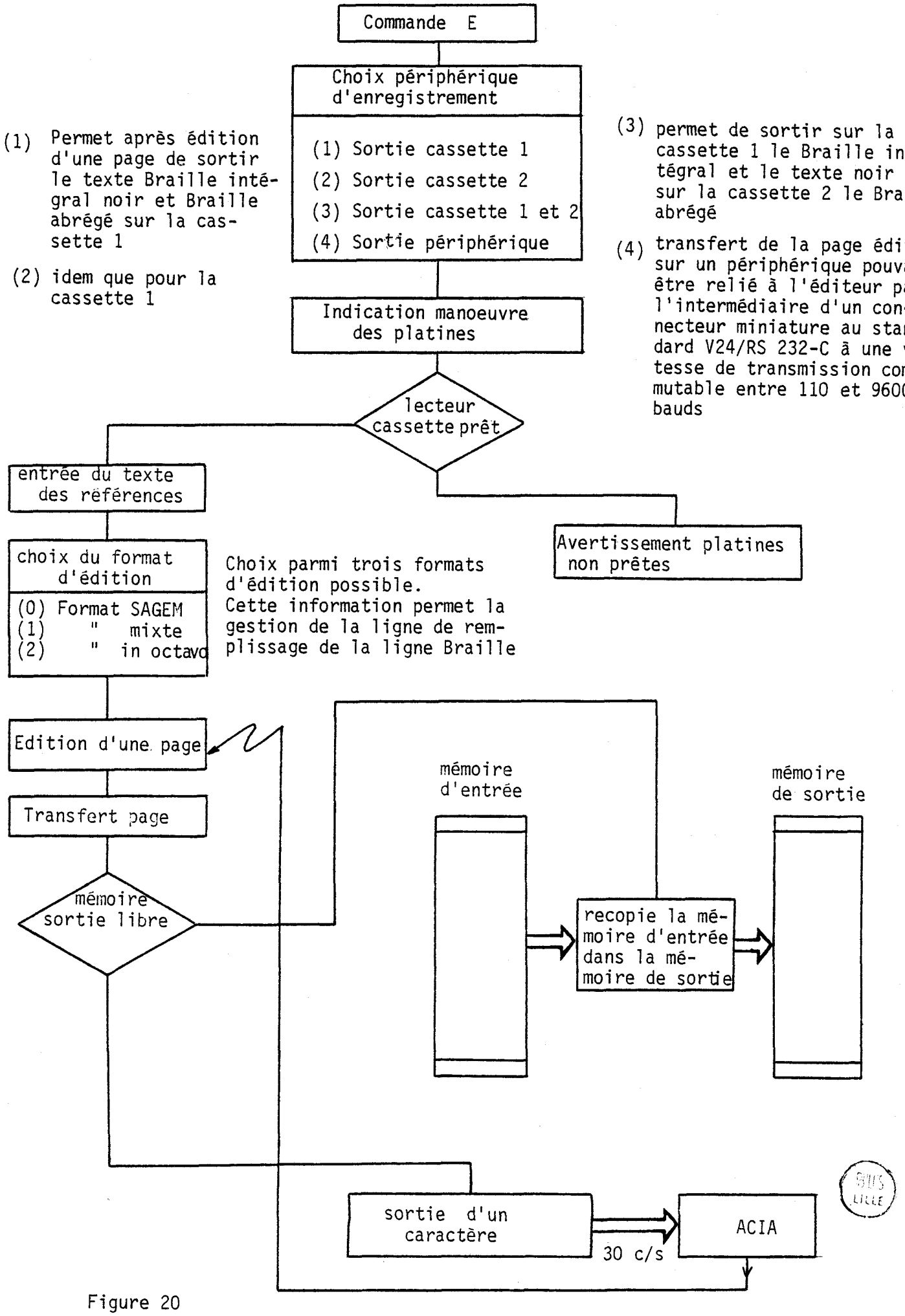

Commande E
Choix périphérique d'enregistrement
(1) Sortie cassette 1
(2) Sortie cassette 2
(3) Sortie cassette 1 et 2
(4) Sortie périphérique
(1) Permet après édition d'une page de sortir le texte Braille intégral noir et Braille abrégé sur la cassette 1
(2) idem que pour la cassette 1
(3) permet de sortir sur la cassette 1 le Braille intégral et le texte noir et sur la cassette 2 le Brail abrégé
(4) transfert de la page éditée sur un périphérique pouvant être relié à l'éditeur par l'intermédiaire d'un connecteur miniature au standard V24/RS 232-C à une vitesse de transmission commutable entre 110 et 9600 bauds
Indication manoeuvre des platines
lecteur cassette prêt
entrée du texte des références
Avertissement platines non prêtes
choix du format d'édition
(0) Format SAGEM
(1) " mixte
(2) " in octavo
Choix parmi trois formats d'édition possible.
Cette information permet la gestion de la ligne de remplissage de la ligne Braille
Edition d'une page
Transfert page
mémoire sortie libre
mémoire d'entrée
mémoire de sortie
recopie la mémoire d'entrée dans la mémoire de sortie
sortie d'un caractère
30 c/s
ACIA

Figure 20

II-1-2- Formatage (F)

Lors de l'édition, les trois textes, noir, Braille intégral et Braille abrégé sont sauvegardés sur cassette, mais comme par définition en moyenne le texte Braille abrégé est plus court que le texte intégral et "noir", la page Braille abrégé est toujours incomplète. Pour sortir le texte Braille abrégé sur un périphérique tel qu'une embosseuse SAGEM ou autre, il est nécessaire de procéder à un formatage comme l'indique la figure 21.

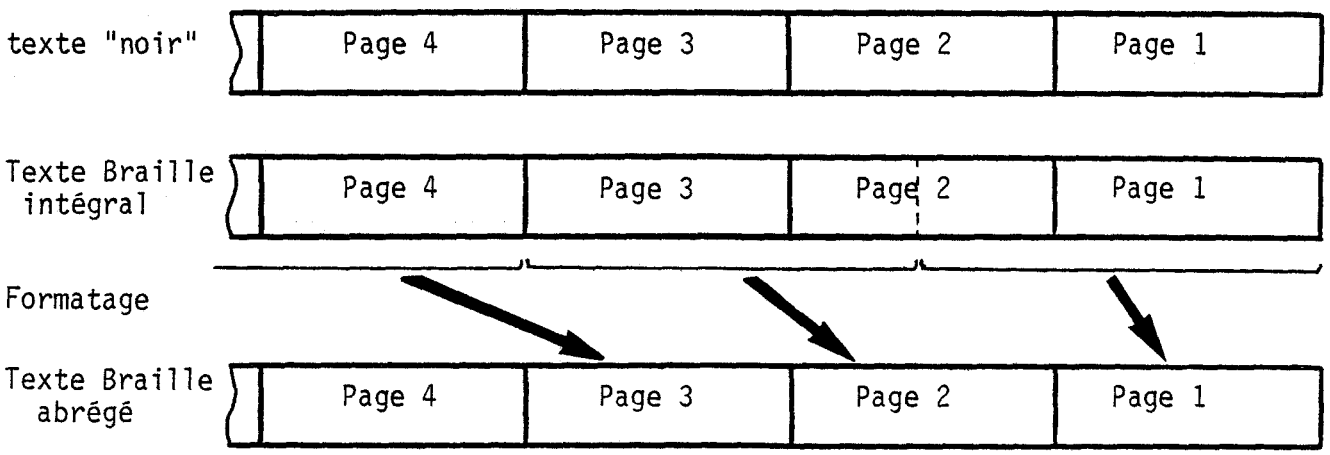


Figure 21

L'opération de formatage permet le passage d'un texte Braille abrégé sous la forme de N caractères par ligne et M ligne par page à N' caractères par ligne et M' lignes par page ; elle permet également la numérotation des pages Braille abrégé qui est évidemment différente de celle de la page d'origine. L'opération de formatage se déroule comme indiqué sur la figure 22 .

Après un dialogue permettant à l'utilisateur d'entrer les paramètres de formatage (format final, n° première page du texte final etc,..), l'éditeur relit le texte Braille abrégé en respectant les mises en page du texte d'origine. En effet, la conservation de la mise en page est assurée si à la saisie du texte origine, l'utilisateur a respecté les règles suivantes : le retour forcé à la ligne et pratiqué à chaque ligne commençant par au moins un espace. Il en résulte qu'un saut de lignes s'effectue à l'aide de la séquence (RC ⌴ RC ⌴ RC ...) . Le retour forcé à la page s'effectue lorsqu'à l'édition le transfert page s'est pratiqué par la commande (Shift "Transfert page").

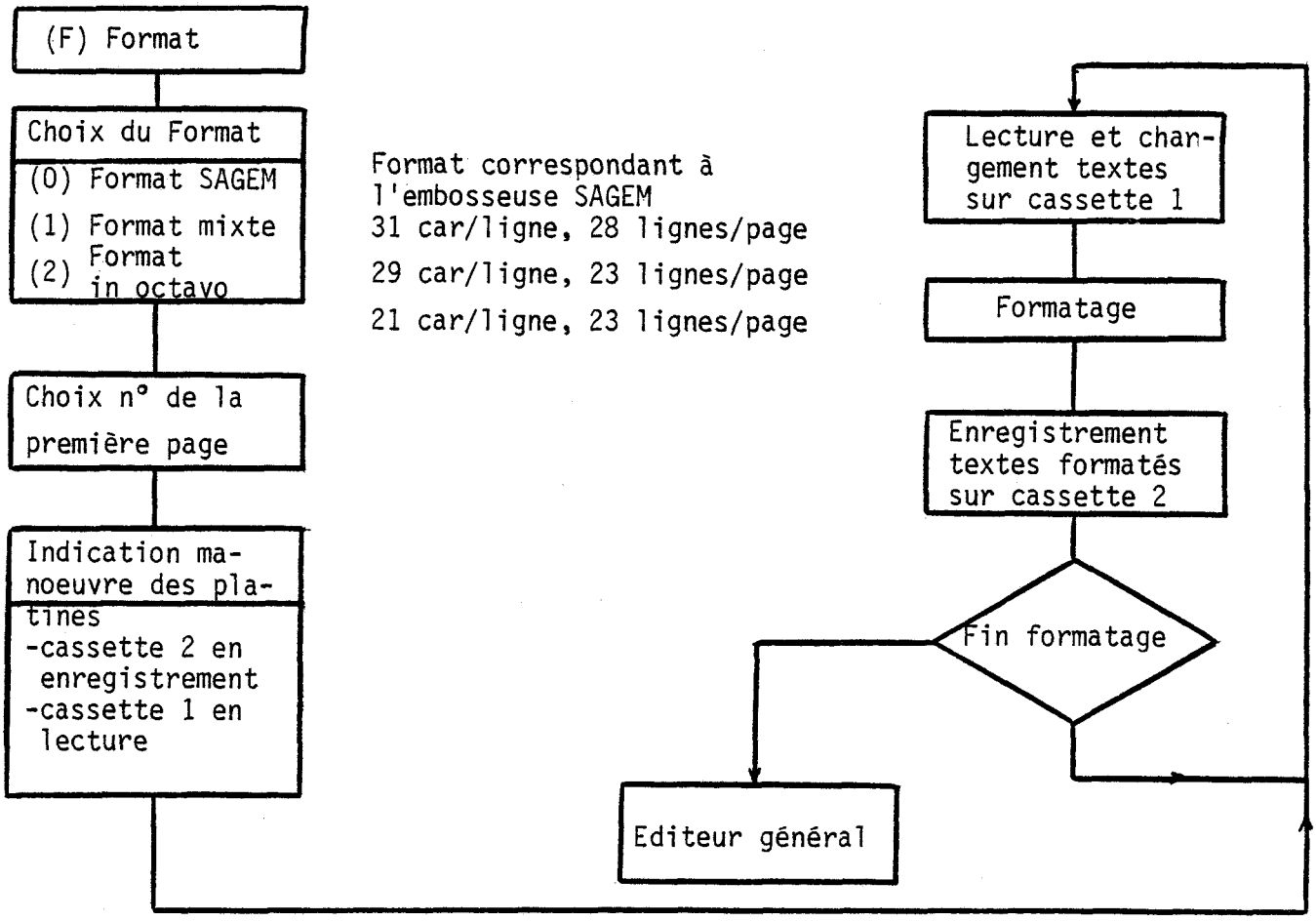


Figure 22

II-1-3- *Test cassette (T)*

Cette opération permet de vérifier le bon fonctionnement du modulateur démodulateur. Elle consiste à écrire automatiquement un message sur une cassette. L'utilisation de la fonction lecture permet alors de vérifier le bon fonctionnement de l'ensemble modulateur démodulateur.

II-1-4- *Test lecture (R)*

Cette commande permet d'une part de faire le test de la lecture après une opération de "test cassette". D'autre part, elle permet de reprendre la suite d'un travail en positionnant la tête de lecture cassette juste à la fin d'une page précédemment entrée. Enfin, elle délivre sur l'écran les informations envoyées par un périphérique.

II-1-5- Lecture (L)

Cette commande permet de lire sur la platine 1 ou sur la platine 2 les textes "noir" seuls et de les visualiser sur l'écran. Elle permet également la visualisation de textes en provenance d'un périphérique par l'intermédiaire d'une entrée/sortie série asynchrone V24.

II-2- DESCRIPTION DES CIRCUITS DE L'EDITEUR

La technologie a été choisie pour permettre l'usage de matériels commercialisés de qualité industrielle. Pour cette raison, il a été décidé de concevoir les circuits sous forme modulaire au format normalisé type EUROPA.

Dans cette technologie, diverses fonctions sont commercialisées et nous avons choisi le maximum de fonctions existantes pour constituer l'appareillage que nous avons mis au point.

Le schéma synoptique des fonctions de l'éditeur est donné ci-dessous (figure 23).

Nous avons dû concevoir les fonctions : gestion de visualisation, modulateur démodulateur cassette et mémoire RAM 8 K octets à double accès.

Description : L'ensemble de l'éditeur se compose de quatre unités distinctes reliées aux trois bus données, adresse et contrôle de l'unité centrale.
Cette unité est constituée d'un microprocesseur du type 6809 de MOTOROLA.

Nous avons choisi ce microprocesseur pour sa richesse des modes d'adressage qui conviennent très bien à la résolution de notre problème. Associé à une horloge d'une fréquence de 1 MHz, ce qui donne des cycles de 1 µs (possibilité de monter jusqu'à 2 MHz). Sa rapidité est largement suffisante pour les problèmes que nous avons à résoudre.

Les modules composant l'éditeur sont :

- un module de mémoire vive (RAM) d'une capacité de 8 K octets,
- un module de mémoire morte (EPROM) d'une capacité de 28K octets,
- des circuits d'entrée/sortie permettant la liaison de l'unité centrale avec l'extérieur (lecteurs-enregistreurs de cassette, clavier ou tout autre périphérique par l'intermédiaire d'un connecteur V24),
- un module de gestion de la visualisation sur écran.

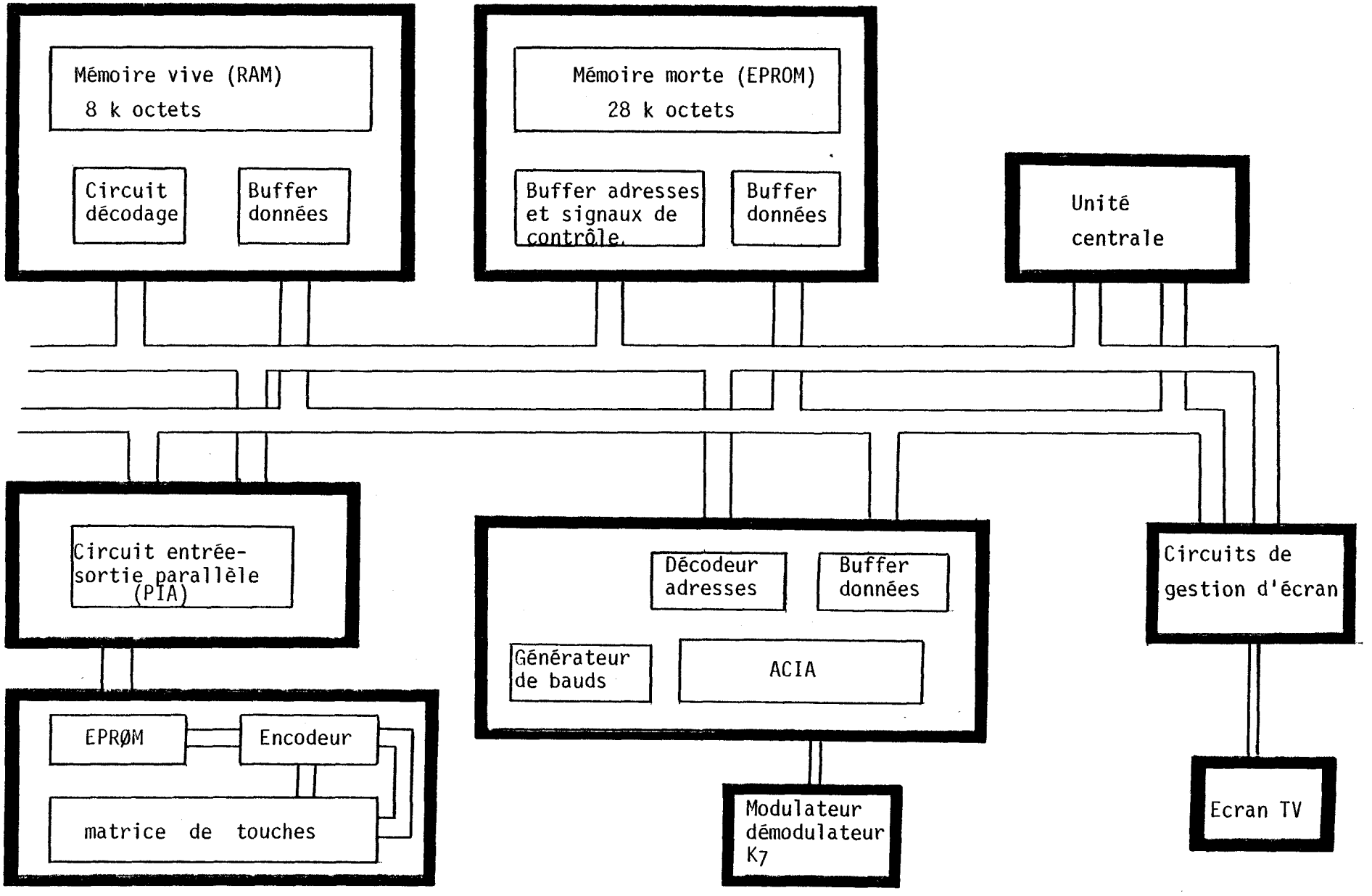

Mémoire vive (RAM)
8 k octets
Circuit décodage
Buffer données
Mémoire morte (EPROM)
28 k octets
Buffer adresses et signaux de contrôle
Buffer données
Unité centrale
Circuit entrée-sortie parallèle (PIA)
Décodeur adresses
Buffer données
Générateur de bauds
ACIA
Circuits de gestion d'écran
EPRØM
Encodeur
matrice de touches
Modulateur démodulateur K7
Ecran TV

Figure 23

II-2-1- Description des circuits de la carte de gestion de visualisation

Le principe de la conception de ce circuit a été exposé dans la première partie. Nous décrivons ici le détail de l'architecture réalisant cette fonction figure 24 .

II-2-2- Module de mémoire vive

Ce module se compose de 8 K octets de mémoire RAM statique. Cet espace mémoire se décompose comme suit :

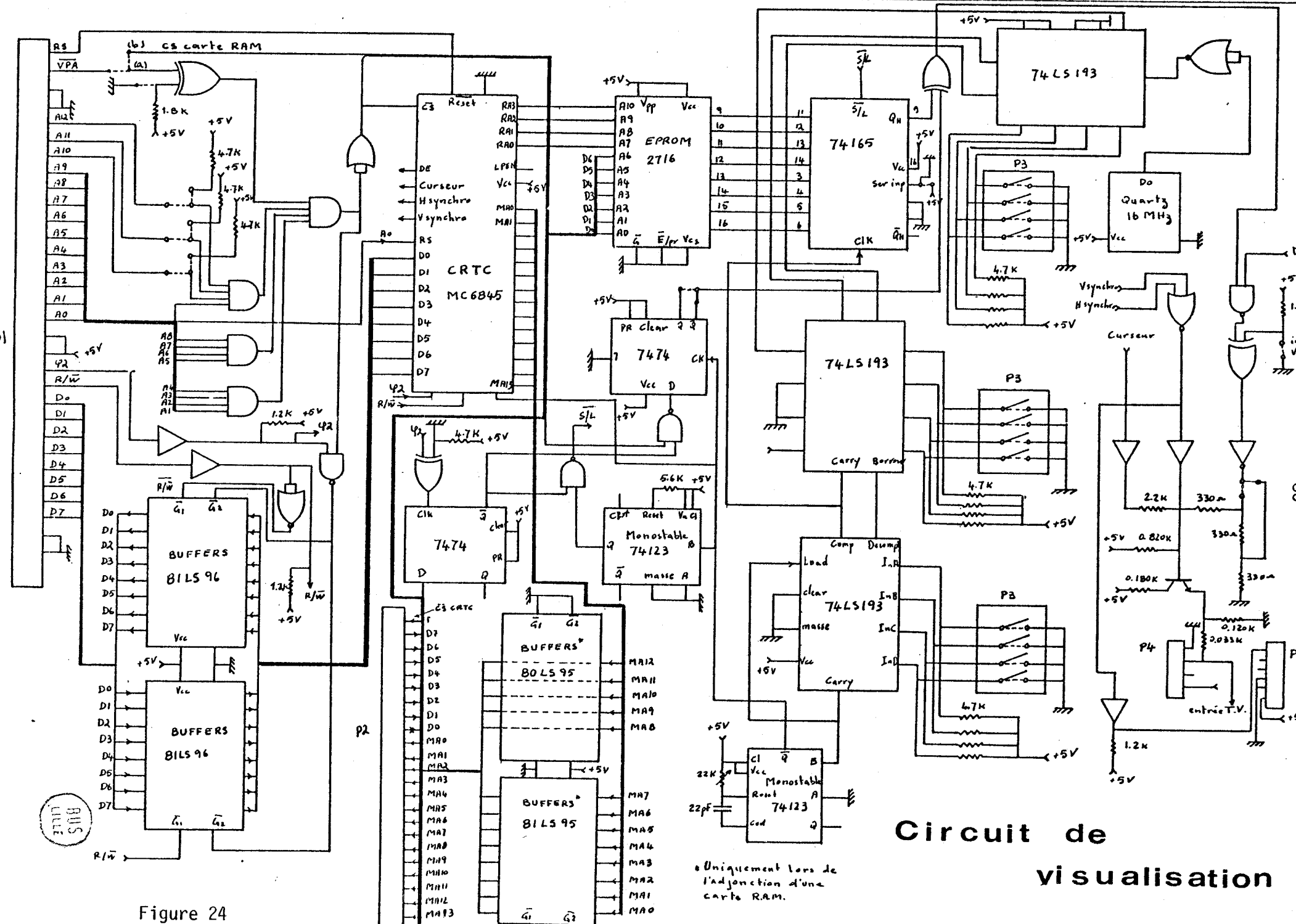
Pile de sauvegarde système	1 K octets pour stockage du texte noir en cours d'édition
Pile de sauvegarde utilisateur	1 K octets pour stockage du texte Braille intégral
Paramètres relatifs à la traduction Braille abrégé	1 K octets pour stockage du texte Braille abrégé
Paramètres relatifs au programme de gestion éditeur	1 K octets de mémoire écran
	3 K octets de mémoire dans laquelle on vide les textes "noir", Braille intégral et Braille abrégé libérant ainsi la place pour l'entrée de nouveaux textes lors de la sortie de ceux-ci sur un support magnétique
RAM VISU et textes édités	

II-2-3- Module de mémoire morte

Ce module contient tous les logiciels d'édition, de transcription en Braille intégral et en Braille abrégé, de formatage, de gestion des périphériques etc.

Cet espace mémoire se décompose comme suit.

Tous les logiciels sont répartis en modules de 4 K octets ce qui permet une remise à jour simple par partitionnement de l'assemblage.


Circuit de visualisation
Figure 24
CRTC MC6845
EPROM 2716
74LS193
74165
7474
74123
BUFFERS 8ILS96
BUFFERS 80LS95
BUFFERS 81LS95
Quartz 16 MHz
Curseur
Vsynchro
Hsynchro
entrée T.V.
* Uniquement lors de l'adjonction d'une carte R.A.M.

Dictionnaire et tables 12 k octets
Programme transcription Braille abrégé
Routines diverses de service
Moniteur gestion des fonctions : édition, lecture, formatage,etc.
Programme de formatage
Programme de gestion visu
Fonction de l'éditeur
Gestion du clavier et codage Braille intégral

II-2-4- Description des circuits de la carte modulateur démodulateur cassette

Nous avons choisi l'enregistrement sur cassette basé sur une modulation type FSR à permutation de fréquence faite sur le signal d'une transmission série asynchrone à 300 bauds.

Ce signal est transmis vers les amplis d'entrée d'un enregistreur à cassette. Il provient de la sortie d'un octet d'un des registres (Braille intégral, Braille abrégé, noir) à travers un interface programmable série asynchrone (ACIA) et un modulateur démodulateur dont le schéma de détail est représenté figure 25 .

II-3- DESCRIPTION DU DUPLICATEUR

Le duplicateur se présente dans un boîtier séparé. Il exploite les informations enregistrées sur un support magnétique (cassette) par l'éditeur. Il peut être relié, soit à un périphérique (embosseuse SAGEM, PED 30, machine à écrire DIABLO) par l'intermédiaire d'une liaison série asynchrone RS 232-C.

Des roues codeuses (figure 26) permettent à l'utilisateur de programmer :

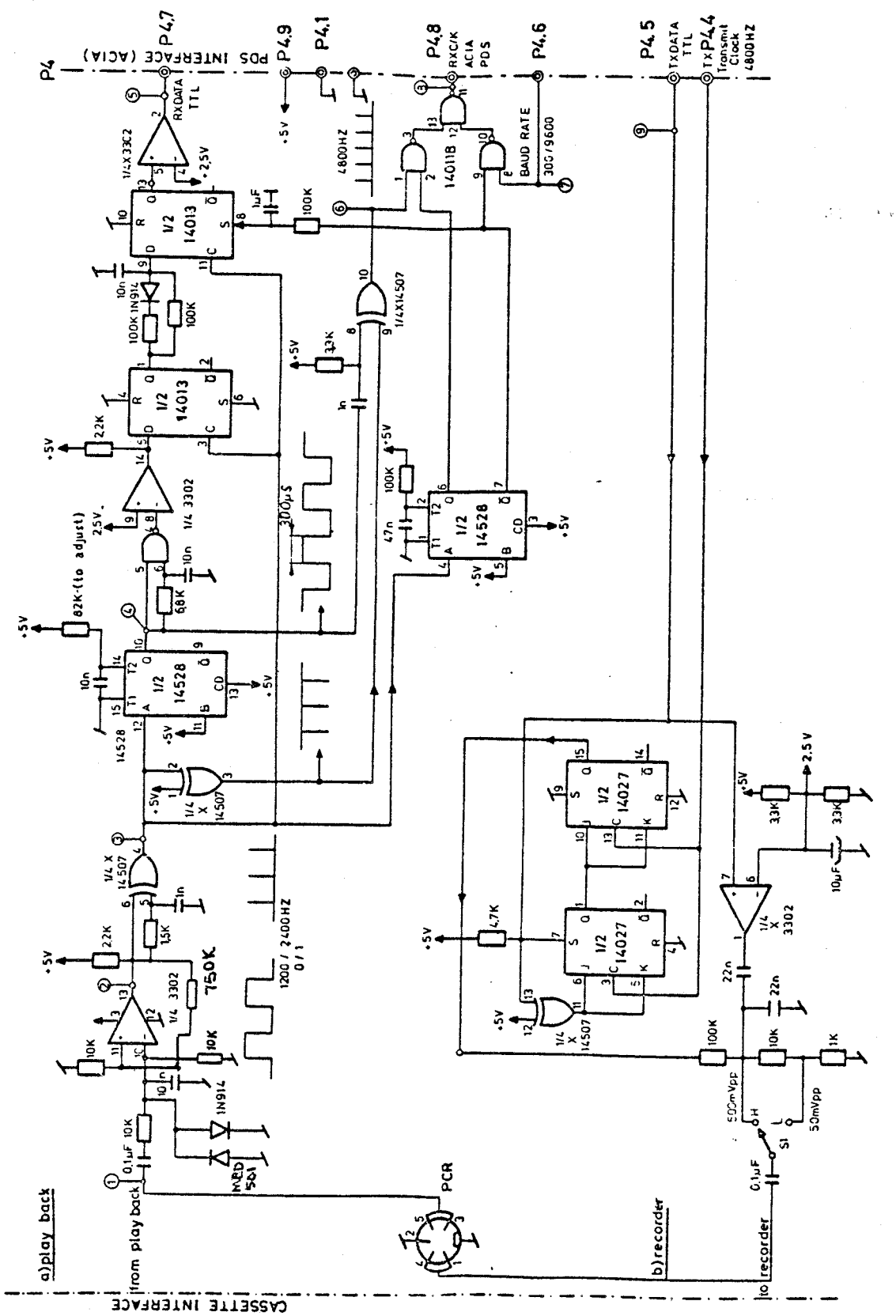


Figure 25 : Modulateur cassette Logibraille

BUS LILLE

1) Le type de texte à dupliquer :
   - texte noir
   - texte Braille intégral
   - texte Braille abrégé

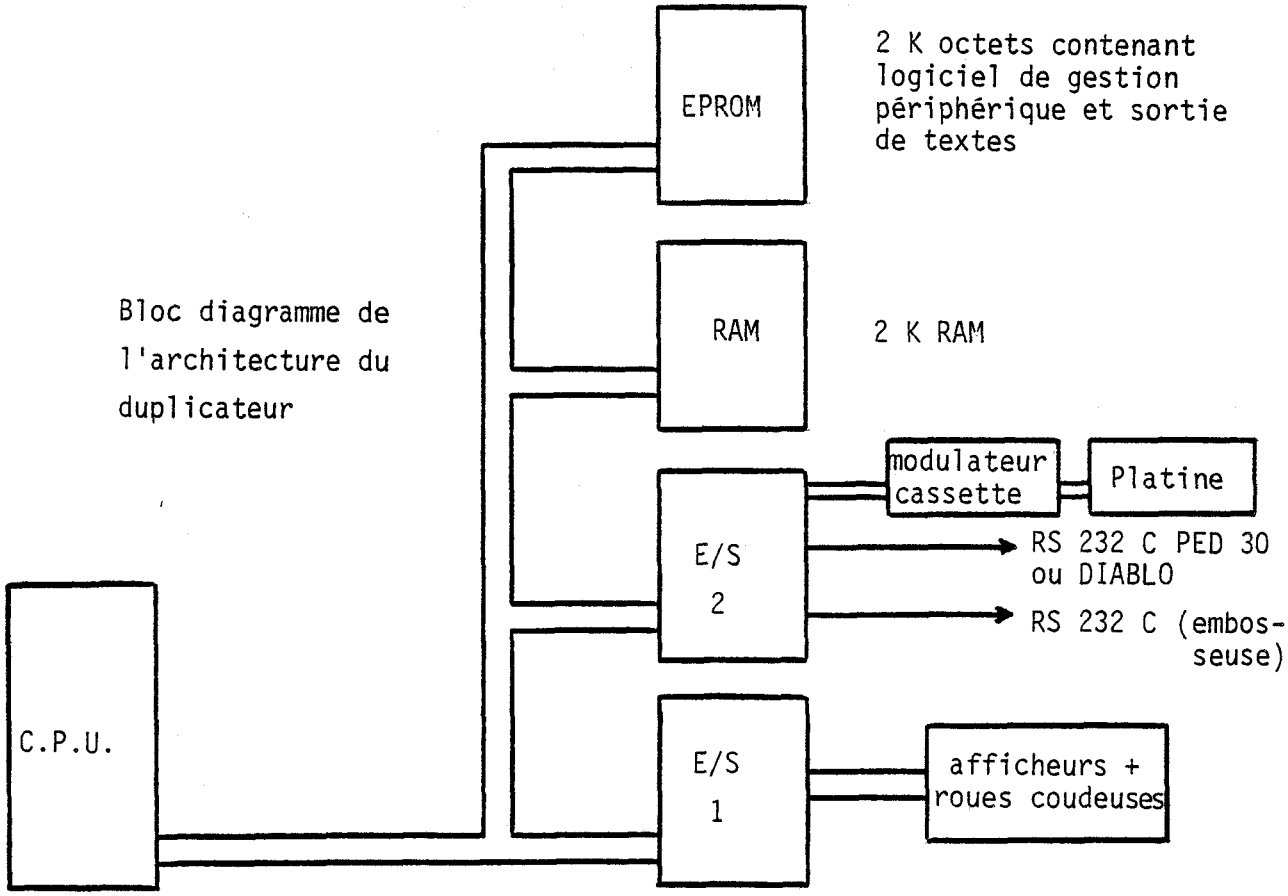
Dans le cas de texte noir , l'utilisateur peut choisir parmi 30 formats d'impression du texte.

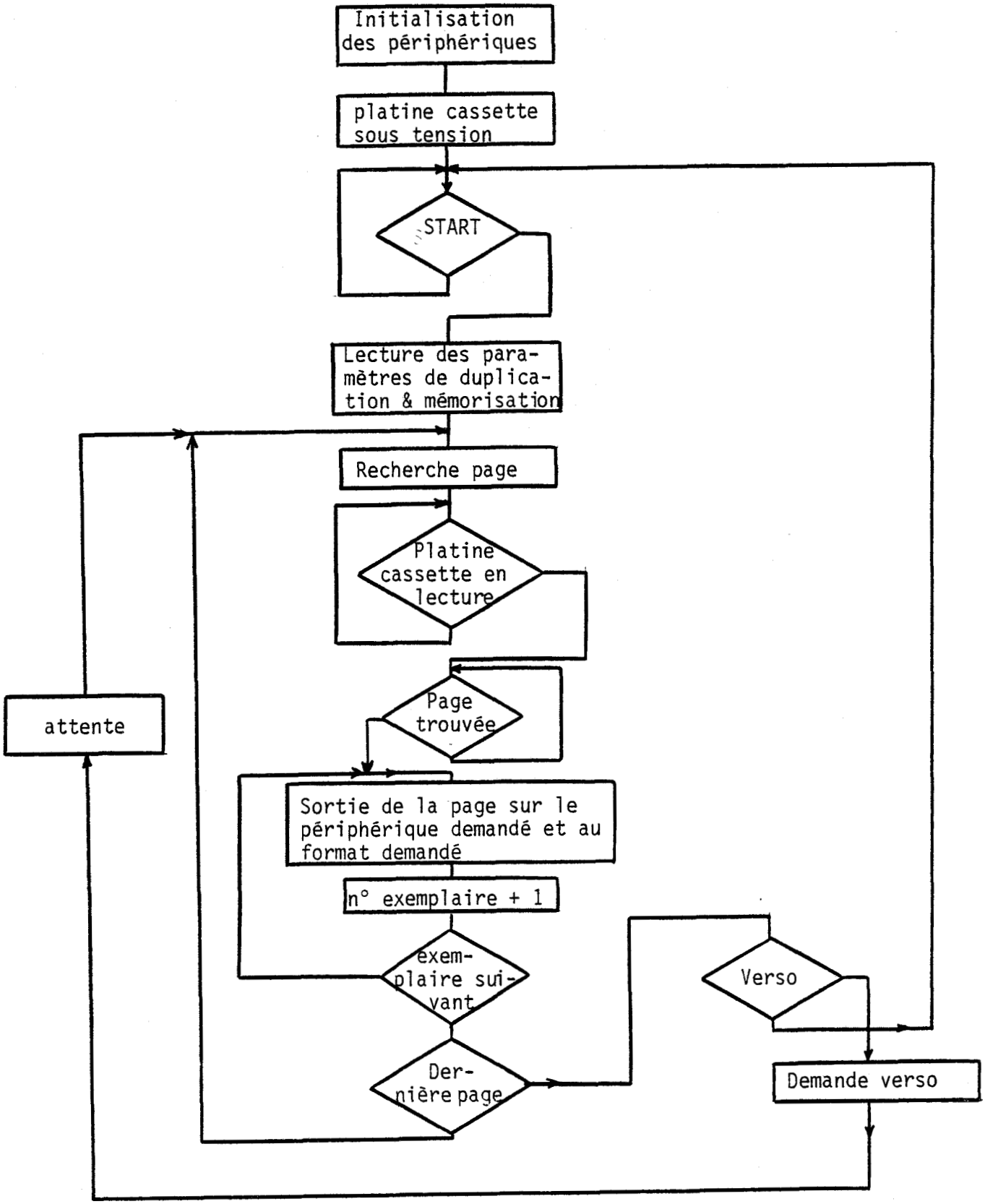
Dans le cas du texte Braille (intégral ou abrégé), l'utilisateur peut choisir soit la sortie des points Braille sur Diablo, soit la sortie du texte sur embosseuse.

2) Le numéro de la première page et celui de la dernière page du texte à sortir, et le nombre d'exemplaires à dupliquer

3) L'impression en recto seul ou recto verso dans le cas de la sortie sur embosseuse.

Bloc diagramme de l'architecture du duplicateur





ORGANIGRAMME DU LOGICIEL DE DUPLICATION

Les programmes de gestion de ces différents périphériques sont contenus dans une mémoire EPROM. La mémoire RAM permet un stockage temporaire des informations lues sur la bande magnétique avant sa sortie par l'intermédiaire d'une des deux sorties (RS 232 C).

PANNEAU D'AFFICHAGE ET PROGRAMMATION

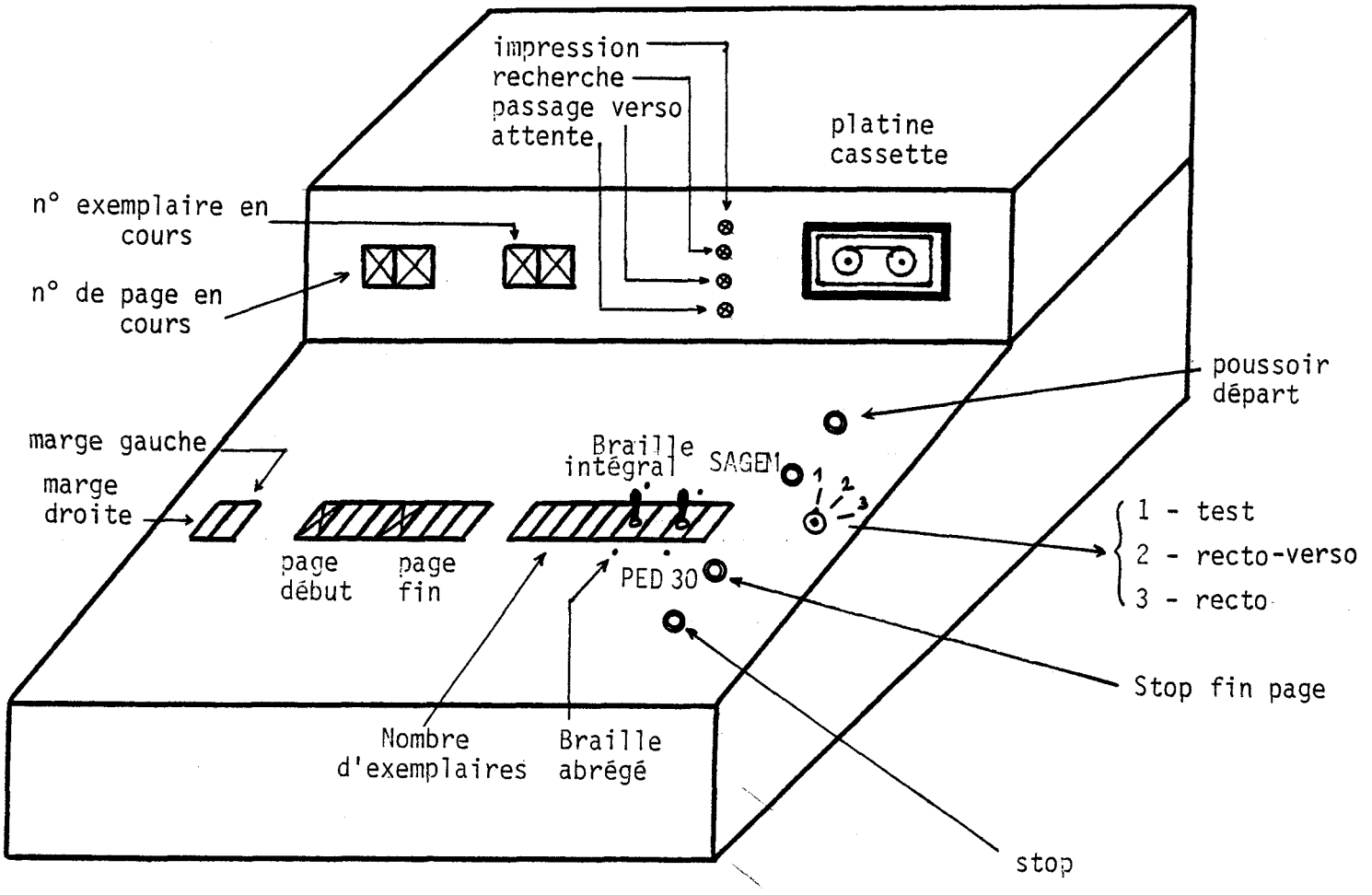


Figure 26

III - PROPOSITION DE DIFFERENTES UTILISATIONS POSSIBLES DU PROCEDE PERMETTANT LA COMMUNICATION VOYANT-AVEUGLE

III-1- UTILISATION INDIVIDUELLE ET TELEMATIQUE DU DISPOSITIF

Sous forme miniaturisée, le dispositif peut être utilisé pour la correspondance voyant-aveugle. Dans ce cas, l'éditeur sera constitué d'un clavier AZERTY, des circuits de gestion d'un écran TV et d'une platine cassette ainsi que les logiciels de transcription Braille intégral et/ou abrégé.

Les textes Braille enregistrés sur une cassette peuvent être envoyés par le voyant à l'aveugle, soit par voie postale, soit par voie téléphonique. L'aveugle disposant d'un générateur de points Braille, peut recevoir et lire à domicile les textes transmis par son correspondant (figure 27).

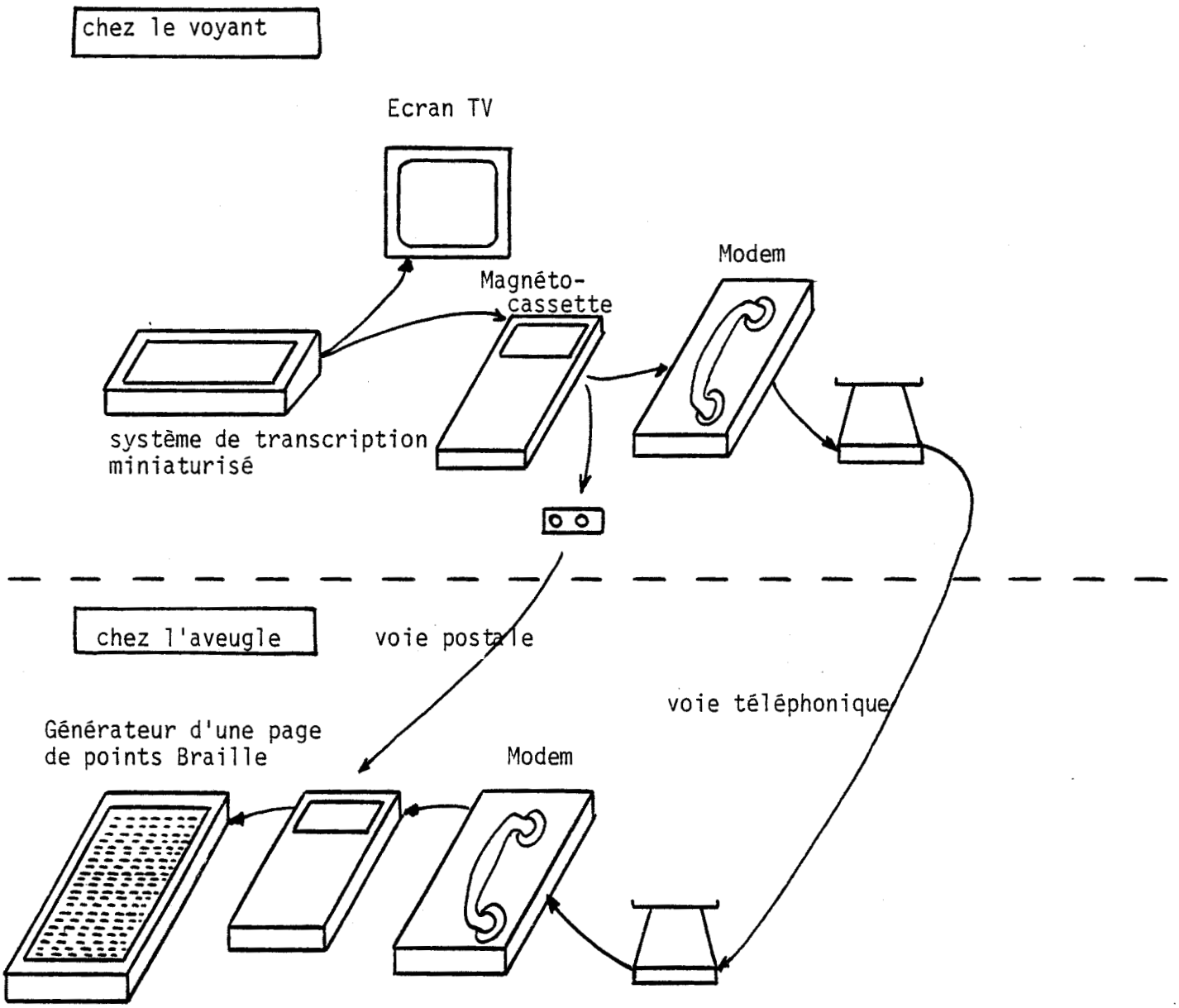


Figure 27

III-2- UTILISATION DU DISPOSITIF COMME MODULE SPECIALISE

Le dispositif peut être utilisé comme module spécialisé relié à un système central disposant d'une mémoire de masse importante et assurant la gestion d'une bibliothèque informatisée Braille. Dans ce cas, les textes saisis et transcrits en Braille sont transmis au système central (figure 28).

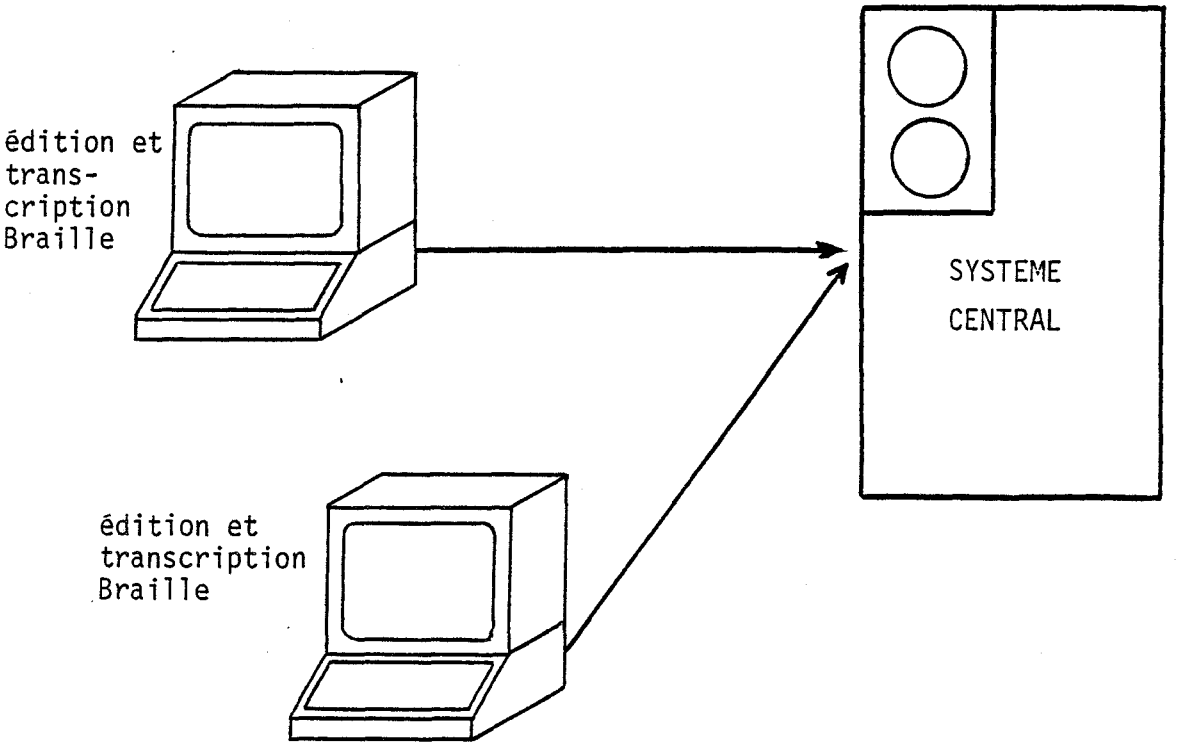


Figure 28

C O N C L U S I O N

Nous avons mené cette étude en pensant à la constitution d'un automate dont l'utilisation faite à partir d'une saisie au clavier d'une machine à écrire ne doit pas nécessiter de formation de son utilisateur.

Ceci nous a conduit à faire apparaître les problèmes de la correspondance entre les syntaxes noir et Braille. Nous avons fait une énumération exhaustive des informations que l'utilisateur doit fournir en supplément de la syntaxe naturelle d'un texte noir dactylographié. Nous avons choisi les solutions qui évitent la mise en oeuvre de logiciels complexes d'analyse du texte sans pour autant conduire à une utilisation fastidieuse de la part de l'utilisateur.

A partir des résultats de cette étude préliminaire et de la connaissance des règles de la transcription Braille, nous avons essayé de bâtir un modèle formel pour constituer un algorithme de transcription systématique.

Nous avons choisi une structure de données qui, associée à ces algorithmes, nous a permis d'aboutir à une optimalisation de la taille du programme et du temps de transcription dont nous avons donné la méthode de calcul du majorant.

L'ensemble constitue un automate qui peut être exploité dans diverses applications. On peut imaginer des application dans le domaine de la production, massive ou non, d'ouvrages en Braille, soit à partir d'une saisie manuelle, soit sous forme d'un module spécialisé de transcription appelé en ligne par un processeur central. Ce module spécialisé peut pratiquer cette transcription, soit en vue de la production d'ouvrages, soit en vue du stockage de données comprimées.

Dans la dernière partie de notre thèse, nous avons essayé de décrire quelques configurations possibles parmi lesquelles existent celles utilisées actuellement dans divers établissements. Les performances du logiciel et la technologie microinformatique utilisée font que cet automate est facilement intégrable dans des architecture miniaturisées. Il paraît alors bien adapté à la conception d'instruments de communication individuelle voyant-aveugle empruntant soit le réseau de télécommunications public, soit le réseau privé.

Enfin, l'expérience acquise au cours de cette étude a servi de base à celle de la constitution d'un automate de transcription inverse.

BIBLIOGRAPHIE

[1] B. HAMPSHIRE

"Establishing Braille Production Facilities in developing Countries"
Swedish Federation of the Visually Handicapped

[2] J. LOPEZ KRAHE

"Etude des méthodes de reconnaissance automatique de caractères imprimés. Application à un système de lecture pour aveugles"
Doctorat d'Université , Paris III

[3] G. BOUVIER

"Système d'acquisition et de reconnaissance de caractères alphanumériques"
Doctorat 3è cycle, Grenoble

[4] A. CHEHIKIAN

"Conception et réalisation d'un automate de reconnaissance de caractères alphanumériques"
Doctorat ès sciences physiques, Grenoble

[5] BLANCHIN, HENRI, LE GUEVEL, LECOGNE et autres

"Extension de l'abrégé orthographique français"
Edition nouvelle, Edition A.V.H. pour les copistes, Paris, 1964

[6] M. TRUQUET

"Transcription en Braille intégral ou abrégé"
Thèse d'état, 30 novembre 1979

[7] J. COURTIN

"Algorithmes pour le traitement interactif des langues naturelles"
Doctorat ès Sciences Mathématiques, Grenoble , 1977

[8] GRANDJEAN

"Conception et réalisation d'un dictionnaire pour analyseur interactif de langues naturelles"
Mémoire CNAM, Université de Grenoble, 1975

[9] J.C. MENEAU

"Bases pour l'élaboration d'une bibliothèque informatisée pour les déficients visuels par traduction automatique de fichiers déjà constitués dans l'imprimerie classique"
Doctorat 3è cycle, Université de Droit et de la Santé de Lille, 1980

.../...

.../...

[10] B. MATHIEU
"Utilisation d'un transcripteur général d'états finis pour la saisie de textes et leur traduction en Braille abrégé"
Congrès AFCET, Novembre 1978

[11] J.P. DUBUS, F. WATTRELOT
"Interpréteur-éditeur Braille automatique autonome avec clavier de machine à écrire"
Le Nouvel Automatisme, Juin-Juillet 1979

[12] C. PAIR, M.C. GAUDEL
"Les structures de données et leur représentation en mémoire"
2è édition INRIA

[13] P. LOOSFELT
"Sur un ensemble technico-pédagogique de traitement numérique et de communication sur écran de télévision"
Thèse de Docteur-Ingénieur, Lille, 15 juin 1977

[14] G.L. GOODRICH, R.R. BENETT
"Computer acces for the blind"
Revue Tekniques, Tektronix, juin 1979

[15] P.W.F. COLEMAN
"Tactile displays:their current state and a new approach"
Conference internationale on computerized Braille production
Today and To morrow, Londres, juin 1979

[16] M. TRUQUET
"Transcription par ordinateur en Braille intégral et abrégé"
Congrès AFCET, Novembre 1978

[17] ELINFA
Notice sur l'enregistreur Braille portatif "digicassette"

[18] J.P. DUBUS, A. MANDAR
"Démarche adoptée pour établir la faisabilité d'un traducteur autonome temps réel Braille abrégé à l'aide d'un microprocesseur .
Description détaillée d'une partie des solutions adoptées"
I.T.B.M., vol. 1, n° 2, 1980

[19] J.P. DUBUS, A. MANDAR
"Méthode d'élaboration des algorithmes de traduction en Braille abrégé par dactylographie d'un texte.
Optimisation de la taille mémoire et du temps de traduction pour implantation sur microprocesseur"
I.T.B.M., vol. 1, n° 4, 1980

.../...

[20] LOGIBRAILLE
Notice technique

[21] J.P. DUBUS, F. WATTRELOT, A. MANDAR
"Description of a system for the editing and duplication of texts either in integral or abbreviated Braille for the blind, or in large size characters for the partially sighted"
Sensory World, n° 41, Décembre 1980

[22] J.P. DUBUS, A. MANDAR
"Dispositif d'édition autonome et de reproduction de textes en Braille intégral et abrégé pour aveugles et de textes noirs en grands caractères pour amblyopes"
Colloque international "Informatique et Braille" , Toulouse, Septembre 1981

[23] AMERICAN FOUNDATION FOR OVERSEAS BLIND
Extension de l'abrégé orthographique français
2ème édition, 1955

[24] WORLD HEALTH ORGANIZATION
Economic aspects of eye Health care
Report on a meeting , Copenhagen, 31 janvier - 1er février 1980

[25] WORLD HEALTH ORGANIZATION
The use of redisual vision by visually disabled persons.
Report on a W.H.O. Meeting, Brussels, 28-30 janvier 1981

[26] P. DEBRAINE
"Système de production Braille I.N.J.A.-I.N.S.T.N. pour la traduction du Braille abrégé et intégral de livres de classes littéraires et scientifiques"
Colloque international "Informatique et Braille", Toulouse, 23-26 septembre 198

[27] M.C. GENNERO, A. POLI
"Condensed "Braille" transmission and error correcting codes"
Colloque international "Informatique et Braille", 23-26 septembre 1981

[28] G. GOUARDERES, B. CAUSSE
"Sur deux problèmes de représentation dans les bases de données textuelles"
Colloque international "Informatique et Braille" , 23-26 septembre 1981

[29] J. CHRISTOPHERSEN
The Nord Braille system
Colloque international "Informatique et Braille", 23-26 septembre 1981

.../...

[30] KARI LARSEN

Handitext. An Interactive reading system for the blind
Colloque international "Informatique et Braille",23-26 septembre 1981

[31] J. FRONTIN, D. LEVY, J. MEEKEL, M. TRUQUET

"Présentation d'un logiciel de transcription de la musique en Braille"
Colloque International "Informatique et Braille", 23-26 Septembre 1981

[32] A. RECUERO

"Production de Braille par ordinateur en Espagne"
Colloque International "Informatique et Braille", 23-26 Septembre 1981

[33] J. MEEKEL, J. FRONTIN, D. LEVY, M. TRUQUET

"Un système de production de textes mathématiques en Braille"
Colloque International "Informatique et Braille", 23-26 Septembre 1981

[34] J.P. DUBUS, F. WATTRELOT

"Description des résultats obtenus sur l'étude et la réalisation d'un système de visualisation de texte à format variable de caractères à l'usage des amblyopes"
Lettre I.T.B.M. à l'éditeur, Vol. 1, n° 3 , 1980

[35] J. MEEKEL, J. FRONTIN, D. LEVY, M. TRUQUET

"Transcription du Braille abrégé français"
Colloque international "Informatique et Braille", 23-26 Septembre 1981

[36] J. KAPLIN

"Methods of inputting text for Braille translation at the RNIB"
Colloque international "Informatique et Braille", 23-26 Septembre 1981

[37] S.C. MACKENZIE

"L'écriture en Braille dans le monde"
U.N.E.S.C.O.

[38] P. GUIRAUD

"La Semautique"
P.U.F.

[39] J. PERROT

"La Linguistique"
P.U.F.

[40] E. DELAVENAY

"La machine à traduire"
P.U.F.

[41] R. ESCARPIT

"L'écrit et la communication"
P.U.F.

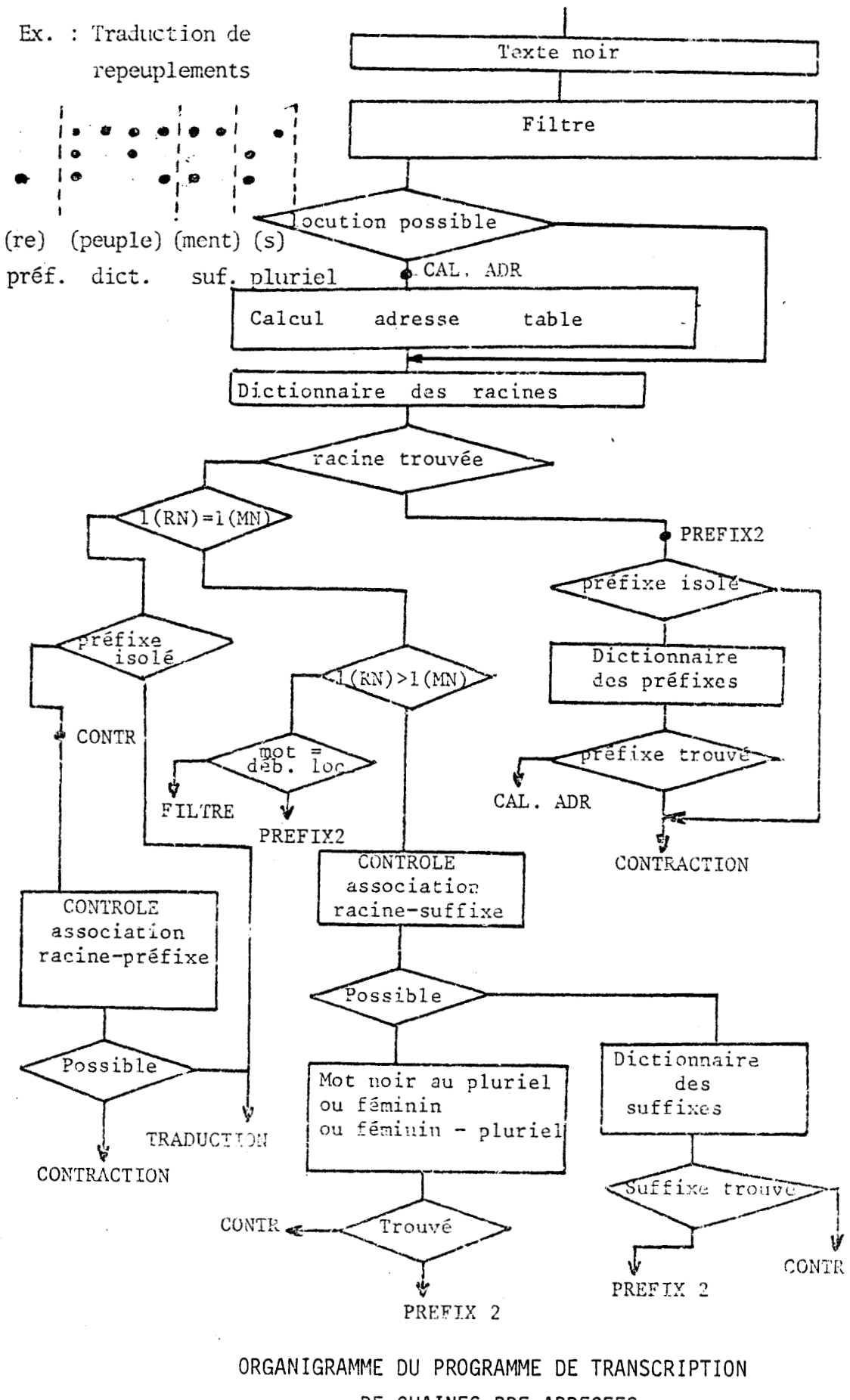
[42] A. DONEDDU

"Structures fondamentales"
Tome I , VUIBERT

A N N E X E

Ex. : Traduction de repeuplements

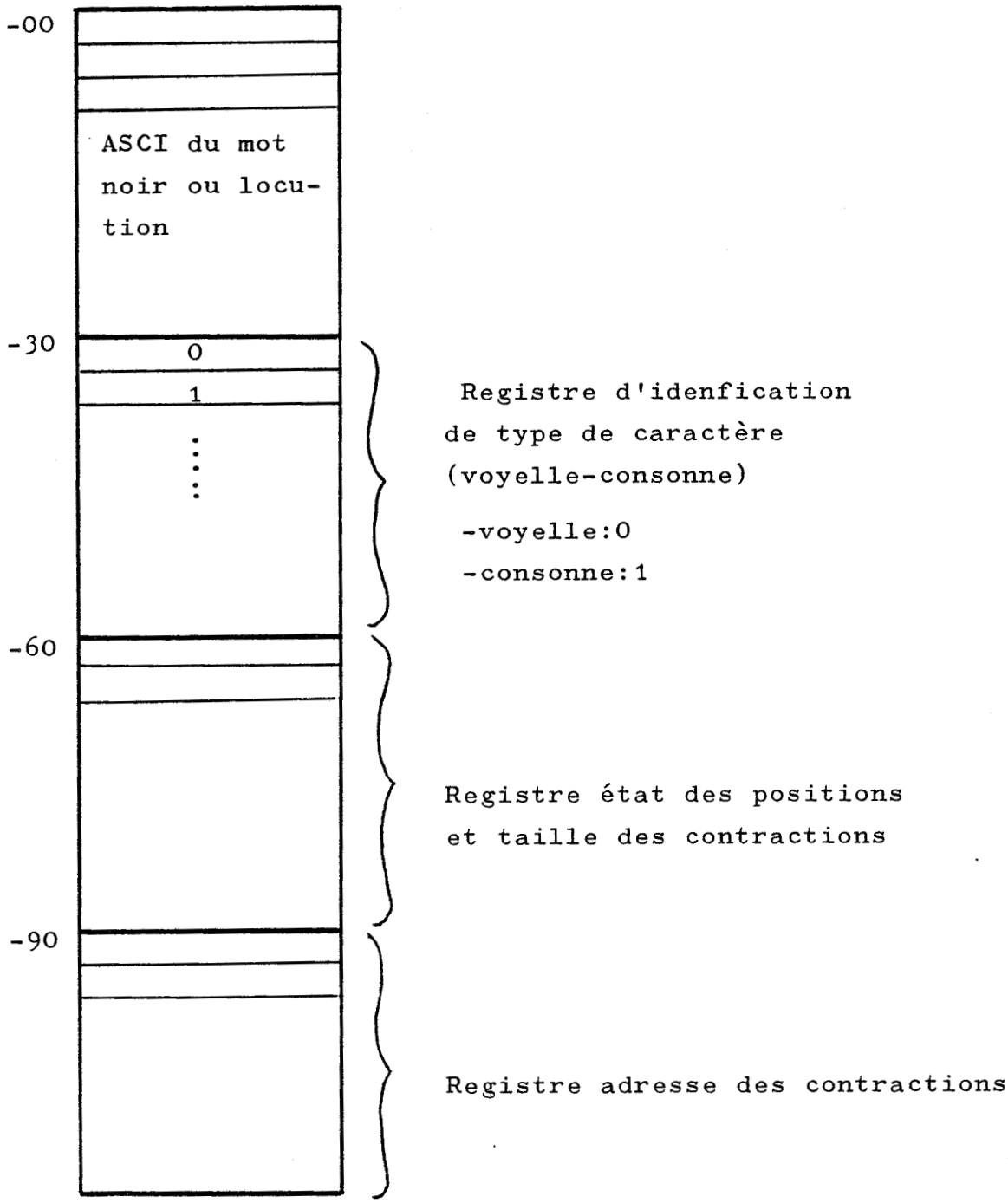
(re) (peuple) (ment) (s)
préf. dict. suf. pluriel



ORGANIGRAMME DU PROGRAMME DE TRANSCRIPTION DE CHAINES PRE-ABREGEES

REGISTRE MOT

Dans le programme de traduction ce registre est appelé: MCOMPL



TRAITEMENT DES SUFFIXES DES MOTS DU DICTIONNAIRE

-Seules les racines des mots s'associant avec les terminaisons suivantes sont munies d'un octet en fin de mot contenant la valeur 00

-AIRE

-EUX

-EUSE

-AUX

-EL,ELLE

-AL

IVE

-IF

MENT

-TE

-Lorsque la racine a été identifiée dans le dictionnaire associée ou non à un prefixe et que le mot entré au clavier est plus long que la racine ,on vérifie l'octet de fin de mot , et si cet octet est nul,on cherche à identifier la fin du mot noir dans le dictionnaire des suffixes.La traduction du mot noir en BRAILLE abrégé sera donc la concaténation du Braille-racine et Braille-suffixe

TERMINAISON EN AIRE

Le mot derivé peut être obtenu de la façon suivante:

-(racine)+aire

-(racine)+(e ou é)+(aire)

TERMINAISON EN EUX

(racine)+(eux)

(racine)-(e ou é)+(eux)

(racine)-(e)+(ieux)

TERMINAISON EN EUSE

(racine)+(euse)

(racine)-(e ou é)+(euse)

(racine)-(e)+(ieuse)

TERMINAISON EN AUX

(racine)+(aux)

(racine)-(l)+(ux)

(racine)-(e)+(aux)

TERMINAISON EN EL,ELLE

(racine)+(l ou lle)

(racine)+(el ou elle)

(racine)-(e)+(iel ou ielle)

TERMINAISON EN AL

(racine)+(al)

(racine)-(e)+(al)

TERMINAISON EN IVE,IF

(racine)+(ve)

(racine)+(f)

TERMINAISON EN MENT

(racine)+(ment)

(racine)+(ement)

(racine)+(ément)

(racine)-(e)+(ément)

Le suffixe "ment" peut être utilisé aprés les suffixes:

-euse

-elle

-aire

-ive

-al

TERMINAISON EN TE

(racine)+(ité)

(racine)-(e)+(ité)

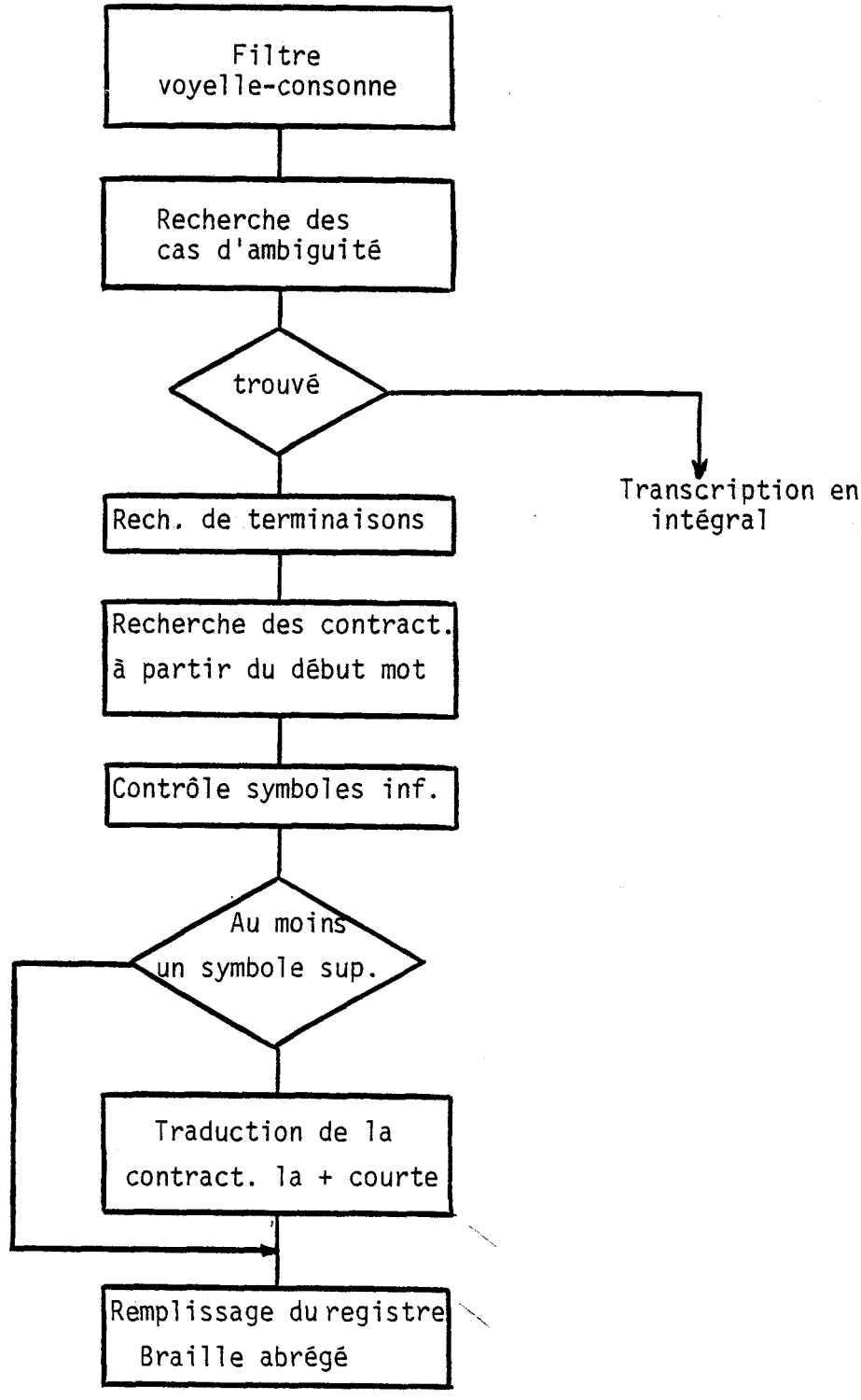
(racine)+(té)

Les lettres qui peuvent être retranchées aux racines avant d'ajouter les symboles fondamentaux cités ci-dessus sont

-	+
e,é	a
é	e
e	i
l	u
e	é

DICTIONNAIRE DES SUFFIXES

Terminaison	Traduction Braille	Terminaison	Traduction Braille
AIREMENT	rm	ITÉ	t
AIRE	r	IVEMENT	vm
ALEMENT	lm	LLEMENT	lm
ALES	lû	LLE	ll
ALE	le	L	l
ALITE	lt	MENT	m
AL	l	NAIRE	r
AUX	x	NELLEMENT	lm
ELLEMENT	lm	NELLE	ll
ELLE	ll	NEL	l
EL	l	NEMENT	m
EMENT	m	S	s
ES	û	TALEMENT	lm
EUSEMENT	sm	TALE	le
EUSES	sû	TALITE	lt
EUSE	se	TAL	l
EUX	x	TAUX	x
E	e	TÉ	t
F	f	USEMENT	sm
IELLEMENT	lm	USES	sû
IELLE	ll	USE	se
IEL	l	UX	x
IEUSEMENT	sm	VEMENT	vm
IEUSE	se	VE	v
IEUX	x	VITÉ	vt
ITAIRE	tr	ÉMENT	m



ORGANIGRAMME DE TRANSCRIPTION PAR CONTRACTIONS

REGLES DE TRANSCRIPTION PAR CONTRACTIONS

n° règle	signification
01	toujours
02	terminaison
03	devant voyelle
04	devant consonne
05	entre voyelles
06	début uniquement
07	début et devant consonne
08	début et terminaison
09	devant consonne et terminaison
11	règle relative aux mots se terminant par "ient"
12	terminaison ou devant consonne, jamais au début, ni devant <<
13	début devant voyelle
14	devant consonne et terminaison sauf devant point
15	devant consonne et terminaison sauf devant point d'interrogation
16	devant la chaîne "quement" (chaîne logi)
17	début et devant consonne sauf après le séparateur "-"
18	toujours sauf au début après le séparateur <(>
19	toujours au début, dans le corps du mot devant consonne en terminaison sauf devant le séparateur <*>
20	devant consonne jamais en terminaison
21	dans le mot devant consonne en terminaison sauf devant le séparateur <*>

TABLE DES CONTRACTIONS DE DEUX LETTRES

Contraction	Type symbole	n° règle	Braille
ai	00	01	2A
an	01	20	2C
ar	00	09	22
au	00	01	4B
bl	00	03	25
br	01	03	3B
ch	00	01	5B
cl	00	03	33
cr	01	03	3A
dr	00	03	34
em	00	04	26
en	01	15	3F
er	01	12	3C
es	00	08	35
eu	01	01	5F
ex	00	20	58
ez	00	02	5A
fl	00	03	32
fr	00	03	31
gl	00	03	26
gn	01	18	3D
gr	00	03	37
im	00	07	39
in	01	19	29
ll	00	05	24
oi	00	01	5D
om	00	09	57
on	00	09	5E
or	01	09	2B
ou	00	01	38
pl	00	03	36
pr	01	03	21
qu	00	01	51
re	01	07	27
ss	00	05	5C
tr	01	03	3E
tt	00	05	57
ui	01	20	3B

TABLE DES CONTRACTIONS DE TROIS LETTRES

Contraction	type symbole	n° règle	Braille
êfl	00	01	32 46 4C
ain	00	09	41 29
ait	00	02	33
ant	00	02	34
com	01	17	2D
con	01	20	3A
dis	01	07	2E
drô	00	01	44 52 34
ent	00	02	32
ess	00	06	35 53
eur	00	09	23
flê	00	01	46 4C 32
frâ	00	01	46 52 31
ien	01	14	2E
ieu	01	01	28
ion	00	09	30
oin	00	19	4F 29
our	00	09	40
pro	01	04	21
que	00	02	51
qui	00	02	51 49
rem	00	07	52 26
ren	00	07	52 3F
ssè	00	01	53 53 5C
uim	00	20	55 39
uin	00	21	55 29
zez	00	02	5A 45 5A

TABLE DES CONTRACTIONS DE QUATRE LETTRES

Contraction	type symbole	n° règle	Braille
able	00	02	25
brui	00	01	42 52 3B
elle	00	02	24
ient	00	11	2E 54
ieur	00	01	28 52
ines	00	07	29 35
inin	00	19	49 4E 29
logi	00	16	4C 47
ouin	00	19	38 29
prom	00	04	21 57
ques	00	02	51 53
quin	00	21	51 29
type	00	02	54 59

TABLE DES CONTRACTIONS DE CINQ LETTRES

Contraction	type symbole	n° règle	Braille
ables	00	02	25 53
ation	00	02	31
clait	00	02	4C 33
concr	00	03	43 5E 3A
drant	00	02	44 52 34
elles	00	02	24 53
flent	00	02	46 4C 32
incom	01	07	29 2D
indis	01	07	29 2E
ition	00	02	39
logie	00	02	4C 47
propr	00	03	21 4F 21
recom	01	07	27 2D
redis	01	07	27 2E
tenir	00	02	54 4E
trans	01	07	3E
venir	00	02	56 4E

TABLE DES CONTRACTIONS DE SIX LETTRES

Contraction	type symbole	n° règle	Braille
aition	00	02	2A 54 30
bilité	00	02	42 4C 54
blable	00	02	42 4C 25
logies	00	02	4C 47 53
mettre	00	02	4D 3E
tement	00	02	54 4D
uition	00	02	3B 54 30
vement	00	02	56 4D

TABLE DES CONTRACTIONS DE SEPT LETTRES

Contraction	type symbole	n° règle	Braille
blables	00	02	42 4C 25 53
fration	00	02	46 52 31
inssent	00	02	29 5C 32
inssiez	00	02	29 5C 49 5A
intrans	01	07	29 3E
logique	00	02	4C 47 51
mission	00	02	4D 30
quement	00	02	51 4D
retrans	01	07	27 3E

TABLE DES CONTRACTIONS DE HUIT LETTRES

Contraction	type symbole	n° règle	Braille
ablement	00	02	25 4D
ellement	00	02	24 4D
inssions	00	02	29 5C 30 53
logiques	00	02	4C 47 51 53

TABLE PERMETTANT L'ANALYSE DU RESTE DU MOT LORSQU'UNE TERMINAISON "IENT" A ETE ISOLEE.

IENT 3	ENT 4	IENT 5	IENT 6	IENT 7	IENT 8
adv	abst	.bienv	appart	circonv	inconven
dev	cont	consc	entret	coeffic	subconsc
dét	emol	defic	ingred	contrev	
esc	grad	effic	interv	disconv	
obt	parv	.excip	omnisc	inconsc	
obv	prov	.impat		ressouv	
pat	prév	.maint			
ret	quot	presc			
rev	sout	.redev			
	souv	.récip			
	subv				
	surv				

PROBLEME DES PRIORITES

oin	o-(in) devant consonne
oin	(oi)-n devant voyelle
ain	a-(in) devant consonne
ain	(ai)-n devant voyelle
uin	u-(in) devant consonne
uim	u-(im) devant consonne
aim	a-im devant consonne ai-m devant voyelle
rem	r-em début devant consonne
ren	r-en début devant consonne
rem	re-m début devant voyelle
ren	re-n début devant voyelle
ess	es-s début

CARACTERES SE TRADUISANT PAR SYMBOLES INFERIEURS

Contractions	Caractère	symbole Braille	Code
eu	←		5F
cr con	:		3A
br ui	;		3B
er	<		3C
gn	=		3D
tr trans	>		3E
en	?		3F
pr pro	!		21
re	,		27
ieu gn	(		28
in	)		29
or	+		2B
an	,		2C
com	-		2D
dis,ien	.		2E

DICTIONNAIRE DES PREFIXES

Préfixe	n° des groupes dans lesquels figure le préfixe	Traduction Braill
ac	50	ac
anti	07	,ti
auto	22	kto
a	47 57	a
com	06 17	-
contre	24	:>e
contr	47	:>
con	08 17 46 47 57	:
co	45	co
de	47	de
dis	18	.
dés	14 52	dés
dé	06 07 08 09 10 22 32 41 43 44 46 50	dé
em	41 55	œ
entre	46	?>e
en	23 09	?
extra	53	x>a
ex	12	x
il	54	il
im	22 45	æ
inter	21 12 24 47	*t<
in	12 13 20 57	*
ir	31 32	ir
mal	52	mal
mé	42 15	mé
par	09 47	p"
per	06	p<

.../...

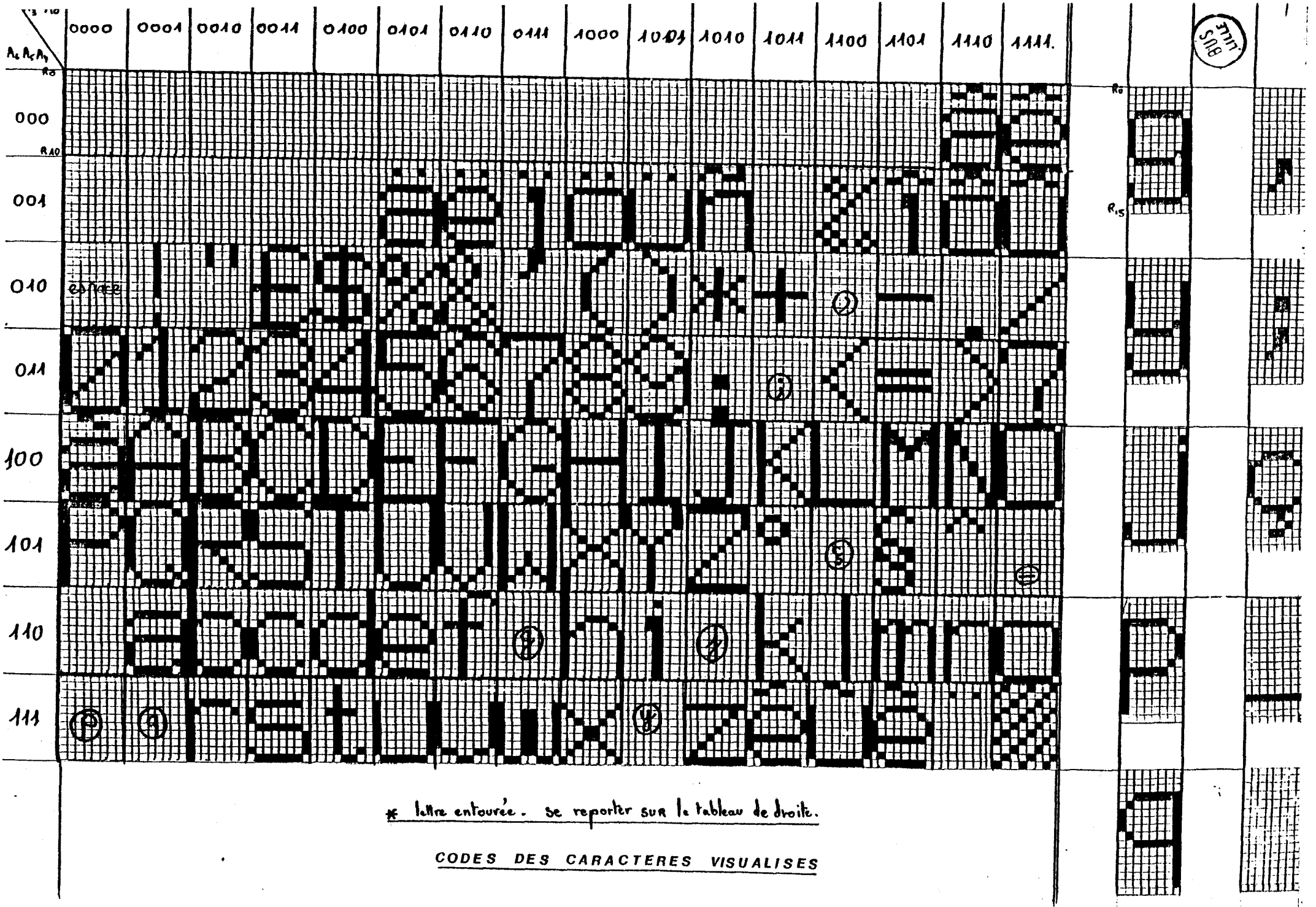
pré	19	!é
ran	56	r,
rem	41	r æ
ren	51	r?
re	10 15 20 40 41 42 43 44 45 46 47	'
ré	40	ré
sou	46 47	sü
super	45	sup<
supra	21	su!a
sur	07 16 20 31 44 47	sur
trans	08 43	.>
télé	53	télé
équi	22	éqi
é	40	é

GROUPEMENT DES PREFIXES

06	07	08	09	10	12
com dé per	dé sur anti	dé trans con	dé en par	dé em re	ex inter in
13	14	15	16	17	18
in	dés	mé re	sur	com con	dis
19	20	21	22	23	24
pré	in sur re	inter supra	auto im dé équi	en	contre inter
31	32	40	41	42	43
ir sur	dé ir	re é ré	re dé rem em	mé re	dé re trans
44	45	46	47	50	51
dé re sur	co im re super	con dé entre re sou	contr sou con sur de a par inter re	ac dé	ren
52	53	54	55	56	57
mal dés	extra télé	il	em	ran	con a in

b_4	b_3	b_2	b_1		b_7 0	0	0	0	1	1	1	1
					b_6 0	0	1	1	0	0	1	1
					b_5 0	1	0	1	0	1	0	1
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	TC_7 (DLE)	SP	0 3456	b @ c 1234 6	P 1234		
0	0	0	1	1	TC_1 (SOH)	DC_1	! 23 5	â 1 1 6	A 1	Q 12345		
0	0	1	0	2	TC_2 (STX)	DC_2	" 4	ê 2 12 6	B 12	R 123 5		
0	0	1	1	3	TC_3 (ETX)	DC_3	# 4 6	î 3 1 4 6	C 1 4	S 234		
0	1	0	0	4	TC_4 (EOT)	DC_4	ou $ 456	ô 4 1 456	D 1 45	T 2345		
0	1	0	1	5	TC_5 (ENQ)	TC_8 (NAK)	% 45	û 5 1 56	E 1 5	U 1 3 6		
0	1	1	0	6	TC_6 (ACK)	TC_9 (SYN)	et & 345	ë 6 12 4 6	F 12 4	V 123 6		
0	1	1	1	7	BEL	TC_{10} (ETB)	' 3	ï 7 12 456	G 12 45	W 2 456		
1	0	0	0	8	FE_0 (BS)	CAN	(6	ü 8 12 56	H 12 5	X 1 34 6		
1	0	0	1	9	FE_1 (HT)	EM	) 3 5	œ 9 2 4 6	I 2 4	Y 1 3456		
1	0	1	0	10	FE_2 (LF)	SUB	* 34	: 2 5	J 2 45	Z 1 3 56		
1	0	1	1	11	FE_3 (VT)	ESC	+ 56	; 23	K 1 3	à [123 56		
1	1	0	0	12	FE_4 (FF)	IS_4 (FS)	2 ,	< 23 6	L 123	è \ 234 6		
1	1	0	1	13	FE_5 (CR)	IS_3 (GS)	- 3 6	= 23 56	M 1 34	ù] 23456		
1	1	1	0	14	SO	IS_2 (RS)	. 2 56	> 3 56	N 1 345	¬ non ^ 34 6		
1	1	1	1	15	SI	IS_1 (US)	é / 123456	? 2 6	O 1 3 5	← 5		DEL

b_4	b_3	b_2	b_1		b_7 0	0	0	0	1	1	1	1
					b_6 0	0	1	1	0	0	1	1
					b_5 0	1	0	1	0	1	0	1
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	TC_7 (DLE)	SP					
0	0	0	1	1	TC_1 (SOH)	DC_1						
0	0	1	0	2	TC_2 (STX)	DC_2						
0	0	1	1	3	TC_3 (ETX)	DC_3						
0	1	0	0	4	TC_4 (EOT)	DC_4						
0	1	0	1	5	TC_5 (ENQ)	TC_8 (NAK)						
0	1	1	0	6	TC_6 (ACK)	TC_9 (SYN)						
0	1	1	1	7	BEL	TC_{10} (ETB)						
1	0	0	0	8	FE_0 (BS)	CAN						
1	0	0	1	9	FE_1 (HT)	EM						
1	0	1	0	10	FE_2 (LF)	SUB						
1	0	1	1	11	FE_3 (VT)	ESC						
1	1	0	0	12	FE_4 (FF)	IS_4 (FS)						
1	1	0	1	13	FE_5 (CR)	IS_3 (GS)						
1	1	1	0	14	SO	IS_2 (RS)						
1	1	1	1	15	SI	IS_1 (US)						DEL



* lettre entourée. se reporter sur le tableau de droite.

CODES DES CARACTERES VISUALISES

RESUME

L'étude porte sur les concepts et les logiciels mis au point pour rendre possible la transduction automatique temps réel par microprocesseur d'un texte noir saisi manuellement en Braille abrégé sans aucun apprentissage préalable de l'utilisateur.

Dans une première partie, l'auteur définit les besoins pour des établissements scolaires ou associations ou pour les individus en matière de communication écrite voyant-aveugle et établit le cahier des charges d'un système autonome de transduction automatique Braille intégral et Braille abrégé.

La deuxième partie est relative à la description d'un modèle formel des règles de transcription d'un texte noir en Braille abrégé. Ce modèle, associé à une structur de données et à un algorithme adapté, permet d'aboutir à un logiciel de taille et de temps d'exécution optimums. Les solutions adoptées et les fonctions de coûts sont calculées pour justifier les choix soutenus.

Dans une troisième partie, l'auteur décrit l'architecture du système et un certain nombre d'applications de l'automate mis au point en matière de communication écrite individuelle ou collective entre voyant et aveugle.

MOTS CLES

Système autonome - Braille intégral - Braille abrégé - Transcription automatique - Communication homme-machine - Analyse morphologique.