UNIVERSITE D'ALGER

ECOLE NATIONALE
POLYTECHNIQUE

Département Economie

SOUS SYSTEME
CONVERSATIONNEL 1130

Sujet étudié par
Ahmed SELLAH
Mikaïl Kamil TIAR

Proposé par M. ADIBA

Juin 1973

UNIVERSITE D'ALGER

ECOLE NATIONALE
POLYTECHNIQUE

Département Economie

SOUS SYSTEME
CONVERSATIONNEL 1130

Sujet étudié par
Ahmed SELLAH
Mikaïl Kamil TIAR

Proposé par M. ADIBA
Juin 1973

A la mémoire de mon père M. H. TIAR

A nos parents.

A Monsieur ADIBA, notre Professeur et Maître,
auquel nous devons notre savoir,

notre reconnaissance.

A Monsieur OUABDESSELAM, Directeur de notre Ecole
et Professeur,
c'est grâce à ses conseils que nous avons pu mener
à bien nos études,

notre profonde gratitude.

A Monsieur BOISRAYON, expert UNESCO, Professeur,
qui nous a formé dans tant de domaines,

notre respectueuse reconnaissance.

A Monsieur BOUMAHRAT, Chef du Centre de Calcul
et Professeur,
c'est avec lui que nous fîmes nos premières armes
dans l'informatique,

nous l'en remercions.

A Monsieur SOLAIR, Professeur de Probabilités-Statistiques,
son concours nous fut précieux pour notre formation,

nos vifs remerciements.

A Monsieur BUDIN, Professeur d'Economie,
Formateur et ami,

notre respectueuse affection.

A Monsieur VALENZA,
il nous a aidés,

nous ne l'oublierons pas.

A Monsieur GIDE,

qui s'est intéressé à nos travaux.

A Messieurs DAMINE et OUFERHAT,

qui nous ont apporté leur précieux concours.

A nos Amis Ahmed MESSILI
Abed BOT,

notre travail en commun nous a rapprochés.

A nos Amis,

A tous ceux qui ont aidé à la réalisation pratique de cet ouvrage,

nos remerciements.

PLAN DE L'ETUDE

I) INTRODUCTION

I.1 DESCRIPTION ET BUT DU PROJET

Le but du projet était essentiellement de fournir à l'utilisateur d'un ordinateur IBM 1130 un sous système conversationnel lui permettant de résoudre la majorité des problèmes de traitement de fichiers par un accès direct.

Ainsi, le système moniteur 1130 qui n'était utilisable qu'en mode "BATCH" permettra un nouveau mode de fonctionnement "mode conversationnel" ; l'utilisateur pouvant alors agir directement lors de la mise au point, l'assemblage ou la compilation et l'exécution de ses programmes.

Tout système conversationnel, pour être performant, doit mettre à la disposition de son utilisateur un langage simple, à utiliser avec des temps de réponse relativement courts, car il ne s'adresse pas nécessairement à des experts en informatique.

Désormais, la pénurie de cartes au centre de calcul de l'E.N.P. ne sera plus un problème, les données, les programmes d'un utilisateur introduits sous "EDIT" (éditeur du sous-système) à partir de la console seront conservés sous forme de fichiers sur disque magnétique.

L'inconvénient majeur d'un conversationnel pour le 1130 est d'immobiliser la machine pour une seule personne ; néanmoins il est intéressant à utiliser lorsqu'on manipule des fichiers de grosse taille.

Les ambitions de ce sous système sont les suivantes :

- faire de l'édition sur des fichiers

c'est-à-dire pouvoir créer un fichier, l'examiner, modifier son contenu au moyen d'un jeu de commandes

- assembler ou compiler un programme se trouvant sur disque
- enfin demander l'exécution et la contrôler.

I.2 ETUDES ET REALISATIONS PRATIQUES

Une étude du système 1130 (qui sera détaillée plus loin) nous a permis de déceler certaines carences.

Essentiellement conçu pour fonctionner en mode "BATCH", il permet difficilement l'intégration d'un système conversationnel. En effet l'assembleur 1130 ne prévoit pas l'assemblage d'un programme se trouvant sur disque.

Un conversationnel nécessite donc un assembleur qui puisse réaliser des opérations de lecture sur disque.

Cette modification n'a pu être apportée faute de n'avoir pas eu la documentation nécessaire (Programme source de l'assembleur).

En résumé nos réalisations pratiques se sont réduites à l'écriture d'un éditeur disposant de qualités conversationnelles mais n'ont pu aboutir à la mise en place d'un véritable sous système.

II) GENERALITES SUR LES SYSTEMES

INTRODUCTION HISTORIQUE

Les systèmes d'exploitation des ordinateurs ont suivi une évolution parallèle aux différentes générations d'ordinateurs.

Les premiers systèmes de programmation furent des systèmes de sous-programme (la machine EDCSAC construite à l'Université de Cambridge en 1949 qui fut le premier calculateur électronique à programme enregistré faisait de la bibliothèque de sous-programme une des bases de la programmation). Ces systèmes comportaient comme simple langage des langages d'assembleur assez lents.

La deuxième génération vit du point de vue système s'affirmer le mode séquentiel avec de gros trains de travaux (BATCH) parallèlement à la naissance de langages évolués (FORTRAN) ; et vers la fin de la deuxième génération, un effort fantastique de programmation a été fait avec la multiplication des langages évolués différents (ALGOLS, COBOL,..) entrainant la toute puissance du BATCH.

La troisième génération après 1965 : avant ce troisième stade technologique on a vu s'annoncer une modification dans l'exploitation des ordinateurs. Dès 1962 pour tenter de supprimer l'inconvénient des systèmes à traitement séquentiel (c'est-à-dire absence totale d'interaction entre programmeurs et la machine) on commença à essayer de faire travailler plusieurs utilisateurs en même temps. On assista donc à la naissance des systèmes multiprocesseurs, multiprogrammés et conversationnels.

- Système à traitement séquentiel :

Dans le mode séquentiel d'utilisation des machines de la première génération chaque utilisateur travaille à tour de rôle sur la machine. Le temps de rotation des programmes est de l'ordre de l'heure.

Pour mieux utiliser le temps de travail de la machine, on passe au mode séquentiel indirect : le temps de rotation de travail est de l'ordre de la minute mais le temps de réponse c'est-à-dire celui qui s'écoule entre le moment où l'utilisateur donne son programme et celui où il obtient les résultats reste de l'ordre de l'heure à cause du groupement de travaux en train important (BATCH).

- Mode conversationnel :

Ce mode de travail est réalisé quand l'utilisateur peut intervenir directement et immédiatement pendant la compilation, l'assemblage, le chargement et l'exécution de ses programmes ceci avec des temps de réponse très courts.

- Système multiprogrammé :

C'est l'exécution simultanée de deux ou plusieurs programmes dans un même ordinateur.

Les systèmes multiprogrammés conversationnels permettent une utilisation constante de l'ordinateur en travaillant en temps partagé avec plusieurs utilisateurs de façon conversationnelle.

II.1 DESCRIPTION DE L'ORDINATEUR IBM 1130 DE L'ECOLE NATIONALE POLYTECHNIQUE

L'ordinateur IBM 1130 est un ordinateur de troisième génération qui travaille avec un système séquentiel.

La configuration du centre de calcul E.N.P.A est la suivante :

- la mémoire centrale de 8 K
- l'unité de traitement 1131 comportant les commutateurs d'entrée pupitre, le clavier et l'imprimante du pupitre, l'unité à un seul disque.

Unités d'entrée/sortie :

- le lecteur de cartes 1442
- le perforateur de cartes 1442

) dispositifs à accès séquentiel, transferts d'information à sens unique, vitesse d'accès moyenne

- l'imprimante 1132 : dispositif à accès séquentiel, transfert d'information à sens unique, accès lent
- l'unité de disque : mode d'accès aléatoire, information relisible, accès rapide

- le clavier pupitre
- l'imprimante pupitre

) dispositifs à accès séquentiel, transferts d'information à sens unique, accès moyen

Modes de fonctionnement de ces dispositifs

- lecteur de cartes 1442
- perforateur de cartes 1442
- clavier pupitre
- imprimante pupitre

) Mode asynchrone avec interruptions fréquentes

- Unité de disque : mode asynchrone avec interruption en fin d'opération
- Imprimante 1132 : mode mixte entre les deux modes précédents

Toutes ces unités d'entrée/sortie ont pour moyen de fonctionnement les interruptions (déroutements).

. Moyens de fonctionnement de ces unités :

1°) Interruptions :

C'est un arrêt technologique du déroulement d'un programme, sans contrôle de celui-ci, pour exécution de certaines actions définies elles aussi de façon technologique.

2°) Priorités :

On introduit des priorités dans la nature des interruptions. Les priorités pour l'IBM 1130 sont décroissantes avec les niveaux d'interruption.

. Niveau 0 (plus forte priorité) : lecteur 1442

. Niveau 1 : imprimante 1132
(adapteur transmission synchrone)

. Niveau 2 : mémoire disque

. Niveau 3 : (unité affichage 2250)
(traceur de courbes 1627)

. Niveau 4 : perforateur 1442
imprimante pupitre
clavier pupitre
(lecteur et perforateur de bandes 1134 - 1055)
(lecteur de cartes 2501)
(imprimante 1403)
(lecteur optique marque 1231)

. Niveau 5 : interruption touche STOP PROG.

N.B. : Les unités citées entre parenthèses ne font pas partie de la configuration du centre E.N.P.A

3°) Canal :

C'est l'intermédiaire entre la machine et le dispositif d'entrée/sortie.

Seule l'unité de disque est reliée à la machine par un canal permettant un travail simultané avec l'unité centrale.

Les autres unités d'entrée/sortie ne sont reliées que par de simples "fils".

4°) Mémoires-tampons (buffers) :

Ce sont des zones de mémoires auxquelles peuvent accèder l'unité centrale et l'unité d'entrée/sortie pour la communication de l'information ; utilisées vu la différence de vitesse de fonctionnement de l'unité centrale et de l'unité d'entrée/sortie.

5°) Codification et transcodage :

Pendant longtemps, chaque constructeur de matériel établissait sa propre codification sans se soucier de celle des autres.

De plus, chaque dispositif d'entrée/sortie nécessite un moyen différent de représentation de l'information.

- lecteur de cartes : code HOLLERITH à 12 bits par caractère
- imprimante 1132 : code EBCDIC à 8 bits par caractère
- imprimante pupitre : code console 8 bits par caractère.

Le transcodage s'effectue par programmes au moyen de tables.

II.2 LE SYSTEME MONITEUR DISQUE 1130 VERSION 2

Ce système permet le fonctionnement permanent de l'ordinateur 1130 dans un contexte de traitement séquentiel par lots (BATCH).

Le système moniteur comporte 7 éléments distincts mais interdépendants :

- le Superviseur (SUP)
- le Programme Utilitaire Disque (DUP)
- l'Assembleur (ASM) et le Macro Assembleur
- le Compilateur Fortran (FOR)
- le Constructeur de charges mémoire (CLB)
- le Chargeur image mémoire (CIL)
- la Bibliothèque du Système.

Le Superviseur contrôle le système moniteur et fournit les liens nécessaires entre les programmes des utilisateurs et les programmes du moniteur.

Le Programme Utilitaire Disque est un groupe de programmes IBM qui exécute les opérations sur disque.

L'Assembleur convertit les programmes symboliques en programmes résultant en langage machine.

Le Compilateur Fortran traduit les programmes Fortran (Fortran 4) en programmes résultant en langage machine.

Le Constructeur de charges mémoire construit les programmes image mémoires à partir des programmes principaux résultant sur disque.

Le Chargeur image mémoire charge en mémoire les programmes ainsi que les sous-programmes qui les accompagnent. Il sert également d'interface aux programmes du moniteur.

La Bibliothèque du Système est un groupe de programmes résidant sur disque, ayant pour objet les fonctions :

d'entrée/sortie
de conversion de données
arithmétiques
d'initialisation de disque.

A la génération du système les programmes IBM sont chargés sur disque par le Chargeur du Système.

ORGANISATION DU DISQUE

+ Description du cylindre 'IDAD = cylindre 0

1) Secteur 0 = secteur 'IDAD
   (Voir schéma ci-contre).

2) Secteur 1 = secteur 'DCOM

Il contient la zone de communication sur disque :

Les paramètres qui doivent être fournis par un programme moniteur à un autre qui sont obtenus à partir de la mémoire sur disque (et non pas de la mémoire centrale).

DCOM se divise en 2 parties. La première partie co tient les paramètres qui se rapportent à toutes les cartouches (dans une unité multidisques). La deuxième partie de DCOM contient les paramètres spécifiques à chaque cartouche :

- l'ID (identification) de la cartouche
- l'adresse de la LET
- l'adresse de protection des fichiers...

Chaque paramètre de la deuxième partie est sous la forme d'une table de 5 mots correspondant chacun à l'une des 5 cartouches possible.

Mots indicateurs de 'DCOM :

. Mot indicateur de zone de travail (WS)

Il contient le nombre de blocs de disque (DB) occupés par chaque programme ou fichier de données dans la WS. Ce mot est remis à 0 après toute opération de stockage vers l'UA ou la WS.

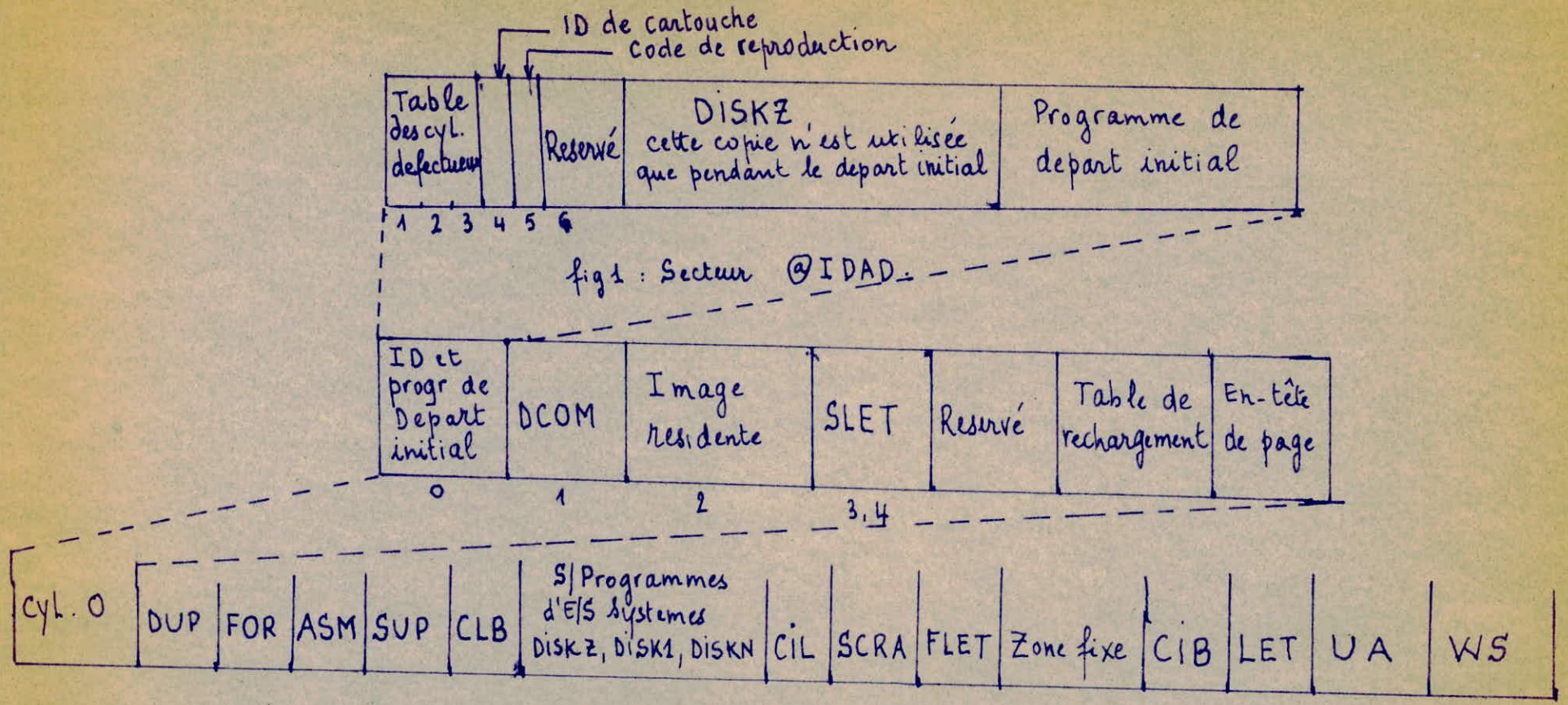


fig 1 : Secteur @IDAD

Description d'1 cartouche système

1) ASM et FOR peuvent être supprimés par l'utilisateur
2) FLET est présente seulement si l'utilisateur a défini une zone fixe pour cette cartouche
3) Zone fixe est définie facultativement par l'utilisateur
4) CIB ne peut être supprimée de la cartouche système par l'utilisateur
5) UA : bibliothèque du système ; des programmes écrits par l'utilisateur peuvent y être ajoutés.

. Mot indicateur de format

Il indique le format de tout programme ou fichier de données dans la WS (DSF, DCI ou DATA).

. Mot indicateur de mode temporaire

Il indique si le JOB en cours est temporaire.

3) Secteur 2 = secteur 'RIAD

Il contient l'image résidente (copie du Moniteur).

4) Secteurs 3 et 4 = secteur 'SLET

C'est la table d'équivalence des emplacements du système.

5) Secteur 5 : Réservé

6) Secteur 6 = secteur 'RTBL

Il contient la table de rechargement.

7) Secteur 7 = secteur 'HDNG

Utilisé pour stocker l'entête de page (imprimée en haut de chaque page imprimée par un programme du Moniteur).

+ Autres zones disque du Système Moniteur

(voir schéma ci-contre)

Le système IBM est chargé sur disque à la suite du cylindre 0 dans l'ordre indiqué dans le schéma.

. Zone des sous-programmes des unités connectées à l'unité centrale ; elle contient :

- les sous-programmes pour le fonctionnement des unités d'entrée/sortie
- les sous-programmes de conversion de code caractères
- les sous-programmes DISKZ, DISK1, DISKN, sous-programmes

d'entrée/sortie disque qui sont placés dans cette zone plutôt que dans la bibliothèque du système car le CLB les traite de façon différente de celle employée pour les sous-programmes dans la bibliothèque.

REMARQUE : DISKZ est placé sur le disque 2 fois, ici et dans le secteur 'IDAD avec le programme d'initialisation où il ne sert que pour lui.

+ Zone des enregistrements de contrôle du superviseur = SCRA

enregistre les LOCAL, NOCAL, FILES et (G2250).

+ Zone FLET

répertoire du contenu de la Zone Fixe (FX) qui la suit directement.

+ Zone intermédiaire image mémoire = CIB

sert au CLB de mémoire auxilliaire pour sauvegarder les COMMON bas et pour construire les charge mémoires.

+ Zone LET

répertoire du contenu de la zone utilisateur (UA).

+ Zone utilisateur = UA

c'est la bibliothèque.

+ Zone de travail = WS

c'est tout ce qui reste. Elle est utilisée pour le stockage temporaire. Elle suit la zone utilisateur et s'étend jusqu'à la fin du disque.

LES PROGRAMMES DU MONITEUR

LE SUPERVISEUR

C'est un groupe de programmes et de zones chargés du contrôle du système moniteur.

Il lit les enregistrements de contrôle dans les fichiers d'entrée, les interprète et appelle le programme approprié pour effectuer cette opération. Il prend le contrôle lors de la procédure de départ initial.

La partie du Superviseur qui se trouve en mémoire centrale est le "Moniteur Résidant".

Le Moniteur Résidant :

composé de 3 parties :

1°) une zone de données pour la communication entre programmes COMMA

2°) le squelette du Superviseur

3°) un des sous-programmes d'entrée/sortie disque.

- COMMA est intercalé dans différentes parties du squelette du Superviseur.

- Squelette du Superviseur : principalement les points d'entrée :

+ LINK : c'est le point d'entrée du sous-programme qui passe du contrôle d'un module à l'autre.

Remarque : le constructeur de charges mémoire (CLB) agit comme Superviseur intermédiaire pour permettre de réaliser le transfert du contrôle d'un module à l'autre.

+ EXIT : c'est le point d'entrée du sous-programme qui passe du contrôle d'un module au Superviseur.

+ DUMP : c'est le point d'entrée pour le sous-programme qui permet une analyse mémoire (copie dynamique ou copie terminale).

+ ILS02 : c'est le point d'entrée pour les interruptions de niveau 2 ; nécessaire dans le moniteur résidant pour réaliser les interruptions disque pour pouvoir charger à partir du disque toute routine nécessaire.

+ ILS04 : c'est le point d'entrée pour les interruptions de niveau 4. L'utilisateur pouvant intervenir à tout moment, il faut que cet ILS appartienne au moniteur résidant.

+ Déroutement pour erreur avant opération : $PRET
  Tous les ISS se branchent à $PRET + 1 (WAIT) en cas d'erreur préopératoire ou instruction PAUSE en FORTRAN.

+ 4 Déroutements pour erreur après opération : $PSTi
  i = 1 à 4 pour chaque niveau d'interruption en cas d'erreur après le début d'opération.

+ Déroutement niveau 5- Touche STOP PROG : état machine conservé

+ Déroutement pour demande d'interruption (INT REQ.)
  Tous les indicateurs sont mis hors fonction et un aiguillage de COMMA indique au Superviseur d'ignorer les enregistrements d'entrée jusqu'au prochain JOB. (Toutefois les parties du moniteur qui ne peuvent être interrompues, par ex. SYSUP, mettent en attente cette demande d'interruption).

- Le sous-programme d'entrée/sortie disque
DISKZ ou DISKI ou DISKN

	Emplacement mémoire	Taille
DISKZ	Fin moniteur résidant +1	480
DISKI	"	660
DISKN	"	930

DISKZ est pris par défaut si l'utilisateur ne spécifie pas quel sous-programme d'entrée/sortie disque il va utiliser.

Programme du Superviseur sur disque

En fonction du point d'entrée dans le moniteur résidant, le CIL va chercher l'un de ces programmes et lui donne le contrôle.

1°) analyseur des enregistrements de contrôle du moniteur (appelé par $EXIT) :

. lit un enregistrement de contrôle du moniteur ou du Superviseur

. imprime cet enregistrement sur l'imprimante principale (1132)

. va chercher le programme du moniteur demandé et lui transmet le contrôle.

2°) Zone des enregistrements de contrôle du Superviseur :

c'est la zone où sont mémorisés les enregistrements de contrôle FILES, LOCAL, NOCAL (et G2250).

L'ASSEMBLEUR

A l'origine des calculateurs la programmation en langage de la machine se faisait de façon purement numérique. La première idée permettant de faciliter le travail d'écritures des programmes a été d'utiliser des codes mnémoniques pour les instructions et de les faire traduire par un employé humain, ... puis par la machine. Cette idée est la base des langages d'assemblage.

Instructions de la machine

Nous ne citerons pas ici toutes les instructions ASM disons simplement quelques mots sur leur format : il peut être de 2 types

1 mot mémoire pour les instructions en format court

2 mots mémoire pour les instructions en format long.

L'adressage symbolique

C'est la fonction fondamentale d'un assembleur. La définition et l'utilisation des symboles correspondent à la création et à la consultation d'un dictionnaire.

L'assembleur 1130 assemble en 2 passages. Au cours du premier on définit par l'évolution du compteur d'adresses ce dictionnaire. Le second passage utilise les valeurs obtenues au premier, pour obtenir le programme résultant.

Conversion et compteur d'adresses : le compteur d'adresses indique l'adresse de l'instruction suivante. L'assembleur manipule ce compteur d'adresses (IAR) pour pouvoir réserver les zones et les remplir de données.

Macro-Assembleur

L'Assembleur 1130 dispose d'un Macro-Assembleur qui permet de définir et conserver un squelette de Macro-instruction.

Une Macro-instruction est constituée comme suit :

Déclaration de Macro		MAC (ou SMAC)	
Prototype	Para 1	nom	Para 2,...
Corps de la Macro	(		
Fin de Macro		MEND	

Instruction d'affectation :

ex. affecter à
Param la valeur Val : Param SET Val
la notation est fonctionnelle.

Branchements et boucles : ils permettront l'assemblage conditionnel.

Facilités d'écriture

Des pseudo-instructions aident la programmation

Pseudo-instruction DC
HDNG (pour les titres)
EJCT (saut de page)
SPAC (saut de lignes)
...

FORMAT DES FICHIERS SUR DISQUE

Fichiers de Type "données" (DATA) :

- Format DONNEES DISQUE (DDF) :

Ce format contient 320 mots binaires par secteur ; ceux-ci sont placés séquentiellement sans indication de début ou de fin et sans mots indicateurs.

Programmes :

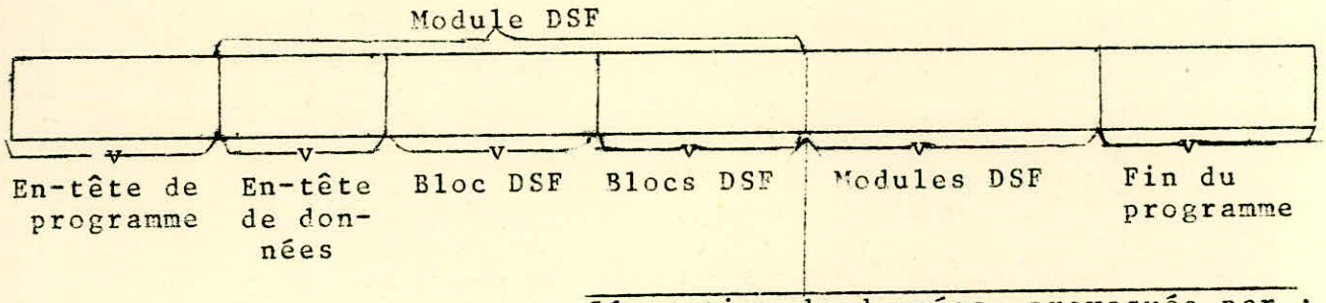
- Types de programme :

Type	Type de programme
1	Principal (absolu)
2	Principal (translatable)
3	Sous-programme, autre qu'un ISS, appelé par LIBF
4	Sous-programme, autre qu'un ISS, appelé par CALL
5	Sous-programme ISS, appelé par LIBF
6	Sous-programme ISS, appelé par CALL
7	Sous-programme ILS

- Sous-types de programme :

Sous-type	Type	Description
0	3,4	Sous-programmes en mémoire centrale
1	3	Sous-programmes d'entrée/sortie disque FORTRAN
2	3	Sous-programmes arithmétiques
3	3	S/P d'entrée/sortie FORTRAN pour unités autres que les disques et S/P de conversion "Z".
3	5	Sous-programmes d'unité "Z"
8	4	Sous-programme de fonction
1	7	ILS02, ILS04 factices

- Format système disque (DSF)



Séparation de données, provoquée par :
- une instruction ORG, BSS, BES ou DAS
- début d'un nouveau secteur
- fin de programme.

. En-tête de données : 2 mots :

1er mot : adresse de chargement

2è mot : nombre de mots entre cet en-tête et l'en-tête de données suivant, plus le nombre de mots de l'en-tête de données suivant.

. Bloc DSF : 2 à 9 mots de données :

1er mot : mot indicateur de données

2è au 9è mot : mots de données.

. Fin de programme : 2 mots :

1er mot : adresse relative de la position disponible suivante

2è mot : zéro.

. En-tête de programme : 12 à 51 mots (éventuellement)

Mots pour tous les types	Contenu
1	Zéro
2	Somme de contrôle si l'entrée est effectuée par cartes ; zéro sinon
3	Type de programme (bits 4 à 7) sous-type (bits 0 à 3) et précision (bits 8 à 15)
4	Longueur réelle du programme ; i. e adresse finale du programme
5	Longueur de la zone COMMON (en mots)
6	Longueur (en mots) de l'en-tête de programme - 9
7	Zéro
8	Longueur du programme y compris l'en-tête (en disque-blocs)
9	Indicateur FORTRAN (bits 0 à 7) ; nombre de fichiers définis (*FILES)
10 et 11	Nom du point d'entrée 1
12	Adresse du point d'entrée 1 (absolue pour les programmes de type 1 ; relative pour tous les autres).

Types 3 - 4 mots	Contenu
13 et 14	Nom du point d'entrée 2
15	Adresse relative du point d'entrée 2
16 à 51	Noms des points d'entrée 3 à 14 éventuellement, suivant le même format que pour le point d'entrée 2. L'en-tête de programme se termine sur l'adresse relative du dernier point d'entrée ; la longueur de l'en-tête de programme est donc variable.

Types 5 - 6 mots	Contenu
13	Numéro d'ISS plus 50
14	Numéro d'ISS
15	Nombre de niveaux d'interruption nécessaires
16	Numéro du niveau d'interruption associé à l'interruption primaire
17	Numéro du niveau d'interruption associé à l'interruption secondaire.

N. B. : Le lecteur perforateur de cartes 1442 est la seule unité exigeant plusieurs niveaux d'interruption.

Type 7 mot 13	Numéro du niveau d'interruption associé.

- Format image mémoire disque (DCI)

	Programme principal	Sous programmes	FLIPR	ZONE LOCAL	ZONE SOCAL	COMMON	T.V.	LOCAL	SOCAL

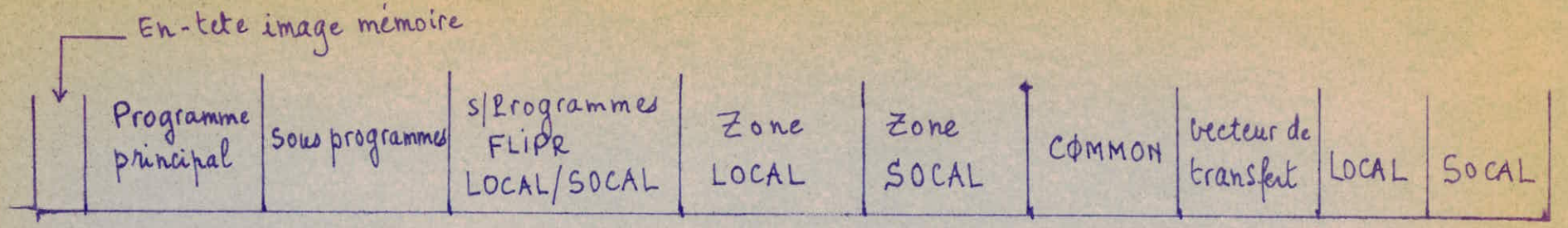
V
En-tête
Image Mémoire

• En-tête image mémoire

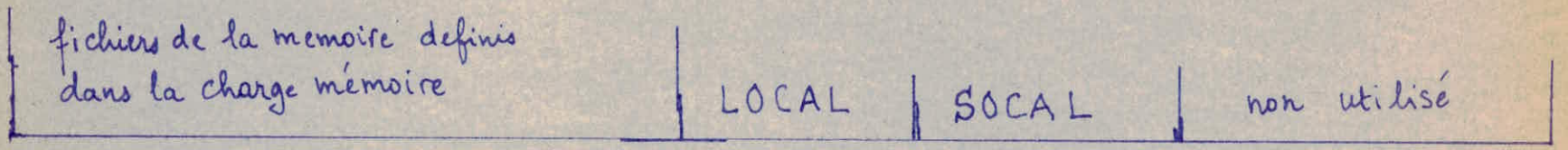
MOT

Symbole	Adresse Relative	Contenu
'XEQ1	1	Adresse d'exécution de la charge mémoire
'CMON	2	Longueur de la zone COMMON (en mots)
'DREQ	3	Indicateur de S/P d'E/S disque : /FFFF = DISKZ /0000 = DISK1 /0001 = DISKN
'FILE	4	Nombre de fichiers définis
'HWET	5	Longueur de l'en-tête image mémoire (en mots)
'LSCT	6	Compte de secteurs des fichiers en W. S.
'LDAD	7	Adresse de chargement de la charge-mémoire
'XCTL	8	Adresse de contrôle du point de sortie DISK1 ou DISKN
'TVWC	9	Longueur du T. V. (en mots)
'WCNT	10	Longueur de la charge-mémoire (en mots)
'XR3X	11	Positionnement pour XR3 au cours de l'exécution de la charge mémoire
'ITVX	12 à 17	Contenu des mots 8 à 13 au cours de l'exécution
	18 à 20	Réservé
	21 à 26	Points d'entrée dans les ISS des unités : 1231, 1403, 2501, 1442, clavier et imprimante pupitre, 1055 et 1134
'OVSW	27	Compte de secteurs des S/P LOCAL et SOCAL
'CORE	28	Dimension (taille) mémoire de la charge mémoire
	29 et 30	Zone de travail de la somme de contrôle de la table de définition de fichier.

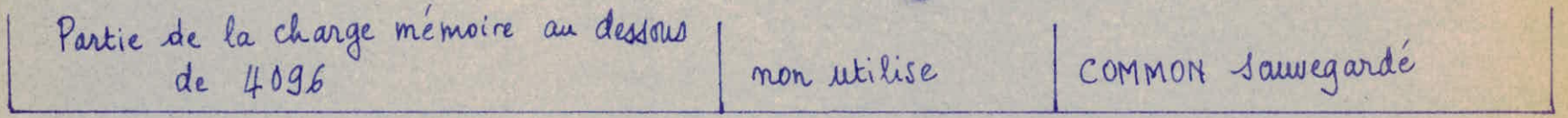
Organisation d 1 programme image mémoire en cours de construction.



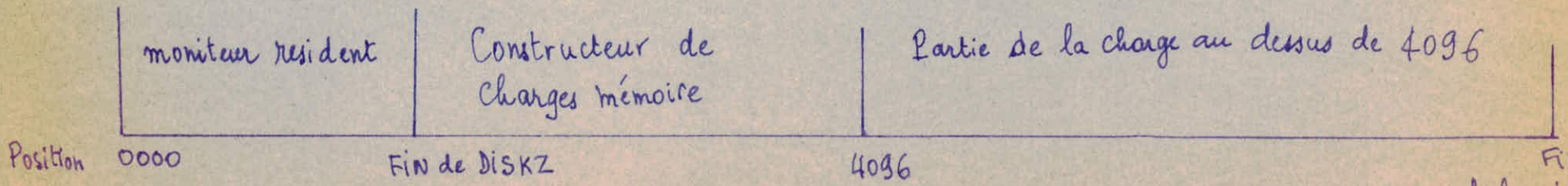
. dans la WS



Mémoire de travail



. zone intermediaire image memoire



. Mémoire centrale

LE CONSTRUCTEUR DE CHARGES MEMOIRE (CLB)

Il construit un programme principal donné en programme image mémoire. Le programme principal et tous les programmes qui lui sont nécessaires (y compris LOCAL et SOCAL) est converti du format système disque (DSF) en format image mémoire disque (DCI). Pendant cette conversion, le CLB génère l'enregistrement d'en-tête image mémoire et le vecteur de transfert (TV).

Le programme image mémoire obtenu peut être immédiatement exécuté ou mis en place sur le disque format DCI.

CLB peut être appelé par le Superviseur (par XEQ)
le DUP
le CIL

Construction de charges mémoire :

- Traitement du contenu de la zone SCRA :

Les enregistrements LOCAL, NOCAL, FILES (et G2250) sont lus et analysés dans la zone des enregistrements de contrôle du superviseur (SCRA) ; des tables sont constituées ; elles serviront lors de la construction du programme image mémoire.

- Conversion du programme principal :

Avant tout autre, le programme principal est converti du format système disque (DSF) en format image mémoire sur disque (DCI).

- Incorporation des sous-programmes :

Tous les sous-programmes appelés sont incorporés à la charge mémoire, sauf le sous-programme d'entrée/sortie disque, les LOCAL et SOCAL.

Si des LOCAL ou SOCAL sont utilisés, le S/P de chargement LOCAL/SOCAL (FLIPR) est inclus dans la charge mémoire.

- Zones prévues pour les LOCAL et SOCAL :

La zone LOCAL (resp. SOCAL) aura la dimension du S/P LOCAL (resp. SOCAL) le plus important.

FLIPR chargera les LOCAL dans la zone LOCAL (les SOCAL dans la zone SOCAL) lors de leur appel.

- Construction de l'en-tête image mémoire :

Le CLB construit l'en-tête image mémoire qui contient les informations nécessaires au CIL pour initialiser la charge mémoire en vue d'exécution. Cet en-tête fait partie du programme image mémoire et réside en mémoire centrale pendant l'exécution.

- Utilisation de la CIB et de la WS :

Le CLB place dans la zone intermédiaire image mémoire (CIB) toutes les parties de la charge mémoire qui, lorsqu'elles sont chargées, occupent une place inférieure à 4096 (le reste de la charge mémoire est placé en mémoire centrale dans la zone qu'il devra occuper).

- Origine de la charge mémoire :

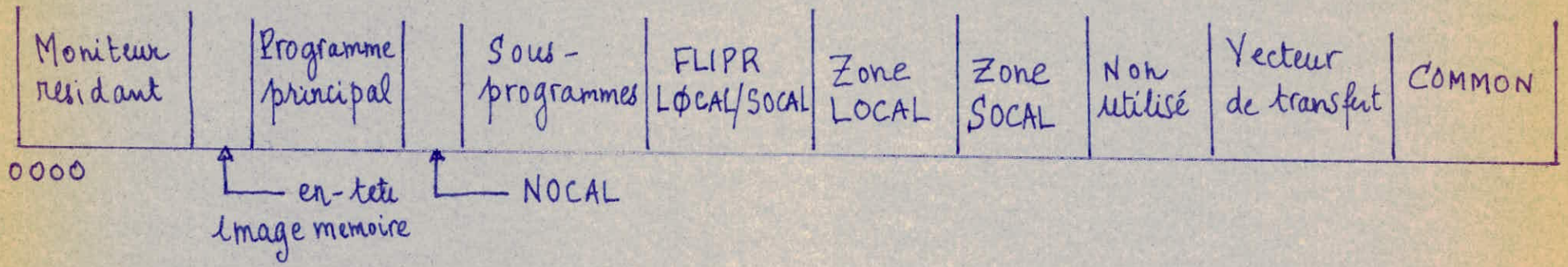
Elle est fonction du S/P d'entrée/sortie disque.

	Origine (en Hexa-) de la charge mémoire
DISKZ	/01FE
DISK1	/02B2
DISKN	/03C0

Nota : Pour les programmes en absolu, l'utilisateur doit fixer le point origine à 30 mots au moins après la fin du S/P d'entrée/sortie disque. Ces 30 mots sont nécessaires à l'en-tête image-mémoire.

- Vecteur de transfert (TV) :

LIBF TV est une table pour les S/P appelés par LIBF.



Disposition d'une charge memoire prête pour l'execution.

Chaque élément de la LIBF TV se compose de 3 mots :

Le premier sert à sauvegarder l'adresse de retour,

les deux autres à effectuer un branchement long vers le S/P appelé.

Un élément de CALL TV est un mot contenant l'adresse absolue du S/P appelé.

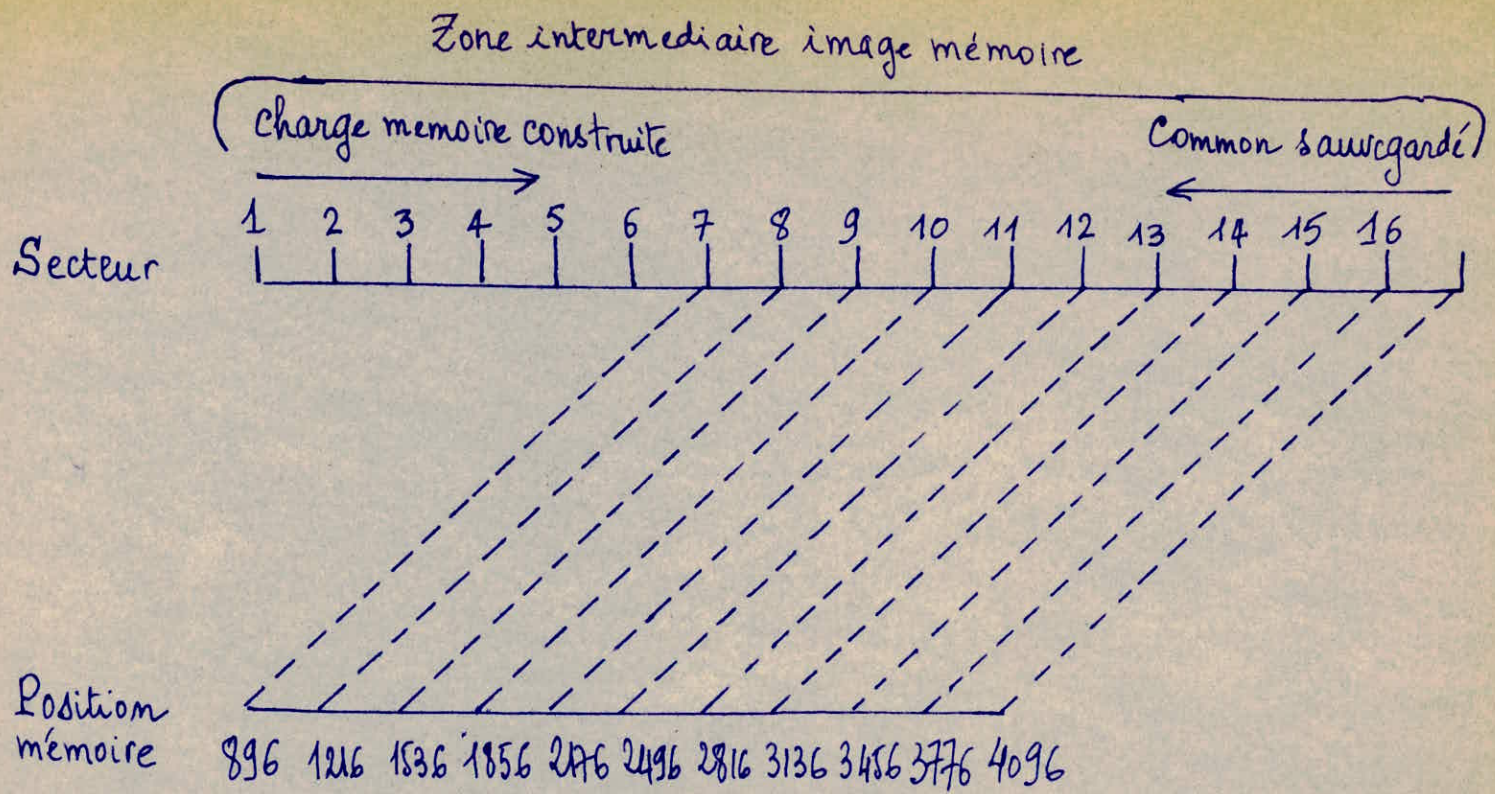
Dans le cas d'un S/P LOCAL (resp. SOCAL) l'élément de CALL TV contient l'adresse de la liaison spéciale du programme LOCAL (resp. SOCAL) et non l'adresse du point d'entrée du S/P.

- Recouvrements système : ce sont les SOCAL

Les recouvrements SOCAL ne sont jamais construits pour des charges mémoire dans lesquelles le programme principal est écrit en langage d'assembleur.

- Le S/P de chargement LOCAL/SOCAL (FLIPR) :

Lit un S/P LOCAL/SOCAL dans l'UA ou la FX et le transfère dans la zone LOCAL/SOCAL, dès qu'il est appelé à l'exécution (si l'exécution suit immédiatement la construction de la charge mémoire, FLIPR lit le LOCAL/SOCAL dans la WS).



. Méthode de sauvegarde de la zone COMMON entre les modules.

LE CHARGEUR IMAGE-MEMOIRE (CIL)

Il sert de chargeur pour les charges mémoire et d'interface pour certaines parties du système moniteur. Il est chargé et prend le contrôle lors de l'entrée dans le squelette du superviseur. C'est lui qui reconnait le point d'entrée ($EXIT, $DUMP, $LINK).

Mise en mémoire du superviseur :

- Point d'entrée $EXIT :

Le CIL charge DISKZ (s'il n'est pas en mémoire) ; il charge ensuite l'analyseur des enregistrements de contrôle du moniteur et lui donne le contrôle.

- Point d'entrée $DUMP :

Le CIL sauvegarde les mots 6 à 4095 dans la zone intermédiaire image mémoire (CIB), charge le programme de DUMP et lui transfère le contrôle. C'est DUMP qui restaure en mémoire centrale ce qui était dans CIB et rend le contrôle à la charge mémoire (DUMP dynamique), ou qui met fin à l'exécution par un CALL EXIT (DUMP terminal).

Mise en mémoire d'un module :

- Point d'entrée $LINK :

Le CIL sauvegarde dans la CIB le COMMON bas (il détermine d'après les renseignements de COMMA le mot qui, dans la zone COMMON éventuellement définie, porte l'adresse la plus basse. Il recherche sur disque le programme à charger ; et met en mémoire s'il n'y est pas le S/P d'entrée/sortie disque nécessaire à ce programme (DISK1 ou DISKN). Il restaure le COMMON bas ; la charge mémoire est mise en mémoire centrale et le contrôle lui est transmis.

. Si le module est en format système disque (DSF) CIL appelle CLB qui construira la charge mémoire et redonnera le contrôle à CIL pour la mise en mémoire du module.

III) GENERALITES SUR LES PROGRAMMES D'EDITION

III.1 CARACTERISTIQUES GENERALES

Un programme d'Edition doit mettre à la disposition d'un utilisateur un jeu de commande lui permettant de créer un fichier, de l'examiner et de modifier son contenu.

Ces programmes sont essentiellement orientés vers le traitement de caractères. Afin de résoudre les problèmes posés par ce genre de traitement l'utilisation d'un calculateur pour manipuler de l'information purement alphanumérique se développe.

Les langages de programmation tendent de plus en plus à introduire des opérations sur des variables de type chaine (PL/1, ALGOL 68) et parallèlement se développent des langages spécialisés (SNOBOL 4).

III.2 EVOLUTION DES PROGRAMMES D'EDITION

Un fichier est en général considéré comme un ensemble de N lignes rangées dans un certain ordre.

Il est nécessaire de spécifier la ligne sur laquelle ou à partir de laquelle on veut faire agir une commande donnée.

Pour se positionner sur une ligne quelconque la méthode classique consiste à donner le rang absolu de la ligne dans le fichier. Cette méthode n'est pas souple, le rang de la ligne peut évoluer au cours des insertions ou des suppressions.

Avec les systèmes conversationnels est apparu un nouveau type d'Editeur appelé "Editeur par contexte". On peut ainsi se positionner sur une ligne en donnant une partie de son contenu.

Parmi les Editeurs par contexte, signalons "EDIT" de C.M.S qui est le seul que nous connaissons.

Un stage de compléments en informatique effectué à l'I.M.A.G en août 1972 nous a fourni l'occasion d'utiliser l'Editeur de CMS et par la même nous a permis d'acquérir une petite expérience sur les programmes d'Edition en général.

Par la simplicité de son langage et ses temps de réponse courts, "EDIT" est agréable à utiliser pour les opérations courantes de création, modification de fichiers, etc...

III.3 UTILISATION DES PROGRAMMES D'EDITION

- Mise au point de programmes source :

Avant de demander la traduction d'un programme source en langage machine, il est nécessaire de pouvoir corriger les erreurs de syntaxe avant d'arriver à un programme au point, acceptable par le traducteur.

- Correction, mise en forme de données :

Les données d'un programme peuvent être créées par l'Editeur sous forme de fichiers ; il est nécessaire de pouvoir les corriger et les remplacer après chaque exécution du programme.

- Mise en page de textes :

Il s'agit de réaliser sur un texte donné la mise en page, le cadrage des titres, etc... par un programme de l'Editeur.

Sous CMS il existe la commande SCRIPT qui permet cette mise en page à condition que le texte ait été créé sous "EDIT", sous forme d'un fichier de type "SCRIPT".

IV) L'EDITEUR 1130

IV.1 DESCRIPTION EXTERNE - LANGAGE DE COMMANDE

L'Editeur 1130 donne la possibilité à l'utilisateur de créer un fichier, de l'examiner, de modifier son contenu au moyen d'un quinzaine de commandes.

Un fichier traité par "EDIT" est défini par son nom (5 caractères au maximum) et son type qui peut être soit ASM, FOR ou DAT.

D'autre part, il est considéré comme un ensemble de N lignes rangées dans un certain ordre et n'excèdant pas 80 caractères chacune (N arbitrairement grand).

Ces lignes sont rangées séquentiellement sur disque en code EBCDIC tassé (2 caractères par mot) dans des enregistrements de longueur fixe de 40 mots chacun.

VI.1.1 UTILISATION - DIALOGUE

Pour rentrer sous le contrôle du conversationnel, l'utilisateur doit taper les cartes de contrôle suivantes et les faire lire par le lecteur de cartes :

```
// ⊬ JOB
// ⊬ XEQ ⊬ EDIT   1      1
*LOCAL, CEDIT, WRITE, PUNCH, MISE
```

S'il veut utiliser la console ou

```
// ⊬ JOB
// ⊬ XEQ ⊬ EDITL   1      1
*LOCAL, CEDIT, WRITE, PUNCH, MISE
```

S'il veut faire une mise à jour automatique de fichier à partir du lecteur.

les "1" "1" devant être tapés sur les colonnes 17 et 19 de la carte.

En retour, sur l'Imprimante console s'imprimera

```
* BONJOUR
CONVERSATIONNEL *EDIT* A VOTRE SERVICE *
```

== : Le double signe égal signifiant attente de commandes du conversationnel.

Les commandes du conversationnel peuvent être alors tapées sur la console (EDIT) ou lues sur cartes (EDITL).

Pour sortir du conversationnel, il faut taper

/* en première colonne

==/*

Pour rentrer sous contrôle de l'Editeur, la commande

EDIT ¥ NOM ¥ TYPE

est nécessaire.

NOM : 1 à 5 caractères alphananumériques commençant obligatoirement par une lettre et sans blanc au milieu

Type : A pour Assembleur
D pour Données
F pour Fortran

Si le fichier n'existe pas encore, le conversationnel répond par un message sur la console :

* FICHIER A CREER. VEUILLEZ CREER UNE ENTREE DANS LA LET

et on attend une autre commande du conversationnel.

Si le fichier est trouvé, on rentre dans l'Editeur et le message

EDIT

s'imprime sur la console.

::: Les ":::" signifiant attente de commande de l'Editeur.

Après chaque commande, l'Editeur imprime la ligne courante (la ligne sur laquelle on est positionné). Toutes les lignes ne doivent pas dépasser 80 caractères ; les caractères au delà de 80 ne sont pas pris en compte.

Toutes les commandes de l'Editeur sont testées sur leur premier caractère. Il suffit de taper l'initiale de la commande (un caractère).

Exemple :

pour

```
:::DELETE
```

taper

```
:::D
```

A la console, le sous programme de lecture (TYPE 0) facilite la frappe.

La touche ← (retour arrière) annule le caractère précédemment tapé.

La touche EFF ZONE (Effacement Zone) annule la ligne tapée.

Si une ligne ne fait pas 80 caractères, la terminer par un FDZ.

IV.1.2 DEFINITION DU LANGAGE DE COMMANDES

Les commandes peuvent se classer en plusieurs catégories :

- Création de fichier :

Une fois qu'une place physique a été allouée au fichier, la création se fait tout simplement par la commande

```
::: INSER                    de l'Editeur
    ligne 1          et en tapant les lignes
    ligne 2          les unes à la suite des autres.
    .                Le /* signifiant fin de création.
    .
    .
    ligne n
    /*
```

- Commandes de positionnement

- Positionnement absolu :

HAUT : se positionner sur la première ligne du fichier.

BAS : se positionner au bas du fichier (au dessous de la dernière ligne du fichier).

LIGNE ∅ n : se positionner sur la ligne de rang n du fichier.

- Positionnement relatif :

+(n) : descendre de n lignes au dessous de la ligne courante. Si n est plus grand que le nombre de lignes restantes "EDIT" répond "FIN DE FICHIER" et on est positionné au bas du fichier).

-(n) : remonter de n lignes au dessus de la ligne courante. Si n est plus grand que le nombre de lignes qui se trouvent au dessus de la ligne "EDIT" répond "HAUT" et on est positionné en haut du fichier.

- Recherches par contexte :

SITUE /chaine de caractères/ : recherche dans le fichier à partir de la ligne courante et plus bas la ligne qui contient la première occurence de la chaine.

Si la chaine n'existe pas dans le fichier "EDIT" répond "FIN DE FICHIER" et on est au bas du fichier.

FIND ∅ (des tabulations et des "slash") chaine sans blanc ETIQUETTE
recherche dans le fichier à partir de la ligne courante et plus bas la ligne qui contient ETIQUETTE dans une colonne déterminée par les caractères "Tabulations" et "slash".

- Commandes de modification :

INSER : insère au dessus de la ligne courante les lignes
ligne 1 tapées après la commande.
ligne 2 Terminer par /+ en début de ligne.
.
.
.
ligne n
/+

DELETE (∅ n) : supprime n lignes à partir de la ligne courante (incluse).

Si n est trop grand "EDIT" supprime jusqu'au bas du fichier et répond "FIN DE FICHIER".

NEW : remplace la ligne courante par la ligne donnée en pa-
ligne ramètre.

CHANGE /chaine 1/chaine 2/ : change dans la ligne courante chaine 1 par chaine 2.

- Impression :

PRINT (Ø n) : imprime n lignes à partir de la ligne courante (incluse).

Si n est trop grand, il a impression de toutes les lignes jusqu'au bas du fichier et de "FIN DE FICHIER".

RANG : donne le rang de la ligne courante dans le fichier.

- Facilité d'écriture :

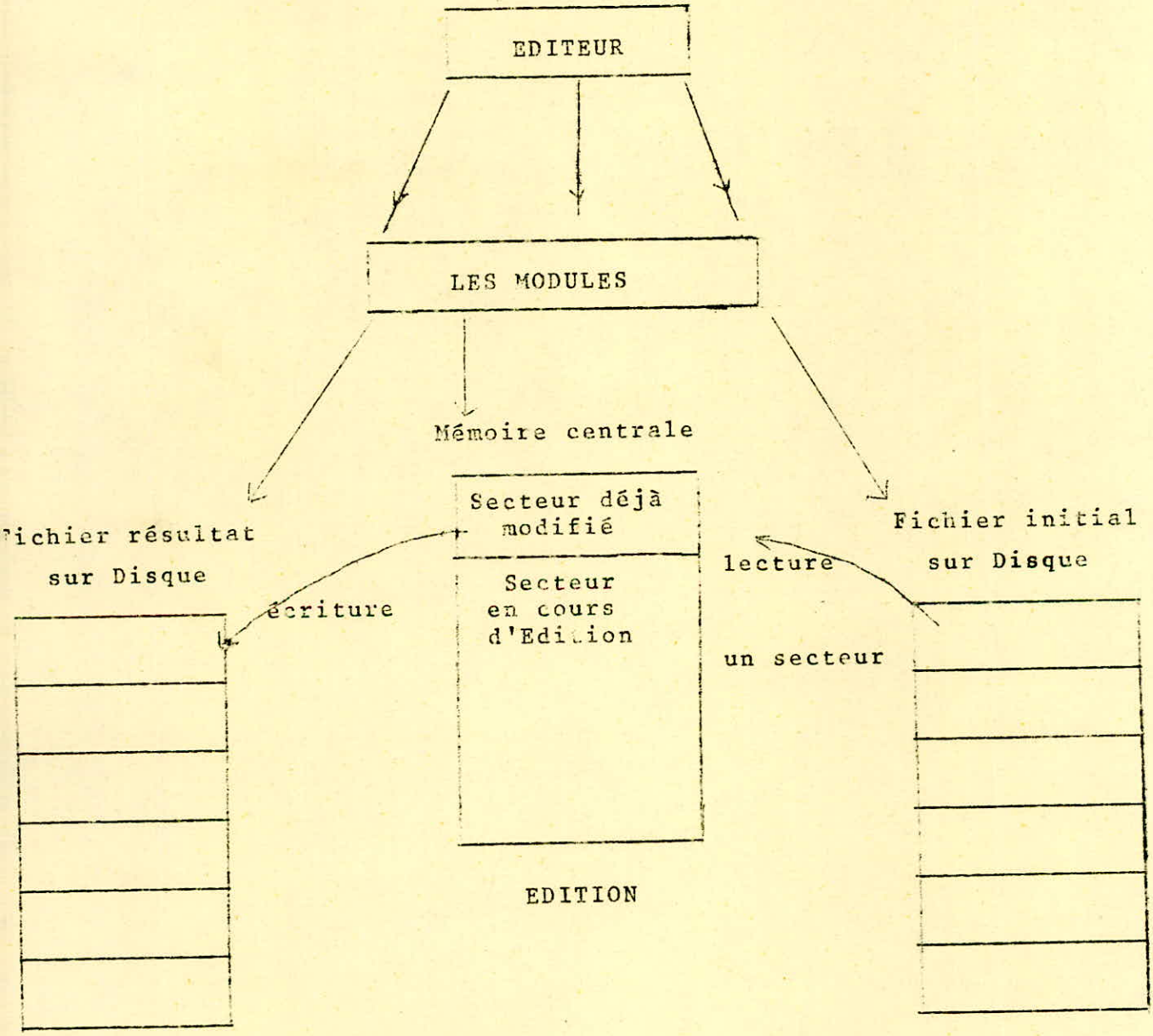
TAB (Ø n1 Ø n2 Ø n3....) permet des intervalles de tabulation n1, n2, n3.

- Sauvegarde du Fichier sur Disque :

GARDE : sauvegarde le fichier modifié et permet de sortir du contrôle d'EDIT et de repartir dans le conversationnel.

IV.2 CONCEPTION INTERNE DE L'EDITEUR

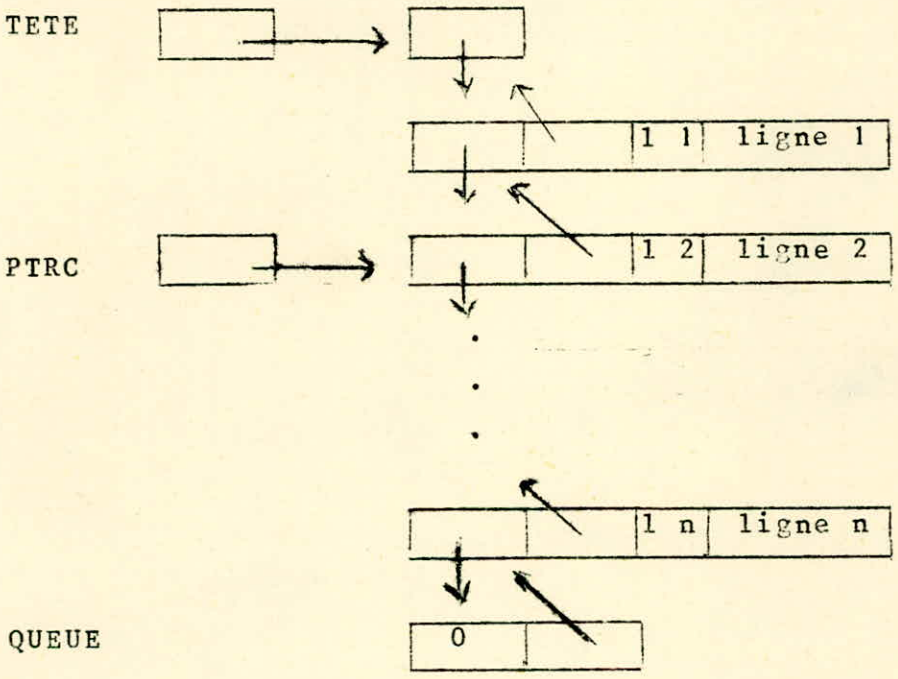
L'Editeur est un ensemble de Modules. Certaines réalisent les opérations de lecture écriture sur disque, les autres servent à traiter les commandes.



- Structure de l'information en mémoire centrale :

En mémoire centrale, l'information est sous forme de lignes chainées en liste bidirectionnelle auxquelles on associe un pointeur. La ligne repérée par ce pointeur à un instant donné s'appelle ligne courante.

Ces lignes sont de longueurs variables.; les blancs de droite ainsi que les 20 premiers blancs dans le cas d'un fichier assembleur sont supprimés. Un mot en tête de chaque ligne indique sa longueur.



TETE : pointe un mot qui contient l'adresse de la première ligne en mémoire centrale à un instant précis.

PTRC : pointe la ligne courante.

QUEUE : c'est une ligne fictive qui contient "0" pour indiquer qu'il n'y a plus de ligne en mémoire centrale.

NREL : ce mot contient le nombre de lignes en mémoire centrale situées au dessus du pointeur courant.

NABS : ce mot contient le rang absolu de la ligne courante dans le fichier.

Avec une telle structure, les insertions, les suppressions se ramènent à de simples modifications de pointeur.

L'Edition se fait secteur par secteur au moyen de 6 zones mémoires :

- Une zone fichier

destinée à recevoir le ou les secteurs en cours d'Edition (au maximum 2 secteurs coexistent en mémoire centrale).

- Une zone carte

destinée à recevoir une chaine de caractères qui sera interprétée comme une commande avec ses paramètres.

- Une zone LECT

Dans cette zone figurent les paramètres (la ligne à insérer, les chaines pour les recherches par contexte, etc...)

- Zone EXT

extension de la zone LECT (utilisée dans le module qui traite la commande CHANGE /chaine 1/chaine 2/, la première chaine se trouvant dans LECT, la seconde dans EXT).

- Une zone LIGNE

sert à recevoir une ligne destinée à être imprimée sur la console.

- Une zone de paramètres utilisés par les différents modules.

Toutes ces zones et paramètres sont placés dans une zone fixe de la mémoire haute : ce sera la zone COMMON.

- Les Modules de l'Editeur

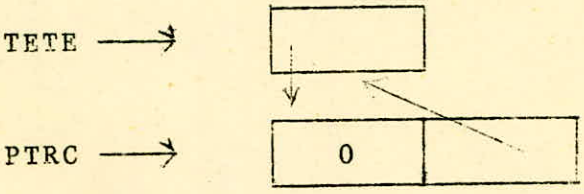
L'Editeur se compose d'un certain nombre de programmes de base nécessaires à l'édition et de modules qui traitent les commandes (nous dirons des primitives).

. Les Programmes de base

+ Le Module INIT :

Ce module permet l'initialisation des paramètres de la zone COMMON et la zone libre de la mémoire centrale.

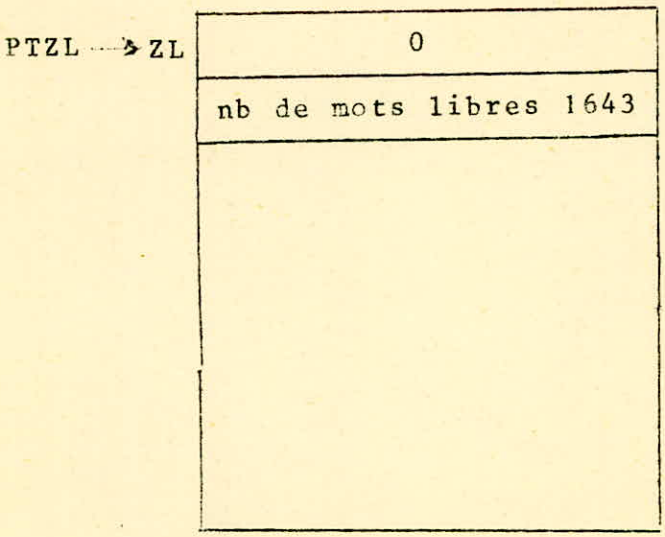
INIT fait entre autres les initialisations suivantes :



Il n'y a aucune ligne en mémoire centrale. PTRC pointe l'élément fictif.

Amener une ligne en m. c revient en fait à l'insérer.

- initialise la zone libre de la mémoire centrale

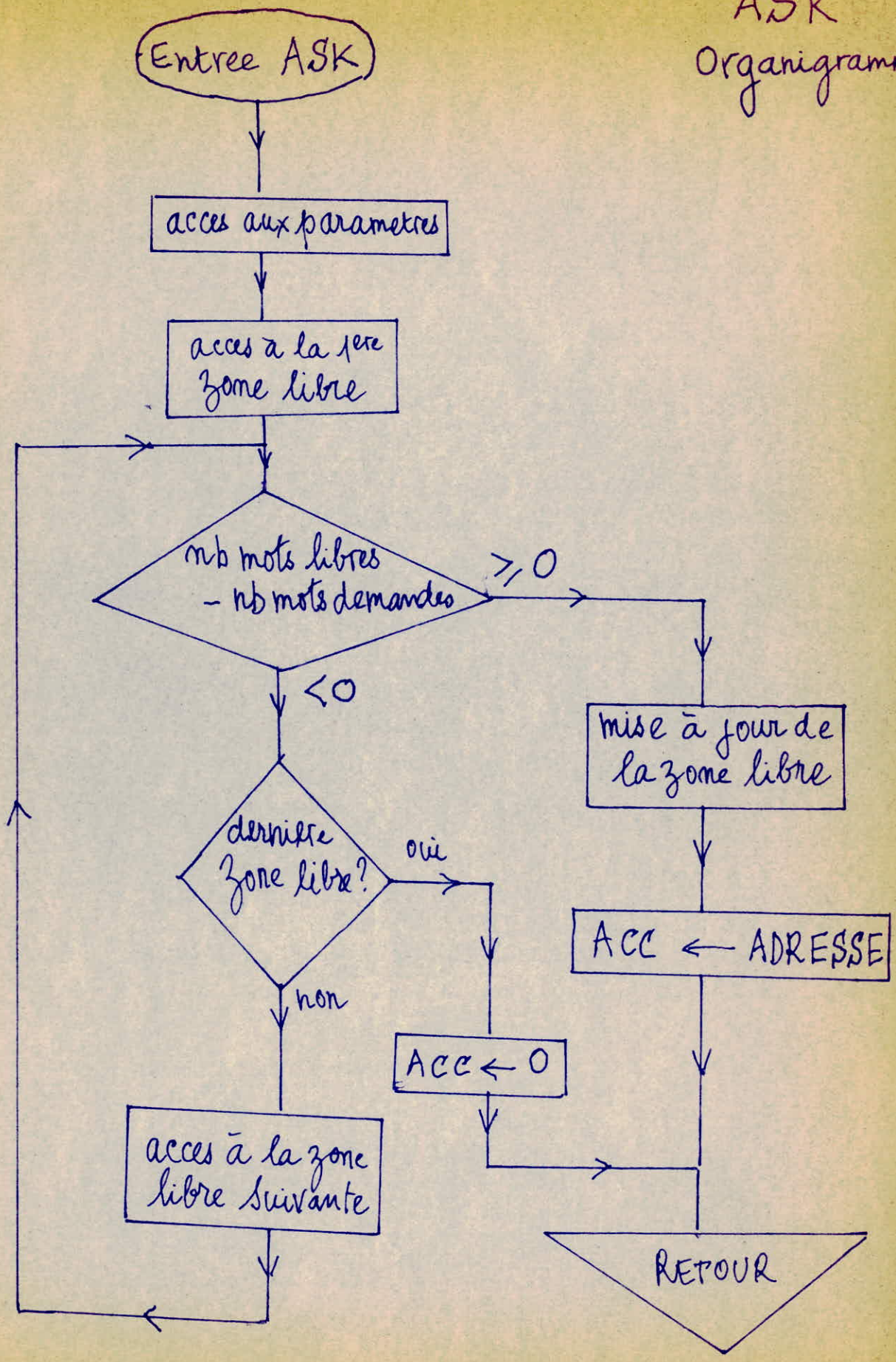


PTZL : pointeur de première zone libre.

ZL contient l'adresse de la zone libre suivante

ZL + 1 : nb de mots libre de cette zone

ASK
Organigramme



+ Le Module de Gestion dynamique de la mémoire centrale.

L'inconvénient d'une petite capacité mémoire (8K) nécessite l'écriture d'un module de gestion de la mémoire centrale afin de diminuer la place mémoire à occuper.

2 sous programmes :

- demande de place mémoire - ASK
- libération de place mémoire - FREE

formes des appels

```
CALL        ASK
DC          n       nombre de mots demandés
```

ASK fournit dans l'accumulateur l'adresse à partir de laquelle les n mots sont accordés ou "0" si pas de place

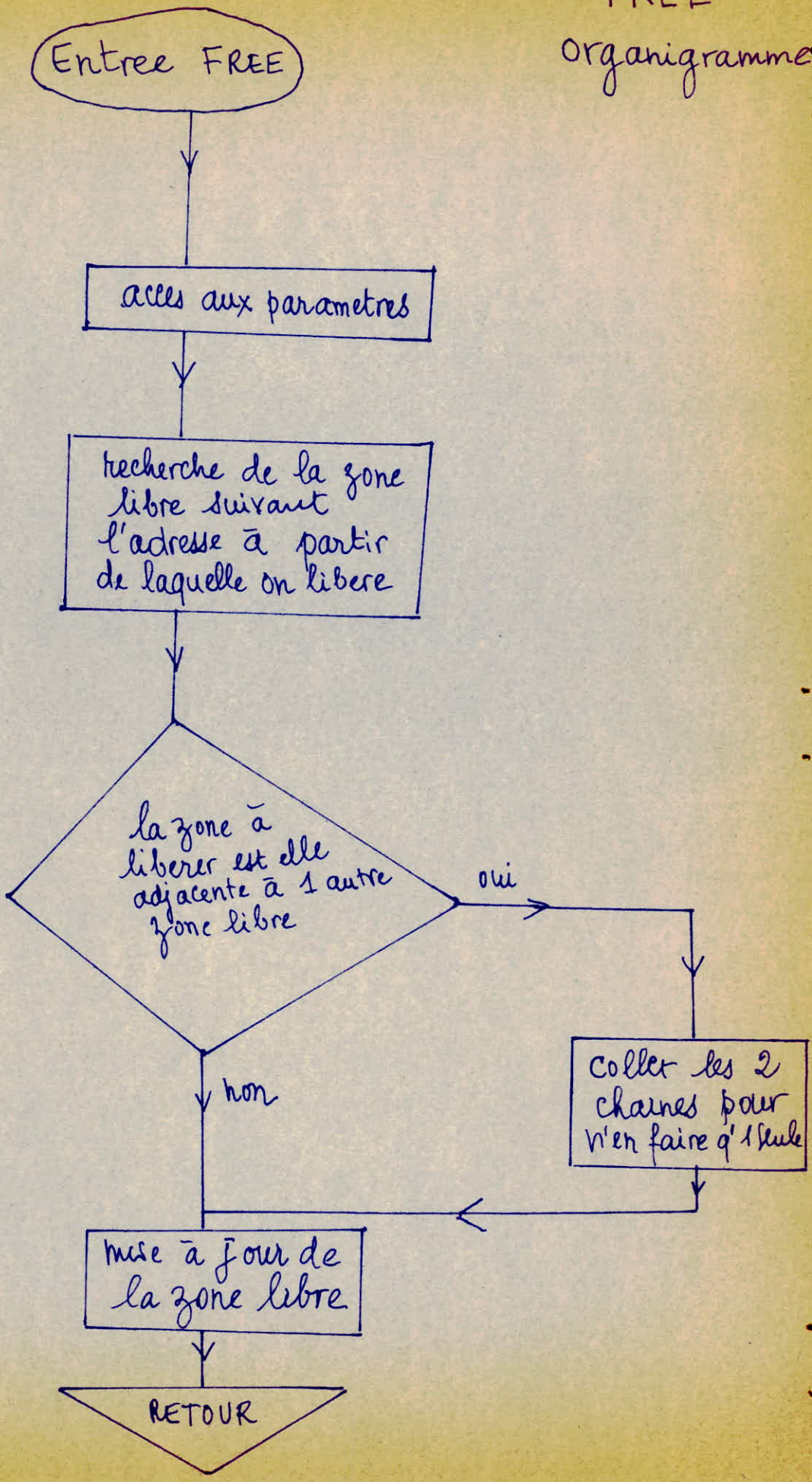
```
CALL        FREE
DC          ADr     à partir de laquelle on libère
DC          n       nb de mots à libérer
```

Exemples d'utilisation de ce module

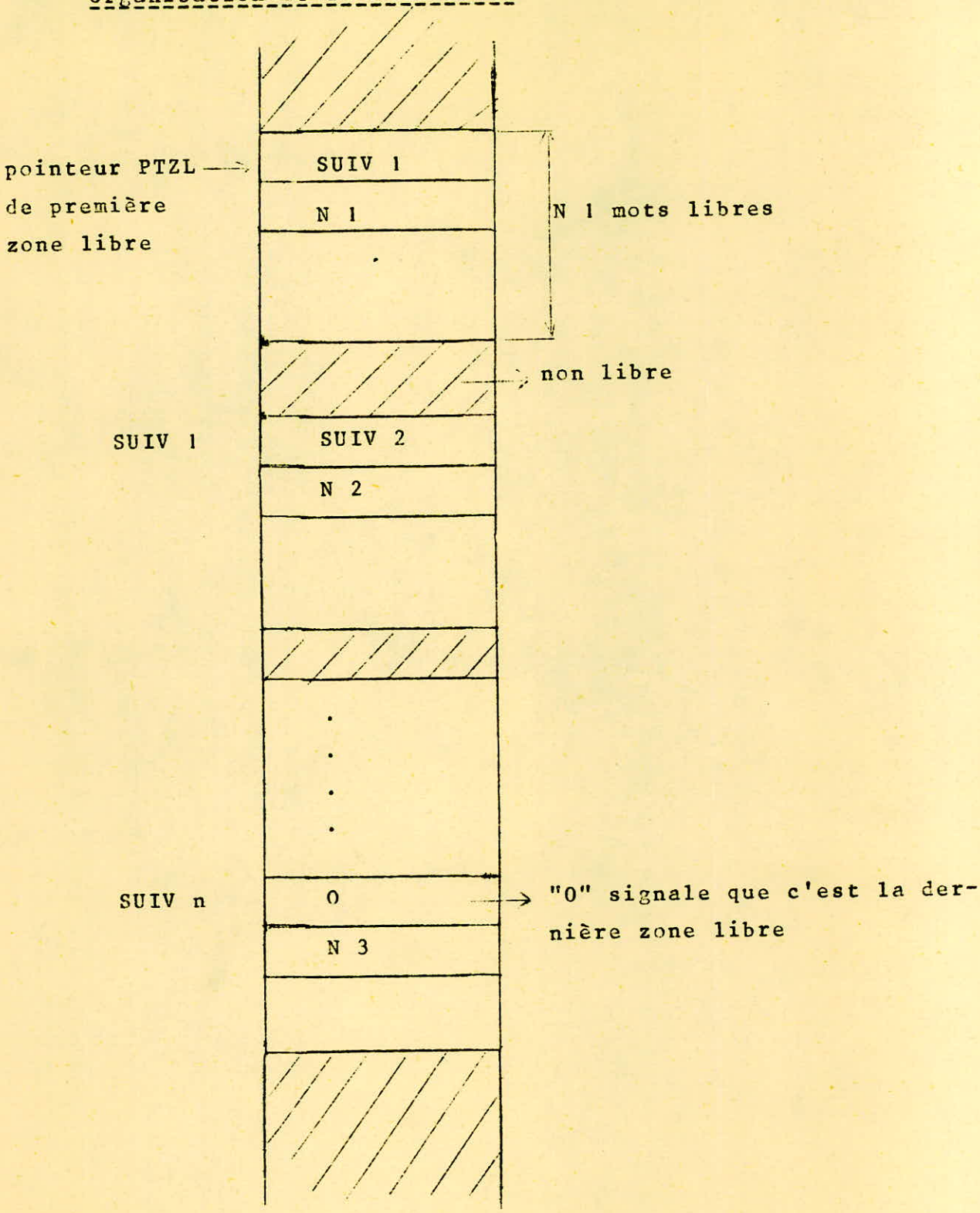
- Demande d'une zone de travail intermédiaire
- Demande de zone pour y mettre les liges du fichier, etc...

FREE

Organigramme

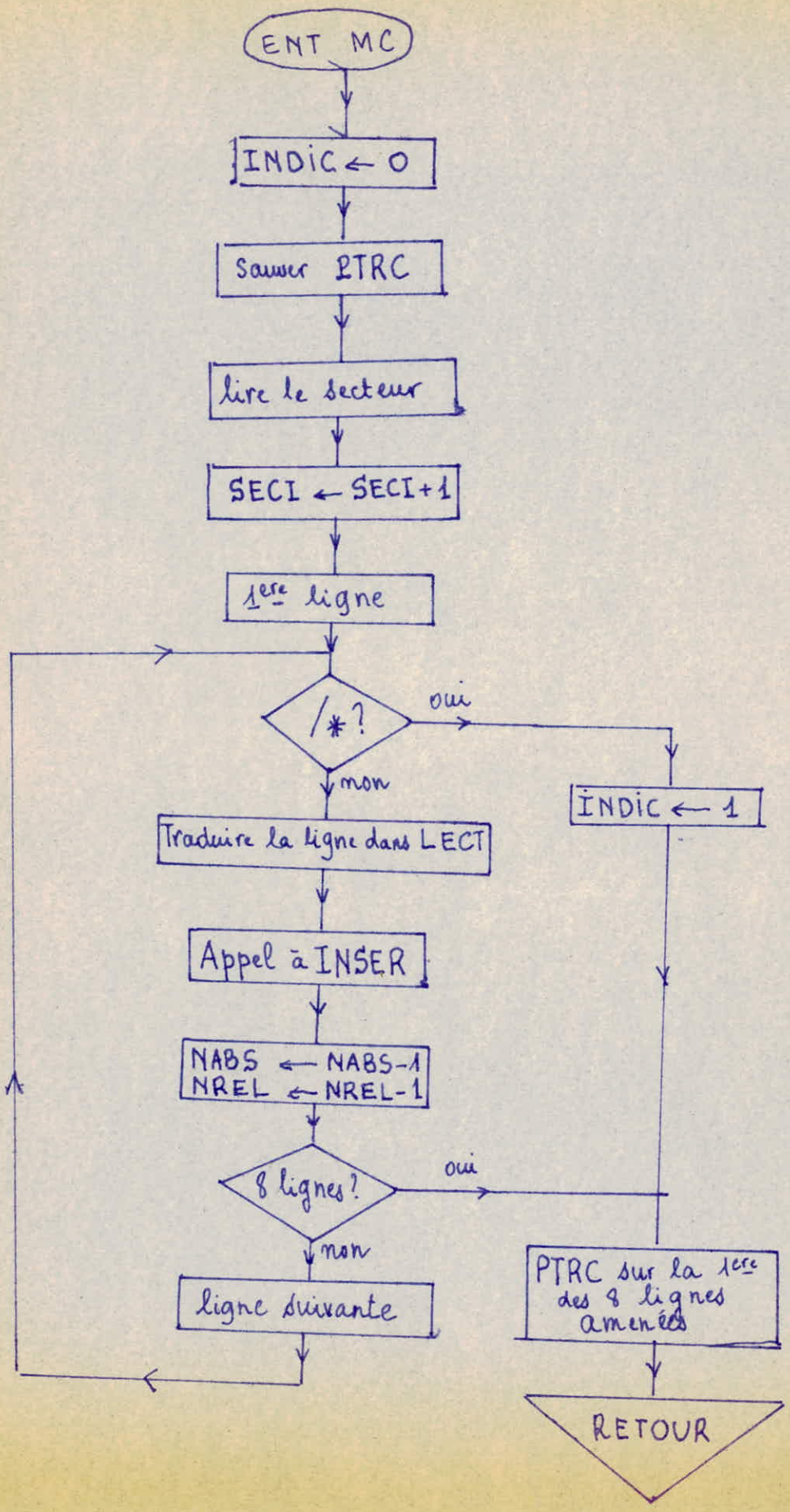


- Organisation de la zone libre



M.C

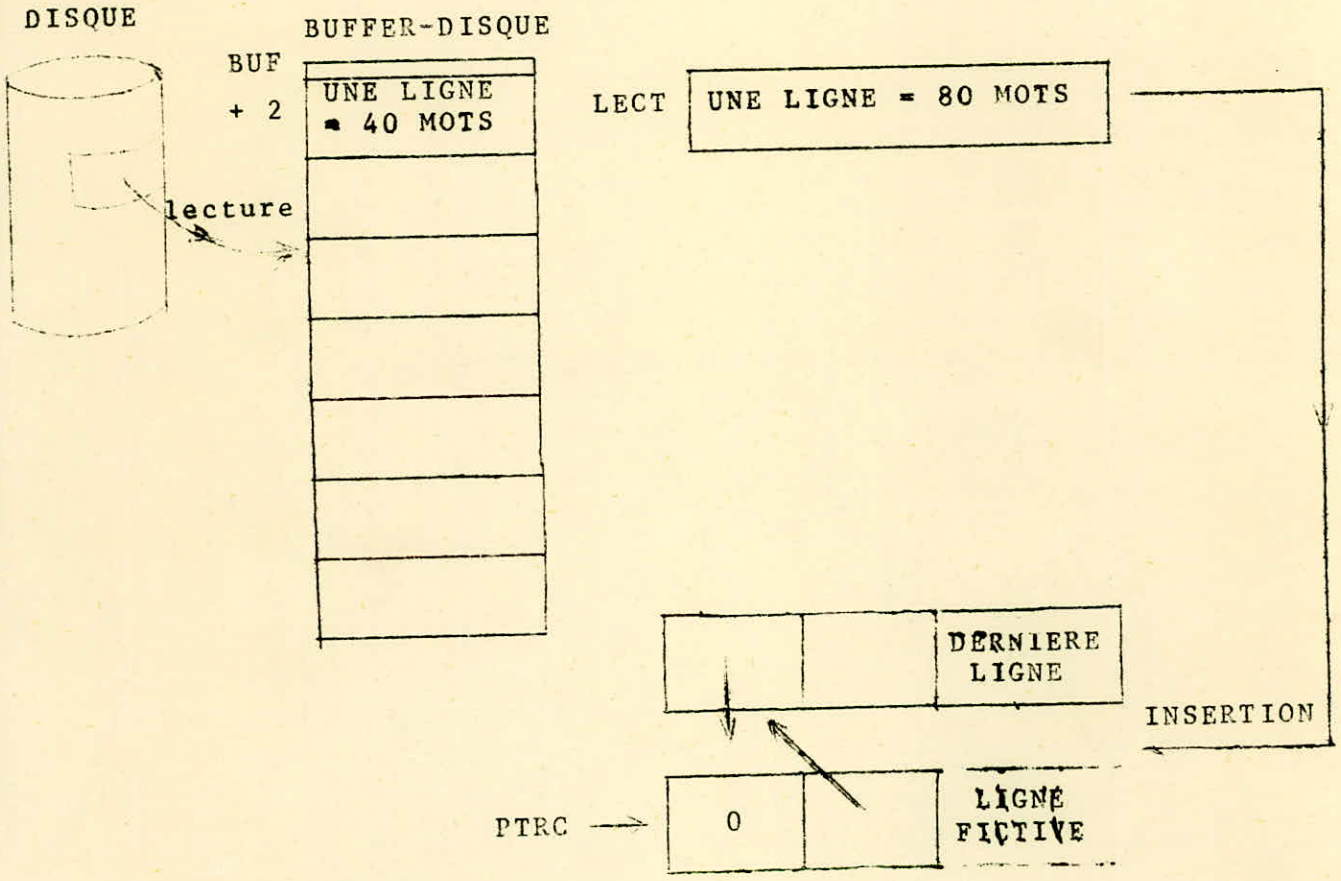
organigramme



+ Le Module MC

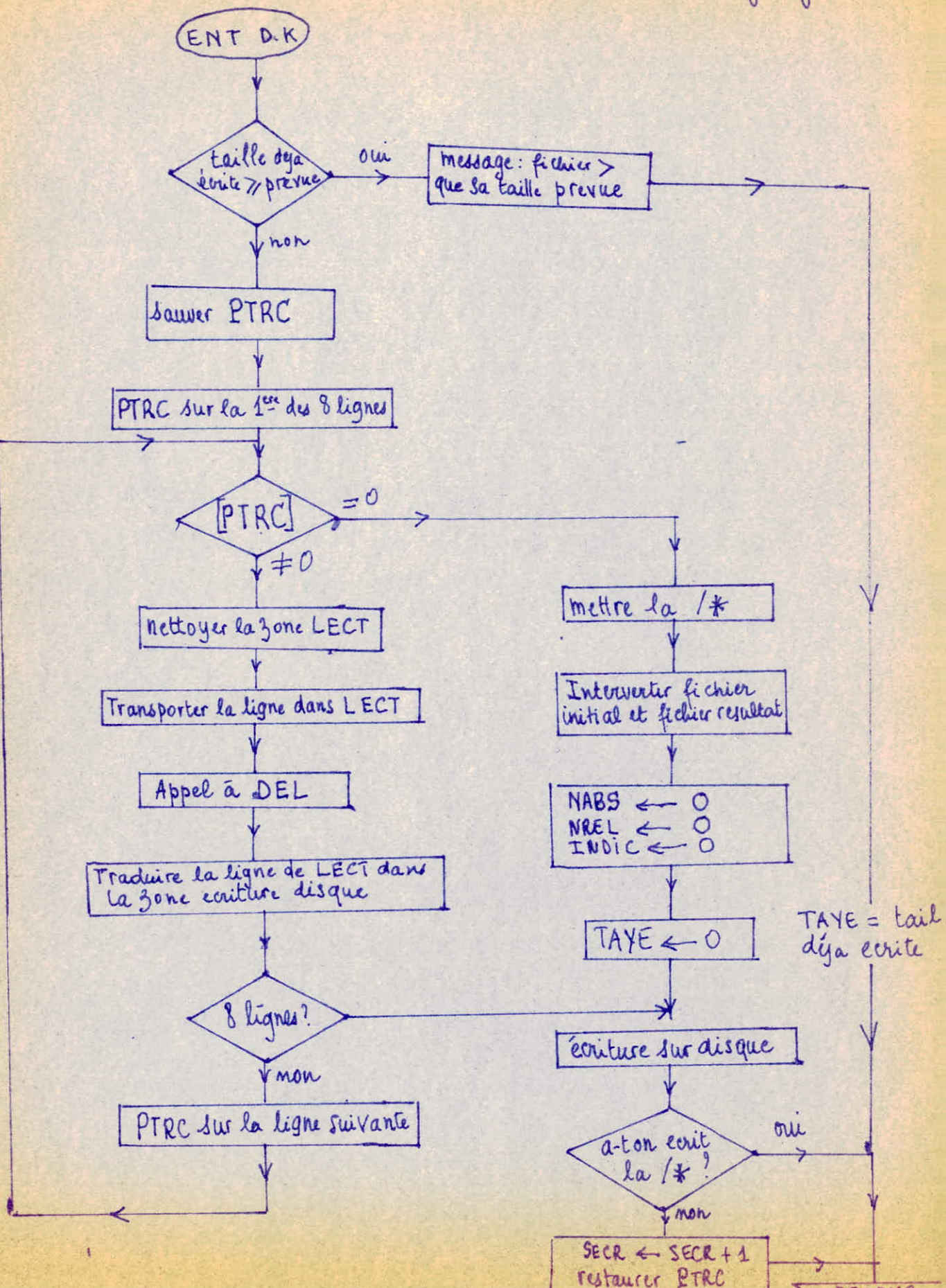
Dès que le secteur en cours d'édition a fini d'être examiné et modifié, il est sauvegardé dans le fichier résultat (qui se trouve temporairement dans la WS) et le secteur suivant doit être amené et chainé en liste double à la suite des lignes restantes en mémoire centrale. Ce travail est réalisé par le module MC.

Ce passage au secteur suivant est automatique dès qu'on est positionné sur la dernière ligne en mémoire centrale et qu'on désire passer aux suivantes.



D.K

organigramme



+ Le Module DK : c'est l'inverse de MC

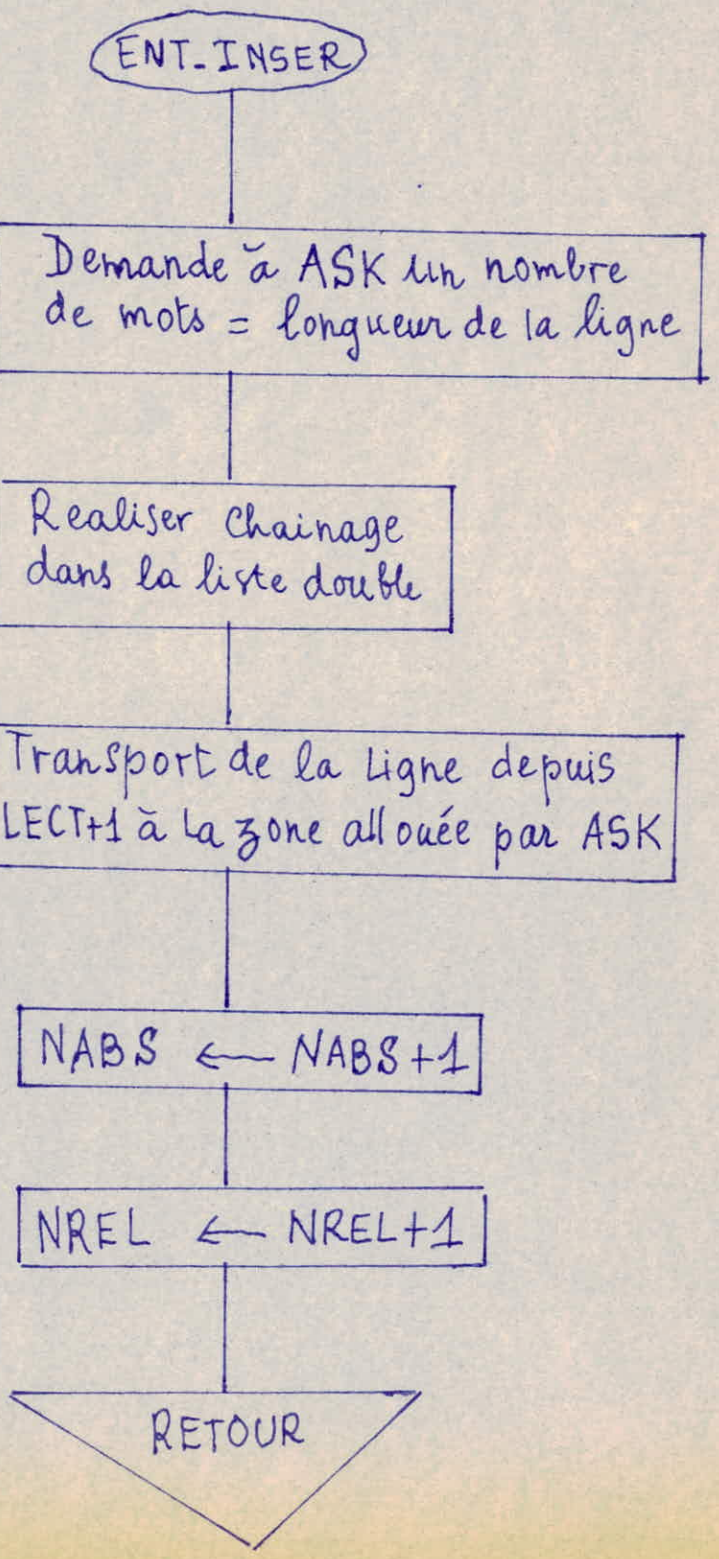
Le module DK écrit sur le fichier résultat le secteur déjà modifié. Dès que ce module rencontre la fin de fichier, il intervertit fichier initial et fichier résultat.

Ce module est appelé par NEXT ou par la commande d'insertion (SPI) lorsqu'il y a plus de 8 lignes au dessus du pointeur.

DK traduit les 8 premières lignes de la mémoire centrale dans le buffer disque et écrit dans le fichier résultat le secteur ainsi constitué.

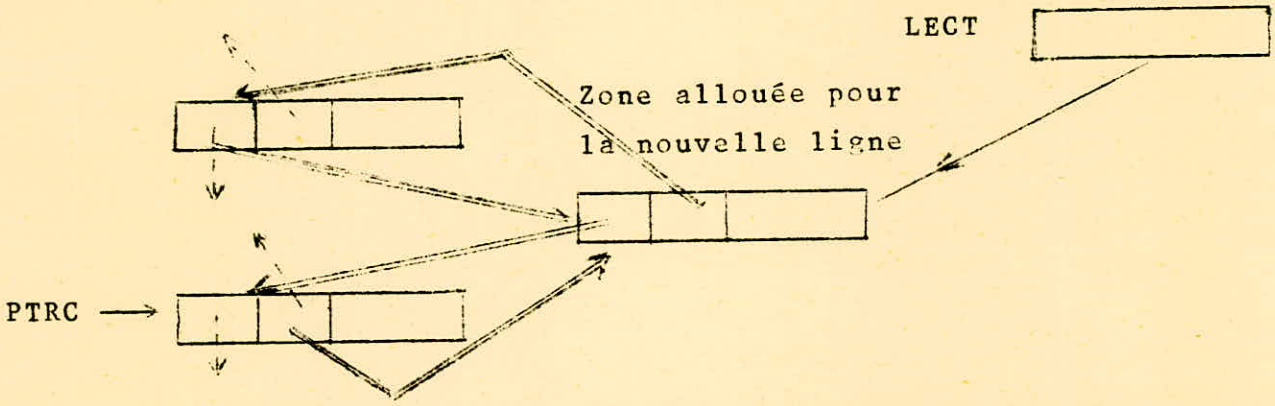
INSER

organigramm



- Les primitives de l'Editeur :

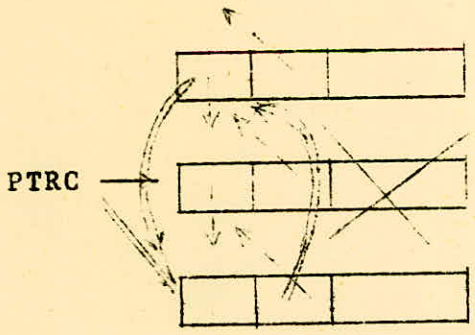
INSER : a pour fonction d'insérer le contenu de la zone LECT au dessus du pointeur.



En double trait, les pointeurs après insertion.

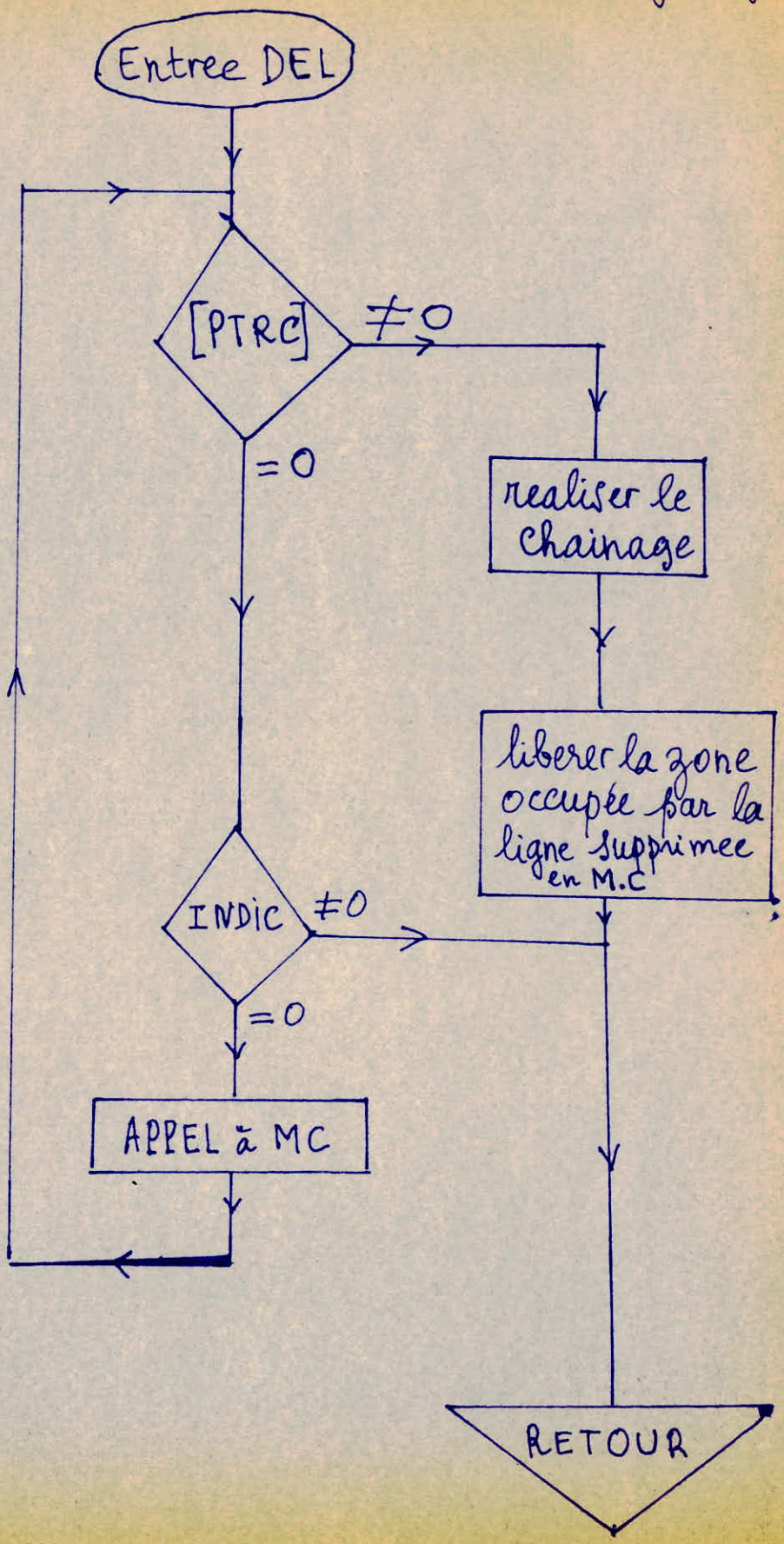
PTRC ne bouge pas de sa place.

DEL : a pour fonction de supprimer la ligne repérée par le pointeur.



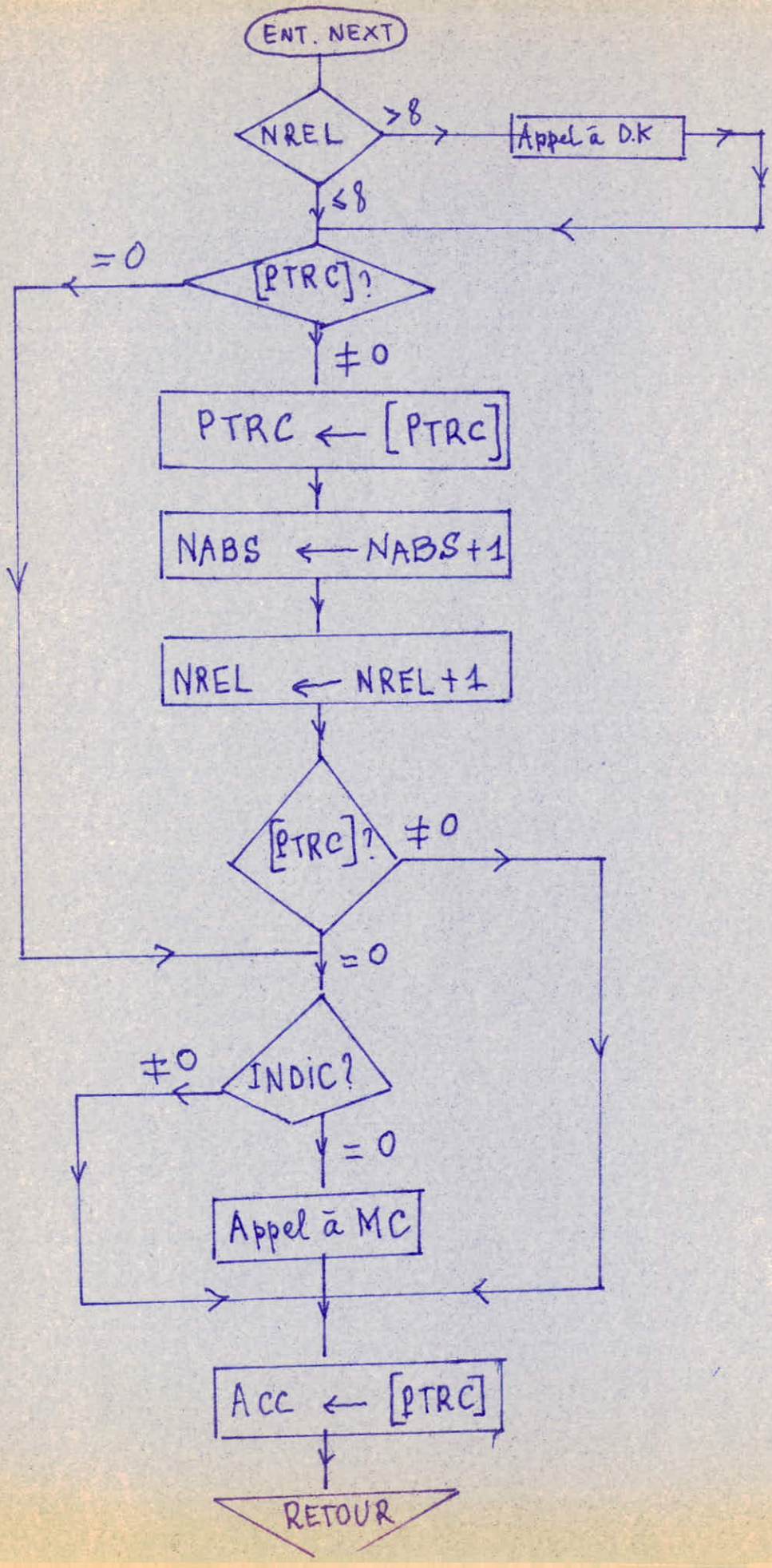
DEL

Organigramme

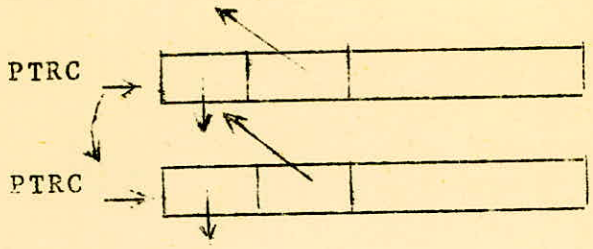


NEXT

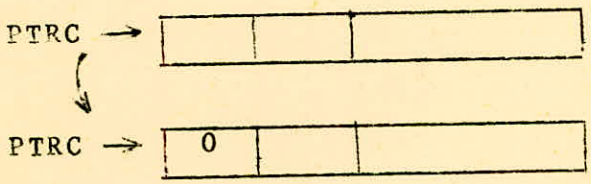
organigramme



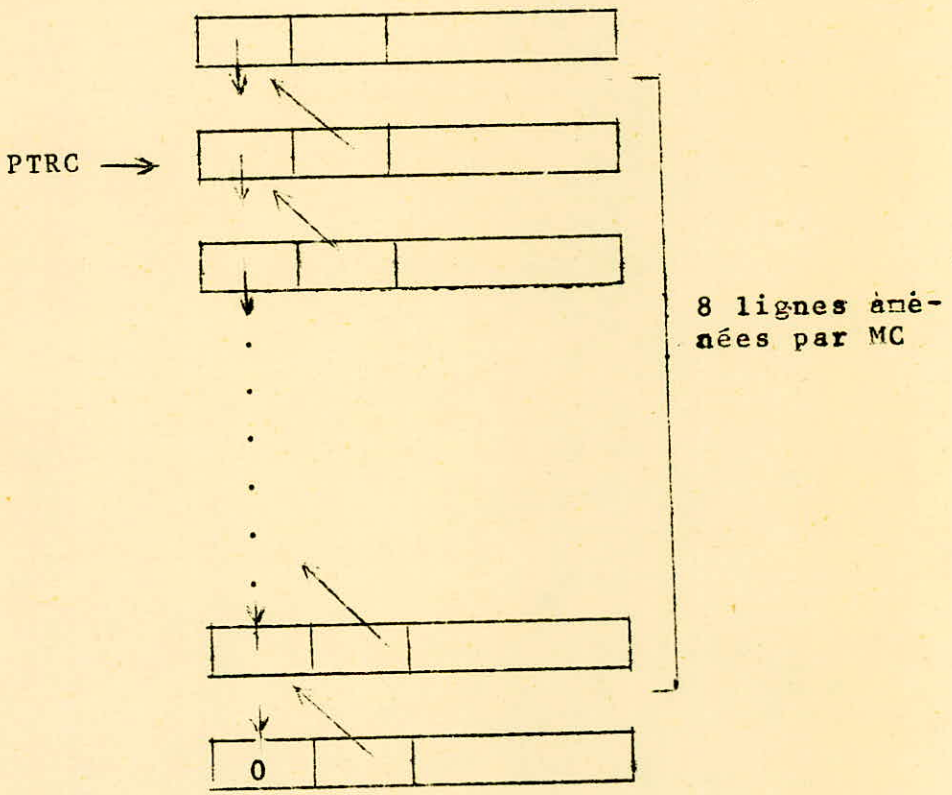
NEXT : permet de se positionner sur la ligne suivante.



Si la ligne suivante est l'élément fictif, on appelle le module MC

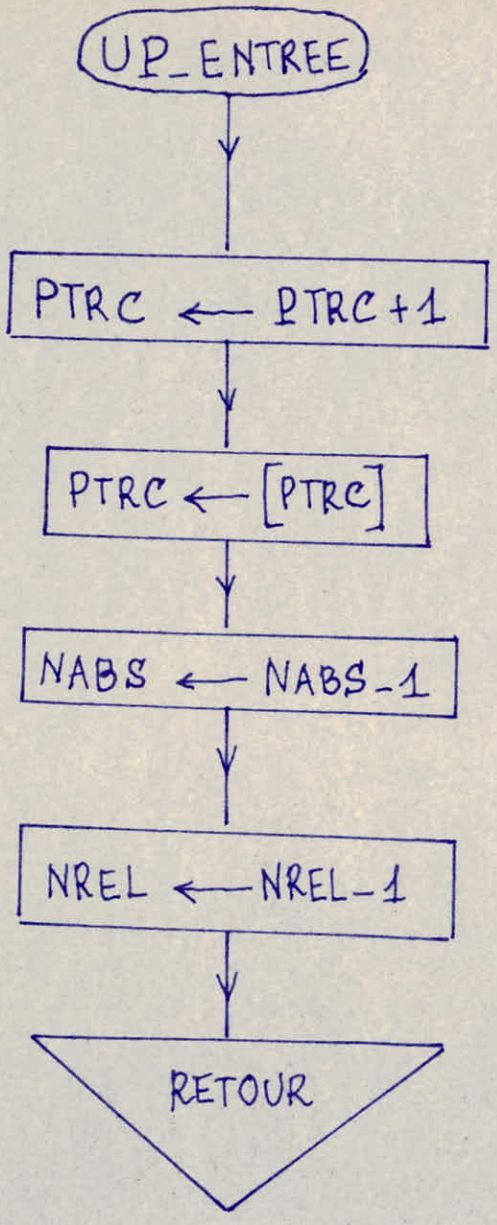


appel à MC :

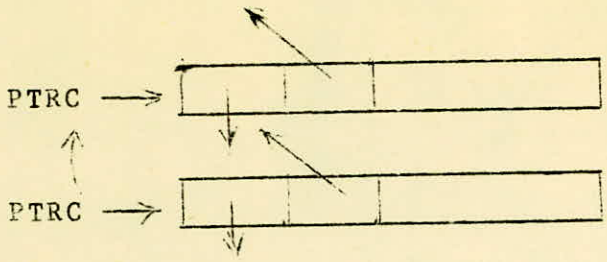


UP

organigramme



UP : permet de se positionner sur la ligne précédente.



PRINT : imprime la ligne courante sur l'imprimante du pupitre.

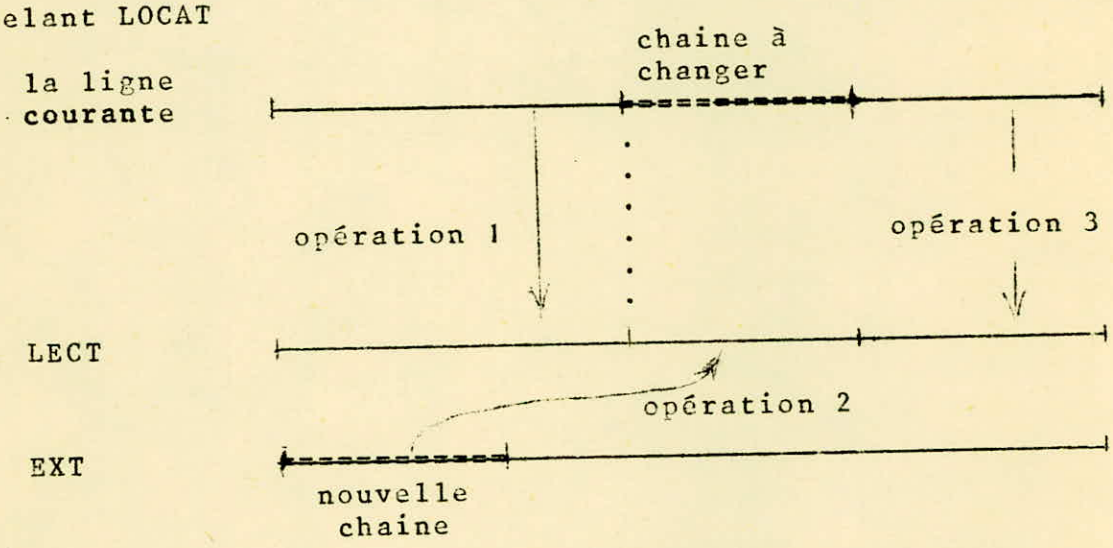
LOCAT : recherche dans la ligne courante la chaine de caractères se trouvant dans LECT.

CHANG : change dans la ligne courante la chaine se trouvant dans la zone LECT par la chaine se trouvant dans la zone EXT

Organigramme : ce schéma est à notre avis plus explicite qu'un organigramme classique.

1) localiser la chaine à changer dans la ligne courante en appelant LOCAT

2)



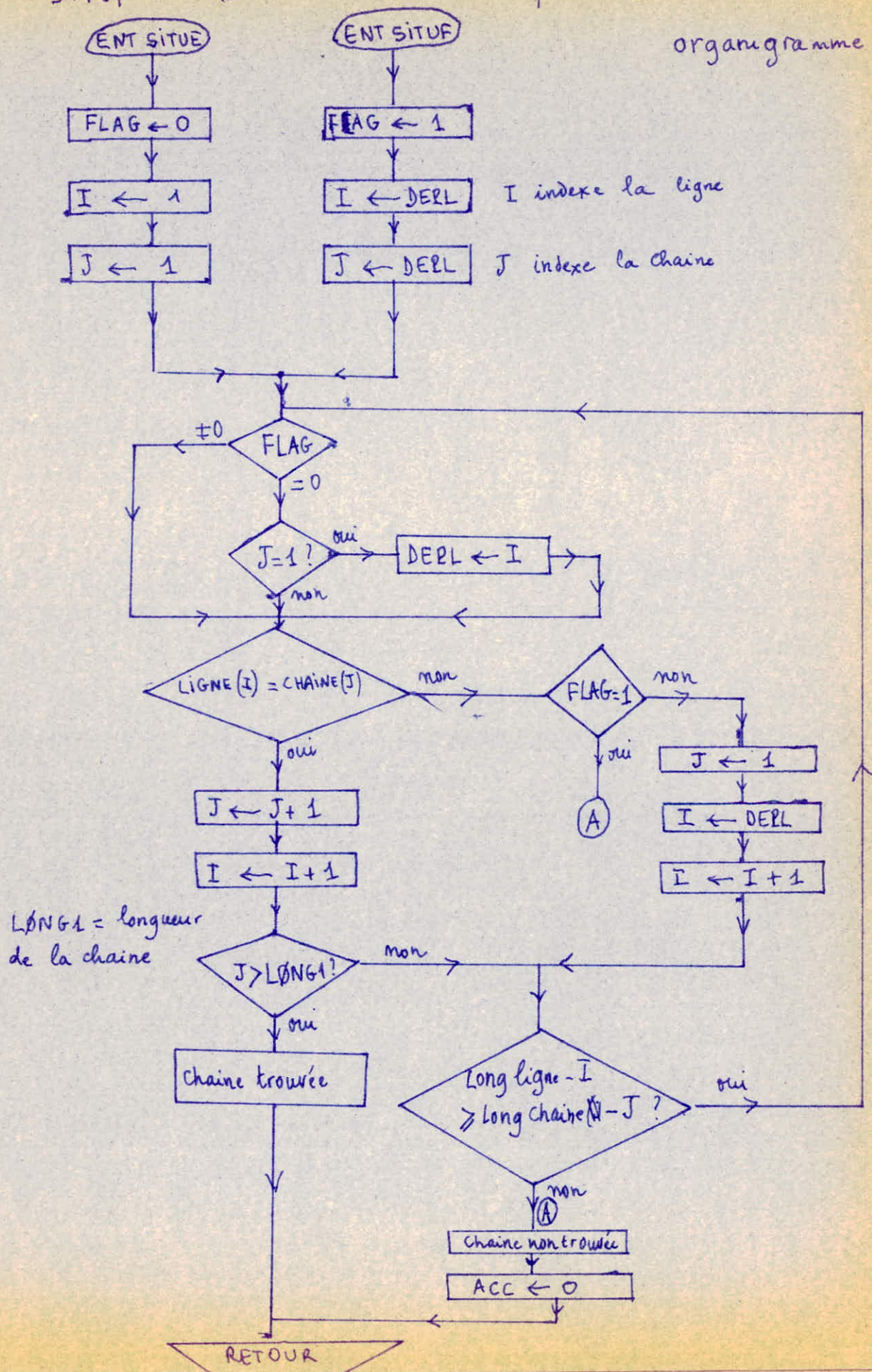
3) appel à DEL pour effacer la ligne pointée

4) insérer la nouvelle ligne ainsi constituée dans la zone LECT.

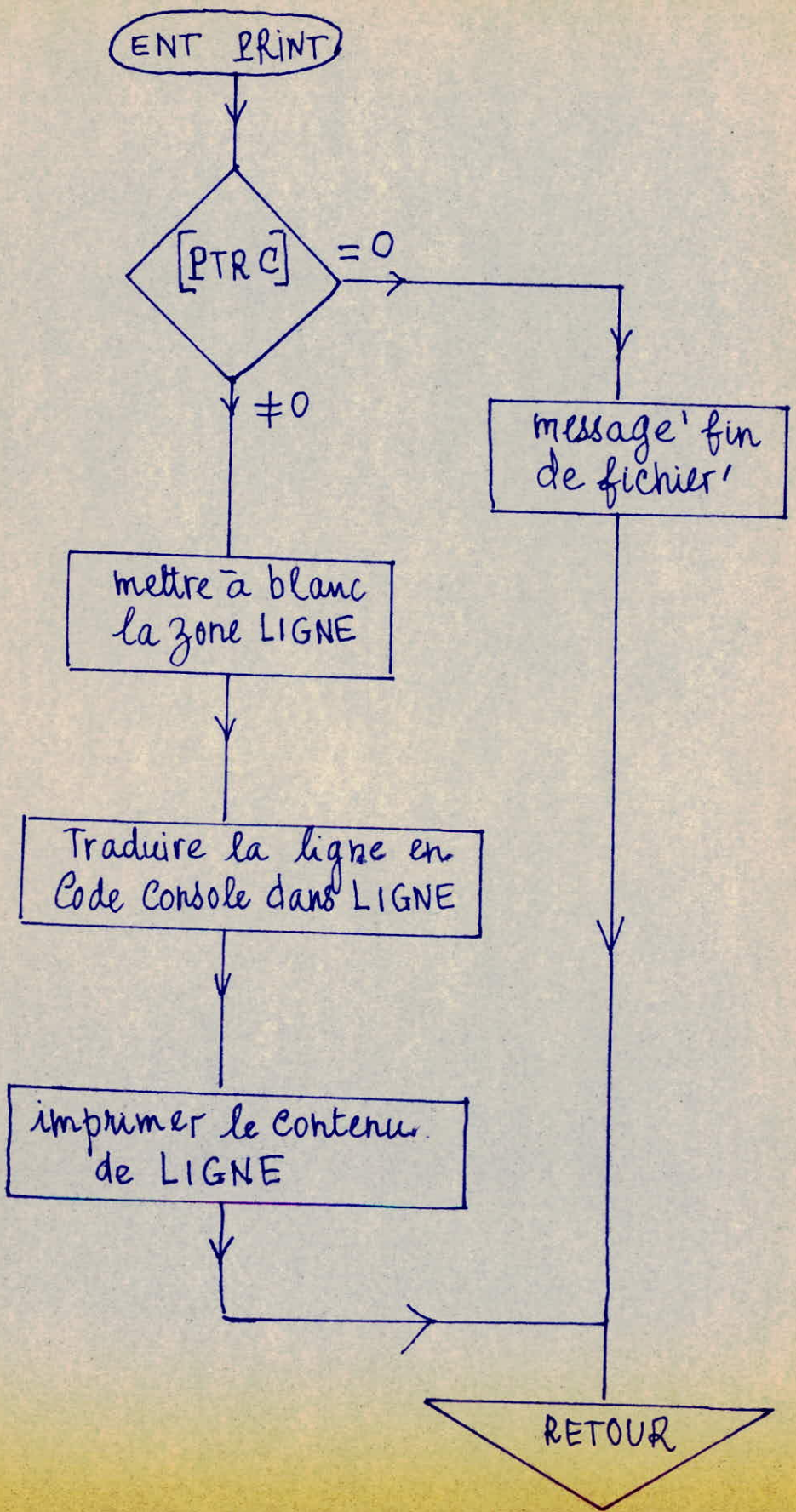
SITUE : recherche 1 chaine dans toute la ligne

SITUF : " " dans 1 champ déterminé

organigramme



PRINT
organigramme



- Interactions entre les différents modules

On a vu que les modules de l'Editeur sont imbriqués les uns dans les autres. Il est nécessaire d'éclaircir la façon dont ils se font tous appel.

- Le module MC revient à l'insertion de 8 lignes consécutivement ; donc

  MC ⟶ INSER

  INSER fait appel à DK dès qu'il y a plus de 8 lignes au dessus du pointeur courant

  INSER ⟶ DK

- NEXT appelle automatiquement "MC" dès que le pointeur courant est sur la dernière ligne en mémoire centrale

  NEXT ⟶ MC

- CHANG fait appel à LOCAT pour localiser la chaine sur la ligne même

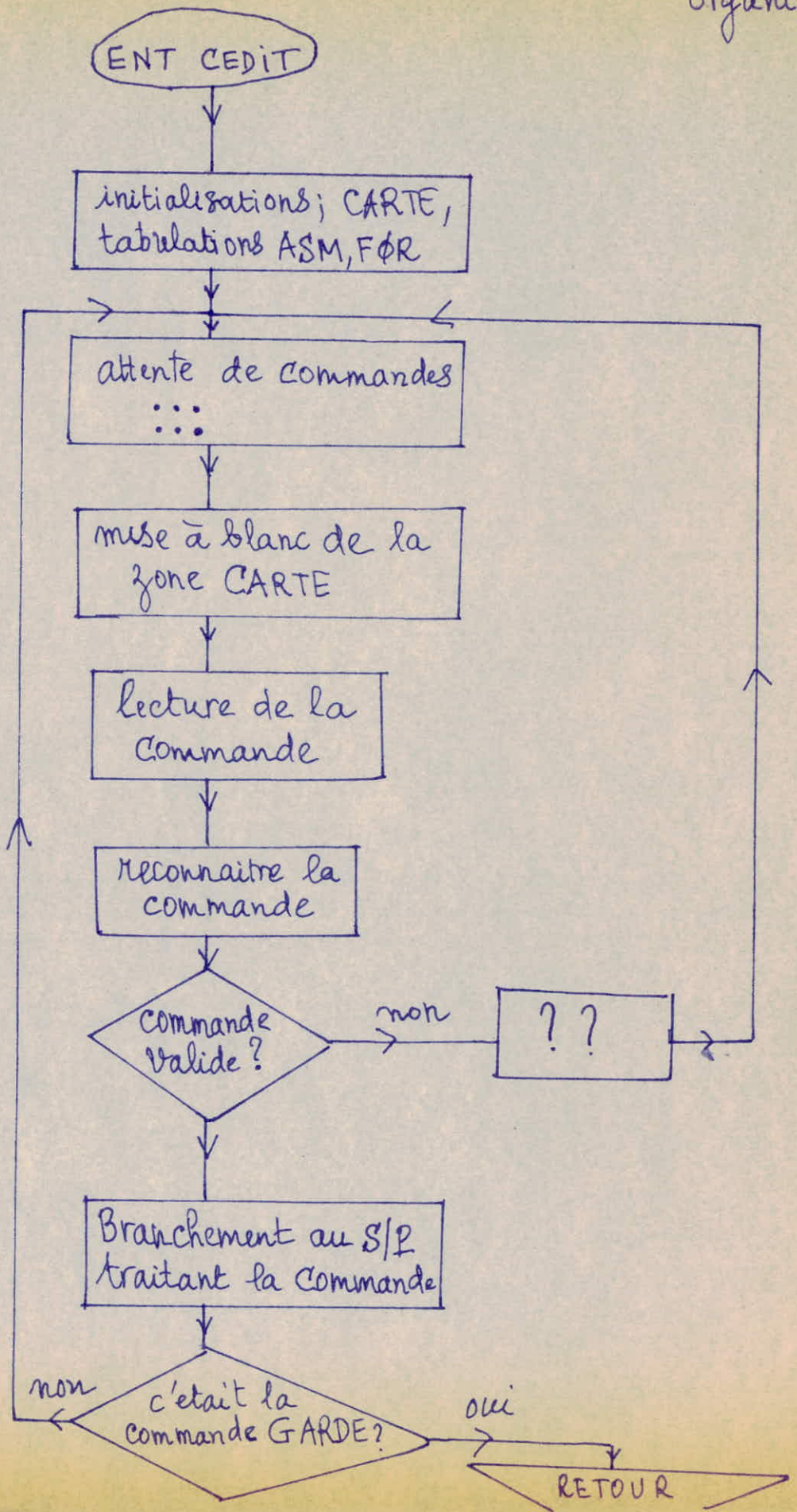
  CHANG ⟶ LOCAT

Presque tous les modules ont besoin de la gestion dynamique de la mémoire centrale.

- DK après avoir écrit sur disque les 8 première lignes appelle DEL pour les effacer en mémoire centrale.

CEDIT : Interprete de Commandes

CEDIT
organigramme



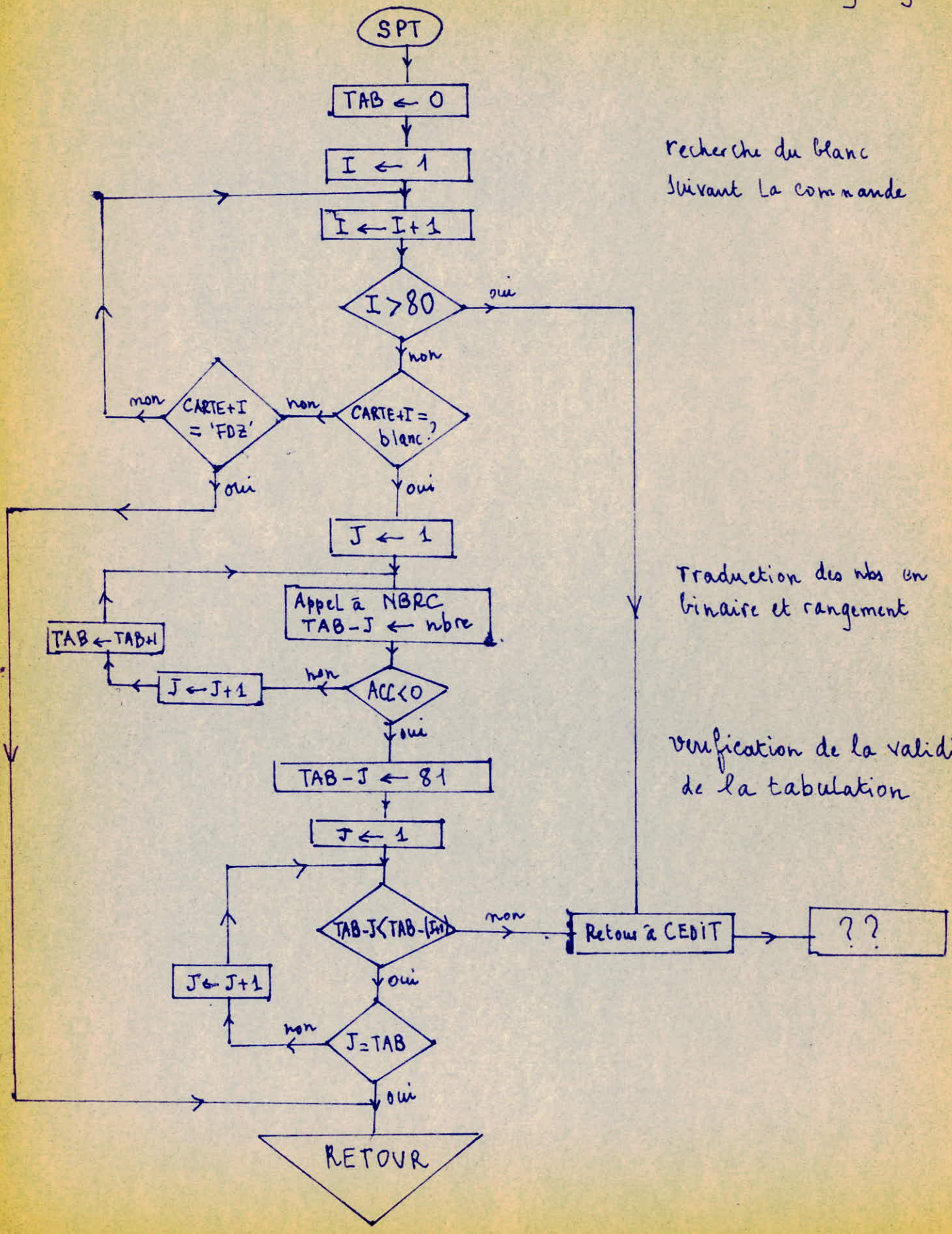
Le MODULE interprête de Commandes = CEDIT

C'est en quelque sorte le superviseur.

Il est chargé de reconnaître les commandes avant de se brancher aux sous programmes qui isolent les paramètres et d'appeler les primitives qui traitent ces commandes.

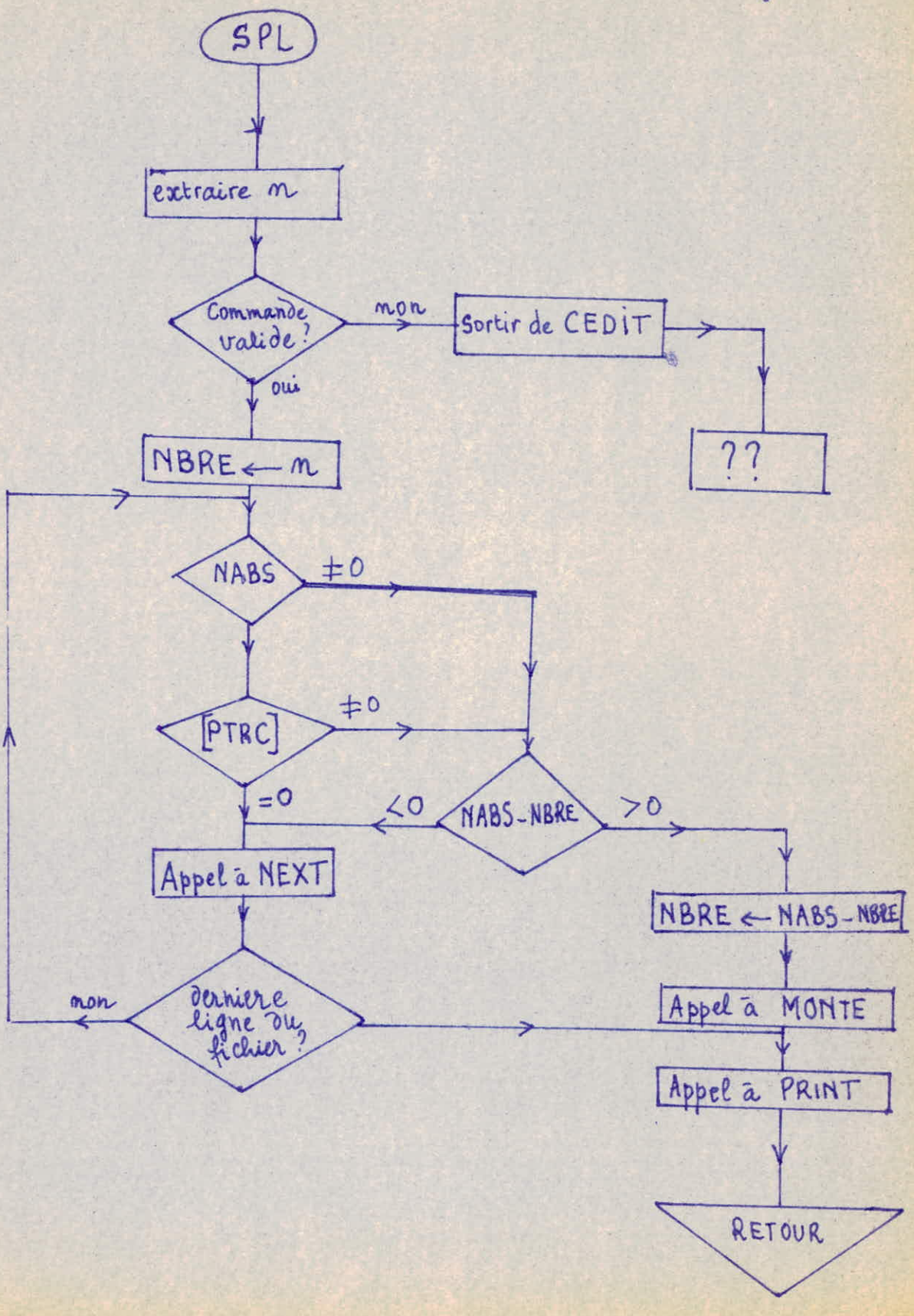
SPT : sous programme de tabulation

Organigramme



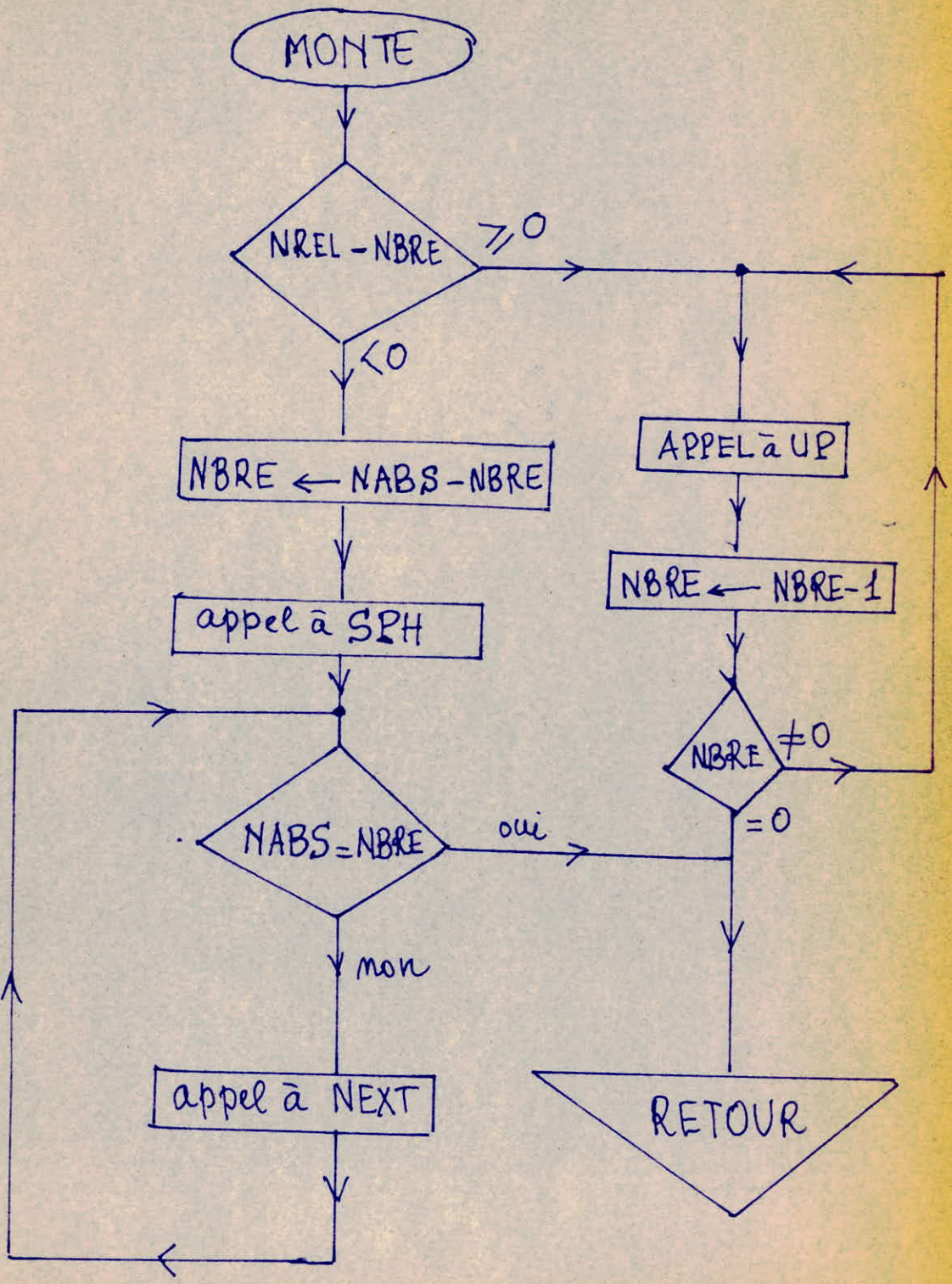
SPL : Sous programme traitant la commande : LIGNE ␣ n

organigramme



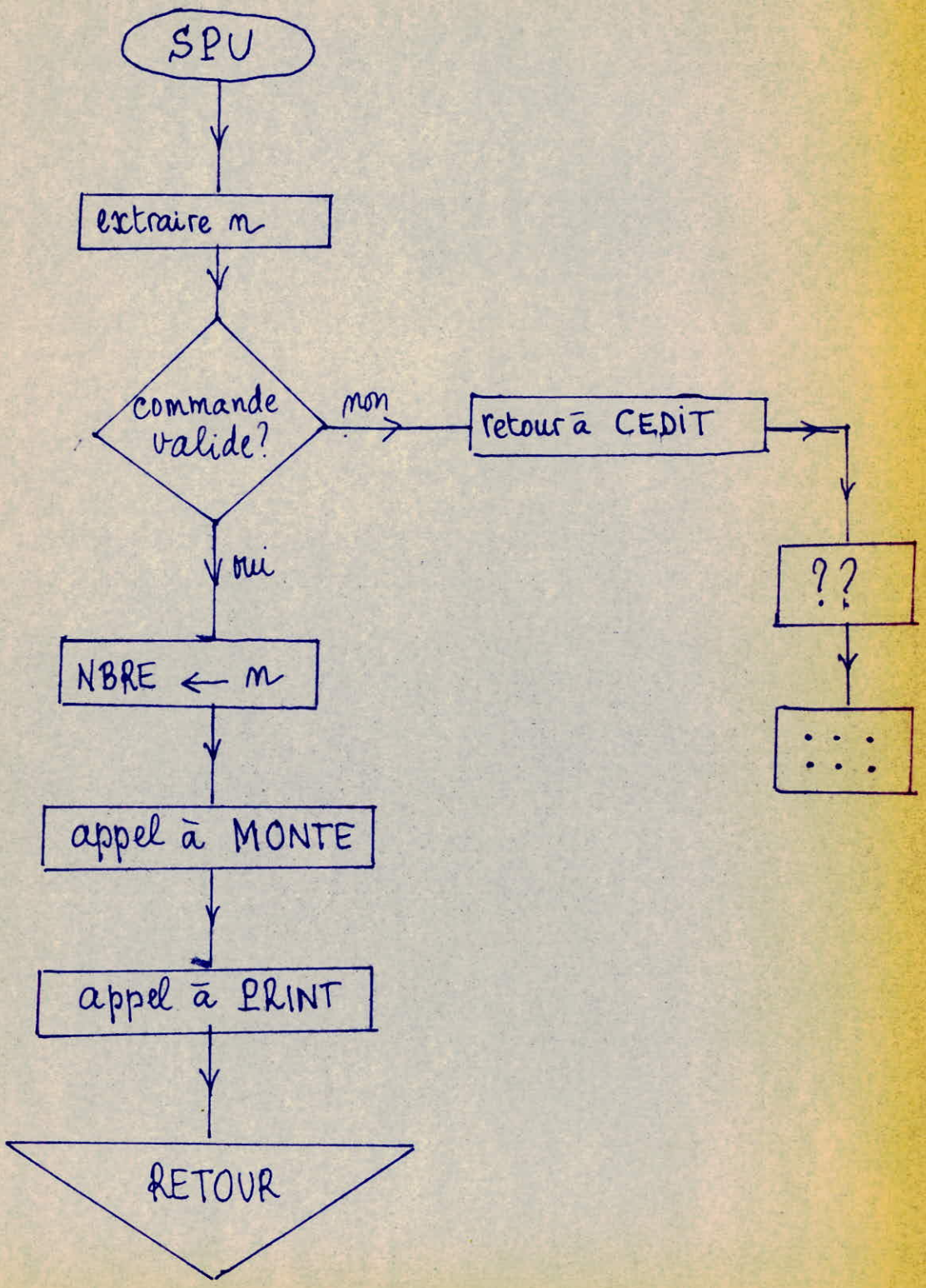
Sous programme MONTE
appelé par SPU et SPL

organigramme



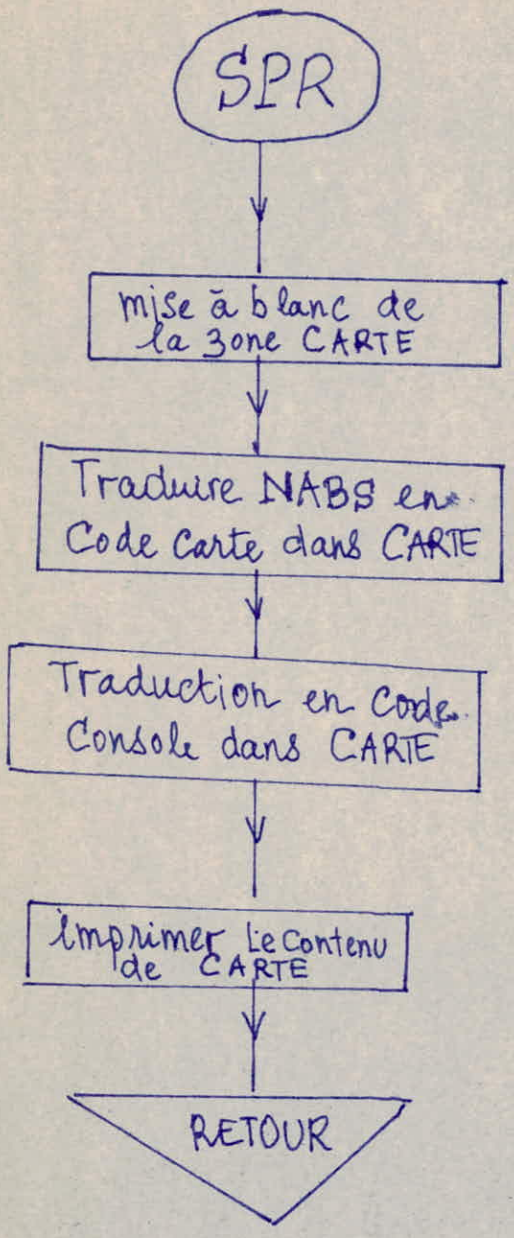
SPU : Sous programme traitant
la commande : — ↵(n)

organigramme



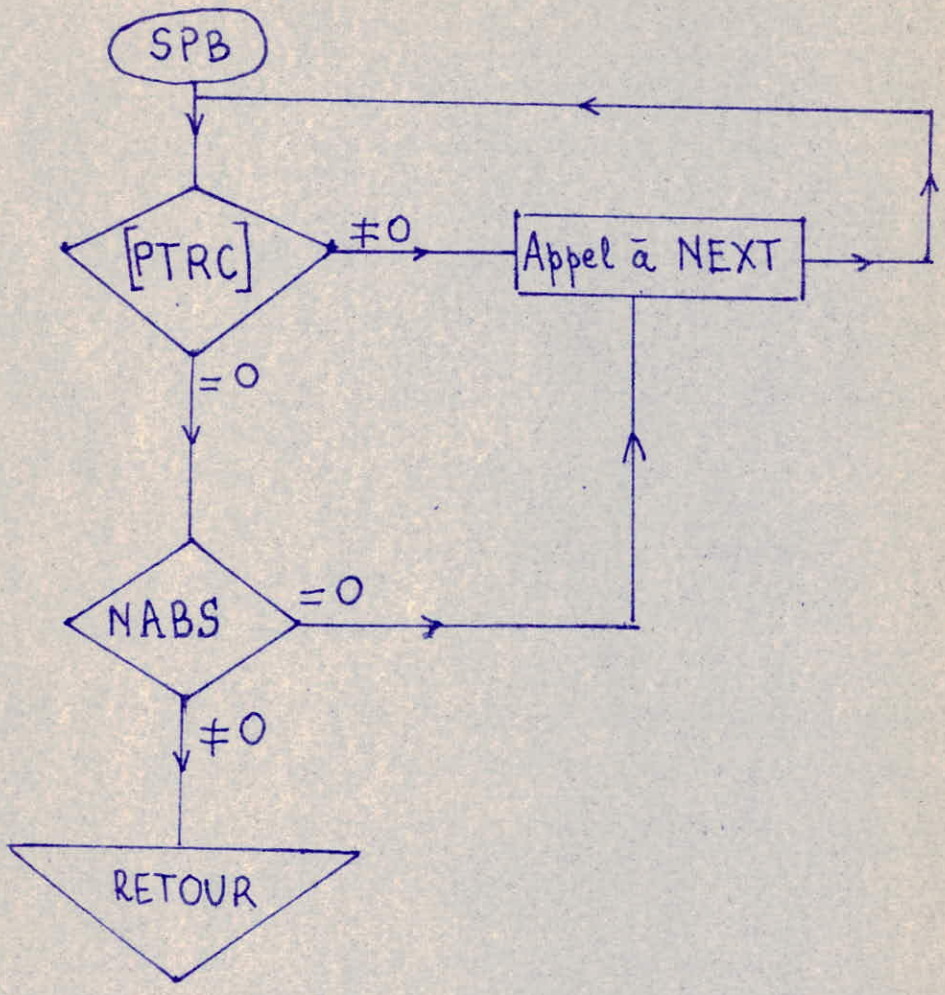
SPR : Sous programme traitant la commande : RANG

organigramme



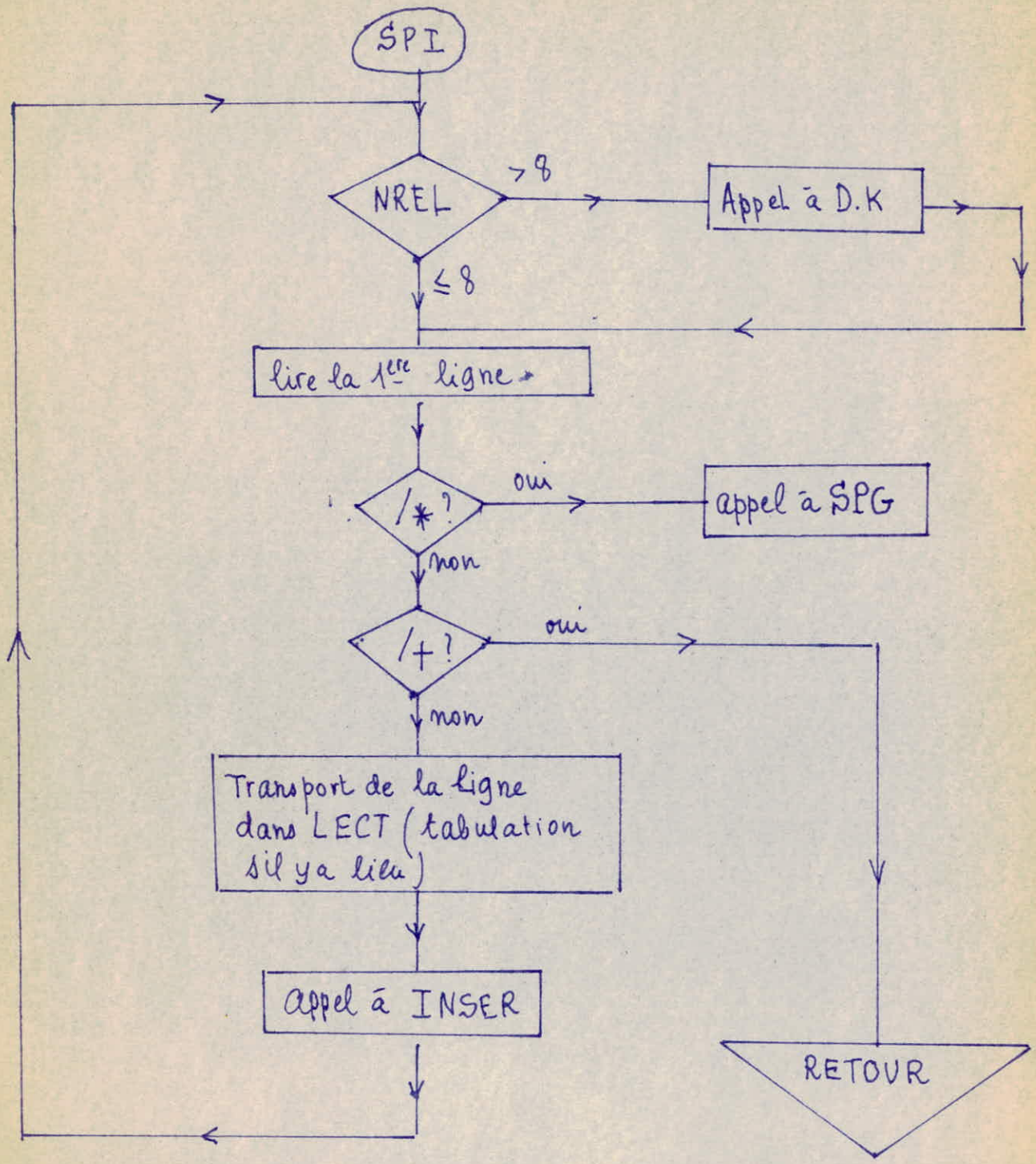
SPB : Sous programme traitant
la commande : BAS

organigramme



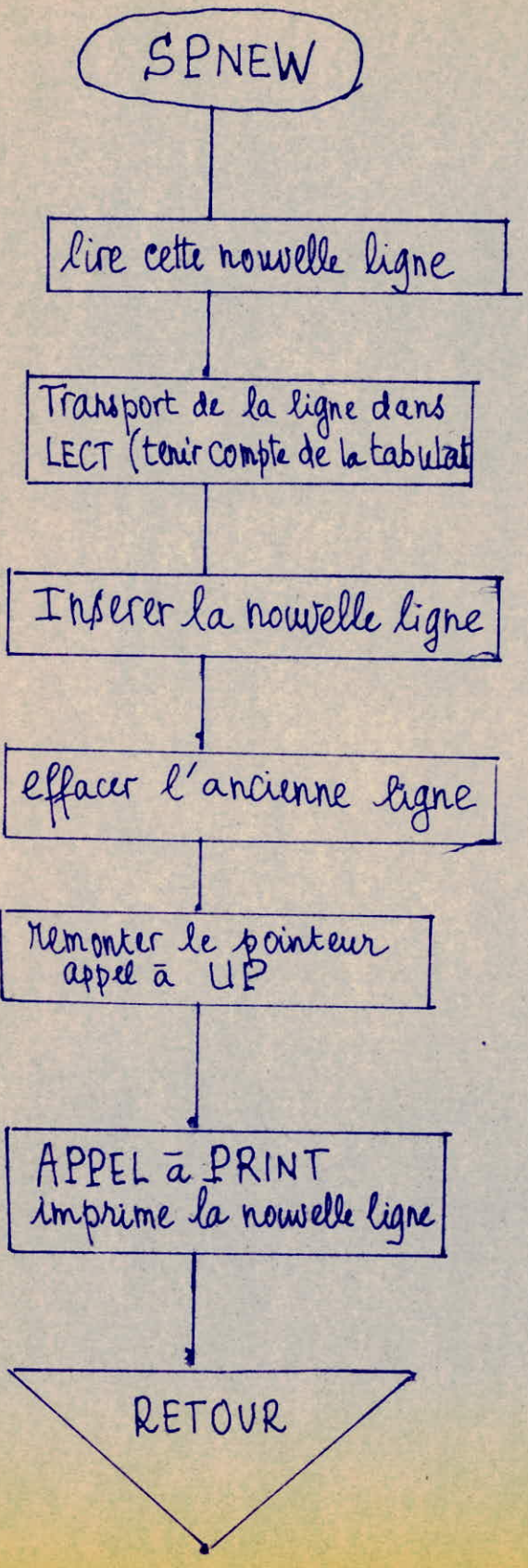
SPI : Sous programme traitant La Commande INSER

Organigramme



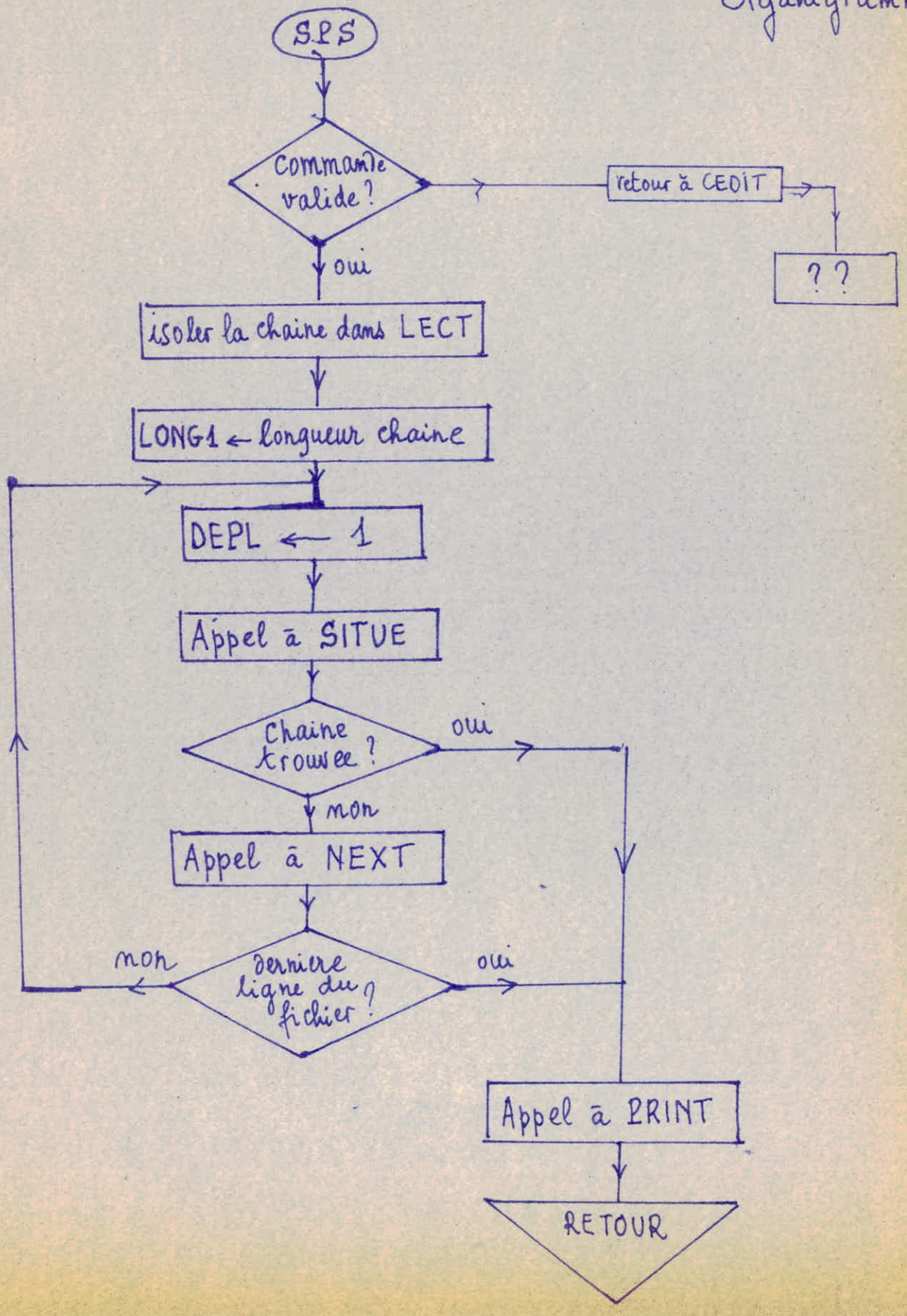
SPNEW : Sous programme traitant la Commande : NEW ligne.

organigramme



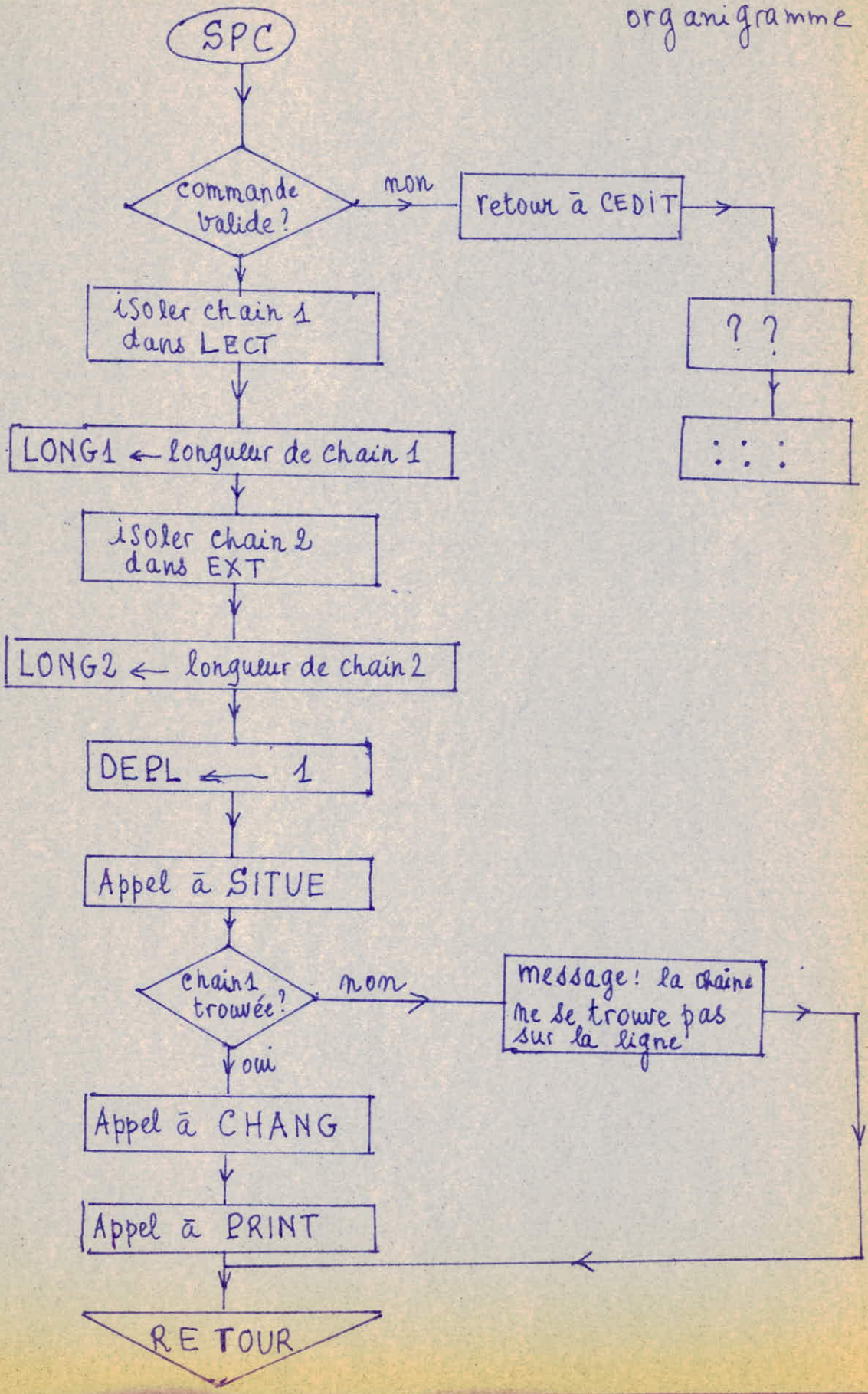
SPS : sous programme traitant
la commande : SITUE /chain/

Organigramme



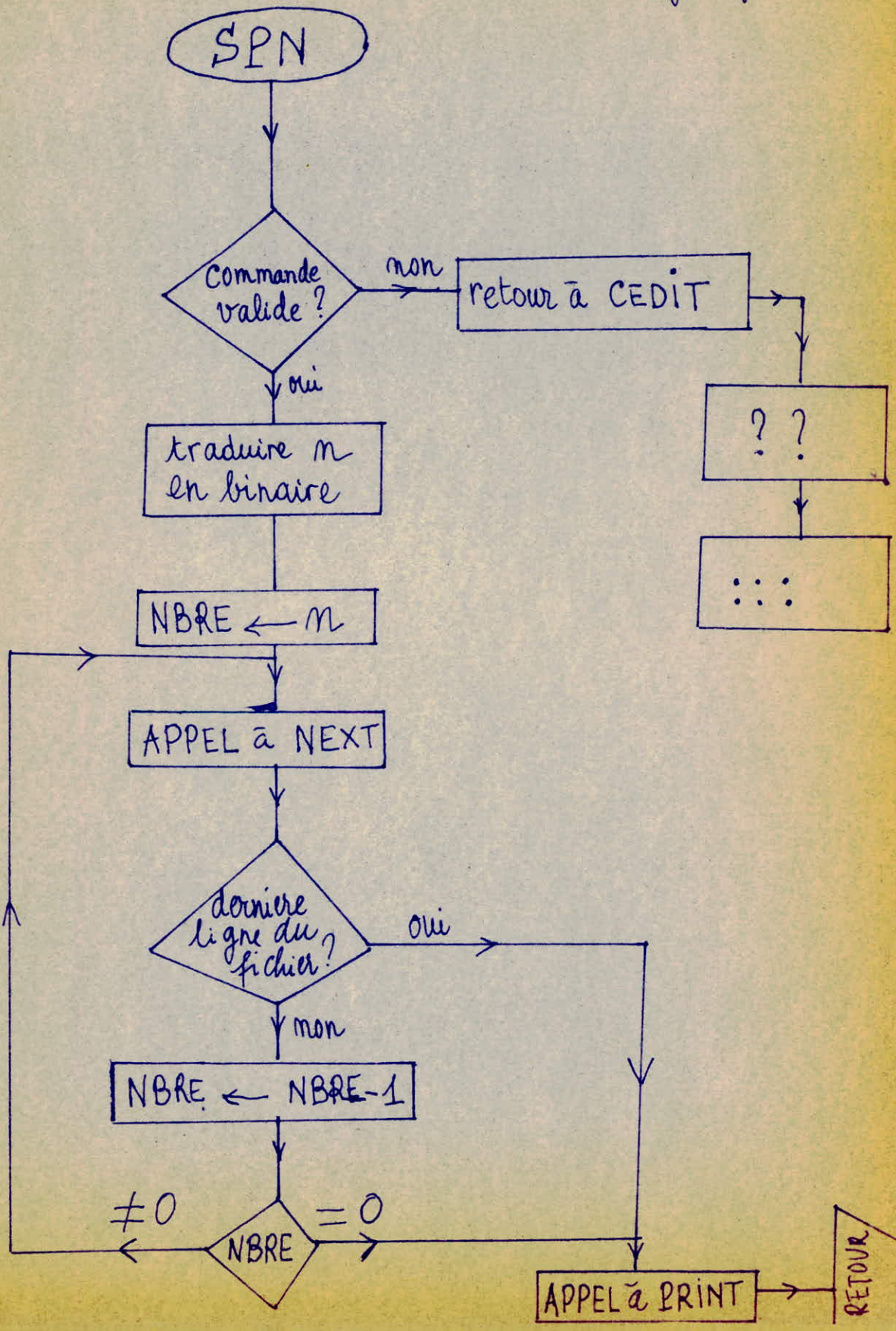
SPC : sous programme traitant
la commande : CHANGE /chain1/chain2/

organigramme



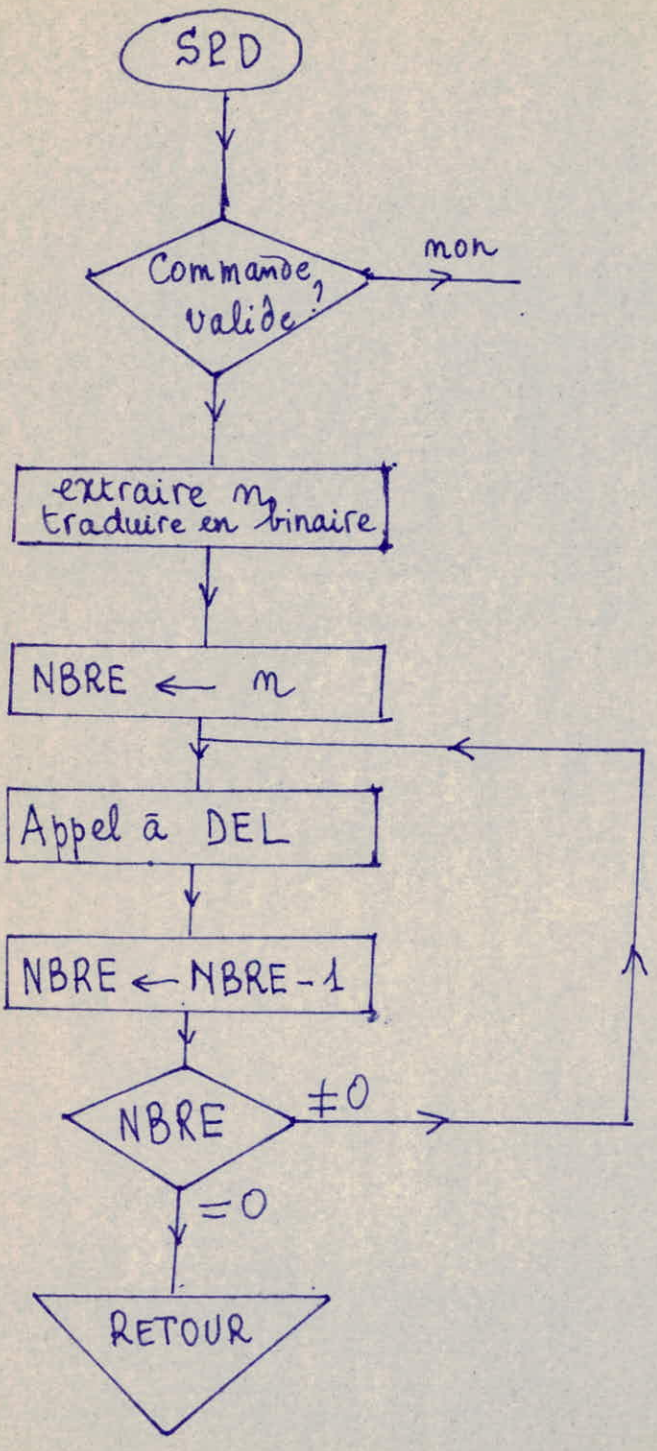
SPN : sous programme traitant la Commande : + ␣ (n)

organigramme

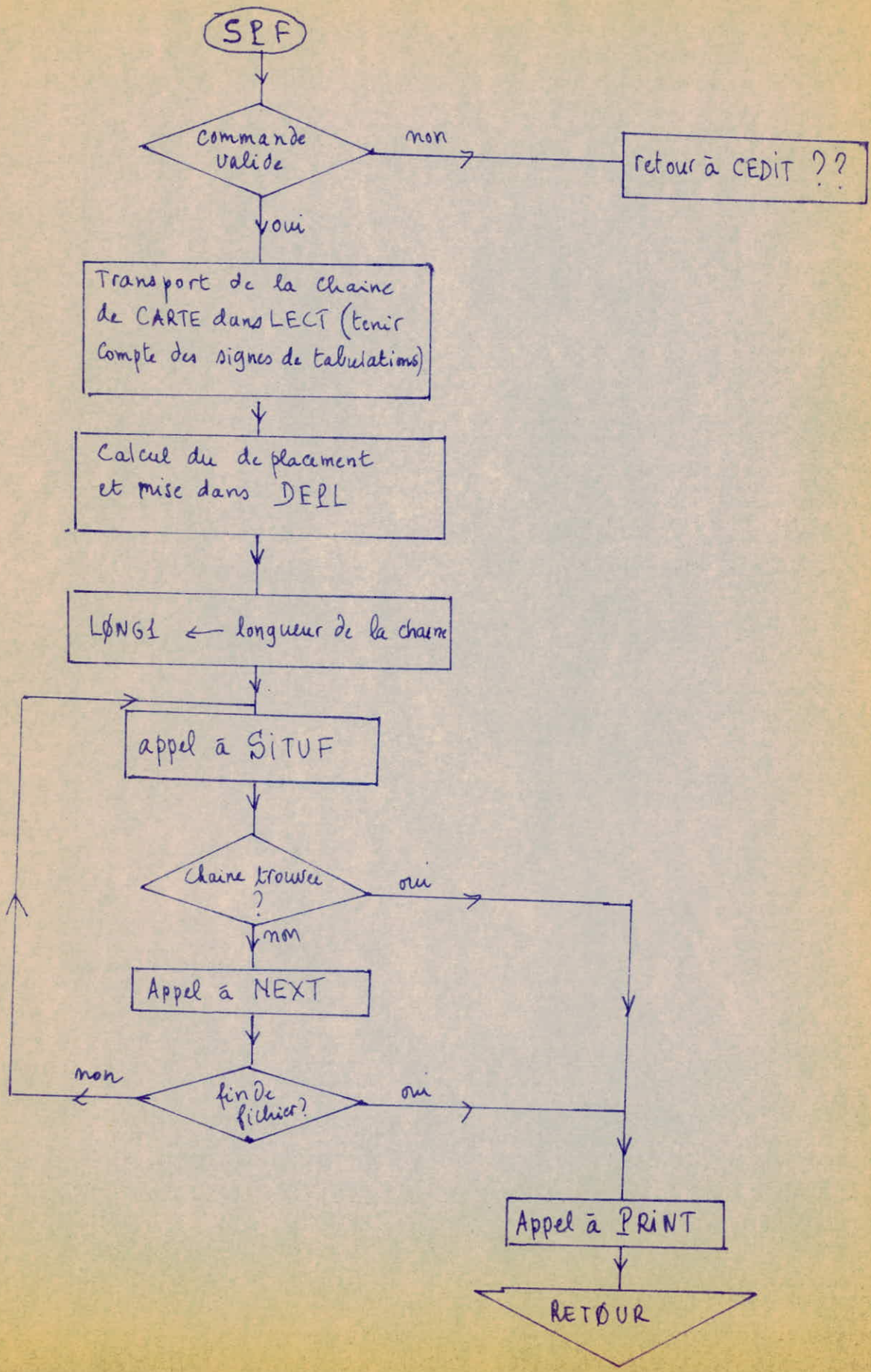


SPD : Sous programme traitant
la commande : DELETE ␣ n

Organigramme

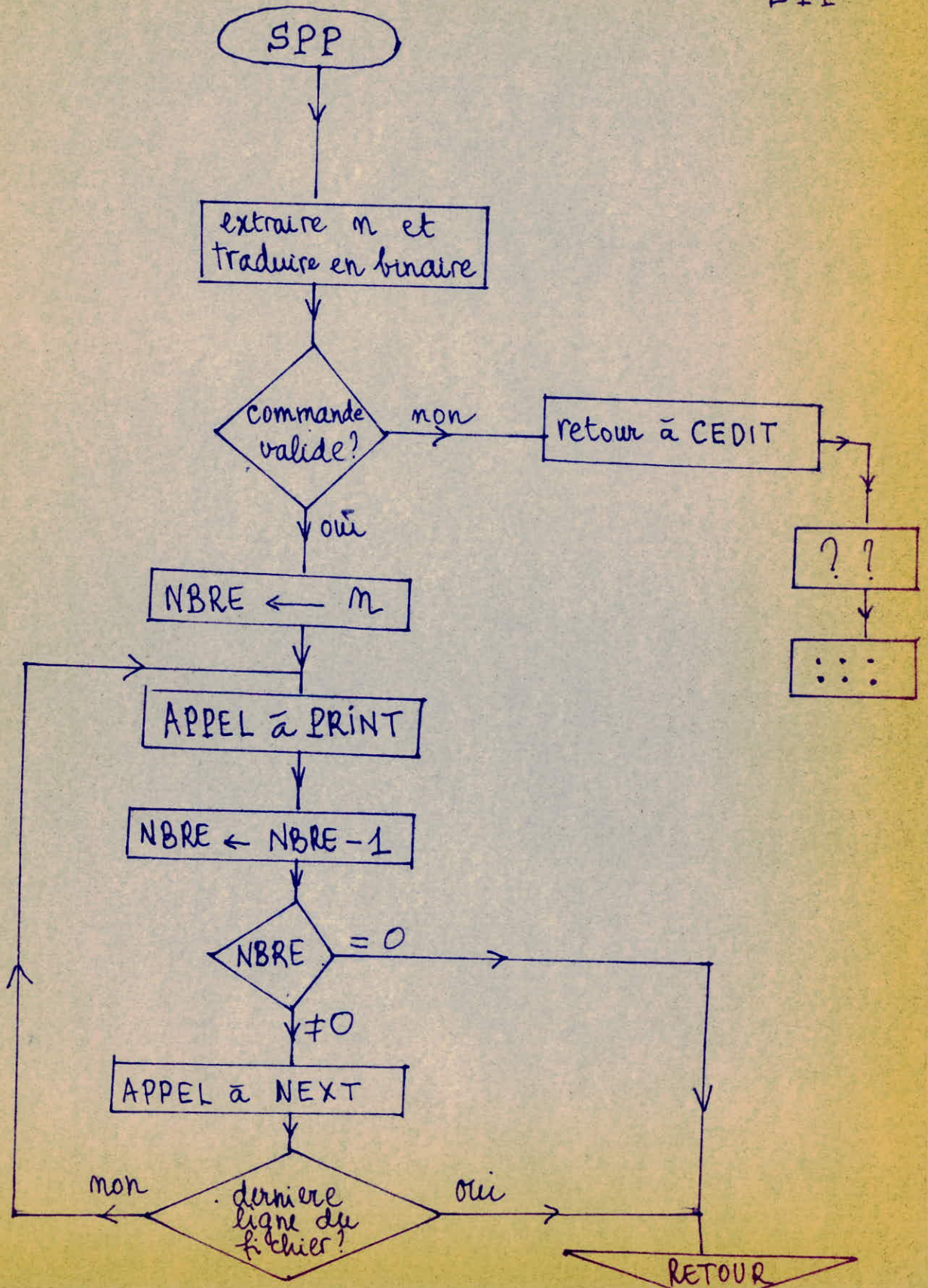


SPF : Sous programme traitant la commande FIND ␣ TITRE



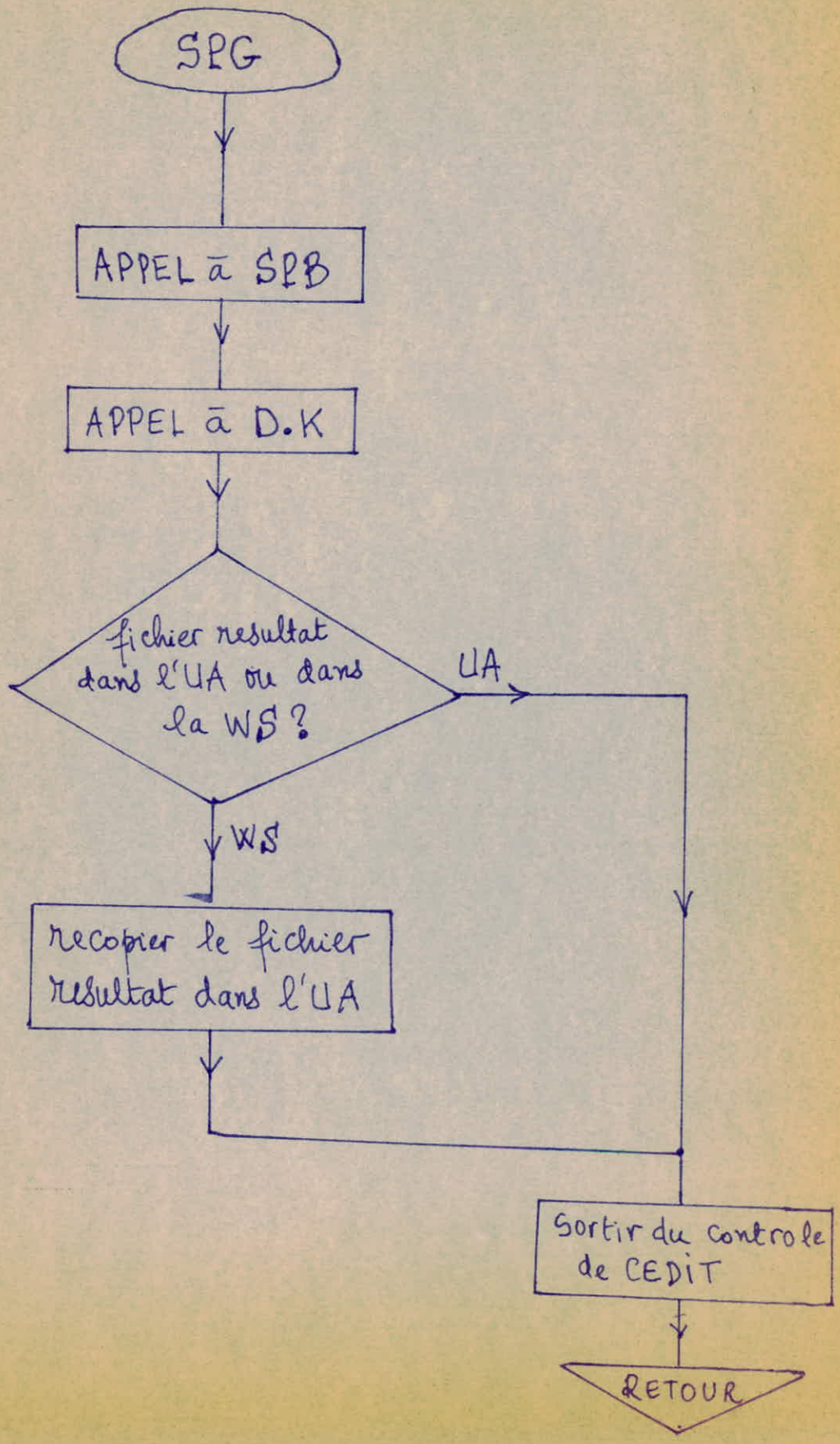
SPP : sous programme traitant la commande : PRINT␣ n

organigramme SPP



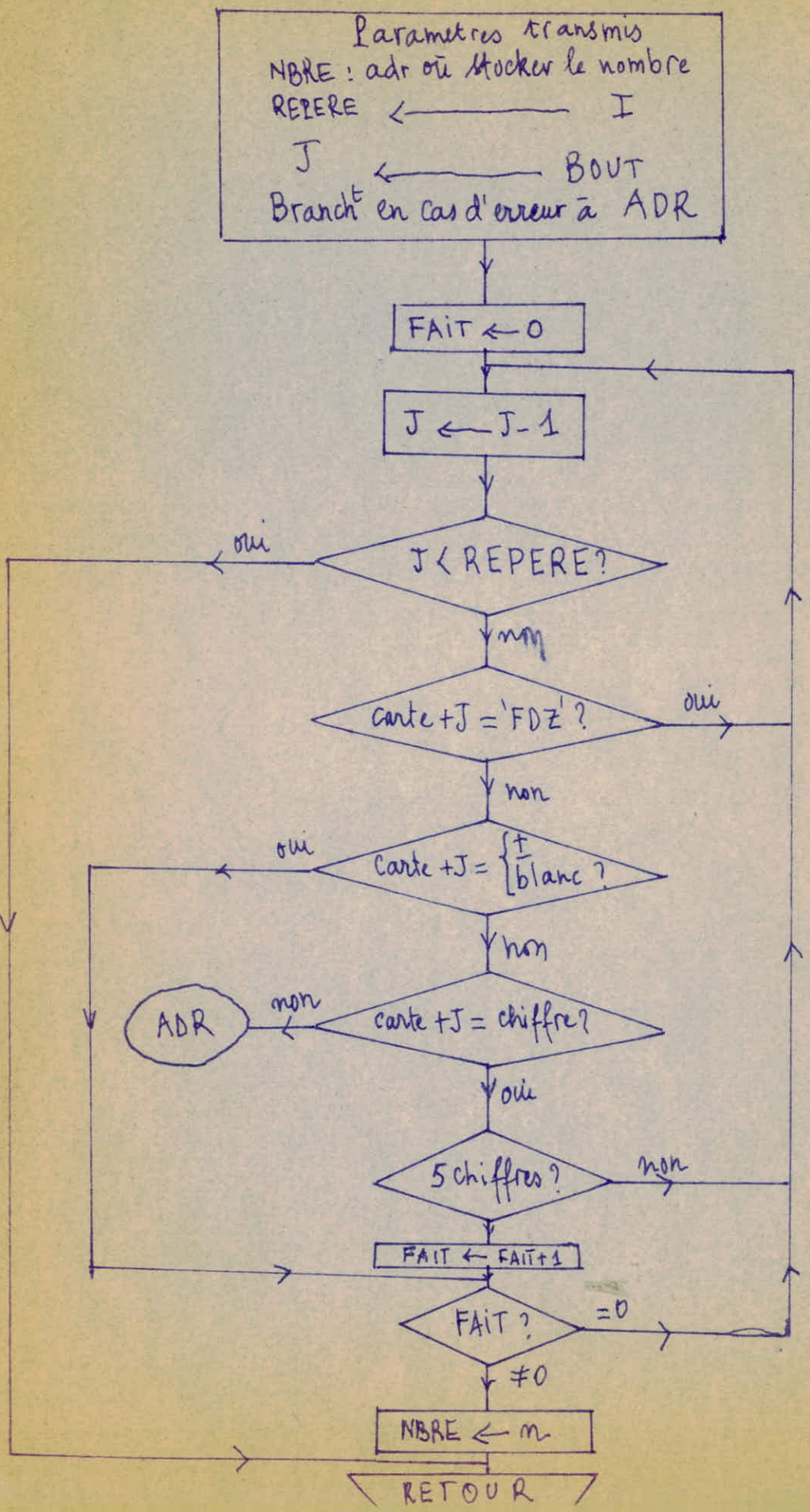
SPG : Sous programme traitant
la commande : GARDE

organigramme



NBRC : sous programme qui recherche et traite
les nombres données en parametres dans les commandes

organigramme



IV.3 CONCLUSIONS SUR L'EDITEUR

En conclusion, vu sa structure modulaire, l'Editeur 1130 peut être modifié assez simplement.

En effet, si l'on veut ajouter certaines commandes il suffit dans "CEDIT" de rajouter une commande ainsi que son sous-programme de traitement en utilisant les primitives déjà écrites ou si l'on estime insuffisantes ces primitives, d'en ajouter tout simplement.

Nous pensons toutefois que le jeu des 15 commandes permises est suffisant pour traiter les fichiers de toutes sortes.

Il faudrait plutôt à notre sens pouvoir étendre ces commandes i. e :

Par exemple, pour une commande de modification

CHANGE /CHAINE 1/CHAINE 2/ ∅ * : étoile signifiant que le changement doit être fait dans toute la ligne

et même

C/CHAINE 1/CHAINE 2/ ∅ * ∅ * : les étoiles signifiant que le changement doit être fait dans tout le fichier.

V) SOUS SYSTEME CONVERSATIONNEL - POSSIBILITES D'EXTENSION

Comme nous l'avons souligné en Introduction, le sous système conversationnel devait comprendre un certain nombre d'éléments. Ces éléments sont au nombre de six (6).

- Un module DISPATCHER (appelé CVERS) chargé d'interpréter les commandes du conversationnel et de se brancher aux modules qui les traitent :

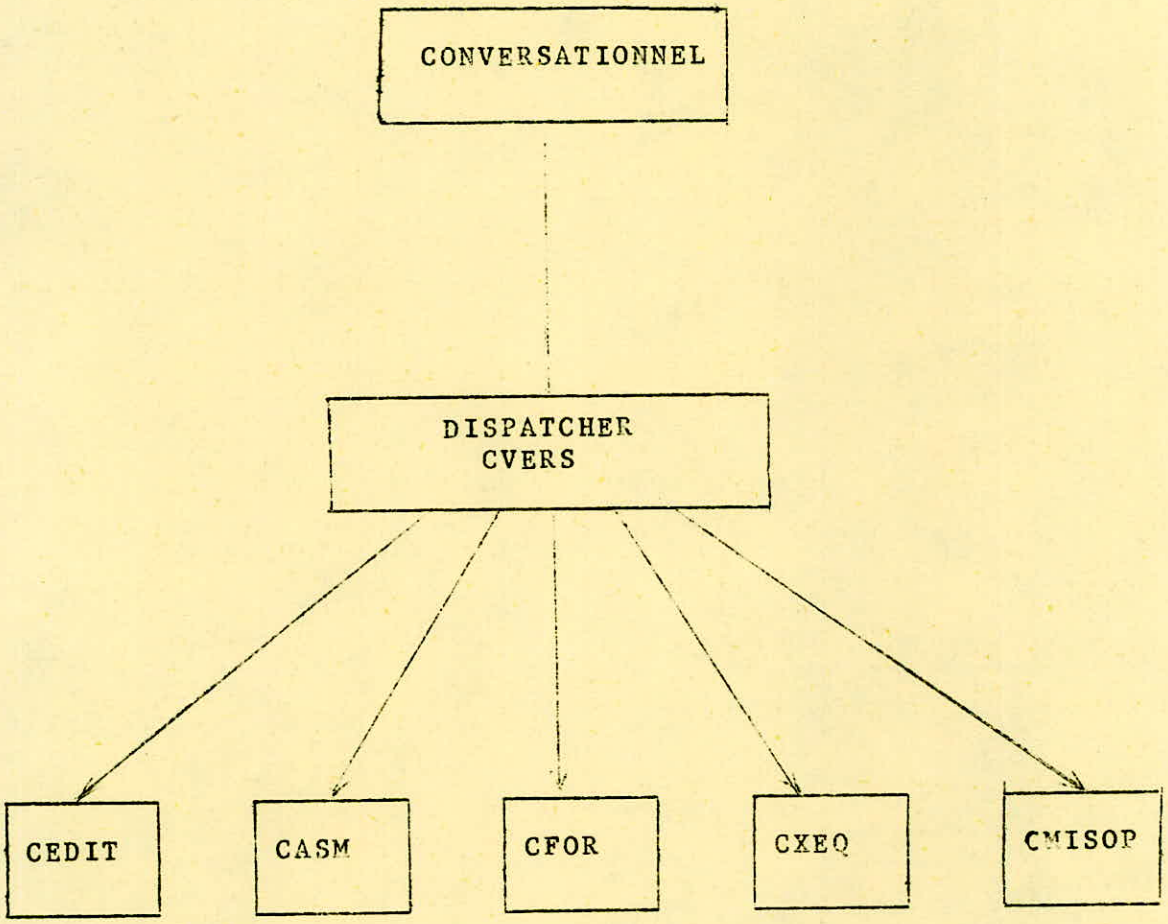
- Un éditeur module CEDIT (voir chapitre IV EDITEUR 1130)

- Un module CASM réalisant la compilation d'un programme se trouvant sur disque

- Un module CFOR réalisant la compilation d'un programme se trouvant sur disque

- Un module CXEQ réalisant le chargement et l'exécution d'un programme

- Un module MISOP qui assure le contrôle de l'exécution d'un programme (MISOP existe déjà au centre de calcul de l'E.N.P.A et il est décrit dans une note technique de M. ADIBA Alger 1972)



Les commandes de CVERS - Utilisation

EDIT ⊬ NOM (⊬ TYPE)	pour appeler CEDIT
ASM/ ⊬ NOM (⊬ TYPE)	pour appeler CASM
FOR/ ⊬ NOM (⊬ TYPE)	pour appeler CFOR
XEQ/ ⊬ NOM (⊬ TYPE)	pour appeler CXEQ
MISE ⊬ NOM (⊬ TYPE)	pour appeler CMISOP

* Module CEDIT :

(voir chapitre IV EDITEUR 1130)

* Module CASM :

Ce module n'a pu être réalisé pour deux raisons :

d'une part, nous ne disposions pas du programme source de l'Assembleur,

et d'autre part, la réalisation de l'Editeur proprement dit ne nous a pas laissé suffisamment de temps pour d'autres extensions possibles.

Néanmoins voici schématiquement la marche que nous aurions suivi pour le réaliser, si toutes les conditions favorables étaient réunies.

. Enoncé du problème :

L'Assembleur 1130 possède un mot de communication avec le Superviseur qui lui indique si l'unité d'entrée du programme source est le lecteur ou le clavier de la machine à écrire (ou le lecteur de bandes perforées), mais ne prévoit rien quand l'unité d'entrée est le disque.

(En conversationnel, tous les programmes sont stockés sur disque).

Assemblage d'un fichier carte

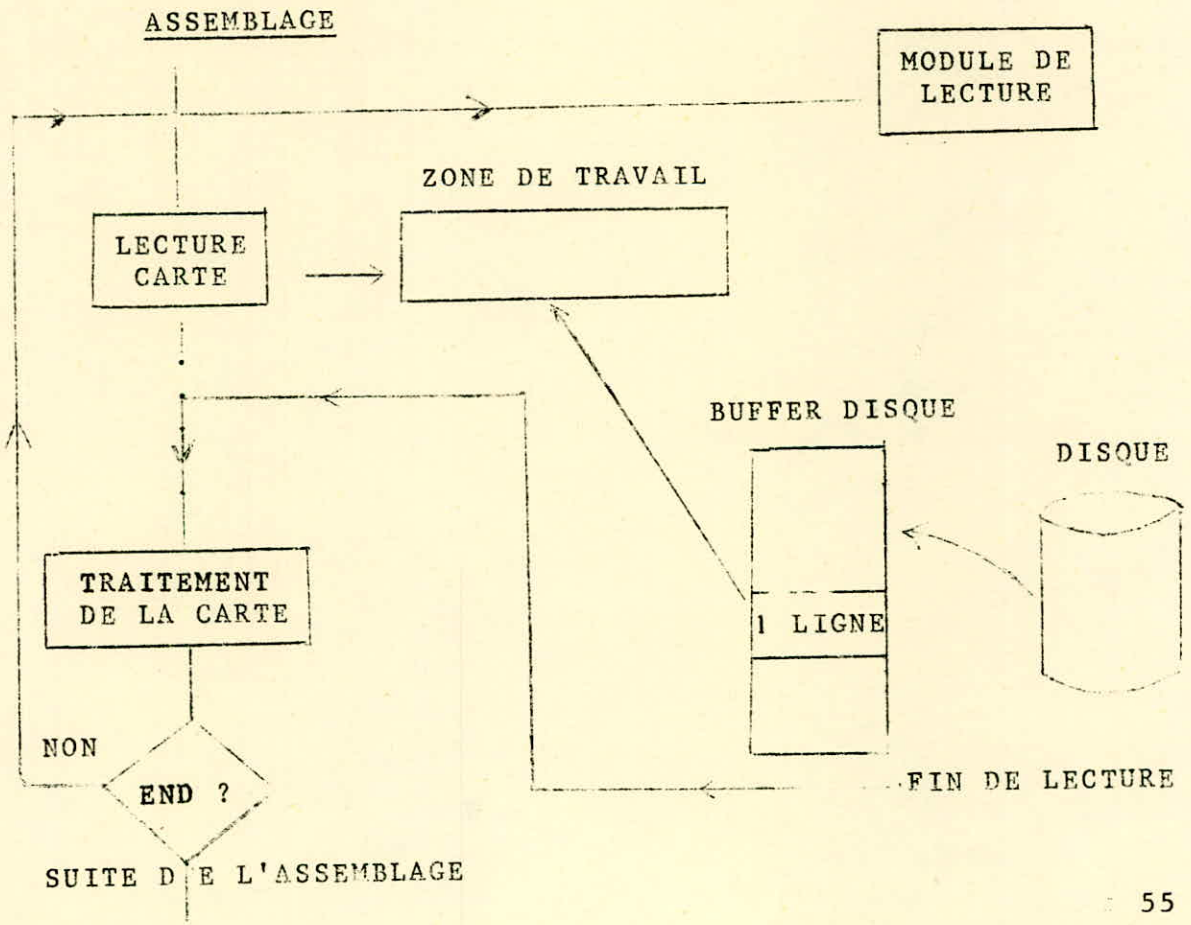
L'Assembleur réalise l'assemblage d'un programme en deux passages.:

au premier passage il lit tout le programme carte par carte et assemble déjà une partie des mots relative aux champs code d'instruction format indirection, etc... et dresse la table des symboles.

au second passage il réalise complètement l'assemblage.

La solution consiste à simuler par programme une lecture par carte à l'Assembleur. Pour cela, il faut être en possession du Programme source de l'assembleur et repérer la séquence du programme qui lit une carte.

Il faut alors écrire un module de lecture qui lise sur le disque secteur par secteur et fasse défiler chaque ligne dans la zone de travail de l'Assembleur.



+ Module CFOR :

Le problème est sensiblement le même que pour l'Assembleur mais nous disposons d'encore moins d'information sur le Compilateur.

+ Module CXEQ :

C'est le travail réalisable le plus rapidement.

Il consiste à transmettre au Chargeur image mémoire (CIL), qui sert de superviseur intermédiaire, le nom du programme à faire exécuter.

+ Module MISOP :

C'est un module de synthèse qui appelerait

- l'Editeur (CEDIT) pour transformer le fichier MAIN
- l'Assembleur (CASM) pour assembler le fichier MAIN
- le Chargeur (CXEQ) pour faire exécuter le fichier MISOP. (cf : MISOP note technique de M^{r} M. ADIBA).

VI) CONCLUSION

En résumé, le bilan des réalisations pratiques se trouve limité à la mise en place d'un Editeur, qui s'il n'est pas intégré dans un système conversationnel complet, présente un intérêt moindre. Il permet toutefois de mettre au point les programmes sources et de réaliser la mise à jour automatique de fichiers.

Afin que notre travail soit utilisable, deux petits modules accompagnent l'Editeur :

* un Imprimeur

WRITE : imprime les lignes d'un fichier, se trouvant sur disque, sur l'imprimante rapide avec les numéros correspondants en vue de faciliter la tâche à l'utilisateur de CEDIT grâce à la commande :

WRIT ⊬ NOM (⊬ TYPE)

* un "Puncheur"

PUNCH : perfore sur cartes un fichier se trouvant sur disque. Ainsi un programme mis au point sous CEDIT peut être perforé grâce à la commande :

PUNC ⊬ NOM (⊬ TYPE)

Une fois perforé on peut facilement demander à l'assembler ou le compiler et l'exécuter sans le secours du conversationnel.

Cette solution (banale) n'est que provisoire dans le temps puisque d'autres viendront améliorer et compléter notre étude.

OUVRAGES CONSULTES

EDITEURS :

+ Editeurs par contexte pour systèmes conversationnels à partage de temps

Thèse de M. ADIBA - avril 1971

+ Introduction aux systèmes CP67 et CMS

par A. AUROUX et C. HANS
Centre Scientifique IBM de Grenoble

SYSTEMES :

+ Système moniteur disque 1130 version 2
Guide du programmeur et de l'opérateur
Programmes 1130-OS-005 et 1130-OS-006

Sujet 1130-36 Référence CF2-0044-0

+ Système d'exploitation des ordinateurs
Cours à l'Institut de Programmation - Grenoble

par O. LECARME

+ Cours à l'Ecole Nationale Polytechnique

de M. ADIBA - Année 1973