MASTER EN SCIENCES INFORMATIQUES

Configuration pluri-processus réalisation d'une liaison GE58-VARIAN par l'intérmédaire d'un contrôleur disque.

Orban de Xivry, D.

Award date:
1979

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES
NOTRE-DAME DE LA PAIX

INSTITUT D'INFORMATIQUE

CONFIGURATIONS PLURI — PROCESSEURS. REALISATION D'UNE LIAISON GE58-VARIAN PAR L'INTERMEDIAIRE D'UN CONTROLEUR DISQUE.

ORBAN DE XIVRY D.

Mémoire présenté en vue de l'obtention du grade de licencié et maître en Informatique.

A Marie

Au seuil de ce mémoire, nous tenons à remercier toutes les personnes qui nous ont permis de mener ce travail à bonne fin et plus particulièrement :

Messieurs J. Brunin et J. Ramaekers, qui par leurs conseils ont su donner une orientation enrichissante à ce travail ;

Messieurs J. Demarteau et L. Jacques de l'Institut d'Informatique, qui ont collaboré à la réalisation pratique de ce mémoire ;

Messieurs les professeurs de l'Institut d'Informatique de Namur qui nous ont donné une formation de base permettant d'aborder ce travail dans les meilleures conditions.

TABLE DES MATIERES

PREMIERE PARTIE

DEUXIEME PARTIE

II. PARTAGE DE L'ESPACE DISQUE

+

+ +

PREMIERE PARTIE

I. LES SYSTEMES PLURI-PROCESSEURS

Les systèmes informatiques dans lesquels travaillent plusieurs processeurs commencent à être relativement répandus. Cependant les termes utilisés dans ce domaine restent souvent assez vagues : systèmes multi-processeurs, systèmes multi-calculateurs etc.... Il est vrai que de nombreuses configurations sont possibles et que les différents articles écrit à ce sujet ne précisent pas suffisamment les termes.(1) Dans ce travail nous parlerons de systèmes pluri-processeurs pour désigner tout système possèdant plus d'un processeur, les autres termes étant réservés pour des configurations bien précises qui seront définies ultérieurement.

Pour préciser les termes nous allons essayer de répartir les systèmes pluri-processeurs en différentes classes représentatives des configurations existantes.

Cette classification des systèmes pluri-processeurs peut se faire suivant deux points de vue :

- un point de vue fonctionnel (Flyn 66)
- un point de vue topologique (Ensl 74)

C'est ce dernier qui a été choisi dans ce travail car il permet une classification plus fine. Topologiquement on peut distinguer trois classes principales :

- les systèmes reliés très indirectement
- les systèmes reliés indirectement
- les systèmes reliés directement.

(1) Les termes employés dans cette partie sont pour la plupart issus de Enslow (ENSL 74)

1. SYSTEMES RELIES TRES INDIRECTEMENT

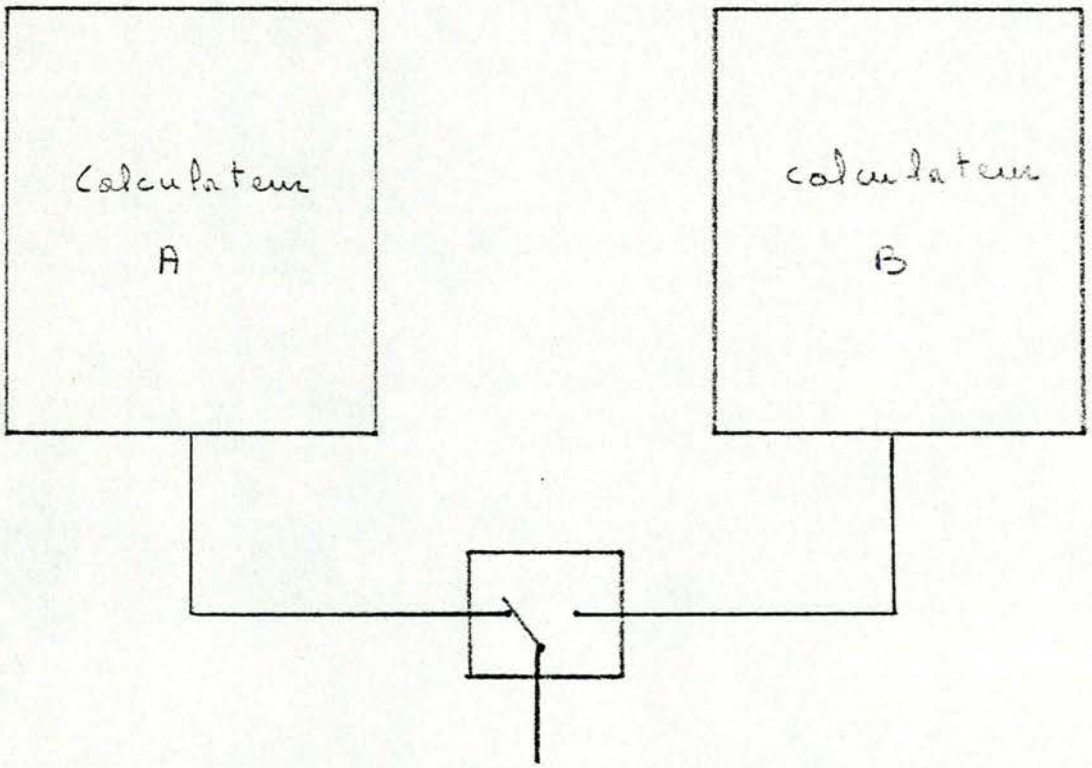
Historiquement, ce sont les premiers et ils ont été développés pour des raisons de fiabilité.

Ils sont obtenus par juxtaposition de calculateurs complets, identiques, capables d'assurer seuls la totalité des travaux (Fig I). Leur seul point commun se situe à l'arrivée des informations. Cela signifie en particulier que chaque calculateur possède ses propres périphériques, ses propres programmes et traite ses propres données.

Au niveau de la réalisation il existe deux possibilités principales :

La première consiste à mettre un interrupteur au point de jonction. Dans ce cas, les informations arrivant ne sont dirigées que vers un calculateur maître. Celui-ci effectuera le traitement et fournira la réponse ; l'autre calculateur pendant ce temps ne fait rien. Il ne servira que lorsque le premier tombera en panne. Dans ce cas là, l'interrupteur sera levé soit de façon automatique soit manuellement sur l'autre calculateur qui assurera à son tour le travail.

La deuxième possibilité est d'autoriser les deux calculateurs à recevoir les informations. Les deux calculateurs traitent ensuite les informations de façon autonome. Toutefois, seul le calculateur maître est autorisé à donner une réponse. Cette solution permet de mettre à jour les informations du calculateur esclave.



Système relié très indirectement

FIG I

Il est également possible que le calculateur esclave compare de temps en temps les résultats obtenus. Cependant cela nécessite un moyen de communication entre les deux machines.

Quelle que soit la philosophie choisie, les systèmes reliés très indirectement présentent l'avantage d'être très simples :

- simplicité de conception
- de maintenance
- de programmation.

Toutefois, ils présentent plusieurs inconvénients :

- ils sont chers (duplication de l'ensemble du matériel)
- ils posent des problèmes pour maintenir à jour les données se trouvant sur les mémoires (central ou externe) privés de chacune des machines.

Cet inconvénient est parfois supprimé pour les mémoires externes en mettant sur les périphériques des interrupteurs qui permettent de les attribuer à l'une ou à l'autre des machines.

- il n'existe aucun moyen de communication entre les calculateurs.

Ce dernier inconvénient est évité par les autres classes de systèmes.

2. SYSTEMES RELIES INDIRECTEMENT

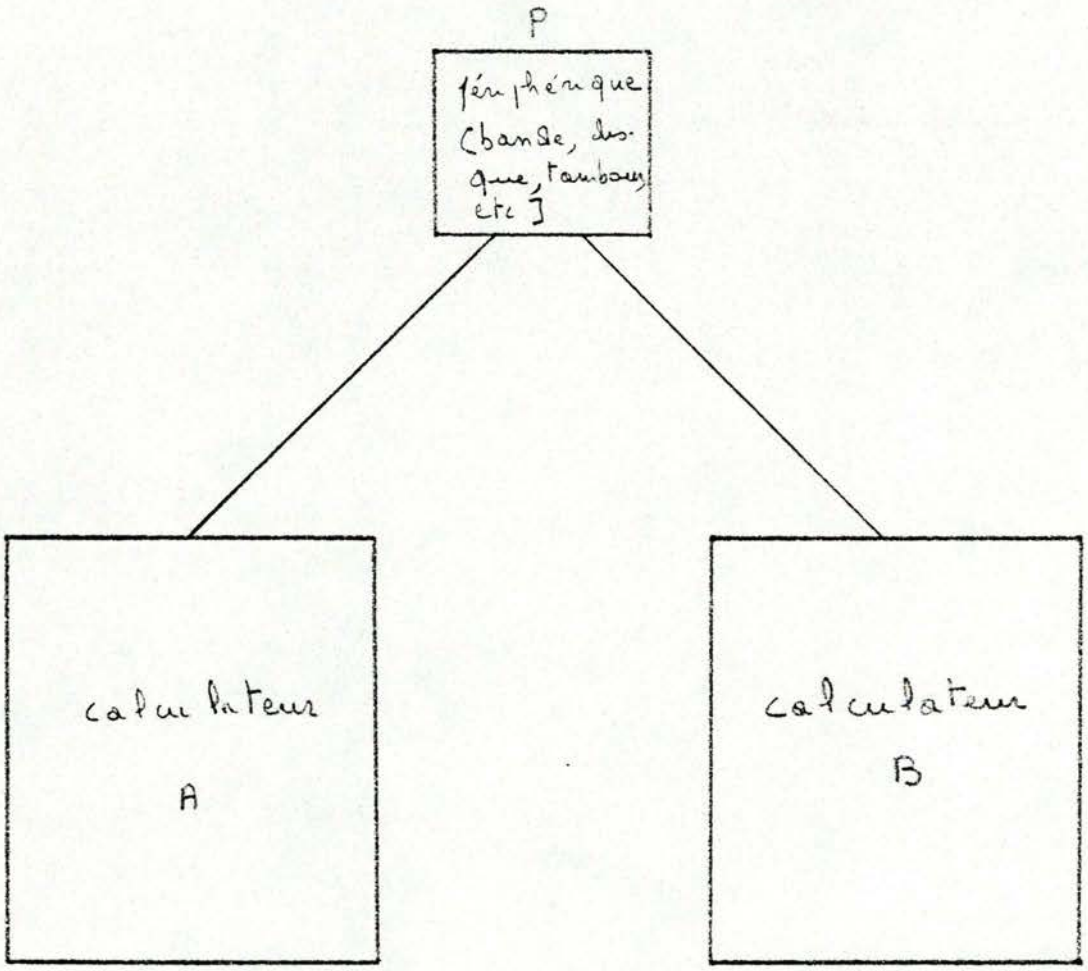
Il s'agit de calculateurs ayant la possibilité de dialoguer mais qui n'ont pas de mémoire centrale commune. L'intermédiaire de dialogue est soit un périphérique (disque, bande, tambour ...) soit une ligne de communication.

La configuration la plus simple dans cette classe de systèmes est celle représentée par la figure 2. Il s'agit de deux calculateurs (A) et (B) reliés par un périphérique (P). Le dialogue entre les deux machines passe alors par le périphérique qui est utilisé comme boîte aux lettres.

Supposons par exemple que le calculateur (A) désire envoyer un message au calculateur (B). Le message est déposé dans une boîte aux lettres par (A). Le destinataire (B) n'aura aucune connaissance de ce message tant qu'il n'aura pas regardé dans la boîte aux lettres. La réponse, s'il y en a une suivra la même procédure.

On voit de suite que l'inconvénient d'une telle organisation est la lourdeur du dialogue.

Un tel système a été réalisé à l'Institut d'Informatique de Namur et est exposé plus en détail dans la deuxième partie de ce travail. Dans ce genre de système la difficulté du dialogue est due à l'impossibilité de signaler le dépôt d'un message dans la boîte aux lettres.



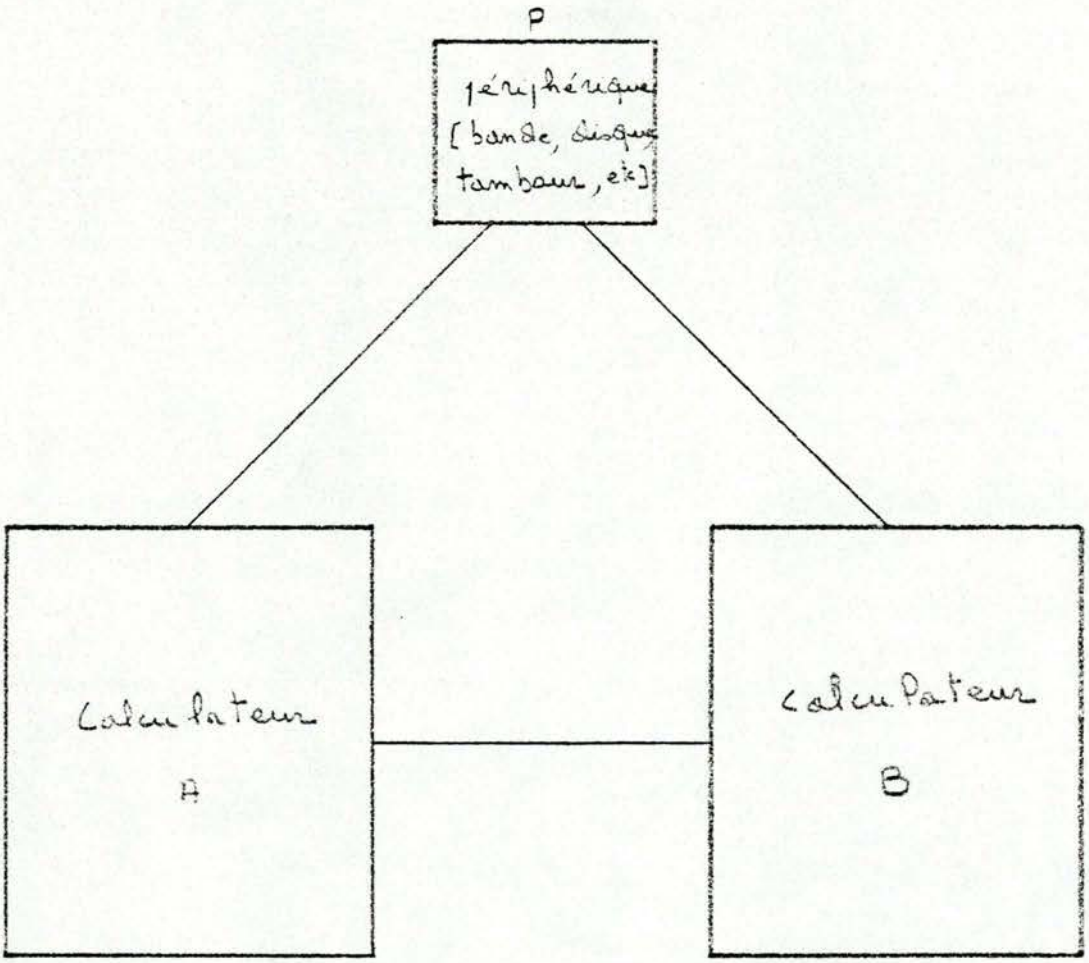
Système relié indirectement

FIG 2

Pour améliorer le dialogue on est donc amené tout naturellement à concevoir des systèmes dans lesquels l'expéditeur peut prévenir le destinataire.

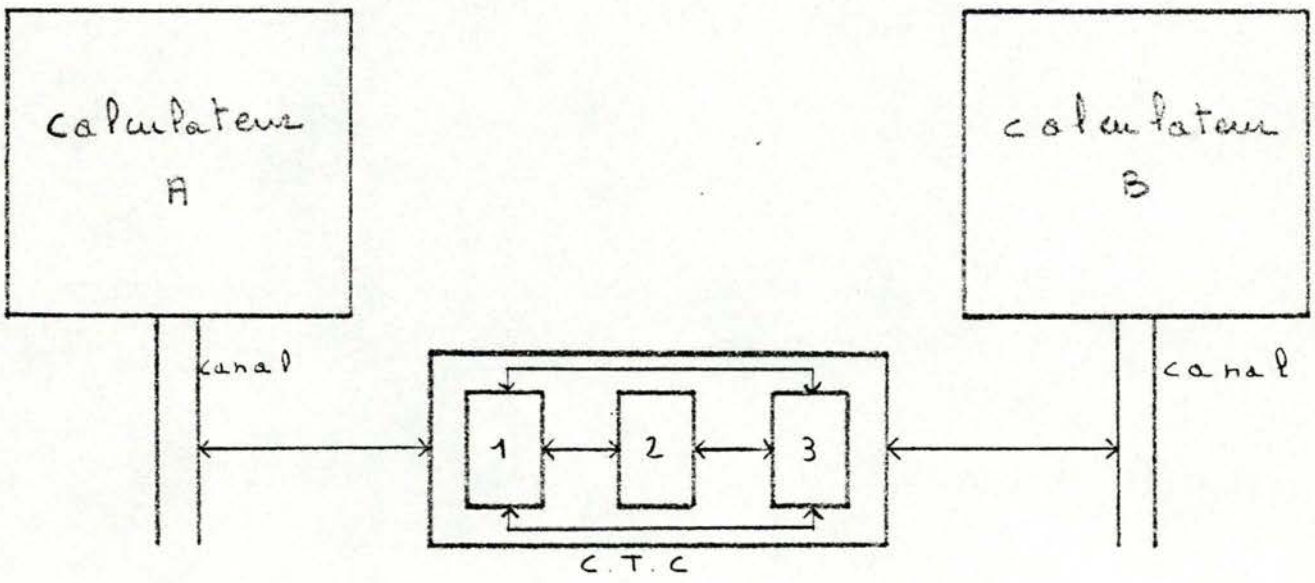
Une première méthode utilisée pour faciliter le dialogue est une configuration en delta (voir fig. 3). Dans ce type de système on utilise comme ci-dessus un périphérique ordinaire (bande, disque, etc...) comme boîte aux lettres. Mais cette fois-ci, l'expéditeur interrompt via la ligne AB le destinataire pour le prévenir de la présence d'un message dans la boîte aux lettres. Cette ligne AB est destinée uniquement à interrompre le destinataire mais n'est jamais utilisée comme moyen de communication directe. Un exemple de système de ce type est le CHILTON ATLAS développé par ATLAS COMPUTER LABORATORY. (BALD 71)

La deuxième méthode consiste à donner un rôle actif au périphérique ; cette idée a été utilisée pour la conception de la configuration ASP (LORI 72) qui utilise comme intermédiaire de dialogue le CTC "chanel-to-chanel"(fig. 4). Le CTC est un périphérique qui peut jouer un rôle actif à un instant donné, dans le sens où il lui est possible de prévenir le destinataire de l'envoi d'un message. Le dialogue entre les deux machines se passe de la façon suivante : lorsqu'un des calculateurs désire entrer en dialogue avec l'autre il signale au CTC qu'il va envoyer un message. Le CTC prévient alors le destinataire de l'envoi d'un message. Lorsque le destinataire est prêt pour la réception, les deux calculateurs effectuent leurs programmes canal de lecture (pour le destinataire) et d'écriture (pour l'expéditeur). Le rôle du CTC se réduit alors à assurer un transfert correct des données.



Configuration en Δ

FIG 3



I: élément de controle du calculateur A

2: mémoire tampon

3: élément de controle du calculateur B

Configuration ASP

FIG 4

Pour terminer cette partie sur les systèmes reliés indirectement nous citerons les connections par ligne de communication (fig. 5). Les calculateurs sont reliés par une ligne de communication et tous les messages passent d'un calculateur à l'autre par ce moyen. Dans cette méthode il est très important de bien définir les protocoles de synchronisation des messages sous peine de perdre ceux-ci.

Cette méthode est employée dans les réseaux ; l'expédition, la réception ainsi qu'une partie du traitement des messages étant assurées par des petits calculateurs "front end processor".

+

+ +

Calculateur relié par ligne de communication

FIG 5

3. SYSTEMES RELIES DIRECTEMENT

Il s'agit de systèmes constitués de plusieurs processeurs[1] qui peuvent accéder à des modules de mémoire communs.

On ne peut faire une liste exhaustive de ce type de système, tant il est possible de concevoir des systèmes différents. On se contentera donc de configurations générales existant rarement à l'état pur, mais qui ne tiennent pas compte des particularités des systèmes. En particulier nous ne tiendrons aucun compte de l'organisation des mémoires et de leur mode d'adressage etc... (pour cela voir par exemple HIGB 73).

Il est possible de distinguer quatre types de systèmes fondamentaux :

a) Les systèmes à processeurs multiples

b) Les systèmes parallèles

c) Les systèmes "pipelines"

d) Les systèmes redondants.

3.1. Systèmes à Processeurs Multiples

Les systèmes à processeurs multiples sont un agencement de plusieurs processeurs et plusieurs modules de mémoires.

(1) Nous désignons par processeur "toute machine capable de communiquer avec la mémoire"(LALI 73).

Il existe quatre façons fondamentales d'assembler les différents modules.

a) avec un bus commun (single time-shared system bus)

b) avec plusieurs bus communs (multiple time-shared system bus)

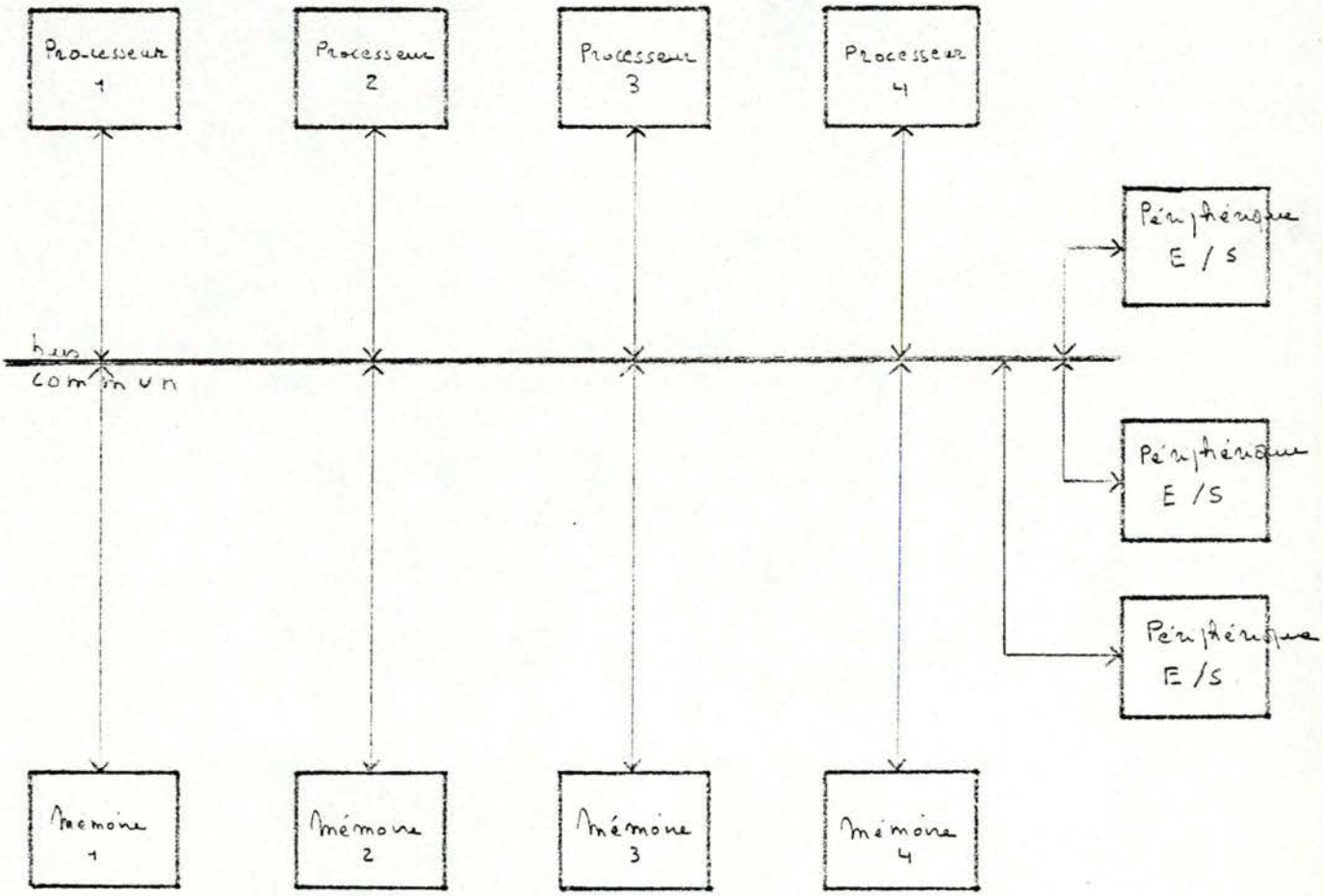
c) les systèmes multiports (multiported system)

d) les systèmes matriciels (crossbar matrice switch).

Dans la suite du paragraphe, nous décrirons ces quatre configurations fondamentales. Cependant, il est à noter que d'une part les systèmes à processeurs multiples sont souvent une combinaison ou/et une modification de ces configurations de base et que d'autre part on ne tient pas compte ici des différences qui peuvent apparaître au niveau des méthodes de contrôle des bus, des méthodes de transfert, etc... (pour cela voir par exemple THUR 72).

3.11. Systèmes avec un Bus Commun

Il s'agit de systèmes dont tous les modules mémoires et tous les processeurs sont reliés à un même bus. (fig. 6). Toute communication entre les constituants du système transite par le bus. A un instant donné le bus ne peut mettre en communication que deux éléments du système. Il est nécessaire dès lors d'assurer une gestion du bus, afin de résoudre les problèmes de simultanéité et aussi que chaque processeur puisse l'employer à son tour. On utilise pour cela des techniques de "time-sharing" ou de multiplexage.



Configuration avec un bus commun

FIG 6

On perçoit immédiatement les avantages de tels systèmes :

- grande flexibilité ; l'ajoute ou la suppression d'un module mémoire ou d'un processeur ne présente pratiquement aucune difficulté.

- coût faible grâce d'une part au fait que tout les modules sont partageables et d'autre part au peu de matériel nécessaire pour réaliser la connexion des différents modules.

Cependant, les limites d'un tel système apparaissent également. Ces limites proviennent évidemment du bus. En effet, celui-ci est le seul moyen de transfert d'un module à un autre. Il s'ensuit qu'il risque d'être rapidement "embouteillé" si de nombreux transferts sont effectués. Lors d'accès simultanés au bus, les processeurs demandeurs doivent attendre que celui-ci soit libre. Une partie de leur potentiel est donc perdu à cause du bus.

Un autre inconvénient de ce type de système est sa fiabilité. En effet, une panne au niveau du bus ou de son contrôleur arrête l'entièreté du système.

Dans ce genre de configuration des différences peuvent apparaître suivant que l'on utilise un bus bi-directionnel ou deux bus uni-directionnel, ainsi que le nombre d'informations transportées par le bus.

Les calculateurs de la série PDP II de Digital Equipment Corporation sont des systèmes avec bus bi-directionnels (PDP II).

3.12. Systèmes avec Plusieurs Bus Communs

Nous venons de voir qu'un système avec un bus commun présentait un certain nombre d'inconvénients. Il est possible de minimiser ces inconvénients en augmentant le nombre de bus (fig. 7).

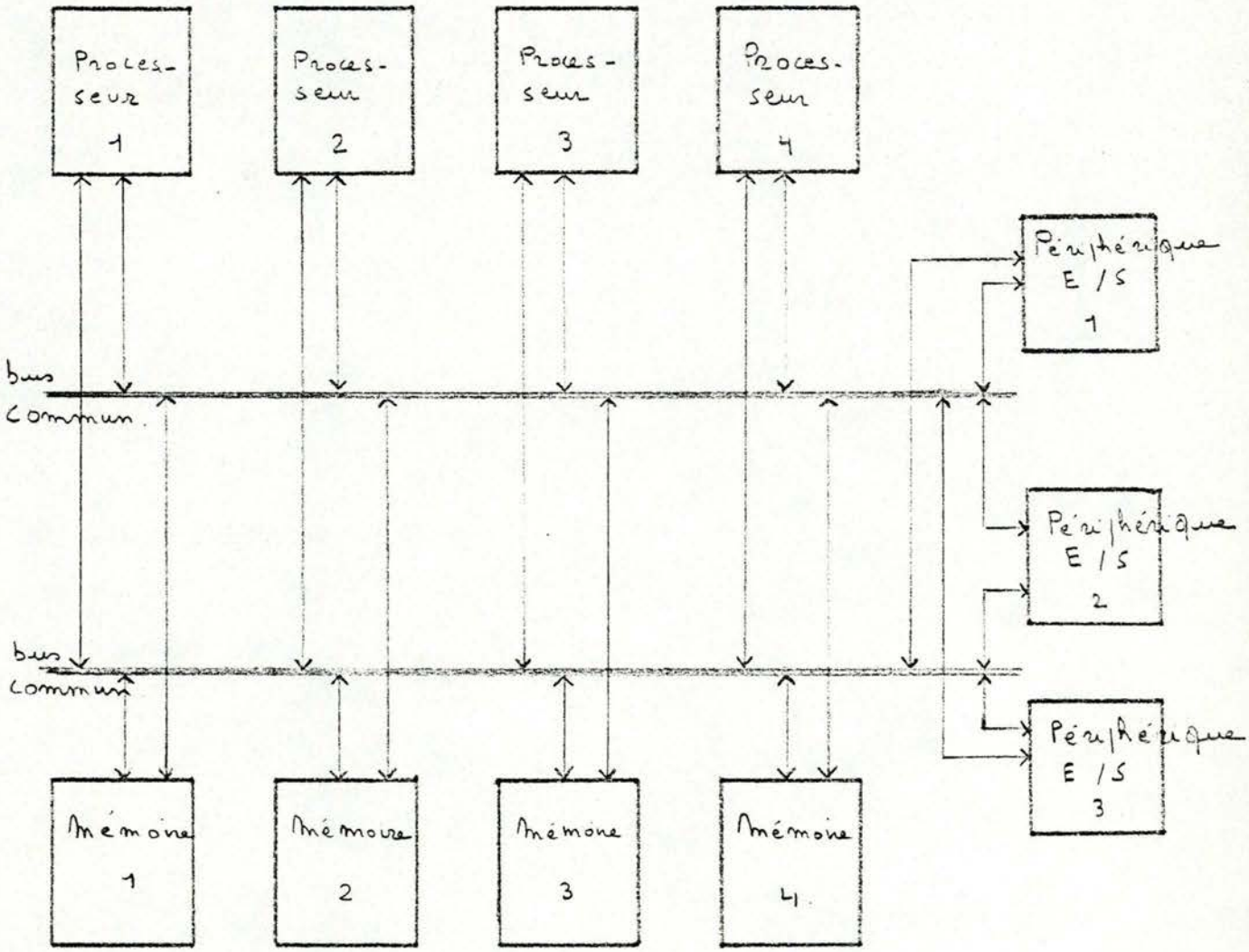
Cette augmentation du nombre de bus vise deux objectifs :

- d'une part la réduction du nombre de conflits au niveau de chacun des bus ce qui permet d'augmenter les performances de l'ensemble du système.

- d'autre part la possibilité de ne pas arrêter tout le système en cas de panne du bus.

Pour réaliser ces objectifs il est nécessaire d'implémenter deux fonctions qui n'existaient pas dans la configuration précédente :

- d'une part les modules actifs (processeurs) doivent être capables de sélectionner un bus particulier.



Configuration avec plusieurs bus commun

FIG 7

- d'autre part les modules passifs (mémoires) doivent pouvoir résoudre le problème des accès simultanés.

On voit aisément qu'un tel système a de meilleures performances que les systèmes avec un bus, en particulier s'il y avait un embouteillage au niveau du bus. Cet accroissement des performances contrebalance l'augmentation du coût du matériel.

Il est à noter qu'il est possible de combiner les deux types de systèmes décrits précédemment. Heart (HEAR 73) par exemple, emploie des processeurs mis deux par deux sur un même bus, ces différents bus étant reliés aux modules mémoires par plusieurs bus communs.

Les deux configurations précédentes sont limitées, d'un point de vue performance, par les conflits se produisant au niveau d'un bus. En effet, il suffit que deux processeurs veuillent utiliser, au même instant, le même bus pour qu'il y en ait un qui doive attendre même si les modules mémoires adressés sont différents.

Les deux configurations suivantes essayent d'éviter cet inconvénient en reportant le problème des conflits au niveau des modules mémoires et en multipliant les "chemins" possibles d'un point à un autre.

3.13. Systèmes Multiports

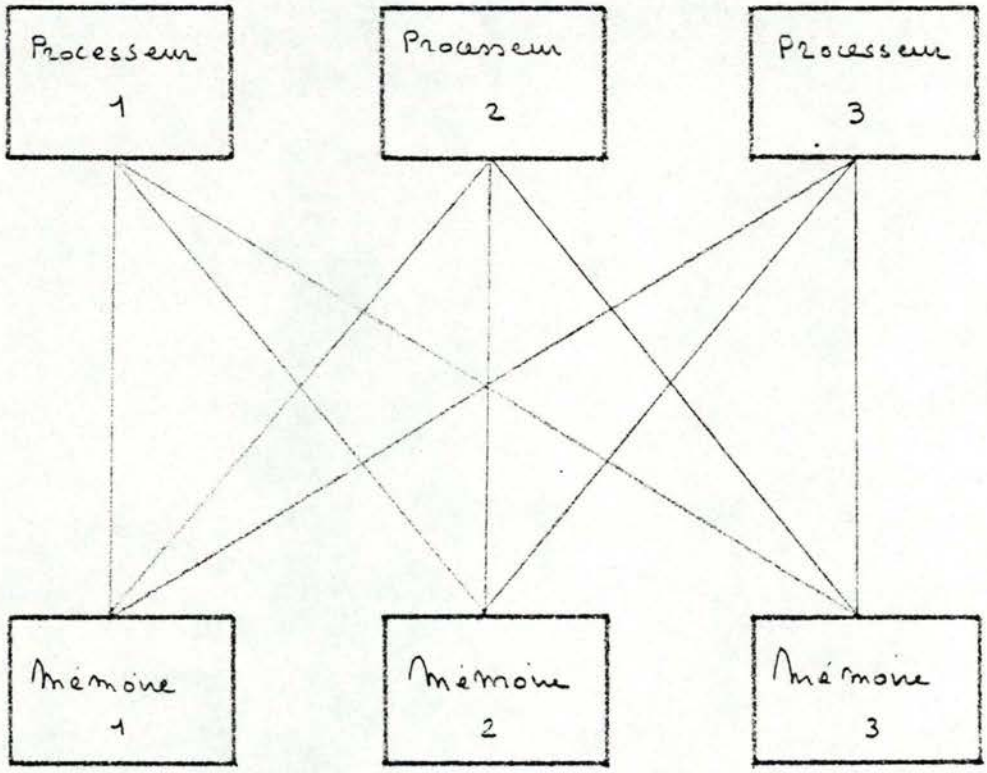
Les systèmes multiports sont tels que chaque processeur est relié à chaque module mémoire par un bus privé (fig. 8). On voit de suite l'avantage d'un tel système par rapport au précédent. En effet, ici il n'y aura conflit que lorsque deux processeurs adresseront au même instant le même module mémoire. Cependant cette configuration présente plusieurs inconvénients.

Le principal est dû au fait que chaque processeur doit être connecté à chaque module mémoire. Le coût du cablage et des connecteurs n'est donc pas négligeable.

De plus, chaque module mémoire doit comporter autant d'entrées qu'il y a de processeurs. Il faut donc ajouter les circuits nécessaires ainsi que ceux nécessaires à la résolution des problèmes de demandes simultanées. Là aussi le coût de ces ajoutes n'est pas négligeable.

D'autre part le nombre de portes se trouvant sur un module donne la configuration maximum qu'il est possible de réaliser. Il convient par conséquent de définir de façon extrêmement précise les configurations limites.

Des modifications souvent utilisées sont : l'ajoute de mémoires privées pour les processeurs, ou bien des modules mémoires accessibles à une partie des processeurs.



Configuration multiport

FIG 8

Il est évident que de telles solutions nuiront à la généralité, et présentent des inconvénients en particulier du point de vue de la flexibilité du système. En effet si un processeur tombe en panne il n'est pas possible à un autre processeur de continuer le travail car il n'aura pas accès aux informations se trouvant dans la mémoire privée.

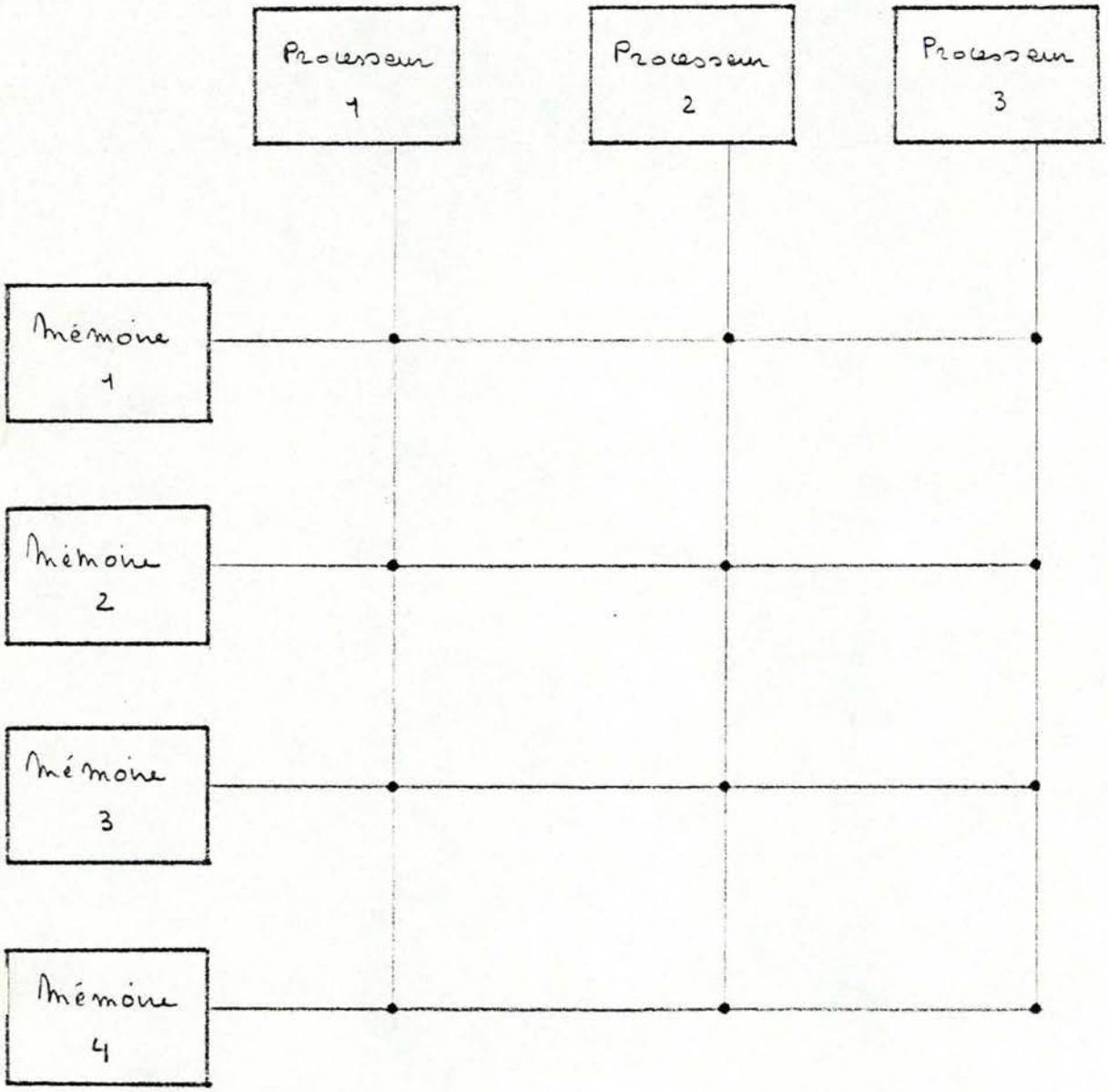
3.14. Systèmes Matriciels

Le croisement matriciel est une méthode pour connecter M éléments d'un type (des processeurs par exemple) à N éléments d'un autre type (des mémoires) sur une base de connection un à un (fig. 9).

Contrairement à ce qui se passe dans les systèmes multiports, ici les bus sont partageables par l'ensemble des modules.

Lors d'un dialogue entre un processeur et un module mémoire, le chemin emprunté pour la connexion apparait comme un bus unique reliant les deux modules. Une fois que le dialogue est terminé, une partie du chemin peut être réutilisée pour effectuer un dialogue entre deux autres modules.

Il est possible, bien entendu, d'avoir plusieurs connexions simultanées à condition toutefois que les chemins empruntés soient mutuellement exclusifs.



Configuration matricielle

FIG 9

Il s'ensuit que les conflits de simultanéité doivent être résolus par la matrice de "switch".

Vu le nombre de fonctions que doit remplir chaque "switch", la matrice devient rapidement très complexe et très onéreuse. Par exemple pour le système Hughes H 4400 (8 processeurs 16 modules mémoires) la matrice de "switch" contient 2, 5 fois plus de composants que l'ensemble des processeurs.

Un autre inconvénient des systèmes matriciels est leur fiabilité. En effet, qu'une panne survienne au niveau d'un des "switch" précédant un module mémoire, et celui-ci devient inaccessible pour tous les processeurs.

En contrepartie, les portes d'entrée de chaque module sont simples. En effet elles ne doivent pas résoudre de problèmes de simultanéité ni même savoir vers qui envoyer leur message, toutes ces fonctions étant assurées par la matrice de "switch". Un exemple de système matriciel est le Multi-interprèteur de Burroughs (DAVI 72).

3. 15. Systèmes Homogènes, Non Homogènes, Distribués et Intégrés

Jusqu'à présent, dans les descriptions de configuration de systèmes à processeurs multiples, nous avons toujours supposé que les processeurs étaient identiques. On parle alors de systèmes homogènes. Cependant, rien n'interdit de prendre des processeurs différents, spécialisés dans telle ou telle fonction. Dans ce cas, on parle de systèmes non homogènes. On utilise d'ailleurs souvent des processeurs spécialisés pour assurer les fonctions d'entrée-sortie (JOSE 72).

Un autre point de référence pour distinguer les différents systèmes à processeurs multiples est leur mode d'organisation d'un point de vue logiciel. Dans ce domaine il existe deux possibilités :

- ou bien chaque processeur possède son propre système d'exploitation
- ou bien il existe un seul système d'exploitation pour l'ensemble des processeurs (ORNS 75).

Dans le premier cas, on parlera de systèmes distribués et dans le second de systèmes intégrés.

Dans les systèmes distribués, il est nécessaire d'avoir un processeur de contrôle qui "distribue" les tâches à exécuter aux autres processeurs. Il faut d'ailleurs remarquer que souvent il est prévu lors de la conception, que les tâches de même type seront exécutées par tel ou tel processeur, ce qui spécialise un peu les processeurs. Ceci n'est toutefois pas obligatoire.

Dans un système intégré, il n'existe qu'un seul système d'exploitation pour tous les processeurs. On voit immédiatement que ce genre de système peut poser beaucoup de problèmes de logiciel (interférence, réentrance, ordonancement des tâches etc...).

Cependant, il est évident que ce sont les systèmes les plus flexibles puisque les tâches peuvent être exécutées par n'importe quel processeur.

La différence fondamentale entre un système distribué et un système intégré réside dans le nombre de files d'attente des tâches à exécuter :

Dans le premier cas chaque processeur a sa propre file d'attente, alors que dans le second cas il n'existe qu'une seule file d'attente à laquelle tous les processeurs accèdent.

Nous pouvons maintenant définir[1] avec précision ce qu'est un système multi-processeurs.

(1) définition donnée par Enslow

C'est un système à processeurs multiples, homogène[1] et intégré. Il a donc les caractéristiques suivantes :

- il y a deux processeurs ou plus
- les mémoires sont partageables par tous les processeurs
- les systèmes d'entrée-sortie sont partageables
- il y a un seul système d'exploitation qui assure le contrôle de l'ensemble.[2]

3.2. Les Systèmes Parallèles

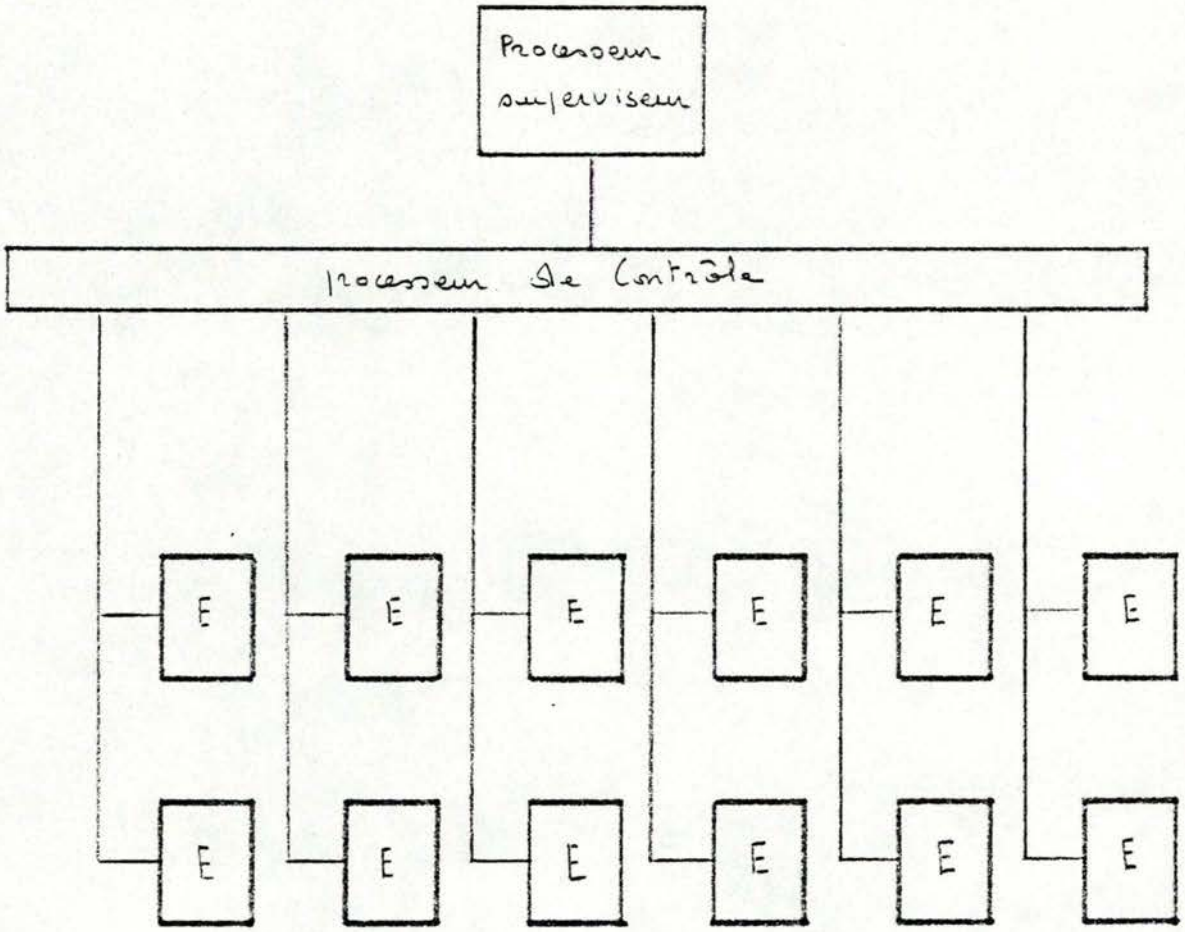
Dans un système à processeurs multiples, l'augmentation de la puissance est obtenue par la multiplication des moyens de calcul. Ceux-ci exécutant en même temps des tâches différentes sur des données différentes. En particulier chaque processeur exécute ses propres instructions.

Dans les systèmes parallèles (fig. 10), au contraire, chaque processeur exécute au même instant la même opération sur les différents items d'une donnée.

Les différents processeurs ne sont pas capables d'opérations autonomes et ne décodent pas d'instructions. Il est donc nécessaire que les processeurs soient coordonnés par un processeur superviseur qui décode les instructions et envoie des signaux de contrôle aux autres.

(1) Certains auteurs admettent que le multi-processeurs soit non homogène

(2) Il s'agit, bien entendu, del'unicitée des tables du système d'exploitation et non des instructions de ce système.



E: processeur de calcul

Système parallèle

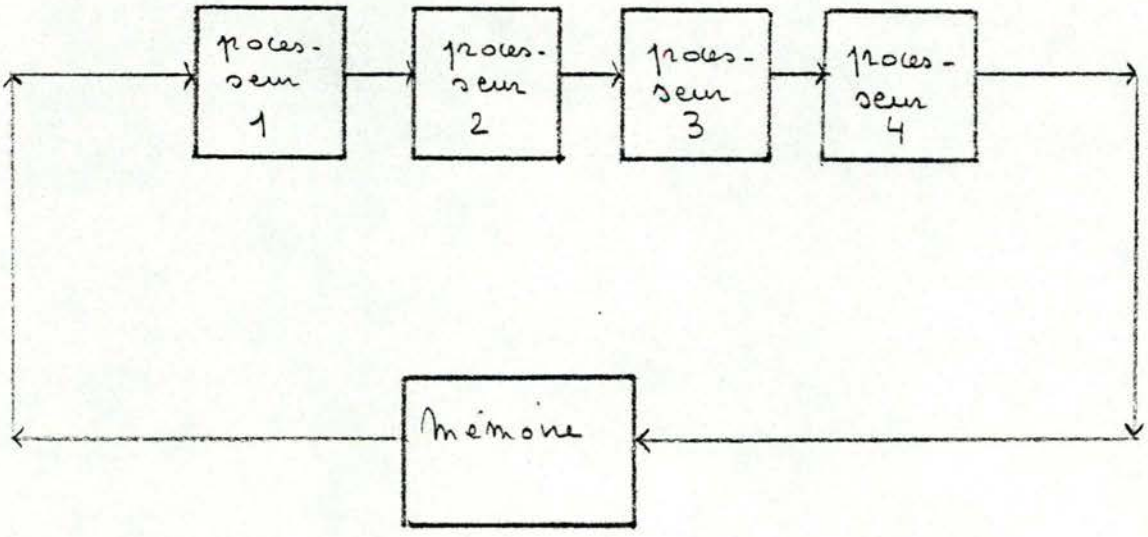
FIG IO

Pour certaines instructions qui n'ont pas besoin de tous les processeurs il est possible pour le superviseur d'inhiber les processeurs inutiles. L'exemple de système parallèle le plus connu est l'ILLIAC IV (BARN 68).

Il est à noter que si les systèmes parallèles répondent bien à des problèmes particuliers, l'ILLIAC IV, par exemple inverse une matrice (700 x 700) en une seconde, les problèmes principaux qu'ils rencontrent sont dus d'une part à la préparation des programmes capables d'employer ces possibilités au maximum, et d'autre part au fait qu'il est difficile de développer le support logiciel (entre autres les compilateurs) capable d'analyser automatiquement les programmes et de produire un code pour une exécution en parallèle.

3.3. Les Systèmes "Pipelines"

Les systèmes "pipelines" (fig. 11) sont basés sur le principe du travail à la chaîne. L'idée directrice est de décomposer le travail à effectuer en N tâches élémentaires. Chaque tâche élémentaire est exécutée à une station de travail et il y a N stations qui travaillent en parallèle. Comme dans la configuration précédente, il y a une notion de travail en parallèle ; toutefois, il ne s'agit pas ici d'effectuer la même opération sur différentes données mais au contraire d'effectuer différentes opérations sur les différents items d'une même donnée.



Système " pipe-line "

FIG II

Prenons un exemple pour bien nous faire comprendre. Supposons que nous devions effectuer l'addition flottante des deux vecteurs de N éléments soit

C = A + B

Une addition flottante peut être décomposée en 5 tâches élémentaires :

1) normalisation des opérandes (100 ns)

2) comparaison des exposants (60 ns)

3) décalage de la mantisse de l'opérande ayant le plus petit exposant (100 ns)

4) addition des mantisses (120 ns)

5) normalisation des résultats (100 ns).

Chaque addition flottante prend donc 480 ns. Dans un système "pipeline" chaque opération est effectuée par une station différente puis le résultat intermédiaire est transféré à la station suivante.

Le temps d'addition est porté à 600 ns car chaque étape de l'addition doit durer 120 ns afin de permettre à l'étape 4 de se dérouler normalement. Toutefois, après le premier résultat on obtient un résultat toutes les 120 ns. Le résultat final sera donc obtenu au bout de 600+(N-1) X 120 ns au lieu de N X 480 ns.

Pour exploiter au maximum les possibilités du "pipeline" il est donc important d'avoir le même type d'opération à effectuer afin de garder le "pipeline" rempli.

Il est à noter d'ailleurs que les opérations :

$$A = B + C \quad \text{et} \quad D = A + E$$

ne peuvent pas être exécutées en "pipeline".

Un autre problème rencontré par le système "pipeline" est celui du changement d'opération. En effet il est assez difficile (circuit de contrôle trop complexe) de mener en "pipeline" une addition suivie d'une multiplication par exemple.

Enfin, un problème relativement complexe à résoudre dans un tel système est celui des "interrupts". En effet, il convient d'identifier quelle est l'opération et quelle est la donnée qui ont provoqué "l'interrupt". Il faut pouvoir également poursuivre ou reprendre les opérations qui se trouvaient à ce moment là dans le "pipeline".

3.4. Les Systèmes Redondants

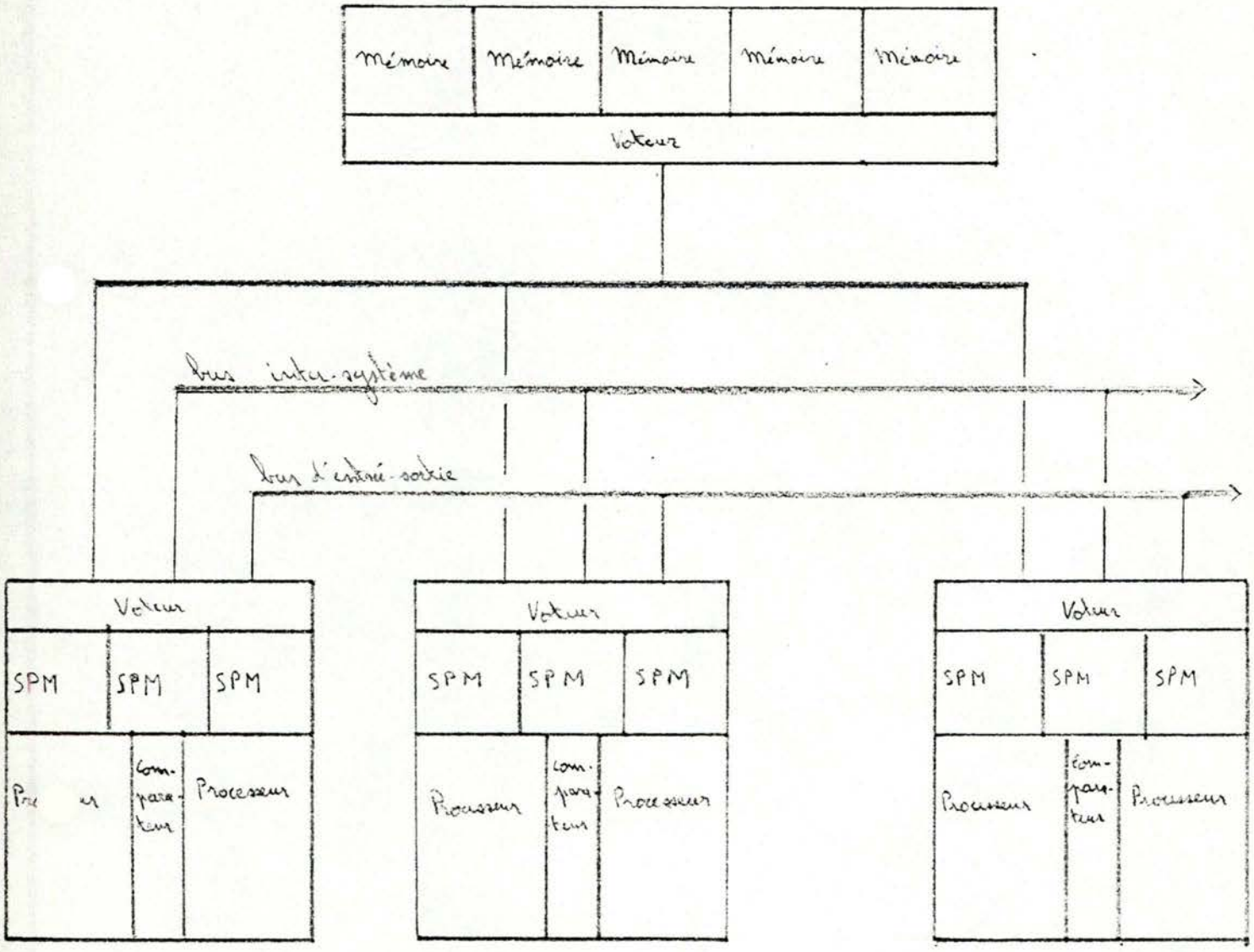
Les systèmes redondants sont développés pour des raisons de fiabilité. Ils ne correspondent à aucune configuration typique, mais plutôt à une idée commune à savoir la redondance des éléments matériels. Une autre caractéristique de ces systèmes est leur extrême modularité ce qui permet de les réparer aisément.

Deux fonctions essentielles se retrouvent dans ce genre de système :

- d'une part la détection des erreurs,

- d'autre part la continuité du travail quelle que soit la panne qui survient.

La détection des erreurs se fait par des techniques du type "vote majoritaire". Ces techniques peuvent s'employer tant au niveau du matériel (HOPK 71) et (KOCZ 71) qu'au niveau du logiciel (WENS 72). Elles consistent à faire exécuter le même travail par plusieurs éléments. Les résultats fournis sont comparés et en cas de désaccord, le module fautif est inhibé. Dès lors il faut reconfigurer le système de façon à pouvoir d'une part poursuivre le travail et d'autre part isoler le module fautif afin de les réparer. Un exemple d'un tel système est donné à la figure 12.



SPM: "scratch pad memory"

Système redondant

FIG I2

En général, il convient de conclure un chapitre comme celui-ci par une évaluation des systèmes présentés. Cette évaluation dans ce cadre-ci n'est guère aisée à faire tant les systèmes présentés sont différents et ont des buts différents. A quoi correspondrait une comparaison entre un système redondant et un système parallèle ou "pipeline" ?
Chaque système est bâti sur des critères qui lui sont propres et il ne peut donc pas être transporté ailleurs. Il n'y a donc pas dans ce domaine, de solution universelle, et on ne peut pas dire qu'une configuration est meilleure qu'une autre. Le choix se fait par le concepteur qui tiendra par conséquent compte des applications qui sont à réaliser, des moyens et des contraintes auxquels il est soumis, et de l'imagination dont il fait preuve.

DEUXIEME PARTIE

I. DESCRIPTION DU SYSTEME

1. BUT RECHERCHE PAR LA REALISATION

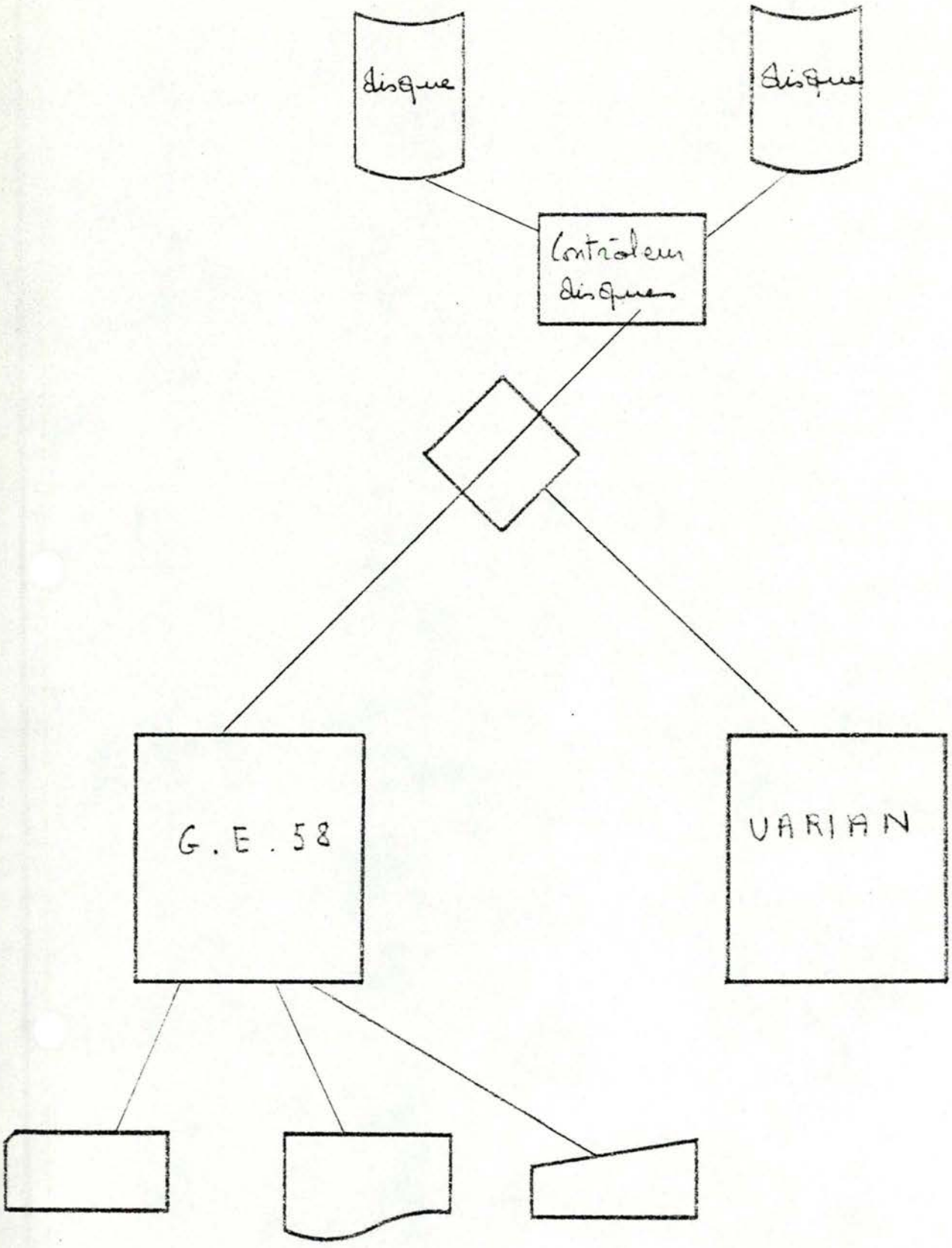
Cette seconde partie décrit la réalisation d'un système relié indirectement, qui a été faite à l' Institut d'Informatique à Namur, entre un GE 58 et un Varian 70, le périphérique intermédiaire étant un contrôleur disque (fig. 1).

Dans ce travail nous nous plaçons du point de vue du programmeur qui possède deux machines utilisant le même contrôleur disque et qui peuvent par conséquent lire ou écrire sur les mêmes disques. Nous n'entrerons donc pas dans les détails de la réalisation matérielle.

En se plaçant au niveau qui vient d'être défini, apparait immédiatement le premier problème : le partage de l'espace disque.

En effet, les deux calculateurs utilisent les mêmes disques et peuvent écrire dans les mêmes secteurs. Dans un système tel que celui-ci on peut considérer qu'il y a trois types d'information :

- les informations appartenant exclusivement au Varian
- les informations appartenant exclusivement au GE58
- les informations partagées par les deux machines.



Configuration GE58-VARIAN

FIG I

A tout instant, chacune des machines doit savoir à quel endroit du disque elle doit écrire ses propres informations ainsi que les informations partagées. Il est donc nécessaire d'assurer une gestion cohérente de l'espace disque afin qu'aucune des informations ne soit perdue. Ce problème est décrit avec plus de détail dans le premier chapitre.

Le deuxième but recherché par la réalisation, est de mettre à la disposition du Varian les terminaux du GE 58. En effet le GE 58 dispose de plusieurs terminaux et en particulier d'un lecteur de cartes et d'une imprimante qu'il serait intéressant de mettre à la disposition du Varian. Ce problème est abordé dans le deuxième chapitre. Toutefois dans cette partie, on s'est intéressé davantage à la fabrication d'un outil, qu'à une étude théorique des terminaux virtuels. Il s'ensuit que nous nous contenterons dans ce chapitre d'exposer la réalisation qui a été faite.

+

+ +

2. DESCRIPTION

Le système Varian - GE 58 est constitué de deux calculateurs travaillant en monoprogrammation (un VARIAN 70 et un GE 58) reliés par l'intermédiaire d'un contrôleur disque.

La configuration est la suivante :

GE 58 :

- une mémoire centrale de IOK mots (16 bits)
- des périphériques :
  - . un clavier alpha-numérique
  - . un clavier de contrôle
  - . un lecteur de carte
  - . un perforateur de cartes
  - . une imprimante

VARIAN 70 :

- une mémoire centrale de 32 K mots (16 bits)
- des périphériques :
  - . un clavier de contrôle
  - . une télétype

CONTROLEUR DISQUE :

- il s'agit d'un contrôleur DSU 050 auquel sont connectées deux armoires à disque. Chaque disque est constitué de 202 cylindres de 10 pistes. Chacune des pistes est divisée en 10 secteurs de 288 caractères. Toute lecture ou écriture sur disque se fait sur un nombre entier de secteurs

Primitivement le contrôleur disque faisait partie des périphériques du GE 58, ce calculateur étant antérieur au Varian. Lors de l'installation du Varian, il a été décidé pour des raisons économiques que le Varian utiliserait lui aussi le DSU 050.

+

+ +

3. CONTRAINTES

Afin de faciliter la compréhension des chapitres suivants, il est nécessaire d'exposer les contraintes auxquelles nous étions soumis lors de la réalisation.

La contrainte principale est l'existence d'un système de gestion de l'espace disque (DOS) sur le GE 58. Ce DOS organise et gère l'espace disque et bien entendu il ignore totalement l'existence du Varian (il faut se souvenir que le GE58 est antérieur au Varian d'au moins trois ans).

A l'analyse il s'est avéré qu'il était pratiquement impossible de modifier le DOS pour plusieurs raisons :

- l'addressage utilisé est un addressage absolu. Cela a pour conséquence qu'il est impossible d'insérer plusieurs instructions sans changer les adresses.

- il n'existe pas qu'un seul point de traitement de la TRF[1] mais plusieurs. Cela implique qu'il faut modifier le DOS en plusieurs endroits et ne pas en oublier.

En fait la solution la plus simple aurait été de réécrire une grande partie, si pas tout, le DOS. Cependant, cela représentait un projet trop vaste pour être réalisé dans le cadre qui nous était proposé.

(1) voir la partie 4 de ce chapitre pour la définition du catalogue disque appelé TRF.

La deuxième contrainte que nous avons rencontré concerne les informations partagées. Le problème se pose ici de la même façon qu'en multi-programmation c'est-à-dire que pendant qu'une des machines ou qu'un utilisateur modifie des informations il faut interdire à l'autre machine ou aux autres utilisateurs de faire des modifications de ces informations.

On sait qu'il est possible de faire des mises à jour d'informations partagées à condition qu'on puisse définir dans les programmes des sections critiques c'est-à-dire des parties de programme pendant lesquelles une seule machine peut modifier l'information. Pour définir ces sections critiques, il est indispensable d'avoir des primitives de synchronisation. On peut considérer que le Varian possède ces primitives. En effet, le programmeur Varian a la possibilité de demander la réservation du contrôleur disque à l'usage exclusif du Varian, et bien sûr de libérer le contrôleur. Entre la réservation et la libération du contrôleur disque par le Varian, le GE 58 ne peut accéder au disque (il doit attendre que le contrôleur soit libre), et par conséquent le Varian peut effectuer les modifications qu'il désire.

Malheureusement le GE 58 ne possède pas de primitives de ce genre et par conséquent le Varian peut modifier une information pendant que le GE 58 effectue lui aussi une modification.

Nous ne pouvons pas par conséquent considérer que nous possèdons des primitives de synchronisation. Il s'ensuit que nous ne possèdons aucun moyen d'éviter la simultanéité.

Dès lors, il dépendra de chaque application traitant des données partagées de résoudre ce problème. Les solutions seront variables suivant les applications. Toutefois celles-ci devront être conçues de façon à ce qu'il soit possible de récupérer les erreurs dues à un traitement simultané.

Nous sommes bien conscients que ceci relève plus de la "recette de cuisine" que d'un travail scientifique rigoureux, mais nous pensons que l'informaticien doit savoir s'adapter aux circonstances et qu'il est parfois amené à se débrouiller avec les "moyens du bord".

Une autre contrainte du système est l'impossibilité de communiquer directement de processeur à processeur, ni même de signaler l'existence d'un message dans la boîte aux lettres. Cela a pour conséquence principale une certaine lourdeur dans le dialogue.

+

+ +

4. GESTION DE L'ESPACE DISQUE ASSURE PAR LE GE 58

Les principes fondamentaux de gestion de l'espace disque pour le GE 58 sont la linéarité de l'espace disque et la continuité de l'espace occupé. Au fur et à mesure de l'occupation du disque le système de gestion construit une table (appelée table de référence ou TRF) donnant une image de la relation qui existe entre un fichier (I) et son emplacement physique (une description précise de la T R F est donnée dans l'annexe I).

Le système de gestion peut assurer 4 opérations :

- créer un fichier
- rechercher un fichier
- agrandir un fichier
- supprimer un fichier.

4. 1. Création d'un Fichier

Créer un fichier, c'est pour le système de gestion créer la référence de ce fichier dans la T R F et allouer de la place sur le disque. Supposons que nous voulions créer un fichier de X cylindres (2) et que l'espace déjà occupé est de Y cylindres. L'opération de création consiste à accorder les X cylindres demandés à la suite des Y cylindres. Dans la T R F il suffit d'ajouter un article contenant le nom du fichier, son adresse début et sa longueur (voir fig. 2).

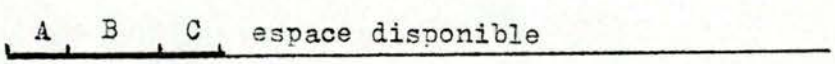
(1) Nous appelons ici "fichier" une zone de l'espace disque identifiée par un nom se trouvant dans la T R F sans tenir compte du contenu.

(2) Le système accorde toujours un nombre entier de cylindres.

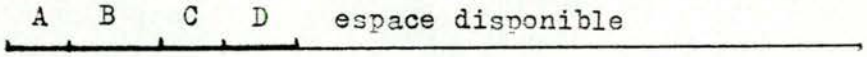
TRF

DISQUE

NOM	F	Adr.	Long.
A	o	I	IO
B	o	II	I5
C	o	26	IO



NOM	F	Adr.	Long.
A	o	I	IO
B	o	II	I5
C	o	26	IO
D	o	36	I5



F :"flag" de suppression logique

Création d'un fichier D

FIG 2

4. 2. Recherche d'un Fichier

Pour retrouver un fichier il suffit de balayer la T R F afin de trouver l'article correspondant au nom. L'article de la T R F donne immédiatement l'emplacement du fichier.

4. 3. Agrandissement d'un Fichier

Nous avons vu que lors de la recherche d'un fichier on se contentait de chercher dans la T R F l'article correspondant au nom. Cela signifie qu'à un nom de fichier correspond un seul article dans la T R F et réciproquement. Cela implique que lorsqu'on veut agrandir un fichier, les cylindres supplémentaires qui sont alloués doivent être ceux qui suivent déjà accordés au fichier. Dans l'article de la T R F seule la longueur du fichier est modifiée.

L'agrandissement d'un fichier peut être parfois une opération longue. En effet, si entre le moment où le fichier a été créé et celui où il est agrandi, d'autres fichiers ont été créés il est nécessaire de décaler ceux-ci (voir fig. 3).

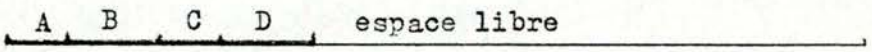
4. 4. Suppression d'un fichier

La suppression d'un fichier est la libération de l'espace disque que le fichier occupait. Cet espace devient alors disponible. Malheureusement si le fichier se trouve au milieu d'autres, en le supprimant on crèe une discontinuité dans l'occupation du disque.

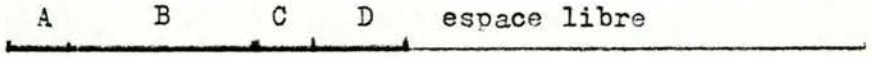
TRF

NOM	F	Adr.	Long.
A	o	I	IO
B	o	II	I5
C	o	26	IO
D	o	36	I5

DISQUE



NOM	F	Adr.	Long.
A	o	I	IO
B	o	II	30
C	o	4I	IO
D	o	5I	I5



F: "flag" de suppression logique

Agrandissement du fichier B

FIG 3

Cette discontinuité ne pouvant être admise, il est nécessaire de décaler les fichiers créés ultérieurement afin de la combler.

Tout comme l'agrandissement, la suppression d'un fichier est une opération relativement lourde. La suppression d'un fichier étant une opération plus fréquente que les agrandissements, elle se fait en deux étapes :

a) Une suppression logique

Elle consiste a rendre inaccessible le fichier bien que celui-ci existe encore. Il n'est plus possible, dès lors d'effectuer la moindre opération sur ce fichier. Cette opération se fait uniquement sur la T R F et consiste a mettre un "flag" signalant que le fichier peut être supprimé (fig. 4).

b) Une suppression physique

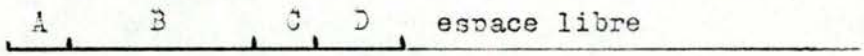
Dans cette étape déclenchée manuellement, on réorganise l'espace disque de façon à ce que tous les fichiers, logiquement supprimés, disparaissent. Les fichiers restants sont compactés vers le début du disque (fig. 5).

En résumé, on peut dire que la gestion de l'espace disque effectuéepar le GE 58 est simple du point de vue logique. Il est important de remarquer que tout fichier (sauf le premier) est susceptible d'être "déplacé" et ce à n'importe quel moment. Cela implique en particulier que dans le cadre de la liaison GE 58 - Varian, le Varian n'est jamais sûr de l'adresse d'un fichier, celui-ci ayant peut-être été déplacé entre le moment où le Varian a consulté la T R F et le moment où il accède au fichier.

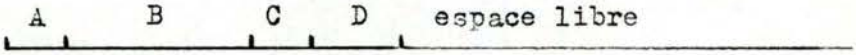
TRF

DISQUE

NOM	F	Adr.	Long.
A	o	I	IO
B	o	II	30
C	o	4I	IO
D	o	5I	I5



NOM	F	Adr.	Long.
A	o	I	IO
B	I	II	30
C	o	4I	IO
D	I	5I	I5



Suppression logique des fichiers B et D

FIG 4

NOM	F	Adr.	Long
A	o	I	IO
C	o	II	IO

Suppression physique des fichiers B et D

FIG 5

II. PARTAGE DE L'ESPACE DISQUE

Le premier objectif à atteindre est le partage de l'espace disque. Le problème peut se poser dans les termes suivants :

> Comment faire pour que chaque machine puisse écrire et lire des informations sur le disque, au moment où elle le désire et être sûre que ces informations seront conservées ?

Il s'agit donc d'attribuer à chaque machine l'espace disque qui lui est nécessaire. Deux solutions sont possibles :

- un partage fait à priori (partage statique)
- un partage fait au fur et à mesure des besoins (partage dynamique).

Nous allons envisager chacune des solutions en nous attachant à voir quels sont leurs avantages et leurs inconvénients dans le cadre de la réalisation qui nous est proposée.

1. PARTAGE STATIQUE

Un partage statique consiste à définir à priori les zones du disque que chaque machine peut utiliser. Il s'agit donc d'allouer à l'initialisation du système une zone disque pour chacune des machines.

A l'intérieur de chacune des zones ainsi définies, chaque machine est maître ; cela nécessite par conséquent un système de gestion (1) pour chaque zone. Les machines dans une telle organisation n'ont pas de possibilité de communication. Si on désire une telle possibilité, il faut définir une troisième zone réservée uniquement à ce rôle. Dans ce cas, il est nécessaire de posséder des mécanismes de protection pour les informations partagées et c'est le cas en particulier des fichiers communs.

Dans le cadre de la liaison GE 58 - VARIAN, un partage statique est réalisable. En effet, il est possible de réserver le début du disque pour le Varian. Le reste du disque reste alors à la disposition du GE 58.

Pour être mené à bien, le partage statique nécessite :

- l'écriture d'un système de gestion pour l'espace disque réservé au Varian
- une légère modification des programmes d'initialisation des disques afin de réserver l'espace Varian
- des programmes de dialogue entre les deux machines.

L'inconvénient majeur d'une solution de partage statique est qu'il faut définir à l'avance la longueur de chaque zone. L'évaluation de la quantité d'espace disque nécessaire à chaque machine doit se faire de façon très précise afin de ne pas pénaliser l'une ou l'autre. Dans la réalisation qui nous est proposée, cette évaluation est difficile à effectuer car les besoins de chaque machine sont très variables d'une application à l'autre.

(1) Nous entendons par "système de gestion" tout mécanisme qui permet de retrouver un fichier à partir de son nom, d'allouer ou de libérer l'espace suivant les besoins d'une machine.

2. PARTAGE DYNAMIQUE

Pour éviter l'inconvénient de devoir définir à l'avance quels sont les espaces nécessaires à chaque machine, on effectue une allocation ou une désallocation dynamique de l'espace c'est-à-dire que l'on accorde ou on libère de l'espace au moment où les machines le désirent.
Une telle solution implique l'unicité du système de gestion de l'espace disque. Le problème de partage se ramène alors à un problème d'accès au système de gestion. Dans une liaison telle que celle réalisée, il y a deux possibilités :

- ou bien une seule machine peut accéder au système de gestion
- ou bien les deux machines peuvent y accéder.

2.1. Accès Autorisé Pour Une Seule Machine

Dans ce cas, nous avons une organisation du type maître-esclave. Seule la machine maître (ici le GE 58) peut accèder au système de gestion. Si la machine esclave désire effectuer une opération de gestion elle devra "demander" au maître de la faire pour elle. Il faut donc établir un dialogue entre les deux machines. Le disque étant la seule possibilité de dialogue on devra l'utiliser comme "boîte aux lettres". La machine esclave y déposera sa demande. La machine maître prendra connaissance du message, le traitera et déposera sa réponse dans la boîte aux lettres. Pour connaître la réponse, la machine esclave devra à son tour consulter la boîte aux lettres.

Dans ce genre de dialogue la difficulté principale réside dans le fait qu'on ne sait pas quand il y a un message dans la boîte aux lettres. Il faut donc régulièrement consulter la boîte aux lettres pour voir si un message a été déposé.

La consultation de la boîte aux lettres peut se faire soit périodiquement, soit de façon aléatoire. Dans le premier cas il convient de définir la période de consultation. Celle-ci doit être un compromis entre des consultations trop fréquentes qui seraient pour la plupart inutiles, et des consultations trop peu fréquentes qui allongeraient inutilement le temps de réponse à une demande. Cette période dépendra évidemment du nombre d'occasion de dialogue qu'il existe dans le système ainsi que du temps qu'il est nécessaire pour répondre à une demande. Dans le cas où les machines consultent la boîte aux lettres de façon aléatoire, il est nécessaire de définir des protocoles de dialogue afin de savoir si une réponse a été déposée ou non.

Les procédures décrites ci-dessus, conviennent dans les cas où on ne dépose qu'un message à la fois, et que celui-ci doit être entièrement traité avant de pouvoir déposer le suivant. Si l'on désire pouvoir déposer plus d'un message, il est nécessaire de disposer de mécanismes de protection des informations en cas de mise à jour simultanée de la boîte aux lettres.

Pour la liaison GE 58 - VARIAN cette solution n'est pas réalisable pour deux raisons principales :

- la première est due au type de gestion assuré par le GE 58. En effet nous avons vu que le GE 58 pouvait déplacer les fichiers : Supposons qu'à un instant donné, le Varian demande l'adresse d'un fichier.

Le GE 58 fournit une réponse qu'il met dans la boîte aux lettres. Entre l'instant où la réponse a été déposée et l'instant auquel le Varian en prend connaissance il se peut que le fichier ait été déplacé. Dès lors l'information donnée par le GE 58 est fausse.

- L'autre raison qui empêche la réalisation provient du fait qu'il n'est pas possible d'implémenter sur le GE 58 la procédure de consultation de la boîte aux lettres sans modifier fortement le système d'exploitation existant déjà. En effet le GE 58 ne possède ni "interrupt" ni "timer" ce qui nous oblige à introduire des séquences de consultation à l'intérieur du système d'exploitation.

2. 2. Accès Autorisé Pour les Deux Machines

Lorsque les deux machines peuvent accèder au système de gestion de l'espace disque, elles ont la possibilité d'occuper et de libérer l'espace disque comme elles l'entendent. Chaque machine gèrera l'espace disque comme si elle était seule, ignorant tout des actions entreprises par l'autre machine.

Il est évident que d'une part il faut qu'elles assurent le même type de gestion et que d'autre part il existe un moyen de protéger les informations partagées pendant leur mise à jour.

Pour la liaison GE 58 - Varian cette solution est réalisable. En effet il suffit de permettre au Varian d'effectuer la même opération de gestion que le GE 58. Toutefois il faut résoudre le problème de simultanéité d'accès à la TRF. Comme nous ne possèdons aucun moyen sûr de résoudre ce problème, il sera nécessaire de trouver des solutions permettant d'adapter la gestion faite par le Varian à celle faite par le GE 58.

+

+ +

3. CHOIX D'UNE SOLUTION

Nous venons de voir qu'il y a deux solutions de partage de l'espace disque pour la liaison GE 58 - Varian.

- soit un partage statique du disque,
- soit un partage dynamique : les deux machines accèdant à la T R F.

Le choix s'est porté sur la seconde solution pour trois raisons :

- la première et la plus importante est que cette solution ne nécessite aucune modification du système d'exploitation du GE 58. On se contentera d'adapter le Varian aux procédures déjà assurées par le GE 58.

- la deuxième raison est qu'un partage statique n'est pas indépendant du disque sur lequel on travaille. En effet le système GE 58 travaille avec deux disques : l'un pour le système, l'autre servant de disque de manoeuvre. Il est possible de réserver une partie du disque système pour le Varian, mais celle-ci ne pourrait se trouver que derrière l'espace réservé au système. Par conséquent l'adresse début de l'espace Varian serait différente suivant que l'on se trouve sur le disque système (l'espace Varian débuterait juste après le système) ou sur le disque manoeuvre (l'espace Varian commencerait dans ce cas au début du disque).

De plus des expériences ont montré que si une telle solution était adoptée, le fichier se trouvant sur le disque système peut être déplacé. Il s'ensuit que sur ce disque le Varian n'est jamais sûr de l'adresse de son fichier, ce qui implique qu'en fait il ne pourrait travailler que sur le disque de manoeuvre.

- la troisième raison du choix qui a été fait est d'ordre esthétique.

Il est plus satisfaisant intellectuellement d'avoir une solution de partage dynamique dans laquelle chaque machine utilise les ressources suivant ses besoins plutôt qu'un partage statique qui fixe d'autorité les limites des besoins des machines.

L'inconvénient de ce choix est qu'il apporte le problème de la simultanéité de traitement des données partagées. Comme nous l'avons déjà dit, nous ne possèdons pas les moyens nécessaires pour résoudre ce problème. Nous essaierons donc plutôt de faire en sorte que les erreurs qui en découlent ne soient pas préjudiciables au reste du système et qu'elles soient récupérables aisément.

+

+ +

4. REALISATION

Comme nous l'avons dit précédemment, il s'agit d'adapter le Varian à la gestion disque assurée par le GE 58. Les manipulations de la T R F que pourra effectuer le Varian seront les mêmes que celles faites par le GE 58. Le Varian ne peut par conséquent qu'assurer une gestion de même type que celle faite par le GE 58. Rappelons que pour le système de gestion de l'espace disque, il existe cinq opérations que l'on peut diviser en trois groupes (1) :

- les opérations se faisant uniquement sur la T R F, c'est-à-dire la création et la suppression logique des fichiers. Ces opérations consistent en l'ajoute d'informations dans la T R F mais ne change rien à l'organisation du disque ni à celle de la T R F.

- les opérations modifiant et, la T R F, et l'organisation du disque à savoir l'agrandissement et la suppression physique du fichier.

- enfin, l'opération de consultation qui ne modifie ni la T R F ni l'espace fichier.

(1) On pourrait considérer qu'il existe un quatrième groupe qui serait les opérations qui modifient les fichier sans modifier la T R F. Cependant ces opérations ne dépendent pas du système de gestion de l'espace disque mais des applications elles-mêmes.

Avant d'aborder de façon plus détaillée chacun des trois groupes, regardons comment se pose le problème de la simultanéité dans le cadre de la réalisation qui nous est proposée. Toute modification de la T R F débute par une lecture (soit LE cet instant) et se termine par une écriture (soit EC cet instant). Cette lecture ou cette écriture se fait en une fois et il n'y a pas de lecture ou d'écriture intermédiaire. Au cours du temps six séquences sont possibles :

1)	LE_{GE58}	EC_{GE58}	LE_{VAR}	EC_{VAR}
2)	LE_{GE58}	LE_{VAR}	EC_{GE58}	EC_{VAR}
3)	LE_{GE58}	LE_{VAR}	EC_{VAR}	EC_{GE58}
4)	LE_{VAR}	EC_{VAR}	LE_{GE58}	EL_{GE58}
5)	LE_{VAR}	LE_{GE58}	EC_{VAR}	EC_{GE58}
6)	LE_{VAR}	LE_{GE58}	EC_{GE58}	EC_{VAR}

Les séquences 1 et 4 ne présentent pas de problème de simultanéité. Nous avons vu qu'il était possible, à partir du Varian, d'interdire l'accès au disque pour le GE 58. Dès lors les situations 2, 5 et 6 ne se présenteront pas. Donc seules des séquences du type 3 peuvent survenir. Nous noterons que dans ce cas ce sont les informations fournies par le Varian qui sont perdues.

Pour toutes les opérations susceptibles d'être exécutées par le Varian on devra par conséquent tenir compte du fait que les résultats de l'opération peuvent être perdus et par conséquent il conviendra de contrôler la validité des informations avant de s'en servir.

4. 1. Opération Modifiant Uniquement la T R F

Supposons que le Varian puisse exécuter ces opérations. Supposons qu'à un instant donné le Varian crèe un fichier. Si cela arrive dans une séquence 3, quelle que soit l'opération effectuée à ce moment par le GE 58, il n'y aura pas de discordance entre la T R F et l'oganisation du disque.

L'article ajouté à la T R F par le Varian aura peut être été sur-écrit, c'est-à-dire que cela apparaîtra comme si le Varian n'avait fait aucune opération de création. Il est par conséquent facile de s'en apercevoir en consultant la T R F.

De façon la plus pratique, cela revient à dire que pour effectuer une création on suivra la procédure suivante :

- réservation du disque pour le Varian
- mise à jour de la T R F
- libération du disque pour le GE 58.

Si l'on désire utiliser le fichier qui a été créé pour y mettre des informations, il conviendra de consulter d'abord la T R F. Nous reviendrons sur le problème de consultation dans la suite de ce chapitre. La suppression logique suit une procédure analogue.

4. 2. Opération Modifiant la T R F et l'Organisation du Disque

Supposons maintenant que le Varian puisse effectuer des opérations d'agrandissement ou de suppression physique. Cela signifie en particulier qu'il peut déplacer les fichiers pendant que le GE 58 les traite.

On voit de suite les dangers d'une telle manipulation. Le fichier ou les fichiers ayant été déplacés, les adresses connues par le GE 58 deviennent fausses à moins que celui-ci ne consulte la T R F avant chaque accès disque. Toutefois, il ne semble pas que ce soit le cas pour le DOS actuellement implémenté.

Il existe une autre situation critique : c'est la réorganisation simultanée du disque. Supposons par exemple que les deux machines effectuent en même temps une suppression physique. Nous avons vu que nous pouvons avoir une séquence de type 3. Dans ce cas, lorsque le Varian entamera sa procédure, le GE 58 aura peut-être déjà déplacé des fichiers alors que les adresses connues par le Varian seront encore les anciennes. Celui-ci effectuera à son tour la procédure de suppression et sur-écrira les fichiers déjà déplacés. On voit donc qu'à la fin des opérations, un certain nombre d'informations du disque risquent donc d'être perdues. Les conséquences de telles erreurs étant trop graves pour être admises, on refusera au Varian toute procédure de suppression physique et d'agrandissement de fichier. Ces procédures restent toutefois faisables à partir du GE 58, c'est-à-dire qu'il est possible d'agrandir à partir du GE 58 un fichier créé par le Varian, et la suppression physique des fichiers supprimés par le Varian sera faite en même temps que celle des fichiers du GE 58.

4. 3. Opérations de Consultation

Ces opérations ne modifient en rien la T R F ou l'organisation du disque. Elle est donc facilement réalisable par le Varian. Toutefois, dans un certain nombre de cas les informations résultant de ces opérations peuvent être fausses.

En effet, si l'on recherche l'adresse dans le fichier pendant que le GE 58 effectue une opération qui modifie l'organisation du disque, l'adresse donnée par la T R F sera fausse. Il convient donc d'interdire au Varian de consulter la T R F pendant que le GE 58 effectue ce type d'opérations. C'est-à-dire lorsque les informations contenues dans la T R F risquent de ne pas être exactes. Les opérations qui réorganisent le disque nécessitent de nombreux accès, ceux-ci se succèdant rapidement. Le temps entre deux accès disque est donc relativement court pendant ces opérations. Soit T le temps maximum entre deux accès durant ces opérations. Supposons que le Varian veuille consulter la T R F. Si pendant un temps T ou supérieur à T le GE 58 n'a pas fait d'accès disque, cela signifie que le GE 58 ne fait aucune réorganisation du disque. Dès lors, le Varian peut consulter la T R F. Par contre si pendant le temps T un accès disque a été effectué par le GE 58, cela signifie que celui-ci est peut être (1) dans une opération de réorganisation du disque. Il faudra dès lors que le Varian attende un nouveau temps T pour voir s'il peut accèder à la T R F.

(1) "peut-être" car à un instant donné et à partir du Varian on ignore ce que fait le GE 58. C'est soit une opération de réorganisation du disque, soit une autre opération mais dans ce cas les accès disque sont plus espacés. Toutefois une application qui ferait des accès disque se succèdant à un rythme inférieur à T empêcherait le Varian de consulter la T R F.

Cette méthode est évidemment très pénalisante pour le Varian puisque tout accès disque fait par le GE 58 pendant la période d'attente oblige le Varian à recommencer celle-ci. Elle permet toutefois de s'assurer que les informations acquises par le Varian lors d'une consultation de la T R F sont exactes.

D'autre part, il est évident que l'information résultant de la consultation de la T R F doit rester exacte tout le temps que le Varian l'utilise. En particulier cela signifie que le GE 58 ne peut déplacer les fichiers qu'une fois les opérations du Varian terminées, et non à partir de l'instant ou la T R F a été lue. Pour cette raison l'opération de consultation réservera le disque à l'usage du Varian ce qui implique que le GE 58 ne pourra l'utiliser que lorsque le Varian le relachera.

+

+ +

5. FICHIERS INTERMEDIAIRES

Les procédures décrites précédemment, bien que parfois un peu lourdes, permettent au Varian et au GE 58 de travailler simultanément. Toutefois dans certains cas une des deux machines peut être pénalisée fortement par l'autre.

Supposons par exemple qu'un utilisateur veuille traiter à partir du Varian un fichier, le traitement demandant beaucoup d'accès disque. Pour lui se présentent trois solutions :

1) Consultation de la T R F
Avant chaque accès au fichier, on consulte la T R F pour connaître l'adresse du fichier, cette adresse ayant pu être changée entre deux accès au fichier.

2) Bloquage du disque
Le contrôleur disque est réservé au Varian lors de la consultation de la T R F et n'est libéré qu'à la fin de l'opération.

3) Fichiers intermédiaires
Les deux premières solutions peuvent parfois être chères : soit on double le nombre d'accès disque soit on empêche le GE 58 de travailler.

Une solution intermédiaire consiste en l'utilisation d'un fichier de travail qui n'est jamais déplacé par le GE 58. En effet lors des réorganisations du disque ne sont déplacés que les fichiers se trouvant "derrière" le fichier à supprimer.

Les fichiers se trouvant en début de disque ne sont par conséquent pas déplacés. Dès lors, si l'on réserve en début de disque de l'espace pour des fichiers intermédiaires et que ceux-ci ne sont ni supprimés ni agrandis (sauf peut-être le dernier) on possède des fichiers dont l'adresse est constante. Le Varian pourra suivre alors la procédure suivante :

- consultation de la T R F et réservation du disque
- copie du fichier à traiter dans un fichier de travail
- libération du disque pour le GE 58
- traitement sur le fichier de travail
- consultation sur la T R F et réservation du disque
- copie du fichier de travail dans le fichier initial
- libération du disque.

Un certain nombre de remarques peuvent être faites à propos d'une telle procédure :

- cette procédure permet aux deux machines de travailler simultanément sans obliger le Varian à de nombreux accès disque supplémentaires.
- Si N est le nombre d'accès disque nécessaire pour traiter le fichier et M le nombre d'accès pour effectuer le transfert du fichier, cette procédure est plus intéressante que la première solution dans le cas ou $2\,(M+1) < N$.

- Il est possible de ne transférer qu'une partie du fichier à traiter si on le désire.

La procédure décrite ci-dessus ne convient pas pour des fichiers volumineux accédés de façon "random", de même que pour les fichiers communs aux deux machines. Dans ces deux cas, on utilisera plutôt une des deux premières solutions. Par contre pour des fichiers séquentiels, cette solution est souvent intéressante en particulier lorsqu'il s'agit de traitement de programme (compilation).

De façon plus générale, il appartiendra à l'utilisateur du Varian de choisir une des trois méthodes possibles. Le choix se fera en fonction du traitement à effectuer, de la longueur du fichier et de l'occupation du GE 58.

Nous avons vu dans ce chapitre comment il est possible d'utiliser les disques à partir du Varian sans gêner le GE 58. Nous avons vu qu'il est possible de créer, de rechercher et de supprimer un fichier à partir du Varian. Le type de gestion de l'espace disque assuré par le GE 58 nous a contraint à prendre des solutions qui peuvent paraître lourdes mais qui permettent aux deux machines de travailler simultanément.

+

+ +

III. TERMINAUX VIRTUELS

Nous avons vu que le GE 58 possède un certain nombre de terminaux que le Varian ne possède pas, comme par exemple un lecteur de cartes, une imprimante rapide, un perforateur. Il serait intéressant que le Varian puisse utiliser lui aussi ces terminaux.

Il est évident que nous n'avons utilisé, pour la réalisation de cette idée, que les moyens qui étaient à notre disposition à savoir la liaison GE 58 - VARIAN via les disques. Par conséquent, il n'est pas question que le Varian puisse utiliser directement les terminaux ; seul le GE 58 est relié physiquement aux terminaux, et c'est donc lui qui effectuera toutes les entrées-sorties. La seule possibilité pour le Varian est de considérer le GE 58 comme un terminal "super intelligent" capable d'effectuer des transactions avec les terminaux et de traiter les messages pour les mettre sur un disque, de façon à ce qu'ils soient accessibles pour le Varian.

Pour le Varian les terminaux n'existent donc pas physiquement mais virtuellement. Ce chapitre explicite plus amplement cette réalisation.

1. ENONCE DU PROBLEME

Quelle que soit la machine que l'on considère, il y a pour elle deux solutions possibles :

- soit elle écrit des informations sur le disque,
- soit elle lit les informations sur le disque.

Dans le premier cas, il faut savoir où écrire, c'est-à-dire dans quelle zone du disque écrire.

Dans le deuxième cas il faut pouvoir retrouver toutes les informations qui constituent l'ensemble logique qui doit être traité.

Toutefois un certain nombre de différences existe entre la position du GE 58 et du Varian.

Pour le GE 58 le traitement consiste :

- d'une part à prendre les informations venant des terminaux et à les écrire sur le disque ;
- d'autre part à lire sur le disque les informations pour les transmettre vers les terminaux.

Pour le Varian le traitement consiste à lire les informations se trouvant sur le disque lorsqu'il le désire et à écrire sur le disque lors de la production d'une sortie.

Pour bien comprendre la réalisation qui a été faite il faut tenir compte en outre que :

- d'une part nous étions soumis aux contraintes exposées plus haut
- d'autre part que nous avions à faire à des fichiers d'entrée-sortie ayant les particularités suivantes :

  . ce sont des fichiers séquentiels

. il peut en exister plusieurs à la fois

. ils sont de longueurs très variables et on ignore à priori leurs longueurs

. ils ont des durées de vie variables mais généralement relativement courtes.

Ces deux derniers points nous ont conduit assez rapidement à éliminer la possibilité de considérer ces fichiers d'entrée-sortie comme des fichiers ordinaires qu'on ajouterait dans la T R F en suivant la procédure normale.

2. REALISATION

2.1. Côté GE 58

Une des contraintes qui nous est imposée est de ne pas modifier le système d'exploitation du GE 58. Cela implique que pour assurer les entrées-sorties du Varian il faut en fait écrire un programme qui sera chargé manuellement par l'utilisateur du Varian. Ce programme assurera toutes les transactions entre le disque et les terminaux et réciproquement. Toutefois on remarque que même pour les transactions allant du disque vers les terminaux il est nécessaire qu'au départ il y ait une intervention manuelle, alors que conceptuellement ceci ne serait pas obligatoire.

2.2. Côté Varian

Pour le Varian le problème est relativement simple. Toute entrée est en fait ramenée à une lecture sur le disque alors que toute sortie consiste à écrire sur le disque et ce, quels que soient les terminaux que l'on utilise.

2.3. Procédure Générale

Comme nous l'avons déjà dit précédemment, il n'est pas très réaliste de considérer les fichiers d'entrée-sortie comme des fichiers ordinaires que l'on inscrit dans la T R F.

Par contre, il est possible de concevoir un fichier spécial que nous appellerons SPOOL, avec un système de gestion spécial qui en assurera l'organisation interne. De plus afin d'éviter les problèmes de déplacement de ce fichier, il est mis en début de disque.

Pour mieux comprendre le système de gestion supposons que nous voulions faire faire par le Varian le travail suivant :

> Il s'agit de rajouter dans un fichier des informations se trouvant sur des cartes perforées, puis d'imprimer tout ou une partie du fichier.

Les cartes sont lues par le GE 58 qui écrit leur contenu sur le disque. Cette opération nécessite une intervention manuelle car il faut charger le programme du GE 58 et placer les cartes. Lorsque toutes les cartes sont écrites sur le disque, dans une zone d'entrée, le Varian peut accèder à son tour à ces informations et les traiter.

On voit immédiatement que deux problèmes se posent :

- d'une part au moment de l'écriture faite par le GE 58, il faut connaître les secteurs disponibles sur le disque.

- D'autre part à la lecture il faut savoir où débutent les informations et quels sont les secteurs à lire.

Pour réaliser l'impression du fichier on suivra la procédure suivante :

Le Varian écrit le fichier dans une zone de sortie sur disque. Il faut que le GE 58 lise le contenu de la zone et le transfère vers l'imprimante. Dans ce cas-ci, se pose pour le Varian les mêmes problèmes que pour le GE 58. De plus, il est nécessaire d'intervenir manuellement pour charger le programme du GE 58 qui assurera le transfert du disque vers le terminal.

On voit donc sur cet exemple que le système de gestion du fichier SPOOL doit remplir au moins deux fonctions :

- au moment de l'écriture dans le fichier SPOOL il faut qu'il donne l'adresse du secteur à écrire,
- à la lecture il faut qu'il donne l'adresse du secteur à lire.

En plus de ces fonctions qui apparaissent nettement dans l'exemple, il est nécessaire que le système de gestion récupère les secteurs rendus disponibles.

La récupération des secteurs libres doit être relativement rapide car les fichiers entrée-sortie sont destinés en principe à ne servir qu'une fois.

De plus, il faudra tenir compte dans l'algorithme général du problème de mise à jour simultanée. Pour exposer le système de gestion nous supposerons d'abord qu'il n'y a qu'une machine qui écrit dans le fichier SPOOL.

Pour connaître l'adresse du 1er secteur dans lequel il est possible d'écrire, il suffit de maintenir une table que nous appellerons table principale (TP), donnant dans l'ensemble des secteurs, ceux qui sont libres et ceux qui ne le sont pas.

De même pour connaître l'ensemble des secteurs utilisés par un fichier à sortir, on lui associe une table (TA) analogue à la TP et qui donne l'ensemble des secteurs utilisés par ce fichier.

De plus comme dans la zone SPOOL il peut exister plusieurs fichiers simultanément. Une table donnera la correspondance entre le nom du fichier et la table de secteur (TA) qui lui est associée. Dans le système utilisé lors de l'application, ces différentes tables sont des tables de bits. Chaque bit représente un secteur et est positionné si le secteur est utilisé.
De plus, le nom du fichier et la table qui lui est associée sont contenus dans un même article. La longueur d'un article est toujours un nombre entier de secteurs.

Pour écrire dans la zone SPOOL l'algorithme est donc relativement simple et est donné par la fiture I. De même la procédure de lecture d'un fichier est donnée par la figure 2.

FIG I

Ecriture d'un fichier

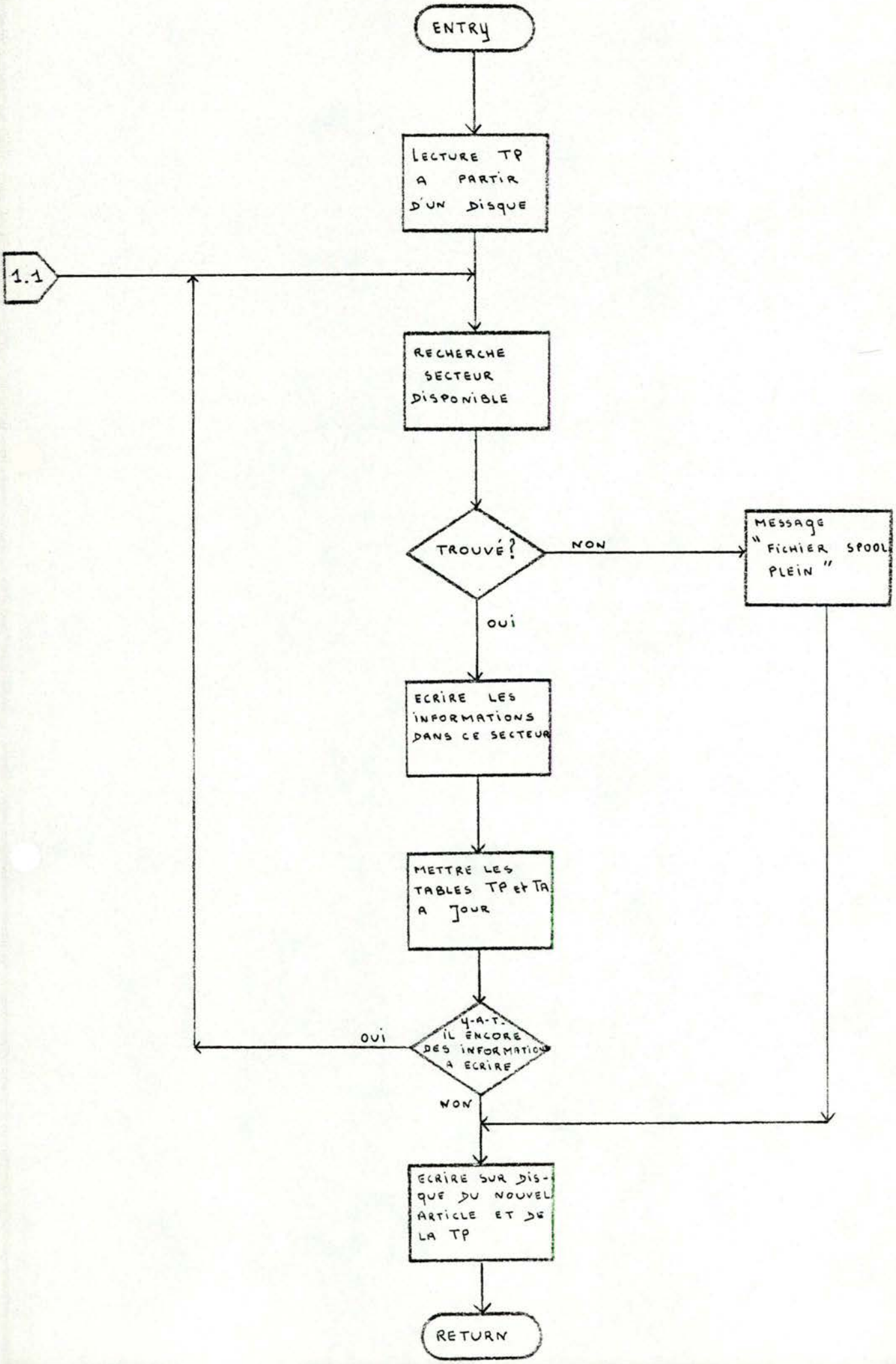
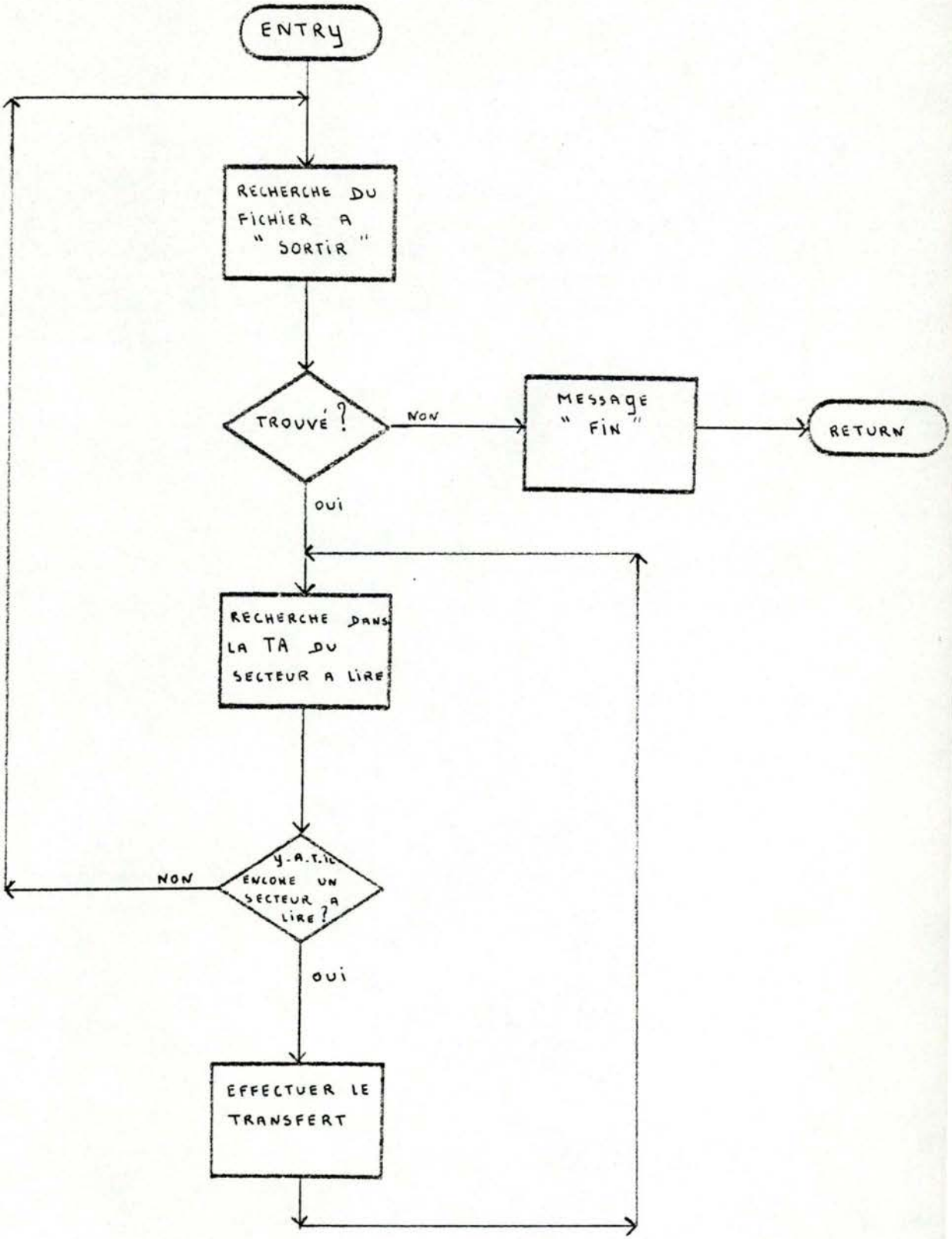


FIG 2

Lecture d'un fichier



On remarquera que dans cette procédure aucune mise à jour des tables n'est effectuée. Ceci ne nous permet pas de savoir quels sont les fichiers qui ont été lus et surtout les secteurs qui peuvent être réutilisés. Pour éviter les problèmes de mise à jour simultanée, seul sera ré-écrit l'article correspondant au fichier qui vient d'être lu. C'est pour cette raison qu'un article sera toujours un nombre entier de secteurs.

Cependant, cette ré-écriture ne suffit pas. Il faut également récupérer les secteurs qu'occupaient les fichiers qui ont été lus. Ces secteurs doivent réapparaître libres dans la TP. Pour cela, il suffit de réaliser un "ou exclusif" entre la TP et la TA du fichier à supprimer. Cette procédure nécessite évidemment une ré-écriture finale de la TP. Elle ne pourra donc être faite que par la machine qui écrit. Pour que celle-ci connaisse les fichiers à supprimer, il suffit que la machine qui a lu, positionne un "flag" dans l'article. De façon à généraliser cette notion de "flag" dans l'application, nous avons utilisé le nombre d'exemplaires désirés. Ce nombre est initialisé par la machine qui écrit, et est décrémenté par la machine qui lit. Lorsque ce nombre devient nul, le fichier peut être supprimé. On peut donc compléter les algorithmes 1 et 2 de façon à ce qu'ils tiennent compte de ce qui vient d'être dit. Nous obtenons alors les algorithmes 3 et 4.

Nous remarquons que dans l'algorithme 3 il est nécessaire d'effectuer deux ré-écritures pour terminer : celle de la TP et celle de l'article qui vient d'être créé.

FIG 3

Modification de la procédure d'écriture d'un fichier

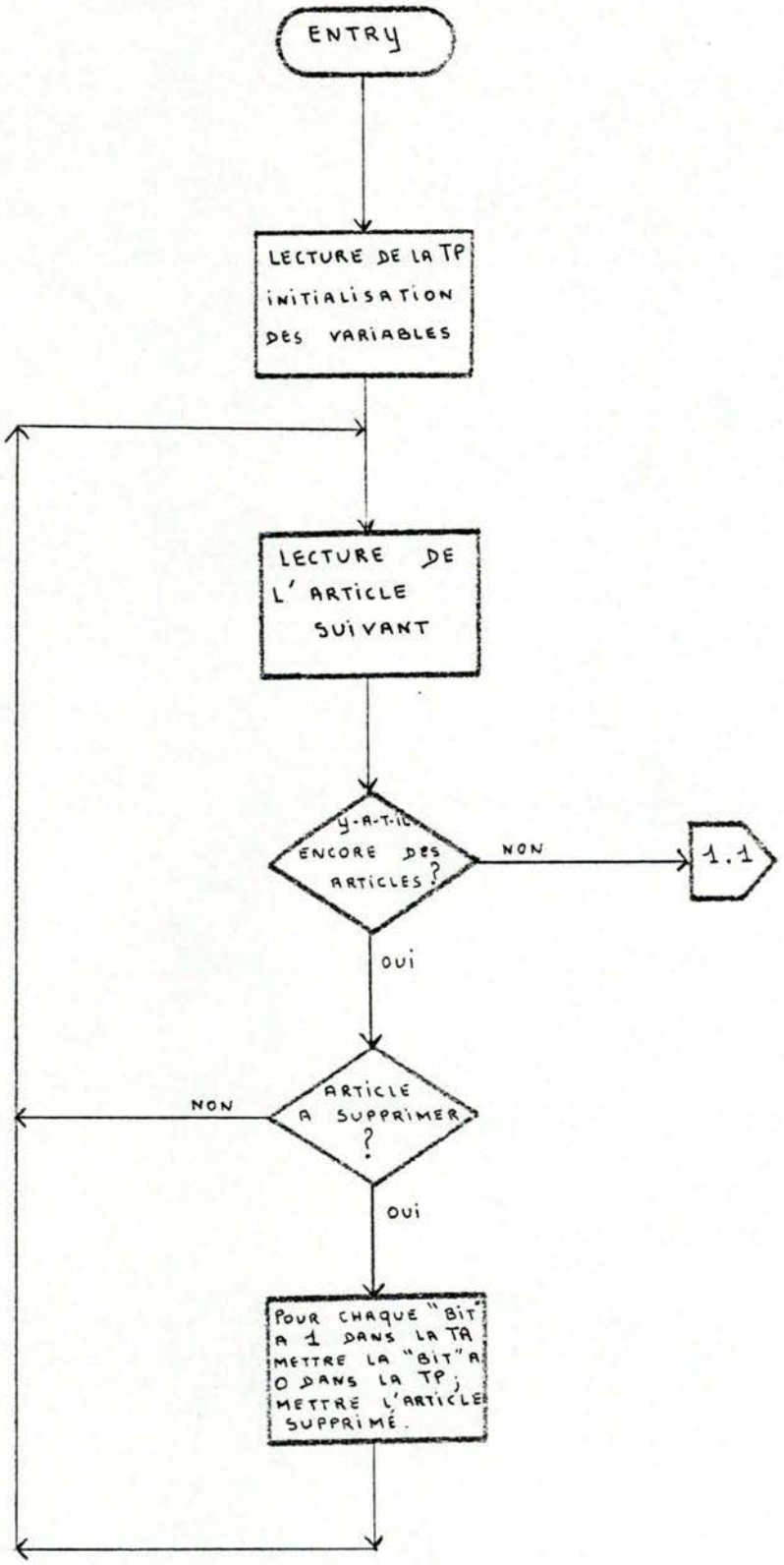
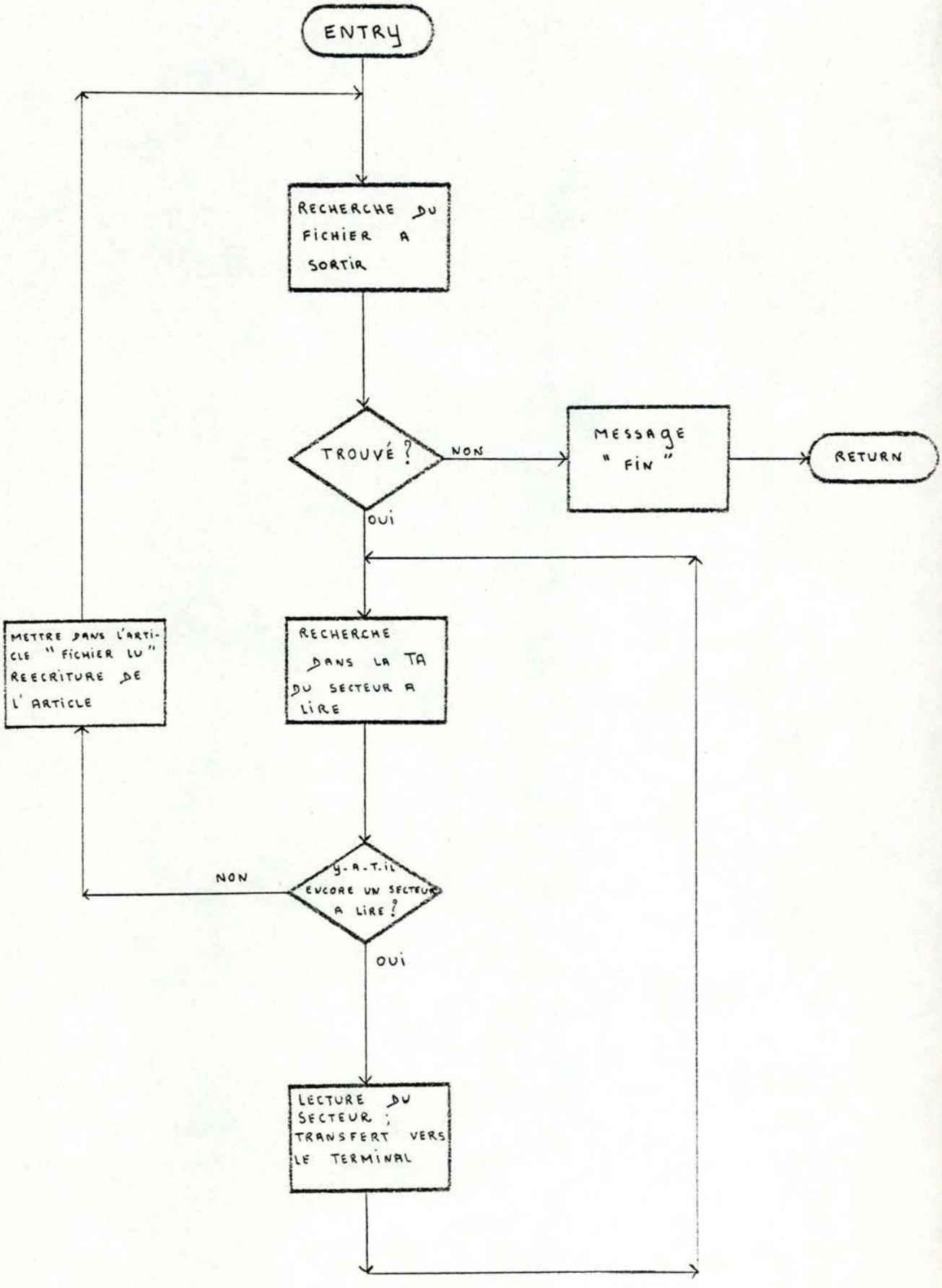


FIG 4

Modification de la procédure de lecture d'un fichier



Il est évident que la concordance entre la TP et l'ensemble des TA doit toujours être respectée. Si pour une raison ou une autre (par exemple "down" de la machine), la deuxième ré-écriture ne peut être faite alors que la première a été effectuée, cette concordance n'est plus respectée.

Il convient alors, avant toute opération, de restaurer les différentes tables. Pour cela, il a été prévu que chacune des tables TA et la table TP contiennent le nombre de secteurs employés. Il y aura concordance si et seulement si la somme des secteurs utilisés dans l'ensemble des TA est égale au nombre de secteurs donné par la TP.

Si la concordance n'est pas respectée, on reconstruit la TP à partir des TA. Celles-ci étant toujours correctes. L'annexe 2 donne les algorithmes complets des procédures de lecture et d'écriture et une description des tables TP et TA.

Tout ce que nous venons de dire est valable parce que nous avons supposé qu'une seule machine écrivait dans le fichier SPOOL. Dans le cadre de la réalisation qui nous est proposée, en fait ce sont les deux machines qui peuvent écrire. Dès lors se pose le problème des mises à jour simultanées des tables et en particulier de la TP.

Comme nous n'avons pas de solution satisfaisante pour résoudre ce problème, il a été décidé de dissocier les tables, chaque machine ayant sa TP et ses TA, ainsi que les secteurs que chaque machine peut employer.

Ce chapitre nous a montré comment il était possible de permettre au Varian d'utiliser les terminaux du GE 58. Nous sommes bien conscients que les procédures décrites ne sont pas idéales. En particulier le fait de devoir intervenir manuellement pour assurer les transferts disque-terminaux peut être contraignant pour les utilisateurs. De même le fait de dissocier les secteurs que chaque machine peut écrire peut être un inconvénient.

Il serait plus intéressant de pouvoir réutiliser les secteurs ayant servi à l'entrée et qui n'ont plus d'utilité pour mettre les informations à sortir.

Il est toutefois intéressant de remarquer que pour l'utilisateur du Varian rien n'est différent d'un système ordinaire. L'utilisateur lit à partir d'un lecteur de carte ou écrit sur une imprimante. Le fait que les informations sont lues ou sont écrites sur le disque est totalement ignoré par l'utilisateur. C'est pour cette raison que nous pouvons parler de terminaux virtuels pour le Varian.

+

+ +

IV. CONCLUSION

Les deux chapitres précédents nous ont montré comment il est possible pour le GE 58 et le Varian de travailler simultanément. Un certain nombre de points ont été réalisés en particulier pour le premier chapitre : le programme de création de fichier est entièrement fait. Il est donc possible à partir du Varian de créer tous les types de fichiers que peut créer le GE 58 y compris les librairies et sous-librairies. Ce programme comprend également une routine de consultation de la T R F qui peut être employée telle quelle pour les procédures de consultation et de suppression qui n'ont pas été poussées plus loin. En ce qui concerne les terminaux virtuels le projet en est resté au stade de l'analyse.

Bien que l'ensemble des procédures décrites dans ces deux chapitres soient réalisables, elles paraissent relativement lourdes (emploi de fichier intermédiaire, obligation d'avoir un fichier réservé spécialement aux entrées-sorties etc...) et ce à cause des contraintes qui nous étaient imposées à savoir :

- la non existence de primitives de synchronisation
- l'existence d'un DOS sur le GE 58 et l'impossibilité de le modifier.

Il serait intéressant de ne plus tenir compte de ces contraintes, et de voir rapidement quelles auraient été les solutions qui auraient pu être choisies.

Pour commencer nous supposerons :

- qu'il existe des primitives de synchronisation qui permettent de protéger chacun des secteurs du disque (1)
- que le DOS du GE 58 a été modifié de façon à tenir compte de ces primitives de synchronisation.

a) Partage de l'espace disque

Il est évident que l'hypothèse que nous venons de faire permet au Varian de manipuler la TRF sans aucun inconvénient. En particulier les problèmes posés par une séquence du type LE_{GE} LE_{VA} EC_{VA} EC_{GE} n'existent plus et par conséquent le Varian peut effectuer les opérations de création et de suppression logiques des fichiers de façon normale, et sans qu'il y ait risque de perdre l'information comme dans le cas précédent.
La procédure de consultation elle, serait une simple recherche, la seule contrainte étant qu'elle ne peut être faite pendant que le GE 58 modifie l'organisation du disque.

(1) Il ne nous importe pas de savoir ici comment seraient réalisées ces primitives. Ce qui est important c'est que le programmeur ait à sa disposition un outil qui lui permette de protéger un ou plusieurs secteurs, interdisant de ce fait l'accès au(x) secteur(s).

b) Terminaux virtuels

Pour que le Varian puisse utiliser les terminaux du GE 58, nous avons été amenés à créer un fichier spécial de SPOOL. Ce fichier est dû d'une part au mode de gestion de l'espace disque assuré par le GE 58 et d'autre part au fait que les fichiers de SPOOL ont une durée de vie très courte.
Les deux hypothèses que nous avons faites au début de ce chapitre ne changent évidemment rien à ce problème là.
Le seul gain que nous ayons réside dans le fait que pour gérer ce fichier SPOOL une seule table TP suffit et que les différentes modifications des TP et des TA peuvent se faire indifféremment par une des deux machines.

Cependant, si l'on suppose maintenant que le DOS ne déplace pas les fichiers, nous pouvons alors considérer les fichiers SPOOL comme des fichiers ordinaires que l'on ajouteraient sur le disque en suivant la procédure normale de création de fichier. De même la suppression du fichier SPOOL suivrait la procédure normale de suppression. Cette solution permettrait d'éviter de conserver un fichier SPOOL relativement grand et qui est rarement employé au maximum.

En conclusion, on peut dire que la réalisation de la liaison GE 58 - VARIAN est faisable bien que les solutions acceptables soient parfois lourdes. Il est évident que si nous avions eu à notre disposition :

- d'une part, des primitives de synchronisation,
- et d'autre part un système d'exploitation qui ne déplace pas les fichiers.

il aurait été possible d'effectuer la réalisation de façon plus aisée et surtout d'avoir une liaison qui permet au Varian et au GE 58 de travailler simultanément sans pour autant être pénalisés.

De plus, la présence de primitives de synchronisation permettrait aux deux machines de dialoguer véritablement ce qui est difficilement réalisable avec le système tel qu'il est.

+

+ +

ANNEXES

ANNEXE I

La T R F tient sur 10 secteurs. Elle est constituée d'articles de longueur fixe de 16 bytes (voir figure I) et il est donc possible de mettre 18 articles par secteur.

Cette table est remplie séquentiellement au fur et à mesure des créations de fichiers. Lors de leur suppression, la T R F est compactée vers le haut de façon à ne pas créer de discontinuités dans son occupation.

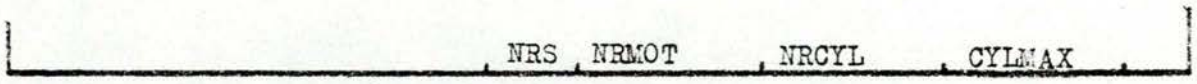
On peut décrire comme suit les différentes opérations possibles :

- création de fichiers

  Cette opération consiste à mettre l'article correspondant au fichier qui a été créé à la fin de la T R F. L'adresse de l'article est donnée par le pointeur se trouvant dans l'article I de la T R F. L'adresse début du fichier est celle donnée par le pointeur NRCYL.
  Il est à noter qu'il est possible de créer des librairies. Lors de leur création, on réserve tout l'espace nécessaire même s'il n'est pas utilisé immédiatement. Les "sous-librairies" créées ultérieurement occuperont cet espace au fur et à mesure et à chaque création un article sera ajouter à la T R F. Cet article devra obligatoirement suivre les articles correspondants à la librairie.

DESCRIPTION DE LA TRF

Article I



NRS: numéro de secteur dans la TRF (I byte)

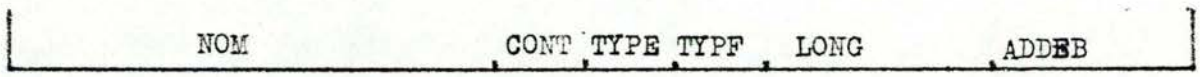
NRMOT: adresse mot dans le secteur (2 bytes)

Pointeur vers le premier article libre dans la TRF

NRCYL: adresse du premier cylindre libre (2 bytes)

CYLMAX: nombre de cylindre maximun (2 bytes)

Autres articles



NOM: nom du fichier (8 bytes)

CONT: contenu du fichier (I byte): = I : data file
2 : source programme
3 : object programme
4 : software programme
5 : fonction et routine
8 : code créé par compilateur

TYPE: type d'enregistrement (I byte): = I : cylindre
2 : piste
3 : secteur

TYPF: type de fichier (I byte): = I : fichier ordinaire
2 : librairies
3 : sous-librairie

LONG: longueur du fichier (20 bits)

ADDEB: adresse début du fichier (20 bits)

FIG I

- Consultation

Cette opération se résume à un "scanning" de la T R F jusqu'à ce que l'article ayant le même nom que celui donné soit trouvé. On obtient alors l'adresse début du fichier ainsi que sa longueur.

- Suppression logique

Cette opération consiste à sur-écrire le nom du fichier par les caractères hexadecimales "FF". Le fichier ne peut plus dès lors être retrouvé.

- Suppression physique

Tous les fichiers dont le nom commence par "FF" peuvent être supprimés. Le disque est réorganisé en fonction de ces suppressions et compacté vers le début ainsi que la T R F.

ANNEXE II

TERMINAUX VIRTUELS

1. Description des tables

Chaque machine possède une table constituée de plusieurs articles de même longueur. Chaque article contient une partie fixe d'une longueur de 12 bytes et que nous appellerons "identifiant" ; l'autre partie est considérée comme un "bitstring" et sa longueur est un paramètre du système de gestion. C'est cette partie que nous avons appelé TP s'il s'agit du premier article et TA dans les autres cas. Le nombre de bit dans ces tables est le nombre de secteur qu'il existe dans le fichier SPOOL. Les bits positionnés à I désignent les secteurs utilisés.

La TP donne l'image des secteurs utilisés par l'ensemble des fichiers alors que chaque TA donne l'image des secteurs utilisés par le fichier dont elle dépend.

Les fig. 1 et 2 donnent une image du fichier SPOOL, des différents articles ainsi qu'une description des identifiants des articles.

On notera que dans l'identifiant de chacune des TP se trouve un certain nombre de paramètres qui sont initialisés par un programme spécial. De plus, on remarquera qu'au début, les deux TP sont complémentaires c'est-à-dire que tout bit se trouvant à I dans une des TP se trouve à 0 dans l'autre et réciproquement.

FICHIER SPOOL

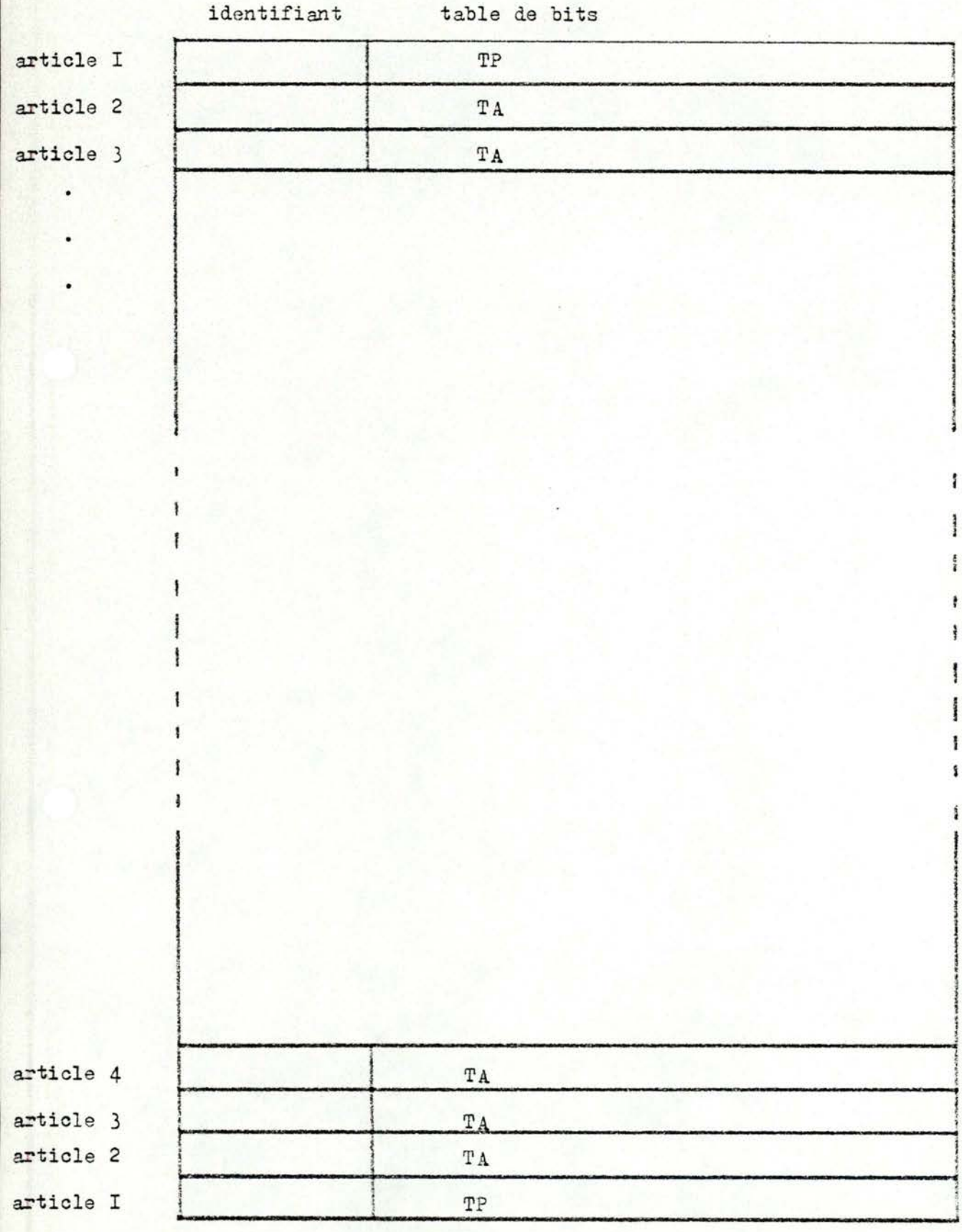
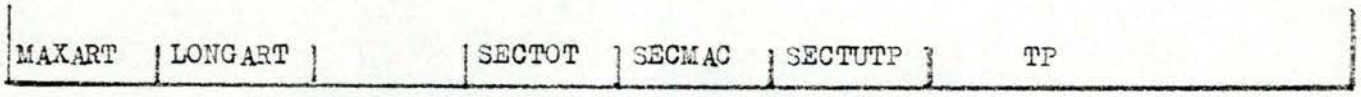


FIG I

DESCRIPTION DES IDENTIFIANTS

Article I



MAXART: nombre maximun d'articles possibles (2 bytes)

LONGART: nombre de secteur d'un article (2 bytes)

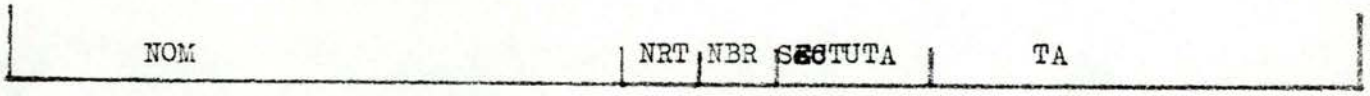
SECTOT: nombre de secteur dans le fichier SPOOL

SECMAC: nombre de secteur du fichier SPOOL attribué à la machine (2 bytes)

SECTUTP: nombre de secteur utilisé (2 bytes)

TP: table de bits (I bit par secteur : si bit=o == secteur non utilisé
si bit=I == secteur utilisé)

Article



NOM: nom du fichier (8 bytes)

NRT: numéro du terminal (I byte)

NBR: nombre d'exemplaires (I byte)

SECTUTA: nombre de secteur utilisé par le fichier (2 bytes)

TA: table de bits (I bit par secteur : si bit=o == secteur non utilisé
si bit=I == secteur utilisé)

FIG 2

Ceci permet de faire varier le nombre de secteurs possibles pour chaque machine sans pour autant devoir changer la longueur totale du fichier SPOOL.

2. Procédure d'écriture et de lecture

Les procédures d'écriture et de lecture des fichiers sont décrites par les organigrammes des figures 3 et 4. La procédure d'écriture peut être décomposée en trois parties :

- une partie de contrôle du fichier SPOOL (fig 3a) :
  - . contrôle de concordance entre la TP et l'ensemble des TA. S'il y a discordance il convient de reconstruire la TP à partir des TA (fig 3d)
  - . récupération des secteurs libres (fig 3d).
- une partie de construction et de contrôle de l'identifiant (fig. 3 b) :
  - . contrôle de place disponible pour mettre l'article
  - . contrôle sur le nom pour ne pas créer deux fichiers du même nom.
- une partie de construction des tables TA et TP (fig 3c) et d'écriture des informations.

Procédure d'entrée (contrôle du fichier SPOOL)

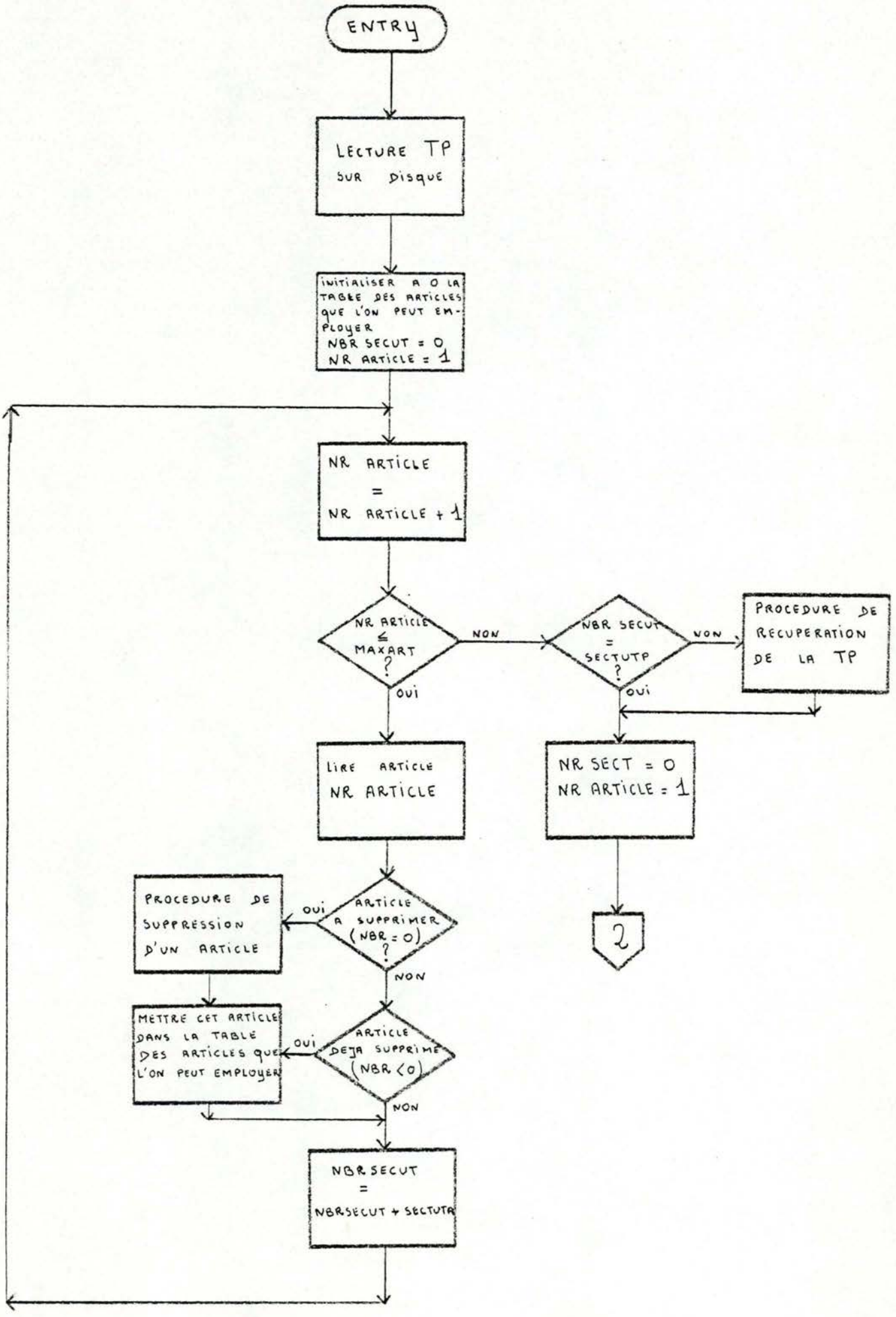


FIG 3a

Procédure d'entrée (contrôle des identifiants)

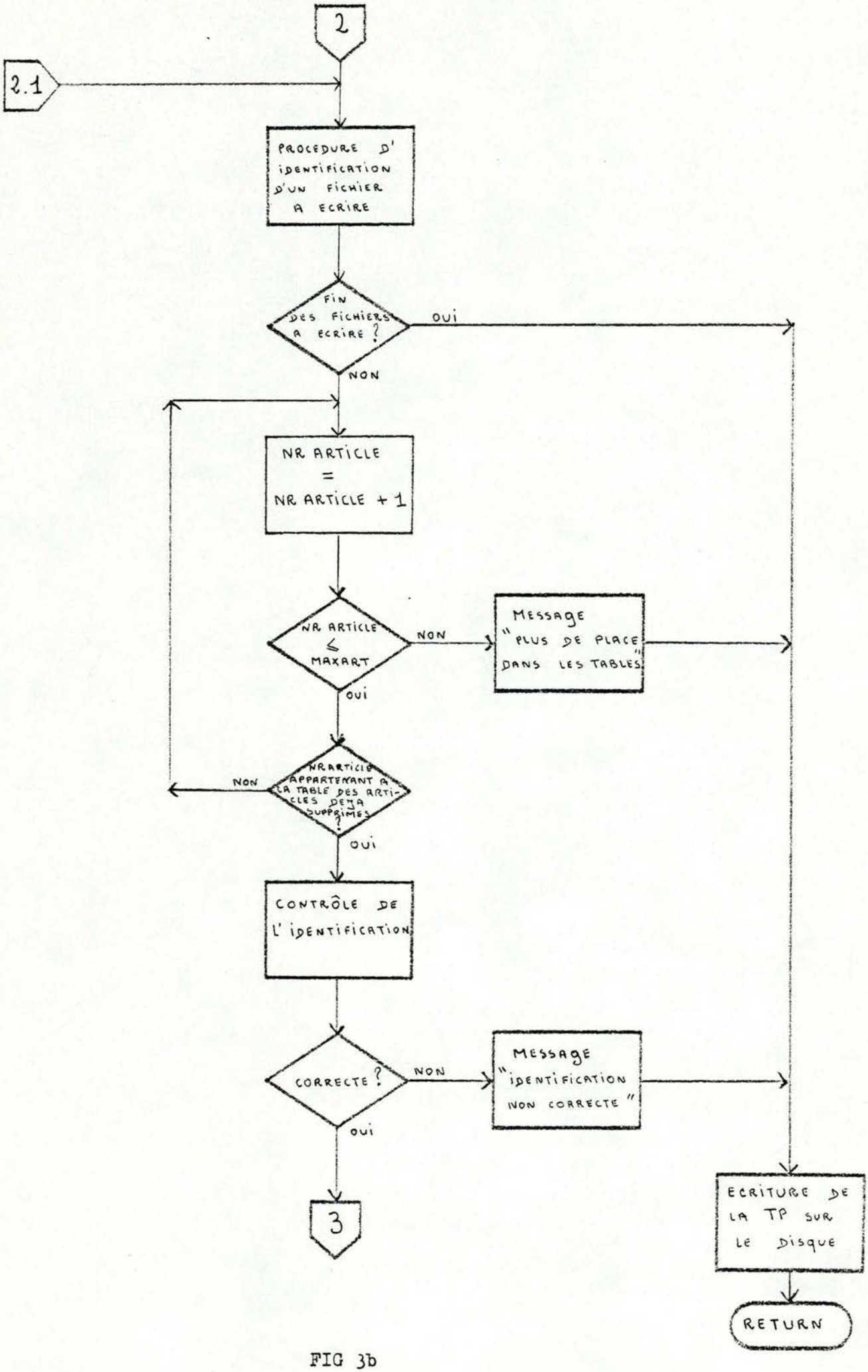


FIG 3b

Procédure d'entrée

(entrée des informations)

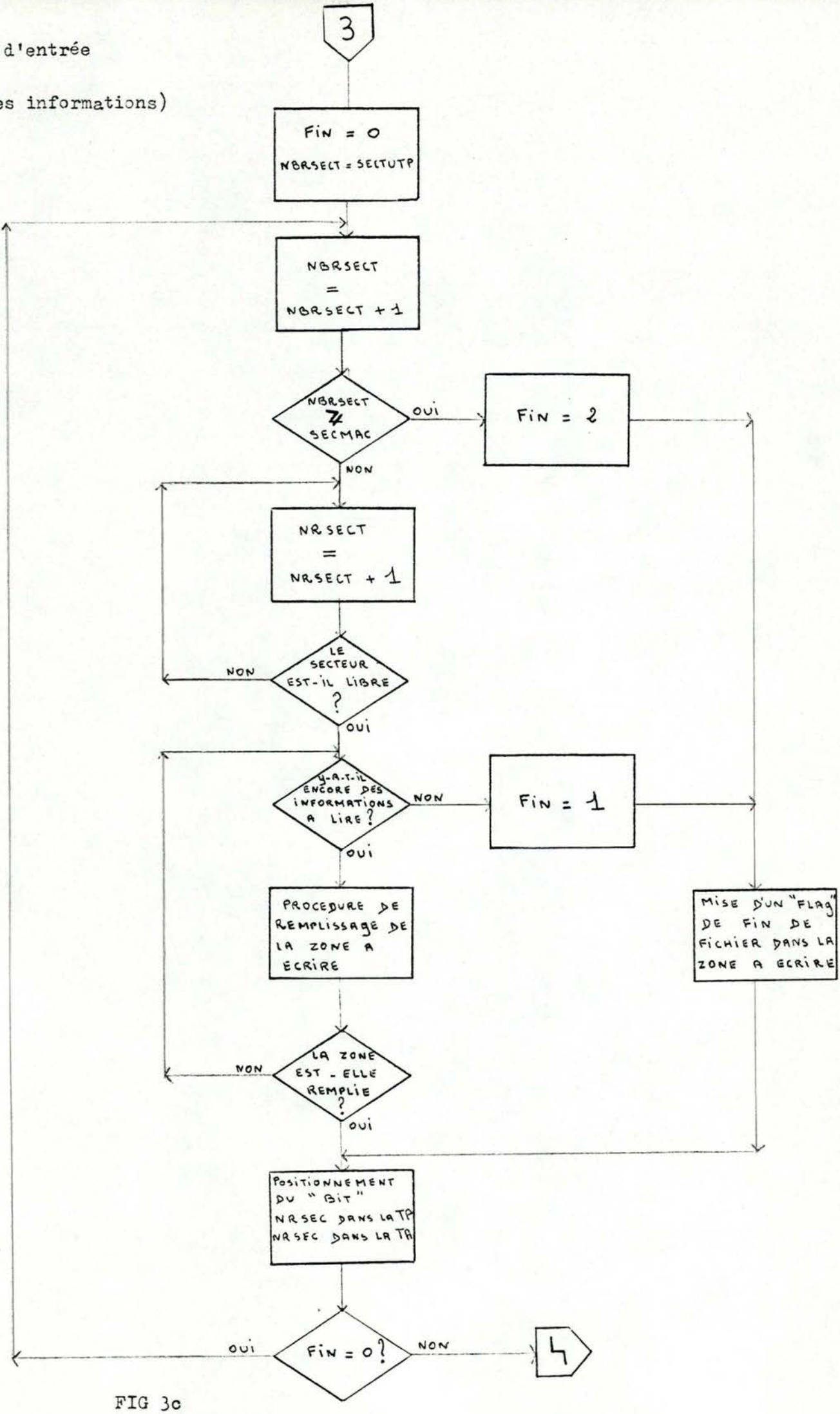


FIG 3c

Procédure d'entrée (entrée des informations)

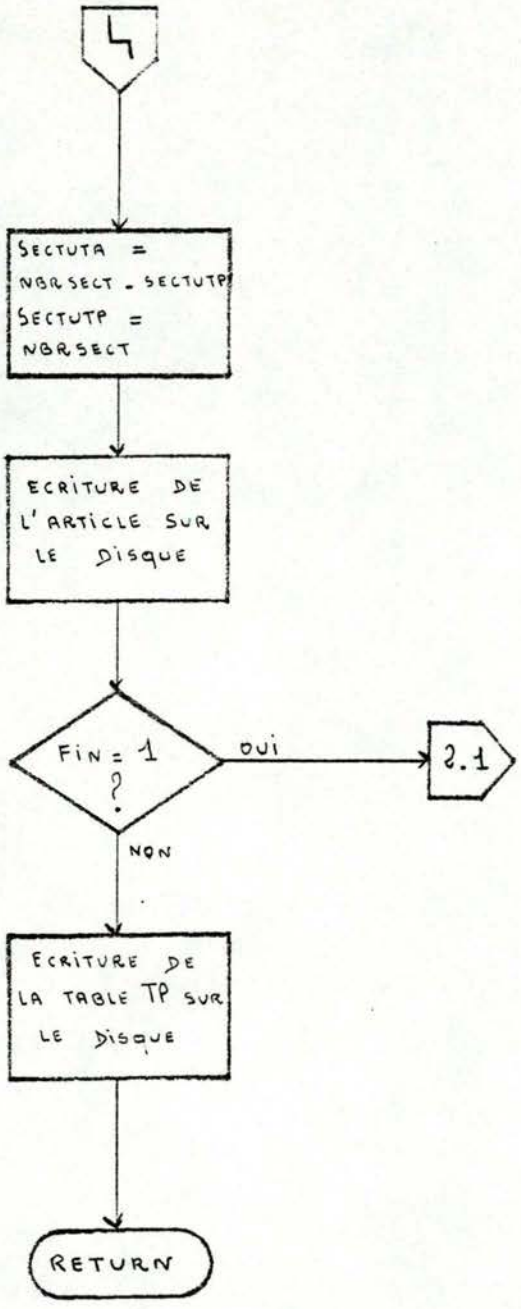


FIG 3c'

Procédure d'entrée (suppression d'un fichier)

Paramêtresreçus en entrée:

- adresse mémoire central de l'articl à supprimer
- adresse mémoire central de la TP
- adresse disque del'article

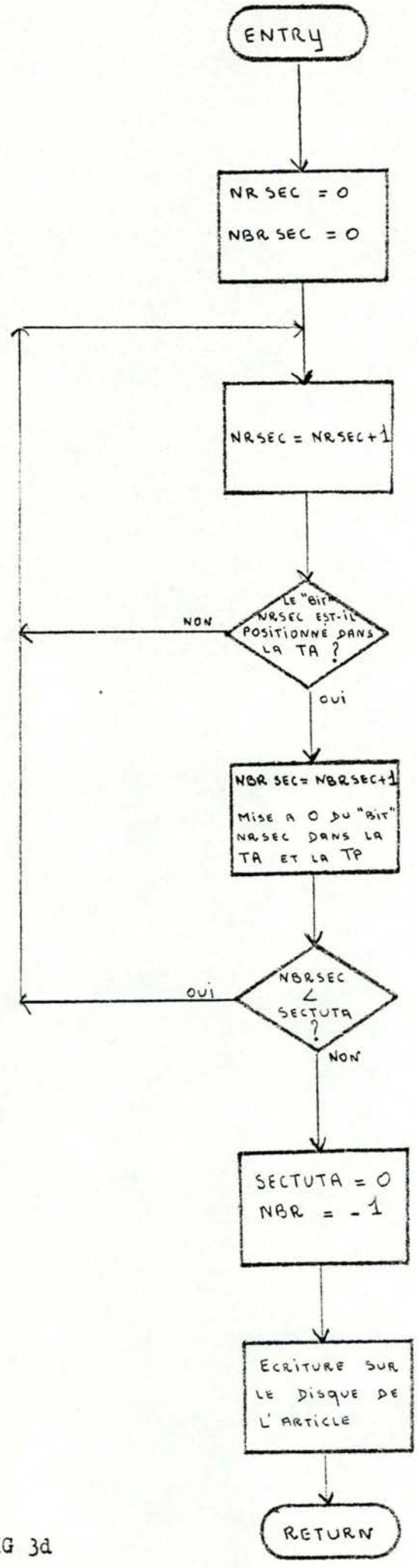


FIG 3d

Procédure d'entrée (récupération de la TP)

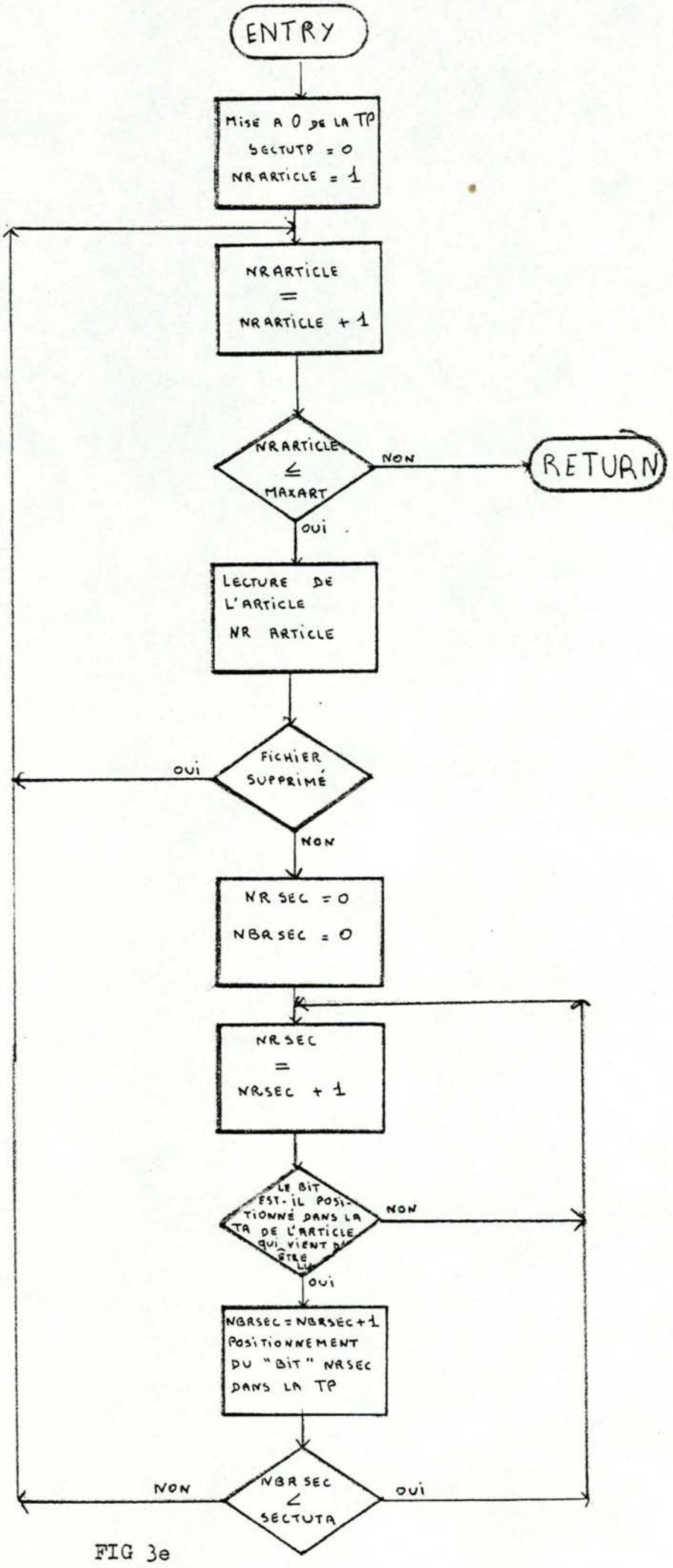


FIG 3e

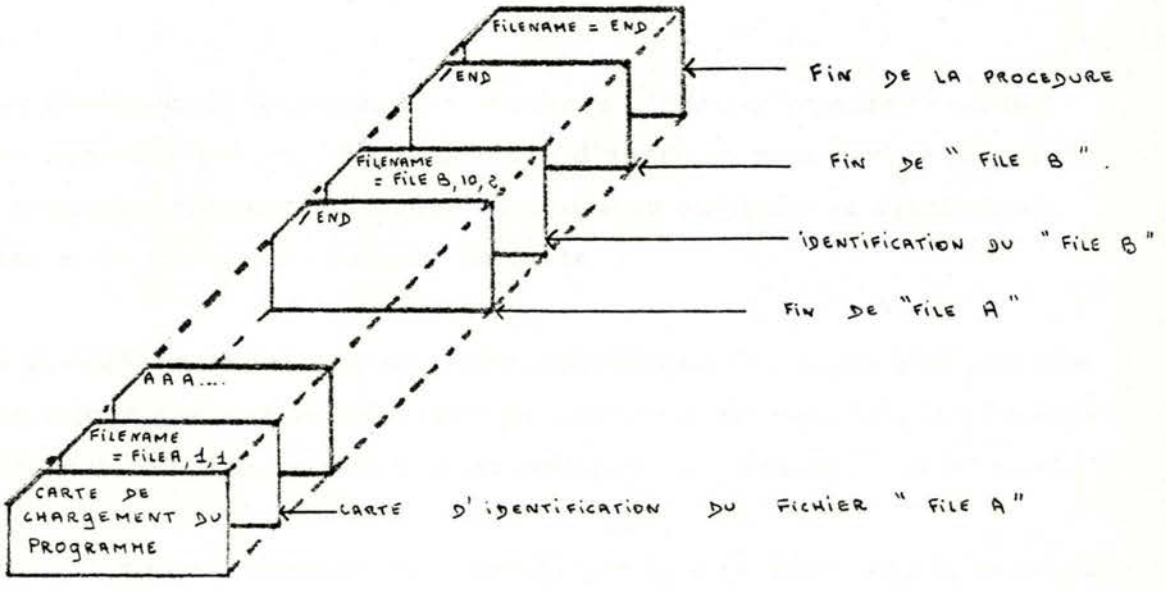

FILENAME = END
/END
FILENAME
= FILE B, 10, 2,
/END
A A A....
FILENAME
= FILEA, 1, 1
CARTE DE
CHARGEMENT DU
PROGRAMME
FIN DE LA PROCEDURE
FIN DE " FILE B ".
IDENTIFICATION DU " FILE B "
FIN DE "FILE A "
CARTE D'IDENTIFICATION DU FICHIER " FILE A "

FIG 3

Procédure de sortie

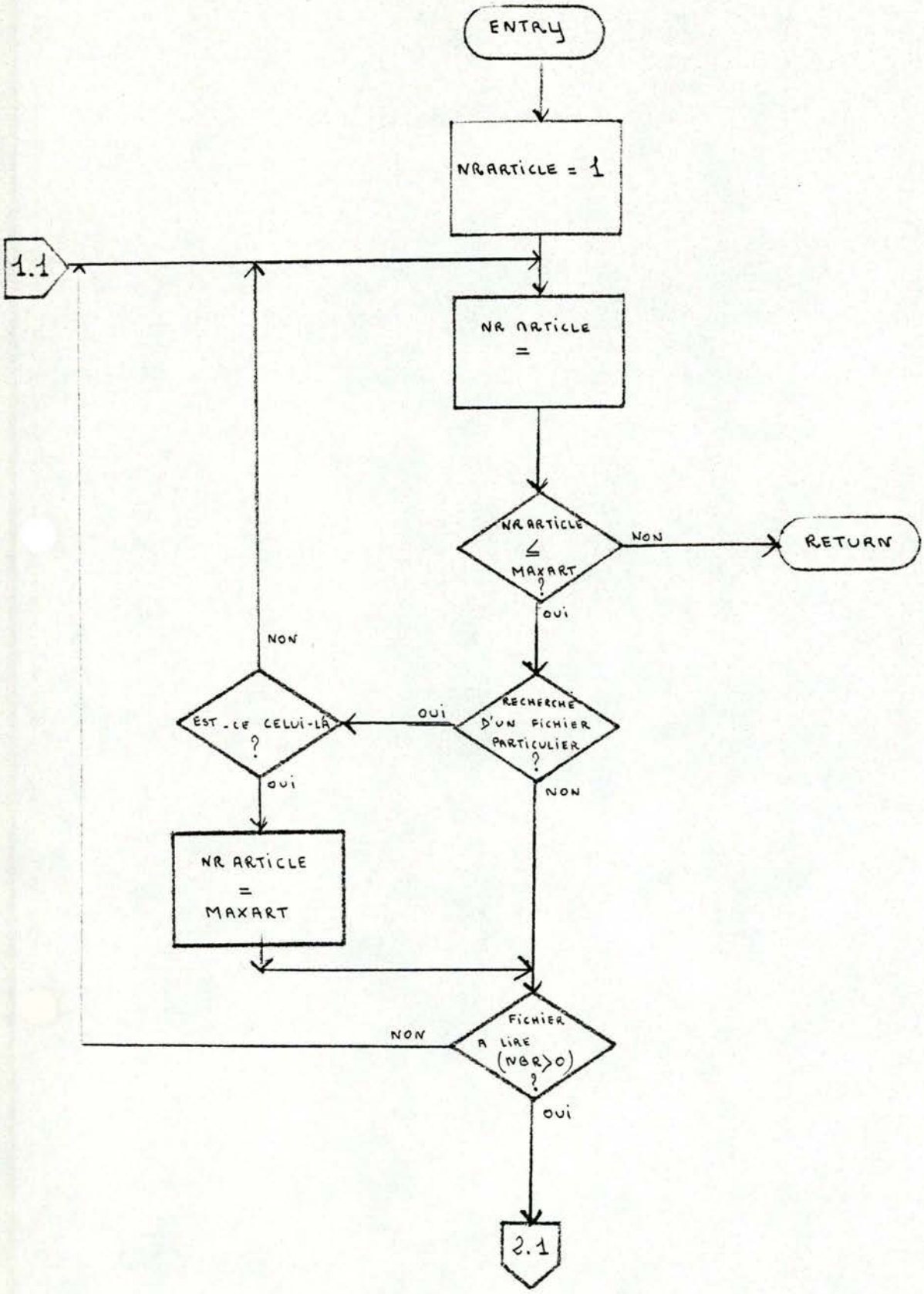


FIG 4

Procédure de sortie

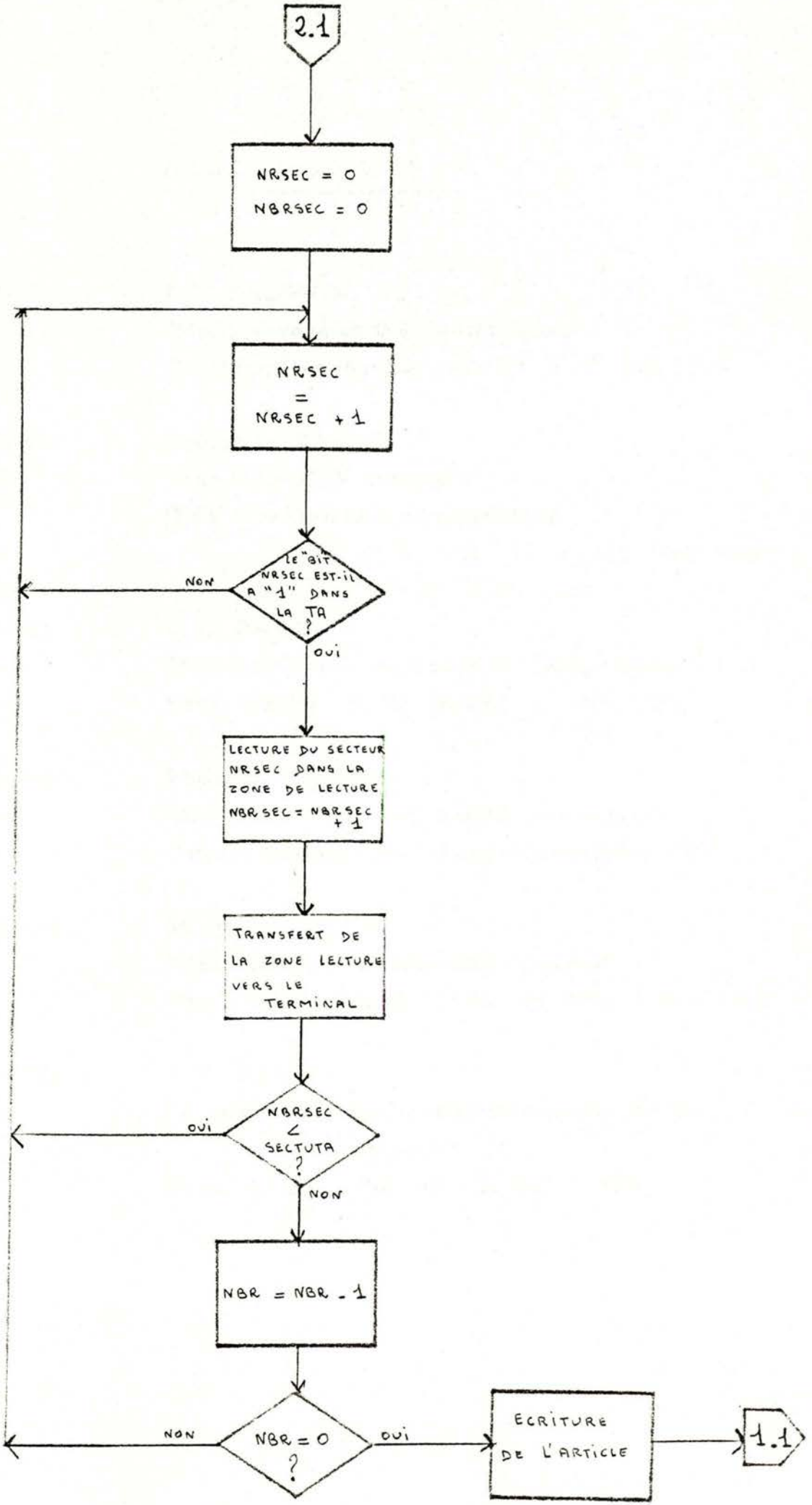


FIG 4'

(HIGB 73) L. C. HIGBIE
"Supercomputer Architecture"
Computer Vol. 16 n° 12 p. 48 déc. 1973

(HOPK 71) ALBERT L. HOPKINS
"A fault tolerant information processing concept for space vehicles"
IEEE Transactions on computers
Vol. C-20 p. 1394 novembre 1971

(JOSE 74) EARL C. JOSEPH
"Innovations in heterogeneous and homogeneous distributed function architecture"
Computer p. 17 mars 1974

(KOCZ 71) Louis J. KOCZELA
"A three failure tolerant computer system"
IEEE transaction on computers
Vol. C-20 p. 1389 novembre 1971

(LALI 73) Théodore A. LALIOTIS
"Main memory technology"
Computer septembre 1973

(LORI 72) Harold LORIN
"Parallelism in hardware and software : real and apparent concurrency"
Prentice Hall 1972

(SEAR 75) B. C. SEARLE

"Microprocessor applications in multiple processor systems"

Computer Vol. 8 N° 10 p. 220 Octobre 1975

+

+ +