



HAL
open science

Partition syntaxique d'un système logique décrit en CASSANDRE

Patrick Liddell

► **To cite this version:**

Patrick Liddell. Partition syntaxique d'un système logique décrit en CASSANDRE. Réseaux et télécommunications [cs.NI]. Université Joseph-Fourier - Grenoble I, 1970. Français. NNT: . tel-00009474

HAL Id: tel-00009474

<https://theses.hal.science/tel-00009474>

Submitted on 13 Jun 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre

THESE

présentée à

LA FACULTE DES SCIENCES DE L'UNIVERSITE DE GRENOBLE

pour obtenir

LE GRADE DE DOCTEUR DE TROISIEME CYCLE

" Mathématiques Appliquées "

par

Patrick LIDDELL



**Partition syntaxique d'un système
logique décrit en CASSANDRE**



Thèse soutenue le 21 Mars 1970 devant la commission d'examen

Monsieur	J. KUNTZMANN	Président
Messieurs	C. BENZAKEN	Examineur
	L. BOLLIET	Examineur

70318

L I S T E D E S P R O F E S S E U R S

Doyen honoraire : Monsieur M. MORET
 Doyen : Monsieur E. BONNIER

PROFESSEURS TITULAIRES

MM.	NEEL Louis	Physique Expérimentale
	KRAVTCHENKO Julien	Mécanique Rationnelle
	CHABAUTY Claude	Calcul différentiel et intégral
	BENOIT Jean	Radioélectricité
	CHENE Marcel	Chimie Papetière
	FELICI Noël	Electrostatique
	KUNTZMANN Jean	Mathématiques Appliquées
	BARBIER Reynold	Géologie Appliquée
	SANTON Lucien	Mécanique des Fluides
	OZENDA Paul	Botanique
	FALLOT Maurice	Physique Industrielle
	KOSZUL Jean-Louis	Mathématiques
	GALVANI Octave	Mathématiques
	MOUSSA André	Chimie Nucléaire
	TRAYNARD Philippe	Chimie Générale
	SOUTIF Michel	Physique Générale
	CRAYA Antoine	Hydrodynamique
	REULOS René	Théorie des Champs
	BESSON Jean	Chimie Minérale
	AYANT Yves	Physique Approfondie
	GALLISSOT François	Mathématiques
Melle.	LUTZ Elisabeth	Mathématiques
MM.	BLAMBERT Maurice	Mathématiques
	BOUCHEZ Robert	Physique Nucléaire
	LLIBOUTRY Louis	Géophysique
	MICHEL Robert	Minéralogie et pétrographie
	BONNIER Etienne	Electrochimie et Electrometallurgie
	DESSAUX Georges	Physiologie animale
	PILLET Emile	Physique Industrielle-Electrotechnique
	YOCCOZ Jean	Physique Nucléaire théorique
	DEBELMAS Jacques	Géologie Générale
	GERBER Robert	Mathématiques
	PAUTHENET René	Electrotechnique
	MALGRANGE Bernard	Mathématiques Pures
	VAUQUOIS Bernard	Calcul Electronique
	BARJON Robert	Physique Nucléaire

MM.	BARBIER Jean-Claude	Physique
	SILBER Robert	Mécanique des Fluides
	BUYLE-BODIN Maurice	Electronique
	DREYFUS Bernard	Thermodynamique
	KLEIN Joseph	Mathématiques
	VAILLANT François	Zoologie et Hydrobiologie
	ARNAUD Paul	Chimie
	SENGEL Philippe	Zoologie
	BARNOUD Fernand	Biosynthèse de la Cellulose
	BRISSONNEAU Pierre	Physique
	GAGNAIRE Didier	Chimie Physique
Mme.	KOFLER Lucie	Botanique
MM.	DEGRANGE Charles	Zoologie
	PEBAY-PEROULA Jean-Claude	Physique
	RASSAT André	Chimie Systématique
	DUCROS Pierre	Cristallographie Physique
	DODU Jacques	Mécanique Appliquée I. U. T.
	ANGLES D'AURIAC Paul	Mécanique des Fluides
	LACAZE Albert	Thermodynamique
	GASTINEL Noël	Analyse numérique
	GIRAUD Pierre	Géologie
	PERRET René	Servo-mécanisme
	PAYAN Jean-Jacques	Mathématiques Pures

PROFESSEURS SANS CHAIRE

MM.	GIDON Paul	Géologie
Mme.	BARBIER Marie-Jeanne	Electrochimie
Mme.	SOUTIF Jeanne	Physique
	COHEN Joseph	Electrotechnique
	DEPASSEL R.	Mécanique des Fluides
	GLENAT René	Chimie
	BARRA Jean	Mathématiques Appliquées
	COUMES André	Electronique
	PERRIAUX Jacques	Géologie et Minéralogie
	ROBERT André	Chimie Papetière
	BIARREZ Jean	Mécanique Physique
	BONNET Georges	Electronique
	CAUQUIS Georges	Chimie Générale
	BONNETAIN Lucien	Chimie Minérale
	DEPOMIER Pierre	Physique Nucléaire-Génie Atomique
	HACQUES Gérard	Calcul numérique
	POLOUJADOFF Michel	Electrotechnique
Mme.	KAHANE Josette	Physique
Mme.	BONNIER Jane	Chimie
MM.	VALENTIN Jacques	Physique
	REBECQ Jacques	Biologie
	DEPORTES Charles	Chimie
	SARROT-REYNAULD Jean	Géologie
	BERTRANDIAS Jean-Paul	Mathématiques Appliquées
	AUBERT Guy	Physique

PROFESSEURS ASSOCIES

MM.	RODRIGUES Alexandre	Mathématiques Pures
	MORITA Susumu	Physique Nucléaire
	RADHAKRISHNA	Thermodynamique

MAITRES DE CONFERENCES

MM.	LANCIA Roland	Physique Atomique
Mme.	BOUCHE Liane	Mathématiques
MM.	KAHANE André	Physique Générale
	DOLIQUE Jean Michel	Electronique
	BRIERE Georges	Physique
	DESRE Georges	Chimie
	LAJZEHOWICZ Joseph	Physique
	LAURENT Pierre	Mathématiques Appliquées
Mme.	BERTRANDIAS Françoise	Mathématiques Pures
MM.	LONGQUEUE Jean-Pierre	Physique
	SOHM Jean-Claude	Electrochimie
	ZADWORNY François	Electronique
	DURAND Francis	Chimie Physique
	CARLIER Georges	Biologie végétale
	PFISTER Jean-Claude	Physique
	CHIBON Pierre	Biologie animale
	IDELMAN Simon	Physiologie animale
	BLOCH Daniel	Electrotechnique I. P.
	MARTIN-BOUYER Michel	Chimie (C. S. U. Chambéry)
	SIBILLE Robert	Construction mécanique (I. U. T.)
	BRUGEL Lucien	Energétique I. U. T.
	BOUVARD Maurice	Hydrologie
	RICHARD Lucien	Botanique
	PELMONT Jean	Physiologie animale
	BOUSSARD Jean-Claude	Mathématiques Appliquées (I. P. G.)
	MOREAU René	Hydraulique I. P. G.
	ARMAND Yves	Chimie I. U. T.
	BOLLIET Louis	Informatique I. U. T.
	KUHN Gérard	Energétique I. U. T.
	PEFFEN René	Chimie I. U. T.
	GERMAIN Jean-Pierre	Mécanique
	JOLY Jean-René	Mathématiques Pures
Melle.	PIERY Yvette	Biologie animale
	BERNARD Alain	Mathématiques Pures
	MOHSEN Tahsin	Biologie (C. S. U. Chambéry)
	CONTE René	Mesures Physiques I. U. T.
	LE JUNTER Noël	Génie Electrique Electronique I. U. T.
	LE ROY Philippe	Génie Mécanique I. U. T.
	ROMIER Guy	Techniques Statistiques quantitatives I. U. T.
	VIALON Pierre	Géologie
	BENZAKEN Claude	Mathématiques Appliquées
	MAYNARD Roger	Physique

MM.	DUSSAUD René	Mathématiques (C. S. U. Chambéry)
	BELORIŁKY Elie	Physique (C. S. U. Chambéry)
Mme.	LAJZEROWICZ Jeannine	Physique (C. S. U. Chambéry)
M.	JULLIEN Pierre	Mathématiques Pures
Mme.	RINAUDO Marguerite	Chimie
MM.	BLIMAN Samuel	E. I. E.
	BEGUIN Claude	Chimie Organique
	NEGRE Robert	I. U. T.

MAITRES DE CONFERENCES ASSOCIES

MM.	YAMADA Osamu	Physique du Solide
	NAGAO Makoto	Mathématiques Appliquées
	MAREZIO Massimo	Physique du Solide
	CHEECKE John	Thermodynamique
	BOUDOURIS Georges	Radioélectricité
	ROZMARIN Georges	Chimie Papetière

Le présent travail a été réalisé dans le cadre d'un contrat passé entre la Délégation Générale à la Recherche Scientifique et Technique et le Service de Mathématiques Appliquées de l'Université de GRENOBLE.

Contrat n° 67.00 - 971

Je tiens à exprimer ici toute ma reconnaissance à :

Monsieur Le Professeur J. KUNTZMANN, Directeur du Service de Mathématiques Appliquées de l'Université de GRENOBLE, qui a dirigé ce travail et qui, par son aide et ses précieux conseils, m'a permis de le mener à bien.

Je suis particulièrement sensible à l'honneur qu'il m'a fait en acceptant de présider le Jury.

Je remercie vivement,

Monsieur C. BENZAKEN, Maître de Conférences à la Faculté des Sciences de GRENOBLE.

Monsieur L. BOLLIET, Maître de Conférences à l'Institut Universitaire de Technologie de GRENOBLE qui ont bien voulu accepter de faire partie du Jury.

Mes remerciements iront aussi à Messieurs F. ANCEAU, J. MERMET, et C. PAYAN, dont la collaboration active et amicale m'a été très précieuse.

Je voudrais également remercier Monsieur J. DOUSSY dont les conseils en programmation, m'ont été fort utiles.

Je ne saurais, enfin, oublier, dans mes remerciements, les personnels des services administratifs et techniques de l'IMAG qui ont assuré la réalisation matérielle de cette thèse.

TABLE DES MATIERES

INTRODUCTION

PRELIMINAIRES

- 1 - Structures generées 3
- 2 - Rappel sur la notion d'unité 5

PREMIERE PARTIE : ETUDE EXHAUSTIVE DES NOTIONS DU LANGAGE CASSANDRE EN VUE DU DECOUPAGE.

CHAPITRE I : LES OPERATEURS DU LANGAGE - DEFINITION DES UNITES ELEMENTAIRES

- 1 - Les unités élémentaires de base 13
- 2 - Autres unités élémentaires 16
- 3 - Opérateurs réalisables par un réseau d'unités élémentaires 21
- 4 - Valeur numérique 23

CHAPITRE II : PARTIE "CONTROLE" D'UNE UNITE

- 1 - Les instructions relatives au contrôle 28
- 2 - Entrées et sorties de l'Unité de contrôle 29
- 3 - Liaison entre unités de contrôle. 32

CHAPITRE III : LES EXPRESSIONS.

- 1 - Forme conditionnelle de l'expression 37
- 2 - Les "Si" imbriqués dans les expressions. 41
- 3 - Algorithme de formation de la structure d'une expression 46

CHAPITRE IV : LES INSTRUCTIONS CONDITIONNELLES

CHAPITRE V : LES REGISTRES

- 1 - Chargement d'un registre 61
- 2 - Les horloges en entrée 64
- 3 - Les conditions en entrée 65
- 4 - Les variables en entrée. 66

CHAPITRE VI : PROBLEMES DIVERS

1 - Partition des variables	71
2 - Coût d'une unité	75
3 - Nombre de fils de connexion d'une unité	75
4 - Regroupement d'unifiés	75

SECONDE PARTIE : LE SYSTEME DE PROGRAMMES

1 - Programmes de traitement généraux	77
2 - Organisation de la chaîne des programmes de découpage	80

EXEMPLE : ETUDE D'UN MULTIPLIEUR 83

CONCLUSION :

BIBLIOGRAPHIE

ANNEXE : Carte syntaxique

PROGRAMMES

INTRODUCTION

A partir de la description d'un système digital en CASSANDRE, donc vu sous son aspect "logique" il est utile de pouvoir donner rapidement au concepteur une idée, même grossière, de la réalisation du système. A partir de cette ébauche le concepteur pourra modifier la description du système jusqu'à obtenir une réalisation, non encore optimisée, qu'il jugera la plus apte à conduire à une solution raisonnable du problème.

On voit nettement apparaître l'utilité d'un système conversationnel afin d'avoir un outil utilisable avec toute la souplesse désirée.

Dans ce travail, nous avons essayé de concilier deux impératifs:

- s'approcher le plus possible du schéma logique
- rester indépendant des technologies.

PRELIMINAIRES

- Un système logique digital se présente comme un ensemble d'organes juxtaposés et interconnectés. Les interconnexions se font selon un schéma déterminé et vis-à-vis d'un organe donné A, un organe B se présente sous l'aspect d'une "boîte noire", un certain nombre de "bornes" permettant d'entrer et de sortir l'information.

Si le système logique étudié est important, sa description en un langage évolué tel que CASSANDRE ne sera pas directement utilisable pour effectuer la réalisation.

Un premier travail est donc de compiler la description du système logique afin d'obtenir une description moins formelle. Le résultat de ce travail n'est généralement pas encore apte à permettre la réalisation du système.

En effet, on se trouve face à une telle masse d'informations que la compréhension en devient délicate. Une réalisation manuelle serait donc longue et fastidieuse, quant à une réalisation automatique elle ne serait pas possible en un temps raisonnable : les algorithmes employés étant combinatoires, les temps de calcul deviennent prohibitifs.

On voit donc apparaître la nécessité de fractionner le système en morceaux plus petits, donc plus aisément réalisables.

Le problème maintenant est de définir un découpage du système qui devra tenir compte de la logique décrite afin de regrouper les éléments de nature similaire, ce qui devrait en faciliter la réalisation et permettre éventuellement d'éviter certaines redondances.

La description en CASSANDRE du système étudié contient des indications permettant la partition du système, peut être pas de façon optimale, mais raisonnablement.

Nous avons donc choisi de découper le système par analyse syntaxique de sa description en CASSANDRE.

Cette méthode présente à nos yeux deux avantages :

- permettre simultanément la compilation des morceaux découpés.
- utiliser toutes les informations que le concepteur a naturellement introduites dans la description par la manière même dont il l'a faite.

Ainsi les deux descriptions suivantes, bien qu'ayant la même signification, ne conduiront pas à la même réalisation.

1° description $\left\{ \begin{array}{l} \underline{\text{Si A alors}} R \leftarrow X \\ \underline{\text{Si B alors}} R \leftarrow Y \end{array} \right.$

2° description $\underline{\text{Si A alors}} R \leftarrow X \underline{\text{sinon si B alors}} R \leftarrow Y$

Le sens est bien le même dans les deux cas puisque l'on interdit en CASSANDRE les chargements multiples, ce qui suppose donc que $A \wedge B = \emptyset$

Avant de passer à l'étude détaillée du découpage syntaxique, nous allons faire quelques remarques générales sur les structures générées par analyse syntaxique et un bref rappel de la notion d'unité en CASSANDRE.

I - STRUCTURES GENEREES

Le découpage se basant sur l'analyse syntaxique du texte, nous sommes amenés à distinguer deux types de structures :

- une structure "élémentaire" dont le niveau le plus fin est le réseau d'opérateurs logiques.

- une structure "emboîtée" due à la nature récursive du langage, qui se superpose à la première.

Ces deux structures sont surtout visibles au niveau des expressions.

La structure élémentaire est parfaitement définie jusqu'à un certain niveau dans la mesure où il ne subsiste pas d'ambiguïtés dans la sémantique du langage. Elle ne l'est pas totalement puisque différentes réalisations d'un être décrit sont parfois possibles selon la technologie employée. Ainsi l'expression />A peut être réalisée de plusieurs façons [Cf p. 13-19]

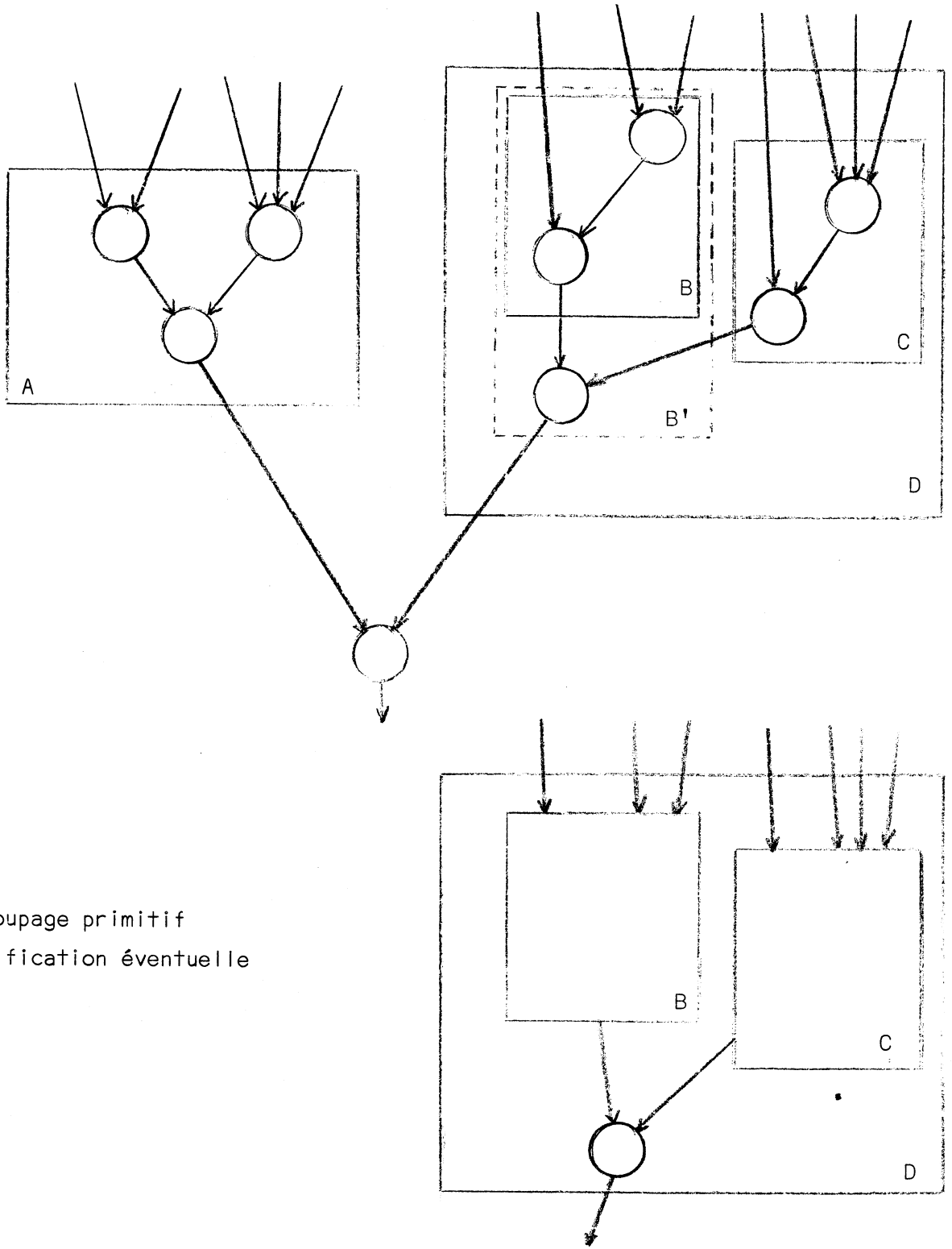
Par contre, la structure emboîtée, si elle est en partie tributaire de la syntaxe du langage du fait même de la méthode choisie, n'est pas déterminée par la sémantique du langage.

Nous avons donc pris certaines options quant à la structure emboîtée, que nous exposerons lors de l'étude systématique du langage.

Il est bien certain que les choix effectués ne mèneront pas toujours à un découpage satisfaisant, d'où la nécessité de pouvoir le modifier. C'est ici qu'un système conversationnel s'avère indispensable afin de pouvoir effectuer des regroupements ou de nouveaux partages dans un laps de temps raisonnable.

Le schéma |1| montre cette notion de structures.

Schéma 1 : Structures



— Découpage primitif
--- modification éventuelle

structure élémentaire dans la "boite noire" D

II - RAPPEL SUR LA NOTION D'UNITE

- Une description en CASSANDRE est une liste d'"Unités", le symbole de base "Unité" les précédant toutes et servant de séparateur.

"Unité" suivi d'un <IDENTIFICATEUR> initialise la description, ou définition de l'unité ainsi identifiée.

Vue de l'extérieur, une unité est entièrement caractérisée par son nom, le nombre, l'ordre et les dimensions de ses entrées et ses sorties.

La description (ou définition) d'une unité est rigoureusement indépendante de celle des autres unités. Ceci permet de considérer, si on le désire, une unité comme une "boîte noire" et de faire tout traitement que l'on juge nécessaire sur cette unité sans se préoccuper des autres.

(1) Unité déclarée externe

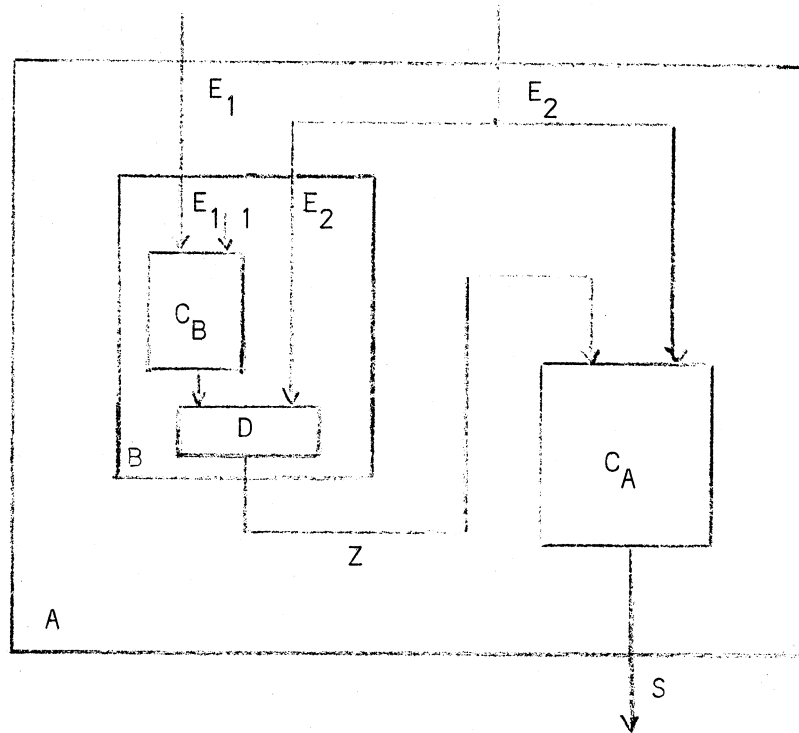
Une unité B peut être déclarée "externe" dans une unité A. Si elle est ainsi déclarée, l'unité B peut être connectée avec d'autres éléments dans l'unité A.

Ceci définit une "hiérarchie" d'unités, une unité utilisée à l'intérieur d'une autre étant inaccessible de l'extérieur de celle-ci.

Deux unités seront dites "de même niveau" si leur connexion s'opère à l'intérieur d'une même autre.

On voit que l'on a ainsi la possibilité de décrire n'importe quelle structure de "boîtes noires" imbriquées et interconnectées.

```
"Unité" A (E1, E2 ; S);  
"Externe" B(;;), C(;;);  
"Signal" Z ;  
Z := B(E1, E2 ; *) ;  
S := C(Z, E2 ; *) ;  
"Unité" B (E1, E2 ; S);  
"Externe" C(;;), D(, ;);  
X := C(E1, 1 ; *) ;  
S := D(X, E2, *) ;
```



Dans l'exemple précédent, deux exemplaires de la même unité C sont utilisées à deux niveaux différents.

Les unités B et C_A sont de même niveau, de même que C_B et D.

(2) Connexion d'une unité externe

Le tableau montre les connexions permises selon que l'unité se trouve ou non utilisée sous un top d'horloge, et selon le type (horloge ou signal) d'entrée-sortie.

Notons que dans le cas où les connexions se font sous horloge, on devra faire l'intersection du signal d'entrée ou de sortie avec cette horloge, sauf si la sortie est connectée sur un registre puisqu'alors l'horloge est directement branchée sur le registre.

HORLOGE	ENTREES		SORTIES		TYPE
	SIGNAL	HORLOGE	SIGNAL	HORLOGE	
NON	SIGNAL	HORLOGE	SIGNAL	HORLOGE	} CONNECTION PERMISE
OUI		SIGNAL	REGISTRE HORLOGE		

(3) Unité externe utilisée sous condition

Que ce soit au niveau d'une instruction conditionnelle, ou d'une expression conditionnelle, les conditions porteront sur tous les branchements effectués, que ce soit sur les entrées ou sur les sorties.

Nous verrons dans le chapitre EXPRESSIONS ce que cela implique.

(4) Numero de duplication

Si plusieurs exemplaires d'une unité externe B sont utilisés dans une unité A, les exemplaires utilisés seront différenciés par un "numéro de duplication" au moment de leur utilisation. L'exemplaire utilisé est entièrement caractérisé par le couple (A,n) où n est le numéro de duplication.

Ceci permet de n'avoir qu'une description d'une unité donnée en bibliothèque. Nous utiliserons ceci pour décrire de façon concise les réseaux d'opérateurs élémentaires.

L'exemple suivant illustre l'utilisation d'une unité externe.

Unité ADD(REE,R(1:4),EDB ; SS(1:4), RES) ;

"Additionneur 4 bits"

Signaux RE1,RE2,RE3,Z(1:2) ;

Externe A(, , ; ,) ;

Z (1:2) := 0:0 ;

A|1|(REE,,Z(1) ; , RE1) ;

A|2|(RE1,,EDB ; , RE2) ;

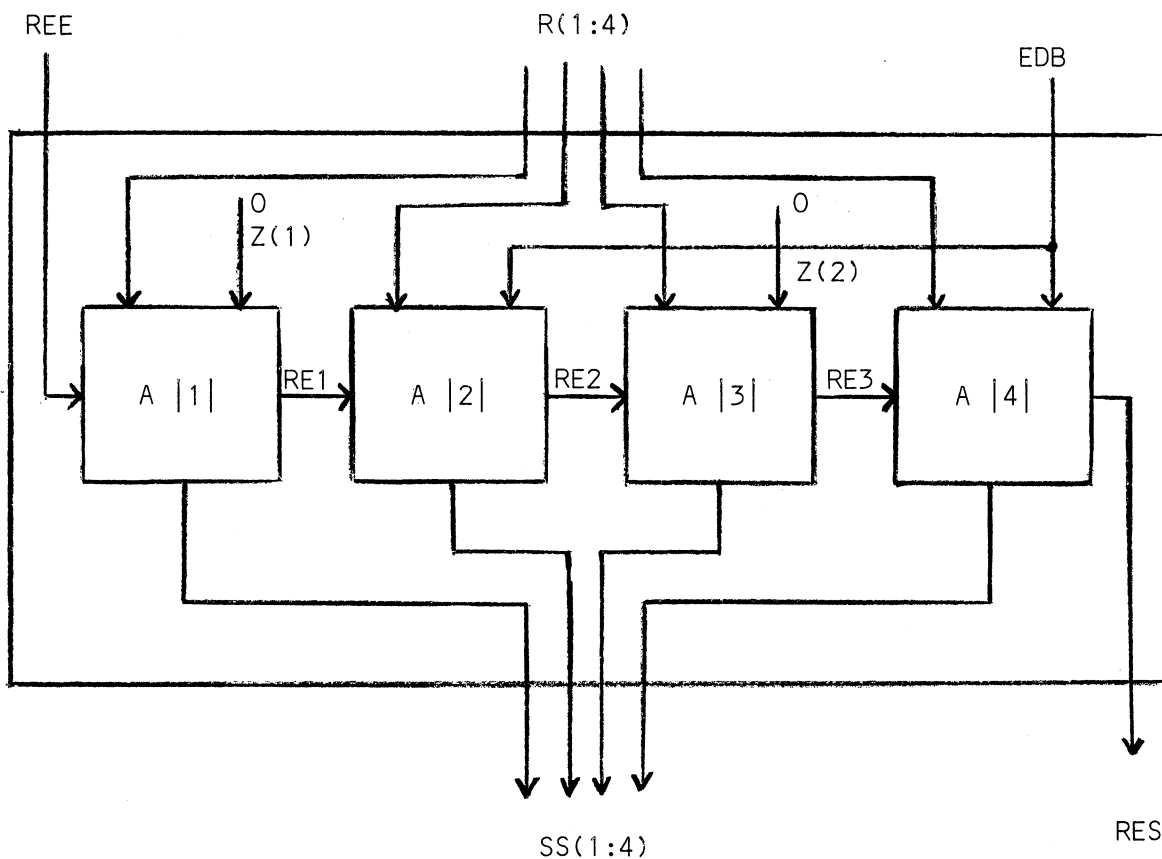
A|3|(RE2,,Z(2) ; , RE3) ;

A|4|(RE3,,EDB ; , RES) ;

'Pour' I = 1 'a' 4 'debut'

A|I|(,R(I), ; SS(I),) ;

'fin' ;



REMARQUE :

Afin de s'insérer au maximum dans le cadre du système CASSANDRE en cours de développement, deux principes de base ont été adoptés :

- conservation à tous les niveaux de la notion d'unité afin de pouvoir faire passer dans une chaîne quelconque du système tout ou partie de la structure générée.

- description en CASSANDRE de la structure générée. Il s'agit évidemment d'un CASSANDRE appauvri, puisque l'on utilise plus que les branchements élémentaires, et éventuellement les boucles "Pour".

Dans toute la suite, nous ne considérons qu'une unité à la fois, qui sera l'unité en cours de traitement.

P R E M I E R E P A R T I E

Etude exhaustive des notions du langage CASSANDRE en vue du découpage.

CHAPITRE I : *Les opérateurs du langage. Définition des unités élémentaires*

CHAPITRE II : *Partie "contrôle" d'une unité .*

CHAPITRE III : *Les expressions.*

CHAPITRE IV : *Les instructions conditionnelles.*

CHAPITRE V : *Les registres.*

CHAPITRE VI : *Problèmes divers.*

C H A P I T R E I

LES OPERATEURS DU LANGAGE
DEFINITION DES UNITES ELEMENTAIRES

Le tableau |p12| donne la liste des opérateurs du langage. Les opérateurs arithmétiques n'entrent pas dans ce tableau.

Nous séparerons ces opérateurs en deux groupes :

- . ceux qui n'introduisent aucun élément technologique.
- . ceux qui correspondent à un ou plusieurs éléments technologiques.

- Le premier groupe comprend :

- . La permutation : * P et * P|n1||n2|
- . Le décalage : * D|n|
- . La rotation : * R|n|
- . La concaténation : &

Ces opérateurs traduisent simplement des regroupements ou des renumérotations sur des nappes de fils.

Nous nous contenterons donc de renommer, si ce n'est déjà fait, les nouvelles variables obtenues en faisant agir l'un de ces opérateurs. Les dimensions de la nouvelle variable seront normalisées, c'est à dire de la forme :

$$(1:n_1, 1:n_2, \dots, 1:n_p)$$

EXEMPLE :

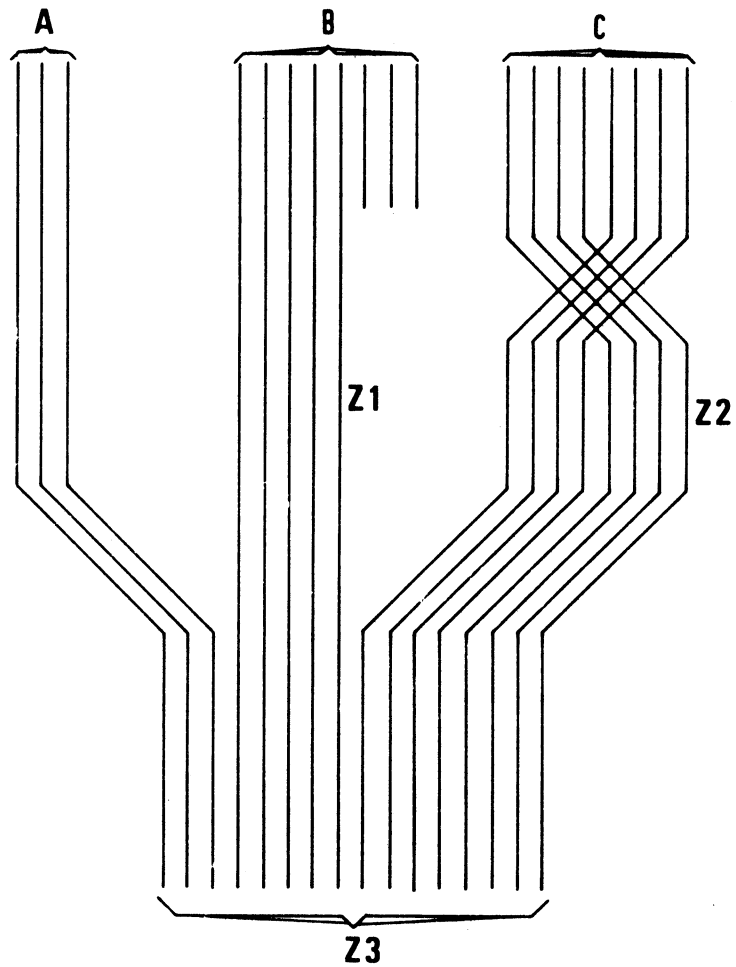
$$A(1:3) \& * D|3|B(1:8) \& * R|4|C(4:12)$$

donne

$$Z_1(1:5) := * D|3|B(1:8)$$

$$Z_2(1:8) := * R|4|C(4:12)$$

$$Z_3(1:16) := A(1:3) \& Z_1(1:5) \& Z_2(1:8)$$



- Dans le second groupe, nous distinguerons les opérateurs donnant lieu à la création d'unités élémentaires et ceux qui sont réalisés par un réseau d'unités élémentaires.

Nous allons tout d'abord définir les unités élémentaires avant de donner la réalisation des opérateurs de relation et de comparaison.

Enfin nous étudierons l'opération de valeur numérique dont nous ferons une unité élémentaire.

Tableau des opérateurs non arithmétiques

	Symbole	Utilisation
U N A I R E	-	Négation
	* P	Permutation des deux premières composantes d'une variable tensorielle
	* P n ₁ n ₂	Permutation des deux composantes spécifiées n ₁ et n ₂
	% τ	Retard relatif. τ durée du retard
	* U τ	génère un signal à partir d'une horloge Un coup ; durée, d' signal généré
	⊙	Dérivation : génère une horloge à partir d'un signal
	* D n	Decalage de n positions à droite si n négatif. à gauche si n positif. Dans les deux cas n positions sont perdues.
	* R n	Rotation de n positions
B I N A I R E	=	Conjonction
	≠	Disjonction
	<	Inférieur
	>	Supérieur
	≤	Inférieur ou égal
	≥	Supérieur ou égal
N / A I R E	+	Ou
	.	Et
	&	Concaténation
	/ + . operateur binaire	Réduction par rapport à l'opérateur situé après

I - LES UNITES ELEMENTAIRES DE BASE

Ce sont principalement celles relatives aux trois opérateurs 'ET', 'OU' et 'NEGATION'. Nous avons distingué pour les opérateurs 'ET' et 'OU' plusieurs types, selon l'utilisation qui en est faite. En effet, les solutions technologiques seront différentes selon qu'un 'OU' est employé par une horloge ou un signal, la rapidité, par exemple, n'étant pas la même.

Nous allons donner rapidement la description de ces unités élémentaires en CASSANDRE.

- NEGATION

Cet opérateur ne peut être utilisé qu'avec des signaux, il n'y en a donc qu'un seul type.

$$\begin{aligned} & \& \text{NEG} (E1; S1) ; \\ & S1 := -E1 ; \end{aligned}$$

- ET

Nous en distinguerons deux types selon que toutes les entrées sont de type signal ou que l'une des entrées est de type horloge.

$$\begin{aligned} & \& \text{ET} \alpha n (E1, \dots, E_n ; S1) ; \\ & S1 := E1, \dots, E_n ; \end{aligned}$$

avec $\alpha = \begin{cases} S \\ H \end{cases}$ et $n =$ nombre d'entrées.

- OU

Quatre types selon qu'il est utilisé pour des signaux, des horloges ou bien qu'il marque des connexions créées par le traitement de l'unité, et susceptibles d'être réalisées par simple soudure dans certains cas.

$$\begin{aligned} & \& \text{OU} \alpha n (E1, \dots, E_n ; S1) ; \\ & S1 := E1 + \dots + E_n ; \end{aligned}$$

avec $\alpha = \begin{cases} S \\ H \end{cases}$

(Notation "&" voir remarque page 81)

La description des OU de connexion doit être différente : en effet dans un OU de connexion, une seule entrée doit être à un - Ceci peut se traduire ainsi :

$$\&OU_{\alpha}(E_1, \dots, E_n, S_1) ;$$
$$E := E_1 \&\dots \&E_n ;$$

Si E alors (S1:=E1) ... (S1:=En) ;

avec $\alpha = \begin{cases} \text{CHO} & \text{connections d'horloges} \\ \text{CSI} & \text{connections de signaux} \end{cases}$

Si l'on a deux entrées à 1, on a alors :

$S_1 := E_i$ et $S_1 := E_j$ ce qui est incorrect sémantiquement (chargement multiple).

Les unités &ET et &OU peuvent posséder dans le cas général n entrées. La réalisation sera fonction du nombre d'entrées des opérateurs technologiques dont on dispose.

- Un coup * U

$$\&UNCOU_{\tau}(E;S)$$
$$S := *U|\tau|E$$

- Dérivation

$$\&DERIV(E;S)$$
$$S := S ;$$

- Retard * | τ |

$$\&RET_{\tau}(E;S)$$
$$S := \%|\tau|E ;$$

L'utilisation des unités précitées exige l'emploi de boucles POUR afin d'avoir une description assez concise de l'unité qui les utilise.

Dans une seconde étape, si l'on désire la numérotation effective de chaque unité élémentaire, pour les schémas logiques, par exemple, il suffira d'exécuter les boucles POUR.

Donnons un exemple :

A,B,C,D,E et Z étant des signaux de dimensions (1:8,1:4),
l'expression $Z := -A+B \cdot C+D \cdot E$
donnera :

```
"Unité" &EXEMPLE(A,B,C,D,E; ?) ;
'Externes' &NEG(;) ; &ET2 (,;) ; &OU3(,,);
'Signaux' &Z1(1:8,1:4);&Z2(1:8,1:4);&Z3(1:8,1:4);
  Pour J=1 à 4 début :
    Pour I=1 à 8 début :
      &Z1(I,J) := &NEG | I+8(J-1) | (A(I,J) ; * ) ;
      &Z2(I,J) := &ET2 | I+8(J-1) | (B(I,J), C(I,J) ; * ) ;
      &Z3(I,J) := &ET2 | 32+I+8(J-1) | (D(I,J),E(I,J) ; * ) ;
      Z(I,J) := &OU3 | I+8(J-1) | (&Z1(I,J),&Z2(I,J),&Z3(I,J) ; * ) ;
    fin ; fin ;
```

Pour les numeros de duplication, nous avons numeroté les éléments du tenseur selon la règle suivante :

$(1:n_1, 1:n_2, \dots, 1:n_p)$ étant un tenseur de dimensions et notant l_1, \dots, l_p les indices des boucles POUR, l'élément courant est donné par :

$$C = | l_1 + (l_2 - 1)n_1 + (l_3 - 1)n_1 \cdot n_2 + \dots + (l_p - 1)n_1 \cdot n_2 \cdot \dots \cdot n_{p-1} |$$

il convient éventuellement d'ajouter à ceci le nombre des éléments précédemment apparus (dans l'exemple, le second emploi de &ET2).

L'exécution des boucles POUR sur l'exemple donné donnerait :

```
&Z1(1,1):=&NEG | 1 | (A(1,1);*) ;
&Z1(2,1):=&NEG | 2 | (A(2,1);*) ;
-----
&Z2(1,1):=&ET2 | 1 | (B(1,1),C(1,1);*) ;
-----
Z(1,1):=&OU2 | 1 | (&Z1(1,1),&Z2(1,1),&Z3(1,1) ; * ) ;
-----
```

II - AUTRES UNITES ELEMENTAIRES

Certaines opérations ont été considérées comme donnant des unités élémentaires car il n'apparaissait pas de réalisation primant sur d'autres.

Ce sont :

- la réduction par rapport aux opérateurs de relations.

$/ < ; / \leq ; / > , / \geq ; / = ; / \neq$

- L'opération de "Valeur Numérique", à laquelle on peut faire correspondre un décodeur.

(1) Reduction par rapport aux operateurs de relations

On suppose comme IVERSON $|B3|$, que l'opération s'effectue de droite à gauche.

EXEMPLE :

$A = 01010$

$/ > A = (0 > (1 > (0 > (1 > 0))) = 0$

puisque

x	y	$x > y$
0	0	0
0	1	0
1	0	1
1	1	0

Ces opérations s'effectuent sur la première dimension si la variable est tensorielle. Il s'agit donc d'une dégénérescence.

$$\begin{array}{l}
 A(1:n) \xrightarrow{\text{ØPREL}} A(1:1) \\
 A(1:n_1, \dots, 1:n_p) \xrightarrow{\text{ØPREL}} A(1:1, 1:n_2, \dots, 1:n_p)
 \end{array}$$

Nous allons donner, pour ces opérateurs une manière d'effectuer le calcul de gauche à droite, puis donner diverses réalisations possibles.

Soit $A(1:n)$, notons a_1, \dots, a_n les éléments de A , soit $k \in [0, n]$

$/ > A$

$$\left. \begin{array}{l} a_1, \dots, a_k = 1 \\ a_{k+1} = 0 \\ a_{k+2}, \dots, a_n = \emptyset \end{array} \right\} \Rightarrow \begin{cases} \text{si } k = 2_p \Rightarrow / > A = 0 \\ \text{si } k = 2_{p+1} \Rightarrow / > A = 1 \end{cases}$$

c'est la parité des 1 à gauche du premier 0 qui donne le résultat
Si le premier élément est 0, le résultat est nul.

$/ < A$

$$\left. \begin{array}{l} a_1, \dots, a_{n-1} = 0 \\ a_n = 1 \end{array} \right\} \Rightarrow / < A = 1$$

c'est la seule configuration donnant un résultat non nul.

$/ \geq A$

$$a_1 = 1 \Rightarrow / \geq A = 1$$

$$\left. \begin{array}{l} a_1, \dots, a_k = 0 \\ a_{k+1} = 1 \\ a_{k+2}, \dots, a_n = \emptyset \end{array} \right\} \Rightarrow \begin{cases} \text{si } k = 2_p \quad / \geq A = 1 \\ \text{si } k = 2_{p+1} \quad / \geq A = 0 \end{cases}$$

$/ \leq A$

$$\left. \begin{array}{l} a_1, \dots, a_{n-1} = 1 \\ a_n = 0 \end{array} \right\} \Rightarrow / \leq A = 0$$

seule configuration donnant un résultat nul.

$/ = A$

parité des 0 de A.

si l'on a un nombre pair de zéro alors $/ = A = 1$

sinon $/ = A = 0$

$/ \neq A$

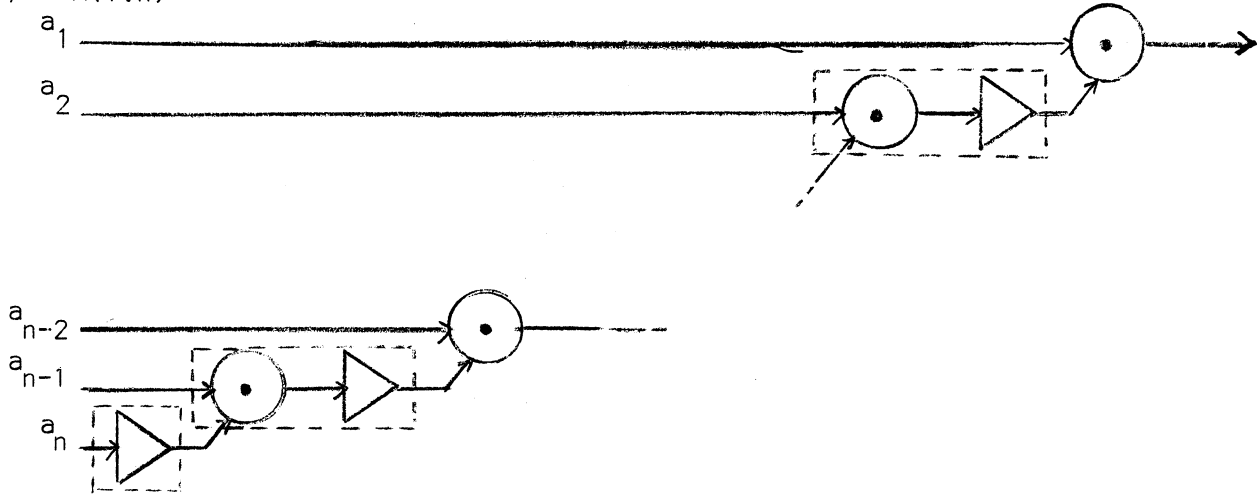
imparité des 1 de A

Si l'on a un nombre pair de zéro alors $/ \neq A = 1$

sinon $/ \neq A = 0$

Donnons, à titre d'exemples, deux réalisations possibles pour les opérateurs $/ >$ et $/ \geq$, en se basant soit sur le calcul de droite à gauche, soit sur le calcul de gauche à droite.

$/ > A(1:n)$

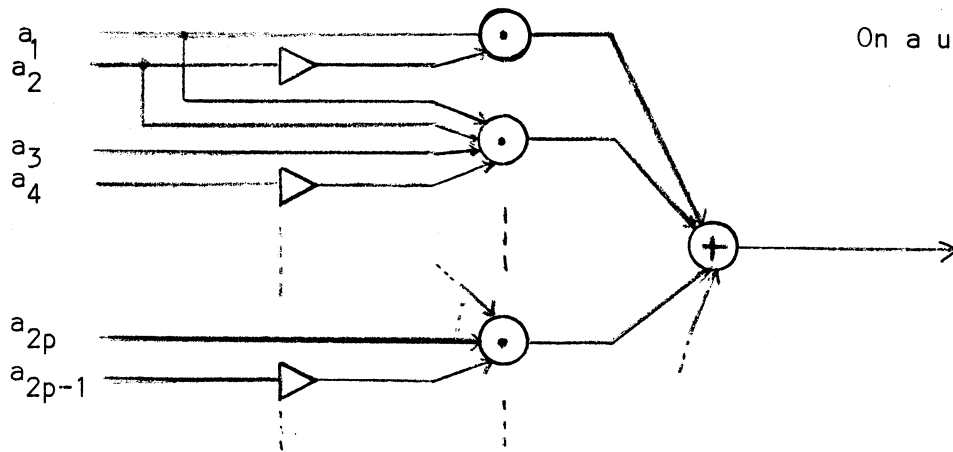


Réseau $2n-2$ couches, ou n couches dont $n-1$ NAND.

On a une autre réalisation possible en se basant sur la manière d'effectuer le calcul de gauche à droite. Il suffit de réaliser :

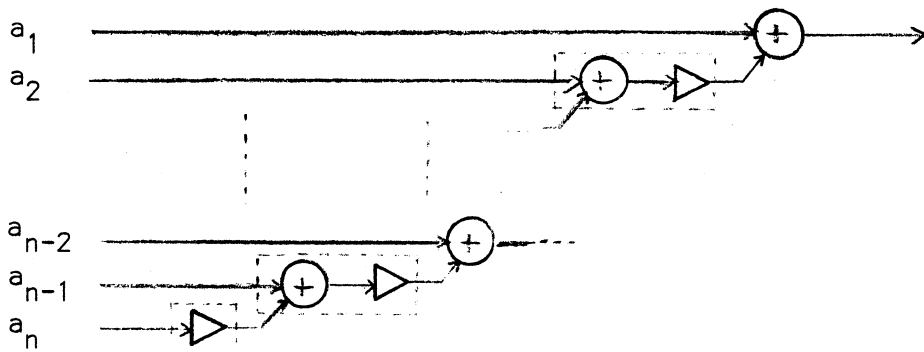
si n pair $\sum_{\substack{p=2 \\ p \text{ pair}}}^n a_1 a_2 \dots a_{2p-1} \bar{a}_{2p}$

si n impair $\sum_{\substack{p=2 \\ p \text{ pair}}}^{n-1} a_1 a_2 \dots a_{2p-1} \bar{a}_{2p} + a_1 a_2 \dots a_n$



On a un réseau en trois couches.

/ $\geq A(1:n)$



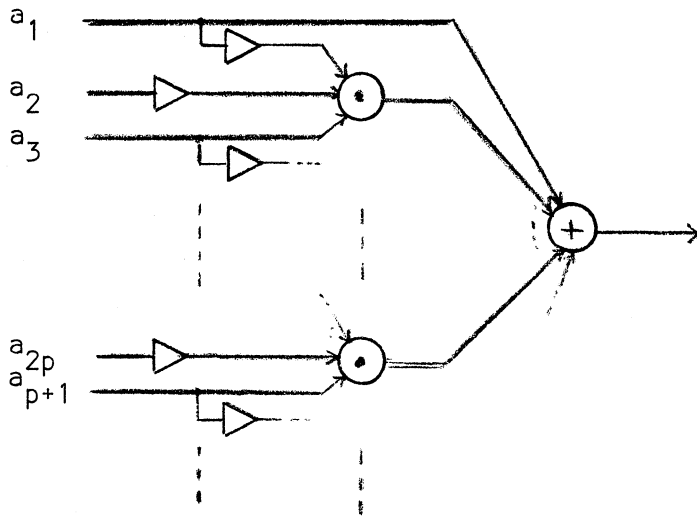
Réseau n couches dont n-1 NØR

On peut réaliser aussi :

$$\sum_{p=1}^n \bar{a}_1 \bar{a}_2 \dots \bar{a}_{2p-2} a_{2p-1} \quad \text{si } n \text{ impair}$$

$$\sum_{p=1}^{n-1} \bar{a}_1, \bar{a}_2 \dots \bar{a}_{2p-2} a_{2p-1} + \bar{a}_1 \dots \bar{a}_n \quad \text{si } n \text{ pair.}$$

On a un réseau en trois couches.



La description en CASSANDRE des unités élémentaires de réduction est :

$$\& R \alpha_{n_1 \dots n_p} (E_1(1:n_1, \dots, 1:n_p) ; S_1(1:1, 1:n_2, \dots, 1:n_p)) ;$$

$$S_1 := / \beta E_1 ;$$

avec $\alpha \left\{ \begin{array}{l} \text{INF} \\ \text{INFG} \\ \text{SUP} \\ \text{SUPG} \\ \text{EG} \\ \text{DIF} \end{array} \right.$ et $\beta \left\{ \begin{array}{l} < \\ \leq \\ > \\ \geq \\ = \\ \neq \end{array} \right.$

Les chiffres $n_1 \dots n_p$ figurant après α dans l'entête de l'unité permettent de différencier les exemplaires dans la bibliothèque des unités |p 76|.

III - OPERATEURS REALISABLES PAR UN RESEAU D'UNITES ELEMENTAIRES

Ce sont tous les opérateurs de comparaison et de relation.

Le tableau |p22| donne leur réalisation. Donnons un exemple de la façon dont un de ces opérateurs serait décrit.

A(1:8) et B(1:8) étant des variables, l'expression

Z(1:8) := A < B

donnerait :

Pour I = 1 à 8 début :

Z₁(I) := &NEG|I|(A(I) ; *) ;

Z(I) := &OU|I|(Z₁(I),B(I) ; *) ;

fin ;

TABLEAU DES OPERATEURS DE COMPARAISON ET DE RELATION.

Opérateurs	Equation Booléenne associée	Réseau associé
$A = B$	$(\bar{A} \cdot \bar{B}) + (A \cdot B)$	
$A \neq B$	$(\bar{A} \cdot B) + (A \cdot \bar{B})$	
$A < B$	$\bar{A} \cdot B$	
$A > B$	$A \cdot \bar{B}$	
$A \leq B$	$\bar{A} + B$	
$A \geq B$	$A + \bar{B}$	

IV - VALEUR NUMERIQUE

Symbole : $\$$

A étant un signal ou plus généralement une expression, $\$ A$ a le sens d'un entier, dépendant de la configuration des bits de A.

Selon la façon dont elle est employée, la "Valeur Numérique" permet de sélectionner un ou plusieurs éléments dans une nappe de fils.

Nous lui ferons donc correspondre un décodeur dont on ne précisera pas la réalisation. Il y a en effet plusieurs façons de réaliser un tel décodeur :

- . en une couche d'opérateurs
- . de manière pyramidale
- . séparation des variables en deux groupes et utilisation d'un réseau matriciel d'opérateurs ET, leurs emplois dépendent de la technologie employée par ailleurs. [B 1]

La description d'une telle unité en bibliothèque sera donc :

$\&VALNUMn(E_1:S_1)$

$S_1 := \$ E_1 ;$

La dimension de S_1 étant 2^n si l'entrée E_1 est de dimension n.

Nous allons étudier sur des exemples les deux possibilités d'emploi d'une "Valeur Numérique".

Comme indice de sélection

Comme borne d'une dimension.

Indice de sélection

C'est l'emploi normal. La figure |1| donne la réalisation de l'exemple suivant :

Registre R(1:8) ;
Signal A(1:4) ;
<H> R (§ A) <== EXP ;

Borne d'une dimension

"Valeur Numérique" peut être employée comme borne gauche ou borne droite. Seul le circuit de sortie de décodeur change.

Précisons les conventions adoptées :

Soit R(1:n) un registre.

R(1:§ A) on charge les positions $[-\infty, § A] \cap [1, n]$

R(§ A:n) on charge les positions $[§ A, +\infty] \cap [1, n]$

Il nous a semblé plus naturel de charger tout le registre, dans le cas où la dimension obtenue est plus grande que celle définie, que de ne rien charger. Cette définition est d'ailleurs imposée par la vérification : en effet, au moment de la vérification du texte CASSANDRE, ne connaissant pas les valeurs possibles de A, on ne peut pas calculer les entiers §A possible pour contrôler qu'ils restent dans les bornes des dimensions définies.

Ceci explique que nous ayons interdit l'emploi de la "Valeur Numérique" sur les deux bornes à la fois, la vérification de la compatibilité n'étant pas possible dans le vérificateur actuel.

Notons que par ailleurs, la réalisation de :

R(§ A, § B)

ne pose pas de difficultés - [cf figure 2].

La figure |3| donne la réalisation de l'exemple suivant :

Registre R (1:4) ;
Signal A (1:3) ;
<H> R(1:§ A) <== EXP ;

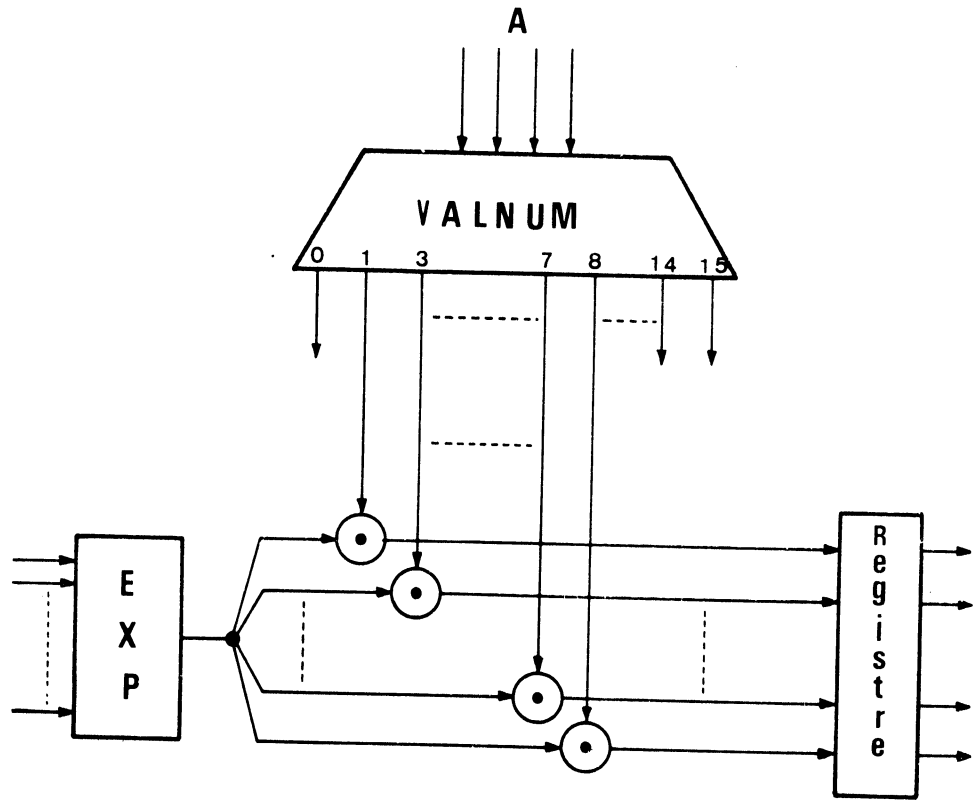


FIGURE 1 $R(\not{A}) \leftarrow EXP$

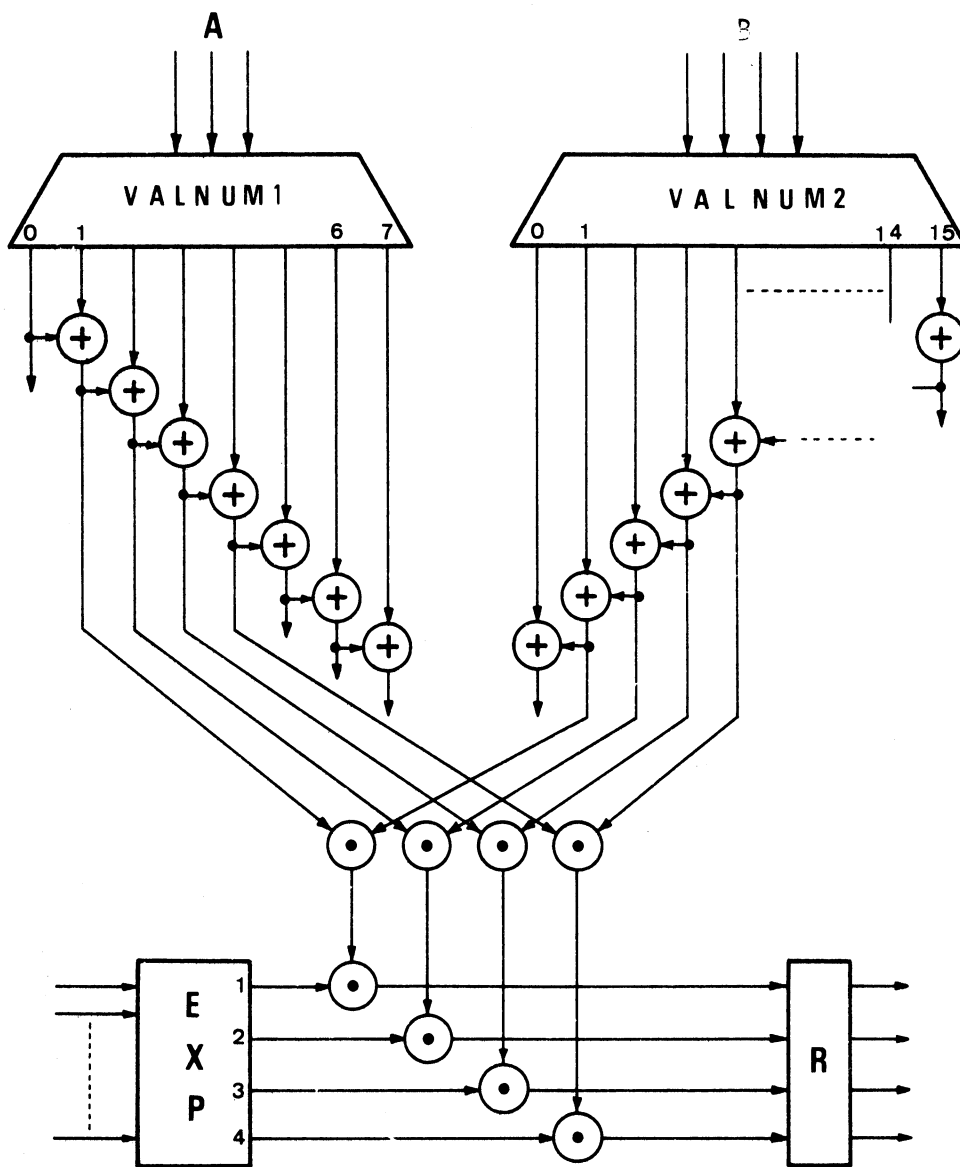


FIGURE 2 : $R(\not{A}, \not{B}) \leftarrow EXP$

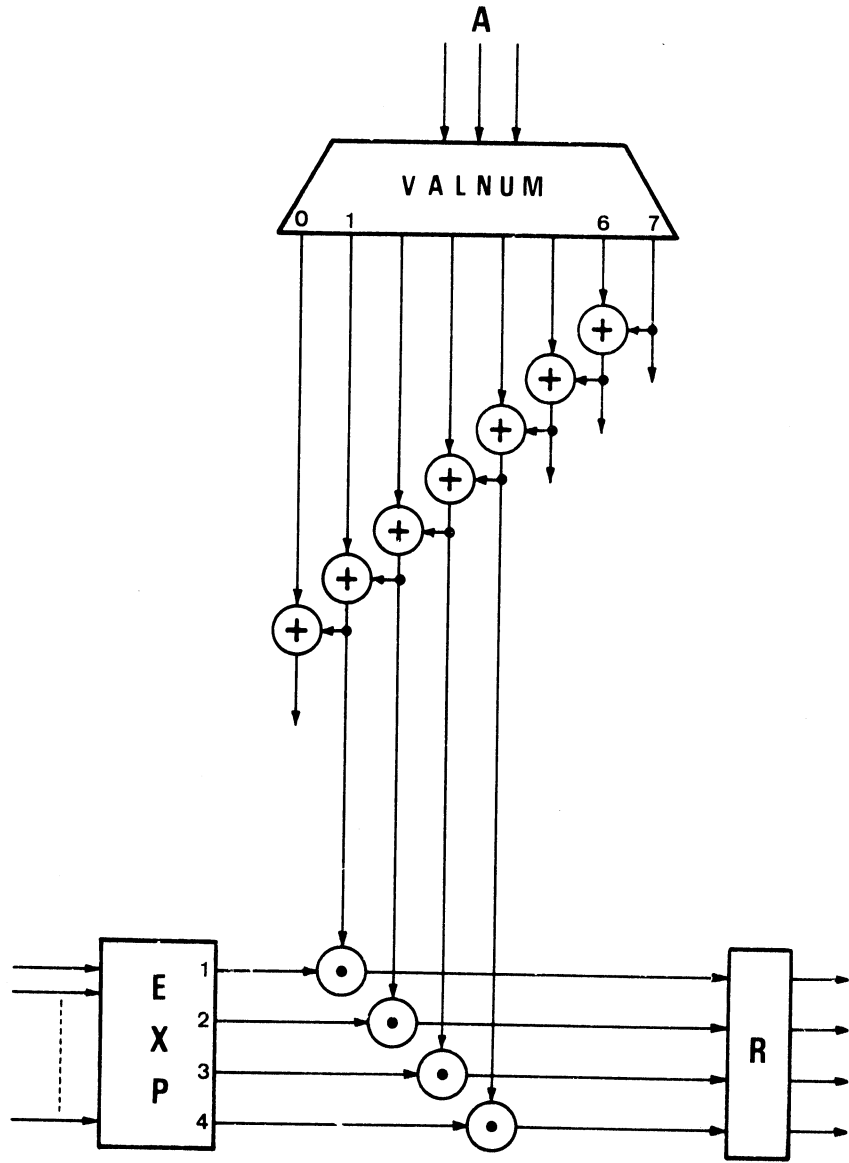


FIGURE 3 : $R(1:\cancel{A}) \leftarrow EXP$

CHAPITRE II

PARTIE "CONTROLE" D'UNE UNITE

La synthèse d'automates requérant des méthodes particulières, nous avons cherché à isoler dans la description de l'unité traitée, toutes les instructions relatives au contrôle de l'unité. Le mieux pour ceci nous a semblé être la création d'une sous unité, que nous appellerons dans ce qui suit "Unité de Contrôle", dont la description fonctionnelle est donnée par les instructions extraites.

Nous allons tout d'abord définir les limites des instructions appartenant au contrôle et de là nous précisons les entrées et les sorties nécessaires à l'unité de contrôle.

I - LES INSTRUCTIONS RELATIVES AU CONTROLE

Ce sont les instructions :

faire EXPDET ;

allera EXPDET ;

il faut y ajouter l'instruction de chargement des registres d'état (spécifiés sous le vocable ETAT dans les déclarations de l'unité)

RDET <= EXPDET ;

En plus de ces instructions, apparaissent dans la description fonctionnelle de l'unité, des identificateurs, considérés comme valeurs d'états de l'unité. Un tel identificateur apparaît en étiquette et conditionne les instructions placées sous sa portée (c'est à dire jusqu'à l'apparition d'une autre étiquette ou la fin de l'unité).

Par ailleurs, la carte syntaxique de l'expression définissant une valeur d'état (EXPDET p |A9 |), montre qu'une expression peut apparaître comme condition logique.

Nous admettrons que l'expression est réalisée en dehors de l'unité de contrôle, et que le résultat est disponible dans l'unité de contrôle.

Nous pouvons donc voir maintenant les entrées et les sorties nécessaires à l'unité de contrôle.

II - ENTREES ET SORTIES DE L'UNITE DE CONTROLE

. L'instruction allera et le chargement d'un registre d'état s'effectuant sous une horloge, il faut donc entrer les diverses horloges conditionnant le déroulement de ces instructions.

. Au cas où une expression est utilisée comme condition logique, on entre la sortie de l'expression.

Les identificateurs d'états définissent les sorties. Nous poserons comme hypothèse de travail qu'à chaque identificateur d'état correspond un fil de dimension 1, commandant un certain nombre de portes. Ce sont donc les fils de sorties d'un décodeur interne à l'unité de contrôle.

Donnons un exemple :

ETA 1 :

----- } liste d'instructions 1

<H> allera si A = B alors ETA2 sinon ETA3 ;

ETA2 :

----- } liste d'instructions 2

<H> allera ETA3 ;

ETA3 :

----- } liste d'instructions 3

<H> allera ETA1 ;

donnera lieu à la création de l'unité de contrôle suivante :

"Unité" CONTROLE (H,Z ; S(1:3)) ;

"horlogemère" H ;

ETA 1 :

S(1:3) := 100 ;

<H> allera si Z alors ETA 2 sinon ETA 3 ;

ETA 2 :

S(1:3) := 010 ;

<H> allera ETA 3 ;

ETA 3 :

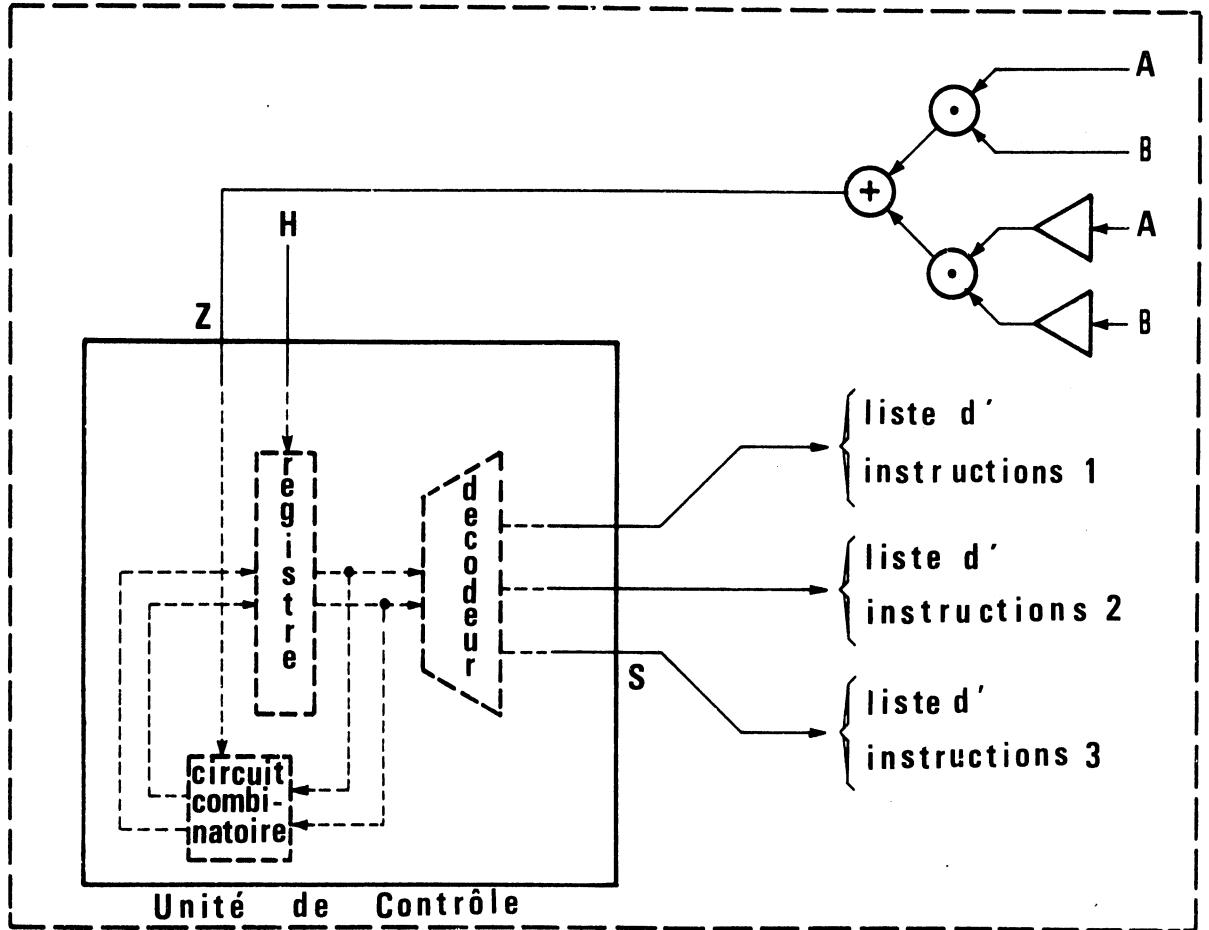
S(1:3) := 001 ;

<H> allera ETA 1 ;

REMARQUE :

Ne cherchant pas à préciser la réalisation de l'Unité de contrôle à ce stade, on précise simplement le fil de sortie actif à l'aide d'une constante.

On ne déclare pas le registre d'état dont la dimension est inconnue avant le codage, et qui existe implicitement dans toute unité CASSANDRE.



UNITE DE CONTROLE D'UNE UNITE DANS LAQUELLE APPARAIT DES ETATS.

III - LIAISON ENTRE UNITES DE CONTROLE

La sémantique de l'expression d'état en CASSANDRE n'étant pas encore entièrement fixée, les possibilités de liaisons entre l'unité de contrôle de l'unité traitée et l'unité de contrôle d'une unité déclarée externe, ne sont pas toutes encore bien définies.

Citons simplement ici la possibilité de réalisation de l'instruction Allera appliquée à un état d'une unité externe par forçage du registre d'état. On force la configuration exacte de l'état (ce qui suppose que les états sont codés) en utilisant les entrées prioritaires de la bascule (SET et CLEAR)

Donnons un exemple simple :

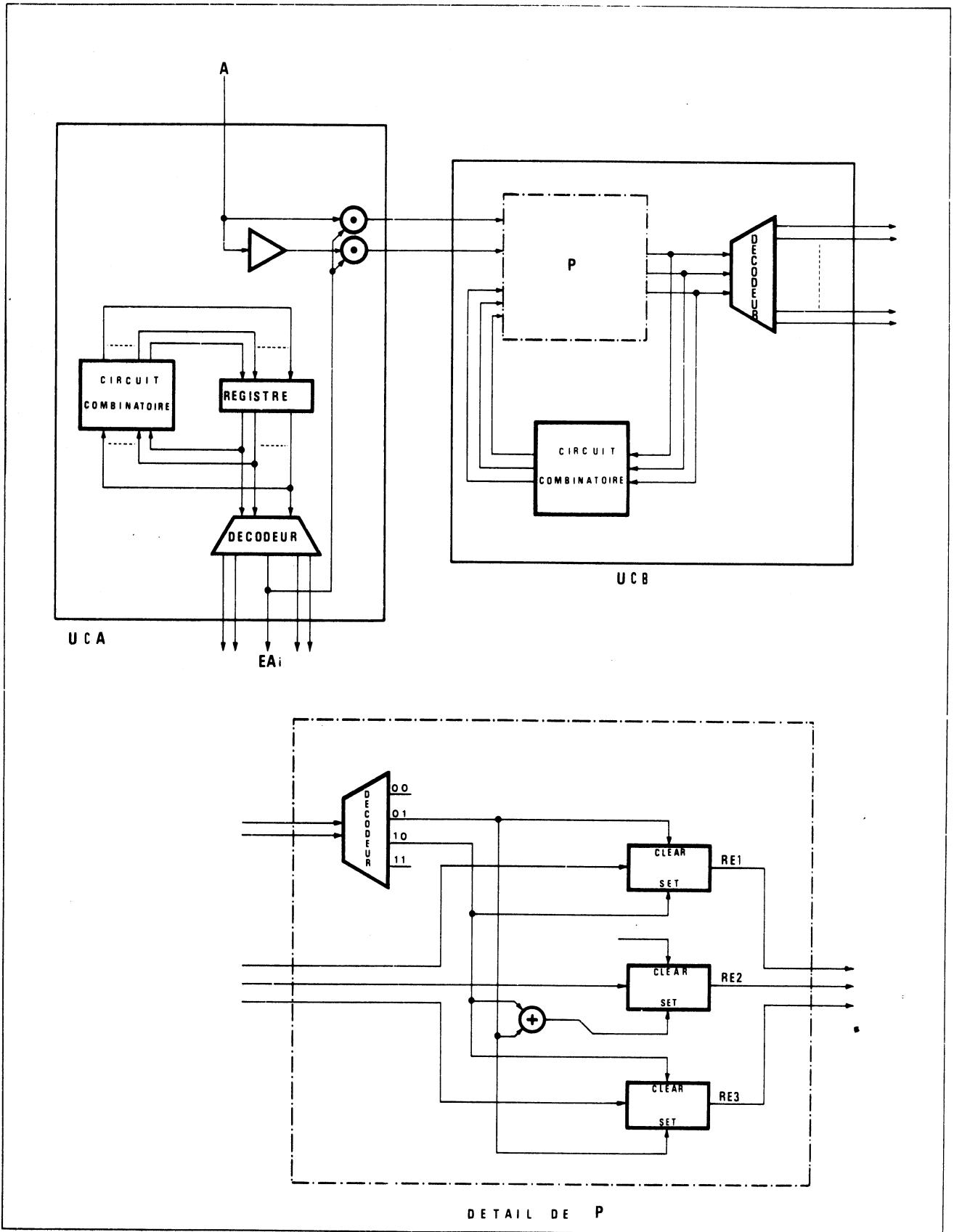
```
'Unité' A (H,E ; S)
  'externe' B, EB1, EB2 ( ; ) ;
  'horlogemère' ;H ;
-----
EAI :
-----
<H> allera si A alors E B1 sinon E B2 ;
-----
```

L'unité de contrôle A (U.C.A) doit fournir à U.C.B. les ordres pour commander le forçage des états. Ceci est réalisable par l'envoi d'un code à U.C.B. Ici, deux bits suffisent pour coder les ordres, par exemple .

EAI	CONDITION A	CØDE
NON	INDIFFERENT	00
OUI	0	01
	1	10

Dans l'U.C.B., nous supposons les états relatifs à l'unité B codés (exemple EB1 par 011, EB2 par 110). Les sorties du décodeur d'ordres sont connectées directement aux entrées prioritaires CLEAR (mise à zéro) ou SET (mise à un) selon le code de l'état à forcer.

Le schéma |p34| montre ceci.



LIAISON ENTRE UNITES DE CONTROLE

CHAPITRE III

LES EXPRESSIONS

Une expression a formellement le sens de la fabrication d'un signal, le temps d'établissement du signal étant supposé inférieur à la période de l'horloge.

Elle se traduira par un réseau logique dont les entrées sont les variables apparaissant dans l'expression, et la sortie, la valeur de l'expression.

L'expression constituera un niveau pour la structure emboîtée. Il n'y aura pas de niveau intermédiaire entre l'expression et les opérateurs élémentaires que nous avons définis. Ceci se justifie par le fait que la syntaxe de l'expression vise à mettre en évidence les règles de priorités des opérateurs élémentaires, entre eux, et ne fait pas apparaître d'entité justifiant d'un niveau de découpage. Naturellement, du fait de la récursivité, les éléments de la structure élémentaire d'une expression pourront être eux même des expressions.

Toutes les variables d'une expression donnent des entrées distinctes pour les unités créées. C'est à dire que l'on ne cherche pas à regrouper les variables de même nom. En effet, la dimension des variables peut être tensorielle et l'on ne possède aucune indication permettant de choisir entre les divers regroupements possibles, au cas où cela serait possible, ce qui ne sera généralement pas le cas.

Ainsi les variables A, B et C étant définies de dimensions (1:12,1:8), considérons

$$Z(1:6,1:4) := A(1:6,1:4) + B.A(4:9,3:7)+C.xD|3|A(1:9,3:7)$$

Le diagramme |1| donne la position des trois occurrences de la variable A et les diagrammes |2| et |3| deux regroupements possibles réduisant le nombre d'entrées.

Par contre, dans le cas de variables uniquement vectorielles, un regroupement pourrait être intéressant.

Puisque nous n'effectuons pas de regroupements des entrées, la structure élémentaire associée à l'expression est donc un arbre.

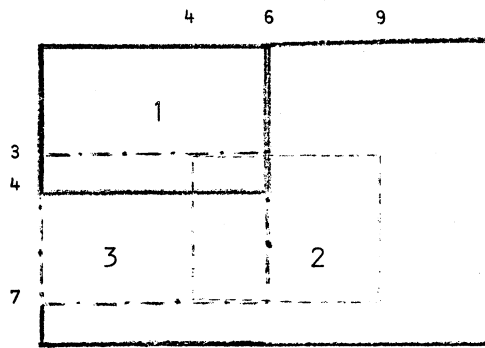


Diagramme 1

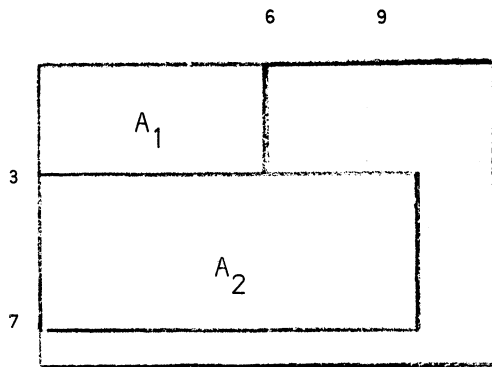


Diagramme 2

A1 (1:6,1:3)

A2 (1:9,3:7)

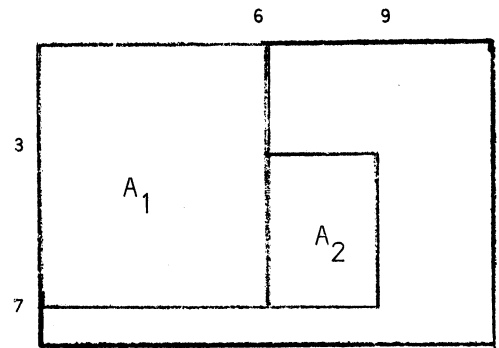


Diagramme 3

A1(1:6,1:7)

A2(6:9,3:7)

La syntaxe de l'expression (carte syntaxique [A8 |]) est la suivante :

EXP si AEX alors EXP sinon EXP
TP
AEX EXP
EXPBE

Nous allons étudier la forme conditionnelle de l'expression. Notons que dans cette forme, le "sinon" est obligatoire.

I - FORME CONDITIONNELLE DE L'EXPRESSION

Nous noterons EXPA l'expression apparaissant après le "alors" et EXPB celle apparaissant après le "sinon". La condition AEX se réduit à un scalaire dans ce cas sa présence va se traduire par un certain nombre de portes ET. Deux possibilités se présentaient :

. Faire l'intersection de toutes les entrées des termes situés sous la portée de la condition avec la condition.

. Faire l'intersection de la sortie de ces termes avec la condition.

Nous avons adopté cette dernière solution qui permet de réduire le nombre de portes ET. Puisque l'on peut toujours se ramener au cas où l'expression sert au branchement d'un signal, ou bien au chargement d'un registre :

Z := EXP
<H> R \Leftarrow EXP.

La dimension de la sortie d'une expression est donc celle du signal (ou du registre) sur lequel elle est connectée.

La sortie d'une expression est donc de dimension inférieure à la somme des dimensions des entrées, sauf dans le cas trivial où l'expression se réduit à un opérateur unaire ne réduisant pas la dimension, auquel cas il y a égalité.

Ainsi, on a $\dim(S) = \dim(A)$
pour l'expression $S := * RA$.

Dans le cas général, on aura toujours
 $\dim(\text{Sortie}) \leq \sum \dim(\text{Entrées}) - 1$

puisqu'alors, au moins, un opérateur binaire apparaît dans l'expression.

Ce choix présente une difficulté dans le cas de l'occurrence d'un "Signal Unité" dans EXPA ou EXPB. En effet, l'unité externe utilisée par le "Signal Unité" peut être employée à un autre endroit de la description et on ne peut donc pas "l'enfermer" dans la boîte noire générée par l'expression qui l'utilise.

D'autre part, les conditions doivent porter aussi sur les entrées spécifiées de l'unité externe qui peuvent être connectées différemment dans un autre emploi.

Il faut donc prévoir comme sorties supplémentaires à l'expression, les conditions. Ces sorties ne seront utilisées que dans le cas où un "signal unité" figurerait dans l'expression.

La structure de la figure |1| découle de ce choix.

Le détail de l'unité externe CONNEX est donné par la figure |2|. Notons simplement ici que les OU employés sont des OU de connexion.

La figure |3| donne la structure de l'exemple suivant, utilisant un "Signal Unité" :

```
S := si A alors ADD(X1, X2 ; *) sinon U ;  
si B alors T := ADD (Y1, Y2 ; *) ;
```

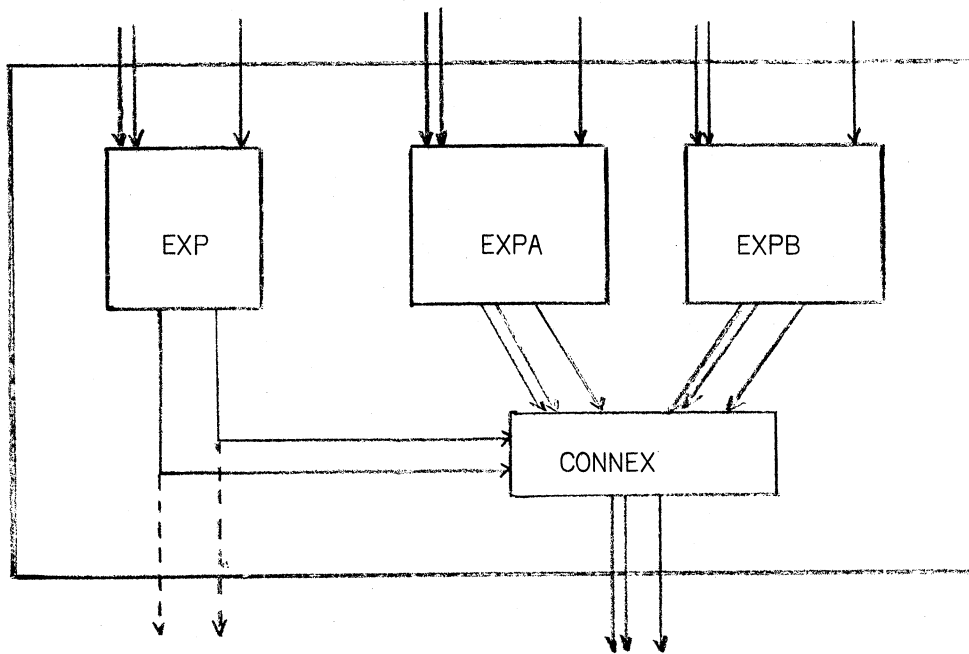


Figure 1 Si EXP alors EXPA sinon EXPB

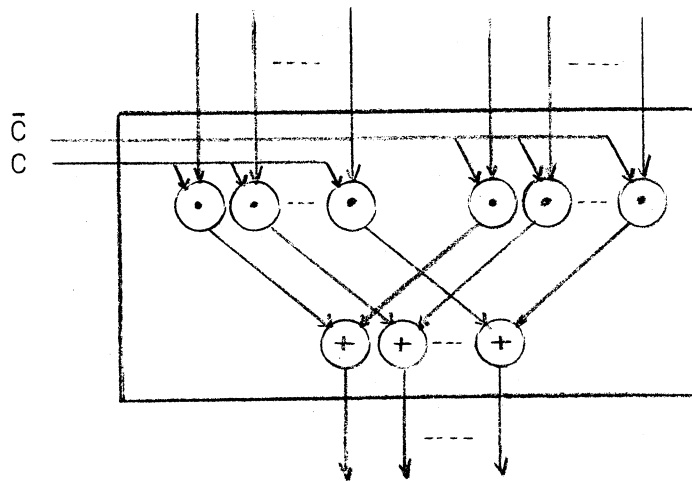


Figure 2 CONNEX

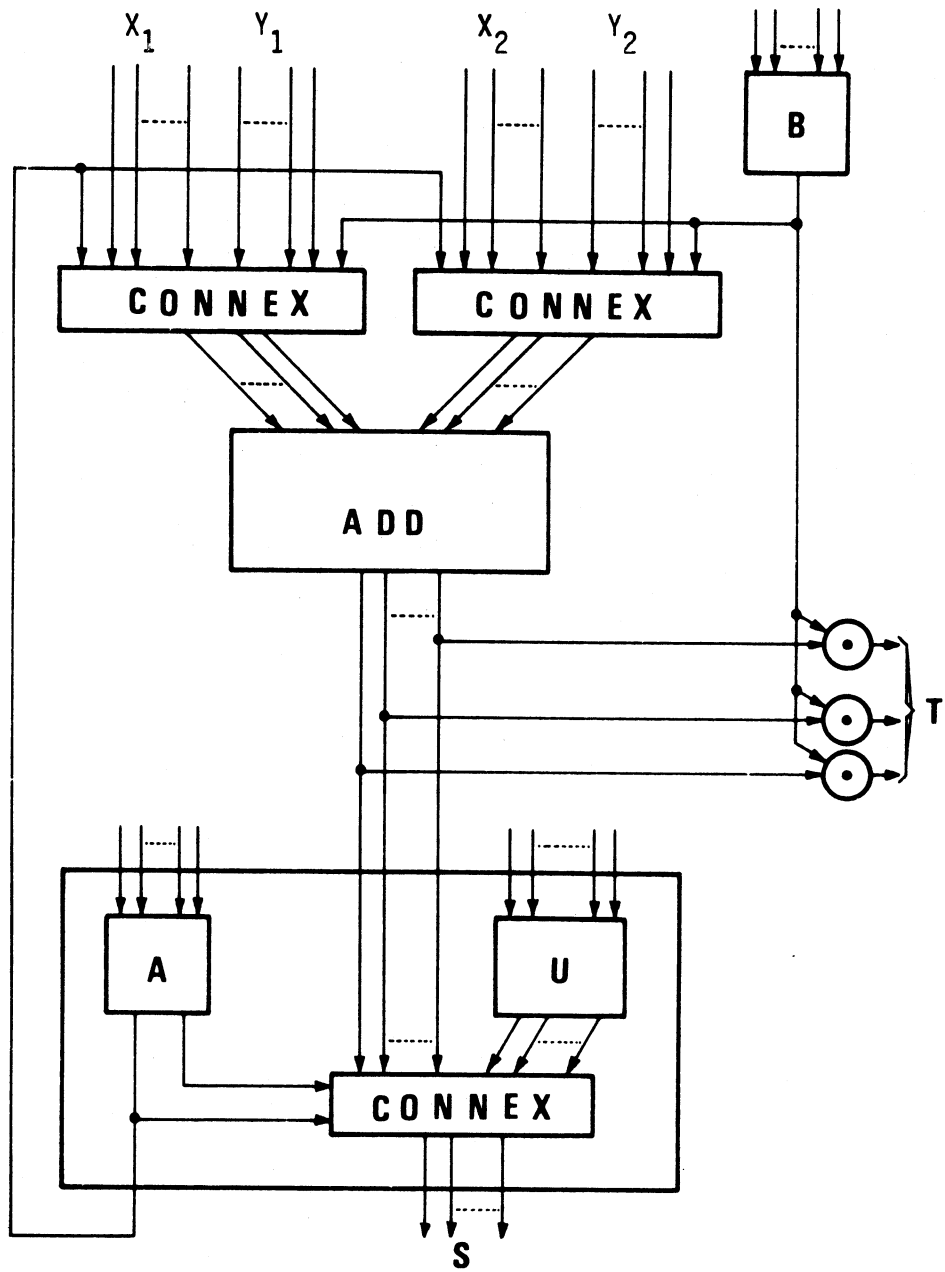


FIGURE 3

Cette description est équivalente à la suivante :

Si A alors (Z₁ := X₁; Z₂ := X₂) sinon si B alors (Z₁ := Y₁; Z₂ := Y₂) ;
Z₃ := ADD (Z₁, Z₂ ; *) ;
S := Si A alors Z₃ sinon U ;
Si B alors T := Z₃ ;

On voit donc la nécessité de conditionner les entrées de l'unité externe
ADD.

II - Les "SI" IMBRIQUES DANS LES EXPRESSIONS

Considérons une expression telle que :

Si A alors si B alors si C alors X sinon

Deux réalisations viennent à l'esprit :

L'une que nous appellerons "conditionnement en série" représentée sur
la fig | 1 |

L'autre que nous appellerons "conditionnement en cascade" représentée
sur la fig | 2 |

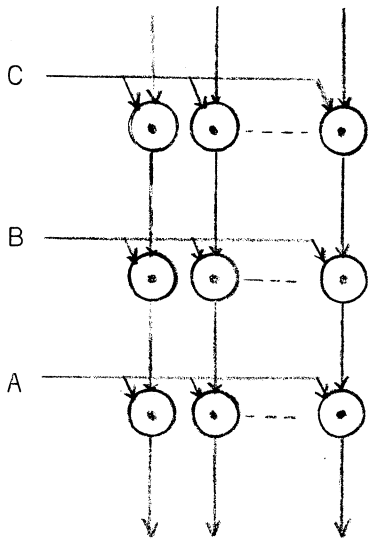


FIGURE 1

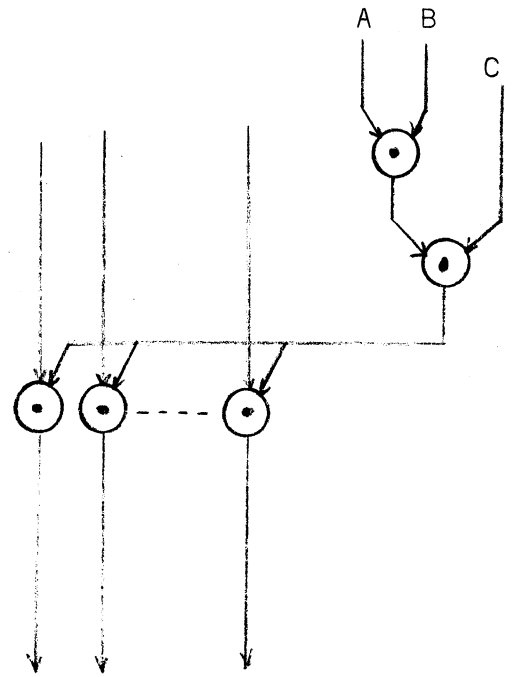


FIGURE 2

La seconde forme est beaucoup plus économique que la première, puisque si X est de dimension n, on a n+2 portes ET contre 3n.

Par contre, elle ne donne pas une structure régulière à l'expression dans le cas général, et introduit une dissymétrie entre le niveau le plus élevé de la structure emboîtée et les autres niveaux:

La réalisation de l'exemple suivant montre cette dissymétrie.

$S := \underline{\text{Si A alors (si B alors si C alors X sinon Y sinon Z) sinon T.}}$

où A,B,C,X,Y,Z et T sont des expressions simples. (figure 1 p 44)

Ceci rend donc pratiquement impossible un traitement récursif.

Nous avons donc opté pour le conditionnement en série, qui représente l'avantage de conserver une structure identique à tous les niveaux de récursivité de l'expression.

Par ailleurs, le gain de portes ET, s'il est sensible dans le cas particulier où l'on a la forme régulière :

si A alors (si B alors ... sinon ...) sinon si ...

l'est beaucoup moins, et peut même être nul, dans le cas général, ainsi que le montre l'exemple suivant :

S := Si A alors U + (si B alors V sinon W) sinon X ;

où A,B,V,W et X sont des expressions simples et U, une variable.

La figure |2| montre la réalisation avec le conditionnement en cascade, qui n'apporte aucun gain ici, puisqu'il faut conditionner la variable U.

La figure |3| donne la réalisation avec le conditionnement en série.

Dans les deux figures |2| et |3|, le rectangle en pointillé délimite l'expression U + (si B alors V sinon W) qui ne créerait pas d'unité dans ce cas puisque nous avons pris comme minimum trois externes pour créer une unité. ("trois" est le paramètre par défaut).

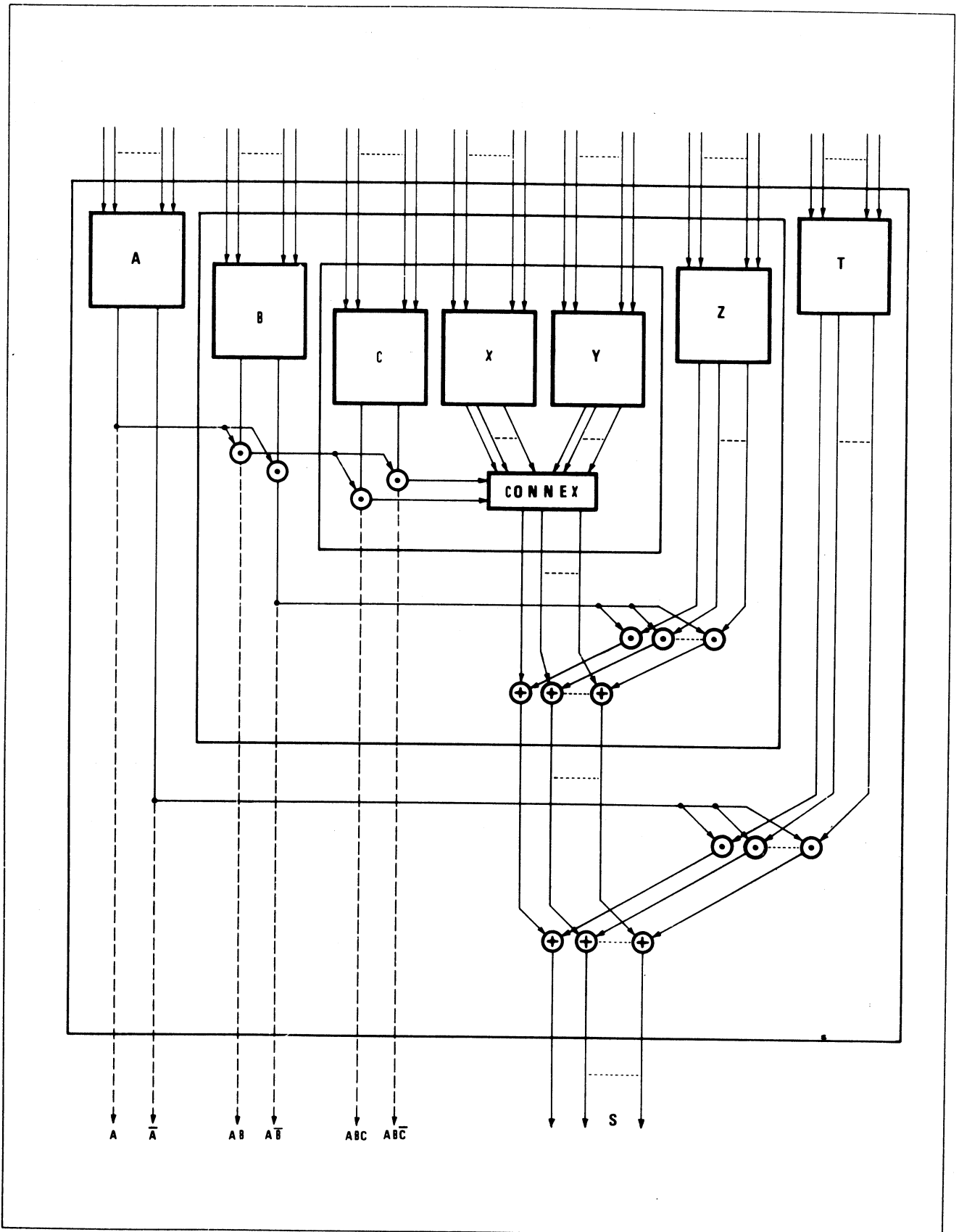


FIGURE 1 SI = SI A alors (si B alors si C alors X sinon y sinon Z) sinon T

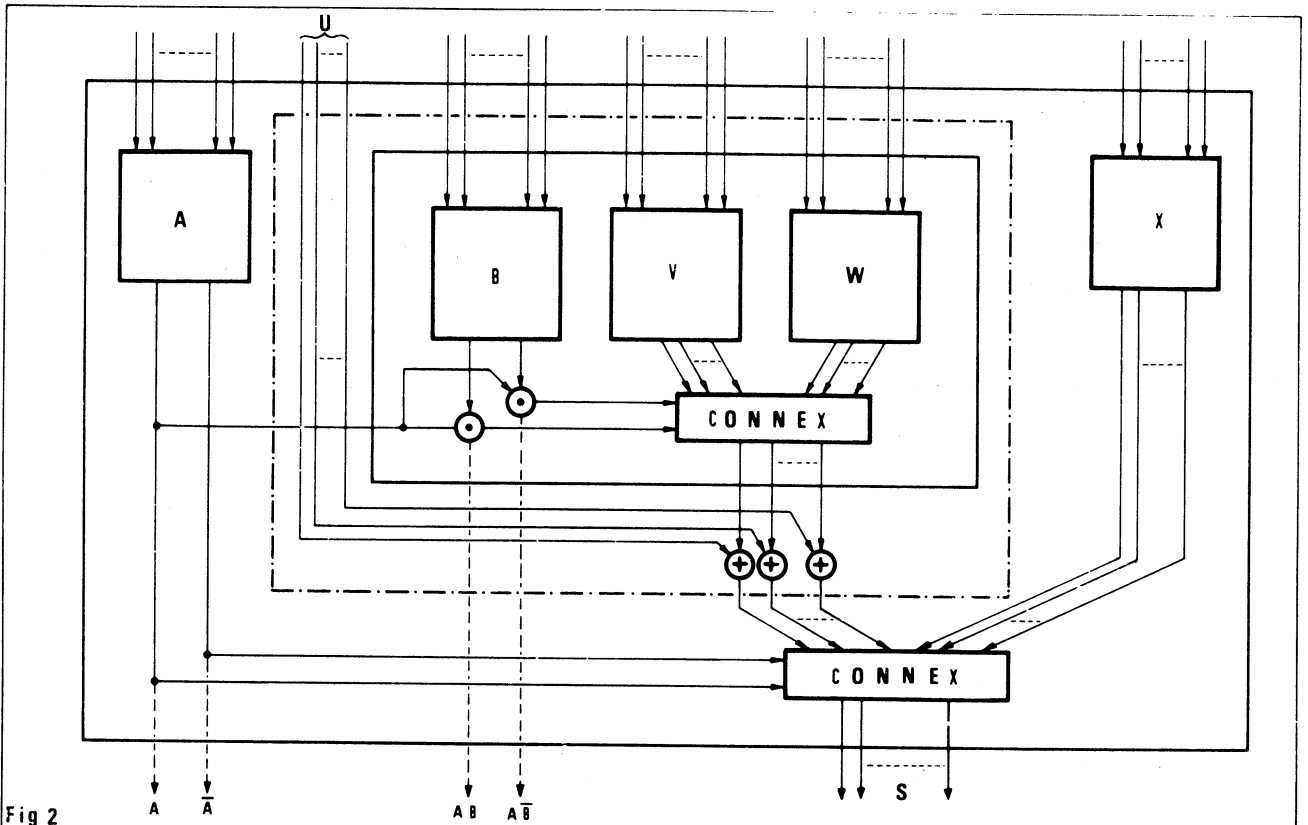


Fig 2

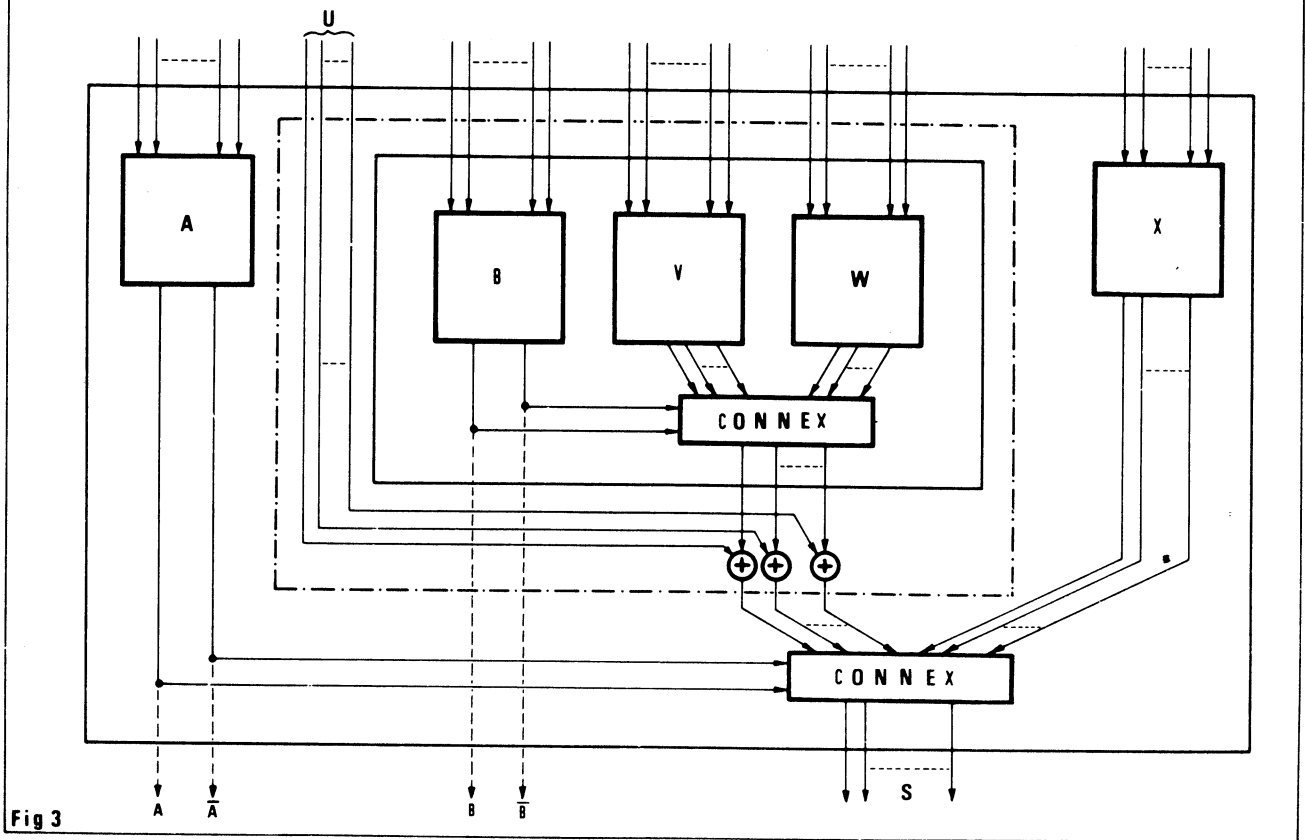


Fig 3

FIGURE 2 CONDITIONNEMENT EN CASCADE.
FIGURE 3 CONDITIONNEMENT EN SERIE.

III - ALGORITHME DE FORMATION DE LA STRUCTURE D'UNE EXPRESSION

Du fait de la génération simultanée des deux structures : élémentaire et emboîtée, nous avons besoin de conserver certaines indications au cours de l'analyse syntaxique.

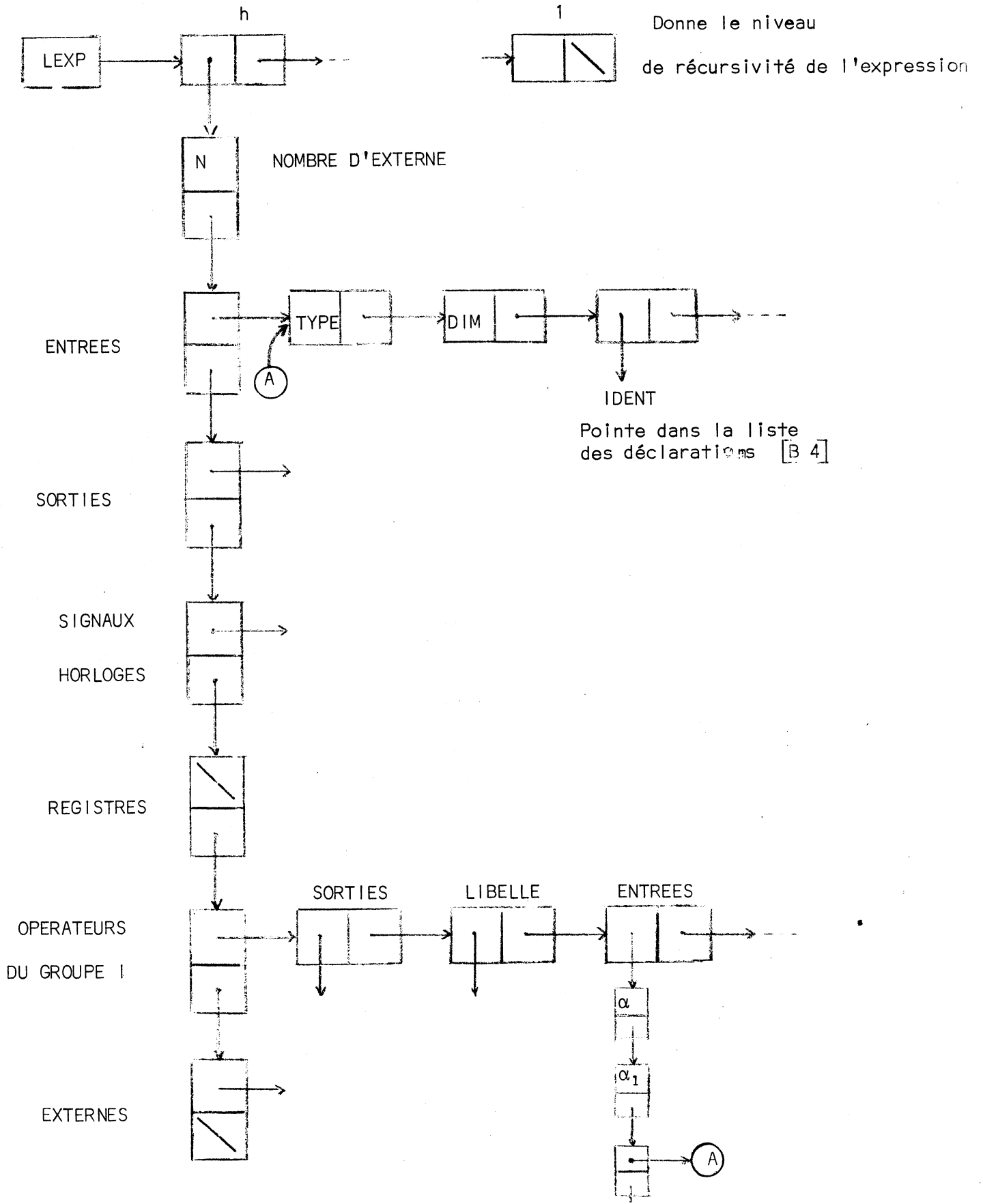
Nous utilisons pour ceci une liste dont la forme est donnée par la figure |p47|, qui a un triple rôle :

- permettre de créer la structure emboîtée.
- connaître la façon dont sont interconnectées les unités créées.
- servir de pile de récursivité pour l'expression : on initialise un nouvel élément dans la liste, chaque fois que l'on rentre dans une expression.

Une pile est utilisée pour la création de la structure élémentaire.

L'algorithme utilisé est expliqué, ci-dessous, de façon schématique, à l'aide d'une grammaire simplifiée de l'expression, comprenant les fonctions nécessaires.

La règle A est une règle quelconque où il y a occurrence d'une expression.



TYPE : Signal, Horloge
DIM : dimension
IDENT : identificateur (i.e le nom)
 α : nombre d'entrée
 α_1 : sélection de la dimension

Une nouvelle chaîne verticale est créée lorsque l'on rentre dans une expression. Elle est rendue à l'espace libre lorsque l'on en sort.

Grammaire simplifiée de l'expression

A f_1 EXP f_2 (i)

EXP si f_1 EXP f_3 (j) alors f_1 EXP f_2 sinon f_1 EXP f_2 (k) f_4
 f_5 TP

TP f_5 FP + TP1 f_6 (l)
 f_5 FP f_7

TP1 f_5 TP + TP1
 f_5 FP

FP PP . FP1 f_8 (m)
PP f_7 (n)

FP1 PP . FP1
PP

PP f_5 SP OPREL f_5 SP f_9 (o)
 f_5 SP

SP KP & SP1 f_{10}
KP f_7 (p)

SP1 KP & SP1
KP

KP f_{11} ID (q)
OPU KP f_{12}

- Conventions :

OPREL : opérateur de relation
ID : identificateur - Terminal pour la grammaire
OPU : opérateur unaire
TP,FP, ... : méta-variables
 f_i : fonction sémantique |P50 |
(I) : référence au tableau |P51 |

Les règles de l'expression mettent en évidence les priorités des opérateurs entre eux.

Ainsi : $A+B.C\&D > E./< F+G$

est interprété comme

$A+(B.((C\&D) >E).(/<F))+G$

REMARQUE :

Certaines règles ont été dédoublées afin de pouvoir utiliser des fonctions sémantiques plus simples et d'en faciliter l'implantation dans la grammaire. Ainsi les deux règles

TP FP + TP1

FP

TP1 FP + TP1

FP

peuvent s'écrire beaucoup plus simplement.

TP FP + TP

FP

Rôle des fonctions

- f_1 : initialise une nouvelle chaîne verticale dans la liste LEXP
- f_2 : crée une nouvelle unité à partir des éléments de la première chaîne verticale de la liste LEXP, s'il y a un nombre suffisant d'externes. L'unité créée est mise en externe sur la chaîne suivante si elle existe, sinon c'est une externe, pour l'unité en cours de traitement. Émet les ordres pour construire l'espace liste et la chaîne codée de cette nouvelle unité vers l'interpréteur [cf Partie Programmation]

Si la création d'une unité ne se justifie pas, les indications de la première chaîne sont reportées au niveau supérieur.

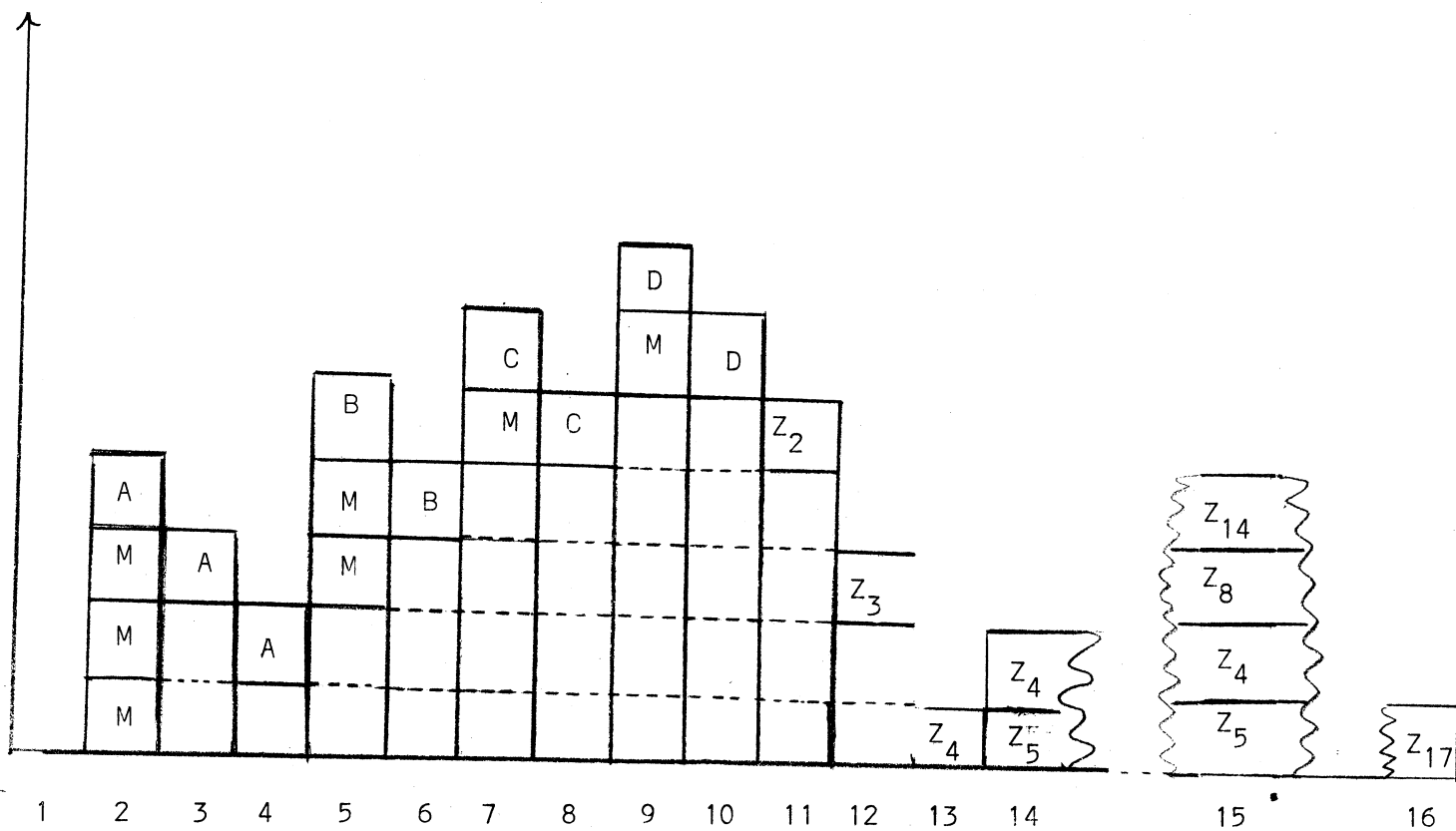
Dans les deux cas, la première chaîne est rendue à l'espace libre.

- f_3 : idem à la fonction f_2 , mais crée une sortie supplémentaire qui est la négation de la sortie normale.
- f_4 : crée la connexion des deux expressions sur la sortie.
- f_5 : empile une marque
- f_7 : supprime la marque qui se trouve sous le premier élément de la pile.
- f_q : en fonction de l'opérateur de relation utilisé, crée les éléments nécessaires à sa réalisation $|P \quad Z|$. Les entrées de l'opérateur sont les deux premiers éléments de la pile. Les éléments créés sont mis dans la première chaîne de la liste LEXP.
- f_{11} : empile l'identificateur qui suit C'est une entrée.
- f_{12} : selon l'opérateur unaire, on assure la réalisation.

- f_6 : +
- f_8 : assure la réalisation des opérateurs .
- f_{10} : &

Le diagramme ci-dessous donne l'évolution de la pile lors de l'analyse de l'expression suivante :

si A+B.C * D alors U.V.+X.Y sinon U.-V+*RX.Y

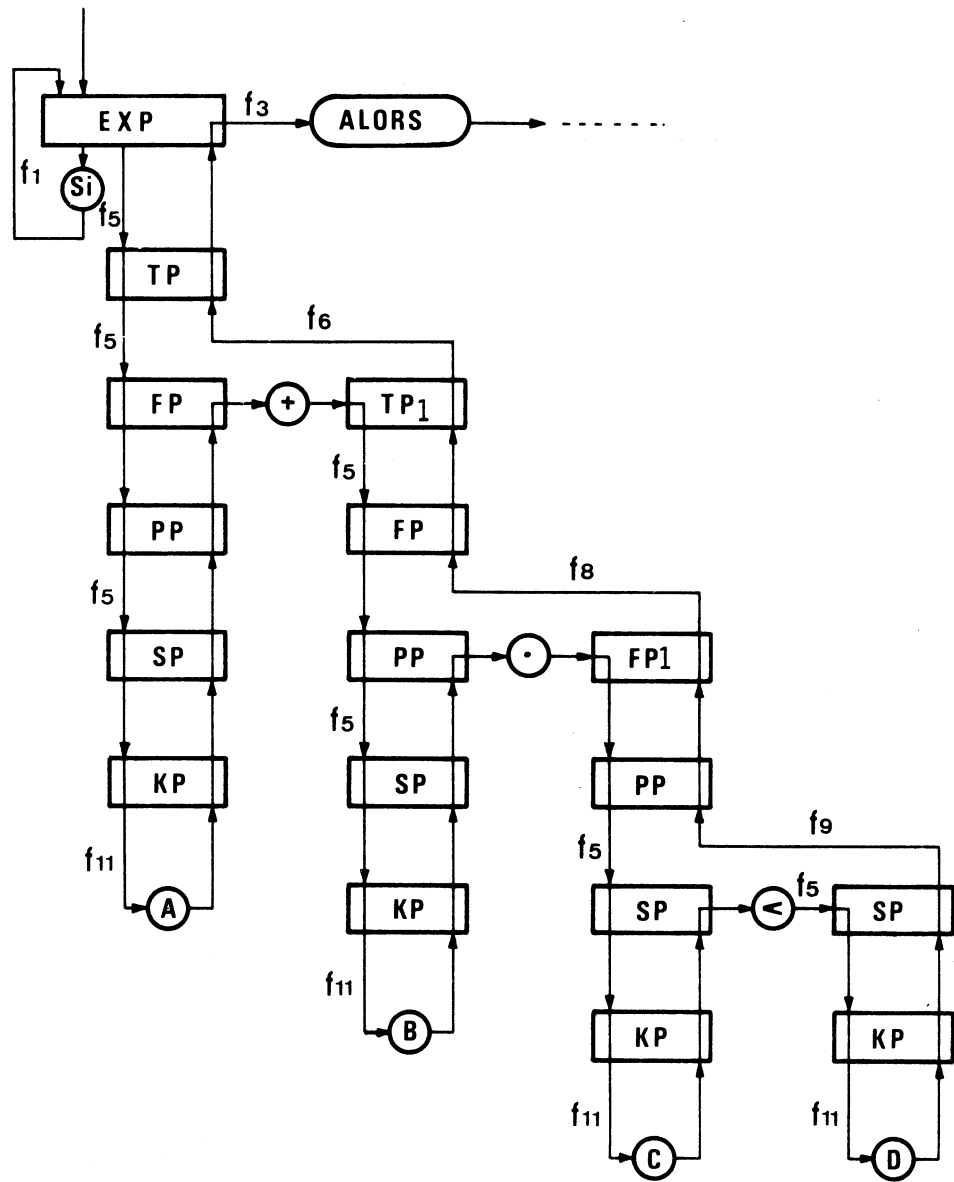


M : Marque

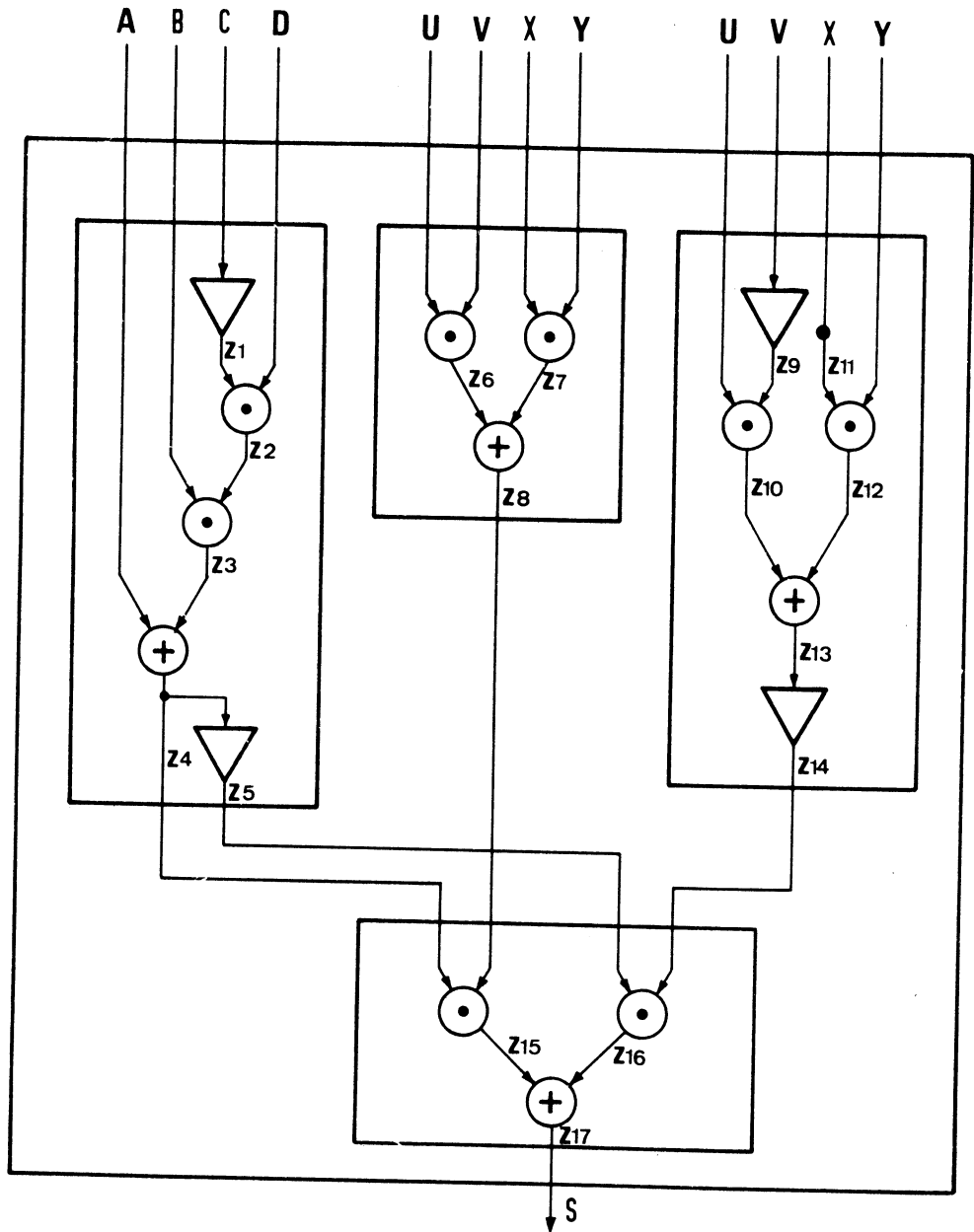
Z_i : Signal intermédiaire créée.

La correspondance entre les points de passages dans la grammaire et la numérotation du diagramme,est la suivante :

(i) : 16	(n) : 4
(j) : 14	(o) : 11
(k) : 15	(p) : 3,6,8,10
(l) : 13	(q) : 2,5,7,9
(m) : 12	



Analyse de $A+B.C < D$



Réalisation de l'expression :

$S := \text{si } A+B.C \text{ D alors } U.V.+X.Y \text{ sinon } U.-V+*RX.Y$

CHAPITRE IV

LES INSTRUCTIONS CONDITIONNELLES

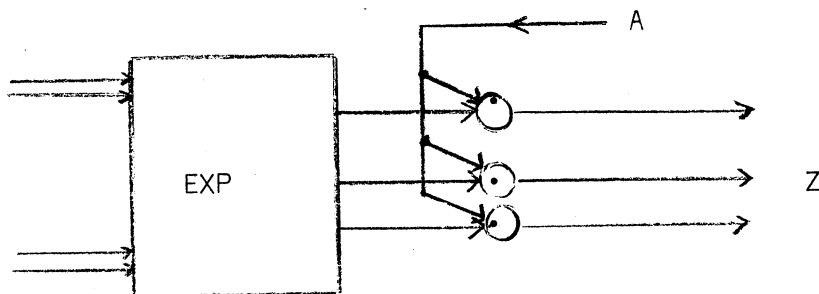
I - LES INSTRUCTIONS CONDITIONNELLES

Elles se situent à deux niveaux, le niveau inférieur étant celui des affectations, donc se plaçant obligatoirement sous une horloge. La carte syntaxique $| A_3 |$ et $| A_6 |$ montre clairement l'imbrication de l'instruction conditionnelle de branchement et de l'instruction conditionnelle d'affectation.

Ceci dit, il n'y a pas de différence de nature entre les deux. On peut considérer, en effet, que dans une affectation conditionnelle, le fait de placer l'horloge avant, est une abréviation commode sans plus. L'horloge se distribue sur toutes les actions élémentaires, sans influencer sur la forme conditionnelle.

Nous aurons donc la même structure dans les deux cas. Nous ferons porter la condition sur le branchement ou l'affectation proprement dits et non sur toutes les variables apparaissant dans une action élémentaire, ceci afin de minimiser les portes ET nécessaires, de la même manière que dans une expression conditionnelle.

Si A alors Z := EXP ;



La seule exception sera dans le cas où il y a un signal unité, puisqu'alors on doit conditionner les entrées comme nous l'avons vu.

Réalisation des conditions

Nous choisissons ici la structure en cascade puisque la forme imbriquée d'une instruction conditionnelle étant :

si C_1 alors (...; si C_2 alors...)(...)... sinon (...).

elle se ramène à la forme régulière:

si C_1 alors si C_2 alors

qui permet de générer aisément une structure en Cascade.

Les conditions seront réalisées dans une boîte noire dont les entrées seront les conditions élémentaires, et, les sorties, les conditions résultantes effectivement nécessaires.

Considérons l'exemple suivant, où a_1, a_2, a_3, a_4 et a_5 sont des actions élémentaires.

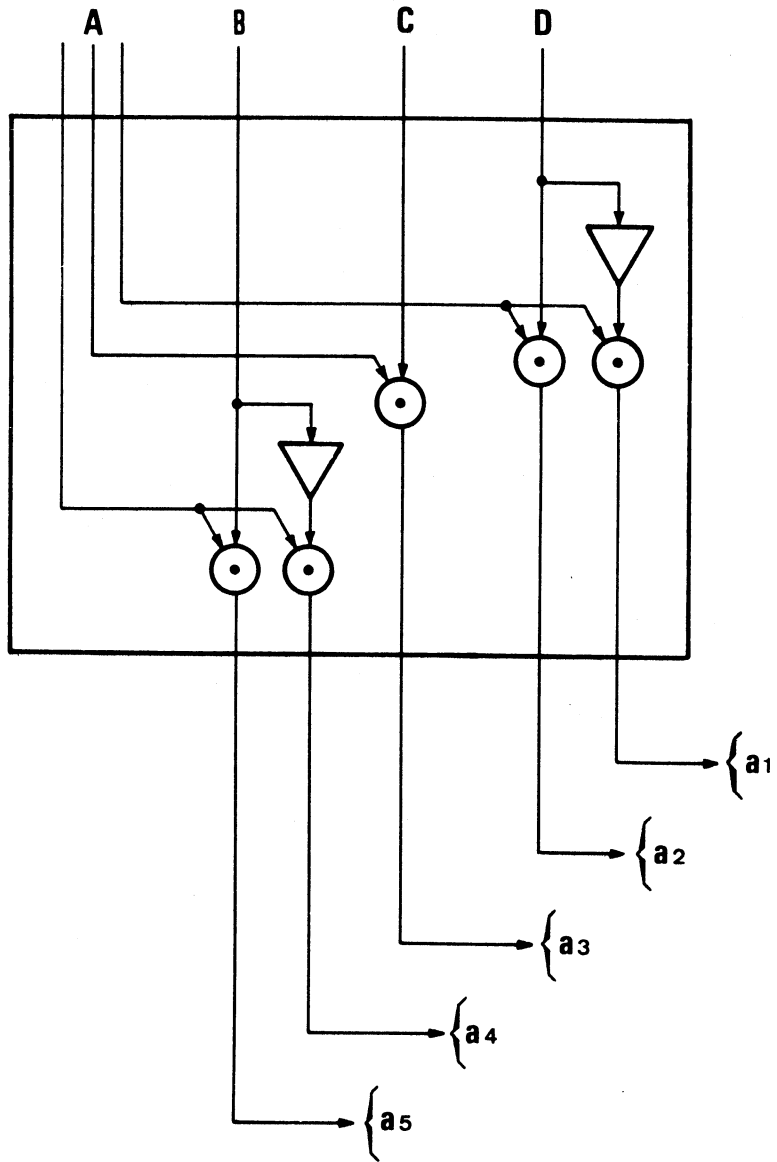
si $A(1:3)$ alors (si B alors a_1 sinon a_2) () (si C alors a_3)
sinon () (si D alors a_4 sinon a_5) () ;

La figure [P 57] montre l'unité réalisant les conditions conditionnant respectivement les actions élémentaires a_1, a_2, a_3, a_4 et a_5 .

Remarquons que si les entrées d'une telle unité peuvent être de dimension quelconque, toutes les sorties sont de dimension (1:1).

Les parenthèses vides correspondent à des conditions élémentaires non utilisées. Ainsi dans l'exemple $A(2)$ n'est pas utilisé, mais $\bar{A}(2)$ l'est. Il doit figurer autant de paires de parenthèses, éventuellement vides, qu'il y a d'éléments dans le tenseur de dimension de la condition.

Pour une condition de dimension $(1:n_1, 1:n_2, \dots, 1:n_p)$, on aura donc $n_1 \cdot n_2 \cdot \dots \cdot n_p$ paires de parenthèses.



Unité de conditionnement

Relation entre les deux niveaux

On a la forme :

Si A alors ... (... ; <H> si B alors ... (... , a_i , ...)...)...

La parenthèse étant conditionnée par le $i^{\text{ème}}$ élément que nous noterons A_i du tenseur A.

Nous supposerons tout d'abord que l'horloge ne dépend pas de la condition A_i , ceci, afin de séparer au maximum les horloges du reste de la logique.

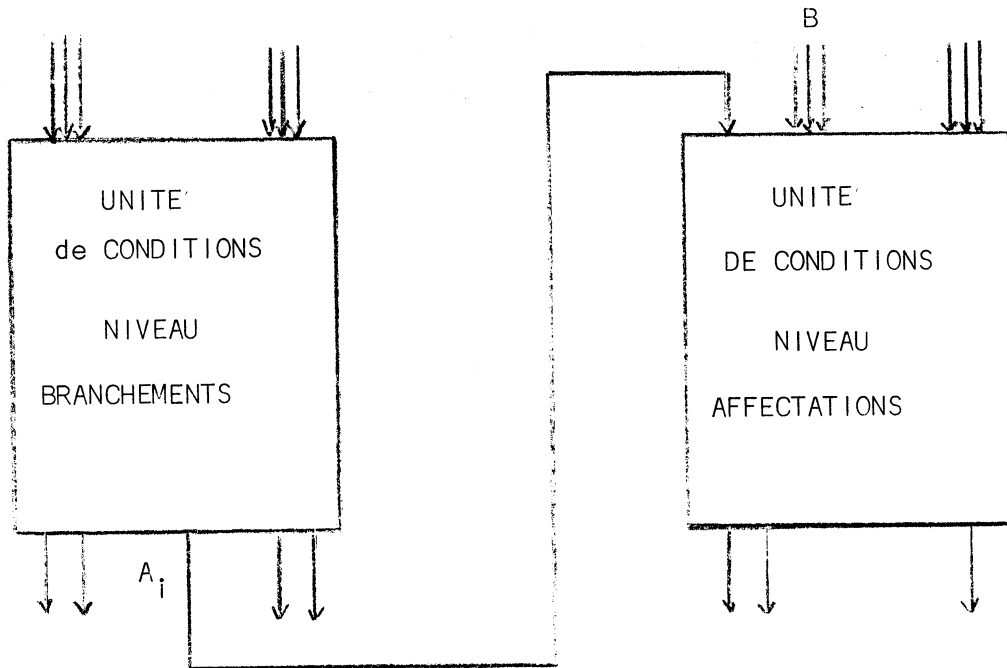
Notons que l'on pourrait prendre le point de vue inverse et conditionner uniquement l'horloge, la condition se répercutant alors sur toutes les actions élémentaires en dépendant. Il semblerait intéressant, dans un développement ultérieur du système de programmes, de permettre le choix entre ces deux options.

Nous introduirons donc la condition A_i au niveau de l'affectation conditionnelle comme un niveau supplémentaire de si c'est à dire que nous interprèterons de la façon suivante :

<H> si A_i alors si B alors ... (... , a_i , ...)...

La condition A_i étant une sortie de l'unité réalisant les conditions au niveau des branchements.

La figure montre cette liaison.



Algorithme pour la réalisation d'une unité de conditionnement.

Nous utilisons une pile. Pour une condition élémentaire, c'est-à-dire le résultat d'une expression ou d'une expression d'état, nous devons disposer de trois indications :

- l'identificateur du signal réalisant la condition élémentaire.
- le numéro de la composante en cours, puisque l'on a à faire à des variables tensorielles.
- un indicateur permettant de savoir si l'on est après un si ou après un alors.

On réalise une condition résultante, chaque fois que l'on rentre dans une parenthèse non vide en faisant le ET des deux éléments situés au sommet de la pile.

Les indications nécessaires à la génération de l'unité de condition sont conservées dans une liste de même forme que celle utilisée pour les expressions.

C H A P I T R E V

LES REGISTRES

On ne préjugera pas de la nature technologique de l'élément de mémorisation à ce niveau. Deux considérations nous en retiennent :

- la possibilité, en CASSANDRE, de spécifier le type de bascule utilisée par un numéro, lors de la déclaration de celle-ci dans l'unité qui l'utilise, imposerait une diversité de traitements difficiles à inclure dans l'analyseur syntaxique.

- Toute fonction bascule non prévue obligerait, pour en permettre le traitement, à incorporer une nouvelle partie aux routines utilisées par l'analyseur, donc à rassembler les modules correspondants.

Nous avons donc adopté la solution suivante :

créer une unité enfermant le registre, sur laquelle on pourra faire tout traitement désirable sans risque de perturber ce qui l'entoure, puisqu'une unité est entièrement indépendante des autres.

Les entrées d'une telle unité peuvent être réparties en trois catégories :

- les horloges
- les conditions éventuelles
- les variables.

Nous allons préciser ces catégories, et en justifier la nécessité.

I - CHARGEMENT D'UN REGISTRE

Il s'effectue nécessairement sous top d'horloge. On considèrera que quand le chargement d'un registre dépend de conditions, le registre ne change pas de valeur lorsque aucune des conditions n'est réalisée.

Ainsi l'instruction conditionnelle :

si A alors R ← X ;

est considérée comme équivalente à :

si A alors R ← X sinon R ← R ;

s'il n'y a aucune autre occurrence du registre R dans la description.

Nous voyons maintenant la nécessité de disposer des conditions en entrées de l'unité enfermant le registre.

Considérons l'exemple suivant :

si A alors R ← X ;

si B alors R ← Y ;

Le tableau ci-dessous donne les valeurs que doit prendre le registre R en fonction des conditions A,B et des variables X,Y.

A	B	X	Y	AX+BY	R	A+B	$(A+B) \oplus (AX+BY)$
0	0	0	0	0	R	0	0
0	0	0	1	0	R	0	0
0	0	1	0	0	R	0	0
0	0	1	1	0	R	0	0
0	1	0	0	0	0	1	1
0	1	0	1	1	1	1	0
0	1	1	0	0	0	1	1
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	0	1	1
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	0	0	1	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0

La colonne AX+BY ne suffit pas pour distinguer entièrement les trois possibilités pour R $\begin{bmatrix} 0 \\ 1 \\ R \end{bmatrix}$. Il faut aussi disposer de A+B afin de séparer les trois cas.

Si nous disposons ici d'une bascule JK par exemple, les branchements

$$J = AX+BY$$

$$K = (A+B) \oplus (AX+BY)$$

donneraient le fonctionnement désiré ainsi que l'on peut le voir, en considérant le tableau d'état de la bascule JK.

J_t	K_t	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	\bar{Q}_t

2 - LES HORLOGES EN ENTREE :

Il y aura toujours au moins une entrée de type horloge puisque toute affectation de registre s'effectue sous un top d'horloge.

Remarquons qu'il est parfaitement possible d'avoir des horloges différentes contrôlant le chargement d'un registre.

Par exemple :

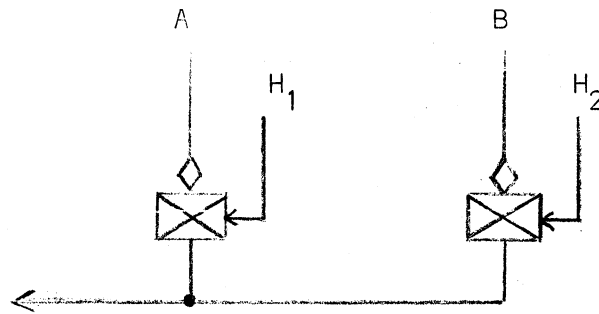
<H1> R \leftarrow A

<H2> R \leftarrow B

est syntaxiquement et sémantiquement correct en CASSANDRE.

La sémantique d'une telle description trouve sa justification par l'existence de certaines logiques.

Si, généralement, les bascules comportent une entrée pour l'horloge de synchronisation, il en existe sur lesquelles les entrées ne sont attaquables que par des impulsions. Les portes de type DCD (Diode Capacitor Diode) permettent de réaliser l'impulsion demandée par intersection de l'impulsion de l'horloge et du niveau du signal. [Cf Logic Handbook - Digital Equipment Corporation]



Les horloge jouent, dans ce cas, le même rôle que des conditions, et l'on peut réaliser des 'OU de connexion' sur les impulsions.

3 - LES CONDITIONS EN ENTREE

On branchera sur une telle entrée l'union des conditions C_i portant sur le registre, C_i pouvant être une combinaison de conditions élémentaires dans le cas de si imbriqués.

EXEMPLE :

Si A alors R ← X ;

Si B alors R ← Y sinon si C alors R ← Z ;

On entrera

$$A+B+\bar{B}C.$$

REMARQUE :

Nous sommes parfaitement conscient du fait qu'il suffit de rentrer l'union des conditions "primaires", c'est à dire ne considérer que les conditions individuelles, et non pas le résultat éventuel d'une suite de conditions (dans le cas de si imbriqués).

Ainsi dans l'exemple précédent, il suffit de rentrer :

$$A+B+C$$

Cette solution présente une difficulté de réalisation que nous n'avons pu surmonter pour l'instant.

Considérons les deux exemples suivants :

(1) Si A alors ... sinon si B alors $R \leftarrow X$

(2) si A alors $R \leftarrow X$ sinon si B alors $R \leftarrow Y$

Dans l'instruction (1), on n'a qu'une occurrence du registre R , et la condition est alors $\bar{A}B$.

Dans l'instruction (2) où il y a occurrence du registre R avec les conditions A et $\bar{A}B$, il suffirait de considérer $A+B$.

Au niveau de l'analyse syntaxique, ne connaissant pas les dimensions, il n'est pas possible de savoir si l'on a bien les mêmes parties du registre R dans un cas tel que l'instruction (2).

4 - LES VARIABLES EN ENTREE

On fournit, pour une position de registre, l'union des variables conditionnées chargeant cette position.

Dans l'exemple précédent, on rentre comme variable :

$$AX + BY + \bar{B}C Z.$$

Il faut montrer que l'on peut réaliser avec ceci, les fonctions à brancher aux entrées du registre.

Soit Z_1, \dots, Z_n des variables conditionnées par C_1, \dots, C_n . En général, le registre demande deux entrées $f_1(C_1, \dots, C_n, Z_1, \dots, Z_n)$ et $f_2(C_1, \dots, C_n, Z_1, \dots, Z_n)$.

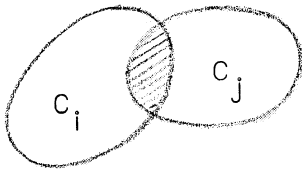
On dispose ici de ΣC_i et de $\Sigma C_i Z_i$ avec C_i de la forme $\bar{A}_i \dots \bar{A}_{i-1} A_i$.

Montrons que les fonctions $g(\Sigma C_i, \Sigma C_i Z_i)$ permettent de réaliser les entrées nécessaires.

Deux cas sont à considérer :

- les conditions sont "disjointes"
i.e un seul C_i vaut 1.
- les conditions sont "non disjointes"
i.e plusieurs C_i peuvent prendre la valeur 1 en même temps.

Ce dernier cas se ramène au précédent puisque l'on ne considère que les parties disjointes.



La partie hachurée est interdite car on aurait un chargement multiple.

Notons que le chargement multiple n'est détectable qu'à la simulation.

On a donc les deux situations suivantes :

- si $C_j = 1$ alors $C_i = 0 \forall i \neq j$

$$\text{donc } \sum C_i = 1 \text{ et } \sum C_i Z_i = Z_j$$

$$\text{d'où } g(\sum C_i, \sum C_i Z_i) = g(1, Z_j)$$

et d'autre part

$$f(C_1, \dots, C_n, Z_1, \dots, Z_n) = f(0, \dots, 1, \dots, 0, 0, \dots, Z_j, \dots, 0)$$

On peut réaliser les mêmes fonctions.

- $C_i = 0$ pour tout i

donc $C_i = 0$ et $C_i Z_i = 0$

On peut mettre n'importe quoi, donc, en particulier
 $g(0,0) = f(0, \dots, 0, Z_1, \dots, Z_n)$

EXEMPLE :

Bascule R.S.

Il suffit de mettre :

sur l'entrée S $\sum_i C_i Z_i$

sur l'entrée R $\sum_i C_i \overline{Z_i}$

c'est-à-dire dans le cas de l'exemple :

$$S := AX + BY + \overline{BC}Z$$

$$R := (A + B + \overline{BC}) \cdot \overline{(AX + BY + \overline{BC}Z)}$$

A B C	R+	S+	Q _{t+1}
0 0 0	0	0	Q _t
0 0 1	\overline{Z}	Z	Z
0 1 0	\overline{Y}	Y	Y
0 1 1	\overline{Y}	Y	Y
1 0 0	\overline{X}	X	X
1 0 1	$\overline{X+Z}$	X+Z	X+Z
1 1 0	$\overline{X+Y}$	X+Y	X+Y
1 1 1	$\overline{X+Y}$	X+Y	X+Y

On introduit ainsi une certaine redondance, difficile à éliminer par un traitement purement automatique, mais, qui, dans le cadre d'un traitement conversationnel pourra être réduite.

REMARQUE :

L'instruction conditionnelle utilisée pour le chargement d'un registre peut être cause d'aléas. Donnons en seulement un exemple :

Si A alors R ← X sinon si B alors R ← Y

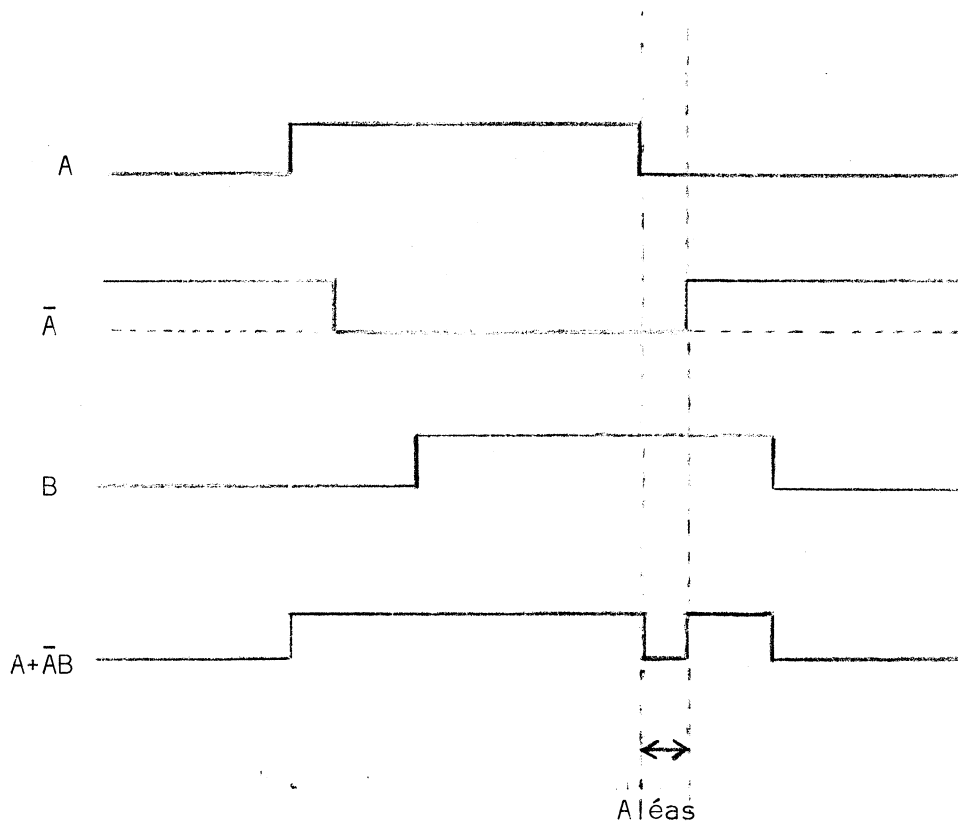
A et B étant des expressions quelconques.

Le registre R sera chargé avec

$$AX + \bar{A}BY.$$

Plaçons nous dans le cas ou X=Y=1

On a les niveaux suivants:



Le concepteur devra avoir présent à l'esprit ce risque d'aléas et en tenir compte pour le traitement particulier qu'il effectuera sur le registre.

Donnons maintenant un exemple :

'Registre' R(1:8) ;

<H> Si A alors R(1:4) ← X ;

<H> Si B alors R(5:8) ← Y sinon si C alors R ← Z ;

On aura une unité comme suit :

'Unité' R R(H,C₁,E₁,C₂,E₂ ; S₁)

'Registre' R(1:8) ;

'horlogemère' H ;

<H> si C₁ alors R(1:4) ← E₁ ;

<H> si C₂ alors R(5:8) ← E₂ ;

S₁ := R ;

Cette unité étant branchée dans l'unité appelante avec sur ses entrées :

RR(H,A+ $\bar{B}C$,AX+ $\bar{B}CZ$ (1:4),B+ $\bar{B}C$,BY+ $\bar{B}CZ$ (5:8);S) ;

CHAPITRE VI

PROBLEMES DIVERS

I - PARTITION DES VARIABLES

Il peut être nécessaire de partitionner les variables apparaissant en partie gauche d'un branchement ou d'une affectation afin d'effectuer les connexions correctes. Ainsi, si la variable $S(1:16)$ est utilisée comme suit :

Si C alors $S(1:12) := A$ sinon $S(8:16) := B$

Nous devons la partitionner :

$S(1:7) := C.A$

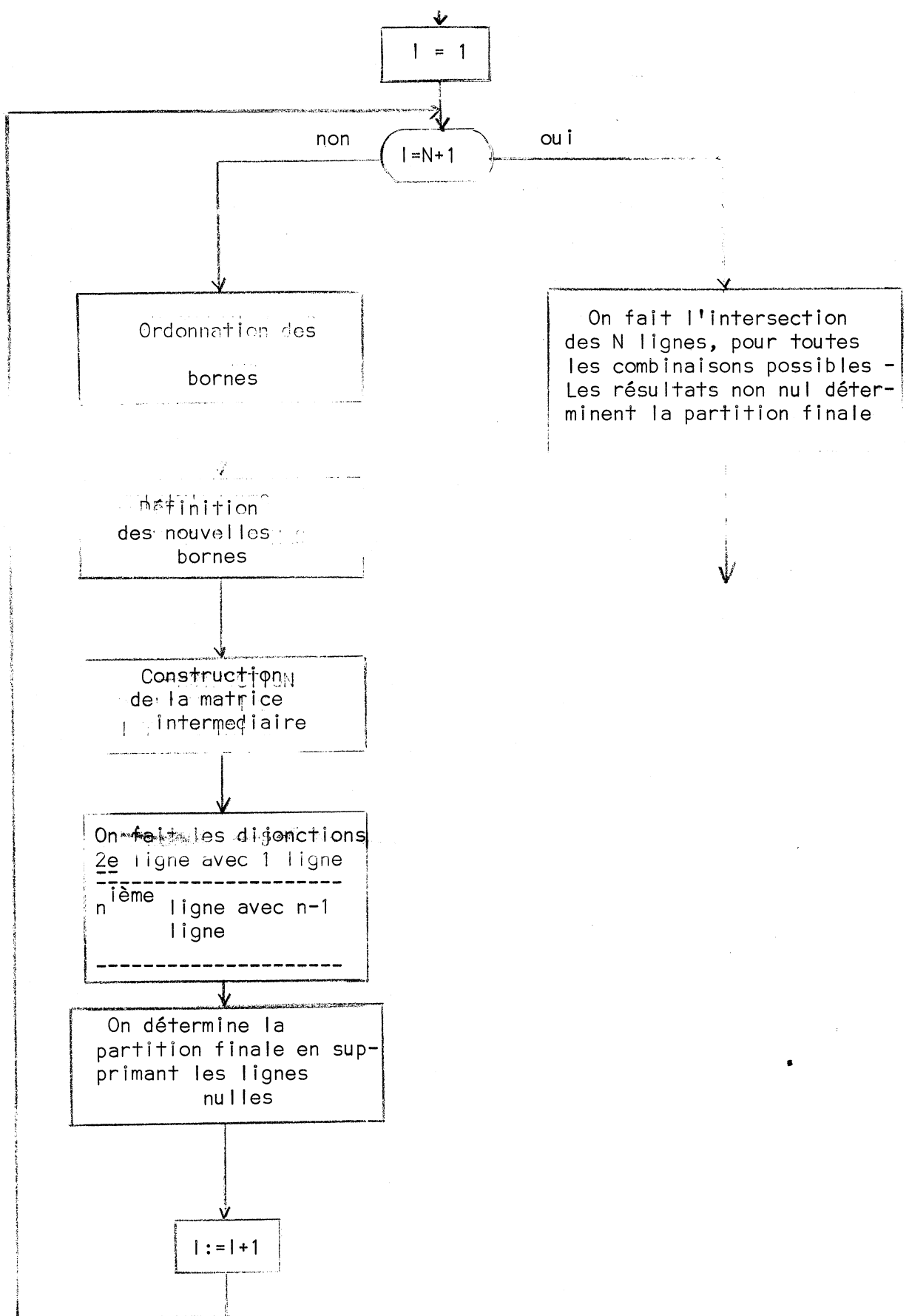
$S(8:12) := C.A + \bar{C}B$

$S(13:16) := \bar{C}B$

Pour les raisons citées chapitre III p35, nous nous contenterons de faire la partition la plus fine.

L'organigramme suivant décrit l'algorithme utilisé.

Nous donnons ensuite un exemple.



ALGORITHME DE PARTITION DES VARIABLES

Soit une variable de dimension tensorielle à N composantes.

Plaçons nous dans une des composantes, soit p le nombre de dimensions utilisées pour cette composante : Les dimensions sont spécifiées par leurs bornes :

$$(g_1;d_1), (g_2;d_2) \dots, (g_p;d_p)$$

Ordonation des bornes

On ordonne les bornes par ordre croissant. A chaque borne on associe le branchement qui est effectué. On a donc maintenant une suite croissante de $2p$ entiers.

Définition des nouvelles bornes

Les nouvelles bornes sont définies à partir de cette suite de la façon suivante :

- les deux éléments extrêmes ne sont pas touchés.
- A chaque entier k, on associe un couple :

$(k-1, k)$ si k était borne gauche

$(k, k+1)$ si k était borne droite

On a une nouvelle suite croissante de $4p-2$ entiers soit $2p-1$ nouvelles paires de bornes.

Construction de la matrice intermédiaire :

Cette matrice a $2p-1$ lignes correspondant aux nouvelles paires de bornes, et p colonnes correspondant aux branchements élémentaires possibles. Dans chaque ligne, il y a un seul élément à 1, qui est déterminé par la borne gauche g :

si $g = k$ ancienne borne gauche, on prend le branchement relatif à k

si $g = k+1$, on prend le branchement relatif à k.

EXEMPLE

S(1:8) := T
 S(3:7) := X
 S(6:14) := Y
 S(16:20) := Z

1	T	G
3	X	G
6	Y	G
7	X	D
8	T	D
14	Y	D
16	Z	G
20	Z	D

Tableau des bornes ordonnées

G : borne gauche

D : borne droite

	T	X	Y	Z
1 2	1	0	0	0
3 5	0	1	0	0
6 7	0	0	1	0
8 8	0	1	0	0
9 14	1	0	0	0
15 15	0	0	1	0
16 20	0	0	0	1

Matrice intermédiaire

S(1:2) := T
 S(3:5) := T+X
 S(6:7) := T+X+Y
 S(8:8) := T+Y
 S(9:14) := Y
 S(16:20) := Z

	T	X	Y	Z
1:2	1	0	0	0
3:5	1	1	0	0
6:7	1	1	1	0
8:8	1	0	1	0
9:14	0	0	1	0
15:15	0	0	0	0
16:20	0	0	0	1

Partition avec les branchements à effectuer

II - COÛT D'UNE UNITE

Le calcul du coût d'une unité peut se faire assez aisément, maintenant que sa description ne comporte plus que des connexions d'unités. Le calcul se fera récursivement à partir du coût donné des unités élémentaires.

Pour une unité U possédant p externes U_1, \dots, U_p apparaissant respectivement k_1, \dots, k_p fois (où k_i est le nombre de numéros de duplication de l'externe U_i), on a :

$$\text{Coût (U)} = \sum_{j=1}^P k_j \text{ coût (U}_j)$$

Il semble intéressant de pouvoir introduire en données, en plus du coût des unités élémentaires, éventuellement, celui d'unités de niveau plus élevé.

Remarquons qu'il ne sera pas nécessaire d'exécuter les boucles POUR pour ce calcul : dans la liste de déclarations, à chaque unité externe est associée la liste de tous les numéros de duplication qu'elle utilise. Le compte est donc aisé.

III - NOMBRE DE FILS DE CONNEXION D'UNE UNITE

Il se déduit immédiatement de l'entête de l'unité. Il suffit de faire la somme des dimensions des entrées et des sorties qui figurent dans cette entête.

IV - REGROUPEMENT D'UNITES

On peut désirer regrouper à l'intérieur d'une unité un certain nombre d'externes en un seul, c'est à dire rajouter un niveau à la structure emboîtée. Il suffit pour cela de retirer de l'unité traitée un certain nombre de déclarations qui deviendront les déclarations du nouvel externe créé. Il faut aussi modifier la chaîne codée décrivant la partie fonctionnelle, donc en faire l'analyse afin d'y apporter les corrections nécessaires.

S E C O N D E P A R T I E

LE SYSTEME DE PROGRAMMES

Comme toutes les applications du langage CASSANDRE, le point de départ du découpage est la bibliothèque d'Unités CASSANDRE précompilées. Les éléments de cette bibliothèque sont obtenus à partir d'un texte CASSANDRE par passage dans la chaîne de traitement général décrite plus loin |p77 |•

Forme d'une unité CASSANDRE en bibliothèque

A une unité correspondent deux fichiers :

- un espace liste, où sont chaînées toutes les "déclarations" de l'unité.

- une chaîne codée, qui n'est autre que la partie fonctionnelle de l'unité, dont tous les symboles de base sont codés. Tous les éléments déclarés (signaux, registres, etc...) sont aussi codés par leur adresse relative dans l'espace liste. Aucune modification n'est apportée au texte source.

Principes de programmation

Tous les programmes sont écrits en Assembleur et Macro-Assembleur 360.

Les divers analyseurs syntaxiques utilisés dans les programmes sont obtenu en utilisant le système de programmes écrits par M. GRIFFITHS et M. PELTIER (IMAG, IBM FRANCE / CENTRE SCIENTIFIQUE), qui permet la génération automatique d'un module syntaxique, à partir d'une grammaire mise sous forme appropriée, et pouvant contenir des appels à des fonctions sémantiques situées dans d'autres modules du programme. |B2| |B5|

Le système est, pour l'instant, destiné à fonctionner sous CP-CMS.

L'arithmétique du langage, nous a conduit à générer un code interprétable, pour éviter de faire des retours en arrière lors de l'analyse des boucles "POUR".

I - PROGRAMMES DE TRAITEMENT GENERAUX

Ces programmes ont été réalisés par Monsieur ANCEAU et Monsieur DOUSSY. Nous avons, pour notre part, collaboré à leurs définitions.

Rappelons brièvement leurs fonctions.

EDITEUR

Ce programme lit le texte Source et fournit une chaîne dans laquelle les différents symboles de base sont codés et les blancs supprimés.

Il se compose uniquement d'un module syntaxique qui provoque la sortie des caractères du code sur un fichier récupérable par le programme suivant. Les identificateurs ne sont pas codés, seuls, leurs caractères le sont.

PROGRAMME DE VERIFICATION.

Ce programme se compose d'un superviseur destiné à gérer les deux modules suivants :

Module de vérification syntaxique et d'analyse des déclarations.

Ce module lit le texte édité et fournit à la bibliothèque, d'une part une chaîne complètement codée sans déclarations, d'autre part, les déclarations mises sous forme de listes, et génère un texte interprétable par le module de vérification dimensionnelle. Dans l'analyse du corps du texte, des fonctions sémantiques exploitent les listes construites au moment des déclarations, pour fournir la nature et les dimensions des identificateurs rencontrés et permettent ainsi leur codage, qui est simplement l'adresse de la liste les décrivant.

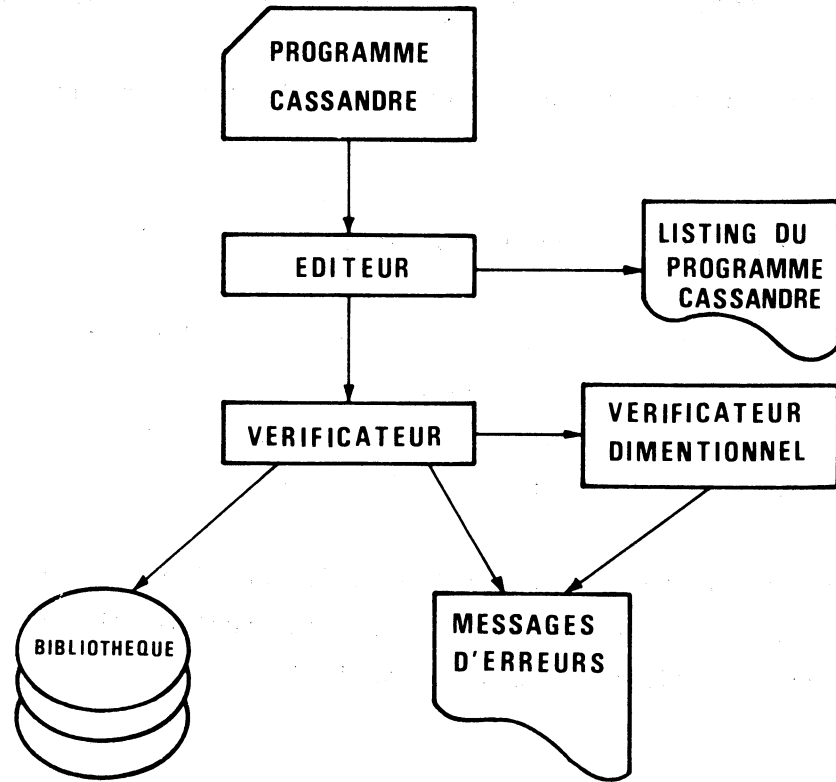
Module de vérification dimensionnelle.

Ce module est un interpréteur qui utilise le code généré à son intention par le module précédent, ses seules sorties sont des messages d'erreurs.

Les instructions de ce code peuvent être de nature :

- arithmétique
- branchement, dans le cas de boucles "pour" ou de "si" arithmétiques,
- dimensionnels
 - . prise en compte des dimensions d'un tenseur
 - . modification des dimensions par les opérateurs de réduction de concaténation et de transposition
- test de compatibilité de dimension.

Ce module ajoute aux listes de déclaration, des renseignements issus de l'exécution des ordres arithmétiques : les numéros d'identification des unités externes utilisées dans une description.



PROGRAMMES DE TRAITEMENTS GENERAUX

II - ORGANISATION DE LA CHAÎNE DES PROGRAMMES DE DECOUPAGE

Elle se compose de trois programmes principaux :

- un analyseur syntaxique
- un interpréteur
- un module de sortie des résultats.

L'organigramme |p82| montre l'enchaînement de ces programmes.

Analyseur syntaxique

Décide de la création de nouvelles unités. Si une unité est créée, il prépare tous les éléments nécessaires à sa réalisation. Sa sortie est une chaîne d'ordres pour l'interpréteur, chaque ordre étant suivi d'un certain nombre de paramètres.

Toutes les listes de travail sont créées dans l'espace liste de l'unité traitée.

Les dimensions ne peuvent pas être calculées au cours de l'analyse car celle-ci se déroule sans "back tracking". L'analyseur émet les ordres nécessaires au calcul des dimensions pour l'interpréteur.

Interpreteur

Il calcule les dimensions d'après les ordres émis par l'analyseur, et fabrique l'espace liste et la chaîne codée de chaque unité créée qui sont rangés dans la bibliothèque. L'exécution des boucles "POUR" a lieu à ce niveau.

Module de sortie des résultats.

Ce module se compose de deux parties :

- un premier programme qui ressort l'espace liste en clair, c'est à dire toutes les déclarations.

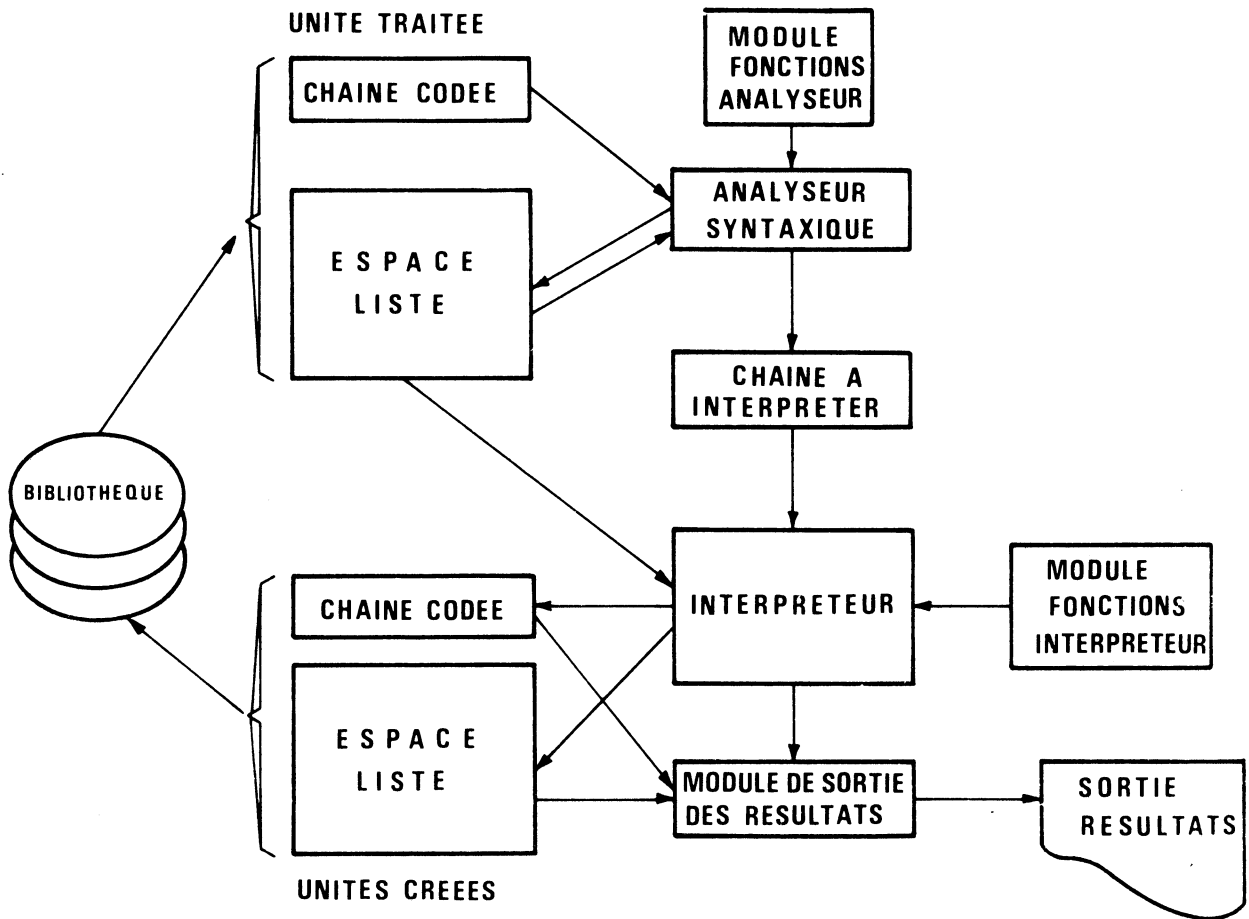
- un analyseur qui sort la chaîne codée en clair. Nous avons employé un analyseur syntaxique ce qui permet de mettre en forme le texte.

Ce module peut être utilisé indépendamment pour sortir en clair n'importe quelle unité de la bibliothèque.

Une option permet sous C.M.S. de recréer un fichier du texte source afin d'utiliser les possibilités de l'éditeur C.M.S. pour les modifications que l'on veut effectuer sur le texte.

REMARQUE :

Le vérificateur n'accepte que les lettres et les chiffres dans le nom d'une unité. Ce nom, tronqué à huit caractères est le nom des deux fichiers obtenus en sortie du vérificateur. Afin d'éviter de détruire un tel fichier, tous les fichiers créés par le programme de découpage sont nommés avec un symbole "&" en tête. On évite ainsi toute confusion.



ENCHAINEMENT DES PROGRAMMES DE DECOUPAGE

EXEMPLE

Nous allons donner en exemple la réalisation d'un multiplieur classique. |B-1|

Il s'agit d'un multiplieur 16 bits × 16 bits qui comprend :

Un registre CA de 33 bits dont on utilise les poids faibles pour stocker le multiplicateur. Le résultat sera disponible dans ce registre.

Un registre COMPT de 4 bits qui sert de compteur

Un additionneur FADD.

La multiplication se fait par additions et décalages successifs, l'exploration du multiplicateur s'effectuant de la droite vers la gauche.

Soit A le multiplicande et B le multiplicateur.

Le produit $P = A.B$ est obtenu par la récurrence :

$$P := \frac{1}{2} P + A b_i$$

avec
$$B = \sum_{i=0}^n 2^i b_i$$

Le schéma |p 86| montre l'organisation générale de ce multiplieur.

On suppose l'initialisation faite de l'extérieur (en utilisant par exemple le forçage d'état d'une unité externe [p. 32])

La description initiale : Unité MULT est donnée p.87
Elle comporte quatre états dont nous donnons rapidement les fonctions :

Le symbole *E permet de générer un vecteur, ou plus généralement un tenseur de 1 de la dimension spécifiée. On en prend la négation afin d'obtenir un vecteur de 0.

(Un symbole permettant la génération d'un tenseur de 0 est en cours d'implémentation).

ETA0

- initialisation du compteur à zéro.
- chargement du multiplicateur dans les poids faibles du registre CA, avec mise à zéro des poids forts.

ETA1

- +1 sur le compteur
- addition du multiplicande aux poids forts du registre si le bit du multiplieur est à 1 (instruction "Si" CA(33) alors...)

ETA2

- décalage à droite du registre CA
- test du compteur pour savoir si l'opération est terminée.

ETA3

- le résultat est disponible.

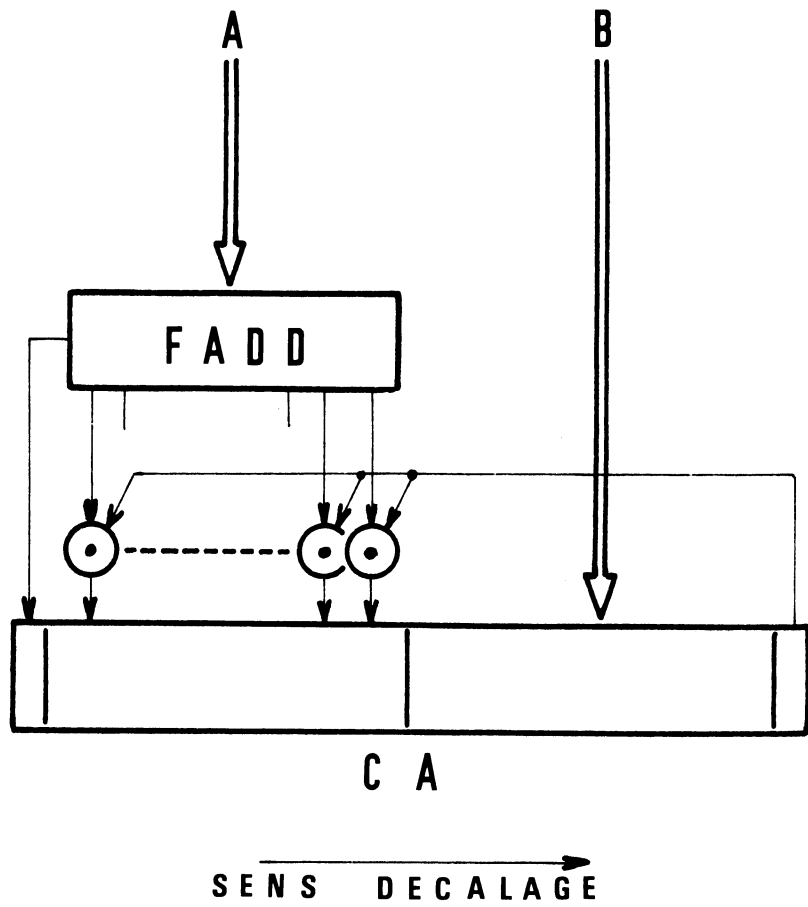
Les unités créées appellent quelques remarques :

L'unité de contrôle &TMULT01 et les deux unités de registres &RMULT01, &RMULT02 possèdent autant d'entrées d'horloges qu'il y a

d'occurrence de l'instruction allera (ou de chargement de registres). On a donc des entrées surabondantes.

Ceci est en cours d'amélioration. On va tester les horloges afin de n'avoir qu'une entrée par horloge différente.

Les inverseurs générés sont inutile. (Constante 0 est généralement disponible). Ceci nous a montré la nécessité d'introduire un symbole pour générer une telle constante.



'UNITE' MULT(H(1:1),A(1:16),B(1:16);C(1:32));

'REGISTRE' CONTR(1:4),CA(1:33);

'SIGNAL' RCONTR(1:4);

'HORLOGEMERE' H(1:1);

'EXTERNE' FADD((1:16),(1:16);(1:17));

ETA0:

<H>

CONTR<--*E(4),

CA<--*E(17)&B,

'ALLERA' ETA1;

ETA1:

FADD(A,CA(2:17));

RCONTR:=*D|1|(CONTR.RCONTR)&1;

<H>

'SI' CA(33) 'ALORS'

CA(1:17)<=FADD(,;*),

CONTR<=RCONTR#CONTR,

'ALLERA' ETA2;

ETA2:

<H>

CA<=0&*D|-1|CA,

'ALLERA' 'SI' /+CONTR 'ALORS' ETA1 'SINON' ETA3;

ETA3:

C:=CA(2:33);

'UNITE' &XMULT01(EE01(1:4),EE02(1:4);SS01(1:4));
 'SIGNAL' XX04(1:4),XX03(1:4),XX02(1:4),XX01(1:4);
 'EXTERNE' &NEG((1:1);(1:1)),&ETS02((1:1),(1:1);(1:1)),&OUS02((1:1),(1:1);(1:1));

'POUR' I1=1 'A' 4

'DEBUT'

XX01(1:1)=-&NEG|I1|(EE01(1:1);*);
 XX02(1:1)=-&ETS02|I1|(XX01(1:1),EE02(1:1);*);
 XX03(1:1)=-&NEG|4+I1|(EE02(1:1);*);
 XX04(1:1)=-&ETS02|4+I1|(EE01(1:1),XX03(1:1);*);
 SS01(1:1)=-&OUS02|I1|(XX02(1:1),XX04(1:1);*);

'FIN';

'UNITE' &RMULT01(HH01(1:1),EE01(1:1),EE02(1:4),HH02(1:1),EE03(1:1),EE04(1:4);SS01(1:4));
 'REGISTRE' CONTR(1:4);
 'HORLOGEMERE' HH01(1:1),HH02(1:1);

SS01(1:4)=-CONTR(1:4);

<HH01(1:1)>

'SI' EE01(1:1) 'ALORS'

CONTR(1:4)<=EE02(1:4);

<HH02(1:1)>

'SI' EE03(1:1) 'ALORS'

CONTR(1:4)<=EE04(1:4);

'UNITE' &RMULT02(HH01(1:1),EE01(1:1),EE02(1:33),HH02(1:1),EE03(1:1),EE04(1:17),HH03(1:1),EE05(1:1),EE06(1:33);SS01(1:33));

'REGISTRE' CA(1:33);

'HORLOGEMERE' HH01(1:1),HH02(1:1),HH03(1:1);

SS01(1:33)=-CA(1:33);

<HH01(1:1)>

'SI' EE01(1:1) 'ALORS'

CA(1:33)<=EE02(1:33);

<HH02(1:1)>

'SI' EE03(1:1) 'ALORS'

CA(1:17)<=EE04(1:17);

<HH03(1:1)>

'SI' EE05(1:1) 'ALORS'

CA(1:33)<=EE06(1:33);

'UNITE' &MULT(H(1:1),A(1:16),B(1:16);C(1:32));

'SIGNAL' CONTR(1:4),CA(1:33),AZ25(1:32),AZ24(1:16),AZ23(1:16),AZ22(1:4),AZ21(1:33),AZ20(1:17),AZ19(1:1),AZ18(1:33),AZ17(1:4),AZ16(1:4),AZ15(1:1),AZ14(1:33),AZ13(1:32),AZ12(1:4),AZ07(1:17),AZ06(1:4),AZ05(1:3),AZ04(1:4),AZ03(1:33),AZ02(1:17),AZ01(1:4),AY01(1:4),RCONTR(1:4);

'HORLOGE:MERE' H(1:1);

'EXTERNE' FADD((1:16),(1:16);(1:17)),&TMULT01('HORLOGE' (1:1),'HORLOGE' (1:1),'HORLOGE' (1:1),(1:1);(1:4)),&RMULT01('HORLOGE' (1:1),(1:1),(1:4),'HORLOGE' (1:1),(1:1),(1:4);(1:4)),&RMULT02('HORLOGE' (1:1),(1:1),(1:33),'HORLOGE' (1:1),(1:1),(1:7),'HORLOGE' (1:1),(1:1),(1:33);(1:33)),&XMULT01((1:4),(1:4);(1:4)),&NEG((1:1);(1:1)),&ETS02((1:1),(1:1);(1:1)),&OUS04((1:1),(1:1),(1:1),(1:1);(1:1));

&TMULT01(H(1:1),H(1:1),H(1:1),AZ15(1:1);AY01(1:4));

&RMULT01(H(1:1),AY01(1),AZ16(1:4),H(1:1),AY01(2),AZ17(1:4);CONTR(1:4));

&RMULT02(H(1:1),AY01(1),AZ18(1:33),H(1:1),AZ19(1:1),AZ20(1:7),H(1:1),AY01(3),AZ21(1:33);CA(1:33));

&XMULT01(RCONTR(1:4),CONTR(1:4);AZ12(1:4));

FADD(AZ23(1:16),AZ24(1:16);AZ07(1:17));

AZ03(1:33)=-AZ02(1:17)&B(1:16);

AZ05(1:3)=-D|1|AZ04(1:4);

AZ06(1:4)=-AZ05(1:3)&1;

AZ13(1:32)=-D|-1|CA(1:33);

AZ14(1:33)=-0&&AZ13(1:32);

RCONTR(1:4)=-AZ22(1:4);

C(1:32)=-AZ25(1:32);

AZ15(1:1)=-OUS04(CONTR(1),CONTR(2),CONTR(3),CONTR(4);*);

AZ19(1:1)=-ETS02(AY01(2),CA(33);*);

'POUR' I1=1 'A' 4

'DEBUT'

AZ01(I1)=-&NEG|I1|(1;*);

AZ04(I1)=-ETS02|1+I1|(CONTR(I1),RCONTR(I1);*);

AZ16(I1)=-ETS02|5+I1|(&Y01(1),AZ01(I1);*);

AZ17(I1)=-ETS02|9+I1|(&Y01(2),AZ12(I1);*);

AZ22(I1)=-ETS02|13+I1|(&Y01(2),AZ06(I1);*);

'FIN';

'POUR' I1=1 'A' 16

- 90 -

'DEBUT'

&Z23(I1):=&ETS02|17+I1|(&Y01(2),A(I1);*);

&Z24(I1):=&ETS02|33+I1|(&Y01(2),CA(1+I1);*);

'FIN';

'POUR' I1=1 'A' 17

'DEBUT'

&Z02(I1):=&NEG|4+I1|(1;*);

&Z20(I1):=&ETS02|49+I1|(&Z19(1),&Z07(I1);*);

'FIN';

'POUR' I1=1 'A' 33

'DEBUT'

&Z18(I1):=&ETS02|66+I1|(&Y01(1),&Z03(I1);*);

&Z21(I1):=&ETS02|99+I1|(&Y01(3),&Z14(I1);*);

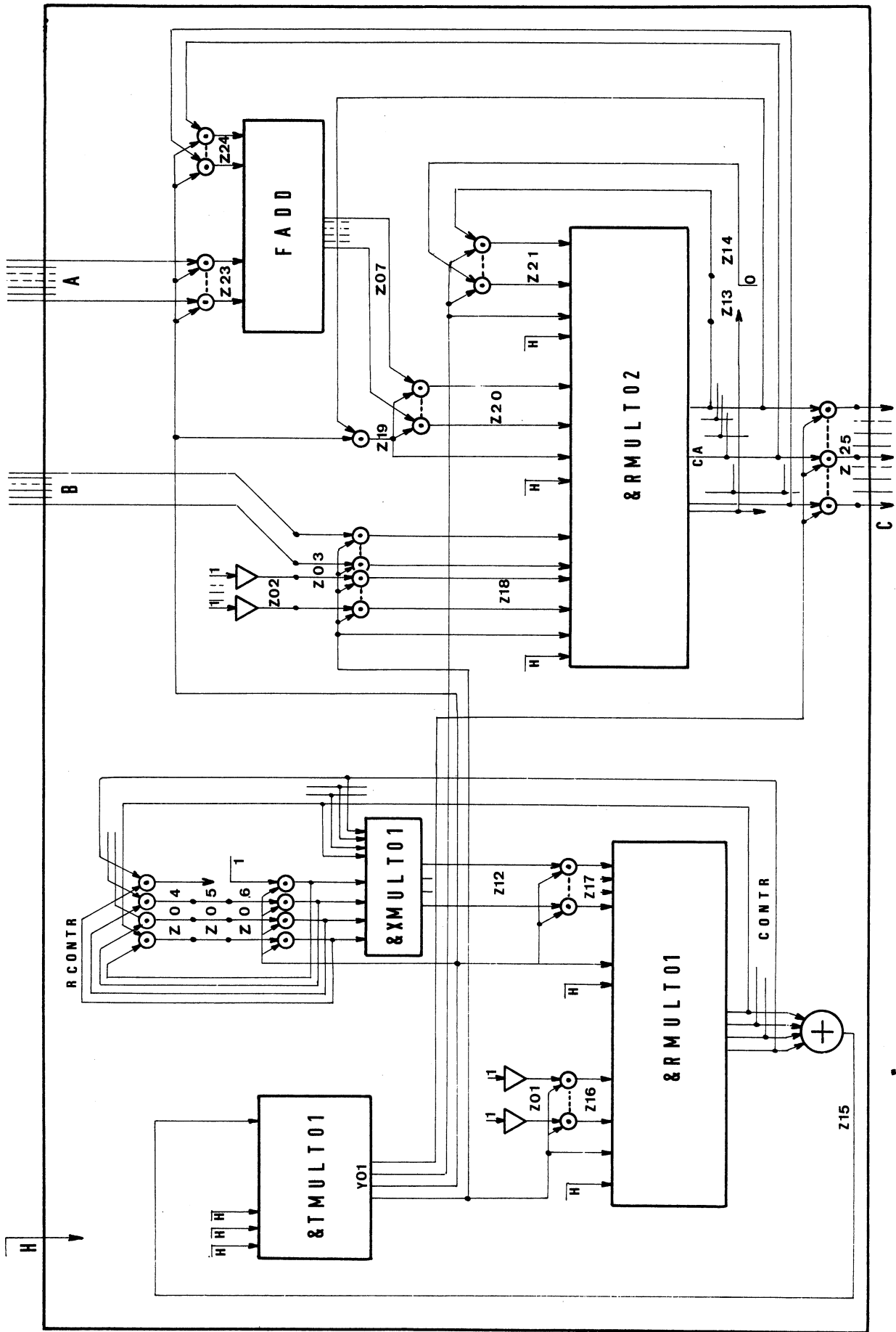
'FIN';

'POUR' I1=1 'A' 32

'DEBUT'

&Z25(I1):=&ETS02|132+I1|(&Y01(4),CA(1+I1);*);

'FIN';



CONCLUSION

Nous avons essayé de montrer dans ce qui précède qu'il était possible de déduire par analyse syntaxique d'un système logique décrit en CASSANDRE, un découpage de celui-ci.

De nombreux problèmes restent à résoudre pour que ce processus fournisse une bonne réalisation, le principal étant de reconnaître si deux expressions sont identiques, ce qui permettrait de réduire le matériel utilisé, donc le coût.

Des améliorations au système apparaissent possibles dans ce but. De plus, il semble indispensable de pouvoir visualiser les résultats, que ce soit au moyen d'une table traçante ou d'une console de visualisation, afin de donner au concepteur un outil plus puissant.

En donnant au concepteur dans ce délai très bref une première partition et une première réalisation du système logique qu'il a décrit, on peut penser ouvrir la voie à des méthodes plus théoriques de synthèse et de composition.

La visualisation qui peut en effet conclure le processus décrit devrait suggérer des réalisations modulaires. Quant à la synthèse en un certain nombre de cellules identiques, problème qui peut s'apparenter à la reconnaissance de formes, le processus décrit fournit la possibilité d'un traitement conversationnel dans lequel le concepteur trouve lui-même progressivement une réalisation en cellules en attendant que la théorie dans ce domaine ait progressé suffisamment pour envisager un degré plus poussé d'automatisation.

La phase qui suit notre travail devrait donc être une phase d'utilisation de l'outil décrit dans cet ouvrage.

Ceci dans l'espoir de faire progresser les problèmes plus théoriques mentionnés ci-dessus.

B I B L I O G R A P H I E

F. ANCEAU, P. LIDDELL, *Système de traitement des descriptions en CASSANDRE - Application à la Simulation, "Colloque sur les Calculateurs embarqués sur fusées et satellites" 3,6 Décembre 1968, CNES Paris.*

L. BOLLIET, *"L'écriture des compilateurs" R.F.T.I. Chiffres vol. 9 n° 1, 1966.*

[1] G. BOULAYE, *"Logique et organes des calculatrices numériques" Dunod.*

M.A. BREUER, *"General survey of design automation of digital computers" Proc of IEEE vol 54, n° 12, december 1966.*

T.D. FRIEDMAN, *"Alert : A program to produce logic designs from preliminary machine descriptions" RC 1578, march 24, 1966, IBM Research.*

T.D. FRIEDMAN, *"Methods used in an automatic logic design generator (Alert)" RC 2226, october 1, 1968, IBM Research.*

T.D. FRIEDMAN, *"Quality of designs from an automatic logic generator" RC 2068, april 25, 1968, IBM Research.*

M. GRIFFITHS, *"Analyse déterministe et compilateurs" Thèse Université de GRENOBLE*

[2] M. GRIFFITHS, P. PELTIER, *"Grammar transformation as an aid to compiler production" Etude n°FF2-0057, mai 1968, Centre scientifique IBM France.*

L. HELLERMAN, *"A catalog of three variable OR-invert and AND-invert logical circuits" IEEE Trans. on Electronic Computers, vol. EC-12, pp.198-223 June 1963.*

[3] K.E. IVERSON, *"A programming language " J. Wiley and Sons. Inc. N.Y. LONDON 1962*

J. KUNTZMANN, *"Algèbre de Boole " Dunod.*

F. LUSTMAN, "Langages exprimant le fonctionnement d'un système" Rapport DGRST, Octobre 1967.

[4] J. MERMET, "Le langage CASSANDRE" Rapport DGRST, Contrat 66 00 069, 31 Mars 1968.

D.L. PARNAS and J.A. DARRINGER, "SODAS and methodology for system design" Fall Joint Computer Conference, 1967, pp. 449-474.

[5] M. PELTIER, "The macro-system" Etude n°FF2-0076, avril 1969: Centre Scientifique IBM France.

J.P. ROTH, "Systematic design of automata Proc. FJCC, 1965, pp. 1093-1099.

J.P. ROTH, "Hardware implementation of microprograms" RC. 1924, August 11, 1965, IBM Research.

H. SCHORR, "Computer-aided digital system design and analyses using a register transfer language" IEEE Trans. on Electronic Computers, Vol. EC-13, pp. 730-737, December 1964.

J. VINCENT-CARREFOUR, "Utilisation d'un calculateur pour la définition des ensembles logiques" L'Onde Electrique, n° 502, janvier 1969.

F.W. ZURCHER, B. RANDELL, "Iterative multi-level modelling - A methodology for computer system design" RC 1938, November 10, 1967, IBM Research

Y. HARRAND, F. ANCEAU, P. LIDDELL, J. MERMET, C. PAYAN, "CASSANDRE : conception assistée des systèmes digitaux" Onde Electrique, n° 502, Janvier 1969.

A N N E X E

CARTE SYNTAXIQUE

NOTE

La syntaxe de "déclarations" d'une unité CASSANDRE ne figure pas sur cette carte - [B4]

Les symboles figurant dans un rectangle sont définis par ailleurs.

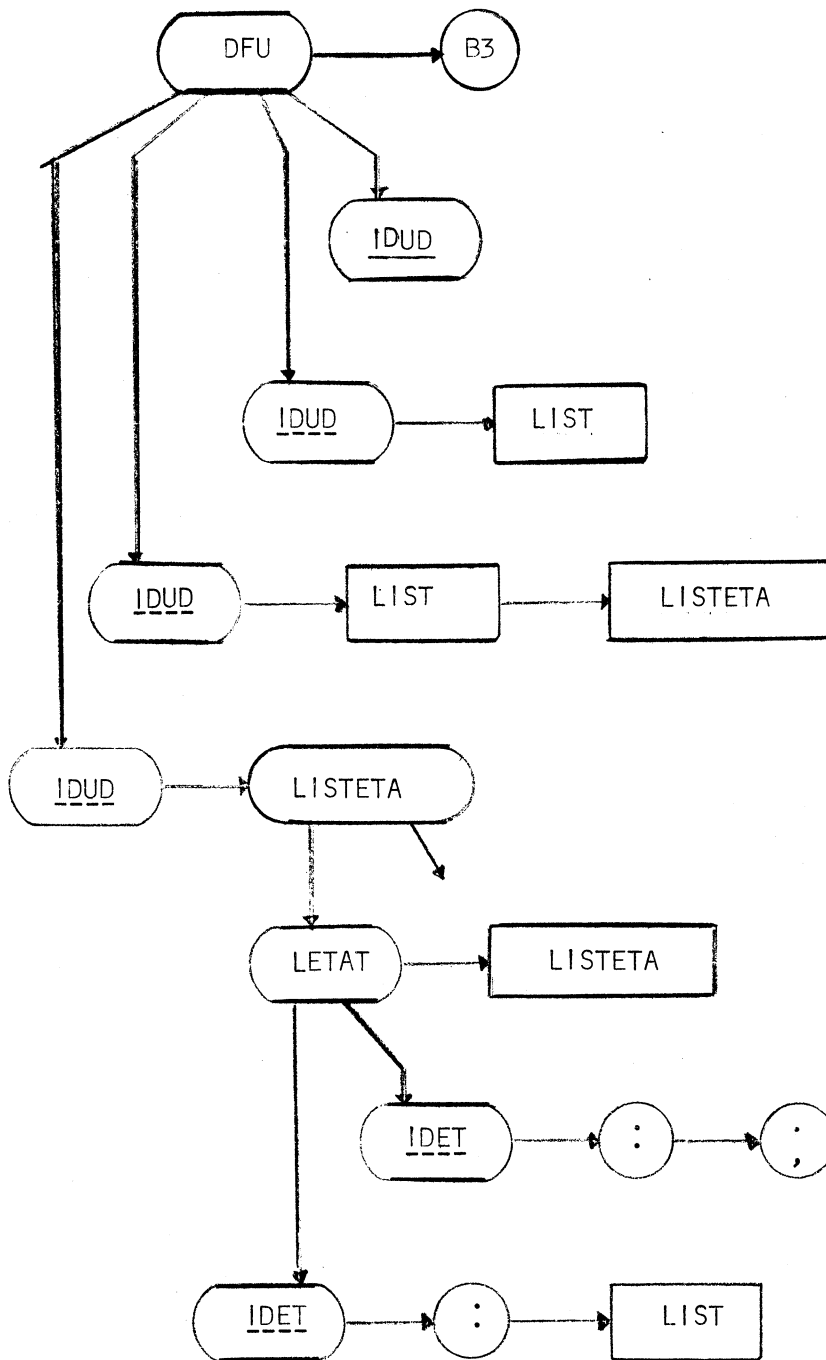
Les symboles soulignés sont des identificateurs déclarés, considérés comme terminaux pour la grammaire

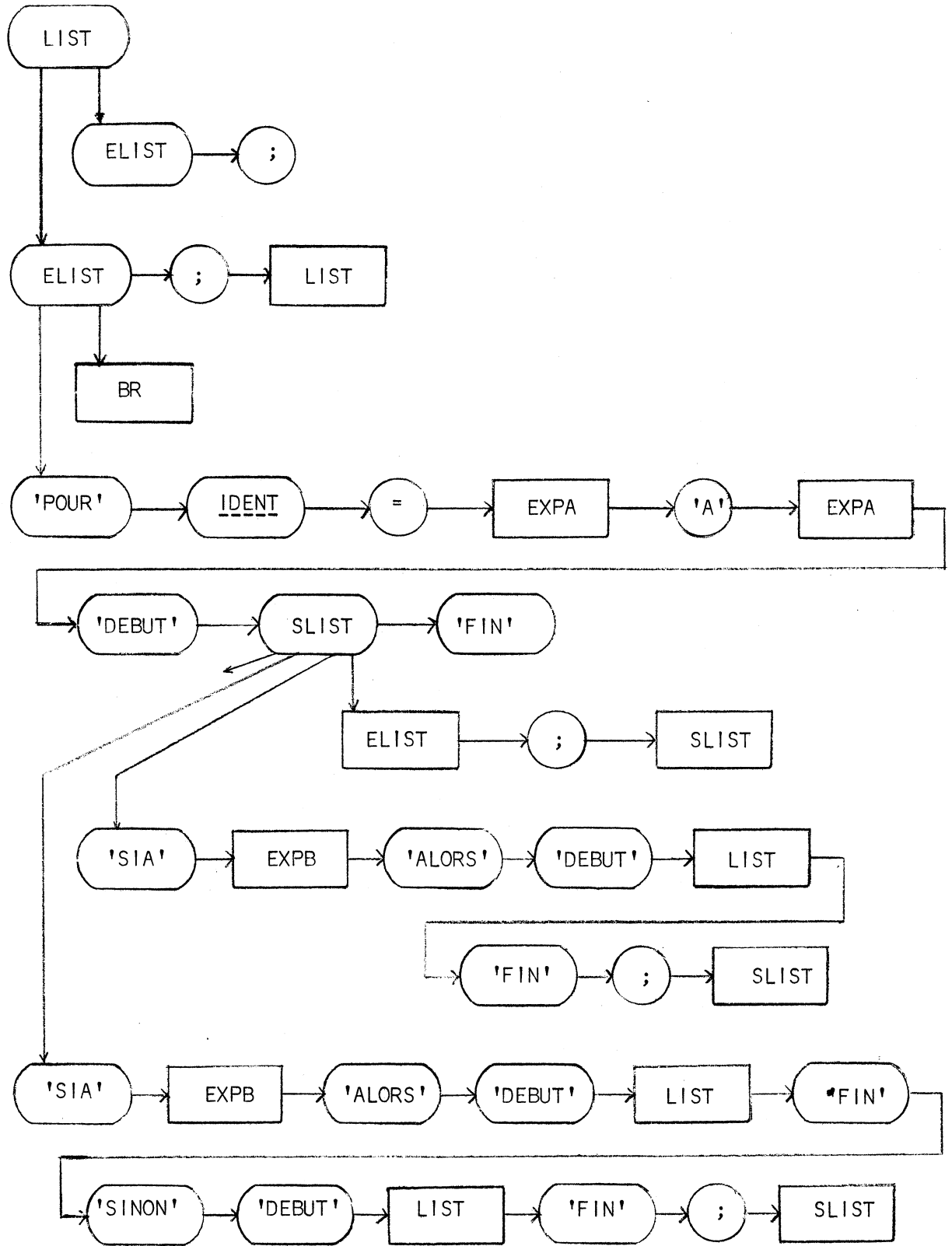
Les symboles figurant entre guillemets sont des symboles de base. Le symbole B3 |f A1 | marque la fin d'une unité, et permet à l'analyseur syntaxique de s'arrêter.

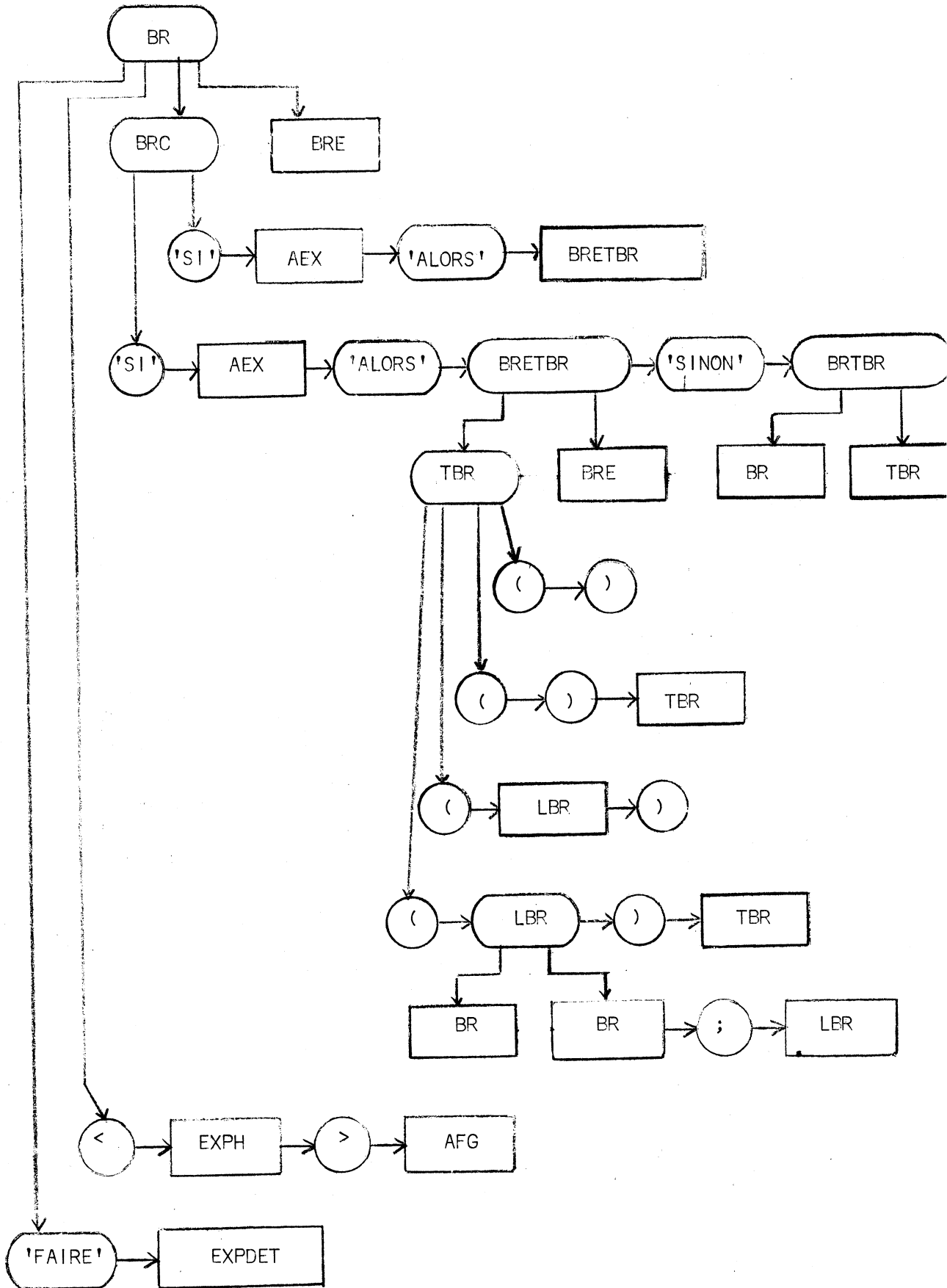
La table des métavariabes |pA8| donne l'endroit où elles sont définies et leur signification, s'il y en a une.

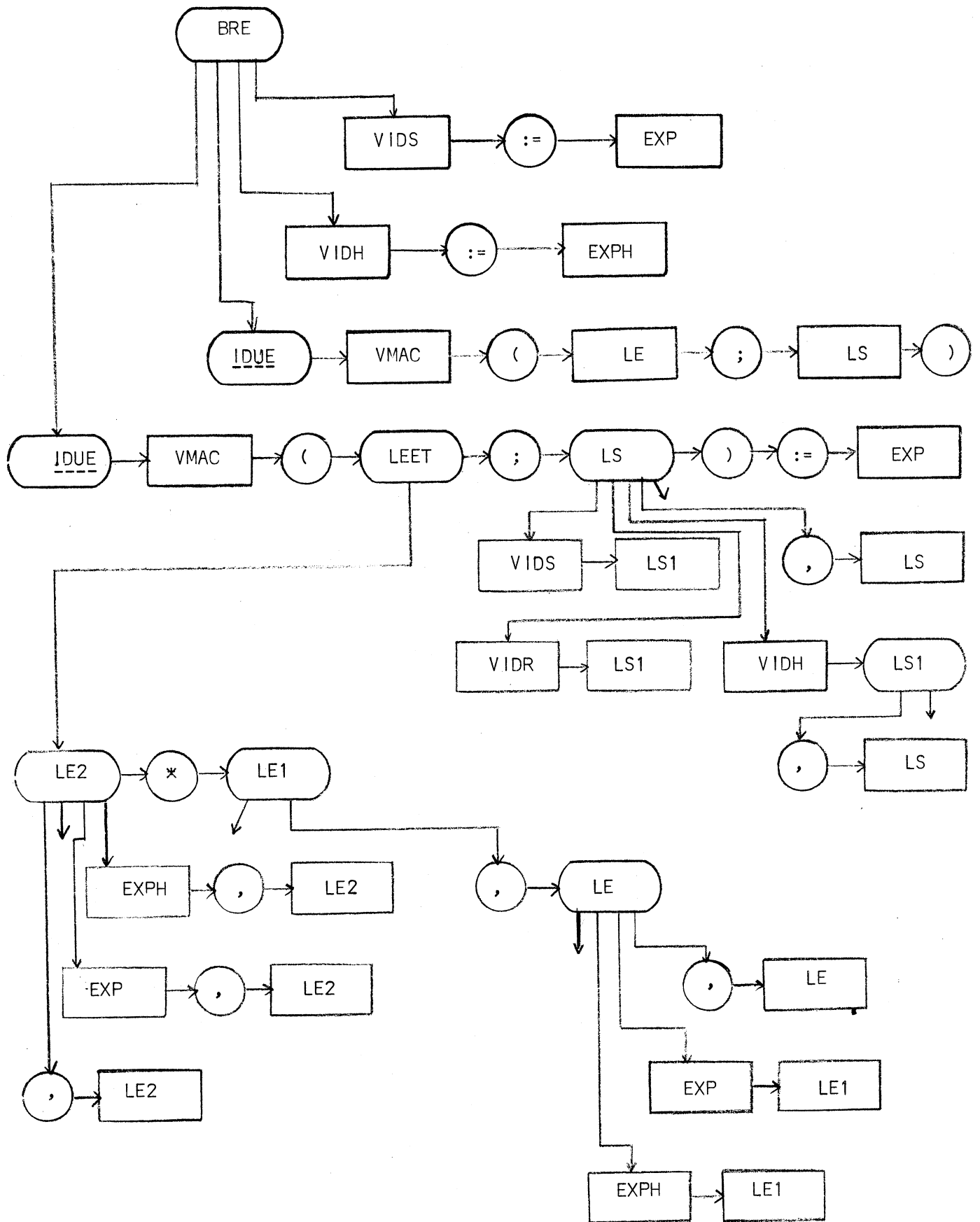
Identificateurs déclarés

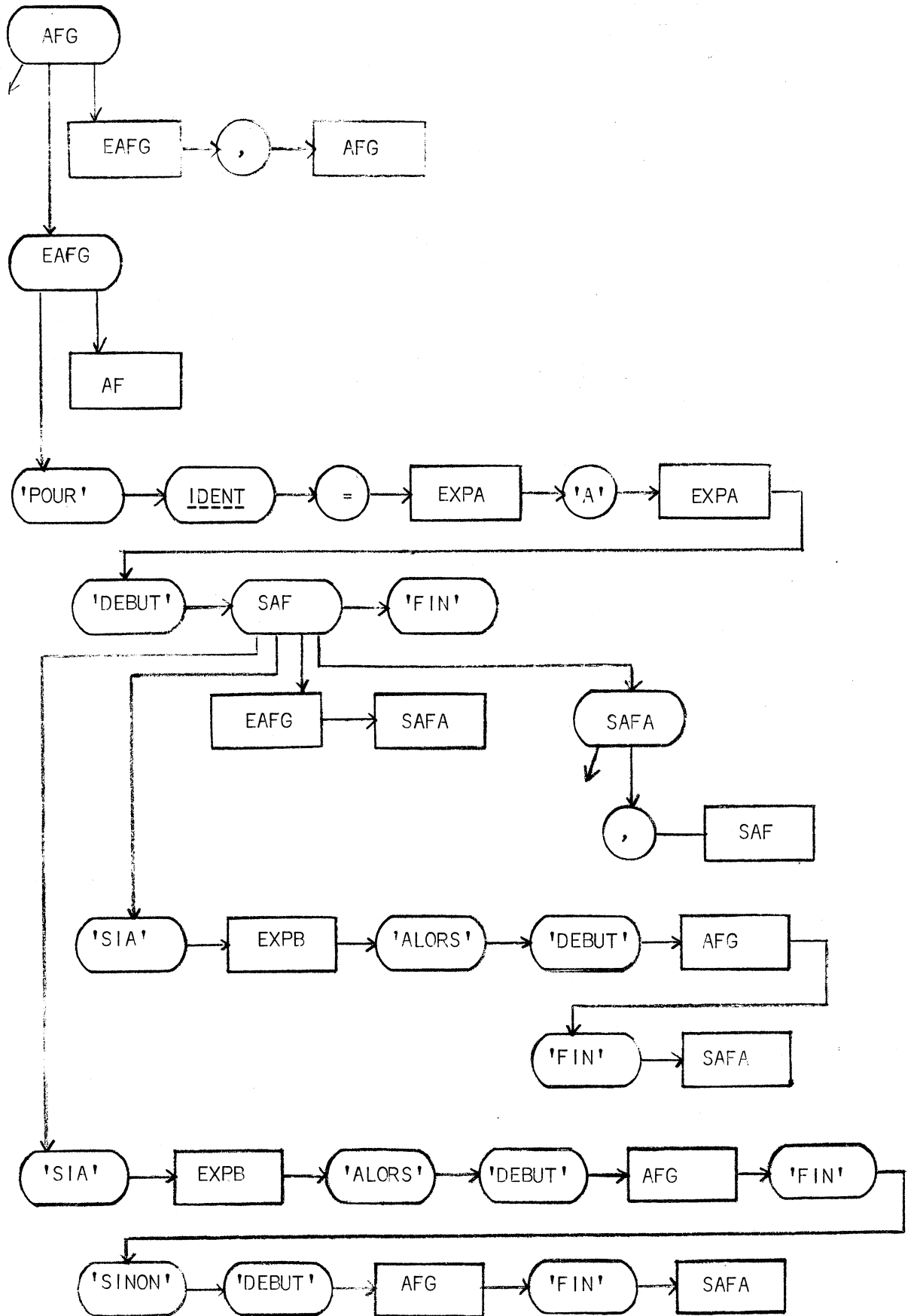
IDUD	: Unité
IDENT	: entier (déclaré implicitement dans une boucle 'POUR')
IDUE	: unité externe
IDS	: signal
IDR	: registre
IDH	: horloge
IDHM	: horloge mère
IDET	: état (état apparaissant en étiquette)
IDETM	: état multiple (états facteurs)
IDETX	: état d'une unité externe
IDRET	: registre d'état (désigné sous le vocable 'ETAT' dans les déclarations).

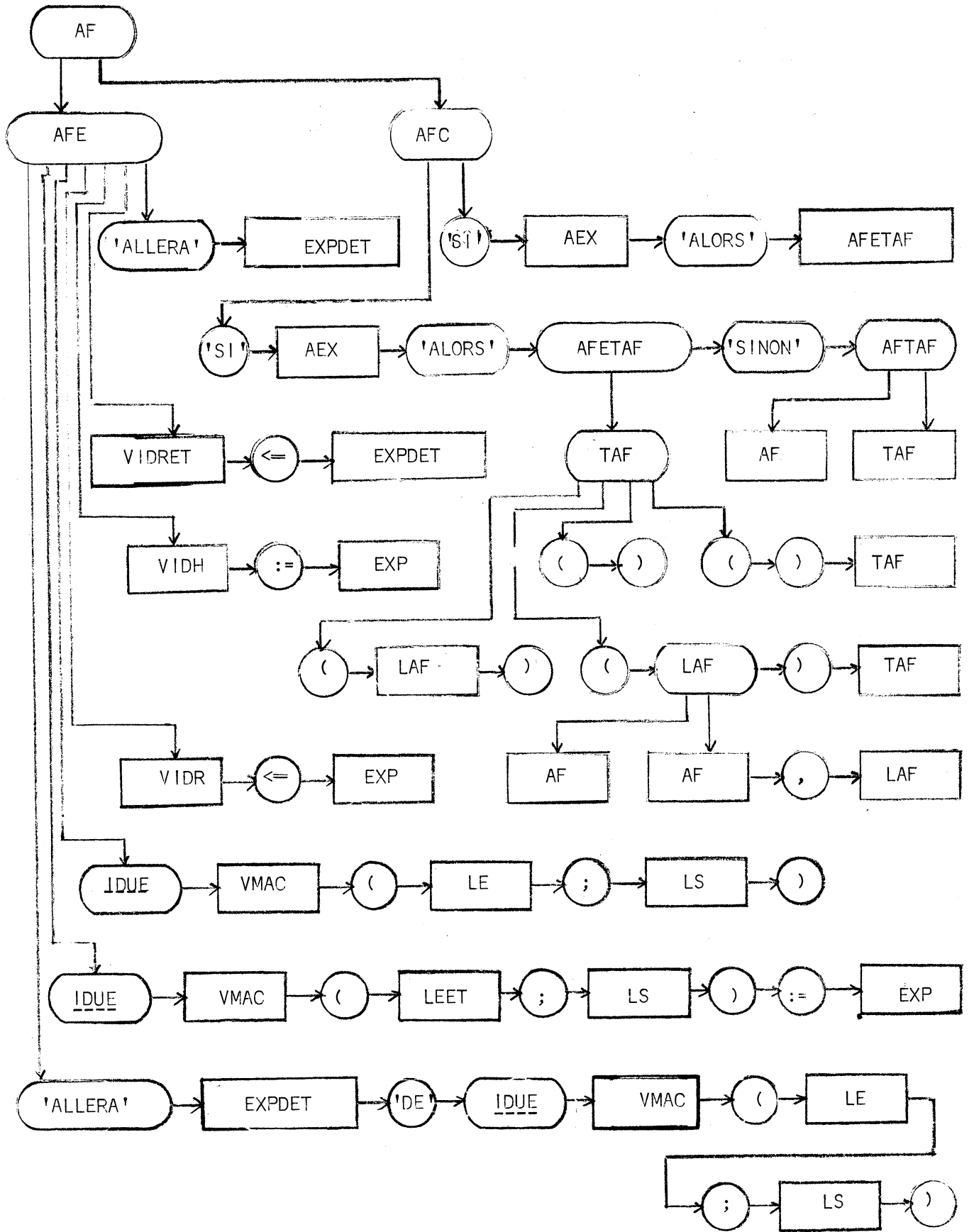


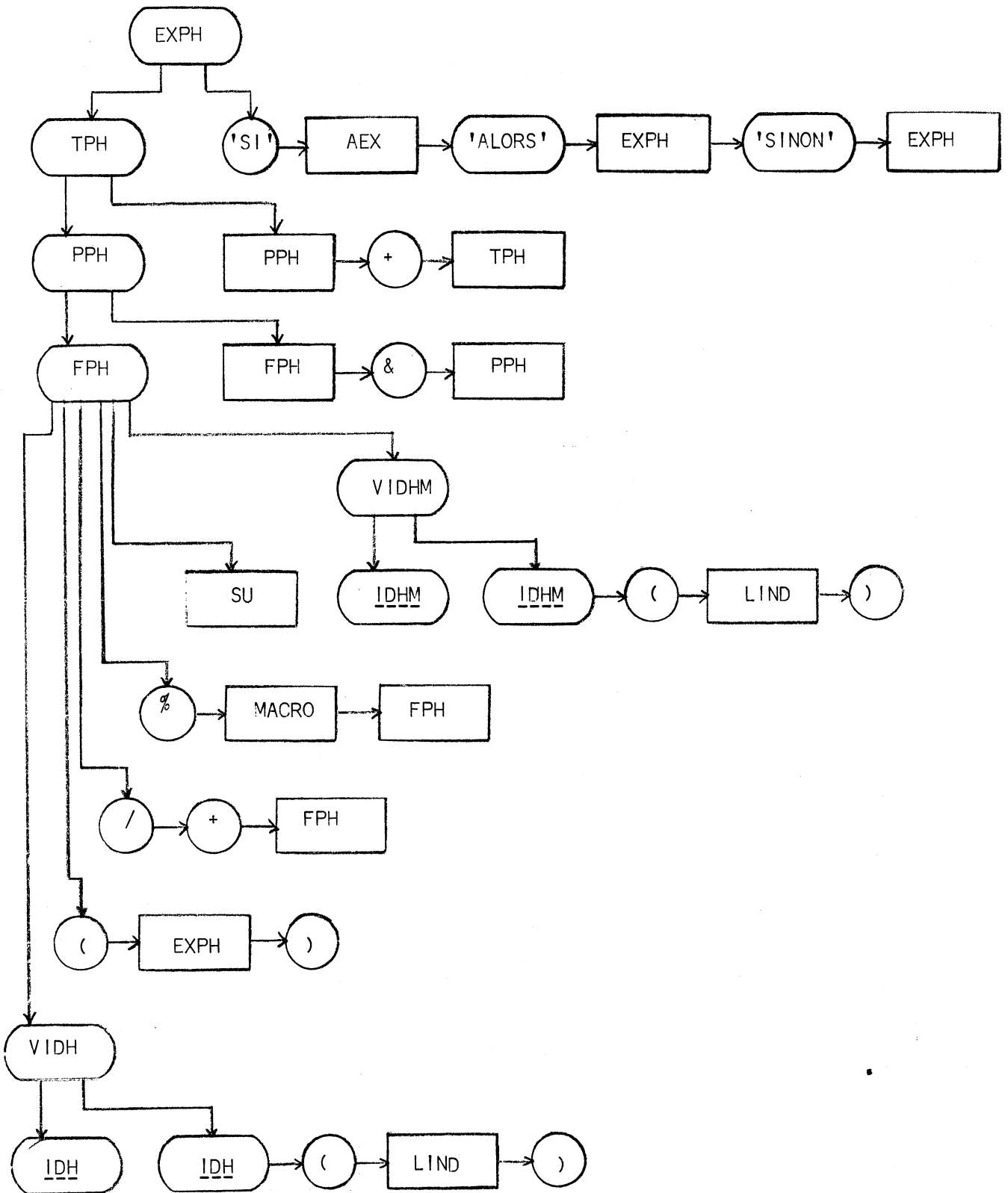


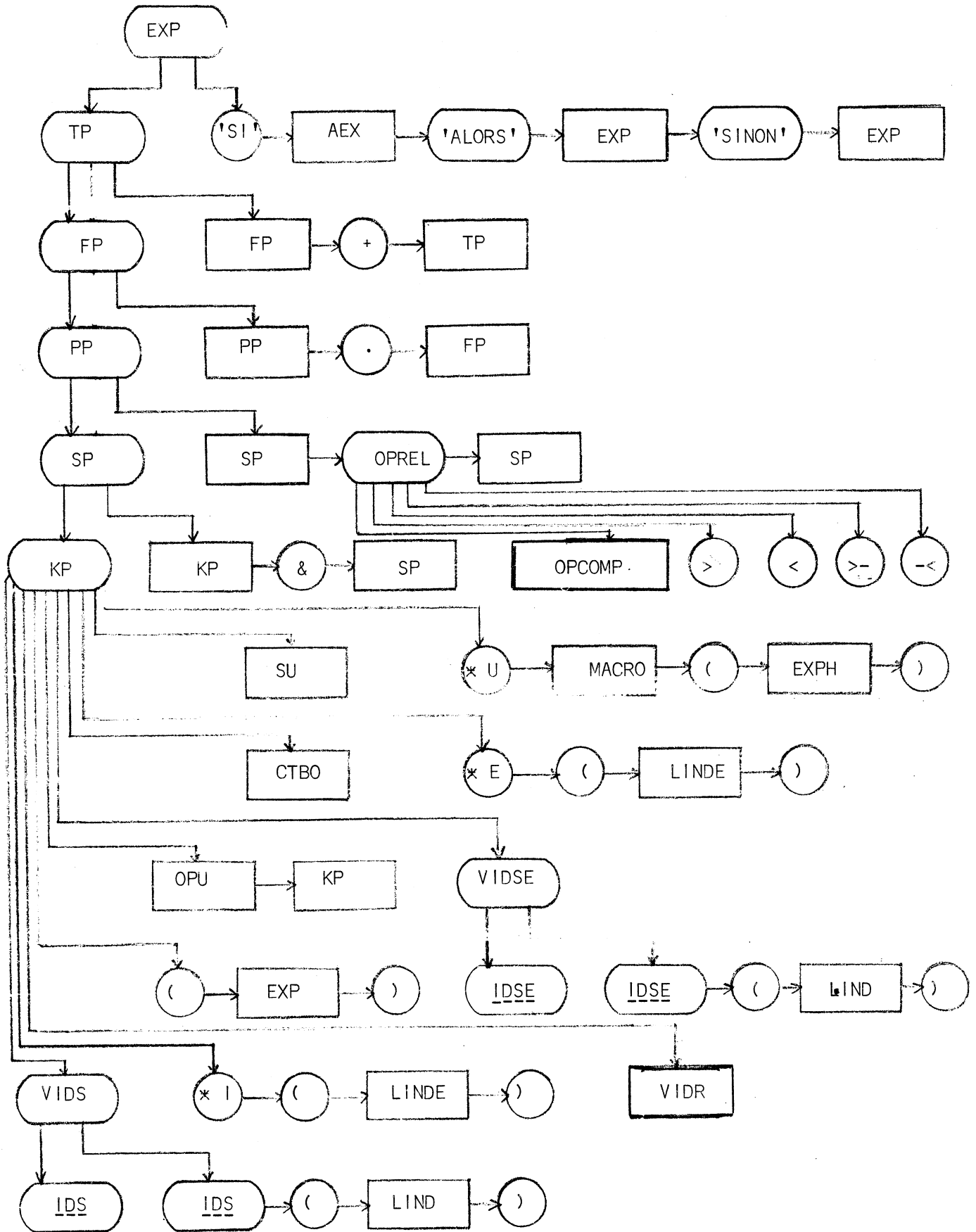


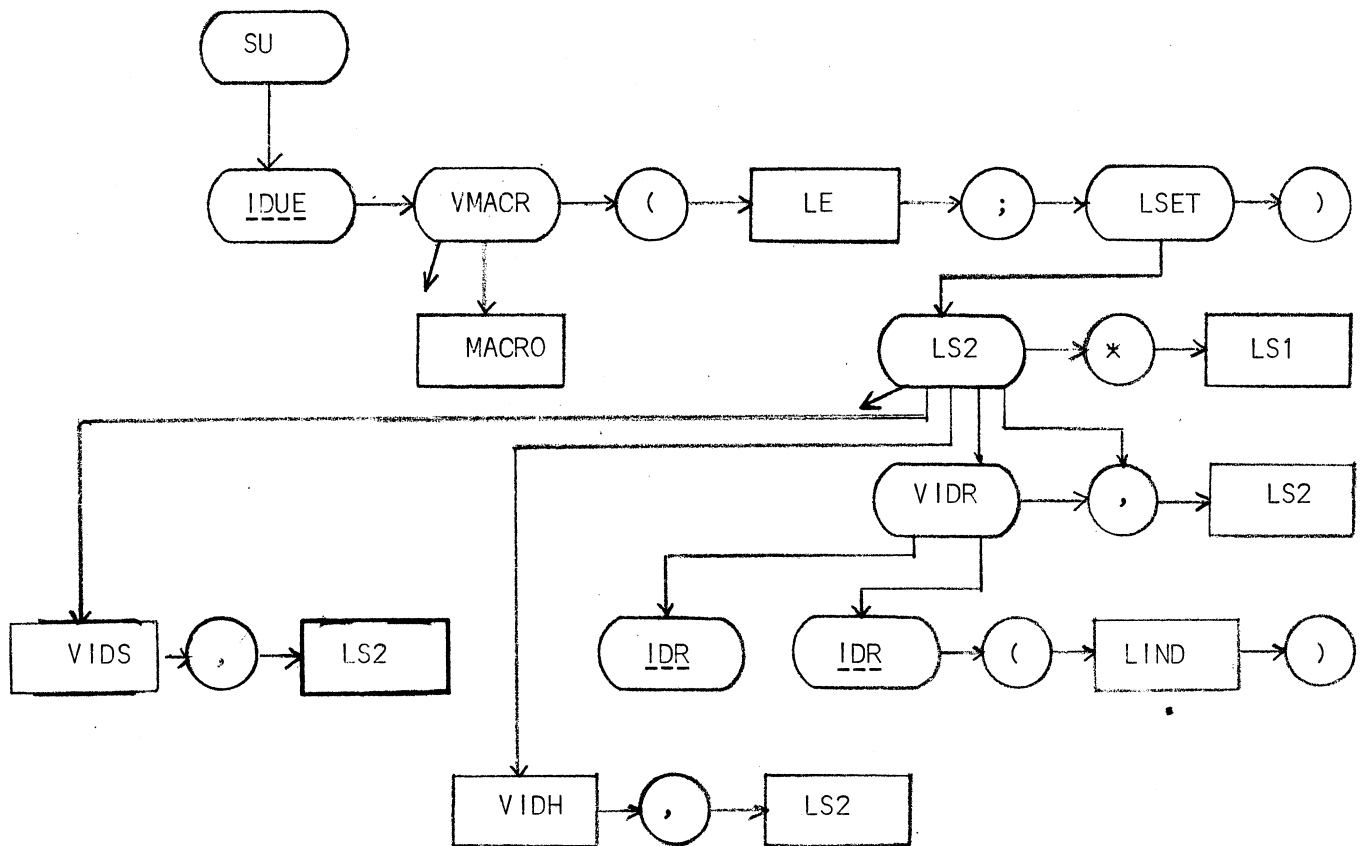
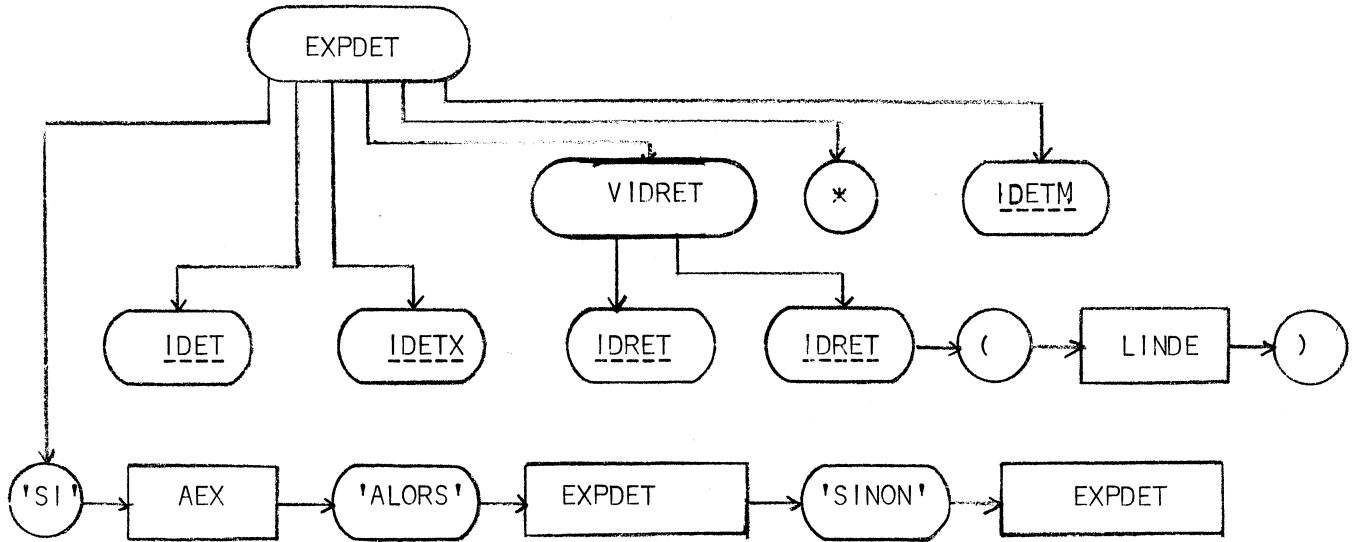


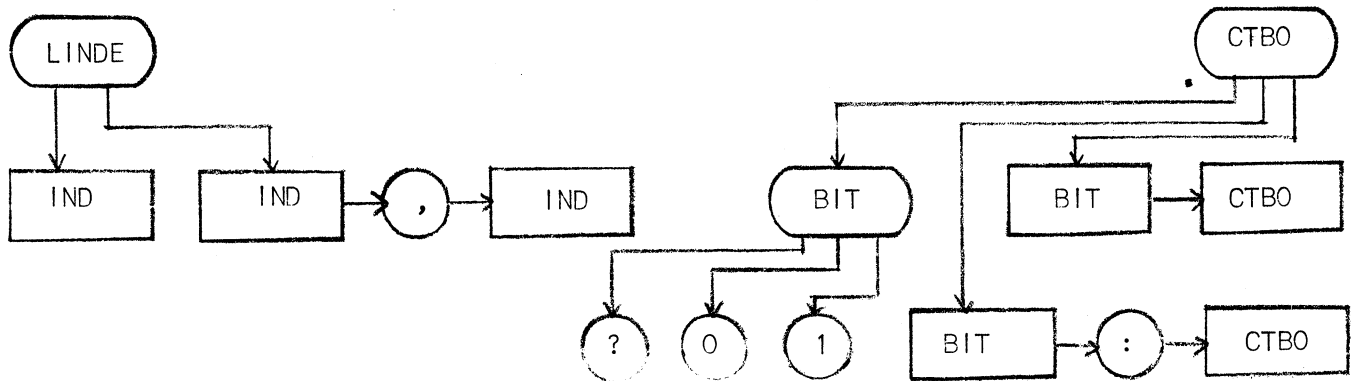
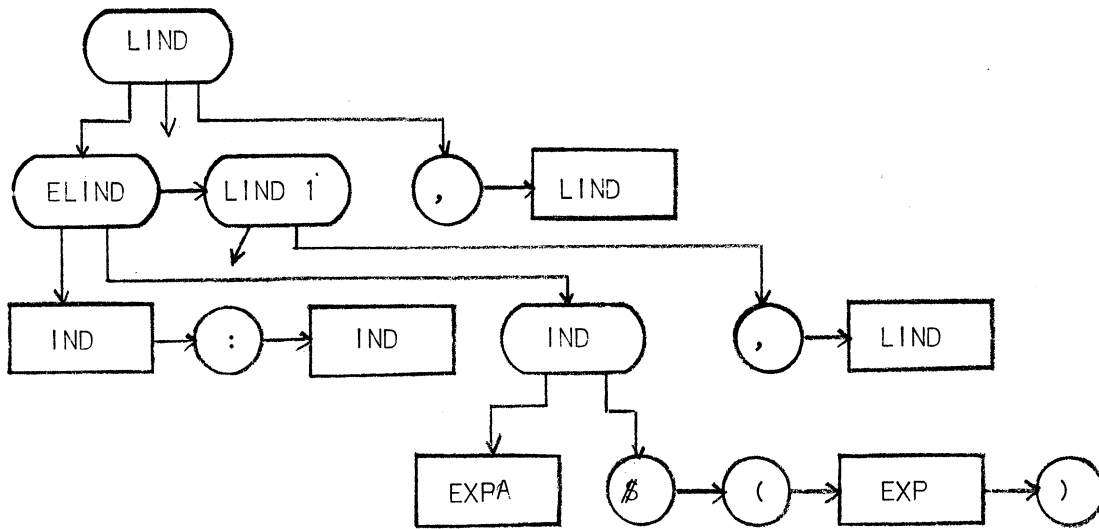
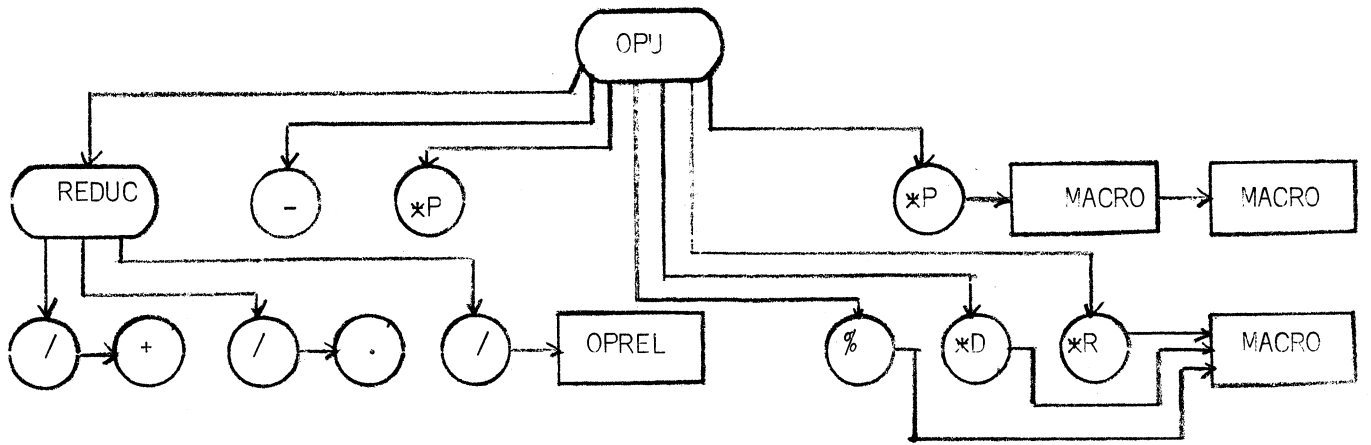


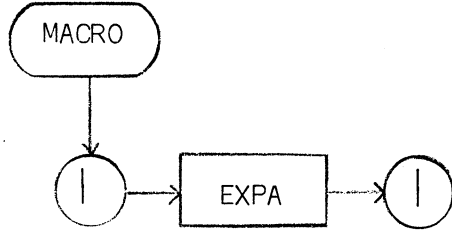
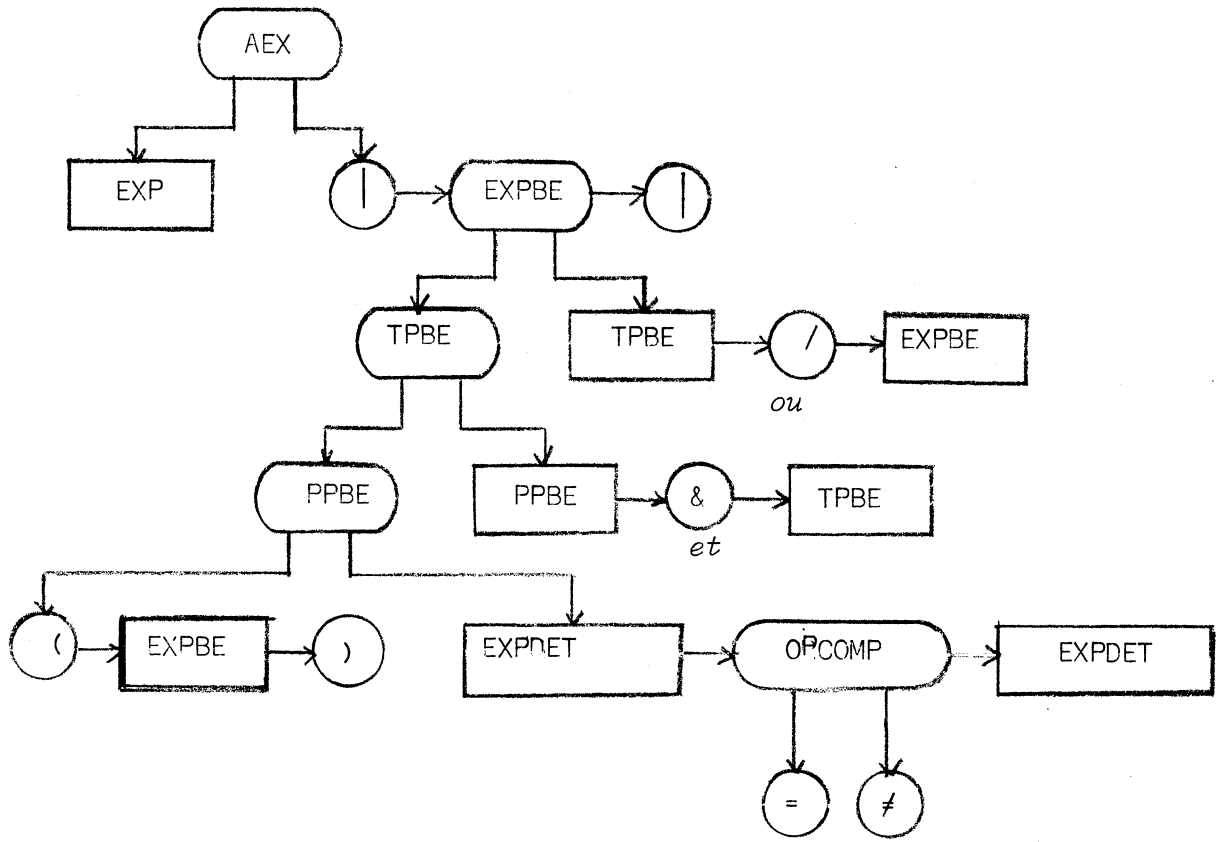


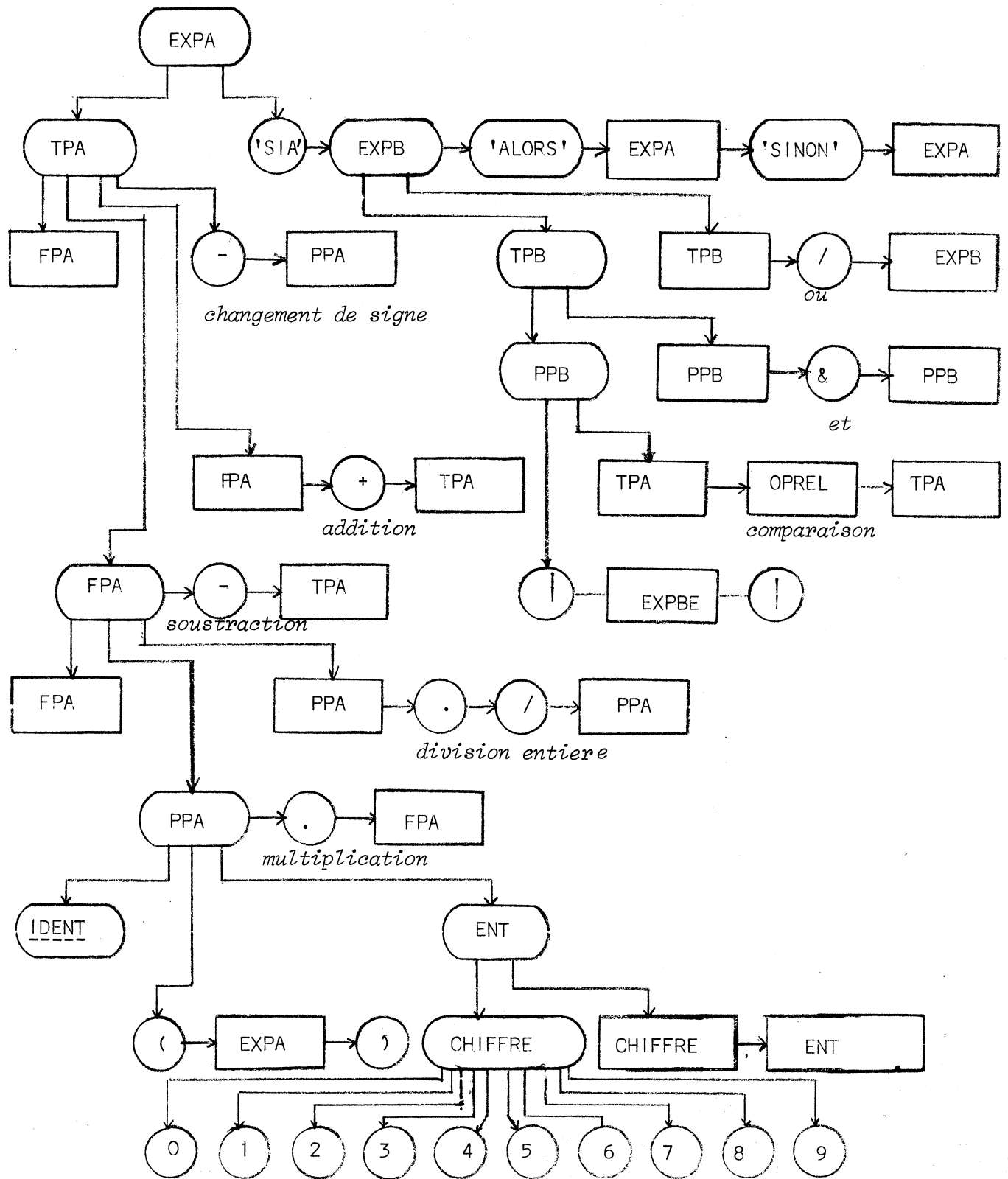












METAVARIABLE	PAGE	SIGNIFICATION
AEX	A11	
AF	A6	Affectation
AFC	A6	Affectation conditionnelle
AFE	A6	Affectation élémentaire
AFETAT	A6	Affectation élémentaire ou tenseur d'affectation
AFG	A5	Affectation généralisée
BIT	A10	
BR	A3	Branchement
BRC	A3	Branchement conditionnel
BRE	A4	Branchement élémentaire
BRETBR	A3	Branchement élémentaire ou tenseur de branchements
BRTBR	A3	Branchement ou tenseur de branchements
CTBO	A10	Constante booléenne
DFU	A1	Définition d'une unité
EAFG	A5	Eléments d'affectation généralisée
ELIND	A10	
ELIST	A2	
ENT	A12	Entier
EXP	A8	Expression
EXPA	A12	Expression arithmétique
EXPB	A12	
EXPBE	A11	
EXPDET	A9	Expression définissant un Etat
EXPH	A7	Expression d'horloge
FP	A8	
FPA	A12	
FPH	A7	
IND	A10	Indice
KP	A8	
LAF	A6	Liste d'affectations
LBR	A3	Liste de branchements
LE	A4	Liste d'entrées
LE2	A4	
LE1	A4	

LEET	A4	Liste d'entrées avec étoile
LIND1	A10	
LETAT	A1	
LIND	A10	Liste d'indices
LINDE	A10	Liste d'indices élémentaires
LIST	A2	Liste d'instructions
LISTETA	A1	Liste d'instructions contrôlées par un état
LS	A4	Liste de sorties
LS1	A4	
LS2	A9	
LSET	A9	Liste de sortie avec étoile
MACRO	A11	
OPCOMP	A11	Opérateurs de comparaison
OPREL	A8	Opérateurs de relation
OPU	A10	Opérateurs unaires
P P	A8	
PPA	A12	
PPB	A12	
PPBE	A11	
PPH	A7	
SAF	A5	
SAFA	A5	
SLIST	A2	
SP	A8	
SU	A9	Signal d'Unité
TAF	A6	Tenseur d'affectations
TBR	A3	Tenseur de branchements
TP	A8	
TPA	A12	
TPB	A12	
VIDRET	A9	
VMAC	A9	
VIDH	A7	
VIDHM	A7	
VIDR	A9	
VIDS	A8	
VIDSE	A8	

P R O G R A M M E S

Dans cette annexe n'apparaissent pas tous les programmes du système, mais seulement les plus caractéristiques.

GRAMMAIRE

P1

```

INPUT SYNTAX=
239 LIU INITIAL DFU ITERMUNIT 'B3'
240 DFU 'IDUD' LIST DSCHAI FETA1 LISTETA BOOLV
      'IDUD' LIST BCOLV
      'IDUD' DSCHAI FETA1 LISTETA BOOLV
      'IDUD' BOOLF
      'IDUD' 'CORPCODE' BOOLF
241 LISTETA LETAT DSCHAI LISTETA
      LETAT
242 LETAT 'IDET' META 'DPT' 'PV' FSCHAI
      'IDET' META 'DPT' FETA2 FSCHAI LIST
      'IDET' META 'DPT' 'DEBUT' LISTETA 'FIN' FSCHAI
243 LIST HORO CNDO ELIST 'PV'
      HORO CNDO ELIST 'PV' LIST
244 POURDEC 'POUR' 'IDENT' EMP 'EG' EXPA ACTPILI EMPCTPI PERMUT CTIDENT 'A' EXPA ACTPILI IPERMUT IDDP DDP I
      EMPV ISPD
245 SIADEC 'SIA' EXPB BAVC 'ALORS'
246 SIDEC 'SI' EMPLXP NL EXP EMPIDS EMPLXP TERMINUNIT 'ALORS' PERMUT BOOLV MAJENT
      'SI' DSCHAI SORTSI 'CC' EXPBE 'CC' 'ALORS' FETA3 FSCHAI PERMUT BOOLV MAJENT
247 ELIST EMPCTPI BOOLF EMPM BR DECAP EMPPOP EMPLCBR TERMINUNIT DCTPIVS
      POURDEC RFI INFEG BAVC 'DEBUT' SLIST 'FIN' IEMP II IADD IDDP SAUVP PERMUT IEMPV ISPD EMPSAUVP B
      ARI CHRFI IDECAP IDECAP DCTPILI DCTPILI
248 SLIST ELIST 'PV' HORO CNDO SLIST
      SIADEC 'DEBUT' LIST 'FIN' CHRFI 'PV' SLIST
      SIADEC 'DEBUT' LIST 'FIN' BAVI PERMUT CHRFI 'SINON' 'DEBUT' LIST 'FIN' CHRFI 'PV' SLIST
      ( )
249 BR EMPLCBR MAJSOR FCGND EMPCTPI BRE DCTPIVS
      EMPLCBR MAJSI BRC
      EMPLCBR MAJSOR EMPCTPI 'INF' EMPLXP NL EXPH EMPIDH EMPLXP TERMINUNIT STHOR 'SUP' DCTPIVS IEMPSA
UV ACTPILI AFC EMPLCBR MAJSOR DSCHAI FETA4 'FAIRE' EXPDEI META2 FSCHAI
250 BRC EMPO EMP1 EMPLCBR SIDEC BRETBR DECAP3
      EMPO EMP1 EMPLCBR SIDEC BRETBR 'SINON' FCI BRTBR DECAP3
251 BRETBR EMPLCBR FBAC EMPLCBR MAJSOR FCGND BRE FDECAP
      TBR
252 BRTBR EMPLCBR FBAC BR FDECAP
      TBR
253 TBR 'PG' 'PD'
      'PG' 'PD' PIP TBR
      'PG' EMPLCBR FBAC LBR 'PD' FDECAP
      'PG' EMPLCBR FBAC LBR 'PD' FDECAP PIP TBR

```


254 LBR BR 'PV' BOOLF BOOLF LBR

255 BRE VIDS 'BRANCH' EMPLXP NL EXP EMPIDS EMPLXP TERMINUNIT BOOLF CNXSIG
VIDH 'BRANCH' EMPLXP NL EXPH EMPIDH EMPLXP TERMINUNIT BOOLF CNXHOR
SU BSUE

256 AFG EAFG 'VIRG' AFG
()
EAFG

257 EAFG FC2 AF FC3 EMPGOP EMPLCAF TERMINUNIT
POURDEC RFI INFEG BAVC 'DEBUT' SAF 'FIN' IEMP I1 IADD IDDP SAUVP PERMUT IEMPV ISPD EMPSAUVP BAR
I CHRFI IDECAP IDECAP DCTPILI DCTPILI

258 SAF SAF
EAFG SAF
SIADEC 'DEBUT' AFG 'FIN' CHRFI SAF
SIADEC 'DEBUT' AFG 'FIN' BAVI PERMUT CHRFI 'SINON' 'DEBUT' AFG 'FIN' CHRFI SAF

259 SAF 'VIRG' SAF
()

260 AF EMPLCAF MAJSOR FCOND EMPCTPI AFE DCTPIVS
EMPLCAF MAJSI AFC

261 AFC EMPO EMP1 EMPLCAF SIDEC AFETAF DECAP3
EMPO EMP1 EMPLCAF SIDEC AFETAF 'SINON' FCI AFCTAF DECAP3

262 AFETAF EMPLCAF FBAC EMPLCAF MAJSOR FCOND AFE FDECAP
TAF

263 AFCTAF EMPLCAF FBAC AFC FDECAP
TAF

264 TAF 'PG' 'PD'
'PG' 'PD' PIP TAF
'PG' EMPLCAF FBAC LAF 'PD' FDECAP
'PG' EMPLCAF FBAC LAF 'PD' FDECAP PIP TAF

265 LAF AF
AF 'VIRG' BOOLF BOOLF LAF

266 AFE DSCHAI FETA5 'ALLERA' EXPDET META2 FSCHAI
VIDH 'BRANCH' EMPLXP NL EXP EMPIDS EMPLXP TERMINUNIT BOOLF CNXHOR
DSCHAI FETA5 VIDRET 'FLG' EXPDET META2 FSCHAI
VIDR 'FLG' EMPLXP NL EXP EMPIDS EMPLXP TERMINUNIT BOOLF CNXHOR
SU BSUE

267 EXPDET 'SI' AEXETA 'ALORS' EXPDET 'SINON' EXPDET
VETA

268 VETA 'RSU'
'IDET'
'IDETM'
'IDETX'
VIDRET

269	AEXETA	'CC' EXPBE 'CC' FSCHAI EMLXP NL EXP EMPIDS EMLXP TERMINUNIT DSCHAI FETA6
270	EXPBE	TPBE 'REDUC' EXPBE TPBE
271	TPBE	PPBE 'CONC' TPBE PPBE
272	PPBE	'PG' EXPBE 'PD' EXPDET OPCOMP EXPDET
273	SU	'IDUE' EMP EMPO PERMUT FISU 'IDUE' EMP MACRO EMPCTPI PERMUT FISU
274	BSUE	PGSU SUITSU
275	PGSU	'PG' FAUXST
276	SUITSU	'IDH' SUITH 'IDS' SUITS
277	SUITH	EMLXP NL EXPH EMPIDH EMLXP TERMINUNIT FSUE FAV SUITE1 SUITE2
278	SUITS	EMLXP NL EXP EMPIDS EMLXP TERMINUNIT FSUE FAV SUITE1 SUITE2 'RSU' FETOIE FAV LE1 FASU 'PV' LS 'PD' BEXP
279	SUITE1	'VIRG' FAUXST SUITSU SUITSUI
280	SUITE2	SUITSUI FAV 'VIRG' FAUXST SUITSU
281	SUITSUI	FASU 'PV' LS 'PD'
282	BEXP	'BRANCH' EMLXP NL EXP EMPIDS EMLXP TERMINUNIT BETOIE
283	LE	'IDS' ETYPS 'IDH' ETYPH
284	ETYPS	EMLXP NL EXP EMPIDS EMLXP TERMINUNIT FSUE FAV LE1 FAV 'VIRG' FAUXST LE FAV
285	ETYPH	EMLXP NL EXPH EMPIDH EMLXP TERMINUNIT FSUE FAV LE1 FAV 'VIRG' FAUXST LE FAV
286	LE1	'VIRG' FAUXST LE ()
287	BSUS	PGSU LE FASU 'PV' LS2 'RSU' BETOIS LS1 'PD'
288	LS2	VIDS EMPIDS FSUS FAV 'VIRG' LS2 VIDR EMPIDR FSUS FAV 'VIRG' LS2 VIDH EMPIDH FSUS FAV 'VIRG' LS2

289 LS FAV 'VIRG' LS2
 ()
 VIDS EMPIDS FSUS FAV LSI
 VIDR EMPIDR FSUS FAV LSI
 VIDH EMPIDH FSUS FAV LSI
 FAV 'VIRG' LS
 FAV

290 LS1 'VIRG' LS
 ()

291 SIEXP 'SI' EMPLXP NL EXP REALSI 'ALORS'
 'SI' DSCHAI SORTSI 'CC' EXPBE 'CC' 'ALORS' FETA3 FSCHAI FXP2

292 EXPH SIEXP EMPLXP NL EXPH BOOLF EMPIDH FXP1 SORSI2 'SINON' EMPLXP NL EXPH BOOLF EMPIDH FXP1 SORSI3 E
 MPICH REALCONEX

293 TPH PPH EMPLXP MAJSI EMPO PERMUT 'PLUS' EMPM TPH1 EMPLXP MAJSI EMPO PERMUT FXPHOU
 PPH PERMUT DECAP

294 TPH1 PPH EMPLXP MAJSI 'PLUS' EMPM TPH1
 PPH

295 PPH FPH EMPO PERMUT 'CONC' EMPO PPH1 FCONC EMPIDH EMPCTPI ELEM2 EMPLXP BOOLF MAJEXT BOOLF
 FPH PERMUT DECAP

296 PPH1 FPH 'CONC' EMPO PPH1
 FPH

297 FPH 'PG' EMPLXP NL EXPH BOOLF EMPIDH FXP1 'PD'
 'REDUC' EMPIDH EMPC 'PLUS' FPH FREDUC
 EMPIDHM VIDHM FPREP EMPLXP MAJENT BOOLF
 EMPIDH VIDH FPREP EMPLXP MAJENT BOOLF
 'TAU' MACRO ACTPILI EMPCTPI FPH FRETARD
 'DERIV' 'PG' EMPLXP NL EXP BOOLF EMPIDS FXP1 'PD' FDERIV
 SU BSUS

298 EXP SIEXP EMPLXP NL EXP BOOLF EMPIDS FXP1 SORSI2 'SINON' EMPLXP NL EXP BOOLF EMPIDS FXP1 SORSI3 EMP
 IDS REALCONEX

299 TP FP EMPLXP MAJSI EMPO PERMUT 'PLUS' EMPM TPI EMPLXP MAJSI EMPO PERMUT RECUPES CREEP CONSQP EMPID
 S BOOLF FOU ELEM1 ELEM2 CONSIP EMPLXP BOOLV MAJEXT
 FP PERMUT DECAP

300 TPI FP EMPLXP MAJSI EMPO PERMUT 'PLUS' EMPM TPI
 FP

301 FP 'PP' EMPLXP MAJSI EMPO PERMUT 'POINT' FPI EMPLXP MAJSI EMPO PERMUT RECUPES CREEP CONSQP EMPIDS FE
 T ELEM1 ELEM2 CONSIP EMPLXP BOOLV MAJEXT
 PP PERMUT DECAP

302 FPI PP EMPLXP MAJSI EMPO PERMUT 'POINT' FPI
 PP

303 PP EMPM SP EMPLXP MAJSI EMPC OPREL EMPM SP EMPLXP MAJSI REALOPREL BOOLV

```

EMPM SP
304 SP KP EMPO PERMUT 'CCNC' EMPO SPI FCNC EMPIDS EMPCTPI ELEM2 EMPLXP BOOLF MAJEXT BOOLF
KP PERMUT DECAP
305 SPI KP 'CCNC' EMPO SPI
KP
306 KP EMPIDSE VIDSE FPREP EMPLXP MAJENT BOOLF
EMPIDS VIDS FPREP EMPLXP MAJENT BOOLF
EMPIDS VIDR FPREP EMPLXP MAJENT BOOLF
EMPC 'EPS' LDI
EMPC 'INDE' LDI
EMPCOP FCTI EMPO EMPO CTBO FCT3 PLUS1 ICHAIDIM IEMP ISPD ACTPILI EMPCTPI PERMUT FPREP BOOLF
'PG' EMPLXP NL EXP BOOLF EMPIDS FXPI 'PD'
'MOINS' CREEP EMPM EMPO KP RECUPES CONSPQ FMOIN
'REDUC' EMPIDS EMPC 'PLUS' KP FREDUC
'REDUC' EMPIDS EMPC 'POINT' KP FREDUC
'REDUC' EMPC OPREL KP FREDOPR
EMPC 'TILDA' IEMP I1 IEMP I2 FKP2 EMPM KP FKPI ITRANSP FKP4
EMPC 'TILDA' MACRO MACRO FKP2 EMPM KP FKPI ITRANSP FKP4
EMPC 'DLG' MACRO FKP3 EMPM KP FKPI IDL FKP4
EMPC 'ROTG' MACRO FKP3 EMPM KP FKPI IROT FKP4
'UNCROUP' MACRO ACTPILI EMPCTPI 'PG' EMPLXP NL EXPH BOOLF EMPIDH FXPI 'PD' FUNCOUP
SU BSUS
307 LDI EMPO 'PG' LDIE 'PD' ICHAIDIM IEMP ISPD ACTPILI EMPCTPI CREEP FPREP BOOLF
308 LDIE IEMP I1 EXPA PLUS1 'VIRG' LDIE
IEMP I1 EXPA PLUS1
309 VIDSE 'IDSE' LINDS
310 VIDS 'IDS' LINDS
311 VIDR 'IDR' LINDS
312 VIDH 'IDH' LINDS
313 VIDHM 'IDHM' LINDS
314 VIDRET 'IDRET' LINDES
315 LINDES 'PG' LINDE 'PD'
( )
316 LINDE EXPA 'VIRG' LINDE
317 LINDS EMP PREPDIM IEMP ISPD ACTPILI EMPCTPI PERMUT
EMP PREPDIM EMPO PERMUT 'PG' LIND 'PD' DECAP IEMP ISPD ICHAIDIM ACTPILI EMPCTPI PERMUT
318 LIND SORTDIM PERMUT PLUS1 PERMUT
'VIRG' SORTDIM PERMUT PLUS1 PERMUT LIND
EXPA IDDP PERMUT PLUS1 PERMUT LINDI
EXPA 'DPT' EXPA PERMUT PLUS1 PERMUT LINDI
319 LINDI 'VIRG' PROGDIM LIND

```

```

( )
320 MACRO 'CC' EXPA 'CC'
321 EXPA SIADEC EXPA BAVI PERMUT CHRFI 'SINON' EXPA CHRFI
TPA
322 TPA FPA 'PLUS' TPA IADD
FPA 'MOINS' TPA ICHS IADD
'MOINS' PPA ICHS
FPA
323 FPA PPA 'POINT' FPA IMULT
PPA 'POINT' 'REDUC' PPA IDIV
PPA
324 PPA 'ENT' EMP IEMP ISPD
'IDENT' EMP LTIDENT IEMPV ISPD
'PG' EXPA 'PD'
325 EXPB TPB 'REDUC' EXPB IOU
TPB
326 TPB PPB 'CONC' TPB IET
PPB
327 PPB TPA EMPC OPREL TPA FBI
'CC' EXPB 'CC'
328 CTBO FCT2 BIT PLUS1 CTBO
FCT2 BIT PLUS1 IDDP IEMP ISPD IPERMUT IDIV IEMP II IPERMUT
FCT2 BIT PLUS1 IDDP IEMP ISPD IPERMUT IDIV IEMP II IPERMUT 'DPT' PLUS1 CTBO
329 BIT 'NO'
'NI'
'PHI'
330 OPREL 'SUP'
'INF'
'SUPG'
'INFG'
OPCOMP
331 OPCOMP 'EG'
'DIF'

```

CROSS REFERENCE TABLE * FUNCTIONS

1	BETOIE	282			
2	BETOIS	287			
3	CNXHOR	255	266		
4	CNXSIG	255			
5	CTIDENT	244			
6	ELEM1	299	301		
7	ELEM2	295	299	301	304
8	FAUXST	275	279	280	284 285 286
9	FBAC	251	252	253	262 263 264
10	FC1	250	261		
11	FC2	257			
12	FC3	257			
13	FCOnc	295	304		
14	FOOND	249	251	260	262
15	FCT1	306			
16	FCT2	328			
17	FCT3	306			
18	FCECAP	251	252	253	262 263 264
19	FDERIV	297			
20	FET	301			
21	FETA1	240			
22	FETA2	242			
23	FETA3	246	291		
24	FETA4	249			
25	FETA5	266			
26	FETA6	269			
27	FETOIE	278			
28	FISU	273			

FONCTIONS

P9

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	
				434	ROUTINE &MAJSOR	
				437	PULL TEMI	
				441	IF B,BOOL,EQ,0	
				444	EXIT	
				446	ENDIF	
				448	CAR TEMI,WORK1	
				452	CCDR 0(WORK1),WORK1	
				456	CAR 0(WORK1),WORK2	
				460	WHILE H,0(WORK2),NE,ANIFT	
				465	IF H,0(WORK2),EQ,0(CONSICNT)	
				469	CAR 0(CONSICNT),WORK3	
				473	PUSH 255	
				482	MVI 0(CONSICNT),X'FF'	
		00000		483	PERMUTER	
				490	PUSH 0	
				499	PERMUTER	
				506	CALL ®UPES	
				509	CREERLIS	
				517	DECAP 1	
				520	CONS 0(CONSICNT),2(CONSICNT)	
				536	MVC 0(2,CONSICNT),2(CONSICNT)	
				537	PUSH 63	
				546	CALL &ELEM1	
				549	IF B,1(WORK3),EQ,98	
				552	B ET1MAJSO	BRANCH
			00168	553	ENDIF	
				555	IF B,1(WORK3),EQ,97	
				558	ET1MAJSO BCTR CONSICNT,0	
				559	BCTR CONSICNT,0	
				560	MVI 1(CONSICNT),97	
			00001	561	LA CONSICNT,2(CONSICNT)	
			00002	562	ENDIF	
				564	CALL &ELEM2	
				567	MVI BOOL,0	
				568	PUSH TEMI	
				576	MVI BOOL,0	
				577	CALL &MAJEXT	
				580	CAR TEMI,WORK1	
				584	CCDR 0(WORK1),WORK1	
				588	B ET2MAJSO	
			001E2	589	ENDIF	
				591	CCDR 0(WORK2),WORK2	
				595	CCDR 0(WORK2),WORK2	
				599	CCDR 0(WORK2),WORK2	
				603	ENDWHILE	
				606	ET2MAJSO CCDR 0(WORK1),WORK1	
				610	CAR 0(WORK1),WORK1	
				614	ACCROCHE 0(WORK1),0(CONSICNT)	
				626	MVI BOOL,0	
				627	RETURN	
0007A	92FF 4000	00000				TYPE ENTRE MARQUE
00012E	D201 4000 4002 00000 00002					
00015C	47F0 C168		00168			
000168	0640					
00016A	0640					
00016C	0261 4001		00001			
000170	4144 0002		00002			
000178	92C0 803A		0003A			
000156	92C0 803A		0003A			
000186	47F0 C1E2		001E2			
000220	92C0 803A		0003A			

LCC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	ENTREES LISTE SYMBOLE CAS A=B	NOMBRE D'EXTERNES	CREATION DE L'EXTERNE CHAINE CODEF	ENTREES	SIGNAUX UNITRAIT
00788	5810 8190	00190		1104	ROUTINE &TERMINU					
00788	5010 818C	0018C		1107	L WORK1,DTSI					
				1108	ST WORK1,PYSI					
				1109	IF H,0(CONSICNT),EQ,ANIFT					
				1113	DECAP 2					
				1118	EXIT					
				1120	ENDIF					
				1122	CAR 0(CONSICNT),WORK1					
				1126	CCDR 0(WORK1),WORK1					
				1130	CAR 0(WORK1),WORK1					
				1134	PULL SAUVP					
				1138	PULL TEM3					
				1142	IF H,0(WORK1),EQ,0(CONSICNT)					
				1146	PUSH SAUVP					
				1154	CALL &RECUP2					
				1157	EXIT					
				1159	ENDIF					
0080E	4111 0002		00002	1161	LA WORK1,2(WORK1)					
				1162	PUSH SAUVP					
0082C	92FF 803A		0003A	1170	MVI 800L,X'FF'					
				1171	CALL &MAJSOR					
				1174	PUSH SAUVP					
				1182	CAR 0(CONSICNT),LCADR					
				1186	CAR 0(LCADR),LCADR					
				1190	IF H,0(LCADR),LT,SAVEPAR+8					
00872	92C0 803A		0003A	1194	MVI 800L,X'00'					
				1195	ENDIF ELSE					
				1198	PUSH TEM3					
				1206	CALL &TRM3					
				1209	CAR SAUVP,WORK1					
				1213	CALL &TRM2					
				1216	SORTIE 51,I=1					
				1220	PUSH SAUVP					
				1228	MVI 800L,X'FF'					
008CA	92FF 8C3A		0003A	1229	ENDELSE					
				1231	CAR 0(CONSICNT),WORK1					
				1235	CCDR 0(WORK1),WORK1					
				1239	CAR 0(WORK1),WORK1					
				1243	CAR UNITRAIT,WORK2					
				1247	CCDR 0(WORK2),WORK2					
				1251	CAR 0(WORK2),WORK2					
				1255	ACCROCHE 0(WORK2),0(WORK1)					
				1267	LA WORK1,2(WORK1)					
0093C	4111 0002		00002	1268	CAR 0(WORK1),WORK1					
				1272	LA WORK2,2(WORK2)					
				1273	CCDR 0(WORK2),WORK2					
				1277	CAR 0(WORK2),WORK2					
				1281	ACCROCHE 0(WORK2),0(WORK1)					
				1293	IF 8,800L,EQ,X'FF'					
				1296	DECAP 1					
				1299	LA WORK2,2(WORK2)					
				1300	CCDR 0(WORK2),WORK2					
				1304	CCDR 0(WORK2),WORK2					
0099A	4122 0002		00002	1308	CAR 0(WORK2),WORK2					

LCC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
				1324	DEGAP 1
				1327	PUSH &RUCUP
				1335	CALL &RUCUP1
				1338	ENDIF ELSE
000AC	4111 C002		00002	1341	LA WORK1,2(WORK1)
				1342	CAR 0(WORK1),WORK1
				1346	ACCROCHE 0(WORK2),0(WORK1)
000A44	4111 C002		00002	1358	LA WORK1,2(WORK1)
				1359	WHILE H,0(WORK1),NE,ANI FT
				1364	CAR 0(WORK1),WORK1
000A60	4122 C002		00002	1368	LA WORK2,2(WORK2)
				1369	CAR 0(WORK2),WORK2
				1373	ACCROCHE 0(WORK2),0(WORK1)
000A96	4111 C002		00002	1385	LA WORK1,2(WORK1)
				1386	ENDWHILE
				1389	CALL &RUCUP2
				1392	ENDELSE
				1394	RETURN

P 12

DIMENSION POUR LA SORTIE

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
				798	ROUTINE \$RECUPES
				801	CCDR 0(CONSICNT),WORK2
				805	CAR 0(WORK2),WORK2
007424	D201 8080 2000 C0080	00000		809	MVC TEM2(2),0(WORK2)
00742A	4120 0000			810	LA WORK2,0
				811	PERMUTER
				818	IF H,0(CONSICNT),NE,X'0000'
				822	MVC TEM2(2),L111
000454	D201 8080 8048 00080	0C048		823	ENDIF
				825	PERMUTER
				832	MVC CRE1(2),ANIFT
000474	D201 C168 8052 00574	00052		833	RECUPBIS IF H,0(CONSICNT),EQ,MARQUEF
				838	CONS WORK2,CRE1
				854	DECAP 1
				857	PUSH CRE1
				865	EXIT
				867	ENDIF
				869	CONS 0(CONSICNT),CRE1
				885	DECAP 1
				888	CONS 0(CONSICNT),CRE1
				904	DECAP 1
000568	4122 0001	00001		907	LA WORK2,1(WORK2)
00056C	47F0 C06E	0C47A		908	B RECUPBIS
				909	EXIT
				911	DS H
000574				912	RETURN

12 MAR 70

LCC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	TYPE
				433	ROUTINE &TRM1	00000050
				436	CCDR 0(WORK3),WORK1	00000060
				440	CAR 0(WORK1),WORK1	00000070
				444	CAR 0(WORK1),WORK1	00000080
				448	IF B,BOOL,EQ,X,FF,	00000090
				451	LA WORK1,2(WORK1)	00000100
				452	MVC TEL(2),0(WORK1)	00000110
				453	ST WORK2,TE2	00000120
				454	CCDR 0(WORK1),WORK1	00000130
				458	CAR 0(WORK1),WORK1	00000140
				462	L WORK2,DTRAD	00000150
				463	WHILE F,WORK2,NE,PTRAD	00000160
				467	IF H,0(WORK1),EQ,0(WORK2)	00000170
				471	B ET1TRM1	00000180
				472	ENDIF	00000190
				474	LA WORK2,6(WORK2)	00000200
				475	ENDWHILE	00000210
				478	LA PARMADD,25	00000220
				479	CALL &ERREUR	00000230
				482	ET1TRM1	00000240
				486	SORTIE 2(WORK2),I=1	00000250
				490	SORTIE 4(WORK2),I=1	00000260
				494	L WORK2,TE2	00000270
				495	B ET2TRM1	00000280
				496	ENDIF ELSE	00000290
				499	MVC MEM2(1,TERMAREA),1(WORK1)	00000300
				500	CALL &SORTIEI	00000310
				503	LA WORK1,2(WORK1)	00000320
				504	MVC TEL(2),0(WORK1)	00000330
				505	CCDR 0(WORK1),WORK1	00000340
				509	CAR 0(WORK1),WORK1	00000350
				513	MVC MEM2(1,TERMAREA),0(WORK1)	00000360
				514	CALL &SORTIEI	00000370
				517	MVC MEM2(1,TERMAREA),1(WORK1)	00000380
				518	CALL &SORTIEI	00000390
				521	ENDELSE	00000400
				523	CAR TEL,WORK1	00000410
				527	MVC MEM2(1,TERMAREA),0(WORK1)	00000420
				528	CALL &SORTIEI	00000430
				531	SORTIE 1(WORK1),I=1	00000440
				535	CAR 0(WORK3),WORK3	00000450
				539	MVC MEM2(1,TERMAREA),0(WORK3)	00000460
				540	CALL &SORTIEI	00000470
				543	MVC MEM2(1,TERMAREA),1(WORK3)	00000480
				544	CALL &SORTIEI	00000490
				547	LA WORK3,2(WORK3)	00000500
				548	CCDR 0(WORK3),WORK3	00000510
				552	EXIT	00000520
				554	DS F	00000530
				555	DS H	00000540
				556	RETURN	00000550

SAUVE POINT LIST POUR DIM
SAUVE WORK2

RESTAURE WORK2

SAUVE POINT LISTE POUR DIM

IDENTIFICATEUR

DIM

PARAM. DIM.

P 14

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT		
					ROUTINE &TRM2		00000570
				560	SORTIE 36, I=1		00000580
				563	ST WORK1, SRI		00000590
00014E	5010 CD56	00E9C		567	CCDR 0(CONSICNT), WORK1		00000600
				568	CCDR 0(WORK1), WORK1		00000610
				572	CAR 0(WORK1), WORK1		00000620
				576	SORTIE 0(WORK1), I=1		00000630
				580	SORTIE 1(WORK1), I=1	NOM DE L'EXTERNE	00000640
				584	L WORK1, SRI		00000650
				588	CCDR 0(WORK1), WORK1		00000660
00018A	5810 CD56	00E9C		589	CAR 0(WORK1), WORK2		00000670
				593	PUSH 0		00000680
				597	PUSH 14		00000690
				606	WHILE H, 0(WORK2), NE, ANIFT		00000700
				615	CAR 0(WORK2), WORK2		00000710
				620	IF B, 1(WORK2), EQ, X'67'		00000720
				624	ET1TRM2		00000730
0001FE	47F0 C000	00216		627	B		00000740
				628	ENDIF		00000750
				630	IF B, 1(WORK2), EQ, X'60'	REGISTRE	00000760
00020A	47F0 C000	00216		633	ET1TRM2		00000770
				634	ENDIF		00000780
				636	IF B, 1(WORK2), EQ, X'5F'	IDS	00000790
000216	9267 803E	0003E		639	ET1TRM2	SORTIE IDSE	00000800
00021A	9267 8126	00126		640	MVI MEM2(TERMAREA), X'67'		00000810
00021E	920E 4001	00001		641	PARM, X'67'	&E	00000820
				642	ENDIF ELSE		00000830
000226	9262 803E	0003E		645	MVI MEM2(TERMAREA), X'62'	SORTIE IDHM	00000840
00022A	9262 8126	00126		646	PARM, X'62'		00000850
00022E	9211 4001	00001		647	MVI PARM, X'62'	&H	00000860
				648	ENDELSE		00000870
				650	CALL &SORTIE1		00000880
				653	CALL &TRM4		00000890
				656	ENDWHILE		00000900
				659	DECAP 2	FIN DES ENTREES	00000910
000246	9248 803E	0003E		664	MVI MEM2(TERMAREA), X'4B'		00000920
				665	CALL &SORTIE1		00000930
				668	CCDR 0(WORK1), WORK1	S	00000940
				672	CAR 0(WORK1), WORK2	PLACE SUR LES SORTIES	00000950
				676	PUSH 0		00000960
				685	PUSH 28		00000970
				694	WHILE H, 0(WORK2), NE, ANIFT		00000980
				699	CAR 0(WORK2), WORK2		00000990
000286	D200 803E	2001 0003E	C0001	703	MVC MEM2(1, TERMAREA), 1(WORK2)	TYPE	00001000
				704	CALL &SORTIE1		00001010
0002C0	D200 8126	2001 00126	00001	707	MVC PARM(1), 1(WORK2)		00001020
				708	CALL &TRM4		00001030
				711	ENDWHILE		00001040
				714	DECAP 2	FIN SORTIES	00001050
0002D6	9248 803E	0003E		719	MVI MEM2(TERMAREA), X'4B'		00001060
				720	CALL &SORTIE1		00001070
				723	SORTIE 75, I=1	X'4B'	00001080
				727	CCDR 0(WORK1), WORK1	PLACE SUR SIG HOR	00001090
				731	CAR 0(WORK1), WORK2		00001100
0002FE	9226 803E	0003E		735	MVI MEM2(TERMAREA), 38	FINT38	00001110

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	
00035E	9221 4001	00001		739	PUSH 0	00001100
00036E	9218 4001	00001		748	PUSH 33	00001110
00037E				757	WHILE H,0(WORK2),NE,ANIFT	00001120
00038E				762	CAR 0(WORK2),WORK2	00001130
00039E				766	IF B,1(WORK2),EQ,95	00001131
00040E				769	MVI 1(CONSICNT),33	00001132
00041E				770	ENDIF ELSE	00001133
00042E				773	MVI 1(CONSICNT),24	00001134
00043E				774	ENDELSE	00001135
00044E				776	MVC MEM2(1,TERMAREA),1(WORK2)	00001140
00045E				777	CALL SORTIEI	00001150
00046E				780	MVC PARM(1),1(WORK2)	00001160
00047E				781	CALL STRM4	00001170
00048E				784	ENDWHILE	00001180
00049E				787	DECAP 2	00001190
00050E				792	MVC TE4(1),BOOL	00001200
00051E				793	MVI BOOL,0	00001210
00052E				794	MVI MEM2(TERMAREA),X*48	00001220
00053E				795	CALL SORTIEI	00001230
00054E				798	CCDR 0(WORK1),WORK1	00001240
00055E				802	CAR 0(WORK1),WORK2	00001250
00056E				806	MVI MEM2(TERMAREA),37	00001260
00057E				807	CALL SORTIEI	00001270
00058E				810	MVI PARM,X*60	00001271
00059E				811	WHILE H,0(WORK2),NE,ANIFT	00001280
00060E				816	CAR 0(WORK2),WORK2	00001290
00061E				820	MVC MEM2(1,TERMAREA),0(WORK2)	00001300
00062E				821	CALL SORTIEI	00001310
00063E				824	MVC MEM2(1,TERMAREA),1(WORK2)	00001320
00064E				825	CALL SORTIEI	00001330
00065E				828	CALL STRM4	00001340
00066E				831	ENDWHILE	00001350
00067E				834	MVI MEM2(TERMAREA),X*48	00001360
00068E				835	CALL SORTIEI	00001370
00069E				838	MVC BOOL(1),TE4	00001380
00070E				839	CCDR 0(WORK1),WORK1	00001390
00071E				843	MVC TEM1(2),0(WORK1)	00001400
00072E				844	CCDR 0(WORK1),WORK1	00001410
00073E				848	CAR 0(WORK1),WORK2	00001420
00074E				852	WHILE H,0(WORK2),NE,ANIFT	00001430
00075E				857	MVI MEM2(TERMAREA),39	00001440
00076E				858	CALL SORTIEI	00001450
00077E				861	CCDR 0(WORK2),WORK2	00001460
00078E				865	MVC TEM2(2),0(WORK2)	00001470
00079E				866	CCDR 0(WORK2),WORK2	00001480
00080E				870	CAR 0(WORK2),WORK2	00001490
00081E				874	MVC MEM2(1,TERMAREA),0(WORK2)	00001500
00082E				875	CALL SORTIEI	00001510
00083E				878	MVC MEM2(1,TERMAREA),1(WORK2)	00001520
00084E				879	CALL SORTIEI	00001530
00085E				882	LA WORK2,2(WORK2)	00001540
00086E				883	MVC TEM3(2),0(WORK2)	00001550
00087E				884	CAR 0(WORK2),WORK3	00001560
00088E				888	CCDR 0(WORK3),WORK3	00001570
00089E				892	MVI BOOLA,0	00001580

X
IDS
EX
EO
SYMB. IDS DU IDH
SAUVE LE BOOLEEN
FIN FINT38
PLACE SUR REGISTRES
FINT37
IDR
NUM FONCTION BASCULE
FIN FINT37
RESTAURE LE BOOLEEN
PLACE SUR EXTERNE
FINT39
SAUTE IND. BOUCLE POUR SORTIES DE L'EXTERNE
LIBELLE
ENTREES

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT		
00048C	411E 0002		0C002	893	ET2TRM2	WHILE H,0(WORK3),NE,ANIFT	00001590
				899	CAR	O(WORK3),WORK3	00001600
				904	LA	WORK1,2(WORK3)	00001610
				908	CAR	O(WORK1),WORK1	00001620
				912	CAR	O(WORK1),WORK1	00001630
				915	IF	B,1(WORK1),EQ,X'67'	00001640
				916	B	ET3TRM2	00001650
0004E0	47F0 C3B2		0C4F8	918	ENDIF		00001660
				921	IF	B,1(WORK1),EQ,X'60'	00001670
				922	B	ET3TRM2	00001680
				924	ENDIF		00001690
0004F8	925F 803E	0003E		927	ET3TRM2	B,1(WORK1),EQ,X'5F'	00001700
				928	MVI	MEM2(TERMAREA),X'5F'	00001710
				931	ENDIF	ELSE	00001720
				932	MVI	MEM2(TERMAREA),X'61'	00001730
				934	ENDElse		00001740
				937	CALL	&SORTIEI	00001750
				941	SORTIE	O(WORK3),I=1	00001760
				945	SORTIE	1(WORK3),I=1	00001770
00051C	4111 0002		00002	946	LA	WORK1,2(WORK1)	00001780
				950	CAR	O(WORK1),WORK1	00001790
				954	SORTIE	O(WORK1),I=1	00001800
				958	SORTIE	1(WORK1),I=1	00001810
000540	41EE C002		0C002	959	LA	WORK3,2(WORK3)	00001820
				963	CCDR	O(WORK3),WORK3	00001830
				966	ENDWHILE		00001840
000554	9248 803E	0003E		967	MVI	MEM2(TERMAREA),X'48'	00001850
				970	CALL	&SORTIEI	00001860
				973	IF	B,BOOLA,EQ,0	00001870
				974	MVI	BOOLA,X'FF'	00001880
000564	92FF 804A	0004A		978	CAR	TEM2,WORK3	00001890
				982	CCDR	O(WORK3),WORK3	00001900
000580	47F0 C35E		0C04A	983	B	ET2TRM2	00001910
				985	ENDIF		00001920
000590	9248 803E	0003E		989	CCDR	TEM3,WORK2	00001930
				990	MVI	MEM2(TERMAREA),X'48'	00001940
				993	CALL	&SORTIEI	00001950
00059C	9223 803E	0003E		996	ENDWHILE		00001960
				997	MVI	MEM2(TERMAREA),35	00001970
				1000	CALL	&SORTIEI	00001980
				1004	CAR	TEM1,WORK2	00001990
				1005	MVI	MEM2(TERMAREA),42	00002000
				1008	CALL	&SORTIEI	00002010
				1013	WHILE	H,0(WORK2),NE,ANIFT	00002020
				1017	CAR	O(WORK2),WORK2	00002030
				1021	CCDR	O(WORK2),WORK3	00002040
				1024	CALL	&TRM1	00002050
0005E0	923F 803E	0003E		1025	MVI	MEM2(TERMAREA),X'3F'	00002060
				1029	CALL	&SORTIEI	00002070
				1033	LA	WORK2,2(WORK2)	00002080
				1037	CAR	O(WORK2),WORK2	00002090
				1038	IF	H,0(WORK2),GT,0	00002100
				1042	LA	WORK3,2(WORK2)	00002110
000604	41E2 C002		00002	1043	CAR	O(WORK3),WORK3	00002120
				1044	CCDR	O(WORK3),WORK3	00002130

FIN DES ENTREES DE L'EXTER

SORTIES

TERMINE ESP LIST

DESCRIPTION
FINT42

::=

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	CONCATENATION
000648	9238 803E	0003E		1046	IF H,0(WORK2),EQ,X'38°	
				1050	WHILE H,0(WORK3),NE,ANIFT	
				1055	CALL &TRM1	
				1058	IF H,0(WORK3),NE,ANIFT	
				1062	MVI MEM2(TERMAREA),X'38°	
				1063	CALL &SORTIEI	
				1066	ENDIF	
				1068	ENDWHILE	
				1071	ENDIF ELSE	
				1074	CALL &TRM1	
				1077	ENDELSE	
00065C	4122 0002	00002		1079	LA WORK2,2(WORK2)	
				1080	CAR 0(WORK2),WORK2	
				1084	ENDIF ELSE	
				1087	CAR 0(WORK2),WORK3	
00067C	D200 803E E001 0003E	0C001		1091	MVC MEM2(1,TERMAREA),1(WORK3)	
				1092	CALL &SORTIEI	
000686	41EE C002	0C002		1095	LA WORK3,2(WORK3)	
				1096	WHILE H,0(WORK3),NE,ANIFT	
				1101	CAR 0(WORK3),WORK3	
				1105	MVI MEM2(TERMAREA),X'2C°	
				1106	CALL &SORTIEI	
0006AA	D200 803E E000 0003E	00000		1109	MVC MEM2(1,TERMAREA),0(WORK3)	
				1110	CALL &SORTIEI	
0006B4	D200 803E E001 0003E	0C001		1113	MVC MEM2(1,TERMAREA),1(WORK3)	
				1114	CALL &SORTIEI	
				1117	MVI MEM2(TERMAREA),X'2C°	
				1118	CALL &SORTIEI	
0006C6	41EE C002	00002		1121	LA WORK3,2(WORK3)	
				1122	ENDWHILE	
				1125	LA WORK2,2(WORK2)	
				1126	CAR 0(WORK2),WORK2	
				1130	CCDR 0(WORK2),WORK3	
				1134	CALL &TRM1	
				1137	ENDELSE	
0006EE	9229 803E	0003E		1139	MVI MEM2(TERMAREA),X'29°	
				1140	CALL &SORTIEI	
0006F6	4122 C002	00002		1143	LA WORK2,2(WORK2)	
				1144	ENDWHILE	
				1147	MVI MEM2(TERMAREA),X'4B°	
				1148	CALL &SORTIEI	
				1151	MVI MEM2(TERMAREA),43	
				1152	CALL &SORTIEI	
				1155	CCDR TEM1,WORK2	
				1159	CAR 0(WORK2),WORK2	
				1163	WHILE H,0(WORK2),NE,ANIFT	
				1169	CAR 0(WORK2),WORK2	
				1173	IF H,0(WORK2),NE,0	
				1177	LA WORK2,2(WORK2)	
				1178	CCDR 0(WORK2),WORK2	
				1182	CCDR 0(WORK2),WORK2	
				1186	CCDR 0(WORK2),WORK2	
				1190	B ET4TRM2	
				1191	ENDIF	
000772	47F0 C5E0	C0726		1190	LA WORK2,2(WORK2)	
000776	4122 0002	00002		1191	LA WORK2,2(WORK2)	
				1193	LA WORK2,2(WORK2)	

FIN FINT42
FINT43

P-18

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	
				1194	CAR O(WORK2),WORK1	00002690
				1198	CCDR O(WORK2),WORK2	00002700
				1202	CAR O(WORK2),WORK2	00002710
	925E 8C3E	0003E		1206	MVI MEM2(TERMAREA),X'5E'	00002720
				1207	CALL &SORTIEI	00002730
	02C0 803E 2000	0C003E	0C000	1210	MVC MEM2(1,TERMAREA),O(WORK2)	00002740
				1211	CALL &SORTIEI	00002750
	0200 803E 2001	0003E	00001	1214	MVC MEM2(1,TERMAREA),I(WORK2)	00002760
				1215	CALL &SORTIEI	00002770
	923C 803E	0003E		1218	MVI MEM2(TERMAREA),X'3C'	00002780
				1219	CALL &SORTIEI	00002790
	4122 C002		0C002	1222	LA WORK2,2(WORK2)	00002800
				1223	CAR O(WORK2),WORK3	00002810
				1227	CCDR O(WORK3),WORK3	00002820
				1231	WHILE H,O(WORK3),NE,ANIFT	00002830
				1236	CALL &TRM1	00002840
				1239	IF H,O(WORK3),NE,ANIFT	00002850
				1243	MVI MEM2(TERMAREA),X'3A'	00002860
	923A 803E	0003E		1244	CALL &SORTIEI	00002870
				1247	ENDIF	00002880
				1249	ENDWHILE	00002890
	9229 803E	0003E		1252	MVI MEM2(TERMAREA),X'29'	00002900
				1253	CALL &SORTIEI	00002910
				1256	CCDR O(WORK1),WORK3	00002920
				1260	WHILE H,O(WORK3),NE,ANIFT	00002930
				1265	CALL &TRM1	00002940
				1268	IF H,O(WORK3),NE,ANIFT	00002950
	923A 803E	0003E		1272	MVI MEM2(TERMAREA),X'3A'	00002960
				1273	CALL &SORTIEI	00002970
				1276	ENDIF	00002980
				1278	ENDWHILE	00002990
	9230 803E	0003E		1281	MVI MEM2(TERMAREA),X'3D'	00003000
				1282	CALL &SORTIEI	00003010
				1285	MVI MEM2(TERMAREA),X'29'	00003020
	9229 803E	0003E		1286	CALL &SORTIEI	00003030
				1289	CCDR O(WORK2),WORK2	00003040
				1293	ENDWHILE	00003050
	9248 803E	0003E		1296	MVI MEM2(TERMAREA),X'48'	00003060
				1297	CALL &SORTIEI	00003070
	922C 803E	0003E		1300	MVI MEM2(TERMAREA),44	00003080
				1301	CALL &SORTIEI	00003090
				1304	CCDR TEN1,WORK2	00003100
				1308	CAR O(WORK2),WORK2	00003110
				1312	WHILE H,O(WORK2),NE,ANIFT	00003120
				1318	CAR O(WORK2),WORK2	00003130
				1322	IF H,O(WORK2),EQ,0	00003140
	4122 C002		00002	1326	LA WORK2,2(WORK2)	00003150
				1327	CCDR O(WORK2),WORK2	00003160
				1331	CCDR O(WORK2),WORK2	00003170
				1335	CCDR O(WORK2),WORK2	00003180
				1339	B ET5TRM2	00003190
	47F0 C744		0C88A	1340	ENDIF	00003200
				1342	LA WORK2,2(WORK2)	00003210
	4122 0002		00002	1343	CAR O(WORK2),WORK2	00003220
				1347	CCDR O(WORK2),WORK3	00003230

SORTIES

LIBELLE EXT.
IDVE

PG

ENTREES EXT

VIRG ENT

PV

VIRG SORT

PD

PV

FIN FINT43

FINT44

SORTIES

12 MAR 70

SOURCE STATEMENT

1351	CALL	&TRM1		00003240
1354	MVI	MEM2(TERMAREA),X'3F'		00003250
1355	CALL	&SORTIEI		00003260
1358	LA	WORK2,2(WORK2)		00003270
1359	CAR	0(WORK2),WORK2		00003280
1363	MVI	MEM2(TERMAREA),X'5E'	LIBELLE EXT	00003290
1364	CALL	&SORTIEI	IDVE	00003300
1367	MVC	MEM2(1,TERMAREA),0(WORK2)		00003310
1368	CALL	&SORTIEI		00003320
1371	MVC	MEM2(1,TERMAREA),1(WORK2)		00003330
1372	CALL	&SORTIEI		00003340
1375	LA	WORK2,2(WORK2)		00003350
1376	CAR	0(WORK2),WORK3		00003360
1380	MVI	MEM2(TERMAREA),X'3C'	ENTREES EXT	00003370
1384	CALL	&SORTIEI	PG	00003380
1388	CCDR	0(WORK3),WORK3		00003390
1393	WHILE	H,0(WORK3),NE,ANIFT		00003400
1396	CALL	&TRM1		00003410
1400	IF	H,0(WORK3),NE,ANIFT		00003420
1401	MVI	MEM2(TERMAREA),X'3A'	VIRG ENT	00003430
1404	CALL	&SORTIEI		00003440
1406	ENDIF			00003450
1409	ENDWHILE			00003460
1410	MVI	MEM2(TERMAREA),X'29'	PV	00003470
1413	CALL	&SORTIEI		00003480
1414	MVI	MEM2(TERMAREA),X'3B'	EVOILE	00003490
1417	CALL	&SORTIEI		00003500
1418	MVI	MEM2(TERMAREA),X'3D'	PD	00003510
1421	CALL	&SORTIEI		00003520
1422	MVI	MEM2(TERMAREA),X'29'	PV	00003530
1425	CALL	&SORTIEI		00003540
1429	CCDR	0(WORK2),WORK2		00003550
1432	ENDWHILE			00003560
1433	MVI	MEM2(TERMAREA),X'4B'	FIN FINT44	00003570
1436	CALL	&SORTIEI		00003580
1438	EXIT			00003590
1439	DS	C	POUR SAUVER LE BOOLEEN	00003600
1439	RETURN			00003610

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	IDS
				1443	ROUTINE &TRM3	00003630
				1446	IF B,1(CONS ICNT),EQ,95	00003640
000904	9221 4001	00001		1449	MVI 1(CONS ICNT),33	00003650
				1450	ENDIF	00003660
				1452	IF B,1(CONS ICNT),EQ,97	00003670
000900	9211 4001	00001		1455	MVI 1(CONS ICNT),17	00003680
				1456	ENDIF	00003690
				1458	IF B,1(CONS ICNT),EQ,76	00003700
000900	9200 4001	00001		1461	MVI 1(CONS ICNT),13	00003710
				1462	ENDIF	00003720
				1464	CALL &CRNOM	00003730
				1467	PERMUTER	00003740
				1474	PULL STRM3	00003750
				1478	CREERLIS	00003760
				1486	PUSH 255	00003770
00093E	92FF 4000	00000		1495	MVI 0(CONS ICNT),X'FF'	00003780
				1496	CAR STRM3,WORK1	00003790
				1500	CCDR 0(WORK1),WORK1	00003800
				1504	CAR 0(WORK1),WORK2	00003810
				1508	WHILE H,0(WORK2),NE,ANIFT	00003820
				1513	PUSH 0	00003830
				1522	PUSH 0(WORK2)	00003840
				1530	CCDR 0(WORK2),WORK2	00003850
				1534	CCDR 0(WORK2),WORK2	00003860
				1538	CCDR 0(WORK2),WORK2	00003870
				1542	ENDWHILE	00003880
				1545	CALL &RECUPES	00003890
				1548	DECAP 1	00003900
				1551	CONS 2(CONS ICNT),0(CONS ICNT)	00003910
				1567	DECAP 1	00003920
				1570	CONS 0(CONS ICNT),2(CONS ICNT)	00003930
000958	D201 4000 4002 C0C00 0C002			1586	MVC 0(2,CONS ICNT),2(CONS ICNT)	00003940
				1587	PUSH 255	00003950
00097A	92FF 4000	00000		1596	MVI 0(CONS ICNT),X'FF'	00003960
				1597	CCDR 0(WORK1),WORK1	00003970
				1601	CAR 0(WORK1),WORK2	00003980
				1605	WHILE H,0(WORK2),NE,ANIFT	00003990
				1610	PUSH 0	00004000
				1619	PUSH 0(WORK2)	00004010
				1627	CCDR 0(WORK2),WORK2	00004020
				1631	CCDR 0(WORK2),WORK2	00004030
				1635	CCDR 0(WORK2),WORK2	00004040
				1639	ENDWHILE	00004050
				1642	CALL &RECUPES	00004060
				1645	DECAP 1	00004070
				1648	CONS 2(CONS ICNT),0(CONS ICNT)	00004080
				1664	CONS 0,0(CONS ICNT)	00004090
000C86	92FF 803A	0003A		1681	MVI BOOL,X'FF'	00004100
				1682	EXIT	00004110
000C8E				1684	DS H	00004120
				1685	RETURN	00004130

SAUVE LA LISTE
MARQUE

LIBELLE DE L'EXTERNE

MARQUE

SORTIES

IND SORTIE BOUCLE POUR

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	ROUTINE	12 MAR 70
000C94	4122 0002	0C002		1689	ROUTINE	ETIRM4	00004150
				1692	LA	WORK2,2(WORK2)	00004160
				1693	SORTIE	0(WORK2),I=1	00004170
				1697	SORTIE	1(WORK2),I=1	00004180
000C88	D200 803E 2000 0003E 00000			1701	CAR	0(WORK2),WORK2	00004190
				1705	MVC	MEM2(1,TERMAREA),0(WORK2)	00004200
000CC2	D200 803E 2001 0003E 00001			1706	CALL	&SORTIEI	00004210
				1709	MVC	MEM2(1,TERMAREA),1(WORK2)	00004220
000CCC	4122 0002	00002		1710	CALL	&SORTIEI	00004230
				1713	LA	WORK2,2(WORK2)	00004240
				1714	CAR	0(WORK2),WORK2	00004250
				1718	IF	B,BOOL,EQ,X,FF,	00004260
				1721	ST	WORK1,SR1	00004270
000CE4	5010 C208	00E9C		1722	ST	WORK2,SR2	00004280
000CE8	5020 C20C	00EAO		1723	ST	WORK3,SR3	00004290
000CEC	50E0 C210	00EA4		1724	LA	WORK1,NLIB2	00004300
000CF0	4110 81C0	001C0		1725	WHILE	H,0(WORK1),NE,ANIFT	00004310
				1730	CAR	0(WORK1),WORK1	00004320
				1734	CAR	0(WORK1),WORK2	00004330
				1738	IF	H,0(CONSICNT),EQ,0(WORK2)	00004340
000D24	47F0 C09C	00D30		1742	B	ETIIRM4	00004350
				1743	ENDIF		00004360
000D28	4111 C002	00002		1745	LA	WORK1,2(WORK1)	00004370
				1746	ENDWHILE		00004380
000D30	4122 C002	0C002		1749	LA	WORK2,2(WORK2)	00004390
				1750	CAR	0(WORK2),WORK2	00004400
				1754	PERMUTER		00004410
				1761	IF	H,0(CONSICNT),LT,0(WORK2)	00004420
000D66	4122 C002	00002		1765	LA	WORK2,2(WORK2)	00004430
000D6A	41E0 0001	00001		1766	LA	WORK3,1	00004440
				1767	WHILE	H,WORK3,LT,0(CONSICNT)	00004450
				1771	CCDR	0(WORK2),WORK2	00004460
				1775	LA	WORK3,1(WORK3)	00004461
000D82	41EE 0001	C 0001		1776	ENDWHILE		00004470
				1779	CAR	0(WORK2),WORK2	00004480
000D96	58E0 8198	00198		1783	L	WORK3,PTRAD	00004490
000D9A	D200 E002 8126 00002 00126	00126		1784	MVC	2(1,WORK3),PAM	00004500
000DA0	D201 E004 2000 00004 C0000	C0000		1785	MVC	4(2,WORK3),0(WORK2)	00004510
				1786	ENDIF	ELSE	00004520
				1789	PERMUTER		00004530
				1796	DOUBLER		00004540
				1805	CALL	&CRSIG	00004550
000DE4	58E0 8198	00198		1808	L	WORK3,PTRAD	00004560
000DE8	D200 E002 8126 00002 00126	00126		1809	MVC	2(1,WORK3),PAM	00004570
000DEE	D201 E004 4000 00004 C0000	C0000		1810	MVC	4(2,WORK3),0(CONSICNT)	00004580
				1811	DECAP	1	00004590
				1814	PERMUTER		00004600
				1821	ENDELSE		00004610
000E12	5820 C20C	C0EAO		1823	L	WORK2,SR2	00004620
000E16	D201 E000 2000 00000 00000	00000		1824	MVC	0(2,WORK3),0(WORK2)	00004630
				1825	SORTIE	4(WORK3),I=1	00004640
				1829	SORTIE	5(WORK3),I=1	00004650
000E30	41EE 0006	C0006		1833	LA	WORK3,6(WORK3)	00004660
000E34	50E0 8198	00198		1834	ST	WORK3,PTRAD	00004670
				1835	IF	F,PTRAD,GT,FTRAD	00004680

SORTIE DIM

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000E44	41F0 0017	00017		1839	LA PARMADD,23
				1840	CALL &ERREUR
				1843	ENDIF
000E4C	48E4 0000	00000		1845	LH WORK3,0(CONSICNT)
000E50	41EE 0001	00001		1846	LA WORK3,1(WORK3)
000E54	40E4 C000	00000		1847	STH WORK3,0(CONSICNT)
				1848	PERMUTER
000E72	5810 C208	00E9C		1855	L WORK1,SRI
000E76	58E0 C210	00EA4		1856	L WORK3,SR3
				1857	ENDIF ELSE
000E7E	D2C0 803E 2000 0003E 00000	00000		1860	MVC MEM2(1,TERMAREA),0(WORK2)
000E88	D2C0 803E 2001 0003E 00001	00001		1861	CALL &SORTIE1
				1864	MVC MEM2(1,TERMAREA),1(WORK2)
				1865	CALL &SORTIE1
				1868	ENDELSE
000E92	4122 0002	C0002		1870	LA WORK2,2(WORK2)
				1871	EXIT
000E9C				1873	DS SR1
000EAO				1874	DS SR2
000EA4				1875	DS SR3
				1876	RETURN

SORTIE LIBELLE

00004690
00004700
00004710
00004720
00004730
00004740
00004750
00004760
00004770
00004780
00004790
00004800
00004810
00004820
00004830
00004840
00004850
00004860
00004870
00004880
00004890

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000EAC	5010 C588			1880	ROUTINE &CRNOM
000EBO	5020 C58C	01434		1883	ST WORK1,SCR1
000EB4	50E0 C590	01438		1884	ST WORK2,SCR2
000EB8	4110 81C2	0143C		1885	ST WORK3,SCR3
		001C2		1886	LA WORK1,LUNICR
				1887	WHILE H,0(WORK1),NE,ANIFT
				1892	CAR 0(WORK1),WORK1
				1896	CAR 0(WORK1),WORK2
				1900	IF H,0(CONSICNT),EQ,0(WORK2)
000EEC	47F0 C054	00F00		1904	B ETICRNOM
				1905	ENDIF
000EFO	4111 0002	00002		1907	LA WORK1,2(WORK1)
				1908	ENDWHILE
000EF8	41F0 0016	00016		1911	LA PARMADD,22
				1912	CALL &ERREUR
000FF0	4122 0002	00002		1915	ETICRNOM
				1916	LA WORK2,2(WORK2)
000F10	4812 0000	00000		1920	LH WORK1,0(WORK2)
000F14	4111 0001	00001		1921	LA WORK1,1(WORK1)
000F18	4012 0000	00000		1922	STH WORK1,0(WORK2)
000F1C	41E0 81FC	001FC		1923	LA WORK3,NOMUNITT
				1924	IF B,1(CONSICNT),EQ,23
000F28	9217 C2C6	01172		1927	MVI CRBOOL,23
				1928	CREERLIS
				1936	CONS ANIFT,0(CONSICNT)
				1952	CONS ANIFT,0(CONSICNT)
				1968	PERMUTER
				1975	ENDIF
				1977	CREERLIS
				1985	PERMUTER
001010	9238 4000	00000		1992	MVI 0(CONSICNT),56
				1993	DECAP 1
				1996	CONS 2(CONSICNT),0(CONSICNT)
				2012	CONS 0(WORK3),0(CONSICNT)
				2028	CONS 2(WORK3),0(CONSICNT)
				2044	CALL &NUMEROT
0010D6	4122 0002	00002		2047	LA WORK2,2(WORK2)
				2048	IF B,CRBOOL,EQ,23
				2051	DECAP 1
				2054	ACCROCHE 4(CONSICNT),0(CONSICNT)
				2066	ACCROCHE 0(WORK2),4(CONSICNT)
				2078	MVC 0(2,CONSICNT),2(CONSICNT)
001132	D201 4000 4002 00000 00002			2079	ENDIF ELSE
				2082	ACCROCHE 0(WORK2),2(CONSICNT)
				2094	ENDELSE
001162	5810 C588	01434		2096	L WORK1,SCR1
001166	5820 C58C	01438		2097	L WORK2,SCR2
00116A	58E0 C590	0143C		2098	L WORK3,SCR3
				2099	EXIT
001172				2101	DS C
				2102	RETURN

NOM DE L'UNITE TRAITEE
CONNEXION

&

12 MAR 70

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	
				2106	ROUTINE &CRSIG	00005430
				2109	ST WORK1,SCR1	00005440
C01178	5010 C2BC	01434		2110	ST WORK2,SCR2	00005450
C0117C	5020 C2C0	01438		2111	ST WORK3,SCR3	00005460
C01180	50E0 C2C4	0143C		2112	IF B,1(CONSICNT),EQ,35	00005470
				2115	B ETOCRSIG	00005480
C0118C	47F0 C02C	011A4		2116	ENDIF	00005490
				2118	IF B,1(CONSICNT),EQ,34	00005500
				2121	B ETOCRSIG	00005510
C01198	47F0 C02C	011A4		2122	ENDIF	00005520
				2124	IF B,1(CONSICNT),EQ,17	00005530
				2127	LA WORK1,NLIB1	00005540
0011A4	4110 81BE	C01BE		2128	ENDIF ELSE	00005550
				2131	LA WORK1,NLIB2	00005560
C011AC	4110 81C0	001C0		2132	ENDELSE	00005570
				2134	WHILE H,0(WORK1),NE,ANIFT	00005580
				2139	CAR 0(WORK1),WORK1	00005590
				2143	CAR 0(WORK1),WORK2	00005600
				2147	IF H,0(CONSICNT),EQ,0(WORK2)	00005610
				2151	B ETICRSIG	00005620
0011E0	47F0 C07C	011F4		2152	ENDIF	00005630
				2154	LA WORK1,2(WORK1)	00005640
C011E4	4111 0002	00002		2155	ENDWHILE	00005650
				2158	LA PARMADD,21	00005660
0011EC	41F0 0015	0C015		2159	CALL &ERREUR	00005670
				2162	LA WORK2,2(WORK2)	00005680
C011F4	4122 0002	00002		2163	CAR 0(WORK2),WORK2	00005690
				2167	LH WORK1,0(WORK2)	00005700
001204	4812 0000	00000		2168	LA WORK1,1(WORK1)	00005710
001208	4111 0001	00001		2169	STH WORK1,0(WORK2)	00005720
C0120C	4012 0000	CC000		2170	CREERLIS	00005730
				2178	PERMUTER	00005740
				2185	MVI 0(CONSICNT),56	00005750
001244	9238 4000	00000		2186	DECAP 1	00005760
				2189	CONS 2(CONSICNT),0(CONSICNT)	00005770
				2205	CALL &NUMEROT	00005780
00128E	4122 C002	00002		2208	LA WORK2,2(WORK2)	00005790
				2209	ACCRUCHE 0(WORK2),2(CONSICNT)	00005800
C012B8	D201 4000	4002 00000		2221	MVC 0(2,CONSICNT),2(CONSICNT)	00005810
C012BE	5810 C2BC	01434		2222	L WORK1,SCR1	00005820
C012C2	5820 C2C0	01438		2223	L WORK2,SCR2	00005830
0012C6	58E0 C2C4	0143C		2224	L WORK3,SCR3	00005840
				2225	RETURN	00005850

+1 SUR LE NUMERO D'ORDRE
WORK1 CONTIENT LE BON N.

VU

Grenoble, le

Le Président de la Thèse

VU

Grenoble, le

Le Doyen de la Faculté des Sciences

Vu, et permis d'imprimer

Le Recteur de l'Académie de GRENOBLE

