Système couplé PDP-8/IBM-360 pour l'assemblage et la transmission de programmes

Peter Jones

N° D'ORDRE

T H E S E

Présentée à la Faculté des Sciences de Grenoble
pour obtenir le titre de
Docteur-Ingénieur en Informatique

PAR

Peter JONES

SYSTEME COUPLE PDP-8/IBM-360
POUR L'ASSEMBLAGE ET LA TRANSMISSION DE PROGRAMMES

Thèse soutenue le 28 Septembre 1970 devant la commission d'Examen

MM.	*GASTINEL*	Président
	BOLLIET	Examinateur
	GRIFFITHS	Examinateur

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 Noel	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 Electrométallurgie
	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.	LONGEQUEUE 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)
	BELORIZKY 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

ERRATA

Page 12, ligne 1 :

"... 360 au du..." devrait être

"... 360 au moment du..."

Page 12, paragraphe 3, ligne 2 :

"tel que défini" devrait être "tel qu'il a été défini"

Page 19, milieu de la page :

Remplacer la phrase :"Mais TAO Y/COM donne..."

par : "Mais TAO Y /COM donne..."

Page 19, paragraphe 4, ligne 2 :

"... d'emplacement donnant..." devrait être

"... d'emplacement. Ceci donne..."

Page 34, paragraphe 2, ligne 2 :

"... peut être" devrait être

"... pourrait être..."

Page 44, paragraphe 3, ligne 2 :

"... est imprédictible..." devrait être

"... n'est pas accessible au programme 360..."

AVANT-PROPOS

Je tiens à remercier les nombreuses personnes qui ont rendu possible la préparation et la présentation de cette thèse.

Monsieur le Professeur N. GASTINEL m'a fait l'honneur de présider au Jury de Thèse.

Monsieur L. BOLLIET, Directeur de la Thèse, m'a donné beaucoup d'aide et d'encouragement pendant la réalisation du travail. Ses suggestions sur le premier manuscrit ont permis plusieurs améliorations.

Monsieur M. GRIFFITHS a bien voulu faire partie du Jury.

Le travail des élèves Messieurs BERGER et JAN, qui ont mis en route la première version de l'assembleur PDP-8 sur 360, a permis un gain de temps important.

La construction de la liaison PDP-8/360 par mes collègues Messieurs TUFFELLI et VUILLOD, était indispensable pour la mise en oeuvre du système de couplage.

La réalisation du premier programme de transmission de données au PDP-8 vers le 360 par mon collègue, Monsieur MONNEROT, a permis de repérer les difficultés de programmation dues au hardware.

Enfin, je remercie le personnel du service de dactylographie et de reproduction de l'IMAG d'avoir rendu possible la publication de cet ouvrage.

TABLE DES MATIERES

CHAPITRE I

INTRODUCTION

INTRODUCTION

Cette thèse décrit la mise en oeuvre d'un système d'assemblage de programmes PDP-8 sur IBM 360, avec transmission du programme assemblé au PDP-8 pour l'exécution. Ce projet a été terminé à l'Institut de Mathématiques Appliquées de Grenoble en Juin 1970.

Le PDP-8 est un petit ordinateur destiné à des applications d'acquisition d'informations et éventuellement de contrôle en temps réel ; il est relativement simple de relier le PDP-8 a des périphériques spécialisés, ce qui permet une adaptation facile à une application particulière. A l'IMAG les applications principales sont les suivantes : la digitalisation de données analogiques, l'acquisition d'images et, les développements de système graphiques.

Un petit calculateur devient beaucoup plus rentable s'il a accès à une machine plus puissante. Par exemple, la tâche simple qui consiste à assembler un programme est assez lourde sur une petite machine. La mémoire de celle-ci est très petite, et ses périphériques sont trop lents pour un gros assemblage. L'impression d'une liste d'assemblage sur un PDP-8 se fait sur un télétype à 10 caractères par seconde. Une imprimante rapide pour le PDP-8 coûterait plus cher que le PDP-8 lui-même, et elle serait peu utilisée. Son achat n'est donc pas justifiée.

Le traitement de grandes quantités de données est difficile sur un petit calculateur à cause du jeu d'instructions restreint, et la petite taille de la mémoire. Les calculs en virgule flottante sont souvent interprétatifs, par exemple : ce qui rend prohibitif le temps de calcul pour un traitement évolué.

Mais le petit calculateur est très adapté pour la collection, la compression, et le prétraitement de données prises en temps réel. Son excellent temps de réponse le rend très intéressant, aussi pour les intéractions en temps réel, notamment les applications graphiques.

Configuration des machines de l'IMAG

La configuration des machines de l'IMAG comprend deux calculateurs IBM 360, modèles 67 et 40, reliés ensemble, et un PDP-8 attaché au 67. La configuration des deux 360 inclut des disques propres à chaque machine, quatre dérouleurs de bande commutables, deux lecteurs/perforateurs de cartes, et deux imprimantes.

Le PDP-8 est muni de plusieurs genres de périphériques, y compris des périphériques spéciaux construits à l'IMAG. Comme périphériques standards, il y a une console de pupitre avec lecteur/perforateur de bande perforée lents, deux dérouleurs de bande magnétique (DECtape), un tambour RM08, un multiplexeur 680 pour des télétypes, un écran cathodique 30N, et un convertisseur analogique-digital. Parmi les périphériques construits à l'IMAG se trouvent une caméra de télévision qui sert à rentrer des images dans la mémoire du PDP-8 et l'interface permettant la communication avec le 360.

On voit que le PDP-8 est très intéressant pour les projets spéciaux : programmes de dessin graphique, acquisition de données analogiques, et l'acquisition d'images. Mais ces programmes ont besoin d'un grand calculateur pour l'assemblage de programme PDP-8. L'assemblage de gros programmes PDP-8 sur le PDP-8 pose beaucoup de problèmes parce que les périphériques du PDP-8 sont lents et la mémoire centrale petite. Ainsi, il n'est pas possible de construire une table de références croisées (outil très utile pour la mise au point de programmes en langage d'assembleur), ni même une grosse table de symboles. Une liste d'assemblage prend beaucoup de temps à sortir sur la console du PDP-8, (télétype) et bloque le passage de programmes plus intéressants. Parce qu'il n'y a pas de traitement par lots sur le PDP-8, chaque programmeur doit s'occuper de l'assemblage de ses programmes.

Le gros calculateur, le 360/67, sert aussi pour le stockage et le traitement des résultats de l'acquisition d'images et la digitalisation de données analogiques. La quantité de données à traiter est énorme. Par exemple, une seule image de 300 points par 300 points contient 90,000 mots de données, chaque mot correspondant à un point de l'image. De même, une minute de données analogiques échantillonnées à 1000 Hz occupe 60,000 mots. La mémoire du PDP-8, avec sa faible capacité de 12,000 mots de 12 bits, est évidemment trop petite pour contenir une telle quantité d'informations. Par contre, les disques et les bandes de l'IBM 360 permettent d'enmagasiner des millions de mots. On peut donc envisager le stockage d'une centaine d'images, ou d'une heure de données analogiques.

Pour les opérations en temps partagé, le 67 dispose de deux unités de contrôle de transmission : un 2701 et un 2702. Le 2702 permet au 67 de communiquer avec plusieurs terminaux de type 2741, et des télétypes. Le 2701 contient deux adaptateurs de transmission. Un IBM 1130 est relié à l'une de ces unités. Le PDP-8 est relié à l'autre par l'intermédiaire d'un adaptateur de données en parallèle, ou "Paralell Data Adapter" (PDA).

BUT DU PROJET

La rentabilité d'un petit calculateur augmente considérablement si on peut le relier à un ordinateur plus puissantLa connexion peut se faire, soit d'une façon non-conversationnelle (bande magnétique, carte perforée, ou ruban perforé), ou d'une façon conversationnelle au moyen d'une interface couplant les deux machines électriquement. Cette deuxième solution a été choisie par l'IMAG pour permettre un dialogue éventuel entre le 360 et le PDP-8. Néanmoins, la conversation revient très cher, et si elle n'est pas indispensable, comme dans le cas de collection de grandes quantités de données, par exemple, la rentabilité à long terme d'un système de communication basé sur l'emploi de bandes magnétiques peut être facilement démontrée.

Les avantages d'un système non-conversationnel sont intéressants du point de vue traitement en routine. D'abord, les plannings des deux machines sont indépendants l'un de l'autre. L'archivage des données est presque automatique parce que les bandes magnétiques utilisées sont lisibles par machines. Enfin, on peut accumuler une quantité arbitraire de données avant de commencer un traitement.

Mais il existe beaucoup de situations où le traitement de données en grands lots n'est pas utile. Souvent, la mise au point d'un nouveau système de collection de données demande un échantillon des résultats finaux pour faire les derniers réglages. Dans ce cas, la possibilité d'accumuler une grande quantité de données n'a pas grand intérêt.

Une connexion conversationnelle permet l'examen [illegible] de ces résultats préliminaires. En outre, ce type de connexion offre des possibilités intéressantes pour la recherche sur les systèmes couplés, en particulier les systèmes graphiques.

Mais il faut se rappeler que dans des conditions d'exploitation, les avantages d'une connexion conversationnelle sont obtenus avec les inconvénients suivants : le planning doit assurer la disponibilité des deux machines à la même heure, l'archivage, n'étant pas une partie intégrante du processus de transfert, doit être prévu, les systèmes qui tournent sur les deux machines doivent être très fiables malgré les complications de programmation qui peuvent arriver dans la conversation entre les machines. Le redémarrage du système après un arrêt intempestif d'une des machines constitue une de ces complications, et pose un problème qui est normalement très difficile à résoudre si l'on veut minimiser la perte du travail déjà accompli.

REALISATION DU PROJET

La mise en exploitation de la liaison à l'IMAG a été réalisé par un travail en équipe. Le projet se divisait naturellemant en quatre projets distincts.

Le premier de ces projets était la conception de l'interface entre les deux machines, et sa construction. Ce travail, essentiellement la construction et la mise au point d'un nouveau dispositif hardware, a été réalisé par Messieurs TUFFELLI et VUILLOD.

Le deuxième était l'écriture d'un assembleur PDP-8 qui tournerait sur le 360. Ceci a été fait comme projet d'élève par Messieurs BERGER et JAN

Pendant l'écriture de cet assembleur, le hardware de l'interface n'était pas encore défini, et la sortie du programme PDP-8 de l'assembleur était dans un format assez rudimentaire. Il a fallu plusieurs modifications à l'assembleur avant sa mise en exploitation. La façon de transférer des programmes au PDP-8 restait encore à définir. Ce travail, réalisé par l'auteur, fait partie de cette thèse.

Le quatrième projet, qui constitue le travail principal de l'auteur, a été l'écriture d'un programme permettant l'envoi de programmes du 360 au PDP-8. Puisque l'interface n'est pas un périphérique standard dans le système IBM, les entrées/sorties sur l'interface ont soulevé un problème délicat à cause de la présence de deux systèmes incompatibles en exploitation à l'IMAG : OS/MVT, et CP/CMS. Ce problème a été résolu en mettant toutes les opérations élémentaires d'entrée -sortie sur l'interface dans un module dont la séquence d'appel est compatible avec les deux systèmes. Cette méthode facilitera un éventuel changement de systèmes.

La séparation des fonctions d'assemblage et de transmissions vers le PDP-8 était impérative, vu la grande taille des programmes PDP-8 écrits à l'IMAG. Pour la mise au point de ces gros programmes, il est très utile de pouvoir faire des petites corrections en octal avant l'envoi au PDP-8. Pour cette raison, la communication entre l'assembleur et le programme de transfert se fait au moyen de paquets de cartes qu'un programmeur peut corriger facilement, pourvu qu'il ait la liste d'assemblage devant lui.

L'auteur est responsable aussi de la mise en exploitation du Disk Monitor System, un software offert par le fabriquant du PDP-8, Equipement Digital. Ce moniteur permet la sauvegarde de programmes PDP-8 sur bande magnétique DECtape. Il offre en outre un éditeur de texte (EDIT), un assembleur PDP-8 (PALD), un compilateur FORTRAN, et un langage conversationnel de calcul scientifique (FOCAL). Le moniteur est conversationnel et permet l'édition, l'assemblage et la mise au point de programmes PDP-8. Mais, vu la lenteur de l'impression d'une liste d'assemblage sur le PDP-8 (10 caractères par seconde), il est plus intéressant d'assembler les programmes PDP-8 sur le 360 s'ils dépassent 300 instructions. Le moniteur est capable d'assembler des programmes beaucoup plus gros, mais le temps d'assemblage devient prohibitif à cause de la lenteur des entrées-sorties.

SUPPORT SYSTEME POUR LE PROGRAMME DE TRANSFERT

Le programme de communication entre le PDP-8 et le 360 est nécessairement divisé en deux parties : une qui tourne sur le 360 et l'autre, écrite en conjonction avec la première, qui tourne sur le PDP-8. Or, puisqu'il fallait insérer la partie 360 dans un système déjà en exploitation, il était nécessaire de mettre cette partie sous un des systèmes disponibles sur le 67 à l'IMAG soit OS/MVT, soit CP/CMS.

La version d'O/S utilisée sur le 67 est MVT (Multiprogrammation avec un nombre Variable de Tâches). Il travaille en conjonction aves ASP (Attached Support Brocessor), un système sur le 40 qui gère les flots d'entrée et de sortie pour MVT, et qui choisit les travaux à passer sur le 67. Puisque O/S est un système de traitement par lots, il est excellent pour les travaux de routine tels que l'assemblage. Mais il n'a pas pu être utilisé pour la mise au point des programmes de connexion du PDP-8 à la 360.

D'abord, il fallait démontrer la fiabilité du hardware de la connexion dans des conditions d'exploitation. Aussi l'obligation de passer un job dans le flot d'entrée pour chaque essai aurait posé des inconvénients : il aurait fallu attendre son démarrage sous MVT, et le programme aurait alorsbloqué une région de MVT pendant son exécution, qui aurait pu inclure des opérations en pas à pas sur le PDP-8.

Vu la possibilité d'erreurs hardware les premiers programmes de transfert ont été réalisés pour un IBM 360 nu ; ceci permettait de distinguer les erreurs dues au hardware des erreurs de programmation. Mais le 67 coûte trop cher pour permettre beaucoup de mise au point de cette façon ; d'ailleurs, l'exploitation n'aurait pas été possible sans gêner les autres utilisateurs. Heureusement, on dispose du système à temps partagé appelé CP (Control Program) qui simule plusieurs machines nues pour des utilisateurs, travaillant d'une façon conversationnelle. Les machines nues simulées s'appellent des machines virtuelles ; chacun des utilisateurs possède une machine virtuelle. CP utilise le mécanisme de pagination, un dispositif du 360/67, pour simuler une mémoire virtuelle pour chaque machine virtuelle. Les machines virtuelles marchent réellement en état "problème"; si une instruction privilégiée est rencontrée, CP simule une interruption programme à la machine virtuelle si le PSW simulé de cette machine indiquait le mode "problème". Mai si le PSW simulé indiquait le mode superviseur, CP simule de déroulement de l'instruction privilégiée qui a provoqué l'interruption effective d'instruction privilégiée sur le 360/67.

CP simule aussi les entrées/sorties pour les machines virtuelles. Lorsqu'il se produit une interruption privilégiée sur une instruction d'entrée/sortie, CP se charge d'effectuer une opération réelle correspondante. Ainsi, une lecture sur un disque système se fait sur un disque réel partagé par toutes les machines virtuelles qui utilisent ce système ; une opération sur un disque propre à la machine se fait sur une portion de disque réel qui à la machine virtuelle. Un vrai disque peut alors contenir plusieurs disques virtuels privés.

Ainsi, chaque machine virtuelle peut avoir un mini-disque simulé de taille arbitraire : 5 cylindres à 200 cylindres.

Les entrées/sorties sur cartes et les sorties sur l'imprimante sont accumulées par CP sur des zones de disques réels qui sont allouées à la machine virtuelle par CP dans la mesure des besoins. Ceci permet à chaque machine virtuelle de faire les lectures de cartes et des impressions à son gré, sans bloquer le lecteur/perforateur réel ou l'imprimante réelle.

Les autres périphériques sont normalement attachés à une machine virtuelle par l'opérateur pendant une session de CP. Tel est le cas pour les dérouleurs de bandes (qui ne peuvent pas être partagés entre plusieurs utilisateurs), et l'adaptateur de données auquel est relié le PDP-8. Quand un périphérique est attaché à une machine virtuelle, CP exécute les opérations d'entrée/sortie d'une façon transparente sur le périphérique et transmet les conditions de fin d'opération à la machine virtuelle sans modification. Ce n'est pas le cas pour la lecture des cartes, par exemple, où il y a recouvrement d'erreur de lecture avant la mise de l'image de la carte sur le disque ; ainsi une machine virtuelle ne traite jamais une erreur de lecture de cartes. Il faut noter que la machine virtuelle travaille toujours en vrai mode "problème", pour permettre à CP d'intercepter, de contrôler, et de traduire toutes les opérations d'entrée/sortie, même celles qui se font sur un périphérique attaché.

Les entrées/sorties pour la console de la machine virtuelle se font sur la console de l'utilisateur. Ainsi, chaque utilisateur est l'opérateur de sa machine virtuelle.

La configuration virtuelle peut varier d'une machine à l'autre. Pour la plupart des utilisateurs de CP elle est la suivante : 360 standard avec 256 K de mémoire, lecteur/perforateur de cartes, imprimante, disque système, et disque privé. Sur le disque système, se trouve un système qui s'appelle CMS ou Cambridge Monitor System.

A l'aide de ce système, l'utilisateur peut ranger des fichiers sur le disque privé. Le disque système est protégé contre l'écriture. Cette précaution permet à plusieurs utilisateurs de CMS de partager inconsciemment, le même disque réel de système CMS. Pour les transferts PDP-360, il faut ajouter le PDA (Parallel Data Adapter) à cette configuration de base. CMS permet à l'opérateur, c'est-à-dire l'utilisateur de CP, d'éditer, d'assembler et d'exécuter des programmes d'une façon semi-conversationnelle. L'édition est conversationnelle mais l'assemblage et l'exécution ne le sont pas. Les programmes exécutés peuvent être logiquement indépendants de CMS, utilisant celui-ci uniquement pour l'assemblage et le chargement en mémoire virtuelle.

Le système CP/CMS a été choisi pour la mise au point des programmes de transfert à cause de son aspect conversationnel, et parce que CP permet de faire les entrées/sorties au niveau des instructions entrées/sorties sur une machine nue. Ceci est très important pour pouvoir connaître à fond les caractéristiques du hardware sans se soucier de la possibilité d'erreurs de programmation dans un système évolué tel que O/S.

Pour les programmes évolués, cependant, le système CMS est trop simple et n'offre pas assez de souplesse dans ses entrées/sorties. Par exemple, on ne peut pas commuter facilement les unités d'entrée/sortie. CMS est orienté aux opérations de fichiers sur le disque privé : les autres périphériques sont d'une importance secondaire. Si on veut assembler un programme qui existe sur bande magnétique, par exemple, il faut d'abord le copier sur le disque privé au lieu de lire le programme directement de la bande. De même, la liste d'assemblage est toujours mise sur le disque privé avant d'être transmise à l'imprimante virtuelle, opération qui exige un transfert préalable sur les disques réels. Les fichiers temporaires créés par l'assembleur sont aussi mis sur le disque privé parce que CP ne permet pas d'allouer dynamiquement des disques temporaires pour l'assemblage. Le fait de mettre tous les fichiers sur le même disque (virtuel ou réel) exige beaucoup de mouvements de bras de disque. Ceci augmente beaucoup le temps de réponse de CP/CMS. L'espace pris par les fichiers temporaires et la liste d'assemblage oblige chaque utilisateur de CMS de garder une importante portion de son disque privé vide. Cette pratique gaspille presque la moitié de la capacité des disques CP.

Puisque les opérations d'entrée/sortie sont essentiellement simulées sous CP, les conditions d'erreurs d'entrée/sortie ne sont pas toujours transmises à la machine virtuelle. Par exemple, si la machine virtuelle essaie de démarrer une opération sur un canal réel qui est occupé, CP ne reflète pas cette condition à la machine virtuelle. Au contraire, CP attend que le canal soit libre avant de démarrer l'opération réelle et de redonner le contrôle à la machine virtuelle. Ainsi, la machine virtuelle ne traite pas le cas du "canal déjà occupé".

CP ne permet pas, non plus, de vérifier la performance des entrées/sorties en temps réel. En effet, les entrées/sorties sous CP ne doivent pas contenir des dépendances de "timing" parce que le temps apparent d'une opération d'entrée/sortie ne correspond pas au temps réellement pris par cette opération. Sous CP il est donc difficile de mesurer la vitesse de transfert de données. Pour ces raisons, CP ne convient pas pour développer un système évolué qui traite les entrées/sorties de façon savante, car le recouvrement des conditions d'erreurs n'est pas complètement testé et l'effet d'un changement de programme sur la rapidité du système est difficile à mesurer. En somme, CP donnera les mêmes résultats qu'une machine nue pour un programme d'entrée/sortie écrit correctement, mais un programme d'entrée/sortie contenant des erreurs dans les procédés de recouvrement d'erreurs pourrait donner des résultats complètement différents. Il existe un certain nombre de conditions d'erreur qui pourraient être cachées ou modifiées par CP.

CHAPITRE II

ASSEMBLEUR PDP-8 SUR 360

MODIFICATION A L'ASSEMBLEUR PDP-8 SUR LE 360

L'assembleur de programmes PDP-8 sur le 360 a été écrit comme projet d'élève au printemps de 1969. Il était au point, mais il lui manquait plusieurs possibilités nécessaires à sa mise en exploitation. Un nombre important de modifications a été fait pour remédier aux défauts suivants :

1°) Les paquets de cartes "objets" produits par l'assembleur consistaient essentiellement d'un dump de l'image mémoire PDP-8 sur 80 colonnes, sans identification du programme sur la première carte, ni numéros de série.

2°) Le langage accepté par l'assembleur était incompatible avec à la fois le langage d'assemblage pour l'IBM 7044, défini par M. CLAUZEL, et le langage PAL D, fourni par le constructeur du PDP-8.

3°) Les entrées/sorties de l'assembleur étaient imbriquées dans l'assembleur lui-même, rendant difficile les modifications aux entrées/sorties sans toucher à l'assembleur lui-même.

4°) La moindre erreur d'assemblage empêchait la perforation d'un paquet de cartes pour transmission au PDP-8. Les instructions erronnées laissaient des zéros dans les cases mémoires PDP-8 correspondantes, plutôt que des "halt".

5°) La fonction de transmission vers le PDP-8 était imbriquée dans l'assembleur lui-même. Ceci rendait impossible la correction, avant la transmission au PDP-8, des erreurs d'assemblage.

La première modification consistait donc à réécrire la partie de l'assembleur qui perfore les cartes. Au lieu de faire tout simplement un dump en EBCDIC sur des cartes, l'assembleur perfore plusieurs types de cartes, et fournit des numéros de série dans les colonnes 73-80. Deux formats de cartes perforées sont prévus. Le format TEXT (pas encore implémenté) est celui des modules objets pour les programmes 360. Ce format permet alors la manipulation de programmes PDP-8

sur le 360 ; un programme PDP-8 en format TEXT peut être mis en mémoire 360 au du chargement d'un programme 360 uqi communiquerait avec lui. Cette technique est utile pour les programmes de systèmes couplés conversationnellement parce que les modifications d'un de ces programmes provoquent souvent des modifications correspondantes sur l'autre machine. Il faut souligner que le programme PDP-8, même en format TEXT, doit être envoyé au PDP-8 pour l'exécution ; l'exécution de ce programme sur le 360, quoiqu'elleest permise par le système 360, donnera des résultats imprévisibles.

Le format OCTAL comprend plusieurs types de cartes. Le type de la carte est indiqué dans les colonnes de 1-4 ; la première colonne contient toujours un point. Le type de carte.PRG annonce le début d'un nouveau programme PDP-8 et fournit l'identification du programme. Le type.FIE indique dans lequel des champs PDP-8 (0,1 ou 2) on voudra ranger les données qui suivent. Le type.OCT indique une adresse PDP-8 en octal, suivie de 1 à 16 contenus de mémoire PDP-8. Ces contenus sont rangés consécutivement en mémoire PDP-8 à partir de l'adresse indiquée. Le type.END indique la fin d'un programme et déclenche la transmission vers le PDP-8. La perfcration de cartes OCTAL est facile à faire à la main beaucoup plus facile que dans le cas des cartes TEXT. C'est pour cela que ce format a été choisi pour être implémenté en premier, car il permet de faire facilement des corrections en octal (des "patches") aux programmes PDP-8, en retenant l'histoire des changements. Les cartes .HLT et .CLR, perforées par le programmeur et mises après la carte .PRG, permettent le remplissage de la mémoire PDP-8 avec des instructions d'arrêt immédiat (HLT), ou des zéros. Le remplissage par des HLT rend plus facile la mise au point de programmes PDP-8 qui pourraient faire un branchement au-delà des limites du programme.

La deuxième modification était de modifier le langage d'assemblage PDP-8 tel que définit par Berger. Lors de l'écriture de l'assembleur, il existait déjà une importante bibliothèque de programmes PDP-8. Les plus importants de ces programmes étaient les programmes graphiques LAGROL et GENIAL de M. Lucas, le système ALGOL 62 pour le PDP-8, écrit par M. Clauzel, et le système SPARTACUS, écrit par Boxsembaum et Guiboud-Ribaud. Avant la mise en exploitation du IBM 360 à l'IMAG, on utilisait un IBM 7044 pour l'assemblage de programmes PDP-8.

L' assembleur de programmes PDP-8 avait été réalisé sur cette machine par M. Clauzel ; le langage d'assemblage ressemblait de très près au langage MACRO 8 défini par le fabricant du PDP-8. Puisque ce langage était perforé en champs fixes sur cartes, tandis que MACRO 8 est écrit en format libre, M. Clauzel a supprimé les artifices de ponctuation du MACRO 8 qui permettent de reconnaître les champs. Dans le MACRO 8, où on peut mettre des caractères blancs à volonté entre les éléments du langage, il faut une virgule après l'étiquette pour la distinguer du code opération, et un slash (/) pour signaler le début d'un commentaire. Sur les bandes perforées du PDP-8, ces conventions ne posent pas de problème parce que les bandes se prêtent facilement à un format libre. Mais sur cartes, ces marques de ponctuation sont nuisibles parce qu'elles ne servent à rien, et parce que les programmeurs déjà habitués aux assembleurs à format fixe les oublient très souvent.

Pendant la transition entre le 7044 et l'IBM 360, on a utilisé le Moniteur pour l'écriture et la mise au point de programmes PDP-8. Le moniteur a donc servi d'intermédiaire jusqu'à la mise en service du programme de transfert PDP-8 vers 360. Une deuxième bibliothèque de programmes PDP-8 a ainsi été créée. Pour le moniteur, on utilise le langage PALD, qui est semblable au MACRO 8, mais qui fournit ni la possibilité de macros (qui sont peu utiles pour le PDP-8, et coûteux en mémoire PDP-8 lors de l'assemblage), ni de pseudo-opérations de virgule flottante. Aussi, il ne vérifie pas la validité des combinaisons de microinstructions. Mais il offre la génération automatique de liens d'une page à une autre, et la possibilité de littéraux en fin de page ou en page zéro.

Or, le langage d'assemblage défini par M. Berger ressemblait plutôt au langage d'assembleur 360, et il était incompatible avec l'assembleur du 7044 et le PALD. Heureusement, il était possible de le modifier pour accepter la partie commune entre le PALD et le langage de M. Clauzel. Le langage de Berger a, en effet subit un élargissement qui lui permet d'assembler à la fois les programmes assemblés sur le 7044, et les programmes en PALD (qui devront être transmis au 360, et perforés sur cartes). Ceci offre la possibilité de mettre au point des parties de programmes sous le moniteur avec son assembleur conversationnel, avant de mettre une version définitive sur les fichiers 360. Les petites modifications aux gros programmes pourront donc être testées au PDP-8 avant

l'incorporation définitive dans le gros programme. Cette possibilité devient très utile pour les programmes PDP-8 où l'addition de petits sous-programmes en langage machine est prévu. Un bon exemple de ce type d'opération est l'addition de fonctions d'animation au système graphique GENIAL. La compatibilité avec le PALD est importante aussi pour la modification et l'assemblage de programmes PDP-8 fournis par le fabricant ; le moniteur fournit un bon exemple.

Le langage de l'assembleur 360, tel que redéfini, est essentiellement PAL-D, mais les deux fautes les plus communes sont "pardonnées". Ces fautes sont les suivantes : soit oublier une virgule pour délimiter une étiquette, soit oublier un slash '/' pour indiquer un commentaire. En revanche, le programmeur doit respecter les conventions suivantes : d'abord, les étiquettes doivent commencer en colonne 1, et être suivies immédiatement par une virgule ou un blanc. En effet, la virgule ne sert plus pour reconnaître l'étiquette, mais elle est tolérée pour la compatibilité avec le PAL-D, qui l'exige. Deuxièmement, les microinstructions combinées (e-g-CLA SNA RTR etc...) doivent être séparées d'un blanc seulement. Les commentaires de ces instructions commencent après deux blancs, ou 1 blanc suivi d'un slash. Troisièmement, pour les autres instructions, le slash n'est pas nécessaire pour commencer le commentaire comme un PAL-D, parce que pour celles-ci la fin de l'instruction est reconnue par contexte (e.g. une instruction de référence mémoire est terminée sur l'adresse). Si on oublie le slash n PAL-D, le commentaire suivant est traité comme partie de l'adresse. Ces conventions ne sont pas gênantes dans la pratique, puisque les programmes PALD sont normalement tapés en colonnes. Voici des exemples pour montrer les différences entre le PALD et l'assembleur 360.
Le - représente un blanc, la notation $\overset{*}{=}$ indique une chaîne d'un nombre arbitraire de blancs, y inclure la chaîne vide.

	INSTRUCTION A ASSEMBLER	INTERPRETATION		COMPATIBILITE
		PALD	ASSEMBLEUR 360	
1	JOJO,*TAD--*Y-	E = JOJO I = TAD A = Y	E = JOJO I = TAD A = Y	OUI
2	JOJO-*TAD--*Y-	I = JOJO A = TAD|Y	E = JOJO I = TAD A = Y	NT
3	-*JOJO,*TAD--*Y-	E = JOJO I = TAD A = Y	DC (ERREUR DANS EA)	N
4	-*JOJO-*TAD--*Y-	I = JOJO A = TAD|Y	DC VALEUR = JOJO	N
5	CLA--*CLL	CLA|CLL	E = CLA I = CLL	N
6	CLA,*CLL	ERREUR (ETIQUETTE DEJA DEFINIE)	E = CLA I = CLL	N
7	CLA--*,	ERREUR (MAUVAISE ETIQUETTE)	ERREUR DANS EA	-
8	-*CLA-CLL-	I + CLA|CLL	I = CLA|CLL	OUI
9	-*CLA--*CLL-	I = CLA|CLL	I = CLA C = CLL	N
10	JOJO-*CLA-CLL-	I = JOJO A = CLA|CLL	E = JOJO I = CLA|CLL	NT
11	JOJO-*CLA--*CLL	I = JOJO A = CLA|CLL	E = JOJO I = CLA C = CLL	N

12	JOJO,$\overset{*}{=}$CLA-CLL	E = JOJO I = CLA| CLL	E = JOJO I = CLA|CLL	OUI
13	JOJO,$\overset{*}{=}$TAD-$\overset{*}{=}$Y-$\overset{*}{=}$COM	E = JOJO I = TAD A = Y|COM	E = JOJO I = TAD A = Y C = COM	NT
14	JOJO,$\overset{*}{=}$TAD-$\overset{*}{=}$Y-$\overset{*}{=}$/COM	E = JOJO I = TAD A = Y C = COM	E = JOJO I = TAD A = Y C = COM	OUI
15	JOJO,$\overset{*}{=}$TAD-$\overset{*}{=}$Y/COM	E = JOJO I = TAD A = Y C = COM	E = JOJO I = TAD A = Y/COM	N
16	JOJO-$\overset{*}{=}$TAD-$\overset{*}{=}$Y-$\overset{*}{=}$COM	I = JOJO A = TAD| Y|COM	E = JOJO I = TAD A = Y C = COM	NT

- = blanc $\overset{*}{=}$ = chaîne de blancs (peut être vide)

E = étiquette

DC = définition de constante

N = non

I = instruction

A = adresse

C = commentaire

EA = expression arithmétique

NT= toléré par l'assembleur PDP-8 sur 360.

| = réunion logique.

Les exemples de lignes (1), (8), (12) et (14) montrent la façon recommandée d'écrire les énoncés d'assemblage PDP-8. Le même énoncé donne le même résultat, quoiqu'on assemble sur le PDP-8 ou sur le 360.

Les exemples (2), (10), (13) et (16) sont des énoncés qui donnent des erreurs en PALD, mais qui sont acceptés par le 360. Ils montrent le "pardon" des erreurs fréquentes d'orthographe. En (2) et (10) on a oublié la virgule après l'étiquette. Le PALD essaie d'interpréter JOJO comme partie de l'instruction s'il est défini ; seul le 360 assemble l'instruction comme le programmeur voulait. En (13), il manque un slash avant le commentaire ; PALD tente un ou inclusif entre les mots du commentaire et l'adresse. En (16), on a omis la virgule après l'étiquette et le slash avant le commentaire. Ce format est interprété comme une instruction très compliquée en PALD. Le 360 fait l'assemblage de ce format correctement : chose très importante car tous les programmes de la bibliothèque 7044 sont perforés dans ce format-ci.

La ligne (3) montre une étiquette qui ne commence pas dans la colonne 1 ; ceci est acceptable au PALD, mais donne une erreur d'assemblage sur le 360. La virgule a été retenue comme signe arithmétique de réunion logique pour la compatibilité avec l'assembleur de M. Clauzel. La distinction entre la virgule comme opérateur, et comme fin d'étiquette, se fait par contexte.

L'énoncé (4) est traité comme l'instruction en (2) par PALD, parce que les blancs devant l'énoncé n'ont aucune importance. Sur le 360, le blanc en colonne 1 indique l'absence d'une étiquette, JOJO devient une expression arithmétique, et le reste est pris comme un commentaire.

L'énoncé (5) est assemblé correctement par PALD, mais le 360 interprète CLA comme étiquette. L'utilisation d'instructions PDP-8 comme étiquettes sur le 360 est possible en certains cas, mais elle est fortement déconseillée.

L'énoncé (6) est assemblé comme l'énoncé (5) sur le 360 parce que la virgule n'a pas de signification. Sur le PDP-8, la tentative de redéfinir CLA donne une erreur.

L'énoncé (7) donne des erreurs d'assemblage sur les deux assembleurs. Sur le PDP-8, on tente de redéfinir CLA. Sur le 360, la virgule constitue une expression arithmétique composé d'un opérateur binaire tout seul ; ceci constitue une erreur.

Les énoncés (9) et (11) sont assemblés correctement sur le PDP-8, mais seulement le CLA est assemblé sur le 360. Un piège ! On ne peut pas tolérer les espaces multiples entre les microinstructions d'un OPR sans exiger un slash avant le commentaire. Ce slash manque dans les programmes PDP-8 assemblés sur le 7044.

L'énoncé (15) est un autre piège parce qu'on utilise le slash pour la division uniquement sur le 360. Si COM est défini comme symbole, la division aura lieu.

Les possibilités suivantes, qui existent en PALD, ne sont pas implémentées sur l'assembleur 360 :

1) Séparation des énoncés par ;

2) Adressage de la page zéro par l'opérateur Z (ceci est d'ailleurs inutile)

3) L'opération EXPUNGE (la table de symboles permanent contient d'une façon inséparable des pseudo-opérations, sur le 360).

4) Les instructions affectées à une étiquette par le signe = (e.g. JOJO = JMS I TOTO). Cette incompatibilité sera enlevée plus tard.

Les possibilités suivantes, implémentées sur ASSPDP, l'assembleur sur 360, ne sont pas disponibles avec PALD :

1) Contrôle de listage (sauf par XLIST)

2) La table de la table des symboles est limitée sur le PDP-8, et il n'y a pas de table de références croisées.

Les instructions CDF et CIF sont codées différemment. Pour accéder au champ 2 en PALD, on doit écrire CDF 20 (ou CIF 20), parce que PALD fait la réunion logique, entre 6201 (CDF) et 20. Le numéro du champ se trouve dans les bits 6-8 de l'instruction. Sur l'assembleur 360, on écrit CDF 2, pour la même instruction. CDF 20 est rejeté (numéro de champ trop grand), mais ASSPDP sera modifié pour l'accepter.

N.B. CDF 2 sur le PDP-8 est accepté par PALD, mais donne une instruction autre que CDF !

En PALD, le slash (/) est toujours pris comme début de commentaire. En ASSPDP, le slash, s'il se trouve immédiatement à droite d'un terme d'une expression arithmétique, sans blancs pour séparer, est pris comme signe de division.

e.g.TAD Y/COM donne TAD Y en PALD, mais une division de Y par COM en ASSPDP.

Mais TAD Y/COM donne le même résultat sur les deux assembleurs.

En ASSPDP, l'étoile (*) peut servir comme signe de multiplication dans une expression arithmétique.

En PALD, l'étoile est prise toujours comme modification du compteur d'emplacement donnant des résultats bizarres, si on essaie de s'en servir comme signe de multiplication dans une expression arithmétique.

e.g.TAD X * 3, quand X = 100, donne TAD 300 en ASSPDP mais donne TAD Y, puis un changement du compteur d'emplacement à 3, en PALD.

En PALD, on utilise $ pour terminer un programme. En ASSPDP, on se sert plutôt de END pour pouvoir indiquer une adresse de départ.

Le but des modifications de l'assembleur ASSPDP a été de conserver l'esprit du langage PALD, mais de lui enlever ses principaux défauts, sans perdre la compatibilité avec les programmes déjà écrits en PALD. En effet, la conversion la plus facile est dans le sens PALD vers le 360 parce que le PALD est plus restreint et offre moins d'outils de mise au point pour les programmes de grande taille.

GRAMMAIRE FORMELLE DE L'ASSEMBLEUR

Le langage accepté par l'assembleur PDP-8 sera maintenant décrit au moyen d'une grammaire formelle de Backus. Les conditions de contexte sont données avec les règles de la grammaire. Les différences entre ce langage et le PALD sont signalées au fur et à mesure.

Les symboles suivants ont des significations spéciales :

Λ	chaîne vide
FDC	caractère fictif de fin de carte (retour chariot sur le PDP-8), précédé éventuellement de blancs.
<chaîne de majuscule>	chaîne de caractères figurant sur la carte
<->	caractère blanc
<->*	chaîne de blancs éventuellement vide
<sansFC>	suite de caractères quelconques, FDC non-inclus.

<Programme> := <début><suite d'énoncés ou vide><fin>

<début> := <options><titre>

<options> := Λ|<.OPTIONS><virbl><suite d'options><-><sans FC><FDC>

<suite d'options> := Λ|<option><viropt>

<viropt> := Λ|<virgule><option><viropt>

<option> := <DECK>|<NODECK>|<REF>|<NOREF>|<LIST>|<NOLIST>

<virbl> := <virgule>|<->

<titre> := Λ|<.TITRE><virbl><sans FC><FDC>

Les énoncés de début .OPTIONS et .TITRE n'existent pas en PALD

<fin> := <><->*<END><-><expression arithmétique><sans FC><FDC>

<suite d'énoncés ou vide> := Λ|<énoncé><FDC><suite d'énoncés ou vide>
<énoncé> := <énoncé-commentaire>|<Modifce>|<énoncé obl.étiqueté>
|<énoncé obl.non-étiqueté>
|<énoncé opt.étiqueté>
<énoncé-commentaire> := <->*<slash><sans FC>
<Modifce> := <->*<étoile><->*<expression arithmétique><sans FC>

Les symboles dans l'expression arithmétique doivent être définis avant la modification du compteur d'emplacement, sur les deux assembleurs.

<énoncé obl.étiqueté> := <énoncé EQU>
<énoncé EQU> := <étiquette><->*<Indication EQU><->*<expression arithmétique>
<sans FC>
<Indication EQU> := <-><EQU><->|<signe =>

La pseudo-opération EQU a été introduite par Berger, mais seul le signe = est compatible avec le PALD. En PALD, l'ordre des EQU est importante car tous les symboles d'une expression arithmétique doivent déjà être définis. L'assembleur sur le 360 ramasse tous les EQU et les traite à la fin du premier passage ; ainsi, les EQU peuvent paraître dans un ordre quelconque. Dans l'implémentation courante sur le 360, les instructions comme valeur d'un EQU ne sont pas autorisées (e.g. JUMP = JMS I TOTO).

<énoncé obl. non-étiq.> := <énoncé racine>|<énoncé de listage>
|<énoncé de champ/page>
<énoncé racine> := <-><->*<Racine><-><sans FC>
<Racine> := <DECIMA>|<OCTAL>

Le PALD accepte aussi DECIMAL parce qu'il tronque toutes les étiquettes à 6 caractères. Sur le 360, une étiquette de 7 caractères est illégal.

<énoncé de listage> := <-><->*<contrôle de listage><-><sans FC>|<espacement>
<contrôle de listage> := <EJECT>|<LIST>|<UNLIST>|<XLIST>
<espacement> := <-><->*<SPACE><-><chif012>
<énoncé de champ/page> := <-><->*<champage><-><chif012>
<champage> := <FIELD>|<PAGE>
<chif012> := <->*|<->*<chiffre><-><sans FC>|<->*<chiffre><chiffre><-><sans FC>
|<->*<slash><sans FC>
<énoncé opt étiqueté> := <énoncé étiqueté>|<énoncé non-étiqueté>
<énoncé étiqueté> := <Partie étiquette><->*<énoncé sans étiquette>
<Partie étiquette> := <étiquette><->|<étiquette><virgule>
<énoncé non-étiqueté> := <-><->*<énoncé sans étiquette>

En PALD, ces 2 règles sont remplacées par :

<Partie étiquette> := <->*<étiquette><virgule>
<énoncé non étiquette> := <->*<énoncé sans étiquette>

Ces règles permettent un énoncé en colonne 1, ou en étiquette dans une colonne quelconque. La virgule après l'étiquette sert donc à séparer l'étiquette de l'énoncé.

<énoncé sans étiquette> := <Instruction IOT>
|<Instruction MRI>
|<Instruction OPR>
|<Instruction CDFCIF>
|<Instruction TEXT>
|<Definition de constante>

<Instruction IOT> := <Mnemo IOT><-><sans FC>

La combinaison des IOT par ou inclusif n'est pas possible sur le 360 ; ce problème est contourné en définissant les mnémotechniques pour les ordres combinés.

<Mnemo IOT> := <ADSC>|<ADCV>|<ADSC>|...

Les productions de <Mnemo IOT> représentent tous les ordres d'entrée-sortie du PDP-8.

<Instruction MRI> := <Mnemo MRI><->*<Indirect ou vide><Partie adresse>
<Mnemo MRI> := <AND>|<TAD>|<ISZ>|<DCA>|<JMS>|<JMP>
<Indirect ou vide> := Λ|<I><->*|<étoile><->*

L'étoile a été introduite par Berger, mais seul le I est reconnu par PALD. L'indication de page zéro (Z) n'est pas implémentée sur le 360.

<Partie adresse> := <Option de FE><Instr. ou expr. arith.><sans FC>

En PALD, il est obligatoire d'intercaler un slash avant le commentaire.

<Option de FE> := Λ|<Indic. de FE><->*
<Indic de FE> := <Indic. FE page courante>|<Indic FE page 0>
<Indic FE page courante> := <Parenthèse gauche>|<signe =>

Seule la parenthèse gauche (est autorisée en PALD ; le = a été introduit par Berger

<Indic FE page 0> := <Parenthèse gauche carrée>|<Parenthèse droite>|<signe dièze>

La parenthèse gauche carrée (est utilisée en PALD. La parenthèse droite) est utilisée sur les machines IBM parce que la parenthèse gauche carrée n'est pas un caractère standard sur les machines IBM. Le dièze(#) a été introduit par Berger.

<Instr ou expr. arith.> := <Instr. comme FE>|<expression arithmétique>
<Instr comme FE> := <Indication IFE> <Ins FE>
<Indication IFE> := Λ|<I :><->*

Le I: n'est pas utilisé en PALD pour indiquer une instruction comme facteur effectif.

<Ins FE> :=<Ins FEIOT>|<Ins FEMRI>|<Ins FEOPR>|<Ins FECDFCIF>

Seulement certaines instructions sont admises comme facteurs effectifs

<Ins FEIOT> := <Mnemo IOT><->
<Ins FEMRI> := <Mnemo MRI><-><->*<Indirect ou vide><expression arithmétique>

Le 360 ne permet pas d'empiler les instructions comme facteurs effectifs. Ainsi, l'instruction TAD (TAD (ISZ X)) est admis en PALD, mais par sur le 360. Cette restriction n'est pas gênante pour les programmes pratiques.

<Ins FEOPR> := <oprcons>
<Ins FECDFCIF> := <cons CDFCIF>
<instruction OPR> := <oprcons><sans FC>
<oprcons> := <suite OPR><->|<suite OPR><slash>

En PALD, seulement le slash indique le début d'un commentaire.Pour les programmes assemblés sur le 7044, le double blanc est utilisé comme délimiteur.

<suite OPR> := <groupe 1>|<groupe 2>
<groupe 1> := <nop>|<raz comp. set><decintvide>|<decint>|<combin>
<nop> := <NOP><->
<raz comp set> := <raz comp set AC><raz comp set L>|<raz comp set AC>
|<raz comp set L>
<raz comp set AC> := <raz comp AC>|<set AC>
<raz comp AC> := <CLA><->|<CMA><->|<CLA><-><CMA><->
<set AC> := <STA><->
<raz comp set L> := <raz comp L>|<set L>
<raz comp L>:= <CLL><->|<CML><->|<CLL><-><CML><->
<set L> := <STL><->
<decintvide> := Λ|<decint>
<decint> := <dec gauche>|<dec droite>|<increment>
<dec gauche> := <RAL><->|<RTL><->
<dec droite> := <RAR><->|<RTR><->
<increment> := <IAC><->
<combin> := <GLK><->|<CIA><->|<raz comp set L><CIA><->
<groupe 2> := <test><specvide>|<test><CLA><-><specvide>|<spec>|<CLA><-><esrhlt>

<test> := <testpas>|<testneg>

<testpos> := <SZA><->|<SMA><->|<SNL><->|<SZA><-><SMA><->
|<SZA><-><SNL><->|<SZA><-><SMA><-><SNL><->

<testneg> := <SNA><->|<SPA><->|<SZL><->|<SNA><-><SPA><->
|<SNA><-> <SZL><->|<SNA><-><SPA><-><SZL><->

<specvide> := Λ|<spec>

<spec> := <LAS><->|<LAS><-><HLT>|<osrhlt>

<osrhlt> := <OSR><->|<HLT><->|<OST><-><HLT><->

Cette partie de la grammaire génère les microinstructions d'un oprcons dans l'ordre logique d'exécution d'opérations multiples. Mais, en PALD, et sur le 360, l'ordre des microinstructions est sans importance ; on peut écrire CLL RTL ou bien RTL CLL pour générer la même instruction. La compatibilité des microinstructions est assurée par ces règles ; si le programmeur code des microinstructions incompatibles, le PALD du PDP-8 n'indique pas d'erreur, mais l'assembleur sur le 360 sort un diagnostic.

<Instruction CDFCIF> := <cons CDFCIF><sans FC>

<cons CDFCIF> := <Mnemo CDFCIF><-><chif 012>

<Mnemo CDFCIF> := <CDF>|<CIF>

Les sémantiques de CDF et de CIF sont différents en PALD, et sur le 360. En langage machine, le champ voulu est indiqué dans les bits 6, 7, 8 de l'instruction. Le PALD fait le ou inclusif entre le mnémotechnique et le chiffre ; ainsi, il faut écrire CDF 20 (et non pas CDF 2) pour spécifier le champ 2 par exemple. L'utilisation de CDF 2 en PALD donne des résultats faux, mais cette forme est exigée sur le 360. Une correction future à l'assembleur 360 lui permettra d'accepter la forme CDF 20. Avec l'implémentation courante, la spécification d'un champ plus grand que 2 donne une erreur.

<Instruction TEXT> := <Mnemo TEXT><-><->*<chif 12><virbl><sans FC>

<Mnemo TEXT> := <TEXT>

<Chif 12> := <chiffre>|<chiffre><chiffre>

Le <sans FC> doit contenir au moins le nombre de caractères spécifié comme texte.En PALD, il n'y a pas de compte de caractères ; on utilise deux caractères non-blancs identiques pour délimiter le texte e.g. TEXT QJOJOQ, tandis que le même énoncé serait codé comme TEXT 4JCJO sur le 360.

<Définition de constante> := <Indic. def. const.><expression arithmétique>
<sans FC>

<Indic.def.const.> := Λ|<DC><->*

En PALD, <Indic def.const.> est toujours nulle. Sur le 360, la pseudo-opération DC a été introduite par Berger.

<expression arithmétique> := <expression><->
<expression> := <terme unitaire>|<expression><plusmoins><terme>
<terme unitaire> := <option unitaire><terme>
<option unitaire> := Λ|<plusmoins>
<terme> := <facteur>|<terme><foisdiv><facteur>
<facteur> := <ovande>|<facteur><ou logique><ovande>
<ovante> := <ovande>|<ovande><et logique><etande>
<etande> := <expression simple>
<expression simple> := <élément>|<parenthèse gauche><expression><parenthèse droite>
<élément> := <étiquette>|<point>|<facteur binaire>|<nombre>

En PALD, il n'y a pas de multiplication ou de division. La possibilité de division sur le 360 rend nécessaire un blanc avant un commentaire qui suit l'expression arithmétique ; aucun blanc n'est permis dans l'expression elle-même. Si un slash suit immédiatement l'expression, il est pris comme signe de division. La division par zéro est permise, et donne zéro comme résultat.

<plus-moins> := + -
<foisdiv> := * /
<ou logique> := | ,
<et logique> := & $

L'interprétation du dollar et de la virgule comme opérateurs logiques ne font pas partie du PALD, mais elle permet la compatibilité avec l'assembleur de Clauzel sur le 7044.

Si une <partie adresse> commence avec une parenthèse gauche, la première parenthèse gauche est prise comme indication de facteur effectif. Ainsi, on ne peut pas écrire TAD(300) au lieu de TAD 300.

<étiquette> := <lettre>|<lettre><letchif>|<lettre><letchif><letchif>|...
|<lettre><letchif><letchif><letchif><letchif><letchif>

En PALD, les étiquettes ayant plus de 6 caractères sont tronquées à 6 i.e. seulement les 6 premiers caractères sont pris en compte. Sur le 360, une étiquette trop longue donne un diagnostic d'erreur.

<point> := .
<facteur binaire> := <B:><douzebits>
<douzebits> := <bit>|<bit><bit>|....|<bit><bit><bit><bit><bit><bit><bit><bit><bit>
<bit><bit><bit>
<nombre> := <chiffre>|<chiffre><nombre>
<lettre> := A|B|C|...|Z
<chiffre> := 0|1|2|...|9
<bit> := 0|1

Les entrées-sorties de l'assembleur ont été séparées de l'assembleur lui-même afin de permettre son adaptation facile à plusieurs systèmes : CP/CMS ou bien O/S, par exemple. Il faut distinguer deux genres de programmes PDP-8 a assembleur : les grands programmes écrits comme projets de recherche, et les travaux pratiques des étudiants qui apprennent la programmation du PDP-8. Ces deux types exigent des performances différentes des entrées-sorties de l'assembleur.

Les grands programmes de recherche occupent plusieurs champs mémoire du PDP-8. Voici quelques exemples :

SPARTACUS	5 module de 4K instructions chacun
ALGOL	1 module de 8K instructions
LAGROL	1 module de 8K instructions

Par "module", on veut dire une section d'un programme qu'on peut assembler indépendamment des autres. La taille de ces programmes exige d'abord un encombrement de mémoire centrale assez importante pour la table des symboles et les références croisées pendant l'assemblage. Aussi faut-il un fichier intermédiaire pour stocker une copie du fichier d'entrée pour le deuxième passage, car la mémoire centrale est trop petite.

Les travaux pratiques sont très petits par contre, mais ils posent un problème spécial parce qu'ils sont nombreux. Dans une session, on voudra assembler 10 ou 20 T.P. à la fois. Chaque T.P. doit être identifié d'une façon unique afin de permettre son repérage. Ainsi, il devient possible d'assembler plusiers programmes ensemble dans un seul "job" O/S. Pendant l'assemblage de chacun de ces petits programmes, l'encombrement mémoire est très faible. Supposons qu'un T.P. occupe 4 pages PDP-8--- soit 600 instructions source. Il pourrait y avoir 100 symboles environ, et 500 entrées dans la table des références croisées. L'encombrement mémoire pour un tel programme serait calculé comme suit :

600 cartes fois 105 octets par carte	63,000 octets
100 symboles fois 20 octets par symbole	2,000 octets
500 références fois 8 octets par référence	4,000 octets
TOTAL	69,000 octets

Il est évident que l'encombrement principal vient du fichier intermédiaire, car chaque "record" inclut non seulement la carte de 80 colonnes perforée par le programmeur, mais aussi des informations supplémentaires ajoutées par l'assembleur pendant le premier passage. Pour un T.P. on peut quand même garder ce fichier en mémoire centrale, car la taille normale d'un programme est de 100K octets. A l'encombrement des tables déjà calculé, il faut ajouter celui de l'assembleur lui-même (moins de 12 K), et des zones tampons d'entrée-sortie (moins de 5K). La suppression d'un support externe pour le fichier intermédiaire permet un gain important lors de l'assemblage de petits programmes.

Le problème des entrées-sorties de l'assembleur est rendu compliqué aussi par la présence de deux systèmes incompatibles sur le 360/67 à l'IMAG : OS/MVT et CP/CMS. Chaque système a des avantages et des inconvénients pour l'utilisateur.

Le système O/S est préféré pour le traitement de plusieurs programmes PDP-8 à assembler. Il permet à plusieurs utilisateurs de corriger des programmes et d'en éliminer les erreurs d'assemblage sans encombrer des consoles conversationnelles. Pour la mise au point d'un programme, cependant, il est plutôt mal commode parce que la correction de fichiers se fait par un procédé ee mise à jour qui est non seulement lourd, mais aussi non-conversationnel.

Le système CP/CMS permet la correction de fichiers d'une façon conversationnelle, mais ce luxe ne devient rentable que lors de la mise au point d'un programme PDP-8. La disponibilité faible de consoles CP/CMS impose des limites sévères sur le nombre d'utilisateurs qui peuvent corriger des programmes à la fois. La correction de programmes ne peut se faire, d'ailleurs, que lorsque le système CP/CMS lui-même est en marche ;il n'est pas possible de préparer les corrections sur cartes avant le début de la session.

L'allocation statique des disques sur CP/CMS constitue un des plus grands désavantages pour l'utilisateur de ce système. Un problème énorme d'encombrement de disques se pose parce que tous les fichiers créés par l'utilisateur, y compris les fichiers temporaires qui sont détruits après la session, sont rangés dans une zone de disque qui est propre à lui ; il n'y a pas de disque banal pour les fichiers temporaires. Pour cette raison, ce système n'est pas rentable pour un grand nombre d'utilisateurs ayant chacun des petits fichiers, car l'espace des fichiers temporaires éventuels doit être affecté à chaque utilisateur, même s'il ne s'en sert pas tout le temps. Lorsque l'assembleur tourne, par exemple, il génère un fichier intermédiaire (pour être relu au deuxième passage et ensuite détruit à la fin de l'assemblage), un fichier de liste qui est destinée à être sorti sur l'imprimante rapide et ensuite détruit, et un fichier d'images de cartes contenant le programme PDP-8 assemblé en forme octal.

Le choix de système à utiliser dépend des besoins spécifiques de l'utilisateur. Le système O/S permet à un assistant de passer une série de travaux pratiques et d'en recevoir toutes les listes ensemble. Ainsi l'assistant peut contrôler le travail de chaque élève, et, si besoin est, de donner des explications plus amples a ceux qui ont des difficultés. Les erreurs grossières de logique sont relativement simples à repérer sur la liste d'assemblage avant l'envoi au PDP-8 pour l'exécution. Enfin, le passage en moniteur O/S par l'assistant libère l'étudiant des soucis de formalités de système qui n'ont pas une relation directe avec la programmation du PDP-8 : cartes de contrôle O/S compliquées, ou bien le démarrage et fin de session CP/CMS.

Le système CP/CMS convient plutôt à l'utilisateur expérimenté en train de réaliser un projet spécifique. Ce genre d'utilisateur peut tirer bénérice des possibilités d'édition par contexte et de réassemblage immédiat sans se noyer dans les détails de l'opération de sa console. Un programmeur expérimenté commet beaucoup moins de fautes d'assemblage qu'un élève. Pour lui, la possibilité d'exécution presque immédiatement après avoir fait des corrections est très importante, car il peut encore se rappeler de ce qu'il vient de faire.

Ces besoins différents ont rendu nécessaire la séparation de l'assembleur lui-même de ses entrées-sorties. Il devient alors possible de changer les procédures d'entrée-sortie sans toucher à l'assembleur lui-même. Cinq (5) modules d'entrée-sortie ont été prévus pour l'assembleur. Ils diffèrent selon leur méthodes d'accéder aux fichiers. Les fichiers s'appellent : SYSIN (entrée), SYSPRINT (listage), SYSPUNCH (sortie perforée), et SYSUT1 (intermédiaire). Les 5 modules sont :

ASSPIOOS — utilise QSAM (macros GET/PUT) pour tous les fichiers

ASSPIOOM — utilise QSAM pour SYSIN, SYSPRINT, et SYSPUNCH, mais garde en mémoire le fichier SYSUT1.

Ces modules sont destinés à l'exploitation sous O/S. Sous CMS, les entrées-sorties de QSAM se font sur les fichiers qui s'appellent FILE SYSIN, FILE SYSPUNCH, etc.

ASSPIOCS — utilise RDBUF/WRBUF (macros de disque CMS) pour tous les fichiers.

ASSPIOCM — RDBUF/WRBUF pour SYSIN, SYSPRINT, et SYSPUNCH, garde SYSYT1 en mémoire

ASSPIOCB — SYSIN sur le lecteur virtuel, SYSPRINT sur l'imprimante virtuelle, SYSPUNCH sur le perforateur virtuel, SYSUT1 sur disque. Pour assembler en lots sous CP/CMS.

De ces modules, ASSPIOOS est déjà en exploitation, car il convient à tous les besoins actuels, même s'il n'est pas aussi performant qu'un module d'entrée-sortie qui garderait le fichier intermédiaire en mémoire.

Plusieurs changements mineurs ont été fait sur l'assembleur pour rendre son utilisation plus commode. D'abord, la liste de sortie indique la version de l'assembleur. Les valeurs des facteurs effectifs ont été mises tout près des instructions générées afin d'améliorer la lisibilité d'un programme assemblé. Une entête en haut de la page s'aligne avec les colonnes préférées des cartes contenant les instructions à assembler.

Les instructions erronées remplissent la mémoire PDP-8 correspondante avec une instruction d'arrêt, plutôt que de laisser cette case pleine de zéros. Ceci réduit le risque de destruction de programme pendant la mise au point.

Certaines parties de l'assembleur, notamment la génération de facteurs effectifs, ont été nettoyées et commentées.

Enfin, la séparation des fonctions d'assemblage et de transfert au PDP-8 a rendu possible la mise au point séparée de chacune. Aussi, il a été possible d'implémenter chaque fonction sur le système qui lui convenait mieux. Pour la plupart des utilisateurs, le système O/S est le plus intéressant pour faire des assemblages, qu'il s'agisse de gros programmes ou de T.P. Mais c'est le système CP/CMS qui est le préféré pour le transfert au PDP-8.

CHAPITRE III

PROGRAMME DE TRANSFERT

OPERATIONS ENTREE-SORTIE SUR LE 360

Sur le 360 les opérations d'entrée-sortie se font indépendamment de l'unité centrale au moyen de canaux. A chaque canal, on peut relier plusieurs périphériques du même genre. Une unité de contrôle sert à gérer des dérouleurs de bandes; une autre sera reliée à des disques, une troisième contrôlera les périphériques de record unitaires (lecteur perforateur de cartes et imprimante) etc... Dans le cas de l'unité d'adaptation de données auquelle est relié le PDP-8, il existe logiquement une unité de contrôle pour chaque périphérique permettant à chacun de ces périphériques de fonctionner indépendamment des autres. Ce n'est pas le cas pour des périphériques comme des bandes ou des disques : une seule bande peut transférer des données à la fois parce que les données doivent obligatoirement passer par l'unique unité de contrôle.

Les canaux peuvent fonctionner en de deux modes : mode simple et mode multiple. Le mode simple établit un chemin pour les données qui part de la mémoire centrale et qui passent par le canal, une unité de contrôle et se termine dans un périphérique. Les opérations qui ne touchent pas à ce chemin (e.g. opérations mécaniques telle que alimentation de cartes, rebobinage de bandes ou mouvement de bras de disque) peuvent s'effectuer sur les autres périphériques pendant un transfert simple. Le mode multiple permet à un canal de faire des lectures et des écritures sur plusieurs unités de contrôle à la fois. Chaque unité de contrôle est affecté à un sous-canal multiple ; chaque sous-canal agit comme un canal simple. Les entrées/sorties en mode multiple sont plus lentes que les entrées/sorties simples parce que le canal doit s'occuper de plusieurs périphériques à la fois. Ceci ne pose aucun inconvénient pour les périphériques lents comme les terminaux, et les périphériques ou le timing n'est pas trop important. Les périphériques rapides, comme les bandes ou les disques font leurs transferts en mode simple à cause du taux d'arrivée des données. Le mode simple est toujours utilisé sur un canal simple. Sur un canal multiple , il est utilisé si l'unité de contrôle le demande.

Avant de démarrer une opération d'entrée-sortie, le programme de l'unité centrale du 360 doit d'abord mettre en place un ou plusiers mots de commandes de canal (CCW). Ensuite, il doit mettre dans le mot de mémoire centrale situé à l'adresse fixe 72 décimal, l'adresse mémoire du premier CCW. Ensuite, il doit masquer les interruptions entrées/sorties et exécuter l'instruction SIO. C'est dans l'exécution de l'instruction SIO elle-même qu'on indique l'adresse du périphérique concerné. Après le SIO, un code condition de 0 dans le CSW indique que l'opération a démarré correctement. Un code condition de 3 indique (sous CP) que le PDA n'est pas attaché à la configuration virtuelle ; un code condition de 2 indique que le canal, étant occupé, n'a pas pu démarrer l'opération ; un code condition de 1 indique que l'opération n'a pas pu démarrer à cause d'une condition d'erreur indiquée dans le mot d'état du canal.

Lorsqu'on exécute une instruction d'entrée/sortie, ou lorsqu'il se produit une interruption d'entrée/sortie, ce mot d'état peut-être transféré du canal en mémoire à une adresse fixe. L'emplacement de ce stockage, appelé le mot d'état du canal (CSW), est le double mot qui se trouve à l'adresse 64-71 décimal. Selon les cas, soit le CSW entier est changé, soit seulement la partie destinée à contenir le mot d'état du canal.

La fin de l'opération d'entrée/sortie demande une interruption entrée/sortie à l'unité centrale. Quand l'interruption est acceptée, les conditions de la fin de l'opération sont mises dans le CSW de la mémoire centrale, et il y a déroutement du programme en cours vers le superviseur. Le superviseur du système range ce CSW stocké dans une zone propre à chacue périphérique avant de permettre d'autres interruptions d'entrée-sortie et de reprendre le programme en cours.

Chaque CCW est composé de 4 parties : la commande, l'adresse de début de zone mémoire, les drapeaux et le nombre d'octets à transmettre. Les commandes pour le PDA sont : lecture et écriture avec ou sans laps de temps, non-opération, et sondage (sense). Le sondage permet de lire un octet de sondage du PDA ; cet octet reflète les conditions intérieures au PDA et donne des informations plus détaillées que celles qui sont stockées dans le CSW. Les drapeaux permettent plusieurs modifications aux opérations de base. Deux de ces drapeaux sont utilisés pour le programme de transfert. Le premier, qui déclenche le "chaînage de

données", permet de concaténer deux zones non-contigues de mémoire centrale dans un seul bloc sur le périphérique. Le deuxième drapeau, qui provoque le "chaînage de commandes", permet au canal d'accomplir plusieurs opérations à la suite avec une seule interruption sur erreur, ou sur la fin de la dernière commande.

Puisqu'on utilise les mêmes emplacements de mémoire pour toutes les opérations sur tous les canaux, il est très important de masquer les interruptions entrées/sorties avant d'exécuter une instruction d'entrée/sortie. De même, lorsqu'une interruption d'entrée-sortie s'est produite, les autres interruptions doivent être masquées jusquà ce que l'interruption courante ait été traitée.

En plus des interruptions d'entrée/sortie causées par une fin d'opération, il y a aussi celles qui ne sont pas dues au programme de l'unité centrale. Il s'agit alors d'interruptions dites asynchrones. Le PDP-8 peut en déclencher une quand il envoie le signal Interruption au BDA. Les interruptions asynchrones sont très utiles pour synchroniser deux machines sans en mettre une dans une boucle d'attente.

PROGRAMMATION DE L'INTERFACE 360-PDP-8

La communication entre le 360 et le PDP-8 se fait au moyen du Parallel Data Adapter 2701 (PDA) qui relie le canal multiple du 360 et une interface sur le PDP-8.

Cette interface transforme les caractéristiques électroniques des signaux du standard IBM au standard DEC et vice-versa. Elle fournit, d'ailleurs, les instructions PDP-8 permettant à celui-ci de communiquer avec le PDA.

Le PDA installé à l'IMAG est un dispositif fourni par IBM qui permet de lire ou d'écrire des mots de 16 bits : seulement 12 de ces bits sont utilisés car la longueur des mots PDP-8 est de 12 bits. En plus de ces 12 signaux de données, il faut en plus des signaux des contrôles entre le PDA et l'interface PDP-8. Les signaux que le PDA peut envoyer vers l'interface PDP-8 sont les suivants : Les 12 bits de données en sortie et les signaux de contrôle, qui s'appellent Sélecté Ecriture, Prêt Ecriture, Sélecté Lecture, Prêt Lecture et Compte Zéro. Les signaux que l'interface PDP-8 peut envoyer au PDA sont les 12 bits de données en entrée et les signaux de contrôle : Demande, Fin de Record, Fin de Fichier, Erreur Parite Ecriture, Supprimer Indication Erreur Parite Lecture, et Interruption. Le signal Erreur Parite Ecriture n'est pas implémenté à l'IMAG, il permettrait au PDP-8 de signaler au 360 s'il a détecté une erreur de parité dans les données qu'il vient de recevoir du 360. Le signal Supprimer Indication Erreur Parite Lecture a toujours la valeur vraie, ce qui empêche le PDA de contrôler la parité de ce qu'il a reçu du PDP-8. L'absence des contrôles de parité par hardware rend nécessaire la vérification des données par programme. Le rôle des autres signaux apparaîtra dans les descriptions suivantes des opérations de transfert. Il est à noter que la terminologie de "lecture" ou "écriture" est orientée au 360 ; ainsi une opération dite d'écriture veut dire que le PDP-8 lit des données du 360 ; une opération dite de lecture implique que le PDP-8 envoie des données vers le 360. Cette terminologie peut être confuse à vue du PDP-8.

L'interface du PDP-8 permet au programme PDP-8 de manipuler presque directement les signaux du PDA. Les mots de données entrent et sortent un à la fois par l'accumulateur du PDP-8. Le programme PDP-8 est responsable de la lecture/écriture des données, la gestion des signaux de contrôle du PDA, et les tests de fin de transfert : le hardware de l'interface transmet directement les signaux entre le PDP-8 et le PDA.

Nous allons maintenant décrire les opérations de transfert et ensuite aborder le problème de démarrage et d'arrêt de ces opérations. Il y a deux types de transfert : PDP-8 vers 360, appelé lecture, et 360 vers PDP-8, appelé écriture. Les transferts se font sur un système de demande-réponde entre les deux machines, ce qui évite la dépendance sur le timing. (En fait, cette règle n'est pas respectée par le 2701 à la fin d'un transfert, comme nous verrons plus loin). Mais tout système de demande/réponse est sujet à un autre problème, celui de l'opération non-terminée à cause de la non-réponse d'une des machines. Cette situation pourrait être provoquée par plusieurs causes : une des machines étant en panne ou pas prête, une défaillance dans la connexion, une intervention opérateur sur une des machines, ou des erreurs de programmation.

Quand une des machines cesse de répondre pendant un transfert, il faut que l'autre soit avertie de la situation. A cette fin le PDA est muni d'un dispositif de laps de temps ; si, lorsque le PDA signale qu'il est prêt à transmettre ou à recevoir un mot de données, le PDP-8 ne répond pas en moins de 2 secondes, le PDA termine l'opération avec indication d'erreur sur le 360. Ceci débloque le canal 360 et permet au programme 360 de prendre une décision concernant l'erreur. Il est à noter que le laps de temps dépend uniquement du PDA, et non pas de l'activité de l'unité centrale. Le transfert marche donc sous CP comme sur une machine nue.

Dans le programme de transfert à l'IMAG, l'occurrence d'un laps de temps est considérée comme une erreur. Donc, le 360 ne doit pas lancer une opération d'entrée-sortie sur le PDA sans s'assurer que le PDP-8 pourra envoyer ou recevoir des données. Le PDP-8 signale qu'il est prêt en envoyant le signal Interruption au 360.

Si le signal Interruption arrive pendant qu'une opération est en cours, le PDA termine l'opération immédiatement avec l'indication erreur Commande Interrompue. Dans la convention du programme de transfert, l'Interruption sert d'indication au 360 que le PDP-8 est prêt pour une nouvelle opération. L'arrivée de Interruption en cours d'une opération d'entrée-sortie constitue, donc, lui aussi une erreur. Le laps de temps, en conjonction avec l'Interruption, permet au 360 de contrôler les interventions manuelles sur le PDP-8 au milieu d'un transfert, ainsi que certaines erreurs de programmation.

Les transferts peuvent se suivre à la que-leu-leu (sans synchronisation par Interruption) pourvu que le PDP-8 soit prêt pour le transfert suivant en 2 secondes. Mais, si le PDP-8 soit faire une opération de durée imprédictible avec les données reçues (sauvegarde d'un programme sur DECtape, ou frappe d'un message), il faut qu'il signale la fin de cette opération en signalant Interruption pour amorcer le transfert suivant. Avant de lancer ce nouveau transfert, le 360 doit contrôler que l'Interruption a été bien reçue et qu'elle n'a pas interrompue un transfert (erreur catastrophique de programmation).

TRANSMISSION VERS LE PDP-8

Pour envouer des données eu PDP-8, il faut que le 360 s'assure que le PDP-8 est prêt à les recevoir. Ensuite, il lance une opération d'écriture sur le sous-canal du PDA. A ce moment, les signaux de Sélecté Ecriture et Prêt Ecriture sont envoués au PDP-8, et le premier mot de données est présenté sur les lignes de Données en Ecriture. Le niveau logique du signal Sélecté Ecriture reste élevé pendant toute l'opération de l'écriture, mais le programme du PDP-8 est insensible à ce signal. Le signal Prêt Ecriture signifie qu'un mot valide se trouve sur les lignes de données. Le PDP-8 doit vérifier ce signal avant d'échantilloner les lignes de données en les lisant dans son accumulateur. Le PDA est insensible à ce processus d'échantillonnage. Le PDP-8 indique qu'il a reçu le mot de données en envoyant le signal Demande au PDA. Le PDA sur réception de Demande, fait tomber Prêt Ecriture jusqu'à ce qu'il y ait un nouveau mot à transmettre au PDP-8. A ce moment-là, le PDA envoie de nouveau Prêt Ecriture au PDP-8, et le PDP-8 peut lire un autre mot de données.

Si le PDP-8 envoie le signal Fin de Record ou Fin de Fichier au lieu de Demande pendant le transfert, le 360 reçoit une indication de longueur incorrecte sur le canal du PDA. Ceci indique au 360 que le PDP-8 n'a pas reçu tous les mots que le 360 tentait d'envoyer. En plus, le PDA génère un signal de Transfert de Données Incomplet (TDI) pour le 360. Ceci se produit parce que le PDA, après la Demande qui précédait la Fin de Record, aurait eu le temps de demander un autre mot du canal 360. Comme ce dernier mot a été refusé par le PDP-8, le compte résiduel stocké par le canal est invalide ; le signal TDI indique ce fait. Seul le 360 peut lire l'indication de TDI.

Après que le PDA^{ait} envoyé Prêt Ecriture pour le dernier mot, et que le PDP-8 a répondu avec un Demande, le PDP-8 doit suivre ce Demande immédiatement par un Fin de Record. Le temps qui sépare le Demande et le Fin de Record doit être moins que 5 microsecondes ; ceci ne permet pas au PDP-8 de savoir si le 360 avait encore des mots à lui envoyer. Seul le 360 aura une indication d'erreur un compte résiduel non-zéro et/ou un TDI. Si le signal Fin de Record n'est pas envoyé dans le délai de 5 microsecondes, le PDA génère un mot de zéros sur les lignes de données et présente ce mot avec à la fois Prêt Ecriture et Compte Zéro.

Le Compte Zéro indique au PDP-8 que le mot de données à été généré par le PDA, et ne vient pas du canal du 360, qui n'aviat rien de plus à envoyer. Si le PDP-8 envoie Fin de Record sur la première arrivée de Compte Zéro, le PDA et le canal 360 devraient terminer l'opération sans indiquer une erreur au 360. En fait il se produit une fausse indication de longueur incorrecte sur le canal 360. Si le PDP-8 répond à Prêt Ecriture et Compte Zéro avec Demande, le PDA continue à présenter des mots zéros au PDP-8 jusqu'à ce que ce dernier envoie Fin de Record ou Fin de fichier.

TRANSMISSION DU PDP-8

L'envoi de données vers le 360 (lecture) se fait d'une façon semblable à l'opération d'écriture. Le 360, sachant d'avance que le PDP-8 est prêt à transmettre, démarre une opération de lecture sur le PDA. Le PDA envoie à son tour les signaux Sélecté Lecture et Prêt Lecture au PDP-8. La ligne Sélecté Lecture reste en état affirmatif pendant la durée de l'opération de lecture, mais n'influence pas le déroulement du programme PDP-8. La ligne Prêt Lecture se lève chaque fois que le PDA est prêt à recevoir un autre mot de données. Une fois que le PDP-8 a contrôlé la présence de Prêt Lecture, il met le mot à transmettre au PDA sur les lignes de Données en Entrée. Ensuite, il lance le signal Demande pour indiquer au PDA la validité du mot qui se trouve sur les lignes de données. Sur réception de Demande, le PDA fait tomber Prêt Lecture jusqu'à ce qu'il ait pu échantillonner les lignes de données et transmettre le mot au canal du 360.

Si le PDP-8 envoie Fin de Record ou Fin de Fichier au lieu de Demande, le 360 reçoit une indication de longueur incorrecte avec le compte de mots restants à transmettre. L'opération se termine normalement sur le 360 si le PDP-8 envoie un nombre de Demandes égal au compte de mots indiqué dans le CCW de lecture sur le 360. Mais le PDP-8 ne peut pas savoir si le compte résiduel est nul sur le 360 en contrôlant la ligne Compte Zéro.

Si le PDP-8 envoie une Demande de trop, le canal 360 termine l'opération en signalant Longueur Incorrecte. Après cette demande supplémentaire, le PDA envoie le signal Compte Zéro au PDP-8. Le PDP-8 peut détecter ce signal, mais la présence de Compte Zéro en lecture constitue une erreur parce que le canal du 360 a déjà terminé l'opération. Il faut noter qu'un signal Fin de Fichier ou Fin de Record est nécessaire pour terminer l'opération sur le PDA.

PROBLEMES SPECIAUX DUS AU TEMPS DE REPONSE DU PDA

La programmation de la correction PDP-8/360 est rendue compliquée par deux situations où le "timing", c'est-à-dire le temps de réponse du PDA, a un effet critique sur les progrès des opérations de transfert. La première situation se produit lors de la terminaison normale d'un transfert ; la deuxième a lieu quand le PDP-8 signale Interruption "très tôt" après une opération de transfert réussie.

On a déjà vu que, pendant une opération d'écriture, le PDA présente un nouveau mot au PDP-8 au début du transfert, et sur chaque Demande suivant. Pour écrire n mots, le PDP-8 doit envoyer Demande n fois pour indiquer qu'il les a reçus. Mais que se passe-t-il après le n-ième Demande ? Supposons d'abord que le PDP-8 n'envoie rien après la n-ième Demande. Alors le PDA, ne sachant pas que l'écriture est finie, demande à son tour encore un mot du canal 360 pour l'envoyer au PDP-8. Or puisque le compte de mots coté 360 est épuisé, le canal génère une indication de longueur incorrecte puisque le périphérique a demandé trop de mots. Le PDA présente alors le signal Compte Zéro au PDP-8 et un mot fictif qui contient des zéros comme données. Si, maintenant, le PDP-8 envoie Fin de Record ou Fin de Fichier, l'opération d'écriture se termine sur le 360 avec une indication de longueur incorrecte au lieu de se terminer normalement. Vue du 360, la siutation est indistinctible de celle où le PDP-8 aurait accepté des mots zéros générés par le PDA. En effet, une opération d'écriture devrait se terminer normalement si et seulement si le signal Fin de Record est reçu par le PDA quand ce dernier venait de présenter Compte Zéro au PDP-8 pour la première fois.

Il y a une autre façon, plus brutale, de terminer une opération d'écriture. Elle consiste à envoyer Fin de Record ou Fin de Fichier tout de suite après le n-ième Demande. Par "tout de suite", on veut dire un délai qui est si court qu'il ne laisse pas au PDA le temps de demander un nouveau mot au canal 360 et ainsi de présenter Compte Zéro au PDP-8. Alors l'opération se termine d'une façon normale sur le 360, sans indication de longueur incorrecte. Cependant, le PDP-8 ne peut pas vérifier la présence du signal Compte Zéro avant l'envoi du signal Fin de Record. Il doit plutôt lancer Fin de Record "dant le vide", sans pouvoir savoir, en testant le signal Compte Zéro, si le 360 avait encore des données à lui transmettre.

On se trouve alors face à un dilemme. Si le PDP-8 prend le temps de vérifier qu'il a tout reçu en contrôlant la ligne Compte Zéro après la dernière Demande, il provoque une situation d'incertitude sur le 360, à savoir que le PDP-8 aurait pu demander trop de mots et aurait alors pris comme données valides des mots zéros générés par le PDA. Autrement, si le PDP-8 envoie Fin de Record dans un temps T après le dernier Demande, sans contrôler Compte Zéro, l'écriture se termine correctement sur le 360, mais il en résulte une incertitude du côté PDP-8 : le 360 aurait pu avoir plus de mots à envoyer au PDP-8.

Il est à noter que le contrôle de Compte Zéro est une opération propre au PDP-8 ; le PDA (et alors le canal 360) ne peut pas savoir si ce contrôle a été effectué. Autrement dit, le PDA ne réagit pas à l'instruction PDP-8 IFSC (test de Compte Zéro, qui n'envoie aucun signal au PDA), mais seulement aux instructions IFDEM (envoi de Demande), IFEOR (envoie de Fin de Record), et IFEOF (envoi de Fin de Fichier).

Quel temps T peut-on permettre entre le dernier Demande et le signal Fin de Record ? Par expérience, on a trouvé que l'écriture se termine correctement si le PDP-8 exécute le dernier IFDEM suivi immédiatement de IFEOR, sans autres instructions intercalées. Dans cette situation, T vaut 3.75 microsecondes, le temps entre deux instructions d'entrée-sortie sur le PDP-8. Mais si l'on s'amuse à intercaler un seul NOP (durée 1.5 microsecondes) entre le dernier IFDEM et le IFEOR, il y a l'indication de longueur incorrecte sur le 360. C'est par cette méthode assez primitive qu'on déduit que $T < 5$ microsecondes.

Comme indiqué ci-dessus, la deuxième situation où le "timing" du PDA devient critique se produit quand on se sert du signal Interruption pour indiquer au 360 que le PDP-8 est prêt pour un autre transfert après la réussite du précédent. On se rappelle que, si Interruption était arrivé pendant le transfert précédent, il se serait produit une erreur de Commande Interrompue. Mais, dans un cas normal, l'Interruption arrive après la Fin de Record du transfert précédent. Sur le 360, on s'attendrait à deux interruptions d'entrée-sortie : une déclenchée par la réception de Fin de Record, et l'autre déclenchée par Interruptio

Or, si Interruption arrive "peu de temps" après Fin de Record, c'est-à-dire avant que l'interruption pour Fin de Record ait pu avoir lieu, les deux interruptions sont superposées et il ne se produit qu'une seule interruption d'entrée-sortie sur le 360. Dans ce cas, le mot d'état (CSW) stocké par le canal 360 contient les indications de fin de record, y compris le compte résiduel, et aussi une indication que le signal Interruption a été reçu. Bien que le CSW ne porte pas d'indication de contrôle unité ("unit check"), le programme 360 doit quand même faire une opération de sondage ("sense") pour vérifier que le transfert précédent s'est déroulé correctement.

Heureusement, le programme 360 peut traiter cette situation assez simplement. Après avoir contrôlé que le transfert précédent s'est achevé correctement, le programme 360 met en place un drapeau de non-attente pour indiquer que Interruption a déjà été reçu. Ensuite, l'indication d'Interruption est enlevé de l'octet d'état (CSW), et le programme 360 continue à se dérouler sans se rendre compte de la réception de Interruption. Mais, lorsque la logique du programme utilisant les sous-programmes de transfert exige une attente du signal Interruption, le drapeau est testé et remis à zéro. Si le drapeau contenait 0 déjà, on sait que Interruption n'a pas été reçu avec la Fin de Record précédent, et qu'on doit attendre la deuxième interruption d'entrée-sortie qui indiquera la réception du signal Interruption. Par contre, si le drapeau contenait 1, on sait que le signal Interruption a déjà été reçu, et que la deuxième interruption n'aura pas lieu. Il faut donc rendre immédiatement le contrôle au programme appelant qui avait demandé l'attente.

Cette deuxième situation se produit dans les transferts pour lesquels le temps de réponse du PDP-8 est imprédictible. Par exemple, si le 360 envoie un champ de programme PDP-8, il faut normalement le sauvegarder sur DECtape. Quand cette opération, qui pourrait durer une ou deux minutes au pire, a été effectuée, le PDP-8 envoie Interruption pour signaler qu'il est prêt pour un autre transfert. Le 360 ne peut pas démarrer l'opération suivante s'il ne sait pas si le PDP-8 est prêt. Dans ce cas, deux interruptions se produiront : une pour la fin du transfert, et l'autre, une ou deux minutes plus tard, pour la réception de Interruption.

Mais supposons que, plutôt que de sauvegarder le champ mémoire immédiatement, on désire en recevoir un autre. Avec le Disk Monitor, par exemple, il est plus intéressant de recevoir les trois champs avant de commencer la sauvegarde ou éventuellement une exécution d'essai. Dans ce cas, il peut arriver que le PDP-8 envoie Interruption rapidement après la Fin de Record, de telle sorte que le programme 360 soit obligé de traiter le cas de deux interruptions superposées.

La même chose peut se produire si le 360 a envoyé un texte ASCII au PDP-8 destiné à être frappé par le PDP-8. Si le texte contient un seul caractère, le temps de frappe est de quelques centaines de microsecondes, le temps de mettre le caractère dans le buffer de sortie du télétype du PDP-8. Mais si le texte contient une ligne entière, le temps de frappe sera de l'ordre de plusieurs secondes. Il y a donc une immense variation possible dans le temps de réponse du PDP-8 ; le programme 360 doit quand même être indépendant du temps de réponse du PDP-8.

DESCRIPTION DETAILLE DU PROGRAMME DE TRANSFERT

Cette description constitue la documentation nécessaire à la maintenance et aux corrections éventuelles du programme de transfert de programmes PDP-8 assemblés sur le 360. Il est fortement conseillé au lecteur de consulter le mode d'emploi avant de commencer l'étude détaillée du programme de transfert.

Le programme de transfert est divisé en deux programmes, un destiné au 360 et l'autre au PDP-8. Le programme 360 tourne sous CMS ; il pourrait donc marcher sur une machine nue si besoin était. Une console CP/CMS a été installée dans la salle PDP-8. A part des utilisations CP/CMS normales, elle a la fonction de permettre le contrôle du 360 (virtuel) par l'utilisateur des programmes de communication PDP-8/360 sans déranger l'opération du 360 réel.

Le programme 360 accomplit trois fonctions principales : la lecture d'un paquet de cartes en format OCTAL et la génération de l'image mémoire du PDP-8, la préparation et le formattage des records nécessaires au transfert, et l'envoi de ces records au PDP-8. Ce programme génère les sommes logiques envoyées avec chaque record pour que le PDP-8 puisse vérifier la validité des données reçues.

Le programme PDP-8 a été écrit comme esclave de celui du 360. Ce programme doit reconnaître les différents types de records envoyés par le 360. Chaque type indique au PDP-8 la signification du record transmis. Le type "texte" indique au PDP-8 un message en code ASCII que le PDP-8 doit frapper sur sa console. Au moyen de ce type de record, le 360 peut envoyer des lignes de texte pour identifier le programme qu'il envoie, ainsi que l'occupation mémoire et l'adresse de départ. Ceci permet à l'opérateur du PDP-8 de sauvegarder correctement un programme sur DECtape avec le Moniteur. Le type de record "programme" annonce au PDP-8 que le 360 envoie une partie de programme PDP-8 en forme binaire. L'action du PDP-8 consiste à stocker cette information dans sa mémoire. Enfin, le type "fin de fichier" indique au PDP-8 que la transmission d'un programme est complète, et que le PDP-8 peut procéder à la sauvegarde ou à une exécution d'essai du programme reçu.

LECTURE DES CARTES OCTAL (MODULE TRAN)

Le programme de transfert sur le 360 est divisé en trois modules. Le premier, TRAN, lit les images de cartes en entrées et génère l'image binaire de la mémoire PDP-8. Il imprime des diagnostics sur le terminal CP/CMS s'il détecte des erreurs dans les cartes. Il appelle WPROG pour accomplir le transfert lui-même, et ensuite passe au paquet de cartes OCTAL suivant. Une condition de fin de fichier sur le fichier d'entrée provoque un retour à CMS. Ce module contient aussi un aiguillage qui permet l'envoi d'une nouvelle version du programme PDP-8 au PDP-8. L'utilisateur normal n'est pas concerné par la présence de cet aiguillage.

FORMAT DES CARTES D'ENTREE POUR LE PROGRAMME DE TRANSFERT

Les cartes de données pour le programme de transfert contiennent un point "." dans la première colonne, suivi de 3 lettres pour identifier le type de la carte. Sauf pour la carte .END, les colonnes 73 à 80 contiennent un numéro de séquence. La carte .END termine le paquet OCTAL pour un programme. L'absence de numéro de série sur cette carte permet de séparer facilement les programmes individuels d'un assemblage multiple. Voici maintenant les différents types de carte avec la fonction de chacun.

Carte .PRG

Cols 1-4	.PRG
6-50	Titre du programme
57-58	Date d'assemblage MM/JJ/AA
61-72	Tableau d'occupation mémoire PDP-8.

Le titre du programme est tiré de la carte .TITRE lue par l'assembleur. Ce titre figure aussi en tête de chaque page de la liste d'assemblage. Il y a assez de place pour 35 caractères environ sur la carte .PRG ; ainsi un paquet octal est facile à identifier. La date d'assemblage, en format /mois/jour/année, permet de retrouver la version la plus récente d'un programme. Le tableau d'occupation mémoire est booléen, avec un bit par page. Ce tableau est perforé en caractères EBCEIC, avec 8 bits par caractère perforé. Le tableau entier pour trois champs PDP-8 contient 3 × 32 bits, à raison d'un bit d'occupation par page ; il faut donc 12 colonnes de carte pour contenir le tableau d'occupation.

Carte .CLR

Cols 1-4	.CLR

Cette carte remet toute l'image de la mémoire PDP-8 à zéro. Les 3 dernières pages du champ 2 sont conservées, car elles contiennent une copie du programme de transfert. Cette carte est à fournir par le programmeur.

Carte .HLT

Cols 1-4 .HLT

Cette carte met une instruction HLT (7402 octal) dans toute l'image mémoire du PDP-8, sauf les 3 dernières pages. Il est très utile, pendant la mise au point d'un programme PDP-8, de remplir la mémoire libre avec des HLT ; si le programme fait un branchement vers une partie de la mémoire non réservée à lui, il tombera probablement sur une instruction HLT qui arrêtera l'exécution immédiatement. Cette carte est fournie par le programmeur.

Carte .FIE

Cols 1-4 .FIE
6 Numéro du champ PDP-8 (0 à 2 en numérique)

Cette carte indique le numéro de champ PDP-8 où il faudra ranger les données à suivre. Ces données sont perforées sur les cartes .OCT qui suivent.

Carte .OCT

Cols. 1-4 .COT
5-8 Adresse de début de rangement
9-72 1 à 6 mots PDP-8 en octal.

Les mots sur cette carte sont rangés en ordre consécutif à partir de l'adresse de début spécifiée, jusqu'à la rencontre d'un blanc ou la colonne 73. Chaque mot occupe 4 colonnes, et les zéros de poids forts sont obligatoirement présents. Le champ PDP-8 où on range les données est 0 par défaut après une carte .PRG; il est changé par les cartes .FIE.

Carte .END

Cols 1-4 .END

Cette carte indique la fin physique d'un programme PDP-8, et provoque le transfert de l'image mémoire PDP-8 accumulée.

Les cartes .PRG, .FIE, .OCT, et .END sont perforées par l'assembleur PDP-8 ou le programmeur ; le programmeur doit insérer lui-même des cartes .CLR ou .HLT s'il le désire.

Le format des cartes OCTAL a été choisi pour permettre une correction aisée de gros programmes PDP-8 par des "patches", c'est-à-dire, des modifications en octal à certaines mémoires PDP-8 avant de faire un transfert. Les patches permettent de sauver un gros assemblage avec deux ou trois petites erreurs du type fréquemment commises : JMS au lieu de JMP, instruction IOT ou expression arithmétique mal épelée, ou oubli de définir une mémoire de manoeuvre. La technique décrite ci-dessous permet de faire des patches d'une façon plus propre que quelques autres qui viennent à l'esprit : corrections aux clés du PDP-8 (source d'erreur, et ne laisse aucune trace), corrections avec ODT (ODT occupe trois pages du PDP-8, ce qui est gênant pour les très gros programmes), ou corrections directement sur les cartes .OCT (ceci ne laisse pas de trace, et ne permet pas de récupérer le programme original).

Supposons que, après une soigneuse examination de la liste d'assemblage, on ait découvert que les patches suivants sont nécessaires avant le prochain essai :

dans le champ 0, il faut mettre 2426 dans la case 3417
3642 dans la case 2005
2423 dans la case 2006
1242 dans la case 2007
542 dans la case 46

dans le champ 2, il faut mettre 4227 dans la case 603
5 dans la case 604

Pour ces corrections, le "paquet de correction" suivant est perforé par le programmeur :

```
.FIE 0                          Champ zéro
.OCT34172426                    Première correction
.OCT2005364224231242            3 mémoires consécutives
.OCT00460542                    Remarquer 0's dans poids fort
.FIE 2                          Champ deux
.OCT060342270005                Correction du champ deux.
```

Il faut noter que les cartes .FIE ont un blanc entre le .FIE et le numéro de champ. Les cartes .OCT n'ont pas de blanc, sauf pour indiquer le début du commentaire. Les nombres octaux contiennent des zéros, le cas échéant, dans les positions de poids fort ; chaque zone de la carte .OCT a une longueur fixe de 4 colonnes.

Le paquet de cartes permettant d'envoyer le programme corrigé au PDP-8 est constitué comme suit :

```
.PRG                            Perforée par l'assembleur
.HLT ou .CLR                    Si désirée
.FIE ou .OCT                    Perforée par l'assembleur
.Paquet de correction
.END
```

Les cartes du paquet de correction peuvent contenir des commentaires pour documenter chacun des patches. Le programme original est reconstitué si on enlève le paquet de correction ; une correction fausse peut être annulé en retirant la carte correspondante.

Malgré l'aise apparente de corriger des programmes en octal, il ne faut pas oublier de corriger le programme source à mesure, et de refaire un assemblage propre aussitôt que possible. Après le réassemblage, il est important de tester le programme dans les cas qui ont provoqué les corrections, afin de vérifier que les changements au symbolique ont produit un programme équivalent à l'ancien avec ses "patches". En somme, la technique de patches permet de gagner du temps pour la correction d'une erreur triviale, et de corriger le résultat d'un assemblage si, par hasard, une erreur obscure dans l'assembleur provoquait une génération de code fausse.

TRANSFERT AU PDP-8 (Module WPROG)

Le module principal (TRAN) appelle un deuxième module, WPROG pour préparer les records à envoyer au PDA. Pour chaque opération élémentaire d'entrée-sortie, il appelle le module de fonctions de base, PDAIO, qui sera décrit plus tard.

WPROG envoie 5 records au PDP-8 au cours d'un transfert. D'abord, il appelle une fonction interne, TEXTPDP, pour envoyer une copie de carte .PRG au PDP-8. Ensuite, il envoie trois records au PDP-8 qui contiennent chacun le contenu d'un champ PDP-8 au moyen du sous-programme interne WPDP. Enfin, il envoie un record au PDP-8 pour indiquer que le transfert est complet.

Le sous-programme TEXTPDP est appelé avec le registre 1 contenant l'adresse de début d'un texte, et le registre 2 indiquant le nombre de caractères à transmettre. Le sous-programme convertit les caractères en EBCDIC en code ASCII, avec chaque caractère cadré dans les 8 bits du milieu d'un demi-mot 360. Ce cadrage à gauche est nécessaire parce que la liaison PDP-8/360 transmet les 12 bits de gauche d'un demi-mot 360. Les codes de retour chariot et avance papier sont effectivement intercalés devant le texte, et le sous-programme interne WPDP est appelé pour faire la transmission.

Le sous-programme WPDP sert à envoyer un bloc de données au PDP-8. Les paramètres d'appel sont les suivants : le registre 2 contient l'adresse de début de la zone à transmettre du 360 ; le registre 3 indique le nombre de mots PDP-8 à envoyer. Les mots PDP-8 de 12 bits doivent être cadrés à gauche dans des demi-mots de 16 bits à cause du câblage de l'interface PDP-8/360. Le registre 4 indique le type de record qui est envoyé. Les types prévus sont :

1	début de transmission d'un programme
2	texte à frapper par le PDP-8
3	programme binaire PDP-8
4	fin de fichier avec contrôle du nombre total de records transmis.

Les records sont envoyés en deux morceaux, appelés record de contrôle et record de données. Par la technique de programmation 360 appelé "chaînage de données", une clôture, c'est-à-dire un mot contenant des 1's partout, est concaténée à la fin de chaque record de contrôle ou de données. Quand le PDP-8 a reçu le nombre de mots prévu pour un transfert, il lit un mot supplémentaire et vérifie qu'il contient des 1's. Ce contrôle remplace le contrôle du signal Compte Zéro qu'on voudrait vraiement effectuer sur le PDP-8. Une deuxième technique de programmation 360, appelé "chaînage de commandes", envoie les deux morceaux d'un record (contrôle, clôture, données, clôture) avec un seul appel à IOPDA, le module qui traite une opération de base.

Le record de contrôle est d'une longueur fixe de 8 mots PDP-8 (16 octets 360). Les mots de ce record ont la signification suivante :

Mot 1	Type du record de données (1 à 4)
Mot 2	Adresse de rangement côte PDP-8
Mot 3	Champ PDP-8 (bits 6-8)
Mot 4	Nombre de mots de données (nég)
Mot 5	Nombre de zéros dans les données (nég)
Mot 6	Somme des mots de données (nég)
Mot 7	Toujours 101010101010 (binaire)
Mot 8	Somme des mots de contrôle (nég).

Puisque le nombre de mots PDP-8 à transmettre est toujours connu, une indication de longueur incorrecte constitue une erreur grave. La longueur du record de données est indiquée dans le record de contrôle.

Les mots d'adresse de rangement et de champ indiquent où il faut ranger une partie de programme PDP-8 en mémoire PDP-8 (dans ce cas, le mot de type contient 3). Le format du record de contrôle a été choisi pour minimiser le traitement sur le PDP-8. C'est pour cela que le champ PDP-8 est mis sur les bits 6-8 afin de permettre une fabrication aisée de l'instruction CDF. Les comptes sont transmis en régatif (complément à 2) pour la même raison.

Les mots zéros sont comptés parce que la vérification de somme logique ne les prend pas en compte, et parce qu'ils peuvent être générés par le PDA en cas d'erreur de programmation.

Essentiellement la fonction de WPDP consiste à générer un record de contrôle pour le record de données donné en paramètres, et de mettre en place les CCW's correspondants. Le premier CCW envoie le record de contrôle ; il est chaîné par données au deuxième, qui envoie la clôture. Le deuxième CCW est chaîné par commande au troisième CCW qui envoie le record de données. Le PDP-8 doit donc envoyer Fin de Record après la réception de la clôture pour que le 360 commence à envoyer les données. Par chaînage de données, le 360 concatène une deuxième clôture aux données ; le PDP-8 doit terminer le transfert des données et de la clôture par un seul Fin de Record.

Avant d'envoyer un record au PDP-8, WPDP attend systématiquement le signal Interruption avant d'appeler IOPDA pour faire le transfert.

OPERATIONS D'ENTREE-SORTIE DE BASE (Module PDAIO)

Le module PDAIO centralise toutes les entrées-sorties sur le PDA. Ainsi, un changement de système peut s'effectuer en mettant à jour uniquement ce module. Il y a quatre points d'entrée dans le mdoule PDAIO.

Le point d'entrée INTWAIT permet au programme appelant PDAIO d'attendre un signal Interruption du PDP-8 avant de démarrer une nouvelle opération. Le contrôle est rendu au programme appelant si ce signal a déjà été reçu, sinon, INTWAIT attend Interruption avant de rendre le contrôle au programme appelant.

Le point d'entrée IOPDA prend comme paramètre une chaîne de CCW contenant au moins un CCW. L'exécution d'une telle chaîne de CCW constitue une opération de base pour l'appelant. Après la fin de ou des commandes canal indiquées dans les CCW, le contrôle est rendu au programme appelant, avec indication d'erreur s'il y a lieu.

Le point d'entrée PDASET fait les fonctions d'initialisation du PDA, et établit le lienage avec le système (CMS dans cette implémentation) pour récupérer et traiter les interruptions de fin d'opération.

Le point d'entrée PDACLR détruit le lienage établit par PDASET. Après l'appel de PDACLR, les interruptions venant du PDA ne sont plus prise en compte par CMS.

Quand le contrôle est rendu à l'appeleur, un code retour est placé dans le registre 15. Le code 0 indique un retour normal. Un code 6 indique une erreur, et le code 12 indique que le PDP-8 a terminé une opération de lecture ou d'écriture par un signal de Fin de Fichier au lieu de Fin de Record.

Le fonctionnement de PDAIO sera décrit maintenant en détail.

La fonction INTWAIT est appelée sans paramètres en utilisant le lienage standard. Il teste un drapeau de non-attente (NOWAITSW) ; s'il contient 1, indiquant qu'un signal Interruption a été reçu à la fin du transfert précédent le contrôle, est rendu tout de suite à l'appeleur. Sinon, le sous-programme interne PDINTW est appelé pour attendre l'arrivée d'une interruption entrée/sortie sur le PDA. Si cette interruption a été provoquée par le signal Interruption, un retour normal s'effectue. Mais si le mot d'état stocké par le canal indique autre chose que Interruption, un sondage est fait sur le PDA pour déterminer la cause de l'erreur. Ensuite, un retour d'erreur a lieu.

Le point d'entrée IOPDA sert à exécuter une chaîne de CCW qui commence à une adresse indiquée comme paramètre. D'abord, un contrôle est fait pour vérifier que le drapeau de non-attente (NOWAITSW) contient 0 ; sinon, une erreur de logique de programmation est indiquée. Ensuite, le programme tente de lancer l'opération par l'instruction SIO. Si le démarrage ne réussit pas, trois cas sont possibles ; ils sont indiqués par le code condition du PSW. Un code condition de 3 indique que le PDA ne fait pas partie de la configuration virtuelle et devrait être attaché. En pratique, ce cas n'arrive pas parce que l'on attend une Interruption avant de démarrer la première opération d'entrée/sortie. Un diagnostic est imprimé, et on donne à l'utilisateur de la machine virtuelle (opérateur de CMS) le choix de relancer l'opération ou d'abandonner avec retour erreur. Si le code condition est 2 (canal occupé), un diagnostic est imprimé et le SIO est relancer Cette condition ne devrait pas se produire sous CP. Un code condition de 1 indique qu'une condition d'erreur ou de fin d'opération existe dans le canal. Dans ce cas, on se branche à SENSE pour en déduire la cause.

Si le SIO a réussi, on attend l'interruption de fin d'opération sur le PDA. Qautre conditions de terminaison sont prévues : elles correspondent toutes à un transfert du nombre exact de mots voulu. Le transfert peut se terminer par le signal Fin de Record ou Fin de Fichier venant du PDP-8. Après ce signal, le PDP-8 aurait pu envoyer Interruption, et ce signal Interruption peut être superposé sur les conditions de fin de transfert. Si le signal est ainsi superposé,

le drapeau de non-attente (NOWAITSW) est mis à un pour indiquer que Interruption a déjà été reçu, et un sondage se fait pour contrôler que le signal Interruption est arrivé après Fin de Record ou Fin de fichier. S'il est arrivé avant, l'octet de sondage (sense byte) portera une indication de Interruption de Commande, et une erreur sérieuse est indiquée.

La routine interne SENSE est appelée par simple branchement lorsqu'il est nécessaire de faire une opération de sondage (sense) sur le PDA. A moins que le sondage soit demandé exprès par une Interruption arrivant presque simultanément avec une fin de transfert, le sondage est effectué seulement si le bit de contrôle d'unité (unit check) est indiqué dans l'octet d'état. L'opération de sondage doit s'effectuer sans le moindre problème ; sinon, une erreur sérieuse de hardware est indiquée. Des diagnostics indiquent les résultats du sondage. (Voir Messages du programme de transfert).

Les points d'entrée PDASET et PDACLR utilisent une fonction CMS, HNDINT, pour établir et détruire le lienage nécessaire au traitement des interruptions venant du PDA, par l'utilisateur. On appelle PDASET avant d'utiliser le PDA, et PDACLR quand on n'en a plus besoin. Ces fonctions remettent le drapeau de non-attente (NOWAITSW) à zéro.

TRANSFERT COTE PDP-8

Sur le PDP-8, il faut charger le programme TRAN, pour recevoir les programmes PDP-8 venant du 360. Ce programme est assez simple. Il lit un record de contrôle et, selon le type indiqué, se branche sur le programme de traitement propre à ce type. Le programme de traitement pour le texte lit un texte et le frappe sur le télétype. Si le type de données est du programme, ces données sont rangées dans le champ et à l'adresse indiquée par le record de contrôle.

Le sous-programme CKSD compare le compte de mots zéros des données, et la somme des données, avec les valeurs prédites dans le record de contrôle.

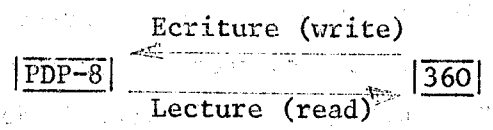
Le sous-programme WR360 lit un bloc de mots venant du PDA en mémoire PDP-8. Il accumule le compte de zéros et la somme pour vérification ultérieure. Lorsque le dernier mot a été reçu, WR360 lit un mot supplémentaire et vérifie que c'est bien une clôture. Enfin il revient à l'appeleur.

Les erreurs produisent des HLT sur le PDP-8 ; une fois que le transfert est au point, les HLT indiquent une erreur due au hardware ou une intervention opérateur.

Le programme PDP-8 inclut un bout de programme pour lire une nouvelle version de TRAN venant du 360. Voir la section "Création d'une nouvelle version du programme de transfert".

INSTRUCTIONS DE L'INTERFACE PDP-8/360

Convention : le lecture (read) et l'écriture (write) se réfèrent au 360 suivant le schéma :



LISTE DES INSTRUCTIONS

Mnemonic	Code	Opération
IFSR	6701	Saut d'une instruction, si le 360 demande une lecture
IFCB	6702	Mise à zéro du buffer
IFLO	6704	Transfert de l'accumulateur dans le buffer
IFRD	6706	Mise à zéro du buffer, puis transfert de l'accumulateur dans le buffer
IFSW	6711	Saut d'une instruction, si le 360 demande une écriture
IFLI	6712	Mise à zéro de l'accumulateur et transfert des lignes IBM dans le buffer
IFLA	6714	Transfert du buffer dans l'accumulateur
IFWR	6716	Mise à zéro de l'accumulateur, puis transfert des lignes IBM dans l'accumulateur
IFBEG	6721	Initialisation du transfert : 0 => Ready, WC, 1 => Eni, Enable, Suppress Pari
IFEOR	6722	Envoi de EOR au 360 + 0 => Ready, WC
IFEOF	6724	Envoi de EOF au 360 + 0 => Ready, WC
IFSC	6731	Saut d'une instruction, si le flag WC = 0 est à 1
IFDEM	6732	Envoi de Demand au 360 + 0 => Ready
IFPAR	6734	Mise à zéro de Suppress Parity
IFDIS	6741	Mise à zéro de Enable
IFIOF	6742	Mise à zéro de Eni
IFINT	6744	Envoi de Interrupt au 360

Fourni par MM. Tuffelli et Vuillod

CREATION D'UNE NOUVELLE VERSION DU PROGRAMME DE TRANSFERT

Pour mettre à jour le programme du 360, il suffit de modifier le fichier TRAN SYSIN (avec l'éditeur), et de réassembler pour avoir le nouveau paquet TEXT correspondant. Ensuite, on en génère un fichier de type MODULE (non-translatable) avec les commandes CMS suivantes :

```
LOAD TRAN    (CLEAR TYPE)
GENMOD TRAN PIMAGE
```

Le fichier MODULE ainsi généré ne contient pas l'image mémoire du PDP-8, car cette image est générée pendant l'exécution. C'est pour cela qu'on spécifie le nom du CSECT de l'image mémoire dans la commande GENMOD.

Si on désire envoyer une nouvelle version du programme PDP-8 au PDP-8, le procédé est plus compliqué. On assemble d'abord la nouvelle version de TRAN PALD, ce qui donne un fichier TRAN OCTAL. Ensuite, on charge TRAN1 sur le PDP-8, soit avec ODT, soit on charge le TRAN1 de la version précédente avec le Moniteur. Il y a une HLT au début de TRAN1. Sur le 360, on appelle TRAN sans spécifier un nom de fichier : TRAN.

On reçoit alors un message demandant un nom d'un fichier de type OCTAL. En réponse, on tape un mot de passe de 9 lettres (voir la liste d'assemblage du module principal de TRAN). Sur le 360, le programme, au lieu de lire un fichier OCTAL et d'appeler WPROG pour envoyer un programme utilisateur au 360, lira plutôt le fichier TRAN OCTAL et enverra l'image des mémoires 7200-7577 du champ 2. L'envoi de cette image est déclenché par la réception du signal Interruption que TRAN1 envoie au début de son exécution. La nouvelle version de TRAN2 est envoyée comme un seul record, sans record de contrôle.

Une nouvelle version de TRAN1 peut être envoyée au PDP-8 comme un programme utilisateur en spécifiant TRAN comme nom de fichier sur le PDP-8. Ainsi, sur le PDP-8, TRAN1 reçoit une nouvelle version de TRAN2, et TRAN2 reçoit une nouvelle version de TRAN1.

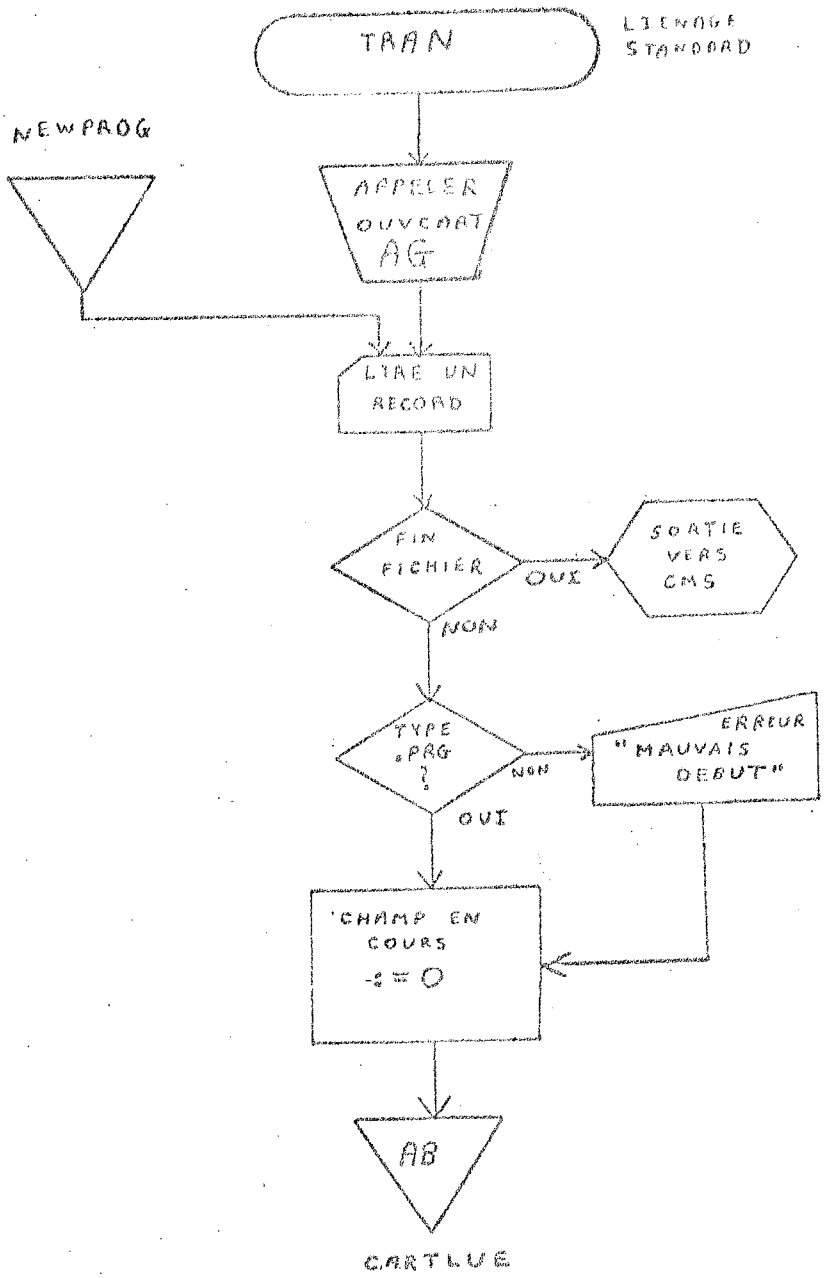
Le programme côté PDP-8 est sauvegardé avec les commandes MONITEUR suivantes :

```
SAVE TRAN ! 27200-7577 ; 7200     pour TRAN2, et
SAVE TRA1 ! 10200 ; 200           pour TRAN1.
```

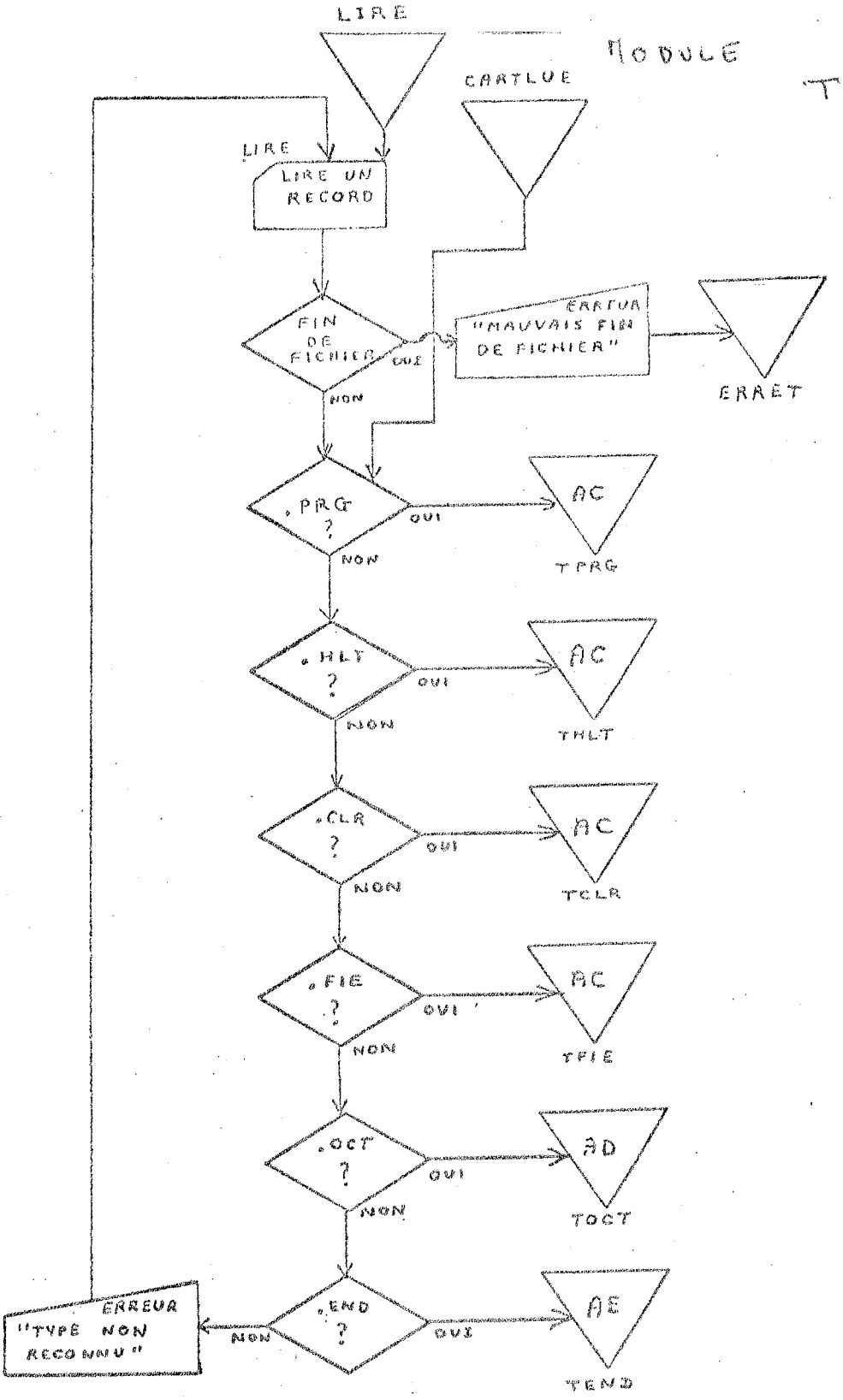
Les utilisateurs ont besoin de TRAN, seulement. La copie de TRAN sur les DEC-tapes des utilisateurs peut se faire avec PIP.

On peut sauvegarder les programmes TRAN et TRA1 sur bande perforée en format BIN en utilisant le programme "Binary Punch" d'Equipement Digital. Mais avant de lancer la perforation, il faut mettre un caractère de champ (CTRL-H pour le champ 1, CTRL-P pour le chap 2) après le "leader" et avant le programme perforé. Le caractère de champ n'est pas inclus dans la somme logique des caractères du PDP-8 de la bande perforée ; ainsi il peut être ajouté par l'opérateur.

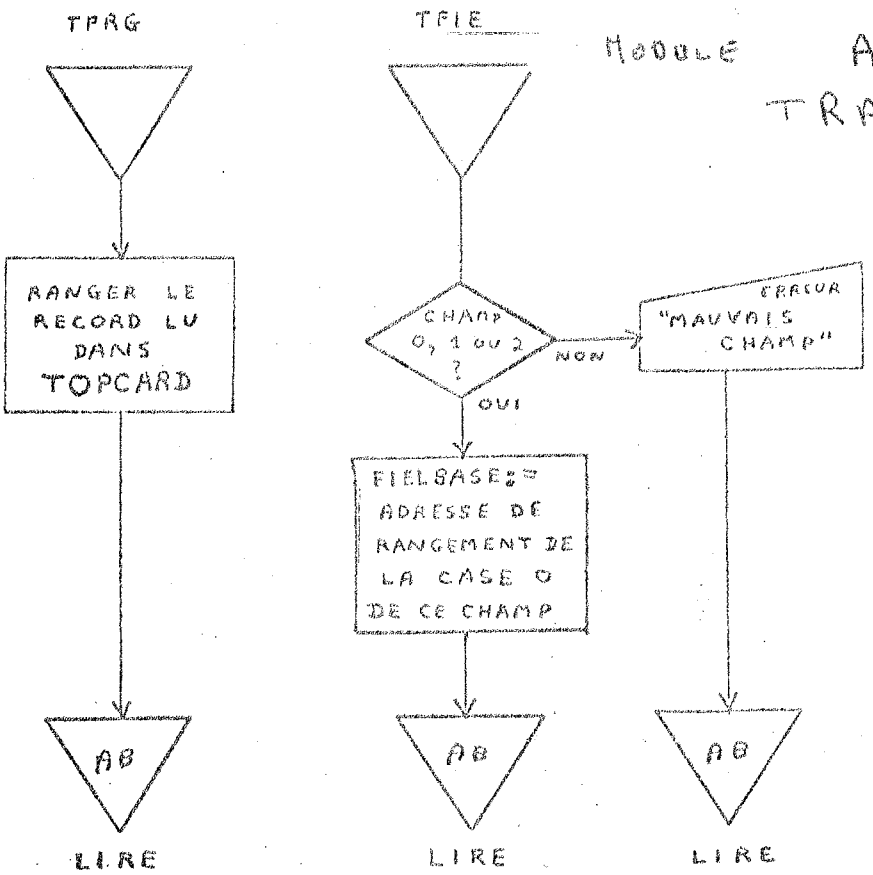
MODULE TRAN AA

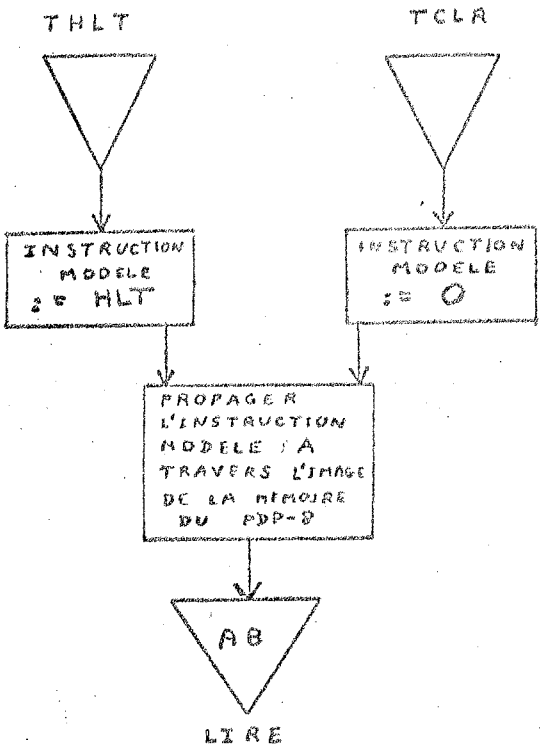


MODULE AB TRAN



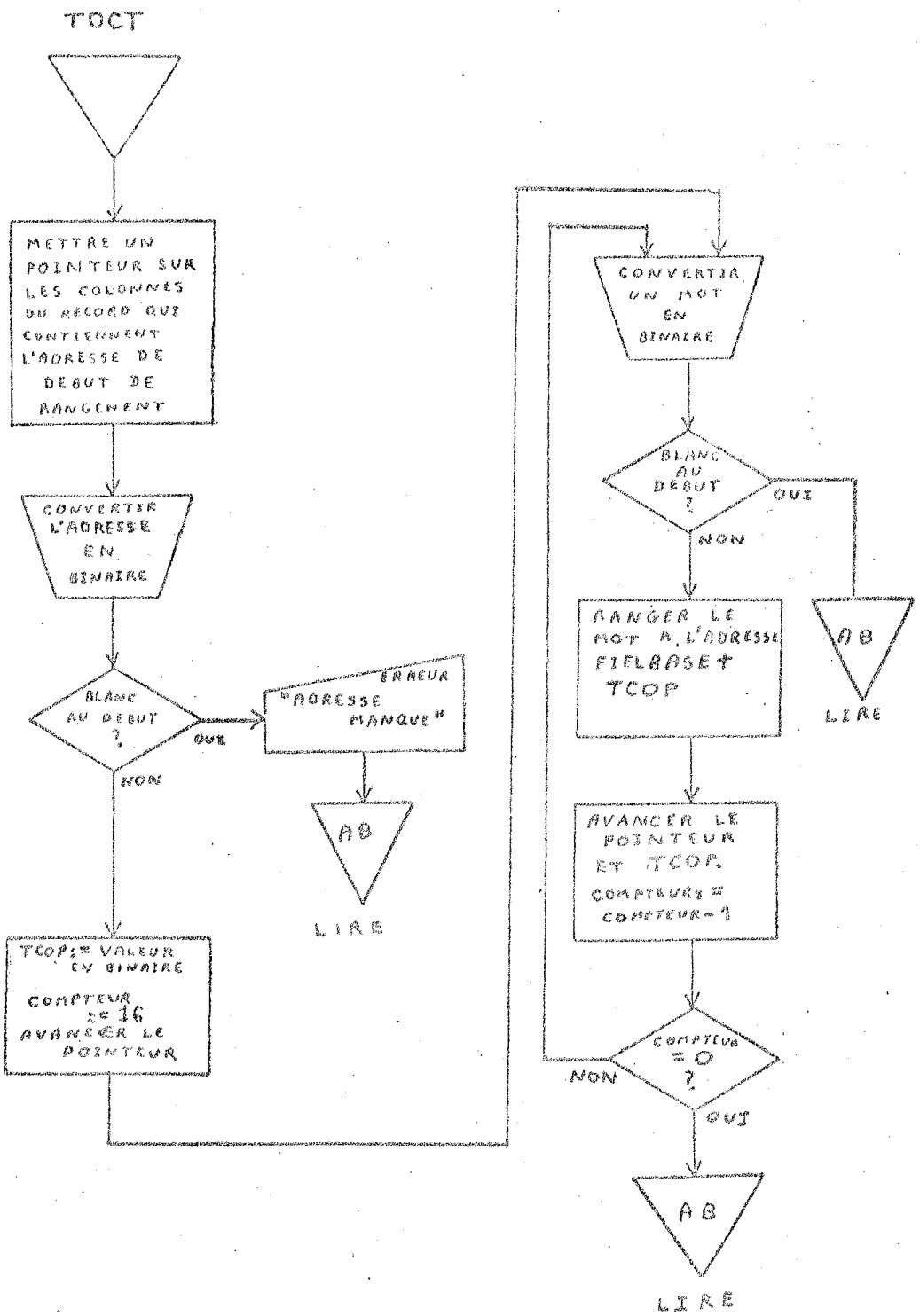
MODULE AC TRAN.



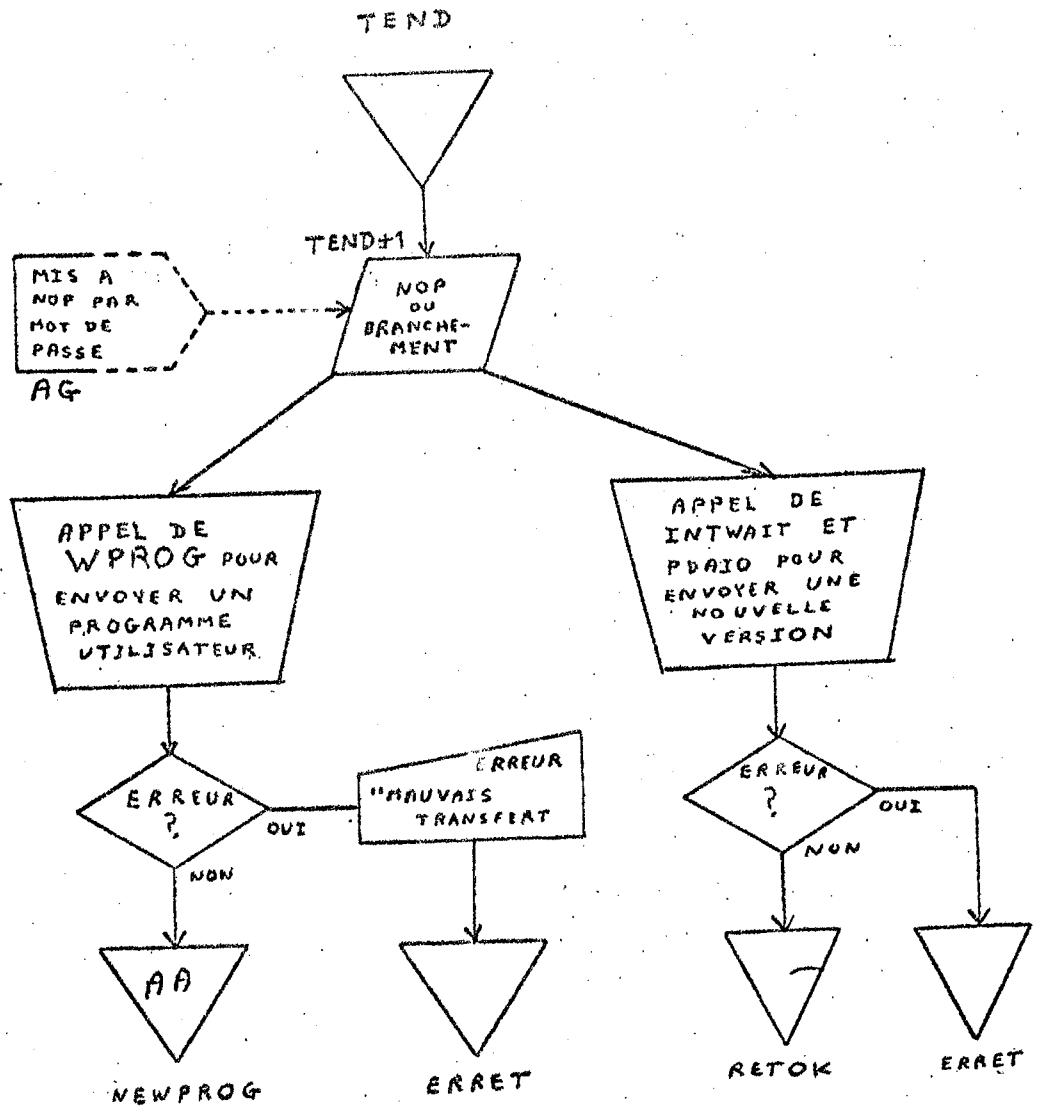


AD

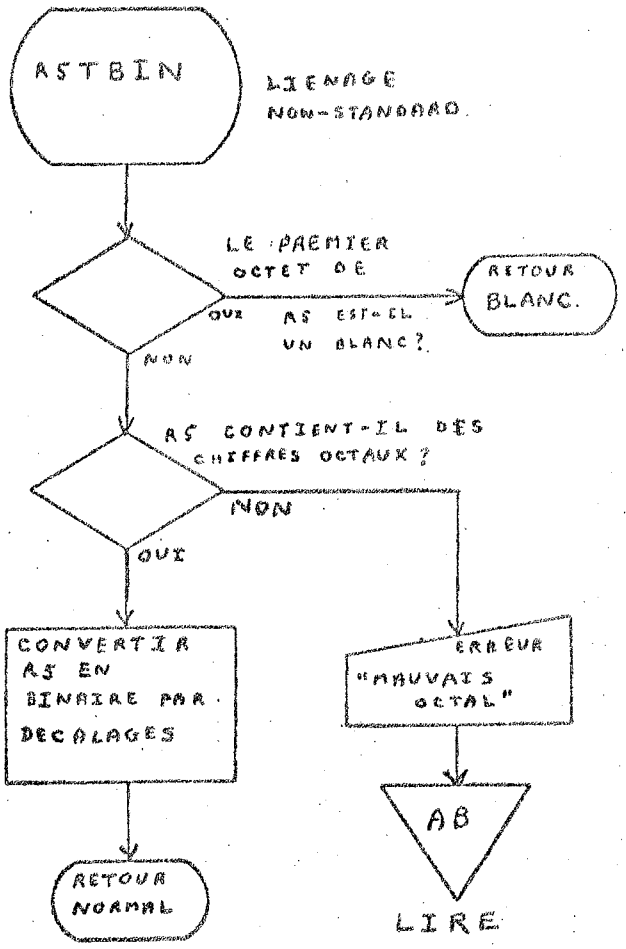
MODULE TRAN



MODULE AE TRAN

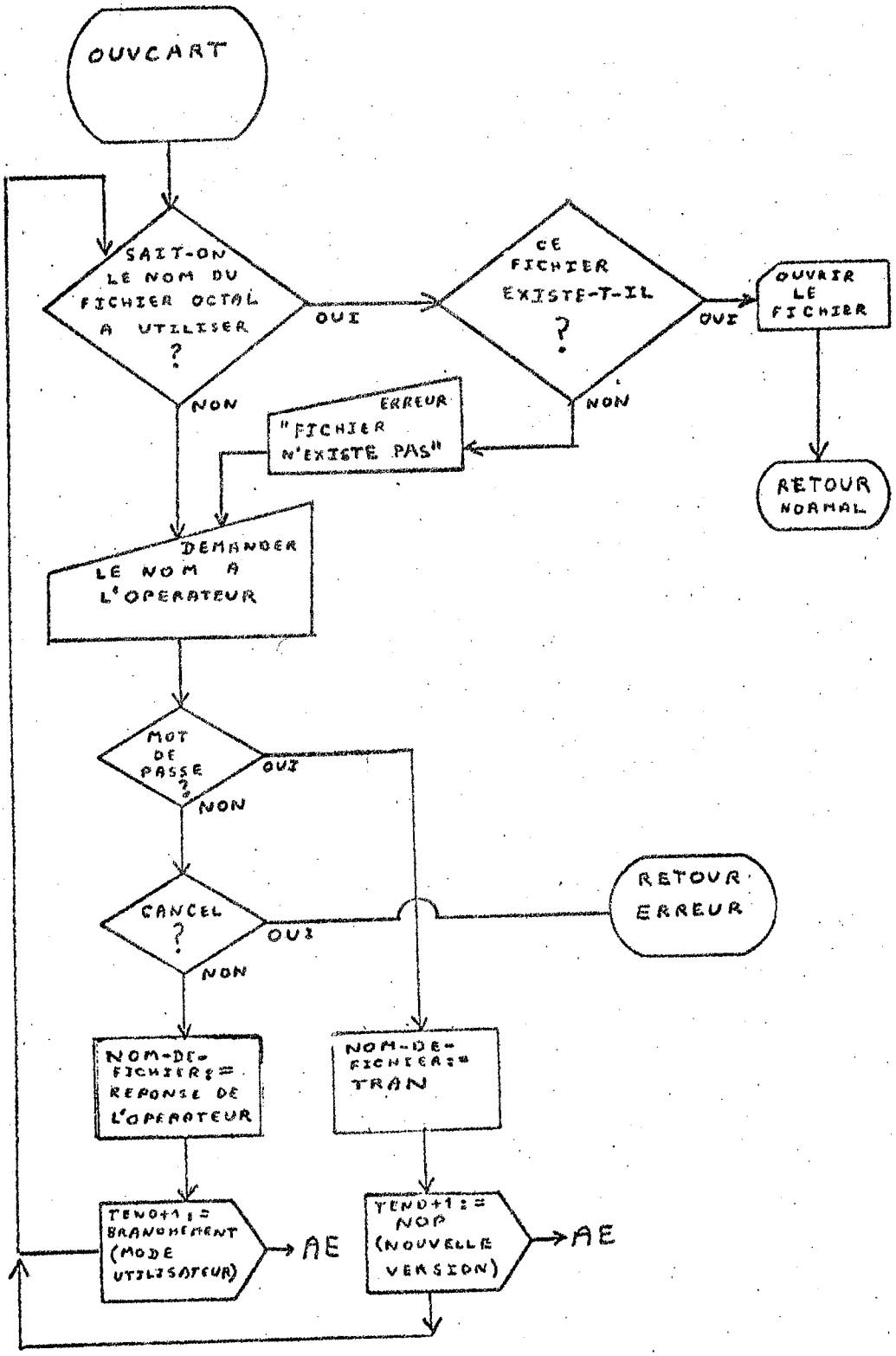


MODULE TRANAF



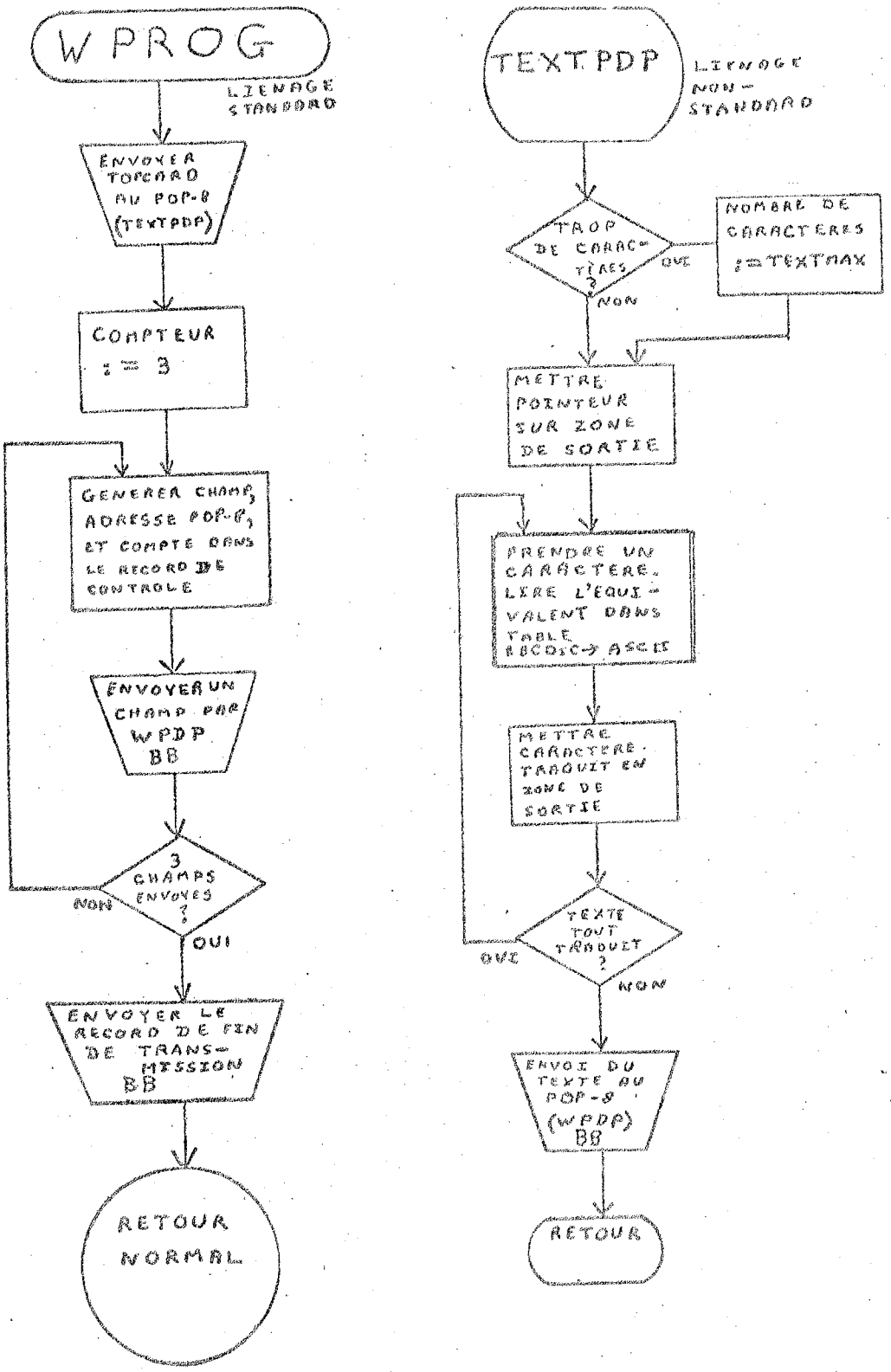
AG

MODULE TRAN



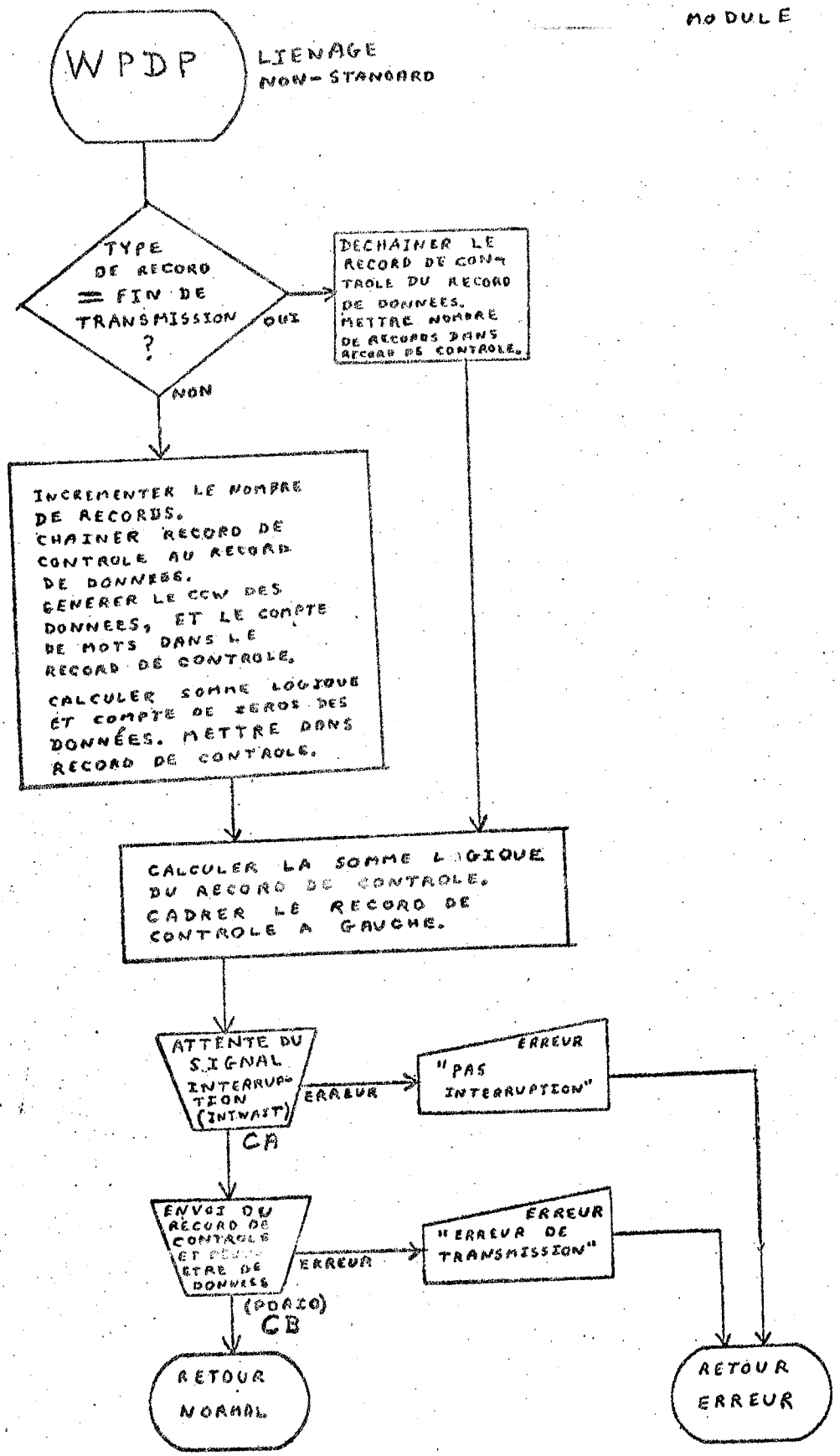
BA

MODULE WPROG



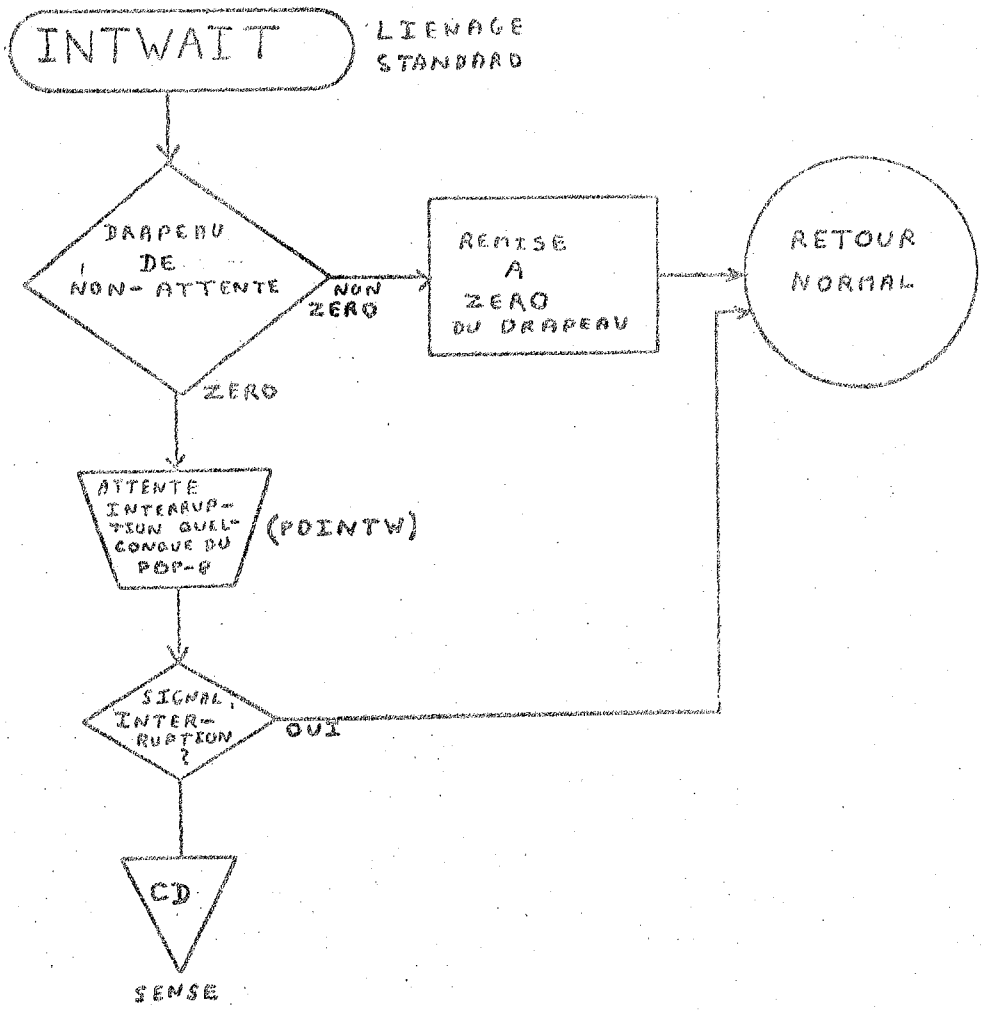
BB

MODULE WPROG

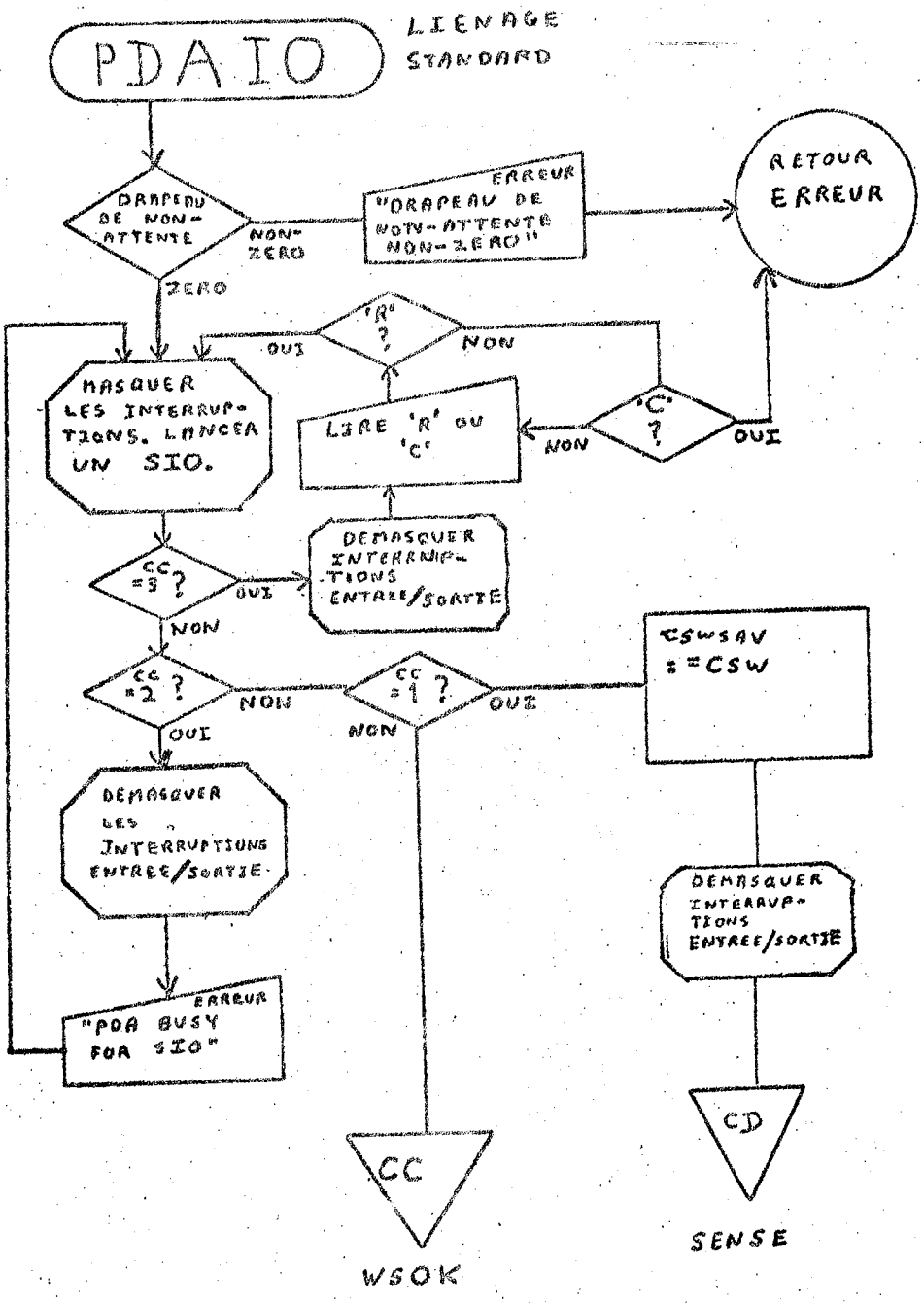


CA

MODULE IOPDA



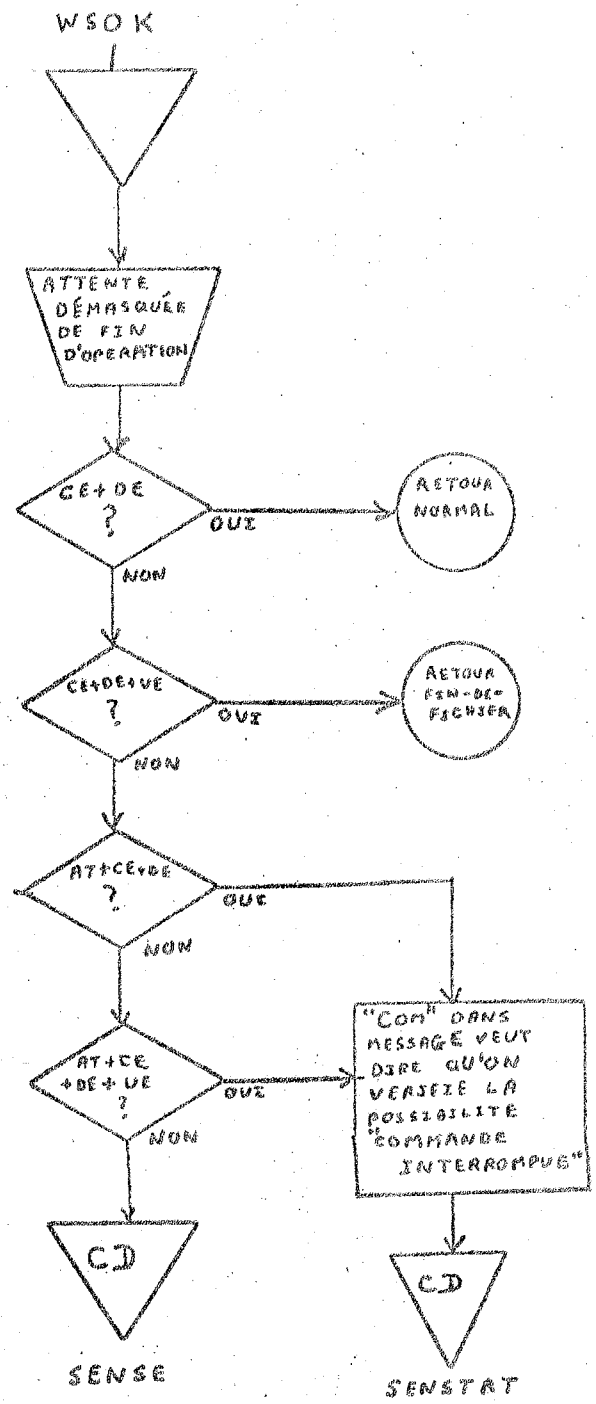
MODULE IOPDA



CC = CODE-CONDITION DU PSW

CC

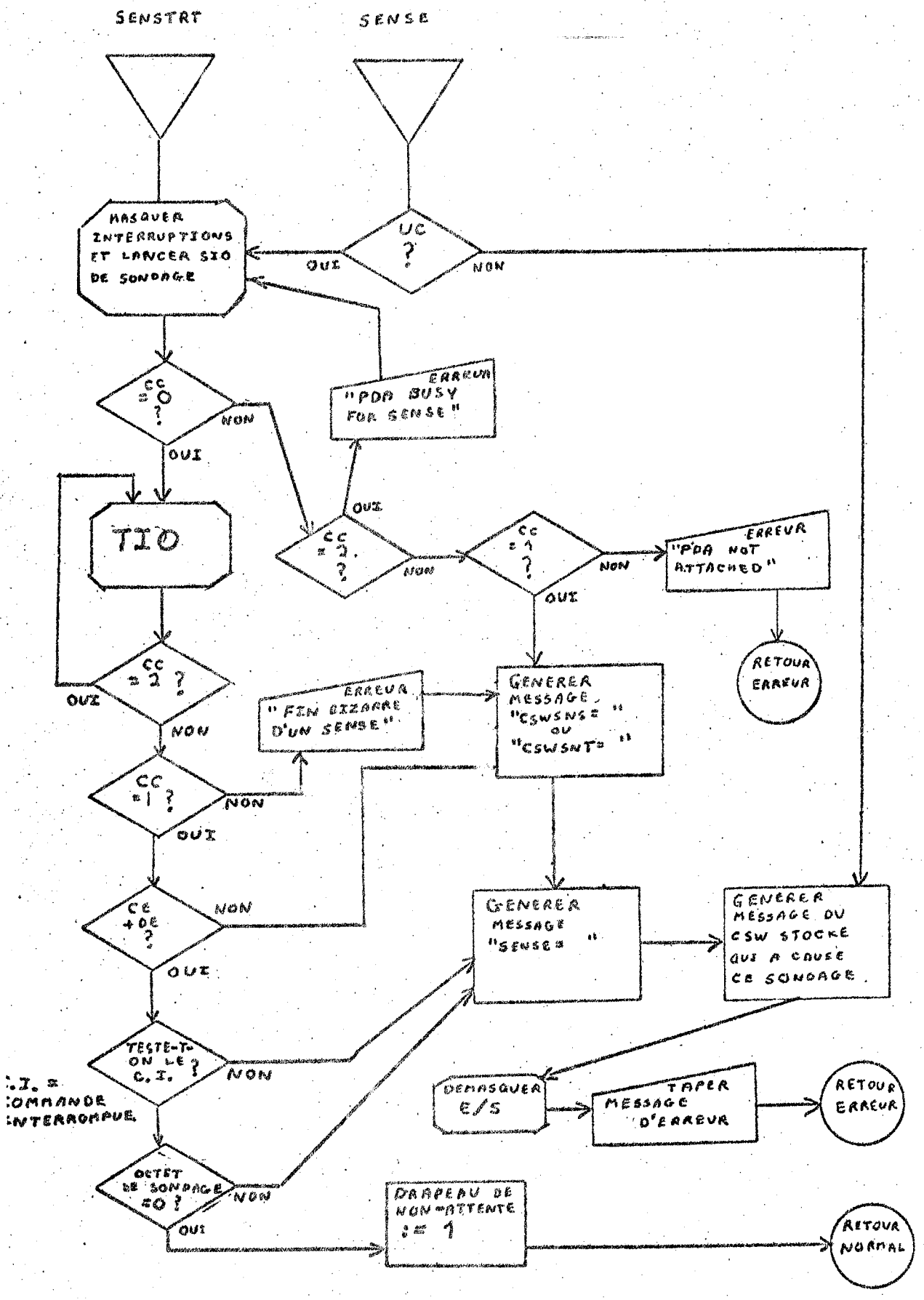
MODULE IOPDA



CE= CHANNEL END
DE = DEVICE END
AT = ATTENTION
UE = UNIT EXCEPTION
UC = UNIT CHECK

C'D

MODULE IOPDA



CHAPITRE IV

STRUCTURE D'UN SYSTEME D'EXPLOITATION D'UN PETIT ORDINATEUR UTILISANT LES RESSOURCES D'UN GRAND ORDINATEUR

SYSTEME DE BIBLIOTHEQUE PDP-8 SUR 360

Dans les grandes usines, on utilise très souvent le contrôle des processus industriels par ordinateur. La configuration comporte un grand ordinateur destiné aux fonctions de gestion de l'entreprise et au contrôle général de la production. Le grand ordinateur peut être relié à plusieurs petits ordinateurs répar dans l'usine. Chacun des petits est dédié au contrôle d'une partie particulière de l'usine et doit s'occuper éventuellement de trois fonctions : le maintien de l'état des processus industriel (c'est-à-dire des températures, pressions, tensions, etc du processus) à l'état désiré, l'acquisition de données sur la production destinée à être traitées statistiquement sur le grand ordinateur, et la réception de commandes du grand ordinateur. Ces commandes peuvent être un signal de démarrage ou d'arrêt des opérations, un changement de paramètres pour le processus ("changer la pression à 2.43 atmosphères et la température à 1400 degrés" par exemple), ou même le chargement d'un nouveau programme sur le petit. Le chargement de programmes à partir du grand calculateur rentabilise la ligne de communication qu'il faut entre chaque petit ordinateur et le grand si on veut contrôler toute l'usine à partir de l'endroit où se trouve le grand calculateur. Aussi, le chargement de programmes par ligne de communication élimine les inconvénients suivants :

1) Il faudrait arrêter l'usine pendant le changement des programmes sur les petits.

2) Un opérateur devrait faire le tour des petits pour charger les nouveaux programmes sur les petits, les vérifier, et les lancer.

3) Le chargement d'un programme sur un petit ordinateur serait long (donc coûteux) à cause du nombre important d'opérations manuelles, et aussi parce que les périphériques du petit sont spécialisés au traitement de données analogiques. Pour cette raison, on dispose rarement de périphériques rapides qui servent uniquement au chargement de programmes.

Le chargement par une ligne de communication de vitesse moyenne (2400 baud par exemple) offre un moyen assez rapide et peu coûteux pour changer d'un programme à l'autre. Dans une minute, on peut transmettre $60 \times \frac{2400}{12} = 12$ K mots de 12 bits.

Mais le plus grand avantage du chargement par ligne téléphonique, c'est le fait de pouvoir synchroniser le changement de processus partout dans une usine à partir d'un horaire déterminé par le grand ordinateur. Ceci est très important dans une usine chimique, ou une raffinerie, où on change très souvent le genre de produits traités.

Le chargement de programmes à partir d'un grand ordinateur implique le problème de planning simultané des horaires de machines ; il faut que le petit et le grand soient disponibles en même temps pour effectuer le transfert. Dans un milieu universitaire, où on désire expérimenter plusieurs systèmes, ce n'est pas toujours le cas. Mais dans une installation industrielle, où l'ordinateur central est dédié à la gestion de l'usine, ce problème a moins d'importance.

Un autre problème à résoudre concerne la fiabilité des programmes qu'on envoie aux petits ordinateurs La réception de ces programmes, ainsi que les tâches de communication avec le grand, et aussi le passage éventuel d'un programme à un autre sur le petit, doivent se faire avec un moniteur sur le petit qui est protégé des erreurs éventuelles du programme qu'il a reçu du grand ordinateur.

Voici une méthode de protection d'un moniteur sur un petit ordinateur qui serait fiable mais qui ne coûterait pas cher en hardware. D'abord, on prévoit deux modes de fonctionnement : mode maître et mode esclave. En mode maître, toute la mémoire du petit ordinateur est disponible et toutes les instructions sont permises. En mode esclave, certaines parties de la mémoire deviendraient inaccessibles, et certaines instructions seraient interdites. Dasn ce mode, la mémoire serait divisée en plusieurs zones. L'accès à la première zone, qui est réservée au moniteur, serait complètement interdit au programme en mode esclave

soit par le système d'adressage mémoire, qui rendrait un tel accès impossible, soit en déclenchant une interruption au moniteur quand une tentative d'accès se fait. On peut trouver un exemple de la première méthode sur le TSS/8, un PDP-8/I avec les modifications nécessaires au travail en temps partagé. Sur le TSS/8, les programmes en mode esclave se déroulent dans un champ PDP-8 autre que le champ 0. Le programme ne peut adresser que le champ où il a été implanté, car les instructions qui permettent de changer de champ sont privilégiées. La deuxième zone de la mémoire serait accessible au programme en mode esclave. Si on désire un système de protection mémoire plus évoluée, et si la structure du petit calculateur permet une séparation nette entre le code du programme qui ne se modifie pas et les données qui sont modifiables, on peut définir des conditions supplémentaires pour permettre l'accès à une adresse mémoire donnée. Par exemple, on peut empêcher la lecture d'une zone de données comme instructions, ou l'utilisation d'une zone d'instructions comme données en mode esclave. Ainsi, le programme esclave ne peut pas se détruire lui-même. Aussi, on pourrait prévoir une zone de mémoire que le programme esclave pourrait lire comme données mais pas exécuter ni écrire dedans. Dans cette zone, le moniteur pourrait ranger les paramètres nécessaires au programme esclave. Ces paramètres pourraient contenir des marges de sûreté (maximum de température, maximum de pression...) que le programme esclave ne doit jamais changer.

Quelles instructions peuvent être utilisées en mode esclave sur un petit ordinateur ? Il est évident qu'il faut restreindre l'emploi des instructions qui gèrent la protection mémoire et l'état majeur de la machine (arrêt ou marche, mode esclave ou mode maître) au mode maître. Sur un grand ordinateur multiprogrammé, il faut aussi empêcher l'emploi des instructions d'entrée-sortie dans le mode esclave, car il est nécessaire de s'assurer qu'un programme n'essaye pas d'utiliser des périphériques qui ne lui ont pas été alloués. Mais sur un petit calculateur avec des périphériques spéciaux, il est inutile et même très lourd de faire faire toutes les opérations d'entrée-sortie par le moniteur. Par exemple, la lecture d'un convertisseur analogique-digital dans l'accumulateur ne peut pas détruire le moniteur. Il n'y a aucune raison, donc, d'empêcher au programme d'exécuter la lecture directement, pourvu qu'on sache d'avance que deux tâches sur le petit ordinateur ne peuvent pas essayer d'utiliser le même convertisseur en même temps.

Les transferts de données en bloc posent un problème plus délicat de protrection mémoire ; il faut éviter que les données lues écrasent le moniteur. Lorsqu'une opération de lecture en bloc a démarrée en mode esclave, le système de protection mémoire doit refuser d'écrire les données lues dans la mémoire protégée. Une autre solution, moins coûteuse en hardware mais plus coûteuse en temps d'exécution, consiste à rendre privilégiées les opérations de transfert en bloc. Ceci entraîne la centralisation de telles opérations dans le moniteur.

Les opérations de communication avec le grand ordinateur doivent être privilégiées pour protéger contre les erreurs du programme transféré. Ainsi, si le petit calculateur reste toujours interruptible par le grand pendant l'exécution du programme transféré, les erreurs de programmation tels que les boucles et les mauvaises terminaisons deviennent récupérables.

Puisque le programme du petit ordinateur serait sans doute généré par le grand, on peut se servir d'un compilateur pour la génération du code du petit. Ainsi, les programmes seraient écrits en langage à haut niveau plutôt qu'un langage d'assemblage propre au petit ; cette méthode donnerait des programmes indépendants des particularités du petit ordinateur, une possibilité d'optimiser le code du programme, et surtout un temps d'écriture et de mise au point considérablement réduit. Sur le PDP-8, par exemple, on perd beaucoup de temps à s'occuper de la division d'un programme en pages de 128 mots. Lorsqu'on passe d'une page à une autre, l'assembleur PDP-8 génère les liens convenables, mais il ne cherche pas à trouver l'emplacement en mémoire qui minimiserait ces liens. Dans un milieu industriel, où le temps de mise au point d'un programme est d'une importance capitale, mais les spécialistes en programmation sont peu nombreux, il est indispensable que les programmeurs des petits ordinateurs disposent d'un compilateur évolué qui leur permette d'exprimer leur problème en termes qui leur sont familiers.

APPENDICE A

MODES D'EMPLOI

MODE D'EMPLOI DE L'ASSEMBLEUR SOUS O/S

Sous O/S, l'assembleur est appelé comme suit :

```
//JOJO     JOB
//JOBLIB DD  DSN = S6400.P0360.JONES.PDPLIB
//S1     EXEC PGM=ASSPDP
//SYSPRINT DD SYSOUT=A
//SYSPUNCH DD SYSOUT=B
//SYSUT1   DD UNIT=2314,SPACE=(101,(x,y))
//SYSIN    DD *
.OPTIONS   DECK
.TITRE                                          Premier programme
           Enoncés du premier programme
           END
.OPTIONS   DECK
.TITRE                                          Deuxième programme
           Enoncés du deuxième programme
           END
.......
/*
```

Le fichier SYSPRINT contient la liste d'assemblage. Le fichier SYSPUNCH reçoit les paquets en format OCTAL du programme assemblé. SYSUT1 doit contenir assez de place pour mettre un record intermédiaire de 101 octets par énoncé à assembler. Ainsi, les paramètres x et y de SPACE doivent satisfaire la relation $x+15y > n$, où n est le nombre d'énoncés entre une carte .TITRE et la carte END. Le fichier SYSIN contient les images de cartes qui sont préparées par le programmeur.

MODE D'EMPLOI DE L'ASSEMBLEUR SOUR CMS

Sous CMS, l'assembleur PDP-8 est disponible dans le fichier ASSP TEXT. Tous les fichiers utilisés par l'assembleur ont le même nom (FILE), et les types SYSPRINT, SYSPUNCH, SYSUT1 et SYSIN. Les noms fixes des fichiers sont dus à l'interprétation très pauvre des macros O/S par le système CMS.

Pour les fichiers à assembler, le type PALD est recommandé comme convention. Pour les listes d'assemblage, le type PALLIST est suggéré ; pour la sortie perforée, le type OCTAL est suggéré. Il faut noter que l'assembleur 360 utilise les types de fichiers SYSIN, LISTING et TEXT. Les types ont été changés pour deux raisons. D'abord les informations dans les fichiers sont différents, et ont un format qui diffère de celui de l'assembleur 360. Deuxièmement, il devient possible d'utiliser le même nom de fichier pour l'assembleur 360 et l'assembleur PDP-8 sans risque de confusion quand on écrit un programme pour le 360 et son pair pour le PDP-8. Aussi, l'utilisateur de CMS est protégé contre l'erreur de charger un fichier OCTAL en mémoire avec la commande LOAD, parce que cette commande exige le type de fichier TEXT en entrée.

Puisque les caractéristiques des fichiers de type PALD et PALLIST ne sont pas connus aux programmes standards de CMS, l'utilisateur doit indiquer ces caractéristiques chaque fois qu'il se sert de l'éditeur ou qu'il sort une liste d'assemblage. Quelques modifications triviales à CMS pourraient résoudre ce problème.

Par exemple, quand on veut éditer un fichier de type PALD, on doit mettre en fonction la sérialisation des cartes (elle est hors fonction par défaut), établir la zone de travail dans la colonne de 1 à 72 (1 à 80 par défaut), et définir les positions de tabulation. Un fichier, PALD MACRO, contient les commandes nécessaires :

```
ZONE   1  72
VERIFY 72
SERIAL XXX
TABSET  1  9  17  25  33  41  49  57  65
```

Lorsqu'on a appelé l'éditeur pour un fichier PALD, on peut lancer ces 4 commandes d'édition en frappant la ligne de commande suivante à l'éditeur :

```
MDEF PALD M # PALD
```

Pour faire l'assemblage d'un programme PDP-8 sous CMS, on se sert du fichier PALD EXEC, qui contient les commandes CMS suivantes :

```
&TYPEOUT   OFF
ERASE FILE SYSUT1
ERASE FILE SYSPRINT
ERASE FILE SYSPUNCH
ERASE &1   PALLIST
ERASE &1   OCTAL
COMBINE FILE SYSIN P1 &1 PALD P1
LOAD  ASSP
START
ERASE FILE SYSUT1
ERASE FILE SYSIN
ALTER FILE SYSPRINT P1 &1 PALLIST P1
ALTER FILE SYSPUNCH P1 &1 OCTAL   P1
OFFLINE PRINTCC &1 PALLIST P1
ERASE &1 PALLIST.
```

Avec un fichier EXEC comme celui-ci, le disque permanent de l'utilisateur de CMS est nettoyé après l'assemblage. Le fichier : nom-de-fichier OCTAL contiendra le programme assemblé en format OCTAL. Pour se servir de cette "procédure cataloguée" PALD EXEC, on doit taper la commande suivante à CMS :

```
PALD nom-de-fichier.
```

En cas d'erreur à l'assemblage, l'assembleur PDP-8 revient avec un code erreur dans le registre 15. L'utilisateur doit chercher ses erreurs sur la liste d'assemblage. Il n'y a pas d'impression d'erreurs sur le terminal CP/CMS comme dans le cas de l'assembleur 360.

Il faut utiliser la commande OFFLINE PRINTCC (plutôt que OFFLINE PRINT) pour profiter du fait que l'assembleur PDP-8 met des caractères de contrôle ASA dans chaque record d'impression. La commande OFFLINE de CMS ne reconnaît que le type de fichier LISTING comme ayant des caractères de contrôle lorsqu'on fait OFFLINE PRINT.

MODE D'EMPLOI DU PROGRAMME DE TRANSFERT

Le programme de transfert des programmes PDP-8 se divise en deux parties : TRAN sur le 360 et TRAN sur le PDP-8.

La partie pour le 360 est dans le fichier TRAN MODULE. Pour transférer un programme PDP-8, il faut d'abord préparer un fichier de type OCTAL, soit à la main, soit à l'aide de l'assembleur PDP-8. Aussi, il faut que le PDA soit attaché à la machine virtuelle à l'adresse 031. Sur le PDP-8, le programme TRAN doit être chargé.

La séquence du transfert est comme ceci sur la console CP/CMS de l'utilisateur userid :

msg cp attacher 031 to userid as 031 s.v.p.

L'opérateur doit alors taper :

a 031 t userid a 031

Pendant ce temps, l'utilisateur frappe la commande :

TRAN nom-de-fichier

Quand le 360 est prêt pour la transmission au PDP-8, il tape un message sur la console CP/CMS :

SENDING TO PDP-8

et le clavier sera bloqué. On peut alors démarrer le PDP-8, pourvu que le message DEV 031 ATTACHED a été frappé par CP.

Sur le PDP-8, il faut appeler le programme TRAN en mémoire PDP-8. Normalement, ceci se fait à l'aide du Moniteur, en frappant le nom du programme, TRAN, sur la console PDP-8 pendant qu'on est sous le contrôle du Moniteur. En cas de panne des DECtapes, on peut aussi charger ce programme en mémoire PDP-8 à partir de rubans perforés en utilisant le chargeur BIN.

Le programme TRAN occupe les mémoires 7200-7577 du champ 2. Tout le reste de la mémoire, sauf la dernière page du champ 0, qui est réservée au Moniteur, est à la disposition de l'utilisateur. L'utilisateur peut, d'ailleurs, se servir de tout le champ 2 pendant l'exécution.

Le programme TRAN démarre en 7200 du champ 2. Une HLT au début permet au PDP-8 d'attendre que le 360 est prêt pour la transmission. Dès que le 360 est prêt, on démarre la transmission en appuyant sur la clé START du PDP-8. La carte .PRG est frappée sur la console du PDP-8 ; si le transfert s'est déroulé correctement, les trois champs du PDP-8 contiennent le programme de l'utilisateur. A la fin du transfert, un point d'exclamation est frappé à la console PDP-8, et le PDP-8 s'arrête. En appuyant sur CONTINUE du PDP-8, on revient au Moniteur pour faire un SAVE du programme transféré. Il faut taper une commande SAVE pour chaque champ à sauvegarder sur DECtape.

S'il se produit une HLT dans le programme de transfert lui-même, ceci indique une erreur opérateur ou une défaillance du hardware. Dans ce cas, il faut se reporter à la liste d'assemblage de TRAN pour déterminer la cause du problème.

MESSAGES DU PROGRAMME DE TRANSFERT

MODULE TRAN

1) TAPER LE NOM D'UN FICHIER OCTAL OU "CANCEL"

Le clavier se débloque pour permettre à l'utilisateur de CMS de frapper le nom du fichier OCTAL qu'il veut envoyer au PDP-8. Si l'utilisateur tape CANCEL, un retour erreur à CMS a lieu.

2) LE FICHIER "xxxxxxxx" N'EXISTE PAS

Le fichier OCTAL demandé par l'utilisateur n'existe pas sur le P-DISK. Le message (1) suit, et permet à l'utilisateur de corriger une erreur de frappe, ou de terminer le transfert.

3) ccccc
MAUVAIS DEBUT

La carte ccccc débute un paquet, et devrait être de type .PRG La carte est quand même acceptée.

4) ccccc
BAD CARD TYPE

La carte cccccc n'est pas d'un type reconnu dans un paquet OCTAL. La carte est ignorée.

5) .FIE f
ERREUR CHAMP

Le caractère f n'est pas 0, 1 ou 2. Le champ reste inchangé.

6) .OCTdddd
ADRESSE MANQUE OU INVAL

Les quatre digits dddd ne sont pas des digits octaux. La carte est ignorée.

7) .OCTccccccccccccc
BAD OCTAL

Un caractère non-octal se trouve dans un nombre octal de 4 digits. Si la carte contient un commentaire, le blanc qui indique son début doit se trouver dans une des colonnes 9, 13, 17, 21...69. Les nombres octaux qui précèdent l'erreur ont été pris en compte ; le reste de la carte est ignorée.

8) MAUVAIS LECTURE

Erreur de lecture du disque dans RDBUF.

9) ERR FIN DE FICHIER

Une fin de fichier a été rencontrée avant la carte .END qui terminerait un paquet. Le dernier paquet n'est pas transmis.

Dans ces exemples, les caractères cccccc représentent une ligne frappée par CMS qui contient la carte erronnée en cours de traitement.

MODULE WPROG

ERR INT CTL

L'interruption venant du PDA n'a pas été provoquée par un signal Interruption du PDP-8. Redémarrer le transfert.

ERR CTL OR DATA

Une erreur a eu lieu dans IOPDA, qui envoyait un record de contrôle éventuellement chaîné à un record de données.
Redémarrer le transfert.

MODULE PDAIO

1) WAITING FOR PDP-8 INTERRUPT

Le 360 attend un signal Interruption venant du PDP-8. Démarrer le programme PDP-8, si nécessaire.

2) INT PENDING FOR SIO

Le PDP-8 a envoyé un signal Interrupt après une opération normale, mais le programme 360 n'a pas contrôlé la réception de ce signal avant de démarrer une nouvelle opération. Ce message indique une erreur logique de programmation : ou bien le PDP-8 n'aurait pas du envoyer Interrupt après l'opération précédente, ou bien le 360 aurait du le contrôler.

3) PDA NOT ATTACHED...TYPE RETRY OR CANCEL

Le PDA n'est pas attaché pour faire une opération. Le clavier se débloque. Taper R pour retenter l'opération quand le PDA a été attaché à la machine virtuelle ; taper C pour terminer le transfert. Dans la version actuelle du programme de transfert, ce message ne doit pas se produire car le programme de transfert attend toujours Interruption avant de démarrer une opération.

4) PDA BUSY SHOULON'T BE

Le canal du PDA est occupé lorsqu'on essaie de démarrer une opération. Ce message ne doit pas se produire sous CP, car CP arrête la machine virtuelle sur le SIO jusqu'à ce que le canal soit libéré.

5) PDA NOT ATTACHED FOR SENSE

Le PDA a pu accomplir une opération, mais lorsqu'on a essayé de démarrer une opération de sondage SENSE, il n'était plus attaché. Erreur catastrophique.

6) PDA BUSY FOR SENSE

Le canal du PDA a répondu "occupé" lors du lémarrage d'une opération de sondage (SENSE) qui suivait une autre opération. Erreur catastrophique.

7) PDA xxx ERROR---CSW=dddddddd dddddddd zzzz

Une erreur du PDA a causé le stockage d'un CSW.
Le xxx indique ou l'erreur s'est produite.

xxx	Endroit de l'erreur
WAT	Attente de signal Interruption
SIO	Instruction SIO d'une opération
END	Fin d'une opération sans Interruption
COM	Fin d'opération suivie immédiatement par Interruption.

Les dddd's sont la représentation hexadécimal du CSW stocké.
Le zzzz peut être :

CSWSNS = dddddddd dddddddd
(CSW stocké au SIO d'un SENSE)
CSWSNT = dddddddd dddddddd
(CSW stocké à la fin d'un SENSE)
SENSE = dd dd est l'octet de sondage.

Les messages CSWSNS= et CSWSNT= indiquent une erreur hardware 360, car une opération de sondage doit s'accomplir parfaitement.

Si le message SENSE=dd apparaît, les bits de dd, c'est-à-dire les bits de sondage, indiquent des informations supplémentaires sur la cause de l'erreur. La signification de ces bits apparaît ci-dessus :

Bit	Valeur (hexadécimal)	NOM	Signification
0	80	Command Reject	Erreur de programme 360
1	40	Command Interrupt	Interruption au milieu d'une commande
2	20	Bus-out Parity	Erreur hardware 360
3	10	-	-
4	08	Data Check	Erreur Parité
5	04	-	-
6	02	Incomplete Data Transfer	Transfert de données Incomplet
7	01	Timeout	Nom-réponse du PDP-8 après 2 secondes.

Le bit 4, avec la version actuelle de l'interface sur le PDP-8, devra toujours être zéro, car l'interface interdit toute indication d'erreur de parité. Les bits 3 et 5 doivent toujours être zéro.

APPENDICE B

BIBLIOGRAPHIE

BIBLIOGRAPHIE

ASSEMBLEURS PDP-8

D. CLAUZEL "Caractéristiques de l'assembleur PDP-8 sur 7044 et langage de commande" - IMAG, Novembre 1966.

BERGER et JAN "Assembleur PDP-8 sur 360"
IMAG - 1968-1969 (Projet).

PROGRAMMATION DU PDP-8

Brochures disponibles de : Program Library, Digital Equipment.
Corporation, Maynard Mass. U.S.A. :

Small Computer Handbook	
Disk Monitor System	DEC-D8-SDAB-O
Editor	DEC-D8-ESAB-D
PALD Assembler	DEC-D8-ASAA-D
ODT writeup	DEC-D8-COCO-D

D. TUFFELLI et B. VUILLOD "Description sommaire de la liaison entre le PDP-8 et l'IBM 360/67"
Note Technique, IMAG.

PROGRAMMATION DU 360

Brochures IBM :

"Principles of Operation" A22-6821

"Assembler Language" C28-6514

"Assembler(F) Programmer's Guide C26-3756

"Supervisor and Data Management Service" C28-6646

"Supervisor and Data Management Macros" C28-6647

"Job Control Language" C28-6539

IBM 2701 Data Adapter Unit (OEMI) A22-6844

IBM 2701 Data Adapter Unit Component Description GA22-6864

"CP-67/CMS User's Guide".

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