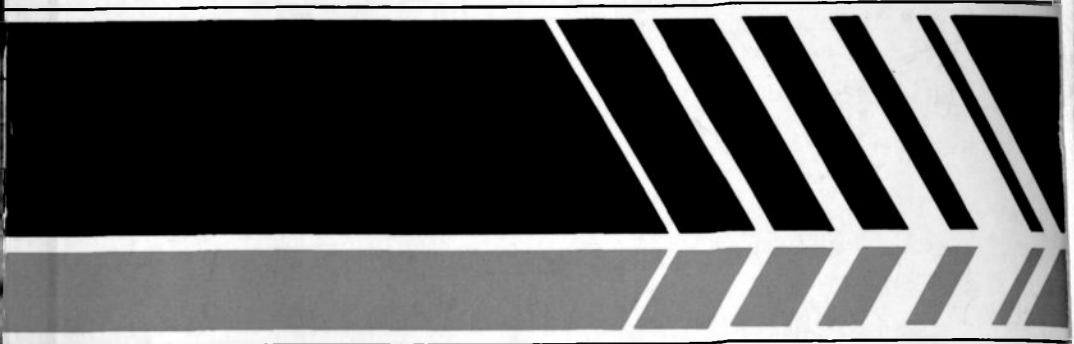
le système PROLOGUE

version 2.2

par Pierre Giraud



éditests

le système PROLOGUE

version 2.2

DANS LA MEME COLLECTION :

□ Série « système d'exploitation »

- **Le système UNIX** - utilisation des commandes par Violaine Prince
- **Le système CP/M pour Z-80** - adaptation du BIOS et compléments par Fabienne et Philippe Gysel
- **Le système CP/M pour 8080** - utilisation et programmation en version 2.2 par Jacques Pinto
- **Le système PASCAL UCSD** - 1/ organisation générale par Thierry Chamoret
- **Le système PASCAL UCSD** - 2/ structure interne par Thierry Chamoret
- **Le concept FORTH** - langage et système par Pascal Courtois
- **Le système MemDos** - conception de programmes par Pierre Clerc
- **Systèmes PC-DOS et MS-DOS - version 2** - introduction et utilisation avancée par Jacques Boyer, Jean-Pierre Lamoitier et Michel Treillet

□ Série « langage de programmation »

- **Le langage ADA** par Daniel-Jean David
- **Le langage APL** par Daniel-Jean David
- **Le langage C** par Jean-Louis Fourtanier et Violaine Prince
- **Le langage FORTRAN** par Daniel-Jean David
- **Le langage D-PROLOG** - initiation au langage de la 5ème génération par Philippe Donz et Rosalie Hurtado
- **Le langage B.A.L.** - initiation et pratique par Didier Lévy

□ Série « communication »

- **Les réseaux locaux d'entreprise** - marchés et technologies par Frédéric Hoste
- **Les techniques de la télématique** par Jérôme Toussaint et Philippe Masson

□ Série « dialogue homme-machine »

- **Synthèse, reconnaissance de la parole** par Marc Ferretti et François Cinare
- **La synthèse d'image** par Francis Martinez

□ Série « informatique »

- **Les opérations arithmétiques dans les ordinateurs** par Ioan Dancea

□ Série « électronique numérique »

- **Choisir un système de développement pour microprocesseurs** par Michel Blanchard, Jean-Claude Cavarroc et Michel Gay
- **Les systèmes à microprocesseurs** par Daniel-Jean David
- **Les circuits programmables** par Jean-Michel Bernard et Henri Breteuil
- **Microprocesseurs et circuits associés** par Roland Dubois
- **Mise en oeuvre du bus IEEE 488** - utilisation et réalisation d'appareils par Gérard Bastide et Jean-René Vellas

□ Série « productique »

- **Les systèmes industriels d'intelligence artificielle** - outils de productique par Lucas Pun
- **Introduction à la robotique** - 1/ notions de base, architecture, systèmes actionneur et sensoriel, modes de fonctionnement par Pierre Lopez et Jean-Numa Foulc

□ Série « bandes dessinées »

- **Les ordinateurs... Bonjour !** par Zévar

le système PROLOGUE

version 2.2

par
Pierre Giraud

éditests

1984

A ma femme

Remerciements

L'auteur remercie l'équipe PROLOGUE (division de Bull Micral) pour sa collaboration et l'aide efficace qui lui a été fournie.

© Editests - 5, place du Colonel Fabien - 75491 Paris cédex 10 - 1984
ISBN 2-86699-027-7
P.S.I. Diffusion -BP 86 - 77402 Lagny-sur-Marne Cédex
Tel. (6) 006 44 35

Imprimé en France

sommaire

AVANT-PROPOS 9

INTRODUCTION 11

Principe de conception de PROLOGUE 11
Principales caractéristiques 12
Portabilité du logiciel 13
Modularité du logiciel 14
Méthodes d'accès évoluées 14
Outils de base 15

CHAPITRE 1 LES VOLUMES ET LES FICHIERS 17

Généralités 17
Les ressources de PROLOGUE 18
Désignation des fichiers 18
Notion de TYPE de fichier 18
Notion de CLES D'ACCES 19

CHAPITRE 2 LE SYSTEME DE FICHIERS 21

Généralités 21
Accès séquentiel indexé 22
Le multicritère 23
La base de données relationnelle 26
Définition du modèle relationnel 27
Caractéristiques optimales de la base de données 28
Les jointures 29

CHAPITRE 3 LES UTILITAIRES DE PROLOGUE 31

Catalogue des fichiers - CAT 34
Catalogue sélectif des fichiers 35
Edition sur imprimante 36
Gestion de la date - DATE 38
Visualisation et modification des implicites - VMPI 40

Manipulation de volumes et de fichiers - CP 42
Création d'un volume 43
Duplication d'un volume 45
Changement du nom d'un volume 47
Duplication de fichier(s) 48
Suppression de fichier(s) 50
Renommer un fichier 51
Copie de supports sectorisés 52
Copie partielle de supports sectorisés 52
Considérations sur la nature des disquettes 54
Edition de texte source - EDV 56
Les mouvements dans le fichier 57
Les mouvements curseur 58
Les suppressions 58
Autres commandes 59
Mode commande 60
Sauvegarde de fichiers - SV 62
Restauration de fichiers - RT 64
Sauvegarde de fichiers séquentiels indexés - SVSI 65
Restauration de fichiers séquentiels indexés - RTSI 67
Modification de support sectorisé - PATCH 68
Les commandes 69
Utilisation des fichiers de commandes - ASG 73
Commandes paramétrées 73
Exécution implicite d'un fichier de commandes 75
Impression des fichiers - SPOOL 77
Activation du système d'impression 78
Insertion d'un fichier liste 78
Abandon d'une liste en cours 79
Inhibition du système de gestion des impressions 79
Etat des périphériques sectorisés - STATUS 81
Interprétation du tableau 81
Transmission de fichiers - TELE 84
Quelques mots sur le protocole de transmission 85
Réorganisation de volume - REORG 87

CHAPITRE 4 LES DECORS 91

Chargement des DECORS 92
Appel du DECOR MS-DOS 92
Appel du DECOR CP/M-86 92
Principe de fonctionnement 93
Mise en oeuvre des DECORS 94
Utilisation des caractères nationaux 96
Partage des fichiers en multiposte 96
Réservation de mémoire 97
Paramètres de DECOR 98
Mémoire allouée au DECOR 98
Table de transcodage des caractères accentués 98

Table des types de fichiers partageables 98
Localisation des paramètres DECOR 99
Valeurs standard par défaut 100
Transfert de fichier CP/M-86 100
Transfert de fichiers MS-DOS 101
Exemples d'utilisations 102
Transferts de fichier CP/M-86 102
Transferts de fichiers MD-DOS 102
Liste de logiciels sous DECOR 103
Logiciels fonctionnant sous DECOR CP/M-86 103
Logiciels fonctionnant sous DECOR MS-DOS 103

CHAPITRE 5 LE LANGAGE BAL 105

Structure d'un programme BAL 106
Instructions générales et branchements 108
Directives et instructions de contrôle 109
Fonctions arithmétiques 110
Fonctions mathématiques 111
Fonctions sur chaînes de caractères 112
Fonctions système 113
Edition et saisie de données 114
Format de saisie/d'édition de données 115
Instructions relatives aux fichiers 116
Instructions relatives aux fichiers séquentiels indexés 117
Instructions relatives aux fichiers multicritères 118
Informations statistiques 118
Instructions relatives à la base de données relationnelle 120
Opérateurs relationnels utilisés par le multicritère et la base de données 121
Instructions de gestion des enregistrements 122
Instructions graphiques 123
Directives du BAL 124
Mise au point d'un programme BAL 125
Liste des commandes sous DEBUG 125

CHAPITRE 6 L'ORGANISATION DU SYSTEME 127

Les extensions du système 128
Structure du noyau environnement 129
Structure du noyau système 129
Présentation du disque système 130
Paramètres de configuration 131
Les paramètres généraux 131
Les paramètres extensions 131
Configuration de l'environnement 131
Configuration du noyau système 131
Disposition des paramètres de configuration 132
Définition des paramètres généraux 134

CHAPITRE 7	MODULES D'EXTENSIONS	137
	Déclaration d'un module extension	**137**
	Description des paramètres	**139**
	Paramètre numérique	139
	Paramètre littéral	139
	Paramètre de lien	140
	Exemples de déclarations	**141**
	Modules environnement	141
	Contrôleur disque optionnel	141
	Contrôleur d'imprimante parallèle	141
	Contrôleur d'imprimante série	141
	Système complet de gestion de deux imprimantes	141
	Contrôleur TTY gérant des lignes asynchrones	142
	Contrôleur mémoire	142
	Modules système	143
	Déclaration d'une méthode d'accès NGF	143
	Déclaration d'une procédure de télétransmission asynchrone	143
	Structure du fichier SYSCONF-S	**143**
	Traitement du fichier SYSCONF	**144**
CHAPITRE 8	LE GRAPHIQUE SOUS PROLOGUE	145
	Description des dessins	**145**
	Espace de travail	145
	Espace de visualisation	146
	Fenêtre et clôture	147
	Ordres de tracés	**151**
	Tracé de lignes	151
	Tracé de symboles	152
	Les attributs de tracés	**152**
	Attributs de couleur	152
	Attributs de type	153
	Attributs de taille	153
CHAPITRE 9	STRUCTURE DES VOLUMES	155
	Gestion de l'espace disque	**155**
	Table d'allocation d'un volume	156
	Organisation du répertoire	**156**
	Informations concernant le volume	157
	Taille du granule (LGR)	157
	Taille du répertoire (LCAT)	158
	Identificateur du répertoire (IND)	158
	Structure du répertoire	159
	Structure des descripteurs noms de fichiers	**161**
	Structure 7-1	163
	Structure 8-3	164
	Notes sur les descripteurs	165
	Utilisation des zones réservées	**166**

CHAPITRE 10	PROGRAMMATION SYSTEME	167
	Avertissement	**167**
	Primitives système	**168**
	Le moniteur multitâche	**170**
	Les tâches immédiates	170
	Les tâches de fond	170
	Primitives du moniteur multitâche	173
	Accès aux ressources disque	**173**
	Primitives du système de fichiers	175
	Principe général de l'appel	**176**
	Structure des programmes	**178**
	Aide à la mise au point	**179**
	Liste des commandes de MM	180
	Exemples de commandes	181
CHAPITRE 11	TELECOMMUNICATIONS	183
	Généralités	**183**
	Notion de réseau	**184**
ANNEXE 1	LES CODES REPONSES	187
	Périphérie	**189**
	Noyau système	**190**
	Commande	**192**
	Système de fichiers	**193**
	Codes erreurs du séquentiel-indexé	**196**
	Codes erreurs du multicritère	**197**
	Codes erreurs de la base de données	**199**
ANNEXE 2	STANDARD DES CODES FONCTIONS	201
	Fonctions d'entrée/sortie	**201**
	Autres fonctions	**201**
	Gestion du curseur	**202**
	Attributs d'impression	**203**
	Attributs vidéo	**203**
ANNEXE 3	EXEMPLES DE PROGRAMMES BAL	205
ANNEXE 4	JEU DE CARACTERES PROLOGUE	209
ANNEXE 5	LISTE DES MOTS-CLES DU BAL	211

avant-propos

PROLOGUE est un système d'exploitation français initialement développé par la société **R2E** devenue depuis **Bull Micral**.

Conçu en 1979 pour habiller les matériels 8 bits de la famille Micral 80 commercialisés par R2E, il a, sous la poussée des utilisateurs, migré sur un nombre important de micro-ordinateurs 8 et 16 bits, hors du groupe Bull.

Grâce à la diffusion des **Micral** et des **Questar/M** par le réseau **Bull**, et aussi à la collaboration de certains constructeurs, PROLOGUE connut rapidement une large audience au niveau européen.

La mutation des micro-processeurs 8 bits (8080,Z-80) vers les 16 bits (8086), en 1982, a permis à PROLOGUE de se libérer du carcan de la taille mémoire et d'atteindre un niveau élevé de fonctions évoluées comme la base de données, la gestion graphique, l'accès X25 et le réseau local, fonctions totalement étrangères, en grande majorité, à des systèmes d'exploitation concurrents pourtant réputés comme les standards du marché.

Le terme "**PROLOGUE**" (ne pas confondre avec le langage de programmation **Prolog**, lui aussi français) ne désigne pas seulement un système d'exploitation, mais plutôt un ensemble cohérent de fonctions et de logiciels.

Système d'exploitation multitâche, mono ou multiposte, d'une structure modulaire et portable, avec ses **DECORS**, son langage **BAL**, ses méthodes d'accès aux fichiers évoluées, ses possibilités en matière de télécommunication, **PROLOGUE** peut se targuer d'être aujourd'hui un système d'exploitation de convergence unique en son genre.

Cet ouvrage, le premier d'une série consacrée à ce système d'exploitation, n'est pas un manuel de référence, mais plutôt une synthèse des fonctions essentielles de **PROLOGUE** et de son environnement que l'auteur se propose de vous faire découvrir.

introduction

Le domaine de **PROLOGUE** est étendu et varié. A travers ses **décors** CP/M 86 et MS/DOS, **PROLOGUE** accède à la quasi totalité des bibliothèques de progiciels liées à ces deux références en micro-informatique.

Ses fonctionnalités sont toutefois plus développées et normalisées que celles de ses homologues américains puisqu'il intègre de nombreuses méthodes d'accès aux fichiers, comme le séquentiel indexé, le multicritère, la base de données relationnelle.

Il s'agit, d'autre part, d'un système d'exploitation fondamentalement multitâche et multiposte. Ce dernier aspect est essentiel car il permet l'intégration de procédures de télécommunications.

Comparé à de nombreux produits disponibles actuellement sur le marché, **PROLOGUE** est le seul système d'exploitation dont l'environnement télécommunication permet de développer des applications personnalisées telles que : émulation de terminaux, transfert de fichiers, conversationnel avec traitement local de l'information, dialogue entre applications, etc.

Le langage **BAL** qui est le langage évolué de base sous **PROLOGUE** et dont les performances et les caractéristiques sont hors du commun mérite une attention particulière, un chapitre lui est consacré.

Le progiciel **DIALOGUE** entièrement écrit en **BAL** est une base de données qui permet de créer, mettre à jour, interroger jusqu'à dix fichiers dans une même application.

PRINCIPES DE CONCEPTION DE PROLOGUE

PROLOGUE est constitué d'un ensemble de modules dont les fonctionnalités et le mode d'utilisation (interfaces) sont clairement spécifiés.

Il admet, de ce fait, des environnements matériel les plus simples aux plus étoffés (disques durs, réseau local ...) et peut être adapté facilement à toute configuration particulière. Certaines de ces fonctionnalités

sont totalement étrangères à des systèmes d'exploitation réputés comme les standards du marché.

Deux niveaux définissent le système PROLOGUE.

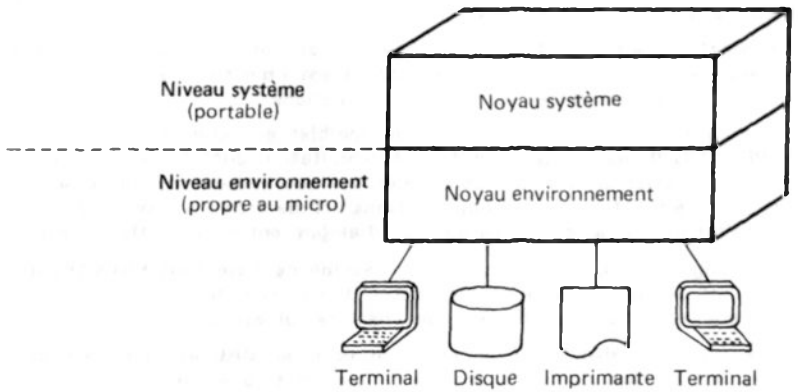
Niveau système :
c'est la **partie portable** de PROLOGUE totalement indépendante du matériel :

- le moniteur multitâche ;
- l'interpréteur de commandes ;
- le noyau de gestion de fichiers ;
- une bibliothèque de primitives système accessibles au programmeur.

Niveau environnement :
c'est la **partie non portable** de PROLOGUE que l'on peut assimiler au BIOS des autres systèmes d'exploitation :

- gestion de la console clavier-écran ;
- gestion des disques souples de base ;
- caractéristiques de l'environnement matériel.

La concaténation de ces deux niveaux constitue le système PROLOGUE.



PRINCIPALES CARACTERISTIQUES

Outre celles que nous venons d'évoquer, PROLOGUE se distingue par un certain nombre de caractéristiques particulières :

- **Multitâche** : deux niveaux de priorité (tâches de fond et tâches prioritaires), primitives de synchronisation et de communication.

- **Multiposte** : gère de un à huit postes indépendants avec, pour chacun, des autorisations d'exploitation particulières et possibilité de commandes initiales.
- **Gestion dynamique de la mémoire** : pour des configurations allant de 128 K octets à 1 M octet, partageables en partitions indépendantes et avec allocation dynamique de segments de taille variable.
- **Gestion d'une horloge temps réel** : exploitée par le multitâche, elle sert également à entretenir l'heure et la date.
- **Système d'enchaînement de commandes** : pour mettre en oeuvre des procédures cataloguées, paramétrables et récursives.
- **Gestionnaire d'impression** : permet un partage de la ressource imprimante dans une configuration multiposte.
- **Gestion dynamique des fichiers** : par allocation dynamique de l'espace disque, peut gérer jusqu'à **895** fichiers sur un support, la taille des fichiers pouvant aller jusqu'à **256** M octets. Protection des fichiers par des clés d'accès en lecture et écriture. Ouverture des fichiers en mode exclusif ou partageable.

PORTABILITE DU LOGICIEL

La portabilité des applications est un des critères fondamentaux que se sont attachés à respecter les concepteurs de PROLOGUE par le biais du langage **BAL** au niveau du code traduit (T-code).

Pour un logiciel système exécutable sur n'importe quel matériel, la portabilité totale n'existe pas sans modification préalable.

Cela tient au fait qu'il n'y a pratiquement pas de systèmes compatibles au plan matériel à 100 %.

Cela provient également de ce que les adaptateurs périphériques utilisés, en particulier pour la gestion des lignes de transmission, sont spécifiques à chaque matériel et que le logiciel doit s'y conformer.

Construire un logiciel portable consiste donc à bien cerner et à minimiser cette part typiquement dépendante du matériel.

Pour y aboutir, les logiciels système qui constituent PROLOGUE sont toujours composés de deux parties :

- un ou plusieurs modules indépendants de la machine cible ;
- un module d'adaptation spécifique à la machine cible.

Ce dernier module appelé "contrôleur", aussi réduit que possible, est le seul qu'il soit nécessaire d'adapter au matériel.

Sa complexité est variable selon la nature du coupleur à gérer (disques souples, interface parallèle, interface série, écran alphanumérique ou graphique, etc.).

Sans intervenir dans le système standard, l'utilisateur peut développer un contrôleur spécifique et l'intégrer dynamiquement dans le système à l'initialisation, ce qui confère à PROLOGUE une grande ouverture à des environnements matériels divers.

MODULARITE DU LOGICIEL

La mise en place de ces modules dans le système se fait aisément par autoconfiguration.

Un fichier **SYSCONF-S** contient la description des modules dont on veut disposer.

Au chargement du système, les éléments décrits sont chargés en mémoire pour être liés au noyau résident.

Ce procédé, très simple, permet à quiconque d'adapter finement la composition d'un système PROLOGUE pour ses besoins propres.

METHODES D'ACCES EVOLUEES

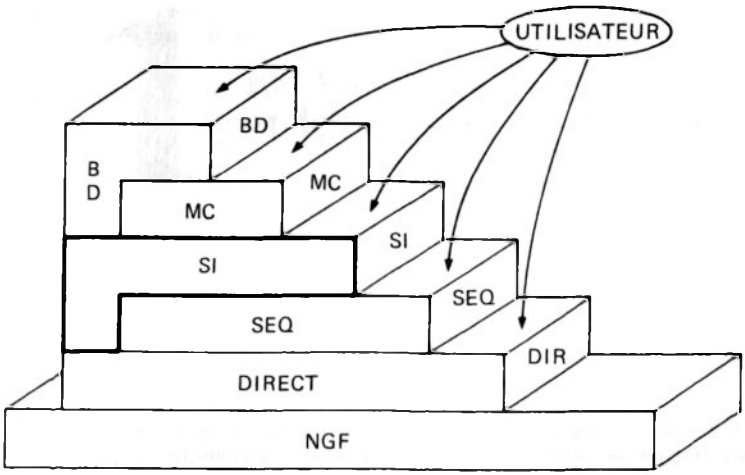
Le système de gestion de fichiers (**SGF**) évolué de PROLOGUE, avec des méthodes d'accès multiples et hiérarchisées, fait partie intégrante du système de base.

Le SGF permet de travailler sur des matériels de configurations différentes d'une façon totalement transparente. De plus, la gestion des accès simultanés à un même fichier ou au même article est également assurée par le SGF.

Les deux méthodes d'accès les plus élémentaires sont l'accès virtuel qui définit des fichiers de variables de tous types, utilisées de la même manière que des variables en mémoire centrale, et l'accès séquentiel qui supporte les fonctions de lecture, modification et ajout.

Au dessus, se greffent de manière hiérarchique, des méthodes d'accès évoluées : le séquentiel-indexé, l'accès multicritère, base de données.

Ce sont toutes ces méthodes d'accès qui sont la richesse première de PROLOGUE : nous y reviendrons plus loin.



Les méthodes d'accès évoluées : une "architecture en couches" accessibles à tous les niveaux

OUTILS DE BASE

Appelés aussi **utilitaires** ou **commandes**, nous les détaillerons par ailleurs ; ils ont une syntaxe d'appel au niveau de l'interpréteur de commandes relativement simple.

CHAPITRE 1

les volumes et les fichiers

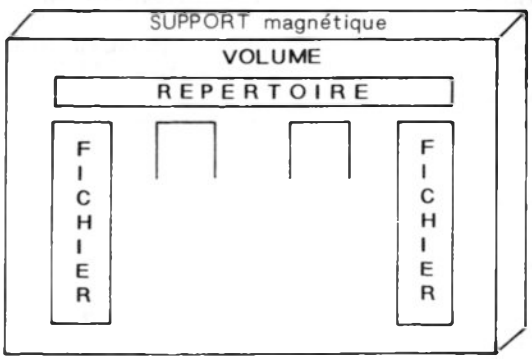
GENERALITES

Un **FICHIER** est un ensemble d'informations organisé logiquement et mémorisé dans un **SUPPORT** altérable (généralement disques magnétiques souples ou durs, fixes ou amovibles et de dimensions variables).

Sous PROLOGUE, cet ensemble représente un **VOLUME** d'informations.

Pour permettre l'accès à un fichier et aux informations qu'il contient, le volume dispose d'un **REPERTOIRE** (appelé aussi directory ou catalogue sur d'autres systèmes d'exploitation).

Outre les noms des fichiers, le répertoire contient également des informations décrivant les caractéristiques du volume et des fichiers.



PROLOGUE gère des **SUPPORTS** magnétiques représentant des **VOLUMES** d'informations. Ces informations sont contenues dans des **FICHIERS** cités dans des **REPERTOIRES**.

LES RESSOURCES DE PROLOGUE

Deux types de ressources sont gérées par PROLOGUE :

- les ressources physiques qui désignent un périphérique. **MD0, FL1, DS1, LA2** sont des ressources physiques.
- les ressources logiques définies par le système et qui sont assignées par configuration ou par l'utilisateur à une ressource physique. **CI, CO, IS, LO** sont des ressources logiques.

PROLOGUE reconnaît les ressources suivantes :

CI	console entrée (clavier) attachée au poste appelant.
CO	console sortie (écran) attachée au poste appelant.
LO	imprimante implicite attachée au poste appelant.
IS	disque implicite système.
IP	disque implicite programme.
IF	disque implicite fichier attaché au poste appelant.
CS	ressource réservée.

Résumé

Une **RESSOURCE** est désignée par un **NOM**. Elle fait référence à :
- un élément **PHYSIQUE** lié à l'environnement du système ;
- un élément **LOGIQUE** défini par le système PROLOGUE et assigné à un élément physique.

DESIGNATION DES FICHIERS

Notion de TYPE de fichier

Une information supplémentaire située en extension du **NOM** d'un fichier permet d'en définir la nature.

Cette information désigne le **TYPE** du fichier, elle intervient dans la désignation d'un fichier.

DEMO-**S** DIALOGUE-**T** PATCH-**X** BASIC-**COM** ESSAI-**BAS**

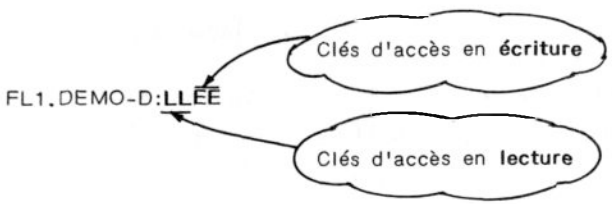
L'utilisateur a toute liberté pour choisir le suffixe qui caractérise le type de ses fichiers. Cependant, quelques suffixes sont réservés à certains types de programmes, il est recommandé de respecter ces règles.

-X	**Programme exécutable** directement sous PROLOGUE. Il s'agit en général d'un fichier contenant un programme écrit en langage assembleur.
-S	**Source.** Il contient du code ASCII produit par un programme tel qu'un éditeur de texte par exemple.
-T	**Traduit.** Il contient le code intermédiaire engendré par le traducteur BAL. Ce fichier ne peut être chargé et exécuté que par l'exécuteur BAL.
-I	**Indexes.** Utilisé conjointement au fichier -D de même nom, par la méthode d'accès séquentiel-indexé. Il contient les indexes des données du fichier -D.
-D	**Données.** Il s'agit d'un fichier engendré ou utilisé par les méthodes d'accès aux fichiers.
-R	**Répertoire.** Utilisé par la méthode d'accès multicritère.
-V	**Valeurs.** Utilisé par la méthode d'accès multicritère.
-U	**Sauvegarde.** Utilisé par les utilitaires SVSI, RTSI.
-$	**Temporaire.** C'est un fichier créé par certains utilitaires de PROLOGUE. Ce suffixe n'apparaît en principe pas lors de l'édition du répertoire, car provisoire.
-ASG	**Fichier de commandes.**

Notions de CLE D'ACCES

Les clés d'accès ont pour objectif de protéger certains fichiers contre des opérations intempestives ou non autorisées.

Elles conditionnent les accès en lecture (et/ou) en écriture.



Si la clé ne comporte que les deux premiers caractères (LL), une spécification correcte de la clé autorisera la lecture, l'écriture et la suppression du fichier.

Si la clé est de la forme bbEE (bb désigne deux espaces), le fichier est en protection écriture. La lecture est toujours possible.

Si la clé est complète (LLEE), la seule spécification de la partie LL n'autorisera que la lecture. La spécification complète de la clé (LLEE) autorisera la lecture, l'écriture et la suppression du fichier.

Les clés ne sont pas éditées lors de la constitution du répertoire.

La clé ne participe pas à l'identification d'un fichier.

Résumé

[RRu.] FF..FF [-TTT] [:LLEE]

RR mnémonique désignant le nom de la ressource périphérique.

Le mnémonique est défini par le module de l'environnement gérant la ressource.

FL floppy
MD mini disque dur

D'autres mnémoniques peuvent être définis, ils dépendent du type de périphérique géré par le contrôleur de l'environnement.

u. désigne le numéro de l'unité adressée. C'est un chiffre de 0 à 9.

FL0 désigne le lecteur de disque numéro **0**.
MD1 désigne le mini disque numéro **1**.

FF.. désigne le nom du fichier. C'est une chaîne de un à huit caractères ASCII.

DEMO **DEMO/01** **//DEMO**

sont des noms de fichiers valides.

-TTT désigne le type de fichier. C'est une chaîne de zéro à trois caractères alphabétiques.

FICHDEM-**BAS** FL1.F/001-D SOURCE-**ASM**

sont des noms de fichiers valides.

:LLEE désigne les clés d'accès au fichier.

LL clé accès lecture.
EE clé accès écriture.

CHAPITRE 2 | le système de fichiers

GENERALITES

Le Système de Gestion des Fichiers (**SGF**) de PROLOGUE est constitué par une succession de couches superposées que l'on appelle des méthodes d'accès.

Le SGF permet de travailler sur des matériels de configurations différentes et ce d'une façon identique quels que soient les périphériques utilisés.

De plus, le SGF assure une gestion des accès simultanés à un même fichier ou au même article.

Il offre deux méthodes d'accès élémentaires qui sont :

- l'accès **VIRTUEL** ou **DIRECT** exploité essentiellement par le langage BAL ; il définit des fichiers de variables de tous types utilisées de la même manière que des variables en mémoire centrale ;
- l'accès **SEQUENTIEL** qui supporte les fonctions de lecture, écriture, modification.

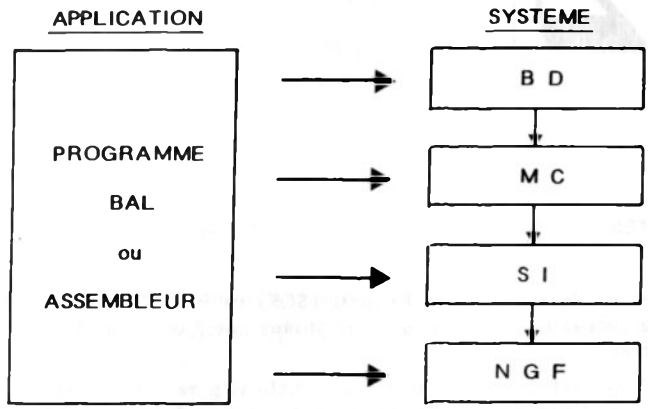
Au-dessus, viennent se greffer de manière hiérarchique des méthodes d'accès évoluées :

- l'accès **SEQUENTIEL-INDEXE** ;
- l'accès **MULTICRITERE** ;
- l'accès **BASE DE DONNEES.**

Au niveau le plus bas de cette architecture, se trouve le **N**oyau de Gestion de **F**ichiers (**NGF**) qui regroupe l'ensemble des fonctions élémentaires d'une gestion de fichiers sur support sectorisé à accès sélectif (disquette, disque dur fixe ou amovible). Ce module fait partie du noyau de PROLOGUE.

Résumé

C'est à partir du NGF que sont bâties les méthodes d'accès évoluées qui forment toutes des modules d'extension du NGF.



ACCES SEQUENTIEL INDEXE

La méthode d'accès **S**équentiel-**I**ndexé (**SI**) est basée sur une structure arborescente du type arbre binaire équilibré. Elle consiste à associer à un enregistrement contenant des données, une **CLE** univoque.

L'exemple le plus courant est la méthode que nous utilisons tous pour consulter un dictionnaire.

C'est sur ce principe que s'appuient les méthodes d'accès plus évoluées telles que :

MULTICRITERE et **BASE DE DONNEES**

Dans un fichier séquentiel-indexé, les clés ont une longueur fixe de **2** à **51** octets, les données (facultatives) ont une longueur qui peut aller jusqu'à **32 000** caractères, en longueur fixe ou variable.

Un fichier séquentiel-indexé se compose de deux fichiers séparés, de même nom générique, situés sur le même volume et identifiables par leur suffixe :

- le fichier d'index (**-I**) qui contient les clés ;
- le fichier des données (**-D**) qui contient les articles.

Le fichier d'index (**-I**) contient toujours la totalité des clés enregistrées dans le fichier. Chaque clé, qui doit être unique, est associée dans l'index à un pointeur vers l'article correspondant dans le fichier des données. Ce pointeur, exprimé en adresse relative par rapport au début du fichier, garantit l'indépendance de la position des fichiers **-I** et **-D**.

Le nombre de clés que peut contenir l'index n'est pas limité et le nombre de niveaux dans l'arbre n'est déterminé que par la longueur des clés définie par l'utilisateur, chaque noeud de l'arbre ayant une longueur fixe de 256 octets.

Le fichier des données (**-D**) contient tous les articles présents à un moment donné dans le fichier. Ils sont enregistrés les uns derrière les autres dans l'ordre de leur insertion.

Lors de la suppression d'un article, la place libérée est récupérée. L'adresse relative de l'article est insérée dans une liste de "trous", cette même liste est utilisée en priorité à chaque nouvelle insertion.

Un fichier Séquentiel-Indexé peut être ouvert selon deux modes :

EXCLUSIF ou **PARTAGEABLE**

En mode partageable, l'utilisateur peut toujours se réserver l'accès exclusif à une clé et son article associé par un mécanisme de blocage des articles clé par clé. Cette méthode permet de gérer les accès concurrents à un même fichier pendant les mises à jour.

En consultation, l'accès s'effectue directement à partir de la clé ou séquentiellement dans l'ordre croissant ou décroissant par rapport au pointeur courant, c'est-à-dire la dernière clé lue.

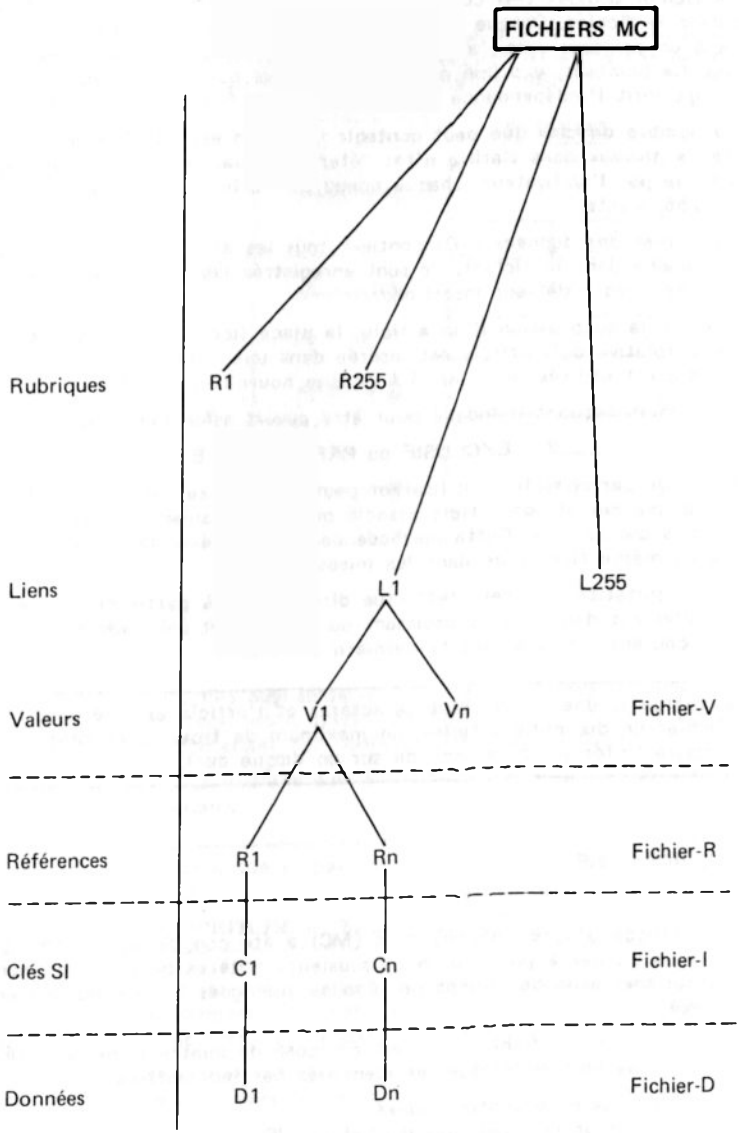
> Pour retrouver une clé de **huit** caractères et l'article associé dans un fichier de **dix mille** articles, un maximum de **trois** accès sont nécessaire (inférieur à la seconde sur un disque dur).

LE MULTICRITERE

La méthode d'accès MultiCritère (**MC**) a été conçue pour permettre l'accès à un fichier à partir d'un ou plusieurs critères de sélection non discriminants et avec des temps de réponse analogues à ceux du Séquentiel-Indexé.

Physiquement, un fichiers MC est composé de quatre fichiers différents, de même nom générique, et identifiés par leur suffixe.

-I et **-D**	idem séquentiel-indexé.
-V	pour le répertoire du fichier MC.
-R	pour les index secondaires.


FICHIERS MC
Rubriques
R1
R255
Liens
L1
L255
Valeurs
V1
Vn
Fichier-V
Références
R1
Rn
Fichier-R
Clés SI
C1
Cn
Fichier-I
Données
D1
Dn
Fichier-D

-V et **-R** sont des fichiers **SI** sans données, la totalité de l'information étant codée dans la clé.

Le répertoire du fichier MC (**-V**) décrit la structure des articles et les index secondaires appelés liens ou fichiers inverses.

> Un article est décomposé en **RUBRIQUES.** Une rubrique est définie par :
> - un **NOM** unique ;
> - un **TYPE** (numérique, texte) ;
> - une **LONGUEUR.**

Cette description de l'article est effectuée par l'utilisateur lors de la création du fichier en fonction de ses besoins et compte tenu de la longueur de l'article qu'il a défini.

Les rubriques sont décrites dans l'ordre où elles se trouvent dans l'article, cependant l'utilisateur n'est pas tenu de les décrire dans leur totalité. Ainsi, il peut à tout moment ajouter une nouvelle description si toutefois il reste suffisamment de place dans l'article.

Il faut noter aussi que les rubriques peuvent être renommées ou redéfinies en plusieurs sous-rubriques.

Toutes les rubriques d'un fichier MC sont interrogeables ensemble ou séparément.

A cette fin, l'utilisateur a la possibilité de décrire un **LIEN** sur ces rubriques.

Un lien représente en fait un type d'interrogation et correspond à un index secondaire.

> Un **LIEN** est constitué de **un** à **dix** noms de rubriques. **255** liens différents peuvent exister. Tous les liens sont enregistrés dans le répertoire.

Pour permettre l'interrogation sur n'importe quelle rubrique du lien dans tous les sens, quel que soit l'ordre de description, l'utilisateur peut demander au MC de calculer et de générer les liens nécessaires.

L'interrogation d'un fichier MC s'effectue à l'aide d'une liste de **un** à **dix CRITERES** de sélection

Un **CRITERE** est composé :

- d'un **NOM de RUBRIQUE**
- d'un **OPERATEUR de COMPARAISON**

'<' '=' '>' '<=' '>=' '<>' **(différent de)**
'(' compris entre deux bornes incluses ')'

d'une **VALEUR.**

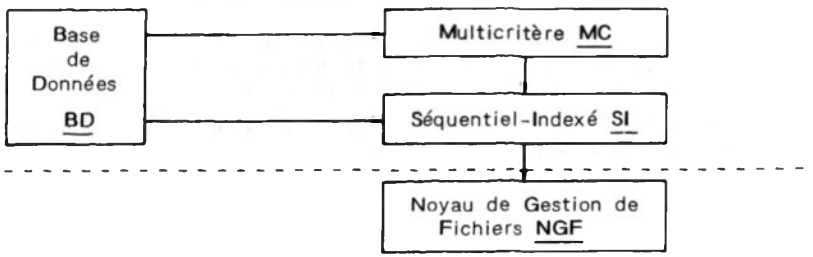
Le nombre de caractères de la VALEUR peut être inférieur à la longueur de la rubrique associée. En particulier, il n'est pas nécessaire de préciser les espaces situés à droite (rubrique de type texte) ou les zéros à gauche (rubrique numérique).

Les temps de réponse obtenus pour une question portant sur **quatre** critères dans un fichier de **27 000** articles sont inférieurs à **trois** secondes pour la première réponse et de l'ordre d'**une demie** seconde pour les suivantes.

LA BASE DE DONNEES RELATIONNELLE

Le module **BD-X** de PROLOGUE est un système de gestion de Base de Données, de type **RELATIONNEL**. Il permet d'accéder à plusieurs fichiers en qualifiant le résultat recherché sans décrire le chemin d'accès à utiliser.

C'est sur la méthode d'accès multicritère que la Base de Données s'appuie entièrement pour gérer ses fichiers ; de ce fait, la totalité des fonctions du Multicritère est disponible au niveau de **BD**.



Elle peut regrouper jusqu'à **255** fichiers, interrogeables ensemble ou séparément à partir d'une question à N critères et de complexité quelconque.

Les fichiers de la Base sont des fichiers Multicritère de longueur fixe. Leurs caractéristiques (longueur de la clé, des données, nombre de rubriques,...) peuvent être différentes pour chaque fichier.

Chaque question peut porter sur N fichiers et ne demande aucune connaissance quant à leur structure ; il suffit d'exprimer les critères de sélection qui qualifient le résultat recherché. Le système évaluera alors la question et déterminera les fichiers concernés ainsi que le chemin à employer.

L'accès à la Base peut s'effectuer de deux façons :

Par un programme (assembleur, BAL,...).
Par le progiciel **DIALOGUE.**

Outre les primitives du système pour la programmation en langage assembleur de la méthode d'accès BD, le langage BAL intègre des verbes puissants pour chaque fonction de la Base.

L'accès à BD par d'autres langages tels que **BASIC, COBOL, PASCAL, C,...** est possible moyennant une interface écrite en assembleur.

Définition du modèle relationnel

Il s'agit d'un ensemble de concepts et de règles permettant de décrire des données.
Un modèle de description est généralement représenté par un formalisme graphique.
Il est mis en oeuvre à l'aide d'un langage de description de données.

Chaque donnée élémentaire est définie par son **NOM** et l'ensemble des **VALEURS** qu'elle peut prendre. Ces données sont reliées entre elles par des **RELATIONS.**

Une relation peut être représentée sous la forme d'un tableau.

Chaque colonne du tableau représente une donnée et s'appelle un **ATTRIBUT** de la relation.

Une ligne du tableau est constituée par l'ensemble des valeurs respectives de chaque attribut et représente l'article de la relation.

L'ensemble de ces relations constitue la **BASE de DONNEES.**

Caractéristiques optimales de la base de données

Nombre maximum de fichiers	255
Nombre maximum de rubriques par fichier	255
Nombre maximum de jointures	255
Nombre maximum de fichiers liés par une jointure	10
Nombre maximum de critères par question	10
Nombre maximum de critères par lien	10
Longueur maximum d'une clé SI	51
Longueur maximum d'une clé MC	40
Longueur maximum d'une valeur MC	40
Nombre maximum d'articles	**capacité des disques**
Longueur maximum d'article	**mémoire disponible**
Longueur maximum d'une rubrique	**mémoire disponible**

*Fichier **CLIENT***

CLI-NR	NOM	VILLE	CODE-POSTAL
0001	DUPONT	MARSEILLE	13100
0002	GIRAUD	ST-LUBIN	28350
0003	JOUBERT	PARIS	75018
0004	GOURCY	MANTES	78310

*Fichier **FACTURE***

FACT-NR	NR-CLI	DATE	NR-PROD	QTE
0018	0003	06.06.84	3050	20
0022	0003	08.06.84	2100	5
0030	0001	12.06.84	1800	100
0045	0002	12.06.84	2100	10
0048	0004	20.06.84	2500	35

Fichier ***PRODUIT***

PROD-NR	DESIGNATION	PRIX
1800	grandiflora	24.50
2100	iris de hollande	35.00
2500	sedum spectabile	45.00
3050	oeillet mignardise	29.90
3100	rosiers polyanthas	57.00

Physiquement, une relation est implantée sous la forme d'un fichier.
Une relation désigne un **FICHIER.**
Un attribut désigne une **RUBRIQUE.**
Une ligne désigne un **ARTICLE.**

LES JOINTURES

Une **JOINTURE** est un fichier multicritère résultat de la concaténation des articles sélectionnés dans les fichiers.

Lors de la description d'une jointure, l'utilisateur donne un nom qui sera celui du fichier résultat.

Les critères de jointure sont précisés dans une liste dont chaque élément est composé de deux noms de rubriques liés par un opérateur de comparaison.

Nom de la jointure : "CLIFAC"

Description : "CLI-NR = NR-CLI"

La jointure **CLIFAC** permet de connaître le détail des factures de chaque client ayant passé une commande. Elle relie la rubrique **CLI-NR** du fichier **CLIENT** à la rubrique **NR-CLI** du fichier **FACTURE.**

Nom de la jointure : "CLIPRO"

Description : "CLI-NR = NR-CLI, NR-PROD = PROD-NR"

La jointure **CLIPRO** relie chaque client à ses factures et chaque facture d'un client aux produits vendus.
Elle relie la rubrique **CLI-NR** du fichier **CLIENT** à la rubrique **NR-CLI** du fichier **FACTURE** et la rubrique **NR-PROD** du fichier **FACTURE** à la rubrique **PROD-NR** du fichier **PRODUIT.**

Dans chaque élément, les rubriques doivent être de même type, de même longueur et appartenir à des fichiers différents.

Du fait de l'unicité des noms de rubriques, il n'est pas nécessaire de désigner les noms de fichiers concernés.

Résultat de la jointure ***CLIFAC***

CLI-NR	NOM	FACT-NR	DATE	NR-PROD	QTE
0001	DUPONT	0030	12.06.84	1800	100
0002	GIRAUD	0045	12.06.84	2100	10
0003	JOUBERT	0018	06.06.84	3050	20
0003	JOUBERT	0022	08.06.84	2100	5
0004	GOURCY	0048	20.06.84	2500	35

Résultat de la jointure ***CLIPRO***

CLI-NR	NOM	DATE	NR-PROD	QTE	DESIGNATION
0001	DUPONT	12.06.84	1800	100	grandiflora
0002	GIRAUD	12.06.84	2100	10	iris de hollande
0003	JOUBERT	06.06.84	3050	20	oeillet mignardise
0003	JOUBERT	08.06.84	2100	5	iris de hollande
0004	GOURCY	20.06.84	2500	35	sedum spectabile

La **JOINTURE** est l'opération qui permet d'établir le lien entre deux fichiers à partir d'une expression logique dans laquelle une rubrique du premier fichier est comparée à une rubrique du second.

CHAPITRE 3 | les utilitaires

PROLOGUE ne dispose pas de commandes résidantes, c'est-à-dire de fonctions opérateur intégrées au système lors du chargement.

> Sous PROLOGUE, une **commande** est un PROGRAMME ; c'est pourquoi nous utiliserons parfois, selon le contexte et surtout par habitude, la notion de commande plutôt que celle d'utilitaire.

CAT	ou "/", édition du répertoire des fichiers.
DATE	visualisation et mise à jour de la date.
VPMI	assignation de ressources implicites.
CP	création et manipulation de volumes/fichiers.
CPS	copie physique de supports sectorisés.
EDV	éditeur de texte source.
SV	sauvegarde de fichiers multi-volumes.
RT	restauration de fichiers multi-volumes.
SVSI	sauvegarde de fichiers séquentiels-indexés.
RTSI	restauration de fichiers séquentiels-indexés.
PATCH	modification de fichiers.
ASG	gestion de fichiers de commandes.
SPOOL	gestion de fichiers d'impression.
STATUS	édition status des périphériques sectorisés.
TELE	transmission de fichiers.
REORG	réorganisation de volume.

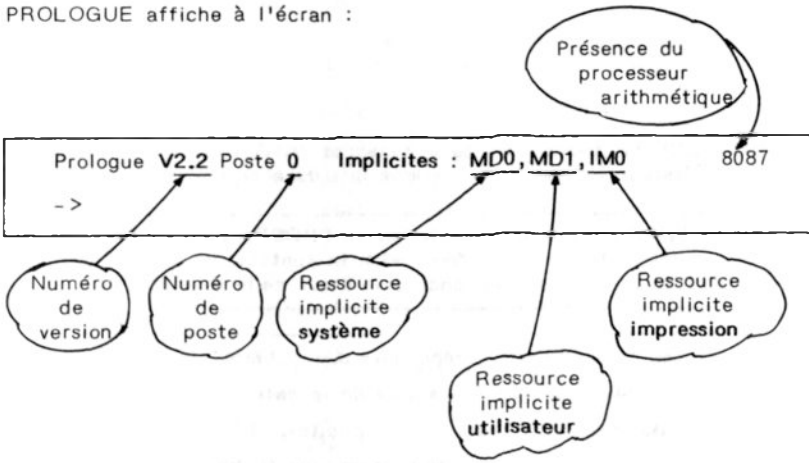
Lorsque PROLOGUE est chargé en mémoire, un signal d'appel représenté par une petite flèche orientée vers la droite ->est affiché à l'écran.

Ce signal signifie que l'interpréteur de commandes de PROLOGUE invite l'utilisateur à introduire, à partir du clavier, une chaîne de caractères, laquelle chaîne sera terminée par appui sur la touche <rc> ou ENTer et constitue la **COMMANDE PROLOGUE.**

Si en réponse à ce signal on introduit immédiatement <rc> ou ENTer (ce qui correspond à une commande vide) :

-> **<rc>**

PROLOGUE affiche à l'écran :



Par **IMPLICITES**, on entend les mnémoniques désignant une ressource périphérique à laquelle une commande fera référence par défaut.

- **IMPLICITE SYSTEME** désigne la ressource à partir de laquelle seront chargés les utilitaires (commandes).
- **IMPLICITE UTILISATEUR** désigne la ressource à partir de laquelle seront exploités les fichiers utilisateur.
- **IMPLICITE IMPRIMANTE** désigne la ressource impression affectée au poste.

Exemples

DATE <rc>

L'utilitaire DATE est chargé à partir de la ressource implicite système (**MD0** dans l'exemple précédent).

MD0 = Mini-Disque unité **0**.

On peut désigner également une ressource explicitement.

FL1.DATE <rc>

L'utilitaire DATE est chargé à partir de la ressource explicite **FL1**.

FL1 = **FL**oppy disque unité **1**.

D'une manière générale, une ressource périphérique est nommée par un **mnémonique** (deux caractères alphabétiques désignant l'élément périphérique) suivi d'un chiffre représentant le numéro de l'unité.

CATALOGUE DES FICHIERS

L'utilitaire CAT

En réalité, cet utilitaire se nomme "/", nous l'appellerons **CAT**, vous pouvez l'appeler DIR, l'auteur n'y voit aucun inconvénient !

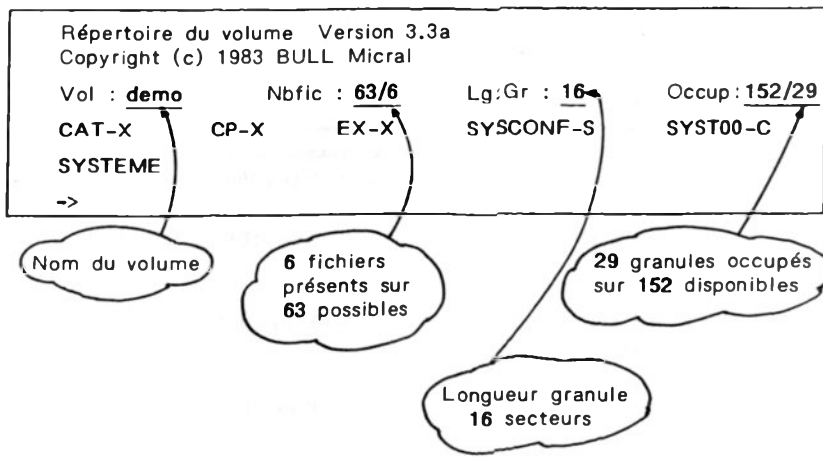
CAT permet d'afficher ou d'imprimer les caractéristiques d'un volume monté sur un support donné ainsi que les noms des fichiers contenus dans le répertoire de ce même volume.

Les différentes options associées permettent d'obtenir :

- la liste de tous les fichiers ;
- la liste des fichiers selon certains critères ;
- la liste des fichiers et leur implantation dans le volume.

CAT <rc>

Affiche la totalité du répertoire du volume monté sur l'unité implicite utilisateur.



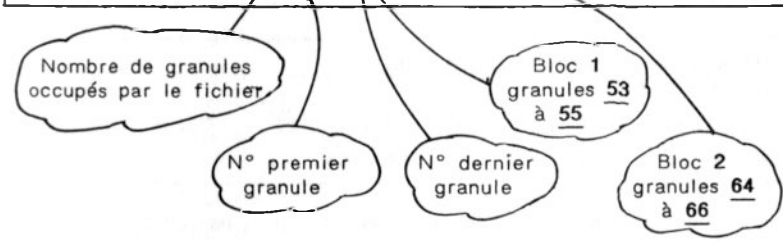
CAT,FL1

Même fonction, mais affiche le répertoire des fichiers du volume monté sur l'unité FL1.

CAT,GR

Affiche la totalité du répertoire avec en plus les implantations et occupations des divers fichiers situés dans le volume monté sur l'unité implicite utilisateur.

```
Répertoire du volume Version 3.3a
Copyright (c) 1983 BULL Micral
Vol : demo        Nbfic : 63/6        Lg:Gr : 16        Occup : 152/29

SYSTEME     :  10: (16,  25)
SYSCONF-S   :   1: (32,  32)
SYST00-C    :   1: (40,  40)
CAT-X       :   5: (48,  52)
CP-X        :   4: (56,  59)
EX-X        :   6: (53,  55)  (64,  66)
```



CAT,FL1,GR

Même fonction, mais le volume est monté sur l'unité FL1.

Un **BLOC** est un ensemble de 1 à n granules contigus.
Un fichier peut être constitué par **17** blocs au maximum.

CATALOGUE SELECTIF des Fichiers

Il est possible d'obtenir la liste des fichiers selon certains critères de sélection.

CAT,*-X

Permet d'obtenir la liste de tous les fichiers de type exécutable (-X) du volume monté sur le support implicite utilisateur.

Et si on désire avoir plus de détails :

CAT,*-X,GR

```
Répertoire du volume Version 3.3a
Copyright (c) 1983 BULL Micral
Vol : demo      Nbfic : 63/6      Lg:Gr : 16      Occup : 152/29
CAT-X          : 5:  (48,  52)
CP -X          : 4:  (56,  59)
EX-X           : 6:  (53,  55)  (64,  66)
```

Pour obtenir les informations relatives à un seul fichier, il suffit d'introduire la commande :

CAT,EX-X

```
Répertoire du volume Version 3.3a
Copyright (c) 1983 BULL Micral
Vol : demo      Nbfic : 63/6      Lg:Gr : 16      Occup : 152/29
EX-X           : 6:  (53,  55)  (64,  66)
```

L'option **GR** n'est pas nécessaire dans ce cas ; par contre, il faut préciser le **NOM** et le **SUFFIXE** du fichier.

L'édition peut être interrompue par appui sur la touche **ESC**ape.

Edition sur imprimante

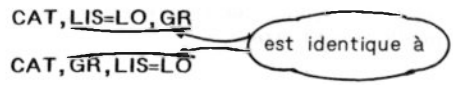
L'usage du mot-clé **LIS=LO** permet d'éditer le répertoire sur l'imprimante assignée au poste de travail.

CAT,FL0,LIS=LO

Edite la totalité du répertoire du volume monté sur l'unité **FL0**.

CAT,FL0,LIS=LO,GR

Edite la totalité du répertoire du volume monté sur l'unité **FL0**, avec les caractéristiques d'implantation des fichiers.



Résumé

CAT[,unité][,GR][,LIS=LO]
CAT,fichier[,GR][,LIS=LO]

unité	désigne le nom et numéro d'unité sur laquelle est monté le volume concerné.
fichier	désignation d'un fichier selon les règles ci-dessous.
GR	mot-clé optionnel spécifiant l'édition des caractéristiques d'implantation des fichiers.
LIS=LO	mot-clé optionnel spécifiant l'édition du répertoire sur l'imprimante assignée au poste de travail.

Spécifications de fichiers autorisées :

-	tous les fichiers.
xyz-	tous les fichiers dont le nom se termine par la chaîne xyz.
xyz*-*	tous les fichiers dont le nom commence par la chaîne xyz.
*-typ	tous les fichiers de même type.
*xyz-typ	tous les fichiers de même type et dont le nom se termine par la chaîne xyz.
xyz*-typ	tous les fichiers de même type dont le nom commence par xyz.

ATTENTION ! un seul * est autorisé par champ.

xyz-typ est interdit.

C'est une règle générale dans PROLOGUE.

L'ordre des options dans une commande est indifférent, sauf si expressemment indiqué.

GESTION DE LA DATE

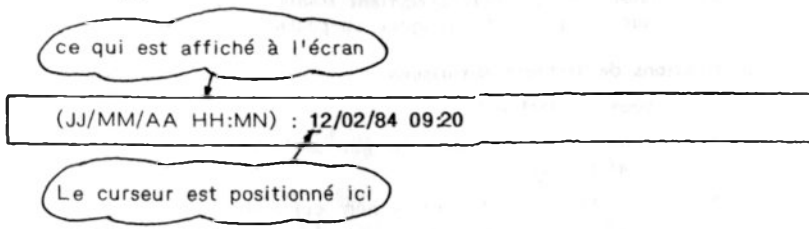
L'utilitaire DATE

PROLOGUE tient systématiquement à jour les informations date et heure.

Lors du chargement du système, ces valeurs sont initialisées d'une manière cohérente, mais quelconque.

L'utilitaire **DATE** permet d'éditer ou d'actualiser les valeurs courantes date, heure et minutes.

DATE <rc>



Nous sommes le **12 Février 1984**, il est **9** heures **20** minutes.

Vous pouvez alors introduire les valeurs adéquates en déplaçant le curseur au moyen des touches ← ou →, puis confirmer par appui sur <rc>.

DATE,20/05/84 14:50 <rc>

Dans cet exemple, date et heure sont passées en paramètres ; il n'y a pas d'affichage dans ce cas.

Un contrôle est effectué au niveau de la syntaxe ainsi que sur la vraisemblance des données introduites. En cas d'erreur, un message est affiché après validation de la commande.

Résumé

DATE[,jj/mm/aa hh:mn]

jj le jour.
mm le mois.
aa l'année.

hh l'heure.
mn les minutes.

Les informations DATE et HEURE sont accessibles au programmeur en langage BAL ou assembleur.

VISUALITION ET MODIFICATION DES IMPLICITES

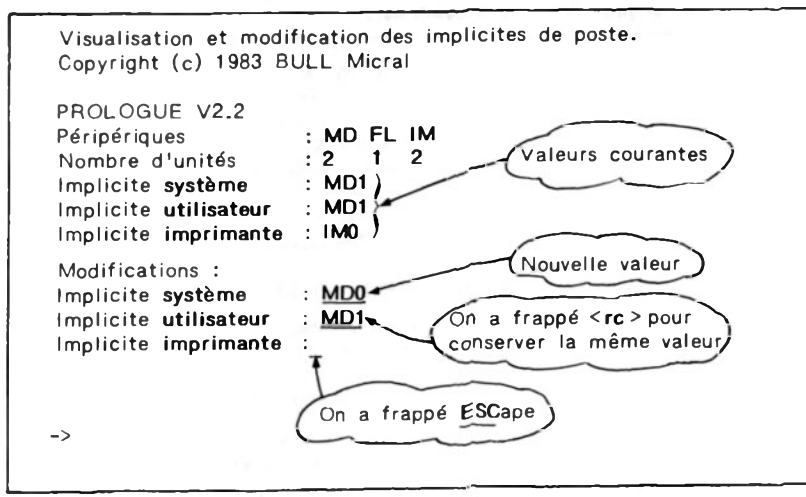
L'utilitaire VPMI

Cet utilitaire affiche ou imprime la liste des périphériques implicites associés au poste de travail.

VPMI permet également de modifier cette liste au gré de l'utilisateur.

VPMI <**rc**>

Voici ce qui est affiché à l'écran :



On distingue, sur la ligne périphérique, les noms des ressources disponibles sur ce système :

MD : 2 unités disque dur (MD0 et MD1).
FL : 1 unité floppy (FL0).
IM : 2 unités d'impression (**IM0** et **IM1**).

Résumé

VPMI [,LIS=LO]	
LIS=O	mot-clé facultatif : il permet d'éditer les mêmes informations sur l'imprimante assignée au poste : - si les assignations affichées conviennent ou si les modifications sont terminées, on retourne à l'interpréteur de commande par appui sur **ESC**ape ; - si l'on ne désire pas modifier une assignation, on répond par <**rc**> à la question posée.

MANIPULATION DE VOLUMES ET DE FICHIERS

L'utilitaire CP

Cet utilitaire permet de procéder à des manipulations destinées à définir ou modifier l'organisation logique des supports sectorisés (disquettes, disques,...).

*OPERATIONS SUR LES **VOLUMES***

CP,CV	pour Créer un Volume.
CP,DV	pour Dupliquer un Volume.
CP,RV	pour Renommer un Volume.

*OPERATIONS SUR LES **FICHIERS***

CP,DF	pour Dupliquer un Fichier.
CP,SF	pour Supprimer un Fichier.
CP,RF	pour Renommer un Fichier.

L'utilitaire CP-CV
Création d'un volume

Créer un volume, c'est organiser logiquement un support disque ou disquette. Il suffit pour cela de préciser l'identification du volume et le nom de l'unité sur laquelle il sera créé.

D'autres informations peuvent optionnellement être ajoutées. Ces options reflètent les choix de l'utilisateur quant à l'organisation propre du volume :

LGR=	taille du granule.
NBFIC=	nombre maximum de fichiers.
PM	demande de prémarquage préalable.

L'opération de **création** s'accompagne de la mise en place systématique du répertoire du volume, à condition que les paramètres de protection d'exploitation spécifiés lors de la configuration du système vous y autorisent.

CP,**CV,FL0,TEXTE,PM**

Un volume nommé **TEXTE** sera créé sur la disquette montée sur **FL0** (lecteur 0), un prémarquage préalable (**PM**) de la disquette devra être effectué.

PM représente un mot-clé, sa présence provoquera l'affichage à l'écran d'un message avec demande de confirmation de l'opération.

Le prémarquage n'est utile que si le support est vierge (neuf) ou dans le cas d'un support dont le prémarquage semble douteux.

Dans l'autre cas, on utilisera la commande :

CP,**CV,FL0,TEXTE**

sans préciser **PM**, si l'on considère que le support est déjà prémarqué.

Si vous effectuez une opération de création sur un volume déjà existant sous PROLOGUE et sans spécifier l'option **PM**, l'utilitaire **CP** vous invite à répondre au message :

Détruire le volume : xxxxxx (O/N)

Nom du volume déjà existant

En répondant Oui, vous provoquerez la suppression de tous les fichiers de l'ancien volume ; un nouveau volume sera créé avec les caractéristiques définies dans la commande CP,**CV**.

Toute autre réponse que Oui provoquera l'abandon de la commande en laissant l'ancien volume intact.

Résumé

CP,**CV**,**unité**,**volume** [,LGR=**xx**] [,NBFIC=**xx**] [,PM]	
unité	définit le nom et numéro d'unité sur laquelle se trouve monté le support à créer.
volume	chaîne de **huit** caractères alphanumériques au maximum définissant le nom du volume.
LGR=**xx**	**xx** spécifie la taille du granule exprimée en nombre de secteurs. La valeur par défaut est **16**.
NBFIC=**xx**	**xx** définit le nombre maximum de fichiers pouvant être répertoriés. Valeur par défaut = 63. Nombre maximal = 895 fichiers.
PM	La présence de ce paramètre indique à **CP** qu'il faut procéder préalablement au prémarquage du support.

L'utilitaire CP-DV
Duplication d'un volume

Cette fonction permet de reproduire les fichiers d'un volume donné, dans un autre volume.

Il est nécessaire, pour que cette opération se déroule correctement, qu'un volume ait été créé sur le support destinataire (commande CP,**CV**).

Dans tous les cas, les caractéristiques propres au volume destinataire ne sont pas modifiées :

- Nom du Volume.
- Taille des Granules.
- Nombre maximum de fichiers.

Les fichiers du volume monté sur l'unité **FL0** sont copiés sur le volume monté sur l'unité **FL1**.

Cette opération ajoute des fichiers dans le volume monté sur **FL1**.

Les fichiers de **FL1** ayant leur NOM et TYPE identiques à ceux de **FL0** sont supprimés et remplacés par la version provenant de **FL0**.

CP,DV,FL0,FL1,**RZ**

La présence du mot-clé **RZ** indique à l'utilitaire **CP** que tous les fichiers du volume monté sur **FL1** doivent être supprimés du catalogue avant l'opération de duplication.

A la fin de cette opération, le volume monté sur **FL1** contient tous les fichiers issus de **FL0**.

Avec ce mot-clé, un message sera affiché à l'écran, l'utilisateur devra confirmer par **O**ui, **N**on ou **A**bandon.

En cas d'erreur périphérique ou de débordement de volume (plus de place dans le volume destinataire ou trop de blocs pour un même fichier), un message est affiché à l'écran ; l'utilisateur devra répondre :

A si abandon de la copie.
C continuer la copie.
S sauter le fichier courant.

Dans tous les cas, les NOMS des fichiers copiés sont affichés à l'écran.

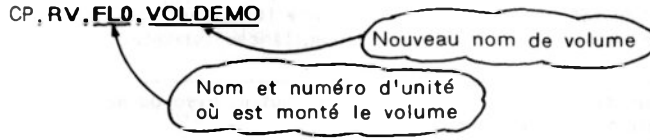
Les caractéristiques propres au volume destinataire ne sont pas modifiées :

- **Nom du volume**
- **Taille des granules**
- **Nombre maximum de fichiers**

Résumé

CP, **DV**, **orig**ine, **dest**ination[, **RZ**]	
DV	mot-clé spécifiant la commande de Duplication de Volume.
orig	nom et numéro de l'unité sur laquelle se trouve le support à dupliquer.
dest	nom et numéro de l'unité sur laquelle est monté le support destinataire.
RZ	mot-clé optionnel. S'il est présent, tous les fichiers seront supprimés du répertoire du volume destinataire.

L'utilitaire CP-RV
Changement du nom d'un volume



Cette commande permet de changer le nom d'un volume, sans modifier les caractéristiques initiales.

Le volume monté sur l'unité **FL0** aura désormais pour nom **VOLDEMO.**

Résumé

CP, RV, **unité**, **volume**	
RV	mot-clé définissant la fonction.
unité	nom et numéro de l'unité sur laquelle est monté le volume dont le nom est à changer.
volume	nouveau nom à donner au volume.

L'utilitaire CP-DF
Duplication de fichier(s)

La commande **CP,DF** permet de reproduire (dupliquer) un fichier ou un ensemble de fichiers dans l'une des circonstances suivantes :

- copie d'un fichier d'un support vers un autre, avec ou sans modification du nom du fichier ;
- copie d'un fichier sur le même ou un autre support avec modification du NOM et (ou) du TYPE ;
- copie d'un fichier de type source vers l'écran ou l'imprimante implicite du poste.

CP,DF,**FICHIER1-S,FL1**

Le fichier **FICHIER1** de type source (**-S**) situé sur le volume implicite utilisateur est transféré sur le volume monté dans l'unité **FL1.**

CP,DF,**MD0,FICH*-*,MD1**

Tous les fichiers situés sur le volume monté dans l'unité **MD0** et dont le nom commence par **FICH** et de type quelconque sont dupliqués sur le volume monté dans l'unité **MD1.**

CP,DF,**FL1.*-T,MD1**

Tous les fichiers de type traduit (**-T**) situés sur le volume monté dans l'unité **FL1** sont dupliqués sur le volume monté dans l'unité **MD1.**

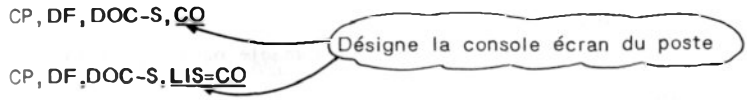
CP,DF,**FL1.PROG1-T,FL0.DEMO-T**

Le fichier **PROG1-T** situé sur le volume monté dans l'unité **FL1** est copié sur le volume monté dans l'unité **FL0** avec le nom **DEMO-T.**

Copie de fichier sur l'écran ou l'imprimante

Deux mots-clés particuliers désignent ces deux périphériques :

CO	pour l'écran du poste de travail.
LO	pour l'imprimante assignée au poste de travail.



Effectue la copie du fichier DOC-S sur l'écran du poste de travail.

CP, DF, DOC-S

CO est pris par défaut.

CP, DF, DOC, LO

Désigne l'imprimante assignée au poste.

CP, DF, DOC, LIS=LO

Effectue la copie du fichier DOC-S sur l'imprimante.

En cas de liste sur l'écran ou l'imprimante, on peut interrompre momentanément l'affichage en appuyant sur la touche ESCape.

Pour continuer, il suffit d'appuyer à nouveau sur ESCape.
Si l'on désire abandonner, l'appui sur la touche <rc> après ESCape renvoie à la flèche ->.

Résumé

CP, DF, **fich1**, **fich2**
CP, DF, **fich1** [, [LIS=] CO]
CP, DF, **fich1** [, [LIS=] LO]

DF	mot-clé signifiant Dupliquer Fichier.
fich1	désigne le fichier origine à dupliquer.
fich2	désigne le nom du fichier destinataire.
CO	mot-clé désignant la sortie écran.
LO	mot-clé désignant la sortie imprimante.
LIS=xx	mot-clé facultatif précédent CO ou LO.

L'utilitaire CP-SF
Suppression de fichier(s)

Cette commande a pour conséquence la suppression du fichier ou de l'ensemble des fichiers du répertoire ainsi que la récupération de l'espace disque occupé par le ou les fichier(s).

CP, **SF, FL0. DEMO-S**

Supprime le fichier **DEMO-S** du volume monté dans l'unité **FL0.**

CP, **SF, MD0. DEMO-•**

Supprime tous les fichiers du volume monté dans l'unité **MD0** ayant pour nom **DEMO**, quel que soit le type.

CP, **SF, MD0. •-T**

Supprime tous les fichiers de type traduit **(-T)** situés sur le volume monté dans l'unité **MD0.**

CP, **SF, DEM•-•**

Supprime tous les fichiers dont le nom commence par **DEM**, ces fichiers sont situés sur le volume monté dans l'unité implicite utilisateur.

Dans le cas de l'utilisation du caractère "•", il y a au prélable demande de confirmation de la suppression après affichage des noms de fichiers concernés.

Résumé

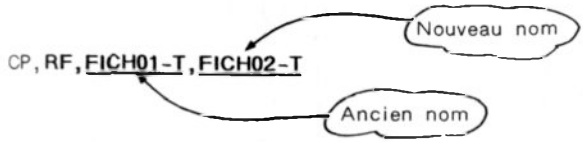
CP, SF. **fich**

SF	mot-clé signifiant Suppression de Fichier.
fich	désignation d'un nom de fichier.

Si l'unité n'est pas précisée, l'unité implicite utilisateur est prise par défaut.

L'utilitaire CP-RF
Renommer un fichier

Cette commande permet de changer le NOM et (ou) le TYPE d'un fichier sans en modifier le contenu.



Le fichier **FICH01-T** situé dans le volume implicite utilisateur est renommé en **FICH02-T.**

CP,RF,FL0.FICH03-S,FL0.FICH03-ASM

Le fichier **FICH03-S** situé dans le volume monté sur l'unité **FL0** voit son type changé en **-ASM.**

Il est impératif de préciser le même **NOM** de support, faute de quoi une erreur sera signalée.

Il est également impératif de préciser, avec le **NOM**, le **TYPE** et éventuellement les clés d'accès du fichier à renommer.

Résumé

CP,RF,**fich1**,**fich2**

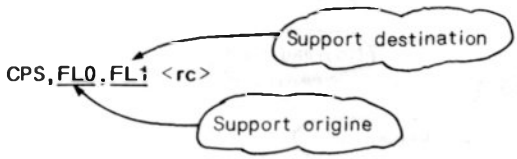
RF	mot-clé signifiant Renommer Fichier.
fich1	désigne le fichier dont le nom est à changer.
fich2	désigne le nouveau nom du fichier.

Si aucune unité n'est précisée, l'unité implicite utilisateur est prise par défaut.

COPIE DE SUPPORTS SECTORISES

L'utilitaire CPS

Cet utilitaire permet d'effectuer des copies intégrales ou partielles de supports sectorisés.

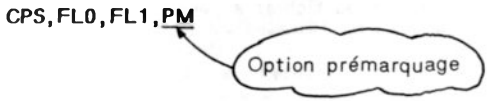


Permet la recopie du volume monté sur l'unité **FL0** vers le volume monté sur l'unité **FL1**.

Dans ce cas, il s'agit d'une copie piste par piste sans aucune modification de l'organisation logique du support destination. Il est impératif que le support destination ait été préalablement prémarqué (formaté).

Durant l'exécution de cette fonction, le nombre de pistes restant à copier est affiché à l'écran. Ce nombre décroît chaque fois qu'une piste est copiée du support origine (**FL0**) vers le support destination (**FL1**).

Le contrôle est rendu à l'interpréteur de commande lorsque la copie est terminée ou si une erreur est intervenue durant l'opération.



Même fonction que la précédente, mais un prémarquage préalable du support destination sera effectué (présence du mot-clé **PM**).

Dans ce cas de figure, si le volume monté sur l'unité destination est un volume **PROLOGUE**, un message de confirmation de l'opération de prémarquage sera affiché à l'écran.

La réponse **O**ui permet de poursuivre tandis que **N**on renvoie à l'interpréteur de commandes.

Copie partielle de supports sectorisés

Cette opération est délicate et demande toute l'attention de l'utilisateur.

En effet, **CPS** est amené à effectuer dans ce cas des copies de secteurs du support origine vers le support destination ; toute erreur dans l'introduction des données peut détruire le volume destination.

CPS <rc>

Appel de CPS sans passer de paramètres.

Puis le dialogue ci-dessous est établi.

Affiché par CPS	Réponses	Explications
ORIGINE	FL0 <rc>	1- nom et unité support origine.
PISTE	10 <rc>	2- numéro de piste début.
SECTEUR	5 <rc>	3- numéro de secteur début.
NB de PISTES	2 <rc>	4- nombre de pistes à transférer.
DESTINATION	FL1 <rc>	5- support destinataire.
PISTE	8 <rc>	6- numéro de piste destinataire.
SECTEUR	4 <rc>	7- numéro de secteur destinataire.
O/N : O		9- Oui pour effectuer l'opération. Non pour abandonner.

Les deux pistes débutant en piste 10, secteur 5 de FL0 sont dupliquées sur FL1 à partir de la piste 8 secteur 4.

A la fin de l'opération, le dialogue reprend en (1) pour permettre l'introduction de nouvelles données.

Si à la première question on répond <rc>, CPS retourne à l'interpréteur de commandes.

En cas d'erreur dans l'introduction des divers paramètres, certains messages d'erreur peuvent être affichés.

Messages	Explication
NOM D'UNITE INCORRECT	Unité spécifiée inconnue (1) (5)
SUPPORT NON SECTORISE	L'unité spécifiée n'est pas un support sectorisé (1) (5)
PARAMETRE INCORRECT	Numéro de piste ou numéro de secteur doivent être décimaux hexadécimaux (2) (3) (4) (6) (7)
PARAMETRE TROP GRAND	Paramètre numérique entré ou calculé > 65 535 (2) (3) (4) (6) (7)
Dans tous les cas, la détection d'une erreur renvoie au message initial ORIGINE (1).	

Outre la fonction essentielle de recopie de supports sectorisés, quelle que soit l'organisation du volume, **CPS** permet également d'effectuer, dans certains cas, des conversions de disquettes.

Cette possibilité dépend des caractéristiques de l'environnement matériel.

Considérations sur la nature des disquettes

On distingue trois types de disquettes :

- les disquettes **SIMPLE FACE, SIMPLE DENSITE** de piste ;
- les disquettes **DOUBLE FACE, SIMPLE DENSITE** de piste ;
- les disquettes **DOUBLE FACE, DOUBLE DENSITE** de piste.

Les conversions de support ne peuvent être effectuées que si le matériel le permet et si le module de l'environnement qui gère les disquettes prend en considération les options **SD** et **SDP**.

Dans le cas contraire, un message d'erreur est affiché.

Le tableau ci-dessous résume les possibilités qui sont offertes en fonction des caractéristiques des disquettes et des options citées dans la commande CPS.

	DISQUETTE ORIGINE		
DISQUETTE DESTINATION	simple face	double face	
		simple densité	double densité
simple face	PM		
double face simple densité	PM SD	PM	
double face double densité	PM SD SDP	PM SDP	PM

Les conversions ne peuvent être effectuées que dans le sens respectant l'égalité ou l'accroissement :

- du nombre de faces ;
- de la densité de piste ;
- du nombre de faces et de la densité de piste.

Résumé

CPS, **orig**, **dest** [, PM] [, SD] [, SDP]

CPS <**rc**> (copie partielle de support)

orig	nom et numéro de l'unité en lecture.
dest	nom et numéro de l'unité en écriture.
PM	demande de prémarquage du support de destination.
SD •	conversion simple face vers double face.
SDP •	conversion simple densité vers double densité.

• Ces deux options dépendent des caractéristiques de l'environnement. Elles peuvent ne pas exister sur certains matériels.

Dans la deuxième forme d'appel de CPS, un dialogue est établi.

Une adresse piste ou secteur peut être exprimée en hexadécimal sous la forme xxxH.

EDITION DE TEXTE SOURCE

L'utilitaire EDV

L'éditeur de textes **EDV** est un éditeur pleine page.

Il permet de **créer** ou de **modifier** des fichiers de type source. Un fichier source contient des informations rangées séquentiellement.

EDV,DEMO

Le fichier **DEMO-S**, situé sur le support implicite utilisateur, va être édité.

Le type -S est pris par défaut si le fichier DEMO- n'existe pas.

EDV,DEMO,DEST=NDEMO

Le fichier **DEMO-S**, situé sur le support implicite utilisateur, va être édité et produire un fichier de sortie **NDEMO** sur le support implicite utilisateur.

Trois fichiers sont ouverts par l'éditeur :

- le fichier **ORIGINE** ;
- le fichier **DESTINATION** ;
- le fichier **INTERMEDIAIRE.**

Le **fichier origine** est de type **-S** par défaut s'il est déjà créé s'il n'existe pas de fichier de même nom ayant pour type trois espaces.

Le **fichier intermédiaire** porte le même nom que le fichier origine, mais le type est **-$$**.

Le **fichier destination** porte par défaut le même nom que le fichier origine, mais temporairement un type **-$.**

Ce n'est qu'en fin d'opération que le fichier intermédiaire sera renommé dans le fichier destination.

EDV travaille dans un écran dont les caractéristiques nombre de lignes et nombre de colonnes sont déclarées dans le système ; de plus, il utilise les fonctions de gestion de curseur que l'on rencontre sur la plupart des claviers.

L'écran est divisé en trois zones :

- une zone **STATUS** occupant les deux lignes supérieures de l'écran et dans laquelle se trouve affiché le nom du fichier en cours de traitement. Cette zone sert également de zone de saisie de commandes spéciales ; on y accède par appui sur la touche **ESC**ape.
- une zone **AIDE** utilisateur occupant huit lignes et qui est affichée au gré de l'utilisateur par appui sur la touche line feed (interligne) ou **CTRL J.** Elle regroupe l'ensemble des fonctions accessibles en mode édition.

- une zone de **TRAVAIL** représentant la **PAGE** du fichier. C'est dans cet espace que l'utilisateur déplace le curseur dans le texte à l'aide des touches appropriées.

Fichier : **DEMO-S** **INSERTION**				
Mvts fichier	: **^C**:page avant		**^W**:desc ligne	**^L**:cent page
	^R:page arrière		**^Z** :montée ligne	
Mvts curseur	: **^F**:mot →	**^D**:car →	**^E**:ligne suiv	**^B**:déb/fin lig
	^A:mot ←	**^S**:car ←	**^X**:ligne préc	**^^**:déb/fin écr
Suppressions	: **^K**:ligne→	**^T**:mot →	**^G**:car →	**^Y**:ligne ent
	^U:ligne←	**^**:mot ←	**EFF**:car ←	
Divers	: **^J**:aide O/N	**^N**:insert cr/lf	**^O**:mode insertion O/N	

Exemple d'écran avec la zone status composée de deux lignes écran, la zone d'aide utilisateur composée de huit lignes et le reste pour l'affichage de la page du fichier.

Les **mouvements dans le fichier**

Ces commandes permettent le déplacement du fichier dans l'espace affichable d'une page.

Elles ne modifient en rien le contenu du fichier.

^C	**DESCENDRE D'UNE PAGE** Le curseur est positionné en début de page. En fin de fichier, si l'on continue à frapper cette commande, le curseur est positionné au début de la dernière ligne.
^R	**MONTER D'UNE PAGE** Le curseur est positionné au début de la dernière ligne de la page. En début de fichier, si l'on exécute cette commande, le curseur se positionne au début de la première ligne de la page.
^Z	**MONTER LA PAGE D'UNE LIGNE** Le curseur reste immobile tandis que la page monte d'une ligne.
^W	**DESCENDRE LA PAGE D'UNE LIGNE** Le curseur reste immobile tandis que la page descend d'une ligne.
^L	**SE POSITIONNER EN MILIEU DE PAGE** La ligne où se trouve le curseur se positionne au milieu de la page. Permet de monter ou de descendre d'une demi-page au maximum.

Les mouvements curseur

^^	**CURSEUR EN DEBUT OU FIN DE PAGE** Le curseur est toujours placé en début de ligne. Si l'on était en début de page, le curseur irait en fin de page et inversement.
^B	**CURSEUR EN DEBUT OU FIN DE LIGNE** Permet de se positionner alternativement au début ou à la fin d'une ligne.
^X	**CURSEUR DESCENDU D'UNE LIGNE** On peut aussi utiliser, si elle existe, la touche flèche vers le bas.
^E	**CURSEUR REMONTE D'UNE LIGNE** On peut aussi utiliser, si elle existe, la touche flèche vers le haut.
^A	**CURSEUR A GAUCHE D'UN MOT** Le curseur se déplace vers la gauche et vient se positionner sur le premier caractère du mot précédent.
^F	**CURSEUR A DROITE D'UN MOT** Le curseur se déplace vers la droite et vient se positionner sur le premier caractère du mot suivant.
^S	**CURSEUR A GAUCHE D'UN CARACTERE** Le curseur se positionne sur le caractère précédent. On peut aussi utiliser la touche ←.
^D	**CURSEUR A DROITE D'UN CARACTERE** Le curseur se positionne sur le caractère suivant. On peut aussi utiliser la touche →.

Les suppressions

^Y	**SUPPRIMER LA LIGNE COURANTE** La ligne où se trouve positionné le curseur est supprimée.
^U	**SUPPRIMER PARTIE GAUCHE DE LA LIGNE** Tout le texte situé à gauche du curseur est supprimé sur la même ligne. Une ligne de texte peut être représentée par une ou plusieurs lignes écran.

^K	**SUPPRIMER PARTIE DROITE DE LA LIGNE** Tout le texte situé à droite du curseur est supprimé sur la même ligne.
^\	**SUPPRIMER MOT A GAUCHE** Le mot est supprimé à gauche à partir de l'emplacement du curseur.
^T	**SUPPRIMER MOT A DROITE** Le mot ou la fin du mot est supprimé à droite à partir de l'emplacement du curseur.
EFF	**SUPPRIMER CARACTERE GAUCHE** Le caractère situé à gauche du curseur est supprimé. Le curseur ne bouge pas.
^G	**SUPPRIMER CARACTERE A DROITE** Le caractère désigné par le curseur est supprimé.

Autres commandes

^N	**INSERER UN RETOUR CHARIOT** Un retour chariot et un retour ligne (**RC+LF**) sont insérés à l'emplacement du curseur.
^M	**FIN DE LIGNE** ou **<rc>** Si on est en mode insertion, un retour chariot est inséré (fin de ligne). Sinon, on se positionne au début de la ligne suivante.
^O	**DEBUT OU FIN D'INSERTION** Cette commande permet de passer en mode insertion ou de revenir au mode normal.
^I	**TABULATION ou TAB** Permet d'insérer le caractère tabulation dans le fichier.
^P	**INSERER LE CARACTERE SUIVANT** Permet d'insérer un caractère quelconque dans le fichier même si c'est un caractère de contrôle.
ESC	**PASSER EN MODE COMMANDE** Le curseur est positionné dans la ligne de STATUS (première ligne de l'écran) dans l'attente d'une commande.

Mode commande

Le mode commande se déroule sur la première ligne de l'écran.

Le caractère "*", affiché sur cette ligne de l'écran, indique à l'utilisateur qu'il peut introduire sa commande.

Deux possibilités sont offertes :

- introduire la commande à la suite de "*" et la valider par <rc> ;
- exécuter la commande précédente en validant par <**rc**> à la suite de "*".

E	**FIN DE SESSION** Permet de revenir à PROLOGUE quand la session de travail est terminée. EDV met à jour définitivement le fichier destinataire tandis qu'il conserve un fichier de type -A contenant l'ancien fichier avant la session d'édition.
V	**RETOUR AU MODE EDITION** Quand on est en mode commande, permet de revenir là où était positionné le curseur en mode édition.
H	**SAUVE LE FICHIER** En mode commande, permet de sauver le fichier en cours de manipulation.
O	**ABANDONNE L'EDITION** Annule toutes les modifications effectuées et retourne au début du fichier. Une confirmation de la commande est demandée pour cette fonction.
Q	**ABANDONNE L'EDITION** Retour à PROLOGUE sans sauvegarde du fichier. Une confirmation de la commande est demandée pour cette fonction.
+B	**POSITIONNE LE CURSEUR EN DEBUT DE FICHIER** Le signe + peut être omis.
-B	**POSITIONNE LE CURSEUR EN FIN DE FICHIER**
	RECHERCHE CHAINE DE CARACTERES EN MEMOIRE
+nF	La chaîne de caractères suivant la lettre **F** est recherchée depuis la position du curseur vers la fin de la mémoire. Le signe + peut être omis.
-nF	La chaîne de caractères suivant la lettre **F** est recherchée depuis la position du curseur vers le début de la mémoire. **n** représente le numéro de l'occurrence, par défaut, **n** est égal à **1**.

RECHERCHE CHAINE DE CARACTERES DANS LE FICHIER

+nN La chaîne de caractères suivant la lettre **N** est recherchée depuis la position du curseur vers la fin du fichier. Le signe + peut être omis.

-nN La chaîne de caractères suivant la lettre **N** est recherchée depuis la position du curseur vers le début du fichier.

n représente le numéro de l'occurrence, par défaut, **n** est égal à **1**.

RECHERCHE CHAINE EN MEMOIRE ET SUBSTITUTION

+nS La chaîne de caractères suivant la lettre **S** est recherchée en mémoire, depuis la position du curseur vers la fin de la mémoire. La chaîne à substituer est séparée de la chaîne à rechercher par le caractère **ESC**ape et se termine par <**rc**>. Le signe + peut être omis.

-nS Idem, mais vers le début de la mémoire.

n représente le numéro de l'occurrence. Par défaut, **n** est égal à **1**.

RECHERCHE CHAINE DANS LE FICHIER ET SUBSTITUTION

+nR Vers la fin du fichier.

-nR Vers le début du fichier.

nQP **PLACER n LIGNES DANS LA RESERVE**

Les **n** lignes situées à partir de la position du curseur sont rangées dans une "**réserve**", puis sont effacées de l'écran.

n/QP **AJOUTER n LIGNES DANS LA RESERVE**

Les **n** lignes situées à partir de la position du curseur sont ajoutées à celles déjà présentes dans la réserve, puis sont effacées de l'écran.

nQG **INSERER LA RESERVE DANS LE FICHIER**

Les lignes contenues dans la réserve sont insérées **n** fois dans le fichier, à partir de la position du curseur.

QT **VISUALISER LA RESERVE**

QK **EFFACER LA RESERVE**

SAUVEGARDE DE FICHIERS

L'utilitaire SV

Il permet d'effectuer des copies de sécurité d'un volume ou plusieurs fichiers, sur un ou plusieurs volumes de sauvegarde.

Un indicatif spécifique **••COPY••** est associé aux volumes créés. Cet indicatif qui n'apparaît que dans les messages, est utilisé par PROLOGUE pour identifier les volumes de sauvegarde, de manière à les distinguer des volumes créés par d'autres utilitaires.

> Un volume créé par **SV** ne peut être copié par l'utilitaire **CP**.
> Tous les volumes de sauvegarde sont numérotés ; la nomenclature utilisée permet de reconnaître le dernier.

SV,DEMO-*,FL0,PM

L'ensemble de fichiers **DEMO** est sauvegardé sur la disquette montée dans l'unité **FL0**.

Un prémarquage de cette disquette est effectué avant la sauvegarde.

SV,DEMO-T,FL0

Le fichier **DEMO-T** est sauvegardé sur la disquette montée dans l'unité **FL0**.

SV,DEMO-T,DEST=FL0

Commande identique à la précédente, la présence du mot-clé DEST=**FL0** ne fait qu'expliciter l'unité destinataire.

SV,MD0,FL0

Sauvegarde du volume monté sur l'unité **MD0** dans le volume monté sur l'unité **FL0**.

Dans ce cas, il est prudent d'effectuer un prémarquage préalable de plusieurs disquettes. Au cas où un débordement de volume sur **FL0** survenait, **SV** demanderait alors de monter le volume suivant et l'opération se poursuivrait.

Résumé

SV,fich, [DEST=]unité [,PM]

fich	désigne le nom du fichier ou du groupe de fichiers dont on désire effectuer la copie de sécurité. Il peut aussi spécifier le nom d'une unité ; dans ce cas, tous les fichiers du volume correspondant seront sauvegardés.
DEST=	mot-clé optionnel. Si présent, il précède le nom de l'unité où seront montés le ou les volumes destinés à recevoir les copies de sécurité.
unité	nom de l'unité destinataire des copies de sécurité.
PM	mot-clé optionnel. Il spécifie le prémarquage préalable de tous les supports nécessaires à la copie de sauvegarde.

RESTAURATION DE FICHIERS

L'utilitaire RT

Il permet de restaurer le ou les volumes **••COPY••** qui ont été créés par l'utilitaire **SV**.

RT,DEMO-T,FL1

Le fichier **DEMO-T** du volume de sauvegarde monté dans l'unité implicite utilisateur est restauré dans le volume monté dans l'unité FL1.

RT,DEMO-•,FL1

Tous les fichiers **DEMO** du volume de sauvegarde monté dans l'unité implicite utilisateur sont restaurés dans le volume monté dans l'unité FL1.

RT,DEMO-T,DEMO-S,DEMO-D,DEST=FL1

Les fichiers **DEMO-T,DEMO-S,DEMO-D** du volume de sauvegarde monté dans l'unité implicite utilisateur sont restaurés dans le volume FL1. Dans le cas où plusieurs noms de fichiers sont précisés, la présence du mot-clé **DEST=**unité est obligatoire.

RT,FL1,MD0

Dans cet exemple, tous les fichiers du volume de sauvegarde monté dans l'unité **FL1** sont transférés dans le volume monté dans l'unité **MDO**.

Résumé

RT,orig,dest
RT,fich1 [,fich2] ... [,fichN], [DEST=],unité

orig	nom unité origine ou nom de fichier.
dest	nom unité destination.
fich1,..n	est la liste des noms de fichiers qui font l'objet de la restauration. Les noms de fichiers spécifiés peuvent comporter le caractère de substitution (•).
DEST=	mot-clé optionnel si la commande comporte un seul nom de fichier à restaurer. Mot-clé obligatoire si la commande comporte plus d'un nom de fichier à restaurer.
unité	nom de l'unité destinataire.

SAUVEGARDE FICHIERS SEQUENTIELS INDEXES

L'utilitaire SVSI

Cet utilitaire permet d'effectuer des copies de sécurité de fichiers séquentiels indexés dans un volume PROLOGUE.

Le fichier de sauvegarde créé par **SVSI** est un fichier séquentiel. Les volumes susceptibles de contenir des copies de sauvegarde de fichiers séquentiels indexés doivent préalablement être prémarqués par l'utilitaire CP.

SVSI, DEMO, DEST=FL1

Le fichier séquentiel indexé **DEMO-I** et le fichier séquentiel associé DEMO-D sont transférés dans le volume monté sur l'unité **FL1**. Le fichier résultant aura pour nom **DEMO-U**.

SVSI, DEMO, DEST=FL1.SAVDEMO

Même opération que la précédente, mais le fichier créé dans le volume monté sur l'unité **FL1** aura pour nom SAVDEMO-U.

SVSI, MD0.DEMO-I

Le fichier séquentiel indexé **DEMO-I** situé dans le volume monté sur l'unité **MD0** est sauvegardé dans le volume monté sur l'unité implicite utilisateur (option par défaut).

> Un fichier séquentiel indexé est un ensemble de deux fichiers portant le **même nom.** L'un contient les indexes (**-I**) et l'autre les données associées (**-D**).

Le fichier séquentiel de sauvegarde est organisé en blocs de **4** K octets contenant les informations suivantes :

> - Description du fichier :
> . longueur enregistrement ;
> . type de fichier ;
> . longueur de la clé (pour le premier bloc seulement).
> - Les enregistrements stockés en séquence et comprenant :
> . longueur de la zone des données ;
> . clé (la valeur doit être croissante) ;
> . l'index ;
> . les données proprement dites.

Résumé

SVSI, **fich1** [, DEST=**fich2**]

fich1	nom du fichier séquentiel indexé à sauvegarder.
DEST=	définit le nom du fichier de sauvegarde ou le nom de l'unité dans laquelle est monté le volume de sauvegarde. Par défaut, le nom de l'unité est l'implicite utilisateur.
fich2	nom du fichier de sauvegarde. Le type **-U** est pris par défaut.

RESTAURATION FICHIERS SEQUENTIELS INDEXES

L'utilitaire RTSI

Cet utilitaire permet la restauration avec réorganisation des fichiers séquentiels indexés sauvegardés par **SVSI**.

Le fichier des clés et, éventuellement, le fichier des données associé sont reconstitués à partir du fichier séquentiel unique produit par **SVSI**.

Si le fichier de sauvegarde est un fichier multi-volumes, les volumes successifs doivent être restaurés dans l'ordre approprié. Un message affiché à l'écran signalera tout changement de volume nécessaire.

RTSI,FL1.DEMO

Cette commande va extraire du volume monté sur l'unité **FL1** le fichier **DEMO-U** et constituer, dans le volume implicite système, le fichier des clés **DEMO-I** et le fichier de données associé **DEMO-D**.

RTSI,FL1.DEMO,DEST=MD1.MODE-I

Même opération que précédemment, mais avec création du fichier **MODE-I** et **MODE-D** dans le volume monté sur l'unité explicite **MD1**.

Résumé

RTSI,fich1 [**,DEST=fich2**]	
fich1	désigne le nom du fichier origine à restaurer.
DEST=	mot-clé annonçant le nom du fichier séquentiel indexé à reconstituer.
fich2	désigne le nom du fichier séquentiel indexé à reconstituer : **fich2-I** et **fich2-D**. Par défaut, **DEST=fich1**.

MODIFICATION DE SUPPORT SECTORISE

L'utilitaire PATCH

Cet utilitaire permet de **VISUALISER** et (ou) **MODIFIER** les données enregistrées sur un support sectorisé (disquette, disque,...) ou dans un fichier.

> **PATCH** accédant à des secteurs physiques d'un support sectorisé, son usage implique certaines précautions, surtout quand il s'agit de modifications de fichiers exécutables.
> Il faut donc être prudent sur son utilisation.

Comme certains utilitaires, **PATCH** n'est accessible à l'utilisateur que si les paramètres de protection d'exploitation spécifiés lors de la configuration du système le permettent.

PATCH, DEMO-X

Visualisation du premier secteur du fichier **DEMO-X** situé dans le volume implicite système.

PATCH, FL1. DEMO-X

Visualisation du premier secteur du fichier **DEMO-X** situé dans le volume monté sur l'unité **FL1.**

PATCH, DEMO-X, NSEC=5

Visualisation du **5e** secteur du fichier **DEMO-X** situé dans le volume implicite système.

PATCH, FL0, NPIS=6

Visualisation du **premier** secteur de la **piste 6.** Ici, le numéro de piste est absolu.

Lors de l'appel de **PATCH**, l'image du premier secteur désigné par les mots-clés ci-dessous est affichée à l'écran.

NSEC=ss	secteur **ss** relatif au début du support.
NPIS=pp	secteur **0** de la piste **pp**.
NPIS=pp, NSEC=ss	secteur **ss** de la piste **pp**.

Le curseur est positionné sur le premier octet du secteur. Pour le déplacer, on utilise les touches :

Flèche	↑ vers le **haut** (- 16 octets).
	↓ vers le **bas** (+ 16 octets).
	← vers la **gauche** (déplacement d'un demi-octet).
	→ vers la **droite** (déplacement d'un demi-octet).
	↖ **home** (déplacement du curseur en haut à gauche).

Pour remplacer un chiffre, il suffit de frapper la valeur hexadécimale correspondante (**0-9, A-F**), le nouveau chiffre sera alors affiché à l'emplacement du curseur.

Les commandes

Outre le caractère hexadécimal représentant une donnée à inscrire dans le secteur, **PATCH** reconnaît certains caractères spéciaux qui tiennent lieu de commande.

La touche **ESC**ape permet à l'utilisateur de quitter PATCH pour retourner à l'interpréteur de commandes. Dans ce cas, si des modifications du secteur ont été effectuées, elles ne seront pas prises en compte.

' '	L'appui sur la barre espace provoque la lecture et l'affichage du secteur suivant. La fin du fichier ou du volume est signalée par un Beep sonore.
H ctrl J	Aide à l'opérateur. Affiche la signification des commandes disponibles et les caractéristiques du fichier ou du volume.
I	Imprime le contenu du secteur affiché avec la traduction en code **ASCII**.
K	Recherche une chaîne **ASCII** ou hexadécimale (chaîne à rechercher est, dans ce cas, précédée du caractère "/"). Le sens de la recherche peut être croissant (de la position du curseur vers la fin du fichier ou du volume) ou décroissant (de la position du curseur vers le début du fichier ou du volume).
L	Lit un secteur déterminé dans un fichier ou lit un secteur relatif au début d'un volume. Le numéro de ce secteur peut être introduit en absolu ou sous la forme **Piste,Secteur**.

M	Ecrit le secteur affiché. Pour être effective, cette commande doit se terminer par **ENT**er ou **<rc>**.
N	Affiche la dernière chaîne cherchée, ainsi que le sens de cette recherche.
P	Lit et affiche le secteur précédent ou indique par un beep sonore le début du fichier.
T	Transfert du secteur en cours vers un autre secteur.
X	Convertit une chaîne **ASCII** en valeur décimale. L'appui sur la touche **ENT**er ou **<rc>** termine cette fonction, la chaîne **ASCII** et la valeur hexadécimale correspondante restent affichées à l'écran jusqu'à la prochaine commande.
Z	Initialise le secteur (sans l'écrire) avec une valeur hexadécimale quelconque.

PATCH,md0,nsec=4

```
   Support : md0                                  Piste : 0        Secteur : 4
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F                   0123456789ABCDEF
-------------------------------------------------------------------------------
10 EE A0 00 00 00 00 00 53 59 53 54 45 4D 45 20 | 00 | ........SYSTEME  | 00
20 20 20 20 20 20 20 58 53 59 53 43 4F 4E 46 58 | 10 | /       XSYSCONFX | 10
4E 53 59 53 20 20 20 58 56 5D 4D 49 20 20 20 58 | 20 | NSYS    XVPMI   X | 20
43 50 53 20 20 20 20 58 45 44 20 20 20 20 20 58 | 30 | CPS     XED     X | 30
53 50 4F 4F 4C 20 20 58 44 41 54 45 20 20 20 58 | 40 | SPOOL   XDATE   X | 40
53 54 41 54 55 53 20 58 54 53 4D 20 20 20 20 58 | 50 | STATUS XTSM     X | 50
41 53 47 20 20 20 20 58 53 56 20 20 20 20 20 58 | 60 | ASG     XSV     X | 60
52 54 20 20 20 20 20 58 53 54 4D 45 4D 20 20 58 | 70 | RT      XSTMEM  X | 70
43 50 20 20 20 20 20 58 50 41 54 43 48 32 20 58 | 80 | CP      XPATCH2 X | 80
01 54 45 53 54 4D 20 20 43 4F 4E 46 49 47 20 58 | 90 | .TESTM  CONFIG  X | 90
4E 45 57 53 49 20 20 58 45 44 56 20 20 20 20 58 | A0 | NEWSI   XEDV    X | A0
44 49 41 4C 44 47 45 53 44 49 41 4C 44 47 4D 53 | B0 | DIALDGESDIALDGMS  | B0
44 49 41 4C 44 47 4C 53 44 41 49 4C 44 47 54 53 | C0 | DIALDGLSDIALDGTS  | C0
44 49 41 4C 44 4C 45 53 44 49 41 4C 44 4C 4D 53 | D0 | DIALDLESDIALDLMS  | D0
44 49 41 4C 44 4C 4C 53 44 49 41 4C 44 4C 43 53 | E0 | DIALDLLSDIALDLCS  | E0
44 49 41 4C 44 4C 53 53 44 49 41 4C 44 4C 58 53 | F0 | DIALDLSSDIALDLXS  | F0
-------------------------------------------------------------------------------
```

PATCH, ED-X

```
   Fichier : ed-x                                  Secteur : 0
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F        0123456789ABCDEF
-------------------------------------------------------------------------
4D 5A 42 00 52 00 00 00 20 00 00 00 FF FF 00 00 | 00 | MZB.R.....  ..... | 00
00 00 01 38 00 01 00 00 1C 00 00 00 00 00 00 00 | 10 | ...3........... | 10
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 20 | ................ | 20
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 30 | ................ | 30
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 40 | ................ | 40
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 50 | ................ | 50
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 60 | ................ | 60
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 70 | ................ | 70
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 80 | ................ | 80
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 90 | ................ | 90
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | A0 | ................ | A0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | B0 | ................ | B0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | C0 | ................ | C0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | D0 | ................ | D0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | E0 | ................ | E0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | F0 | ................ | F0
-------------------------------------------------------------------------
```

PATCH, MD0, NPIS=0, NSEC=2

```
   Support : md0                    Piste : 0      Secteur : 2
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F        0123456789ABCDEF
-------------------------------------------------------------------------
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 00 | ................ | 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 10 | ................ | 10
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 20 | ................ | 20
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 30 | ................ | 30
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 40 | ................ | 40
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 50 | ................ | 50
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 60 | ................ | 60
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 70 | ................ | 70
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 80 | ................ | 80
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 90 | ................ | 90
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | A0 | ................ | A0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | B0 | ................ | B0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | C0 | ................ | C0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | D0 | ................ | D0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | E0 | ................ | E0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | F0 | ................ | F0
-------------------------------------------------------------------------
Chaine à convertir : M  D  0
                     4D 44 30
```

La commande **X** a été frappée

puis la chaine **M D 0** etc jusqu'à <rc>. Pour chaque caractère frappé, apparait au dessous, la valeur hexadécimale correspondante.

```
   Support : md0                    Piste : 0          Secteur : 0
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F       0123456789ABCDEF
-------------------------------------------------------------------------
                  MENU                    | 00 | ................ | 00
  ' ' : Secteur suivant                   | 10 | ................ | 10
    P : Secteur précédent                 | 20 | ................ | 20
    L : Lire un Secteur ou Piste,Secteur  | 30 | ................ | 30
    M : Ecriture secteur en cours         | 40 | ................ | 40
    T : Translater le Secteur en cours    | 50 | ................ | 50
    K : Rechercher une Chaine             | 60 | ................ | 60
    N : Nouvelle Recherche                | 70 | ................ | 70
    I : Imprimer le secteur en cours      | 80 | ................ | 80
    S : Impression enchainée              | 90 | ................ | 90
    R : Visualisation enchainée           | A0 | ................ | A0
    Z : Remplissage du secteur avec une valeur | B0 | ............ | B0
    X : Convertir une chaine en hexadécimal | C0 | .............. | C0
 home : Curseur en haut à gauche          | D0 | ................ | D0
  Esc : Sortir                            | E0 | ................ | E0
        Frapper une touche pour retour    | F0 | ................ | F0
-------------------------------------------------------------------------
 Caractéristiques :  43968  Secteur ( 1374  Pistes de 32  Secteurs )
```

Résumé

PATCH,fich [,NPIS=p] [,NSEC=s]
PATCH,unité [,NPIS=p] [,NSEC=s]

fich	désigne le nom du fichier concerné. Dans ce cas, le type de fichier (s'il existe) doit être précisé.
unité	désigne le nom de l'unité concernée.
NPIS=p	mot-clé optionnel : p désigne le numéro de piste. Si un nom de fichier est précisé, p représente le numéro de piste relative au début du fichier. Si p n'est pas mentionné, le numéro de piste est pris égal à zéro.
NSEC=s	mot-clé optionnel : s désigne le numéro de secteur. Si un nom de fichier est précisé, s représente le numéro de secteur relatif au début de la piste. Si s n'est pas mentionné, le numéro de secteur est pris égal à zéro. Si seulement l'unité est précisée, s désigne le numéro de secteur absolu par rapport au début du volume ou de la piste.

UTILISATION DES FICHIERS DE COMMANDES

L'utilitaire ASG

Cet utilitaire permet de substituer un fichier de commandes au clavier.

Lors de l'appel de **ASG**, celui-ci transmettra chaque ligne du fichier à l'interpréteur de commandes, comme si ces commandes avaient été introduites à partir du clavier.

Si vous devez effectuer périodiquement une sauvegarde de n fichiers, au lieu d'introduire à chaque fois au clavier n commandes **CP,DF ...**, vous créerez une fois pour toutes un fichier de commandes que vous soumettrez à **ASG** le moment venu.

Soit, par exemple, le fichier **COMMAND-S** créé par l'utilitaire EDV.

```
;Les lignes commençant par un point-virgule sont des
;lignes commentaires.
;Copie du fichier DEMO1-S situé sur MD0 vers FL0.
CP,DF,MD0.DEMO1-S,FL0
;Copie du fichier DEMO1-T vers FL0
CP,DF,MD0.DEMO1-T,FL0
;Copie du fichier DEMO1-D sur FL0
CP,DF,MD0.DEMO1-D,FL0
;Copie du fichier DEMO1-I,FL0
CP,DF,MD0.DEMO1-I,FL0
```

puis, plus tard ... exécution du fichier COMMAND-S.

ASG.COMMAND <**rc**>

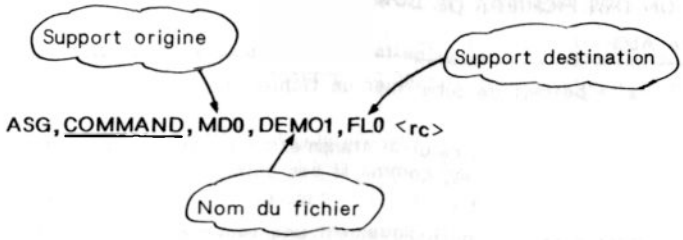
Nom du fichier de commandes.
Le type **-ASG** est pris par défaut.

-> Retour à la flèche quand c'est fini.

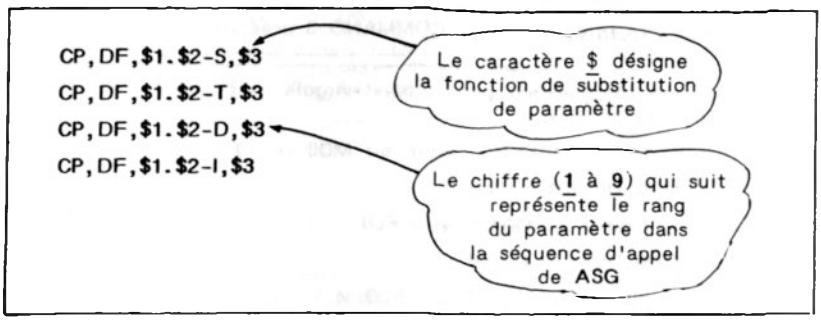
Commandes paramétrées

Afin de généraliser l'utilisation de ASG, il est possible de communiquer des paramètres au fichier de commandes lors de l'appel de **ASG**.

Toujours dans le cadre du premier exemple, en ne connaissant pas à priori le nom du fichier à copier, les noms des supports origine et destination, **ASG** offre la possibilité de communiquer ces paramètres.

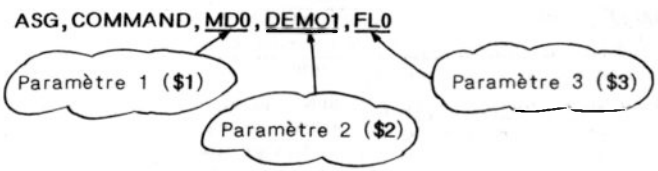


Le fichier **COMMAND-S** aura la structure suivante :



Dans cet exemple, il s'agissait de dupliquer des fichiers de même nom, mais de type **-S, -T, -D, -I**, situés sur **MD0** et à destination de **FL0**.

La substitution des paramètres est faite comme ci-dessous :



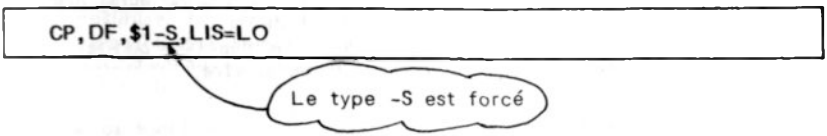
Exécution implicite d'un fichier de commandes

L'interpréteur de commande permet l'exécution de l'utilitaire ASG d'une manière implicite si le fichier de commandes créé a pour suffixe **-ASG.**

Cette particularité permet à l'utilisateur de définir, dans certaines conditions, de nouvelles commandes.

Imaginons une commande **LIST** permettant de lister à l'écran un fichier source de type **-S.**

On crée, à l'aide de **EDV**, le fichier de commandes **LIST-ASG** à qui l'on passera comme paramètre le nom du fichier source à lister.



Pour lister le fichier **DEMO-S**, il suffira d'écrire :

LIST,DEMO

au lieu de **CP,DF,DEMO-S,LIS=CO.**

LIST ayant pour suffixe **-ASG**, l'interpréteur de commandes charge implicitement l'utilitaire **ASG-X** avec comme paramètres **LIST,DEMO.**

On pourrait imaginer aussi une commande du style **COPY** qui permet d'effectuer des transferts de fichiers au lieu d'écrire la commande **CP,DF, ...**

```
CP,DF,$1,$2
```

```
COPY,MD0.DEMO-T,MD1
COPY,FL0.DEMO-T,MD0.DEMON-T
```

etc ...

Résumé

ASG, fich [, **paramètres**]

fich désigne le nom du fichier source contenant les commandes. Le fichier est de type -**ASG** par défaut.
Il est, en principe, utilisé en mode **partageable**, ce qui signifie qu'il peut être exploité simultanément par plusieurs programmes.

ASG,FICH

paramètres **liste optionnelle** de paramètres séparés par des virgules. Ces paramètres se substituent à des variables insérées dans les lignes source du fichier.
Les variables substituables sont représentées par le symbole **$** suivi du numéro du paramètre.

ASG,FICH,**P1**,**p2**, ... **p8**,**p9**
↓ ↓
$1 $9

IMPRESSION DES FICHIERS

L'utilitaire SPOOL

C'est un utilitaire qui permet la mise en oeuvre d'une ou de plusieurs imprimantes dans des conditions d'efficacité et de performances optimales.

SPOOL permet de résoudre les problèmes tels que :

- partager des ressources d'impression entre plusieurs utilisateurs, en accès simultané ;
- réguler la charge d'une imprimante, tout en se réservant la possibilité de procéder à des éditions différées ;
- optimiser le temps d'exécution des programmes qui sollicitent fréquemment l'imprimante.

Principe de fonctionnement succinct

Il consiste à enregistrer temporairement, dans un volume, une succession de fichiers correspondant, chacun, à un texte à imprimer.

L'impression de ces fichiers s'effectue en parallèle avec l'exécution des programmes ou en différé.

Ces opérations exigent, au préalable, l'activation du système de gestion des impressions (le module d'extension **SPLO1** doit être présent dans la configuration du système).

SPOOL offre quatre commandes possibles :

- activation du système de gestion des impressions ;
- insertion d'un fichier en queue de la file des fichiers à imprimer ;
- abandon d'une liste en cours d'impression ;
- désactivation du système de gestion des impressions.

Activation du système d'impression

Impression simultanée sur l'imprimante implicite (par exemple **IM0**) après ouverture d'un fichier liste.

Le fichier liste est créé sur le support implicite système.

SPOOL,AC

Ou sur le support MD1.

SPOOL,AC,TEMP=MD1

Impression simultanée sur l'imprimante **IM1** après ouverture du fichier liste sur le disque système.

SPOOL,AC,DEST=IM1

Impression différée, c'est-à-dire lors de la fermeture du fichier liste.

SPOOL,AC,LDIF

Si l'on ne désire pas de sortie sur imprimante.

SPOOL,AC,NL

Cette possibilité permet de travailler sur un système dépourvu d'imprimante.

Insertion d'un fichier liste

Permet d'ajouter un fichier source (éditable) dans la file des fichiers listes à imprimer.

Ce fichier sera édité sur l'imprimante implicite (par exemple **IM0**).

SPOOL,IL,DEMO-ASM

Ou sur l'imprimante **IM1**.

SPOOL,IL,DEMO-ASM,DEST=IM1

> Le fichier à imprimer doit nécessairement se trouver sur le support spécifié lors de l'activation du système de gestion des impressions (SPOOL,AC).

Abandon d'une liste en cours

Cette commande provoque l'abandon de l'impression en cours et le passage à la liste suivante éventuellement en attente dans la file.

Cette opération concerne l'imprimante implicite.

```
SPOOL,AB
```

Ou une imprimante explicite, par exemple **IM1**.

```
SPOOL,AB,DEST=IM1
```

Inhibition du système de gestion des impressions

Cette commande peut être frappée à tout moment. Elle agit de manière différée après achèvement des opérations en cours (impression ou création d'un fichier liste).

Aucun fichier nouveau ne sera pris en compte après cette commande.

Dès la validation par <**rc**>, le message :

DESACTIVATION EN COURS

est affiché à l'écran.

```
SPOOL,DS
```

Ou sur une imprimante explicite.

```
SPOOL,DS,DEST=IM1
```

Résumé

SPOOL,AC [,DEST=**imp**] [,TEMP=**sup**] [,**option**] SPOOL,IL,**fichier** [,DEST=**imp**] SPOOL,AB [,DEST=**imp**] SPOOL,DS [,DEST=**imp**]	
AC	mot-clé spécifiant l'**activation** du système d'impression.
IL	mot-clé spécifiant l'**insertion** d'un fichier liste à imprimer.
AB	mot-clé spécifiant l'**abandon** d'une liste en cours d'impression.
DEST=	mot-clé optionnel. Il précède le mnémonique désignant le nom et le numéro d'unité d'impression.
imp	nom et numéro d'unité d'impression. IM0 ... IM4 l'unité, par défaut, est IM0.
TEMP=	mot-clé optionnel. Il précède le nom du support destiné à recevoir les files d'attentes d'impression.
option	il s'agit d'un des mots-clé ci-dessous :
NL	si l'on ne souhaite pas de sortie imprimante par exemple sur un système dépourvu d'imprimante.
LDIF	impression différée après fermeture du fichier utilisateur.
LISTS	impression simultanée après ouverture du fichier liste utilisateur. C'est cette option qui est prise par défaut.

ETAT DES PERIPHERIQUES SECTORISES

L'utilitaire STATUS

Les modules de gestion des périphériques sectorisés (disques dur, disquettes,...) du noyau environnement de PROLOGUE tiennent à jour des informations relatives à l'état opérationnel de ces périphériques.

Tout accès en lecture/écriture, tout incident de lecture ou d'écriture affectant ces périphériques est comptabilisé pour chaque unité disponible dans le système.

L'utilitaire **STATUS** permet de renseigner l'utilisateur sur le nombre et la qualité des accès disque qui ont été effectués depuis le chargement de PROLOGUE en mémoire.

STATUS <rc>

Donne l'état de l'unité implicite utilisateur sous forme d'un tableau affiché à l'écran :

SUPPORT **FL.1** **ERR 02 en LEC.** Nr PISTE **149** Nr SECT, **3**

ECHANGES	ERR LEC.	ERR ECR.	ERR POSIT.	SECT.INV.
1895	**5**	**0**	**0**	**0**

Interprétation du tableau

Nous venons de visualiser le status de l'unité implicite utilisateur FL1.

Il y a eu, depuis le démarrage de PROLOGUE, **1895 accès lecture/ écriture.**

5 erreurs de lecture.

La **dernière erreur** est survenue lors de la lecture du **secteur 3** de la **piste 149.**

On constate, dans l'exemple cité, qu'aucune erreur d'écriture ni de positionnement n'est apparue.

La colonne **SECT.INV.** n'est utilisée que si l'on s'adresse à un disque dur. Elle permet de connaître le nombre de secteurs invalides.

STATUS,LIS=LO

Edition du status de l'unité implicite utilisateur sur l'imprimante implicite du poste de travail.

STATUS, MD0

Afficher le status de l'unité de disque dur **MD0.**

```
SUPPORT MD.0  ERR 02 en LEC.  Nr PISTE  667  Nr SECT 9
ECHANGES    ERR LEC.   ERR ECR.   ERR POSIT.   SECT.INV.
----------  --------   ---------  ----------   ---------
  12507        10          0          0           3

Secteurs invalides    piste
        3               12
        9               120      liste des secteurs invalides
        1               132
```

Une petite différence ici avec l'apparition de **3** secteurs invalides.

Dans ce cas, des secteurs de remplacement ont été substitués automatiquement aux secteurs invalides du disque dur **MD0.**

Ces substitutions ont été effectuées d'une manière transparente pour l'utilisateur.

L'utilisation de **STATUS** permettra de vérifier que le nombre de secteur invalides n'augmente pas dans des proportions suspectes.

Attention

La gestion des statistiques par les modules de l'environnement étant optionnelle, l'appel de l'utilitaire STATUS peut, sur certains systèmes, entraîner l'affichage d'un message informant l'utilisateur de l'absence de cette fonction.

Résumé

STATUS [**unité**] [,LIS=LO]	
unité	désigne le nom de l'unité de disque dont on désire connaître l'état. L'unité implicite utilisateur est prise par défaut.
LIS=LO	mot-clé spécifiant l'édition du status sur l'imprimante implicite du poste de travail.
	INTERPRETATION DU TABLEAU
SUPPORT	indique le NOM et le Numéro de l'unité examinée.
ERR xx	xx représente le numéro de la dernière erreur détectée.
Nr PISTE	donne le numéro de la piste concernée.
Nr SECT	donne le numéro du secteur de la piste ci-dessus.
ECHANGES	représente le nombre total d'opérations de lecture et d'écriture depuis le chargement de PROLOGUE.
ERR LEC	nombre cumulé d'erreurs de lecture.
ERR ECR	nombre cumulé d'erreurs d'écriture.
ERR POSIT	nombre cumulé d'erreurs de positionnement de piste.
SECT INV	nombre total de secteur invalides (ne concerne que les les disques durs).

TRANSMISSION DE FICHIERS

L'utilitaire TELE

TELE permet le transfert de fichiers entre deux systèmes équipés de PROLOGUE et d'une liaison **V24 (RS232C)**.

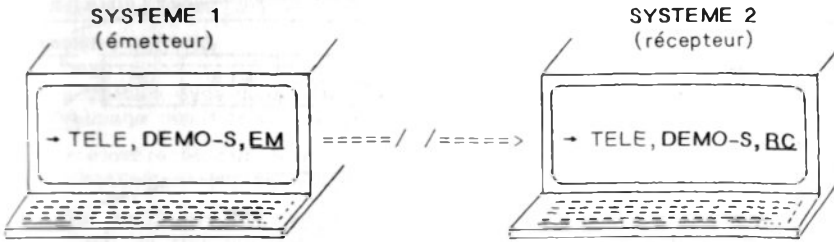
Cet utilitaire n'a de sens que si les supports (disquettes notamment) de ces deux systèmes sont incompatibles ou distants.

C'est le cas, par exemple, d'un système dont les disquettes sont à sectorisation logicielle et l'autre à sectorisation matérielle. Il peut aussi s'agir de disquettes à sectorisation logicielle mais dont les caractéristiques sont incompatibles.

Conditions à remplir pour qu'une transmission puisse s'effectuer :

- les deux systèmes doivent transmettre/recevoir à la même vitesse et être configurés selon les valeurs ci-après, le récepteur prêt avant l'émetteur :
  - . vitesse : **9600 Bauds** par défaut
  - . données : **8 bits**
  - . parité : **sans**
  - . stop : **2**
- que la liaison physique soit réalisée avec un câble approprié selon le principe de base ci-dessous :

Fonction	Pin	Nom	Connexion	Nom	Pin
Emission	2	TD	—— / / ——→	RD	3
Réception	3	RD	←—— / / ——	TD	2
Demande pour émettre	4	RTS	——→ + + ←——	RTS	4
Prêt à émettre	5	CTS	←—— + + ——→	CTS	5
Poste données Prêt (modem)	6	DSR	←—— + + ——→	DSR	6
Détection Porteuse	8	CD	←—— ——→	CD	8
terminal prêt (micro)	20	DTR	——→ + + ←——	DTR	20
Masse	1-7	SG	←—— / / ——→	SG	1-7



Le fichier **DEMO-S** est transmis du système 1 vers le système 2.

Quelques mots sur le protocole de transmission

Pour transmettre les fichiers d'un site à un autre, **TELE** utilise un protocole de transmission relativement simple :

- les fichiers sont transférés par blocs de 256 octets ;
- chaque bloc est précédé d'un caractère spécial **STX** (02) signalant le début du bloc et se termine par un autre caractère spécial **ETX** (03).

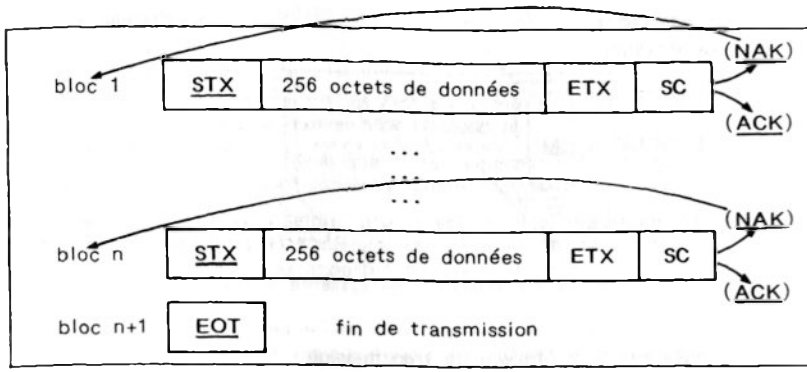
ETX est suivi d'un caractère utilisé comme somme de contrôle. Ce caractère est le résultat d'un **'OU' exclusif** des 258 caractères qui le précède (STX à ETX inclus).

C'est la comparaison de ce caractère avec la somme de contrôle calculée lors de la réception qui détermine une éventuelle erreur de transmission.

A la réception, si une erreur est détectée, le récepteur répond par un accusé de réception négatif **NAK** (15) et l'émetteur retransmet le même bloc de données.

Si aucune erreur n'est détectée, le récepteur répond par un accusé de réception positif **ACK** (06) et l'émetteur transmet le bloc suivant.

La fin de transmission est signalée par un caractère début de bloc particulier : **EOT** (04) ; dans ce cas, le récepteur ferme le fichier et TELE rend le contrôle à l'utilisateur.



Résumé

TELE,fich, EM [,vitesse]
TELE,fich, RC [,vitesse]

fich	désigne le nom du fichier à émettre ou à recevoir.
EM	mot-clé spécifiant l'EMISSION.
RC	mot-clé spécifiant la RECEPTION.
vitesse	valeur de la vitesse de transmission exprimée en bauds, 9600 bauds est pris par défaut.

Pour abandonner la transmission, on utilise la touche **ESC**, ce qui entraîne la suppression du fichier reçu.

Remarque : si un fichier séquentiel indexé doit être transmis par TELE, il faut au préalable effectuer une sauvegarde de ce fichier par **SVSI** et transmettre le fichier résultant de type -U.

TELE est également utilisable avec un modem en full-duplex.

REORGANISATION DES VOLUMES

L'utilitaire REORG

La gestion dynamique des volumes par le système PROLOGUE impose un découpage des fichiers en blocs (nommés extensions).

Le nombre de ces extensions est limité à 18 pour chaque fichier et tout débordement provoque une erreur (45).

Afin de récupérer l'espace disque disponible mais non utilisable (le volume est un vrai gruyère à cet instant), **REORG** effectue le regroupement des extensions de sorte à disposer d'un maximum d'espace disque linéaire.

Cet utilitaire, écrit en langage **BAL**, ne demande aucune autre ressource que le volume à traiter ; il essaie de garantir l'intégrité des données, notamment en cas de reprise après une coupure de courant.

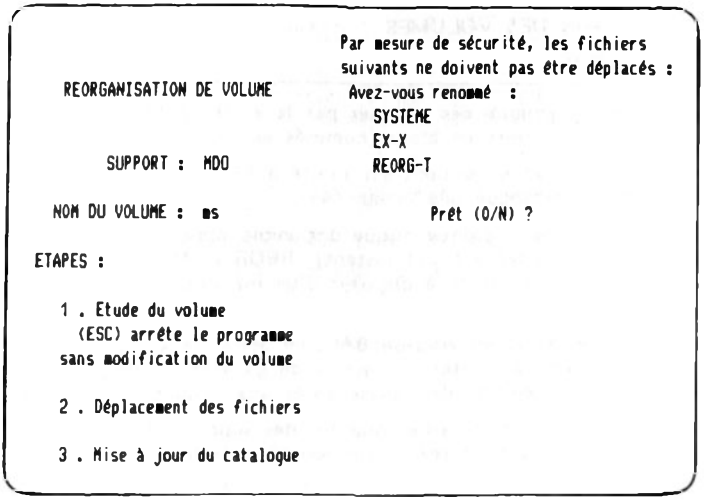
Il ne peut toutefois travailler que sur des supports propres, c'est-à-dire exempts d'erreurs d'entrée-sortie non récupérables.

REORG <rc>

L'écran ci-dessous est alors affiché :

```
Réorganisation de volume  Version 1.0 Provisoire du 10/07/1984
Copyright (c) 1984 BULL Micral



                              ATTENTION

                        CONTEXTE MONO-POSTE

                           OBLIGATOIRE


     Les erreurs dues au support peuvent provoquer une destruction
    partielle de certains fichiers. Dans le doute, il est conseillé
    d'effectuer une sauvegarde avant le processus de réorganisation

(RET) pour continuer, (ESC) pour arrêter
```

Un dialogue est établi ; il suffit de répondre aux questions posées puis de valider l'opération à la question **Prêt (O/N) ?**



Le programme **REORG** demande dans l'ordre :

- l'identification du support à traiter (MD0, FL0, etc.) ;
- la validation des noms de fichiers qu'il ne doit pas restructurer (fichiers **SYSTEME, EX-X** et **REORG-T**). Il est impératif que ces fichiers ne soient pas déplacés afin de permettre une reprise éventuelle en cas d'incident ;
- l'autorisation de commencer le traitement.

> La durée du traitement est liée à la capacité du volume, au nombre de fichiers qu'il contient et ... à son désordre. Cette opération peut nécessiter une demi-heure ou davantage.

La première phase (la plus longue) est celle de l'étude du volume ou **REORG** détermine la structure future du volume et la liste des transferts à effectuer.

La deuxième phase consiste à effectuer les déplacements des fichiers.

La troisième phase est celle de la mise à jour du répertoire.

En cas d'interruption involontaire du traitement (panne de secteur, par exemple), il suffit de relancer l'exécution de REORG, la gestion des reprises étant automatique.

En cas d'erreur liée à la disponibilité des données (support non monté, périphérique protégé en écriture,...), un message d'attention est affiché et le programme est en attente d'une intervention de l'utilisateur. Il

est toutefois possible d'abandonner le traitement en appuyant sur la touche **ESC**ape.

En cas d'erreur grave (lecture et/ou écriture impossible), le traitement dépend de la phase dans laquelle se trouve REORG.

Si l'on est dans la phase d'analyse, il y a abandon.

Si l'on est dans une autre phase, un message signale à l'utilisateur qu'il y a eu impossibilité de transfert et indique le nom du fichier concerné sur lequel il risque d'y avoir une incohérence.

CHAPITRE 4 les décors

Comme au spectacle, PROLOGUE recrée le décor du système d'exploitation "hôte" et exécute les programmes qui sont habituellement supportés par ce système, comme si ce dernier était réellement présent en mémoire.

Ce mode de fonctionnement explique que le mot **DECOR** ait été emprunté au vocabulaire du théâtre pour désigner une fonction propre à PROLOGUE.

DECOR désigne un utilitaire qui émule les principales fonctions d'un système d'exploitation hôte de PROLOGUE.

DECOR CP/M	émule CP/M-86 version 1.12*
DECOR MS-DOS	émule MS-DOS version 1.25*

Le décor n'a pas pour but de recréer un système d'exploitation complet sous PROLOGUE, mais d'émuler en priorité les fonctionnalités standard du système hôte.

---

* - *CP/M est une marque déposée de Digital Research.*
* - *MS-DOS est une marque déposée de Microsoft.*

CHARGEMENT DES DECORS

Appel du décor MS-DOS

MSDOS <rc>

```
                                  tm
                    DECOR MS-DOS  Version 1.1a
                    Copyright (c) 1983 BULL Micral


                          Msdos / Prologue Units Equals :

                                * A: <=> MD0
                                * B: <=> MD1
                                * C: <=> FL0
A>dir c:
SYSTEME          40960   1-06-83  12:00p
SYSCONF  S         640   1-06-83  12:00p
SYST00   C         256   1-06-83  12:00p
/        X       19584   1-06-83  12:00p
CP       X       16384   1-06-83  12:00p
EX       X       24576   1-06-83  12:00p
A>
```

Appel du décor CP/M-86

CP/M <rc>

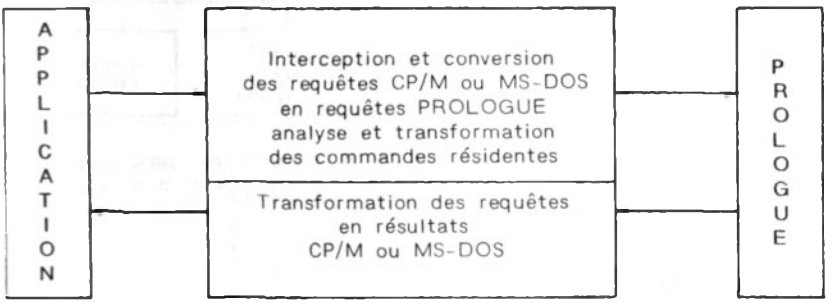
```
                                  tm
                    DECOR CP/M-86  Version 1.2b
                    Copyright (c) 1983 BULL Micral


                          CP/M-86 / Prologue Units Equals :

                                * A: <=> MD0
                                * B: <=> MD1
                                * C: <=> FL0
A>dir c:
C: SYSTEME       : SYSCONF  S    : SYST00   C    : /   X
C: CP        X   : EX        X
A>
```

PRINCIPE DE FONCTIONNEMENT

Le principe de fonctionnement du décor est relativement simple :

- un processus '**aller**' transforme les requêtes de type CP/M ou MS-DOS en requêtes PROLOGUE ;
- un processus '**retour**' transforme à son tour les résultats des requêtes PROLOGUE en résultats compatibles avec les requêtes CP/M ou MS-DOS.

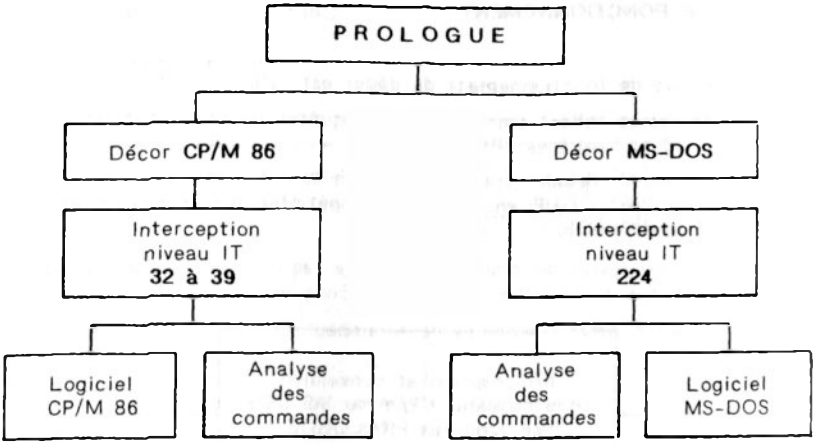
Outre ce processus de transformation de requêtes, le décor procède au traitement des commandes résidentes reconnues.



Les fonctionnalités de la plupart des commandes internes de CP/M-86 et MS-DOS sont traitées sous DECOR.

Par contre, les commandes ayant leurs équivalents sous PROLOGUE ou dépendant étroitement de caractéristiques matérielles spécifiques ne sont pas prises en compte par DECOR.

Les commandes telles que FORMAT, COPY, etc., sont réalisées par l'utilitaire CP-X appelable directement sous DECOR.



Les commandes internes telles que DIR, ERA, TYPE, REN sont émulées ainsi que l'usage de certaines touches comme ctrl C, P, S, etc.

MISE EN OEUVRE DES DECORS

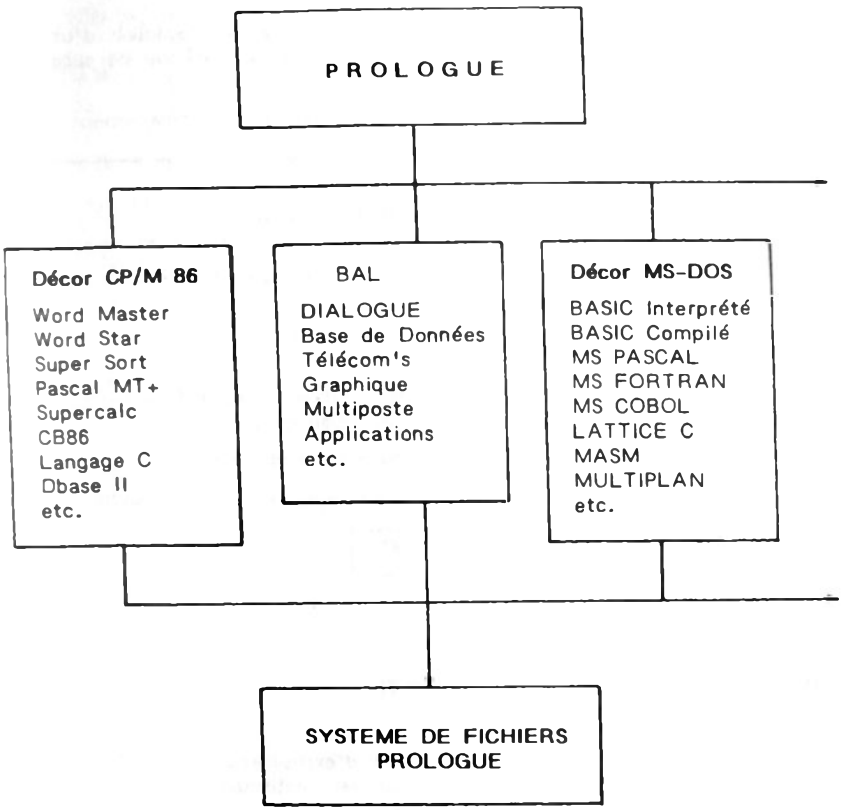
Afin de ne pas pénaliser les utilisateurs de PROLOGUE, les logiciels hôtes peuvent être lancés directement à partir de l'interpréteur de commande de PROLOGUE, sans avoir à appeler le décor approprié.

Cette possibilité est due au fait que l'interpréteur de commande reconnaît l'origine du fichier exécutable à partir de son suffixe.

Système	Suffixe
PROLOGUE	X, T, ASG
MS-DOS	COM ou EXE
CP/M-86	CMD

Grâce à ce principe, l'utilisateur étant sous contrôle de l'interpréteur de commande de PROLOGUE peut appeler un programme exécutable d'origine quelconque. Le décor approprié sera chargé en fonction du type de fichier, puis le programme sera sollicité.

-> **MP <rc>**	Charge de **MULTIPLAN** via décor **MS-DOS** - **MP** est du type **.COM**
-> **WM,SOURCE-ASM <rc>**	Charge **WM** via décor **CP/M** **WM** est du type **.CMD**



Convergence des systèmes d'exploitation vers PROLOGUE

UTILISATION DES CARACTERES NATIONAUX

DECOR est un utilitaire de PROLOGUE et, à ce titre, il gère les caractères du standard PROLOGUE (voir tableau en annexe). Ainsi, un programme d'application conçu dans ce contexte peut accéder à tous les caractères nationaux.

Cette flexibilité rend moins aisée l'exécution de certains logiciels étrangers limités aux seuls caractères ASCII codés sur 7 bits ou utilisant le 8ème bit pour un usage interne.

Pour remédier à ces difficultés et permettre à ces logiciels d'utiliser les caractères nationaux, les DECORS ont adopté la méthode de substitution de codes.

L'utilisateur peut choisir un des deux modes de fonctionnement :

ASCII standard (sans caractères accentués).

Substitution des codes ASCII (caractères nationaux).

Dans le deuxième mode, il peut accéder aux caractères :

é è § ç à ° ¨ ù

Ces deux modes sont exclusifs, on ne peut avoir en même temps à l'écran un caractère accentué substitué au code original.

Au chargement du DECOR, on est dans le mode ASCII non accentué.

Pour passer d'un mode à l'autre, il faut appuyer sur la touche

contrôle ⓐ

PARTAGE DES FICHIERS EN MULTIPOSTE

CP/M-86 et MS-DOS sont des systèmes d'exploitation monotâches et monopostes. De ce fait, il n'existe pas, sur les systèmes natifs, de primitives qui permettent le partage des fichiers.

Sous DECOR, les fichiers sont ouverts en mode **EXCLUSIF**, ce qui peut nuire au bon fonctionnement de certains programmes dans une configuration multiposte.

Si, sur le poste 0, on lance la commande :

```
EDLIN,SOURCE-TXT (appel de l'éditeur ligne MS-DOS)
```

EDLIN est ouvert en mode partageable.
SOURCE-TXT est ouvert en mode exclusif.

Puis, sur le poste 1 :

```
EDLIN,ESSAI-TXT
```

EDLIN étant ouvert en mode partageable, la commande est acceptée, mais si par contre on lance la commande :

```
EDLIN,SOURCE-TXT
Erreur 44 fichier non partageable
```

sera affiché à l'écran car **SOURCE-TXT** est déjà ouvert en exclusif sur le poste 0.

RESERVATION DE MEMOIRE

Une gestion de mémoire est introduite sous DECOR pour l'utilisation de logiciels natifs dans un contexte multiposte.

En standard, le DECOR met à la disposition de l'utilisateur un maximum de **128 K octets** de mémoire pour l'exécution de ses programmes. Cette limite peut cependant être modifiée en fonction de la capacité mémoire de la machine.

```
L'unité de mémoire est exprimée en "paragraphe"
(1 paragraphe = 16 octets).
```

Les deux premiers octets du secteur 3 du fichier décor-X contiennent la taille maximale allouée au décor (**2000**h par défaut, soit **128 K octets**).

Pour changer cette valeur, on utilisera l'utilitaire **PATCH-X.**

Taille K octets	Valeur
64	1000h
128	2000h
192	3000h
256	4000h

PARAMETRES DE DECOR

Mémoire allouée au DECOR

Cette valeur est inscrite dans les **octets 0** et **1** du **secteur 3** du fichier décor-X.

Table de transcodage des caractères accentués

Cette table débute à l'octet **4** du **secteur 3** du fichier décor-**X.** Elle permet le transcodage jusqu'à **16** caractères et doit se terminer par un caractère nul (00).

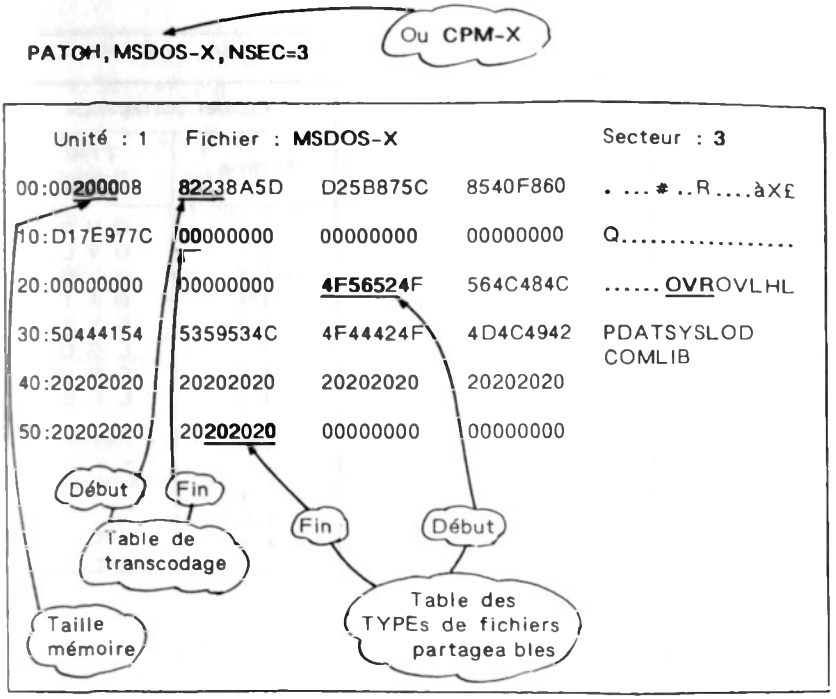
En standard, elle est initialisée avec les caractères accentués français. On y accède avec l'utilitaire **PATCH-X** en y inscrivant le nouveau code interne (8 bits PROLOGUE) suivi du code externe (7 bits) à substituer.

> Le codage des caractères accentués du DECOR étant différent de celui de PROLOGUE, il est donc indispensable qu'un texte saisi sous décor soit édité ou imprimé sous décor. L'édition ou l'impression de ce texte sous PROLOGUE donnera des graphismes différents à la place des caractères accentués.

Table des types de fichiers partageables

Elle débute à l'**octet 28h** du **secteur 3** du fichier décor-**X.** Elle peut contenir treize suffixes ; la table doit se terminer par un espace.

Localisation des paramètres DECOR



Dans cet exemple, les fichiers de type :

-OVR -OVL -HLP -DAT -SYS -LOD -COM -LIB

sont partageables.

Valeurs standard par défaut

FICHIER DECOR **secteur 3** CP/M-X ou MS-DOS X					
Table de substitution				**Fichier partageable**	
Adresse	**CODE ASCII**			**Adresse**	**TYPE fichier**
	Interne		**Externe**		
0 4	8 2	é	2 3	2 8	O V R
0 6	8 A	è	5 D	2 B	O V L
0 8	D 2	§	5 B	2 E	H L P
0 A	8 7	ç	5 C	3 1	D A T
0 C	8 5	à	4 0	3 4	S Y S
0 E	F 8	°	6 0	3 7	L O D
1 0	D 1	¨	7 E	3 A	C O M
1 2	9 7	ù	7 C	3 D	L I B
1 4	0 0			4 0	- - -
1 6	- -	-	- -	4 3	- - -
1 8	- -	-	- -	4 6	- - -
1 A	- -	-	- -	4 9	- - -
1 C	- -	-	- -	4 C	- - -
1 E	- -	-	- -	4 D	- - -

TRANSFERT DE FICHIERS CP/M-86

L'utilitaire LCPM

L'utilitaire **LCPM-X** permet de transférer un logiciel CP/M-86 dans un volume PROLOGUE.

Avant d'effectuer le transfert d'un fichier issu d'une disquette CP/M-86, il est souhaitable de visualiser le répertoire de cette disquette.

LCPM,DIR,FL0

Affiche à l'écran le catalogue de la disquette CP/M-86 montée sur le lecteur 0.

Puis le choix du fichier à transférer dans le volume PROLOGUE étant fait :

LCPM,FL0.DEMO-BAS,MD0

Le fichier **DEMO.BAS** situé sur le lecteur de disquettes 0 est transféré dans le volume monté sur l'unité **MD0**.

Résumé

LCPM, **fich1**, **fich2** [,**Ux**] LCPM, **DIR**, **sup**

fich1	désigne le support et le nom du fichier à transférer.
fich2	désigne le nom du support et du fichier à créer.
Ux	mot-clé **U** suivi du numéro de USER x. Obligatoire si le fichier se trouve dans un USER différent de 0.
DIR	mot-clé spécifiant la demande d'édition du répertoire de la disquette.
sup	désigne le nom et l'unité où se trouve la disquette.

TRANSFERT DE FICHIERS MS-DOS

L'utilitaire LMSDOS

L'utilitaire **LMSDOS-X** permet de transférer un logiciel MS-DOS dans un volume PROLOGUE.

Avant d'effectuer le transfert d'un fichier issu d'une disquette MS-DOS, il est souhaitable de visualiser le répertoire de cette disquette.

LMSDOS,DIR,FL0

Affiche à l'écran le catalogue de la disquette MS-DOS montée sur le lecteur 0.

Puis le choix du fichier à transférer dans le volume PROLOGUE étant fait :

LMSDOS,FL0.DEMO-BAS,MD0

Le fichier **DEMO.BAS** situé sur le lecteur de disquettes 0 est transféré dans le volume monté sur l'unité **MD0**.

Résumé

LMSDOS, **fich1**, **fich2**	
LMSDOS, **DIR**, **sup**	
fich1	désigne le support et le nom du fichier à transférer.
fich2	désigne le nom du support et du fichier à créer.
DIR	mot-clé spécifiant la demande d'édition du répertoire de la disquette.
sup	désigne le nom et l'unité où se trouve la disquette.

EXEMPLES D'UTILISATIONS

Transferts de fichiers CP/M-86

```
LCPM, MBASIC-COM, MD0
LCPM, FL0.BASCOM-COM, MD0
LCPM, FL0.EXEMPLE-A86, FL1.SOURCE-ASM
LCPM, FL1.PROG1-BAS, PROG1-BAS, U5
```

Transferts de fichiers MS-DOS

```
LMSDOS, MBASIC-COM, MD0
LMSDOS, FL0.BASCOM-COM, MD0
LMSDOS, EXEMPLE-BAS, MD0.EXEMPLE-S
```

LISTE DE LOGICIELS SOUS DECOR

Logiciels fonctionnant sous DECOR CP/M-86

- MS BASIC Interprété
- MS PASCAL
- PASCAL MT+
- C BASIC
- DBASE II
- DDT86
- ASM86
- GENCMD
- SID86
- SUPERCALC
- WORDSTAR
- WORDMASTER
- LINK.EXE idem
- LIB.EXE idem
- CREF.EXE idem

Logiciels fonctionnant sous DECOR MS-DOS

- MULTIPLAN
- MS BASIC Interprété
- MS BASIC Compilé
- MS PASCAL
- MS COBOL
- LATTICE C
- DEBUG.COM
- EDLIN.COM
- MASM.EXE
- LINK.EXE
- LIB.EXE
- CREF.EXE

CHAPITRE 5 | le langage B.A.L.

BAL (Business Application Language) est un langage de programmation orienté gestion avant tout, mais aussi d'un usage polyvalent, performant, riche en instructions, d'une grande souplesse d'utilisation et capable de satisfaire les programmeurs les plus exigents.

S'il rappelle le **BASIC** par certains verbes, **BAL** ne lui ressemble guère.

> Les variables doivent être déclarées avant leur utilisation. Par rapport au BASIC, ce principe impose, à l'utilisateur, un peu plus de rigueur dans l'écriture de ses programmes.
>
> Les instructions sont décrites dans des segments.

Si, pour les habitués du Basic ces deux points paraissent contraignants, il n'en est rien, car le fait de structurer les données n'apporte que des avantages, surtout lorsqu'il s'agit d'assurer la maintenance des programmes.

La production d'un programme BAL nécessite trois phases :

- une phase de **création** d'un fichier source ;
- une phase de **traduction** ;
- une phase d'**exécution.**

Pour être exécutable, un programme BAL doit d'abord être constitué dans un fichier au moyen d'un éditeur de texte, puis ce fichier de type source (-S) doit être soumis à un utilitaire appelé **TRADUCTEUR BAL** (TR-X) lequel engendre un fichier de même nom que le fichier source, mais avec le suffixe -T et dont le contenu est un code intermédiaire appelé **T-CODE.**

Le fichier (-T) est ensuite soumis pour exécution à un autre utilitaire appelé **EXECUTEUR BAL** (EX-X) lequel va dérouler les instructions.

STRUCTURE D'UN PROGRAMME BAL

Un programme **BAL** est constitué de **DECLARATIONS** de variables et d'**INSTRUCTIONS** opérant sur ces variables.

Les **VARIABLES** sont déclarées et peuvent être structurées.
Les **INSTRUCTIONS** sont écrites dans des **SEGMENTS.**
Le segment représente l'unité de programmation.

```
PROGRAM "nom du programme"

        ;DECLARATION DES VARIABLES
        DCL A$=10,B#,trav%
        DCL Z$=20
        FIELD=M,Z
        DCL Z1$=2
        DCL Z2=12
        DCL Z3$=1(10)
        FILLER=5
        DCL z4%

        ;CORPS DU PROGRAMME
        SEGMENT 0
              A="chaîne"
        10    FOR I=1 TO 10
              PRINT=1:CLEAR,TABV(1),I,TAB(I),A,"***"
              NEXT I
        LDGO.SEG 5
              GOTO 10
        ESEG 0

        SEGMENT 5
              PRINT=1:"SEGMENT 5"
        RET.SEG
        ESEG 5

END
```

Résumé

TR,DEMO	effectue la traduction de **DEMO-S** et produit un fichier **DEMO-T.**
TR,DEMO,LIS=LO	effectue la traduction de DEMO-S, produit un fichier DEMO-T et édite, sur l'imprimante assignée au poste, la liste du fichier traduit.
EX,DEMO	l'exécuteur **EX-X** est chargé en mémoire puis, à son tour, charge le fichier **DEMO-T** et l'exécute.
EX,DEMO,DB	l'exécuteur **EX-X** est chargé en mémoire ainsi que le fichier DEMO-T, mais le contrôle est donné au **DEBUG** pour aider à la mise au point.
-> **DEMO**	est une manière d'exécuter un programme BAL. PROLOGUE se rendant compte que le fichier **DEMO-T** existe, charge automatiquement l'exécuteur **EX** qui, à son tour, chargera DEMO-T et l'exécutera.

INSTRUCTIONS GENERALES ET BRANCHEMENTS

Instruction	Fonction	Exemple
LET	Affectation.	LET A=10 ou A=10 LET A="chaîne"
DATE	Mise à jour date et heure.	DATE(1)="84" DATE(8)="5" année ..... 1/10e seconde
DATA	Déclaration de données en mémoire.	DATA 10,"texte",1256,15.56
READ	Lecture données situées en mémoire.	READ=0:A,B,C(I)
RESTORE	Repositionner le pointeur READ.	RESTORE
OP	Saut si caractère frappé au clavier.	OP 1000,car
GOTO	Branchement inconditionnel.	GOTO 1000
IF	Branchement sur condition.	IF B(i)="ABC" GOTO 1000 IF V=5 THEN 1000 ELSE 2000
GOSUB	Branchement vers sous-programme.	GOSUB 1000 :::: 1000 ::::
RETURN	Fin de sous-programme.	1000 :::: RETURN
FOR NEXT STEP	Exécution de boucle	FOR I=1 to 100 STEP j NEXT I
LOAD	Chargement d'un programme objet.	ASSIGN=1,"FL0.SPR1" LOAD=1,/E000
CALL	Appel d'un sous-programme objet.	CALL PARAM,/E000
OF v GOTO	Branchement indexé.	OF v GOTO 10,20,30,500 1 2 3 4 ....
ON v GOTO	Branchement signé.	ON v GOTO 100,200,300 < = >
ON ERROR GOTO	Branchement sur erreur.	ON ERROR GOTO 1000,e
RESUME	Fin traitement ON ERROR GOTO.	RESUME 1000

DIRECTIVES ET INSTRUCTIONS DE CONTROLE

Instruction	Fonction	Exemple
PROGRAM	Début d'un programme.	PROGRAM "DEMO-BAL"
DCL	Déclaration de variables.	DCL N#,VAR%,B#(256) DCL CHAIN$=80
FIELD	Structuration de variables, permet de réaliser des équivalences.	DCL T#(32) FIELD=M,t(10) ::: DCL i# ..... i est équivalent à t(10) FIELD=4,MEM$ ... MEM est en mémoire virtuelle
FILLER	Réserve de l'espace sans création de variable.	FILLER=8 ... saut de 8 octets
SEGMENT	Début d'un segment de programme.	SEGMENT 2 SEGMENT 6,nolist
ESEG	Fin de segment.	ESEG 5
END	Fin de programme.	END
LDGO.SEG	Chargement et exéc. d'un segment progr.	LDGO.SEG 3 LDGO.SEG i
RET.SEG	Retour de segment	RET.SEG
LOAD.GO	Chargement et exéc. utilitaire PROLOGUE.	ASSIGN=1,"MD0.CP" LOAD.GO=1,"DF,MD0.DEM-S,FL0
CHAIN	Enchaînement d'un programme BAL.	ASSIGN=1,"PRG1" ::: CHAIN=1 PRG1 est exécuté sur STOP après le programme en cours.
STOP	Arrêt d'un programme enchaînement progr. si CHAIN demandé.	STOP
WAIT	Suspension exécut.	WAIT 3 ;attente 3 sec.
PAUSE	Suspension exécut.	PAUSE "<rc> pour continuer"

FONCTIONS ARITHMETIQUES

Instruction	Fonction	Exemple
CONV	Conversion donnée ascii → interne	R=CONV(V)
VAL	Valeur numérique d'une chaîne.	R=VAL("ABC-45.78 DEF") R=-45,78
STRN	Conversion donnée interne → ascii	V=STRN(R)
ABS	Valeur absolue.	R=ABS(V)
INT	Valeur entière.	R=INT(V)
MOD	Valeur absolue du reste division.	R=MOD(V,M)
ROUN	Arrondi.	R = ROUN(V,P)
FIX	Cadrage numérique.	R=FIX(V,L)
FP	Partie décimale.	R=FP(V)
SGN	Donne le signe.	R=SGN(V)
RND	Donne un nombre aléatoire.	R=RND(1000) R=RND

FONCTIONS MATHEMATIQUES

Ces fonctions nécessitent, pour être exécutées, la présence du module "bibliothèque mathématique" dans l'exécuteur BAL.

Instruction	Fonction	Exemple
SIN	Donne le sinus d'un angle exprimé en radian.	R=SIN(V)
COS	Donne le cosinus d'un angle exprimé en radian.	R=COS(V)
TAN	Donne la tangente d'un angle exprimé en radian.	R=TAN(V)
ATN	Donne l'arc tangent en radian.	T=ATN(V)
SQR	Donne la racine carrée d'un nombre.	R=SQR(V)
EXP	Donne l'exponentielle d'un nombre.	R=EXP(V)
LOG	Logarithme népérien d'un nombre.	R=LOG(V)

ONCTIONS SUR CHAINES DE CARACTERES

Instruction	Fonction	Exemple
LEN	Longueur déclarée d'une chaîne.	DCL V$=10 R=LEN(V) → 10
LEN$	Longueur utilisée d'une chaîne.	V="ABCD" R=LEN$(V) → 4
CHR$	Conversion binaire ascii.	A=CHR$(/41) → "A"
LEFT	Extraction gauche.	R=LEFT(V,N)
RIGHT	Extraction droite.	R=RIGHT(V,N)
SUBSTR	Extraction sous-chaîne.	R=SUBSTR(V,P,L)
MOVE	Transfert de zone en mémoire.	R=MOVE(V,P,L)
SHR	Cadrage à droite d'une chaîne.	R=SHR("P.GIR ") -> "P.GIR"
SHL	Cadrage à gauche d'une chaîne.	R=SHL(" P.GIR") -> "P.GIR"
GENER	Génération d'une suite de caractères.	R=GENER(5,"*") → *****
SPACE	Génération d'espace.	R=SPACE(6) = GENER(6," ")
INDEX	Donne la position d'un caractère.	R=INDEX(C1,C2) R=INDEX("P.GIR", "I")
INSTR	Test d'inclusion d'une chaîne.	R=INSTR(C1,C2,N) R=INSTR("P.GIR", "GI",1)
INCLUD	Substitution de chaîne.	R=INCLUD(N,C,L)
INV	Inversion d'une chaîne.	V="1234567890" R=INV(V) -> 0987654321
TRAN	Transcodage chaîne de caractères.	R=TRAN(T,S,L,V)
SMALL	Force chaîne en minuscules.	R=SMALL ("P.GIR-45") -> "p.gir-45"
LARGE	Force chaîne en majuscules.	R=LARGE("p.gir-45") -> "P.GIR-45"

FONCTIONS SYSTEME

Instruction	Fonction	Exemple
PROCESS	Donne le numéro du poste où se déroule le programme.	R=PROCESS
COL	Donne la position du curseur colonne.	R=COL(n° logique)
LIN	Donne la position du curseur ligne.	R=LIN(n° logique)
CONF	Donne les paramètres du système.	R=CONF(i) i=1 nombre lignes écran 2 nombre colonnes écran 3 nombre lignes/page imp. 4 nombre col/ligne impr. 5 code langue 6 version PROLOGUE 7 indice version 9 nombre de touches fonct.
DATE	Donne la date et heure courante.	R=DATE(i) i=1 année 2 mois 3 jour/mois 4 quantième 5 heure 6 minute 7 seconde 8 dixième seconde
VPTR	Donne l'adresse d'une variable.	I=VPTR(V)
PEEK	Lecture du contenu d'une case mémoire.	I=PEEK(M)
POKE	Ecriture dans une case mémoire.	M=POKE(I)
INP OUT	Lecture d'un port. Ecriture dans un port.	INP V,P OUT P,V
IO	Entrée/sortie. directe/disque.	IO=1,/40,S:100,E,D(1),512

EDITION ET SAISIE DE DONNEES

Instruction	Fonction	Exemple
PRINT	Edition données. Ecran ou Imprimante.	PRINT=1:"TEXTE" PRINT=2:"TEXTE" PRINT=p:VAR
ASK	Entrée de données à partir du clavier.	ASK=1:=VAR ASK=1:"TEXTE"=VAR
MASK	Détermine le mode de contrôle de saisie des données.	MASK 2 ASK=1: ....
MASK 001	Refuse tout caractère invalide.	
MASK 002	Emission d'un beep à chaque caractère invalide.	
MASK 004	Annule saisie si erreur et saisie obligatoire.	
MASK 008	<rc> non exigé si zone de saisie complète.	
MASK 016	Débranchement sur caractère invalide E=étiq.	
MASK 032	Pas de modif. de la variable si 1er caractère = <rc>.	
MASK 064	Caractères frappés non affichés à l'écran.	
MASK 128	Interdit le débranchement sur touche **ESC**ape.	
HOME	Curseur HOME.	PRINT=p:HOME, ...
CLEAR	Curseur HOME et effacement écran.	PRINT=p:CLEAR, ...
TAB	Positionne curseur	PRINT=p:TAB(10) .. TAB(X,Y)
TABV	Saut de ligne.	PRINT=p:TABV(2),"TEXTE"
BELL	Emission beep.	PRINT=p:BELL, ...
PAGE	Saut de page.	PRINT=p:PAGE, ...
ATB	Définit un attribut de visualisation.	PRINT=p:ATB(a),"TEXTE", ... ASK=1:ATB(1),"???",ATB(0)=V
ATB(0)	Attribut normal.	
ATB(1)	Préférentiel pour le périphérique utilisé.	
ATB(2)	Inversion.	
ATB(3)	Clignotement.	
ATB(4)	Souligné.	
ATB(5)	Sous-brillance.	
ATB(6)	Sur-brillance.	
ATB(7)	Force majuscules.	
ATB(8)	Force minuscules.	
ATB(9)	Recopie écran.	
ATB(10)	Ecriture penchée.	
ATB(11)	Ecriture large.	
ATB(21)	Effacement de la fin de ligne depuis le curseur.	
ATB(22)	Effacement de la fin de page depuis le curseur.	

Formats de saisie/d'édition de données

Instruction	Fonction	Exemple
FM	Définit un format associé à une variable.	S=FM(/1,X3,ZZ,N.ZZ) PRINT=1:((S)),R1,R2
FMT	Permet de spécifier un format externe à l'instruction PRINT.	10 FMT (/1,X3,ZZ,N.ZZ) PRINT=1,10:R1,R2

SYMBOLES ENTRANT DANS LA COMPOSITION DES FORMATS

Symbole	Définition
U	Tout caractère accepté en saisie.
W	Idem U, sauf les fonctions du curseur.
A	Entrée alphabétique A-Z ou espace, longueur fixe.
D	Idem A, mais longueur variable.
Z	Numérique 0-9 zéros significatifs obligatoires.
N	Numérique 0-9 zéros non significatifs absents.
B	Alphanumérique obligatoire A-Z espace 0-9.
C	Idem B, mais longueur variable.
+	Signe + doit précéder le 1er chiffre significatif.
-	Signe - nécessaire avant le 1er chiffre significatif.
E	Entrée chaîne quelconque.
L	Cadrage à gauche.
F	Suppression des espaces à droite.
V	Séparateur décimal implicite.
*	Remplace les zéros non significatifs par *.
S(c)	Définit le séparateur des milliers.
S	Edite le séparateur des milliers d'un nombre.
Xn	Edition de n espaces.
/n	Saut de n lignes.
"TEXTE"	Chaîne de littéraux entre guillemets.

INSTRUCTIONS RELATIVES AUX FICHIERS

Instruction	Fonction	Exemple
ASSIGN	Assignation d'un n° log. à un fichier.	ASSIGN=1,"fichier",SQ ASSIGN=2,F1,BD,WR,EX:100,E
CFILE	Création d'un fichier.	CFILE=1,D=128:100,E CFILE=2:100,E
OPEN	Ouverture d'un fichier.	OPEN=1 OPEN=2:100,E
CLOSE	Fermeture d'un fichier.	CLOSE=1 CLOSE=2:100,E
DFILE	Suppression d'un fichier.	DFILE=1 DFILE=2:100,E
RENAME	Renommer un fichier.	RENAME=1,"NOUVEAU":100,E
EXTEND	Etendre un fichier.	EXTEND=1,200:100,E 200 secteur rajoutés
FIELD	Déclaration de variables en mémoire virtuelle.	FIELD=1 DCL A$=1(256) DCL N#,B%,c$=1(10) FIELD=1,C DCL c1$,c2$,c3$,c4$
ATB	Modification des attributs de partageabilité ou d'exclusivité.	ATB=1,EX
CATALOG	Recherche d'un fichier dans un volume.	CATALOG=1,"fichier" ... :100,E,VAR CATALOG=1,FICH,VAR
VOLUME	Caractéristiques d'un volume.	VOLUME=1:100,E,VAR VOLUME=1,VAR

La majorité des instructions relatives aux fichiers revêt la forme suivante :

INSTRUCTION	= numéro logique [:**ERR**] [variable] [,longueur]
:ERR	= numéro de ligne, variable
:100,E	= si erreur, aller à l'étiquette **100** et ranger le code erreur dans la variable **E**

INSTRUCTIONS RELATIVES AUX FICHIERS SEQUENTIELS INDEXES

Instruction	Fonction	Exemple
INSERT	Crée un article.	INSERT=1,clé,donnée,long INSERT=2,clé,index,donnée
SEARCH	Recherche et lecture article si la clé est trouvée.	SEARCH=1,clé,var,long
SEARCH.M **SEARCH.L** **SEARCH.ML**	Idem SEARCH, mais avec blocage de l'article. Enregistrement précédé de sa longueur. Blocage, longueur et clé dans l'article.	
UP	Permet de lire séquentiellement les enregistrements dans un ordre décroissant des clés à partir de l'enregistrement courant.	UP=1,var,long UP=1,index,var,long UP=1:100,E,var
UP.M **UP.L** **UP.ML**	Idem UP, mais avec blocage de l'article. Enregistrement précédé de sa longueur. Blocage, longueur et clé dans l'article.	
DOWN	Permet de lire séquentiellement les enregistrements dans l'ordre croissant des clés à partir de l'enregistrement courant.	DOWN=1,var,long DOWN=1,index,var,long DOWN=1:100,E,var
DOWN.M **DOWN.L** **DOWN.ML**	Idem DOWN, mais avec blocage de l'article. Enregistrement précédé de sa longueur. Blocage, longueur et clés dans l'article.	
MODIF	Modification d'un article.	MODIF=1,clé MODIF=1,clé,index,var,long
DELETE	Suppression d'un article	DELETE=1,clé DELETE=1,clé,index

INSTRUCTIONS RELATIVES AUX FICHIERS MULTICRITERES

Instruction	Fonction	Exemple
KEY	Description d'une rubrique. Permet de définir le nom de la rubrique, sa longueur, son type $ # %, le nombre de décimales.	KEY=1, nomrubr, long KEY=1,rub, long, type, nbdec
CKEY	Création des rubriques. Provoque l'écriture des rubriques dans le dictionnaire.	CKEY=1 CKEY=1:100,E
RKEY	Réinitialisation de la table d'interrogation.	RKEY=1
LINK	Description des liens.	LINK=1,"NOM","VILLE","ADR" LINK=1,I1,I2,I3,option
CLINK	Création des liens entre les diverses rubriques.	CLINK=1 CLINK=1:100,E
COUNT	Dénombrement multicritères.	COUNT=1,"VILLE='PARIS'",var
POSIT	Positionnement dans le dictionnaire selon le critère en tête des réponses.	POSIT=1,"VILLE='PARIS'",var POSIT=1,critère,var
POSIT.D	Idem POSIT, mais en queue des réponses.	

Informations statistiques

Instruction	Fonction	Exemple
STAT	Permet d'obtenir des informations sur l'historique des fichiers MC et sur les opérations effectuées.	STAT=1,var STAT=1,:100,E,var,long

La zone définie var permet de recevoir :

Pos	Type	Description
0	#	Nombre d'octets transmis par MC.
1	#	Numéro de version MC lors de la création du fichier MC.
2	%	Année de création du fichier MC.
4	#	Mois de création du fichier MC.
5	#	Jour de création du fichier MC.
6	#	Heure de création du fichier MC.
7	#	Minute de création du fichier MC.
8	#	Réservé.
9	%	Nombre d'articles dans le fichier de données.
11	#	Nombre de rubriques déclarées.
12	#	Nombre de liens déclarés.
13	%	Nombre d'interrogations et/ou dénombrements.
15	%	Nombre de mise à jour (insertions/modifications).

INSTRUCTIONS RELATIVES A LA BASE DE DONNEES RELATIONNELLE

Le tableau ci-dessous n'est pas un inventaire du jeu d'instructions disponibles, mais un simple aperçu.

Instruction	Fonction	Exemple
FILE	Déclaration d'un fichier.	FILE=1,"NOMFIC",MC(n)
LFILE	Liste des fichiers du dictionnaire.	LFILES=1,V
KEY	Définition d'une rubrique.	KEY=1,"NOM",1
FKEY	Redéfinition d'une rubrique.	FKEY=1,"RUB"
NKEY	Renommer une rubrique.	NKEY=1,"RUB","RUB1"
JOIN	Déclaration d'une jointure. Chaque élément de la liste se compose de définitions de rubrique.	JOIN=1 "NOMJOIN", liste NOMRUBi = NOMRUBj
CJOIN	Création d'une jointure.	CJOIN=1:100,E CJOIN=1("CLIFACT"):100,E
DJOIN	Destruction d'une jointure.	DJOIN=1("CLIFAC")
RJOIN	Renommer une jointure.	RJOIN=1("CLIFAC"),("FACLI")
LJOIN	Liste des jointures.	LJOIN=1,var
LINK	Description d'un lien fichier ou base.	LINK=1,"NOM,VILLE" LINK=1("CLIFAC"),"NOM",(A)
CLINK	Création d'un lien.	CLINK=1
LLINK	Liste des liens.	LLINK=1,var

OPERATEURS RELATIONNELS UTILISES PAR LE MULTICRITERE ET LA BASE DE DONNEES

Opérateur	Symbole	Exemple
Egalité	=	Nom = 'DUPOND'
Différence	< >	Ville <> 'PARIS'
Inférieur à	<	Age < '32'
Supérieur à	>	Age > '32'
Inférieur ou égal à	< =	Age <= '70'
Supérieur ou égal à	> =	Age >= '30'
Compris entre deux valeurs bornes incluses	(,)	Nom ('durand','giraud')

INSTRUCTIONS DE GESTION DES ENREGISTREMENTS

Instruction	Fonction	Exemple
LET	Instruction d'affectation optionnelle d'écriture/ lecture en mémoire du système ou en mémoire virtuelle. Voir FIELD=N.	LET A=10 idem A=10
READ	Lecture séquentielle	READ=1,VAR(1) READ=1,V,16 ; lit 16 octets READ=1:100,E,V,16
WRITE	Ecriture séquentielle.	WRITE=1,V WRITE=1,V,16 ; écrit 16 octets WRITE=1:100,E,V,16
BACKSPACE	Positionner sur l'enregistrement précédent.	BACKSPACE=1:100,E
MODIF	Modification d'un enregistrement.	MODIF=1:100,E,tamp,long

INSTRUCTIONS GRAPHIQUES

Pour être exploitables, ces instructions nécessitent un exécuteur BAL avec l'option GR (graphique).

Il est également nécessaire que les modules d'extension graphique soient déclarés dans le fichier SYSCONF-S.

Instruction	Fonction	Exemple
WINDOW	Définition fenêtre.	WINDOW=1,(X1,Y1),(X2,Y2)
VIEWPORT	Définition clôture.	VIEWPORT=1,(X1,Y1),(X2,Y2)
CLEAR	Effacement clôture.	CLEAR=n,COLOR=V
POINT	Tracé de symboles.	POINT=1:V,L
LINE	Tracé de ligne.	LINE=1:V,L
PEN	Sélection attributs de tracé.	PEN=n,COLOR.PT=V PEN=n,TYPE.PT=V PEN=n,SIZE.PT=V PEN=n,TYPE.LN=V PEN=n,COLOR.LN=V PEN=n,TYPE.LN=V PEN=n,SIZE.LN=V
CONF	Lecture paramètres associés à Nr log.	CONF=n:L
PREAD	Lecture attribut couleur d'un point.	PREAD=n,(X,Y):V

DIRECTIVES DU BAL

Directive	Fonction	Exemple
INCLUDE	Inclusion d'un fichier source lors de la traduction.	INCLUDE "MENU"
REM ou ;	Commentaire.	REM ceci est un commentaire 10 A=1.20 ;commentaire ;commentaire
• ou •	En début de ligne.	Saut de page à l'édition du listing lors de la traduction
<	Autorise la liste.	< commentaire
>	Interdit la liste.	> fin de liste
MSIZE	Taille maximale des segments.	MSIZE n

MISE AU POINT D'UN PROGRAMME BAL

Cette facilité est offerte au programmeur lorsqu'il demande l'exécution de son programme avec l'option **DB (DEBUG)**.

EX,DEMO,DB

L'option DEBUG permet :

- d'exécuter les instructions une à une (pas à pas) ;
- d'insérer des points d'arrêt ;
- d'examiner et modifier le contenu des variables ;
- de lancer l'exécution du programme à partir d'une adresse quelconque.

Afin que ces opérations puissent être possibles, il est nécessaire d'effectuer une traduction du programme avec un listing donnant les adresses des instructions.

Liste des commandes sous DEBUG

Commande	Description
S <rc>	L'exécution est faite pas à pas, la barre d'espacement provoque l'avancement du programme.
B seg,adr	Implante un point arrêt fugitif à l'adresse adr du segment seg.
G <rc>	Permet d'exécuter le programme jusqu'à la rencontre d'un point d'arrêt.
A seg,adr	Implante un point arrêt permanent à l'adresse adr du segment seg.
G adr	Lancement à partir d'une adresse appartenant au segment courant.
C <rc>	Annule le mode pas à pas et tous les points d'arrêts permanents.
ESC	Retour à DEBUG si le programme boucle.
E <rc>	Sortie définitive de DEBUG et retour à l'interpréteur de commande (->).
D var	Affiche la valeur courante de la variable spécifiée.
M var=valeur	Remplace le contenu de la variable spécifiée par valeur.

CHAPITRE 6 l'organisation du système

Dans le but de faciliter les développements et d'assurer une souplesse de configuration et de portabilité sur des machines différentes, le système d'exploitation PROLOGUE est constitué par des modules indépendants dont les fonctionnalités (spécifications et modes d'utilisation) sont clairement définies.

Sa conception est telle qu'il admet les environnements matériels les plus simples aux plus étoffés et peut être adapté facilement à toute configuration particulière.

Deux niveaux de logiciel définissent le système PROLOGUE :

- le niveau **SYSTEME** ;
- le niveau **ENVIRONNEMENT.**

Le **niveau système** est construit autour d'un certain nombre de modules logiciels ; il constitue la partie **portable** de PROLOGUE.

Le **niveau environnement** est plus spécialement attaché au matériel, les modules qui le composent réalisent l'implantation de **PROLOGUE** sur une machine donnée ; il est, de ce fait, **non portable.**

On appelera respectivement les modules constituant ces deux niveaux :

NOYAU SYSTEME	SYSTEME PROLOGUE
NOYAU ENVIRONNEMENT	minimal

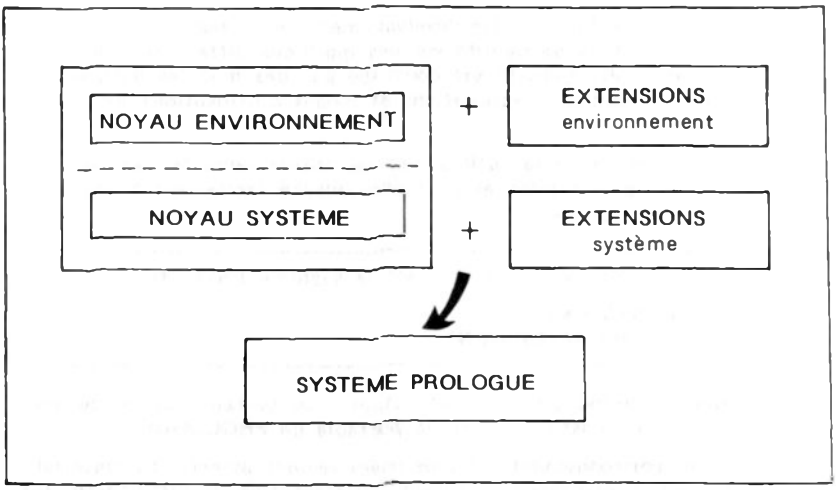
La concaténation des noyaux système et environnement définit le système PROLOGUE minimal pour une machine donnée.

LES EXTENSIONS DU SYSTEME

D'autres modules logiciels écrits en langage assembleur viennent compléter le système minimal ; on les appelle les modules d'**EXTENSIONS** :

- les **extensions SYSTEME** ;
- les **extensions ENVIRONNEMENT.**

Il s'agit par exemple des contrôleurs de périphériques non intégrés dans le noyau environnement (disque dur, imprimante) ou des méthodes d'accès aux fichiers non intégrées dans le noyau système (séquentiel indexé, base de données, etc.).



Résumé

PROLOGUE est résident en mémoire ; il est vu par l'utilisateur comme un tout, UNIQUE et COHERENT.

Le principe des extensions permet à l'utilisateur d'agir sur la configuration par :

- la **DECLARATION** de modules extension dans un fichier particulier dont le nom est immuable : **SYSCONF-S** et par l'inclusion de ces modules dans le disque système ;
- des **PARAMETRES** de configuration associés aux modules d'extension environnement ou système.

STRUCTURE DU NOYAU ENVIRONNEMENT

Prenons par exemple une machine type dont le sigle sera '**xy**' et disposant en version de base d'une console clavier-écran et de deux lecteurs de disquettes.

Les modules qui composeront le noyau environnement seront, au niveau élémentaire :

- **INxy-X** : initialisation propre de l'environnement.
- **COxy-X** : contrôleur clavier-écran (console).
- **FLxy-X** : contrôleur floppy (disquettes).

Ces modules seront liés pour produire un module unique **NENVxy-X**, autrement dit le noyau environnement de la machine '**xy**'.

> Trois modules suffisent pour composer un noyau environnement minimal.
>
> Le choix de placer un contrôleur de périphérique dans le noyau environnement ou comme module extension est laissé libre et dépend essentiellement des caractéristiques de l'implantation et de la souplesse recherchée.
>
> Il est toutefois souhaitable que le même module puisse être indifféremment lié au noyau ou inclus comme module d'extension.

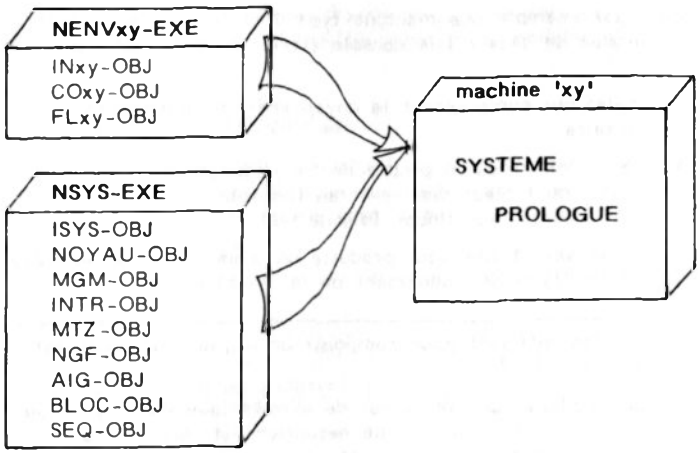
STRUCTURE DU NOYAU SYSTEME

Les modules élémentaires composant le noyau système sont les suivants :

- **ISYS-OBJ** : initialisation du système.
- **NOYAU-OBJ** : noyau du système.
- **MGM-OBJ** : module gestion de mémoire.
- **AIG-OBJ** : aiguillage de l'interface Système de Fichiers.
- **INTR-OBJ** : interpréteur de commandes.
- **MTZ-OBJ** : moniteur multitâche.
- **NGF-OBJ** : noyau de gestion de fichiers.
- **BLOC-OBJ** : méthodes d'accès fichier par BLOC.
- **SEQ-OBJ** : méthode d'accès fichier SEQUENTIEL.

Ces modules sont liés pour produire le module unique **NSYS-EXE**, **noyau portable** du système PROLOGUE.

Le système d'exploitation PROLOGUE de la machine **'xy'** est obtenu par la concaténation des fichiers binaires **NENVxy-EXE** et **NSYS-EXE**.



PRESENTATION DU DISQUE SYSTEME

Le disque système PROLOGUE contient au minimum les fichiers ci-dessous :

- **SYSTEME**	Système d'exploitation PROLOGUE de base configuré pour une machine cible donnée.
- **SYSCONF-X**	Programme initial pour le traitement du fichier de configuration SYSCONF-S.
- **SYSCONF-S**	Fichier source optionnel contenant la déclaration des modules d'extension pour une configuration donnée.
- **SYSERxx-S**	Fichier source optionnel contenant les libellés des codes erreurs système. **xx** varie selon la langue. **SYSERF-S** = français **SYSERGB-S** = anglais
- **SYSTnn-C**	Fichier source optionnel des commandes initiales à exécuter sur le poste **nn**.

En outre, le disque système doit comporter tous les fichiers binaires (**-X**) des modules extension déclarés.

La structure d'un fichier **SYST**nn-**C** est identique à celle d'un fichier source créé pour un enchaînement de commandes (utilitaire **ASG-X**).

SYST00-C correspond au fichier des commandes initiales à exécuter sur le poste 0.

Dans le cas où l'utilisateur désire agir sur les paramètres de configuration initiaux du système, les fichiers **NENVxy-EXE** et **NSYS-EXE** peuvent être fournis, ainsi qu'un programme permettant le paramétrage et la concaténation des deux modules.

PARAMETRES DE CONFIGURATION

Il existe deux types de paramètres de configuration liés au noyau système **NSYS** et au noyau environnement **NENV**.

Les paramètres généraux

Ils sont définis à des adresses fixes dans les fichiers.

NENVxy-X ou **NSYS-X**

Les paramètres extensions

Ils sont définis explicitement lors de la déclaration des modules extension dans le fichier.

SYSCONF-S

Configuration de l'environnement

Aux modules composant l'environnement, peuvent être associés des paramètres permettant une relative adaptation à la configuration de la machine.

La structure de ces paramètres ne sera pas évoquée ici puisqu'elle est liée étroitement à un environnement spécifique.

Configuration du noyau système

Ces paramètres sont définis à des adresses fixes dans le premier secteur du fichier **NSYS-X**.

Ces paramètres se divisent en paramètres généraux pour ce qui est du système et en paramètres spécifiques à chaque poste de travail dans le cas d'une machine multiposte.

On accède à ces paramètres soit par l'utilitaire **PATCH-X**, soit par l'intermédiaire d'un utilitaire de configuration écrit en **BAL** par exemple.

Disposition des paramètres de configuration

Les paramètres de configuration du noyau système, décrits précédemment, sont définis à des adresses fixes appartenant au **secteur 2 du fichier NSYS-X**.

Format du secteur de configuration

Offset	Long	Valeur	Signification
00h	3	JMP	Traitement des appels faits par le module INxy-X.
03h-**05**h	3	???	Relai avant exécution d'un programme chargé par l'interpréteur de commandes.
06h-**07**h	2	0000	Réservés à zéro.
08h	1	31h	Indicateur de structure.
09h-**0F**h	7	0	Réservés à zéro.
10h	1	?	Code langue.
11h-**12**h	2	????	Taille de la partition système en K octets.
13h-**15**h	3	ascii	Mnémonique disque implicite système (IS).
16h	1	?	Nombre de postes (01 à 08 max).
17h-**1A**h	9	0	Réservés à zéro.
1Bh-**1C**h	2	?	Caractéristiques première imprimante.
1Dh-**1F**h	16	0	Réservés à zéro.
20h-**3F**h	16	?	Réservés.
40h-**7F**h	64	?	Réservés à zéro.
SUIVENT LES PARAMETRES DU (des) POSTES			

Offset	Long	Valeur	Signification
80h-8Fh	16		**Paramètre du poste 0.**
80h-82h	3	ascii	Mnémonique implicite système (IS). Exemple : MD0.
83h-85h	3	ascii	Mnémonique implicite fichier (IF). Exemple : MD1.
86h-88h	3	ascii	Mnémonique implicite impression (IL). Exemple : IM0.
89h-8Ah	2	??	Taille partition poste en K octets.
8Bh	1	?	Indicateur commande initiale.
8Ch-8Dh	2	??	Protections d'exploitation.
8Eh-8Fh	2	0000	Réservés à zéro.
IDEM pour les postes suivants			
90h-9Fh			**Paramètres poste 1**
A0h-AFh			**2**
B0h-BFh			**3**
C0h-CFh			**4**
D0h-DFh			**5**
E0h-EF0			**6**
F0h-FFh			**7**

Résumé

Les paramètres de configuration de PROLOGUE sont définis au niveau des modules composant le système.

Ils sont accessibles à l'utilisateur, sans nécessiter un système de développement (assembleur + éditeur de liens).

DEFINITION DES PARAMETRES GENERAUX

NOM	Long	Valeur	Signification
KLAN	1		**CODE LANGUE SYSTEME** Définit la langue de base des messages du système et des utilitaires. Le séparateur décimal utilisé en BAL est la virgule, sauf pour la Grande-Bretagne où on utilise le point.
		0	**Français**
		1	Anglais
		2	Allemand
		3	Espagnol
		4	Italien
		5	Norvégien
		6	Suédois
		7	Danois
		8	Hollandais
		9	Finlandais
		10	Portugais
		11	Grec
PMSYS	2	2	**TAILLE PARTITION SYSTEME** Définit en nombre de K octets la taille de la partition mémoire système. Valeur courante : 2 K octets par poste de travail. PMSYS peut être nul.
SYSIMP	3		**DISQUE IMPLICITE SYSTEME** Spécifie la ressource implicite sur laquelle seront recherchés les modules extension.
		MD0	Par exemple.
SYSNBP	1		**NOMBRE DE POSTES CONNECTES** Donne en binaire le nombre de postes qui peuvent être connectés au système.
		1	Par exemple 1, si monoposte.
CARIM1	2		**CARACTERISTIQUES DE LA PREMIERE IMPRIMANTE**
		66	1e octet = nombre de lignes par page.
		132	2e octet = nombre de colonne par ligne.
DSIP	3	ascii	**DISQUE IMPLICITE PROGRAMMES** Donne le mnémonique de la ressource implicite programmes du poste.
		FL0	Par exemple.

→

NOM	Long	Valeur	Signification
DSIF	3	ascii FL1	**DISQUE IMPLICITE FICHIERS** Donne le mnémonique de la ressource implicite fichiers du poste. Par exemple.
DSIL	3	ascii IM0	**IMPRIMANTE PAR DEFAUT** Donne le mnémonique de la ressource implicite imprimante. Par exemple.
PMPOS	2	bin 80h	**PARTITION MEMOIRE** Donne en nombre de K octets la taille de la partition mémoire associée au poste pour l'exécution des programmes utilisateur. 128 K octets.
PCOMIN	1	bin 00 01	**COMMANDE(s) INITIALE(s)** Ce paramètre indique la présence du fichier **SYST**nn-C (**nn** = n° du poste) sur le disque système du poste (**DISP**). Pas de commande initiale. Présence de commande(s) initiale(s).
NPREX	2	bin	**NIVEAU DE PROTECTIONS D'EXPLOITATION** Chaque bit à 1 spécifie une autorisation d'exploitation particulière. |7|6|5|4|3|2|1|0| b0 Sauvegardes autorisées par : **CP**,DV ou **CPS**, ou **CP**,CV ou **CP**,RV. b1 Ecriture en format libre autorisée par : PATCH, TSM, CPS. b2 Instructions système disponible en BAL Peek, Poke, IO, etc. b3 Usage configurateur autorisé. b4 Fichiers NGF au format 8-3 autorisés. b5 Usage du traducteur et debug autorisé. Les bits 6 ... 15 sont réservés.

CHAPITRE 7 modules d'extension

Une des caractéristiques de PROLOGUE est sa configurabilité grâce à des modules d'extension liés dynamiquement avec le système.

Un module d'extension (environnement ou système) est constitué par un fichier programme (**-X**) présent sur le disque système et déclaré dans le fichier de configuration **SYSCONF-S**, également sur le disque système.

Un module d'extension est reconnu par PROLOGUE de deux façons :

- Par son **nom** qui, lors de l'initialisation, est chaîné dans la liste des modules présents dans la configuration du système.
- Par son **lien** au système qui dépend de la nature du module. Le lien peut être :
  - soit un **autre module** préalablement identifié par le système et auquel se raccroche le module déclaré ;
  - soit un **module virtuel** présent dans le système.

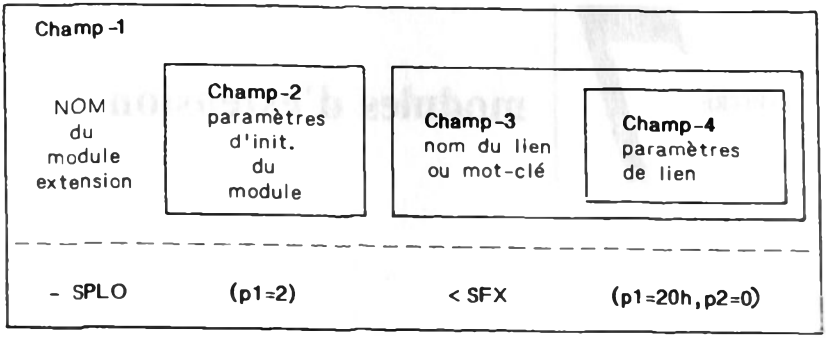
Un module virtuel est défini par un mot-clé auquel correspond un type de lien.

CO LO ES NGF SFX INTR

sont des mots-clés définissant des modules virtuels.

DECLARATION D'UN MODULE EXTENSION

La déclaration d'un module extension est faite sur une ou plusieurs lignes source et doit être conforme à la structure ci-après.



La signification de ces divers champs est la suivante :

Champ-1 Obligatoire -SPLO

Il est précédé du caractère '-' et a pour valeur le nom du **module extension** (-X) situé sur le disque implicite système.

Champ-2 Optionnel (p1=2)

Il donne la liste des paramètres d'initialisation propres au module.

Le **champ-2** est séparé du **champ-1** par le caractère '(' et se termine par le caractère ')'.

Champ-3 Optionnel < SFX

Il spécifie le nom du lien ou le mot-clé auquel se raccroche le module désigné par le **champ-1.**

Le **champ-3** est toujours séparé du champ-2 par le caractère '<'.

Champ-4 Optionnel (p1=20h,p2=0)

De structure identique au champ-2, il donne la liste des paramètres dynamiques relatifs au lien établi entre les deux modules.

Le **champ-4** est séparé du champ-3 par le caractère '(' et se termine par le caractère ')'.

Les **champs-3** et **4** peuvent être répétés si le module chargé en mémoire présente plusieurs liens de nature différente avec le système.

DESCRIPTION DES PARAMETRES

Un paramètre est toujours représenté sous la forme :

Pi = v

i est le numéro du paramètre précédé de la lettre P ou **p**.
v est la valeur associée à **Pi** précédée du caractère =.

p1=2 P2=20h p3='NGF'

Quand plusieurs paramètres doivent être représentés, il faut les séparer par une virgule.

(p1=1,p2='MD',p3='NGF')

Paramètre numérique

P1=**2** P1=**5E**h P1=**9600**

Il peut être donné en DECIMAL ou en HEXADECIMAL ; dans ce cas, le dernier chiffre doit être suivi de la lettre 'H' ou '**h**'.

Les caractères autorisés sont '**0**' à '**9**' en base décimale et 'A' à 'F' en plus en base hexadécimale.

Paramètre littéral

P1=**'NGF'** P1=**'chaîne de caractères'** P1=**''''**

C'est une chaîne de caractères encadrée par le caractère ' (2Dh) ; il ne peut se prolonger au-delà d'une ligne.

Si le caractère ' doit être représenté dans le littéral, il faut le doubler : **''**.

Paramètres de lien

CO	Permet de lier le module de traitement des appels CO pour les postes en extension (multiposte).
LO	Permet de lier le module de traitement des appels LO (contrôleur imprimante LOxx ou EMLO).
	Pas de paramètres de lien.
ES	Permet de lier un contrôleur de ressource périphérique disque, bande ou spécial. ES permet aussi de définir un mnémonique d'accès à un module.
	Trois paramètres de lien obligatoires : **p1** : mnémonique de la ressource. **p2** : nombre d'unités supportées. **p3** : nom de la méthode d'accès référencée par la ressource.
NGF	Permet de lier une méthode d'accès à des supports sectorisés (disques, disquettes, mémoire).
	Deux paramètres obligatoires : **p1** : numéro du relais système utilisé pour atteindre la méthode d'accès. **p2** : niveau de la méthode d'accès.
SFX	Permet de lier une méthode d'accès hors NGF.
	Deux paramètres obligatoires : **p1** : numéro du relais système utilisé pour atteindre la méthode d'accès. **p2** : niveau de la méthode d'accès.
INTR	Permet de lier les utilitaires résidents en mémoire avec le système.

EXEMPLES DE DECLARATIONS

Modules environnement

Les exemples ci-dessous sont formels, ils ne correspondent pas nécessairement à un cas réel.

Contrôleur disque optionnel

```
- DSxx < ES (p1='DS',p2=1, p3='NGF' )
```

Le contrôleur disque est déclaré au niveau E/S avec le mnémonique 'DS' ; il comprend une unité (**p2=1**), ce qui signifie que l'on ne pourra adresser que l'unité **DS0**.

Le paramètre **p3** indique la nature **NGF** du contrôleur.

Contrôleur d'imprimante parallèle

```
- LOxx < LO ; pas de paramètre, ni d'initialisation, ni de lien
```

Le module **LOxx** est rattaché directement au mot-clé **LO**.

Contrôleur d'imprimante série

```
- LOSxx (p1=4,p2=9600) < LO
```

Le module **LOSxx** a pour paramètres d'initialisation :

p1 : numéro de la voie V24 sur laquelle est connectée l'imprimante.
p2 : vitesse exprimée en bauds.

Système complet de gestion de deux imprimantes

```
- EMLO < LO                ;EMLO qui traite les appels LO
- SPLO (p1=2) <  SFX (p1=20h, p2=0)
              <  ES (p1='IM',p2=2,p3='SPLO')
- SGLO (p1=2) <  SPLO      ;lien entre SGLO et SPLO
- LOxx <  SGLO (p1=0)      ;déclaration contrôleur IM0
                           ;avec paramètre de lien donnant le
                           ;numéro de l'imprimante
- LO2xx (p1=3) < SGLO (p1=1) ;déclaration contrôleur IM1
                           ;avec paramètre d'initialisation donnant
                           ; le numéro de voie de sortie
```

SPLO a un paramètre d'initialisation qui est le nombre d'imprimantes supportées. Il est lié au système de deux façons :

- SFX pour le déclarer comme méthode d'accès hors NGF, s'adressant par le relai **20h** et de niveau **0** ;
- ES pour que l'imprimante puisse être adressée par son mnémonique, bien qu'il n'y ait pas d'accès ES possible.

Ici, SPLO recouvre SGLO qui n'est pas vu par le système.

Contrôleur TTY gérant des lignes asynchrones

```
- TTYxx (p1=2,p2=5,p3=8) < PTTY
```

Le contrôleur **TTYxx** gère trois voies asynchrones (voies **2**, **5** et **8**). Il est rattaché à la méthode d'accès **PTTY**.

Contrôleur mémoire

Le module **MEMDISK-X** permet la gestion d'un volume en mémoire de la même manière qu'un disque.

Cette extension est intéressante dans la mesure où le matériel dispose d'une capacité mémoire élevée, 256 K octets minimum à 1 M octet.

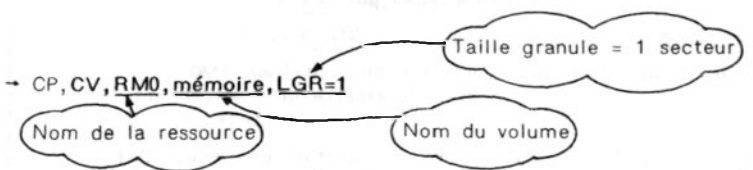
Un ou plusieurs volumes mémoire peuvent être déclarés. L'accès au volume se fera par un mnémonique défini dans la déclaration du contrôleur.

```
- MEMDISK (p1=64) < ES (p1='RM',p2=1,p3='NGF')
```

Le contrôleur mémoire est déclaré au niveau E/S avec le mnémonique 'RM' ; il comprend une unité (**p2=1**), ce qui signifie que l'on ne pourra adresser que la ressource mémoire **RM0**.

Le premier paramètre **p1=64** définit le nombre de 'pistes' de **16** secteurs de **256** octets. Ainsi, **p1=64** signifie que la capacité mémoire utilisée sera de **256*16*64**, soit **315984** caractères.

La création effective du volume en mémoire se fera à l'aide de la commande **CP**.



On pourra ensuite procéder à toutes les manipulations autorisées sur un volume.

Modules système

Déclaration d'une méthode d'accès NGF

```
- SI  < NGF (p1=20h,p2=2)      ;séquentiel indexé
- MC  < NGF (p1=20h,p2=3)      ;multicritère
- BD  < NGF (p1=21h,p2=2)      ;base de données
      < NGF (p1=21h,p2=3)
```

Le paramètre **p1** indique le numéro de relais par lequel on accède à la méthode d'accès.

Le paramètre **p2** indique le niveau de la méthode (**0** à **3**).

Le module **BD** travaillant sur deux niveaux doit être lié deux fois.

Déclaration d'une procédure de télétransmission asynchrone

```
- PTTY < SFX (p1=20h,p2=0)
       < ES  (p1='LA',p2=3,p3='PTTY')
```

La procédure **PTTY** est redéclarée au niveau **ES** pour permettre un accès sur mnémonique, bien qu'aucune fonction de niveau E/S ne soit définie.

p1='LA' représente le mnémonique d'accès à la procédure.
p2=3 signifie que **PTTY** gère **3** lignes asynchrones (**LA0, LA1, LA2**).

STRUCTURE DU FICHIER SYSCONF-S

Le fichier **SYSCONF-S** étant un fichier source, il peut être créé et modifié par un éditeur de texte du type :

EDV-X	PROLOGUE
Word Master	décor CP/M
Word Star	décor CP/M
EDLIN	décor MS/DOS

Ce fichier peut aussi être construit par un utilitaire spécial et conversationnel écrit en BAL ou BASIC, par exemple.

Les caractères de tabulation **TAB** (09h) et interligne **LF** (0Ah) sont filtrés systématiquement.

Le caractère **Blanc** ou espace (20h) est filtré partout sauf dans les '**littéraux**' et fait office de séparateur.

Hormis dans les littéraux, tous les caractères rencontrés à partir du caractère point virgule ';' (3Bh) sont ignorés jusqu'à la fin de la ligne définie par **CR** (0Dh). Cela permet de définir des commentaires à l'intérieur du fichier SYSCONF.

TRAITEMENT DU FICHIER SYSCONF

Le fichier **SYSCONF-S** est traité à l'initialisation du système par le programme **SYSCONF-X**.

Ce programme est activé implicitement sur le poste **0**, après le chargement du système, une fois l'initialisation des tâches de l'interpréteur de commandes effectuée.

La première action du programme **SYSCONF-X** est de charger en mémoire le fichier des libellés des erreurs système (1 à 82). Ces libellés sont définis dans le fichier source **SYSERxx-S** présent sur le disque système.

Puis, pour chaque module extension, les opérations ci-dessous sont exécutées :

- chargement du module extension (xxxx-**X**) au sommet de la partition commune, c'est-à-dire en fin de mémoire ;
- insertion du nom du module dans la liste des modules système et mise à jour du nom du module (égal au nom du fichier) dans l'entête du module ;
- initialisation du module. Chaque paramètre de la liste (si celle-ci existe) est passé au module qui l'utilise pour son initialisation. Un appel supplémentaire est fait pour clore la phase d'initialisation ;
- mise à jour des liens (module ou mot-clé) référencé par le nom du lien.

Le traitement est fait successivement pour chaque module déclaré dans SYSCONF-S jusqu'à la rencontre de la fin du fichier source (contrôle Z = 1Ah).

La partition commune doit être de taille suffisamment grande pour permettre le chargement de tous les modules extension.

En cas d'erreur de configuration détectée lors du traitement de SYSCONF, celle-ci est affichée sur l'écran du poste 0 et la configuration est abandonnée.

Si un module déclaré dans SYSCONF-S est absent du disque système, un message d'attention est affiché, mais la configuration est poursuivie.

CHAPITRE 8 le graphique

Le logiciel graphique offre à l'utilisateur des primitives graphiques portables proches de la norme **G.K.S** (Graphical Kernel System) qui répond à un besoin de standardisation en matière d'utilisation du graphique sur micro-ordinateurs.

Le système graphique est intégré à PROLOGUE sous forme de modules d'extension paramétrables.

> Les fonctionnalités offertes à l'utilisateur sont indépendantes des caractéristiques des différents matériels.

Elles sont accessibles en langage BAL ou assembleur, mais aussi en d'autres langages de programmation.

DESCRIPTION DES DESSINS

Espace de travail

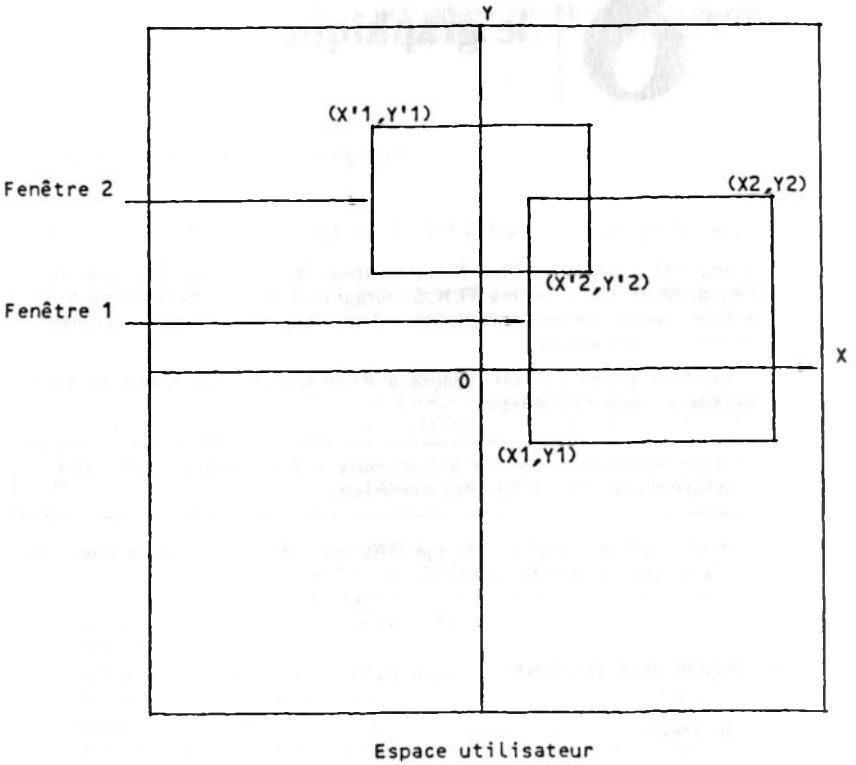
Les applications graphiques traitées sont des applications en deux dimensions.

L'utilisateur dispose d'un **espace de travail virtuel.** Cet espace est défini dans un repère orthonormé dont les coordonnées sont des entiers signés variant de **-32768** à **+32767** en X et en Y.

Le système de coordonnées dans lequel les dessins sont définis est appelé système de coordonnées de l'utilisateur.

Dans l'espace de travail, l'utilisateur dispose d'une **fenêtre** qui en délimite une partie rectangulaire dont les côtés sont parallèles aux axes du système de coordonnées utilisateur.

Seule la partie du dessin qui se trouve à l'intérieur de la fenêtre est affichée, le système effectuant un découpage de tous les tracés par rapport à cette fenêtre.



Définition explicite de fenêtres
en précisant à chaque fois deux points diagonaux

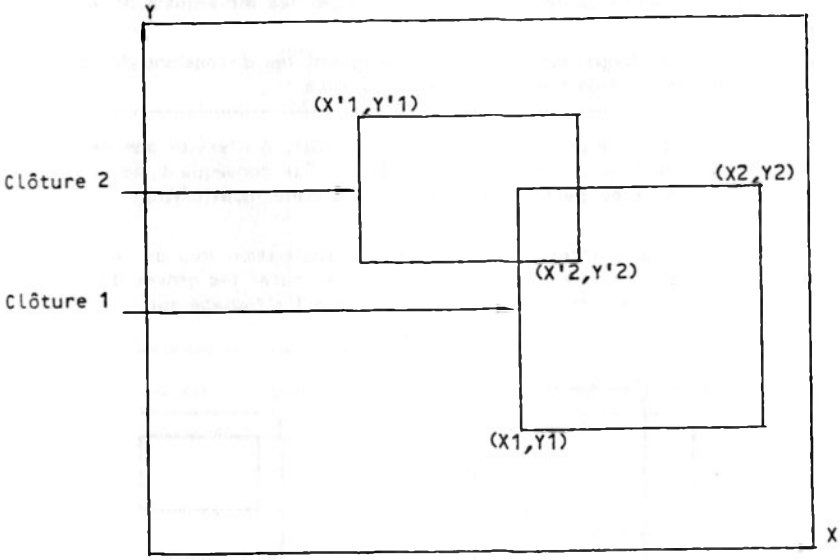
Espace de visualisation

Dans l'espace de **visualisation** (l'écran en général), on dispose d'une **clôture** qui en délimite une partie rectangulaire et dont les côtés sont parallèles aux bords de cet espace.

La taille de la clôture est inférieure ou égale à tout l'espace et ses dimensions sont précisées dans le système de coordonnées de l'écran.

Les coordonnées de l'espace de visualisation ou coordonnées réelles sont des entiers positifs de valeur inférieure ou égale aux dimensions de celui-ci.

Par convention, l'origine (X=0, Y=0) est située en bas à gauche de l'écran.



Définition explicite de clôtures
en précisant à chaque fois deux points diagonaux

L'affichage consiste à projeter le contenu de la fenêtre dans la clôture.
Il existe une indépendance totale entre l'espace virtuel de travail et l'espace de visualisation.

Fenêtre et clôture

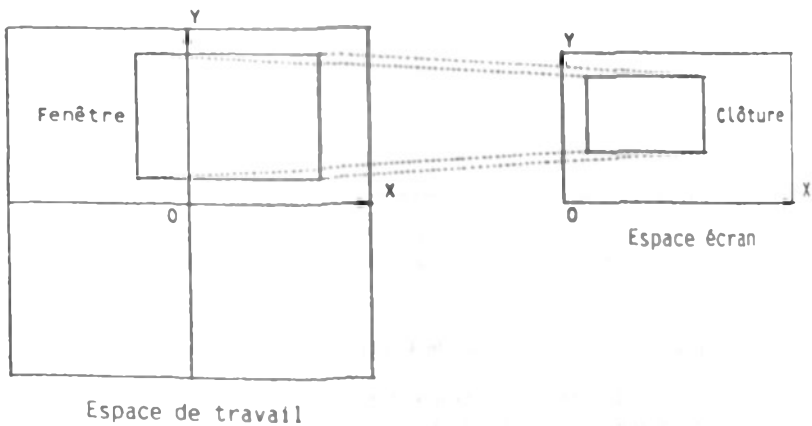
En changeant indépendamment les dimensions de la fenêtre et (ou) de la clôture, on peut obtenir des représentations différentes dans l'espace de visualisation.

Ainsi un même objet représenté dans une fenêtre peut subir des effets de **réduction** ou d'**agrandissement** dans l'espace de visualisation.

> L'effet d'**agrandissement** ou de **réduction** est obtenu en changeant les dimensions de la clôture sans changer les dimensions de la fenêtre.
>
> L'effet de **loupe** est obtenu en changeant les dimensions de la fenêtre sans modifier celles de la clôture.

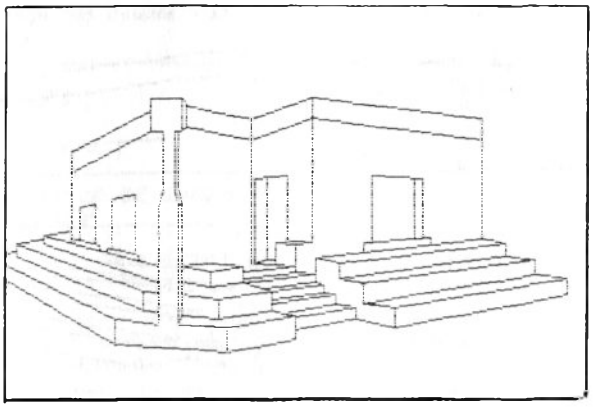
Du fait que l'espace de travail soit virtuel, il n'existe pas de mémorisation des dessins affichés par le système. Par conséquent, le changement de fenêtre ou de clôture n'entraîne aucune modification de l'image affichée.

Il n'y a pas d'effet immédiat sur la visualisation lors de la définition d'une fenêtre ou d'une clôture ; il faut réexécuter les ordres de tracé qui composent les différents dessins pour que l'affichage soit modifié.

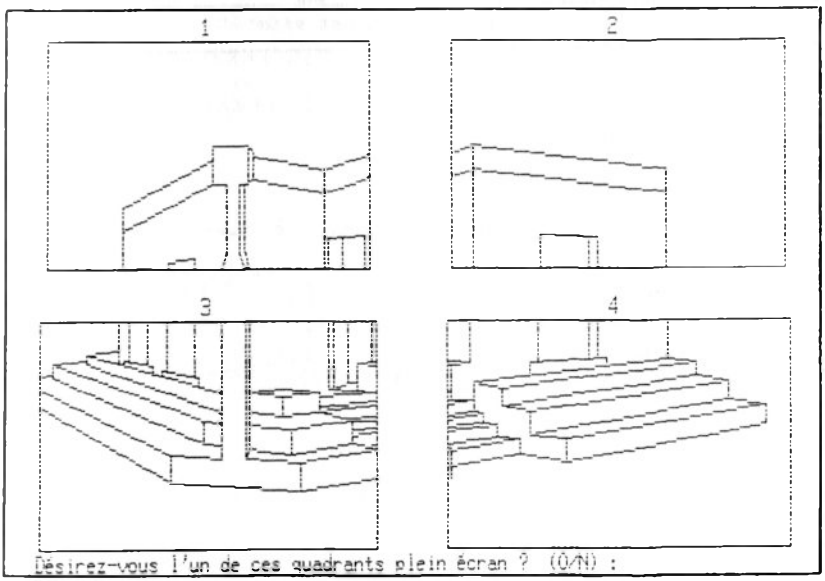


Transformation fenêtre-clôture

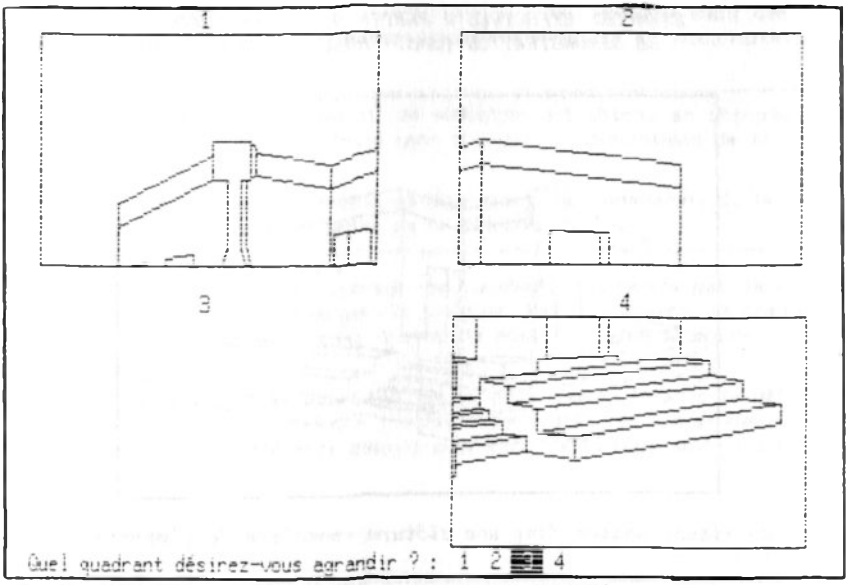
EXEMPLES EFFECTUES A PARTIR D'UN PROGRAMME DE DEMONSTRATION (doc. PROLOGUE-BULL)



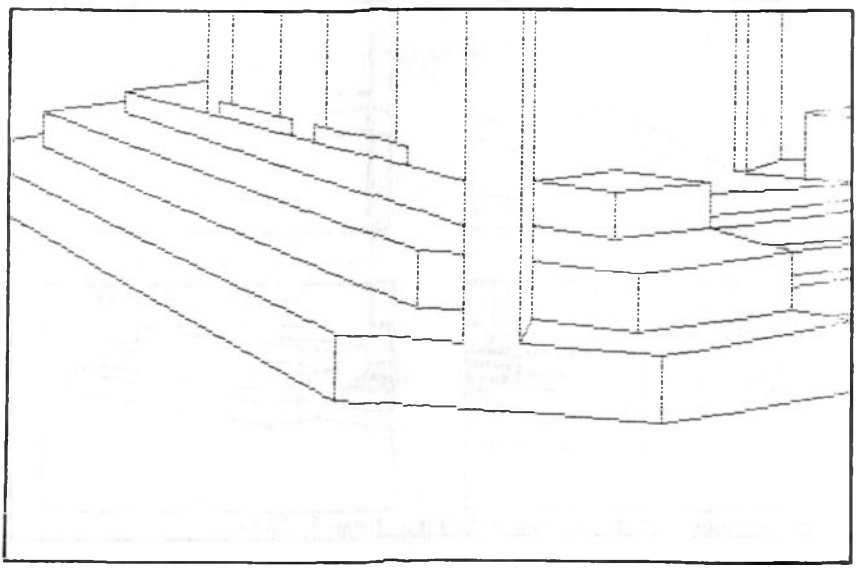
a- Visualisation dans une clôture (totalité de l'écran)



b- Visualisation dans quatre clôtures



c- Le cadran choisi est effacé



d- Visualisation du cadran numéro 3 plein écran

ORDRES DE TRACES

L'affichage des dessins s'effectue au fur et à mesure des ordres de tracés.

- **SYMBOLES** ;
- **LIGNES** ;
- **CERCLE, ARC** ;
- **TEXTE** ;
- **REMPLISSAGE DE SURFACE** ;
- **HACHURAGE.**

Un ordre de tracé s'adresse toujours à un périphérique désigné par un numéro logique ; ce peut être un écran, une table traçante,...

Tous les points précisés dans les ordres de tracés sont définis en coordonnées utilisateur : **-32767 à +32768.**

Lignes et symboles sont affectés d'attributs de tracé.

Trois attributs de tracé peuvent être définis :

- attribut de **COULEUR** ;
- attribut de **TYPE** ;
- attribut de **TAILLE.**

Tracé de lignes

Il permet de tracer des lignes entre différents points donnés.

L'ordre de tracé est envoyé avec un tableau contenant les coordonnées utilisateur des différents points à joindre par des lignes.

Un découpage de ces lignes est effectué par rapport à la fenêtre associée de sorte à ne tracer, dans la clôture correspondante, que les segments de droite intérieurs à la fenêtre.

Trois attributs de tracé sont possibles :

- la **couleur** de la ligne ;
- le **type de ligne** (continu, pointillé, tireté) ;
- l'**épaisseur** de la ligne.

16384 points différents peuvent être envoyés, ce qui correspond à **16383** lignes.

Tracé de symboles

Il permet de tracer un même symbole en différents points donnés.

On définit, dans un tableau, les coordonnées utilisateurs des différents points où le symbole doit être tracé. Seuls les points intérieurs à la fenêtre associée seront tracés dans la clôture correspondante.

Trois attributs sont affectés au tracé de symboles :

- la **couleur** du symbole ;
- le **type** de symbole (croix, point, ...) ;
- la **taille** du symbole.

LES ATTRIBUTS DE TRACE

Tous les ordres de tracé sont accompagnés des attributs propres à chaque tracé.

Une valeur d'attribut se caractérise par un nombre de **0** à **255**, **256** valeurs différentes sont possibles pour chaque attribut.

Dans la pratique, les attributs sont spécifiques au type de périphérique géré par l'environnement graphique, selon le standard ci-dessous.

Attributs de couleur

Valeur	Couleur correspondante
0	Noir
1	Rouge
2	Vert
3	Marron
4	Bleu
5	Magenta
6	Cyan
7	Gris clair
8	Gris foncé
9	Rouge clair
10	Vert clair
11	Jaune
12	Bleu clair
13	Magenta clair
14	Cyan clair
15	Blanc

Attributs de type

Type de symbole

Valeur	Type de symbole
1	. (point)
2	+ (plus)
3	X (croix)
4	O (rond)
5	□ (carré)
6	◊ (losange)

Type de ligne

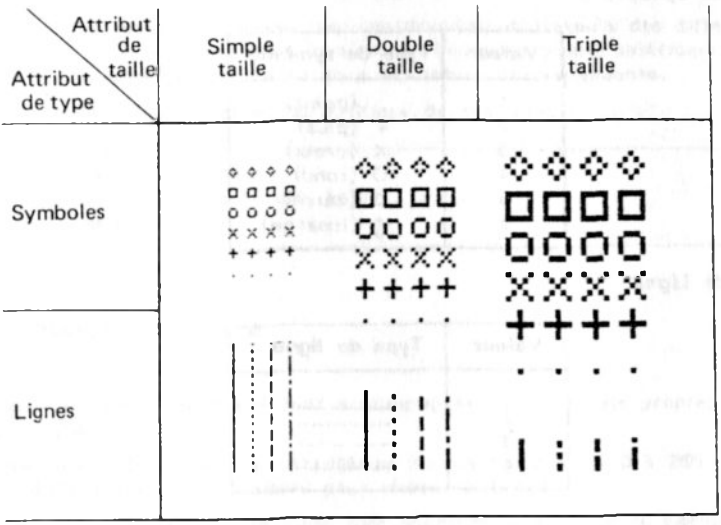
Valeur	Type de ligne
1	————
2	.....
3	- - - - -
4	-.-.-.-.-

Attributs de taille

Valeur	Type de taille
1	simple taille
2	double taille
3	triple taille

Pour le symbole POINT, la simple taille correspond à la taille du plus petit élément adressable (pixel).

Récapitulatif



Les attributs de type et de taille dans le graphisme de PROLOGUE

CHAPITRE 9 structure des volumes

Le **NGF** (**N**oyau de **G**estion de **F**ichiers) est un module système regroupant les fonctions élémentaires d'une gestion de fichiers sur supports sectorisés à accès aléatoire (disquette, disque dur, fixe ou amovible ...).

Il offre à l'utilisateur, à partir d'une gestion **dynamique** de l'espace disponible sur le support et d'un accès direct aux enregistrements, la possibilité de :

- définir le(s) type(s) d'organisation de ses fichiers ;
- construire la (les) méthodes(s) d'accès à ses enregistrements.

L'espace disque adressable par le **NGF** comprend :

- une **table d'allocation** des granules ;
- un **répertoire** de longueur variable.

L'espace restant disponible est utilisé pour l'allocation et désallocation dynamique **des granules** attribués aux fichiers définis dans le répertoire.

Un volume PROLOGUE est toujours contenu sur un seul support physique.

GESTION DE L'ESPACE DISQUE

Le **NGF** affranchit l'utilisateur des problèmes d'estimation préalable de l'espace disque nécessaire à ses fichiers.

Il alloue, au fur et à mesure de ses besoins, un quantum d'espace disque appelé **GRANULE**.

> Un **granule** est un multiple de secteurs de 256 octets défini à l'avance par l'utilisateur, en fonction de la capacité du support utilisé et de la nature des fichiers à gérer (taille, taux d'expansion).
>
> La dimension du granule est un paramètre (**LGR**) fixé par l'utilisateur au moment de la création du VOLUME. Il permet de rechercher une certaine optimisation dans la gestion de l'espace disque.
>
> Le paramètre LGR, une fois défini, s'adresse à l'ensemble des fichiers du volume et ne peut être modifié.

Table d'allocation d'un volume

Pour mémoriser l'espace disque alloué à l'ensemble des fichiers du volume, nous avons vu que le NGF dispose des deux premiers secteurs disque du volume.

Les deux premiers octets de cette table représentent la capacité physique maximale du volume exprimée en granules.

Un bit à **0** (zéro) de la table des granules représente un granule disponible.

Le nombre maximal de granules adressables par cette table est égal à **4080 GRANULES** ((512 - 2) * 8 = 4080 bits), ce qui autorise un espace maximal théorique de **255** millions d'octets.

La table d'allocation du volume, gérée conjointement avec le descripteur disque d'un fichier, permet au NGF de suivre l'évolution de ce fichier sur le volume.

ORGANISATION DU REPERTOIRE

> Le répertoire regroupe les noms des fichiers et les descripteurs associés.

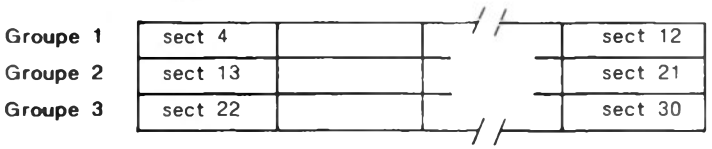
Il commence **toujours** au secteur numérique **4**.
Il comprend **des groupes** de **neuf** secteurs décomposés comme suit :

- **Un secteur** contenant les NOMS des fichiers.
  Chaque nom est contenu sur **huit octets.**
  Le secteur regroupe donc 256/8 soit **32** noms de fichiers.
- **Huit secteurs** contenant les **DESCRIPTEURS** associés.
  Chaque descripteur occupe **64** octets.
  Le secteur contient donc 256/64 soit **quatre** descripteurs.

Les huit premiers octets du secteur **4** (première place disponible pour un NOM de fichier) sont réservés pour mémoriser des informations relatives au VOLUME.

Il en est de même pour le descripteur associé, c'est-à-dire les 64 premiers octets du secteur 5.

STRUCTURE DU REPERTOIRE



32 noms de fichiers

32 descripteurs associés aux noms des fichiers

à l'exception du groupe 1 qui contient 31 descripteurs.

Informations concernant le volume

Ces informations sont situées dans les huit premiers octets du **secteur 4** du répertoire et le premier descripteur du **secteur 5**.

*Taille du granule (**LGR**)*

Cette taille est exprimée en secteurs de **256** octets. Les valeurs possibles sont :

1 - 2 - 4 - 8 - 16 - 32 - 64 - 128 - 0 (c'est-à-dire 256).

*Taille du répertoire (**LCAT**)*

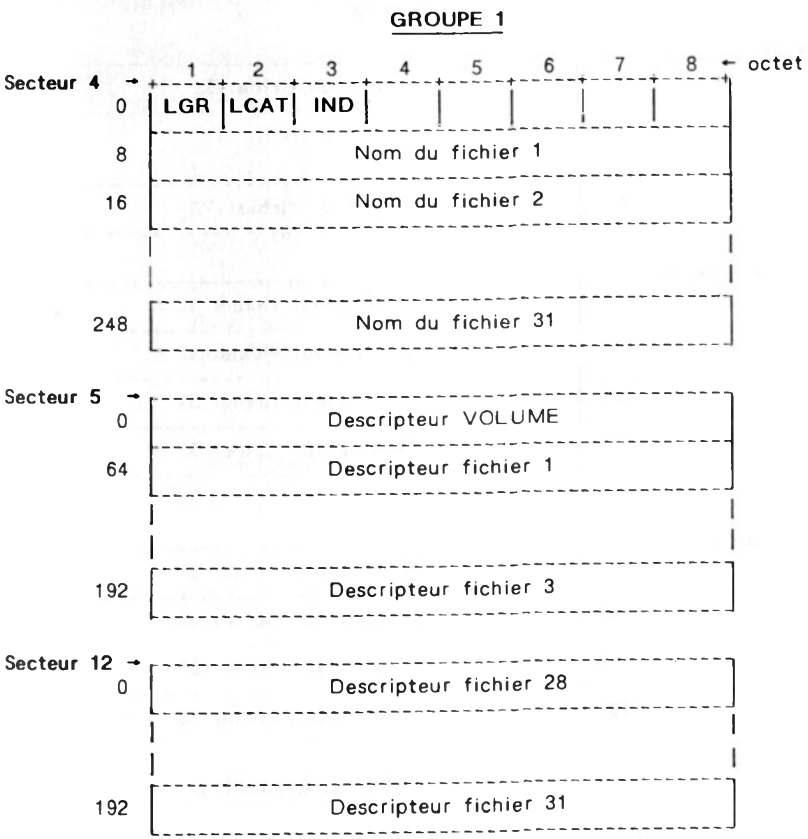
Cette taille, exprimée en secteurs, représente le numéro du premier secteur physique n'appartenant pas au répertoire. C'est le premier secteur disponible pour les données des fichiers.

> Le nombre maximal de fichiers sur un volume **NGF** étant limité à **895**, la taille maximale du répertoire est de **256** secteurs.

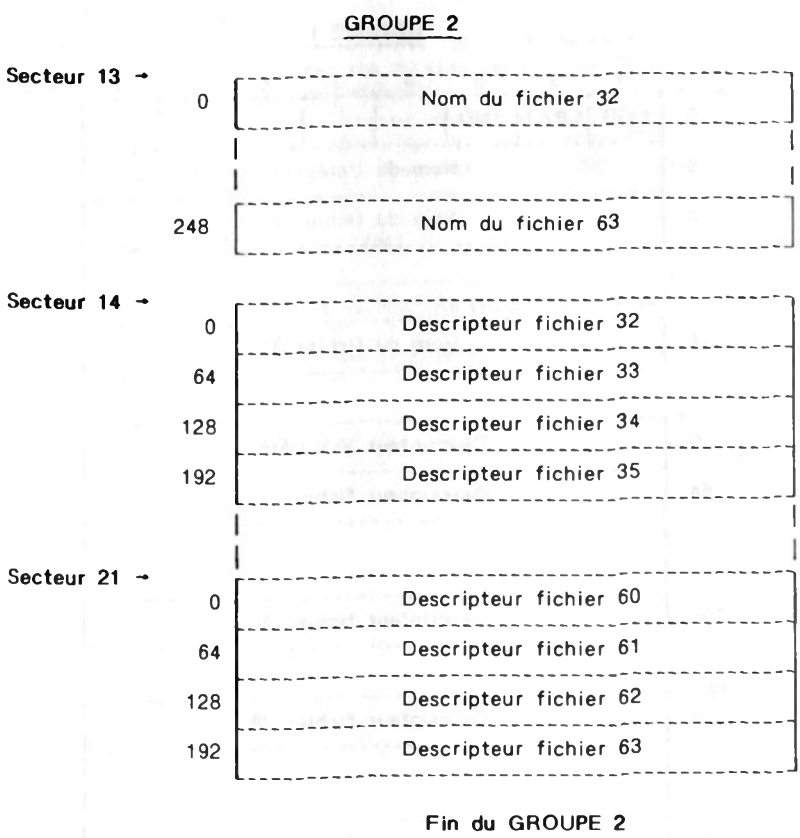
*Identificateur du répertoire (**IND**)*

> Cet identificateur permet au **NGF** de reconnaître un volume géré par lui.

Structure du répertoire



Le premier groupe s'adresse à 31 fichiers.



Et ainsi de suite pour les groupes 3,...

Résumé : organisation du répertoire

Secteurs	Informations
0 à 3	Table d'allocation des granules.
4	Données du volume et NOMS des fichiers **1** à **31**.
5 à 12	Descripteurs des fichiers 1 à **31**.
13 (*)	NOMS des fichiers **32** à **63**.
14 à 21	Descripteurs des fichiers **32** à **63**.
22 (*)	NOMS des fichiers **64** à **95**.
23 à 30	Descripteurs des fichiers **64** à **95**.
:	:
:	:
:	:
247 (*)	NOMS des fichiers **864** à **895**.
248 à 255	Descripteurs des fichiers **864** à **895**.
256 (*)	

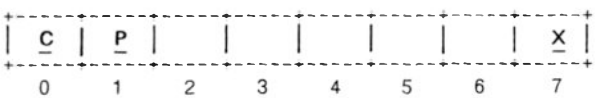
(*) Ces valeurs sont des longueurs possibles du répertoire.

STRUCTURE DES DESCRIPTEURS NOMS DE FICHIERS

Deux types de structures sont possibles :

STRUCTURE 7-1

Le **NOM** du fichier est représenté sur **sept** caractères, l'**EXTENSION (TYPE)** sur **un** caractère.



Le **NOM** du fichier est rangé dans les sept premiers octets de la zone NOM de FICHIER du catalogue.

L'**EXTENSION** est rangée dans le huitième octet de cette même zone.

STRUCTURE 8-3

Le **NOM** du fichier est représenté sur **huit** caractères au maximum, l'**EXTENSION** sur **trois** caractères.

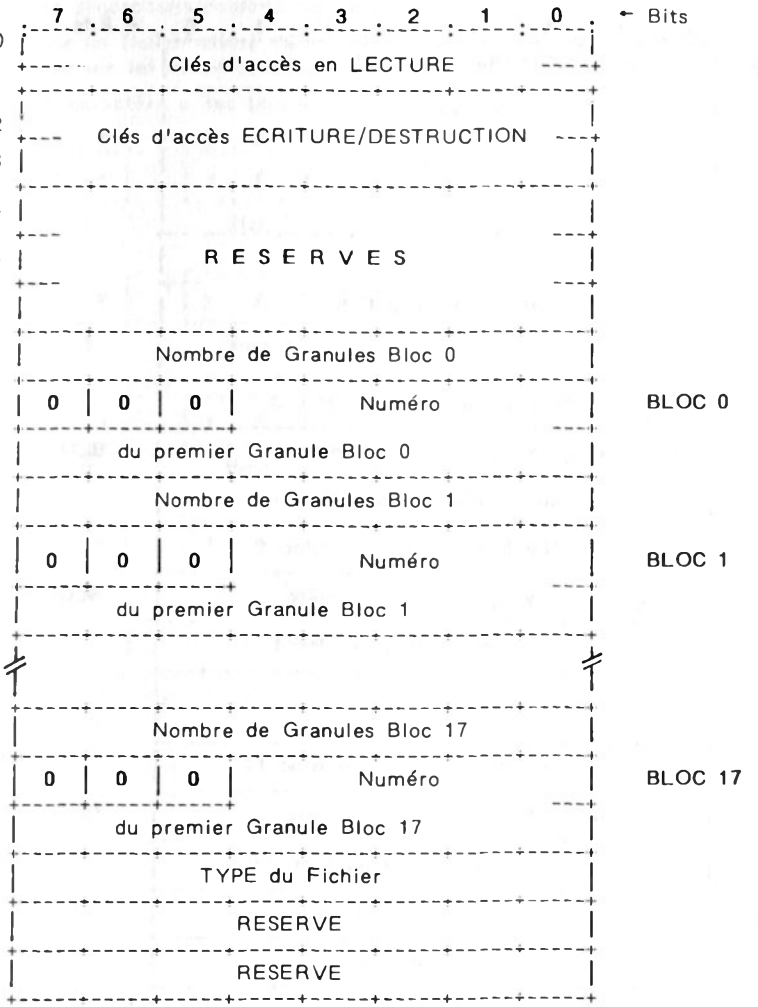
Les **sept** premiers caractères du **NOM** sont rangés dans la zone NOM de FICHIER du catalogue.

Le **dernier** caractère (8e) et l'**extension** sont rangés dans le DESCRIPTEUR DISQUE associé.

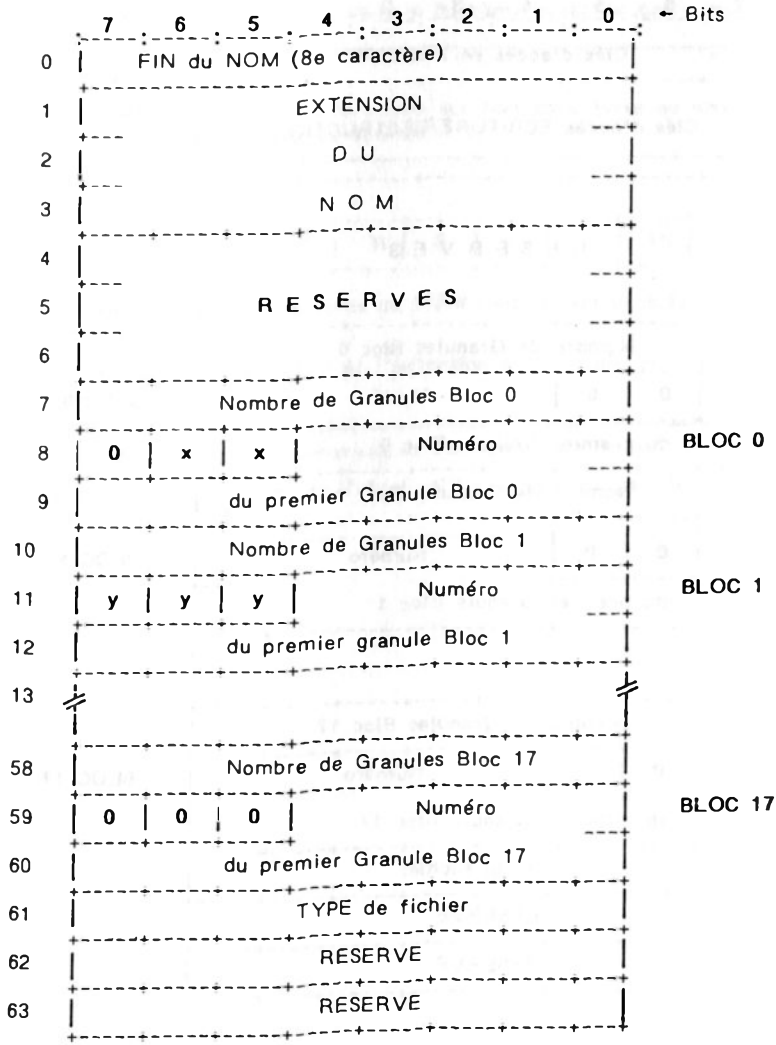
Le **NGF** optimise la mémorisation des noms de fichiers.

Il essaie de mémoriser en **7-1** et, s'il n'y parvient pas, il mémorise en **8-3**.

Structure 7-1



Structure 8-3

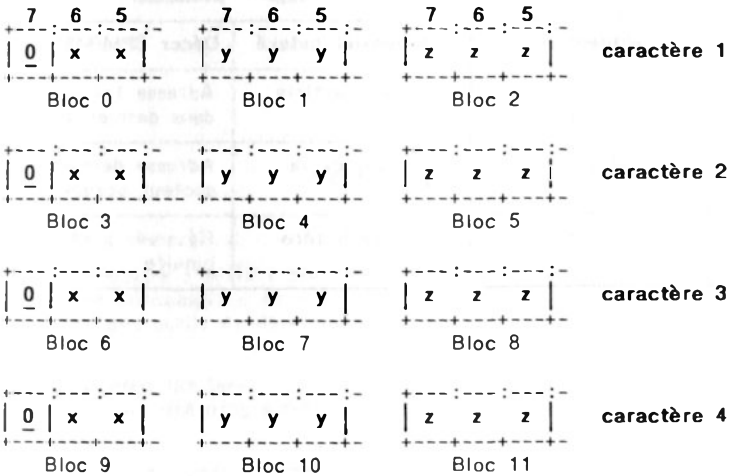


Notes sur les descripteurs

Clés de protection descripteur 8-3

Pour un fichier donné de structure 8-3, les **clés de protection** sont réparties sur les douze premiers blocs d'extension.

Un caractère a ses bits éclatés comme suit :



La clé : P R O T est codée comme ci-dessous.

0	0	1	0	1	0	0	0	0	P	(50h)
0	0	1	0	1	0	0	1	0	R	(52h)
0	0	1	0	0	1	1	1	1	O	(4Fh)
0	0	1	0	1	0	1	0	0	T	(54h)

UTILISATION DES ZONES RESERVEES

Les octets **4, 5, 6** et **62, 63** du descripteur définissent deux zones réservées à des fichiers de nature particulière.

Dans le cas d'un fichier NGF simple, ces zones sont à zéro binaire.

Octets	Séquentiel	Séquentiel-indexé	Décor CPM/MSDOS
4-5	Longueur de l'enregistrement	Longueur article	Adresse 1er octet dans dernier sect.
6	Réservé à zéro	Longueur de la clé	Adresse dernier secteur occupé.
62-63	Pointeur fin de fichier	Réservés à zéro binaire	Réservés à zéro binaire

CHAPITRE 10 programmation système

AVERTISSEMENT

Ce chapitre s'adresse plus spécialement aux programmeurs "système" ayant une connaissance du micro-processeur **8086** et de ses compatibles, ainsi que des outils de développement associés, assembleur, éditeur de liens.

Nous nous limiterons à la présentation des principes généraux relatifs à la structure des programmes et à l'utilisation des primitives système de PROLOGUE.

> Pour un usage pratique, nous vous invitons à vous référer au "guide du programmeur système" **PROLOGUE 86**.

PRIMITIVES SYSTEME

Les primitives système représentent un **ensemble de fonctions standard** permettant l'accès au système d'exploitation PROLOGUE à partir d'un programme écrit en langage assembleur.

On distingue cinq catégories de primitives :

- Les primitives générales du système qui regroupent les fonctions d'analyse d'identificateurs de fichiers, de chargement de programme, d'acquisition de paramètres, etc.
- Les primitives mettant en oeuvre les périphériques tels l'écran, le clavier, l'imprimante.
- Les primitives du système de fichiers qui permettent l'accès aux ressources périphériques de manière directe ou par le biais d'une méthode d'accès particulière ou structurée sur le NGF (SI, MC, BD, ...).
- Les primitives du moniteur multitâche qui permettent la gestion de tâches, de délais, d'événements.
- Les primitives de gestion de la mémoire pour l'allocation et la désallocation dynamique de segments de taille variable dans l'espace mémoire libre.

Les primitives système représentent la **seule interaction** qu'un programme peut avoir avec le système.

Le respect de cette règle garantit au programmeur la portabilité de son logiciel.

Toute autre référence faite au système ou à l'environnement matériel court le risque de ne pas être portable, soit sur d'autres matériels, soit sur des versions futures de PROLOGUE.

Nom	Code	Fonction
PRIMITIVES GENERALES		
ASBN	03h	Conversion ASCII → binaire base n.
BNDC	04h	Conversion binaire → décimal ASCII.
BNHX	05h	Conversion binaire → hexadécimal ASCII.
PSYS	06h	Lecture des paramètres généraux du système.
LDAT	07h	Lecture de la date et heure.
MDAT	08h	Mise à jour de la date et heure.
MEMO	09h	Obtention du mnémonique d'une ressource.
PPOS	0Ah	Lecture/écriture des paramètres de poste.
PRIMITIVES METTANT EN OEUVRE LES PERIPHERIQUES		
PRFO	19h	Paramétrage des fonctions console.
LCFO	1Ah	Lecture du paramétrage des fonctions console.
STCS	1Ch	Status et caractéristiques console.
TPCI	12h	Test si caractère frappé au clavier.
CI	10h	Lecture d'un caractère frappé au clavier.
CLAV	15h	Entrée d'un message à partir du clavier.
CO	11h	Affichage d'un caractère à l'écran.
MES	16h	Affichage d'un message à l'écran.
TPLO	14h	Test si imprimante prête.
LO	13h	Sortie d'un caractère vers l'imprimante.
EDER	17h	Edition d'une erreur à l'écran.
EDRU	18h	Edition d'une erreur utilisateur à l'écran.
PRIMITIVES DE GESTION DE LA MEMOIRE		
ALSM	50h	Allocation d'un segment de mémoire.
LBSM	51h	Libération d'un segment de mémoire.
XTSM	52h	Extension d'un segment de mémoire.
STSM	53h	Status mémoire.

Les primitives du moniteur multitâche et les primitives du système de fichiers vont être étudiées dans les paragraphes suivants.

LE MONITEUR MULTITACHE

C'est le coeur de PROLOGUE ; il offre, au niveau programmeur, un certain nombre de services :

- La **gestion des tâches** : création, activation, etc.
- La **gestion et le contrôle de délais** programmés.
- La **synchronisation des tâches** sur des événements.
- Des **facilités** pour la gestion de sections de programmes non réentrants.

Outre ces services, **MTZ** gère deux catégories de tâches :

- les **tâches immédiates** ;
- les **tâches de fond.**

Les tâches immédiates

Une tâche immédiate est définie par une courte séquence d'instructions activée par l'interruption horloge.

Huit tâches immédiates peuvent être actives à un instant donné.

Les tâches de fond

On distingue deux types de tâches :

- Les tâches de fond prioritaires dont la priorité est définie lors de la création de la tâche.
- Les tâches de fond de même priorité.

Un superviseur de commutation distribue le temps CPU disponible entre les tâches actives à un instant donné.

Une tâche peut se trouver dans l'un des quatre états suivants :

LIBRE	: la tâche est déclarée, mais n'a pas été activée.
PRET	: il s'agit d'une tâche libre qui a été activée et qui attend d'être exécutée.
ACTIF	: il s'agit d'une tâche (et d'une seule) qui, à l'état PRET, dispose effectivement du temps CPU.
ATTENTE	: il s'agit d'une tâche dont l'exécution a été suspendue. Elle restera en attente jusqu'à ce qu'un événement la ramène à l'état prêt.

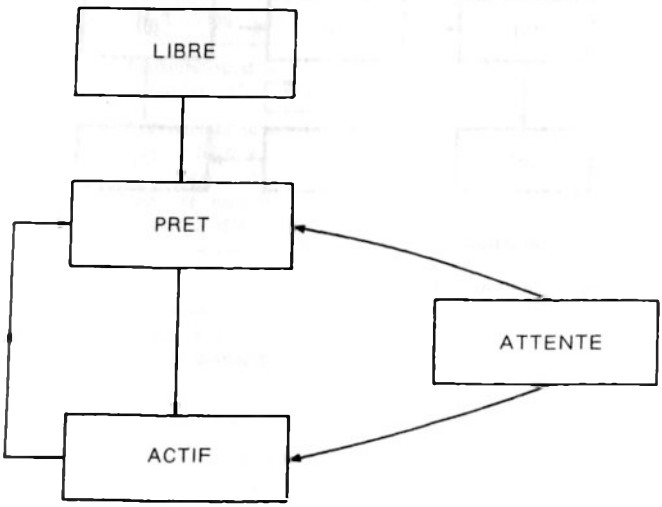
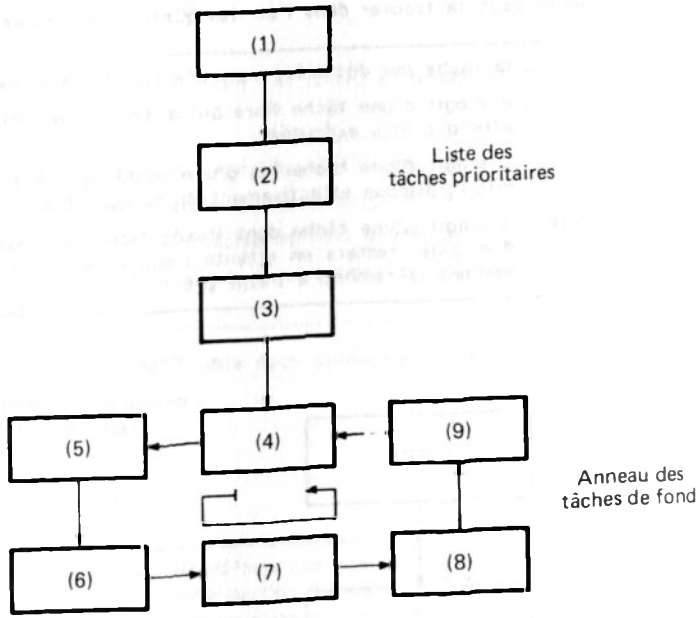


Schéma de transition des états d'une tâche



Tâches prioritaires : niveau 1 à 255

Tâches de fond : niveau 0

Priorité des tâches

Primitives du moniteur multitâche

Nom	Code	Fonction
GESTION DES TACHES		
ACRI	3Fh	Activation d'une tâche immédiate.
RETI	40h	Fin de traitement d'une tâche immédiate.
CRES	38h	Création d'une tâche prioritaire ou de fond.
ACTI	39h	Activation d'une tâche.
DSAC	3Ah	Désactivation d'une tâche.
EXIT	3Bh	Fin d'exécution d'une tâche.
SUPR	3Ch	Suppression d'une tâche.
SYNCHRONISATION ENTRE TACHES		
EVAT	3Dh	Armement d'une attente d'événement.
EVPR	3Eh	Emission d'un événement.
PRIO	37h	Activation/désactivation du superviseur.
DMEX	32h	Demande d'exclusion mutuelle.
FMEX	33h	Fin d'exclusion mutuelle.
NPOS	34h	Lecture numéro de poste actif.
ABQT	35h	Abandon d'un quantum de temps.
GESTION DE DELAIS		
HORL	30h	Armement d'une horloge de tâche.
RAZH	41h	Désactivation d'une horloge de tâche.
ARTO	42h	Armement d'un temporisateur.
DRTO	43h	Désarmement d'un temporisateur.
WAIT	31h	Attente d'un délai ou d'un événement.
WMUL	36h	Attente d'événements multiples.

ACCES AUX RESSOURCES DISQUE

Les ressources disques sont des ressources sectorisées à accès aléatoire gérées par le **NGF.**

L'accès à ces ressources se fait par les primitives du SGF (**20H** ou **21H** dans le registre **AH**) ; l'aiguillage de la requête dépend du code fonction fourni dans le registre **AL** et du numéro de la ressource.

Fonctions d'accès direct au périphérique

> Ces fonctions permettent l'accès direct aux secteurs physiques du périphérique, sans aucune structure de fichier. Le contrôle est donné directement à l'environnement qui gère le contrôleur.

Fonctions d'accès au NGF

> Ces fonctions permettent un accès aux ressources disque avec la structure de fichiers dynamiques définie par le **NGF**.

Un fichier NGF n'a aucun format interne particulier. Il est accédé par l'adresse du secteur (de 256 octets) relative au début du fichier, quelle que soit son implantation physique.

La méthode d'accès Bloc

La méthode d'accès par Bloc de données est bâtie sur la structure et l'organisation des fichiers du NGF.

Elle bénéficie de la gestion dynamique de l'espace disque alloué aux fichiers par le NGF et permet un accès direct à des blocs physiques de longueur quelconque.

Dans cette méthode, un fichier est considéré comme une suite continue de blocs physiques de taille fixe adressables par leur numéro, à partir du début du fichier (0, 1 ... n).

> La méthode d'accès **PAR BLOC DE DONNEES** permet de gérer des enregistrements de longueur fixe mais quelconque.

La méthode d'accès séquentiel

> La méthode d'accès séquentiel (SEQ) permet l'adressage à partir du début d'un fichier, d'enregistrements de longueur fixe définie à la création du fichier.

Primitives du système de fichiers

Nom	Code	Fonction
ANFC	01h	Analyse d'un nom de fichier.
CHGT	02h	Chargement d'un fichier objet -X.
SF	20h	Entrée générale du système de fichiers.
ACCES PHYSIQUE AU SUPPORT		
	00h	Caractéristiques d'un support.
	49h	Status d'un support.
	82h	Prémarquage d'un support à sectorisation matérielle.
	85h	Prémarquage d'une piste à sectorisation logicielle.
	86h	Prémarquage SASI.
	01h	Purge et dévalidation d'un tampon.
	40h	Lecture directe d'un support.
	44h	Lecture avec mise en mémoire.
	80h	Ecriture directe sur un support.
	84h	Ecriture avec mise en mémoire.
ACCES DIRECT		
	30h	Création et ouverture non partageable.
	38h	Création et ouverture partageable.
	34h	Ouverture d'un fichier non partageable.
	3Ch	Ouverture d'un fichier partageable.
	60h	Lecture n secteurs.
	A0h	Ecriture n secteurs.
	24h	Fermeture d'un fichier.
	2Ch	Renommer un fichier.
	28h	Agrandir un fichier.
	20h	Supprimer un fichier.
ACCES NIVEAU BLOC		
	64h	Lecture d'un bloc de données.
	A4h	Ecriture d'un bloc de données.

Nom	Code	Fonction
ACCES SEQUENTIEL		
	31h	Création et ouverture non partageable.
	39h	Création et ouverture partageable.
	35h	Ouverture non partageable.
	30h	Ouverture partageable.
	61h	Lecture enregistrement suivant.
	A1h	Modification de l'enregistrement courant.
	A5h	Ajout d'un enregistrement.
	29h	Positionnement enregistrement précédent.
	25h	Fermeture fichier.

PRINCIPE GENERAL DE L'APPEL

Tous les appels aux primitives système sont faits par l'interruption logicielle **INT xx** avec les paramètres :

En entrée : **AH** = N° de la primitive

Au retour : **AH** = valeurs des flags CF, ZF, PF, SF

La programmation typique d'un appel sera la suivante :

```
Entrée : MOV     AH,N°_de_primitive
         INT     xx          ;exécuter (IT logicielle)
         SAHF                ;mettre à jour les indicateurs
                             ;à partir de AH
         RET                 ;retour à l'appelant
              ...
              ...
              ...
         CALL    entrée      ;appel primitive
              ...
              ...
```

Le numéro de l'interruption logicielle (**xx**) est un paramètre variable selon le matériel. **Sa valeur est fixée par le noyau environnement** ; elle est transmise au programme au moment de l'appel.

En général, le numéro de l'interruption logicielle utilisée en standard est **110** décimal.

> Pour être **portable**, tout module ou programme doit donc prendre en compte la valeur de cette interruption et mettre à jour dynamiquement, au moment de l'appel, l'instruction **INT xx**.

Dans l'exemple ci-après, la zone mémoire référencée **IT:** pourra contenir le numéro du vecteur d'appel des primitives. Ce numéro est fourni dans le registre **AH** lors de l'exécution de la première instruction du programme, ou **CL** si module extension.

```
;POINT D'ENTREE DU PROGRAMME
DEBUT : MOV     CS:BYTE PTR [IT],AH     ;range n° d'IT
        ...     suite programme
```

Exemple d'appel de primitives système :

```
; Entrée avec, dans le registre AH, le numéro de la primitive
APPEL:  DB      OCDh            ;code instruction INT
IT:     DB      110             ;niveau IT (valeur standard)
        SAHF                    ;range les indicateurs
        RET                     ;retour appelant

; Primitive affichage d'un caractère à l'écran
; Entrée : CL=caractère à afficher
CO      MOV     AH,11h          ;fonction
        JMP     APPEL           ;exécution

; Primitive lecture d'un caractère du clavier
; Sortie : AL=caractère frappé au clavier
CI      MOV     AH,10h          ;fonction
        JMP     APPEL           ;exécution

; et ainsi de suite ...

; Pour afficher un caractère à l'écran, on écrira :
        MOV     CL,'x'          ;afficher 'x'
        CALL    CO              ;appel de la primitive
```

STRUCTURE DES PROGRAMMES

Il s'agit de fichiers programmes exécutables de type **-EXE**. Ces fichiers sont le résultat d'une édition de lien de programme(s) objets (**-OBJ**), issus d'un assemblage ou d'une compilation.

Pour être exécutable directement sous PROLOGUE, le fichier **-EXE** doit être renommé en **-X**.

-> **CP,RF,fich-EXE,fich-X**

Si par hasard cette opération venait à être oubliée, le fichier **-EXE** serait exécuté sous DECOR.

Un programme peut être chargé en mémoire soit par l'interpréteur de commande, soit indirectement par un autre programme. Le retour se fera toujours à l'appelant.

Le chargement est fait soit dans la partition du poste appelant, soit dans la partition commune si la place est insuffisante. En fin d'exécution, la place occupée par le programme est libérée.

La structure des segments composant un programme n'est pas imposée mais, en général, on peut distinguer :

- un segment de type **CODE** contenant les instructions du programme en principe non modifiées pendant l'exécution ;
- un segment de type **DATA** contenant les variables propres du programme qui peuvent être initialisées au chargement.

La syntaxe d'appel est spécifique à la primitive référencée ou à la catégorie à laquelle elle appartient.

S'il y a lieu, des paramètres d'appel sont passés par les registres du 8086 : **AL, BX, CX, DX.**

AL représente en général un **code de fonction** s'il est défini.

Si la fonction fait appel à un descripteur mémoire associé, ce dernier est basé dans le registre **DS** et son déplacement (offset) dans le registre **BX.**

Si une adresse est transportée dans le descripteur (ou dans les données associées), ce sera en général une adresse complète :

OFFSET+SEGMENT (deux mots de 16 bits) et exceptionnellement une adresse SEGMENT (un mot de 16 bits).

Au retour, **AL** contient généralement un code réponse et l'indicateur ZF positionné.

ZF=1 → AL=0	OK pas d'erreur
ZF=0 → AL=code réponse NOK	

Des valeurs retour peuvent être rendues soit par les registres **AL, BX, CX, DX**, soit dans le **descripteur** (en mémoire).

AIDE A LA MISE AU POINT

Le moniteur d'aide à la mise au point **MM-X** est un programme utilitaire de PROLOGUE permettant l'exécution contrôlée de programmes écrits en langage assembleur.

Deux méthodes d'appel de **MM-X** :

-> **MM <rc>**

ou

-> **MM,nom_du_programme-X**

Dans les deux cas, le contrôle est donné au programmeur, soit immédiatement, soit après chargement du programme à mettre au point dont le nom a été spécifié lors de l'appel de **MM**.

A chaque fois que le contrôle est donné au programmeur, **MM** affiche le contexte du **CPU**.

```
- CPHZSIDO  AX   BX   CX   DX   BP   SI   DI   ES   IP   Instr
 1--111--  FF00 0000 0000 0000 0000 0000 0000 0100 0000
* (attente commande)
                                  registres          instruction
```

représente l'état des flags du CPU

Si flag à "**1**", un **1** est affiché.
Si flag à "**0**", un - est affiché.

C = Carry	**Z** = Zéro	D = Direction
P = Parité	**S** = Signe	O = Overflow
H = Half carry	**I** = Interruption	

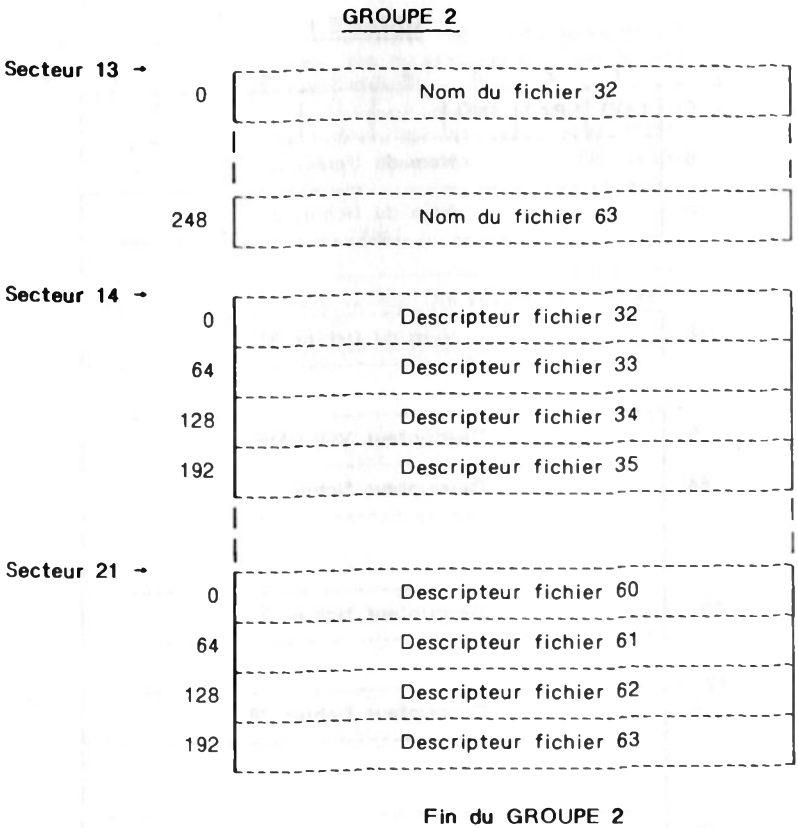
CS:IP pointe sur la prochaine instruction à exécuter dont la traduction symbolique est donnée en fin de ligne.

Liste des commandes de MM

Commande	Fonction
E	Abandon de MM, retour au système.
S	Retour au système, MM restant actif.
espace	Avance pas à pas instruction.
V	Visualisation complète des registres.
R	Modification d'un registre.
M	Modification mémoire.
D	Dump mémoire format octet.
W	Dump mémoire format mot.
L	Liste des instructions en clair.
G	Exécution avec pose de pièges.
P	Lecture/écriture sur port E/S.
C	Chargement d'un programme + paramètres.
F	Initialisation mémoire.
&	Définition d'une base.
O	Mode calculatrice.
B	Pose d'un piège conditionné ou d'une trace. Visualisation table des pièges ou traces. Suppression piège ou trace.

Exemples de commandes

- **CDS:0400** <rc>	Visualise l'octet pointé par DS:0400. Sa modification est obtenue en entrant une nouvelle valeur hexa (2 digits) suivis de <rc>.
- **B2000:0,M,CX=20** <rc>	Pose un piège en 2000:0 avec, comme condition d'arrêt : CX=20h.
- **B100:2100,n=10** <rc>	Pose d'une trace en 100:2100 avec, comme condition de trace : 10e passage.
- **OFFFF** <rc> FFFF décimal:$65535	Conversion hexadécimal en décimal.
- **O14C/2** <rc> 00C6 décimal:$198	Effectue la division par 2 de 14Ch.
- **O20+CX** <rc> 0025 décimal:$37	Addition de 20h au registre CX. (CX contenait 5 initialement).
- **BL2000:0** <rc>	Libération du piège en 2000:0.
- **FDS:BX,'MD0'** <rc>	Initialise la zone mémoire pointée par DS:BX avec la chaîne de caractères "MD0".
- **FDS:SI,55,256** <rc>	Initialise la zone mémoire pointée par DS:SI avec 256 octets égaux à 55h.



Et ainsi de suite pour les groupes 3,...

CHAPITRE 11 télécommunications

GENERALITES

La **TELECOMMUNICATION** a pour but la transmission à distance d'une certaine quantité d'information. Elle implique l'intervention d'un **TRANSMETTEUR** (émetteur), d'un **DESTINATAIRE** (récepteur) et d'une **LIAISON** de communication (voie de transmission du message.

Les communications entre ordinateurs sont des transferts d'information, selon plusieurs **MODES** (asynchrone, synchrone, bit (HDLC, SDLC)...).

Les règles qui permettent de gérer une transmission sont appelées **PROTOCOLES** ou **PROCEDURES** de transmission.

La procédure **BSC** (Binary Synchronous Communication), par exemple, définit un ensemble de règles qui permettent de gérer une transmission synchrone de données entre deux unités centrales éloignées l'une de l'autre.

"Synchronous Communication" signifie qu'après reconnaissance d'une séquence spéciale de synchronisation, le récepteur est synchronisé avec l'émetteur pendant toute la durée de la communication.

> Toutes les procédures de télécommunication sous PROLOGUE disposent, au niveau application, d'une interface unifiée appelée interface **SF** (Système de Fichiers) identique à celle utilisée pour l'accès aux fichiers.

Ce principe permet aux langages évolués d'avoir accès aux télécommunications, sans qu'il soit nécessaire de développer des jeux d'instructions spécifiques à ce service.

L'analogie entre la fonction à exécuter et la primitive SF utilisée apparaît dans le tableau ci-après.

Fonction à exécuter	Primitive SF utilisée	Verbe BAL
Etablissement connexion	**OUVRIR** liaison	**OPEN**
Envoi d'un message	**ECRIRE** données	**WRITE**
Réception d'un message	**LIRE** données	READ
Rupture de la connexion	**FERMER** liaison	CLOSE

Le catalogue PROLOGUE compte les procédures les plus demandées :

- **ASYNCHRONE** ;
- **BSC-2780, 3780, 3741** ;
- **BSC-3270** ;
- **VIP.**

Les deux dernières **BSC-3270** et **VIP** sont multi-stations, c'est-à-dire qu'elles acceptent plusieurs terminaux sur une même ligne physique.

Dans le cas où l'ordinateur dispose de plusieurs voies de transmission, il est possible d'associer une ligne logique à l'une ou l'autre des voies physiques. Cette opération se fait au moyen de paramètres dans le fichier SYSCONF.

Il est également possible d'intégrer, dans un système, plusieurs procédures et de les exécuter simultanément sur des voies différentes ou par sessions alternées si elles utilisent la même voie ; c'est le cas, par exemple, du **MICRAL 9050** de **Bull.**

Certains logiciels comme PTTY (asynchrone) ou X25 sont réentrants et, le cas échéant, gèrent simultanément plusieurs lignes de transmission.

NOTION DE RESEAU

L'objectif de la constitution d'un réseau est de permettre à deux applications de communiquer entre elles sans qu'elles aient à se préoccuper du (ou des) moyen(s) d'acheminement de l'information.

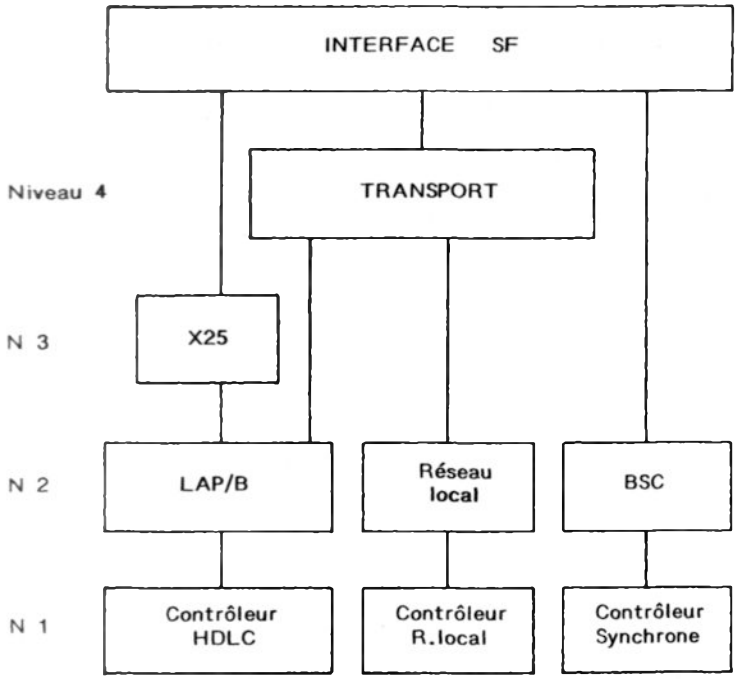
Le dialogue entre les deux applications se fait en établissant une connexion dite de "TRANSPORT".

Le rôle du niveau "TRANSPORT" consiste à établir un lien entre le site adressé et le moyen de l'atteindre. La nature du réseau (local, distant), son type (paquet, "datagramme", ...) ou sa topologie (maille, en

boucle, en ligne) sont autant de paramètres rendus transparents pour l'utilisateur.

> L'interface unifiée de PROLOGUE régit l'échange d'information entre deux niveaux de protocoles.

Par ce moyen, l'utilisateur choisit librement le niveau auquel il désire communiquer (liaison, réseau, transport). Il peut également définir, dans sa configuration, un chemin logique de progression entre les couches existantes.



Correspondance entre l'architecture des logiciels de réseau et les couches définies par le modèle OSI (*)

(*) OSI 7498 (ou ISO) est une norme internationale qui propose une architecture de systèmes ouverts. Cette norme détermine sept niveaux de protocoles permettant l'échange d'information entre des équipements divers. Elle sert aujourd'hui de référence dans la plupart des réalisations modernes en matière de réseaux.

ANNEXE 1
les codes réponses

Les **codes réponses** sont des informations retournées à l'utilisateur par la plupart des primitives du système.

Un code réponse doit être considéré soit comme un code erreur indiquant une exécution anormale d'une fonction, soit comme un indicateur d'une situation particulière (exemple : Fin de Fichier).

Les codes réponses vont de **0** à **127** (décimal), la valeur **0** indique systématiquement une exécution correcte de la fonction.

Ces codes se répartissent en **quatre** catégories, selon leur origine et leur nature :

- Catégorie PERIPHERIE (**1** à **15**)

Cette catégorie regroupe les codes réponses retournés par une fonction d'**E/S** (entrée/sortie) adressée à une ressource périphérique ; ils proviennent des modules de gestion de l'environnement système.

- Catégorie NOYAU SYSTEME (**20** à **29**, **56** à **59**)

Cette catégorie regroupe les codes retournés par le moniteur multitâche (**20** à **26**), le module de gestion de mémoire (**27** à **29**) ainsi que certains codes à caractère général.

- Catégorie COMMANDE (**30** à **39**)

Cette catégorie regroupe les codes retournés par l'interpréteur de commandes et qui correspondent en général à des erreurs syntaxiques dans la commande frappée par l'utilisateur ou à une impossibilité d'exécution de la commande.

- Catégorie NOYAU DE GESTION DE FICHIERS (**40** à **49**, **50** à **55**, **60** à **64**)

C'est l'ensemble des codes réponses retournés par le noyau de gestion de fichier (**NGF**).

La description des codes réponses donnée ci-après est faite par catégorie et par valeur croissante.

Pour chaque code réponse, sont fournis un libellé explicatif (nature de l'erreur) ainsi qu'une description des causes probables.

Les codes réponses soulignés peuvent survenir au niveau opérateur dans des conditions de fonctionnement normal et correspondent à de simples erreurs de commandes, de manipulation ou de configuration.

PERIPHERIE

Erreur	Nature de l'erreur	Cause probable de l'erreur
0 1	PERIPHERIQUE NON PRET	- porte du lecteur disquette ouverte, disquette mal installée.
0 2	ERR DE LECTURE/ECRITURE	- le secteur adressé ne peut être lu ou écrit correctement malgré plusieurs tentatives ; - panne coupleur d'E/S ; - lecteur mal réglé ; - mauvaise qualité du média.
0 3	ERR DE POSITIONNEMENT	- la piste adressée ne peut être localisée ; - causes identiques ERR 02.
0 4	ERR DESCRIPTEUR D'E/S	- la fonction demandée n'est pas supportée par le contrôleur de la ressource adressée ; - l'adresse secteur demandée est au-delà de la capacité prévue du support.
0 5	PROTECTION ECRITURE	- la disquette est protégée par une languette ; - unité placée en protection écriture manuellement.
0 6 0 7 0 8 0 9 1 0 1 1 1 2 1 3 1 4 1 5	Les erreurs 06 à 15 sont spécifiques Elles dépendent de la nature et de la gestion de la ressource périphérique adressée.	

NOYAU SYSTEME

Erreur	Nature de l'erreur	Cause probable de l'erreur
2 0	PARAMETRE INCORRECT	- numéro de tâche égal à 0.
2 1	Réservé	
2 2	TACHE DEJA CREEE	- tentative de création de tâche sous un numéro déjà attribué.
2 3	TACHE INCONNUE	- le numéro de tâche référencé dans la requête n'est pas attribué.
2 4	TABLE DES DESCRIPTEURS DE TACHES SATUREE	- aucun descripteur de tâche n'est disponible pour créer la tâche demandée.
2 5	TACHE DEJA ACTIVE	- tentative d'activer une tâche précédemment activée.
2 6	Réservé	
2 7	ALLOCATION MEMOIRE IMPOSSIBLE	- le plus grand segment libre dans la partition adressée a une taille inférieure à celle demandée.
2 8	ADRESSE PARAGRAPHE INCORRECTE	- l'adresse paragraphe spécifiée dans une fonction de désallocation est invalide.
2 9	SEGMENT NON ALLOUE PAR L'APPELANT	- tentative de désallouer un segment sous un numéro de poste différent de l'allocation.
5 6	FONCTION NON SUPPORTEE	- la fonction système spécifiée n'est pas disponible dans la configuration actuelle.

Erreur	Nature de l'erreur	Cause probable de l'erreur
5 7	FONCTION ATTRIBUT ERRONE	- bit de désignation erroné ; - le fichier est partagé et on demande le mode exclusif.
5 8	FONCTION INTERDITE	- la primitive adressée par l'appelant est supportée par le système mais est interdite dans la configuration.
5 9	STRUCTURE DE VOLUME INCORRECTE	- le nombre de granules du support ne peut être géré par le NGF.

COMMANDE

Erreur	Nature de l'erreur	Cause probable de l'erreur
3 0	SYNTAXE INCORRECTE	- caractère invalide dans un nom de fichier ; - trop de caractères dans un champ ; - séparateur ou mot-clé incorrect.
3 1	RESSOURCE PERIPHERIQUE INCONNUE	- le numéro de ressource fourni dans le descripteur d'appel SF est inexistant.
3 2	NUMERO D'UNITE INCORRECT	- le numéro d'unité spécifié dans la désignation d'une ressource périphérique est supérieur au nombre total d'unité gérées par cette ressource.
3 3	Réservé	
3 4	Réservé	
3 5	Réservé	
3 6	STRUCTURE PROGRAMME INCORRECT	- le contenu du fichier dont l'exécution a été demandée est incorrect.
3 7	Réservé	
3 8	PROGRAMME TROP GRAND	- la taille du programme dont le chargement a été demandé est supérieure à la taille du plus grand segment mémoire disponible.
3 9	Réservé	

SYSTEME DE FICHIERS

Erreur	Nature de l'erreur	Cause probable de l'erreur
4 0	FICHIER INCONNU	- le fichier (nom+type) spécifié dans le descripteur d'appel est inexistant dans le catalogue de l'unité adressée.
4 1	FICHIER DEJA CREE	- tentative de création d'un fichier sous un nom déjà présent dans le catalogue.
4 2	FICHIER NON OUVERT	- le numéro logique utilisé dans un descripteur ne correspond pas à un fichier ouvert.
4 3	FICHIER NON FERME	- tentative de suppression d'un fichier non préalablement fermé.
4 4	FICHIER NON PARTAGEABLE	- tentative d'ouverture d'un fichier déjà ouvert en exclusif ; - recopie d'un fichier sur lui-même ; - partage abusif d'un fichier.
4 5	TROP DE BLOCS D'EXTENSION	- le nombre maximal de blocs composant un fichier est de 18 ; - réorganiser le volume.
4 6	DEBORDEMENT DE VOLUME	- tous les granules du volume adressé sont occupés.
4 7	CLES ACCES INVALIDES	- accès à un fichier avec des clés incorrectes.
4 8	FIN DE FICHIER	- le secteur adressé suit le dernier secteur alloué du fichier.

Erreur	Nature de l'erreur	Cause probable de l'erreur
4 9	CATALOGUE SATURE	- le nombre maximal de fichiers créés sur le volume est atteint (ce nombre est défini à la création du volume).
5 0	NUMERO LOGIQUE INVALIDE	- le numéro logique fourni est égal à 0 ou supérieur au maximum permis.
5 1	TROP DE FICHIERS OUVERTS	- le système ne peut attribuer un numéro logique sur une fonction d'ouverture (ce nombre maxi fait partie des paramètres de configuration).
5 2	ERREUR Nr DE POSTE	- le poste appelant n'est pas celui qui a ouvert le fichier.
5 3	ADRESSE SECTEUR	- l'adresse secteur fournie dans le descripteur est située au-delà de la fin du fichier.
5 4	FORMAT DE VOLUME INCORRECT	- l'unité périphérique adressée ne contient pas un volume PROLOGUE.
5 5	FONCTION INCORRECTE	- la fonction NGF spécifiée dans le descripteur n'est pas reconnue par le système.
6 0	TYPE DE FICHIER INCORRECT	- le fichier n'est pas un fichier séquentiel.
6 1	Réservé	
6 2	LONGUEUR ENREGISTREMENT INCORRECTE	- tentative de création d'un fichier séquentiel avec une longueur d'enregistrement nulle.

Erreur	Nature de l'erreur	Cause probable de l'erreur
6 3	PERTE DE DONNEES EN LECTURE	- la longueur de la zone de réception est inférieure à la longueur de l'enregistrement lu.
6 4	DEBUT DE FICHIER	- tentative de positionnement en deçà du début d'un fichier séquentiel.

CODES ERREURS DU SEQUENTIEL-INDEXE

Erreur	Signification
70	Longueur de clé trop petite.
71	Longueur de clé trop grande.
72	Trop de fichiers ouverts simultanément.
73	Index nul.
74	Clé déjà bloquée.
75	Débordement du fichier des clés.
76	Structure du fichier des clés incorrecte.
77	Débordement du fichier des données.
78	Clé non trouvée.
79	Clé trouvée mais index faux.
80	Trop de niveaux dans le graphe.
81	Clé déjà existante.
82	Taille du tampon plus petite que la clé.

CODES ERREURS DU MULTICRITERE

Erreur	Nature de l'erreur	Cause probable de l'erreur
8 3	FONCTION INEXISTANTE Nr LOGIQUE INCORRECT	
8 4	TROP DE FICHIERS OUVERTS SIMULTANEMENT	- paramétrage système incorrect.
8 5	INCOHERENCE DU DICTIONNAIRE	- erreur fatale, recréer le dictionnaire ; - panne matériel ; - erreur multicritère.
8 6	TROP DE LIENS	- nombre maxi de liens atteint (s'assurer qu'il n'y a pas de liens redondants ; utiliser option ensemble minimal).
	TROP DE RUBRIQUES	- nombre maxi de rubriques atteint (nécessaire de déclarer des rubriques que l'on n'interrogera jamais).
8 7	CARACTERISTIQUE RUBRIQUE INCORRECTE	- vérifier caractéristique.
	LONGUEUR RUBRIQUE INCORRECTE	- somme des longueurs des rubriques > 65535.
	DESCRIPTION RUBRIQUE DEJA EXISTANTE	- même nom pour deux rubriques.
	DESCRIPTION LIEN DEJA EXISTANTE	- même liste, même type.
	REDEFINITION DE RUBRIQUE INEXISTANTE	- vérifier l'orthographe.
	RUBRIQUE A RENOMMER INEXISTANTE	- vérifier l'orthographe.
8 8	DESCRIPTION RUBRIQUE INEXISTANTE	- une rubrique non déclarée figure dans la description de lien ou inter/dénombrement.

Erreur	Nature de l'erreur	Cause probable de l'erreur
8 9	NOMBRE DE CRITERES INCORRECTS	- trop de rubriques dans une description de lien ; - trop de rubriques dans une interrogation.
9 0	DESCRIPTION DE LIEN INEXISTANT	- aucun lien déjà créé ne peut répondre à la question (créer le nouveau lien).
9 1	VALEUR MULTICRITERE TROP LONGUE	- long. rubr. lors de inter./ dénombr. a une longueur supérieure à celle déclarée.
9 2	INCOMPATIBILITE ENTRE DATA et DICO	- données modifiées sans passer par multicritère (erreur matérielle).
9 3	OPERATEUR INCONNU	- vérifier l'opérateur.
9 4	TYPE DE DESCRIPTEUR INCORRECT	
9 5	Réservé	
9 6	FIN DES REPONSES A UNE INTERROGATION FIN DES RUBRIQUES FIN DES LIENS	- il faut d'abord se positionner dans le dico par une fonction interrogation.

CODES ERREURS DE LA BASE DE DONNEES

Erreur	Signification
170	Code fonction inexistant.
171	Trop de fichiers ouverts simultanément.
172	Numéro logique incorrect. Descripteur incorrect. Mode d'ouverture incorrect.
173	Incohérence du dictionnaire.
174	Longueur de clé incorrecte. Type de fichier incorrect.
175	Description de jointure incorrecte. Nombre de joints binaires incorrect. Opérateur incorrect. Jointure sur le même fichier. Jointure impossible. Longueur/types de rubriques différents.
176	Description/nom de fichier existant.
177	Description/nom de fichier inexistant.
178 179	Nom de rubrique existant. Nom de rubrique inexistant.
180	Description/nom de jointure existant.
181	Description/nom de jointure inexistant.
182	Trop de fichiers dans le dictionnaire. Trop de rubriques dans le dictionnaire. Trop de jointures dans le dictionnaire.
183	Pas de POSIT avant DOWN ou UP.
184	Clé trouvée mais index faux.
185 186 187	Pas de réponse pour ce fichier. Réservé. Réservé.

ANNEXE 2

standard des codes fonctions

FONCTIONS D'ENTREE/SORTIE

Fonctions d'entrée/sortie	Codage	Notes
ESCape	1Bh	1
Retour Chariot	0Dh	
Line-Feed	0Ah	
Back-Space	08h	
TABulation	09h	

Notes

1- En entrée, ESCape est souvent utilisé pour terminer un traitement ou revenir à l'interpréteur de commandes.

En sortie, ESCape annonce une fonction particulière dont la signification est donnée par le caractère qui suit.

AUTRES FONCTIONS

Autres fonctions	Codage	Notes
Mode TRANSPARENT	ESC,"T",1,L	1
Recopie d'écran ALPHA-NUMERIQUE	ESC,"O"	2
Recopie d'écran GRAPHIQUE	ESC,"Q"	3

Notes

1- Le mode TRANSPARENT permet de transmettre une chaîne de caractères à un périphérique sans qu'aucune interprétation ne soit faite par le système.

l,L sont deux octets qui spécifient la longueur de la chaîne de caractères qui suit.

l représente l'octet poids faible.
L représente l'octet poids fort.

2- Cette fonction permet une recopie de l'écran sur une imprimante quelconque. Elle est toujours implantée sur un système et peut être activée par une touche spéciale du clavier.

3- Cette fonction permet une recopie d'un écran graphique sur une imprimante graphique. Elle est liée au contrôleur imprimante et peut ne pas être implantée sur un système. Dans ce cas, elle est filtrée. Elle peut être activée par une touche spéciale du clavier.

GESTION DU CURSEUR

Gestion du curseur	Codage	Notes
Curseur droite	**06h**	1
Curseur gauche	**08h**	1
Curseur haut	**0Bh**	1
Curseur bas	**05h**	1
Home	**1Ch**	1
Effacement écran	**0Ch**	1,2
Positionnement curseur	**ESC,"f",x,y**	3
Effacement fin écran	**ESC,"J"**	4
Effacement fin de ligne	**ESC,"K"**	4
Insertion ligne	**ESC,"L"**	4
Suppression ligne	**ESC,"M"**	4
Insertion caractère	**ESC,"C"**	4
Suppression caractère	**ESC,"P"**	4

Notes

1- Ces codes fonction sont utilisés aussi bien en entrée qu'en sortie. En entrée, elles peuvent être matérialisées par une touche du clavier.

2- Si sortie vers une imprimante, il y a saut de page.

3- **x**+20h désigne le numéro de colonne.
y+20h désigne le numéro de ligne.

1B 66 20 20 positionne le curseur en colonne 0 et en ligne 0 (1er caractère en haut à gauche de l'écran), soit l'équivalent de la fonction HOME.

1B 66 25 30 positionne le curseur en colonne 5 et ligne 10.

4- Toutes ces fonctions sont optionnelles.

ATTRIBUTS D'IMPRESSION

Attributs d'impression	Codage	Notes
Caractères compactés	ESC,"x"	
Caractères penchés	ESC,"y"	
Caractères double hauteur	ESC,"t"	
Caractères double largeur	ESC,"z"	

Notes

Ces fonctions sont optionnelles ; elles sont liées au périphérique d'impression utilisé et au module de gestion spécifique.

ATTRIBUTS VIDEO

Gestion des attributs vidéo	Codage	Notes
Retour attribut normal	ESC,"a"	1,4
Attribut préférentiel	ESC,"p"	2,4
Inversion vidéo	ESC,"b"	3
Clignotant	ESC,"c"	3
Souligné	ESC,"d"	3
Sous-brillance	ESC,"e"	3
Sur-brillance	ESC,"h"	3
Transcodage min/MAJ	ESC,"i"	4
Transcodage MAJ/min	ESC,"j"	4

Notes

1- Cette fonction permet de retourner à l'attribut normal de visualisation ou d'impression.

2- L'attribut préférentiel est un attribut imposé par le contrôleur environnement ; il n'est pas nécessairement le même d'un système à un autre. La sélection de cet attribut se traduira, par exemple, par une inversion vidéo sur une machine X, de la sur-brillance sur une machine Y ou encore du souligné sur une machine Z.

3- Ces attributs peuvent ne pas exister sur certains matériels. Dans ce cas, le système, soit les filtre, soit affecte l'attribut préférentiel.

4- Ces attributs sont toujours implantés dans un système. L'attribut préférentiel peut être, sur certains matériels, le forçage des caractères en majuscules.

ANNEXE 3

exemples de programmes B.A.L.

```
            program "MAPDISK"

                    dcl     tW
                    dcl     pW
                    dcl     i%
                    dcl     j%
                    dcl     l%
                    dcl     k%
                    dcl     lect3W
                    dcl     er1W
                    dcl     lgt0%
                    dcl     nbg0%
                    dcl     mW
                    dcl     eW
                    dcl     lgr1%

                    dcl     v$=3,c$=1
                    dcl     t1W(512)
                    dcl     m1skW(8)

                    dcl     s4$=256
                    field=m,s4
                    dcl     lgrW    ;longueur granule
                    dcl     lcatW   ;taille catalogue
                    dcl     indW    ;indicateurs

            segment 0
(0000)
(0000)              lect3=/40
(0004)              for i=1 to 8
(0008)                      read=0:m1sk(i)
(000E)              next i
(0016)
(0016)              print=1;tabv(1),"GEOGRAPHIE D'UN VOLUME",tabv(1)
(003E)
(003E)      10      print=1;tabv(1),atb(0),"Nom du support ",atb(1)
(0062)              ask=1,l=80:=v
(0068)              print=1;tab(2),atb(0)
(0079)              print=1;tabv(1),atb(1),"1 ",atb(0),"- Edition écran"
(00A1)              print=1;tabv(1),atb(1),"2 ",atb(0),"- Edition imprimante"
(00CE)              print=1;tabv(1),"? "
(00DD)              ask=1,i=10:=p
(00E6)
(00E6)              assign=3,v:10,er1
(00F0)              io=3,lect3,0:t1(1),512  ;lecture 512 octets
(0100)              lgt0=t1(1)              ; nb granules sur volume
(0105)
(0105)              io=3,lect3,4:s4,256     ;lecture secteur 4
(0113)              lgr1=lgr*256            ;taille granule en octets
(0119)
```

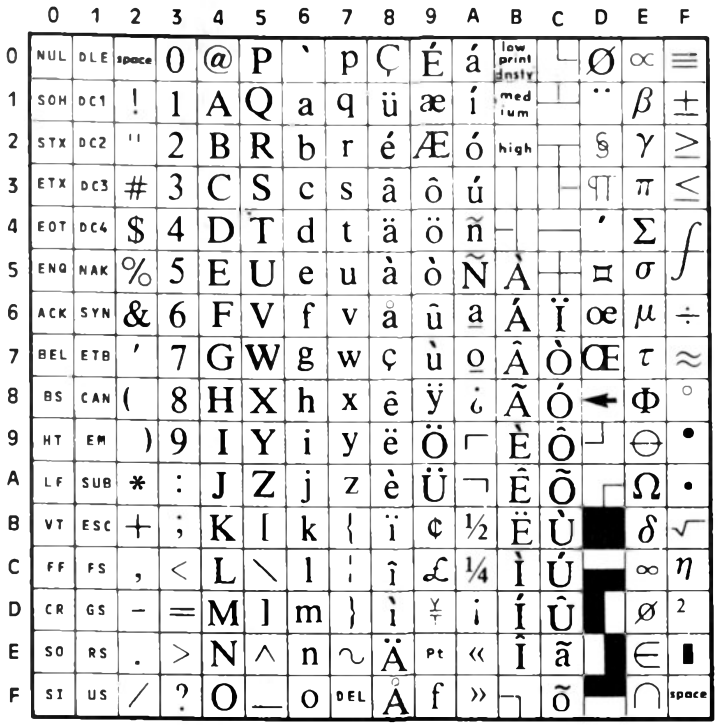
```
(0119)              print=p:clear
(0121)              if p=1 goto 11
(0128)              print=p:"GEOGRAPHIE du VOLUME ",V
(0145)              print=p:tabv(1)
(014F)
(014F)    11        if lgt0 )=0 goto 12
(0156)              lgt0=256+lgt0
(015C)    12        nbg0=t1(2)
(0161)              if nbg0 )=0 goto 15
(0168)              nbg0=256+nbg0
(016E)    15        lgt0=lgt0+nbg0*256
(0177)
(0177)              nbg0=0                      ; nb granules libres
(0178)              i=3                         ; octet courant examine
(017F)              j=8                         ; nb granules dans octet courant
(0183)              k=0                         ; nb granules examines
(0187)    20        l=0                         ; nb granules affiches sur ligne
(0188)              op 22,e
(018F)              goto 27
(0192)
(0192)    22        if e=27 goto 80
(0199)    25        op 27,e
(019D)              goto 25
(01A0)
(01A0)    27        print=p:tabv(1)
(01AA)              print=p:(zzzz),k            ;edite  adresse
(01B6)              print=p:":"
(01BE)              t=0
(01C2)              op 80,e                     ;pour abandon
(01C6)
(01C6)    30        e=t1(i)
(01CA)              for m=1 to j
(01CE)                      c="X"
(01D3)                      if e and m1sk(m) <> 0 goto 40
(01DE)                      c="."
(01E3)                      nbg0=nbg0+1
(01E8)    40
(01E8)                      if t>0 goto 45
(01EF)                      print=p:" "
(01F8)                      t=16
(01FC)    45                print=p:c
(0202)                      t=t-1
(0207)              next m
(020E)              i=i+1
(0213)              k=k+j
(0217)              l=l+j
(0218)              if k=lgt0 goto 65
(0221)              if k+8 > lgt0 goto 60
(0228)    50        if l<64 goto 30
(0232)              goto 20
(0235)
(0235)    60        j=lgt0-k
(0239)              goto    50
(023C)
(023C)    65        gosub   70
(023F)              goto    80
(0242)
(0242)    70        print=p:tabv(2),"Volume ... ",atb(1),v,atb(0)
(0264)              print=p:tabv(1)
(026E)              print=p:atb(1),(zzzzz),lgr1
(027D)              print=p:atb(0)," octets / granule"
(029A)              print=p:tabv(1)
(02A4)              print=p:atb(1),(zzzzz),lgt0
(0283)              print=p:atb(0)," Granules dont"
(02CD)              print=p:tabv(1)
(02D7)              print=p:atb(1),(zzzzz),nbg0
(02E6)              print=p:atb(0)," Granules disponibles"
(0307)              return
(0308)
(0308)    80        stop
(0309)              data /80,/40,/20,/10,/08,/04,/02,/01

          eseg 0
```

```
GEOGRAPHIE du VOLUME sd0

0000: XXXXXXXXXXXXX.X XXXXX.....XXXXX. X...XX..XXXXXXXX XXXXXX..X...XX..
0064: X....XXXXX..X... ....X.......XXXX X...XXXXX..... .....X..X.......
0128: ...XX...X..... ..XXXXXX.XX....X ...XXXXX...XXXX. ...XXXXXXX.....
0192: X...XXX.X.....XX .XXXXX..XX..X... ....XXXXXXXXXX X...X...X......X
0256: X...XXXXXXXXX... X...XXXX.XXXXXXX .XXXXXX.XXXXXXX XXXXXXXXXXXXXXX
0320: XXX.....XXXXXX.X XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0384: XXXXXXXXXX.X... XXXXXXXXX.X..XX XXXXXXXX......X .....XXXXXXXXXX
0448: XXX.........XXX XXX.....X...XXXX XXXX....XXXXX... X.......XXXXXX.
0512: ..XXX...XXXXXXX X.XX..X.XX...XXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0576: XXXXX...XXX.... X.......X....... X.......X....... XX......X.......
0640: XXXXXXXXXXXXXXX X.......X....... X.......XX...... XXXXXXXXXXXXXXX
0704: XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX X.......XXXXXX. XXXX....XXX.....
0768: XXXX....X....... X.......XXX..... XXXXXXXXXXXXX. XXXXXXXXXX.....
0832: ....X...XXX..... XX......X....... XXXXXXXXXXXXX. XXXXXX.XXXXXX.
0896: XXXXXX..XXXX.... XXXX....XXXXX... XX......XXXXX... XXXXXXXXXXX....
0960: XXX.....XXXXX... XXXXX...XXXX.... XXXX....XXXX.... XXXXX...XXXX....
1024: XXX.....XXX..... ....X...X....... XXX.....XXX..... XXX.....XXX.....
1088: XXX.....XXXX.... X.......X....... XX......XX...... X.......X.......
1152: X.......X....... X.......X....... X.......X....... X.......X.......

Volume ... sd0
04096 octets / granule
01216 Granules dont
00567 Granules disponibles
```

Géographie d'un volume obtenue à partir de Mapdisk
X représente un granule occupé,. un granule libre

ANNEXE 4

jeu de caractères Prologue



ANNEXE 5

liste des mots clés B.A.L.

ABS	ASK	ASSIGN	ATB
BACKSPACE	BELL		
CALL	CATALOG	CFILE	CHAIN
CHR	CJOIN	CKEY	CLEAR
CLINK	CLOSE	COL	CONF
CONV	COS	COUNT	
DATA	DATE	DCL	DEFSEG
DELETE	DFILE	DJOIN	DOWN
END	ERROR	ESEG	EXP
EXTEND			
FIELD	FILE	FILLER	FIX
FKEY	FM	FMT	FOR
FP			
GENER	GOSUB	GOTO	
HOME			
IF	INCLUD	INCLUDE	INDEX
INP	INSTR	INT	INV
IO			
JOIN			
KBF	KEY		
LARGE	LDGO.SEG	LEFT	LEN
LET	LFILE	LIN	LINE
LINK	LJOIN	LKEY	LLINK
LOAD	LOG		
MASK	MOD	MODIF	MOVE
NEXT	NKEY		
OF	ON	OP	OPEN
OUT			
PAGE	PAUSE	PEEK	PEN
POINT	POKE	POSIT	PREAD
PRINT	PROCESS	PROGRAM	
READ	RECORD	REM	RENAME
RESUME	RETURN	RET.SEG	RIGHT
RJOIN	RKEY	RND	ROUN
SEARCH	SEGMENT	SGN	SHL
SHR	SIN	SMALL	SPACE
SQR	STAT	STOP	STRN
SUBSTR			
TAB	TABV	TAN	
UP			
VAL	VIEWPORT	VOLUME	VPTR
WAIT	WINDOW	WRITE	

Achevé d'imprimer en octobre 1984
sur les presses de l'imprimerie Laballery et C^ie^
58500 Clamecy
Dépôt légal : octobre 1984

N° d'impression : 410023
N° d'édition : 86699-27-1
ISBN : 2-86699-027-7

le système PROLOGUE

Cet ouvrage est une présentation détaillée de la dernière version du système d'exploitation Prologue, développé par la division Prologue de Bull-Micral (anciennement R2E). Prologue est maintenant disponible sur toute une gamme de matériels 8/16 bits allant du Micral au PC d'IBM. Il possède l'avantage d'offrir des possibilités d'évolution croissante tout en conservant les bases, garantissant de cette manière la pérennité des logiciels développés d'une version à une autre.
L'audience de Prologue n'est pas limitée au seul territoire national puisqu'une enquête récente menée aux USA a montré que les utilisateurs américains s'estimaient fort satisfaits des performances de Prologue.
Les fonctionnalités de Prologue sont analysées :
le système de fichiers ; les utilitaires ;
les décors CP/M 86 et MS-DOS qui permettent d'utiliser sous Prologue les logiciels tournant sous ces deux systèmes d'exploitation ;
l'organisation du système ; les modules d'extension ;
le graphisme ; la programmation système ; les télécommunications (une autre caractéristique importante de Prologue).
Un chapitre est également consacré au langage BAL, compagnon indispensable de Prologue.

DIFFUSÉ PAR PSI

ISBN 2-86699-027-7 110 FF

IMPRIME EN FRANCE

couverture AGRAPH