cii division petits ordinateurs et applications systèmes

COMPAGNIE INTERNATIONALE POUR L'INFORMATIQUE

mitra 15

manuel d'utilisation

Moniteur de base MOB

Gamme	:	MITRA 15
Systèmes	:	MOB – MOB.E
Objet	:	Ce manuel décrit le moniteur de base MOB de l'ordinateur MITRA 15. On y trouve la description des commandes, l'organisation des E/S, le traitement des déroutements et l'utilisation des modules du moniteur.
Remarques	:	Cette édition correspond à la version 05 du moniteur. Les modifications sont indiquées par un trait vertical en marge du texte.
Nombre de pages	:	73
Date d'édition	:	MARS 1974

Pour commander ce document, envoyez votre demande à l'adresse ci-contre en reproduisant intégralement la "référence document".

Compagnie Internationale pour l'Informatique
CIDOC 68, route de Versailles LOUVECIENNES 78

Référence document :
4055 U3/FR

Moniteur de base MOB

manuel d'utilisation

SOMMAIRE

4055 U3/FR

SOMMAIRE
(Suite)

Bibliographie

4029 Manuel de présentation – Hardware
4399 Manuel de présentation – Software
4057 Manuel de référence
4055 Moniteur de base MOB – Manuel d'utilisation
4154 Moniteur de base MOB – Fiche opérateur
4111 Moniteur temps réel MTR – Manuel d'utilisation
4117 Moniteur temps réel disque MTRD – Manuel d'utilisation
4086 Assembleur MITRAS 1 – Fiche opérateur
4087 Assembleur MITRAS 2 – Fiche opérateur
4155 Assembleur MITRAS 1 S.A. – Fiche opérateur
4083 Editeur-chargeur, éditeurs de liens – Manuel d'utilisation
4084 Editeur-chargeur, ECH – Fiche opérateur
4085 Editeur de liens EDL – Fiche opérateur
4260 Editeur de liens étendu EDL-E – Fiche opérateur
4267 Editeur de liens disque EDL-D – Fiche opérateur
4076 Utilitaires – Manuel d'utilisation
4081 Manipulation de fichier binaire COPY – Fiche opérateur
4082 Correction de texte source RCEDIT – Fiche opérateur
4303 PATCHD – Fiche opérateur
4304 DUMPDS – Fiche opérateur
4305 UTIL – Fiche opérateur
4056 Bibliothèques mathématiques – Manuel d'utilisation
4259 Bibliothèques mathématiques – Fiche opérateur
4332 Module d'enchaînement BATCH – Manuel d'utilisation
4253 Bibliothécaire BIB – Manuel d'utilisation
4268 Bibliothécaire BIB – Fiche opérateur
4113 SGF 50 bandes – Manuel d'utilisation
4109 Langage LP15 – Manuel d'utilisation
4235 Langage LP15 – Fiche opérateur
4112 Langage FORTRAN IV – Manuel d'utilisation
4110 Langage BASIC – Manuel d'utilisation
4256 Langage BASIC mono-console autochargeable – Fiche opérateur
4284 Langage BASIC mono-console chargeable – Fiche opérateur
4115 Macro-générateur MAG15 – Manuel d'utilisation
4295 Macro-générateur MAG15 – Fiche opérateur
4247 Langage de programmation de gestion LPG 15 – Manuel d'utilisation
4242 Langage symbolique d'enseignement LSE 15 – Manuel d'utilisation
4149 Spécifications générales d'installation – Manuel de présentation
4245 Guide du couplage – Manuel d'utilisation
4276 Pupitre 1520 – Fiche opérateur
4266 Téléimprimeur de service 15001 – Fiche opérateur
4262 Lecteur/perforateur de ruban 15060 – Fiche opérateur
4367 Lecteur de cartes 15122 – Fiche opérateur
4263 Lecteur de cartes 15120 – Fiche opérateur
4268 Perforateur de cartes 15160 – Fiche opérateur
4265 Imprimante 15410 – Fiche opérateur
4369 Imprimante 15413 – Fiche opérateur
4354 Disque rapide 15200 – Fiche opérateur
4356 Disque amovible en cartouche 15270 – Fiche opérateur
4370 Disque amovible en cartouche 15280 – Fiche opérateur
4355 Dérouleur de bande magnétique 15300/310 – Fiche opérateur
4420 Catalogue des produits programmes MITRA 15
4422 Catalogue des produits (hardware)

1. Introduction

Le MOB (Moniteur de Base) a été particulièrement développé pour assurer avec une configuration minimale de 8 K mots mémoire et télescriptrice :

- Le contrôle de l'ordinateur (traitement des déroutements et des interruptions, dialogue avec l'opérateur).
- Le traitement des Entrées/Sorties.
- Des fonctions de service (conversions etc...).

Ces fonctions principales sont traitées selon un point de vue temps-réel :

- Connection de programmes à un niveau d'interruption (MOB-E).
- Multiprogrammation assurée par le dispositif de priorité hardware des interruptions.
- Modules réentrants.

Le MOB agit par ailleurs sur un environnement logique par le biais des étiquettes opérationnelles (voir manuel de référence MITRA 15, chapitre "Entrées/Sorties"). Il offre donc toutes possibilités d'adaptation à un environnement physique par simple adjonction des modules de contrôle des périphériques utilisés (lecteur de ruban rapide, lecteur de cartes, imprimante, etc...).

Plus généralement, de par sa modularité et grâce au système de génération, le MOB peut s'adapter parfaitement aux besoins de l'utilisateur par adjonction de modules réentrants au noyau résident ou de modules de programme immédiats. Il est de plus compatible avec les moniteurs plus puissants.

Le Moniteur de Base existe sous deux versions principales : le MOB (version minimum) et le MOB-E (version étendue). Le MOB-E possède, en plus d'une plus grande souplesse d'utilisation, un jeu de commandes supplémentaires ainsi que, pour les commandes déjà existantes dans le MOB, des options supplémentaires, principalement destinées à la mise au point des programmes. Elles sont décrites dans le chapitre 9 de ce manuel.

2. Système MOB

On appellera système MOB, l'ensemble des éléments de programmes standard utilisables sur une configuration minimale (unité centrale - 8 K mots de mémoire ferrite - téléscriptrice - 16 registres (bloc 0 et bloc 1)).

Ce système MOB est susceptible d'extensions, tant du point de vue moniteur, que du point de vue processeurs et les bibliothèques.

2-1. NOYAU RESIDENT

C'est lui qui assure :

- Le contrôle de l'ordinateur,
  - . attente de l'interruption pupitre,
  - . édition des messages opérateurs,
  - . analyse des commandes.
- Le chargement et le lancement des programmes,
- La gestion des entrées-sorties,
- Le traitement des déroutements,
- Le traitement des interruptions.

2-2. PROCESSEURS

- Assembleur MITRAS 1.
- Editeur de liens.
- Editeur-chargeur.
- Programmes de service (traitement des textes source et binaire).

2.3. BIBLIOTHEQUES

- Bibliothèques mathématiques (virgule fixe, virgule flottante, conversion de format, etc...).
- Bibliothèque temps réel.

Etc... .

La mise en oeuvre (détaillée dans la fiche opérateur) est simple : chargement du moniteur par bootstrap, prise de contrôle par interruption pupitre, chargement et lancement des processeurs et des programmes au niveau zéro par les commandes %LOAD et %RUN, utilisation des modules de bibliothèque par inclusion au programme utilisateur (édition de liens) pour leur version CLS par inclusion au noyau résident, par génération pour leur version CSV.

2-4. EXTENSIONS DU MOB

Les extensions du MOB sont de quatre types :

■ Extensions propres au MOB-E

- Commandes et options de mise au point
- Module M:DUMP
- Zone commune étendue

■ Adjonction de modules IMT par génération

Ceci concerne :

- les handlers,
- les modules simulant les instructions optionnelles (traitement des déroutements),
- les programmes immédiats utilisateur (connectés à une IT).

Ceci permet d'adapter le MOB à n'importe quel environnement.

■ Adjonction de modules BT par édition de liens

Ceci concerne :

- des modules de bibliothèque standard (version CSV),
- des modules utilisateur (qui doivent être réentrants, c'est-à-dire travailler dans le TWB de l'appelant),
- des modules superviseur standard (Adjonction des commandes de mise au point par exemple).

■ Processeurs

Avec une configuration mémoire plus élevée, utilisation de MITRAS 2, LP15, FORTRAN, etc... .

Les commandes et options supplémentaires du MOB-E et le module M:DUMP sont décrits dans le chapitre 9 de ce manuel.

La simulation des instructions optionnelles est précisée au chapitre 7.

Les éléments qui concernent les handlers et les modules de bibliothèque standard, n'étant pas spécifiques d'un moniteur particulier, sont décrits respectivement dans le chapitre Entrées/Sorties du manuel de référence MITRA 15 et le manuel d'utilisation des bibliothèques.

Les éléments qui concernent les encombrements et les modes opératoires sont décrits dans les manuels de génération et les manuels d'opération des bibliothèques IMT et BT de génération.

3. Structure du système et organisation de la mémoire

3-1. STRUCTURE DU NOYAU RESIDENT

Le noyau résident est toujours implanté à partir de l'adresse absolue 0 de la mémoire vive. Il comporte :

■ Les données système :

. adresses de communication avec la micromachine,

. zone de sauvegarde sur déroutement,

. adresses des tables-système,

. adresses de communications entre sous-programmes communs,

. constantes-système (pointeurs de contexte ; tables d'entrée-sortie ; etc...).

■ Les sous-programmes communs (ou modules du superviseur)

Ce sont les modules accessibles à l'utilisateur par CSV ou bien des modules strictement internes.

■ Les handlers

Ce sont des modules de contrôle physique des transferts sur périphérique. Ils constituent chacun une "tâche immédiate" connectée à l'interruption associée au périphérique.

Le MOB dans sa version minimale ne comprend que le Handler téléscriptrice. Des handlers supplémentaires sont intégrables à la génération et sont disponibles sous forme IMT.

■ Les tâches immédiates

Ce sont des modules de programme liés à un niveau d'interruption. On peut distinguer :

- Le programme de traitement de l'interruption pupitre.
- Les programmes de traitement des interruptions "coupure secteur" et "retour secteur".
- Les tâches immédiates utilisateur incluses à la génération réalisant le traitement de l'environnement temps réel.

3.2. ORGANISATION DE LA MEMOIRE

- Zone moniteur

A partir de l'adresse zéro jusqu'à la fin du noyau résident y compris les tâches immédiates utilisateur.

- Zone utilisateur

Depuis la fin du noyau résident jusqu'au début de la zone commune.
Cette zone reçoit les programmes utilisateur et les processeurs. Ces programmes sont connectés au niveau zéro ou à un autre niveau (exceptés les processeurs qui sont toujours au niveau zéro).

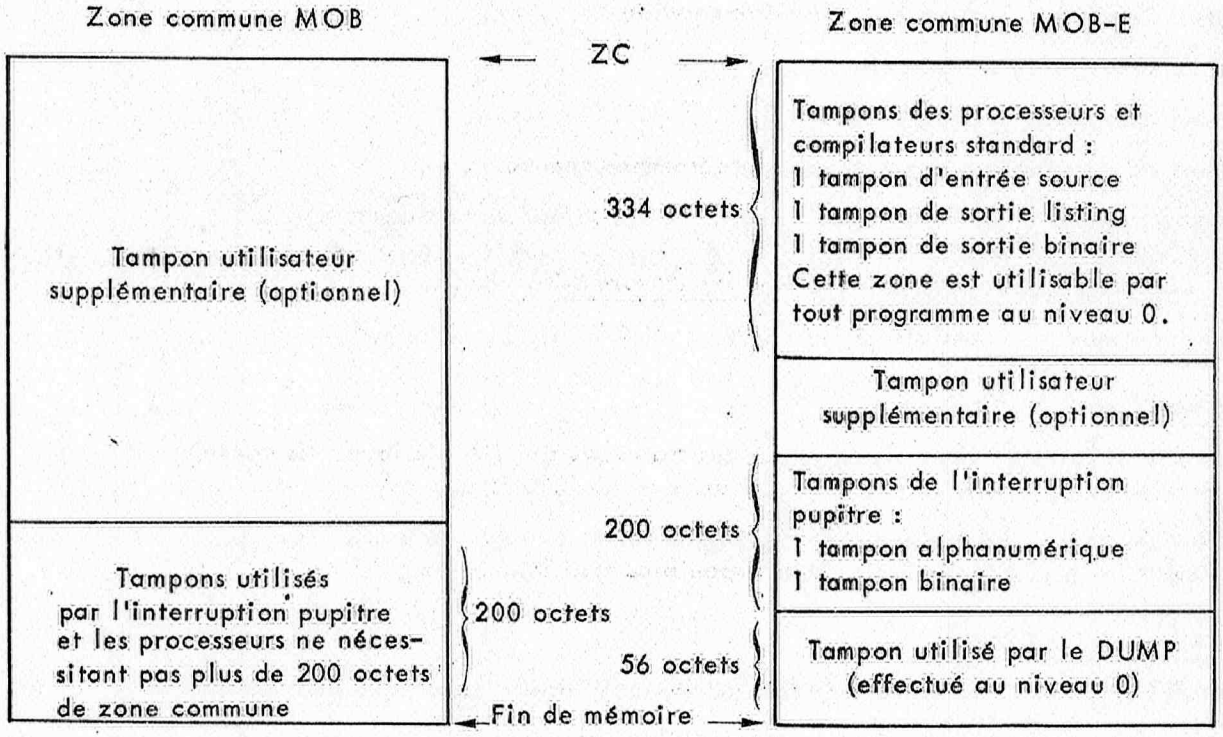
La multiprogrammation est assurée par le système de priorité hardware des interruptions.

zone commune.

La zone commune sert aux communications entre des programmes distincts (non simultanément en mémoire) ou entre des tâches immédiates et un programme au niveau zéro. Elle assure en particulier les communications entre le programme immédiat de traitement de l'interruption pupitre (commandes moniteur) et les processeurs. Les processeurs l'utilisent également pour leurs tampons.

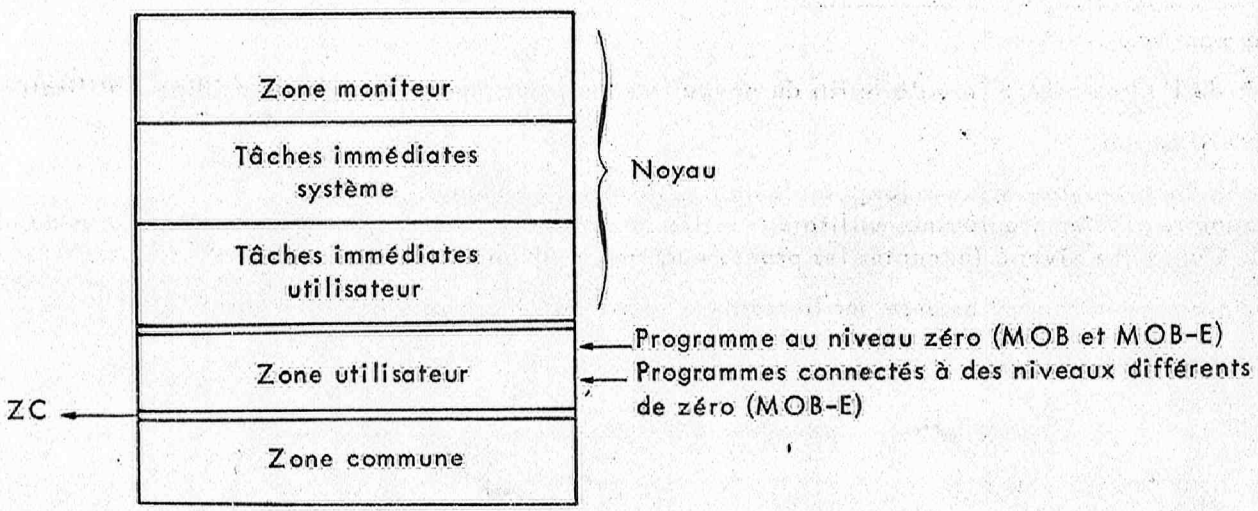
Les besoins minima de zone commune du moniteur sont de 200 octets (1 tampon alphanumérique de 80 octets et un tampon binaire de 120 octets) mais la taille de cette zone commune peut être fixée librement à la génération de système.

L'adresse de la zone commune est disponible dans le quatrième mot de la CDS du programme utilisateur (adresse G + 6). Cette adresse de la zone commune est implantée par le moniteur lors du traitement de la commande %LOAD et est exprimée relativement à la base G du programme utilisateur. Cette zone commune prend deux structures légèrement différentes selon qu'on considère le MOB ou le MOB-E.



Pour assurer un fonctionnement normal de son système, l'utilisateur devra vérifier la compatibilité entre la taille de la zone commune de son moniteur et les besoins des processeurs standard qu'il utilise. Les besoins en zone commune des processeurs standard figurent dans les notices opérateur correspondantes.

En standard, les MOB ont 200 octets de zone commune et les MOB-E en ont 590, mais la taille de la zone commune peut être fixée librement à la génération.



La compatibilité d'accès à la zone commune est assurée par le fait qu'au lancement d'un programme le moniteur transmet dans X l'adresse de la zone utilisable par le programme (exprimée relativement au début de la zone commune). Sous MOB, cette valeur vaut zéro, mais il n'en est pas de même sous MTR ou MTRD.

4. Organisation des entrées/sorties

Une entrée-sortie peut se décomposer en cinq phases :

1 - Réservation éventuelle d'un tampon.

2 - Demande de transfert.

3 - Initialisation du transfert.

4 - Déroulement du transfert.

5 - Fin de transfert et contrôle de validité.

La première et la deuxième phase sont à la charge du demandeur d'entrée-sortie (appel CSV M : IO).

La troisième phase est assurée par le moniteur après l'appel à M : IO. Le retour au programme appelant est fait après cette initialisation.

Les deux dernières phases (4 et 5) sont assurées par le système d'une façon totalement indépendante du programme appelant, en simultanéité apparente avec le programme en cours d'occupation de l'unité centrale, le programme appelant pouvant suivre l'évolution du transfert s'il le désire.

4-1. INTERFACE D'ENTREE-SORTIE

Une grande partie du traitement réalisé par le noyau résident est commune à tous les types de transfert dans les phases 2, 3 et 5 décrites au paragraphe précédent :

a - Analyse de la demande de transfert pour un traitement préliminaire.

b - Mise en place des paramètres nécessaires à l'initialisation physique et au contrôle du transfert après une éventuelle boucle d'attente.

c - Branchement au module de contrôle du transfert propre au périphérique (handler).

d - Prise en compte et traitement de certaines erreurs en fin de transfert.

L'ensemble de ces traitements communs à tous les transferts, effectués entre l'appel de l'utilisateur et la prise en charge du transfert par un handler, puis entre la fin de transfert et sa validation pour le compte de l'utilisateur, constitue l'interface d'entrée-sortie.

Cet interface peut être divisé en deux parties :

- 1re partie : M : IO comportant a, b et c, donc précédant le lancement effectif du transfert.
- 2e partie : M : IO2 comportant d suivant la fin effective du transfert.

4-2. HANDLER

L'initialisation effective du transfert de ou vers un périphérique, le contrôle de cette initialisation, le suivi éventuel du transfert et la prise en compte de fin de transfert dépendent du type de périphérique, c'est-à-dire du coupleur.

Par suite, à chaque coupleur doit correspondre un module de contrôle, c'est ce module de contrôle qu'on appellera HANDLER.

D'une façon identique à l'interface, le handler se décompose en deux parties :

H 1 handler d'initialisation.

H 2 handler de contrôle de transfert.

Le handler 2 est un programme immédiat, en mode maître, déclenché par les interruptions en provenance du coupleur.

Selon le type de périphérique, le coupleur associé fonctionnera suivant l'un des deux modes suivants :

a - Transfert par bloc de données,

b - Transfert donnée par donnée.

Dans le cas d'un transfert de bloc selon le mode b, le handler commandera le transfert des données une à une et assurera le remplissage ou le vidage du bloc de données, soit sous le contrôle complet du handler 2, soit par relance du handler 1 par le handler 2 pour chaque donnée à transférer.

Les handlers sont inclus au noyau résident par génération suivant la configuration à contrôler.

4-3. APPEL D'ENTREE-SORTIE : CSV M : IO

Au call superviseur (CSV) pour demande de transfert, est associé un bloc de commande : le CB (control block). Ce bloc de commande contient les paramètres définis par l'utilisateur. En fin de tranfert, il comporte également des informations d'état rendues par le système. (Voir description du CB au paragraphe de description de M : IO).

Au moment de l'appel, le registre A doit contenir l'adresse du CB relative à G.

La séquence d'appel est alors la suivante :

```
LEA     CB
CSV     M : IO
```

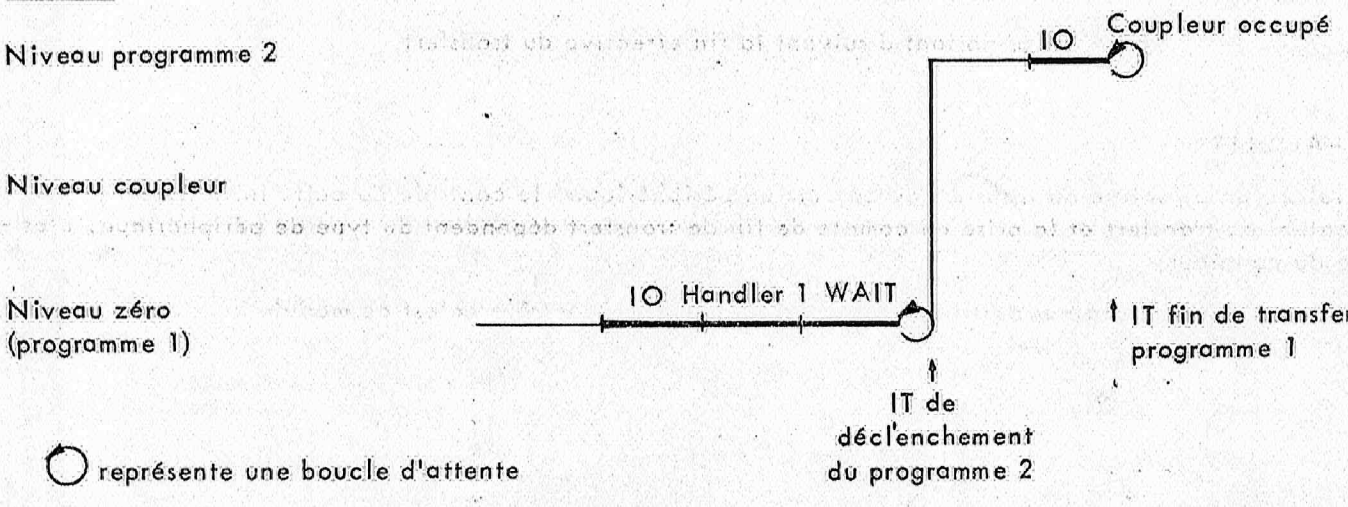
Le moniteur associe à la demande de transfert, un évènement dynamique défini par l'adresse du CB : c'est le bit zéro de l'octet zéro du CB. Ce bit est positionné à 1 dès la reconnaissance de l'appel de transfert. Il est remis à zéro en fin de transfert (évènement activé).

Il n'y a pas de file d'attente d'entrée-sortie. Le moniteur MOB ne rend le contrôle au programme appelant qu'après l'initialisation effective du transfert.

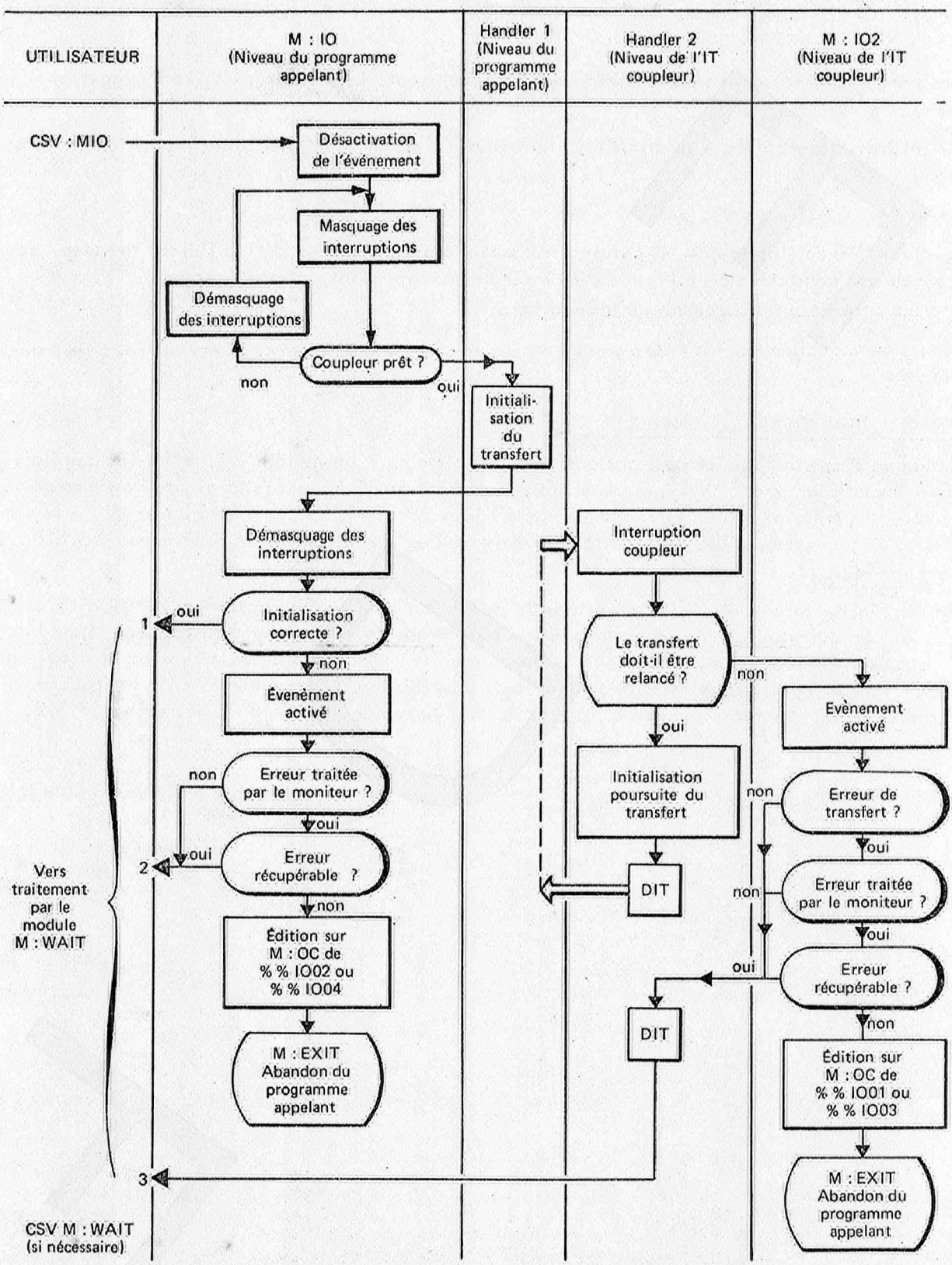
L'utilisateur peut alors faire appel au module M : WAIT (attente activation d'évènement, l'évènement étant ici la fin de transfert).

Un programme ne doit pas demander d'entrées/sorties sur un périphérique connecté à une interruption moins prioritaire que lui-même : l'absence de files d'attentes entraînera un blocage du système.

Exemple :



4-4. SCHEMA D'ORGANISATION D'UNE ENTREE-SORTIE SOUS MOB



4-5. ATTENTE FIN DE TRANSFERT : MODULE M : WAIT

Du point de vue de l'utilisateur, le transfert est commencé dès la prise en compte de l'appel d'E/S par le module M : IO. Celui-ci commence par positionner à 1 le bit événement de l'octet 0 du CB. Il désactive l'évènement.

Le transfert se termine après remise à zéro du bit évènement. Le moniteur active l'évènement (cas 2 et 3 du schéma).

Après l'initialisation correcte du transfert, le retour a lieu en 1. L'utilisateur peut alors faire appel au module M : WAIT.

Le rôle de M : WAIT est de :

- Désactiver l'IT de l'appelant, si celui-ci est connecté à un niveau d'IT différent de zéro, pour permettre aux programmes connectés à un niveau d'IT inférieur à l'appelant, d'être exécutés.
- Mettre en attente un programme au niveau zéro.
- Effectuer le branchement sur "adresse de reprise en cas d'erreur", si celle-ci est spécifiée dans le CB d'entrée-sortie.

4-6. ETIQUETTES OPERATIONNELLES

Une demande d'Entrée/Sortie n'est pas adressée directement à un handler. En effet les handlers gèrent directement l'environnement physique. Une liaison directe aux handlers rendrait la programmation des Entrées/Sorties toujours tributaire d'un environnement physique précis. C'est pour se libérer de cette contrainte qu'a été utilisé le système des "étiquettes opérationnelles" de façon qu'un programme travaille sur un environnement logique.

Une Entrée/Sortie sera adressée à une étiquette opérationnelle (voir description du CB d'Entrée/Sortie au chapitre 8). La correspondance entre étiquette opérationnelle et handler sera fait par le module M : IO. Cette correspondance est fixée à la génération du moniteur. Sous MOB-E, elle est modifiable par la commande %ASSIGN. Les étiquettes opérationnelles, décrites avec le module M : IO et la commande %ASSIGN représentent des fonctions logiques (Entrée de commande, Sortie de binaire, etc...).

5. Interruptions

Le MOB assure le contrôle des interruptions du système qui dépendent de la configuration hardware.

Le MOB contrôle :

- L'interruption pupître,
- Les interruptions de coupure et retour secteur,
- Les interruptions de tous les coupleurs dont les handlers ont été inclus à la génération,
- Toute autre interruption à laquelle a été connecté un programme immédiat (à la génération pour MOB ou également au moment du RUN pour MOB-E).

Interruption pupitre

Cette interruption est déclenchée par pression sur le bouton poussoir correspondant du panneau de commande.

Le module de programme immédiat lié à cette interruption :

- Edite "retour chariot" "interligne" % sur la téléscriptrice,
- Assure l'entrée d'une commande opérateur sur M : OC,
- Appelle la section d'analyse de commande phase 1 (analyse syntaxique),
- Contrôle les paramètres de la commande,
- Appelle la section d'analyse de commande phase 2 (exécution),
- En fin d'éxécution, se désactive pour attendre une nouvelle interruption.

Les interruptions de coupure et retour secteur

Au niveau 31 est lié le programme immédiat de coupure secteur qui initialise le programme immédiat de retour secteur et effectue un RAZ du Mitra.

Au niveau 30 est lié le programme immédiat de retour secteur qui réinitialise le système et édite à la téléscriptrice :

%%
ZZ CS
SYS READY

La réinitialisation effectuée consiste en :

- réinitialisation du noyau moniteur (en particulier des tables d'entrée/Sortie).
- réinitialisation des contexes de tous les programmes immédiats (exceptés ceux des niveaux 30 et 31) et des handlers avec le premier élément de leur PRT. Tous les programmes immédiats devront donc être lancés par leur première section et tous les handlers devront avoir comme premier LPS le handler 2.
- libération des handlers : chaque handler possède à l'adresse héxadécimale 16 de leur LDS un mot d'occupation destiné à les protéger contre les interruptions parasites ; ce mot est réinitialisé à zéro.
- mise en attente du système (attente d'interruption).

Remarque : Le MOB1 ne possède pas cette protection et le système devra être rechargé en cas de coupure secteur.

■ Les interruptions des coupleurs (traités par handler)

Ces interruptions sont traitées par le système d'Entrée/Sortie composé de M : IO, M : WAIT et des handlers inclus à la génération.

Autres interruptions

Les autres interruptions qui n'auraient pas été connectées à une tâche immédiate à la génération sont initialement associées à un DIT de façon à protéger le système contre les interruptions parasites.

Elles pourront être connectées dynamiquement à une tâche utilisateur par la commande %RUN dans le cas MOB-E.

La gestion des interruptions connectées à une tâche immédiate à la génération se limite au branchement à la tâche lors de l'arrivée de l'interruption correspondante.

Fin d'une tâche

. Fin normale :

Une tâche se termine normalement pour un appel moniteur CSV M : EXIT. Le système désactivera alors le niveau de la tâche appelante.

. Fin anormale :

En cas de déroutement dans un programme immédiat il y a réinitialisation de l'ensemble du système (sauf sous MOB 1) car le programme immédiat fautif à vraisemblablement perturbé l'ensemble du système (chapitre 6).

6. Traitement des déroutements

Un déroutement a pour origine un incident détecté à la fin d'une micro-instruction lors de l'éxécution d'un instruction ou d'une fonction de couplage d'Entrée/Sortie. On parlera respectivement de déroutement "programme" et de déroutement "coupleur".

6-1. DIFFERENTS TYPES DE DEROUTEMENT

- Adresse inexistante.
- Violation de protection mémoire.
- Défaut de parité sur mémoire ferrite.
- Violation de mode.
- Instruction inexistante.
- Horloge de garde d'E/S.

Lorsqu'elle détecte l'incident, la micro-machine :
- protège dans les octets d'adresse 4 à 9 de la mémoire, le contenu de L et P et des indicateurs,
- indique par des bits du mot d'adresse 2 de la mémoire, quelle est la cause du déroutement, (mot d'état déroutement)
- réalise l'appel à la section zéro du superviseur (module M : TRAP).

6-2. SIMULATION DES INSTRUCTIONS OPTIONNELLES

Le déroutement instruction inexistante peut être mis à profit pour réaliser la simulation de certaines instructions optionnelles. Cela conduit à concevoir 4 types de déroutement :

. Type 0 Pas de simulation.

. Type 1 Simulation de DIV et SHC.

. Type 2 Simulation des instructions flottantes FAD, FMU, FSU, FDV (traitement réentrant) et des instructions sur chaines de caractères (MVS, CPS, TRS).

. Type 3 Simulation de DIV et SHC et des instructions flottantes et des instructions sur chaines de caractères.

6-3. M : TRAP TYPE 0

6-3.1. Déroutement "programme" au niveau zéro

Le module M : TRAP de type 0 provoque :

■ L'édition sur la téléscriptrice (M : OC) du message de déroutement suivant :

. Mot d'état déroutement.

. P absolu
. L absolu
. Indicateurs

} sur l'instruction ayant provoqué le déroutement.

%% DR xx

xx = 01 si déroutement dû à un coupleur.

xx = 02 si déroutement programme.

■ L'ABORT du programme en cours

Le programme en cours est abandonné et retiré logiquement du système.

Mot d'état déroutement : les différents chiffres binaires de ce mot ont le sens suivant :

Bit 15 PG (1 déroutement sur programme, 0 déroutement dû à un coupleur),

Bit 14 ES (Horloge de garde E/S),

Bit 13 II (instruction inexistante),

Bit 12 PA (Parité mémoire),

Bit 11 AI (Adresse inexistante),

Bit 10 PM (Protection mémoire),

Bit 9 VM (Violation de mode).

Plusieurs bits peuvent être positionnés simultanément (en particulier AI et PA).

6-3-2. Déroutement "Coupleur" ou déroutement "programme" à un niveau différent de zéro.

Le système est considéré comme perturbé dans son ensemble et une réinitialisation générale est déclenchée:

- Réinitialisation du moyen moniteur (en particulier des tables d'E/S)

- Réinitialisation des contextes de tous les programmes immédiats (exceptés ceux des niveaux 30 et 31) et des handlers avec le premier élément de leur PRT. Tous les programmes immédiats devront donc être lancés par leur première section et tous les handlers devront avoir comme premier LPS le handler 2.

- Libération des handlers : chaque handler possède à l'adresse héxadécimale 16 de leur LDS un mot d'occupation destiné à les protéger contre les interruptions parasites ; ce mot est réinitialisé à zéro.

- Edition de :

%% DR xxxx
SYS READY
où xxxx est en héxadécimal, le mot d'état déroutement.

- Mise en attente du système (attente d'interruption).

Remarque :

Tous les MOB n'ont pas cette protection, un MOB ne possédant pas cette protection devra alors être rechargé.

6-4. LES AUTRES TYPES DU MODULES M : TRAP

Les autres types de module M : TRAP analysent le type de déroutement pour déterminer s'il est dû à l'absence du hardware correspondant à une instruction dont il simule l'exécution.

Pour les instructions simulées, le déroutement est transparent pour l'utilisateur.

Dans tous les autres cas, le traitement est identique au traitement offert par le module M : TRAP de type0.

Notes :

1 - L'utilisation de la bibliothèque mathématique virgule fixe, implique l'utilisation des instructions DIV et SHC.

L'utilisation de la bibliothèque mathématique virgule flottante, implique l'utilisation des instructions flottantes et de SHC.

2 - L'exécution par déroutement des instructions flottantes, nécessite l'existence dans le programme utilisateur, d'une TWB de 32 mots au lieu des 16 mots standard.

3 - Si un déroutement se produit au cours de la simulation d'une instruction inexistante, il y aura édition du message de déroutement et abandon du programme ayant utilisé cette instruction. Les informations éditées concerneront l'instruction simulée et non l'instruction déroutante. En effet, seule l'instruction simulée interesse alors l'utilisateur.

Rappelons que le traitement des instructions flottantes est réentrant.

4 - Si un déroutement se produit au cours du traitement du message de déroutement, la machine sera alors arrêtée après affichage de :

- Voyants adresse : &00FF
- Voyants donnée : Mot d'état du deuxième déroutement.

5 - Le CSV de la section 0 est autorisé par le hardware. Le résultat sera le même que celui d'un déroutement "programme" mais le mot d'état déroutement sera nul.

7. Communications avec l'opérateur (M : OC)

7-1. COMMANDES OPERATEUR

■ Principe

Le module d'analyse des commandes est appelé par le module traitant l'interruption pupitre et travaille donc au niveau de celle-ci.

Lorsqu'une interruption pupitre est prise en compte, MOB édite sur M : OC (télescriptrice) le caractère % et connecte la téléscriptrice en attente d'une commande. Sur l'entrée du caractère "retour-chariot", la commande est exécutée.

Les commandes opérateur du MOB sont % LOAD (chargement d'un IMT) et % RUN (lancement du dernier programme chargé). Aucune simultanéité n'est donc nécessaire entre l'entrée des commandes et le niveau zéro. L'interruption pupitre aborte donc le niveau zéro.

Sous MOB-E, le jeu des commandes opérateur comportant des commandes de mise au point il est nécessaire d'assurer la simultanéité de fonctionnement entre l'entrée et le traitement des commandes et le niveau zéro. Ce dernier ne sera donc pas aborté par l'interruption pupitre. Une commande spécialisée (%Y) permettra d'aborter le niveau zéro. (cf chapitre 9).

Le caractère "←" suivi de "retour-chariot" annule la commande en cours d'entrée.

■ Normes utilisées pour les commandes

D'une manière générale, seule la première lettre d'une commande est utilisée pour la reconnaître.

- Une commande commence par %.

- Une commande est terminée sur le premier blanc (espace) rencontré. Tout caractère présent dans la suite de l'enregistrement est considéré comme commentaire.

- Le caractère "/" sépare des niveaux hiérarchisés de paramètres et en particulier le nom de la commande de ses options.

- Le caractère ":" a deux utilisations :

1 - Séparateur : il a le sens général d'affectation numérique. Les éléments qu'il sépare ne sont donc pas syntaxiquement permutables. Il est utilisé dans ce sens dans les commandes SNAP et MODIFY (voir chapitre 9).

2 - Caractère alphabétique : il figure à ce titre dans les éléments du genre étiquette opérationnelle, type de périphérique, etc... sous la forme :

M : xx T : yy C : zz D : t

Exemples : T : PR M : BI etc...

- Le caractère "." a le sens général d'affectation symbolique. Les éléments qu'il sépare ne sont donc pas permutables.

- Le caractère "," sépare deux paramètres ou groupes de paramètres dont l'ordre peut être quelconque.

- Le caractère ";" a le sens d'un caractère suite et figure, quand il est utilisé, comme dernier caractère d'une commande. Dans MOB, un point-virgule comme dernier caractère d'une commande reconnecte la téléscriptrice en entrée pour une nouvelle commande après l'exécution de la précédente sans qu'il soit besoin de refaire une interruption pupitre.

- Le caractère "@" indique que ce qui suit est une adresse.
- Le caractère "&" est le premier caractère d'un nombre hexadécimal.
- Les caractères "+" et "-" ont respectivement le sens d'addition et de soustraction. Les éléments qu'ils séparent ne sont pas permutables.
- Les accolades { } indiquent que le terme inclus doit être présent.
- Les crochets [ ] indiquent que l'un des termes superposés doit être présent.
- Les points de suspension "..." indiquent que le dernier élément spécifié peut être répété.
- Deux termes superposés s'excluent mutuellement.
- La présence des séparateurs "/", ":", ".", "," est liée à celle des éléments qu'ils séparent.

Exemple : une commande étant spécifiée comme suit :

$$\left\{\begin{array}{l}\%\ \text{LOAD}\\ \%\ \text{L}\end{array}\right\} \quad / \quad \left[\ @\ [\&]\ \text{xxxx}\ \right], \quad [\text{S}]$$

Les écritures suivantes sont correctes :

```
% LOAD
% L
% L/S
% L/@ xxxx              Adresse décimale
% LOAD/@ xxxx, S        Adresse décimale
% L/S, @ & xxxx         Adresse hexadécimale.
```

% LOAD Commande de chargement

. Définition :

Commande assurant le chargement en mémoire d'un module de programme ou de données au format IMT sur M : BI.

. Forme :

$$\left\{\begin{array}{l}\%\ \text{LOAD}\\ \%\ \text{L}\end{array}\right\} \quad / \quad \left[\ @\ [\&]\ \text{xxxx}\ \right], \quad [\text{S}]$$

. Sens des options :

[@[&] xxxx] - Quand cette option est présente, le chargement est effectué à partir de l'adresse spécifiée exprimée relativement à la fin de la zone moniteur.

- Quand cette option est absente, le chargement est effectué à partir de la première adresse libre après le moniteur.

[S] - Edition en fin de chargement sur M : OC du contenu de la PRT du programme sous la forme suivante :

```
xxxx } L - G }
xxxx } P - G } Section n
                :
xxxx } L - G }
xxxx } P - G } Section 1
```

. Effet en cas de chargement d'un module de programme

Le chargeur (loader) charge l'IMT du programme y compris les 16 octets du bloc fin (comprenant le contexte) et laisse après chargement, en G + 6, l'adresse relative à G de la zone commune. Il édite ensuite sur M : OC (téléscriptrice) les informations suivantes :

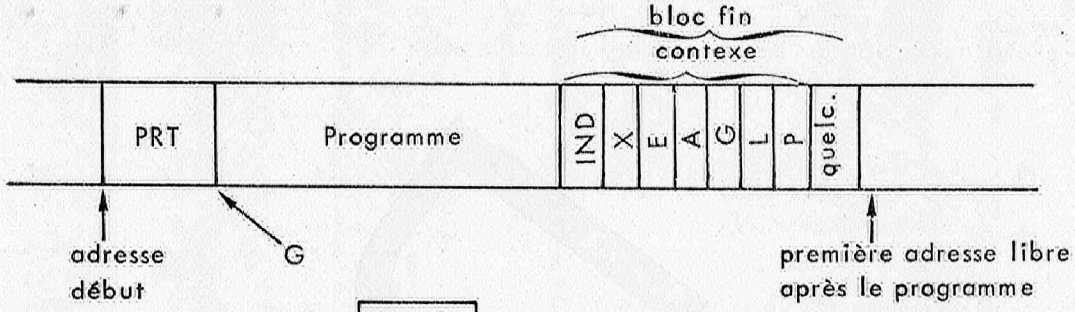
xxxx
xxxx
xxxx
xxxx
} SRD_n ... SRD_1 — PRT si l'option "S" est présente.

Les adresses éditées sont relatives à G.

yyyy — Adresse début d'implantation (absolue).

yyyy — Valeur de la base G (absolue).

yyyy — Première adresse libre après le programme (absolue).



. Rappel un SRD est composé de 2 mots

L - G
P - G

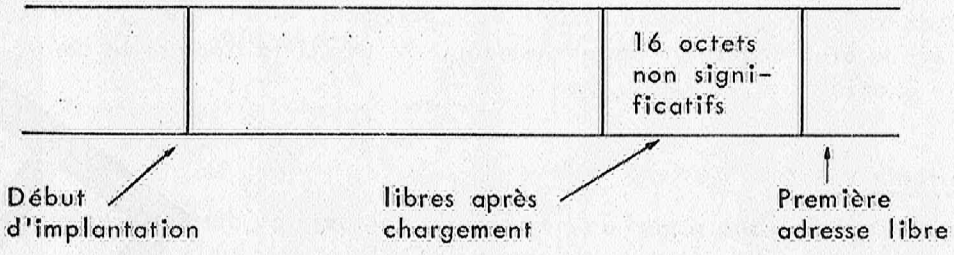
. Effet en cas de chargement d'un module de données

Le chargeur (loader) charge le bloc de données IMT y compris 16 octets non significatifs. Après chargement, il édite sur M : OC (téléscriptrice) :

xxxx — Adresse début d'implantation (absolue).

xxxx — Adresse début d'implantation (absolue).

xxxx — Première adresse libre (absolue).



Rappelons qu'un module de données est obtenu par édition de liens avec l'option DA.
L'adresse début d'implantation est éditée deux fois, la base G n'ayant pas de sens pour un module de données.

. Erreurs au chargement

Le moniteur édite sur M : OC (téléscriptrice) les informations suivantes :

%% LD01 Erreur d'E/S au chargement.

%% LD02 Parité (ou checksum) incorrecte au chargement.

%% LD03 Séquencement incorrecte au chargement.

%% AC02 Tentative de chargement sur le moniteur ou sur la zone commune.
Le module chargé n'est pas au format IMT.

Remarques :

1 - Le chargement de blocs de données n'affecte pas la prise en compte du dernier programme chargé qui reste donc le programme qui sera lancé par % RUN.

2 - Le tampon d'entrée binaire est un bloc de 120 octets situé en début de zone commune.

% RUN Commande de lancement

. Définition :

Commande lançant l'exécution du dernier programme chargé.

. Forme :

{ % RUN }
{ % R }

. Effet :

Chargement des registres avec les valeurs suivantes :

A Bits 0 à 7 : Niveau de la tâche appelante (ici, niveau de l'interruption pupitre).
Bits 8 à 15 : Niveau de la tâche lancée (ici, niveau zéro).

E Chargé avec zéro.

X Chargé avec zéro (représente l'adresse relative au début de la zone commune des tampons utilisables).

P, L Valeurs des bases de programme et de données locales courantes du dernier programme chargé.

Le moniteur utilise pour ce faire les 16 octets dits "de contexte" réservés à la fin du programme. Seule une commande LOAD concernant un nouveau module de programme, ou bien une erreur irrécupérable (déroutement en particulier) ou bien encore un appel au module M : EXIT fait perdre le contrôle du niveau zéro au dernier programme chargé.

. Remarque importante :

Après un RUN le système ne dispose plus en mémoire du contexte du dernier programme chargé, un nouveau RUN de ce programme n'est donc pas possible sans un LOAD au préalable.

. Rappel

Dans un programme écrit en assembleur, la section de lancement est spécifiée dans la directive END. Dans un programme écrit en LP15, il s'agit de la MAIN SECTION. Une option est de toute façon prise par défaut par l'éditeur de liens.

7-2. MESSAGES OPERATEUR

Les messages suivants sont édités sur M : OC (téléscriptrice) :

Messages	Signification
Au chargement du système	
SYSTEM READY	Le système vient d'être chargé. Il est en attente d'une interruption pupitre.
Concernant les commandes	
%	Attente commande opérateur.
%% AC01	Erreur de syntaxe dans une commande.
%% AC02	Exécution impossible d'une commande.
Au chargement	
%% LD01	Erreur d'E/S au chargement.
%% LD02	Parité ou checksum (somme de contrôle) incorrecte au chargement.
%% LD03	Séquencement incorrecte au chargement.
Erreurs d'entrée-sortie	
%% IO01	Erreur logique détectée en fin de transfert.
%% IO02	Erreur logique détectée à l'initialisation d'un transfert.
%% IO03	Erreur physique détectée en fin de transfert.
%% IO04	Erreur physique détectée à l'initialisation du transfert.
Déroutements	
xxxx xxxx xxxx xxxx xxxx xxxx xxxx %% DRnn	Mot détat déroutement. P absolu } sur l'instruction ayant L absolu } provoqué le déroutement. Indicateurs P - G } sur le dernier appel L - G } au moniteur Indicateurs nn = 01 déroutement "coupleur" nn = 02 déroutement "programme".
Incidents graves	
%% CS SYS READY	Coupure secteur ayant entraîné une réinitialisation du système (cf chapitre 5).
%% xxxx SYS READY	Déroutement ayant entraîné une réinitialisation du système (cf chapitre 6).

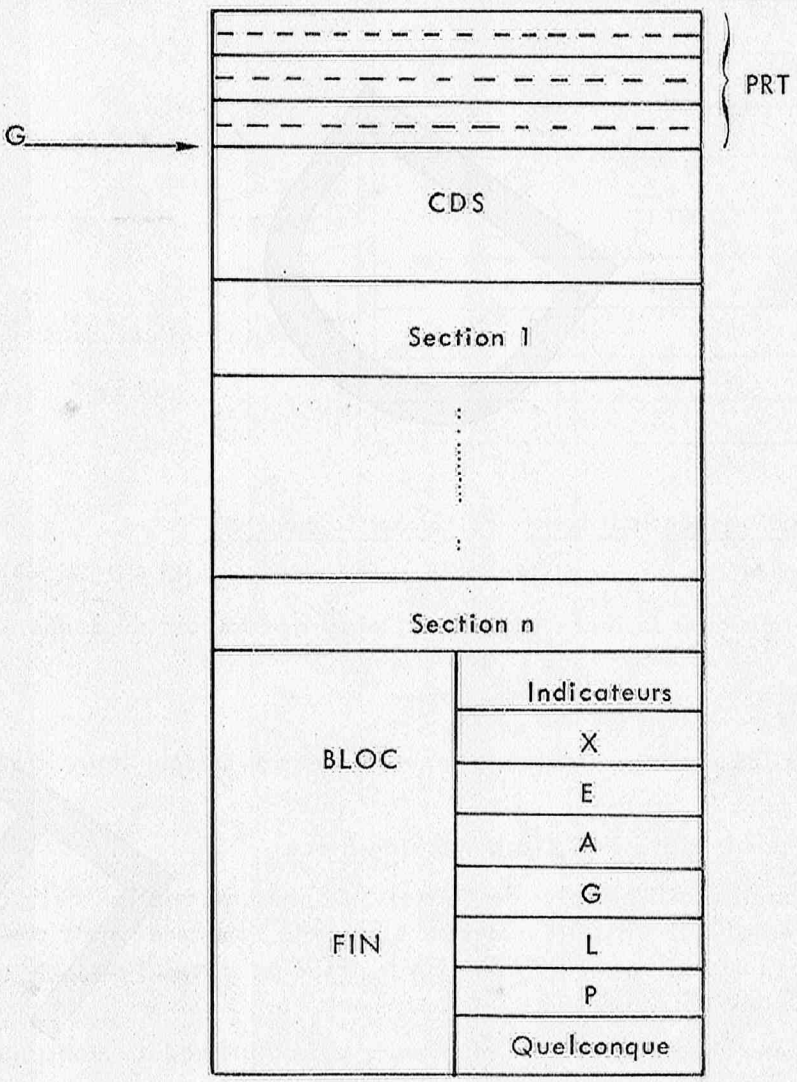
8. Utilisation des modules du moniteur

8-1. PRINCIPES GENERAUX DES APPELS MONITEUR

■ Structure d'un programme

Un programme en mémoire se compose de :

- Sa PRT,
- Une section de données communes (CDS),
- Des sections (LDS-LPS),
- Une zone fin de programme de 16 octets contenant le contexte (14 octets).



Description du TWB (Task Working Block)

	G							
PMG	+ &00	+ 0	(P) - (G)					
LMG	+ &02	+ 2	(L) - (G)					
IND	+ &04	+ 4		PR	MA	MS	C	O
AZC	+ &06	+ 6						
T0	+ &08	+ 8						
T1	+ &0A	+ 10						
T2	+ &0C	+ 12						
T3	+ &0E	+ 14						
T4	+ &10	+ 16						
T5	+ &12	+ 18						
T6	+ &14	+ 20						
T7	+ &16	+ 22						
N3	+ &18	+ 24						
N2	+ &1A	+ 26						
N1	+ &1C	+ 28						
N0	+ &1E	+ 30						
T8	+ &20	+ 32						
T9	+ &22	+ 34						
Tn								

utilisés par CSV et RSV

En cas d'extension de du TWB

Utilisation de la section de données communes (CDS) par le moniteur

Le moniteur peut utiliser dans le cas standard, les seize premiers mots de la CDS (TWB) :

- pour ranger l'adresse de retour dans la tâche appelante, ainsi que la base de données locales de l'appelant et les indicateurs,

- comme mémoires de travail.

Les programmes qui font appel au moniteur, doivent normalement commencer leur CDS par une réservation de seize mots.

Ces seize mots sont désignés par le sigle TWB (Task Working Block).

Il est à noter que certains modules utilisent plus de 16 mots. Dans la description des modules, le nombre de mots mémoire utilisé est indiqué. L'utilisateur devra réserver le nombre de mots correspondant . De plus, l'utilisation du traitement des instructions FAD, FSV, FMU, FDV par simulation (module n : TRAP de type 2 ou 3), nécessite un TWB étendu à 32 mots.

Cette procédure permet la réentrance des modules moniteur, puisqu'un module moniteur travaille toujours dans une zone appartenant au programme appelant.

Procédure d'appel d'un module du moniteur MOB

L'utilisateur doit d'abord mettre en place les paramètres de l'appel, puis effectuer l'appel au superviseur.

CSV M : <nom du module>

Remarque importante :

D'une manière générale, les registres sont modifiés, à moins que le contraire ne soit explicitement spécifié.

Exemple :

Appel d'entrée-sortie :

a - Constitution du CB (Control Block).

b - LEA CB (adresse du CB par rapport à G dans A)

C - CSV M : IO (M : IO interface d'E/S).

■ Sous-programmes utilisables par appel CSV

M : IO	Appel d'entrée-sortie.
M : WAIT	Attente fin de transfert.
M : EXIT	Fin anormale d'un programme.
M : KEY	Lecture des clés Ecriture des voyants du pupitre.
M : DCBN	Conversion décimal-binaire.
M : HXBN	Conversion hexadécimal-binaire.
M : BNDC	Conversion binaire-décimal.
M : BNHX	Conversion binaire-hexadécimal.
M : ASEB	Conversion ASCII - EBCDIC.
M : EBAS	Conversion EBCDIC - ASCII.
M : LOAD	Chargement en mémoire d'un module IMT (TWB étendu).
M : MOVE	Déplacement d'une chaîne d'octets.
M : EDIT	Edition d'une zone mémoire en hexadécimal (TWB étendu).
M : CMPA	Comparaison arithmétique de deux mots de 16 bits.

8-2. M : IO APPEL D'ENTREE-SORTIE

. Séquence d'appel :

```
LEA     CB
CSV     M : IO
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du bloc de commande d'entrée-sortie (Control Block). Un CB doit toujours avoir une adresse-mot (adresse paire).

. Fonction :

Exécution des demandes d'entrée/sortie (voir organisation des Entrées/Sorties sous contrôle du MOB).

• Description du CB (Bloc de commande) :

Le CB peut comporter les 8 à 12 octets suivants :

	0	Octet évènement
	1	Indicateurs
	2	Ordre
	3	Etiquette opérationnelle
	4	Adresse du buffer (*)
	5	relative à G
	6	Nombre d'octets
	7	à transférer
Optionnel	8	Adresse de branchement sur (*)
	9	erreur ou fin anormale (traitement standard)
Optionnel	10	Information supplémentaire
	11	(adresse disque par exemple)

(*) En mode asservi, la référence indiquée sera toujours relative à G.

En mode maître, où les références sont normalement rendues absolues au chargement, cette référence devra être précédée à l'assemblage, du caractère " # " pour en maintenir la translatabilité (voir manuel MITRA 15 de référence). A la compilation LP15, elle devra être précédée du caractère "." .

- Octet 0 : octet évènement

Bit 0	= 1	Un transfert associé à un M : IO utilisant ce CB est en cours. L'évènement (fin de transfert) est désactivé.
	= 0	Le transfert est terminé. L'évènement (fin de transfert) est activé.
Bit 1	= 1	Erreur ou fin anormale.
Bit 2	= 0	Erreur logique (par exemple : CB incorrect).
	= 1	Erreur physique (par exemple : signalée par le coupleur).
Bit 3	= 0	Erreur fin de transfert.
	= 1	Erreur à l'initialisation du transfert.

Bits 4, 5, 6 et 7 : Bits d'erreur ou de fin anormale positionnés par le handler concerné et spécifiques de celui-ci (voir Entrées/Sorties, manuel de référence MITRA 15).

Les bits 2 et 3 n'ont de sens que si le bit 1 est à 1.

- Octet 1 : type de contrôle

0	1	2	3	4	5	6	7
U	E	S					

U = 1 : contrôle total du transfert par l'utilisateur : aucune erreur n'est traitée par le moniteur.

E = 1 : adresse de branchement en cas d'erreur ou de fin anormale (voir M : WAIT).

S = 1 : présence d'une information supplémentaire (adresse disque par exemple).

- Octet 2 : ordre

Il définit l'opération d'entrée/sortie demandée. Les fonctions standard sont les suivantes :

Contenu hexadécimal	Ordre
00	Lecture avec filtrage
01	Lecture sans filtrage
10	Lecture avec format, avec filtrage
11	Lecture avec format, sans filtrage
20	Lecture arrière
30	Rebobinage on-line
40	Saut avant de n blocs
50	Saut avant de n fichiers
60	Saut arrière de n blocs
70	Saut arrière de n fichiers
80	Ecriture
90	Ecriture avec format
A0	Ecriture de tape-mark
B0	Avance
C0	Non affecté
D0	Positionnement de bras
E0	Non affecté
F0	Rebobinage off-line.

Leur exploitation précise par chaque handler figure dans le manuel de référence, chapitre "traitement software des entrées-sorties".

- Octet 3 : étiquette opérationnelle

Equivalent numérique de l'étiquette opérationnelle indiquée à l'assemblage.

Exemple :

M : BO équivaut à 2.

Si l'on utilise MITRAS 1, il est nécessaire d'indiquer directement l'équivalent numérique donné dans le tableau suivant.

Etiquettes opérationnelles MOB

Numéros à l'assemblage (décimal)	Etiquettes opérationnelles	Fonction	Mode
1	BI	Entrée binaire	binaire
2	BO	Sortie binaire	binaire
3	CI	Entrée commande	alphanumérique
4	OC	Organe de commande	alphanumérique
5	EI	Entrée d'éléments	binaire ou alphanumérique

Numéros à l'assemblage (décimal)	Etiquettes opérationnelles	Fonction	Mode
6	EO	Sortie d'éléments	binaire ou alphanumérique
7	LO	Sortie de liste	alphanumérique
8	LL	Sortie d'implantation	alphanumérique
9	DO	Sortie de diagnostic	alphanumérique
10	SI	Entrée source	alphanumérique

L'affectation à une étiquette opérationnelle alphanumérique, assure toute conversion de code nécessaire entre le code interne (EBCDIC) et le code utilisé par le périphérique.

Les étiquettes opérationnelles correspondent à des handlers. On dit qu'elles sont assignées au périphérique correspondant. Les assignations sont fixées à la génération et sous MOB-E modifiables par sa commande % ASSIGN. Les assignations standard des moniteurs sont indiquées dans les fiches opérateur correspondantes.

- Octets 4 et 5 : Adresse tampon

Adresse-octet de l'information à transférer de/vers la mémoire vive.

- Octets 6 et 7 : Longueur

Longueur en octets de l'information à transmettre.

- Octets 8 et 9 : Adresse de reprise

Cette information est significative si E = 1.

Ces octets indiquent l'adresse de branchement en cas d'erreur ou de fin anormale, indiquée par l'utilisateur.

C'est le module M : WAIT qui, le cas échéant, effectue le branchement.

- Octets 10 et 11 : Information supplémentaire

Cette information est significative si S = 1.

Ces octets indiquent, dans le cas du disque, l'adresse disque utilisée (en nombre de secteurs à partir de 1).

. Exemple de sortie sur M : LO

```
EXEC    CDS
        RES     16
CB      DATA    0
        DATA, 1 &80
        DATA, 1 M : LO
        DATA    # BUFF
        DATA    13
BUFF    TEXT    "SORTIE SUR LO"
        FIN

LOC     LDS
        RES     2
```

```
        FIN
PROG    LPS     LOC
DEB     LEA     CB
        CSV     M : IO
        CSV     M : WAIT
        CSV     M : EXIT
        FIN     DEB

        END     PROG
```

. Eléments communiqués en sortie

Le registre A contient l'adresse relative à G du CB, ce qui permet facilement un appel au module M : WAIT après M : IO.

Les registres E et X sont quelconques.

L'octet évènement contient les informations suivantes :

. Erreur (ou non) de transfert,

. Code d'erreur le cas échéant.

. Messages d'erreur (cas du bit U de l'octet 1 à zéro)

%% 1001 Erreur logique décelée en fin de transfert.

%% 1002 Erreur logique décelée à l'initialisation du transfert.

%% 1003 Erreur physique décelée en fin de transfert.

%% 1004 Erreur physique décelée à l'initialisation du transfert.

. Mémoires de TWB utilisées : T0 à N0

. Modules appelés : M : EBAS, Handlers 1, Handlers 2.

8-3. M : WAIT ATTENTE DE FIN DE TRANSFERT

. Séquence d'appel

```
LEA     CB
CSV     M : WAIT
```

Au moment de l'appel du CSV, le registre A doit contenir l'adresse relative à G du bloc de commande de l'entrée-sortie que l'on veut contrôler.

. Fonction

Mise en attente de fin d'entrée-sortie.

Si le transfert est terminé au moment de l'appel de M : WAIT, celui-ci rend le contrôle à l'utilisateur (éventuellement à une adresse de reprise en cas d'anomalie, si cela est demandé par l'octet 1 du CB).

Si le transfert n'est pas terminé au moment de l'appel de M : WAIT, celui-ci :

- range le niveau de la tâche appelante dans l'octet zéro du CB d'Entrée/Sortie,
- Désactive ce niveau s'il est non nul, se met en attente de fin de transfert s'il est nul.

C'est donc le module M : WAIT qui assure un fonctionnement évolué du MOB dans un environnement temps réel. En effet, comme il désactive provisoirement le niveau d'un programme en attente d'entrée-sortie, il permet la prise en compte, sans blocage, de programmes moins prioritaires. Ce programme provisoirement suspendu sera repris au moment de la fin de transfert, son niveau étant alors réactivé par le handler 2 du coupleur concerné par l'entrée-sortie.

. Eléments communiqués en sortie

A < 0 en cas d'erreur d'entrée-sortie.

Dans ce cas, l'utilisateur doit analyser la cause de l'erreur indiquée dans l'octet 0 du CB.

A ⩾0 aucune erreur.

. Mémoires de TWB utilisées : T0 à N0

8-4. M : EXIT FIN DE PROGRAMME

. Séquence d'appel

```
CSV     M : EXIT
```

. Fonction

Fin de programme.

. Si le programme est au niveau zéro, le contrôle est rendu au moniteur qui se met en attente d'interruption pupitre (boucle d'attente BRU $ au niveau 0).

. Si le programme n'est pas au niveau zéro, son niveau est désactivé, ce qui permet de prendre en compte tout niveau inférieur.

. Mémoires de TWB utilisées : Néant.

8-5. M : KEY CLES ET VOYANTS

. Séquence d'appel

LDX = { 0 Lecture de clés
1 Ecriture des voyants de donnée
2 Ecriture des voyants adresse

```
CSV     M : KEY
```

. Fonction

Ecrire le contenu du registre A sur les voyants "données" du pupitre (LDX = 1) ou sur les voyants "adresse" du pupitre (LDX = 2) ou lire les clés du pupitre pour les ranger dans le registre A (LDX = 0).

. Eléments communiqués en sortie

En lecture, le registre A contient la valeur indiquée par les clés du pupitre.

. Mémoires de TWB utilisées : Néant.

8-6. M : DCBN CONVERSION DECIMAL-BINAIRE

. Séquence d'appel

```
LEA     CHAINE
CSV     M : DCBN
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne décimale à convertir.

. Fonction

Conversion en binaire sur un mot de 16 bits d'un nombre décimal représenté par une chaîne de caractères EBCDIC, terminée par un séparateur : caractère autre qu'un chiffre décimal (0 à 9). Le nombre à convertir doit être tel que $0 \leqslant n \leqslant 65535$.

. Eléments communiqués en sortie

E et T1 Résultat de la conversion (sur un mot).

X et T0 Adresse relative à G du séparateur de fin de chaîne (sur un mot).

A { $\geqslant 0$ Nombre de caractères de la chaîne, s'il n'y a pas eu de débordement.
{ < 0 S'il y a eu débordement.

. Mémoires de TWB utilisées : T0 à N0

8-7. M : HXBN CONVERSION HEXADECIMAL-BINAIRE

. Séquence d'appel

```
LEA     CHAINE
CSV     M : HXBN
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne hexadécimale à convertir.

. Fonction

Convertir en binaire (sur un mot de 16 bits) un nombre héxadécimal représenté par une chaîne de caractères EBCDIC terminée par un séparateur : caractère autre qu'un chiffre hexadécimal (0 à 9 et A à F). Le nombre à convertir doit être tel que :

$\&0 \leqslant N \leqslant \&FFFF$.

. Eléments communiqués en sortie

E et T1 Résultat de la conversion (sur un mot).

X et T0 Adresse relative à G du séparateur de fin de chaîne (sur un mot).

A { $\geqslant 0$ Nombre de caractères de la chaîne s'il n'y a pas eu débordement.
{ < 0 S"il y a eu débordement..

. Mémoires de TWB utilisées : T0 à N0

8-8. M : BNDC CONVERSION BINAIRE-DECIMAL

. Séquence d'appel

```
LDE     NOMBRE
LEA     CHAINE
CSV     M : BNDC
```

Au moment de l'exécution du CSV, le registre A doit contenir, l'adresse relative à G du premier octet de la chaîne résultat et le registre E doit contenir le nombre à convertir.

. Fonction

Convertir un nombre binaire positif de 15 bits (&0 ⩽ n ⩽ & 7FFF) en une chaîne de chiffres décimaux sous forme de cinq caractères EBCDIC. Ces caractères sont cadrés à droite avec suppression des zéros non significatifs de gauche.

. Eléments communiqués en sortie

A E et X quelconques.

T0 Adresse relative à G du début de la chaîne.

. Mémoires de TWB utilisées : T0 à N0.

8-9. M : BNHX CONVERSION BINAIRE-HEXADECIMAL

. Séquence d'appel

```
LDE     NOMBRE
LEA     CHAINE
CSV     M : BNHX
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne résultat et le registre E doit contenir le nombre à convertir.

. Fonction

Convertir un nombre binaire de 16 bits (&0 ⩽ n ⩽ & FFFF) en une chaîne de chiffres hexadécimaux sous forme de quatre caractères EBCDIC.

. Eléments communiqués en sortie

A E et X quelconques.

T0 Adresse relative à G de la chaîne.

. Mémoires de TWB utilisées : T0 à N0.

8-10. M : ASEB CONVERSION ASCII-EBCDIC

. Séquence d'appel

```
LDX     = Nombre de caractères de la chaîne.
LEA     CHAINE
CSV     M : ASEB
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne à convertir et le registre X doit contenir le nombre de caractères de la chaîne.

. Fonction

Conversion d'une chaîne codée en ASCII (ISO à 7 éléments) en une chaîne codée en EBCDIC (ISO à 8 éléments). Les caractères convertis remplacent un à un les caractères initiaux. La valeur du bit zéro des caractères ASCII est ignorée.

La table de conversion ASCII-EBCDIC est donnée à la page suivante.

. Mémoires de TWB utilisées : T0 à N0

Code ASCII (hexadécimal)	Code EBCDIC correspondant (hexadécimal)	Code ASCII (hexadécimal)	Code EBCDIC correspondant (hexadécimal)	Code ASCII (hexadécimal)	Code EBCDIC correspondant (hexadécimal)
00 ou 00	00	30 30	F0	60 60	CA
01 81	01	31 B1	F1	61 E1	CB
02 82	02	32 B2	F2	62 E2	CC
03 03	03	33 33	F3	63 63	CD
04 84	37	34 34	F4	64 E4	CE
05 05	2D	35 35	F5	65 65	CF
06 06	2E	36 36	F6	66 66	86
07 87	2F	37 B7	F7	67 E7	87
08 88	16	38 B8	F8	68 E8	88
09 09	05	39 39	F9	69 69	89
0A 0A	15	3A 3A	7A	6A 6A	91
0B 0B	0B	3B 3B	5E	6B EB	92
0C 0C	0C	3C 3C	4C	6C 6C	93
0D 8D	0D	3D BD	7E	6D ED	94
0E 8E	0E	3E 8E	6E	6E EE	95
0F 0F	0F	3F 3F	6F	6F 6F	96
10 90	10	40 C0	7C	70 F0	97
11 11	11	41 41	C1	71 71	98
12 12	12	42 42	C2	72 72	99
13 13	13	43 43	C3	73 F3	A2
14 14	3C	44 44	C4	74 74	A3
15 95	3D	45 C5	C5	75 F5	A4
16 96	32	46 C6	C6	76 F6	A5
17 17	26	47 47	C7	77 77	A6
18 18	18	48 48	C8	78 78	A7
19 99	19	49 C9	C9	79 F9	A8
1A 9A	3F	4A CA	D1	7A FA	A9
1B 1B	27	4B 4B	D2	7B 7B	C0
1C 9C	1C	4C CC	D3	7C FC	6A
1D 1D	1D	4D 4D	D4	7D 7D	D0
1E 1E	1E	4E 4E	D5	7E 7E	A1
1F 9F	1F	4F CF	D6	7F FF	07
20 A0	40	50 50	D7		
21 21	4F	51 D1	D8		
22 22	7F	52 D2	D9		
23 A3	7B	53 53	E2		
24 24	5B	54 D4	E3		
25 A5	6C	55 55	E4		
26 A6	50	56 56	E5		
27 27	7D	57 D7	E6		
28 28	4D	58 D8	E7		
29 A9	5D	59 59	E8		
2A AA	5C	5A 5A	E9		
2B 2B	6E	5B DB	4A		
2C AC	6B	5C 5C	E0		
2D 2D	60	5D DD	5A		
2E 2E	4B	5E DE	5F		
2F AF	61	5F 5F	6D		

8-11. M : EBAS CONVERSION EBCDIC-ASCII

. Séquence d'appel

LDX = Nombre de caractères de la chaîne

LEA CHAINE

CSV M : EBAS

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne à convertir et le registre X doit contenir le nombre de caractères de la chaîne.

. Fonction

Conversion d'une chaîne codée en EBCDIC (ISO à 8 éléments), en une chaîne codée en ASCII (ISO à 7 éléments). Les caractères convertis remplacent un à un les caractères initiaux. La parité des caractères ASCII obtenue est paire; un caractère EBCDIC n'ayant pas de correspondant en ASCII, est remplacé par un blanc. La table de conversion utilisée est donnée à la page suivante.

Mémoires de TWB utilisées : T0 à N0.

Code EBCDIC (hexa)	Code* ASCII (hexa)	Code EBCDIC (hexa)	Code* ASCII (hexa)	Code EBCDIC (hexa)	Code* ASCII (hexa)	Code EBCDIC (hexa)	Code* ASCII (hexa)	Code EBCDIC (hexa)	Code* ASCII (hexa)	Code EBCDIC (hexa)	Code* ASCII (hexa)
00	00	30	A0	60	2D	90	AD	C0	7B	F0	30
01	81	31	A0	61	AF	91	6A	C1	41	F1	B1
02	82	32	96	62	A0	92	EB	C2	42	F2	B2
03	03	33	A0	63	A0	93	6C	C3	C3	F3	33
04	A0	34	A0	64	A0	94	ED	C4	44	F4	B4
05	09	35	A0	65	A0	95	EE	C5	C5	F5	35
06	A0	36	A0	66	A0	96	6F	C6	C6	F6	36
07	FF	37	84	67	A0	97	F0	C7	47	F7	B7
08	A0	38	A0	68	A0	98	71	C8	48	F8	B8
09	A0	39	A0	69	A0	99	72	C9	C9	F9	39
0A	A0	3A	A0	6A	FC	9A	A0	CA	60	FA	A0
0B	8B	3B	A0	6B	AC	9B	A0	CB	E1	FB	A0
0C	0C	3C	14	6C	A5	9C	A0	CC	E2	FC	A0
0D	8D	3D	95	6D	5F	9D	A0	CD	63	FD	A0
0E	8E	3E	A0	6E	BE	9E	A0	CE	E4	FE	A0
0F	0F	3F	9A	6F	3F	9F	A0	CF	65	FF	A0
10	90	40	A0	70	A0	A0	A0	D0	7D		
11	11	41	A0	71	A0	A1	7E	D1	CA		
12	12	42	A0	72	A0	A2	F3	D2	4B		
13	93	43	A0	73	A0	A3	74	D3	CC		
14	A0	44	A0	74	A0	A4	F5	D4	4D		
15	0A	45	A0	75	A0	A5	F6	D5	4E		
16	88	46	A0	76	A0	A6	77	D6	CF		
17	A0	47	A0	77	A0	A7	78	D7	50		
18	18	48	A0	78	A0	A8	F9	D8	D1		
19	99	49	A0	79	A0	A9	FA	D9	D2		
1A	A0	4A	DB	7A	3A	AA	A0	DA	A0		
1B	A0	4B	2E	7B	A3	AB	A0	DB	A0		
1C	9C	4C	3C	7C	C0	AC	A0	DC	A0		
1D	1D	4D	28	7D	27	AD	A0	DD	A0		
1E	1E	4E	2B	7E	BD	AE	A0	DE	A0		
1F	9F	4F	21	7F	22	AF	A0	DF	A0		
20	A0	50	A6	80	A0	B0	A0	E0	5C		
21	A0	51	A0	81	A0	B1	A0	E1	A0		
22	A0	52	A0	82	A0	B2	A0	E2	53		
23	A0	53	A0	83	A0	B3	A0	E3	D4		
24	A0	54	A0	84	A0	B4	A0	E4	55		
25	A0	55	A0	85	A0	B5	A0	E5	56		
26	17	56	A0	86	66	B6	A0	E6	D7		
27	1B	57	A0	87	E7	B7	A0	E7	D8		
28	A0	58	A0	88	E8	B8	A0	E8	59		
29	A0	59	A0	89	69	B9	A0	E9	5A		
2A	A0	5A	DD	8A	A0	BA	A0	EA	A0		
2B	A0	5B	24	8B	A0	BB	A0	EB	A0		
2C	A0	5C	AA	8C	A0	BC	A0	EC	A0		
2D	05	5D	A9	8D	A0	BD	A0	ED	A0		
2E	06	5E	BB	8E	A0	BE	A0	EE	A0		
2F	87	5F	DE	8F	A0	BF	A0	EF	A0		

* Code ASCII parité paire

8-12. M : LOAD CHARGEMENT D'UN MODULE IMT EN MEMOIRE

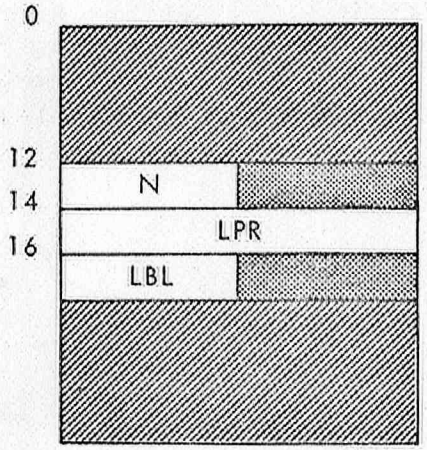
. Séquence d'appel

- Lecture de l'en-tête (1er enregistrement).
- Mise en place des paramètres dans le TWB.
- CSV M : LOAD.

. Fonction

- Chargement en mémoire d'un module IMT (format non disque) de programme (sans overlay) ou de données. La lecture et le dépouillement de l'en-tête de l'IMT est à la charge de l'utilisateur, une partie des éléments de l'en-tête étant utilisés pour fixer les éléments à communiquer en entrée pour M : LOAD.
- Edition sur M : OC de l'implantation et, si demandé, de la PRT du programme chargé (voir commande %LOAD)

. Informations de l'en-tête à exploiter



N = 1 programme / 3 données

LPR Longueur du programme y compris la PRT et le contexte.

LBL Longueur des enregistrements (120).

. Eléments à communiquer en entrée

Le module M : LOAD utilise un TWB étendu à 64 octets comme zone de travail et comme interface d'Entrée /Sortie. L'utilisateur qui utilise M : LOAD devra en tenir compte.

Les éléments du TWB à initialiser avant l'appel sont :

G + 20	T8	CB / INDI
G + 22	T9	Ordre / Etiquette
G + 24	T10	ADTAMP
G + 26	T11	LBL
G + 2C	T14	ADTAMP
G + 2E	T15	LBL
G + 30	T16	SORPRT
G + 32	T17	N
G + 38	T20	ADPR
G + 3A	T21	LPR

hexadécimal

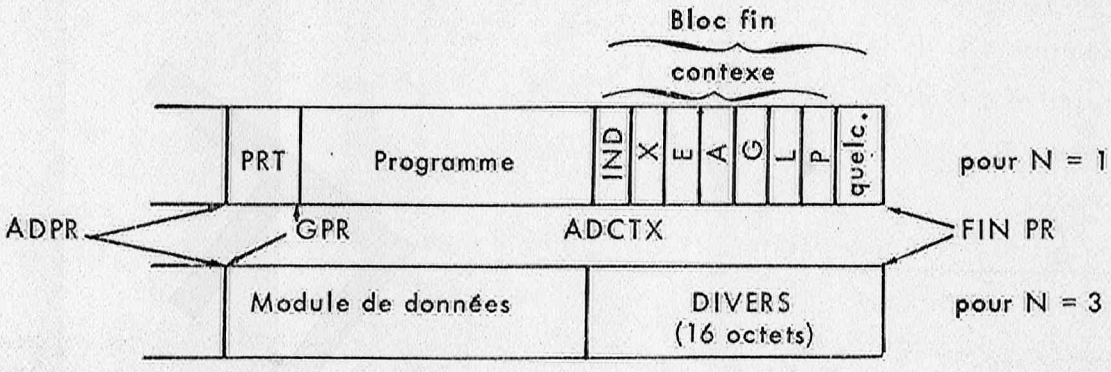
CB	= 0	Octet événement.
INDI	= 0	Format IMT non disque.
Ordre	= 0	Lecture.
Etiquette	=	Etiquette opérationnelle (normalement 1 pour M : Bi).
ADTAMP	=	Adresse du tampon de lecture par rapport au G de l'appelant.
LBL	= 120	Longueur enregistrement (à prendre dans l'en-tête).
SORPRT	= 1	Edition de la PRT.
	= 0	Pas d'édition de la PRT.
N	= 1	Programme.
	= 3	Données (à prendre dans l'en-tête).
ADPR	=	Adresse d'implantation du programme par rapport au G de l'appelant.
LPR	=	Longueur du programme en octets y compris la PRT et le contexte (à prendre dans l'en-tête).

. Eléments communiqués en sortie : Les éléments de TWB utilisés en sortie sont :

G + 38	T20	ADPR
G + 3A	T21	GPR
G + 3C	T22	FIN PR
G + 3E	T23	ADCTX

hexadécimal

ADPR	=	Adresse absolue d'implantation du programme (pour N = 1).
GPR	=	Base G du programme chargé (en absolu) (pour N = 1).
FIN PR	=	Première adresse libre après le module chargé (en absolu).
ADCTX	=	Adresse relative à la base G de l'appelant du contexte du programme chargé.



Accumulateur : A < 0 incident au cours du chargement.

Après chargement d'un module de programme l'utilisateur le connectera à un niveau défini par chargement avec ADCTX du pointeur de contexte correspondant (accessible par l'intermédiaire du pointeur de la table des pointeurs de contexte situé à l'adresse absolue 000A).

Il n'y a pas de contrôle de validité de l'implantation.

Le jeu normal des interruptions assurera le lancement du programme par le procédé classique d'échange de contextes.

Un programme ne peut charger un autre programme et le connecter au même niveau que lui-même.

. Editions sur M : OC

Ces éditions sont identiques à celles exécutées par % LOAD, soit :

PRT → absente pour un module de données.

ADPR

GPR → égale à ADPR pour un module de données.

FIN PR

. Mémoires de TWB utilisées :

TWB étendue à 64 octets, soit : T0 à T23.

. Modules appelés

M : MOVE, M : EDIT, M : IO, M : WAIT, M : ERED, M : BNHX.

8-13. M : MOVE DEPLACEMENT D'UNE CHAINE D'OCTETS

. Séquence d'appel

```
LEA     CHAIN2
XAX
LEA     CHAIN1
LDE     NOMBRE
CSV     M : MOVE
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G de la chaîne émettrice, le registre X doit contenir l'adresse relative à G de la chaîne réceptrice et le registre E le nombre d'octets à transférer.

. Fonction

Déplacement en mémoire d'une chaîne d'octets. Le module opère par adresses décroissantes.

. Eléments communiqués en sortie

A — Adresse relative à G de la chaîne émettrice.

E — Adresse relative à G de la chaîne réceptrice.

. Mémoires de TWB utilisées : T0 à N0.

8-14. M : EDIT EDITION D'UNE ZONE MEMOIRE

. Séquence d'appel

```
LEA     ZONE
LDE     NOMBRE
CSV     M : EDIT
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse par rapport à G de la zone à éditer et le registre E le nombre d'octets à éditer.

. Fonction

Editer en hexadécimal une zone mémoire. Cette édition est effectuée mot par mot sur M : OC à raison de 1 mot par ligne.

Une erreur d'entrée-sortie entraîne l'abandon du module.

. Mémoires de TWB utilisées :

TWB étendue à 56 octets, soit : T0 à T19.

. Modules appelés

M : MOVE, M : BNHX, M : IO, M : WAIT.

8-15. M : CMPA COMPARAISON ARITHMETIQUE DE DEUX MOTS DE 16 BITS

. Séquence d'appel

```
LDA     FIRST
LDE     SECOND
CSV     M : CMPA
```

Au moment du CSV, le registre A doit contenir le premier terme de la comparaison et le registre E le deuxième terme.

. Fonction

Comparaison arithmétique sur 16 bits des deux termes communiqués. La fonction du CMPA est distincte de celle de l'instruction CMP en ce sens que CMP considère les deux termes comme des nombres algébriques (15 bits de mantisse et un bit de signe) alors que CMPA considère les deux termes comme des nombres absolus (16 bits de mantisse). Ce module est utile lorsqu'on veut comparer des adresses dont l'une au moins peut être supérieure à 7FFF (hexadécimal).

. Eléments communiqués en sortie

A < 0	Premier terme < second terme
A = 0	Premier terme = second terme
A > 0	Premier terme > second terme

. Mémoires de travail utilisées : T0 à N0.

9. MOB. E : Communications opérateur Modules moniteurs

9-1. GENERALITES

Les besoins étant différents suivant que l'on désire exploiter un système, ou que l'on désire mettre au point de nouveaux programmes, pour ne pas pénaliser l'utilisateur disposant d'une mémoire réduite, les commandes ayant trait à la mise au point constituent une extension du MOB.

Pour simplifier la lecture, la description de l'ensemble des commandes sera reprise, ainsi que le tableau des messages opérateur.

En ce qui concerne les modules moniteur, seul sera spécifié ici le module supplémentaire M : DUMP.

Rappelons que l'extension de la zone commune du MOB-E est décrite au chapitre 3.

9-2. COMMUNICATIONS OPERATEUR - LES COMMANDES

Principe

Les commandes sont entrées sur l'étiquette opérationnelle M : OC.

Les commandes reconnues par MOB-E sont les suivantes :

% LOAD	Chargement d'un IMT
% RUN	Lancement du dernier programme chargé
% ASSIGN	Assignation des étiquettes opérationnelles
% DISPLAY	Edition des informations système
% DUMP	Edition de mémoire (post mortem)
% Y	Abandon du niveau zéro
% SNAP	Edition de mémoire (en cours de programme)
% TRACE	Edition des registres (en cours de programme)
% MODIFY	Modification mémoire
% HALT	Arrêt sur instruction
% NEXT	Exécution d'un pas
% EXECUTE	Reprise du programme utilisateur
% PERFORM	Calcul en hexadécimal.

Les commandes sont entrées sur l'étiquette opérationnelle M : OC (téléscriptrice) après prise en compte d'une interruption pupitre.

Lorsqu'une interruption pupitre est prise en compte, MOB-E édite sur M : OC le caractère % et connecte la téléscriptrice en entrée d'une commande. Sur l'entrée de "retour-chariot", la commande est analysée et exécutée au niveau de l'interruption pupitre. Il n'y a pas cependant de blocage de niveau zéro (excepté pour les commandes HALT et NEXT). L'impression des DUMP s'effectue au niveau zéro, ce qui rend ceux-ci interruptibles par l'interruption pupitre. Une grande souplesse d'utilisation est ainsi assurée.

Le caractère "←" suivi de "retour-chariot" annule la commande en cours d'entrée.

Normes utilisées pour les commandes

D'une manière générale, les deux premières lettres d'une commande sont utilisées pour la reconnaître (excepté pour les commandes LOAD, RUN et Y où seul le premier caractère est utilisé pour ce faire).

- Une commande commence par %.

- Une commande est terminée sur le premier blanc (espace) rencontré. Tout caractère présent dans la suite de l'enregistrement est considéré comme commentaire.

- Le caractère "/" sépare des niveaux hiérarchisés de paramètres et en particulier, le nom de la commande de ses options.

- Le caractère ":" a deux utilisations :

1 - Séparateur : il a le sens général d'affectation numérique. Les éléments qu'il sépare ne sont donc pas syntaxiquement permutables. Il est utilisé dans ce sens dans les commandes SNAP et MODIFY.

2 - Caractère alphabétique : il figure à ce titre dans les éléments du genre étiquette opérationnelle, type de périphérique, etc... sous la forme :

M : xx T : yy C : zz D : t

Exemples : T : PR M : Bi etc...

- Le caractère "." a le sens général d'affectation symbolique. Les éléments qu'il sépare ne sont donc pas permutables.

- Le caractère "," sépare deux paramètres ou groupes de paramètres, dont l'ordre peut être quelconque.

- Le caractère ";" a le sens d'un caractère suite et figure, quand il est utilisé, comme dernier caractère d'une commande, reconnecte la téléscriptrice en entrée pour une nouvelle commande après l'exécution de la précédente, sans qu'il soit besoin de refaire une interruption pupitre.

- Le caractère "@" indique que ce qui suit est une adresse.

- Le caractère "&" est le premier caractère d'un nombre hexadécimal.

- Les caractères "+" et "-" ont respectivement le sens d'addition ou de soustraction. Les éléments qu'ils séparent ne sont pas permutables.

- Les accolades { } indiquent que l'un des termes superposés doit être présent.

- Les crochets [ ] indiquent que le terme inclus peut ne pas être présent.

- Les points de suspension "..." indiquent que le dernier élément spécifié peut être répété.

- Deux termes superposés s'excluent mutuellement.

- La présence des séparateurs "/", ":", ".", "," est liée à celle des éléments qu'ils séparent.

Exemple : Une commande étant spécifiée comme suit :

{ % DUMP / { [M] }
{ % DU   } { [A] , [@ [&] xxxx] , [@ [&] yyyy] }

Les écritures suivantes sont correctes :

% DUMP

% DU

% DU/M

adresse en hexadécimal ↓ , adresse en décimal

% DUMP/A,@ &xxxx, @ yyyy

% DUMP/ @ &xxxx

% LOAD Commande de chargement

. Définition

Commande assurant le chargement en mémoire d'un module de programme ou de données au format IMT sur M : BI.

. Forme

{ % LOAD / % L } / [A] , [@ [&] xxxx] , [S]

. Sens des options

[A] - Quand cette option est présente, l'adresse spécifiée est absolue.
- Quand cette option est absente, l'adresse est relative à la première adresse libre, après le moniteur.
- Cette option n'a pas de sens si aucune adresse n'est spécifiée.

[@ [&] xxxx] - Adresse de chargement. Cette adresse est absolue, si l'option [A] est présente. Elle est relative à la première adresse libre après le moniteur, si l'option [A] est absente.

- Quand cette option est absente, le chargement est effectué à la première adresse libre après le moniteur.

[S] - Edition en fin de chargement sur M : OC du contenu de la PRT du programme, sous la forme suivante:

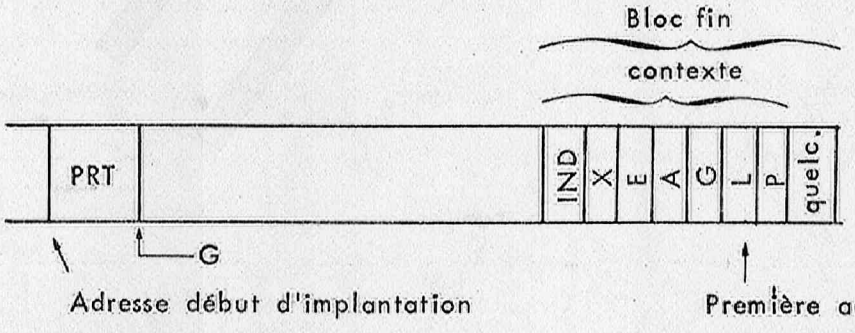
```
xxxx  L - G  }
xxxx  P - G  }  Section n
                :
xxxx  L - G  }  Section 1
xxxx  P - G  }
```

. Effet en cas de chargement d'un module de programme

Le chargeur (loader) charge l'IMT du programme y compris les 16 octets du bloc fin (comprenant le contexte) et laisse après chargement, en G + 6, l'adresse relative à G de la zone commune. Il édite ensuite sur M : OC (téléscriptrice) les informations suivantes :

```
xxxx  }
xxxx  }  SRDn
  I   }
  I   }  PRT si l'option "S" est présente.
  I   }  Les adresses éditées sont relatives à G.
xxxx  }
xxxx  }  SRD1
yyyy     Adresse de début d'implantation
yyyy     Valeur de la base G.
yyyy     Première adresse libre après le programme.
```

Les trois adresses ci-dessus sont absolues.



. Rappel : Un SRD est composé de deux mots

L - G
P - G

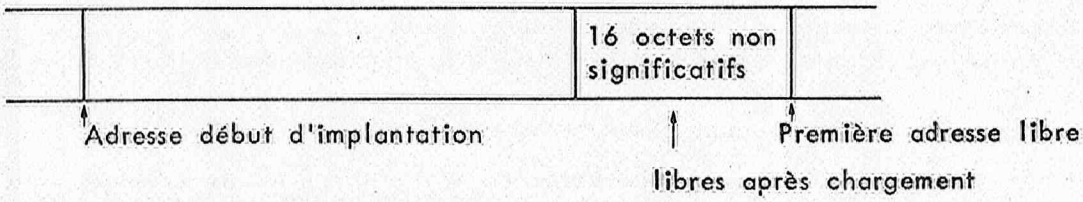
. Effet en cas de chargement d'un module de données

Le chargeur (loader) charge le bloc de données IMT y compris 16 octets non significatifs.
Il édite ensuite sur M : OC (téléscriptrice) les informations suivantes :

xxxx Adresse de début d'implantation (absolue).

xxxx Adresse de début d'implantation (absolue).

xxxx Première adresse libre (absolue).



Rappelons qu'un module de données est obtenu par édition de liens avec l'option de liens avec l'option DA.

. Erreurs au chargement

Le moniteur édite sur M : OC (téléscriptrice) les informations suivantes :

%% LD01 Erreur d'entrée-sortie au chargement.

%% LD02 Parité (ou somme de contrôle) incorrecte au chargement.

%% LD03 Séquencement incorrecte au chargement.

%% AC02 Tentative de chargement sur le moniteur ou sur la zone commune.
Le module chargé n'est pas au format IMT.

. Remarques

1 - Le chargement de blocs de données n'affecte pas la prise en compte du dernier programme chargé qui reste donc le programme qui sera lancé par % RUN.

2 - Le tampon d'entrée binaire est un bloc de 120 octets, situé en début de zone commune.

3 - Il faut bien prendre garde que le % RUN initialise toujours A, E et X.

4 - Le chargement est refusé dans les 256 derniers octets de la mémoire (partie de zone commune utilisée par le moniteur (voir chapitre 3)).

. Exemples

% LOAD Chargement à la première adresse libre après le moniteur.

% LOAD/@&1000 Chargement en +&1000, relativement à la première adresse libre après le moniteur.

% L/A,@&2500 Chargement à l'adresse décimale absolue 2500.

% LOAD/A,@&2FF0,S Chargement à l'adresse absolue 2FF0 et édition de la PRT.

■ % RUN Commande de lancement

. Définition

Commande lançant l'exécution du dernier programme chargé.

. Forme

{ % RUN / % R } / [N [&] xx], [S [&] xx], [L [&] xxxx], [P [&] xxxx], [A [&] xxxx], [E [&] xxxx], [X [&] xxxx]

. Sens des options

[N [&] xx] - Si cette option est présente, [&] xx indique le niveau d'interruption auquel sera connecté le programme. Le programme sera effectivement lancé quand le niveau correspondant sera actif, c'est-à-dire quand l'interruption correspondante sera activée si le niveau n'est pas nul, quand plus aucune interruption ne sera active si le niveau est nul.

- Si cette option est absente, le niveau est pris par défaut.

[S [&] xx] Cette option précise par rapport à quoi sont exprimées les valeurs spécifiées dans les options [L [&] xxxx] et [P [&] xxxx]

- Si cette option est présente, la valeur éventuellement indiquée par l'option [P [&] xxxx] est exprimée relativement à la base P de la section de numéro [&] xx et la valeur éventuellement indiquée par l'option [L [&] xxxx] est exprimée relativement à la base L de la section de numéro [&] xx. Ce sont les éléments de PRT de la section de numéro [&] xx qui sont utilisés à cet effet.

- Si cette option est absente, les valeurs éventuellement spécifiées dans les options [L [&] xxxx] et [P [&] xxxx] sont exprimées relativement à la base G du dernier programme chargé.

[L [&] xxxx] - Si cette option est présente, le registre de base locale est chargé avec la valeur spécifiée. Deux cas sont possibles :

. Option [S [&] xx] présente : la valeur est exprimée relativement a la base L de la section de numéro [&] xx (élément de PRT).

. Option [S [&] xx] absente : la valeur est exprimée relativement à G.

- Si cette option est absente, deux cas sont possibles :

. Option [S [&] xx] présente : le registre de base locale est chargé avec la base L de la section de numéro [&] xx (élément de PRT).

. Option [S [&] xx] absente : le registre de base locale est chargé avec la valeur courante (figurant dans le contexte du programme)

[P [&] xxxx] - Si cette option est présente, le registre programme est chargé avec la valeur spécifiée. Deux cas sont possibles :

. Option [S [&] xx] présente : la valeur est exprimée relativement à la base P de la section de numéro [&] xx (élément de PRT).

. Option [S [&] xx] absente : la valeur est exprimée relativement à G.

- Si cette option est absente, deux cas sont possibles :

. Option [S [&] xx] présente : le registre programme est chargé avec la base P de la section de numéro [&] xx (élément de PRT).

. Option [S [&] xx] absente : le registre programme est chargé avec la valeur courante (figurant dans le contexte du programme).

[A [&] xxxx] - Si cette option est présente, le registre A est chargé avec la valeur spécifiée.

- Si cette option est absente, le registre A est chargé avec la valeur suivant

bits 0 à 7 : niveau de la tâche appelante (ici, niveau de l'interruption pupitre).
bits 8 à 15 : niveau de la tâche lancée.

[E [&] xxxx] - Si cette option est présente, le registre E est chargé avec la valeur spécifiée.

- Si cette option est absente, le registre E est chargé à zéro.

[X [&] xxxx] - Si cette option est présente, le registre X est chargé avec la valeur spécifiée.

- Si cette option est absente, le registre X est chargé avec zéro.

(Ce zéro représente l'adresse relative au début de la zone commune (ZC) utilisateur. Il y a compatibilité à ce niveau entre tous les moniteurs standard MITRA 15).

. Remarque

Après un RUN le contexte du dernier programme chargé n'est pas conservé par le système. Un nouveau RUN ne sera exécutable que si l'utilisateur fourni au système les données nécessaires (RUN avec options).

. Rappel

Dans un programme écrit en assembleur, la section initiale est spécifiée dans la directive END. Dans un programme écrit en LP15, il s'agit de la MAIN SECTION. Une option est de toute façon prise par défaut par l'éditeur de liens.

. Exemples :

% R

% RUN

% RUN/N&00

% RUN/A&0001,E&0001,X&0002

% RUN/S02,L&0000,P&000A,A&FFFF

% R/S&01,L12,P24,X10

% RUN/N4

% RUN/N&F

% R/S01

% R/N4,S&0A

■ % ASSIGN Commande d'assignation

. Définition

Commande modifiant les assignations des étiquettes opérationnelles aux périphériques.

. Forme

$$\left\{\begin{matrix}\%\ \text{ASSIGN}\\ \%\ \text{AS}\end{matrix}\right\}\ /M : 0_1\,0_2,\ T : t_1\,t_2,\ \left[D : [\&]\,d\right],\ \left\{\begin{matrix}\text{BN}\\ \text{AN}\end{matrix}\right\}$$

. Sens des options

$M : 0_1\,0_2$ Spécifie l'étiquette opérationnelle affectée.
Sous MOB, elle doit être une des neuf suivantes :

M : BI, M : BO, M : CI, M : EI, M : EO, M : LO,

M : LL, M : DO, M : SI

L'étiquette oprérationnelle M : OC réservée au dialogue opérateur, est toujours assignée à la téléscriptrice et ne peut être réassignée.

T : t_1 t_2 — Spécifie le type du périphérique à assigner à l'étiquette opérationnelle. Sous MOB, ce type doit être un des suivants :

- T : NO — Annulation d'étiquette. Une entrée-sortie demandée sur l'étiquette opérationnelle ainsi assignée ne sera pas effectuée.
- T : TY — Clavier téléscriptrice (entrée et sortie).
- T : PT — Lecteur-perforateur de ruban téléscriptrice.
- T : PR — Lecteur de ruban perforé rapide.
- T : PP — Perforateur de ruban rapide.
- T : LP — Imprimante.
- T : CR — Lecteur de cartes.
- T : CP — Perforateur de cartes.
- T : DC — Disque à têtes fixes.
- T : 9T — Bandes magnétiques 9 pistes.
- T : DM — Disques à têtes mobiles (amovibles).

Il est bien évident que ne pourront être spécifiés que des types de périphérique dont le handler a été inclu à la génération (voir fiche opérateur du moniteur MOB-E).

D : [&] d — d est le numéro de périphérique pour un coupleur multipériphérique (par exemple bandes).

$0 \leqslant d \leqslant F$ (hexadécimal)

Par défaut, d vaut zéro.

[{BN / AN}] — Spécifie si l'entrée-sortie sera binaire ou alphanumérique. Cette option n'est utile que pour les étiquettes opérationnelles M : EI et M : EO.
Les autres étiquettes opérationnelles sont définitivement alphanumériques ou binaires.

. Etiquettes opérationnelles standard du MOB

Nom	Numéro binaire	Fonction	Mode
M : BI	1	Entrée binaire (Binary Input)	Binaire
M : BO	2	Sortie binaire (Binary Output)	Binaire
M : CI	3	Entrée de commande (Command Input)	Alphanumérique
M : OC	4	Organe de commande (Operator Communication)	Alphanumérique
M : EI	5	Entrée d'éléments (Element Input)	Binaire ou Alphanumérique
M : EO	6	Sortie d'éléments (Element Output)	Binaire ou Alphanumérique
M : LO	7	Sortie de liste (Listing Output)	Alphanumérique
M : LL	8	Journal de bord (Listing Log)	Alphanumérique
M : DO	9	Sortie de diagnostics (Diagnostic Output)	Alphanumérique
M : SI	10	Entrée langage source (Source Input)	Alphanumérique

. Exemples

% AS/M : BI, T : PT — Les entrées binaires demandées sur M : BI se feront sur le lecteur de ruban de la téléscriptrice.

% ASSIGN/M : EO, T : PP, AN — Sortie alphanumérique sur le perforateur de ruban rapide (code ASCII implicite).

% AS/M : EI, T : 9T, D : 4, BN — Les entrées binaires demandées sur M : EI se feront sur le dérouleur de bandes magnétiques numéro quatre.

■ % DISPLAY Commande de sortie des informations système

. Définition

Commande visualisant sur M : OC les adresses remarquables du système.

. Forme

{ % DISPLAY
{ % DI

. Fonction

Impression en hexadécimal sur M : OC de la liste suivante :

xxxx	Première adresse libre après le moniteur.
yyyy	Adresse de la zone commune.
zzzz	Taille mémoire en octets.
tttt	Adresse de début d'implantation du dernier programme chargé, (zéro, s'il n'y a pas eu de programme chargé).
uuuu	Valeur de la base G du dernier programme chargé (ou zéro).
vvvv	Première adresse libre après le dernier programme chargé (ou zéro).
wwww	Adresse du contexte du dernier programme chargé ou adresse du contexte de niveau zéro moniteur (BRU $) si aucun programme n'a été chargé.

% DUMP Commande d'édition mémoire

. Définition

Edition sur M : LO en hexadécimal d'une zone mémoire à raison de huit mots par ligne. L'édition sera toujours arrondie à un nombre entier de lignes. L'édition sera faite au niveau zéro.

. Forme

$$\left\{\begin{matrix}\%\ \text{DUMP}\\ \%\ \text{DU}\end{matrix}\right\} / \left\{\begin{matrix}[\text{M}]\\ [\text{A}]\end{matrix}\right\} , \left\{\left[@\,[\&]\ \text{xxxx}\right], \left[@\,[\&]\ \text{yyyy}\right]\right\}$$

. Sens des options

[M]	Edition de toute la mémoire.
[A]	Quand cette option est présente, les adresses spécifiées sont absolues. Quand cette option est absente, les adresses spécifiées sont relatives à la base G du dernier programme chargé.
[@ [&] xxxx]	Edition du mot adressé (une ligne de dump). Cette adresse est absolue, si l'option A est présente, sinon elle est relative à la base G du dernier programme chargé.
[@ [&] xxxx], [@ [&] yyyy]	Edition de la zone mémoire comprise entre les adresses spécifiées. Ces adresses sont absolues si l'option A est présente, sinon elles sont relatives à la base G du dernier programme chargé. Si aucune adresse n'est spécifiée, édition de la zone mémoire centrale comprise entre les adresses début et fin d'implantation du dernier programme chargé.

. Remarque

Le DUMP s'exécutant au niveau zéro, il est interruptible par l'interruption pupitre. Une seule commande % DUMP peut être prise en compte à la fois. Plus précisement une nouvelle commande % DUMP spécifiée pendant l'exécution d'un DUMP précédemment demandé est acceptée et provoque l'abandon du DUMP en cours.

. Exemples

```
%DU/A,@&1200,@&1210
   1200    0040  01B0  0030  0034  0000  0000  0000  0D30
   1210    0000  0000  0000  0000  0000  0000  0000  0000
%DU/@&1200,@&1210,A
   1200    0040  01B0  0030  0034  0000  0000  0000  0D30
   1210    0000  0000  0000  0000  0000  0000  0000  0000
%DU/@&1200,@&1210
   2400    0000  0000  0000  0000  0000  0000  220A  5344
   2410    F729  4242  210A  0404  F70D  4042  4606  2204
```

↳ Adresse absolue du premier mot de la ligne de DUMP

% Y Abandon du niveau zéro

. Définition

Commande d'abandon du programme en cours au niveau zéro.

. Forme

% Y

. Remarque

Le DUMP s'exécutant au niveau zéro, il peut être arrêté par une commande % Y.

% SNAP Commande d'édition mémoire dynamique

. Définition

Edition dynamique sur M : LO en hexadécimal d'une zone mémoire, à raison de huit mots par ligne.

. Forme

$$\left\{\begin{matrix}\%\ \text{SNAP}\\ \%\ \text{SN}\end{matrix}\right\} / \left\{\left[\begin{matrix}[\text{A}]\\ \text{S}\ [\&]\ xx\end{matrix}\right]\right\}, @\ [\&]\ zzzz : \left\{\left[\begin{matrix}[\text{M}]\\ @\ [\&]\ xxxx\end{matrix}\right], \left[@\ [\&]\ yyyy\right]\right\}$$

. Sens des options

[A]	Si cette option est présente, toutes les adresses figurant dans la commande sont absolues.
[S [&] xx]	Si cette option est présente, toutes les adresses figurant dans la commande sont relatives à la base P de la section du numéro [&] xx.
	Si aucune des deux options [A] ou [S [&] xx] n'est présente, toutes les adresses figurant dans la commande sont relatives à la base G du dernier programme chargé.
@ [&] zzzz	Spécifie l'adresse de l'instruction au passage de laquelle l'édition de mémoire sera exécutée. Après édition, le programme reprend à l'endroit où il a été interrompu.
[M]	Edition de toute la mémoire.
[@ [&] xxxx]	Edition de l'élément adressé (une ligne de dump).
[@ [&] xxxx], [@ [&] yyyy]	Edition de la zone mémoire comprise entre les adresses spécifiée, édition de la zone mémoire comprise entre les adresses début et fin d'implantation du dernier programme chargé.

. Remarques

- Une seule commande % SNAP peut être prise en compte à la fois.
- On ne peut interrompre un SNAP en cours d'exécution pour spécifier une nouvelle commande % SNAP.
- Si aucun SNAP n'est en cours, une nouvelle commande % SNAP annule et remplace la précédente.
- Les deux moments où l'on spécifie normalement une commande % SNAP sont :

. après chargement, avant le lancement,

. après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

- On ne peut spécifier, pour une même adresse programme, qu'une seule commande dynamique (% TRACE, % SNAP, % HALT, % NEXT). Ceci signifie en particulier que, quand on a spécifié une commande % TRACE avec deux adresses, toutes les adresses intermédiaires tracées sont interdites comme adresse programme d'une commande % SNAP.
- On n'effectue de SNAP que sur un programme au niveau zéro.

. Exemples

% SNAP/S01, @&24
% SNAP/@&24, S01
% SN/ @36, S01

Dump à l'adresse &24 (ou 36) du LPS de la section 01 du dernier programme chargé ainsi que des registres.

% SN/A, @&2014 :@&1007, @&3004
% SN/A, @&2014 :@&1007, @&3004, A

Dump à l'adresse absolue &2014 de la zone mémoire comprise entre les adresses absolues &1000 et &300F.

■ % TRACE Commande d'édition dynamique des registres

. Définition

Edition dynamique sur M : LO des six premiers registres des indicateurs et de l'instruction courante (après exécution de celle-ci).

. Forme

{% TRACE / % TR} / {[A] / [S [&] xx]} , @ [&]xxxx, [@ [&]yyyy]

. Sens des options

[A] Si cette option est présente, toutes les adresses figurant dans la commande sont absolues.

[S [&] xx] Si cette option est présente, toutes les adresses figurant dans la commande sont relatives à la base P de la section de numéro [&] xx.

Si aucune des deux options [A] ou [S [&] xx] n'est présente, toutes les adresses figurant dans la commande sont relatives à la base G du dernier programme chargé.

@ [&] xxxx Spécifie l'adresse de la première instruction tracée.

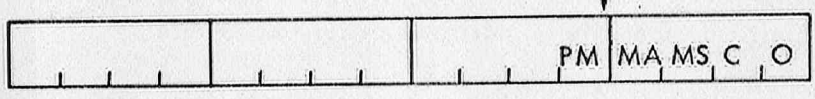
[@ [&] yyyy] Spécifie l'adresse de la dernière instruction tracée. Toutes les instructions exécutées entre @[&] xxxx et [@ [&] yyyy] seront tracées.

Si cette option est absente, seule l'instruction d'adresse [@[&] xxxx] sera tracée.

. Format de l'édition (hexadécimal)

X = xxxx E = xxxx A = xxxx G = xxxx L = xxxx P = xxxx I = xxxx xxxx

Reg. X Reg. E Reg. A Base G Base L Base P Indicateurs Binaire de l'instruction tracée



. Remarques

- Une seule commande % TRACE peut être prise en compte à la fois.
- On ne peut interrompre une TRACE en cours d'exécution pour spécifier une nouvelle commande % TRACE.
- Si aucune TRACE n'est en cours, une nouvelle commande % TRACE annule et remplace la précédente.
- Les deux moments où l'on spécifie normalement une commande % TRACE sont :

après chargement, avant le lancement,

. après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

- On ne peut spécifier, pour une même adresse programme, qu'une seule commande dynamique (% TRACE, % SNAP, % HALT, % NEXT). Ceci signifie en particulier que, quand on a spécifié une commande % TRACE avec deux adresses, toutes les adresses intermédiaires tracées sont interdites à une autre commande dynamique.

- On n'effectue de TRACE que sur un programme au niveau zéro.

- Si au cours de l'exécution d'une commande % TRACE, on passe par un CSV ou un CLS, la section appelée ne sera pas tracée. Ceci a pour but de ne pas alourdir exagérément le nombre d'instructions tracées et ceci respecte l'ordre normal de la mise au point (mise au point préalable des sections appelées).

. Exemples

% TRACE/A,@&2000

% TR/S02,@&2010,@&2016

% TR/@&2000, A

% MODIFY Commande de modification mémoire

. Définition

Modification d'un ou plusieurs mots mémoire. Les adresses spécifiées doivent être des adresses mot.

. Forme

{% MODIFY / % MO} / {[A] / [S [&] xx, [P]]} ,@[&]aaaa :[&]xxxx,[[&]xxxx]...,[@[&]aaaa :[&]xxxx,[[&]xxxx]...] .

. Sens des options

[A] — Les adresses @[&] aaaa figurant dans la commande sont absolues.

[S [&] xx, P] — Les adresses @[&]aaaa figurant dans la commande sont relatives à l'une des bases de la section de numéro & xx.

C'est la base P si l'option [P] est présente.

C'est la base L si l'option [P] est absente.

Si aucune des options [A] et [S [&] xx, [P]] n'est présente, les adresses @ [&] aaaa figurant dans la commande sont relatives à la base G du dernier programme chargé.

@[&] aaaa — Adresse à partir de laquelle on désire modifier la mémoire.

[&] xxxx — Valeur à implanter sur un mot mémoire.
L'adresse de ce mot mémoire par rapport à la dernière adresse @ [&] aaaa spécifiée, est la suivante :

@[&] aaaa : [&] xxxx, [&] xxxx, [&] xxxx, ...

@[&] aaaa @[&] aaaa + 2 @ [&]aaaa + 4 etc...

. Exemples

% MODIFY/A,@ &1000 : &0000, &0001, &0002, @&1020 : &1000, &1001

Résultat

Adresse	Contenu
1000	0000
1002	0001
1004	0002
1020	1000
1022	1001

% MODIFY/S01, P,@ &0004 : &C700, @&002A : &0024, &1136

% MO/S01, @&00F0 : &FFFF, &FFFF, &FFFF

% MODIFY/@&0022 : &0001, &0000,@&003C : &0000,@&004E : &0000

% HALT Commande d'arrêt sur instruction

. Définition

Arrêt sur adresse pour entrée de commandes après édition sur M : OC des six premiers registres, des indicateurs et de l'instruction adressée (après l'exécution de celle-ci). L'exécution d'une commande % HALT précède normalement l'entrée des commandes % SNAP, % TRACE et % NEXT, la reprise se faisant par la commande % EXECUTE.

. Forme

{ % HALT / % HA } / [{ [A] / [S [&] xx] } , @ [&] aaaa]

. Sens des options

[A] L'adresse indiquée est absolue.

[S [&] xx] L'adresse indiquée est relative à la base P de la section de numéro [&] xx.

Si aucune des options [A] et [S [&] xx] n'est présente, l'adresse indiquée est relative à la base G du dernier programme chargé.

[@ [&] aaaa] Spécifie l'adresse de l'instruction après l'exécution de laquelle le contrôle est donné sur M : OC pour l'entrée de commandes.

Préalablement ont été imprimés sur M : OC les six premiers registres, les indicateurs et l'instruction adressée.

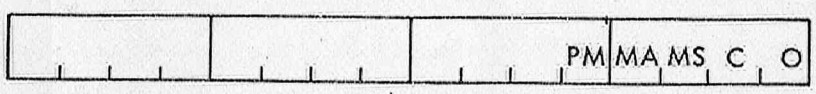
. Absente d'options

Si aucune option n'est précisée (% HALT), l'adresse prise en compte sera celle qui avait été spécifiée dans la précédente commande % HALT ou qui résultait de l'exécution de la dernière commande % NEXT.

. Format de l'édition (hexadécimal)

X = xxxx E = xxxx A = xxxx G = xxxx L = xxxx P = xxxx I = xxxx xxxx

Reg. X — Reg. E — Reg. A — Base G — Base L — Base P — Indicateurs — Instruction adressée et dernière exécutée



. Remarques

- Une seule commande % HALT peut être prise en compte à la fois.

- Une nouvelle commande % HALT annule et remplace la précédente (voir cependant le cas "absence d'options").

- Les deux moments où l'on spécifie normalement une commande % HALT sont :

. après chargement, avant lancement,

. après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

- On ne peut spécifier, pour une même adresse programme, qu'une seule commande dynamique (% TRACE, % SNAP, % HALT, % NEXT). Ceci signifie en particulier que, quand on a spécifié une commande % TRACE avec deux adresses, toutes les adresses intermédiaires tracées sont interdites dans une commande % HALT.

- On n'effectue de HALT que sur un programme au niveau zéro.

- Une commande % HALT donnée n'est exécutée qu'une fois. Si l'on veut l'exécuter une seconde fois (dans le cas d'une boucle par exemple), on devra la réactiver en frappant % HALT avant de poursuivre (% EXECUTE).

. Exemples

% HALT/A,@&2000

% HALT/@&2000, A

% HA/@24

% HALT

% HA/S02,@&F0

Un exemple complet d'enchaînement des % HALT et % EXECUTE sera donné dans la description de la commande % EXECUTE.

% NEXT Commande d'exécution d'une séquence d'instructions

. Définition

Implantation d'une "HALT" à l'adresse courante augmentée d'un nombre de mots (nombre d'instructions). L'exécution sera lancée par % EXECUTE.

Au passage de l'adresse où est implantée la "HALT", il y aura édition sur M : OC des six premiers registres des indicateurs et de l'instruction adressée aprés exécution de celle-ci.

Une commande % NEXT suit toujours une commande % HALT ou % NEXT. La première instruction exécutée sera l'instruction suivant l'instruction éditée dans cette commande % HALT ou % NEXT précédente.

A l'arrêt après édition, le contrôle est rendu sur M : OC et l'entrée de commandes est possible, la reprise se faisant par % EXECUTE.

. Forme

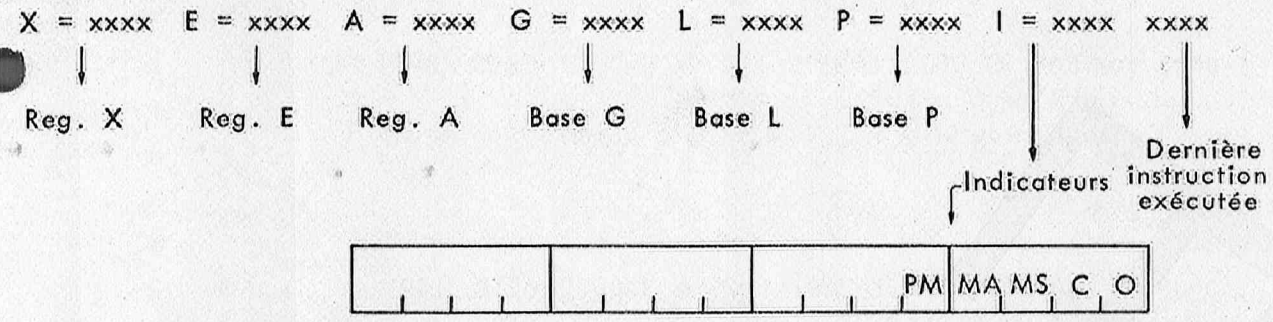
$$\left\{\begin{matrix}\%\ \text{NEXT}\\ \%\ \text{NE}\end{matrix}\right\} \ / \ \left[\text{N}\ [\&]\ \text{xx}\right]$$

. Sens des options

[N [&] xx] Implantation d'une HALT à l'adresse courante augmentée de [&]xx mots (instructions).

Valeur par défaut : N1

. Format d'édition



. Remarques

- Une seule commande % NEXT peut être prise en compte à la fois.
- On ne peut interrompre un NEXT en cours d'exécution pour spécifier une nouvelle commande % NEXT.
- Si aucun NEXT n'est en cours, une nouvelle commande % NEXT annule et remplace la précédente.
- Une commande % NEXT n'a de sens que si au moins une commande % HALT a déjà été exécutée. Une commande % NEXT ne se spécifie donc normalement qu'après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.
- On ne peut spécifier pour une même adresse programme, qu'une seule commande dynamique (% TRACE, % SNAP, % HALT, % NEXT). Ceci signifie en particulier que, quand on a spécifié une commande % TRACE avec deux adresses, toutes les adresses intermédiaires tracées sont interdites dans une commande % NEXT.
- On n'effectue de NEXT que sur un programme au niveau zéro.
- Si au cours de l'exécution d'une commande % NEXT, on passe par un CSV ou un CLS, les instructions de la section appelée ne seront pas comptées (voir commande % TRACE, remarques).

. Exemple 1

```
% NEXT
% EX
% NE
% EX
% NEXT/N&0A
% EX
% NEXT/N2
```

. Exemple 2

```
% HA/A,@&=78A
% R
  X=0049    E=0008 A=18A0 G=1B12 L=2692 P=278C I=0002 F005 (SLLS=5)
            * Halte en 278A
            * Après exécution de SLLS=5, P=278C
% NE;
% EXE
  X=0049    E=0008 A=18A0 G=1B12 L=2692 P=2792 I=0002 C003 (BE $+3)
            * Equivalent à % HA/A,@&278C
            * Après exécution de BE $+3, P=2792 (Branchement satisfait)
```

```
% NE/N3;
% EXE
  X=0049    E=0008 A=18A0 G=1B12 L=2692 P=2794 I=0002 C403 (BAN $+3)
           * Equivalent à % HA/A,@&2792
           * Après l'exécution de BAN $+3, P=2794 (Branchement non satisfait)
% EN/N3;
% EXE
  X=0049    E=0008 A=0000 G=1B12 L=2692 P=279A I=0002 0E1B (LBR)
           * Equivalent à % HA/A,@&2798
           *Après exécution de LBR, P=279A
% NE;
% EXE
  X=0049    E=0008 A=0000 G=1B12 L=2692 P=279C I=0002 030C (EOR)
           *Equivalent à % HA/A,@&279A
           *Après exécution de EOR, P=279C.
```

■ % EXECUTE Commande de reprise du programme utilisateur

. Définition

Un programme au niveau zéro suspendu à la suite de l'exécution d'une commande % HALT ou % NEXT sera relancé par % EXECUTE.

Une commande % EXECUTE ne peut remplacer % RUN et réciproquement, une commande % RUN ne peut remplacer une commande % EXECUTE.

. Forme

{ % EXECUTE }
{ % EX   }

. Fonction et exemple

L'analyse de l'exécution du programme ci-contre peut se faire de la façon suivante :

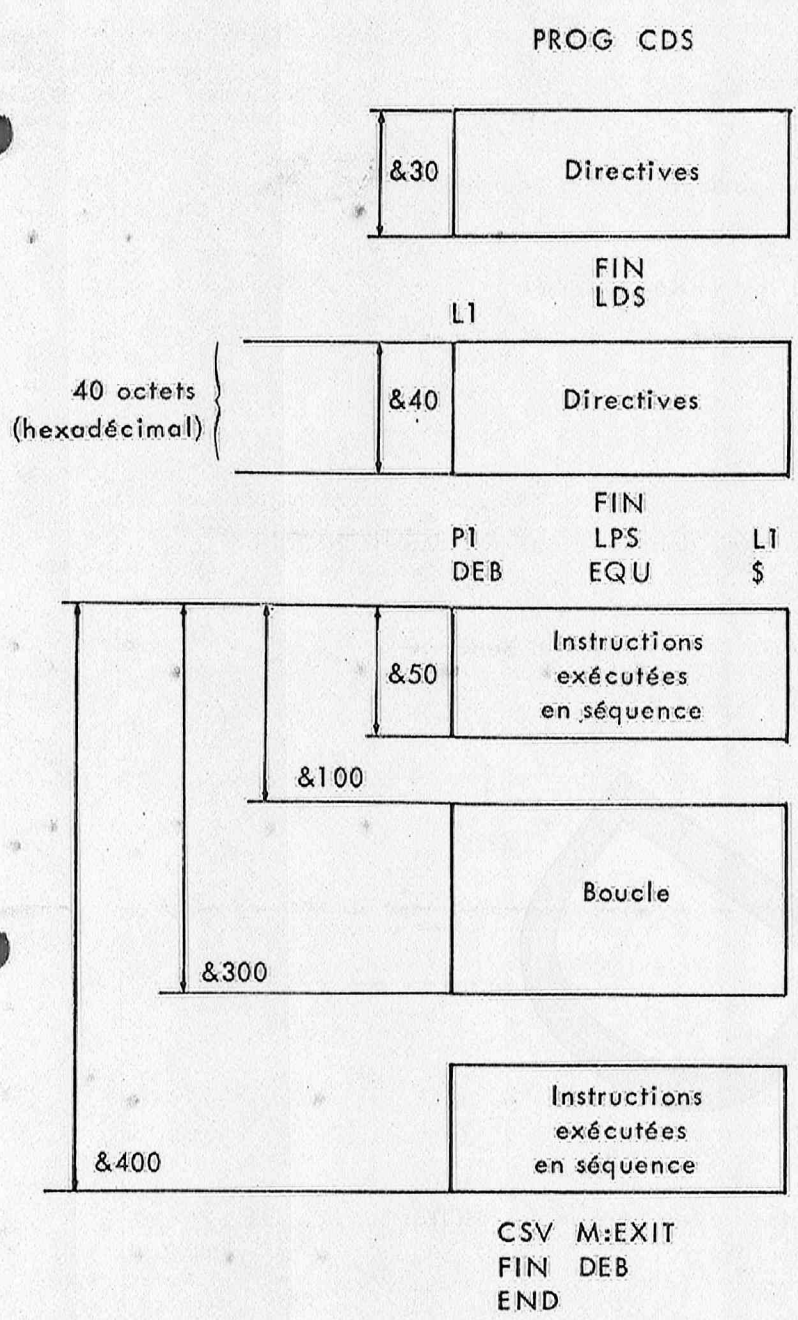
```
% LOAD
% HALT/S01,@&20
% RUN
   :      Déroutement du début de programme
   .
```

Exécution de l'instruction adressée dans le HALT précédent

Edition des registres et indicateurs ainsi que de l'instruction adressée

Passage en entrée de commandes

```
% NEXT/N10
```



Exécution des 10 instructions suivantes

Edition des registres et indicateurs ainsi que de la dernière instruction

Passage en entrée de commandes

% SNAP/@170 : @ &30, @&70

% HALT/S01, @ &300

% EXECUTE

⋮ Relance et déroulement du programme

Exécution du SNAP et éditions

⋮ Poursuite du déroulement, premier passage dans la boucle

Exécution de l'instruction adressée dans le HALT précédent

Edition des registres et indicateurs ainsi que de l'instruction adressée

Passage en entrée de commandes

% TRACE/S01, @&202, @&228

% HALT

% EXECUTE

⋮ Relance et déroulement du programme, deuxième passage dans la boucle

Exécution du TRACE et éditions

⋮ Poursuite du déroulement, deuxième passage dans la boucle

Exécution de l'instruction adressée dans le HALT précédent

(réactivation du % HALT/S01, @ &300)

Edition des registres et indicateurs ainsi que de l'instruction adressée

Passage en entrée de commande

% MODIFY
% MODIFY } Corrections

% DUMP Vérification

% EX Relance du programme en fonctionnement normal

% PERFORM Commande de calcul

• Définition

Effectuer des calculs sur les bases et les registres avec édition du résultat sur M : OC.

• Forme

{% PERFORM / % PE} , { X / E / A / G / L / P / [&]aaaa } {+ / -} [&] bbbb

• Sens des options

X Valeur courante du registre X (Index).

E Valeur courante du registre E (Extension).

A	Valeur courante du registre A (Accumulateur).
G	Valeur courante de la base G (Base générale).
L	Valeur courante de la base L (Base locale).
P	Valeur courante de la base P (Base programme).
[&] aaaa	Valeur numérique.
+	L'opération est une addition.
-	L'opération est une soustraction.
[&] bbbb	Deuxième terme de l'opération.

. Exemples

```
% PERFORM/L + &00F0
% PE/P - 24
```

9-3. COMMUNICATIONS OPERATEUR - MESSAGES OPERATEUR

Les messages suivants sont édités sur M : OC (téléscriptrice) :

Messages	Signification
Au chargement du système	
SYSTEM READY	Le système vient d'être chargé en mémoire. Il est en attente d'interruption pupitre.
Concernant les commandes	
%	
%% AC01	Erreur de syntaxe dans une commande.
%% AC02	Exécution impossible d'une commande.
%% AC03	Exécution impossible d'une commande spécifique de MOB.E
Au chargement	
%% LD01	Erreur d'entrée-sortie au chargement.
%% LD02	Parité ou checksum (somme de contrôle incorrecte au chargement.
%% LD03	Séquencement incorrecte au chargement.
Erreurs d'entrée-sortie	
%% IO01	Erreur logique détectée en fin de transfert.
%% IO02	Erreur logique détectée à l'initialisation du transfert.
%% IO03	Erreur physique détectée en fin de transfert.
%% IO04	Erreur physique détectée à l'initialisation du transfert.
Déroutements	
xxxx	Mot d'état déroutement.
xxxx xxxx xxxx	P absolu L absolu Indicateurs } sur l'instruction ayant provoqué le déroutement.
xxxx xxxx xxxx	P-G L-G Indicateurs } sur le dernier appel moniteur
%% DRnn	nn = 01 Déroutement coupleur nn = 02 Déroutement programme.
Incidents graves	
%% CS SYS READY	Coupure secteur ayant entraîné une réinitialisation du système (of chapitre 5).
%% DR xxxx SYS READY	Déroutement ayant entraîné une réinitialisation du système (of chapitre 6).

9-4. MODULES MONITEUR DU MOB-E

Liste

M : IO	Appel d'entrée-sortie.
M : WAIT	Attente de fin de transfert.
M : EXIT	Fin de programme.
M : KEY	Clés et voyants.
M : DCBN	Conversion décimal-binaire.
M : HXBN	Conversion hexadécimal-binaire.
M : BNDC	Conversion binaire-décimal.
M : BNHX	Conversion binaire-hexadécimal.
M : ASEB	Conversion ASCII-EBCDIC.
M : EBAS	Conversion EBCDIC-ASCII.
M : LOAD	Chargement d'un module IMT.
M : MOVE	Déplacement d'une chaîne d'octets.
M : EDIT	Edition d'une zone mémoire (TWB étendu).
M : DUMP	Edition d'une zone mémoire (TWB normal).

Seuls sont décrits dans la suite, les modules ne figurant que dans MOB-E.

M : DUMP Edition d'une zone mémoire

. Séquence d'appel

```
LEA     CB
LDE     ADEBUT
LDX     AFIN
CSV     M : DUMP
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse par rapport à G du bloc de commande, le registre E doit contenir l'adresse par rapport à G du début de la zone mémoire à éditer, le registre X doit contenir l'adresse par rapport à G de la fin de la zone mémoire à éditer.

. Fonction

Impression, sur le périphérique associé à l'étiquette opérationnelle spécifiée dans le CB, de la zone mémoire spécifiée à raison de huit mots par ligne, étant entendu qu'il est sortie un nombre entier de lignes.

. Bloc de commande

```
CB   DATA, 1    0
     DATA, 1    0
     DATA, 1    80
     DATA, 1    M : xx   (Etiquette opérationnelle utilisée.
                          En cas d'utilisation de MITRAS 1, c'est le numéro binaire lui-même,
                          qui devra être spécifié).
     DATA       TAMPON   (Adresse du tampon fourni par l'utilisateur)
     DATA       58       (Taille minimum du tampon en octets)
```

• Format d'impression

Adresse absolue 1er octet de la ligne	Contenu de 16 octets par adresses croissantes							
aaa0	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
bbb0	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
|	|	|	|	|	|	|	|	|
|	|	|	|	|	|	|	|	|
zzz0	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx

avec aaa0 ⩽ ADEBUT ⩽ aaaF

zzz0 ⩽ AFIN ⩽ zzzF

• Eléments communiqués en sortie

A, E et X quelconques.

• Mémoires de TWB utilisées

T0 à N0 (TWB standard).

• Modules appelés

M : BNHX, M : IO, M : WAIT.

Listing

annexe A

(début de mémoire d'un MOB)

Ce listing est donné à titre d'exemple. Des différences de détail peuvent exister entre différents MOB.

```
XCALL/ASSEMH,SI,L0,60
      1                 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      2                 *                                                                               *
      3                 *                          MONITEUR MINIMUM                                     *
      4                 *                                                                               *
      5                 *                          VERSION DE BASE                                      *
      6                 *                                                                               *
      7                 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


      8                       PAGE
      9                 CDS   CDS     DUM           *                                                 *
     10                 *                           * * * * * * * * * * * * * * * * * * * * * * * * * *
     11 0000           PMG   RES     1             * P-G APPELANT                                      *
     12 0002           LMG   RES     1             * L-G APPELANT                                      *
     13 0004           IND   RES     1             * INDICATEURS                                       *
     14                 *                           *                                                 *
     15 0006           AZC   RES     1             * ADR. ZC/G                                         *
     16 0008           T0    RES     1             *                                                   *
     17 000A           T1    RES     1             *                                                   *
     18 000C           T2    RES     1             *                                                   *
     19 000E           T3    RES     1             *                                                   *
     20 0010           T4    RES     1             *                                                   *
     21 0012           T5    RES     1             *                                                   *
     22 0014           T6    RES     1             *                                                   *
     23 0016           T7    RES     1             *                                                   *
     24 0018           N3    RES     1             * NIVEAU S/P  3                                     *
     25 001A           N2    RES     1             * NIVEAU S/P  2                                     *
     26 001C           N1    RES     1             * NIVEAU S/P  1                                     *
     27 001E           N0    RES     1             * NIVEAU S/P  0                                     *
     28                 *                           *                                                 *
     29                 * MEMOIRES UTILISEES PAR LE MODULE M:LOAD                                       *
     30                 *BLOC DE COMMANDE           *                                                 *
     31 0020           CB    RES,1   1             * OCTET EVENEMENT                                   *
     32 0021           INDIC RES,1   1             * INDICATEUR = 0 ( NON DISQUE )                     *
     33 0022           ORDRE RES,1   1             * ORDRE = 0 LECTURE EN M                            *
     34 0023           ETIQ  RES,1   1             * ETIQUETTE = 1 ( BI )                              *
     35 0024           ADR   RES     1             * ADR/G BUFFER MEMOIRE                              *
     36 0026           LONG  RES     1             * COMPTE D'OCTETS                                   *
     37 0028           ACCU  RES     1             *                                                   *
     38 002A           ADTAMP RES    1             * ADR/G TAMPON ( 120 )                              *
     39 002C           LBL   RES     1             * TAILLE ENREGISTREMENT ( =120 )                    *
     40 002E                 RES     1             *                                                   *
     41 0030           NOV   RES,1   1             * INUTILISE EN VERSION MINI                        *
     42 0031           SORPRT RES,1  1               * = 1 SORTIE DE LA PRT                           *
     43                                            * = 0 PAS DE SORTIE                               *
     44 0032           N     RES,1   1             * = 1 PROGRAMME                                     *
     45                                            * = 3 DONNEES                                      *
     46 0033           TB1   RES,1   1             * INUTILISE  OCTET DE TRAVAIL                       *
     47 0034                 RES     2             *                                                   *
     48 0038           ADPR  RES     1             * ADR/G DE CHARGEMENT DU PROGRAMME                  *
     49 003A           LPR   RES     1             * LONGUEUR IMT AVEC PRT ET CONTEXTE                 *
     50 003C           FINPR RES     1             * ADRESSE FIN DE PROGRAMME                          *
     51 003E           ADCTX RES     1             * ADR/G DU CONTEXTE DU PROGRAMME CHARGE             *
     52       003A     GPR   EQU     LPR           * VALEUR DE G DU PROGRAMME                          *
     53       003D     TB2   EQU     NOV           * OCTET DE TRAVAIL                                  *
     54                 *                           *                                                 *
     55                 *       MEMOIRE UTILISEE PAR M:EXIT, M:IO, M:WAIT                             *
     56       0010      A:CB  EQU     T4            *                                                   *
     57                 *                           *                                                 *
     58                 *                           *                                                 *
```

```
                        * MEMOIRES UTILISEES PAR LES MODULES M:ASEB M:EBAS
                        *
0008            PTAMP   EQU     T0          * POINTEUR ADRESSE COURANTE
000A            SAVCAR  EQU     T1          * VALEUR COURANTE DU CARACTERE
000C            SAVEX   EQU     T2          * VALEUR COURANTE DE L'INDEX
                        *
                        *       MEMOIRES UTILISEES PAR M:ANAC, M:MOVE
                        *
                ZANAC   IDS     DUM
0008            A:ADC   EQU     T0
000A            CARA    EQU     T1
0014            C:ZERO  EQU     T6          * MOT NUL POUR APPEL ADRESSE/G DANS X
0016            C:ADT   EQU     T7          * ADRESSE COURANTE DU TAMPON
0018            ICOM    EQU     N3          * BOOLEEN TYPE DE LA COMMANDE
0000                    RES     16
0020                    RES     12
0038            C:CHA   RES     1           * ADRESSE DEBUT DE CHARGEMENT
                        *
                        FIN
                        *
                        * MEMOIRES UTILISEES PAR LES MODULES M:EDIT, M:ERED
                        *
                ZEDIT   IDS     DUM
0000                    RES     16
0020            CBD     RES     2           * DEBUT DU CB DE SORTIE SUR.OC
0024            ADTP    RES     1           * ADRESSE DU TAMPON
0026            LGTP    RES     1           * LONGUEUR DU TAMPON
0028                    RES     1
002A            TPHX    RES     2           * TAMPON D'EDITION HEXA
002E            LGZ     RES     1           * LONGUEUR DE LA ZONE
0030            ADZ     RES     1           * ADRESSE DE LA ZONE
                        *
0016            IDX     EQU     T7          * INDEX OCTET COURANT
0016            TYPER   EQU     T7          * UTILISE PAR M:ERED
                        FIN
                        *                   * * * * * * * * * * * * * * * * * * * *
                _       FIN
                        PAGE
                        *                   * * * * * * * * * * * * * * * * * * * *
                SUPER   LDS                 * DONNEES COMMUNES AU SUPERVISEUR     *
                        *                   * * * * * * * * * * * * * * * * * * * *
0000                    RES     1           *
                        *                   * * * * * * * * * * * * * * * * * * * *
0002            TYPDER  RES     1           * TYPE DE DEROUTEMENT
0002            TYDR    EQU     TYPDER      * UTILISE PAR M: TRAP
0004            PMGDER  RES     1           * P - G DU PROGRAMME DEROUTE
0006            LMGDER  RES     1           * L - G DU PROGRAMME DEROUTE
0008            INDDER  RES     1           * INDICATEURS DU PROGRAMME DEROUTE
                        *                   * * * * * * * * * * * * * * * * * * * *
000A    000A    T:CPT   DATA    CPT         * POINTEUR TABLE 'POINTEURS DE CONTEXTE'
000C            T:PRTS  RES     1           * POINTEUR TABLE 'PRT SUPERVISEUR'
000E            A:SYS   RES     1           * ADRESSE FIN DE SYSTEME MINI
0010    1F38    A:ZC    DATA    &1F38       * TAILLE MEMOIRE QO
                0010    A:BUF   EQU     A:ZC    * ADR. TAMPON - G DU PUPITRE
                        *                   * * * * * * * * * * * * * * * * * * * *
0012    000A    T:IOT   DATA    IOT         *
0014    000A    T:IOT1  DATA    IOT1        *
0016    000A    T:OLT   DATA    OLT         *
0018    000A    T:OLT1  DATA    OLT1        *     TABLES DU SYSTEME D'E/S
001A    000A    T:OLTB  DATA    OLTB        *
001C    000A    T:DCT   DATA    DCT         *
001E    000A    T:DCT1  DATA    DCT1        *
0020    000A    T:DCT2  DATA    DCT2        *
                        *                   * * * * * * * * * * * * * * * * * * * *
0022    000A    P:CTXO  DATA    R:CTXO      * POINTEUR CONTEXTE O
0024    000A    T:ISEB  DATA    ISEB        * POINTEUR TABLE CONVERSION ISO<==>EBCDIC
                        *                   * * * * * * * * * * * * * * * * * * * *
0026    000A    P:IO    DATA    R:IO        * RELAIS BRANCHEMENT VERS M:IO
0028    000A    P:WAIT  DATA    R:WAIT      * RELAIS BRANCHEMENT VERS M:WAIT
002A    000A    P:RACK  DATA    R:RACK      * RELAIS BRANCHEMENT VERS M:CSUM
002C    000A    P:EBAS  DATA    R:EBAS      * RELAIS BRANCHEMENT VERS M:EBAS
002E    000A    P:HXBN  DATA    R:HXBN      * RELAIS BRANCHEMENT VERS M:HXBN
0030    000A    P:BNHX  DATA    R:BNHX      * RELAIS BRANCHEMENT VERS M:BNHX
0032    000A    P:DCBN  DATA    R:DCBN      * RELAIS BRANCHEMENT VERS M:DCBN
0034    000A    P:MOVE  DATA    R:MOVE      * RELAIS BRANCHEMENT VERS M:MOVE
0036    000A    P:EDIT  DATA    R:EDIT      * RELAIS BRANCHEMENT VERS M:DUMP
0038    000A    P:ERED  DATA    R:ERED      * RELAIS BRANCHEMENT VERS M:ERED
003A    000A    P:SPER  DATA    R:SPER      * RELAIS BRANCHEMENT VERS M:SPER
```

```
137                     *
138 003C 0000A  L:ANAC  DATA    N:ANAC          * POINTEUR (LOST) ANALYSEUR DE COMMANDES *
139                     *                       * * * * * * * * * * * * * * * * * * * * *
140 003E 0000A  T:CBCC  DATA    CBBC            *                                         *
141 0040 0000A  T:CDER  DATA    CDER            *                                         *
142                     *                       * * * * * * * * * * * * * * * * * * * * *
143 0042 0000   K:ZERO  DATA    0               * CONSTANTE ZERO                          *
144 0044 0001   K:UN    DATA    1               * CONSTANTE UN                            *
145 0046 0002   K:DEUX  DATA    2               * CONSTANTE  2                            *
146 0048 0003   K:TROI  DATA    3               * CONSTANTE : 3                           *
147 004A 0004   K:QUAT  DATA    4               * CONSTANTE : 4                           *
148 004C 0006   K:SIX   DATA    6               * CONSTANTE :6                            *
149 004E 6667   K:1:10  DATA    &6667           * 1/10 CADRE *3                           *
150                     *                       *                                         *
151                     *                       * * * * * * * * * * * * * * * * * * * * *
152 0050   00   CBBC    DATA,1  0,0,&80,M:BC    * C.B. STANDARD POUR LIASR33             *
    0051   00
    0052   80
    0053   04
153 0054 0028           DATA    40,6            *                                         *
    0056 0006
154 0058 4040           DATA    &4040           *                                         *
155                     *                       * * * * * * * * * * * * * * * * * * * * *
156 005A   C4   CDER    TEXT    DRIMLUAL        *                                         *
    005B   D9
    005C   C9
    005D   D6
    005E   D3
    005F   C4
    0060   C1
    0061   C3
157                     *                       * * * * * * * * * * * * * * * * * * * * *
158                     *                       * * * * * * * * * * * * * * * * * * * * *
159                                             * TABLE DE TRANSCODAGE  ASCII <-> EBCDIC  *
160 0062 0001   ISEB    DATA    &0001           *     NULL - SOH                          *
161 0064 0203           DATA    &0203           *     STX - ETX                           *
162 0066 372D           DATA    &372D           *     EOT - ENQ                           *
163 0068 2E2F           DATA    &2E2F           *     ACK - BELL                          *
164 006A 1605           DATA    &1605           *     BS  - HT                            *
165 006C 1508           DATA    &1508           *     LF  - VT                            *
166 006E 0C0D           DATA    &0C0D           *     FF  - CR                            *
167 0070 0E0F           DATA    &0E0F           *     SO  - SI                            *
168 0072 1011           DATA    &1011           *     DLE - DC1                           *
169 0074 1213           DATA    &1213           *     DC2 - DC3                           *
170 0076 3C3D           DATA    &3C3D           *     DC4 - NAK                           *
171 0078 3226           DATA    &3226           *     SYN - ETB                           *
172 007A 1819           DATA    &1819           *     CAN - EM                            *
173 007C 3F27           DATA    &3F27           *     SUB - ESC                           *
174 007E 1C1D           DATA    &1C1D           *     FS  - GS                            *
175 0080 1E1F           DATA    &1E1F           *     RS  - US                            *
176 0082 404F           DATA    &404F           *     .   -                               *
177 0084 7F7B           DATA    &7F7B           *         - $                             *
178 0086 5B6C           DATA    &5B6C           *     $   - %                             *
179 0088 507D           DATA    &507D           *     &   - '                             *
180 008A 4D5D           DATA    &4D5D           *     (   - )                             *
181 008C 5C4E           DATA    &5C4E           *     *   - +                             *
182 008E 6B60           DATA    &6B60           *     ,   - -                             *
183 0090 4B61           DATA    &4B61           *     .   - /                             *
184 0092 F0F1           DATA    &F0F1           *     0   - 1                             *
185 0094 F2F3           DATA    &F2F3           *     2   - 3                             *
186 0096 F4F5           DATA    &F4F5           *     4   - 5                             *
187 0098 F6F7           DATA    &F6F7           *     6   - 7                             *
188 009A F8F9           DATA    &F8F9           *     8   - 9                             *
189 009C 7A5E           DATA    &7A5E           *     :   - ;                             *
190 009E 4C7E           DATA    &4C7E           *     <   - =                             *
191 00A0 6E6F           DATA    &6E6F           *     >   -                               *
192 00A2 7CC1           DATA    &7CC1           *     @   - A                             *
193 00A4 C2C3           DATA    &C2C3           *     B   - C                             *
194 00A6 C4C5           DATA    &C4C5           *     D   - E                             *
195 00A8 C6C7           DATA    &C6C7           *     F   - G                             *
196 00AA C8C9           DATA    &C8C9           *     H   - I                             *
197 00AC D1D2           DATA    &D1D2           *     J   - K                             *
198 00AE D3D4           DATA    &D3D4           *     L   - M                             *
199 00B0 D5D6           DATA    &D5D6           *     N   - O                             *
200 00B2 D7D8           DATA    &D7D8           *     P   - Q                             *
201 00B4 D9E2           DATA    &D9E2           *     R   - S                             *
202 00B6 E3E4           DATA    &E3E4           *     T   - U                             *
203 00B8 E5E6           DATA    &E5E6           *     V   - W                             *
```

```
204 00BA E7E8            DATA    &E7E8          *     X     - Y                 *
205 00BC E94A            DATA    &E94A          *     Z     - [                 *
206 00BE F05A            DATA    &F05A          *     /     - )                 *
207 00C0 5F60            DATA    &5F60          *     |     - -                 *
208 00C2 7981            DATA    &7981          *                               *
209 00C4 8283            DATA    &8283          *                               *
210 00C6 8485            DATA    &8485          *                               *
211 00C8 8687            DATA    &8687          *                               *
212 00CA 8889            DATA    &8889          *                               *
213 00CC 9192            DATA    &9192          *                               *
214 00CE 9394            DATA    &9394          *                               *
215 00D0 9596            DATA    &9596          *                               *
216 00D2 9798            DATA    &9798          *                               *
217 00D4 99A2            DATA    &99A2          *                               *
218 00D6 A3A4            DATA    &A3A4          *                               *
219 00D8 A5A6            DATA    &A5A6          *                               *
220 00DA A7A8            DATA    &A7A8          *                               *
221 00DC A9C0            DATA    &A9C0          *           - {                 *
222 00DE 6AD0            DATA    &6AD0          *     ~     - }                 *
223 00E0 A1FF            DATA    &A1FF          *     #     - DEL               *
224                *                           * * * * * * * * * * * * * * * * *
225                *     TABLE 'DVT' DES CONFIGURATIONS DE CONTROLE DES IT       *
226 00E2           DVT   EQU     $              *                               *
227                *-------------------------------------------------------------*
228                *     CONFIGURATION          RANG   TYPE                     *
229                *-------------------------------------------------------------*
230 00E2 E040            DATA    &E040          * 32   INEFFECTIVE              *
231 00E4 E040            DATA    &E040          * 31   INEFFECTIVE              *
232 00E6 E040            DATA    &E040          * 30   INEFFECTIVE              *
233 00E8 E040            DATA    &E040          * 29   INEFFECTIVE              *
234 00EA E040            DATA    &E040          * 28   INEFFECTIVE              *
235 00EC E040            DATA    &E040          * 27   INEFFECTIVE              *
236 00EE 6005            DATA    &6005          * 26   MINIDISQUE               *
237 00F0 E040            DATA    &E040          * 25   INEFFECTIVE              *
238 00F2 E040            DATA    &E040          * 24   INEFFECTIVE              *
239 00F4 E040            DATA    &E040          * 23   INEFFECTIVE              *
240 00F6 E040            DATA    &E040          * 22   INEFFECTIVE              *
241 00F8 E040            DATA    &E040          * 21   INEFFECTIVE              *
242 00FA E040            DATA    &E040          * 20   INEFFECTIVE              *
243 00FC E040            DATA    &E040          * 19   INEFFECTIVE              *
244 00FE E040            DATA    &E040          * 18   INEFFECTIVE              *
245 0100 E040            DATA    &E040          * 17   INEFFECTIVE              *
246 0102 E040            DATA    &E040          * 16   INEFFECTIVE              *
247 0104 E040            DATA    &E040          * 15   INEFFECTIVE              *
248 0106 6040            DATA    &6040          * 14   LECTEUR RAPIDE           *
249 0108 6080            DATA    &6080          * 13   PERFORATEUR RAPIDE        *
250 010A E040            DATA    &E040          * 12   INEFFECTIVE              *
251 010C E040            DATA    &E040          * 11   INEFFECTIVE              *
252 010E E040            DATA    &E040          * 10   INEFFECTIVE              *
253 0110 E040            DATA    &E040          *  9   INEFFECTIVE              *
254 0112 6011            DATA    &6011          *  8   IMPRIMANTE 300L/MN       *
255 0114 E040            DATA    &E040          *  7   INEFFECTIVE              *
256 0116 6008            DATA    &6008          *  6   ASR33 DE SERVICE         *
257 0118 6004            DATA    &6004          *  5   IT PUPITRE               *
258 011A E040            DATA    &E040          *  4   INEFFECTIVE              *
259 011C E040            DATA    &E040          *  3   INEFFECTIVE              *
260 011E E040            DATA    &E040          *  2   INEFFECTIVE              *
261 0120 E040            DATA    &E040          *  1   INEFFECTIVE              *
262 0122 E040            DATA    &E040          *  0   INEFFECTIVE              *
263                *-------------------------------------------------------------*
```

```
                        *
                        *
0124            ØLT     EQU     $
                        *
0124            ØLT1    EQU     $
                        *
0124            ØLTB    EQU     $
0124 0000               DATA    Ø
0126 1800               DATA    §1800
0128 2000               DATA    §2000
012A ØØ01               DATA    §1
012C ØØ01               DATA    §1
012E 1800               DATA    §1800
0130 2001               DATA    §2001
0132 ØØ01               DATA    §1
0134 ØØ01               DATA    §1
0136 ØØ01               DATA    §1
0138 1801               DATA    §1801
                        *
013A            ØCT     EQU     $
                        *
013A            ØCT1    EQU     $
                        *
013A            ØCT2    EQU     $
                        *
013A            IØT1    EQU     $
013A 0003               DATA    3
013C 0000               DATA    Ø
013E 0000               DATA    Ø
0140 0000               DATA    Ø
                        *
0142            IØT     EQU     $
0142   00               DATA,1  Ø
0143   00               DATA,1  Ø
0144   00               DATA,1  Ø
0145   00               DATA,1  Ø
                        *
```

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*  ØL  **          FONCTION                                    *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
** ØL  * ALPHA  * BINAIRE* ENTREE * SORTIE*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*  ØL  *     T     *     D     *     MV    *
*                                                               *
*  BI  *     2     *     Ø     *     Ø     *
*  BØ  *     2     *     Ø     *     Ø     *
*  CI  *     1     *     Ø     *     1     *
*  ØC  *     1     *     Ø     *     1     *
*  EI  *     2     *     Ø     *     Ø     *
*  EØ  *     2     *     Ø     *     1     *
*  LØ  *     1     *     Ø     *     1     *
*  LL  *     1     *     Ø     *     1     *
*  DØ  *     1     *     Ø     *     1     *
*  SI  *     2     *     Ø     *     1     *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*  T   *              ØRGANE                                   *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
** T ** ALPHA * BINAIRE * ENTREE * SORTIE *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                               *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*          BASE L DES HANDLERS                                  *
*                                                               *
* BASE L DE HTY000                                              *
* BASE L DE HPR000                                              *
* BASE L DE HPP000                                              *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*          ØCCUPATION DES HANDLERS                              *
*                                                               *
*                                                               *
*                                                               *
*                                                               *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
                        PAGE
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *                                             *
                        *       TABLES DE L'ANALYSEUR DE CØMMANDES    *
                        *              VERSIØN MINIMUM                *
                        *                                             *
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *                                             *
                        *     TABLE DES PØINTEURS DE TABLES DE M:ANAC *
                        *                                             *
0146 0000A      M:ANAC  DATA    A:CDES      * PØINTEUR LISTE DES TYPES DE CØMMANDES
0148 0000A              DATA    A:L         * PØINTEUR LISTE ARGUMENTS DE %LØAD
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *                                             *
                        *       TABLE DES TYPES DE CØMMANDES          *
                        *                                             *
                        * * * * * * * * * * * * * * * * * * * * * * *
014A   D3       A:CDES  TEXT    LR          * TYPES DE CØMMANDES (VERSIØN MINI)
014B   D9
014C   FF               DATA,1  §FF         * MARQUE FIN DE TABLE
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *                                             *
                        * TABLES DES ARGUMENTS DES DIFFERENTES CØMMANDES *
                        *                                             *
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *       %LØAD,A:Ø&XXXX,S                      *
                        *       %RUN                                  *
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *       ***     ARGUMENTS DE %LØAD     ***    *
                        * * * * * * * * * * * * * * * * * * * * * * *
014D   7C       A:L     TEXT    )S          *
014E   E2
                        * * * * * * * * * * * * * * * * * * * * * * *
                        *       ***     ARGUMENTS DE %RUN      ***    *
                        * * * * * * * * * * * * * * * * * * * * * * *
014F   FF               DATA,1  §FF         * MARQUE FIN DE TABLE
                        * * * * * * * * * * * * * * * * * * * * * * *
```

```
                      PAGE
              * * * * * * * * * * * * * * * * * * * * * * * * *
              *                                               *
              *          STRUCTURES TRAITEES PAR LE LOADER    *
              *                                               *
              * * * * * * * * * * * * * * * * * * * * * * * * *
              *                                 * * * * * * * * * * * * * * * * * * * * * *
              :ACTAM IDS     DUM                * EN TETE DU BLOC IMT                     *
     0000     CTRL   RES,1   1                  * CONTROLE &FF OU 7F POUR LE DERNIER      *
     0001     SEQ    RES,1   1                  * SEQUENCEMENT                            *
     0002     CSUM   RES,1   1                  * COMPLEMENT A 256 DE LA  CHECKUM DU BLOC *
     0003     LENR   RES,1   1                  * LONGUEUR UTILE DE L'ENREGISTREMENT      *
     0004     INFO   RES,1   104                *                                         *
              *                                 * * * * * * * * * * * * * * * * * * * * * *
                     FIN
              *                                 * * * * * * * * * * * * * * * * * * * * * *
              :ACTX  IDS     DUM                * BLOC FIN IMT                            *
     0000     M      RES     1                  * BIT 13 = 1 SI MODE MAITRE               *
     0002     NAPTG  RES     1                  * NOMBRE ADRESSES PTG                     *
     0004     NOSI   RES     1                  * NUMERO DE SECTION INITIALE              *
     0006     NSTIMT RES     1                  * NOMBRE DE SECTEURS IMT + PTG            *
     0008     G:CTX  RES     1                  * BASE G DU CONTEXTE                      *
     000A     L:CTX  RES     1                  * BASE L DU CONTEXTE                      *
     000C     P:CTX  RES     1                  * BASE P DU CONTEXTE                      *
     000E     NBS    RES,1   1                  * NOMBRE DE SECTIONS                      *
     000F     OV     RES,1   1                  * = 0 PAS D'OVERLAYS                      *
              *                                 * * * * * * * * * * * * * * * * * * * * * *
                     FIN
                     FIN
                     PAGE
              ***********************************************
              *                                             *
              *                   M:TRAP                    *
              *                                             *
              ***********************************************
              *          SECTION  0  DU SUPERVISEUR         *
              *          TRAITEMENT DES DEROUTEMENTS        *
              *                                             *
              *  INTERFACE :                                *
              *  -----------      (2) = TYPE DU DEROUTEMENT *
              *                   (4) = P-G     |           *
              *                   (6) = L-G     | PROGR.    *
              *                   (8) = INDICATEURS | DEROUTE *
              *                                             *
              *  APPEL :  AUTOMATIQUE SUR DEROUTEMENT       *
              *           LANCEMENT AVEC BASE L=0           *
              *                                             *
              *  SORTIE :     ABORT DU PROGRAMME AYANT      *
              *  --------     PROVOQUE LE DEROUTEMENT       *
              *                                             *
              *                                             *
              ***********************************************
                                           *                *
                                           *                *
              M:TRAP LPS    SUPER          *                *
                                           *                *
     0000 F408       STM                   *MASQUE PASC DE POSSIBILITE D INTRRUPTION
     0002 0002       LDA    TYDR           *ENTRE LE DER ET LE STM
     0004 2901       AND    =1             *ACC=0 SI DER E/S
     0006 5118       STA    #N3            *     =1 SI DER PROG
     0008 F902       LDR    =2             *
     000A 1704       ADM    4              *P ET L ABSOLU EN (4) ET (6)
     000C F902       LDR    =2             *
     000E 1706       ADM    6              *
     0010 6446       LEA    &K:DEUX        *DUMP DES CODES DEROUTEMTNTS
     0012 2108       LDE    =8             * P,L ET INDICATEURS
     0014 581A       SPA    #N2            *
     0016 D736       BRU    @P:EDIT        * EN CAS DE DEROUTEMENT  *
     0018 2000       LDA    =0             * DUMP DE    (G)    P-G  *
     001A 2106       LDE    =6             *            (G+2) L-G   *
     001C 581A       SPA    #N2            *            (G+4) INDIC *
     001E D736       BRU    @P:EDIT        * EN CAS DE DEROUTEMENT  *
                                           * APRES APPEL SUPERVISEUR*
     0020 F908       LDR    =8             * NIVEAU DU PROGRAMME *2 *
     0022 4D19       LBL    #N3+1          * CODE DEROUTEMENT = 0.  *
     0024 0000       CSV    M:EMED         *                        *
     0026 0000       CSV    M:EXIT         * EDITION DU MESSAGE     *
                                           *        DR NUMERO RC NL *
                                           * PUIS ABORT DU NIVEAU   *
                                           * AYANT PROVOQUE LE      *
                                           * DEROUTEMENT            *
              *                            **************************
                     FIN
```