Editeur Chargeur ECH
Editeur de liens EDL, EDL-E, EDL-EX, EDL-D, EDL-DX

manuel d'utilisation

SOMMAIRE

GENERALITES 1-1

Définitions 1-1

Organisation des fichiers binaires 1-1
Organisation sur support externe (non disque système) 1-1
Fichier IMT 1-4
Fichier BT 1-5

Organisation dans la zone GI-GO du disque système 1-6

Organisation dans les zones EP, SL et UL du disque système 1-7
Zone EP (programme IMT) 1-8
Zone SL ou UL (modules BT) 1-9

EDITEUR DE LIENS EDL
EDITEUR DE LIENS ETENDU EDL-E 2-1

Fonctions de l'éditeur de liens (EDL) et de l'éditeur de liens étendu (EDL-E) 2-1

Format de la commande 2-2

Messages d'erreur 2-3

EDITEUR DE LIENS ETENDU (EDL-EX) 3-1

Fonctions de l'éditeur de liens étendu 3-1

Format de la commande 3-2

Messages d'erreur 3-2

Etiquettes opérationnelles utilisées 3-4

EDITEUR DE LIENS DISQUE (EDL-D) 4-1

Fonctions de l'éditeur de liens disque (EDL-D) 4-1

Format de la commande 4-3
Format de la commande %TREE 4-4

Messages d'erreur 4-5

Etiquettes opérationnelles utilisées 4-7

4083 U4 FR
70 pages

SOMMAIRE (suite)

EDITEUR DE LIENS DISQUE ETENDU EDL-DX 5-1

Fonctions de l'éditeur de liens disque étendu (EDL-DX) 5-1
Format de la commande 5-3
Format de la commande %TREE 5-5
Messages d'erreur 5-6

EDITEUR-CHARGEUR ECH 6-1

Fonctions de l'éditeur-chargeur 6-1
Format de la commande (M:OC) 6-1
Messages d'erreur 6-2
Etiquettes opérationnelles utilisées 6-3

FORMATS BT et IMT A-1

Format BT A-1
En-tête sur support séquentiel A-1
En-tête sur disque A-1
Modules A-1
Format IMT A-2
Organisation logique du format IMT sur disque A-2
Description des unités logiques A-3
En-tête "programme" A-3
En-tête "données" A-3
Blocs programme (ou données, ou overlay) A-3
Fin de programme sans overlays A-4
Fin de programme avec overlays A-4
Fin d'overlay A-5
Blocs PTG A-6
Organisation logique du format IMT sur support séquentiel A-6
Structure générale A-6
Description des unités logiques A-8
En-tête programme A-8
En-tête données A-8
Blocs programme (ou données, ou overlays) A-8
Fin de programme sans overlays A-9
Fin de programme avec overlays A-9
Fin d'overlay A-10
Blocs PTG A-11
Table PTG A-11

ARTICLES BT B-1

Contrôle d'enregistrement B-1
Article 2 : Fin de segment (Hexa : 02) B-1
Article 3 : Fin de module (Hexa : 03) B-1
Article 4 : Déclaration de CDS (Hexa : 04) B-2

SOMMAIRE (suite et fin)

Article 5 : Déclaration de LDS (Hexa : 05) B-
Article 6 : Déclaration de LPS (Hexa : 06) B-
Article 7 : CALL (Hexa : 07) B-
Article 8 : Définition d'origine (Hexa : 08) B-
Article 9 : Chargement de valeur prédéfinie (Hexa : 09) B-
Article 10 : Chargement absolu (Hexa : 0A) B-
Article 11 : Définition de référence en avant (Hexa : 0B) B-4
Article 12 : Remplissage (Hexa : 0C) B-4
Article 13 : Définition de champ (Hexa : 0D) B-4
Article 14 : Addition de référence en avant (Hexa : 0E) B-4
Article 15 : Soustraction de référence en avant (Hexa : 0F) B-5
Article 16 : Addition de valeur prédéfinie (Hexa : 10) B-5
Article 17 : Soustraction de valeur prédéfinie (Hexa : 11) B-6
Article 18 : Fin d'expression (Hexa : 12) B-6
Article 19 : Positionnement du pool (Hexa : 13) B-6
Article 20 : Remplissage de pool (Hexa : 14) B-7
Article 21 : Référence externe (Hexa : 15) B-7
Article 22 : Définition externe (Hexa : 16) B-8
Article 23 : Déclaration de COMMON (Hexa : 17) B-8
Article 24 : Initialisation de COMMON (Hexa : 18) B-8
Article 25 : Fin d'initialisation du COMMON (Hexa : 19) B-9
Article 26 : Addition d'adresse de pointeur de COMMON (Hexa : 1A) B-9
Article 27 : Addition d'adresse de COMMON (Hexa : 1B) B-9
Article 28 : Positionnement du Pool (Hexa 1C) B-10
Article 29 : Signification des indicateurs I et J B-10

TRAITEMENT DES OVERLAYS C-1

Généralités C-1
Description C-1
Fonctionnement C-2
Déroulement des opérations C-2
Fonctions de la section M:C VLD C-3
Contenu du LDS de la section SHUNT C-3
Format de la table PRT2 C-4
Format de la table NUOV C-4
Tampon PTG C-5

1. Généralités

L'éditeur de liens (EDL), les éditeurs de liens étendus (EDL-E, EDL-EX), l'éditeur de liens disque (EDL-D) l'éditeur de liens disque étendu (EDL-DX) et l'éditeur chargeur (ECH) ont pour but de traiter des modules objets dans le format BT (Binaire Translatable) pouvant provenir d'assemblages ou de compilations séparés.

EDL, EDL-E, EDL-EX, EDL-D et EDL-X transforment les modules BT en un module IMT (Image Mémoire Translatable) acceptable par le chargeur du système.

ECH transforme les modules BT en un programme exécutable qu'il charge directement en mémoire (les modules BT que traite ECH sont normalement issus de MITRAS 1).

ECH, EDL et EDL-E, EDL-EX sont exploitables sous MOB et MTR.

EDL-D et EDL-DX sont exploitables sous MTRD et MMT.

Remarque :

Les normes de commandes des éditeurs de liens suivent les normes générales des commandes MITRA 15. Celles-ci sont décrites dans les manuels d'utilisation des moniteurs.

1-1. DEFINITIONS

Un enregistrement binaire sur support séquentiel a une longueur de 120 octets. Sur disque il a une longueur de 120 octets pour le format BT et de 256 octets pour le format IMT.

Un module BT est un binaire issu d'un seul assemblage ou compilation, et est composé d'enregistrements binaires.

Un module IMT est un binaire issu d'une seule édition de liens, et est composé d'enregistrements binaires.

Un fichier binaire est composé d'une suite d'enregistrements binaires. Il comprend un ou plusieurs modules. Il est terminé par une marque fin de fichier.

Pour le support bande magnétique, la marque fin de fichier est représentée par l'enregistrement particulier à ce type de support, le "tape mark".

Pour les autres supports, la marque fin de fichier est représentée par un enregistrement binaire dont les quatre premiers caractères sont %EOD, en code EBCDIC ou ASCII suivant les supports.

1-2. ORGANISATION DES FICHIERS BINAIRES

Les fichiers BT ou IMT peuvent avoir trois organisations distinctes :

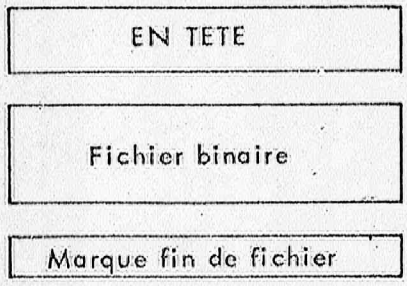
- sur un support externe (non disque système)
- dans la zone GI-GO du disque système
- dans l'une des zones EP, SL ou UL du disque système.

1-2.1. Organisation sur support externe (non disque système)

Généralités

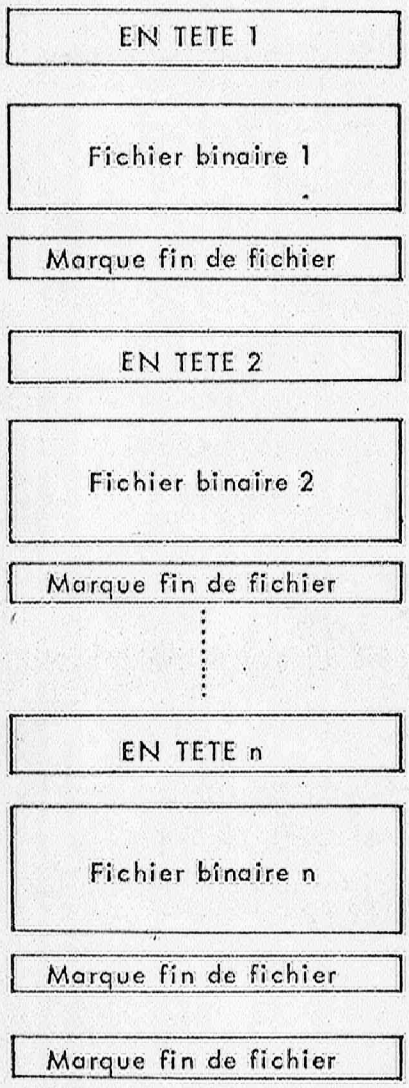
Un fichier binaire sur support externe se présente soit seul soit incorporé à un ensemble de fichiers. Cet ensemble possède une organisation sous forme de bibliothèque externe.

- Fichier binaire seul



Un fichier binaire seul est précédé d'un en-tête et suivi d'une marque fin de fichier.

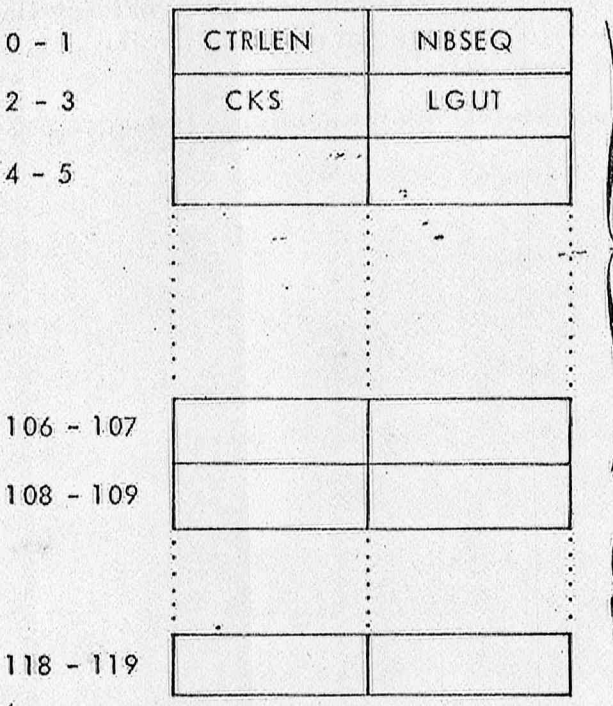
- Bibliothèque externe



Une bibliothèque externe est composé d'un ou plusieurs fichiers binaires. Chacun d'eux est précédé de son en-tête et suivi d'une marque fin de fichier. Le dernier fichier de la bibliothèque est suivi de deux marques fin de fichier consécutives.

Chacun des enregistrements constituant un fichier binaire possède en tête quatre octets de contrôle.

D'autre part les douze derniers octets sont réservés pour un commentaire éventuel (séquencement en alpha-numérique par exemple).



CTRLEN = contrôle d'enregistrement
Format BT = &81 enregistrement "en-tê
dernier enregistrement
de l'unité logique.
= &01 premier enregistremen
enregistrement couran
Format IMT = &7F enregistrement "en-tê
enregistrement fin de
bloc logique.
= &FF premier enregistrement
enregistrement courant

NBSEQ = numéro binaire de séquence de l'enregistrement (modulo 256).

CKS = somme de contrôle

LGUT = longueur utile de l'enregistrement

Pour le format IMT, la notion de premier ou de dernier enregistrement physique ainsi que celle de séquencement ne sont valables que pour un bloc logique. Un bloc logique est un bloc programme, un bloc PTG, un bloc overlay...

Exemple :

EN-TETE	Séquencement = 0 Contrôle d'enregistrement = &7F
BLOC PROGRAMME	Séquencement de 0 à n Contrôle d'enregistrement = FF pour les enregistrements courants 7F pour le dernier enregistrement
BLOC PTG DU PROGRAMME	Séquencement de 0 à n Contrôle d'enregistrement : FF pour les enregistrements courants 7F pour le dernier enregistrement
PREMIER BLOC OVERLAY	Séquencement de 0 à n Contrôle d'enregistrement : FF pour les enregistrements courants 7F pour le dernier enregistrement
BLOC PTG DE L'OVERLAY	Séquencement de 0 à n Contrôle d'enregistrement : FF pour les enregistrements courants 7F pour le dernier enregistrement
DERNIER BLOC OVERLAY	Séquencement de 0 à n Contrôle d'enregistrement : FF pour les enregistrements courants 7F pour le dernier enregistrement
BLOC PTG DE L'OVERLAY	Séquencement de 0 à n Contrôle d'enregistrement : FF pour les enregistrements courants 7F pour le dernier enregistrement

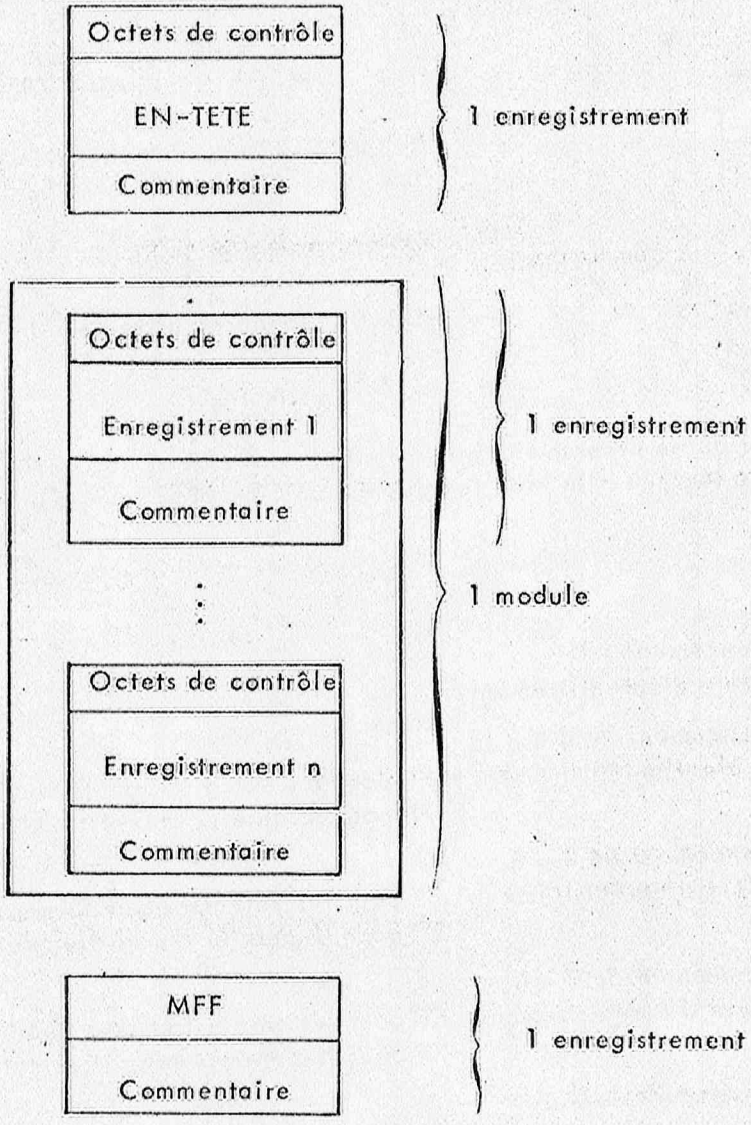
Pour les notions de blocs programmes, blocs PTG de blocs overlays, se reporter à l'annexe A.

La somme de contrôle est égale au complément à deux de la somme sans report de tous les octets utiles de l'enregistrement, non compris celui de la somme de contrôle. L'addition sans report de tous les octets de l'enregistrement (y compris celui de la somme de contrôle) donne zéro.

Le premier enregistrement physique d'un fichier contient un "en-tête" de 14 octets qui l'identifie. Cet "en-tête" est la première unité logique du fichier.

■ Fichier IMT

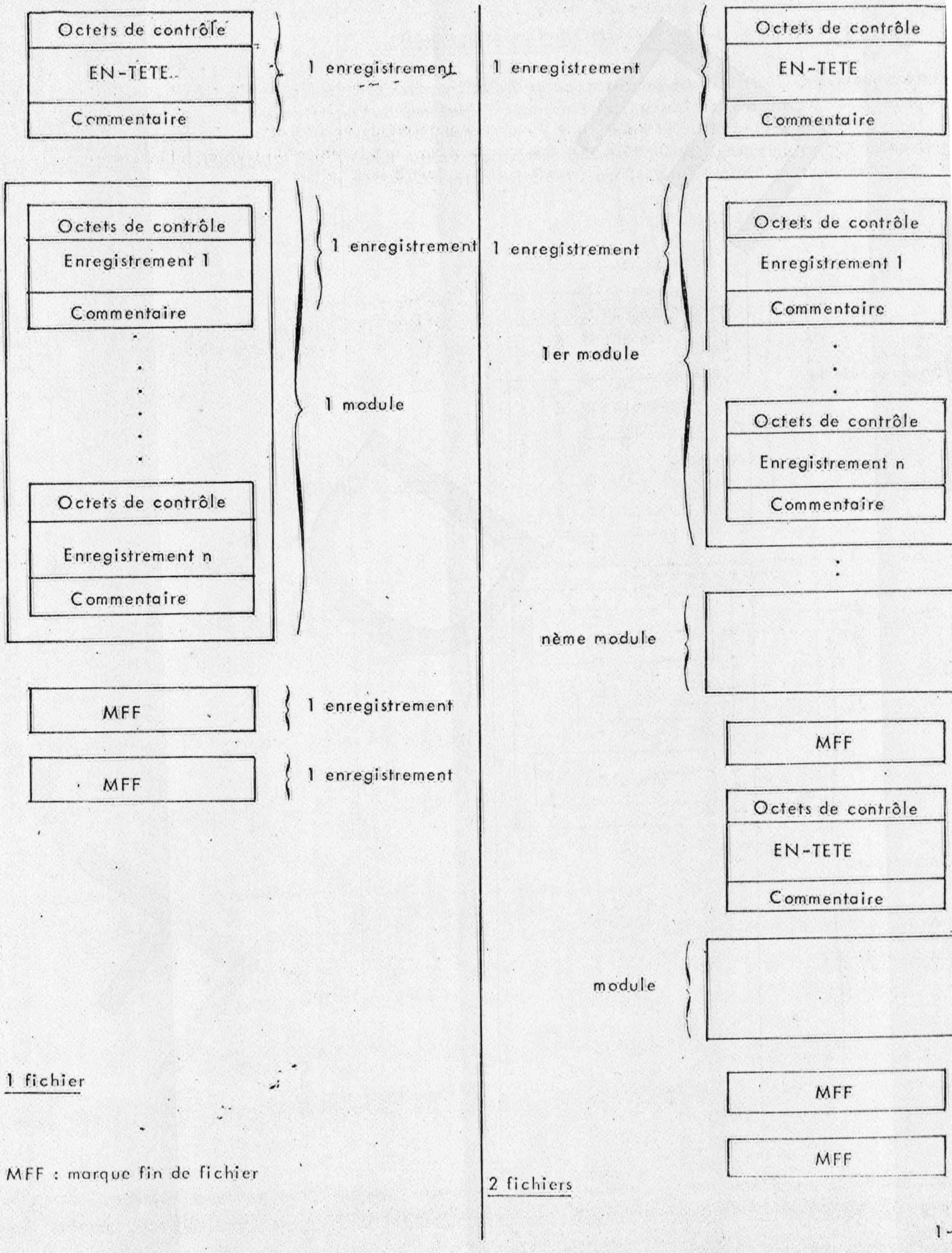
Les fichiers IMT sont sortis par les Editeurs de liens.



MFF : marque fin de fichier

Fichier BT

Les fichiers BT sont sortis par les Assembleurs et Compilateurs, ou entrés par les Editeurs de liens et l'Editeur-chargeur.



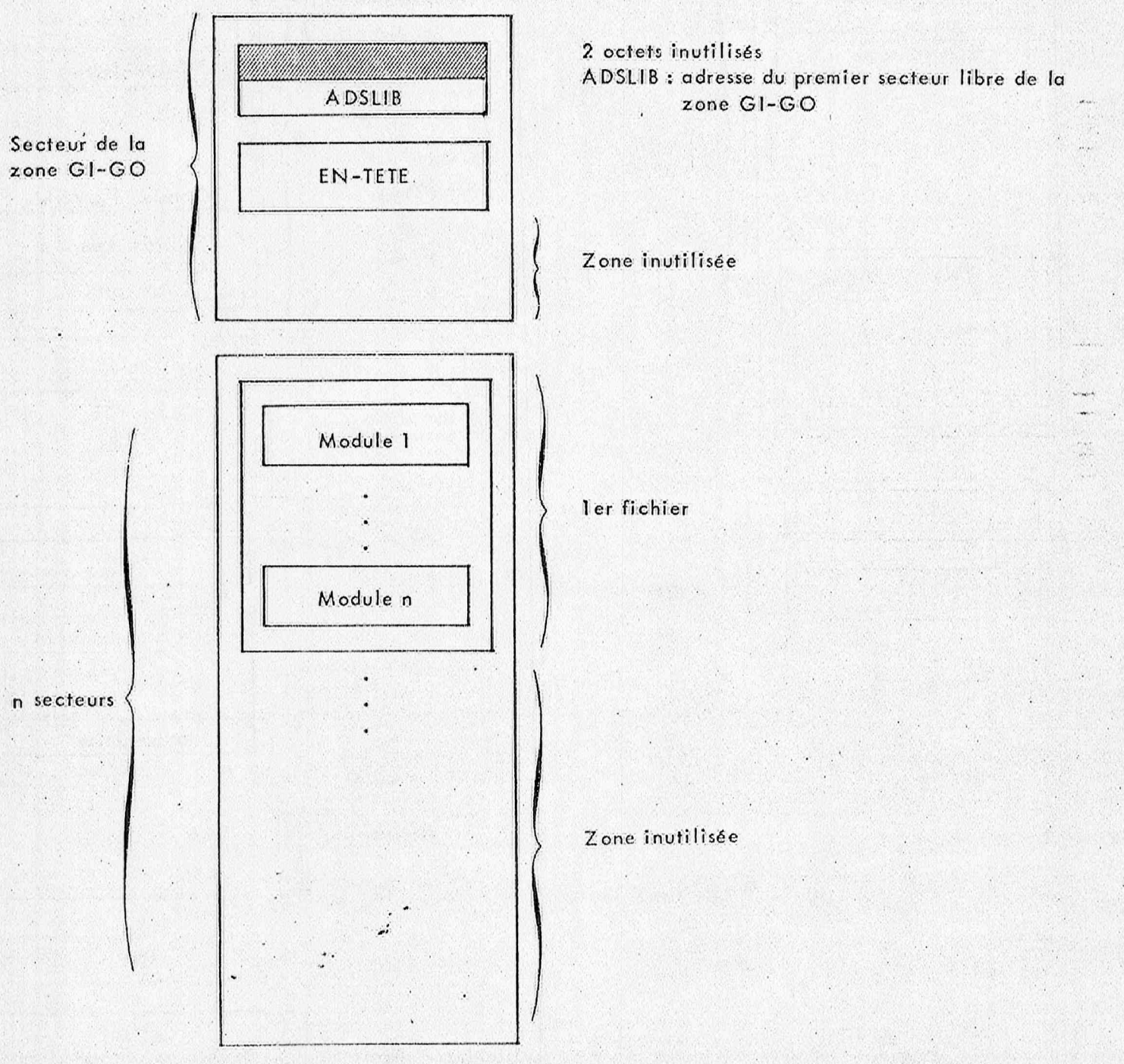
Remarque :

En règle générale, un fichier est constitué d'un seul module. Le regroupement peut se faire par l'option P du programme COPY.

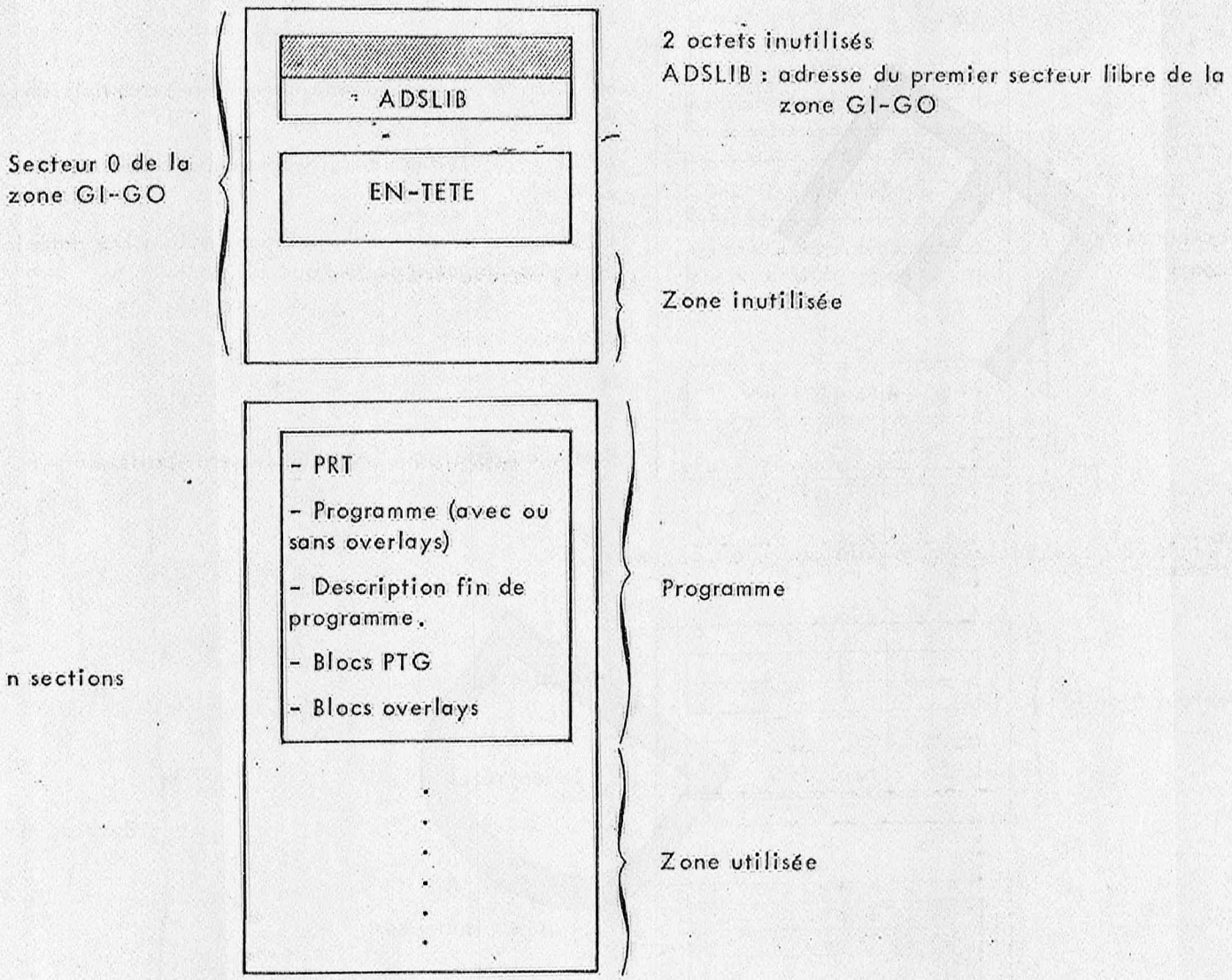
1-2.2. Organisation dans la zone GI-GO du disque système

La zone GI-GO est utilisée comme zone de travail par les processeurs et comme zone de liaison entre les processeurs. Dans ce dernier cas, elle ne peut contenir qu'un seul fichier BT ou IMT dont l'en-tête se trouve dans le secteur 0 de la zone. Il y a deux enregistrements de 120 octets par secteur dans le cas d'un fichier BT, et un enregistrement de 256 octets par secteur dans le cas d'un fichier IMT. Les quatre premiers octets du secteur 0 sont réservés, mais seuls les octets 2 et 3 sont utilisés.

- Cas d'un fichier BT



- Cas d'un fichier IMT

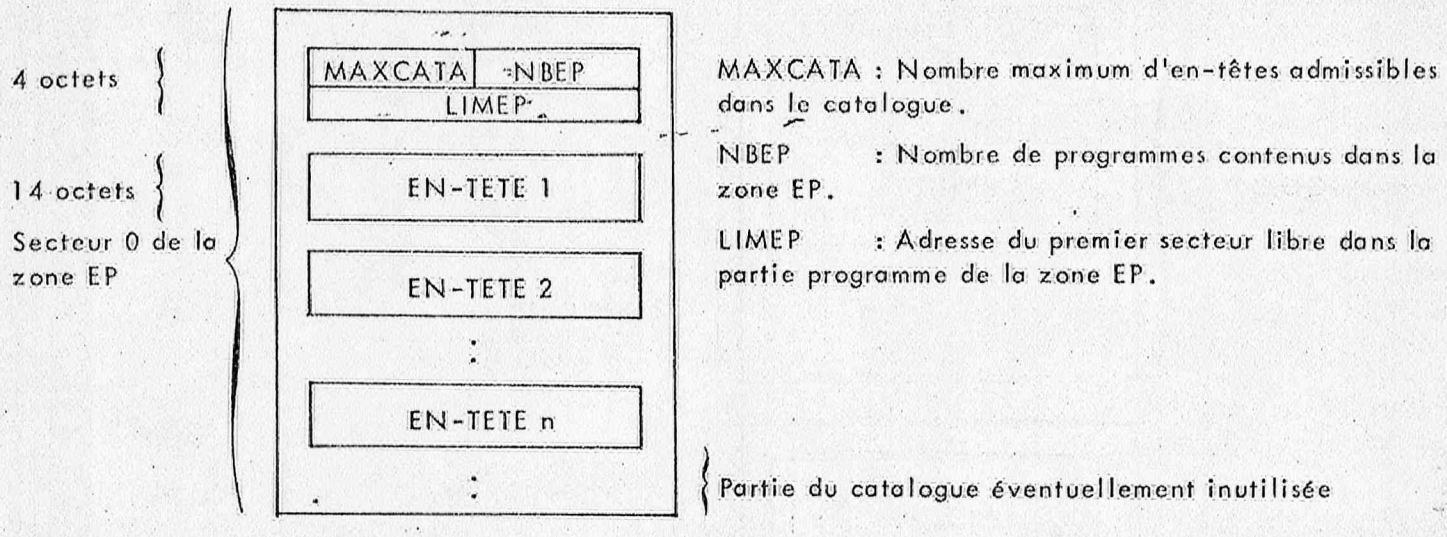


1-2.3. Organisation dans les zones EP, SL et UL du disque système

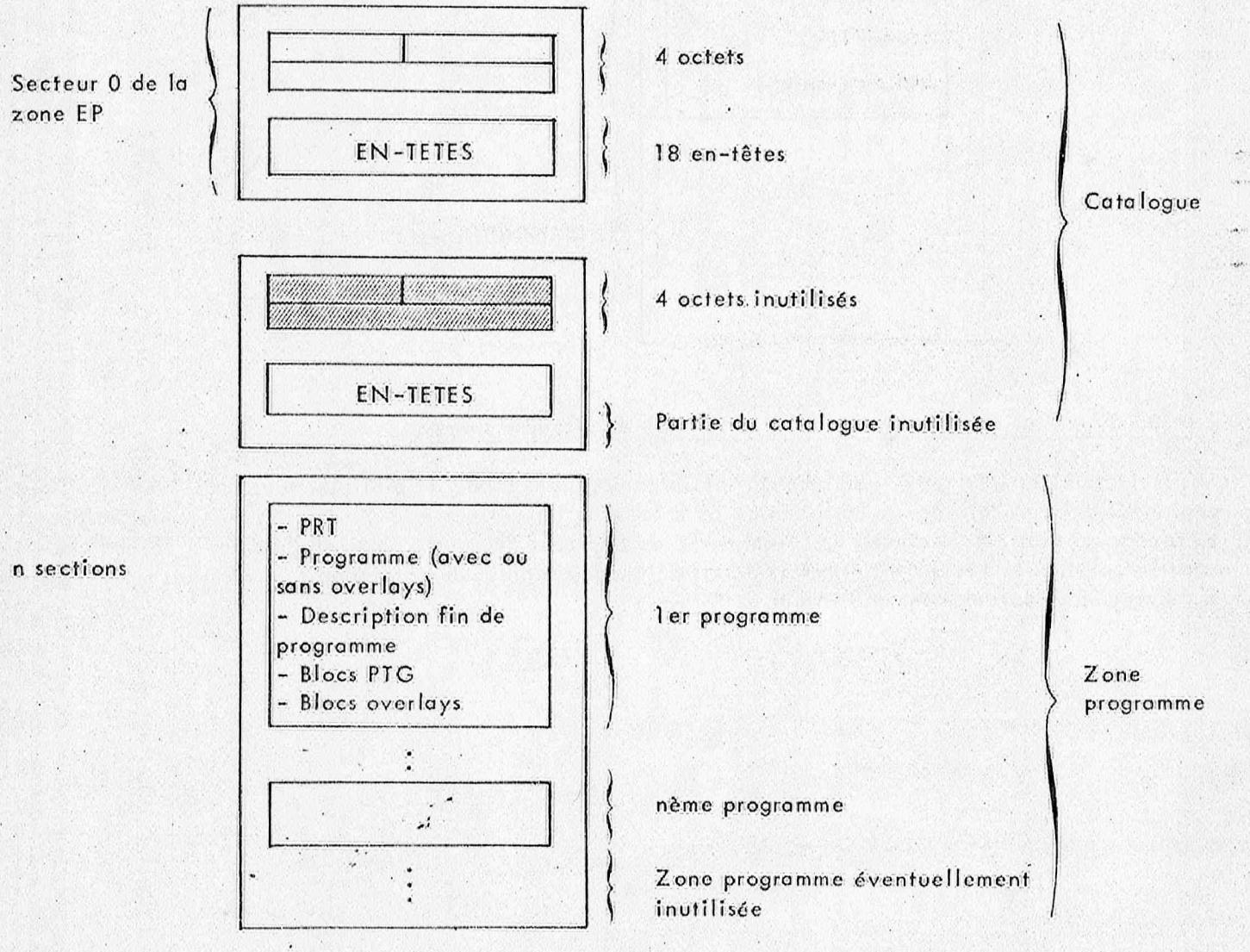
Les fichiers BT et IMT sur disque système sont organisés sous forme de bibliothèques. Les "en-têtes" sont regroupés dans un catalogue. La taille de ce catalogue est définie à la génération du système et peut s'étendre sur plusieurs secteurs. Les fichiers BT ou IMT sont chargés à partir du premier secteur libre suivant le catalogue. Les quatre premiers octets de chaque secteur du catalogue sont réservés mais seuls ceux du secteur 0 de la zone ont une valeur définie.

Zone EP (programme IMT)

- Catalogue de la zone EP : exemple de catalogue sur 1 secteur

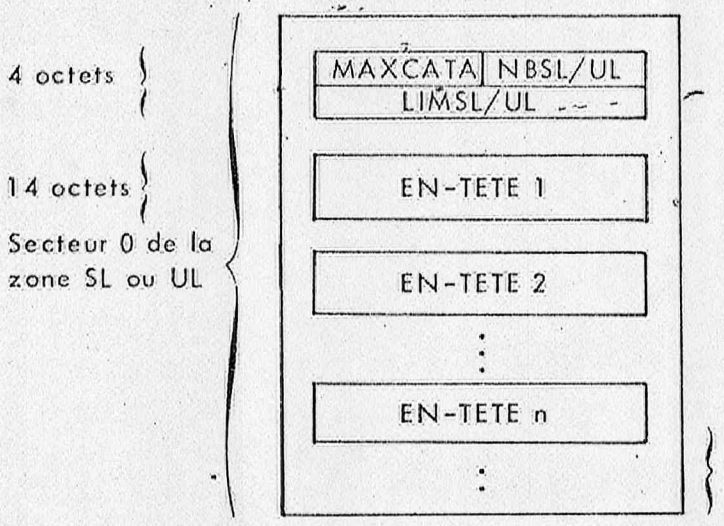


- Zone EP : exemple avec un catalogue de n secteurs

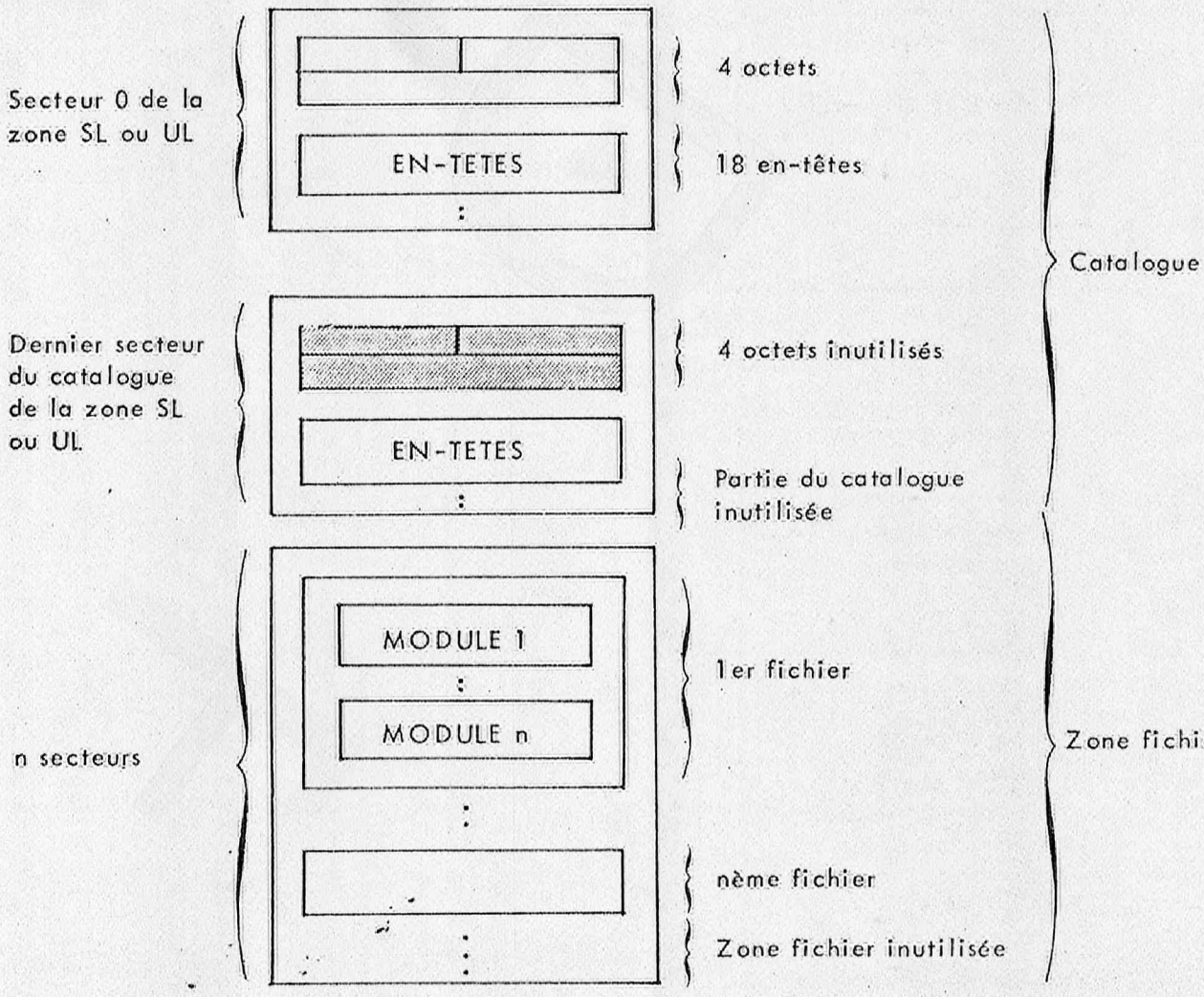


■ Zone SL ou UL (Modules BT)

- Catalogue : exemple avec un catalogue sur 1 secteur



- Description de la zone SL ou UL : exemple avec un catalogue de n secteur



Remarque :

En règle générale, un fichier est constitué d'un seul module. Le regroupement peut se faire par la commande FETCH du BIBLIOTHECAIRE.

5. Editeur de liens disque étendu (EDL-DX)

Il est destiné aux systèmes comportant un disque et l'option 1529 ou 1539.

L'Editeur de liens disque étendu traite le BT issu des Assembleurs, du LP15 et du Fortran.

5-1 FONCTIONS DE L'EDITEUR DE LIENS DISQUE ETENDU (EDL-DX)

- Lecture des modules BT à éditer (sur M:BI ou M:GI).

- Recherche éventuelle dans le catalogue de UL ou SL des modules appelés par CLS dans le BT déjà lu. Ils sont ensuite rajoutés derrière le BT d'origine.

- Modification éventuelle de la table standard des sous-programmes communs.

- Dans le cas d'un programme ayant une structure d'overlays, recherche dans le catalogue de UL ou SL des modules décrits dans une commande "TREE" de l'éditeur de liens disque. Ils sont ensuite rajoutés derrière le BT d'origine.

- Calcul des bases des segments.

- Calcul et mise en place des références en avant.

- Calcul des expressions.

- Calcul et mise en place des références et définitions externes.

- Choix entre CLS ou CSV pour les appels de section.

- Détermination de la PRT du module de programme IMT.

- Traitement des articles de "pool".

- Traitement des "common" Fortran.

- Production de programmes IMT ayant une structure d'overlays.

. calcul des bases des segments, suivant l'implantation en mémoire des branches d'overlays auxquelles ils appartiennent.

. génération derrière la racine du programme, d'une section appelée "SHUNT", capable de charger une branche d'overlays en mémoire, lors de l'appel d'une section non présente en mémoire.

. génération derrière le bloc fin de programme, d'une table d'overlays donnant l'adresse secteur sur disque l'adresse d'implantation en mémoire ainsi que la longueur de chaque branche d'overlay.

- Edition suivant les options demandées par l'utilisateur de :

. la table d'implantation des segments,

. la liste des références non satisfaites,

. la liste des sous-programmes communs appelés (utilisation de CSV).

Remarque :

Quand l'éditeur de liens disque étendu effectue une recherche en bibliothèque UL ou SL, c'est le nom présent dans le catalogue qui est utilisé pour satisfaire les CALL SECTION. On ne pourra donc pas utiliser la fonction RENAME du bibliothécaire sur des fichiers BT destinés à être recherchés automatiquement en bibliothèque EDL-DX.

L'Editeur de liens disque étendu travaille en deux passes. La première lui permet essentiellement de :

- Transférer les modules BT depuis le support séquentiel assigné à l'étiquette opérationnelle M:BI vers la zone GI-GO du disque.

- D'effectuer deux phases préliminaires lui permettant :

. dans le cas d'une structure sans overlays, de rechercher dans les bibliothèques SL ou UL les modules BT absents et de les transférer dans la zone GI-GO du disque à la suite du BT d'origine.

. dans le cas d'une structure avec overlays, de rassembler dans la zone GI-GO les modules BT dans un ordre compatible avec la description de l'arbre d'overlays et de rechercher les sections manquantes dans les bibliothèques SL ou UL.

- De déterminer l'implantation des sections.

- De calculer les valeurs à affecter aux références.

La deuxième passe lui permet de générer le programme IMT.

- Dans le cas d'une structure sans overlays, la composition est la suivante :

. PRT
. Programme
. Bloc fin de programme
. PTG (en mode maître).

- Dans le cas d'une structure avec overlays, la composition est la suivante :

. PRT (les SRD des sections appartenant à une branche d'overlays sont remplacés par la SRD de la section "SHUNT").
. Racine
. Section SHUNT
. Bloc fin de programme
. Table d'overlays
. PTG de la racine (en mode maître)
. Première branche d'overlay
. PTG associée (en mode maître)
⋮
. nième branche d'overlay
. PTG associée (en mode maître).

Lorsque l'éditeur de liens étendu rencontre une instruction d'appel de section (CLS ou CSV), il effectue tout d'abord une recherche de la section dans la table des sections superviseur et, en cas de succès, génère un CSV. Sinon, il effectue une recherche dans la table des sections utilisateur et, en cas de succès, génère un CLS.

A la fin de la seconde passe, suivant les options demandées par l'utilisateur, l'éditeur de liens étendu sort sur M:LL la table d'implantation des segments, la liste des références non satisfaites et la liste des sous-programmes communs appelés, selon le format suivant :

TABLE D'IMPLANTATION DES SEGMENTS

```
AAA   XXXXXX          NNNN
BBB   YYYYYY          MMMM
 ⋮      ⋮              ⋮
CCC   ZZZZZZ          PPPP
```

↑ nom du segment — └ Adresse relative à G du début du segment exprimée en hexadécimal

Numéro de section pour les LPS exprimé en décimal

LISTE DES COMMUNS UTILISES

```
      XXXXXX          NNNN
      YYYYYY          MMMM
        ⋮              ⋮
      ZZZZZZ          PPPP
```

nom du commun — └ Adresse relative à G du début du commun exprimée en hexadécimal

REFERENCES NON SATISFAITES

XXXXXX
YYYYYY
:
ZZZZZZ

LISTE DES S/P COMMUNS APPELES

M:XXXX
M:YYYY
:
M:ZZZZ

Remarque :

Le fonctionnement de la section "SHUNT" et des overlays est décrit dans l'Annexe D.

5-2 FORMAT DE LA COMMANDE

Dans le cas d'une utilisation sous MTRD, il est possible de lancer l'Editeur de liens disque étendu par %CALL/LINKDX/paramètres.

%C[ALL] LINKDX/[SM / MM / DA], [PM], [XXXXXX.[NN]], [BI / GI], [BO / GO], [EI / CI], [UL], [SL], [OV], [SG], [FR], [CL], [C

Signification des paramètres :

DA — Modules de données

MM — Programme à exécuter en mode maître

SM — Programme à exécuter en mode esclave.

Dans le cas où plusieurs paramètres de ce type sont spécifiées dans la commande, le dernier est pris en compte ; quand le paramètre est absent, SM est pris par défaut.

PM — Programme ayant accès aux zones mémoires protégées. Quand le paramètre est absent, le programme n'a pas accès aux zones mémoires protégées.

XXXXXX — Nom du programme qui figurera dans l'en-tête de l'IMT. Ce nom comporte de 1 à 6 caractères alphanumériques. Quand le paramètre est absent, le nom pris par défaut est composé de six blancs.

.NN — Deux chiffres décimaux indiquant le numéro de version de l'IMT produit. Quand le paramètre est absent, le numéro pris par défaut est : 00.

BI — BT entré sur support séquentiel assigné à M:BI.

GI — BT entré dans la zone GI-GO du disque.

Quand le paramètre est absent, GI est pris par défaut.

BO — IMT sortie sur support séquentiel assigné à M:BO.

GO — IMT sortie dans la zone GI-GO du disque.

L'IMT est toujours sortie dans la zone GI-GO du disque. Cependant si le paramètre BO est présent, l'IMT est également sortie sur le support séquentiel assigné à M:BO.

EI — Commandes %TREE lues sur le périphérique assigné à M:EI (M:EI peut être assignée à un fich SGF15).

CI — Commandes %TREE lues sur le périphérique assigné à M:CI, quand le paramètre est absent, CI est pris par défaut.

UL — Recherche des modules BT manquants dans la bibliothèque UL.

SL : Recherche des modules BT manquants dans la bibliothèque SL.

Ces modules doivent être décrits dans la commande %TREE de telle manière qu'ils puissent être ajouter à la suite du BT se trouvant sur M:G1.

Dans le cas où les deux paramètres sont indiqués, la recherche commence par la bibliothèque UL. Si ni l'un ni l'autre n'est indiqué, aucune recherche n'est effectuée.

OV : Donne une structure d'overlays au programme IMT généré.

Si le paramètre est absent, le programme IMT généré aura une structure linéaire.

,SG : Demande d'édition de la table d'implantation des segments.

,FR : Demande d'édition des références non satisfaites.

,CL : Demande d'édition des noms de sous-programmes communs appelés.

Si aucun paramètre n'est présent, SG, FR, CL sont tous les trois implicites et le caractère / en fin de commande est obligatoire.

,CV : Demande de modification de la table standard des sous-programmes communs.
Si le paramètre CV est absent, la table standard est retenue.

■ Format de la commande %CSV

Dans le cas où le paramètre CV de la commande "LINKDX" est présent, l'éditeur de liens lit sur M:C1 la la commande %CSV. Celle-ci sert à modifier la table des sous-programmes communs.

Toutes les commandes %CSV doivent être regroupées immédiatement avant les commandes %TREE.

$$\%CSV/\left\{\begin{matrix}M:\\S:\end{matrix}\right\}\ X[XXX]=N[NN]\ \left\{\begin{matrix}/\\,\\;\end{matrix}\right\}\ldots\ldots$$

$\left\{\begin{matrix}M:\\S:\end{matrix}\right\}$ indique s'il s'agit d'un module M: ou S:

X[XXX] nom du module sur 1 à 4 caractères alphanumériques

= sépare le nom du module de son numéro

N[NN] numéro du module sur 1 à 3 caractères décimaux, on doit avoir $1 \leqslant NNN \leqslant 255$

$\left\{\begin{matrix}/\\,\\;\end{matrix}\right\}$ / fin de commande %CSV
, sépare deux redéfinitions de modules
; caractère de suite.

Remarques :

Tous les modules M: possédant dans la table un numéro supérieur au plus grand numéro de module M: se trouvant dans la commande %CSV, sont supprimés.

Tous les modules S: possédant dans la table un numéro supérieur au plus grand numéro de module S: se trouvant dans la commande %CSV, sont supprimés.

Exemple :

```
%CSV/M:WAIT = 4, S:KILL = 12 ;
%CSV/S:UTI = 15, M:BNDC = 8.
```

Le module M:WAIT prend le numéro 4 dans la table M:
Le module S:KILL prend le numéro 12 dans la table S:
Le nouveau module S:UTI est inséré dans la table S: et prend le numéro 15
Le module M:BNDC garde le numéro 8.

Le numéro maximum de la section M: décrite étant 8, tous les modules M: ayant un numéro supérieur à 8 sont supprimés.

Le numéro maximum de la section S: étant 15, tous les modules S: ayant un numéro supérieur à 15 sont supprimés.

■ Format de la commande %TREE

Dans le cas où le paramètre OV de la commande "LINKD1" est présent, l'éditeur de liens disque étendu lit sur M:C1 ou M:E1 la commande %TREE. Celle-ci sert à décrire l'arbre d'overlays qui représente la structure de recouvrement du programme IMT généré.

Dans le cas où les étiquettes opérationnelles M:C1 et M:B1 sont assignées au même support, toutes les commandes "TREE" doivent être regroupées immédiatement derrière la commande LINKDX les modules BT étant regroupés à la suite des commandes "TREE".

La commande est décrite ci-dessous selon la formulation de Backus :

%TREE/ ARBRE

où ARBRE est définit ainsi :

<ARBRE>: : =<DB> <DB>(<DN>)

<DN> : : =<ARBRE> <DN>,<ARBRE>

<DB> : : =<NOM> <DB>+<NOM>

DB : description de branche

DN : description de noeud

NOM : nom de section (nom du premier LPS du module)

L'éditeur de liens disque étendu détecte la fin de la commande soit par le caractère "Retour chariot", soit par le caractère "blanc".

Un caractère "point virgule" signifie que la commande continue sur l'enregistrement suivant qui commence également par le libellé "%TREE/".

Les opérateurs servant à décrire l'arbre sont au nombre de quatre.

+ indique la poursuite de la même branche.

(indique la rencontre d'un noeud de l'arbre.

) indique que pour la branche suivante, on doit remonter au noeud précédent.

, indique qu'il s'agit d'une branche parallèle à la précédente.

Il ne peut y avoir deux opérateurs qui se suivent, sauf si le premier est un ")".

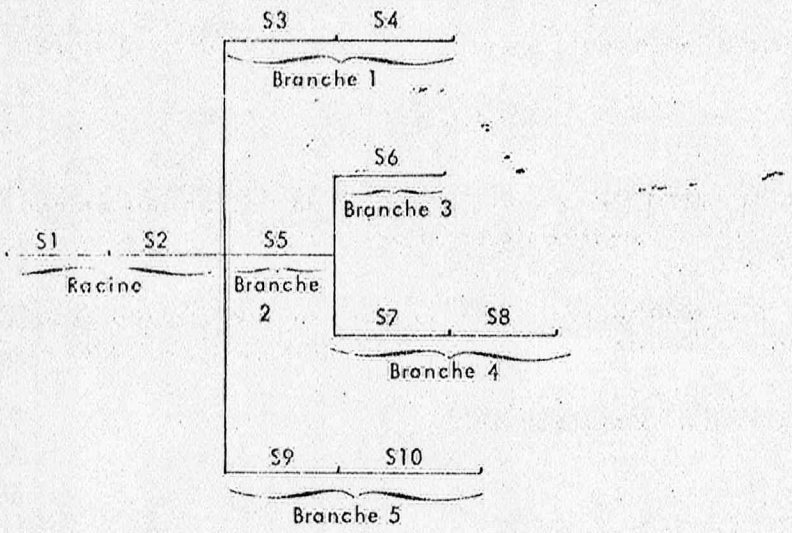
Exemple :

%TREE/S1+S2(S3+S4,S5(S6;

%TREE/,S7+S8),S9+S10)

où S1, S2, S3.....S10 sont des noms de section (nom du LPS).

La structure de l'arbre est alors la suivante :



Remarques :

L'ordre d'apparition des sections dans le BT doit être identique à celui de la commande.
La section initiale doit se trouver dans la racine.
Dans le cas d'un programme ayant une structure d'overlay, il est nécessaire de réserver un TWB étendu (32 mots).

5-3. MESSAGES D'ERREUR

Quand l'Editeur de liens disque étendu détecte une erreur, il édite un message du type :

ERREUR NO XY

sur M:DO où X et Y sont deux chiffres décimaux. Le premier chiffre X constitue le niveau de sévérité de l'erreur. Si le niveau de sévérité d'une erreur est supérieur à trois, l'édition de liens est abandonnée, sinon l'éditeur de liens disque prend une option et poursuit.

Dans le cas d'erreurs détectées lors de la deuxième passe, l'éditeur de liens disque indique également le nom du segment et l'adresse (par rapport au début du segment) à laquelle s'est produite l'erreur.

Dans le cas :

- de l'erreur 53, le nom du segment est édité.
- de l'erreur 29, le nom de la référence est édité.
- de l'erreur 30, le nom du LPS est édité.

Dans le tableau ci-dessous, la phase où est détectée l'erreur est signalée dans la colonne "type" par les codes suivants :

- CL Erreur détectée au décodage de la commande LINKDX
- CT Erreur détectée au décodage de la commande TREE.
- P1 Erreur détectée à la première passe.
- P2 Erreur détectée à la deuxième passe.

Numéro	Signification	Type
10	Absence de section de lancement en fin de module. La dernière section du module est prise comme section de lancement.	P1
11	Section de lancement multi-définie, la première spécifiée est prise comme section de lancement.	P1
13	La taille d'un "common" Fortran est différente entre plusieurs modules. La plus grande taille est prise en compte.	P1
20	Référence non définie. Elle est forcée à zéro.	P1
21	Référence externe non définie. Elle est forcée à zéro.	P1
22	Indicateur I incorrect. On prend I = 0.	P2
23	CLS ou CSV non satisfait. Génération d'un module.	P2
24	Erreur dans une définition de champ. La valeur est tronquée.	P2
25	Addition de deux valeurs translatables (adresses) relatives à la même base.	P2
26	Soustraction de deux valeurs translatables (adresses) relatives à des bases différentes Génération d'une valeur nulle.	P2
27	Expression non homogène. Génération d'une valeur nulle.	P2
28	Débordement de champ. La valeur est tronquée.	P2
29	Double définition de référence externe. La deuxième est ignorée. Edition du nom de la référence.	P1
30	Absence du segment de données locales associé à un LPS. Le précédent LDS le plus proche lui est associé. (Edition du nom du LPS).	P1
31	"Common" Fortran non défini.	P2
33	Pointeur mot contenant une adresse impaire.	P2
40	Erreur de format du binaire. Format non BT.	P1
41	Format incorrect de la commande TREE (erreur de libellé %TREE/, nom de section de plus de 6 caractères, opérateur en suivant un autre différent de ")", parenthèses droite et gauche non équilibrées.	CT
42	Format incorrect de la commande LINKD1 (absence de virgule ou de parenthèse, nom de programme de plus de 6 caractères, numéro de version ne comportant pas deux chiffres).	CL
43	Option inexistante.	CL
44	Fichier GI vide ou ne contenant pas de binaire du format BT.	P1
45	Section de lancement n'appartenant pas à la racine.	P1
46	Erreur somme de contrôle, due, soit à une erreur d'écriture lors de la production du BT, soit à une erreur de lecture à l'édition de liens.	P1
47	Erreur de séquencement. Les enregistrements binaires sont déclassés.	P1
48	Une déclaration de CDS a été rencontré et elle ne figure pas en tête du premier module.	P1
49	Débordement des tables de l'éditeur de liens.	P1
50	Niveau de sévérité égal ou supérieur à trois dans un module BT.	P1

Numéro	Signification	Type
52	Tentative d'édition de modules ne comportant aucun LPS, en l'absence de l'option DA.	P1
53	Double définition de segment-Edition du nom de segment	P1
54	Le nombre de segments dans le programme est supérieur à 255.	P1
55	OPEN erroné lors de la lecture des commandes %TREE sur un fichier SGF15 assigné à M:EI.	P1
56	CLOS erroné lors de la lecture des commandes %TREE sur un fichier SGF15 assigné à M:EI.	P1
57	Nombre de branches d'overlay supérieur à 85.	P1
58	Programme dont la taille mémoire est supérieure à &FFFF.	P1
59	Sous-programme FORTRAN utilisant des communs et rattaché à un programme principal générant des articles de pool de CDS.	P1
60	Erreur commande TREE. Section dans le désordre. Edition du nom de la section.	P1
61	Erreur commande TREE. Section se trouvant dans la zone GI-GO et non décrite dans la commande. Edition du nom de la section.	P1
62	Erreur commande TREE. Section non décrite dans la commande et se trouvant dans un module en bibliothèque SL ou UL dont la première section est décrite dans la commande. Edition du nom de la section.	P1
63	Erreur commande TREE. Section décrite dans la commande et ne se trouvant ni dans la zone GI-GO, ni en bibliothèque SL ou UL. Edition du nom de la section.	P1
64	Erreur commande TREE. Nom de CDS ou de LDS se trouvant dans la commande. Edition du nom de la section.	P1
65	Erreur commande TREE. Tentative d'inclusion d'un module, se trouvant en bibliothèque SL ou UL, à l'intérieur d'un module BT se trouvant en zone GI-GO. Edition du nom de la section.	P1
66	Erreur commande TREE. Description d'arbre erronée.	P1
70	Erreur commande CSV. Erreur de syntaxe.	P1
71	Erreur commande CSV. Erreur de numéro.	P1

5-4. ETIQUETTES OPERATIONNELLES UTILISEES

M:BI	Entrée des fichiers BT (suivant option)
M:BO	Sortie du programme IMT (suivant option)
M:OC	Dialogue opérateur (EDL-DX lancé par %LOAD-%RUN)
M:CI	Lecture de la commande TREE
M:EI	Lecture de la commande TREE
M:LL	Listes et tables
M:DO	Messages d'erreur
M:GI	Entrée des fichiers BT
M:GO	Sortie du programme IMT
M:UL	Recherche dans la bibliothèque UL
M:SL	Recherche dans la bibliothèque SL

Annexe A – Formats BT et IM

A-1. FORMAT BT

Le format "Binaire Translatable" (BT) est constitué d'un certain nombre d'unités logiques de longueur variable qui sont :

- l'en-tête
- le ou les modules
- la marque fin de fichier

Sur le disque, il y a deux unités physiques consécutives de 120 octets par secteur.

Sur un support séquentiel, les unités logiques se présentent chacune sous la forme d'une suite d'enregistrements physiques de 120 octets.

A-1.1. EN-TETE sur support séquentiel

0 - 1	C1	C2
2 - 3	C3	C4
4 - 5	C5	C6
6 - 7	0	F
8 - 9	4	V
10 - 11	RCN	
12 - 13	LBL	

C1 à C6 : Nom du fichier en EBCDIC

F : 1 ⟶ fichier BT Fortran
F : 0 ⟶ autre fichier BT

V : Numéro de version du fichier

RCN : Nombre total d'enregistrements du fichier (dans le cas où le fichier est sorti par le bibliothécaire).

LBL : Longueur des enregistrements

A-1.2. EN-TETE sur disque (y compris dans la zone GI-GO)

0 - 1	C1	C2
2 - 3	C3	C4
4 - 5	C5	C6
6 - 7	0	F
8 - 9	4	V
10 - 11	RCN	
12 - 13	ADSTR	

C1 à C6 : Nom du fichier en EBCDIC

F : 1 ⟶ fichier BT Fortran
F : 0 ⟶ autre fichier BT

V : Numéro de version du fichier

RCN : Nombre d'enregistrements BT du fichier (dans le cas où le fichier est sorti par le bibliothécaire).

ADSTR : Adresse du premier secteur du disque occupé par le fichier

A-1.3. Modules

Les modules sont constitués d'articles BT. Les articles BT sont décrit dans l'annexe B.

A-2. FORMAT IMT

Le format "Image Mémoire Translatable" (IMT) constitue le format binaire chargeable en mémoire pour exécution.

Ce format est constitué d'un certain nombre d'unités logiques de longueur variable, qui sont :

- l'en-tête
- le programme suivi du descripteur "fin de programme", et éventuellement de la table des overlays.
- la table PTG (voir paragraphe 3-2.8) (si elle existe) associée au programme.
- les différents overlays (s'ils existent) comprenant chacun un descripteur "fin d'overlay" suivi de la table PTG associée (si elle existe).

Sur le disque, les unités physiques sont consécutives (256 octets) et n'entraînent pas de fractionnement supplémentaire des unités logiques.

Sur un support externe, autre que le disque, les unités logiques se présentent chacune sous la forme d'une suite d'enregistrements physiques qui sont les unités physiques de transfert de (ou) vers un support externe.

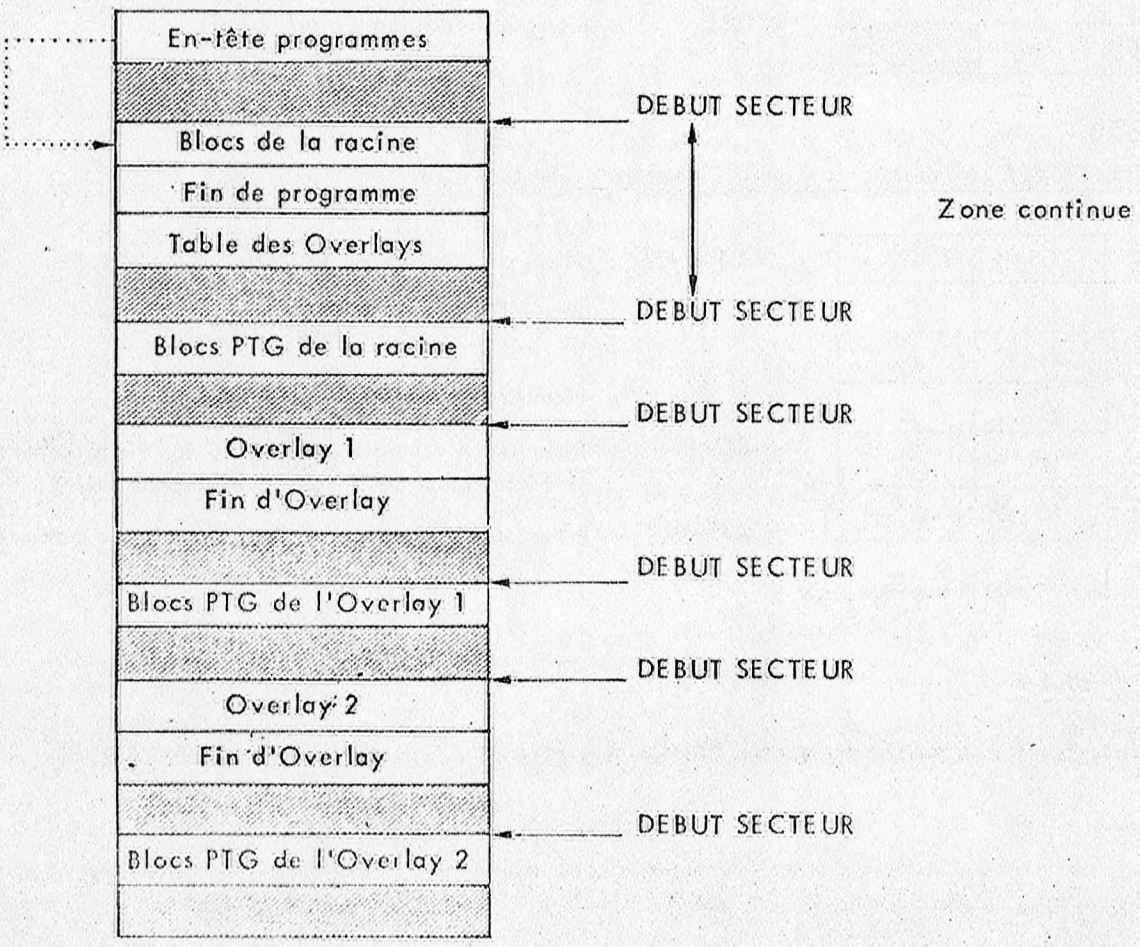
Il existe deux types de format IMT :

- Le format "programme" qui correspond à une structure complète de programme (exécutable en mode maître ou esclave, avec ou sans overlays, etc...).
- Le format "données" qui correspond à une zone de données directement chargeable sans autre intervention du chargeur.

A-2.1. Organisation logique du format IMT sur disque

Structure générale

Cette structure est représentée sur le schéma suivant :



Description des unités logiques

Rappelons que, comme pour tout fichier implanté dans une zone de format standard du disque, l'en-tête d'un fichier IMT est séparé physiquement du fichier lui-même et est catalogué dans le dictionnaire de la zone à laquelle appartient le fichier.

• EN-TETE "PROGRAMME" (y compris dans la zone GI-GO)

0 - 1	C1	C2
2 - 3	C3	C4
4 - 5	C5	C6
6 - 7	NOV	TMEM
8 - 9	1	V
10 - 11	LPR	
12 - 13	ADSTR	

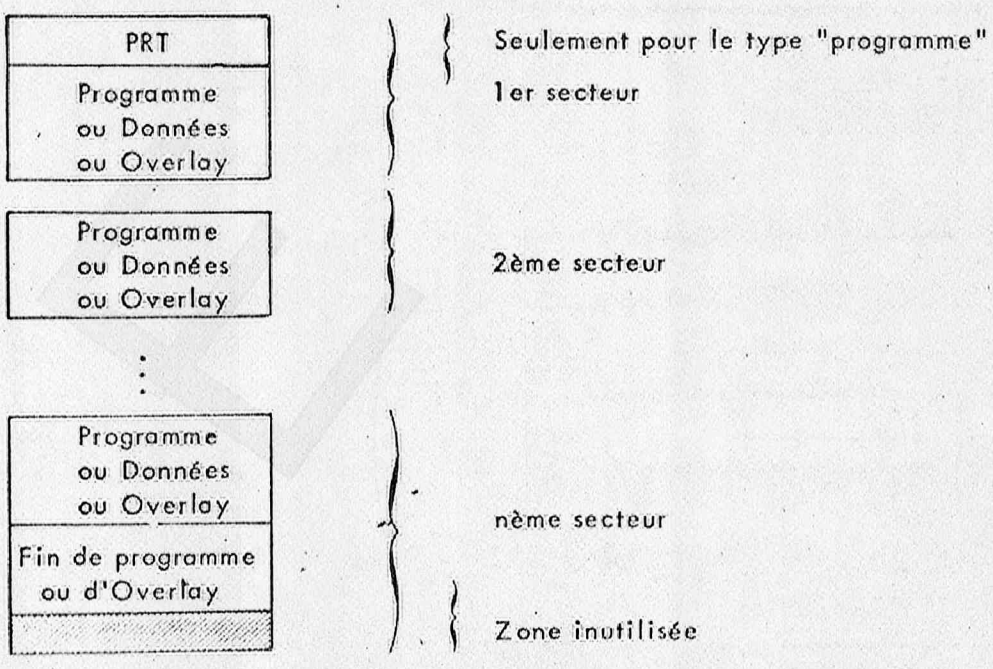
C1 à C6 : Nom du programme en EBCDIC
NOV : Nombre de branches d'overlay
TMEM : Emplacement mémoire occupé par la racine et la plus gra branche d'overlay exprimé en nombre de pages (1 page = 256 octets).
V : Numéro de version du programme
LPR : Longueur du programme (en octets) : Racine + section SH(+ bloc fin + table d'overlay.
Sur disque, on ne fait pas de somme de contrôle.
ADSTR : Adresse du premier secteur contenant le programme.

• EN-TETE "DONNEES" (y compris dans la zone GI-GO)

0 - 1	C1	C2
2 - 3	C3	C4
4 - 5	C5	C6
6 - 7	0	TMEM
8 - 9	3	V
10 - 11	LPR	
12 - 13	ADSTR	

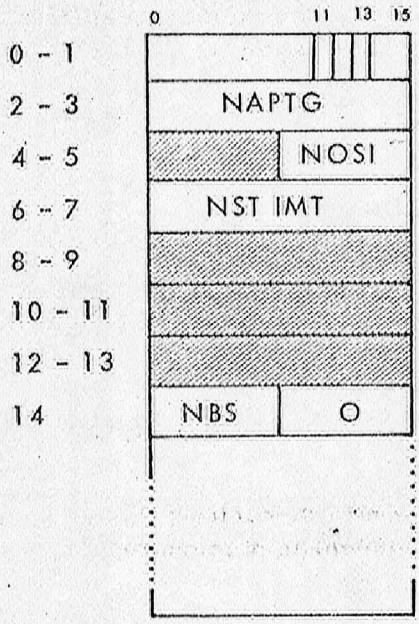
C1 à C6 : Nom des données en EBCDIC
TMEM : Emplacement mémoire occupé par le programme exprimé en nombre de pages (1 page = 256 octets).
V : Numéro de version des données
LPR : Longueur des données (en octets) à charger en mémoire.
ADSTR : Adresse du premier secteur contenant les données.

• BLOCS PROGRAMME (ou Données, ou Overlay)



Remarque : Pour le type "données", le bloc fin de programme est constitué de 16 octets nuls.

• Fin de programme sans overlays



Octet 1 : bit 13 = 1 si mode maître.
bit 11 = 1 si accès aux zones protégées.

NAPTG : Nombre d'adresses contenues dans la table PTG (absence de table PTG si NAPTG = 0).

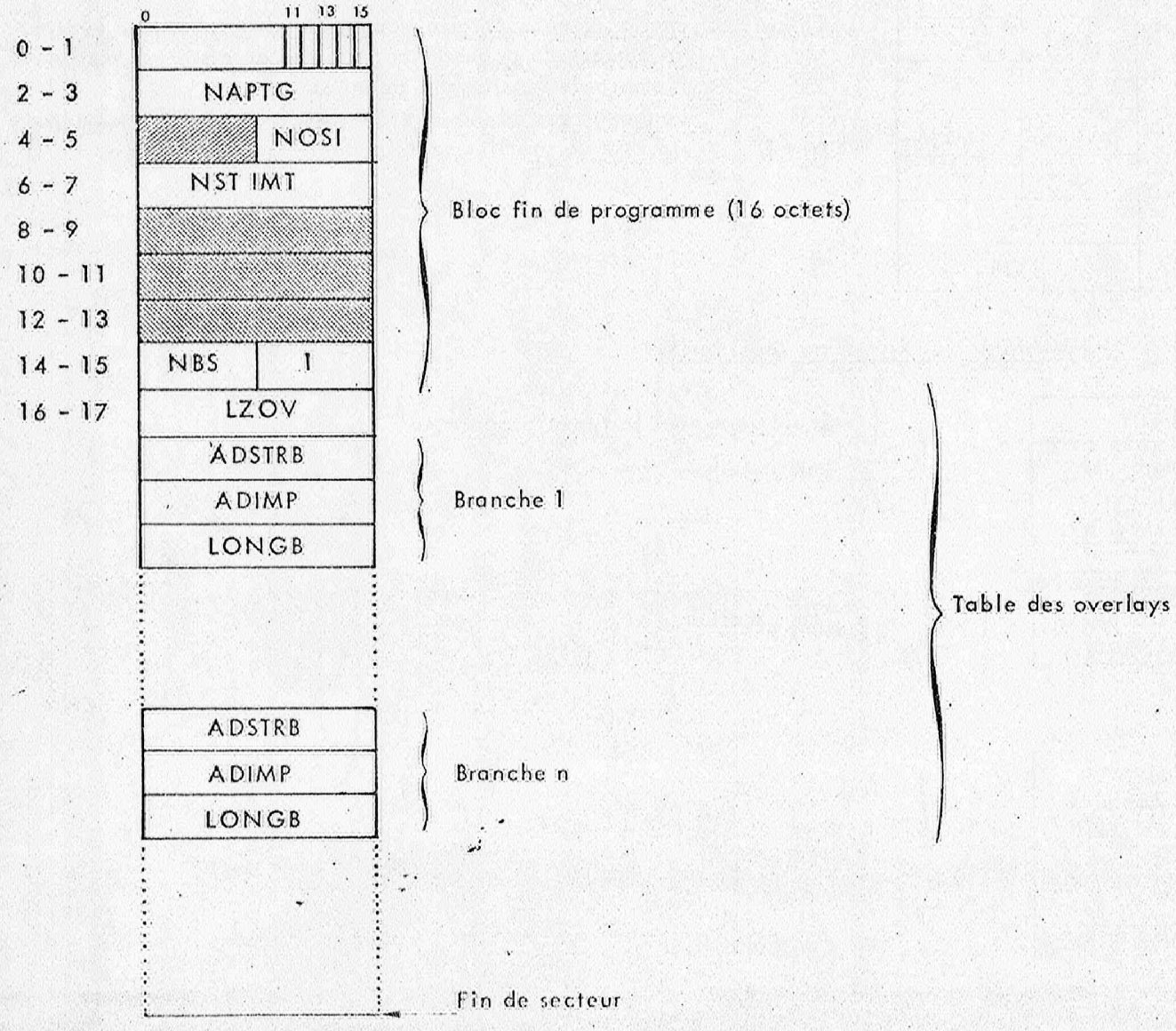
NOSI : Numéro de la section initiale du programme

NSTIMT : Nombre total de secteurs du programme en IMT (y compris la table PTG de la racine, les overlays et les tables PTG des overlays) mis en place par le bibliothécaire.

NBS : Nombre de sections

Fin de secteur

• Fin de programme avec overlays



Bloc fin de programme (16 octets)

Branche 1

Table des overlays

Branche n

Fin de secteur

Octet 1 : bit 13 = 1 si mode maître
bit 11 = 1 si accès aux zones protégées

NAPTG : Nombre d'adresses contenues dans la table PTG (absence de table PTG si NAPTG = 0).

NOSI : Numéro de la section initiale du programme

NST IMT : Nombre total de secteurs du programme IMT (y compris la table PTG de la racine, les overlays et les tables PTG des overlays) mis en place par le bibliothécaire.

NBS : Nombre de sections

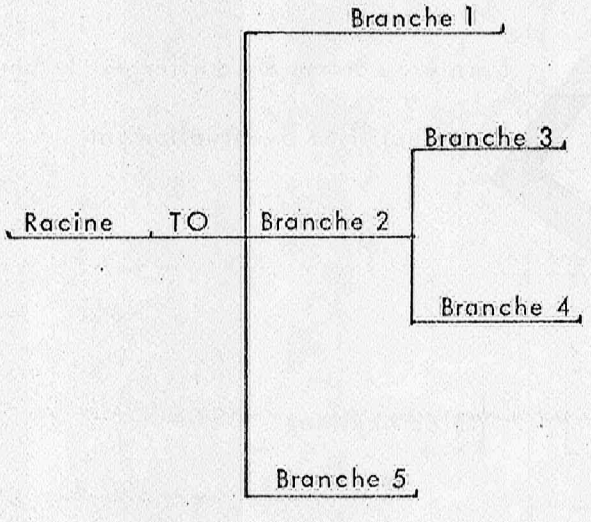
LZOV : Longueur de la zone mémoire des overlays (emplacement mémoire nécessaire à la plus grande extension des branches d'overlays y compris la table d'overlays) (1).

ADSTRB : Adresse secteur par rapport au début de la zone disque, de la branche d'overlay sur le disque.

ADIMP : Adresse d'implantation de la branche d'overlay, relativement à G (elle doit pouvoir adresser 64K octets).

LONGB : Longueur en octets de la branche d'overlay.

(1) Exemple :



TO : Table des overlays

LZOV : TO + Branche 2 + Branche 4

• FIN d'overlay

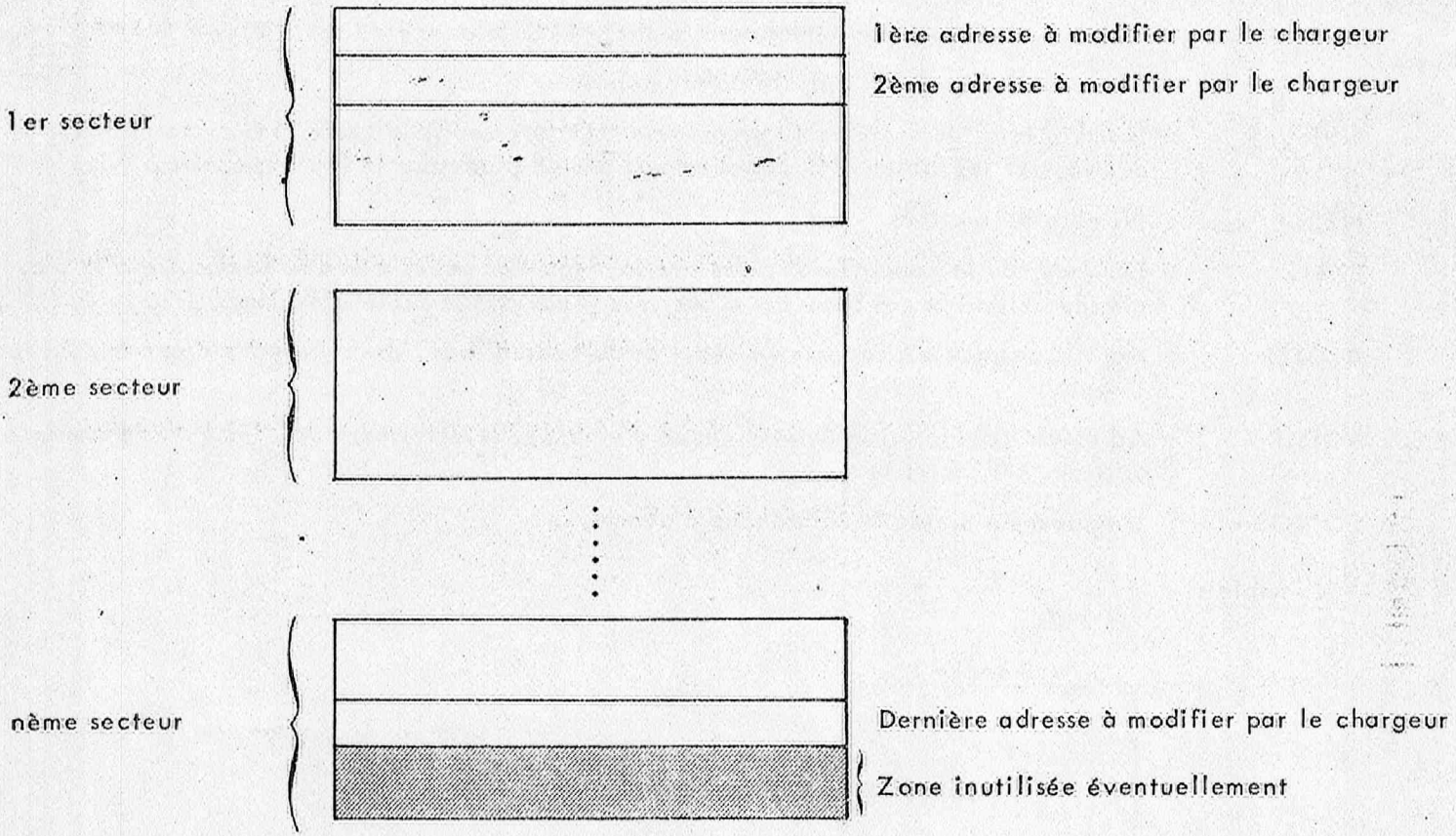
0 - 1	NAPTGB

NAPTGB : Nombre d'adresses contenues dans la table PTG de la branche d'overlay (pas de table PTG si NAPTGB = 0).

Remarque :

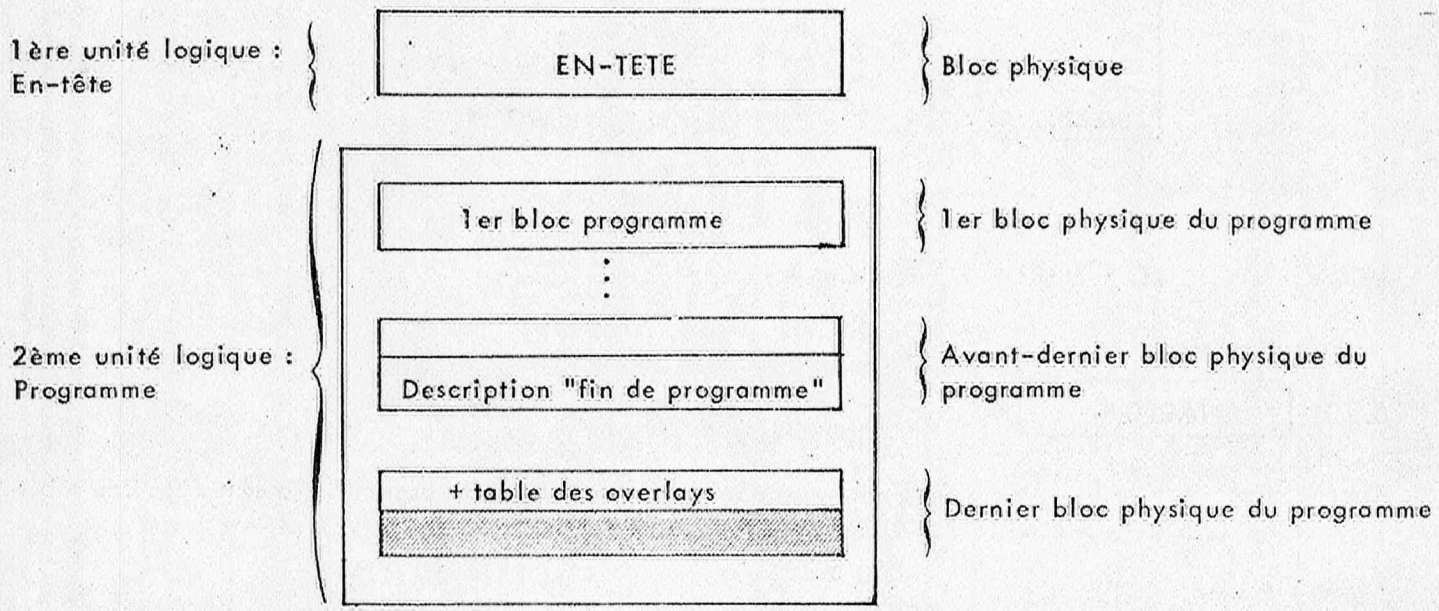
Ces deux octets resteront inutilisés une fois chargés en mémoire.

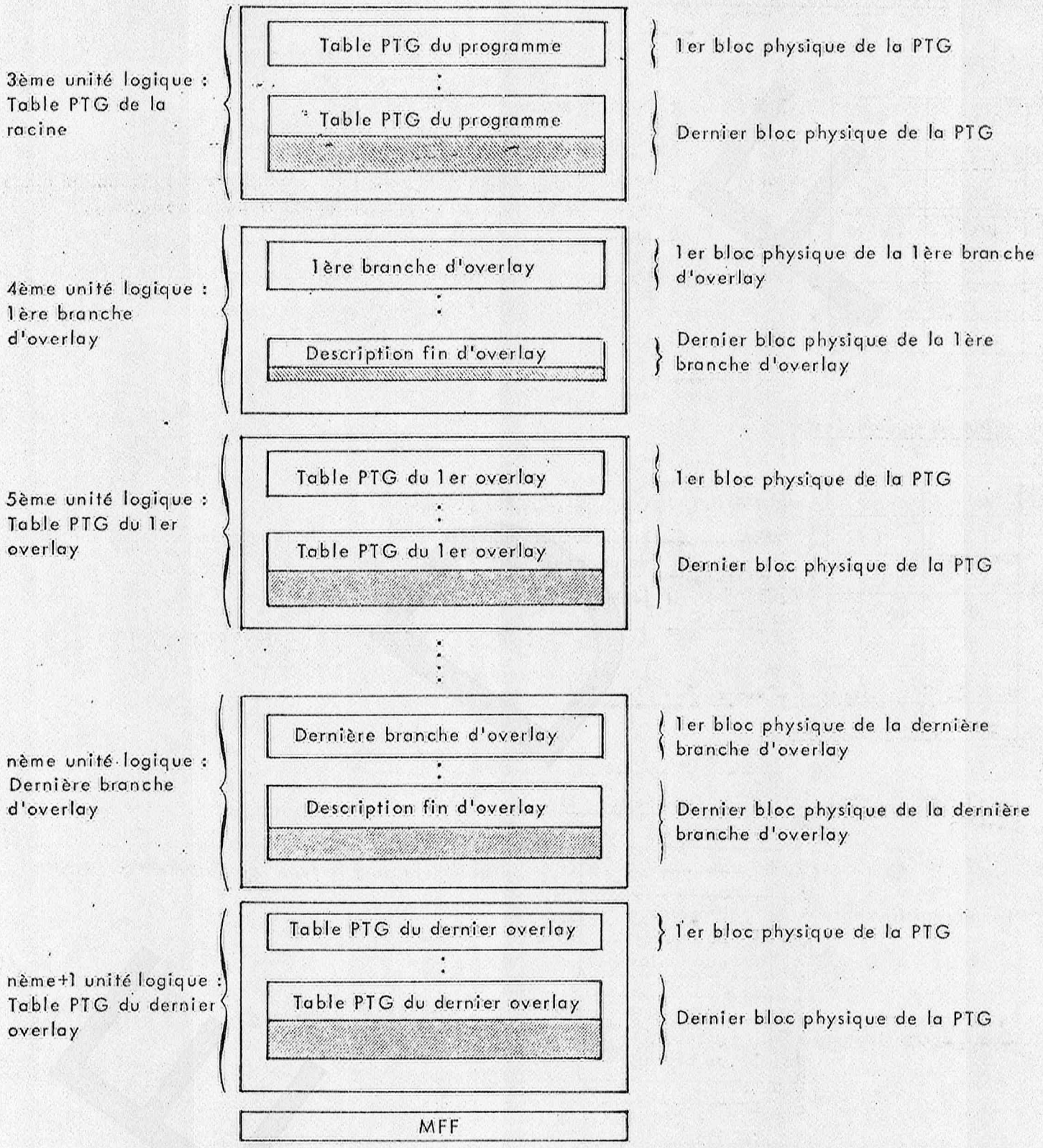
• Blocs PTG



A-2.2. Organisation logique du format IMT sur support séquentiel

■ Structure générale




3ème unité logique : Table PTG de la racine
Table PTG du programme
Table PTG du programme
1er bloc physique de la PTG
Dernier bloc physique de la PTG
4ème unité logique : 1ère branche d'overlay
1ère branche d'overlay
Description fin d'overlay
1er bloc physique de la 1ère branche d'overlay
Dernier bloc physique de la 1ère branche d'overlay
5ème unité logique : Table PTG du 1er overlay
Table PTG du 1er overlay
Table PTG du 1er overlay
1er bloc physique de la PTG
Dernier bloc physique de la PTG
nème unité logique : Dernière branche d'overlay
Dernière branche d'overlay
Description fin d'overlay
1er bloc physique de la dernière branche d'overlay
Dernier bloc physique de la dernière branche d'overlay
nème+1 unité logique : Table PTG du dernier overlay
Table PTG du dernier overlay
Table PTG du dernier overlay
1er bloc physique de la PTG
Dernier bloc physique de la PTG
MFF

Description des unités logiques

• EN-TETE Programme

C1	C2
C3	C4
C5	C6
NOV	TMEM
1	V
LPR	
LBL	

C1 à C6 : Nom du programme en EBCDIC

NOV : Nombre de branches d'overlay

TMEM : Emplacement mémoire occupé par la racine et la plus grande branche d'overlay exprimé en nombre de pages (1 page = 256 octets).

V : Numéro de version du programme

LPR : Longueur du programme (en octets) : Racine + section SHUNT + bloc fin + table d'overlay.

LBL : Longueur des enregistrements.

• EN-TETE Données

C1	C2
C3	C4
C5	C6
0	TMEM
3	V
LPR	
LBL	

C1 à C6 : Nom des données en EBCDIC

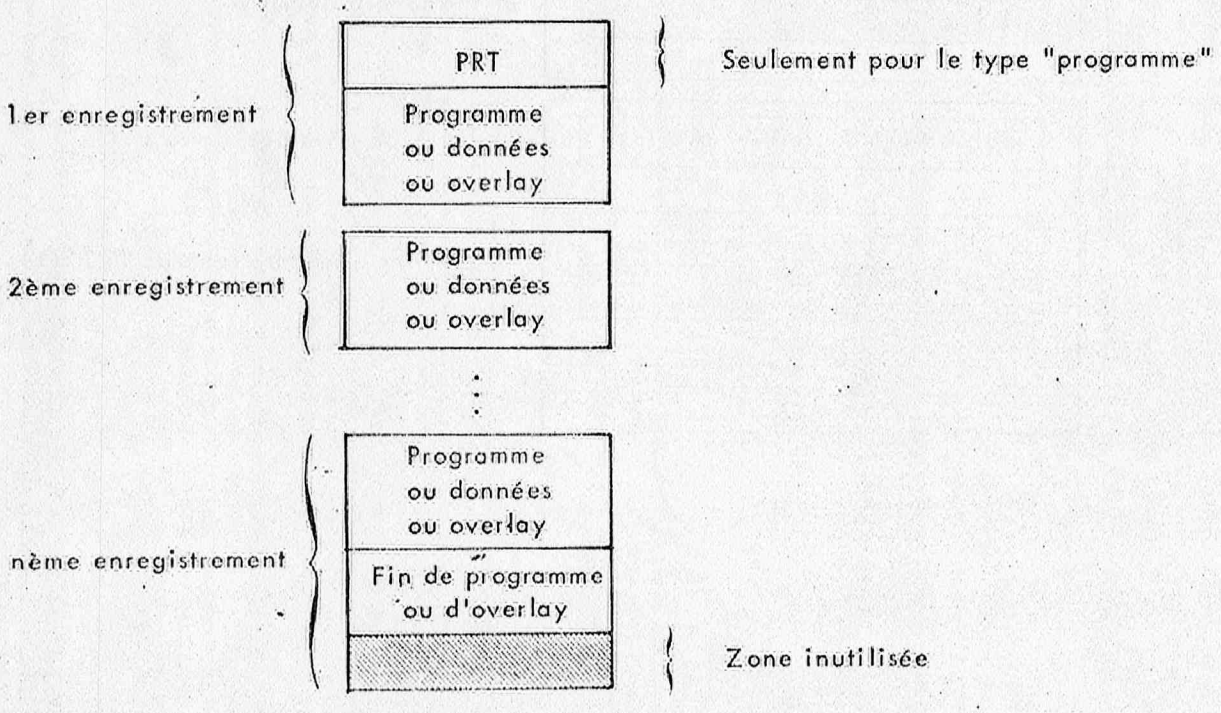
TMEM : Emplacement mémoire occupé par le programme exprimé en nombre de pages (1 page = 256 octets).

V : Numéro de version des données

LPR : Longueur des données (en octets) à charger en mémoire.

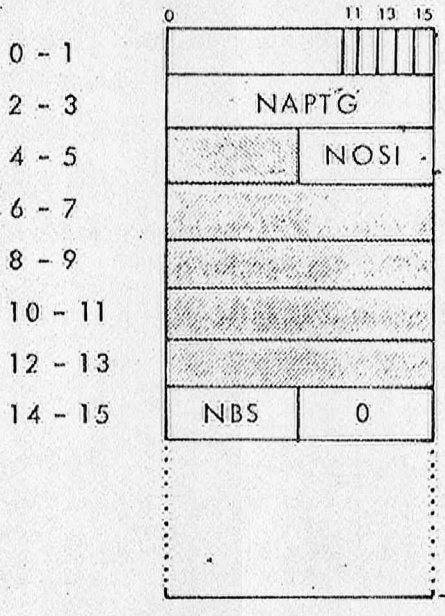
LBL : Longueur des enregistrements.

• BLOCS "PROGRAMME" (ou Données, ou Overlay)



Remarque : Pour le type "données", le bloc fin de programme est constitué de 16 octets nuls.

• FIN DE PROGRAMME sans overlays



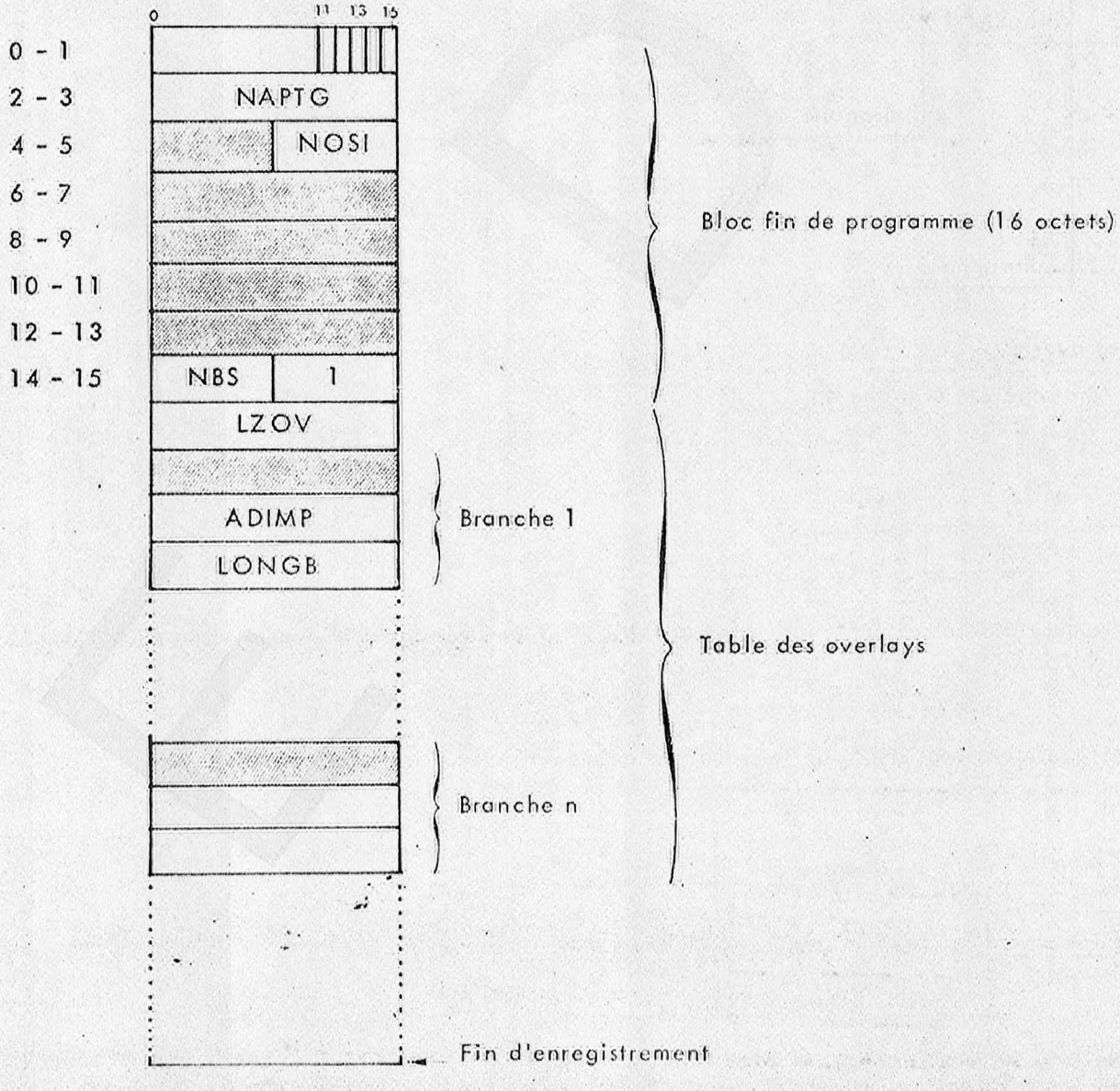
Octet 1 : bit 13 = 1 si mode maître
bit 11 = 1 si accès aux zones protégées

NAPTG : Nombre d'adresses contenues dans la table PTG (absence de table PTG si NAPTG = 0).

NOSI : Numéro de la section initiale du programme

NBS : Nombre de sections

• FIN DE PROGRAMME avec overlays



Octet 1 : Bit 13 = 1 si mode maître
Bit 11 = 1 si accès aux zones protégées

NAPTG : Nombre d'adresses contenues dans la table PTG de la racine (absence de table PTG si NAPTG = 0).

NOSI : Numéro de la section initiale du programme

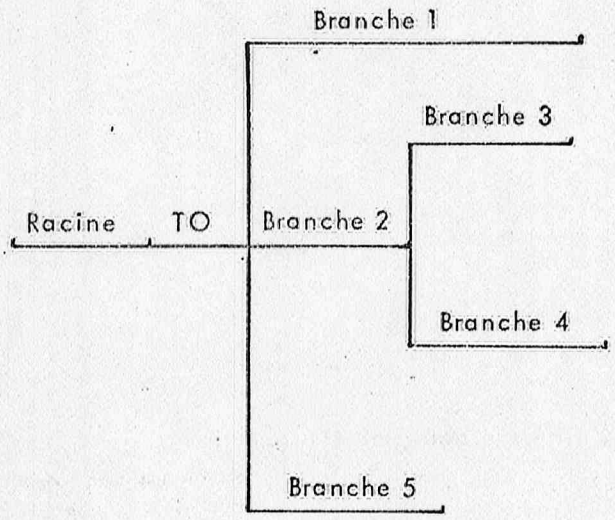
NBS : Nombre de sections

LZOV : Longueur de la zone mémoire des overlays (emplacement mémoire nécessaire à la plus grande extension des branches d'overlays, y compris la table des overlays). (1)

ADIMP : Adresse d'implantation de la branche d'overlay, relativement à G (elle doit pouvoir adresser 64K octets).

LONGB : Longueur en octets de la branche d'overlay.

(1) Exemple :



TO = Table des overlays

LZOV = TO + Branche 2 + Branche 4

• Fin d'overlay

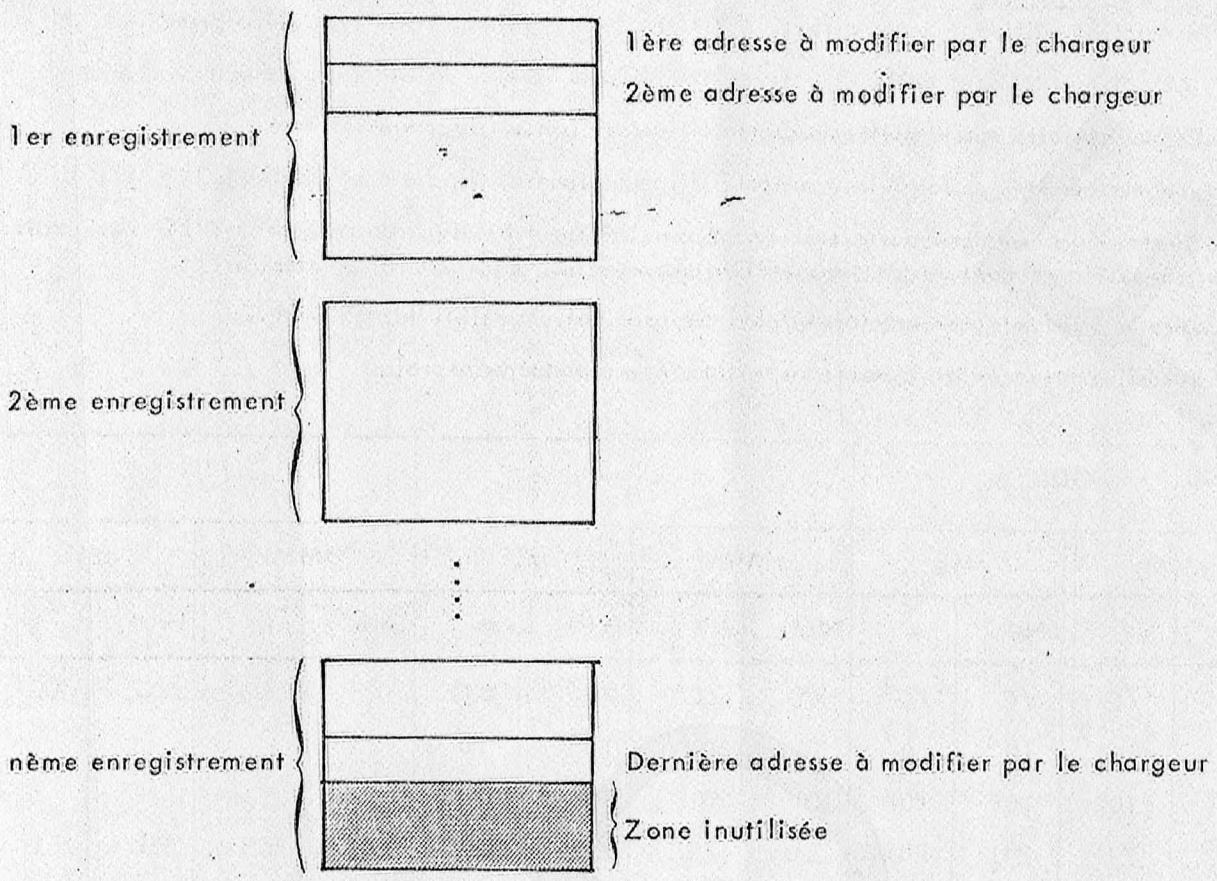
0 - 1	NAPTGB

NAPTGB : Nombre d'adresses contenues dans la table PTG de la branche d'overlay (pas de table PTG si NAPTG = 0).

Remarque :

Ces deux octets resteront inutilisés une fois chargés en mémoire.

• Blocs PTG



A-3. TABLE PTG

La table PTG est créée par l'éditeur de liens dans le cas d'un programme exécutable en mode maître. Il y range la liste des adresses des mots mémoire contenant des références auxquelles le chargeur devra rajouter la contenu de la base G.

En effet, l'éditeur de liens ne connaissant pas la valeur de la base G :

- En mode esclave, la base G est rajoutée au moment de l'exécution.

- En mode maître, la base G n'étant pas rajoutée lors de l'exécution, c'est le loader qui le fait, au moment du chargement en mémoire, sur toutes les adresses se trouvant dans la table PTG.

Il est possible, au moment de l'assemblage, de mettre en début de zone opérande d'une directive "DATA" ou "GEN" le caractère spécial "#". Dans ce cas, lors d'un chargement d'un programme édité en mode maître, le loader ne rajoutera pas le contenu de la base G à ce mot mémoire. Ceci permet en particulier au moment de l'exécution d'accéder à la zone des données système.

Le tableau ci-dessous résume les divers cas possibles. Il indique dans quels cas les adresses des mots mémoire sont mis dans la table PTG, le type des instructions concernées ainsi que la zone mémoire sur laquelle va porter l'instruction.

SI	Segment dans lequel est déclarée la directive "DATA" ou "GEN".
SR	Segment dans lequel se trouve le mot mémoire référencé par la directive "DATA" ou "GEN".
MM	Le programme sera exécuté en mode maître.
ME	Le programme sera exécuté en mode esclave
#	La zone opérande de la directive "DATA" ou "GEN" est précédée du caractère spécial "#"
N#	La zone opérande de la directive "DATA" ou "GEN" n'est pas précédée du caractère spécial "#".

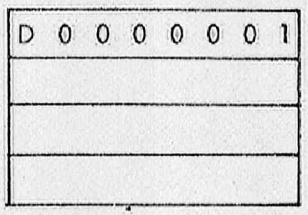
T0	L'instruction utilisant cette directive "DATA" ou "GEN" est de type zéro.
T0P	L'instruction utilisant cette directive "DATA" ou "GEN" est de type zéro prime.
T2	L'instruction utilisant cette directive "DATA" ou "GEN" est de type deux.
ZS	La zone mémoire sur laquelle va porter l'instruction est la zone système.
ZU	La zone mémoire sur laquelle va porter l'instruction est la zone utilisateur.
PTG	L'adresse de ce mot mémoire sera rangé par l'éditeur de liens dans la table PTG. Le loader lui rajoutera le contenu de la base G au moment du chargement en mémoire.
/G	L'adresse implantée en mémoire après chargement est relative à la base G.
A	L'adresse implantée en mémoire après chargement est en absolue.

SR / SI	CDS				LDS				LPS			
	MM		ME		MM		ME		MM		ME	
	N#	#	N#	#	N#	#	N#	#	N#	#	N#	#
CDS	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	/G T0 T0P ZU	A T2 ZU PTG	/G T2 ZS	/G T2 ZU	/G T2 ZU
LDS	A T0 T0P ZU PTG	/G T0 T0P ZS	/G T0 T0P ZU	/G T0 T0P ZU	A T0 T0P ZU PTG	/G T0 T0P ZS	/G T0 T0P ZU	/G T0 T0P ZU	A T2 ZU PTG	/G T2 ZS	/G T2 ZU	/G T2 ZU

Annexe B – Articles BT

B-1. CONTROLE D'ENREGISTREMENT

Bien que les quatre octets de contrôle d'enregistrement ne soient pas un article BT, pour simplifier leur analyse, les processeurs les considèrent comme étant l'article 1.



Octet 0 : D=0 pour un enregistrement courant
D=1 pour le dernier enregistrement
pour l'enregistrement "en-tête"

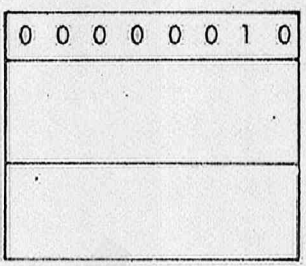
Octet 1 : Numéro de séquence d'enregistrement. Zéro pour le premie
et module 256.
(Le numéro de séquence de l'"en-tête" est toujours égal à 0

Octet 2 : Somme de contrôle. Elle est égale au complément à deux d
la somme sans report de tous les octets utiles de l'enregis-
trement, non compris celui de la somme de contrôle.
L'addition sans report de tous les octets de l'enregistrement,
(y compris celui de la somme de contrôle) donne zéro.

Octet 3 : Longueur utile en octets de l'enregistrement.

B-2. ARTICLE 2 - FIN DE SEGMENT (HEXA : 02)

Il marque la fin d'un segment. Il déclare sa dimension sans le "pool" dans le cas d'une CDS ou d'un LDS. Dans le cas d'un LPS, il fournit éventuellement l'adresse de lancement dans le segment.

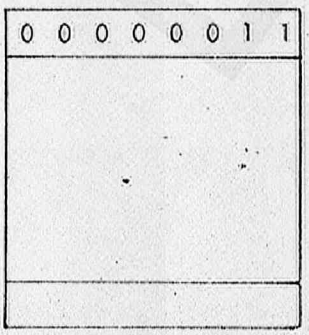


Octets 1 et 2 : Dimension en octet du segment

Octets 3 et 4 : Adresse de lancement (dans le cas d'un LPS)

B-3. ARTICLE 3 - FIN DE MODULE (HEXA : 03)

Il permet de connaître la fin du module d'assemblage, le niveau de sévérité ainsi que le nom de la section de lancement.

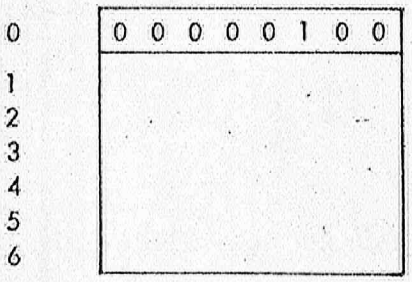


Octets 1 à 6 : Nom de la section de lancement (en EBCDIC).

Octet 7 : Niveau de sévérité du module d'assemblage.

B-4. ARTICLE 4 - DECLARATION DE CDS (HEXA : 04)

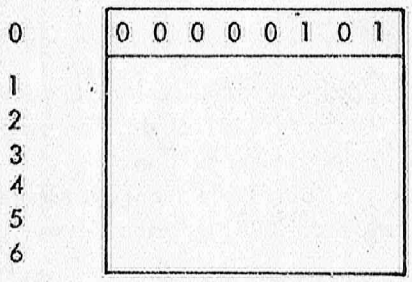
Il comprend le nom de la CDS.



Octets 1 à 6 : Nom de la CDS (en EBCDIC).

B-5. ARTICLE 5 - DECLARATION DE LDS (HEXA : 05)

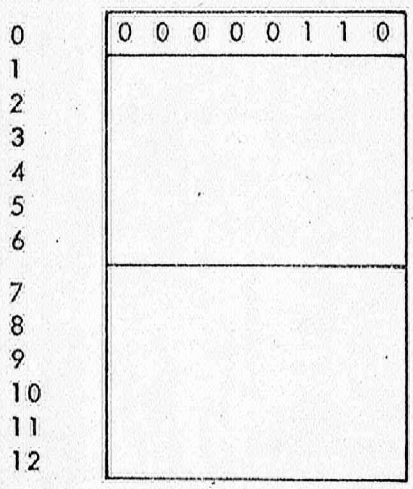
Il comprend le nom du LDS.



Octets 1 à 6 : Nom du LDS (en EBCDIC).

B-6. ARTICLE 6 - DECLARATION DE LPS (HEXA : 06)

Il comprend le nom du LPS ainsi que celui du LDS associé.

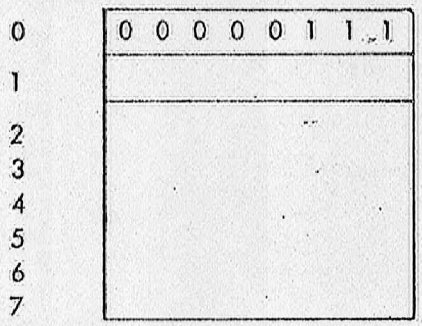


Octets 1 à 6 : Nom du LPS (en EBCDIC)

Octets 7 à 12 : Nom du LDS associé (en EBCDIC).

B-7. ARTICLE 7 - CALL (HEXA : 07)

Il donne le nom du LPS référencé dans un CLS ou un CSV.

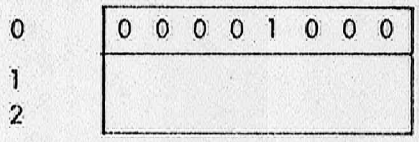


Octet 1 : Code opération { CSV = 7 / CLS = 8

Octets 2 à 7 : Nom de la section appelée.

B-8. ARTICLE 8 - DEFINITION D'ORIGINE (HEXA : 08)

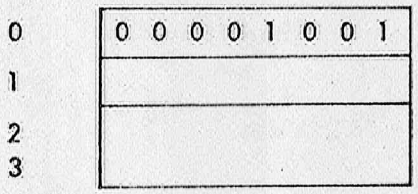
Il permet de faire progresser le compteur d'emplacement du nombre d'octets désignés.



Octets 1 et 2 : Nombre d'octets

B-9. ARTICLE 9 - CHARGEMENT DE VALEUR PREDEFINIE (HEXA : 09)

Il provoque le chargement de deux octets qui sont précédés d'un octet indicatif définissant le type de la référence (CDS, LDS...). Les deux octets de chargement contiennent une valeur prédéfinie.

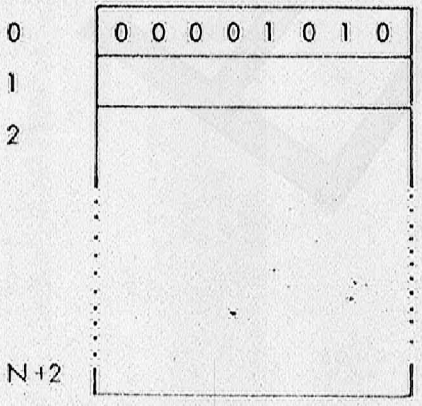


Octet 1 : Indicateur I (voir détail au paragraphe B-28).

Octets 2 et 3 : Valeur prédéfinie.

B-10. ARTICLE 10 - CHARGEMENT ABSOLU (HEXA : 0A)

Il permet le chargement de N octets. Ce nombre est contenu dans l'octet 1. Il doit toujours être inférieur à 104 octets.



Octet 1 : Nombre N d'octets à charger

Octets 2 à N+2 : N octets à charger

B-11. ARTICLE 11 - DEFINITION DE REFERENCE EN AVANT (HEXA : 0B)

Il précise à l'éditeur de liens le numéro et la valeur d'une référence en avant. Il devra les garder dans une table de symboles afin de satisfaire les références en avant.

0	0 0 0 0 1 0 1 1	Octet 1 : Indicateur I (voir paragraphe B-28).
1		Octets 2 et 3 : Valeur de la référence en avant.
2		Octets 4 et 5 : Numéro de la référence en avant.
3		
4		
5		

B-12. ARTICLE 12 - REMPLISSAGE (HEXA : 0C)

Cet article est ineffectif.

0	0 0 0 0 1 1 0 0

B-13. ARTICLE 13 - DEFINITION DE CHAMP (HEXA : 0D)

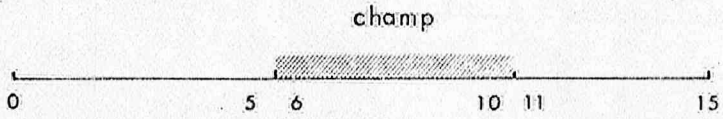
Un champ est la partie d'un mot sur laquelle vont porter les divers articles d'addition ou de soustraction qui suivent la définition de champ. Ces opérations se terminent par un article de fin d'expression.

L'article de définition de champ permet de connaître la taille de ce champ (16 bits maximum).

0	0 0 0 0 1 1 0 1	Octet 1 : Longueur du champ (en nombre de bits)
1		Octet 2 : Adresse de fin de champ par rapport au début du mot (en nombre de bits).
2		

Exemple :

Longueur du champ = 5
Adresse de fin de champ = 10



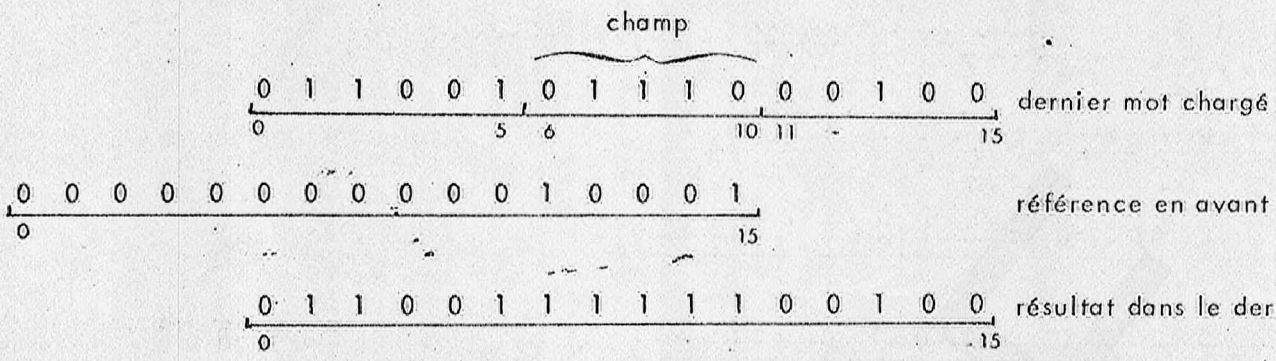
B-14. ARTICLE 14 - ADDITION DE REFERENCE EN AVANT (HEXA : 0E)

Dans le champ du dernier mot chargé, il provoque l'addition de la référence en avant à la valeur contenue dans le champ. (Le champ du dernier mot chargé est défini par le dernier article de définition de champ).

0	0 0 0 0 1 1 1 0	Octets 1 et 2 : Numéro de la référence en avant.
1		
2		

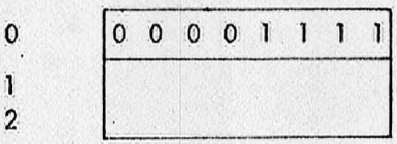
Exemple :

Longueur du champ = 5
Adresse fin de champ = 10



B-15. ARTICLE 15 - SOUSTRACTION DE REFERENCE EN AVANT (HEXA : 0F)

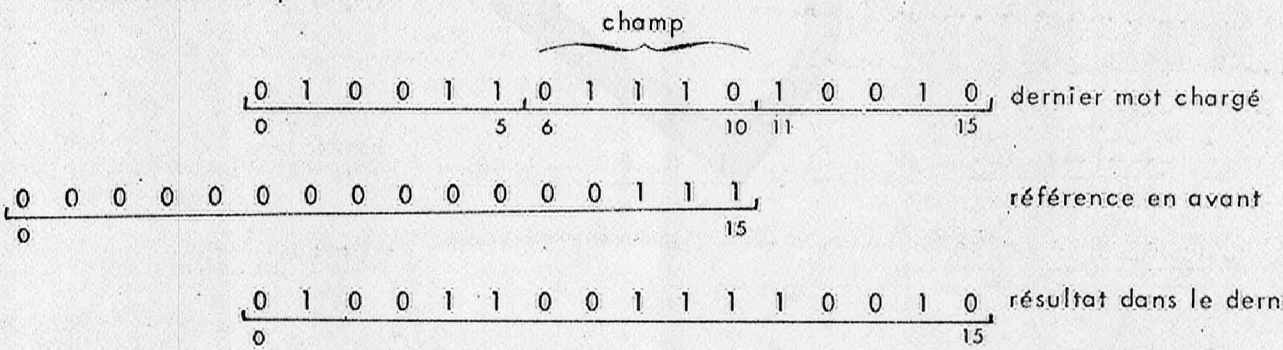
Dans le champ du dernier mot chargé, il provoque la soustraction de la référence en avant à la valeur contenue dans le champ. (Le champ du dernier mot chargé est défini par le dernier article de définition de champ).



Octets 1 et 2 : Numéro de la référence en avant

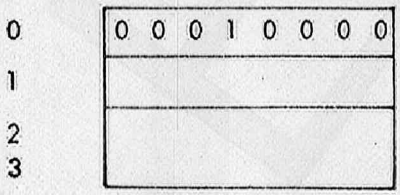
Exemple :

Longueur du champ = 5
Adresse fin de champ = 10



B-16. ARTICLE 16 - ADDITION DE VALEUR PREDEFINIE (HEXA : 10)

Dans le champ du dernier mot chargé, il provoque l'addition de la valeur prédéfinie à la valeur contenue dans le champ. (Le champ du dernier mot chargé est défini par le dernier article de définition de champ).

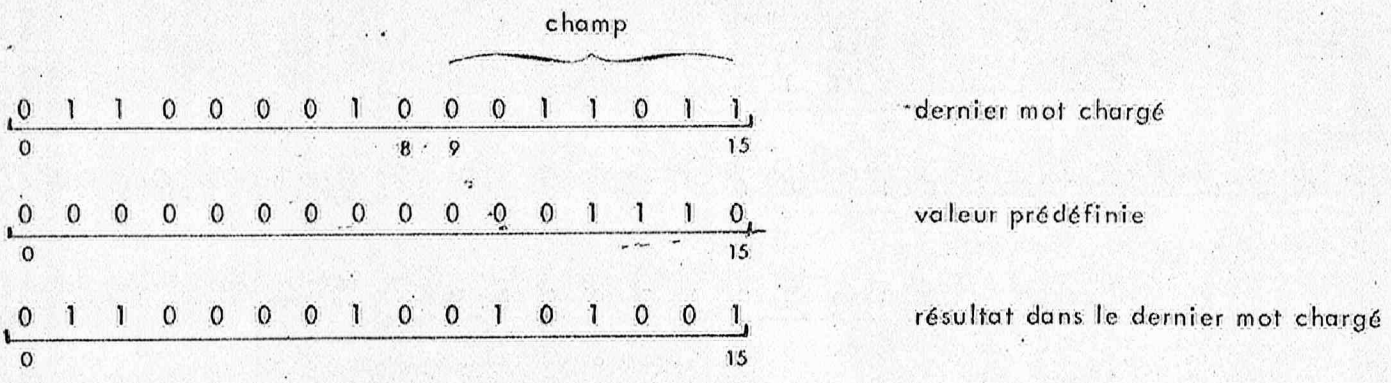


Octet 1 : Indicateur I (voir paragraphe B-18)

Octets 2 et 3 : Valeur prédéfinie

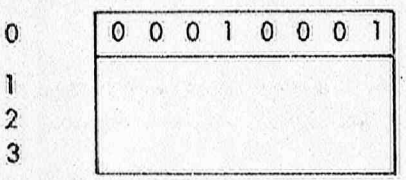
Exemple :

Longueur de champ = 7
Adresse fin de champ = 15



B-17. ARTICLE 17 - SOUSTRACTION DE VALEUR PREDEFINIE (HEXA : 11)

Dans le champ du dernier mot chargé, il provoque la soustraction de la valeur prédéfinie à la valeur contenue dans le champ. (Le champ du dernier mot chargé est défini par le dernier article de définition de champ).

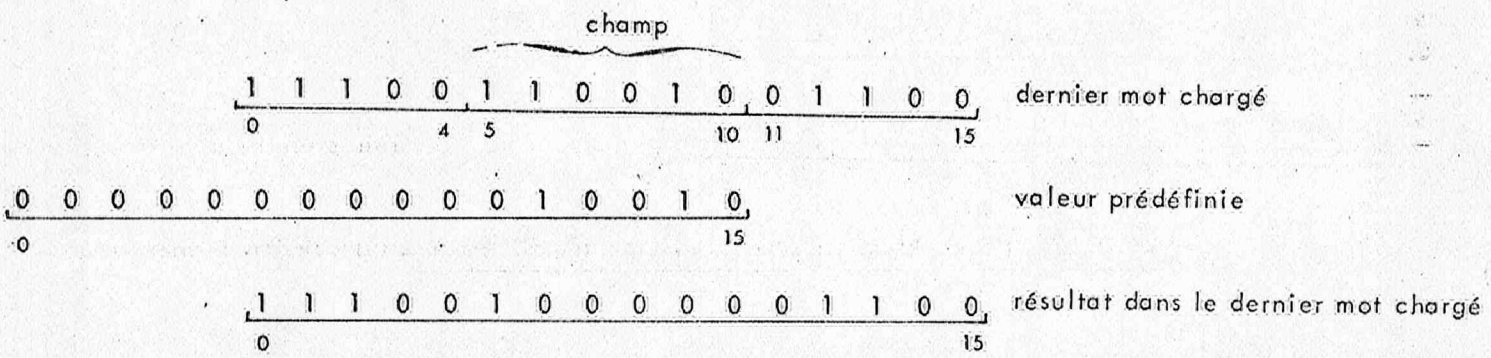


Octet 1 : Indicateur I (voir paragraphe B-18)

Octets 2 et 3 : Valeur prédéfinie

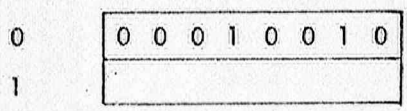
Exemple :

Longueur du champ = 6
Adresse fin de champ = 10



B-18. ARTICLE 18 - FIN D'EXPRESSION (HEXA : 12)

Il indique la fin de l'expression à évoluer et le type du résultat de l'évaluation.

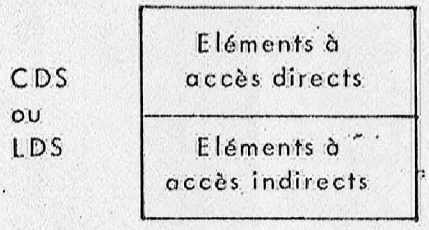


Octet 1 : Indicateur J (voir au paragraphe B-18)

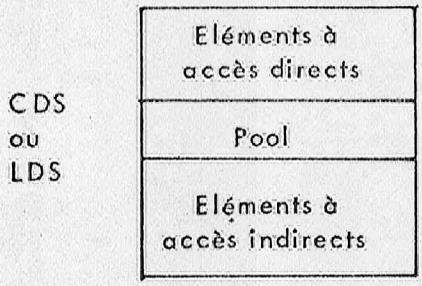
B-19. ARTICLE 19 - POSITIONNEMENT DU POOL (HEXA : 13) (EDL-E et EDL-D)

Il indique à quel emplacement de la CDS ou du LDS courant est situé le "pool".

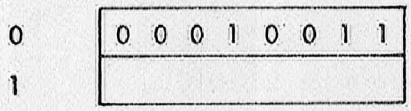
Une CDS ou un LDS est constitué d'éléments à accès direct et d'éléments à accès indirects.



Lorsque le compilateur génère des éléments à accès direct après avoir déjà généré des éléments à accès indirect, il déclare ces nouveaux éléments à accès direct comme article de "pool", l'éditeur de liens se chargeant de créer une zone de "pool" et de les y insérer.



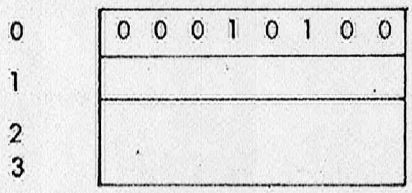
Ceci est nécessité par le fait que les éléments à accès directs ne pouvant généralement pas avoir une adresse relative au début de section supérieure à 256, il convient de les mettre en début de section.



Octet 1 : Adresse du début du "pool" par rapport au début du segment.

B-20. ARTICLE 20 - REMPLISSAGE DE "POOL" (HEXA : 14) (EDL-E et EDL-D)

Il permet d'insérer des éléments dans le "pool".

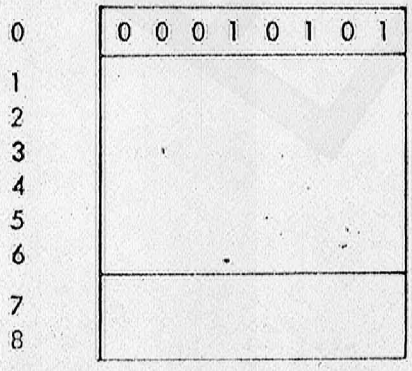


Octet 1 : Indicateur I (voir paragraphe B-28).

Octets 2 et 3 : Valeur prédéfinie ou numéro de référence

B-21. ARTICLE 21 - REFERENCE EXTERNE (HEXA : 15)

Il permet d'établir une association entre un nom symbolique et un numéro de référence.

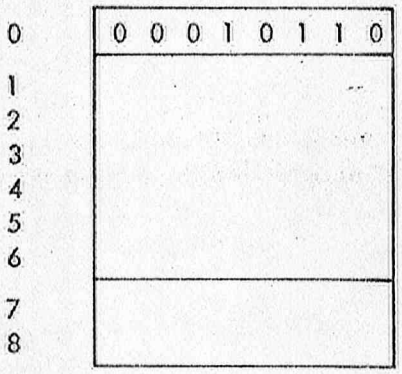


Octets 1 à 6 : Nom de la référence (en EBCDIC)

Octets 7 et 8 : Numéro de la référence.

B-22. ARTICLE 22 - DEFINITION EXTERNE (HEXA : 16)

Il permet d'associer un nom symbolique à un numéro de référence dont la valeur sera définie dans le module.



Octets 1 à 6 : Nom de la référence (en EBCDIC)

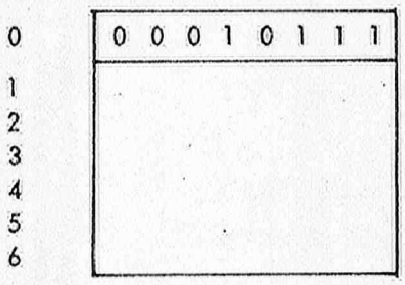
Octets 7 et 8 : Numéro de la référence de définition

ARTICLES "COMMON" FORTRAN (EDL-E et EDL-D)

B-23. ARTICLE 23 - DECLARATION DE "COMMON" (HEXA : 17)

Un "common" est une zone de données communes à plusieurs unités de programme.

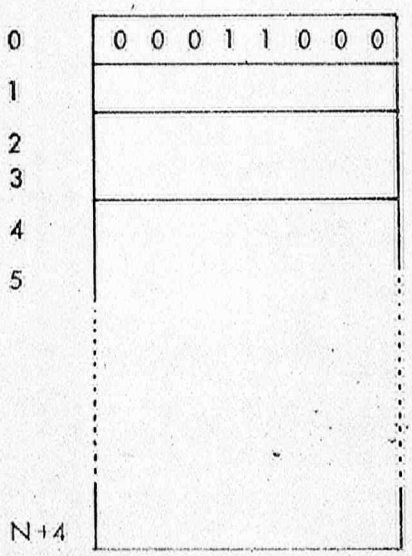
Cet article déclare un "common".



Octets 1 à 6 : Nom du "common" Fortran (en EBCDIC)

B-24. ARTICLE 24 - INITIALISATION DU "COMMON" (HEXA : 18)

Il permet de donner à l'éditeur de liens les éléments nécessaires à la création du "common".



Octet 1 : Nombre N d'octets à initialiser

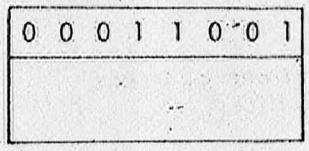
Octets 2 et 3 : Adresse du début de l'initialisation par rapport au début du "common".

Octets 4 à N+4 : Valeurs d'initialisation.

(L'article 24 suit l'article 23 ou un autre article 24).

B-25. ARTICLE 25 - FIN D'INITIALISATION DU "COMMON" (HEXA : 19)

Il termine les initialisations de "common".

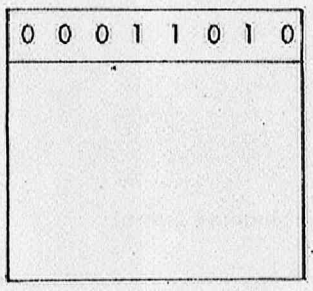


Octets 1 et 2 : Taille en octets du "common"

(L'article 25 suit un article 23 ou un article 24).

B-26. ARTICLE 26 - ADDITION D'ADRESSE DE POINTEUR DE "COMMON" (HEXA : 1A)

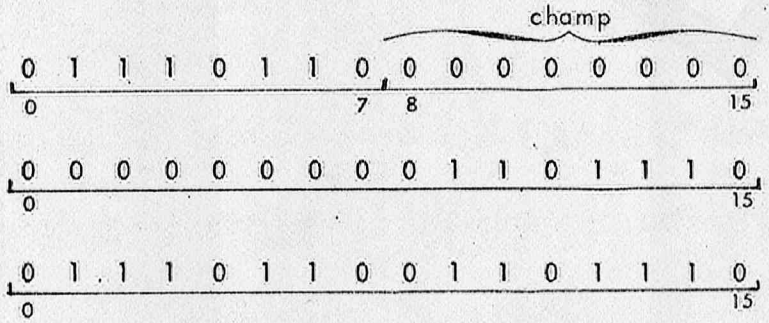
Dans le champ du dernier mot chargé, il provoque l'addition de l'adresse de pointeur de "common" à la valeur contenue dans le champ. (Le champ du dernier mot chargé est défini par le dernier article de définition de champ).



Octets 1 à 6 : Nom du "common" en EBCDIC

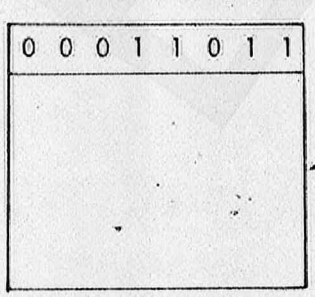
Exemple :

Taille du champ : 8
Adresse fin de champ : 15



B-27. ARTICLE 27 - ADDITION D'ADRESSE DE "COMMON" (HEXA : 1B)

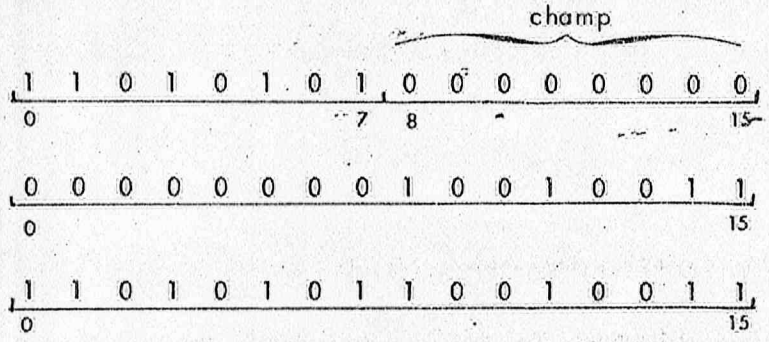
Dans le champ du dernier mot chargé, il provoque l'addition de l'adresse du "common" à la valeur conten dans le champ. (Le champ du dernier octet chargé est défini par le dernier article de définition de champ



Octets 1 à 6 : Nom du "common" en EBCDIC

Exemple :

Taille du champ : 8
Adresse fin de champ : 15



B-28. ARTICLE 28. POSITIONNEMENT DU POOL (HEXA : 1C) (EDLE et EDLD)

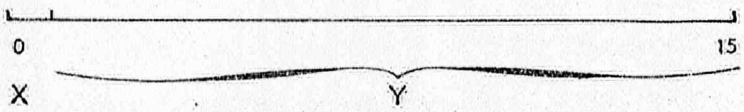
Pour la définition du POOL se reporter à l'article 19.

0	0	0	0	1	1	1	0	0
1								
2								

Octets 1 et 2 : adresse sur 1 mot du début du "POOL" par rapport au début du segment

B-29. SIGNIFICATION DES INDICATEURS I ET J

Indicateur I

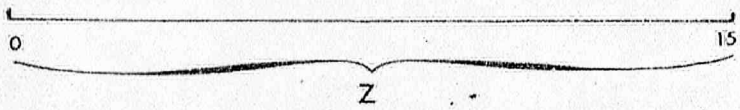


Y = 0 Absolu
1 CDS
2 LDS
3 Debut de "pool" de LDS
4 Fin de "pool" de LDS
5 Zone programme LPS
6 Début de "pool" de CDS
7 Fin de "pool" de CDS
8 Référence en avant

X = 1 Elément devant rester relatif à la base G

Indicateur J

Intervient dans l'article "fin d'expression" et indique le type du résultat de l'évaluation.



Z = 1 Par rapport à la valeur du compteur d'emplacement
2 Par rapport à la base G (ne pas mettre dans la table PTG)
3 Par rapport au début du segment
4 Par rapport à la base Z (ne pas mettre dans la table PTG)
5 Adressage étendu par rapport au début du segment.

Exemple 1 :

BT sorti par l'assembleur MITRAS 2

Les éléments d'un article sont séparés par un "-".

Les octets d'un éléments sont séparés par une ",".

NOMCDS	CDS		04-D5, D6, D4, D3, C3, C4, E2-
	RES	16	08-00, 20-
	FIN		02-00, 20-00, 00-
NOMLDS	LDS		05-D5, D6, D4, D3, C4, E2-
	RES	4	08-00, 08-
	DATA	LD3-LD1	0A-02-00, 00-
			0D-10-0F-
			0E-00, 00-
			0F-00, 01-
			12-00-
LD1	DATA	LD0	0B-02-00, 0A-00, 01-
			0A-02-00, 00
			0D-10-0F-
			0E-00, 02-
			12-00-
LD2	RES	3	08-00, 06-
LD3	DATA	3	0B-02-00, 12-00, 00-
			0A-02-00, 00-
			0D-10-0F-
			10-00-00, 03-
			12-00-
	DATA	-1	0A-02-00, 00-
			0D-10-0F
			11-00-00, 01-
			12-00-
LD0	DATA	LD2	0B-02-00, 16-00, 02-
			0A-02-00, 00-
			0D-10-0F-
			10-02-00, 00-
			12-00
	TEXT	"ABCD"	0A-04-C1, C2, C3, C4-
	FIN		02-00, 1C-00, 00-
NOMLPS	LPS	NOM LDS	06-D5, D6, D4, D3, D7, E2-D5, D6, D4, D3, C4, E2-

```
                REF         REFEXT      15-D9,C5,C6,C5,E7,E3-00,03-
                DEF         DEFEXT      16-C4,C5,C6,C5,E7,E3-00,04-
VALDEF          LDA         =2          0A-02-20,02-
DEFEXT          STA         LD2         0B-05-00,02-00,04-
                LDE         LD1         0A-06-11,0C,01,0A,12,12-
                STE         LD3
                CSV         M:IO        07-07-D4,7A,C9,D6,40,40-
                BRU         REFAVT      0A-02-C7,00-
                                        0D-08-0F-
                                        10-00-00,00-
                                        0E-00,05-
                                        12-01-
REFAVT          LDX         =0          0B-05-00,0C-00,05-
                                        0A-02-22,00-
                FIN         VALDEF      02-00,0E-00,00-
                END         NOMLPS      03-D5,D6,D4,D3,D7,E2-00-
```

Exemple 2 :

BT sorti par le compilateur LP15

```
COMMON SECTION CSLP:                    04-C3,E2,D3,D7,40,40-
                                        13-00-
   WORD A;                              14-00-00,00-
   ARRAY 4 BYTE T = ('ABC');            14-07-00,00-
                                        0A-03-C1,C2,C3-
                                        08-00,01-
FIN
EXTERNAL SECTION M%IO,M%EXIT;
SECTION SLP
LDS:                                    02-00,04-00,00-
                                        05-7F,D3,D7,40,40,40-
                                        13-00-
   WORD P;                              14-00-00,00-
   LABEL ETI;
LPS:                                    02-00,00-00,00
                                        06-E2,D3,D7,40,40,40-7F,D3,D7,40,40,40-
   ACR:=M%OC;                           0A-02-2E,04
   M%IO;                                07-08-D4,7A,C9,D6,40,40-
```

```
IF(# 41 EQ AC) THEN GOTO ETI        0A-02-2B,41-
                                    0A-02-C3,00-
                                    0D-08-0F-
                                    0E-00,01-
                                    12-01-
                                    0A-02-C7,00-
                                    0D-08-0F-
                                    0E-00,01-
                                    12-01
   ELSE AC:=0;                      0A-02-C7,00-
                                    0D-08-0F-
                                    0E-00,02-
                                    12-01-
                                    0B-05-00,0C-00,00-
                                    0A-02-20,00-
                                    0B-05-00,0E-00,02-
   ETI: M%IO;                       0B-05-00,0E-00,01-
                                    07-08-D4,7A,C9,D6,40,40-
FIN                                 0A-02-F1,00-
MAIN SECTION MSLP
LDS:                                02-00,12-00,00-
                                    13-00-
   WORD P1;                         14-00-00,00-
LPS:                                02-00,00-00,00-
                                    06-D4,E2,D3,D7,40,40-7F,E2,D3,D7,40,40-
   SLP;                             07-08-E2,D3,D7,40,40,40-
   M%EXIT;                          07-08-D4,7A,C5,E7,C9,E3-
FIN                                 0A-02-F7,01-
STOP                                02-00,06-00,00-
                                    03-D4,E2,D3,D7,40,40-00-
```

Annexe C – Traitement des overlays

C-1 GENERALITES

Un programme ayant une structure d'overlays est composé de deux parties principales :

- La racine qui est composée d'une ou plusieurs sections à la suite desquelles l'éditeur de liens disque rajoute une section nommée "SHUNT".

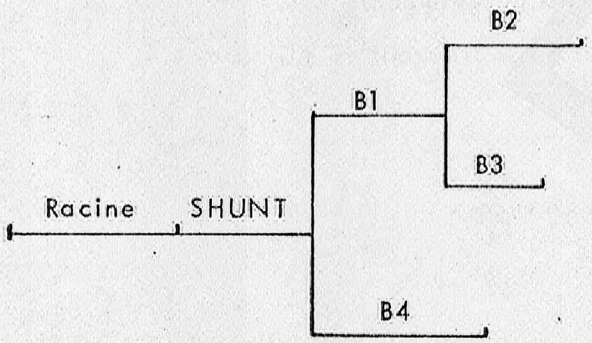
- Les branches d'overlays dont chacune est composée d'une ou plusieurs sections.

Lors du chargement du programme en mémoire, seule la racine est chargée, chaque branche d'overlay n'étant chargée que lorsqu'elle est appelée par un CLS.

C-2 DESCRIPTION

Pour décrire un programme ayant une structure d'overlays, on utilise une représentation sous forme d'arbre la racine en étant le tronc et les divers overlays en étant les branches.

Exemple :



où

Racine est la racine du programme
B1 est la branche d'overlay numéro 1
B2 est la branche d'overlay numéro 2
B3 est la branche d'overlay numéro 3
B4 est la branche d'overlay numéro 4

La structure d'overlay permet, dans le cas où des sections ne sont pas utilisés simultanément par le programme, de n'avoir en mémoire que les sections utiles à un moment donné de l'exécution.

Dans l'exemple donné :

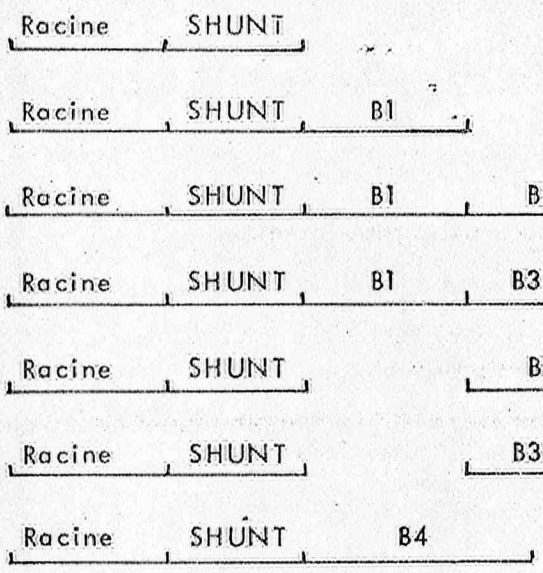
- La racine peut soit appeler en mémoire seulement l'une des quatre branche d'overlay, soit les branches d'overlays numéro 1 et 2 simultanément, soit les branches d'overlays numéro 1 et 3 simultanément.

- La branche d'overlay numéro 1 peut soit appeler la branche d'overlay numéro 2, soit la branche d'overlay numéro 3.

- Les branches d'overlays numéro 2 et 3 peuvent appeler la branche d'overlay numéro 1.

- La branche d'overlay numéro 4 ne peut appeler aucune des autres branches d'overlays.

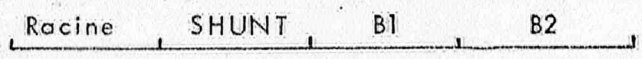
On peut donc envisager les diverses configurations suivantes en mémoire à un moment donné de l'exécution du programme :



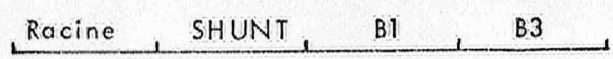
Il ressort de l'exemple ci-dessus que :

- Lorsque deux branches d'overlays sont parallèles, elles ne peuvent coexister en mémoire. Il y a recouvrement de l'une par l'autre lors des divers appels. Par exemple, si la branche B2 se trouve en mémoire au moment de l'appel de la branche B3, cette dernière recouvrira la précédente.

Phase 1 : La racine, la section SHUNT et les branches B1 et B2 se trouvent en mémoire.

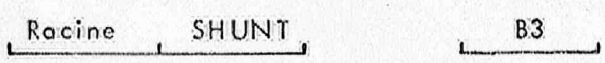


Phase 2 : La branche B3 est appelée et chargée en mémoire en recouvrant la branche B2.

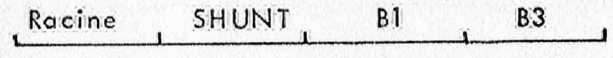


- Il n'est pas nécessaire pour qu'une branche d'overlay puisse être appelée que celle qui doit la précéder en mémoire soit déjà chargée. Le fonctionnement suivant est en effet possible .

Phase 1 : La racine appelle la branche B3.



Phase 2 : La branche B3 appelle la branche B1



Nota :

Pour retourner d'une section appelante à une section appelée ne se trouvant pas dans la même branche d'overlay, il suffit de faire un RTS.

C-3 FONCTIONNEMENT

C-3.1. Déroulement des opérations

Lors de la création de la PRT par l'éditeur de liens disque, celui-ci remplace les SRD des sections de trouvant en overlay par la SRD de la section SHUNT. Il remplit d'autre part la table PRT2 se trouvant dans le

LDS de la section SHUNT avec les SRD des sections se trouvant en overlay.

Au moment de l'appel d'une section par CLS, si celle-ci ne se trouve pas en mémoire, c'est donc la section SHUNT qui est lancée. Elle effectue alors les actions suivantes :

- Protection des registres A, E et X
- Appel de la section moniteur M:OVLD
- Appel de la section moniteur M:LOAD qui charge en mémoire la branche demandée.
- Restitution des registres A, E et X.

En cas de chargement incorrect, sortie du message :

%%LDOA

et abandon du programme.

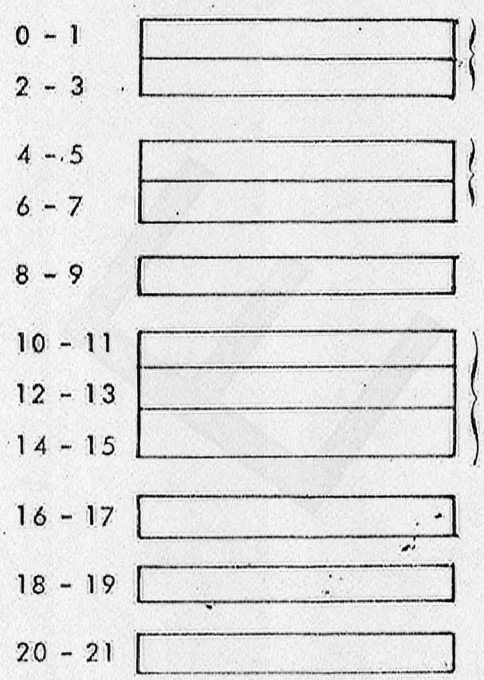
Remarque :

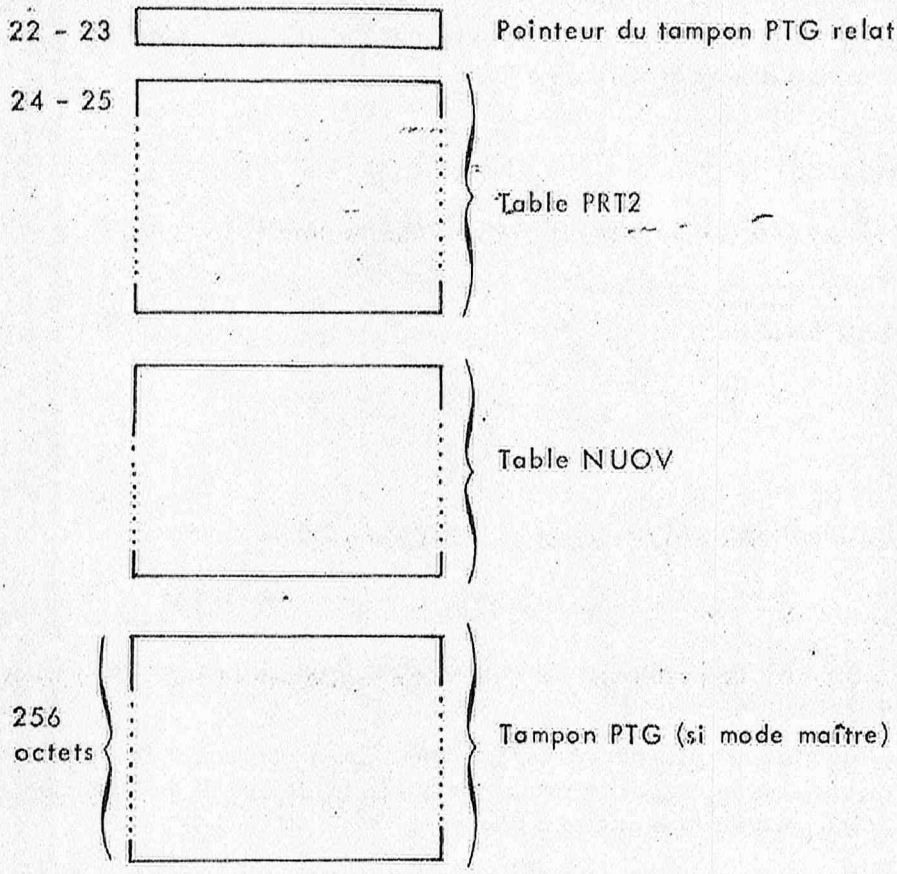
La section SHUNT appelle les modules superviseur M:OVLD, M:LOAD et M:SPER.

C-3.2. Fonctions de la section M:OVLD

- L'adresse de retour est décrémentée de 2 afin de permettre de réexécuter l'appel à la section se trouvant en overlay après le chargement effectif de la branche.
- Le numéro de la section en overlay appelée est pris par M:OVLD dans le mot contenant le CLS. Le numéro est diminué du nombre de sections contenues par la racine, et le résultat sert de pointeur pour aller chercher dans la table MUOV le numéro de la branche à charger.
- Recherche des branches recouvertes (totalement ou en partie) par la branche à charger. Détermination des sections contenues par ces branches et remplacement de leurs SRD par celle de la section SHUNT.
- Détermination des numéros de sections contenues par la branche à charger et chargement de leur SRD propre dans la PRT à l'aide du contenu de la table PRT2.
- Chargement de l'interface d'appel pour M:LOAD.

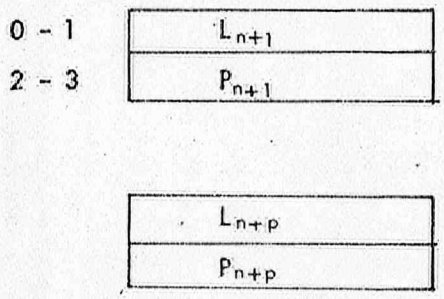
C-3.3. Contenu du LDS de la section SHUNT





■ Format de la table PRT2

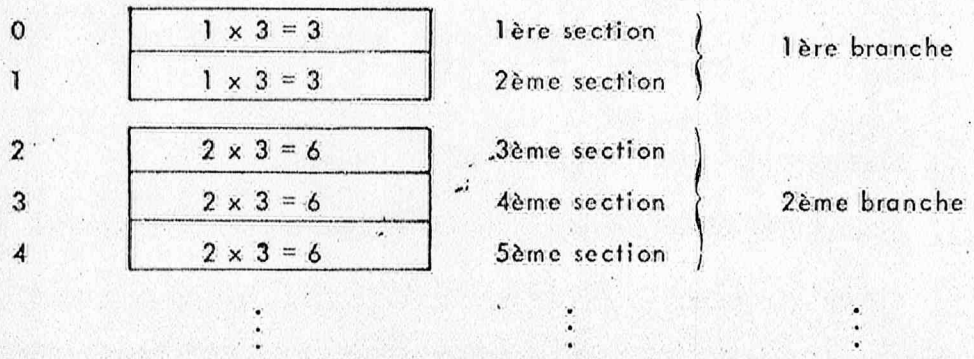
Contient les SRD des sections contenues dans les branches d'overlay dans l'ordre ou elles sont décrites dans les cartes %TREE.

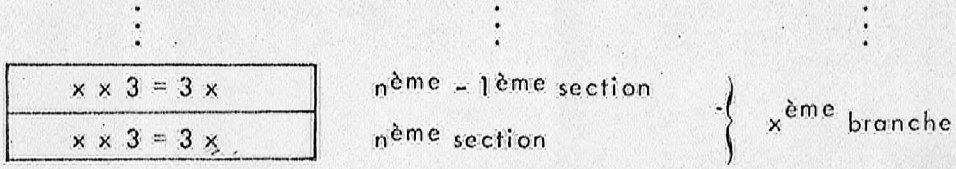


n = nombre de sections contenues par la racine

p = nombre de sections contenues par les branches d'overlay

■ Format de la table NUOV





■ Tampon PTG

Dans le cas d'un programme édité en mode maître cette table a une longueur de 256 octets.

Dans le cas d'un programme édité en mode esclave, cette table a une longueur de 0 octets.