Réseaux
et systèmes
d'information

Bull

MANUEL D'UTILISATION

SOLAR

IOCS16

Moniteur de gestion des entrées-sorties

LOGICIEL

LOGICIEL

LOGICIEL

LOGICIEL

LOGICIEL

Bull

1 164 213 02 030

Réseaux
et systèmes
d'information

Bull

Manuel d'utilisation

Bull
Sems

SOLAR

ENTREES - SORTIES

IOCS16

. . . . . .

Logiciel

SUJET : Moniteur d'Entrées/Sorties des SOLAR 16-35.
16-70, 16-90

OBSERVATION :

VERSION LOGICIEL :

DATE : MAI 1985

REF Bull-Sems : 1 164 213 02 030 02 /FR

Réseaux
et systèmes
d'information

Bull

Bull
Sems

Imprimé en France

| Vos suggestions sur la forme et le fond de ce manuel seront les bienvenues. Une feuille destinée à recevoir vos remarques se trouve à la fin du présent manuel.

- TABLE DES MATIERES -

1 DESCRIPTION D'IOCS16

1.1 DESCRIPTION SOMMAIRE DU FONCTIONNEMENT

Du point de vue fonctionnel, IOCS16 comporte deux types de modules :

a) Un module se déroulant sous niveau software de la tâche appelante.

Ce module est activé par le superviseur chaque fois qu'une tâche fait une SVC IOCS.
Son rôle est le suivant :

- il analyse la demande en vérifiant l'IOCB
- il attribue au demandeur les ressources nécessaires à l'échange (unité physique, buffer du "pool")
- il active le module du driver chargé d'initialiser les échanges sur le périphérique concerné
- il analyse le compte-rendu du driver après qu'il ait initialisé les échanges
- suivant le type de demande rend le contrôle à l'appelant ou le met en attente d'événement fin d'échange.

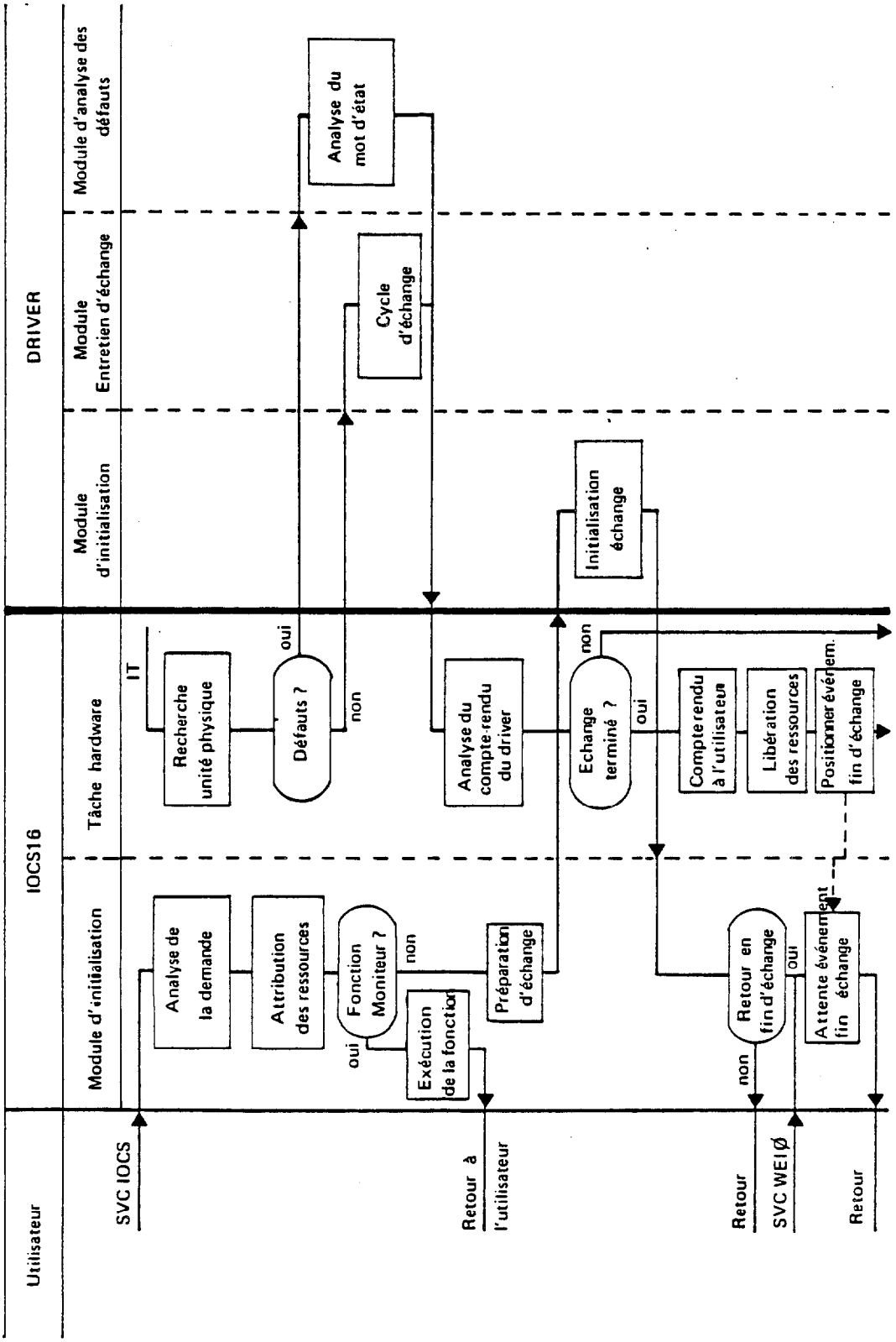
b] Des tâches hardware de niveaux déterminés à la configuration du système.

Ces tâches prennent en compte les interruptions en provenance des périphériques.
Leur rôle est le suivant :

- reconnaître l'origine de l'interruption (le périphérique qui a provoqué l'interruption]
- reconnaître la nature de l'interruption (normale ou exception)
- activer le module du driver chargé d'analyser la nature de l'interruption
- en fin d'échange libérer les ressources qui ont été attribuées lors de l'initialisation de l'échange
- positionner l'événement "fin d'échange".

1.2 GESTION DES RESSOURCES ET EVENEMENTS FIN D'ECHANGE

IOCS16 gère des demandes en provenance de plusieurs tâches software portant. éventuellement, sur les mêmes périphériques.
Lorsque plusieurs tâches requièrent une même ressource (périphérique, buffer contexte canal) elles sont mises en attente dans une file suivant leur priorité. En général, ces files sont réalisées à l'aide d'un sémaphore d'exclusion.



DEROULEMENT D'UN ECHANGE

a) Ressource unité physique

Chaque unité physique est considérée comme une ressource gérée par un sémaphore d'exclusion. A chaque unité physique est affecté un sémaphore d'accès à cette ressource (SEMGEN sémaphore à un seul accès). Une ressource unité physique est attribuée à une tâche :

- soit pour le déroulement d'un échange. Dans ce cas l'accès à la ressource unité physique est libéré en fin d'échange
- soit parce que celle-ci a demandé l'attachement (fonction PUA). Dans ce cas la ressource sera libérée sur la fonction détachement (PUD) si l'unité physique était attachée.

b) Ressource "pool buffer"

L'accès au pool est géré par un sémaphore d'exclusion (SEMBUF) dont le nombre d'accès est égal au nombre de buffers du pool.
Lorsqu'une tâche fait une demande d'échange avec utilisation du "pool buffer", on lui attribue un accès à cette ressource. Cet accès sera libéré en fin d'échange.

c) Ressource "contexte canal"

Les canaux HDC peuvent gérer 8 échanges simultanés pour les UT 16-70 et 16-90. Pour les IOP 16-M, ils sont limités à 2 et dans ce cas l'accès à un tel canal est filtré par un sémaphore d'exclusion à deux accès. L'UT 16-35 ne dispose que d'un seul canal HDC. L'attribution de la ressource "contexte canal" n'est faite que si le périphérique est connecté à un canal du type HDC.

d) "Fin d'échange"

A chaque demande d'échange (exceptées les demandes avec utilisation du pool buffer), IOCS16 associe un sémaphore privé réalisé dans le mot 4 de l'IOCB correspondant à cet échange. A l'initialisation de l'échange, le compteur d'accès à ce sémaphore est nul. L'accès à ce sémaphore est fourni par la fin d'échange (événement fin d'échange). Lors d'une demande avec retour en fin d'échange, IOCS16 demande l'accès à ce sémaphore avant de rendre le contrôle à l'utilisateur. Si la fin d'échange n'est pas encore arrivée la tâche est suspendue et réactivée par l'événement fin d'échange. Il en est de même lorsqu'une tâche se met en attente de fin d'échange sur une SVC WEIO.

- on associe en outre à chaque unité physique un sémaphore privé utilisé pour synchroniser les demandes d'échange et les fins d'échange portant sur cette unité physique lorsque celle-ci est attachée à une tâche (sémaphore SEMATT).

1.3 LES TABLES SYSTEME

Pour communiquer entre eux, les différents modules d'IOCS16 et les drivers utilisent des tables [Tables Systèmes). Ces tables sont décrites dans le présent paragraphe.

1.3.1 Tables d'unités symboliques : TBS

Cette table est organisée en octets. Elle est pointée par les numéros d'unités symboliques. Elle fait correspondre à chaque numéro d'unité symbolique un numéro d'unité fonctionnelle.
Elle peut être modifiée par le programme superviseur (BOS16 par exemple) lors des affectations d'unités fonctionnelles aux unités symboliques.

TBS

N° FU associée à la SU 0	N° FU associée à la SU 1
N° FU associée à la SU 2	
No FU associée à la SU i	

La taille de cette table (nombre d'octets) est contenue dans la mémoire NUSMAX.
Les unités symboliques sont numérotées à partir de 0.

1.3.2 Table d'unités fonctionnelles : TBF

Cette table et organisée en mots. Elle est pointée par les numéros d'unités fonctionnelles et fait correspondre à chaque numéro d'unité fonctionnelle l'adresse de la table unité physique sur laquelle elle pointe (voir 1.3.5).

TBF

Adresse TUP correspondant à la FU1
Adresse TUP correspondant à la FU2
Adresse TUP correspondant à la FUi

La taille de cette table est contenue dans la mémoire NUFMAX. Les unités fonctionnelles sont numérotées à partir de 1.
Cette table est initialisée à la génération.

1.3.3 Table des niveaux d'interruption des unités fonctionnelles : TBNFU

Cette table est organisée en octets. Elle est pointée par les numéros d'unités fonctionnelles et fait correspondre à chaque numéro d'unité fonctionnelle le numéro d'interruption correspondant.

TBNFU

No IT correspondant à la FU = 00	No IT correspondant à la FU = 01
No IT correspondant à la FU = 04	

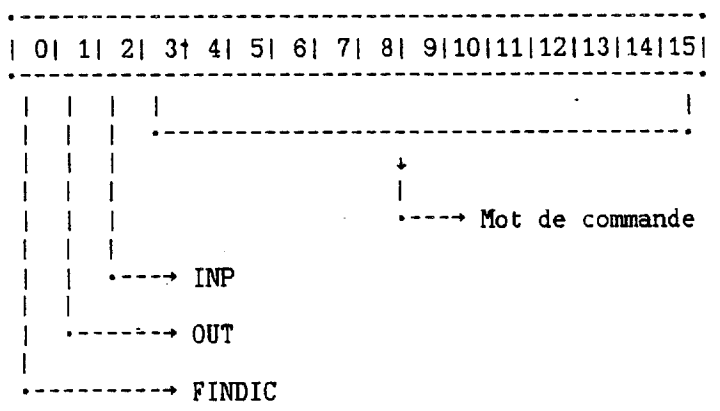
La taille de cette table (en mot) est égale à (NUFMAX + 1) /2. Elle est initialisée à la génération.

1.3.4 Table des mots de commande : TBMCOM

Cette table est organisée en mots. Elle est pointée par les numéros d'unités fonctionnelles et fait correspondre à chaque numéro d'unité fonctionnelle d'une part un mot de commande d'autre part un certain nombre d'indicateurs caractérisant cette unité fonctionnelle.

TBMCOM	Mot de commande associé à la FU1
	Mot de commande associé à la FU2
	Mot de commande associé à la FUi

Forme d'un élément de la table :



Signification des indicateurs :

INP (bit 2) :
- 0 les entrées sont autorisées sur l'unité fonctionnelle
- 1 les entrées sont refusées sur l'unité fonctionnelle

OUT (bit 1) :
- 0 les sorties sont autorisées sur l'unité fonctionnelle
- 1 les sorties sont refusées sur l'unité fonctionnelle

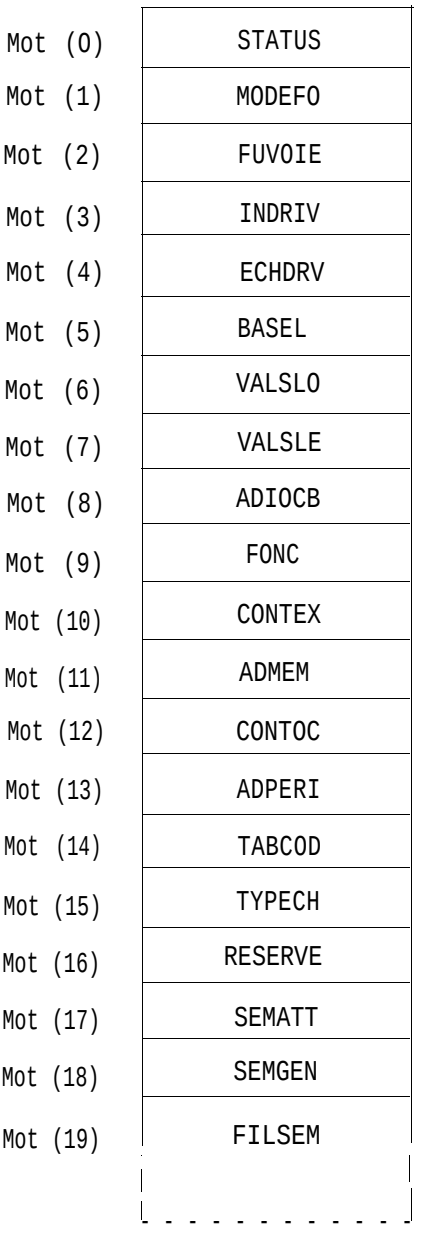
Ces deux indicateurs permettent à IOCS16 de s'assurer que le sens de l'échange est possible sur le périphérique concerné.

FINDIC (bit 0) : Indicateur servant éventuellement au driver pour distinguer une unité fonctionnelle particulière lorsqu'il est utilisé par plusieurs unités fonctionnelles. Il s'agit d'une convention avec le driver. Par exemple ce bit sera à 1 pour l'unité fonctionnelle TP, ce qui permettra au driver "bande papier" de savoir qu'il doit envoyer les codes "perfo on" et "perfo off" en début et fin d'échange.

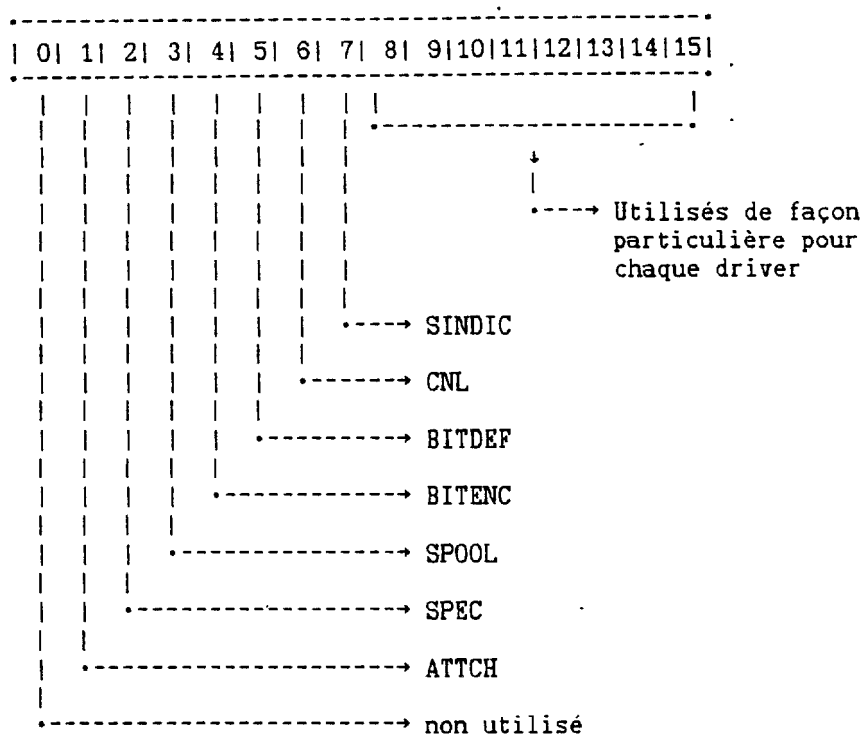
La taille de cette table (nombre de mots) est contenue dans la mémoire NUFMAX. La table est initialisée à la configuration.

3.5 Table unités physiques TUP xxx

A chaque unité physique est associée une table d'unité physique TUP xxx qui contient toutes les informations (statiques et dynamiques] propres à une unité physique pour un échange donné. C'est là que le driver trouvera toutes les données propres à un échange.



Mot (0) : STATUS

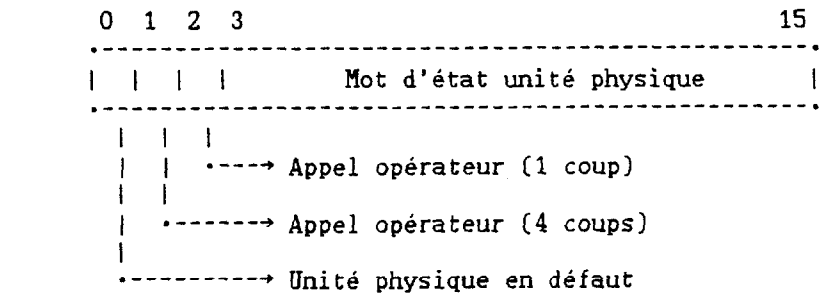


Signification des différents indicateurs lorsqu'ils sont à 1

ATTCH	L'unité physique est attachée (bit modifié par une fonction spéciale adressée au moniteur).
SPEC	Ce bit est réservé pour une fonction spéciale non standard que l'utilisateur voudrait inclure à son système.
SPOOL	L'échange demandé utilise le pool de buffers. Ce bit est positionné à 1 ou 0 à chaque demande à IOCS16.
BITENC	Un échange est en cours sur l'unité physique.
BITDEF	Un défaut a eu lieu en cours d'échange, obligeant à abandonner l'échange. Ce bit est positionné à 1 par le module d'analyse des défauts du driver.
CNL	L'unité physique fonctionne en mode canal (bit positionné à la configuration).
SINDIC	Indicateur caractérisant l'unité fonctionnelle en échange sur l'unité physique. C'est en fait le BIT FINDIC qui est recopié par IOCS16 à chaque demande d'échange ou de positionnement. Par exemple : le driver "bande papier" trouvera ce bit à 1 lorsqu'il travaillera pour l'unité fonctionnelle TP ce qui lui permettra de savoir qu'il doit envoyer les codes "perfo on" et "perfo off" en début et fin d'échange. Chaque driver pourra l'utiliser de façon personnelle. L'octet de droite est utilisé de façon particulière par chaque driver pour mémoriser des informations.

Les bits 8 à 15 sont utilisés de façon particulière par chaque driver. Cependant, ceux des périphériques gérant l'appel opérateur utilisent les bits 13, 14, 15 pour comptabiliser le nombre d'appels. Lors de la prise en compte d'un appel, les sous-programmes superviseurs (TAPS, RMDEF) doivent mettre à zéro ces trois bits.

Mot (l) : MODEFO



IOCS16 range les mots d'état des unités physiques que lui fournissent les modules d'analyse des défauts des drivers. Ces mots sont scrutés par les différents moniteurs (BOS16 par exemple) qui peuvent alors signaler par un message d'erreur les défauts périphériques qui se sont produits.

Mot (2) : FUVOIE

Octet de gauche : numéro de FU qui est en cours d'échange
Octet de droite : numéro de voie (pour les périphériques multiplexés uniquement).

Mot (3) : INDRIV

Adresse du driver (chargée à la configuration).

Mot (4) : ECHDRV

Adresse à lancer à la prochaine interruption du périphérique (adresse d'entretien d'échange).

Mot (5) : BASEL

Il s'agit de l'adresse sur laquelle pointe la base L dans le local du driver. Cette adresse est chargée dans le mot (5) à la génération.
La base L est chargée par cette valeur chaque fois qu'il fait appel à un module du driver. Les modules des drivers n'auront donc pas à se préoccuper de charger les bases.

Mot (6) : VALSLO

Valeur du registre SLO de l'appelant qui a initialisé l'échange.

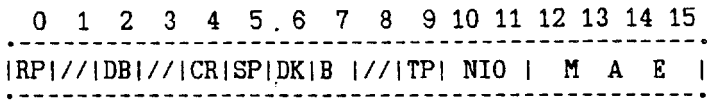
Mot (7) : VALSLE

Valeur du registre SLE de l'appelant qui a initialisé l'échange.

Mot (8) : ADIOCB

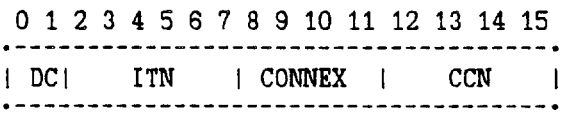
Adresse de l'IOCB pour l'échange considéré.

Mot (9) : FONC



RP : 0	Ce bit est toujours à 0
DB : 1	Mode DEBUG (Positionné par GI016)
CR : 1	La fonction demandée au canal est un compte-rendu
CR : 0	La fonction demandée au canal est une initialisation d'échange
SP : 1	Micro-programmation spécifique
SP : 0	Micro-programmation standard
DK : 1	Le périphérique est un disque à têtes mobiles
DK : 0	Le périphérique n'est pas un disque
B : 0	Canal à mots
B : 1	Canal à octets ou fausse lecture disque (si DK = 1)
TP :	type processeur = 1 s'il s'agit d'un périphérique en mode canal géré par UT 16-70, 16-90 ou IOP16-R
NI0 :	Numéro de processeur d'entrées-sorties qui gère l'échange
MAE :	Extension d'adresse buffer (poids forts)

Mot (10) : CONTEX



DC = 00 : le canal est de type LDC
DC = 11 : le canal est de type MDC
DC = 10 : le canal est de type HDC

ITN : Niveau d'interruption normale

CONNEX : Type de connexion du périphérique
0000 coupleur standard
0001 coupleur asynchrone 1 voie
0010 coupleur multiplexé 4 voies
0011 coupleur multiplexé 16 voies
0100 coupleur multiplexé 8 voies
0101 coupleur CMF
0110 coupleur multiplexé 4 voies MUX4-U

CCN: Numéro de contexte hardware

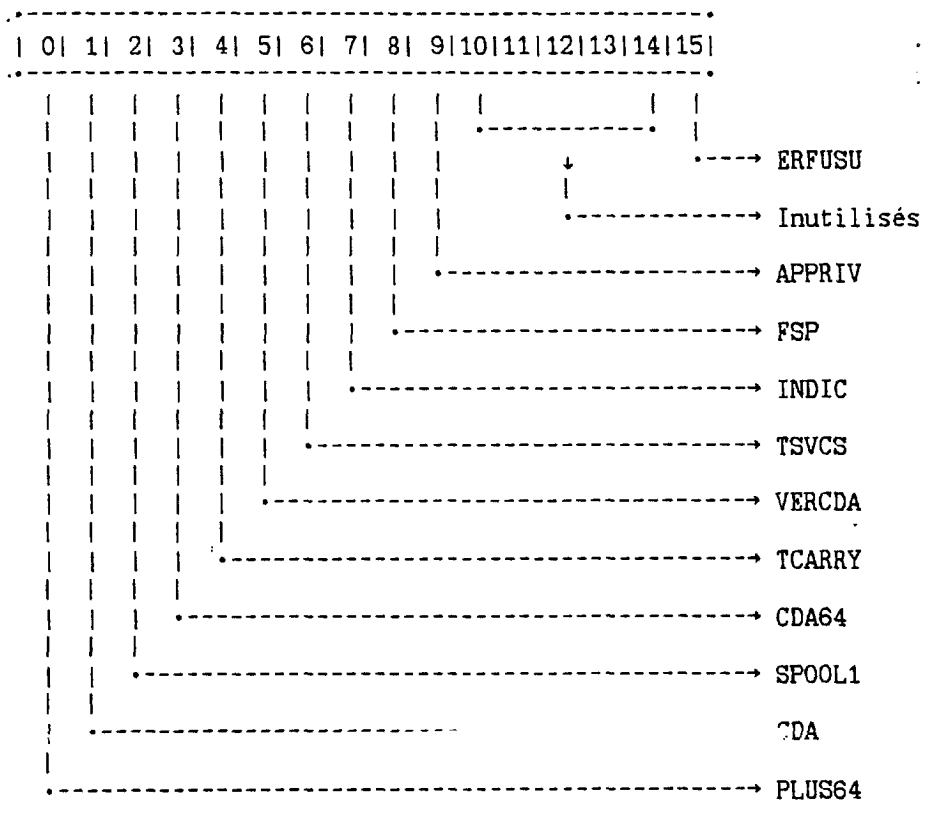
Mot (11) : ADMEM
Adresse mémoire du buffer d'entrées-sorties (poids faibles)

Mot (12) : CONTOC
Compte de mots ou d'octets (suivant le coupleur) à échanger.

Mot (13) : ADPERI
Adresse du périphérique

Mot (14) : TABCOB
Adresse mémoire de la table des codes d'arrêt ou adresse disque (si DK = 1).

Mot (15) : TYPECH



Signification des différents indicateurs lorsqu'ils sont à 1 :

PLUS64	Echange avec interface système
CDA	Echange avec buffer en CDA
SPOOL1	Echange avec utilisation du pool buffer
CDA64	Echange avec interface système ou buffer en CDA
TCARRY	Appelant en mode maître
VERCDA	Vérification d'une adresse en CDA
TSVCS	Valeur du bit SVCS du registre STATUS à l'initialisation
INDIC	Type d'adressage = 0 : adressage d'un IOCB = 1 : adressage d'un buffer
FSP	L'échange en cours est une fonction spéciale
APPRIV	Appelant en mode privilégié
ERFUSU	Unité d'échange non gérée par IOCS16

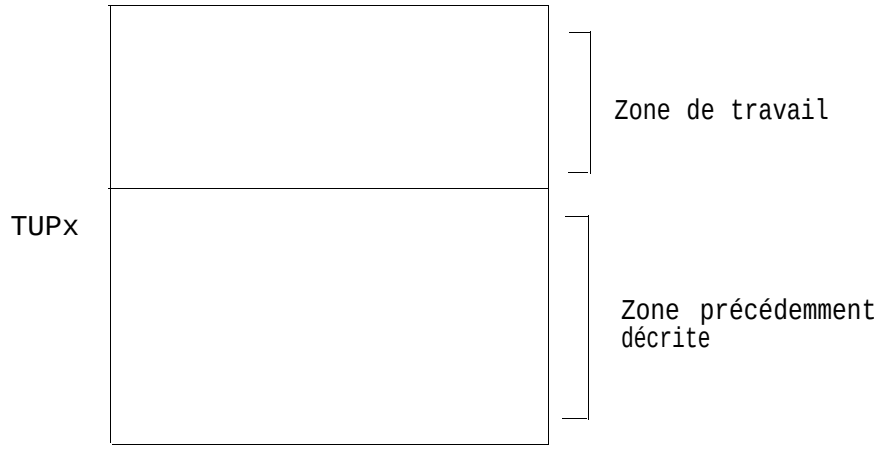
Mot (16): Réservé Usage Futur.

Mot (17) : SEMATT
Sémaphore privé d'attachement.

Mot (18) : SEMGEN
Sémaphore d'exclusion gérant la ressource unité physique.

Mot (19) : FILSEM
Début de la file du sémaphore d'exclusion. Le nombre de mots réservés à la file du sémaphore d'exclusion dépend du nombre des tâches du système auquel il sera inséré (maximum 8 mots c'est-à-dire 128 tâches).

Remarque :
La table unité physique que l'on vient de décrire est la forme la plus simple.
Certains drivers ont besoin d'une zone de travail supplémentaire, la table unité physique la plus générale aura la forme suivante :

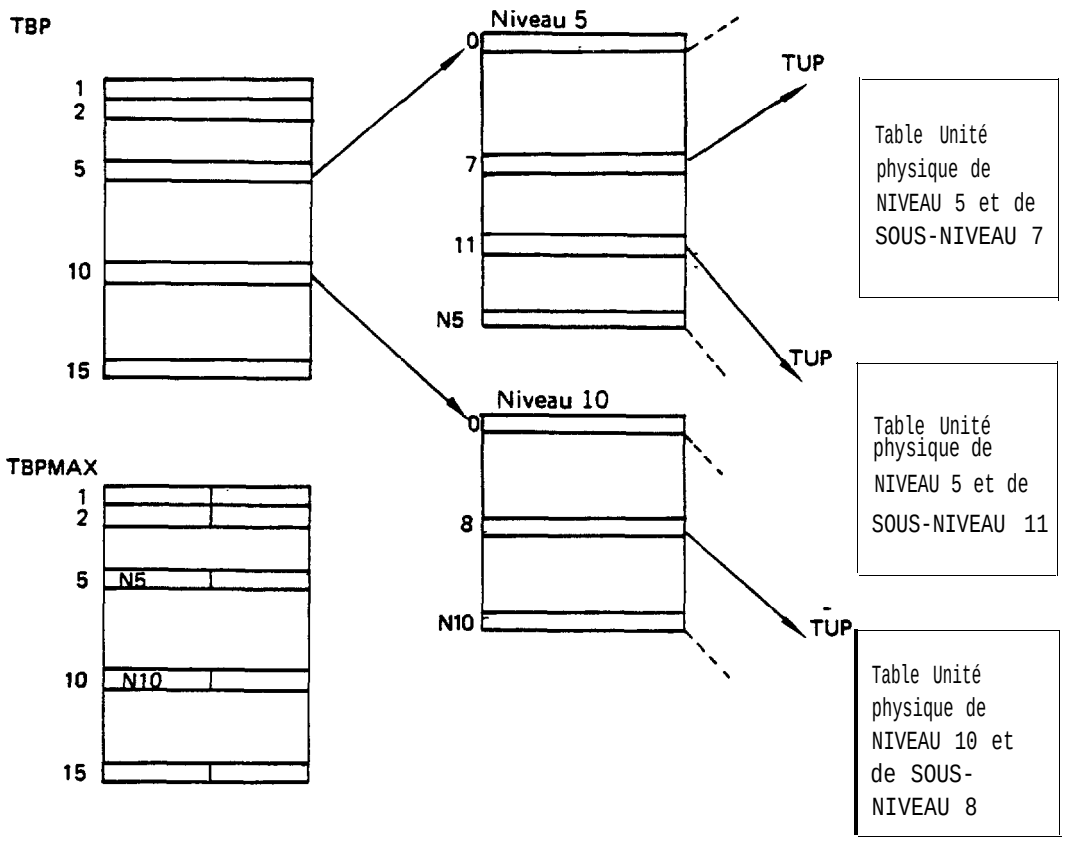


.3.6 Table des adresses de tables unités physiques : TBP

L'accès aux tables d'unités physiques est fait en deux étapes :

A) En fonction du niveau d'interruption traité on recherche, dans la TBP, l'adresse de la table des unités physiques de ce niveau.

B) En fonction du sous-niveau appelant on a l'adresse de la table d'unité physique.



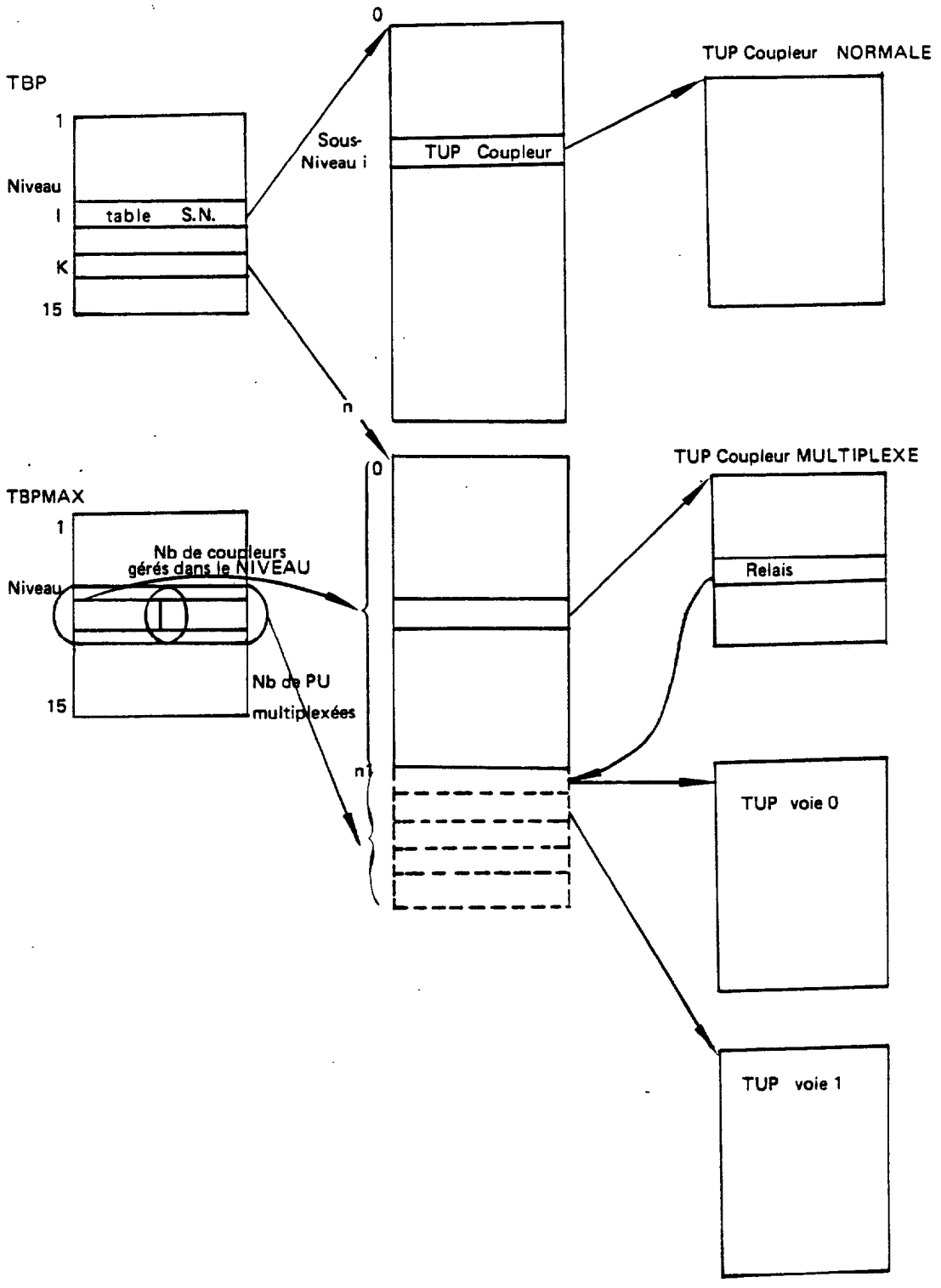
L'octet gauche de la table TBPMAX indique le rang maximum du sous-niveau géré dans chacun des niveaux gérés par IOCS16.
Dans le cas où l'on a plusieurs périphériques multiplexés sur le même coupleur, l'organisation précédemment décrite ne peut plus s'appliquer telle quelle.
En effet, dans ce cas, il faut faire correspondre à un numéro de niveau et de sous-niveau plusieurs tables unités physiques associées aux différents périphériques multiplexés sur le même coupleur.
Pour résoudre ce problème on adoptera l'organisation suivante :

- on associera à ce coupleur multiplexé une table "unité physique fictive" ou "coupleur" dont l'emplacement est déterminé par la méthode ci-dessus.
- on associera à chaque "voie" d'un coupleur multiplexé une table unité physique. Cette table est gérée par le driver en entretien d'échange. Une table unité physique "coupleur" n'étant jamais directement associée à un échange, on peut lui donner une forme simplifiée.

Forme d'une table "unité physique fictive" :

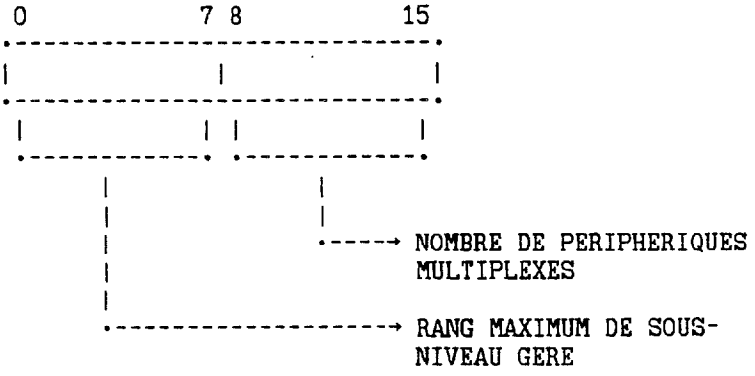
Mot (0)	STATUS
Mot (1)	MODEFO
Mot (2)	FUVOIE
Mot (3)	INDRIV
Mot (4)	ECHDRV
Mot (5)	BASEL
Not (6)	VALSLO
Not (7)	VALSLE
Mot (8)	RELAI
Mot (9)	FONC
Mot (10)	CONTEX
Mot (11)	ADMEM
Mot (12)	CONTOC
Mot (13)	ADPERI

Les mots (1), (2), (6), (7), (9), (10), (11), (12) sont sans signification pour une telle organisation.
Le mot (8) contient l'adresse d'une table indiquant les adresses des unités physiques pour chacune des voies.
Cette organisation permet de faire le lien entre une unité physique coupleur et les tables unités physiques associées aux différentes voies du coupleur multiplexé.


TBP
1
Niveau
I
table S.N.
K
15
0
Sous-
Niveau i
TUP Coupleur
n
TUP Coupleur NORMALE
TBPMAX
1
Niveau
Nb de coupleurs
gérés dans le NIVEAU
15
Nb de PU
multiplexées
0
n
TUP Coupleur MULTIPLEXE
Relais
TUP voie 0
TUP voie 1

1.3.7 Table de description des niveaux hardware d'entrées-sorties (TBPMAX)

- cette table est organisée en mots
- elle est pointée par les priorités des tâches hardware
- elle contient pour chaque niveau géré :
  - . Le rang maximum du sous-niveau géré
  - . Le nombre de périphériques multiplexés dans le niveau.



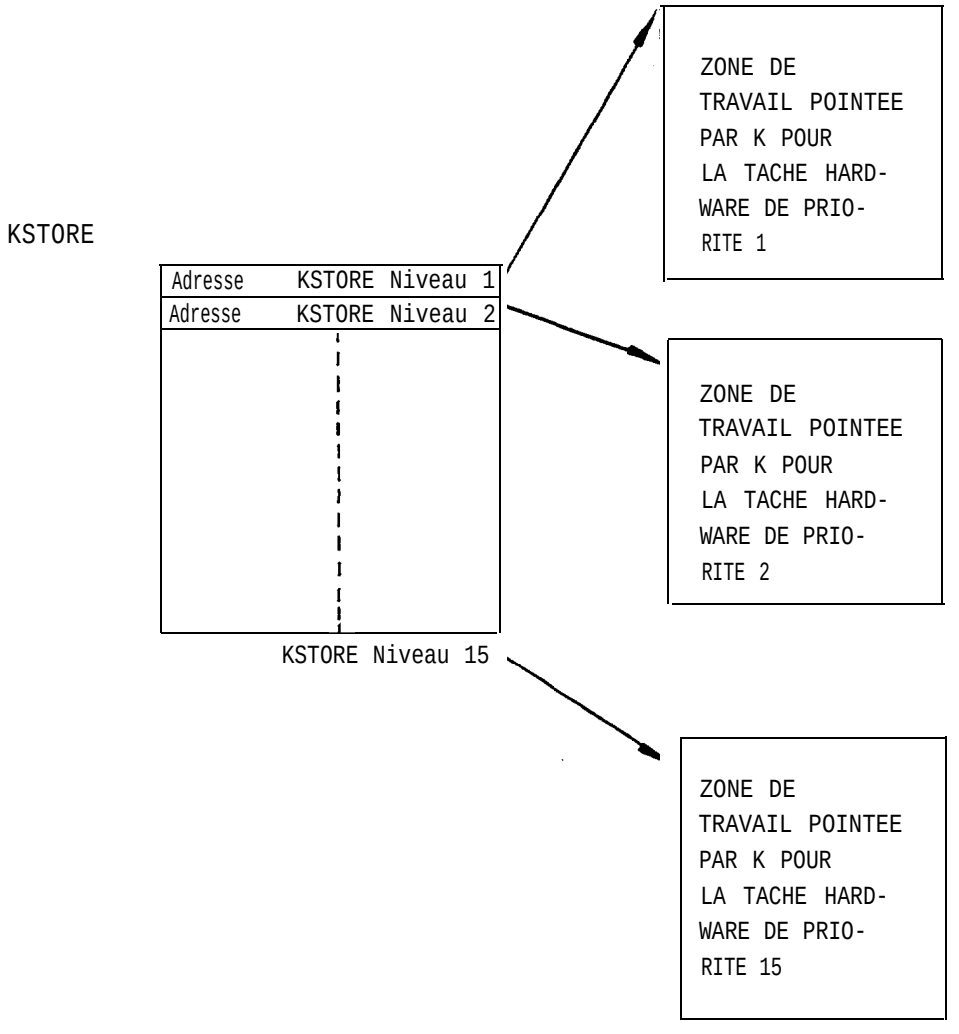
1.3.8 Table des fonctions spéciales moniteur - TBFSM

Cette table est organisée en mots. Elle contient les adresses des sous-programmes exécutant les fonctions spéciales adressées au moniteur.
Elle est pointée par les numéros des fonctions spéciales (bits 2 à 7 de l'octet de fonction).
En standard celui-ci gère un certain nombre de fonctions spéciales.
L'utilisateur pourra intégrer à la configuration du système les fonctions qu'il jugera utiles.
Aux fonctions non existantes correspondront des adresses de programmes nulles.
La taille de la table est contenue dans la mémoire NFSMAX.

TBFSM	Adresse du module exécutant la fonction 0
	Adresse du module exécutant la fonction 1
	Adresse du module exécutant la fonction i

1.8.9 Table des KSTORE des tâches hardware

Les tâches hardware gérées par IOCS16 ont leurs zones de travail pointées par le registre K dans les tables d'IOCS16.



Nota : Pour un niveau non géré par IOCS16, la zone de travail n'est pas réservée dans les tables et la position correspondante dans la table KSTORE vaut 0.

1.4 LES TABLES DE GESTION DU DISQUE

On rappelle qu'un disque peut être partagé en plusieurs. unités fonctionnelles qui sont en fait des portions de disque (Voir Manuel de Référence).
Pour gérer une unité fonctionnelle disque, IOCS16 a besoin de connaître outre les paramètres qui caractérisent une unité fonctionnelle quelconque, les paramètres suivants :

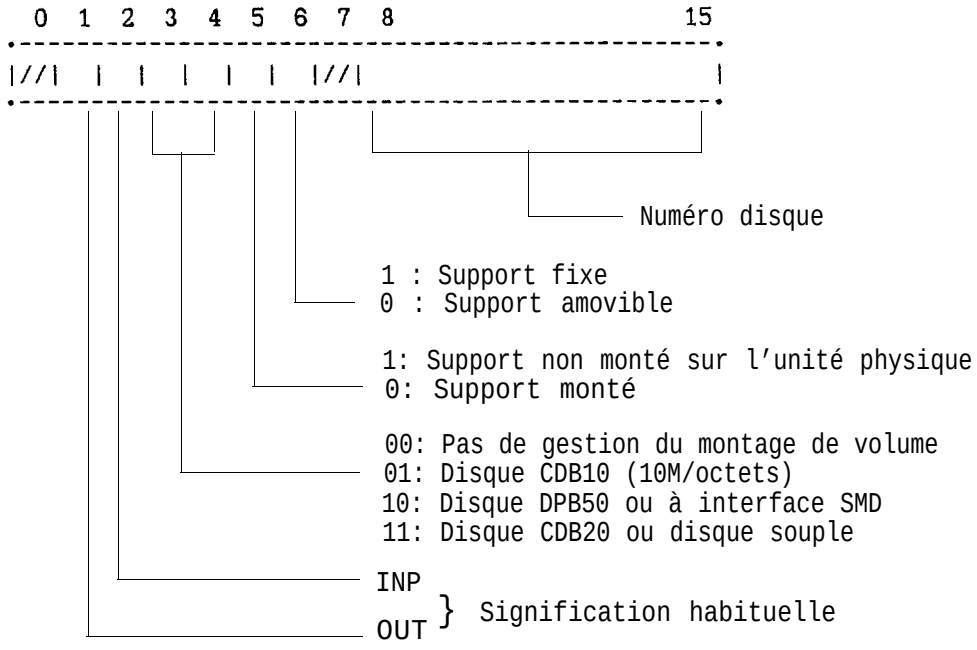
- l'adresse disque du début de la zone affectée à l'unité fonctionnelle.
- le nombre de secteurs de cette zone.

Ces informations sont contenues dans des tables propres à la gestion du disque. Pour accéder à ces dernières on associe à chaque unité fonctionnelle disque un numéro de "disque" (les numéros allant de 0 à n - 1 si n est le nombre d'unités fonctionnelles disque).

1.4.1 Numéro de disque

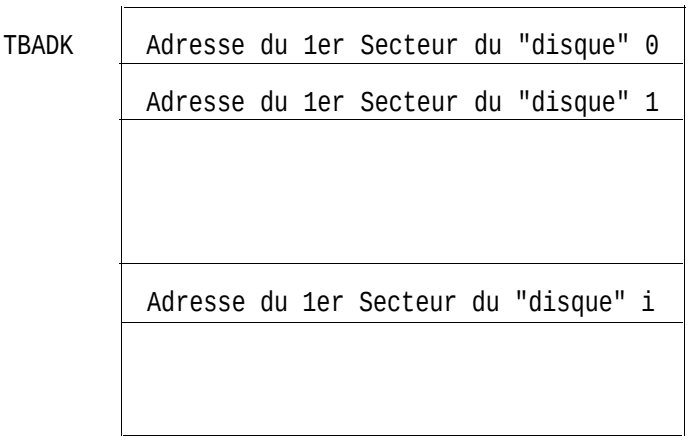
Pour les unités fonctionnelles autres que les unités fonctionnelles disque, on trouve dans la table TBMCOM un mot de commande qui est envoyé avant chaque échange par le driver.
Les échanges disque ne nécessitant pas l'envoi d'un mot de commande, on trouvera pour les unités fonctionnelles disque non pas un mot de commande mais un numéro de disque.

L'élément de la table TBMCOM correspondant à une unité fonctionnelle disque a le profil suivant :



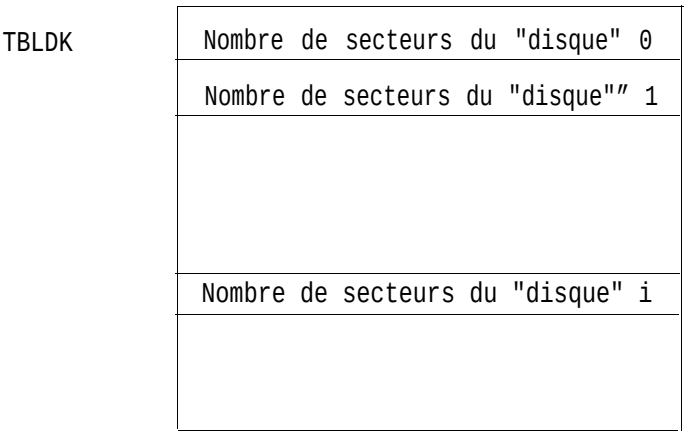
4.2 Table des adresses disque TBADK

- cette table est organisée en mots
- elle est pointée par les numéros de "disque"
- elle contient l'adresse sur le disque "réel" du début de la zone affectée à l'unité fonctionnelle disque
- sa taille est contenue dans la mémoire NDKMAX (nombre maximum d'unités fonctionnelles disque).



1.4.3 Table des longueurs de disque TBLDK

- cette table est pointée par les numéros de "disque"
- elle contient le nombre de secteurs affecté à chaque unité physique
- sa taille est contenue dans la mémoire NDKMAK.



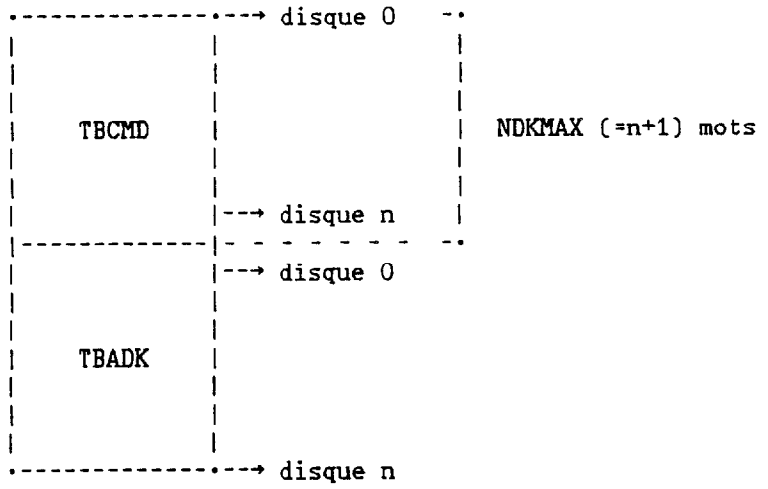
1.4.4 Organisation des tables TBADK et TBLDK pour une organisation "grand disque"

Une organisation est dite "grand disque" lorsque 2 mots sont nécessaires pour en contenir l'adresse ou la taille.
Cette organisation est utilisée dès qu'un disque de moyenne capacité a été décrit.
Les tables TBADK et TBLDK sont découpées en 2 sous-tables :

- la 1ère contenant les adresses ou les longueurs poids faibles
- la 2ème contenant les adresses ou les longueurs poids forts.

1.4.5 Table des mots de commande supplémentaire TBCMD

- cette table est organisée en mots
- elle est pointée par les numéros de disque
- elle est créée pour chaque FU “disque”
- elle contient un mot de commande supplémentaire
- elle se trouve devant la table TBADK et elle a la même longueur (NDKMAX).



- on accède à la table TMCMD par le contenu des mémoires TBADK et NDKMAX. pour le "disque" i

  TBCMD (i) = @ TBADK - NDKMAX + i

- dans le cas des disques souples gérés par le driver DRVFDD les bits 14 et 15 de TBCMD contiennent le numéro de voie (de 0 à 3).

1.5 LES TABLES DU DISPOSITIF “CHIEN DE GARDE IOCS16”

Trois tables sont nécessaires au fonctionnement du dispositif "CHIEN DE GARDE IOCS16".

1.5.1 Table FUSURV :

Table de bytes contenant les numéros des FU déclarées surveillées (dans l’ordre de déclaration).

1.5.2 Table DOGINI :

Table de bytes contenant les valeurs initiales des chiens de garde associés aux FU. Les valeurs sont codées sur 7 bits.
Le bit de poids fort indique s’il s’agit de minutes (=1) ou de secondes.

1.5.3 Table DOGCOM :

Table de mots contenant les compteurs chargés à chaque échange par la valeur initiale du chien de garde convertie en secondes.

2 COMMENT ECRIRE UN DRIVER

Ce chapitre a pour but d'indiquer la procédure à suivre pour la réalisation d'un driver.

2.1 DESCRIPTION GENERALE D'UN DRIVER

En règle générale on peut distinguer sept modules dans un driver :

- lecture mot d'état unité physique
- initialisation d'un échange effectif
- initialisation d'une fonction spéciale de positionnement
- entretien d'un échange effectif
- analyse des défauts
- initialisation de l'unité physique
- "tuer" un échange

Les modules "entretien d'un échange" et "analyse des défauts" sont appelés par les tâches hardware qui gèrent les niveaux E/S. Tous les autres modules se déroulent sous le niveau software de l'appelant.

2.2 LES MODULES D'UN DRIVER

Un module de driver a une structure de sous-programme. Généralement ces modules sont réentrants.
Tout driver doit disposer de deux points d'entrée.

2.2.1 Entrée dite "INITIALISATION"

Ce point d'entrée est commun à plusieurs modules il est paramétré par le registre d'index.

- lecture mot d'état périphérique (PUSI) X = 0
- initialisation d'un échange (INIT) X = 1
- initialisation d'une fonction spéciale de positionnement (FSP) X = 2
- traitement des défauts (DEF) X = 3
- initialisation de l'unité physique (SCLEAR) X = 4
- tuer un échange (KILL) X = 5

2.2.2 Entrée dite "ENTRETIEN"

C'est le point d'entrée dans un driver, sous niveau hardware, pour un périphérique fonctionnant en programmé prioritaire. Il est à noter que, pour un périphérique fonctionnant en mode canal, cette séquence ne sera jamais activée.

2.3.1 Lecture mot d'état

Le but de cette fonction est de transmettre à l'appelant le mot d'état unité physique.

2.3.2 Initialisation d'un échange

Ce module doit réaliser les fonctions suivantes :

- tester l'occupation du coupleur de façon à savoir s'il doit ou non poursuivre l'échange. Dans le cas où le périphérique est occupé à l'initialisation, le driver utilisera généralement la prochaine interruption pour relancer l'échange.
- en fonction des modalités de l'échange, il met à jour l'adresse du périphérique dans le mot ADPERI de la table d'unité physique.
- pour les périphériques fonctionnant en programmé prioritaire, il met à jour dans le mot ECHDRV de la TUP l'adresse du module qui sera activé lors de la prochaine interruption pour l'entretien de l'échange.
- pour les périphériques fonctionnant en canal, il réserve un contexte canal (cette fonction est réalisée par le sous-programme d'IOCS16 SPCCN) s'ils sont gérés par UT 16-35 ou IOP 16-M.
- dans le cas où le périphérique fonctionne en mode canal, le module initialise les registres du canal en vue de l'échange (cette fonction est réalisée par le sous-programme d'IOCS16 IPINI).
- il envoie le mot de commande associé à la FU qui pointe sur l'unité physique concernée (ce mot de commande peut être modifié par le module, en fonction des paramètres de l'échange).

2.3.3 Entretien d'un échange

Ce module est activé pour les périphériques fonctionnant en programmé prioritaire. Il doit assurer les fonctions suivantes :

- réaliser les opérations d'entrées-sorties proprement dites
- mettre à jour les données de l'échange dans la TUP (compte d'octets, adresse du prochain mot à transférer).
- déterminer le module à activer lors de la prochaine interruption et charger l'adresse de ce module dans le mot ECHDRV de la table d'unité physique.

2.3.4 Initialisation d'une fonction spéciale de positionnement

Ce module se déroule sous niveau software, il doit interpréter et initialiser la fonction requise de la façon suivante :

- tester l'occupation du périphérique
- lancer la commande correspondant à la fonction demandée
- charger dans ECHDRV l'adresse du module qui sera lancé à la prochaine interruption.

2.3.5 Traitement des défauts

1) Périphérique fonctionnant en programmé prioritaire.
Ce module devra lire le mot d'état périphérique et éventuellement le remettre en forme pour en faire un mot d'état "unité physique" et le transmettre à IOCS16. Ce mot d'état doit être analysé de la façon la plus complète. En effet c'est à ce module de décider si l'échange doit ou non être interrompu. Si un défaut fatal est détecté le module positionnera BITDEF à 1 dans le STATUS de la TUP.

2) Périphérique fonctionnant en mode canal :
Après demande de compte-rendu canal (sous-programme SPLIB d'IOCS16), le driver détermine l'origine de l'interruption.
- fin d 'échange : le compte d'octets restant à transmettre est nul. Il ne s'agit pas, alors, d'un défaut bien que délivré par une interruption exception.
- défaut : le compte d'octets restant à transmettre n'est pas nul. C'est alors au driver de décider si l'échange est, ou non, fatal.

En fin d'échange le contexte canal doit être libéré (sous-programme LIBCCN d'IOCS16).

2.3.6 Initialisation

Ce module doit initialiser les sémaphores de la TUP :

- sémaphore d'attachement (SEMATT) par la valeur 1
- sémaphore d'exclusion (SEMGEN) par la valeur 1 et mise à 0 de la file d'attente.

Ces fonctions sont réalisées par le sous-programme INITP d'IOCS16.

2.3.7 "Tuer" un échange

Cette fonction intervient lorsque l'on désire mettre fin à un échange. Suivant le type de périphérique, le driver prend l'initiative du traitement à effectuer.
Dans le cas où le périphérique dispose du mot de commande "RESET COUPLEUR", le driver l'envoie au coupleur. Dans le cas contraire, ou s'il n'est pas recommandé de l'envoyer (bande magnétique par exemple), le driver devra récupérer et acquitter les interruptions ultérieures intervenant hors échange.
Dans tous les cas avant le retour, le driver devra effectuer les traitements suivants :

- libération des ressources (retour à IOCS16 avec X = 2)
- réveil de la tâche qui était en attente de fin d'échange.

Remarque : l'envoi de la commande "RESET COUPLEUR" inhibe les interruptions ultérieures sur celui-ci.

2.4 INTERFACE IOCSl6-DRIVER

2.4.1 Initialisation :

C = Valeur de la base C d'IOCS16
L = Valeur de la base L du driver
W = adresse de la table d'unité physique
X = fonction è activer.

PUSI X = 0: (lecture mot d'état)
INIT X = 1 : (initialisation d'un échange effectif)
FSP X = 2 : (initialisation d'une fonction spéciale de positionnement)
DEF X = 3 : (traitement des défauts sous-niveau hardware)
CLEARS X = 4 : (initialisation d'une table d'unité physique)
KILL X = 5 : ("Tuer" un échange).

Autres paramètres

a) Appelant en mode maître et SVCS = 0
SLO = 0
SLE = 'FFFF (0 si pas de DRPS)
Y = adresse absolue de l'IOCB

b) Appelant en mode maître et SVCS = 1
SLO appelant
S L E "
Y = adresse absolue de l'IOCB

c) Appelant en mode esclave
SLO appelant
S L E "
Y = adresse absolue de l'IOCB si l'échange est en IBMOD (utilisation du pool buffer)
Y = adresse relative de l'IOCB sinon.

Autres fonctions réalisées

Les informations suivantes, contenues dans la table d'unité physique, sont mises à jour par le noyau avant l'activation du driver :

- adresse de l'IOCB (ADIOCB) : même valeur que le registre Y
- adresse de la table d'échange (ADMEM) : cette adresse est absolue si :

  . L'appelant est en mode maître et SVCS = 0
  . L'échange a lieu en IBMOD
  . Le périphérique fonctionne en mode-canal.

Elle est relative dans les autres cas.

- adresse de la table des codes d'arrêt : il s'agit d'une adresse absolue si l'appelant est en mode maître et d'une adresse relative s'il est en mode esclave ou privilégié.
- numéro de FU (octet gauche de FUVOIE) sur lequel porte l'échange.
- compte d'octets (CONTOC) :
  - bit 0 = 0 lecture
    = 1 écriture
  - bit 1 = 0 échange normal
    = 1 échange multi C.A étendu
  - bits 2 à 15 = compte d'octets maximum ou LBUFI
- valeur du registre SLO de l'appelant (VALSLO) : il a la valeur 0 pour un appelant en mode maître (avec SVCS = 0), il a la valeur du registre SLO de l'appelant sinon.
- valeur du registre SLE de l'appelant (VALSLE) :
  il a la valeur - 1 ('FFFF) pour un appelant en mode maître avec SVCS = 0, il a la valeur de SLE de l'appelant dans les autres cas.

Valeur de X	Signification du compte-rendu	Mesures prises par IOCS16
0	Echange ou fonction de positionnement en cours	IOCS16 ne fait rien
1	Transmission d'un mot d'état sur les bits 5 à 15 de A avec eventuellement BITDEF = 1	- Mémorisation du mot d'état dans MODEFO - Si BITDEF = 1 abandon de l'échange (libération des ressources) avec compte-rendu éventuel dans le mot (3) de I'IOCB en cours
2	Défaut de fonctionnement du coupleur (exemple : coupleur répond occupe alors qu'il devrait être libre)	- Abandon de l'échange (libération des ressources) - Eventuellement compte-rendu dans le mot (3) de I'IOCB en cours
3	Défaut de fonctionnement du canal (exemple fin d'échange sur compte d'octets non nul)	- Abandon de l'échange (libération des ressources) - Eventuellement compte-rendu dans le mot (3) de I'IOCB en cours
4	Inutilisé	IOCS16 ne fait rien
5	Demande impossible à satisfaire dans la position actuelle du périphérique. Exemple : Demande d'écriture d'un bloc alors qu'on est en fin de bande sur un dérouleur.	- Abandon de l'échange (libération des ressources) - Eventuellement compte-rendu dans le mot (3) de I'IOCB Le mot d'état transmis en compte-rendu sera le dernier mot d'état mémorisé dans MODEFO
6	Fin d'échange normale ou fin de fonction de positionnement	- Libération des ressources - Eventuellement compte-rendu dans le mot (3) de I'IOCB
7	Paramètres incorrects	- Abandon de l'échange (libération des ressources) - Eventuellement compte-rendu dans le mot (3) de I'IOCB

c = valeur de la base C d'IOCS16
L = valeur de la base L du driver
W = adresse de la table d'unité physique
SLO-SLE = valeurs prises dans la table d'unité physique et décrites au paragraphe précédent.

Nota : Dans le cas de périphériques multiplexés la base W ne contient pas l'adresse de la table d'unité physique mais l'adresse de la table d'unité physique du coupleur (ou TUP coupleur). Il en est de même pour les registres SLO et SLE qui doivent être initialisés avec les valeurs contenues dans VALSLO et VALSLE de la table d'unité physique de la voie. Tous les registres non sauvegardés par contexte, le sont systématiquement en début de la tâche hardware d'IOCS16.

2.5 INTERFACE DRIVER IOCS16

- le module rend le contrôle à IOCS16 par une instruction RSR
- il peut restituer l'ensemble des registres dans un état quelconque exceptés :

C : qui doit rester inchangé
W : qui doit contenir l'adresse de la table unité physique
X : qui contient un compte-rendu (voir tableau de la page précédente)
En fin d'échange, le driver doit transmettre à IOCSl6, dans CONTOC, le compte d'octets restant à transmettre (Le bit 0 doit toujours être à 0).

2.6 INTEGRATION D'UN DRIVER A IOCS16

Un driver doit être structuré comme un segment comportant :

- un local
- une section programme
- éventuellement une section table.

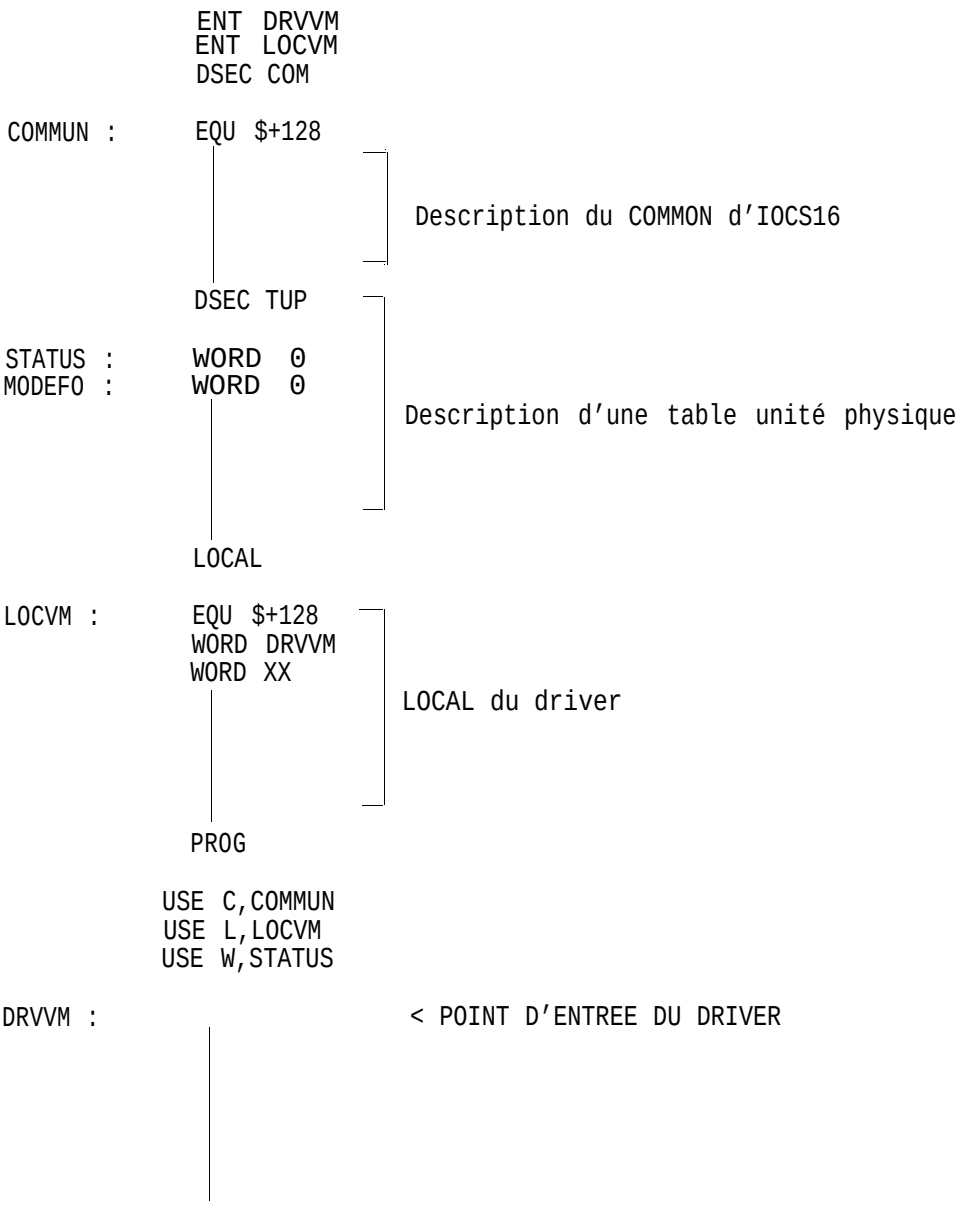
Il sera assemblé isolément puis relié à IOCS16 par édition de liens.
Il pourra utiliser le commun d'IOCS16 et devra alors déclarer cette section en DUMMY ou DSEC.

Règles à observer :

- le premier mot implanté est le premier mot du local du driver
- ce premier mot contient l'adresse de lancement initial du driver
- les mots supplémentaires sont toujours au-dessus de la TUP.
- les symboles à définir en ENTRY sont :

DRVxxx	pour le point d'entrée du driver initialisation
LOCxxx	pour l'adresse sur laquelle pointera la base L
ECHxxx	pour le point d'entrée du driver en traitement d'interruption (uniquement pour les périphériques multiplexés)
xxx	est le type de périphérique.

Exemple : Driver disque (VM)



3 COMMENT ECRIRE UN MODULE DE FONCTION SPECIALE MONITEUR

Certaines fonctions spéciales moniteur sont traitées en “standard” (voir manuel de référence IOCS16).

Cependant, un utilisateur peut très simplement intégrer une nouvel le fonction spéciale.

3.1 INTERFACE IOCS16 - FONCTION MONITEUR ET FONCTION MONITEUR - IOCS16

3.1.l Appel d’un module fonction spéciale

La table des adresses de programme de fonctions spéciales moniteur (TBFSM) permet à IOCS16 d’associer à chaque numéro de fonction spéciale une adresse de programme.

Cette table est chargée à la configuration. IOCS16 donne le contrôle à un module de fonction spéciale par une instruction BSR.

Il lui fournit à l’entrée les registres suivants :
C : pointe sur le COMMON
Y : pointe sur l’IOCB correspondant à la demande
W : pointe sur la table unité physique correspondant à la demande.

Remarque :
Dans le cas où un module de fonction spéciale a besoin d’une section LOCAL il devra lui-même charger sa base L contrairement à un module de driver. En effet IOCS16 ignore l’adresse du LOCAL associé à cette fonction spéciale.

3.1.2 Retour à IOCS16

- s’effectue par une instruction RSR
- l’état des registres peut être quelconque
- lorsqu’un module de fonction spéciale rend le contrôle à IOCS16, celui-ci transmet à l’utilisateur le contenu du registre A tel qu’il l’a reçu. On peut donc utiliser le registre A pour transmettre à l’utilisateur une information ou un compte-rendu.

- un module de fonction spéciale peut être assemblé isolément puis relié à IOCS16 par édition de liens à condition que l'IOCS16 ait été configuré pour le recevoir.
- le COMMON d'IOCS16 devra être décrit en tant que DSEC dans le module d'assemblage. La base C pointe cette DSEC (IOCS16 l'initialise avant de donner le contrôle à la fonction).
- le module pourra comporter une section LOCAL. Dans ce cas le programme devra lui-même charger sa base.

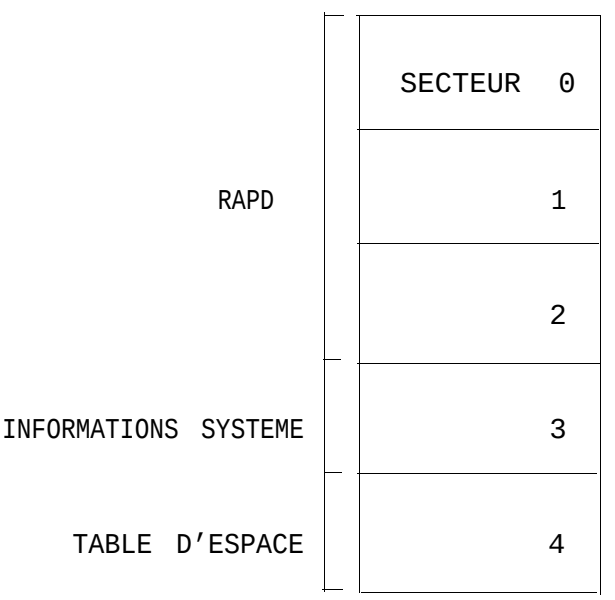
Règles à respecter : Le point d'entrée du module de fonction spéciale devra avoir pour nom symbolique Fxxxxx où xxxxx représente le code mnémonique associé à la fonction lors de la configuration.
Cette étiquette devra avoir été définie par une directive ENT.

4 LE LOGICIEL DE GESTION DU MONTAGE DE VOLUME

4.1 UTILISATION DES SUPPORTS DISQUE

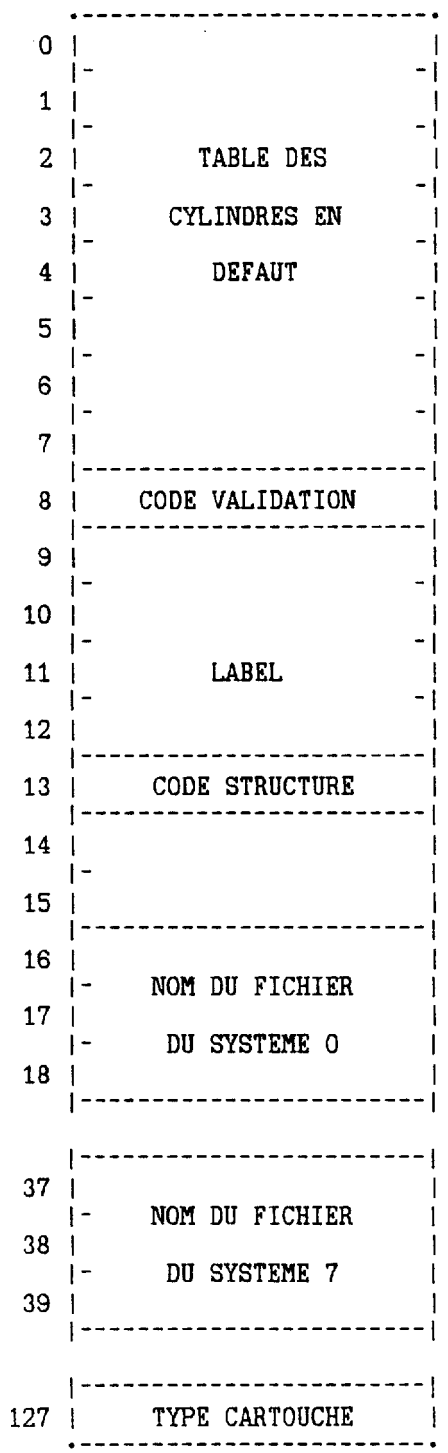
Avant toute utilisation, un support disque doit être FORMATE. Cette opération, indispensable, réalisée par un programme autonome a pour but d'écrire sur le support les informations nécessaires au fonctionnement et à la gestion du support. Ces informations élaborées par le programme de formatage sont les suivantes :

- les entêtes (HEADER) de secteurs nécessaires au fonctionnement du coupleur.
- la table des cylindres en défaut ou table de glissement indispensable au driver pour le calcul des adresses physiques.
- le programme de chargement et lancement (BOOTSTRAP) des systèmes disque (RAPD).
- un identificateur (label) du support.
- des tables systèmes renfermant différents indicateurs nécessaires à la prise en charge du support par le système : TABLE D'ESPACES.



Ces différentes informations sont situées dans les secteurs 0 à 4 du premier cylindre valide du support.

4.1.1 Structure du secteur 3



- TABLE DES CYLINDRES EN DEFAUT : contient les numéros de cylindre en défaut ou '7FFF s'il n'y en a pas.
- CODE VALIDATION : '89AB
- LABEL : identificateur du support (1 à 7 caractères ASCII, suivi du caractère '8D).
- CODE STRUCTURE :

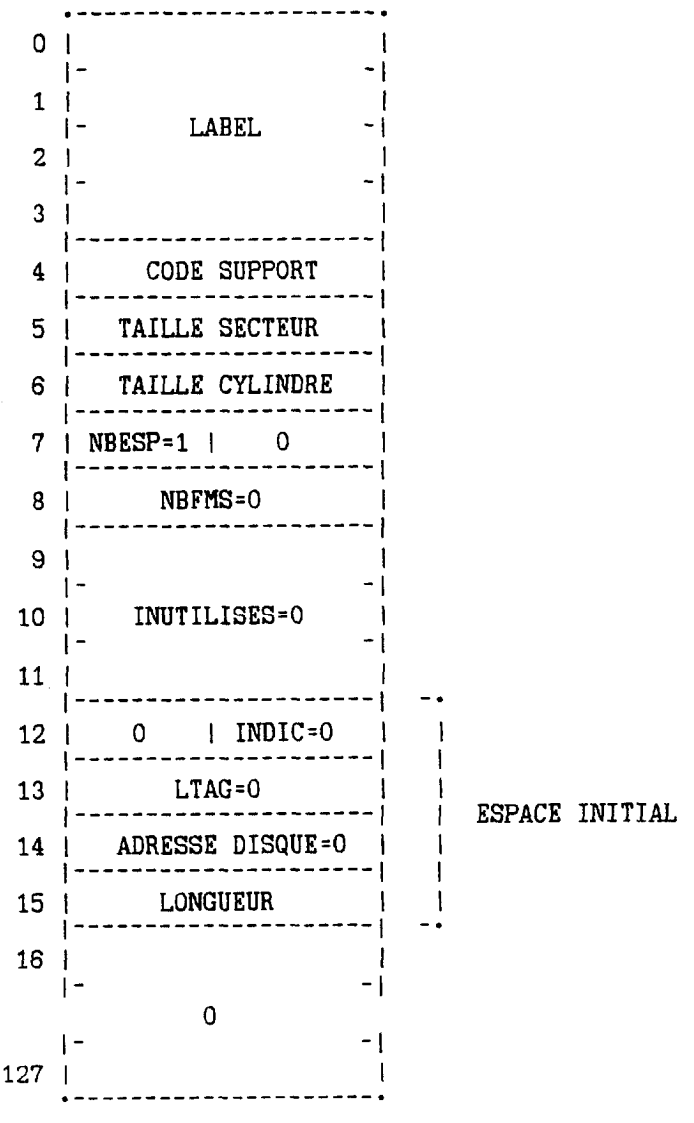
  0 : volume non structuré (secteur 4 sans signification)
  1 : volume structuré, le support est un disque CDB10
  2 : volume structuré, le support est un disque DPB50
  4 : volume structuré fixe le support est un disque CDB20
  5 : volume structuré mobile " " " "
  6 : volume structuré, le support est un disque souple

7 : volume structuré, le support est un disque à interface SMD
8 : volume structuré, le support est un disque WINCHESTER.

- La zone contenant les noms des fichiers systèmes n'est renseignée que pour les supports systèmes.
- TYPE CARTOUCHE :

0 : cartouche banale
n : cartouche de régénération. N = N* du système REGEN.

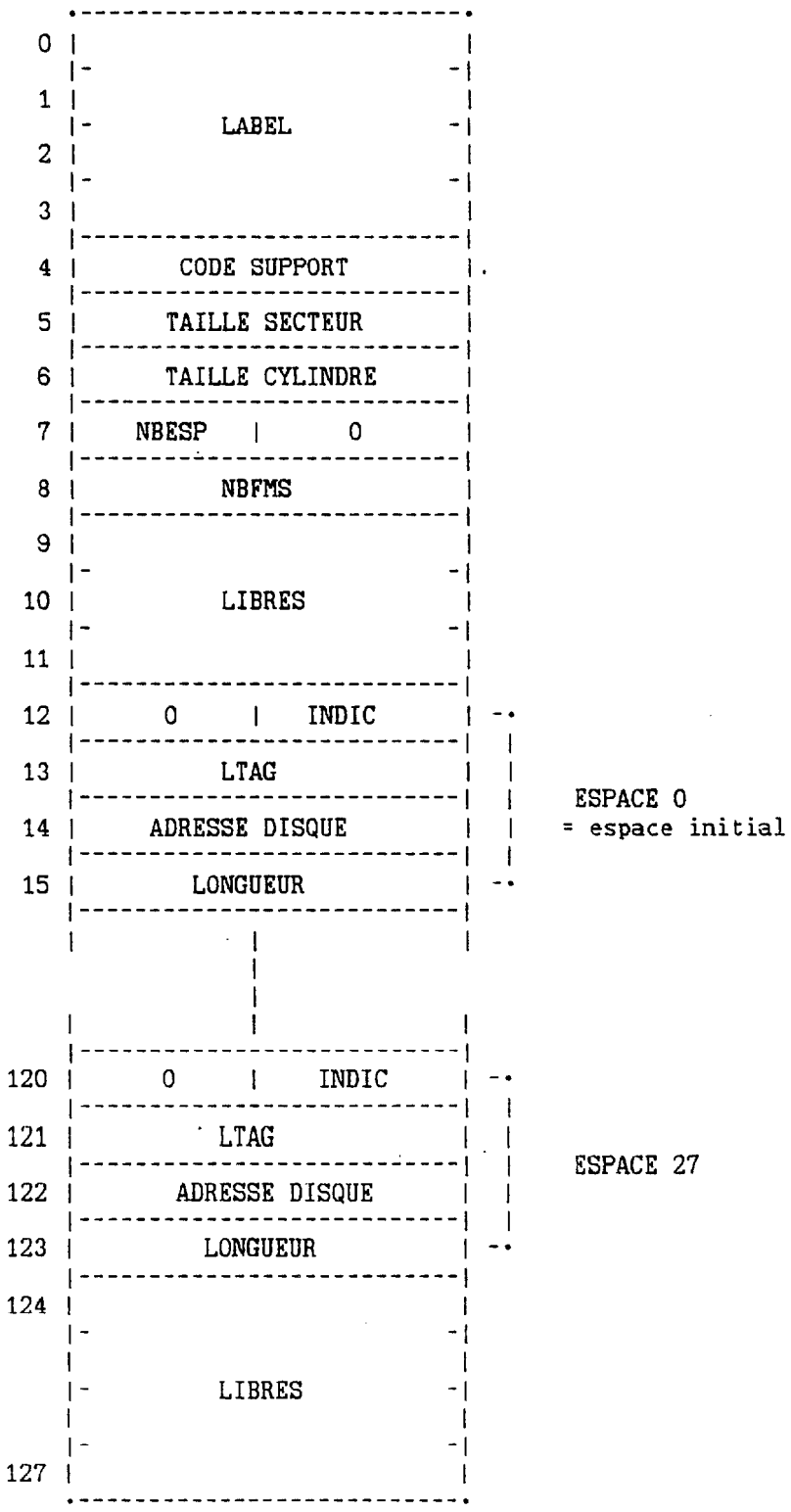
4.1.2 La table d'espaces après formatage (secteur 4)



LABEL	: Recopie des informations contenues dans le secteur 3.
CODE SUPPORT	: Recopie des informations contenues dans le secteur 3.
TAILLE SECTEUR	: Taille du secteur en mots (l28).
TAILLE CYLINDRE	: Taille du cylindre en secteur (fonction du support).
NBESP	: 1
NBFMS	: 0
INDIC	: 0 [Espace accessible en lecture et écriture)
LTAG	: 0 (Espace non géré par FMS16)
ADRESSE DISQUE	: 0
LONGUEUR	: Nombre de cylindres formatés.

4.1.3 La table d'espaces structurée par FUP4 (SDEF).

Après formatage du support, l'utilisateur dispose d'un disque susceptible d'être "monté" sur une unité physique.
Préalablement à tout accès, il doit compléter la structure définie par le programme de formatage au moyen de l'utilitaire FUP4 (voir notice FUP) et définir l'environnement du système de fichiers FMS16 à l'aide du même utilitaire. Le secteur 4, lorsque cette opération a été réalisée, se présente ainsi :



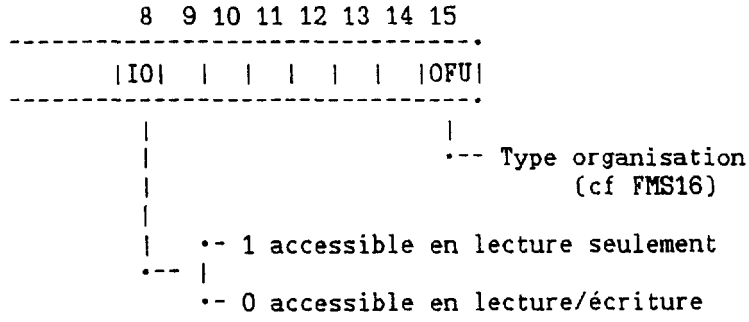
Signification des différentes valeurs :

a) Paramètres communs à l'ensemble des espaces

LABEL : recopie des informations contenues dans le secteur 3.
CODE SUPPORT : " " " " " " " "
TAILLE SECTEUR : Taille du secteur en mot (128).
TAILLE CYLINDRE : Taille du cylindre en secteur (fonction du support).
NBESP : Nombre d'espaces valides décrits sur le support.
NBFMS : Nombre d'espaces disque gérés par FMS16 (cf. Notice FMS16).

b) Paramètres relatifs à un espace

INDIC :



LTAG : Longueur de la Taille d'Allocation des Granules.
si LTAG = 0 l'espace n'est pas géré par FMS16
ADRESSE DISQUE : Adresse physique, exprimée en cylindres, de l'espace.
LONGUEUR : Taille de l'espace disque exprimée en cylindre.

4.2 UTILISATION DU LOGICIEL

4.2.1 Description interne

Le logiciel de gestion du montage des volumes se présente sous la forme de trois fonctions spéciales moniteur (cf. Manuel de référence) :

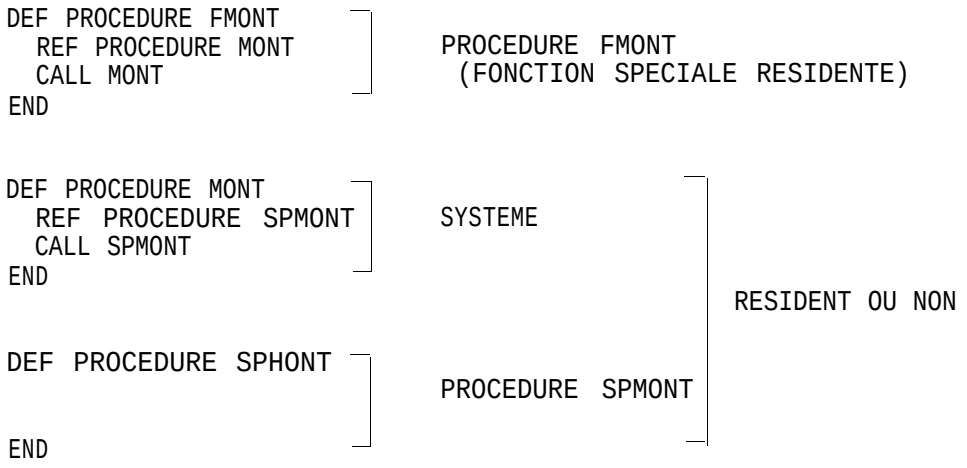
- FREAD : cette fonction spéciale a pour but de communiquer au système et aux processeurs la structure du support monté sur une unité physique. Cette requête appelée par les systèmes et par certains processeurs est RESIDENTE.

- FMONT et FDMONT : ces fonctions spéciales accessibles à travers les dialogues opérateur du système sont activées uniquement lors des "montages" et "démontages" des volumes. Il est donc possible de les rendre NON RESIDENTES et de les inclure dans la structure d'OVERLAY du système. Pour cette raison chaque fonction spéciale FMONT et FDMONT se compose de 2 modules :

    * un module résident constituant la fonction spéciale moniteur dont le seul rôle est de réaliser un appel vers un sous-programme qui appellera lui-même le sous-programme effectuant le traitement.

    * le module ou sous-programme de traitement proprement dit.

Exemple : Fonction spéciale FMONT



2.2 La bibliothèque BVOL16

COMPOSITION :

La bibliothèque BVOL16 support du logiciel comporte 5 articles. C'est un fichier indexé catalogué sous le nom de BVOL16-:S ; les articles constituant ce fichier sont :

- FREAD : fonction spéciale moniteur de communication de structure disque.
- FMONT : fonction spéciale moniteur de demande de montage de volumes.
- SPMONT : module complément du précédent.
- FDMONT : fonction spéciale moniteur de demande de démontage de volumes.
- SPDMON : module complément du précédent.

UTILISATION

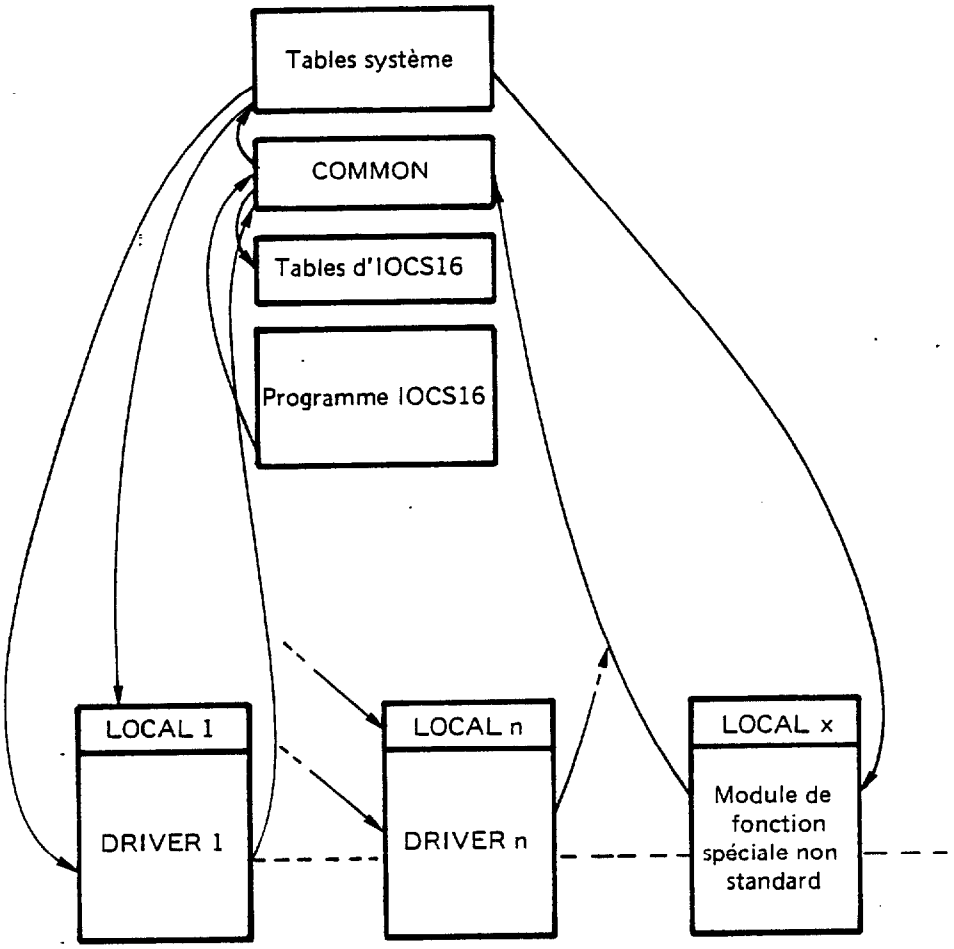
La scrutation de la bibliothèque à l'édition de liens du système doit intervenir :

- à l'édition de liens de la racine pour les fonctions spéciales moniteur FREAD, FMONT, FDMONT.
- à l'édition de liens de la racine pour les sous-programmes SPMONT et SPDMON dans le cas où ces fonctions doivent être résidentes.
- à l'édition de liens des branches gérant le montage et démontage de volumes dans le cas où ces fonctions doivent être à structure d'overlay.

4.3 GENERATION

Le logiciel du montage de volume est automatiquement intégré lors de la génération par l'emploi des macro-instructions "système standard" (%BOS16, %RTES16, etc...) et de celles relatives aux volumes disque (%FUI.., %FUESP..).

5 INTRODUCTION STRUCTURE D'IOCS16



5.1 LE NOYAU D'IOCS16

Certains éléments sont indépendants de la configuration du système et sont nécessaires au fonctionnement d'un IOCS16 destiné à une configuration minimale. Ils constituent le "noyau" d'IOCS16 qui comprend :

- le COMMON
- les tables
- le programme

Le noyau est indépendant de la configuration du système et n'a donc pas besoin d'être "généré". Il est fourni sous une forme symbolique directement assemblable par ASM.

5.2 ELEMENTS DEPENDANTS DE LA CONFIGURATION DU SYSTEME

Ce sont :

- les tables système
- les drivers
- les modules de fonctions spéciales non standard.

5.2.1 Les tables système

Elles sont décrites dans le § 1.3 du présent manuel.

Elles renferment toutes les informations caractérisant d'une part le système d'entrées-sorties (unités physiques, unités fonctionnelles, unités symboliques), d'autre part les fonctions d'IOCS16.

Ces tables sont évidemment particulières à une application donnée et nécessitent donc d'être générées. C'est le but de GIO16.

5.2.2 Les drivers

Chaque driver est structuré comme un segment utilisant d'une part le COMMON d'IOCS16, d'autre part un local qui lui est propre.

Chaque driver constitue un module assemblé isolément et link-éditable avec IOCS16.

Les drivers des périphériques conventionnels sont fournis sous une forme binaire link-éditable.
On reliera à IOCS16 uniquement les drivers nécessaires à l'installation.

Dans le cas où l'utilisateur veut introduire un périphérique de type nouveau, il pourra écrire un driver approprié à ce périphérique. Ce driver pourra également être relié à IOCS16 par édition de liens comme cela est fait pour les périphériques conventionnels, à condition que ce nouveau type de périphérique ait été défini lors de la configuration.

5.2.3 Les modules de fonctions spéciales

Certaines fonctions spéciales dites standard sont incluses dans le noyau d'IOCS16. Si l'utilisateur veut en introduire d'autres, il devra écrire le module correspondant (voir paragraphe 3).

Ce module pourra être relié à IOCS16 par édition de liens comme cela est fait pour les drivers.

6 GENERATION

6.1 PRESENTATION DE GIO16

GIO16 se présente sous la forme d'un fichier symbolique comportant :

- une séquence d'initialisation
- une bibliothèque de macro-définitions

Pour décrire sa configuration, l'utilisateur devra écrire un jeu de macro-instructions. Le contenu de la bibliothèque sera décrit ultérieurement.

6.2 MISE EN OEUVRE

La génération d'IOCS16 est indissociable de la génération des systèmes (BOS16, RTES16, MPESl6, etc..) ; elle est à ce titre, transparente à l'utilisateur dont la seule intervention se résume à la constitution du "fichier des macro-instructions utilisateur" de GIO16.

6.3 LES MESSAGES D'ERREUR

Les messages d'erreur peuvent être de deux sortes :

a) Messages de la forme :

```
ERM n
<Macro-instruction erronée>
```

Pour connaître les différentes causes d'erreurs, se reporter au manuel de référence de MACP.

On rappelle que les erreurs dans l'écriture d'une macro-instruction ne peuvent être détectées que si elle est précédée par le caractère %. Pour des raisons de sécurité, il est donc conseillé de faire précéder toute macro-instruction par le caractère % (en colonne 1).

b) Messages émis par le générateur :
Lorsque celui-ci trouve une erreur dans l'écriture d'une macro-instruction, il ignore la macro-instruction erronée et imprime un message d'erreur.

Les différents messages sont :

- MACRO-INSTRUCTION INTERDITE :
La macro-instruction écrite précédemment n'est pas autorisée dans l'étape actuelle : elle ne respecte pas l'ordre demandé

- PARAMETRE ? INCORRECT :
Le paramètre spécifié par ? n'est pas correct. Pour plus de précisions se reporter à la description de la macro-instruction en cause

- INCOMPATIBILITE ENTRE ? ET ? :
Les deux paramètres ne sont pas compatibles.

- PARAMETRES ? , ? IDENTIQUES
Les deux paramètres doivent être différents.

Des détails concernant les différentes causes d'erreurs sont donnés dans la description de chaque macro-instruction.

6.4 LA CORRECTION DES ERREURS

Les erreurs ne peuvent être corrigées en cours de génération. L'utilisateur devra donc corriger le fichier des macros utilisateur et relancer à nouveau la génération.

7 DESCRIPTION DE LA BIBLIOTHEQUE DE MACRO-INSTRUCTIONS

NOTATIONS

XXX	Chaîne de caractères de type "symbole" dont le nombre de caractères est limité au nombre de X
999	Chaîne de caractères de type "décimal" dont le nombre de caractères est limité au nombre de 9.
FFF	Chaîne de caractères de type "hexadécimal" dont le nombre de caractères est limité au nombre de F.
|	Indique un choix entre plusieurs possibilités (exemple : Y|N indique la possibilité de choix entre Y et N).
[---]	Indique un paramètre qui peut être omis.

7.2 STRUCTURE D'UN JEU DE MACRO-INSTRUCTIONS

Lors de l'écriture d'un jeu de macro-instructions, l'utilisateur doit respecter un certain ordre.
Toute macro-instruction qui ne respectera pas cet ordre sera ignorée et le message "MACRO-INSTRUCTION INTERDITE" sera imprimé.

On distingue deux phases :

7.2.1 Dimensionnement et définitions

Cette phase permet de fixer les paramètres du système et de définir les périphériques non standard présents dans la configuration. Les macro-instructions autorisées sont :

%UC 16-
%IOP RAPIDE
%NTACHES
%SUMAX
%INPMMAX
%POOL
%FSMON
%DEFPU
%SYMBEXT

Pour décrire les systèmes "standard" on peut utiliser les macro-instructions :

%BOS16
%RTESl6
%MPES16
%TSM16
%MUTEX16

Ce type de macro-instructions génère les macro-instructions %NTACHES, %SUMAX. %POOL, appropriées pour chaque système.

Cette phase se termine à la rencontre de la première macho-instruction %NIVEAU.

7.2.2 Configuration

Dans cette phase, l'utilisateur décrit toutes les unités physiques de son système, toutes les unités fonctionnelles qu'il utilise, et les liaisons unités physiques - unités fonctionnelles.
C'est dans cette phase que seront générées les tables d'unités physiques.

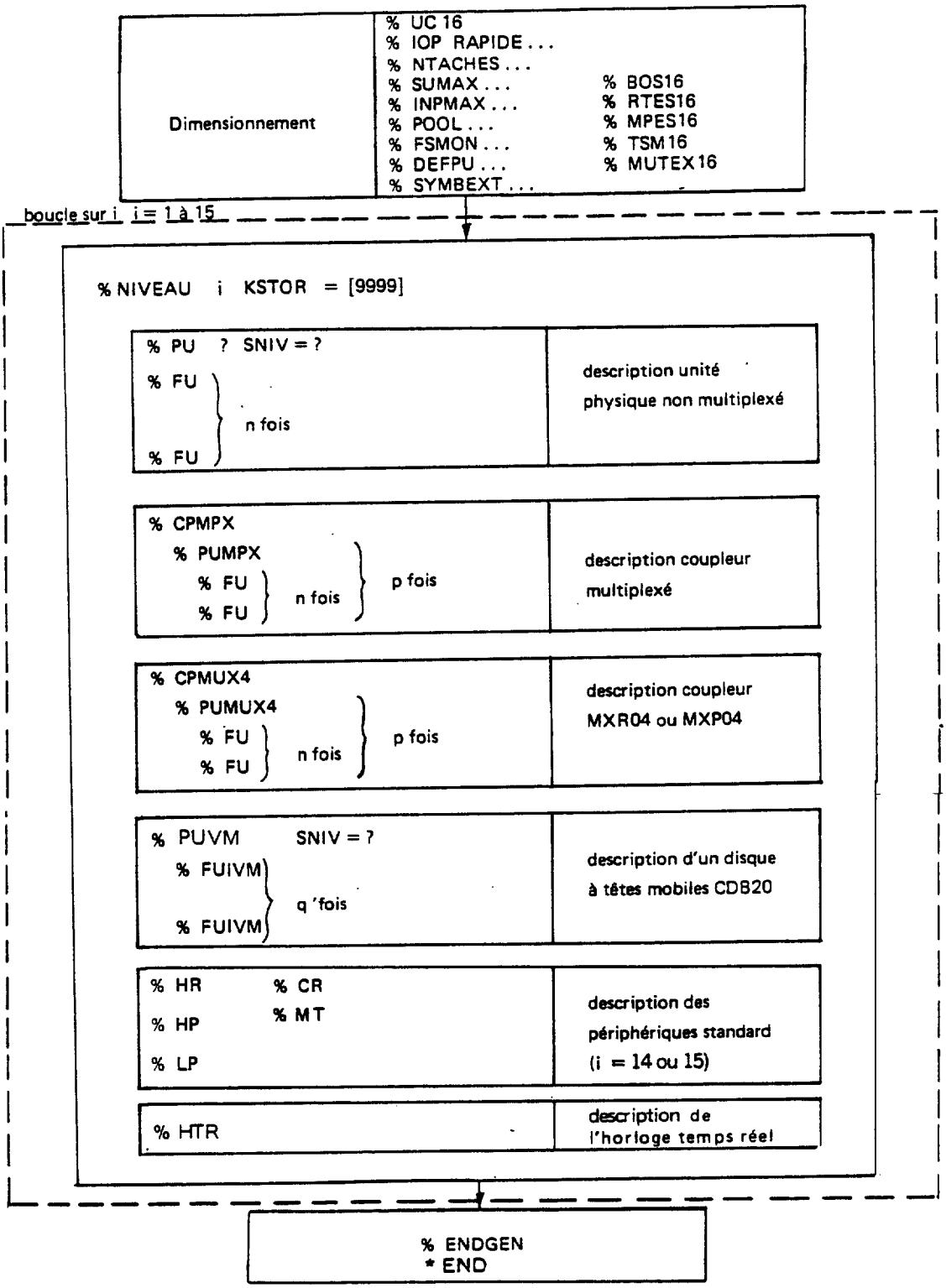
Cette description se fait par niveau : chaque niveau est déclaré par la macro-instruction %NIVEAU 999 KSTOR=[999]
A chaque niveau peuvent correspondre plusieurs sous-niveaux. Chaque unité physique ou chaque coupleur multiplexé est rattaché à un sous-niveau ; pour chaque niveau on décrira donc tous les sous-niveaux occupés.

Après chaque description d'unité physique (%PU) on décrira les unités fonctionnelles rattachées à cette unité.

Après chaque description de coupleur multiplexé occupant le niveau i, on décrira le(s) périphérique(s) connecté(s) à ce coupleur par des macro-instructions %PUMPX, et pour chaque périphérique connecté les unités fonctionnelles qui lui sont associées (par %FU).

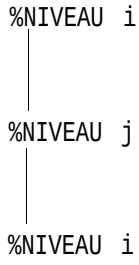
L'utilisation des gestionnaires d'écran SIM ou MCS (MUTEX16) impose le choix des drivers DRVMC1 (pour SIM) ou DRVMC2 (pour MCS) et des macros associées (%CPMUX4, %PUMUX4).
Dans tous les autres cas, on utilisera le driver DRVASY. Les macros relatives à ce driver sont décrites dans le M.R. DRVASY.

Pour décrire les périphériques dits "standard", on peut utiliser les macro-instructions disponibles dans la bibliothèque : %CR, %LP, %MT etc...
Pour décrire les périphériques reliés à un coupleur asynchrone multiplexé, on dispose de macro-instructions particulières %CPMUX4 et %PUMUX4.
Pour la description des macro-instructions, se reporter aux pages suivantes.
Cette phase se termine par la macro-instruction %ENDGEN.



NB : i peut varier de 1 à 15
Chaque rectangle correspond à un ou plusieurs sous-niveaux. On peut avoir plusieurs rectangles identiques à l'intérieur d'un même niveau. Chaque rectangle peut être omis. Il est inutile de définir un niveau sur lequel aucun périphérique n'est connecté.

Remarques :
La description d'un niveau doit être complètement terminée avant la prochaine macro-instruction %NIVEAU.
Ex. : cette configuration n'est pas autorisée



On peut décrire les niveaux dans n'importe quel ordre.

7.3.1 Phase dimensionnement

```
%IOP RAPIDE=?,?,?
```

Cette macro-instruction a pour but, lorsque la configuration comporte 1 IOP ou plus, d'en préciser le type.

Syntaxe :

%IOP RAPIDE=9[,9][,9]

le paramètre représente le numéro d'IOP (l, 2 ou 3).

Erreurs détectées :

L'erreur "PARAMETRE INCORRECT" est détectée si :

- les paramètres ne sont pas de type décimal
- les paramètres sont différents de l, 2 ou 3.

L'erreur "PARAMETRES IDENTIQUES" est détectée si l'on a spécifié deux fois le même numéro d'IOP.

Option prise par défaut :

Tous les IOP seront considérés comme étant des IOP16-M.

```
%UC 16-?
```

Cette macro-instruction permet de préciser le type d'unité centrale.

Syntaxe :

%UC 16-35|70|90 [PAS DE COMPTAGE]

Erreurs détectées :
L'erreur "PARAMETRE INCORRECT" sera détectée si le paramètre est différent des trois types d'UC (35, 70, 90).

Option par défaut : UC 16-35

Remarque : Un des effets de cette macro-instruction est de générer des zones de sauvegarde des contextes canaux LDC pour l'UC 16-35 (8 mots par sous-niveau et par IOP).

Le paramètre "PAS DE COMPTAGE" a pour effet de ne pas générer la table des "compteurs d'échange" dont la taille est égale à 2 (mots) x Fumax (n° de FU maximum déclarée hormis la FU '7E)

%NTACHES=?

Cette macro-instruction permet d'indiquer le nombre maximum de tâches susceptibles de travailler avec IOCS16.

Syntaxe :

%NTACHES=999

Erreurs détectées : L'erreur "PARAMETRE INCORRECT" est détectée si :

- le paramètre n'est pas de type décimal
- le paramètre est supérieur à 128.

Option prise par défaut :
Cette macro-instruction peut être omise, l'option prise par défaut sera alors :

%NTACHES=48

%SUMAX=?

Cette macro-instruction permet à l'utilisateur d'indiquer le nombre maximum d'unités symboliques de son système.

Syntaxe :

```
%SUMAX=999
```

Erreurs détectées :

L'erreur "paramètre incorrect" est détectée si :

- le paramètre n'est pas de type décimal
- le paramètre est supérieur à 128.

Option prise par défaut :
Si cette macro-instruction est omise, on prendra par défaut :

```
%SUMAX=16
```

%INPMAX=?

Cette macro-instruction permet à l'utilisateur de fixer le nombre maximum de caractères qui seront échangés lors d'un échange avec code d'arrêt en cas de non concordance avec les codes de la table des codes d'arrêt.

Syntaxe :

```
%INPMAX=9999
```

Erreurs détectées :
L'erreur "paramètre incorrect" est détectée dans l'un des cas suivants :

- le paramètre n'est pas de type décimal
- le paramètre est supérieur à 4096.

Options prises par défaut :
On prendra par défaut 80 caractères si cette macro-instruction est omise, soit la macro-instruction

```
%INPMAX=80
```

```
%POOL NBUF=? LBUF=?
```

Cette macro-instruction permet de décrire "le pool buffer" en fixant le nombre de buffers du pool d'une part (NBUF=?) et la longueur en mots de chaque buffer d'autre part (LBUF=?).

Remarque :
Le nombre de buffers est limité à 31.

Syntaxe :

```
%POOL NBUF=99 LBUF=9999
```

Erreurs détectées :
L'erreur "paramètre incorrect" est détectée dans l'un des cas suivants :

- Pour le paramètre NBUF :
  - . Le paramètre n'est pas de type décimal
  - . Le paramètre est supérieur à 31.

- Pour le paramètre LBUF :
  - . Le paramètre n'est pas de type décimal
  - . Le paramètre est supérieur à 4096.

Option prise par défaut :
Si cette macro-instruction est omise, on prendra par défaut un nombre de buffers égal à 1 et une longueur de 80 mots, c'est-à-dire :

```
%POOL NBUF=1 LBUF=80
```

```
%FSMON ? NUM=?
```

Cette macro-instruction permet de définir une fonction spéciale moniteur non standard.

Signification des paramètres :

a) Code mnémonique de la fonction spéciale :
Le premier paramètre est le code de la fonction spéciale. C'est une chaîne de type symbole de 5 caractères au maximum.
Cette chaîne doit être unique car, si XXXXX est le code mnémonique d'une fonction spéciale, GIO16 associe à cette fonction un module dont le point d'entrée a pour nom symbolique FXXXXX dans la table TBFSM.
Ce module devra être assemblé isolément puis relié par édition de liens à l'IOCS16 généré.
Les codes affectés aux fonctions spéciales standard sont interdits, ils sont connus implicitement par GIO16.

b) Numéro de la fonction (NUM=?)
C'est un nombre décimal qui indique le numéro attribué à la fonction spéciale définie. Il doit être différent des numéros attribués aux fonctions spéciales standard, c'est-à-dire supérieur ou égal à 8.

Remarque :
Si l'utilisateur donne pour nom à une fonction spéciale moniteur un nom déjà attribué à une fonction spéciale standard, aucune erreur ne sera détectée par GIO16, mais il y aura une erreur lors de l'assemblage (double définition d'étiquettes). Il lui faudra donc recommencer la génération.

Fonctions spéciales moniteur standard :
Les fonctions standard sont incluses dans l'IOCS16 généré sans qu'il soit besoin de les définir par une macro-instruction FSMON. On trouvera dans le manuel de référence IOCS16 la description de ces fonctions.

Les fonctions standard sont :

```
FCLEAR      num  = 0
FPUSI       num  = 1
FPUA        num  = 2
FPUD        num  = 3
FKILL       num  = 6
FCLSEL      num  = 7
```

Les fonctions "gestion de volume" sont :

```
DMONT       num  = 8
MONT        num  = 9
READ        num  = 10
```

Syntaxe :

```
%FSMON XXXXX NUM=999
```

Erreurs détectées :
L'erreur "paramètre incorrect" est détectée dans l'un des cas suivants :

- Pour le premier paramètre (nom de FSMON) :
  - . Ce paramètre n'est pas de type symbole
  - . Ce paramètre a plus de 5 caractères
- Pour le paramètre NUM :
  - . Ce paramètre n'est pas de type décimal
  - . Il est supérieur à 128
  - . Il est inférieur à 8.

```
%DEFPU ? CODE=? LZONE=? SP=?
```

Cette macro-instruction permet de définir un type de périphérique non standard.

Signification des paramètres :

a) Type de périphériques
On sait que les fonctions particulières à un périphérique (échanges, fonctions de positionnement) sont réalisées par le driver correspondant à ce périphérique.
Plusieurs périphériques peuvent utiliser le même driver à condition d'être de même type.
On associe donc à chaque type de périphérique un driver chargé d'exécuter les fonctions qui lui sont propres.
Le type de périphérique est représenté par une chaîne de type symbole de trois caractères au maximum.
Lors de la génération, GIO16 associe automatiquement à un périphérique de type XXX un driver tel que :

- son point d'entrée ait pour nom symbolique DRVXXX
- l'adresse sur laquelle pointe la base L dans son LOCAL ait pour nom symbolique LOCXXX.

Lorsqu'un utilisateur veut introduire dans son installation un périphérique de type particulier, il doit écrire le driver correspondant et le définir par une macro-instruction DRFPU afin que GIO16 puisse le reconnaître dans la suite de la génération.

b) Longueur des codes transférés (CODE=?)
Ce paramètre précise si le type de périphérique transfère des codes de 8 bits (lecteur de ruban par exemple) ou de plus de 8 bits (lecteur de cartes par exemple).

On notera :
CODE=8 pour les périphériques à code 8 bits
CODE=16 pour les périphériques à code supérieur à 8 bits.

c) Longueur de la zone de travail (LZONE=?)
On sait qu'un driver trouve les informations nécessaires à un échange donné dans la table unité physique du périphérique sur lequel est effectué l'échange.
La table unité physique est décrite dans ce manuel.
Il peut arriver qu'un driver ait besoin "d'agrandir" la table physique de façon à pouvoir disposer d'une zone de travail (cas des drivers disque). Le paramètre LZONE=? précise la longueur de cette zone de travail. (Elle est limitée à 128 mots).
Cette zone de travail est générée au-dessus de la table unité physique (TUP) standard.

d) Spécification "microprogrammé" (SP=?)
Certains périphériques peuvent être microprogrammés de manière spécifique. Ce paramètre vaut Y si le périphérique est microprogrammé et N sinon. Ce paramètre peut être omis, l'option prise par défaut est N.

Les périphériques gérés par les drivers contenus dans les bibliothèques de MOL (Module Objet Link-éditable) BDRV16 (pour les drivers standards) et BTRA16 (pour les drivers de transmission) sont reconnus implicitement par GI016

DRIVERS STANDARDS		PARAMETRES GENERES		
DRV--	périphérique/coupleur	CODE	LZONE	SP
CMF	coupleur multi-fonctions	8	231	N
CR	lecteur de cartes	8	0	N
DK	disque à têtes fixes	16	0	N
DP	disque DPB50	16	15	N
FDD	floppy-disque	16	52	N
HTR	horloge temps réel	16	0	N
LP	imprimante ligne	8	0	N
MT	bande magnétique	16	3	N
RP	lecteur/perfo ruban papier	8	0	N
SAS	disque WINCHESTER	16	68	N
SMD	disque à interface SMD	16	64	N
VM	disque à cartouche CDB10 ou CDB20	16	26	N

DRIVERS DE TRANSMISSION		PARAMETRES GENERES		
DRV--	périphérique/coupleur	CODE	LZONE	SP
BDG	terminaux lecteur de badges	8	36	N
DLC	coupleur de ligne HDLC (niveau 1)	8	10	N
EDC	coupleur synchrone avec option EDC	8	16	Y
MC1	MUX4 (SIM)	8	21	N
MC2	MUX4 (MCS)	8	21	N
x25	coupleur de ligne HDLC (niveaux 1 et 2)	8	80	Y

Remarque : Les paramètres décrits dans ces tableaux peuvent être redéfinis par la macro %DEFPU

Syntaxe :

```
%DEFPU XXX CODE=8|16 LZONE=999 SP=[Y|N]
```

Erreurs détectées :

L'erreur "paramètre incorrect" est détectée dans l'un des cas suivants :

- Pour le paramètre "type" :
  - . Le paramètre n'est pas de type symbole
  - . Sa longueur est supérieure à 3.

- Pour le paramètre CODE :
  - . Il est différent de 8 et de 16

- Pour le paramètre LZONE :
  - . Il n'est pas de type décimal
  - . Il est supérieur à 128

- Pour le paramètre SP :
  - . Il est autre que Y ou N

Remarque :

Si le type de périphérique est l'un de ceux connus implicitement par GIO16, celui-ci ne détectera pas d'erreur lors de la génération, mais il y aura une erreur d'assemblage (double définition d'étiquettes).

%SYMBEXT ?

Cette macro permet d'indiquer le nom d'un module externe utilisé dans une ou plusieurs macro-instructions (en particulier %TUP...). Ce module devra être relié par édition de liens avec le noyau d'IOCS16.

Syntaxe :

```
%SYMBEXT XXXXXX
```

Exemple :

```
%SYMBEXT TRANSC
```

3.2 Les macros “standard” système

Ces macro-instructions servent à définir des systèmes conventionnels dits "standard". L’utilisateur peut simplifier la phase d’initialisation et de dimensionnement en utilisant les macro-instructions suivantes :

%BOS16

%RTES16

%MPES16

%TSM16

%MUTEX16

Ces macro-instructions regroupent les macro-instructions %NTACHES, %SUMAX, %POOL selon le tableau ci-dessous :

macros \ macro systèmes	TSM16	%RTES16	%MUTEX16	%BOS16	%MPES16
%NTACHES=	128	128	128	48	128
%SUMAX=	0	30	30	29	1
%POOL NBUF=	0	*	*	*	*
%POOL LBUF=	0				

* : non générée par la macro "système standard"

7.3.3 Phase configuration

```
.-------------------------.
|    %NIVEAU ? KSTOR=?    |
.-------------------------.
```

Cette macro-instruction indique le début de la description d'un niveau d'interruption : toutes les macro-instructions qui suivent jusqu'à la prochaine macro-instruction %NIVEAU concernent les périphériques reliés aux différents sous-niveaux de ce niveau.

Signification des paramètres :

- le paramètre NIVEAU précise le numéro de niveau. C'est un nombre compris entre 1 et 15.
- le paramètre taille (KSTOR=?) donne la taille en mots de la pile K de la tâche gérant le niveau. Ce paramètre doit être inférieur à 512.

Si le paramètre KSTOR est omis, on prendra par défaut la taille standard suffisante pour l'utilisation des drivers standard (30 mots).

Syntaxe :

```
%NIVEAU 99 KSTOR=[9999]
```

Erreurs détectées :

L'erreur "paramètre incorrect" sera détectée si :
- l'un des deux paramètres n'est pas de type décimal
- le paramètre "niveau" est supérieur à 15
- le paramètre "KSTOR" est supérieur à 512.

```
%PU ? SNIV=? MODE=? ADR=? ITN=? CONNEX=? IOP=?
```

Cette macro-instruction définit une unité physique non multiplexée connectée sur le niveau précédemment décrit.
Elle est obligatoirement suivie par une ou plusieurs macro-instructions %FU (description des unités fonctionnelles associées à cette unité physique).

Signification des paramètres :

- Paramètre "nom de pu" :
Le premier paramètre décrit le type de périphérique. Si celui-ci n'est pas standard, il doit obligatoirement être décrit auparavant par une macro-instruction DEFPU.

- Paramètre SNIV :
Ce paramètre indique le sous-niveau sous lequel est connecté l'unité physique. Il est fonction du mode de fonctionnement. Il doit être omis si le mode est PS, inférieur à 16 si le mode est PP, et inférieur à 48 dans les autres cas.

- Paramètre MODE :
Ce paramètre indique le mode de fonctionnement du périphérique. Il vaut :

. PS	si programmé simple
. PP	si programmé prioritaire
. LDC	(Low speed Data Channel)
. MDC	(Médium speed Data Channel)
. HDC	(High speed Data Channel)

- Paramètre ADR :
Il s'agit de l'adresse du coupleur du périphérique. C'est un nombre hexadécimal.

- Paramètre ITN :
Il s'agit du niveau des interruptions canal. Ce paramètre doit être omis si le mode est PP ou PS, il est obligatoire dans les autres modes :

. HDC, ITN aura une valeur comprise entre 0 et 7.
. HDC, ITN sera inférieur à 16.
. LDC, ITN sera inférieur à 64.

- Paramètre CONNEX :
Il précise le mode de connexion (c'est-à-dire le type de coupleur sur lequel est connecté le périphérique). Il peut être :

. STD Standard
. ASYV1 Asynchrone une voie

Il peut être omis, l'option prise par défaut est STD.
Les périphériques reliés par un coupleur de type MXP04 font l'objet de macro-instructions spéciales.

- Paramètre IOP :
Ce paramètre précise le numéro du processeur qui gérera les entrées-sorties.
Ce paramètre est interdit si le mode est PP.
Ce paramètre vaut 0, 1, 2, 3. S'il est omis, le processeur de calcul (CPU n. 0) se chargera de gérer les entrées-sorties.

Syntaxe :

```
%PU XXX SNIV=99 MODE=HDC|MDC|LDC|PP|PS ADR='FFFF ITN=[9]
                CONNEX=[STD|ASYV1] IOP=[9]
```

Erreurs détectées :

L'erreur "paramètre incorrect" est détectée si :

- Pour le paramètre "nom pu" :
  . Il n'est pas de type symbole
  . Il a plus de 3 caractères.

- Pour le paramètre SNIV :
  . Si le mode est PP, il est supérieur à 16
  . Si le mode est différent de PP il est supérieur à 48.

- Pour le paramètre MODE :
  . Il est différent de PP, HDC, MDC, LDC ou PS.

- Pour le paramètre ADR :
  . Ce paramètre n'est pas de type hexadécimal.

- Pour le paramètre ITN :
  . Il est présent à tort car le mode est PP ou PS
  . Il est supérieur à 7 si le mode est HDC
  . Il est supérieur à 15 si le mode est MDC
  . Il est supérieur à 63 si le mode est LDC.

- Pour le paramètre CONNEX :
  . Il est différent de STD et ASYV1

- Pour le paramètre IOP :
  . Il est supérieur à 3
  . Il est présent si le mode est PP ou PS.

Remarques :

- l'utilisation d'un type de périphérique non défini ne donnera pas lieu à une détection d'erreur lors de la génération et ne pourra donc pas être corrigée à ce moment-la. Cette erreur sera détectée à l'assemblage (symboles non définis). Ce type d'erreur est fatal et nécessite de recommencer la génération.

%TUP-?=?

Cette macro-instruction permet d'initialiser l'extension de la dernière TUP générée, au moyen d'un nombre ou d'un symbole externe défini par %SYMBEXT. Elle peut suivre une macro %PU, %PUMUX4, %PUMPX, %PUDX.

Le premier paramètre donne le déplacement par rapport au début de la TUP.
Le deuxième donne la valeur ou le symbole à y charger. Attention, aucune vérification n'est faite par rapport au paramètre LZONE de la macro %DEFPU qui définit la longueur de l'extension de TUP.

```
%FU ? CDE=? SENS=? FINDIC=?
```

Cette macro-instruction définit une unité fonctionnelle (autre que disque) associée à la dernière unité physique (PU, PUMPX, PUMUX4) déclarée.

Signification des paramètres :

- Paramètre "FU" :
  . Il précise le numéro de FU et peut être exprimé en décimal (<l26) ou par sa clé symbolique (TS, LP, etc...).

- Paramètre CDE :
  . C'est le mot de commande. Il est de type hexadécimal.

- Paramètre SENS :
Ce paramètre indique le sens des échanges qui seront effectués sur cette unité fonctionnelle.
Il peut être égal à :
  . I si elle n'effectue que des entrées
  . O si elle n'effectue que des sorties
  . IO si elle effectue des entrées et des sorties.

- Paramètre FINDIC :
  . Ce paramètre vaut 0 ou 1, sa signification dépend du type de périphérique sur lequel pointe la FU.

Syntaxe :

```
%FU XX|999 CDE='FFFF SENS=I|O|IO FINDIC=0|1
```

Erreurs détectées :

- Pour le paramètre "FU" :
L'erreur "paramètre incorrect" sera détectée si :
  . Il est de type décimal supérieur à 125,
  . Le symbole n'est pas reconnu (voir MEMO16).

- Pour le mot de commande (CDE) :
L'erreur "paramètre incorrect" sera détectée s'il n'est pas de type hexadécimal.

- Pour le sens (SENS) :
L'erreur "paramètre incorrect" sera détectée si il est autre que I, O ou IO.

- Pour le paramètre FINDIC :
L'erreur "paramètre incorrect" sera détectée s'il est différent de 0 ou 1.

Remarque :
Deux unités fonctionnelles ne peuvent avoir le même numéro, l'erreur n'est pas détectée au moment de la génération mais au moment de l'assemblage. Cette erreur est fatale et nécessite de recommencer la génération.

```
%CPMPX ? SNIV=? MODE=? ADR=? ITN=? CONNEX=? IOP=? NBV=?
```

Cette macro-instruction permet de définir un coupleur multiplexé. Pour chaque coupleur multiplexé, l'utilisateur devra écrire une macro-instruction CPMPX. S'il n'est pas connu implicitement par GIO16 le coupleur doit avoir été décrit par une macro-instruction DEFPU.

Cette macro-instruction est obligatoirement suivie par une ou plusieurs macro-instructions PUMPX.

Signification des paramètres :

Les sept premiers paramètres ont la même signification que ceux de la macro-instruction %PU.

Paramètre NBV :
Ce paramètre précise le nombre de voies utilisées sur le coupleur que l'on définit. Ce nombre est inférieur ou égal à 16.

Syntaxe :

```
%CPMPX XXX SNIV=99 MODE=HDC|MDC|LDC|PP ADR='FFFF ITN=99
           CONNEX=[MUX8|MUX16] IOP=9 NBV=99
```

Erreurs détectées :

- pour les sept premiers paramètres, les erreurs sont les mêmes que dans la macro-instruction %PU
- pour le paramètre NBV, l'erreur "paramètre incorrect" sera détectée si :
  . Il n'est pas de type décimal
  . Il est supérieur à 16.

Remarques :

- Les coupleurs disque, floppy disque, ainsi que les coupleurs de type MXP04 ne peuvent être décrits par cette macro-instruction.
- l'utilisation d'un coupleur non défini ne donnera pas lieu à une détection d'erreur lors de la génération. L'erreur ne sera détectée qu'au moment de l'assemblage (symboles non définis), et nécessitera de recommencer la génération.

```
%PUMPX ? VOIE=?
```

Cette macro-instruction définit un périphérique connecté sur le coupleur multiplexé CPMPX décrit dans le paragraphe précédent.

Signification des paramètres :

- Paramètre "typu" :
Ce paramètre précise le type de périphérique. C'est une chaîne de 3 caractères alphanumériques au maximum.
S'il n'est pas l'un de ceux reconnus implicitement par GIO16, il doit obligatoirement avoir été l'objet d'une macro-instruction DEFPU.

- Paramètre VOIE :
Ce paramètre précise sur quelle voie s'effectue l'échange. Il doit être inférieur au nombre de voies déclaré dans la macro-instruction CPMPX.

Syntaxe :

```
%PUMPX XXX VOIE=99
```

Erreurs détectées :
L'erreur "paramètre incorrect" est détectée si :

- Pour le paramètre "typu" :
  . Il n'est pas de type symbole
  . Il a plus de trois caractères.

- Pour le paramètre VOIE :
  . Il n'est pas de type décimal
  . Il est supérieur au nombre de voies du coupleur.

L'erreur "macro-instruction interdite" est détectée si %PDMPX n'a pas été précédée de %CPMPX.

Remarque :
- l'initialisation d'un type de périphérique non défini ne donnera pas lieu à une erreur lors de la génération. L'erreur ne sera détectée qu'au moment de l'assemblage et nécessitera de recommencer la génération.

```
%PUDX ? BEGIN=? NUMBER=? STEP=? ADR=?
```

Cette macro décrit un coupleur démultiplexé, c'est-à-dire un coupleur connu comme une seule unité physique, mais utilisant plusieurs sous-niveaux d'interruption égaux à :

```
BEGIN
BEGIN + STEP          NUMBER fois
BEGIN + STEP + STEP
```

%PUDX doit être suivi d'une ou plusieurs macro %FU.

Signification des paramètres :

. Paramètre "BEGIN" :
C'est la valeur du sous-niveau de plus petit rang que l'utilisateur veut gérer.

. Paramètre "NUMBER" :
C'est le nombre de sous-niveau que l'utilisateur désire gérer.

. Paramètre "STEP" :
C'est la différence existant entre le sous-niveau de rang j et le sous-niveau de rang i. I est le rang du premier sous-niveau géré et j le rang du second.

```
%CPMUX4 MODE=? ADR=? IOP=? CONNEX=?
```

Cette macro-instruction permet de définir un coupleur du type MXP04. Elle est obligatoirement suivie de une ou plusieurs macro-instructions PUMUX4.

Signification des paramètres

- Paramètre MODE :
Ce paramètre indique le mode de fonctionnement des périphériques connectés sur le coupleur
  . PP si programmé prioritaire
  . LDC si Low Data Channel
  . MDC si Medium Data Channel
  . HDC si High Data Channel

- Paramètre ADR :
C'est un nombre hexadécimal qui précise l'adresse du coupleur.

- Paramètre IOP :
C'est un nombre décimal inférieur à 4 qui précise le numéro du processeur qui gère les entrées-sorties.
Ce paramètre peut être omis :

  - en mode PP
  - en mode canal ; il prend alors la valeur 0 (CPU).

- Paramètre CONNEX :
Il précise le mode de connexion (c'est-a-dire le type du coupleur) ; il peut être MUX4, ASYV1, STD, CMF, MUX4U. Par défaut CONNEX=MUX4.

Syntaxe :

```
%CPMUX4  MODE=PP|LDC|MDC|HDC  ADR='FFFF
          IOP=[9]  [CONNEX=[MUX4|ASYV1|STD|CMF|MUX4U]]
```

Erreurs détectées :
L'erreur "paramètre incorrect" sera détectée si :

  - le paramètre MODE est autre que PP, LDC, MDC ou HDC
  - le paramètre ADR n'est pas de type hexadécimal
  - le paramètre IOP est supérieur à 3
  - le paramètre CONNEX est différent de MUX4, ASYV1, STD, CMF, MUX4U.

Remarque : Un coupleur de type MXP04 est un coupleur dont le sous-niveau d'entrée et le sous-niveau de sortie d'une même voie sont différents.

```
%PUMUX4 ? SNIV1=? SNIV2=? ITN=? VOIE=?
```

Cette macro-instruction définit une unité physique connectée sur un coupleur de type MXP04. Elle doit être précédée de la macro CPMUX4.

Remarque préliminaire :

Si le périphérique peut fonctionner en entrée et sortie, les sous-niveaux des interruptions entrée et sortie sont différents.

- pour un fonctionnement en half duplex, on considérera le périphérique comme une seule unité physique : donc une seule macro PUMUX4 précisant le sous-niveau entrée et le sous-niveau sortie.
- pour un fonctionnement en full duplex, on considérera le périphérique comme 2 unités physiques : donc 2 macros PUMUX4, l'une en entrée, l'autre en sortie. La déclaration de la ou des FU associée(s) (%FU) doit être faite après la macro %PUMUX4. Deux macros %PUMUX4 doivent donc être séparées par une ou des macro(s) %FU ; seule la macro %FU peut être placée entre les macros %PUMUX4 associées à une même macro %CPMUX4.

Signification des paramètres

- Paramètre "nompu"

Ce paramètre précise le type de périphérique. S'il n'est pas connu implicitement, il doit avoir été défini auparavant par une macro-instruction DEFPU.

- Paramètre SNIV1

Ce paramètre est obligatoire. Il précise le sous-niveau (réception si fonctionnement en half-duplex) des interruptions du périphérique.
C'est un nombre décimal inférieur à 16 si le mode de fonctionnement (défini dans CPMUX4) est "programmé prioritaire", ou HDC et MDC, inférieur à 48 en mode LDC.

- Paramètre SNIV2

Le paramètre SNIV2 n'est utile que dans le cas d'un périphérique en entrée-sortie fonctionnant en half-duplex. Il indique alors le sous-niveau émission des interruptions du périphérique. IL doit être omis dans tous les autres cas. Il suit les mêmes règles que SNIV1 mais doit être supérieur de celui-ci et tel que SNIV2>SNIV1.

- Paramètre ITN

Ce paramètre est un nombre décimal qui précise le niveau des interruptions canal. Il est inférieur à 64 en LDC, 8 en HDC, 16 en MDC.
Il est obligatoire en mode LDC, MDC, HDC (paramètre donné dans la macro CPMUX4) et interdit en mode PP.

- Paramètre VOIE

C'est un nombre décimal inférieur à 4 qui précise sur quelle voie du coupleur est connecté le périphérique.

Syntaxe :

```
    %PUMUX4  XXX  SNIV1=99  SNIV2=[99]  ITN=[99]  VOIE=9
```

Erreurs détectées :

- "macro interdite" si elle n'est pas précédée de %CPMDX4
- "paramètre incorrect"
  - . Le type de périphérique n'est pas de type symbole, ou a plus de 3 caractères.
  - . SNIV1 ou SNIV2 est supérieur à 15 en mode PP, HDC et MDC ou supérieur à 47 en mode LDC
  - . ITN est supérieur à 63
  - . VOIE est supérieur à 3
- "paramètre interdit"
  - . ITN est présent alors que le MODE est PP.

Remarque :

- l'utilisation d'un type de périphérique non défini provoquera une erreur d'assemblage. Il faudra recommencer la génération.

```
%PUVM SNIV=? MODE=? ADR=? ITN=? IOP=? NBV=?
```

Notations :
On appelle unité d'échange multiplexée, l'ensemble des unités gérés par le même coupleur.

Cette macro-instruction permet de décrire une unité d'échange multiplexée disque CDB10 ou CDB20. Pour chaque unité d'échange multiplexée présente dans son installation, l'utilisateur devra décrire une macro-instruction %PUVM.

Signification des paramètres :

- Paramètre SNIV :
Ce paramètre indique le sous-niveau d'interruption sur lequel est connecté l'unité d'échange multiplexée.

- Paramètre MODE :
Ce paramètre précise le mode de connexion. Ce ne peut être que HDC ou MDC.

- Paramètre ADR :
Il indique l'adresse du coupleur. C'est un nombre hexadécimal.

- Paramètre ITN :
Ce paramètre est obligatoire. Il précise le numéro d'interruption canal.

- Paramètre IOP :
Ce paramètre précise le processeur d'entrée-sortie qui gère le canal. Il peut varier entre 0 et 3. S'il est omis, par défaut, le processeur de calcul servira à gérer les entrées-sorties.

- Paramètre NBV :
Ce paramètre précise le nombre de voies. Il est inférieur à 5.

Syntaxe :

```
%PUVM SNIV=99 MODE=MDC|HDC ADR='FFFF ITN=99 IOP=[9] NBV=99
```

Erreurs détectées :
Les erreurs détectées sur les paramètres SNIV, ADR, ITN, IOP sont les mêmes que dans la macro-instruction PU.

- Pour le paramètre MODE :
L'erreur "paramètre incorrect" est détectée si le paramètre est différent de HDC ou MDC.

- Pour le paramètre NBV :
L'erreur "paramètre incorrect" sera détectée si :
. Le paramètre n'est pas décimal
. Le paramètre est supérieur ou égal à 5.

```
%FUDK ?   ADRDK= ?   NSECT= ?
```

Cette macroinstruction sert à définir une unité fonctionnelle disque à têtes fixes. Elle doit obligatoirement être précédée d'une macroinstruction %PUDK ou d'une autre macroinstruction %FUDK.

Signification des paramètres :

- Paramètre "numfu"
Il représente le numéro de FU (décimal ou symbolique].

Paramètre "adresse" (ADRDK=?)
Il représente le numéro de secteur du disque où commence la FU disque. Il doit être inférieur à 8192 (256 pistes de 32 secteurs).

- Paramètre "nombre de secteurs" (NSECT=?)
Il précise le nombre de secteurs occupés par la FU disque. Il doit être inférieur à 8193.

- La somme adresse de début de FU + nombre de secteurs doit être inférieure à 8193.

Syntaxe :

```
%FUDK XX|999 ADRDK= 9999 NSECT= 9999
```

Erreurs détectées :

L'erreur "paramètre incorrect" est détectée si :

- le paramètre "NUHFU"" est supérieur à 125 ou, s'il est exprimé en symbolique, n'est pas reconnu,
- le numéro de secteur (ADRDK) est supérieur à 8191,
- le nombre de secteur est supérieur à 8192,
- la somme no de secteur + nombre de secteurs est supérieure à 8192.

Remarque : pas de gestion d'espace pour ce type de disque.

%FUIVM ? VOIE=? FIXE=?

Cette macro-instruction permet la description d'une FU initiale disque CDB10 ou CDB20 appartenant à l'un des disques de l'unité d'échange multiplexée définie précédemment par une macro-instruction PUVM.

Signification des paramètres :

- Paramètre "numfu" :
Ce paramètre indique le numéro de FU et peut être exprimé en décimal (< 126) ou en symbolique (D1,. E1, etc...).

- Paramètre VOIE :
Ce paramètre précise le numéro de voie sur laquelle est connectée la FU. Ce nombre peut prendre les valeurs de 0 à 3. Il doit être inférieur au nombre de voies défini dans la macro-instruction PUVM.

- Paramètre FIXE :
Ce paramètre vaut Y si la FU se trouve sur le plateau fixe et N si elle se trouve sur le plateau mobile.

Syntaxe :

```
%FUIVM XX|999 VOIE=9 FIXE=Y|N
```

Erreurs détectées :

L'erreur "macro-instruction interdite" est détectée si cette macro-instruction n'a pas été précédée d'une macro-instruction PUVM.

L'erreur "PLUS DE 28 FU POUR LE VOLUME" est détectée.

L'erreur "paramètre incorrect" est détectée si :

- Pour le paramètre "numfu"
  . Il est de type décimal et supérieur à 125,
  . Le symbole n'est pas reconnu (cf. MEMO16).

- Pour le paramètre "voie"
  . Il n'est pas de type décimal
  . Il est égal ou supérieur au nombre de voies défini dans PUVM.

Remarque : voir remarque de la macro %FU.

%FUESPVM?

Description d'une FU "espace" associée aux disques définis par la macro %FUIVM.
Le paramètre "numfu" obéit aux règles habituelles de ce paramètre.

```
%PUDP SNIV=? MODE=? ADR=? ITN=? IOP=? NBV=?
```

Cette macro décrit une unité d'échange multiplexée "disque DPB50" qui peut contenir 1 à 4 unités de 50 méga-octets.

Les paramètres, la syntaxe et les erreurs sont les mêmes que dans la macro %PUVM.

```
%FUIDP ? VOIE=?
```

Cette macro décrit une FU initiale "disque DPB50" appartenant à l'un des disques de l'unité d'échange multiplexée définie précédemment par une macro %PUDP.

Les paramètres ont la même signification que les 2 premiers paramètres de %FUIVM (idem pour la syntaxe et les erreurs).

Remarque : voir remarque de la macro %FU

```
FUESPDP ?
```

Description d'une FU "espace" associée au disque défini par la macro %FUIDP. Le paramètre "numfu" obéit aux règles habituelles de ce paramètre.

```
%CPSMD SNIV=? ADR=? ITN=? IOP=?
```

Cette macro décrit une unité d'échange multiplexée disque à interface SMD laquelle peut comprendre jusqu'à quatre unités.

Signification des paramètres :

SNIV	est un nombre décimal représentant le sous-niveau d'interruption sur lequel est connectée l'unité d'échange multiplexée.
ADR	est un nombre hexadécimal représentant l'adresse du coupleur (exemple : '38).
ITN	est un nombre décimal représentant le numéro d'interruption canal.
IOP	est un nombre décimal représentant le numéro du processeur d'entrée-sortie qui gère le canal. Il peut varier de 0 à 3, s'il est omis, c'est le processeur de calcul (0) qui est pris par défaut.

```
%FUISMD ? UNIT=? FIXE=?
```

Cette macro-instruction décrit une FU initiale d'un disque à interface SMD. Chacun des disques est identifié par son numéro d'unité. Ce dernier correspond au numéro affiché sur la face avant de l'unité.

Signification des paramètres :

- Paramètre 'numfu' : numéro de la FU initiale. Il peut être exprimé en décimal (< 126) ou en symbolique (D1, D2, E1, etc ...).

- Paramètre 'UNIT' : numéro d'unité décimal de 0 à 7.
Dans une configuration comportant plusieurs disques, les macros FUISMD doivent respecter l'ordre croissant des numéros d'unité.

- Paramètre 'FIXE' :
  = Y pour la partie fixe d'un disque
  = N pour la partie mobile.

Remarque : voir remarque de la macro %FU.

```
%FUESPSMD ?
```

Description d'une FU "espace" associée au disque défini par la macro %FUISMD. Le paramètre "numfu" obéit aux règles habituelles de ce paramètre.

```
%CPSAS SNIV=? MODE=? ADR=? ITN=? IOP=?
```

Cette macro décrit une unité d'échange multiplexée "disque WINCHESTER" pouvant comprendre 3 unités.

La signification et la syntaxe des paramètres est identique à celle décrite pour la macro %PUVM.

Les erreurs détectées sont les mêmes que pour la macro %PUVM.

```
%FUISAS ? UNIT=?
```

Cette macro-instruction permet la description d'une FU initiale 'disque WINCHESTER'.

Signification des paramètres :

numfu : (1er paramètre) indique le numéro d'unité fonctionnelle exprimé en décimal ou symbolique

UNIT : indique le numéro d'unité sur lequel est connectée la FU (0 à 2).

Remarque : voir remarque de la macro %FU.

```
%FUESPSAS ?
```

Description d'une FU "espace" associée au disque défini par la macro %FUISAS. Le paramètre "numfu" obéit aux règles habituelles de ce paramètre.

```
%PUFDD SNIV=? MODE=? ADR=? ITN=? IOP=?
```

Cette macro-instruction décrit une unité d'échange pouvant contenir 1 à 4 unités de disques souples.

Signification des paramètres :

SNIV	représente le sous-niveau sous lequel est connecté l'unité physique
MODE	représente le mode de fonctionnement du canal. Il ne peut être que HDC, MDC ou LDC
ADR	représente l'adresse coupleur
ITN	représente le niveau d'interruption canal
IOP	représente le numéro de processeur d'entrée/sortie (0) par défaut.

```
%FUIFDD ? VOIE=?
```

Cette macro-instruction définit une FU initiale disque souple.

Signification des paramètres :

"numfu"	:	précise le numéro de FU (décimal ou symbolique)
VOIE	:	précise le numéro de voie et varie de 0 à 3

Remarque : voir remarque macro %FU

```
%FUESPFDD ?
```

Description d'une FU "espace" associée au disque défini par la macro %FUIFDD. Le paramètre "numfu" obéit aux règles habituelles de ce paramètre.

Elles ont pour but la définition des périphériques conventionnels dits "standard". Si l'utilisateur a dans son installation ce genre de périphérique, il peut éviter de le décrire à l'aide des macro-instructions présentées dans les paragraphes précédents et utiliser les macro-instructions suivantes :

%HR

Description d'un lecteur rapide de bandes perforées et de la FU 5.

Cette macro-instruction n'est autorisée que si le lecteur rapide est défini comme suit :

MACRO	NIVEAU	S/NIV	MODE	ADR.	ITN	IOP
%HR	15	4	PP	'8	-	-

Description de l'unité fonctionnelle définie par %HR

MACRO	Nom	Numéro	CDE	SENS	FINDIC
%HR	HR	5	'1		0

%HP

Description d'un perforateur rapide et de la FU 6.

Cette macro-instruction n'est autorisée que si le perforateur rapide est défini comme ci-après :

MACRO	Niveau	S/Niveau	Mode	Adr.	ITN	IOP
%HP	15	5	PP	'C	/	/

Description de l'unité fonctionnelle définie par %HP

MACRO	Nom	Numéro	CDE	SENS	FINDIC
%HP	HP	6	'1	0	0

```
%CR IOP=? RACK=?
```

Description d'un lecteur de carte et de la FU 7.

Signification des paramètres :

Les paramètres IOP=? Et RACK=? sont optionnels.

- Paramètre IOP :
Ce paramètre précise le numéro de processeur qui gérera les entrées-sorties ; ce paramètre vaut 0, 1, 2, 3. Le processeur 0 (CPU) est pris par défaut.

- Paramètre RACK :
Ce paramètre ne peut être décrit que si l'on a défini un numéro d'IOP.
Ce paramètre précise si le coupleur se trouve dans le rack de base (ON), ou d'extension (OFF). Il est pris "ON" par défaut.

Syntaxe :

```
%CR [IOP=9 [RACK.=ON|OFF]]
```

Erreurs détectées :

L'erreur "paramètre incorrect" est détectée si :
- le paramètre IOP est supérieur à 3,
- le paramètre RACK est différent de ON, OFF.

Cette macro-instruction n'est autorisée que si le lecteur de cartes est défini comme ci-après :

MACRO	Niveau	S/Niveau	Mode	Adr.	ITN	IOP
%CR	14	4	LDC	'10	4	0
%CR IOP=x	14	4	LDC	'10	4	x
%CR IOP=x RACK=OFF	14	4	LDC	'810	4	x

Description de l'unité fonctionnelle définie par la macro %CR (quelque soient les paramètres associés).

MACRO	Nom	Numéro	CDE	SENS	FINDIC
%CR---	CR	7	'0	I	0

```
%LP  IOP=?  RACK=?
```

Description d'une imprimante rapide et de la FU 8.

Les paramètres, la syntaxe et les erreurs sont les mêmes que dans la macro %CR.

Cette macro-instruction n'est autorisée que si l'imprimante est définie comme ci-après :

MACRO	Niveau	S/Niveau	Mode	ADR.	ITN	IOP
%LP	14	7	LDC	'40	7	0
%LP IOP=x	14	7	LDC	'40	7	x
%LP IOP=x RACK=OFF	14	7	LDC	'840	7	x

Description de l'unité fonctionnelle définie par la macro %LP (quelque soient les paramètres associés).

MACRO	Nom	Numéro	CDE	SENS	FINDIC
%LP---	LP	8	'45	0	0

```
%MT BPI=? MODE=? IOP=? BACK=? NBV=?
```

Description d'un coupleur bande magnétique.
Les paramètres "IOP", "BACK", "NBV" sont optionnels.

Signification des paramètres :

- Paramètre BPI :
Ce paramètre précise la densité employée : 1600 BPI ou 800 BPI; il ne doit pas être omis.

- Paramètre MODE :
Ce paramètre précise le mode de fonctionnement du périphérique. Il est obligatoire, et vaut :

. LDC (Low Speed Data Channel)
. MDC (Médium Speed Data Channel)
. HDC (High Speed Data Channel).

- Paramètre IOP :
Ce paramètre précise le numéro du processeur qui gérera les entrées-sorties. Ce paramètre vaut 0, 1, 2, 3. S'il est omis, le processeur de calcul (CPU n° 0) se chargera de gérer les entrées-sorties.

- Paramètre BACK :
Ce paramètre ne peut être décrit que si l'on a défini un numéro d'IOP.
Ce paramètre précise si le coupleur se trouve dans le rack de base (ON) ou d'extension (OFF). Il est pris "ON" par défaut.

- Paramètre NBV :
Ce paramètre précise le nombre de voies utilisées sur le coupleur que l'on définit. Il vaut 1 ou 2 et 2 par défaut. S'il vaut :

. 1 on génère une unité fonctionnelle FU 9
. 2 on génère 2 unités fonctionnelles FU 9 et FU 10.

Syntaxe :

```
%MT BPI=800|1600 MODE=LDC|HDC|HDC [IOP=9 [RACK=ON|OFF]] [NBV=1|2]
```

Erreurs détectées :

L'erreur "paramètre incorrect" sera détectée si :

- le paramètre BPI est différent de 800 ou 1600

- le paramètre MODE est différent de HDC, MDC, LDC

- le paramètre IOP est supérieur à 3

- le paramètre BACK est différent de ON ou OFF

- le paramètre NBV est différent de 1 ou 2.

Cette macro-instruction n'est autorisée que si le dérouleur de bande magnétique est défini comme ci-après :

	Niveau	S/Niveau	Mode	Adr.	ITN	IOP
%HT BPI=? MODE=xDC	1à15	2	xDC	'18	2	0
%HT BPI=? MODE=xDC IOP=x	1à15	2	xDC	'18	2	x
%HT BPI=? MODE=xDC IOP=x RACK=0FF	1à15	2	xDC	'818	2	x

Description des unités fonctionnelles par la macro %MT

MACRO	Nom	Numéro	CDE	SENS	FINDIC
%MT BPI=800--- NBV=1l	T1	9	'0	IO	1
%MT BPI=1600--- NBV=1	T1	9	'0	IO	0
%MT BPI=800--- [NBV=2]	T1 T2	9 10	'0 '0	IO IO	1 1
%MT BPI=1600--- [NBV=2]	T1 T2	9 10	'0 '0	IO IO	0 0

7.3.5 Configuration de la console de service

La console de service est généralement gérée par le driver DRVASY. Les macro-instructions de définition d'unité physique et fonctionnelle sont produites en même temps que le binaire du driver grâce au pré-générateur GENASY (cf. MR DRVASY).

7.3.6 Configuration horloge temps réel

```
%HTR ? SNIV=? FREQ=? ADR=?
```

Cette macro-instruction sert à définir l'horloge temps réel. Elle ne peut être utilisée qu'une seule fois.

Signification des paramètres

- 1er paramètre =

  . MFI : horloge du coupleur MFI gérée par DRVHTR
  . CMF : " " " " CMF " " DRVHTR
  . par défaut : " " " " " " " DRVCMF

- SNIV : indique le sous-niveau des interruptions horloge. C'est un nombre décimal inférieur à 16

- FREQ : indique la fréquence des interruptions horloge (en nombre de tops par seconde). C'est un nombre décimal inférieur ou égal à 1000.

- ADR : représente l'adresse de l'horloge (nombre hexadécimal).

Les différentes formes d'écriture sont :

FORME DE LA MACRO	Paramètres retenus				DRIVER
	TYPE CPL	SNTV	FREQ	ADR	
%HTR	CMF	0	50	'17E0	DRVCMF
%HTR CMF	CMF	0	50	'17FF	DRVHTR
%HTR MFI	MFI	0	50	'17FC	DRVHTR
%HTR SNiV=n FREQ=f ADR=@	CMF	n	f	@	DRVCMF
%HTR CMF SNIV=n FREQ=f ADR=@	CMF	n	f	@	DRVHTR
%HTR MFI SNIV=n FREQ=f ADR=@	MFI	n	f	@	DRVHTR

Syntaxe :

```
%HTR [MFI|CMF] [SNIV=99 FREQ=9999 ADR='FFFF]
```

Erreurs détectées :

- "paramètre incorrect" si :
  . SNIV supérieur à 15
  . FREQ supérieur à 1000.

- "macro interdite" si ce n'est pas la première macro %HTR rencontrée.

Remarques :

- la macro %HTR définit une unité fonctionnelle gérée de manière spécifique par IOCS16.

- le driver correspondant sera intégré dans le superviseur.

```
    %FU ?    CHIEN= ? ?
```

Cette macro-instruction peut être placée à n'importe quel endroit et permet d'associer un chien de garde à une FU.

Signification des paramètres :

- 1er paramètre : numéro de FU exprimé en décimal ou en symbolique

- 2me paramètre : valeur du chien de garde exprimé en décimal ; il doit être inférieur à 128

- 3me paramètre : indique que le chien est exprimé en minutes (MIN) ou en secondes (SEC)

Syntaxe :

```
          %FU XX|999 CHIEN=999 MIN|SEC
```

Erreurs détectées :

- pour le 1er paramètre :

  - il est de type décimal et supérieur à 125.
  - le symbole n'est pas reconnu (cf. MEMO16)

- pour le second paramètre, il est supérieur à 127
- pour le troisième paramètre, il est différent de MIN ou SEC

Remarque : en cas de double déclaration pour une même FU, la seconde est seule considérée.

7.3.8 Fin de génération

```
%ENDGEN
```

Cette macro-instruction termine la génération. Toute macro se trouvant après celle-ci sera donc rejetée.
Elle est obligatoirement suivie de la directive *END qui provoque le retour au superviseur.

Bull .3.9 Description de svstèmes BOS16

Cet exemple décrit une gestion d'espace dynamique.

La configuration comprend :

- un disque CDB20 support système,
- un dérouleur de bande magnétique 800 BPI
- une imprimante,
- une console de service gérée par DRVASY (pour la macro %MFI, se reporter au M.R. DRVASY).
- un lecteur de carte.

Dans cet exemple on utilise les macro-instructions "standard système" et "périphérique" afin de simplifier l'écriture des macro-instructions.

```
 ----------------------------------------------
‹              FICIOC-BG
<----------------------------------------------
%BOS16
%POOL NBUF=4 LBUF=80
<------------NIVEAU 14------------------------
%NIVEAU 14 KSTOR=30
<------------DISQUE VM----------------------
%PUVM SNIV=1 MODE=HDC ADR='30 ITN=1 IOP=1 NBV=1
%FUIVM DB VOIE=0 FIXE=N
%FUESPVM D1
%FUESPVM D2
%FUIVM DA VOIE=0 FIXE=Y
%FUESPVM D3
%FUESPVM D4
%FUESPVM D5
%FUESPVM D6
%FUESPVM D7
%FUESPVM D8
%FUESPVM D9
‹
<------------- BIDE GENETIQUE----------------
%MT BPI=800 MODE=MDC
<-------------LECT.CARTE,IMPRIMANTE------------
%CR
%LP
‹
<------------NIVEAU 15------------------------
%NIVEAU 15
%MFI ASY TTY
%ENDGEN
<----------------------------------------------
```

7.4 RECAPITULATION DES MACRO-INSTRUCTIONS

```
%UC 16-35|70|90 [PAS DE COMPTAGE]

%IOP RAPIDE=9[,9][,9]
%NTACHES=999
%SUMAX=999
%INPMAX=9999
%POOL NBUF=99 LBUF=999
%FSMON XXXXX NUM=999

%DEFPU XXX CODE=8|16 LZONE=999 SP=[Y|N]

%SYMBEXT xxxxxx
%BOS16
%RTES16
%MPES16
%TSM16
%MUTEX16
%NIVEAU 99 KSTOR=[999]

%PU XXX SNIV=99 MODE=PP|LDC|MDC|HDC ADR='FFFF ITN=[99]
                          CONNEX=[STD|ASYV1] IOP=9

%TUP-99=xxxxxx

%FU XX|999 CDE='FFFG SENS=I|O|IO FINDIC=0|1

%CPMPX XXX SNIV=99 MODE=PP|LDC|MDC|HDC ADR='FFFF ITN=[99]
                          CONNEX=[MUX8|MUX16] IOP=9 NBV=9

%PUMPX XXX VOIE=99

%PUDX xxx BEGIN=99 NUMBER=99 STEP=9 ADR='FFFF

%CPMUX4 MODE=PP|LDC|MDC|HDC ADR='FFFF IOP=[9]
                          [CONNEX=[STD|MUX4|ASYV1|CMF|MUX4U]]

%PUMUX4 XXX SNIV1=99 SNIV2=[99] ITN=[99] VOIE=9

%PUVM SNIV=99 MODE=MDC|HDC ADR='FFFF ITN=99 IOP=[9] NBV=9

%FUIVM XX|999 VOIE=9 FIXE=Y|N

%FUESPVM XX|999

%PUDP SNIV=99 MODE=MDC|HDC ADR='FFFF ITN=99 IOP=[9] NBV=9

%FUIDP XX|999 VOIE=9

%FUESPDP XX|999

%HR
%HP

%CR [IOP=9 [RACK=ON|OFF]]

%%LP [IOP=9 [RACK=ON|OFF]]

%MT BPI=800|1600 MODE=LDC|MDC|HDC [IOP=9 [RACK=ON|OFF]] [NBV=1|2]

%HTR [MFI|CMF] [SNIV=99 FREQ=9999 ADR='FFFF]

%PUFDD SNIV=99 MODE=LDC|MDC|HDC ADR='FFFF ITN=99 IOP=9

%FUIFDD XX|999 VOIE=9
```

```
%FUESPFDD  XX|999
%CPSND  SNIV=99  ADR='FFFF  ITN=99  IOP=9
%FUISMD  XX|999  UNIT=99  FIXE=Y|N
%FUESPSMD  XX|999
%CPSAS  SNIV=99  MODE=MDC|HDC  ADR='FFFF  ITN=99  IOP=9
%FUISAS  XX|999  UNIT=9
%FUESPSAS  XX|999
%FUDK  XX|999 ADRDK=9999 NSECT=9999
%FU  XX|999 CHIEN=999 MIN|SEC
%ENDGEN
```

Réseaux
et systèmes
d'information

Bull

VOS REMARQUES SUR CE DOCUMENT

Bull Sems

TITRE

IOCS16

N° DE REFERENCE

Bull-Sems : 1 164 213 02 030 02

DATE

MAI 1985

ERREURS DETECTEES

AMELIORATIONS SUGGEREES

*-> Vos remarques et suggestions seront attentivement examinées.
si vous désirez une réponse écrite, veuillez indiquer ci-après
votre adresse postale complète.

NOM : ..... DATE .....

SOCIETE : .....

ADRESSE : .....

*-> Remettez cet imprimé à un responsable Bull-Sems ou envoyez le
directement à

Bull-Sems
Méthodes G.I.
Rue de Provence
38130 - ECHIROLLES

Réseaux
et systèmes
d'information

Bull

Distribution codes/Codes de diffusion			
Customers : Clients :			
Internal : Interne :			

DELIVERY ADDRESS

ETIQUETTE ADRESSE

Bull MTS

1, Rue de Provence
B.P. 208
38432 ÉCHIROLLES CEDEX / FRANCE

Bull

Sems