Réseaux
et systèmes
d'information

Bull

Manuel de référence

Bull

Sems

SOLAR

RTES16
ADDENDUM A

Logiciel

SUJET : Création d'une version téléchargeable de RTES16

OBSERVATION : Première mise a jour du document 1 164 324 00 036 03 d'avril 1985. Suivre les directives de mise à jour et placer la présente feuille directement après la couverture de votre manuel pour indiquer que celui-ci est à l'indice 04.

VERSION LOGICIEL :

DATE : JANVIER 1986

REF Bull-Sems : 1 164 324 00 036 04/FR

Réseaux
et systèmes
d'information

Bull

Bull

Sems

(C) Bull-Sems 1986

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.

Ce document est fourni à titre d'information seulement. Il n'engage pas la responsabilité de Bull-Sems en cas de dommages résultant de son application. Des corrections ou modifications au contenu de ce document peuvent intervenir sans préavis ; des mises à jour ultérieures les signaleront éventuellement aux destinataires.

Réseaux
et systèmes
d'information

Bull

RTES 16

MANUEL DE REFERENCE

A en haut de page indique le changement complet de la page par rapport à l'IE précédent

I en marge indique la partie modifiée par rapport à l'IE précédent

SOMMAIRE

Pages

AVANT PROPOS

Le lecteur trouvera essentiellement dans ce manuel des exemples pratiques de génération de RTES16 et de mise en oeuvre de son application.

Les chapitres qui le constituent font fréquemment référence à des notions de système ou de langage supposées acquises et qui ne sont que partiellement reprises.

Si ce manuel est donc en quelque sorte le dernier à lire, son contenu n'en est pas moins de première importance.

Dans ce manuel les références à IOCS16 et FMS16 ont été respectivement remplacées par IOCS et FMS.

1 - PRESENTATION

RTES16 (REAL TIME EXECUTIVE SYSTEME16) est le moniteur disque conçu pour l'exploitation des SOLAR 16-35 et du SOLAR 16-70 en multiprogrammation dans un contexte Temps Réel.

II permet en outre la production de programmes par l'utilisation du moniteur Background BACKM en simultanéité avec le déroulement de l'application temps réel.

Par sa conception modulaire orientée vers la souplesse de création, de mise en œuvre et d'exploitation des programmes de l'usager, RTES16 permet non seulement une configuration agréable et progressive de son application, mais surtout offre un ensemble d'outils puissants destiné à son évolution et son optimisation.

Souplesse de mise en œuvre et efficacité des services doivent être assortis, en temps réel, avec une sécurité absolue : ainsi, RTES16 assure la protection hardware du moniteur et des programmes en cours d'exploitation vis à vis des programmes en cours de mise au point ; il donne à l'usager des éléments permettant un redémarrage dans un contexte précis après une disparition du secteur.

Dans un système temps réel, la notion de gestion du temps est prépondérante : temps de réponse, évolution dans le temps, utilisation du temps machine sont autant d'impératifs essentiels.

Les caractéristiques multitâches de RTES16 permettent d'utiliser au mieux le temps machine ; d'autre part l'usager contrôle à tout instant et de façon précise l'évolution dans le temps qui traduit "l'histoire" de son application ; enfin RTES16 assure des temps de réponse performants en liaison directe avec la rapidité des fonctions "système" microprogrammées du Solar 16.

RTES16 présente une entière compatibilité ascendante à l'égard des autres systèmes du software de base du SOLAR 16, au niveau de l'utilisation des outils fondamentaux que sont le moniteur d'entrée-sortie IOCS, le système de fichier FMS et le flottant programmé.

Le respect de la norme FORTRAN temps réel en cours d'élaboration (appel, temporisation de tâche par exemple) confère en outre à RTES16 un aspect d'universalité et de standardisation des services proposés.

Enfin, l'utilisation d'un disque lui assure sa grande capacité de stockage des informations, et, par l'intermédiaire du système de fichiers, la sécurité d'accès et de partage de ces informations. Sous RTES16, le disque est par définition l'outil prépondérant de stockage des données et de communications entre tâches.

1.1 - CONFIGURATION MATERIEL

- 64 K mots à 1024 K mots de mémoire centrale
- 1 téléimprimeur ou 1 visualisation alphanumérique
- Horloge temps réel secteur 50 Hz ou horloge à quartz à fréquence sélectable jusqu'à 1000 Hz
- Scheduler microprogrammé (MTS16)
- DRPS16
- 1 disque à têtes fixes de 256 K mots ou 1 disque à cartouches de 2.5 M octets.

Remarque :

Les options CDA et DAP 16 (flottant double précision) sont gérées en standard par RTES16.

1.2 - BULLETIN TECHNIQUE

FONCTIONS	CARACTERISTIQUES
SYSTEME DISQUE	SOUPLESSE D'EMPLOI STOCKAGE DE DONNEES ET DE PROGRAMMES SYSTEME DE FICHIERS MULTI-ACCES MODULAIRE
MULTIPROGRAMMATION	SCHEDULER SOLAR16 MICROPROGRAMME 128 PRIORITES SOFTWARE * 16 PRIORITES HARDWARE (PLUS DE 200 SOUS NIVEAUX). UTILISATION OPTIMALE DE L'UNITE CENTRALE
TACHES RESIDENTES SUR D I S Q U E	UTILISATION EFFICACE DE LA MEMOIRE CENTRALE
PROTECTION HARDWARE DES PROGRAMMES	PROTECTION INTERTACHE (DRPS) SIMPLIFICATION DE LA MISE AU POINT FIABILITE DU SYSTEME
RELOCATION HARDWARE DES PROGRAMMES	GEREE PAR LE SYSTEME UTILISATION OPTIMALE DE LA MEMOIRE CENTRALE
GESTION DES ENTREES/SORTIES	ORGANISEE AUTOUR DE L'IOCS STANDARD ET DES DRIVERS DE PERIPHERIQUES CONVENTIONNELS ET INDUSTRIELS.
FORTRAN IV TEMPS REEL	LANGAGE STANDARD SIMPLIFICATION DE L'ECRITURE ET DE LA MAINTENANCE DES PROGRAMMES
BACKGROUND	PRODUCTION SIMULTANEE DE PROGRAMMES EXPLOITATION TYPE CENTRE DE CALCUL
DIALOGUE OPERATEUR FOREGROUND	MULTI-USAGES - CHARGER, SUPPRIMER DES TACHES ON-LINE - LANCER, ARRETER DES TACHES - IMPRIMER L'ETAT DU SYSTEME, D'UN PERIPHERIQUE...

2 - INTRODUCTION A RTES16

La dénomination ("temps réel" recouvre un ensemble de concepts spécifiques repris et développés en quelques lignes.
Au delà des fonctions classiques réalisées par les moniteurs "temps réel", ce chapitre présente la ((philosophie)) du système RTES16 et met l'accent sur des aspects pratiques de son utilisation.

2.1 - CONCEPTS FONDAMENTAUX

2.1.1 - Système et application

L'emploi de ces deux termes n'est pas indifférent ; on désignera sous le nom de système (ou système RTES16) le produit software qui, dimensionné aux besoins de l'usager, est prêt à accepter les programmes qu'il doit gérer en temps réel.
Ce système sera configuré à partir de programmes du software de base livrés au client, parmi lesquels figure le noyau du système (standard, non configuré par l'usager).
Le terme application recouvre d'une part le système, d'autre part l'emploi qu'en fait l'usager, c'est à dire l'ensemble des tâches qu'il fait gérer à RTES16 pour contrôler son processus.

2.1.2 - Multiprogrammation

La multiprogrammation est la technique par laquelle un système exécutif gère l'allocation des ressources d'une installation (utilisation de l'unité centrale, des périphériques, de la mémoire centrale,...) entre plusieurs tâches concurrentes.
Cette technique d'exploitation accroît l'efficacité du calculateur, son temps de réponse, et sa souplesse opérationnelle puisque les délais logiques d'une tâche sont "masqués" par l'exécution d'autres tâches.
L'efficacité de la multiprogrammation réside dans la présence simultanée de plusieurs tâches en mémoire centrale ; certaines d'entr'elles passant une partie de leur temps de présence en attentes diverses (fin d'opérations d'entrée-sortie, synchronisation, temporisations...) d'autres sont disponibles pour l'utilisation des ressources de la machine.

2.1.3 - Priorité et "Scheduling"

La notion de priorité est nécessaire pour régler les conflits d'accès aux ressources du système.
Dans RTES16 chaque tâche d'une application a sa propre priorité ; tout accès à une partition de la mémoire pour-exécution; tout accès à un dispositif périphérique sont gérés à travers le critère de priorité relative des divers demandeurs par le "Scheduler".

RTES16 exploite la structure multitâche et le système de scheduling microprogrammés du SOLAR 16 à la différence de la plupart des moniteurs actuels, cette structure fait partie intégrante de la machine et se reflète sur le système par ses performances et son temps de réponse.

2.1.4. - "Foreground" et "Background"

Alors que la sphère des programmes temps réel (ou Foreground) effectue des acquisitions de données, contrôle un processus industriel ou des expériences de laboratoire, en un mot répond aux "stimuli" externes de l'application, un ensemble de programmes moins critiques et d'intérêt plus général, le Background, utilise les ressources laissées disponibles dans un but de production de nouveaux programmes destinés par exemple à contrôler de futurs processus, et dans un but d'exploitation de ces programmes (calculs scientifiques ou traitement de données off-line par exemple).

2.2 - CARACTERISTIQUES DE RTES16

Les principales caractéristiques du système développées dans les chapitres suivants résultent de l'analyse des applications temps réel et des critères suivants qui s'en dégagent :

- Facilité de mise en oeuvre de l'application
- Protection du système et des tâchescritiques vis à vis des tâches en cours de mise au point.
- Possibilité d'intervention ON-LINE de l'opérateur pour surveiller et corriger l'évolution de son application, ou la compléter.
- Gestion optimisée de la mémoire centrale et du disque.

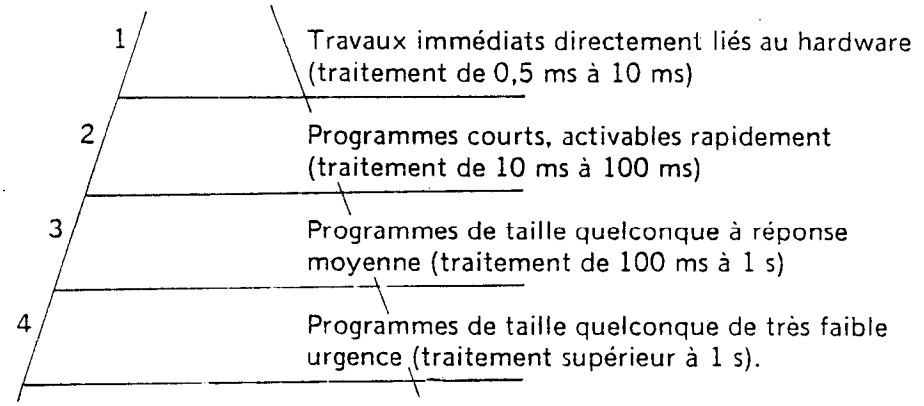
2.2.1 - Dialogue système-opérateur

L'opérateur exerce un ultime contrôle sur l'application par l'intermédiaire de commandes d'une part et des messages et diagnostics d'erreurs qui lui sont signalés d'autre part.
Cette fonction de dialogue, puissante et interactive, est vitale pour l'application, et donc assortie d'un maximum de sécurité (mot de passe, messages d'erreurs...).
L'opérateur peut à l'aide de ces commandes :

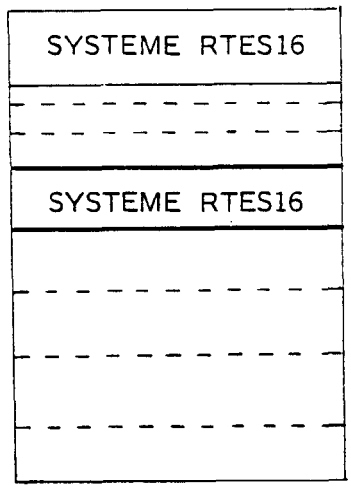
- introduire en ligne de nouvelles tâches
- modifier l'heure
- initialiser le système
- visualiser des états du système, les sauver sur disque.

2.2.2. - Gestion mémoire

L'analyse des applications "temps réel" permet de dégager une structure partitionnée des différents travaux liés à l'application. Cette structure est schématisée ci-après.



La gestion proposée s'inspire de ce schéma. Les programmes de type 1 et 2 sont en général résidents en mémoire. Par contre, les programmes 3 et 4 sont résidents sur disque. D'où le schéma ci-après de la géographie mémoire.



Zone partitionnée des programmes résidents en mémoire.

Zone partitionnée des programmes résidents sur disque.

La zone des programmes résidents-disque est découpée en partitions de tailles différentes. Ce partitionnement permet l'exécution des tâches de l'application suivant une hiérarchie utilisateur.

Une tâche non résidente qui occupe une partition est une tâche qui doit se terminer par un "SVC EXIT" ; l'allocation de la partition occupée par cette tâche ne pouvant se faire que lorsque la tâche est terminée.

Les tâches s'exécutant dans une même partition sont en général exclusives.
Un paragraphe de cette introduction est consacré plus précisément à la réalisation de cette gestion mémoire.

2.2.3 - Background

La fonction de production de programmes est réalisée par le moniteur BACKM. version adaptée du système BOS16 et d'une utilisation compatible. Ce moniteur occupe une partition d'au moins 2 Kmots en dessous de 32K pendant toute la durée de l'activité background.

Les programmes exploités sous BACKM s'exécutent dans la ((partition background)) ; cette partition background est en fait constituée d'une ou plusieurs partitions temps réel consécutives de type non résident.

Un programme background ne répond pas à la condition de terminaison dans un temps donné imposée aux tâches temps réel de l'application. Lorsque la partition Background est requise pour l'exécution de l'une de ces tâches, le système en sauvegarde l'image sur le disque (SWAP - OUT)

Ce mécanisme de SWAP n'est appliqué qu'au Background, toute tâche "temps réel" restant dans sa partition jusqu'à la fin de l'exécution en cours, puis étant "écrasée" par la prochaine tâche alimentée dans la même partition (technique de l'OVERLAY sur les partitions, à dissocier de l'OVERLAY sur les BRANCHES d'une tâche).

2.2.4 - Gestion des Entrées-Sorties

Toutes les opérations d'entrée-sortie sont réalisées par un moniteur unique, IOCS, qui se retrouve dans tous les systèmes d'exploitation développés de façon standard pour les calculateurs de la gamme SOLAR 16.

Cet interface d'entrée-sortie, compatible pour l'ensemble des systèmes, d'exploitation, est en outre modulaire, c'est à dire que l'utilisateur peut aisément lui ajouter de nouveaux drivers de périphériques.

Ainsi, RTES16 présente des drivers et des requêtes adaptés aux fonctions industrielles (entrées -sorties analogiques et digitales).

2.2.5 - Gestion du disque

Une caractéristique de RTES16 est l'unicité du type d'accès au disque quelles que soient les fonctions à réaliser.

- traitement des fichiers de l'usager
- alimentation de tâches
- overlay sur les branches d'une tâche.

Cette unicité de type d'accès, réalisée par l'intermédiaire du système FMS (File Management System), assure l'optimisation d'utilisation de l'espace disque, la protection d'accès aux divers fichiers et la modularité du système basé sur un environnement disque.

2.2.6 - Autres fonctions de RTES16

RTES16 assure en outre :

- la prise en compte et le traitement des interruptions "système" du SOLAR 16 (alarmes)
- la prise en compte et le traitement de la disparition et la réapparition du secteur
- la gestion des ressources du système
- l'activation des tâches de l'application à une heure donnée, après un délai initial donné, l'activation périodique des tâches, la temporisation des tâches.

2.3 - ORGANISATION DE LA MEMOIRE CENTRALE

L'occupation mémoire de RTES16 se divise en cinq zones.

la zone occupée par le système (FMS, IOCS, Noyau de RTES16, Drivers, zone d'overlay du dialogue)

la zone réservée à la gestion des informations dynamiques du système (ZUEP, ZIOCB, ZDF, ZDM, ZWCB, ZGIN)

la zone de pavé utilisateur : ZWB (Zone Working Bloc)

la Zone des Blocs de Contrôle des Tâches de l'application ZBCT

la Zone des Données Résidentes ZDR et/ou la Zone de Communication CDA.

la zone des partitions des tâches de l'application (partitions de type résident et non résident).

2.3.1 - Principe

La zone occupée par le système comprend des tables dont la taille dépend de la configuration physique de l'installation et du découpage du (des) disque (s) en unités fonctionnelles.

Les zones réservées à la gestion des informations dynamiques du système sont également configurables ; c'est l'espace de travail du système. Il gère dans ces zones les demandes d'activation de tâches, les attentes d'événements, les ressources définies par l'usager, les opérations sur horloge, les demandes d'ouvertures et d'accès aux fichiers...

La zone des blocs de contrôle de tâches est configurable lors de la génération du système : elle est constituée de blocs adjacents de 38 mots, un bloc par tâche de l'application, chaque bloc contenant la PST de la tâche et les informations systèmes associées.

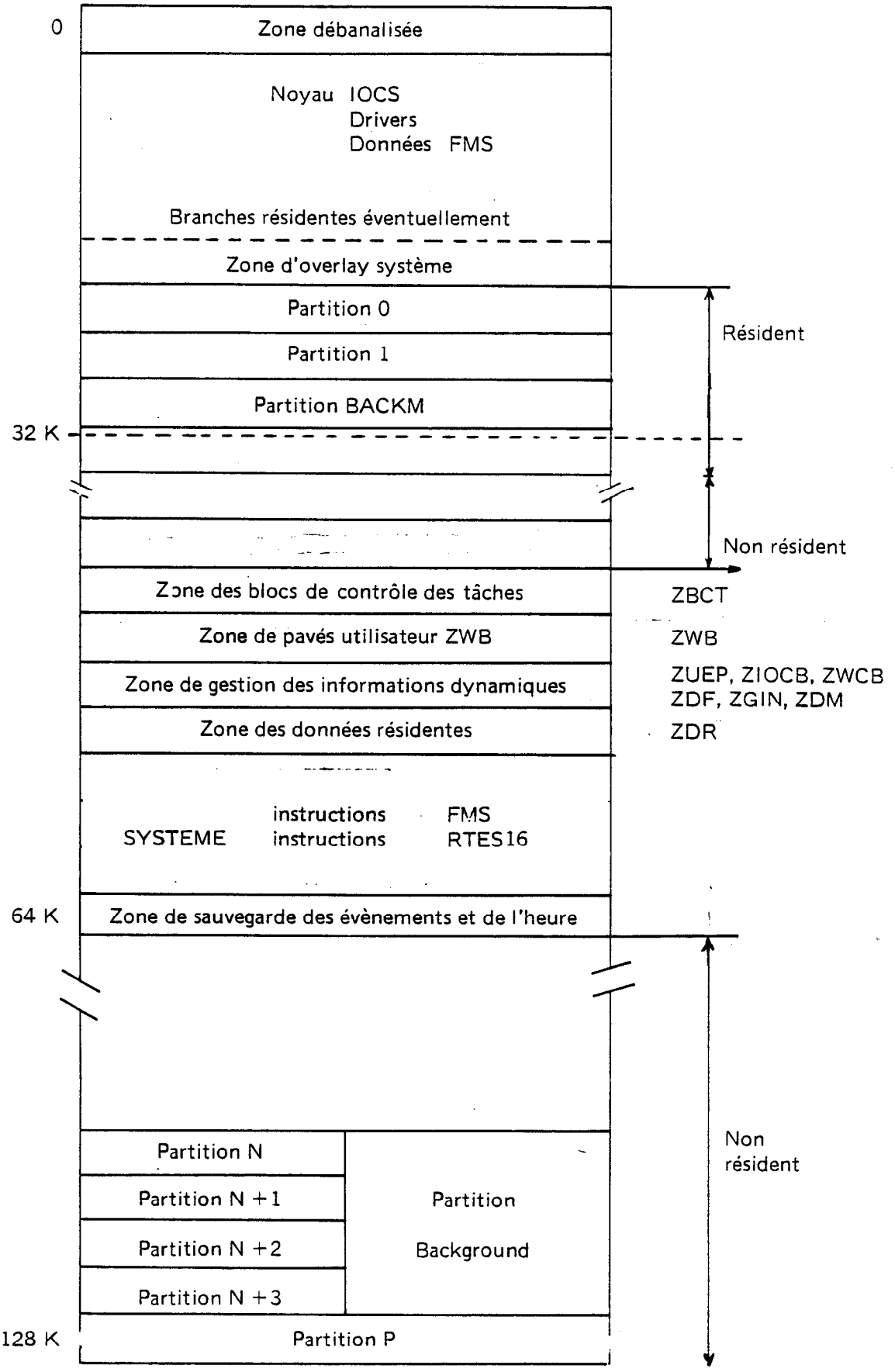
La zone-des données résidentes permet le transfert d'information entre tâches et la sauvegarde des données modifiables entre deux exécutions d'une même tâche.

La zone de communication CDA remplit la même fonction que la ZDR, d'une façon plus performante.

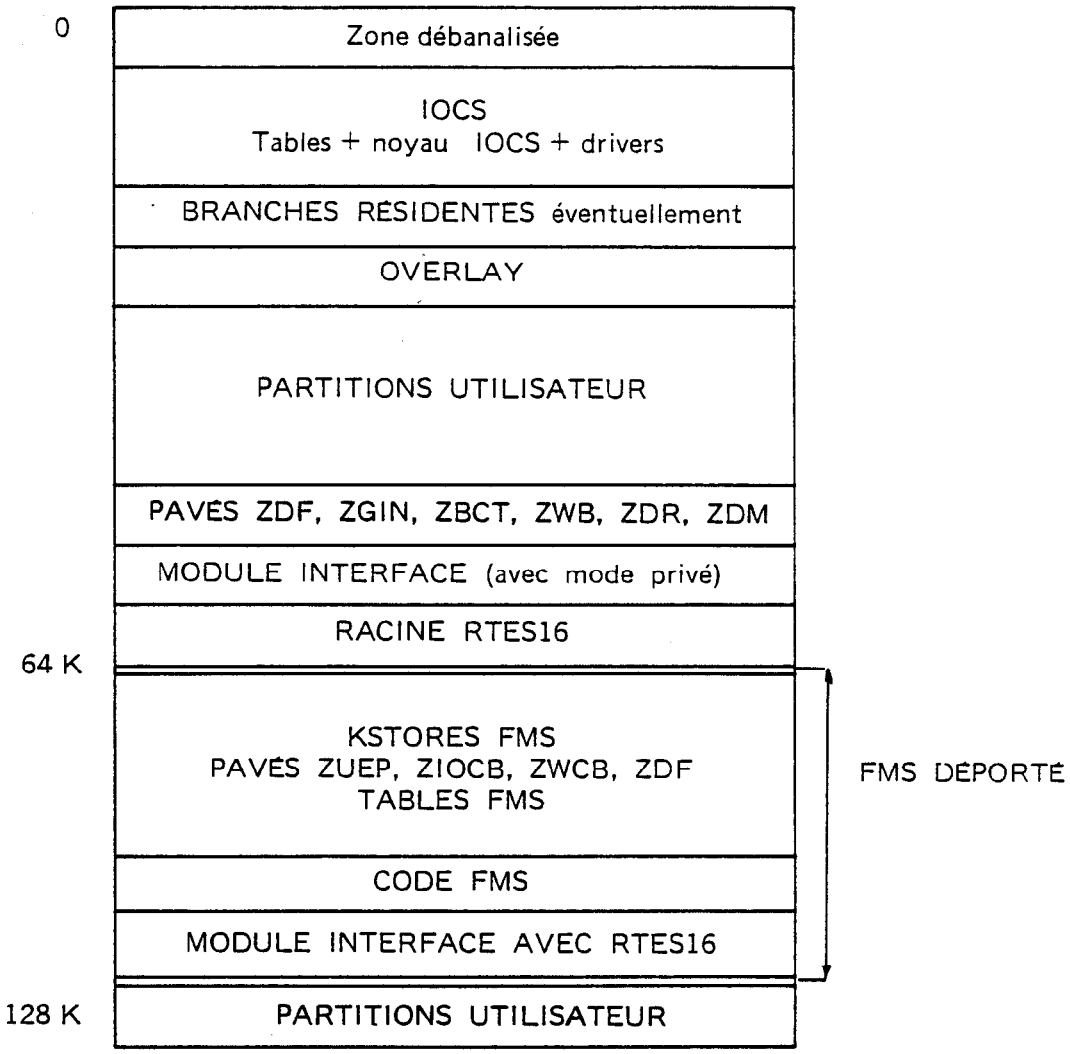
La zone des partitions s'implante sur toute la mémoire résiduelle. Le partitionnement est réalisé par l'utilisateur pour lui permettre de résoudre ses différents problèmes ou conflits : temps de réponse, étreinte fatale, encombrement mémoire.

2.3.2 - Carte de la mémoire

— Sans prise en compte du mode privilégié



- Avec prise en compte du mode privilégié



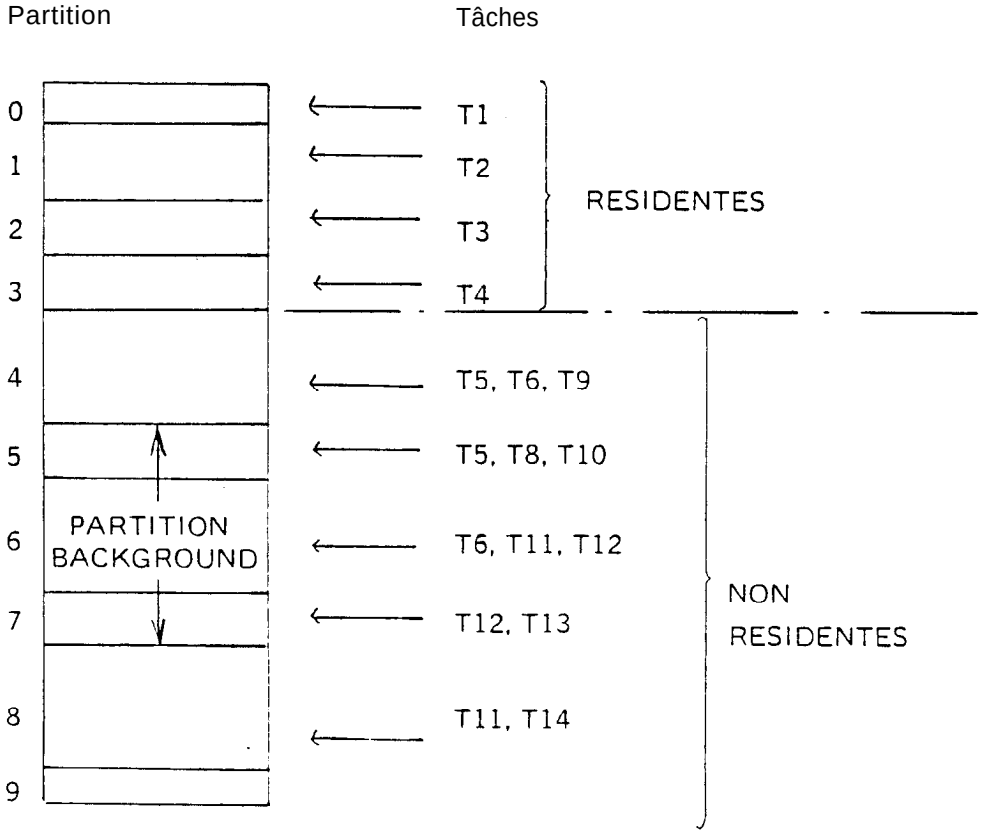
2.3.3 - Gestion des partitions

Les règles suivantes régissent l'utilisation des partitions :

- RTES16 gère au maximum 128 partitions.
- A un instant donné, une seule tâche peut occuper une partition. Cette règle générale peut, dans le cadre d'applications particulières, être contournée par utilisation de création dynamique de tâches : TASKC.
- A un instant donné, une tâche présente en mémoire occupe une seule partition (pas de chevauchement sur deux ou plusieurs partitions).
- Dans le temps, une même tâche peut être exécutée dans différentes partitions, plus précisément dans le sous-ensemble des partitions (4 au maximum) déclarées accessibles à cette tâche par l'opérateur.
- Au moment de son activation, une tâche non présente est chargée depuis le disque dans une partition qui lui est accessible avec son image initiale ; c'est la technique de l'"overlay" dans les partitions.
- Le moniteur background BACKM occupe sa partition pendant toute la durée de l'activité background.
- Les processeurs background s'exécutent dans une ou plusieurs partitions non résidentes consécutives ; la taille de cette "partition background" est configurable et comprise entre 3 K et 64 Kmots.

  Les partitions temps réel ainsi recouvertes sont rendues accessibles par le mécanisme de SWAP qui les particularise.

2.3.4 - Exemple de partitionnement :



2.4 - TACHE UTILISATEUR

2.4.1 - Définitions

L'unité de base de programme sous RTES16 est la "tâche" ; elle consiste en un programme ou un ensemble de programmes écrits en langage FORTRAN, PL16 ou ASSEMBLEUR ASM16.

Elle est produite sur disque par utilisation des processeurs du software de base (compilateur, éditeur de liens...) ; ultérieurement, elle est "chargée" sous RTES16 par une commande de l'opérateur qui l'introduit dans l'application.

Une tâche est alors identifiée en tant qu'entité fonctionnelle de RTES16 par sa priorité ou son nom (tâches mères - tâches filles) et son type (hardware ou software) et par son contexte qui regroupe les informations statiques et dynamiques relatives à la tâche.

Au niveau externe, une tâche est identifiée par son numéro d'appel ; l'opérateur-dispose de commandes d'intervention ON-LINE qui font référence à une tâche par son numéro d'appel ; les autres tâches de l'application font référence à une tâche donnée, dans certaines requêtes programmées, également par son numéro d'appel. II s'agit d'un numéro symbolique, la priorité étant au contraire un numéro fonctionnel.

Au moment de l'écriture de la tâche, l'usager lui assigne une "priorité et un type par défaut" : ce sont les caractéristiques fonctionnelles avec lesquelles elle s'exécutera si l'opérateur, par la commande appropriée, ne les modifie pas.

Le numéro d'appel, spécifié lors de la création de la tâche, est immuable.

2.4.2 - Exécution dans une partition

On a vu qu'une partition est une zone contigue de mémoire centrale utilisée pour l'exécution de tâches.

Toutes les tâches utilisateur s'exécutent dans des partitions, qu'elles soient résidentes en mémoire ou résidentes sur disque, de type hardware ou de type software.

Au moment de l'écriture de la tâche, l'usager lui assigne une partition "par défaut" : c'est la partition dans laquelle elle s'exécutera normalement si l'opérateur, par la commande appropriée ne modifie pas ON-LINE cette affectation. Par cette commande: il peut alors allouer à une tâche une autre partition, ou une listé de partitions "possibles" (4 au maximum)..

A l'exception des tâches nécessitant un temps de réponse minimum (tâches hardware en particulier), les tâches utilisateur sont d'une façon générale résidentes sur disque et sont chargées en mémoire centrale dans une partition avant leur exécution.

L'état de résidence d'une tâche est figé non pas à sa création, mais au moment de son introduction sous RTES16.

Une tâche résidente sur disque libère sa partition en fin d'exécution ; elle n'est pas réécrite sur disque (pas de SWAP-OUT) ; lors de sa prochaine activation, si la partition antérieurement allouée a été occupée par une autre tâche, elle sera lue sur disque dans sa version initiale. D'une exécution à l'autre, une tâche résidente sur disque ne pourra donc pas se communiquer des données par des mémoires de travail.

Remarque : Le ((contexte)) des tâches est résident en mémoire centrale : les registres d'une tâche ne sont donc pas détruits d'une exécution à l'autre (exceptions : les deux registres RA et RX détruits par la SVC EXIT de fin de tâche).

2.4.3 - Protection intertâche

Les tâches temps réel particulièrement critiques doivent s'exécuter dans un environnement présentant un maximum de sécurité vis à vis des autres tâches de l'application ; de même, le moniteur, coeur de l'application, doit être intégralement protégé des tâches en cours de mise au point.

RTES16 utilise la protection intertâche câblée du SOLAR 16 (DRPS16). Le moniteur s'exécute en mode maître ; d'une façon générale, les tâches de l'utilisateur (et les processeurs Background) opèrent en mode esclave et sont protégéesindividuellement. Pour exécuter les instructions privilégiées qui leur sont interdites, elles disposent de requêtes programmées au moniteur qui réalisent toutes les opérations de contrôle assurant à l'application un maximum de sécurité.

La protection intertâche est assurée par RTES16 sous forme de protection inter-partitions : au moment de l'activation d'une tâche, après lecture en mémoire depuis le disque, ses deux registres de protection sont initialisés avec les adresses absolues qui délimitent la partition occupée ; l'espace de travail d'une tâche est donc celui de la partition qu'elle occupe.
Le processus est le même pour les tâches qui résident en mémoire ; cependant il n'est pas "dynamique", de telles tâches s'exécutent en permanence dans la même partition : les registres de protection sont donc initialisés au moment du chargement (unique) de la tâche en mémoire.

2.4.4 - Etats d'une tâche

Ils caractérisent l'ensemble des tâches de type software.

- Les états de fonctionnement :
Ils correspondent à l'état des files "système" du scheduler microprogrammé (cf Manuel de Présentation du SOLAR 16) et des files internes de RTES16.

. inexistante : la tâche n'a pas été intégrée sous RTES16.

. armée : le système a enregistré au moins une demande d'activation immédiate

. en attente ou masquée : la tâche est en attente d'une fin d'opération d'entrée-sortie, d'un accès à une ressource, d'un événement ou d'une combinaison d'événements, d'un délai...

. éligible : armée et non masquée

. non prête : la tâche est éligible, mais le système doit réaliser pour elle un "complément de scheduling" avant de la lancer (la charger en mémoire si elle n'est pas présente ; attendre qu'elle soit validée si elle est inhibée)

. activable : éligible et prête

. active : la tâche activable de plus haute priorité est la tâche active.

- Les états de présence :

. résidente : la tâche réside en mémoire centrale ; cet état est figé lors de l'intégration de la tâche sous RTES16.

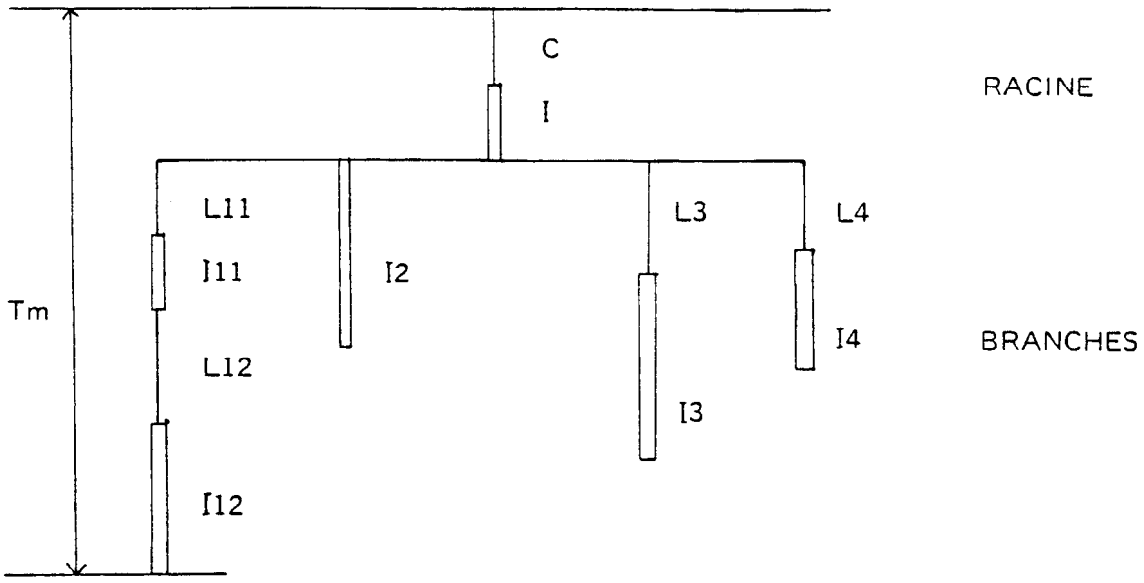
. présente : une tâche résidente est par définition toujours présente
une tâche non résidente libère sa partition par la requête CALL EXIT ; elle devient non présente dès que cette partition est allouée à une autre tâche.

2.4.5 - Structure d'"OVERLAY" (cf Manuel de Référence de BOS16

Les tâches de l'application de niveau software peuvent être structurées en "OVERLAY "

Une telle tâche se compose :

- d'une racine (données communes C + instructions I)
- de N branches (données locales LJ + instructions IJ)
  $N \in [0,255]$



Une tâche en "overlay" est exécutée dans une partition dont la taille minimale Tm est égale à la taille de la racine plus la taille de la plus grande branche (taille de l'image Mémoire)

Le chargement en mémoire d'une branche se fait par écrasement de la précédente.

La racine est présente en mémoire pendant toute la durée du déroulement de la tâche ; l'appel d'une branche provoque l'écrasement de la précédente sans sauvegarde sur disque ; le transfert des paramètres entre branches doit se faire par les registres, par le système de fichiers ou par les données communes de la racine.

L'activation et l'abandon d'une tâche en "overlay" se font dans la racine; la tâche est reprise à l'instruction qui suit la demande d'abandon précédente.

L'activation par le scheduler comporte au plus 2 phases :

- chargement de la racine (cas d'une tâche non présente)
- lancement de la tâche au point de reprise dans la racine

Le lancement d'une branche par la racine ou une autre branche comporte au plus 2 phases :

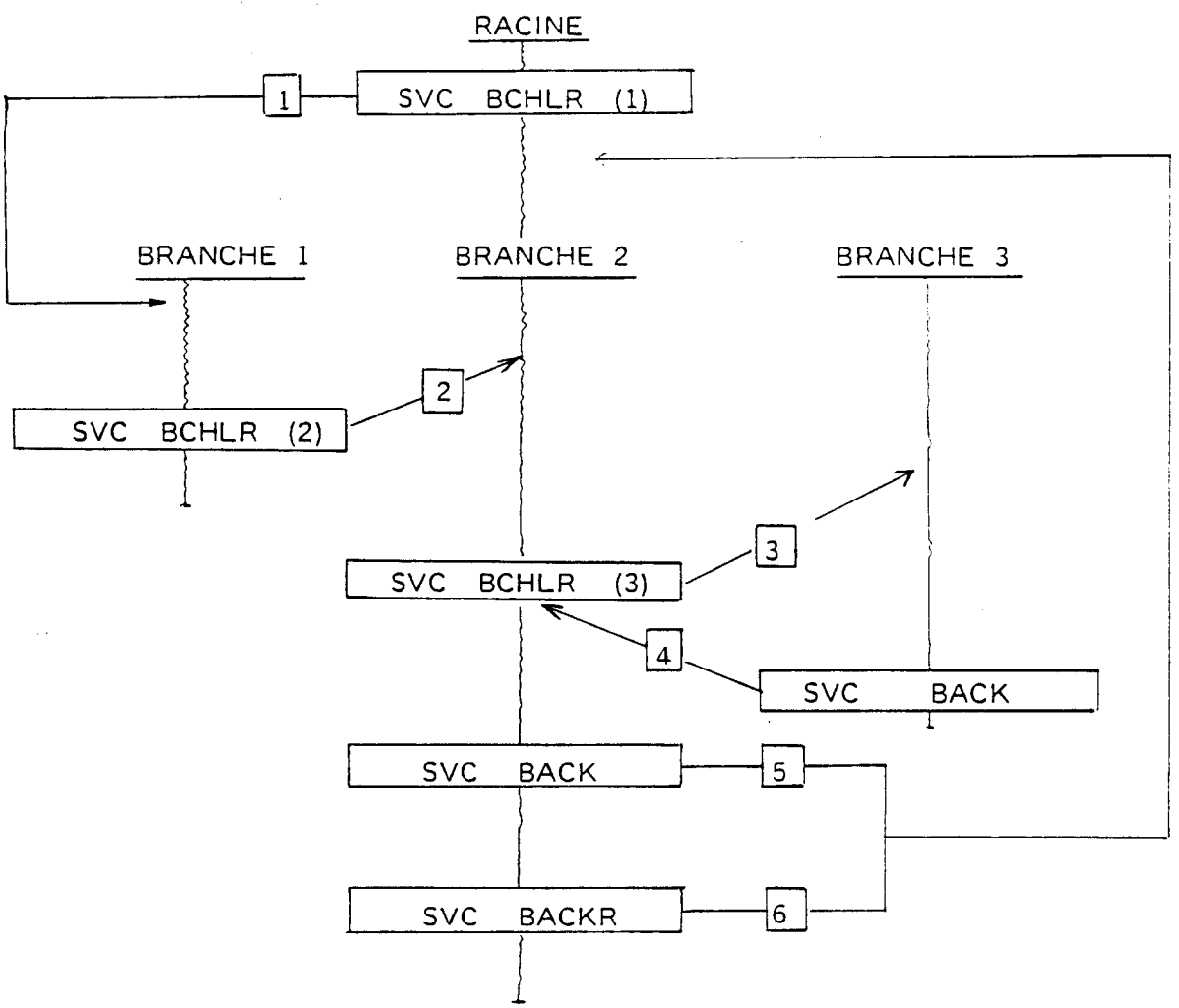
- chargement de la branche
- lancement à une adresse fixe (cas de la première activation d'une branche)

Les requêtes programmées disponibles pour la gestion et le déroulement d'une tâche en "overlay" sont au nombre de 3 :

- chargement et lancement d'une branche (requête BCHLR) depuis la racine ou depuis une branche.
- retour à la racine (requête BACKR)
- retour à l'appelant, racine ou branche (requête BACK)

Remarque : Le retour d'une branche appelée à la branche appelante se fait à un seul niveau.

Exemple :



1 chargement et lancement BRANCHE 1

2 chargement et lancement BRANCHE 2 ; BRANCHE 1 est écrasée

3 chargement et lancement BRANCHE 3 ; BRANCHE 2 est écrasée

4 chargement et relancement BRANCHE 2 après appel BRANCHE 3

5 le retour à la branche appelante se fait à un seul niveau : ici, BACK est exécuté comme BACKR

6 retour à la racine après le dernier appel d'une branche depuis la racine

2.4.6 - Communications entre tâches :

Le fonctionnement asynchrone des tâches d'une part, le fait qu'elles résident en mémoire centrale ou sur disque d'autre part, le fait enfin que la notion de protection intertâche leur impose des espaces de travail en mémoire disjoints, rendent indispensable l'existence de moyens souples et puissants d'intercommunication.

RTES16 présente divers outils complémentaires pour répondre à cet objectif.

2.4.6.1 - Les fichiers de l'utilisateur :

Le lecteur trouvera dans le Manuel de Référence de FMS les requêtes programmées relatives à l'accès aux fichiers.

Avertissement :

Cettenotice de FMS ne décrit pas exactement l'interface tel que l'utilisateur le voit, mais tel que le système le voit. RTES16 analyse et complète éventuellement chaque requête, en particulier il transmet la valeurdes paramètres USR et USRP (n° d'usager privé et public) fournis par l'usager lors de l'écriture d'une tâche...

Rappelons simplement que ce système disponible sous RTES 16 permet de manipuler des informations d'une manière relativement indépendante de l'organisation physique sur le support.

L'organisation de ces informations et les méthodes d'accès se font d'une façon logique en liaison avec la logique du traitement de l'utilisateur.

Les volumes d'information sont personnalisés et identifiés logiquement.

Le système de fichier permet de gérer des informations de caractère partageable ou non partageable, temporaire ou permanent.

Le partage de l'accès à l'information est réalisé à l'aide de clés et d'identificateurs. Le système de fichiers gère automatiquement les conflits d'accès simultanés à l'information.

2.4.6.2 - La zone des données résidentes :

La zone commune des données résidentes constitue un moyen complémentaire du Système de Fichiers pour la communication entre tâches ; en raison de l'encombrement en mémoire centrale qui résulte de cette méthode, il est nettement conseillé de limiter son emploi aux cas :

- où le volume des informations communes n'est pas prohibitif
- où intervient de façon critique la notion de rapidité d'accès à ces informations.

L'organisation de la zone des données résidentes est à la charge de l'usager ; elle résulte de conventions entre les tâches de son application.

Aucune protection n'est réalisée par le système : l'usager doit synchroniser les tâches faisant accès à cette zone pour gérer les informations qu'elle contient.

Pour l'accès à ces données résidentes, les tâches disposent de 2 requêtes :

- REDGET, pour lire dans un buffer une portion de la zone des données résidentes.
- REDPUT, pour transférer un buffer dans une portion de la zone des données résidentes.

Remarque : RTES16 effectue un contrôle de débordement d'accès à cette zone.

2.4.6.3 - La zone de communication : CDA

La CDA permet la communication de données inter-tâche aisée quel que soit le mode de fonctionnement.

Comme pour la ZDR, les tâches utilisatrices de la CDA assurent elles-mêmes leur synchronisation.

Pour l'accès à ces données communes les tâches disposent :

- des instructions RCDA et WCDA
- des entrées-sorties directes en CDA par le moniteur IOCS
- des entrées-sorties directes en CDA par le moniteur FMS
- de l'intégration et du lancement de tâches dans la CDA.

L'emplacement mémoire de la CDA est défini à la configuration du système par l'utilisateur.

Elle est constituée d'une ou plusieurs partitions contigües, ne peut pas dépasser 64 K mots et doit appartenir à la même page de 64 K.

L'utilisation en Fortran des instructions de l'option CDA se fait à travers les sous-programmes de la bibliothèque BI BRTE.

2.4.6.4 - Echanges de blocs d'informations

Cet outil. de communication inter-tâcheest basé sur l'utilisation de l'allocateur mémoire de RTES16.

Les blocs d'informations échangés transitent par le système qui les stocke dans la ZBCT (pavés de 38 mots).

Pour réaliser ces échanges les tâches disposent de 2 requêtes programmées :

SEND pour envoyer un bloc d'informations dans la zone système (38 mots)

RECEIVE pour recevoir un bloc d'informations stocké dans le système (38 mots).

L'utilisation de la ZDR correspond à des besoins différents :

- la gestion de la ZDR est à la charge de l'utilisateur
- les informations de la ZDR sont partageables en lecture et en écriture et sont non protégées.

La gestion des blocs transmis par SEND et RECEIVE est réalisée par le système et les informations d'un bloc ne sont pas partageables (un pavé de la ZBCT est alloué par la requête SEND et désalloué par la requête RECEIVE).

2.4.6.5 - Echanges de blocs d'informations utilisateurs

Des blocs d'informations, de taille configurables à la génération du système, sont à la disposition des utilisateurs. Cette zone est appelée ZWB.
Pour acquérir ou libérer un de ces blocs, il existe une requête : GESPAV.
Le blocage sur attente de libération de bloc est optionnel.

2.4.6.6 - Ressources de l'usager :

La ressource représente le moyen utilisé par la tâche ; elle permet la coexistence et l'exécution parallèle des tâches.

On distingue :

- les ressources "hardware" qui appartiennent au système et dont il vérifie l'utilisation afin d'éviter tout blocage du système.

Exemple : les périphériques (ressources discrètes)
la mémoire (ressource continue)

- les ressources "software"
c'est l'ensemble :

. des services de base du moniteur accessibles par requêtes programmées (événements, horloges software...)

des ressources créées par l'usager, qui représentent ce qu'il désire, dont l'utilisation est laissée à son entière responsabilité, et dont il se sert pour résoudre ses problèmes d'exclusion entre tâches.

Dans RTES16, l'usager dispose de trois requêtes programmées pour créer des ressources software (en donnant leur nom et le nombre d'accès simultanés autorisés), demander un accès à ces ressources, et libérer un accès à ces ressources.

Ce sont les requêtes :

- RESDEF pour créer (définir) une ressource
- RRQST pour accéder à une ressource
- RRLSE pour libérer un accès à une ressource.

Ces deux dernières requêtes permettent aux tâches utilisateur en mode esclave de réaliser les instructions de même nom-avec un maximum de sécurité.

2.4.6.7 - Paramètre de travail et compte rendu de traitement :

Au moment de l'appel d'une tâche, la tâche appelante ou l'opérateur (tâche de dialogue opérateur) peuvent préciser la nature du travail à réaliser à l'aide d'un paramètre de travail (un mot de 16 bits).

Ce paramètre peut- ne pas être précisé : cette absence lui confère alors la valeur nulle.

Réciproquement, la tâche appelée peut renvoyer un compte rendu de traitement à l'appelant, sous la même forme d'un mot de 16 bits. Cette notion de compte rendu est liée à celle d'appel avec attente ; elle apparaît dans les requêtes d'appel de tâche STARTW et TRNONW exposées ci-après.

2.4.6.7 - Evénements :

Un événement est une donnée qui correspond à "quelque chose que l'on peut attendre", "quelque chose qui peut arriver". Il est sans mémoire et ne garde pas trace de ses différentes apparitions.

Plusieurs tâches peuvent attendre l'arrivée d'un même événement : par définition, un événement est partageable.
Un événement a deux états :

- l'état SET ; un événement est à l'état SET quand il est arrivé
- l'état RESET ; un événement est à l'état RESET quand il a disparu.

RTES16 gère 256 événements (numéros 0 à 255) ;
Ces événements sont scindés en 8 classes disjointes de 32 événements chacune

Classe	Numéros d'événements
0	0 à 31
1	32 à 63
2	64 à 95
7	224 à 255

L'interprétation que fait l'usager des 32 événements d'une classe est liée à son application, et dépend des opérations qu'il désire réaliser sur une classe d'événements.

Pour utiliser les événements, on dispose de 8 requêtes programmées ; deux d'entr'elles font référence à une classe d'événements ; les autres font référence à un événement ; ce sont :

- WEVENT pour attendre qu'un événement soit dans l'état SET
- WEVAND pour attendre que certains événements spécifiés d'une même classe soient simultanément dans l'état SET
- WEVOR pour attendre que l'un des événements d'une liste spécifiée d'événements d'une même classe soit dans l'état SET
- SEVENT pour signaler l'arrivée d'un événement (le mettre à l'état SET)
- SEVDEL pour signaler l'arrivée différée d'un événement (le mettre à l'état SET après un délai donné)
- OFFDEL pour supprimer l'arrivée différée d'événement demandée par SEVDEL
- REVENT pour mettre un événement dans l'état RESET
- TEVENT pour tester l'état d'un événement.

2.4.6.8 - Phase d'avancement

La phase d'avancement est un outil d'intercommunication entre tâches utilisateur ; elle permet de savoir dans quelle phase logique de son déroulement se trouve une tâche.

La phase d'avancement consiste en un numéro compris entre 0 et 255 ; par une requête programmée, une tâche précise au système, à chaque franchissement de points stratégiques, quelle est sa phase d'avancement.

Par une seconde requête programmée, une autre tâche peut savoir quelle est la phase d'avancement d'une tâche dont le numéro est spécifié en paramètre.

La phase d'avancement a une signification spécifique de chaque tâche qui résulte d'une convention entre deux ou plusieurs tâches utilisateur.

Cette notion est par exemple relativement intéressante pour un redémarrage après disparition du secteur : après avoir effectué un certain nombre d'actions standard, RTES16 active une tâche usager spécialisée qui peut prendre des décisions en fonction de la phase d'avancement de chaque tâche du système.

Les requêtes permettant l'accès à la phase d'avancement sont :

- STADEF : la tâche appelante définit sa phase d'avancement
- STAGET : La tâche appelante demande quelle est la phase d'avancement d'une tâche précisée en paramètre.

2.5 - BACKGROUND DE RTES16

2.5.1 - Caractéristiques générales

La fonction background de RTES16 est réalisée par un moniteur présentant les services et la souplesse du système d'exploitation BOS16 et d'une utilisation compatible.

Cette fonction permet en particulier :

- d'utiliser simplement tous les processeurs du software de base dont :
  - les compilateurs FORTRAN, PL1600
  - le macroprocesseur MACP
  - l'assembleur ASM
  - l'éditeur de liens EDILE
  - l'éditeur de texte EDIT16
  - le chargeur disque BUILDER
  - des utilitaires de manipulation de fichiers
- d'exploiter les programmes produits indifféremment par un des systèmes de la série BOS ou en Background,
- d'utiliser le système d'entrées-sorties- IOCS et le système de fichiers FMS intégrés à RTES 16.
- d'exploiter des programmes en mode train de travaux ou conversationnel, ces programmes pouvant être structurés en overlay, c'est à dire ayant une taille supérieure à la taille mémoire occupée.
- d'utiliser les bibliothèques système ou utilisateur.
- de produire sur disque des taches ultérieurement intégrées sous RTES16, c'est à dire de faire évoluer une application par adjonction ou modification de tâches en background.

La fonction background est optionnelle ; son utilisation nécessite un calculateur dont la taille mémoire minimale est de 32 K mots.
La console utilisée pour le dialogue avec le moniteur background peut être la console système ou une console spécialisée.

Ce chapitre présente les concepts et éléments "système" relatifs au background : il ne reprend pas la description du langage de contrôle ou des requêtes que l'usager trouvera dans le' Manuel de Référence de BOS 16.
Certains points spécifiques du moniteur background font, néanmoins l'objet d'un chapitre du manuel d'utilisation de RTES16.

2.5.2 - Définitions

2.5.2.1 - Moniteur background

Le moniteur background BACKM est une tâche de RTES16 de faible priorité qui occupe pendant toute la durée de l'activité background une partition de taille d'au moins 2K mots.

Le numéro de la partition est paramétrable à la configuration du système. Elle doit se situer en dessous de 32K.

Les requêtes programmées de BACKM sont celles du système BOS16.

Les requêtes programmées spécifiques du temps réel (START, WEVOR ...) et issues des processeurs exécutés en background sont ineffectives (ignorées) en standard.

En règle générale, les commandes et leur syntaxe sont celles du système BOS16

Certaines commandes relatives à la configuration de système (CONF, ZCONF, INIT) n'existent pas en background.

2.5.2.2 - Processeur Background

Un processeur background est un programme exploité sous le moniteur BACKM.

C'est soit un processeur du software de base (assembleur, compilateur, éditeur de texte...) soit un programme utilisateur.

En règle générale, un processeur background s'exécute en mode esclave : le moniteur RTES16, les tâches temps réel et le moniteur BACKM sont ainsi protégés vis à vis des programmes en cours de mise au point en background.

Un processeur est activé par les commandes CALL. ou RUN ; il est alors chargé en mémoire et occupe une ou plusieurs partitions consécutives.

Ces partitions sont de type non résident et sont récupérées par le mécanisme de swap au profit des tâches temps réel qui requièrent l'une d'elles : le processeur est interrompu, la zone' mémoire qu'il occupe est écrite sur disque dans la zone de swap, et il est ramené en mémoire et relancé après exécution de la tâche dans la partition allouée.

D'une façon plus générale, plusieurs tâches peuvent s'exécuter dans les partitions du processeur background après un "swap-out" ; il ne sera rechargé en mémoire et relancé que lorsque toutes les partitions qu'il utilise sont à nouveau disponibles.

2.5.2.3 - Vacation background

Sous RTES16 une vacation background correspond à une phase d'activité du moniteur ou des processeurs background en parallèle avec l'exploitation en temps réel de l'installation.

Après l'intégration initiale du moniteur background. une vacation débute par émission sur le dispositif de dialogue foreground (unité fonctionnelle DI, le téléimprimeur en général) de la commande :

BACK, paramètres

d'activation du moniteur BACKM, et se termine par émission, sur le dispositif de dialogue de l'opérateur background. de la commande :

EBOS

2.5.3 - Gestion de la mémoire

2.5.3.1 - Principe :

Le système de partitionnement de la mémoire centrale a été étendu et adapté à la fonction background compte tenu des critères suivants :

- la place occupée en mémoire par le moniteur BACKM en cours de vacation est récupérée en fin de vacation (si la partition est de type non résident) ; la partition de BACKM est alors accessible aux tâches temps réel.

- la place occupée en mémoire par les processeurs du background recouvre une ou plusieurs partitions, le nombre de ces partitions pouvant évoluer dynamiquement d'un job à un autre en fonction en paramètres précisé dans la commande JOB, par défaut, l'espace alloué est de 4 K mots ; pour une taille donnée, le nombre de partitions occupées par les processeurs du background dépend de la taille de ces partitions.

- l'activité background ne répondant pas à la condition de terminaison dans un temps donné imposée aux autres tâches du système, les partitions allouées pour un job sont récupérées par le mécanisme de swap au bénéfice des tâches temps réel.

Au lancement d'une vacation, l'opérateur foreground précise le numéro de la première partition utilisée par les processeurs background pour cette vacation :

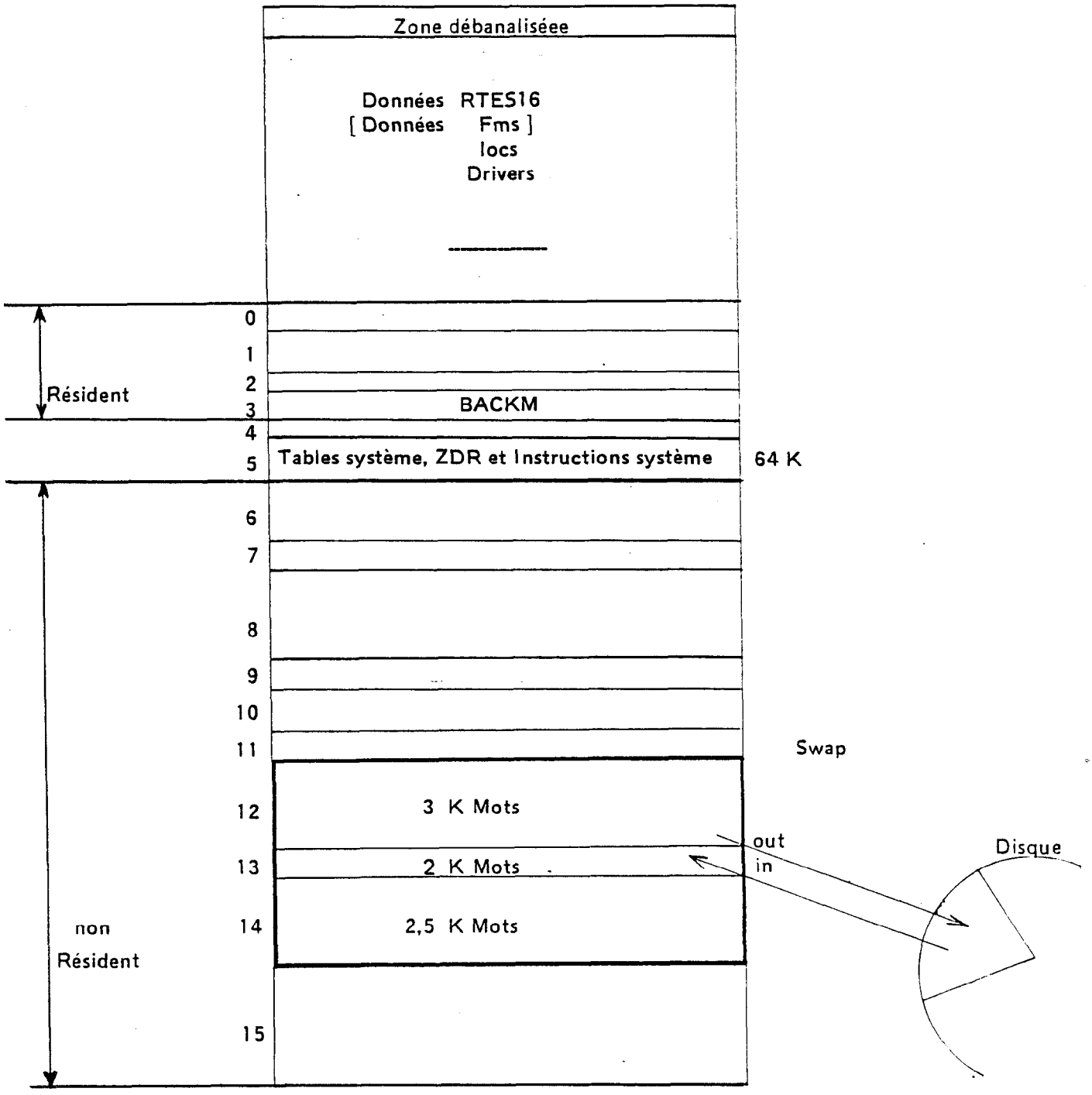
EXEMPLE :

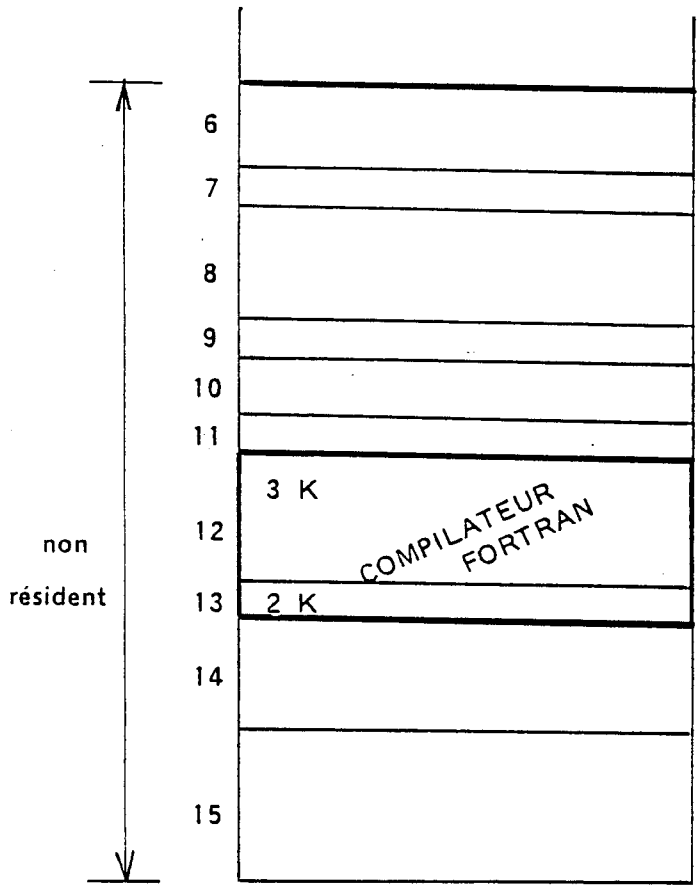
BACK, 12

Pour cette vacation, l'espace mémoire des processeurs background commence au début de la partition de numéro 12.

On a donné dans les pages suivantes un exemple de carte mémoire et d'utilisation des partitions background pendant une vacation au cours de laquelle s'enchaînent un job de production de programme FORTRAN et un job d'exploitation de ce programme.

Carte mémoire





Exemple 1 :

. Job de production de programmes

. Steps :

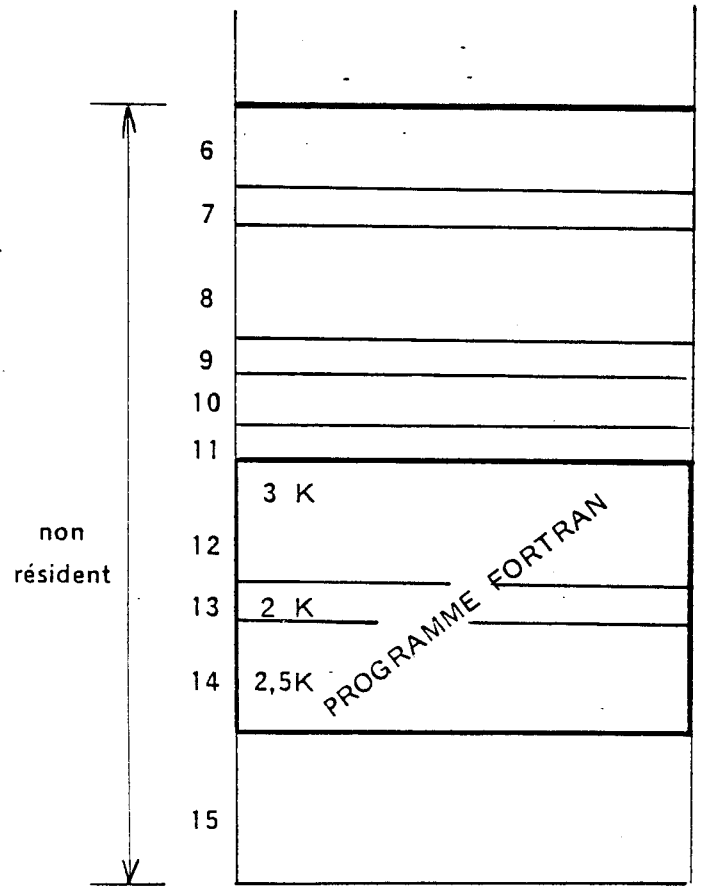
— compilateur fortran
— éditeur de liens
— builder

. Taille requise

5 K Mots

. Partitions processeurs *

12, 13.



Exemple 2 :

. Job d'exploitation

. Steps :

— exécution de programmes fortran

. Taille requise

7,5 K Mots

. Partitions processeurs *

12, 13, 14.

* L'ensemble de ces partitions doit appartenir à la même page de 64 K.

2.5.3.2 - Commande JOB du moniteur background

Cette commande, par ailleurs analogue à son homonyme dans le système BOS16, comporte un paramètre (optionnel) supplémentaire en background : la taille mémoire requise pour l'exécution du prochain JOB (par défaut, 4 K mots). La valeur maximale autorisée est 64 K mots.

Exemple :

JOB FTXAB, PX, D2,,7

7 K mots sont requis pour l'exécution du job FTXAB.

Cette indication de taille mémoire est valable jusqu'à la prochaine carte JOB.

REMARQUE : II n'y a pas en background de contrôle de temps sur l'exécution d'un job ; le paramètre associé est ignoré en background dans la commande JOB.

2.5.4 - Gestion des périphériques

2.5.4.1 - Principe :

Le fonctionnement normal d'une installation exploitée sous RTES16 suppose, en cours de vacation background. l'exclusion d'accès entre les tâches temps réel et le background, aux périphériques non partageables tels que :

- lecteur, perforateur de ruban
- lecteur de cartes et imprimante rapide
- dérouleur de bande magnétique
- table traçante
- téléimprimeur
- dispositif de visualisation alpha-numérique

Cette règle stricte peut néanmoins être modulée, par exemple pour l'imprimante, en tenant compte des données suivantes :

. les processeurs background de production de programme ne s'attachent pas de périphérique en cours de vacation.

. les tâches temps réel susceptibles d'utiliser l'imprimante en parallèle avec le background réaliseront dans l'ordre les fonctions d'attachement, de saut page, d'impression, de saut page et de détachement, puis le SVC (EXIT) de fin de tâche.

2.5.4.2 - Moyens de gestion

Lors du lancement d'une vacation par la commande BACK, le background a implicitement accès :

- aux unités fonctionnelles associées à la console background, c'est à dire TS, TK simulés respectivement par les unités fonctionnelles F7, F8 de l'installation.

- en lecture seulement aux unités fonctionnelles disque D2 à D8 (ou au sous-ensemble de ces FU qui a été effectivement configuré).

- aux unités fonctionnelles de numéros supérieurs à '19 (F5) en lecture et en écriture.

2.5.4.2.1 - Commande GIVE

La commande :

GIVE, FU1, FU2, FU3

de RTES16 permet à l'opérateur système de donner au background l'autorisation d'accès à d'autres FU de l'installation.

Exemple :

GIVE, CR, LP, T1

Cas de FU disque

Pour les FU disque D1 à D8, implicitement accessibles en lecture, cette commande autorise l'accès en écriture aux FU spécifiées :

Exemple :

GIVE, D3, D4

Autres cas

Enfin, cette même commande permet d'autoriser au background l'accès à une unité fonctionnelle de l'installation, en tant qu'unité fonctionnelle type du background ;

Exemple :

Etant donné une installation comprenant les unités fonctionnelles

LP imprimante
F2 seconde imprimante

La commande :

GIVE, LPF2

spécifie que l'unité fonctionnelle F2 est accessible au background en tant qu'unité fonctionnelle LP.
Ainsi par exemple, après la commande LO LP la liste EDITEX sera imprimée sur F2.

Le tableau des affectations possibles des FU "Foreground" aux FU "Background" est donné en annexe.

2.5.4.2.2 - Commande TAKE

La commande symétrique de GIVE

TAKE, FU1, FU2...

supprime l'autorisation d'accès aux unités fonctionnelles spécifiées (ou uniquement l'autorisation d'accès en écriture pour les unités fonctionnelles disque).

Exemple : TAKE, LP, D3

Remarque : Les unités fonctionnelles dont le numéro est supérieur à 25 (19), et en particulier l'unité fonctionnelle ME de numéro 127 (7F) sont en standard accessibles aux processeurs du background sans utilisation préalable de la commande GIVE.

2.5.4.3 - Défauts de périphériques

En cours de vacation background, un partitionnement des périphériques est implicitement établi :

- console background et périphériques autres que le (s) disque (s) dont une unité fonctionnelle est rendue accessible au background par la commande : GIVE, FU.

Les défauts de tels périphériques sont signalés par le moniteur background : le traitement réalisé est analogue à celui du système BOS16 (message ERB 13. abandon du Job en cours).

- autres périphériques (dont le (s) disque (s))

Les défauts de ces périphériques sont pris en compte par RTES16 et signalés à l'opérateur système.

2.6 - PROCESSEUR FOREGROUND

2.6.1 - Principe

Un processeur foreground de RTES16 est un programme d'intérêt général pour l'application temps réel ; c'est soit un programme du software de base, soit un programme spécifique de l'application.

Exemple de processeurs du software de base :

- le flottant programmé peut être utilisé par plusieurs tâches de l'application ; il doit alors être intégré en tant que processeur foreground dans une partition de RTES16.

Exemples de processeurs spécifiques :

Cette notion de processeur a un double intérêt :

- Adjonction de nouvelles commandes opérateur. Un tel processeur peut correspondre à un programme spécifique directement activable par des commandes spéciales qu'il fait connaître au superviseur dans sa phase d'initialisation par la requête SVC CAMO.
On trouvera par exemple des commandes de positionnement de périphériques, des commandes d'initialisation de la zone des données résidentes...

- Adjonction de nouveaux sous-programmes superviseur.
Des sous-programmes spécifiques appelés par plusieurs taches de l'application peuvent être intégrés sous RTES16 en tant qu'extension des requêtes programmees standard ; ils sont déclarés au système par la SVC NEWS et sont appelables par l'instruction SVC.
Ils doivent être réentrants.

2.6.2 - Règles

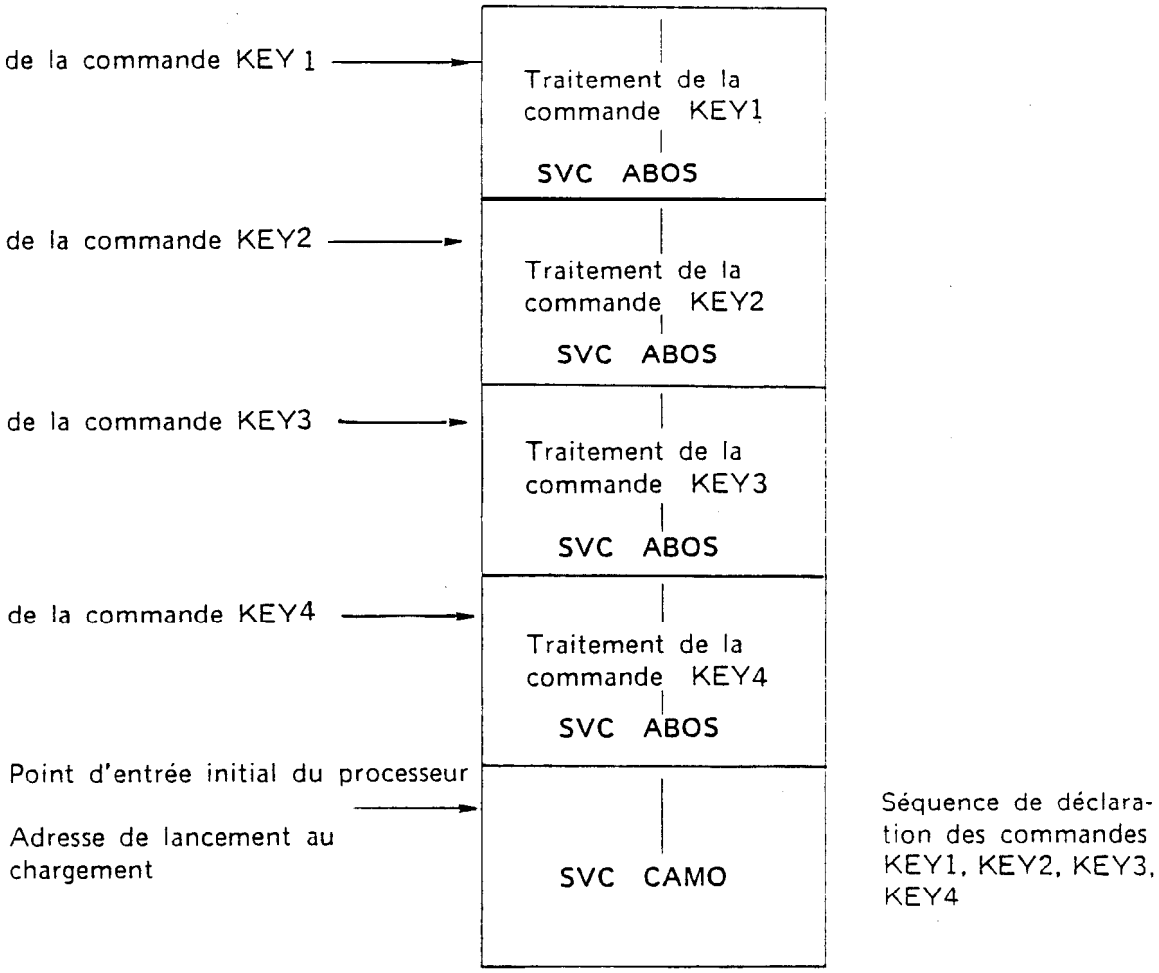
Un processeur foreground est un programme buildé en mode maître, sans structure d'overlay, intégré dans une partition de type résident, et ne contenant pas de déclaration de PST.

La commande d'intégration on-line d'un processeur (commande EXEC) réalise le chargement dans une partition et le lancement du processeur avec une priorité réservée au système (celle de la tâche de dialogue opérateur)

2.6.3 - Structure d'un processeur foreground

a)- Processeur extension du dialogue opérateur

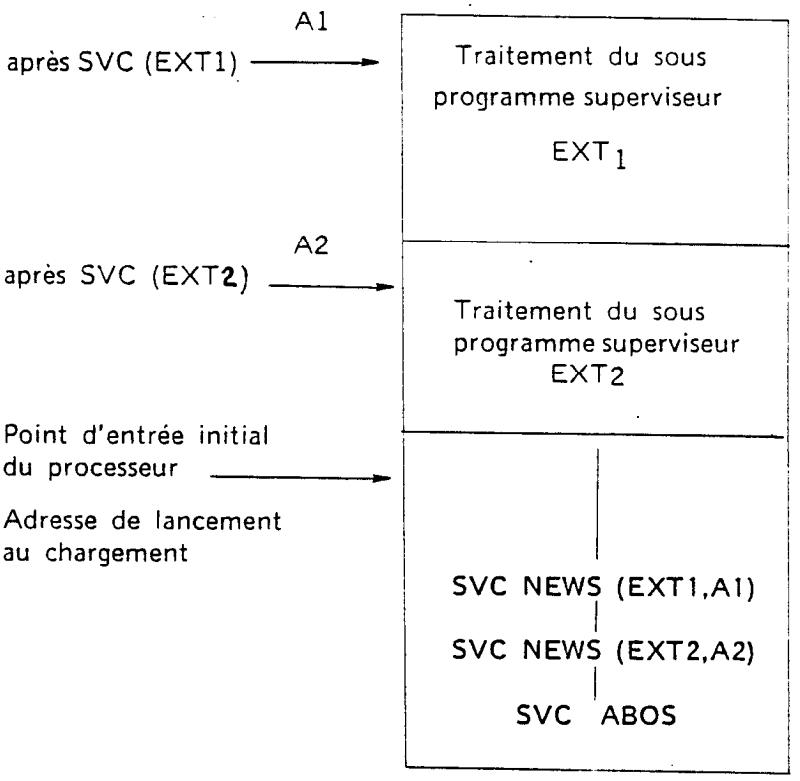
Adresse de branchement au
processeur après émission :



On trouvera plus loin une description précise des conditions d'utilisation de la SVC CAMO.

b)- Processeur extension des requêtes programmées

Adresse de branchement au processeur



Séquence de déclaration des requêtes EXT1 et EXT2

On trouvera dans le Manuel de Référence du système BOS16 (paragraphe 3.3.9) une description précise des conditions d'écriture d'un sous-programme superviseur.

Remarque : Un même processeur peut contenir à la fois une extension des requêtes programmées et une extension des commandes du dialogue opérateur.

3 - DESCRIPTION

Ce chapitre présente une description des services à caractéristique temps réel rendus à l'utilisateur du système RTES16.

II comprend les sections suivantes :

- requêtes programmées
- commandes opérateur
- détection des erreurs.

3.1 - LES REQUETES PROGRAMMEES

3.1.1 - Définition

Une tâche utilisateur demande des services au système par l'intermédiaire de requêtes programmées. Ces requêtes sont des appels à des sous-programmes superviseur assemblés ou compilés dans le programme utilisateur et interprétés par le système au moment de l'exécution.

Elles sont constituées d'une séquence d'instructions et en règle générale d'une table de paramètres qui spécifient. l'opération demandée.

3.1.2 - Syntaxe des requêtes

Les requêtes du système RTES16 (à l'exception des requêtes à IOCS et à FMS que le lecteur trouvera décrites dans les Manuels de Référence de ces produits) sont présentées sous leur syntaxe FORTRAN et ASSEMBLEUR ASM 16 ; à l'aide de règles simples, le lecteur fera la correspondance entre ces deux syntaxes d'appel des requêtes et la syntaxe d'appel dans le langage PL16.

3.1.2.1 - Syntaxe FORTRAN

Elle se présente sous la forme suivante:

CALL RNAME (P1, P2. P3,..., Pn)
où RNAME est un nom de sous-programme externe faisant partie d'une bibliothèque système et intégré au programme utilisateur par édition de liens ;

P1, P2, P3,..., Pn sont des variables ou constantes entières ou des tableaux entiers constituant la liste des paramètres de la requête (s'il y en a).

Exemple :

```
CALL WAIT (ID, IUD, IERR)
```

Les requêtes proposées dans le projet de norme FORTRAN Temps Réel (communication de l'Instrument Society of America ISA-S61) sont traitées par RTES16 ; ce sont par exemple :

- CALL START (i, j, k, m)
- CALL TRNON (i, j, k,)

CALL WAIT (i, j, k,)

3.1.2.2 - Syntaxe ASSEMBLEUR :

La forme générale d'appel d'une requête est la suivante :

```
LAD     RPB   < A : =   adresse Request Parameters Block

SVC     RNAM
```

où RPB est l'adresse de la table des paramètres de la requête (RPB : Request Parameters Block) et RNAM est un symbole valeur désignant la requête sollicitée.

Lorsque la requête n'a aucun paramètre, l'instruction d'appel SVC suffit.

3.1.2.3 - Syntaxe PL 16

Selon le type de programmation adoptée dans ce langage, la syntaxe de la requête dérive de la syntaxe FORTRAN ou de la syntaxe ASSEMBLEUR.

Règles :

1 Une tâche écrite en PL16 peut appeler une requête avec la syntaxe FORTRAN, en utilisant le même sous-programme bibliothèque, à condition :

. que les divers paramètres de l'appel soient des pointeurs (les paramètres passés en FORTRAN étant des adresses)

. que le nombre de paramètres de l'appel soit chargé dans le registre accumulateur (ce chargement étant généré implicitement par le compilateur FORTRAN)

Exemple :

. Appel FORTRAN CALL RUN (NT, IERR, IPAR)
. Appel PL 16 CALL RUN (@NT,@IERR,@IPAR, RA:= 3)

2 Une tâche écrite en PL 16 peut appeler une requête avec une syntaxe voisine de la syntaxe assembleur à condition d'avoir auparavant déclaré l'instruction SVC ; le contenu de la table des paramètres est alors le même qu'en assembleur.

Exemple :

Supposons que la requête de numéro '42 soit la requête TRNON.

```
INSTRUCTION SVC (3,'1C00) ; <<déclaration de l'instruction SVC
CONSTANT TRNON = '42 ;      <<déclaration de la constante TRNON
RA: = @RPB (0) ;            <<A: = pointeur des paramètres
SVC (TRNON) ;               «appel de requête
```

3.1.3 - Paramètres d'une requête :

Ils spécifient les diverses entités du travail demandé à la requête. En règle générale, chaque requête travaille sur des paramètres spécifiés au moment de l'appel.
En assembleur, le registre accumulateur est interprété comme adresse de la table des paramètres ; dans les rares cas où la requête n'a besoin que d'un paramètre (EXIT par exemple), l'accumulateur contient alors la valeur de ce paramètre.
Les paramètres du Request Parameters Block (RPB) sont eux-mêmes soit des adresses, soit des valeurs.

Toutes les adresses fournies en paramètre d'appel à une requête sont considérées comme des adresses relatives à la tâche appelante.

y compris l'adresse de RPB fournie dans l'accumulateur. Ainsi, en cours de traitement, les requêtes vérifient la vraisemblance de telles adresses : une tâche en mode esclave doit fournir des adresses inclues dans la partition où elle s'exécute.

Remarque :
Certaines requêtes ont des paramètres optionnels ; ces paramètres figurent entre crochets dans la syntaxe FORTRAN de l'appel.

3.1.4 - Exécution d'une requête :

Le traitement est fait, en tout ou partie, avec la priorité de la tâche appelante.
Pour les traitements complexes, une partie de ce traitement est réalisé :
- soit sur un niveau hardware (cas de requêtes sur date, délai ou période par exemple)
- soit dans une tâche software système (cas nécessitant un ((complément de scheduling)))

3.1.5 - Compte rendu d'une requête :

II y a deux issues possibles au traitement réalisé par une requête.
- La requête est acceptée, le travail demandé a été exécuté ; dans ce cas, le compte rendu de requête (dont l'adresse se trouve précisée dans le mot 1 du RPB) est égal à 1. L'appelant reprend le contrôle en séquence après l'appel dans tous les cas sauf pour les 3 requêtes BCHLR, BACK et BACKR.

- La requête est refusée par le système. Le travail demandé n'a pas été exécuté. La tâche appelante est alors arrêtée en cours d'exécution de cette requête ; un message est imprimé sur le périphérique associé à l'unité symbolique TE (Task Errors) afin d'informer l'opérateur de cette anomalie.

Après prise en compte de ce message, l'opérateur pourra relancer la tâche erronée : elle se poursuivra alors toujours en séquence après l'appel et devra impérativement utiliser le compte rendu de requête qui lui est renvoyé.

Dans le cas d'un refus de requête, ce compte rendu a une valeur supérieure ou égale à 2 et sera en règle générale interprété de la façon suivante :

Valeur du compte rendu	Signification
2	L'entité spécifiée n'existe pas
3	La requête est illogique ou redondante
4	L'appelant n'a pas accès à cette requête
5	La requête est incompatible avec l'état actuel de l'entité sur laquelle elle travaille
6	II existe au moins un paramètre incorrect
7	On ne sait pas enregistrer la requête : le système est sous-dimensionné
8	Requête non traitée par RTES/D ou Valeur incorrecte du registre K (mode esclave).

Le lecteur trouvera, dans l'exposé ci-après de chaque requête, une interprétation plus spécifique de ce compte rendu.

> Ce compte rendu de requête, fourni à l'adresse précisée dans le mot 1 du RPB, est également fourni dans le registre accumulateur (ASSEMBLEUR) en retour de requête.

Ceci afin d'en faciliter le traitement immédiat d'une part, et d'autre part pour pallier au cas ou l'adresse de compte rendu précisée est elle-même erronée.
A l'exception des requêtes relatives à IOCS (IOCS. WEIO) et FMS (FMS, FMSS, FMSI, FMSD), le compte rendu de toutes les requêtes standard de RTES16 répond à ce schéma logique simple pour chaque requête décrite dans le manuel de référence, sa signification est donnée plus explicitement.
Pour les requêtes IOCS et WEIO, le lecteur se reportera au manuel de référence de IOCS.

Sous RTES16 les erreurs logiques détectées par FMS, supérieures ou égales à 32 ('20) sont considérées comme fatales pour les tâches de l'application dont le numéro d'usager USR est inférieur à 128 (suspension de l'appelant, impression d'un message ERR sur la SU TE...).
Pour les numéros USR supérieurs à 128, le système ne suspend pas la tâche en erreur : elle est chargée d'analyser et de traiter tous les compte rendus de FMS.
Selon les caractéristiques de l'application, l'utilisateur prendra en compte ce partitionnement des numéros USR en fonction des degrés de sévérité des erreurs qu'il désire traiter.

Le numéro USR de RTES16 est 128, les tâches de l'application ne doivent pas utiliser ce numéro.
Le numéro USR du BACKGROUND de RTES16 est 129, les tâches de l'application ne doivent pas utiliser ce numéro.

3.1.6 - Fonctions réalisées :

Le lecteur trouvera dans les paragraphes suivants les requêtes spécifiques du système RTES16
- ACTIVATION, TEMPORISATION, FIN DE TACHES
- INHIBITION DE TACHES
- OVERLAY
- EVENEMENTS
- DONNEES RESIDENTES
- RESSOURCES DE L'USAGER
- PHASE D'AVANCEMENT
- ACQUISITION DE LA DATE ET DE L'HEURE.

Pour chaque requête, la présentation est la suivante

- N O M
- B U T
- SYNTAXE FORTRAN
- SYNTAXE ASSEMBLEUR
- ERREURS
- COMMENTAIRES.

D'autres requêtes, présentées dans les Manuels de Référence de IOCS ou de FMS, sont également accessibles aux tâches de l'application ; leur description n'est pas reproduite dans ce Manuel.
Le lecteur est invité à se reporter au manuel de référence BIBIND pour les requêtes spécifiques aux entrées-sorties industrielles.
Enfin, les requêtes CAMO, TAPS, ABOS, EMAD, RBOS, NEWS, TEST et AFSU des systèmes de la série BOS sont traitées par RTES16. Certaines ont fait l'objet d'une adéquation aux fonctions de multiprogrammation et de temps réel.

CAMO : Cette requête permet de communiquer au superviseur une table contenant les commandes associées à un processeur.
Sous RTES16, elle est accessible aux processeurs foreground ; elle est refusée aux tâches de l'application (compte rendu d'erreur dans l'accumulateur en retour de requête : RA = 4, l'appelant n'a pas accès).
En standard, RTES16 accepte au maximum 9 tables de clés de processeurs ; en cas de saturation, l'intégration du processeur extension du dialogue provoque l'impression du message ERC 07 (système sous dimensionné).

TAPS : Cette requête réalise la scrutation de la table des mots d'état de périphériques, et, en cas d'apparition de défaut, active la tâche système d'impression des alarmes :
ERS 02 Appel 4 coups
ERS 03 Appel 1 coup
ERS 04 Défaut de périphérique.
Elle est ignorée sous niveau hardware
Elle ne provoque jamais de suspension de la tâche appelante.

ABOS : Exécutée par un processeur, cette requête rend le contrôle au système.
Exécutée par une tâche de l'application, elle est strictement identique à EXIT.

EMAD : Refusée aux tâches en mode esclave.

RBOS : Exécutée par un processeur, elle rend l'état communiqué par ABOS.
Exécutée par une tâche de l'application, elle est strictement identique à DATAG.

NEWS : Cette requête transmet au système l'adresse d'un nouveau sous-programme superviseur.
Cette adresse est fournie dans le registre RA et le numéro du nouveau sous-programme superviseur dans le registre RY. Ce numéro doit appartenir à la plage :

0 à 139

En retour de la requête NEWS, RTES16 fournit dans le registre RB l'adresse antérieure du sous-programme associé à ce numéro. Cette requête ne doit pas être exécutée dans une tâche en mode esclave ou dans une tâche non résidente.

TEST : Cette requête transmet à l'appelant :

- dans RA : la valeur de la base C de RTES16
- dans RB : la valeur de la base L de RTES16
- dans RY : la valeur de la base C de IOCS
- dans RX : le numéro du système
  23 pour RTES16

AFSU : Cette requête permet à l'appelant de connaître l'affectation d'une SU :

- paramètre d'entrée : RA = N° de la SU
  RY = '15
- paramètre de retour : RA = '8000 : No SU incorrect
  '8001 : RY ≠ '15
  No de la FU affecté à la SU

Activation de tâche (RUN) :

RUN

But

Activer une tâche et lui transmettre un paramètre de travail.

Syntaxe Fortran

CALL RUN (NT, IERR [,IPAR])

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à activer.

IERR : Variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

IPAR : Variable ou constante entière qui représente un paramètre de travail transmis à la tâche appelée (par défaut, IPAR = 0)

Exemple : CALL RUN (SCRUT, IRESP, JOBT)

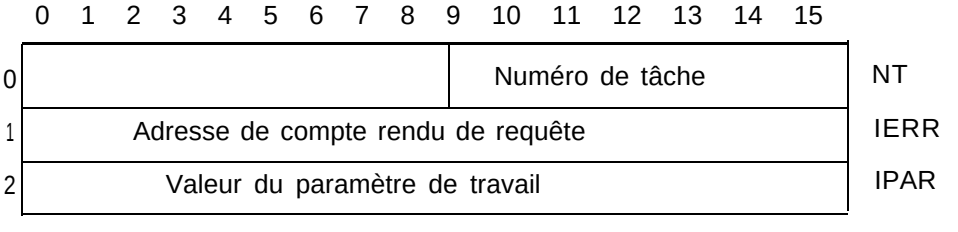
> Activer la tâche SCRUT et lui transmettre le paramètre de travail JOBT. Le compte-rendu de la requête est à placer en IRESP.

Syntaxe Assembleur

Format de la requête

```
LAD  RPB  <  A  :  = adresse de la table des paramètres
SVC  RUN
-------  <  Première  instruction  en  retour
```

Table des paramètres



Exemple : TBAPP : WORD 50 < Numéro de tâche
WORD CONRDU < Adresse compte rendu
WORD '1F < Valeur paramètre

```
LAD  TBAPP
SVC   RUN      <  Activation  tâche  50
CPI   1
JG    YAERR  <  Traitement  refus  requête
```

> Activer la tâche de numéro 50 et lui transmettre le paramètre de travail '1F ; le compte rendu de la requête est à placer en CONRDU.

Erreurs

IERR = 2 . La tâche NT est inconnue du système

IERR = 3 . La tâche NT est une tâche hardware

IERR = 4 . Le numéro NT est supérieur au numéro maximum des tâches de l'application

IERR = 5 . Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.

IERR = 6 . Il existe au moins un paramètre incorrect
- NT négatif
- adresse de compte rendu erronée
- adresse de RPB erronée.

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 - Sauf cas d'erreur sanctionnés par un refus de requête, à chaque demande d'activation d'une tâche correspondra son exécution effective : RTES16 gère le cumul des appels de tâches.
Pour chaque tâche, l'amplitude du cumul est limitée à la valeur précisée, par programme, dans sa PST.

2 - L'appel réalisé par RUN est un appel "sans attente", c'est-à-dire que RTES16 ne suspend pas l'appelant jusqu'à la fin d'exécution de l'appelé ; seule intervient la priorité relative des deux tâches dans le déroulement du processus.

3 - Au retour chez l'appelant après exécution de la requête, les variables et/ou la table associée sont disponibles.

4 - L'appel d'une tâche se fait toujours avec transmission d'un paramètre (valeur nulle par défaut).

5 - Une tâche peut s'appeler elle-même.

Activation différée d'une tâche (START) :

START

But

Activer une tâche après un délai initial, et lui transmettre un paramètre de travail.
La tâche peut être activée périodiquement après écoulement du délai initial.

Syntaxe Fortran

CALL START (NT, ID, IUD, IERR [,IPAR[, IP , IUP]])

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à activer.

ID : variable ou constante entière qui représente, en unités spécifiées par IUD, le délai initial d'activation de la tâche.
Si ID =0, l'activation est immédiate.
Le délai initial ne peut excéder 24 heures.

IUD : variable ou constante entière qui précise l'unité du délai
IUD = 0 top de l'horloge de base
IUD = 1 milliseconde
IUD = 2 seconde
IUD = 3 minute

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requëte acceptée
IERR $\geqslant$ 2 requête refusée.

IPAR : variable ou constante entière qui représente un paramètre de travail transmis à la tâche appelée (par défaut, IPAR = 0)

IP : variable ou constante entière qui précise, en unités spécifiées par IUP, la période d'activation de ta tâche.
Si IP= 0, l'activation est unique
La période ne peut pas excéder 24 heures.

IUP : variable ou constante entière qui précise l'unité de la période.
IUP = 0 top de l'horloge de base du système
IUP = 1 milliseconde
IUP = 2 seconde
IUP = 3 minute.

Exemple : CALL START (SCAN, 70, 2, IRESP)

Activer la tâche SCAN dans 70 secondes. Le compte rendu de la requête est à placer en IRESP.

CALL START (CASE, 5, 3, TICKER, YEAR, 10, 2)

Activer la tâche CASE dans 5 minutes, puis toutes les 10 secondes et lui transmettre le paramètre de travail YEAR. Le compte rendu de la requête est à placer en TICKER.

Syntaxe Assembleur

Format de la requête :

```
LAD     RPB
SVC     START
```

Table des paramètres :

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Numéro de tâche	NT
1	Adresse de compte rendu de requête	IERR
2	Valeur du paramètre de travail	IPAR
3	Délai	ID
4	unité	IUD
5	période	IP
6	unité	IUP

Erreurs

IERR = 2 . La tâche NT est inconnue du système

IERR = 3 . La tâche NT est une tâche hardware

IERR = 4 . Le numéro NT est supérieur au numéro maximum des tâches de l'application

IERR = 5 . Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.

IERR = 6 . Il existe au moins un paramètre incorrect
- NT négatif
- adresse du RPB erronée
- adresse de com te rendu erronée
- (IUD, IUP) ∉ |0,3|
- délai ou période>24 heures.

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 Sauf cas d'erreur sanctionnés par un refus de requête, à chaque demande d'activation d'une tâche correspondra son exécution effective : RTES16 gère le cumul des appels de tâches.
Pour chaque tâche, l'amplitude du cumul est limitée à la valeur précisée, par programme, dans sa PST.

2 L'appel réalisé par START est un appel "sans attente", c'est-à-dire que RTES16 ne suspend pas l'appelant jusqu'à la fin d'exécution de l'appelé ; seule intervient la priorité relative des deux tâches dans le déroulement du processus.

3 Au retour chez l'appelant après exécution de la requête. les variables et/ou la table associée sont disponibles.

4 L'appel d'une tâche se fait toujours avec transmission d'un paramètre (valeur nulle par défaut).

5 Une tâche peut s'appeler elle-même.

6 Une tâche peut être activée cycliquement. avec cumul de plusieurs périodes égales ou différentes, à chaque période étant par exemple associé un type de travail à réaliser précisé par le paramètre IPAR.

7 Dans le cas d'une activation cyclique de tâche, si le temps de cycle est inférieur au temps d'exécution, le système n'accumule pas les appels internes correspondants : il n'autorise qu'une entrée au cumul par requête cyclique indépendante.
Au début du cycle i, si le traitement correspondant au cycle i -1 n'est pas terminé, il n'y aura pas d'activation pour le cycle i : un appel est perdu, et cette anomalie est signalée à l'opérateur.
On évite ainsi un "glissement" dans le temps du processus, ainsi qu'un risque de saturation des tables du système.

8 L'appel de tâche n'est immédiat que si ID et IP sont tous deux nuls.

Activation différée d'une tâche et attente de l'appelant (STARTW) :

STARTW

But

Activer une tâche après un délai initial, lui transmettre un paramètre de travail, et être suspendu jusqu'à la fin de l'exécution demandée.

Syntaxe Fortran

CALL STARTW (NT, ID, IUD, IERR, IPAR, ICR)

NT — variable ou constante entière qui précise le numéro d'appel de la tâche à activer.

ID — variable ou constante entière qui représente en unités spécifiées par IUD. le délai initial d'activation de la tâche.
Si ID = 0, l'activation est immédiate.
Le délai initial ne peut pas excéder 24 heures.

IUD — variable ou constante entière qui précise l'unité du délai.
IUD = 0 top de l'horloge de base du système
IUD = 1 milliseconde
IUD = 2 seconde
IUD = 3 minute.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geq$ 2 requête refusée.

IPAR : variable ou constante entière qui représente un paramètre de travail transmis à la tâche appelée (par défaut, IPAR = 0)

ICR — variable entière chargée par le compte rendu d'exécution de la tâche appelée, avant réactivation de l'appelant.

Exemple : CALL STARTW (INVER, 0,0, IERR, 14. CONRDU)

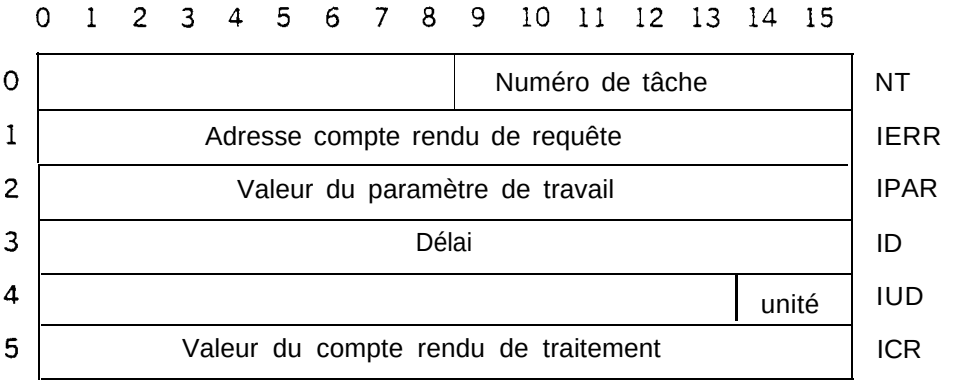
Activer immédiatement la tâche INVER avec le paramètre de travail 14.
Si la requête est acceptée, l'appelant est suspendu jusqu'à la fin de l'exécution demandée ; en retour, IERR est égal à 1 et CONRDU contient le compte rendu de travail demandé.
Si la requête est refusée, IERR est supérieur ou égal à 2 et la tâche INVER n'est pas activée.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     STARTW
```

Table des paramètres :



Erreurs

IERR = 2 . La tâche NT est inconnue du système
IERR = 3 . La tâche NT est une tâche hardware
IERR = 4 . Requête interdite aux tâches hardware
. Le numéro NT est supérieur au numéro maximum des tâches de l'application.
IERR = 5 . Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.
IERR = 6 . Il existe au moins un paramètre incorrect
- NT négatif
- adresse de RPB erronée
- adresse de compte rendu IER R erronée
— IUD ∉ [0,3]
— délai initial > 24 heures.
IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 Sauf cas d'erreur sanctionnés par un refus de requête, à chaque demande d'activation d'une tâche correspondra son exécution effective : RTES16 gère le cumul des appels de tâches.
Pour chaque tâche, l'amplitude du cumul est limitée à la valeur précisée, par programme, dans sa PST.

2 Au retour chez l'appelant après exécution de la requête, les variables et/ou la table associée sont disponibles.

3 L'appel d'une tâche se fait toujours avec transmission d'un paramètre (valeur nulle par défaut).

4 La tâche appelante est suspendue jusqu'à la fin de l'exécution demandée : pendant toute la durée de cette attente, elle reste dans sa partition.
La tâche appelée ne peut pas s'exécuter dans la même partition que la tâche appelante.

5 La notion d'attente introduite avec STARTW est incompatible avec la notion d'activation cyclique de START.

6 Cette requête est refusée aux tâches hardware.

7 Une tâche ne peut pas s'auto-activer par STARTW.

Activation d'une tâche à une heure donnée (TRNON) :

TRNON

But

Activer une tâche à une heure donnée, et lui transmettre un paramètre de travail.
Cette requête permet également l'activation cyclique de la tâche après la première activation à l'heure précisée.

Syntaxe Fortran

CALL TRNON (NT, ITIME, IERR [,IPAR [,IP ,IUP]])

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à activer.

ITIME : tableau entier de 3 mots qui précise l'heure d' (e) (première) activation.
ITIME (1) : heure
ITIME (2) : minute
ITIME (3) : seconde

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

IPAR : variable ou constante entière qui représente un paramètre de travail transmis à la tâche appelée (par défaut, IPAR = 0)

IP : variable ou constante entière qui précise, en unités spécifiées par IUP, la période d'activation de la tâche.
Si IP = 0, l'activation est unique.
La période ne peut pas excéder 24 heures.

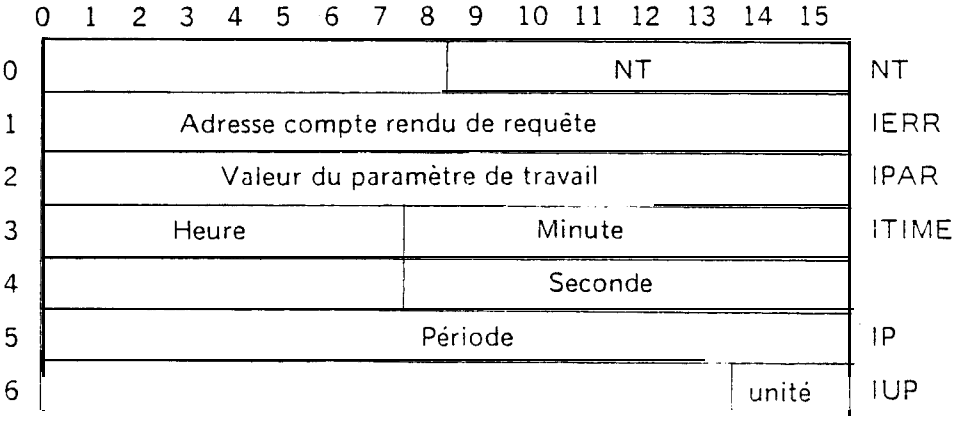
IUP : variable ou constante entière qui précise l'unité de la période
IUP = 0 top de l'horloge de base du système
IUP = 1 milliseconde
IUP = 2 seconde
IUP = 3 minute

Syntaxe Assembleur

Format de la requête :

```
LAD     RPB
SVC     TRNON
```

Table des paramètres



Erreurs

IERR = 2 . La tâche NT est inconnue du système

IERR = 3 . La tâche NT est une tâche hardware

IERR = 4 Le numéro NT est supérieur au numéro maximum des taches de l'application.

IERR = 5 Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.

IERR = 6 II existe au moins un paramètre incorrect.
- NT négatif
- adresse de RPB erronée
- adresse de compte rendu erronée
- IUP ∉ [0,3]
- Période > 24 heures
- heure erronée H ∉ [0,23] ,MN ,SC ∉ [0,59]

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 Sauf cas d'erreur sanctionnés par un refus de requête, à chaque demande d'activation d'une tâche correspondra son exécution effective : RTES16 gère le cumul des appels de tâches.
Pour chaque tâche, l'amplitude du cumul est limitée à la valeur précisée, par programme, dans sa PS%

2 L'appel réalisé par TRNON est un appel "sans attente", c'est-à-dire que RTES16 ne suspend pas l'appelant jusqu'à la fin d'exécution de l'appelé ; seule intervient la priorité relative des deux tâches dans le déroulement du processus.

3 Au retour chez l'appelant après exécution de la requête les variable; et/ou' la table associée sont disponibles.

4 L'appel d'une tâche se fait toujours avec transmission d'un paramètre (valeur nulle par défaut)

5 Une tâche peut s'appeler elle-même.

6 Le tableau de 3 mots ITIME doit préciser une heure "vraisemblable"
ITIME (1) ∈ [0,23]
ITIME (2) ∈ [0,59]
ITIME (3) ∈ [0,59.]

7 Si, à l'instant où la requête est enregistrée, l'heure dans la journée en cours est déjà passée, l'opération demandée sera réalisée, à l'heure précisée, le jour suivant.

Activation d'une tâche à une heure donnée, et attente de l'appelant (TRNOMW) :

TRNONW

But

Activer une tâche à une heure donnée, lui transmettre un paramètre de travail, et être suspendu jusqu'à la fin de l'exécution demandée.
Cette requête est analogue à STARTW, le délai initial étant remplacé par l'heure d'activation.

Syntaxe Fortran

CALL TRNONW (NT, ITIME. IERR, IPAR, ICR)

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à activer.

ITIME : tableau entier de 3 mots qui précise l'heure d'activation.
ITIME (1) : heure
ITIME (2) : minute
ITIME (3) : seconde

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geq$ 2 requête refusée.

IPAR : variable ou constante entière qui représente un paramètre de travail transmis à la tâche appelée (par défaut, IPAR = 0).

ICR : variable entière chargée par le compte rendu de traitement de la tâche appelée avant réactivation de l'appelant.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     TRNONW
```

Table des paramètres

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0										Numéro de tâche							NT
1	Adresse compte rendu de requête																IERR
2	Valeur du paramètre de travail																IPAR
3	Heure								Minute								ITIME
4									Seconde								
5	Valeur du compte rendu de traitement																ICR

Erreurs

IERR = 2 . La tâche NT est inconnue du système
IERR = 3 . La tâche NT est une tâche hardware
IERR = 4 . Requête interdite aux tâches hardware
. Le numéro NT est supérieur au numéro maximum des tâches de l'application.
IERR = 5 . Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.
IERR = 6 . Il existe au moins un paramètre incorrect
— NT négatif
— adresse de RPB erronée
— adresse de compte rendu (IERR ou ICR) erronée
— Heure erronée (H $\in$ [0,23] ,MN, SC $\in$ [0,59])
IERR = 7 . On ne peut enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 Sauf cas d'erreur sanctionnés par un refus de requête. à chaque demande d'activation d'une tâche correspondra son exécution effective : RTES16 gère le cumul des appels de tâches.
Pour chaque tâche, l'amplitude du cumul est limitée à la valeur précisée, par programme, dans sa PST.

2 Au retour chez l'appelant après exécution de la requête, les variables et/ ou la table associée sont disponibles.

3 L'appel d'une tâche se fait toujours avec transmission d'un paramètre (valeur nulle par défaut).

4 L'heure précisée dans ITIME doit être correcte.

5 Si l'heure spécifiée est ultérieure à l'heure actuelle, l'activation aura lieu dans la journée en cours : sinon elle aura lieu le lendemain.

6 Une tâche ne peut pas s'auto-activer par TRNONW.

Suspension d'une tâche (WAIT) :

WAIT

But

Suspendre l'exécution de l'appelant pendant un délai spécifié.

Syntaxe Fortran

CALL WAIT (ID, IUD, IERR)

ID : variable ou constante entière qui représente un délai en unités spécifiées par IUD
Si ID = 0 la requête est ineffective
Le délai ne peut pas excéder 24 heures.

IUD : variable ou constante entière qui précise l'unité du délai
IUD = 0 top de l'horloge de base du système
IUD = 1 milliseconde
IUD = 2 seconde
IUD = 3 minute.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Exemple : CALL WAIT (5,2, IRESP)

Suspendre la tâche pendant 5 secondes. Le compte-rendu de la requête est à placer en IRESP.

Syntaxe Assembleur

Format de la requête

LAD	RPB
SVC	WAIT

Table des paramètres

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	Réserve																
1	Adresse compte rendu de requête																IERR
2	Délai																ID
3															Unité		IUD

Erreurs

IERR = 4 Cette requête est interdite sous niveau hardware

IERR = 6 Il existe au moins un paramètre incorrect
- IUD ⩽ 0,3
- délai supérieur à 24 heures
- adresse de compte rendu erronée
- adresse de RPB erronée

IERR = 7 On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 La tâche reste dans sa partition pendant toute la durée de la suspension.

2 La durée minimale de suspension est au moins égale à 1 top de base.

Début de tâche (DATAG) :

DATAG

But

Acquisition du paramètre de travail.

Syntaxe Fortran

CALL DATAG (IDATA)

IDATA : variable entière chargée avec la valeur du paramètre de travail pour la prochaine exécution de la tâche appelante.

Exemple : Une tâche demande l'activation de la tâche NT :

```
CALL START (NT, 80, 1, IERR, IPAR)
```

Le paramètre de travail IPAR vaut par exemple 27.
L'activation de NT aura lieu dans 80 millisecondes.
Lorsque la tâche NT sera lancée, elle commencera par faire l'acquisition de son paramètre de travail.

```
CALL DATAG (IWORK)
```

Cette requête chargera la variable IWORK avec la valeur 27.

Syntaxe Assembleur

Format de la requête

```
SVC DATAG
```

Cette requête fournit dans l'accumulateur la valeur du paramètre de travail.

Commentaires

1 Cette requête réalise l'acquisition du paramètre de travail d'une tâche. Ce paramètre étant optionnel, son utilisation est facultative. Réciproquement, pendant une exécution, une tâche peut demander plusieurs fois quelle est la valeur du paramètre de travail de l'exécution en cours.

Le schéma logique d'une tâche intégrée sous RTES16 est donc le suivant :

```
100   CALL DATAG (IDATA)

          TRAITEMENT DE L'APPEL

      CALL EXIT (ICRDU)
      GO TO 100
```

2 Lorsque l'appelant est une tâche hardware, la valeur du paramètre de travail est sans signification.

Fin de tâche (EXIT) :

EXIT

But

Signaler la fin d'exécution de la tâche appelante.

Syntaxe Fortran

```
CALL EXIT [(ICRDU)]
```

ICRDU : variable entière représentant le compte rendu du traitement effectué (nul par défaut)

Exemple :

La tâche T1 a demandé l'activation de la tâche T4 et sa suspension jusqu'à la fin du traitement associé :

```
CALL STARTW (T4, ID, IUD, IERR, PT, ICR)
```

Après son activation, T4 récupère le paramètre de travail PT dans une variable propre IDATA :

```
CALL DATAG (IDATA)
```

En fin d'exécution, T4 renvoie le compte rendu de traitement CRT

```
CALL EXIT (CRT)
```

Cette dernière requête réactive la tâche T1 à la suite de l'instruction CALL STARTW : T1 trouve alors la valeur du compte rendu (CRT) dans sa variable ICR.

Syntaxe Assembleur

Format de la requête

```
LA ICRDU < A = compte rendu de traitement
SVC EXIT
```

Avant l'exécution de cette requête, l'accumulateur doit être chargé avec la valeur du compte rendu de traitement.

Commentaires

1 Cette requête est interdite sous niveau hardware (compte-rendu d'erreur dans l'accumulateur en retour de la requête : RA : = 4). Pour acquitter le niveau il faut programmer une instruction ACQ.

2 Exécutée sous niveau software, cette requête désarme la tâche en cours s'il n'y a pas d'appels stockés, libère la partition occupée si la tâche n'est pas résidente, et donne le contrôle au scheduler.
S'il existe des appels cumulés pour cette tâche, elle reste armée mais libère sa partition avant de rendre le contrôle au scheduler, ceci afin d'autoriser le chargement dans la partition d'une tâche plus prioritaire.

3 Une tâche qui libère sa partition est considérée comme présente tant que cette partition n'a pas été affectée à une autre tâche.

4 Le compte rendu fourni en paramètre de la requête n'a d'utilité que dans le cas ou la tâche qui termine un traitement a été activée par une requête STARTW ou TRNONW, c'est-à-dire dans le cas où l'appelant attend ce compte rendu.

Important

1 Lorsqu'une tâche, non résidente, est activée, elle est alimentée en mémoire dans son état initial (image mémoire générée par le builder) si elle n'est pas présente en mémoire.

Dans le cas contraire où, depuis sa précédente activation, sa partition n'a pas été récupérée pour une autre tâche, elle est activée sans chargement préalable en mémoire.
Donc en règle générale, une tâche non résidente ne peut pas utiliser de mémoires de travail internes comme moyen de transfert de paramètres d'une exécution à la suivante.
Cette règle s'applique à la Kstore. qui fait partie intégrante de la tâche.
En particulier, la requête SVC EXIT ne doit pas être réalisée par un sous-programme de la tâche, l'adresse de retour stockée dans la Kstore

pouvant être détruite en cas de rechargement de la tâche.

Le sous-programme CALL EXIT (et CALL CEXIT) de la bibliothèque temps réel BIBRTE-RT sauvegarde l'adresse de retour dans la tâche appelante dans le registre RB, qui est donc détruit en retour de sous-programme.

2 Avant l'exécution de la requête SVC EXIT, une tâche non résidente doit s'assurer, par utilisation de la requête WEIO, que toutes ses demandes d'entrée-sortie en mode retour immédiat (IMOD) sont terminées.

Suppression d'appels de tâche (OFF) :

OFF

But

Suppression des demandes d'activation différées et périodiques d'une tâche faites avec un paramètre de travail spécifié.

Syntaxe Fortran

CALL OFF (NT, IERR, IPAR)

NT : variable ou constante entière représentant le numéro d'appel d'une tâche.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

IPAR : variable ou constante entière qui représente un paramètre de travail d'une tâche.

Exemple : CALL OFF (150, IERS, P2)

> Une tâche, T50 ou autre, décide que la tâche T50 ne doit pas répondre à toute éventuelle demande d'activation antérieure différée ou périodique faite avec le paramètre de travail P2.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     OFF
```

Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Numéro de tâche	NT
1	Adresse compte rendu de requête	IERR
2	Valeur du paramètre de travail	IPAR

Erreurs

IERR = 2 . La tâche NT est inconnue du système
IERR = 3 . La tâche NT est une tâche hardware
IERR = 4 . Le numéro NT est supérieur au numéro maximum des tâches de l'application.
IERR = 6 . Il existe au moins un paramètre incorrect
- NT négatif
- adresse de RPB ou de compte rendu erronés

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 Si IPAR = 0 tous les appels antérieurs, différés ou périodiques, à NT sont effacés.

2 Cette requête est ineffective s'il n'existe aucun appel antérieur différé ou périodique à NT avec le paramètre spécifié.

3 Dans le cas d'un appel différé de tâche avec attente, si l'appel est effacé par la requête OFF, la tache appelante est relancée et reçoit un compte rendu de traitement égal à 0.

Inhibition d'une tâche (INHIB) :

INHIB

But

Inhiber une tâche : la figer dans son état actuel.

Syntaxe Fortran

CALL INHIB (NT, IERR)

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à inhiber ; si NT = 128 (numéro d'appel sans signification), la tâche à inhiber est la tâche appelante.

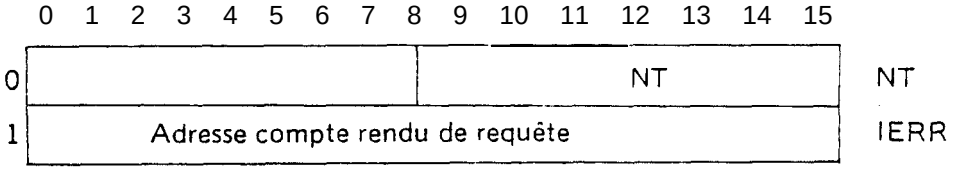
IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     INHIB
```

Table des paramètres



Erreurs

IERR = 2 . La tâche NT est inconnue du système
IERR = 3 . On ne peut pas inhiber une tâche hardware
IERR = 4 . Cette requête est interdite sous niveau hardware
. On ne peut pas inhiber une tâche système.
IERR = 5 . La tâche NT est déjà inhibée (requête INHIB ou commande STOP).
IERR = 6 . Il existe au moins un paramètre incorrect.
- NT négatif
- adresse de RPB
- adresse de compte rendu de requête.

Commentaires

Une tâche peut s'auto-inhiber.

La tâche inhibée est suspendue dans l'état ou elle se trouve : elle conserve ses ressources et la partition qu'elle occupe si elle est présente en mémoire ; si elle n'est pas présente, elle ne sera pas chargée dans une partition pendant toute la durée de l'inhibition.

La commande opérateur VALI, NT met fin à cette inhibition.

Les demandes d'activation de cette tâche sont enregistrées ; les appelants avec attente restent suspendus jusqu'à la fin de l'exécution demandée, après la demande de validation.

Chargement et lancement d'une branche (BCHLR) :

BCHLR

But

Chargement en mémoire principale d'une branche de la tâche appelante, et lancement à son point d'entrée avec mémorisation du point de retour dans la racine ou la branche appelante.

Syntaxe Fortran

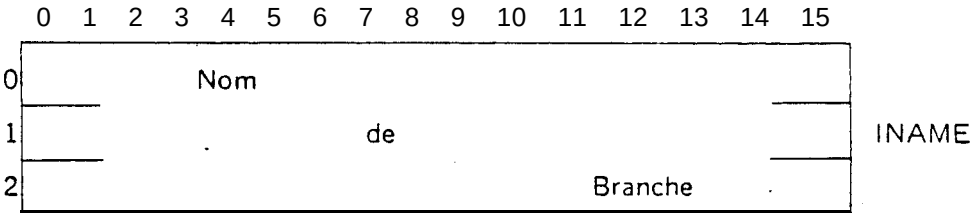
Se reporter au chapitre "SEGMENTATION" du MANUEL D'UTILISATION FORTRAN.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     BCHLR
```

Table des paramètres



Nom de branche : 6 caractères, 2 caractères par mot.

Erreurs

IERR = 2 . La tâche appelante n'a pas accès à la branche de nom spécifié.
IERR = 3 . Informations non valides sur disque
. La tâche appelante n'a pas de structure d'overlay
IERR = 4 . Cette requête est interdite aux tâches hardware.
IERR = 5 . Disque non prêt.
IERR = 6 . Adresse de RPB incorrecte.
IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Dans la syntaxe assembleur, le compte rendu de requête est chargé dans l'accumulateur.

Commentaires

1 Une tâche s'exécute dans une partition dont la taille minimale est égale à celle de la racine plus celle de la branche-maximale.

L'accès d'une tâche (éventuellement structurée en Overlay) à une partition donnée est contrôlé au moment de l'affectation de partition (s) à cette tâche, c'est-à-dire :
- au moment de son intégration sous RTES16 par la commande d'intégration ON-LINE
- lors de l'exécution de la commande qui permet de modifier l'affectation initiale de partition à une tâche (non résidente)

2 On rappelle (cf. Chapitre : OVERLAY) que la mémorisation du point de retour dans une branche appelante se fait à un seul niveau.

3 L'adresse du point d'entrée dans une branche se trouve dans le premier mot de la branche.

Fin de branche (BACK):

BACK

But

Fin d'exécution de la branche en cours : retour à l'appelant (directement à la racine, ou à la branche appelante après son chargement en mémoire principale).

Syntaxe Fortran

Se reporter au Chapitre "SEGMENTATION" du Manuel D'utilisation FORTRAN.

Syntaxe Assembleur

Format de l'appel

```
SVC  BACK
```

Erreurs

IERR = 3 . La tâche appelante n'a pas de structure d'overlay
. Cette requête est exécutée dans la racine
. Informations non valides sur disque

IERR = 4 . Cette requête est refusée sous niveau hardware

IERR = 5 . Disque non prêt.

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Dans la syntaxe ASSEMBLEUR, le compte rendu de requête est chargé dans l'accumulateur.

Commentaires

1 Si l'appelant de la branche en cours est la racine, cette requête effectue un branchement à l'adresse de retour dans la racine.

2 Si l'appelant est une branche, la phase de retour est précédée de la phase de chargement en mémoire principale de la branche appelante : la branche qui fait SVC BACK est "écrasée".

Fin de branche (BACKR) :

BACKR

But

Signaler la fin d'exécution de la branche en cours : retour direct à la racine à l'adresse qui suit le dernier appel d'une branche depuis la racine.

Syntaxe Fortran

Se reporter au Chapitre "SEGMENTATION" du MANUEL D'UTILISATION FORTRAN.

Syntaxe Assembleur

Format de l'appel

```
SVC  BACKR
```

Erreurs

IERR = 3 . La tâche appelante n'a pas de structure d'overlay
. Cette requête est exécutée dans la racine
IERR = 4 . Requête refusée sous niveau hardware
Dans la syntaxe ASSEMBLEUR, le compte rendu de requête est chargé dans l'accumulateur.

Commentaires

1 Cette requête est particulièrement adaptée au principe d'appel séquentiel des branches : une portion de tâche, relativement linéaire et de taille importante, est découpée en branches qui s'appellent en cascade, la dernière retournant directement à la racine chargée d'enchaîner les fonctions à réaliser.

2 La requête BACK est au contraire mieux adaptée au principe d'appel d'une branche considérée comme sous-programme.

Attente d'un événement (WEVENT) :

WEVENT

But

Attendre l'arrivée d'un événement

Syntaxe Fortran

CALL WEVENT (IEV, IERR)

IEV : variable ou constante entière représentant un numéro d'événement.

IERR : variable entière chargée par le compte rendu de requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Exemple :

```
CALL WEVENT (13, IERR)
```

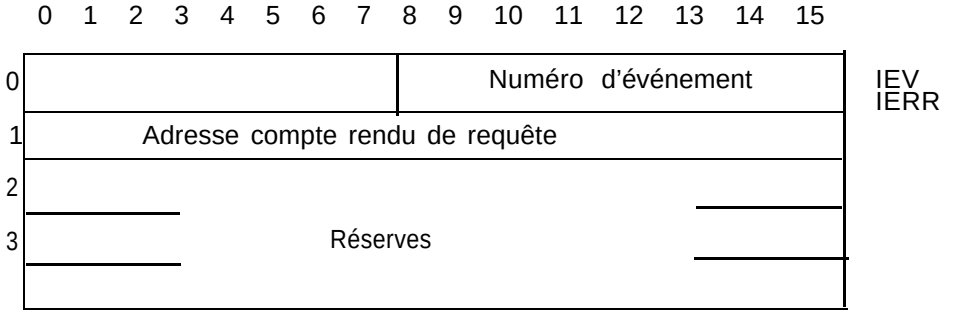
Suspendre la tâche appelante jusqu'à l'arrivée de l'événement 13 (jusqu'à ce qu'il soit dans l'état "SET")

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     WEVENT
```

Table des paramètres



Erreurs

IERR = 4 . Requête interdite aux tâches hardware
IERR = 6 . Il existe au moins un paramètre incorrect
- adresse de RPB
- adresse de compte rendu de requête
- numéro d'évènement ∉ [0,255]

IERR = 7 . On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1 L'appelant n'est pas suspendu si l'événement est déjà dans l'état SET.

2 Pendant sa suspension, la tâche reste dans sa partition.

Attente de plusieurs événements (WEVAND) :

WEVAND

But

Attendre l'arrivée d'une liste d'événements : suspendre la tâche appelante jusqu'à ce qu'ils soient simultanément dans l'état SET.

Syntaxe Fortran

CALL WEVAND (IERR, IA, IB, IC,...)

IA, IB, IC... : variables ou constantes entières représentant une liste d'événements. Les événements de cette liste doivent appartenir à la même classe :

0 à 31
ou 32 à 63
. . . . . .
. . . . . .
ou 224 à 255

I E R R : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Exemple :

CALL WEVAND (IERR, 32, 35, 60, 40, 63)

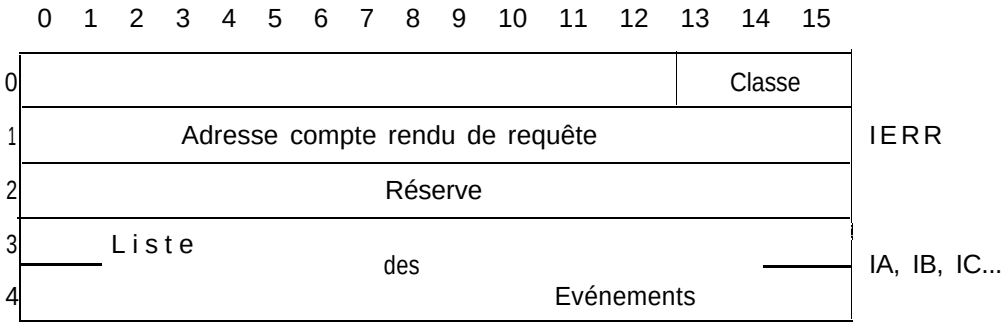
Attendre l'arrivée des événements 32 et 35 et 60 et 40 et 63.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     WEVAND
```

Table des paramètres



Exemple :
Reprenons en langage ASSEMBLEUR la précédente requête écrite en FORTRAN :
- les événements de la liste appartiennent à la classe 1
- dans cette classe, le rang des événements spécifiés est respectivement :

Numéro d'évènement	32	35	60	40	63
Numéro d'évènement dans la classe	0	3	28	8	31

Les mots 3 et 4 du RPB (32 bits) doivent contenir un bit par événement attendu :

événement 0	bit 0
événement 1	bit 1
événement 31	bit 31

On en déduit le format-du RPB.

```
RPB:WORD 1          < Numéro de classe
    WORD CONRDU     < Adresse compte rendu
    WORD 0          < Réservé
    WORD '9080      < Bits 0, 3, 8
    WORD '0009      < Bits 28, 31
```

Erreurs

IERR = 4 . Requête interdite aux tâches hardware

IERR = 6 . Il existe au moins un paramètre incorrect
- adresse de RPB
- adresse de compte rendu de requête
- Numéro de classe

IERR = 7 . On ne peut pas enregistrer la requête. Le système est sous-dimensionné.

Commentaires

1 L'appelant n'est pas suspendu si tous les événements attendus sont déjà dans l'état SET.

2 Pendant sa suspension, la tâche reste dans sa partition.

Attente d'un événement parmi plusieurs (WEVOR) :

WEVOR

But

Attendre l'arrivée de l'un des événements d'une liste : suspendre la tâche appelante jusqu'à ce qu'il en existe un dans l'état SET.

Syntaxe Fortran

CALL WEVOR (IERR, ICR, IA, IB, IC...)

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée.

ICR : variable entière ou tableau entier chargé par le compte rendu de traitement : il précise quel est l'événement de la liste qui a provoqué le réveil de l'appelant.
- le tableau doit comprendre autant d'éléments qu'il y a d'événements attendus dans la liste,
- au retour de la requête, le ou les éléments du tableau contiennent soit un numéro d'événement arrivé, soit la valeur initiale (chargée par exemple à -1 par l'utilisateur avant la requête),
- les numéros des événements arrivés sont rangés par ordre de numéro croissant dans le tableau,
- il y a au moins un élément du tableau chargé par la requête (1 seul événement arrivé) au plus tous les éléments du tableau sont chargés (tous les événements de la liste sont arrivés).

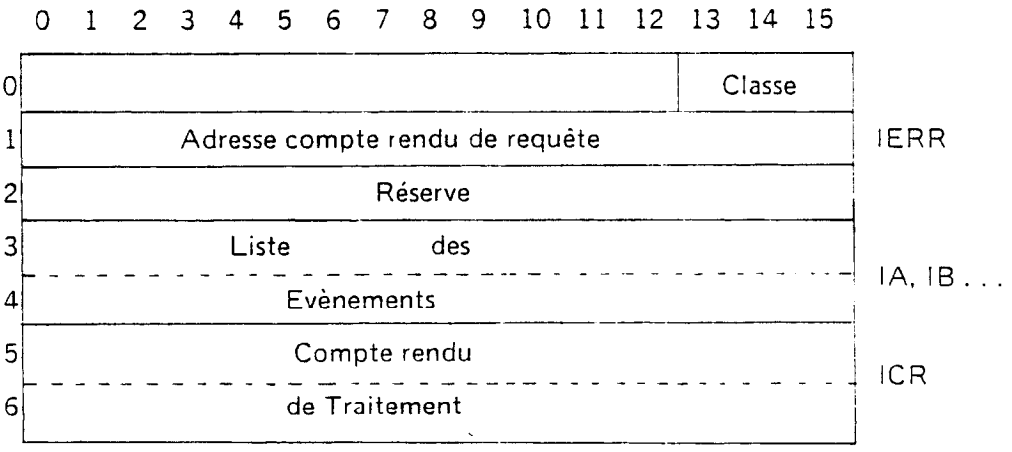
IA, IB, IC... : variables ou constantes entières représentant une liste d'événements d'une même classe.

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     WEVOR
```

Table des paramètres



Erreurs

IERR = 4 . Requête interdite aux tâches hardware
IERR = 6 . II existe au moins un paramètre incorrect
- adresse de RPB
- adresse de compte rendu de requête
- numéro de classe.

IERR = 7 . On ne peut pas enregistrer la requête, le système est sous-dimensionné.

Commentaires

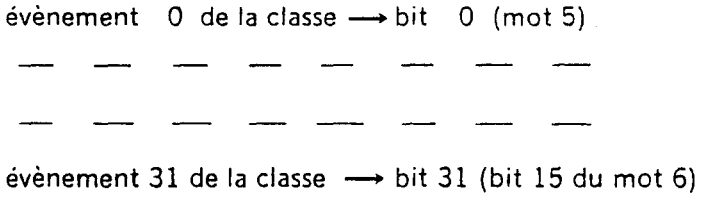
1 L'appelant n'est pas suspendu si l'un des événements attendus est déjà dans l'état SET.

2 Pendant sa suspension, la tâche reste dans sa partition.

3 Lorsque la condition d'attente est satisfaite, la tâche est relancée en séquence après la requête ; le compte rendu de traitement lui indique quel est, dans la liste précisée, l'événement qui a provoqué le réveil de l'appelant.
Dans le cas précisé par la remarque 1, ce compte-rendu peut spécifier plusieurs événements.

4 Le format des mots 3 et 4 du RPB est analogue à celui des mêmes mots du RPB de la requête WEVAND.

Le format des mots 5 et 6 du compte-rendu de traitement est le même :



Signaler l'arrivée d'un événement (SEVENT) :

SEVENT

But

Mettre l'événement dans l'état "SET"

Syntaxe Fortran

CALL SEVENT (IEV, IERR)

IEV : Variable ou constante entière représentant un numéro d'événement.

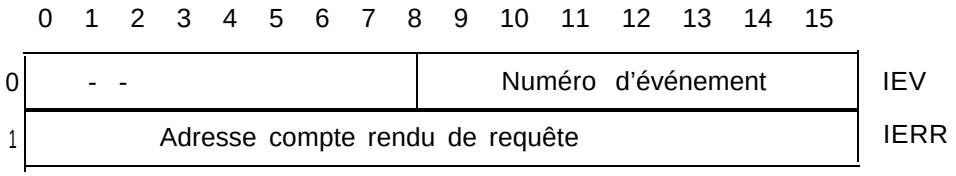
IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Syntaxe Assembleur

Format de la requête

```
LAD     RPB
SVC     SEVENT
```

Table des paramètres



Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect
- adresse de RPB
- adresse de compte rendu de requête
- numéro d'évènement ∉ [0,255]

IERR = 7 . Système sous-dimensionné.

Commentaires

1 Cette requête est ineffective si l'événement est déjà dans l'état "SET".

2 Le réveil des tâches pour lesquelles cette requête réalise la condition de réveil est effectif même si la plus prioritaire d'entr'elles, une fois relancée, fait un (plusieurs) RESET d'événement,

Signaler l'arrivée différée d'un événement (SEVDEL) :

SEVDEL

But

Signaler au système qu'il devra le mettre dans l'état "SET" après un délai spécifié.

Syntaxe Fortran

```
CALL SEVDEL (IEV, IERR, ID, IUD)
```

. IEV : variable ou constante entière représentant un numéro d'événement

. IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

. ID : variable ou constante entière qui représente, en unités spécifiées par IUD, le délai d'arrivée de l'événement.
Si ID = 0, la requête est équivalente à SEVENT.
Le délai ne peut pas excéder 24 heures

. IUD : variable ou constante entière qui précise l'unité du délai
IUD = 0 top de l'horloge de base
IUD = 1 milliseconde
IUD = 2 seconde
IUD = 3 minute

Syntaxe Assembleur

. Format de la requête

```
LAD RPB
SVC SEVDEL
```

. Table des paramètres

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15
0		Numéro d'événement
1	Adresse compte rendu de requête	
2	Délai	
3		Unité

Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte-rendu de requête
- Numéro d'évènement $\notin [0,255]$
- Délai incorrect

IERR = 7 . Système sous-dimensionné.

Commentaires

1 Le principe "d'événement-délai", joint à la notion d'attente combinée d'événements, permet de résoudre des problèmes de temps "enveloppe".
Précisons sur un exemple :

```
CALL SEVDEL    (IV1, IERR, 500, 1)
CALL WEVOR     (IERS, ICR, IV1, IV2, IV5)
```

Cette séquence permet de suspendre une tâche jusqu'à l'arrivée de l'un des événements IV2 ou IV5 jusqu'à concurrence de 500 millisecondes, le compte-rendu ICR indiquant au réveil de la tâche qui est à l'origine du réveil, le délai, IV2 ou IV5.

Effacement de la requête d'arrivée différée d'un événement (OFFDEL)

OFFDEL

But

Supprimer l'effet de toute éventuelle requête SEVDEL portant sur l'événement spécifié.

Syntaxe Fortran

CALL OFFDEL (IEV, IERR)

IEV : variable ou constante entière représentant un numéro d'événement

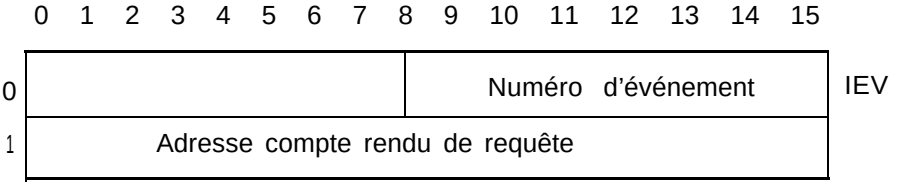
IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geq$ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

LAD RPB
SVC OFFDEL

. Table des paramètres



Exemple :

. La requête : CALL SEVDEL (55, IERR, 3, 2)
demande au système de mettre l'événement 55 à l'état "SET" dans 3 secondes.

. La requête : CALL OFFDEL (55. CRENDU)
demande au système d'effacer toute éventuelle action différée relative à l'événement 55 prescrite par SEVDEL.

Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu de requête
- Numéro d'évènement $\notin$ [0,255]

IERR = 7 . Système sous-dimensionné.

Commentaires

1 Cette requête est ineffective si aucune requête SEVDEL portant sur l'événement spécifié n'est en cours.

Signaler la disparition d'un événement (REVENT) :

REVENT

But

Mettre l'événement dans l'état "RESET".

Syntaxe Fortran

```
CALL REVENT (IEV, IERR)
```

IEV : variable ou constante entière qui précise un numéro d'événement.

IERR : variable entière chargée par la réponse à la requête

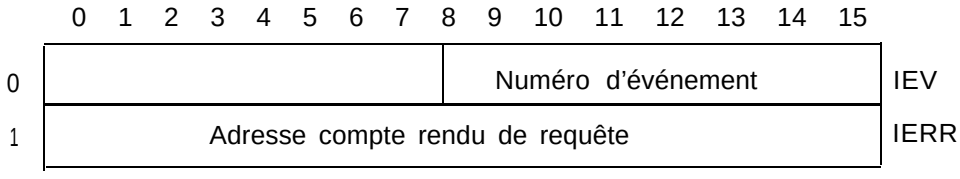
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     REVENT
```

. Table des paramètres



Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte-rendu de requête
- Numéro d'évènement ∉ [0,255]

IERR = 7 . Système sous-dimensionné.

Commentaires

1 Cette requête est ineffective si l'événement est déjà dans l'état "RESET".

Test de l'état d'un événement (TEVENT) :

TEVENT

But

Savoir si l'événement est "SET" ou "RESET".

Syntaxe Fortran

CALL TEVENT (IEV, IERR. ICR)

IEV : constante ou variable entière représentant un numéro d'événement

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

ICR : variable entière chargée par l'état de l'événement :
ICR = 1 état SET
ICR = 2 état RESET

Exemple :

```
CALL TEVENT (4, IRP, CONRDU)
```

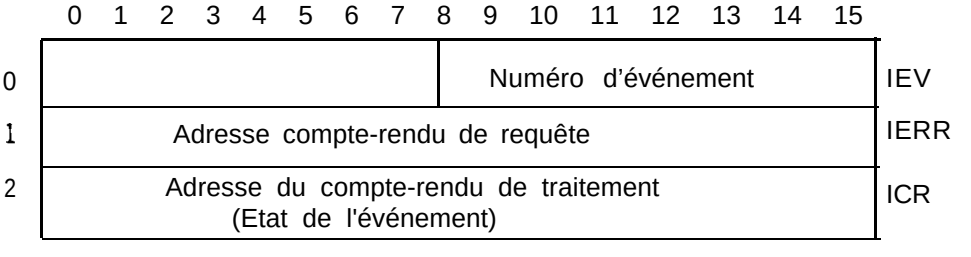
Tester l'événement 4 : placer son état instantané en CONRDU

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     TEVENT
```

. Table des paramètres



Erreurs

IERR = 4 . Requête interdite aux tâches hardwares

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu de requête
- Numéro d'évènement ∉ [0,255
- Adresse compte rendu de traitement

Commentaires

1 La tâche appelante teste l'événement, sans être suspendue.

Transférer dans un buffer une portion de la zone des données résidentes :

REDGET

But

Lire une portion de la zone de données résidentes : la transférer dans un buffer de l'appelant.

Syntaxe Fortran

CALL REDGET (IN, IERR, IDEP, IADR)

IN : variable ou constante entière qui précise le nombre de mots à transférer

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

IDEP : variable ou constante entière qui situe la portion de la zone des données résidentes par un déplacement depuis son adresse origine.

IADR : variable entière ou tableau entier qui précise le nom du buffer chez l'appelant.

Exemple : CALL REDGET (25, IRESP, 50, ITAB)

> Ranger dans le tableau ITAB les 25 mots de la zone des données résidentes de rang 50 à 74 inclus.

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     REDGET
```

. Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Déplacement depuis le début de la ZDR	IDEP
1	Adresse de compte rendu de requête	IERR
2	Adresse du buffer	IADR
3	Nombre de mots à transmettre	IN

. Modalités d'exécution . Le transfert n'est pas masqué : les divers usagers de la zone des données résidentes doivent résoudre par eux-mêmes les problèmes éventuels d'exclusion d'accès.

Erreurs

IERR = 6 Il existe au moins un paramètre incorrect
Adresse de RPB, de compte rendu ou de buffer
Nombre de mots ou déplacement négatifs (sur 16 bits)
La zone précisée par lDEP et IN n'est pas inclue dans le commun des données résidentes
Buffer non inclu dans la partition

Commentaires

1 La gestion de cette zone est laissée à la charge de l'usager (protection, exclusion d'accès...)

2 Cette requête est acceptée sous niveau hardware.

Transfert d'un buffer en zone de données résidentes :

REDPUT

But

Ecrire une portion de la zone de données résidentes : y transférer un buffer propre à la tâche appelante.

Syntaxe Fortran

CALL REDPUT (IN, IERR, IDEP, IADR)

IN : variable ou constante entière qui précise le nombre de mots à transférer

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

IDEP : variable ou constante entière qui situe la portion de la zone des données résidentes par un déplacement depuis son adresse origine.

IADR : variable entière ou tableau entier qui précise le nom du buffer chez l'appelant.

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     REDPUT
```

. Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0	Déplacement depuis le début de la ZDR
1	Adresse compte rendu de requête
2	Adresse du buffer
3	Nombre de mots à transmettre

Modalités d'exécution

Le transfert n'est pas masqué : les divers usagers de la zone des données résidentes doivent résoudre par eux-mêmes les problèmes éventuels d'exclusion d'accès.

Erreurs

IERR = 6 Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu
- Adresse dans la partition
- Déplacement négatif (sur 16 bits)
- Nombre de mots négatif (sur 16 bits)
- La zone précisée par IDEP et IN n'est pas inclue dans la zone des données résidentes
- Buffer non inclu dans la partition

Commentaires

1 La gestion de cette zone est laissée à la charge de l'usager (protection, exclusion d'accès,...).

2 Cette requête est acceptée sous niveau hardware.

Définition d'une ressource :

RESDEF

But

Définir une ressource par son numéro et le nombre d'accès simultanés maximum.

Syntaxe Fortran

CALL RESDEF (IR, IAC, IERR)

IR : variable ou constante entière qui représente le numéro de la ressource

IAC : variable ou constante entière qui précise le nombre maximum d'accès simultanés à la ressource : IAC doit être compris entre 1 et 127

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant 2$ requête refusée

Exemple :

CALL RESDEF (12. 4. IERR)

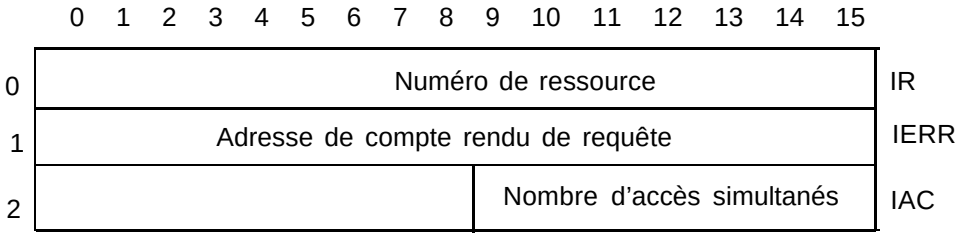
Définition de la ressource usager de numéro 12 à 4 accès simultanés possibles

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     RESDEF
```

. Table des paramètres



Erreurs

IERR = 3 . Ressource déjà définie

IERR = 4 . Il existe au moins un paramètre incorrect

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu
- Nombre d'accès $\notin [1,127]$

IERR = 7 . RTES16 ne peut pas enregistrer la requête : système sous-dimensionné.

Commentaires - L'utilisation d'une telle ressource est laissée à l'entière responsabilité de l'usager : il en fait ce qu'il désire, une erreur d'utilisation ne pouvant entraîner que la dégradation (blocages, "deadlock") de certaines tâches, et non une destruction du système.

Demande d'accès à une ressource :

RRQST

But

Demander un accès à une ressource.

Syntaxe Fortran

CALL RRQST (IR, IERR)

IR : variable ou constante entière qui précise le numéro de la ressource.

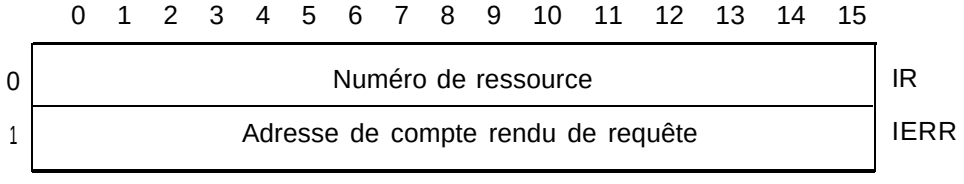
IERR : variable entière chargée par la réponse à la requête.
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée ·

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     RRQST
```

. Table des paramètres



Erreurs

IERR = 2 . La ressource IR n'a pas été définie

IERR = 4 . Requête interdite aux tâches hardware.

IERR = 6 . Il existe au moins un paramètre incorrect.
- Adresse de RPB
- Adresse de compte rendu de requête.

Commentaires

1 Il y a suspension de l'appelant si aucun accès à la ressource n'est disponible, et réveil lorsqu'un accès à la ressource étant libéré l'appelant est le plus prioritaire de tous ceux qui ont demandé l'accès.

Libération d'un accès à une ressource :

RRLSE

But

Libérer un accès à une ressource.

Syntaxe Fortran

CALL RRLSE (IR, IERR)

IR : variable ou constante entière qui précise le numéro de la ressource.

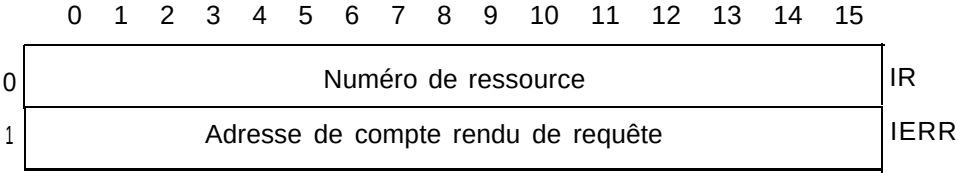
IERR : variable entière chargée par le compte rendu de requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     RRLSE
```

. Table des paramètres



Erreurs

IERR = 2 . La ressource IR n'a pas été définie

IERR = 3 . Requête illogique : création illicite d'un accès à une ressource

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu de requête.

Définition de la phase d'avancement :

STADEF

But

Définir la phase d'avancement.

Syntaxe Fortran

CALL STADEF (IPH, IERR)

IPH : variable ou constante entière qui précise la phase d'avancement de l'appelant.

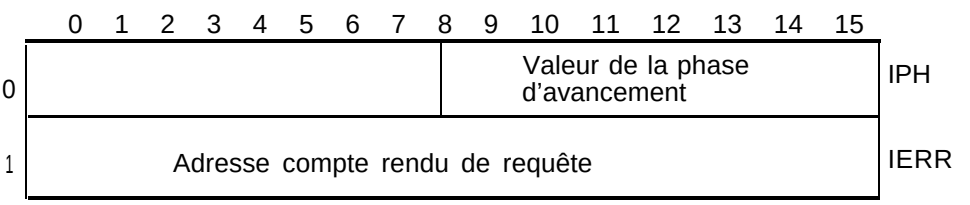
IERR : variable chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     STADEF
```

. Table des paramètres



Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu de requête
- IPH ∉ [0,255]

Commentaires

Cette requête est acceptée, mais ineffective, sous niveau hardware.

Acquisition de la phase d'avancement :

STAGET

But

Connaître la phase d'avancement d'une tâche.

Syntaxe Fortran

CALL STAGET (NT, ICR, IERR)

NT : variable ou constante entière qui représente le numéro de la tâche

ICR : variable entière chargée par la valeur de la phase d'avancement de la tâche.

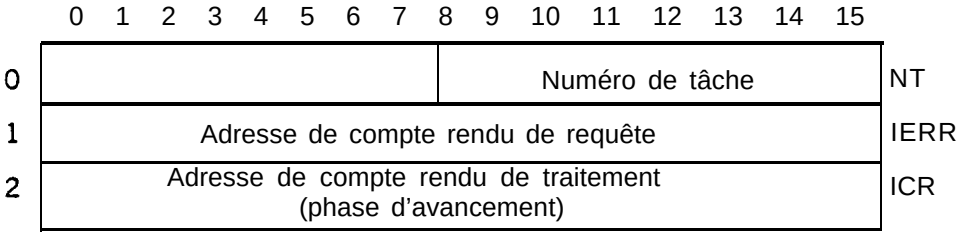
IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     STAGET
```

. Table des paramètres



Erreurs

IERR = 2 . La tâche NT est inconnue du système

IERR = 3 . La tâche NT est une tâche hardware

IERR = 4 . Le numéro NT est supérieur au numéro maximum des tâches de l'application

IERR = 6 . Il existe au moins un paramètre incorrect :
- Adresse de RPB
- Adresse de compte rendu de requête
- NT négatif

Commentaires

1 La signification donnée à la valeur de la phase d'avancement est spécifique de chaque tâche et résulte de conventions entre les tâches de l'application.

Initialisation de la date et de l'heure :

WTIME

But

Actualiser la date et l'heure

Syntaxe Fortran

CALL WTIME (IDATIM, IERR)

IDATIM : tableau entier de 7 mots contenant la date et l'heure

IERR : variable chargée par le compte rendu de requête

IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

Syntaxe Assembleur

Format de la requête

```
LAD RPB
SVC WTIME
```

Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Adresse du tableau date et heure	IDATIM
1	Adresse compte rendu de requête	IERR

Erreurs

IERR = 6 Il existe au moins un paramètre incorrect
. Adresse de RPB
. Adresse de compte rendu de requête
. Adresse du tableau date et heure
. Paramètres non vraisemblables (JOUR > 31, etc)

Commentaires

Le tableau IDATIM est chargé par l'appelant avant l'exécution de la requête selon le schéma ci-dessous :

0	Année	[78,99]
1	Mois	[1,12]
2	Jour	[1,31]
3	Heure	[0,23]
4	Minute	[0,59]
5	Seconde	[0,59]
6	Milliseconde	[0,1000 - P]

— P : pas de l'horloge de base.

Les informations du tableau doivent être en format binaire.
Si les paramètres Année, Mois, Jour sont égaux à 0 la date sera inchangée.

Lors du traitement de la requête, RTES16 recalcule la date d'échéance des opérations sur horloge.

- Opération sur délai :
  Heure échéance = nouvelle heure + délai restant.

- Opération sur date :

  si heure d'échéance> nouvelle heure -> l'opération sera activée à la date d'échéance.

  si heure d'échéance ⩽ nouvelle heure ⟶ l'opération sera activée immédiatement.

Le système met à jour le coupleur multi-fonctions si la FU temps réel est configurée (% PUTRL présente dans les macro-instructions d'lOCS).

Lecture de la date et de l'heure :

TIME

But

Lire la date et l'heure.

Syntaxe Fortran

CALL TIME (IDATIM, IERR)

IDATIM : tableau entier de 7 mots chargé par la date et l'heure

IERR : variable chargée par le compte rendu de requête

IERR = 1 requête acceptée

IERR $\geqslant$ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     TIME
```

. Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Adresse compte rendu de traitement (7 mots)	IDATIM
1	Adresse compte rendu de requête	IERR

Erreurs

IERR = 6 . Il existe au moins un paramètre incorrect

- Adresse de RPB
- Adresse de compte rendu de requête
- Adresse de compte rendu de traitement

Commentaires

Le tableau IDATIM est chargé par la requête selon le schéma ci-dessous :

0	Année	[73,99]
1	Mois	[1,12]
2	Jour	[1,31]
3	Heure	[0,23]
4	Minute	[0,59]
5	Seconde	[0,59]
6	Milliseconde	[0,1000-p] → p : pas de l'horloge de base

Les informations du tableau sont données en format binaire.

Mettre une tâche en attente sur un sémaphore privé :

RWAIT

But

Simuler l'instruction sur sémaphore WAIT du SOLAR 16 (interdite aux tâches en mode esclave). Le sémaphore privé ainsi utilisé doit être situé dans la zone des données résidentes et peut être un sémaphore avec ou sans file de paramètres.

Syntaxe Fortran

CALL RWAIT (IDEP, IERR)

IDEP : variable ou constante entière qui situe le sémaphore privé dans la ZDR par un déplacement depuis son adresse origine.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR $\geqslant$ 2 requête refusée

Syntaxe Assembleur

. Format de la requête

```
LAD     RPB
SVC     RWAIT
```

. Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Déplacement depuis le début de la ZDR	IDEP
1	Adresse de compte rendu de requête	IERR

Erreurs

IERR = 2 . IDEP est incorrect (débordement de la ZDR) .

IERR = 4 . Cette requête est interdite sous niveau hardware.

IERR = 6 . II existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu de requête
- IDEP négatif

Activer une tâche en attente sur un sémaphore privé :

RACT

But

Simuler l'instruction sur sémaphore ACT du SOLAR 16 (interdite aux tâches en mode esclave). Le sémaphore privé (avec ou sans file de paramètres) doit être situé dans la zone des données résidentes.

Syntaxe Fortran

```
CALL RACT (IDEP, IERR [, IPAR] )
```

IDEP : variable ou constante entière qui situe le sémaphore privé dans la ZDR par un déplacement depuis son adresse origine.

IERR : variable chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

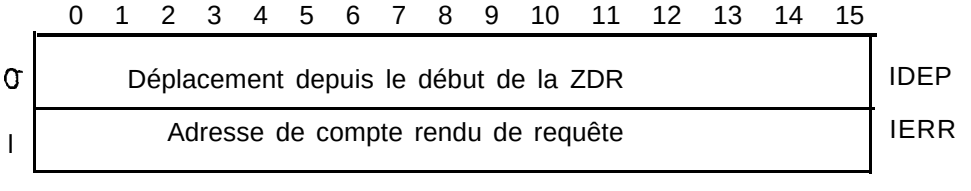
IPAR : variable ou constante entière qui représente la valeur du paramètre (cas d'un sémaphore privé avec file seulement).

Syntaxe Assembleur

. Format de la requête

```
[LY    IPAR]
 LAD  RPB
 SVC  RACT
```

. Table des paramètres



Erreurs

IERR = 2 . IDEP est incorrect (débordement de la ZDR)
. Relativement à IDEP, IPAR est incorrect (débordement de la ZDR)

IERR = 6 . Il existe au moins un paramètre incorrect
- Adresse de RPB
- Adresse de compte rendu d'erreur
- IDEP négatif

Activation d'une "tâche fille" :

TCALL

But

Activer un programme identifié par le nom de son fichier image mémoire stocké sur disque, et lui transmettre un paramètre de travail.

Syntaxe Fortran

CALL TCALL (NT, IFU, IERR, IPAR, NOMFIC)

NT : variable ou constante entière précisant le numéro d'appel de la tâche à activer.

IFU : variable ou constante entière précisant le numéro de la FU disque support du fichier image mémoire de la tâche à activer.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

IPAR : variable ou constante entière représentant le paramètre de travail transmis à la tâche appelée.

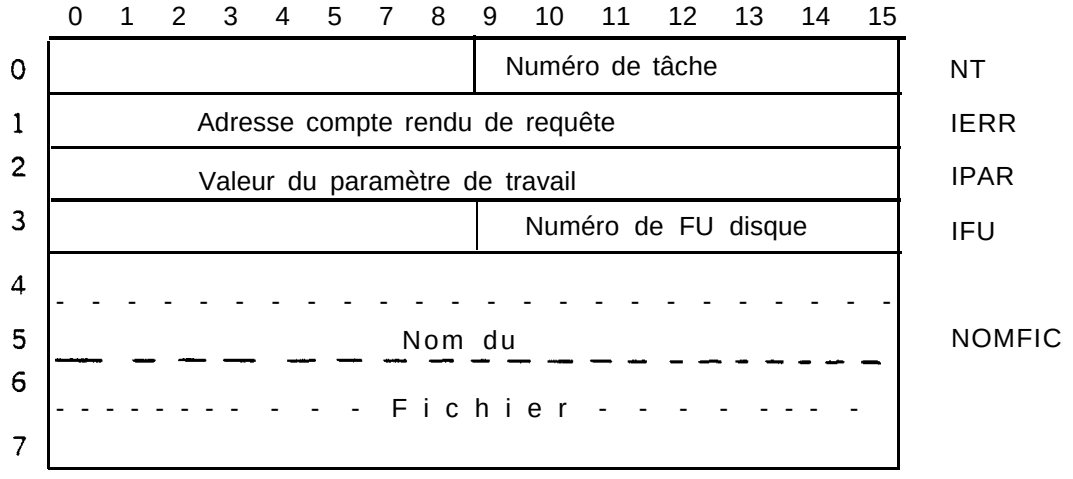
NOMFIC : tableau entier représentant le nom du fichier support de la tâche à activer.

Syntaxe Assembleur

Format de la requête

```
LAD RBP
SVC TCALL
```

Table des paramètres



Erreurs

IERR = 2. La tâche NT est inconnue du système.

IERR = 3. La tâche NT est une tâche hardware.

IERR = 4. Le numéro NT est supérieur au numéro maximum des tâches de l'application.

IERR = 5. Le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.

IERR = 6. II existe au moins un paramètre incorrect
- NT négatif
- adresse de compte rendu erronée
- adresse de RPB erronée

IERR = 7. On ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaires

1- Les programmes, identifiés par le nom du fichier image mémoire associé, sont baptisés tâches filles.

- Créer et activer une tâche fille consiste à lancer un programme stocké sur disque en lui associant un numéro d'appel, c'est-à-dire en fait une priorité. Le nombre de tâches filles n'est donc limité que par le volume de la mémoire secondaire.

- On appelle tâche mère, la tâche non résidente unique, intégrée sous le système par la commande TASK, qui réserve lors de son intégration les différentes ressources nécessaires à l'exécution de ses tâches filles :
  - une priorité
  - un numéro d'appel
  - un bloc de contrôle de tâche (PST + extension)
  - éventuellement un bloc de contrôle d'overlay
  - une liste de 1 à 4 partitions accessibles
  - un numéro d'usager pour FMS.

2 - A un instant donné, le système peut gérer plusieurs processus tâche mère-tâches filles : toutes les tâches non résidentes peuvent être à la limite des tâches mères.

3 - Si l'une des tâches filles est structurée en overlay, la tâche mère doit avoir une structure d'overlay (réservation d'un bloc de contrôle d'overlay lors de l'intégration de la tâche mère).

4 - La tâche mère doit avoir une taille d'image mémoire au moins aussi grande que la plus grande des tâches filles : cette règle n'est pas impérative mais conduit à une plus grande sécurité dans la mesure où lors de l'intégration de la tâche mère le système vérifie que sa taille est compatible avec la ou les partitions qui lui sont allouées.

5 - La tâche mère et ses tâches filles peuvent être indifféremment en mode maître ou en mode esclave ; si l'une d'elles est en mode maître, une seule partition lui sera accessible, celle dont l'adresse correspond à l'adresse donnée au builder lors de la création sur disque de l'image mémoire de la tâche.

6 - Le non respect de ces règles conduit à un refus du système lors de la tentative d'approvisionnement des tâches filles ; l'appel correspondant est alors perdu. Un message de la forme :
ERS 06 NT = nn 'hhhh FICNAM-PW est imprimé sur l'unité symbolique SA.

nn numéro d'appel de la tâche fille à intégrer

'hhhh code hexadécimal indiquant le motif de refus

'0003 . nombre de PST dans la tâche fille à intégrer différent de 1
la tâche fille à intégrer a une structure d'overlay alors que la tâche mère correspondante n'en a pas.

'0007 . on ne peut pas enregistrer l'appel : le système est sous-dimensionné.

'0009 . la tâche fille à intégrer est en mode maître et son adresse d'implantation ne correspond pas au début de la partition spécifiée lors de l'intégration de la tâche mère.

'4nnn
'6nnn erreurs détectées par FMS lors de la demande de lecture sur disque du descripteur de fichier ou de la racine de la tâche fille à intégrer.
exemple :
'6003 : article plus long, la taille du fichier image mémoire de la tâche fille à intégrer est supérieure à la taille de la partition réceptrice.

7 - Une tâche fille a accès aux requêtes de RTES16 sans restriction aucune par rapport aux tâches non résidentes. Elle doit contenir une PST ayant la structure standard (voir Annexe) ; néanmoins, les paramètres figurant dans les quatre mots d'extension de la PST (numéro de partition, cumul d'appel, numéro de priorité) sont ignorés par le système : ces paramètres sont des attributs de la tâche mère dont le numéro d'appel est précisé dans le RPB de la requête TCALL (ou TCALL W)

Activation d'une "tâche fille" avec attente

TCALLW

But

Activer un programme identifié par le nom de son fichier image mémoire sur disque, lui transmettre un paramètre de travail et être suspendu jusqu'à la fin de l'exécution demandée.

Syntaxe Fortran

CALL TCALLW (NT, IFU, IERR, ICR, IPAR, NOMFIC)

NT : variable ou constante entière précisant le numéro d'appel de la tâche à activer.

IFU : variable ou constante entière précisant le numéro de la FU disque support du fichier image mémoire de la tâche à activer.

IERR : variable entière chargée par la réponse à la requête.
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

ICR : variable entière chargée par le compte rendu d'exécution de la tâche appelée, avant la réactivation de l'appelant.

IPAR : variable ou constante entière représentant le paramètre de travail transmis à la tâche appelée.

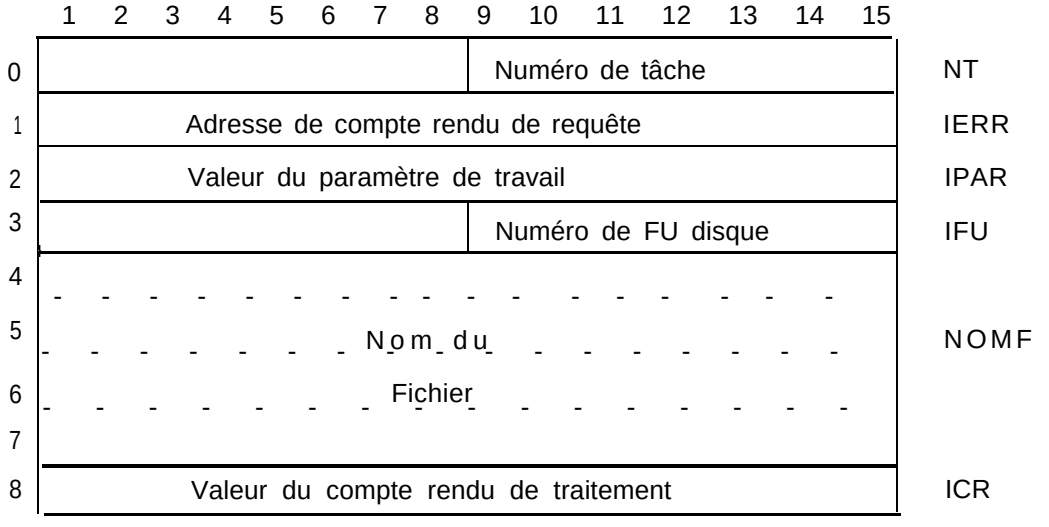
NOMFIC : tableau entier représentant le nom du fichier support de la tâche à activer.

Syntaxe Assembleur

Format de la requête

```
LAD RP8
SVC TCALLW
```

Table des paramètres



Erreurs

IERR = 2 . la tâche NT est inconnue du système.

IERR = 3 . la tâche NT est une tâche hardware.

IERR = 4 . le numéro NT est supérieur au numéro maximum des tâches de l'application.

IERR = 5 . le système ne peut pas enregistrer la requête, le nombre maximum d'appels stockables pour NT étant déjà atteint.

IERR = 6 . il existe au moins un paramètre incorrect
- NT négatif
- adresse de compte rendu erronée
- adresse de RBP erronée

IERR = 7 . on ne peut pas enregistrer la requête : le système est sous-dimensionné.

Commentaire

1 - Une tâche fille ne peut pas activer, à l'aide de la requête TCALLW, une autre tâche fille de la même famille, c'est-à-dire ayant la même tâche mère (ou le même numéro d'appel).

2 - Les cas de refus d'approvisionnement d'une tâche fille activée par TCALLW sont identiques à ceux de la requête TCALL ; à la suite d'un tel cas de refus, la tâche appelante est réveillée par le système qui force un compte rendu de traitement égal à 0.

Acquisition des frontières et du numéro de partition occupée (FREEM)

FREEM (A)

But

Faire l'acquisition de l'adresse de début et de l'adresse de fond de la partition occupée, de son numéro, et de la taille image mémoire de la tâche appelante.

Syntaxe Assembleur

Format de la requête

```
SVC FREEMA      ('34)
```

Le retour se fait en séquence ; les registres A, B, X, Y ont la signification suivante :

A = taille de l'image-mémoire de l'appelant

B = taille de la partition occupée, (B = 0 pour partition de 64 K)

X = numéro de la partition occupée

Y = adresse de début de la Partition occupée (Format SLO-SLE)

Commentaire

Cette requête n'est jamais refusée ; elle est accessible à toutes les tâches software de l'application ; elle ne doit pas être utilisée par les processeurs foreground.

Remarque

Cette requête est spécifique à RTES16 et porte le n° 52, depuis l'adjonction des requêtes FREEM (n° 51) et FREEM64 (n° 26) qui, compatibles avec les autres logiciels SOLAR, donnent les limites de la zone mémoire libre.

Acquisition des limites de la zone disponible

FREEM

FREEM64

But

Permettre à des processeurs ou des tâches de connaître les limites de la zone mémoire libre par des adresses relatives à SLO.

Syntaxe Assembleur

Format de la requête

SVC FREEM (‘33 : pour les adresses < 32 Kmots)

SVC FREEM64 (‘1A : pour les adresses > 32 Kmots)

Le retour se fait en séquence avec, dans les registres :

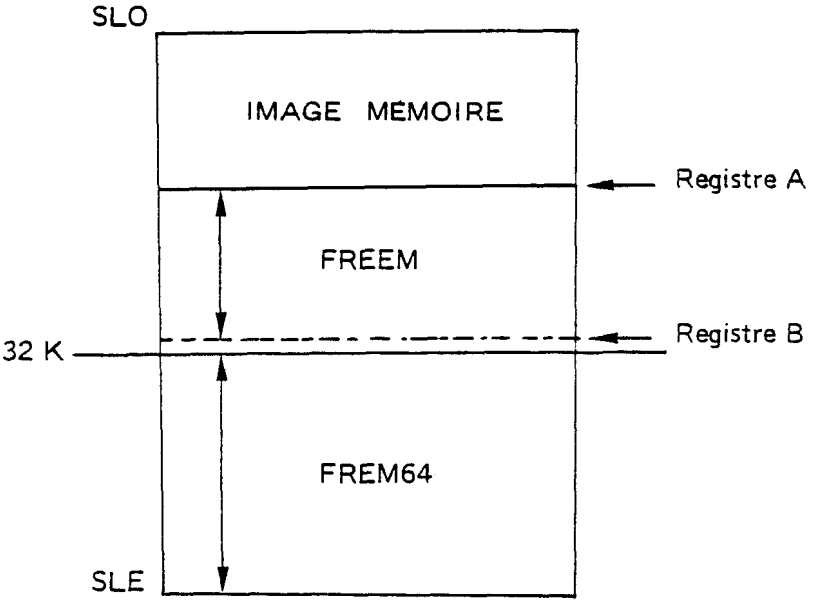
A = @ début mémoire libre

B = @ fin mémoire libre

Commentaire

Ces deux requêtes assurent la compatibilité du moniteur RTES16 avec les autres logiciels SOLAR (en particulier, les langages).

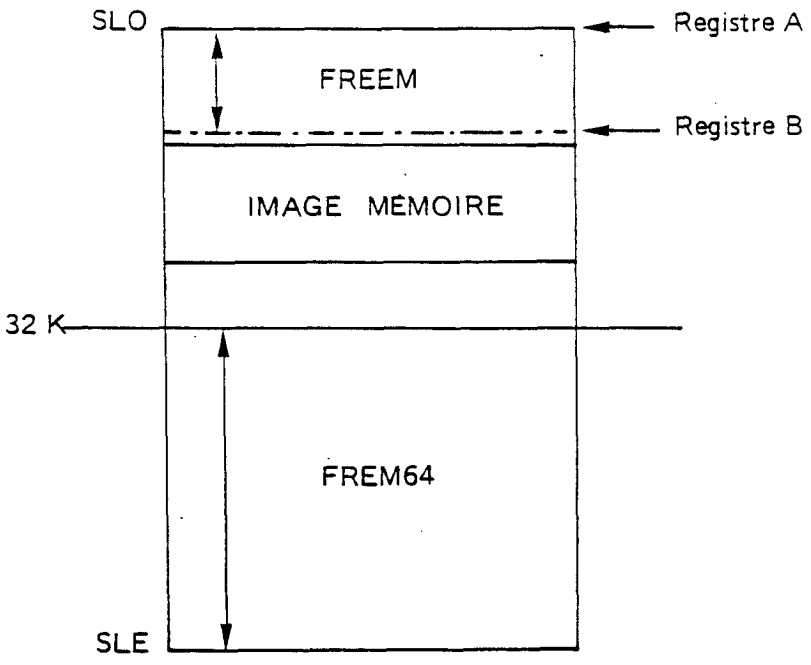
Exemple 1 - Cas adresse d’implantation égale à zéro



Remarques :

- La zone libre se situe derrière l’image mémoire.
- L’adresse de fin est limitée à 32K - 51.
- Si l’image mémoire est supérieure à 32K mots on a zéro dans les registres A et B.

Exemple 2 - Cas adresse d'implantation différente de zéro



Remarques :

- La zone libre se trouve devant l'image mémoire.
- On a dans le registre A zéro et dans B l'adresse d'implantation - 52.

La requête FREM64 ('1A) permet d'obtenir dans les registres A et B les limites de la zone disponible située au delà des 32 K.
En ce qui concerne les images mémoires buildées en mode maître, la requête FREEM donne dans les registres A et B les adresses absolues de début et fin de la zone libre.

Acquisition du mot d'état d'un périphérique (RMDEF)

RMDEF

But

Faire l'acquisition du mot d'état d'une unité physique, combiné à deux indicateurs :

défaut périphérique ou appel opérateur

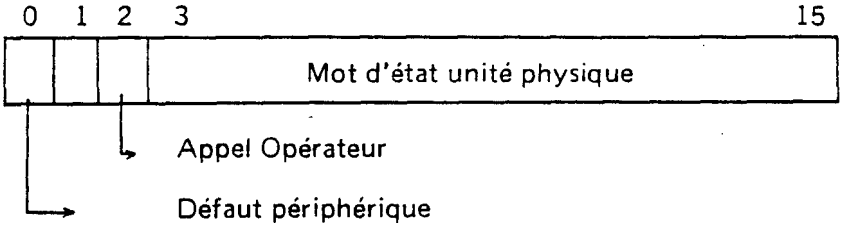
et effacer les défauts ou appels relatifs à cette unité enregistrés par le système mais pas encore traités, par l'intermédiaire d'une requête TAPS par exemple.

Syntaxe Assembleur

Format de la requête

```
LY  NUMFU  « Y = numéro unité fonctionnelle ou numéro
SVC RMDEF  «      d'unité symbolique
```

Le retour se fait en séquence et l'on trouve dans le registre accumulateur le mot d'état de l'unité physique ; ce mot d'état a le format suivant :



On trouvera dans les Manuels d'utilisation de IOCS, la description des mots d'état des périphériques.

Erreurs

Si la valeur du registre Y en entrée de requête ne correspond pas à un numéro de FU ou de SU gérée par IOCS, la valeur de l'accumulateur en retour est égale à 'FFFF (valeur de mot d'état improbable).

Commentaire

Cette requête permet de réaliser la prise en compte et le traitement spécifique d'un défaut ou d'un appel sur les périphériques de l'installation pour lesquels l'utilisateur désire réaliser un traitement différent du traitement standard de RTES16.

Envoi d'un bloc d'information dans une zone système

SEND

But

Transférer un buffer de la tâche en cours dans une zone gérée dynamiquement par le système.

Syntaxe Fortran

CALL SEND (IADR, IERR, ICR)

IADR : tableau entier contenant le bloc d'information (38 mots maximum)

IERR : variable entière chargée par le compte rendu de la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

ICR : variable entière chargée en retour de requête par une clé d'identification de la zone système allouée pour la sauvegarde du bloc.

Syntaxe Assembleur

Format de la requête

```
LAD RPB
SVC SEND
```

Table des paramètres

Adresse du bloc d'information	IADR
Adresse du compte rendu de requête	IERR
Clé d'identification de la zone système réceptrice	ICR

Erreurs

IERR = 6 . il existe au moins un paramètre incorrect
- adresse du RPB
- adresse du compte rendu de requête
- adresse du bloc d'information

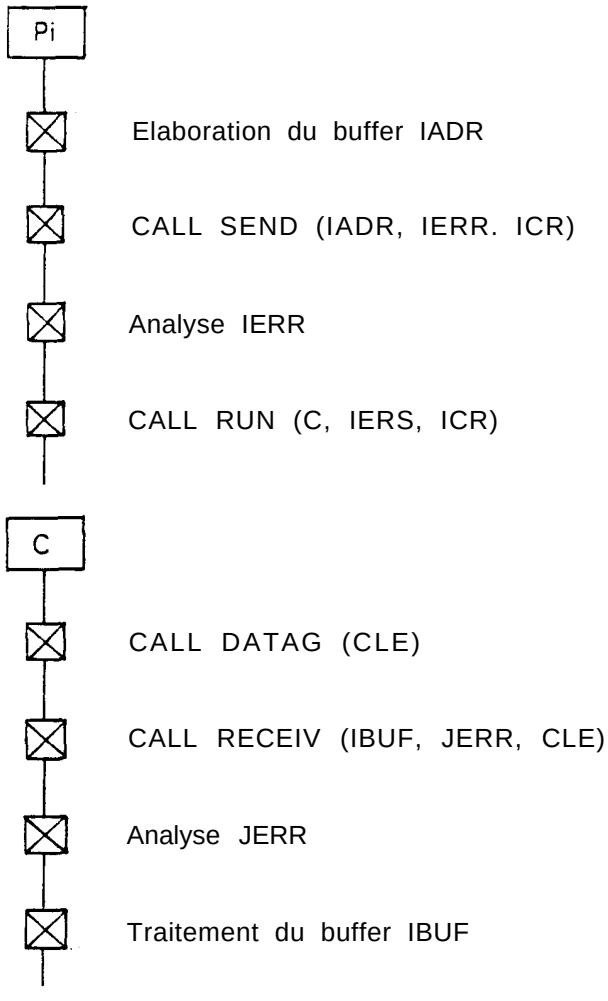
IERR = 7 . système sous dimensionné (zone ZBCT).

Commentaire

Une ou plusieurs tâches "productrices" d'informations peuvent envoyer au système par la requête SEND un bloc de données, puis activer une tâche "consommatrice" (requêtes RUN, START, TCALL...) en donnant en paramètre d'appel IPAR la clé d'identification de la zone système réceptrice du bloc de données ; la tâche appelée (avec ou sans délai, résidente ou non) fait l'acquisitiondu paramètre d'appel (donc de la clé) par la requête DATAG, puis du bloc transmis par la requête RECEIV.

Exemple

Soit Pi une tâche “productrice” et C une tâche “consommatrice” ; l’appel de C par Pi peut être ainsi schématisé :



Réception d'un bloc d'information stocké dans une zone système

RECEIV

But

Transférer dans un buffer de la tâche en cours un bloc d'information antérieurement envoyé dans une zone système par la tâche en cours ou par une autre tâche.

Syntaxe Fortran

CALL RECEIV (IADR, IERR, ICR)

IADR : tableau entier destiné à la réception du bloc d'information (38 mots minimum)

IERR : variable entière chargée par le compte rendu de la requête
IERR = 1 requête acceptée
IERR ⩾ 2 requête refusée

ICR : variable entière représentant la clé d'identification de la zone système allouée lors de la requête SEND antérieure.

Syntaxe Assembleur

Format de la requête

```
LAD RPB
SVC RECEIV
```

Table des paramètres

Adresse du buffer récepteur	IADR
Adresse du compte rendu de requête	IERR
Clé d'identification de la zone système origine	ICR

Erreurs

IERR = 6 . il existe au moins un paramètre incorrect
- adresse du compte rendu de requête
- adresse du RPB
- adresse du buffer récepteur.

Commentaire

L'information transférée par une zone système n'est ni partageable ni réutilisable ; par ailleurs, une zone système allouée par la requête SEND doit être désallouée par la requête RECEIV sous peine de saturation de cette zone système : au niveau de l'application, le nombre de requêtes SEND doit être égal, dans le temps, au nombre de requêtes RECEIV.

Création de tâche :

TASKC

But

Permettre l'exécution parallèle d'un groupe de plusieurs tâches dans une même partition de type résident ou non résident afin d'autoriser avec une performance maximum :

- le partage de données communes (sans passage par la CDA ou la ZDR)
- le partage de programmes réentrants (sans passage par SVC)
- l'accès à des ressources communes (sans passage par SVC)

Syntaxe Fortran

CALL TASKC (NT, IERR, INAM, ISTAT, IUSR, IAPPMAX. IUSRP, IPRTY)

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à créer.

IERR : variable entière chargée par la réponse à la requête :

IERR = 1 requête acceptée

IERR ⩾ 2 requête refusée

INAM : nom d'un programme "EXTERNAL" contenant le code de la tâche à créer

ISTAT : variable ou constante entière qui précise le mode d'exécution de la tâche à créer :

ISTAT = '8000 mode maître

ISTAT = '0000 mode esclave

IUSR : variable ou constante entière qui précise le numéro d'usager privé de la tâche à créer.

IAPPMAX : variable ou constante entière qui précise le nombre maximum d'appels cumulables pour la tâche à créer.

IUSRP : variable ou constante entière qui précise le numéro d'usager public de la tâche à créer.

IPRTY : variable ou constante entière qui précise la priorité de la tâche à créer.

Commentaire

Dans le cas d'utilisation FORTRAN, les bibliothèques FORTRAN non réentrantes doivent être linkéditées dans chaque tâche "fille" à l'aide des commandes ZDEF (LKLOAD) ou ZLNK (EDILE). Se reporter au manuel d'utilisateur de la chaîne de préparation de programmes.

Format de la requête

```
LAD RPB
SVC TASKC
```

Table des paramètres

	N° de tâche à créer	NT
Adresse du compte rendu de requête		IERR
Valeur initiale de RA		
RB		
RX		
RY		
RC		
RL		
RW		
RK		
RP		INAM
RS		ISTAT
Réservé		
Réservé		
0	No Usager privé de la tâche à créer	IUSR
0	Nbre maximum d'appels cumulables de la tâche à créer	IAPPMAX
N° Usager public de la tâche à créer	0	IUSRP
Réservé		
0	Priorité de la tâche à créer	IPRTY

Erreurs

IERR = 2 . numéro de tâche ou priorité incorrects

IERR = 3 . il existe déjà une tâche de même numéro d'appel ou de même même priorité connue du système

IERR = 4 . cette requête est interdite aux tâches hardware

IERR = 6 . le mode d'exécution de la tâche à créer précisé dans le RPB (RS) est différent de celui de l'appelant

IERR = 7 . le système est sous-dimensionné : on ne peut pas enregistrer la requête

Commentaires

. Grâce à cette requête, une tâche de l'application, résidente ou non, peut créer dynamiquement une ou plusieurs tâches sur son code dans la partition qu'elle occupe.

. Le processus de-filiation fonctionne à plusieurs niveaux, c'est-à-dire qu'une tâche créée peut à son tour créer d'autres tâches dans son groupe;

. Les tâches d'un groupe sont activées par les requêtes standard de RTES16 ; elles se synchronisent entre elles ou avec les autres tâches de l'application en utilisant les requêtes standard (événements, sémaphores, horloge...)

. Les tâches créées dynamiquement ne peuvent être tuées que par une autre tâche du groupe (groupe = ensemble des tâches appartenant à la même image mémoire) et uniquement lorsqu'elles sont dans l'état dormant.
Seule, la tâche principale du groupe, (déclarée par la commande TASK) ne peut pas être tuée par la requête KILL.

. Lorsqu'un groupe est implanté dans une partition de type non résident, la partition est libérée en fin d'exécution de la tâche principale (requête EXIT). : II est impératif qu'à cet instant le groupe soit inactif et d'autre part que toutes les tâches créées dynamiquement aient été tuées.

. Seule la tâche principale d'un groupe peut avoir une structure d'overlay.

. Intéraction avec les commandes SAVE et INIT :
La commande SAVE peut être émise après des requêtes TASKC ;
Mais après la commande INIT, il faut forcer l'alimentation de la partition (dans le cas d'une partition non résidente) avant l'activation d'une des tâches créées par requête.

Suppression d'une tâche :

KILL

But

Permettre la destruction d'une tâche d'un groupe de plusieurs tâches s'exécutant en parallèle dans une même partition.

La tâche a été créée par la requête TASKC et doit être inactive (désarmée, inhibée, en erreur) et elle ne doit pas avoir d'appels en attente.

Syntaxe Fortran

```
CALL KILL (NT, IERR)
```

NT : variable ou constante entière qui précise le numéro d'appel de la tâche à «tuer".

IERR : variable entière chargée par la réponse à la requête :

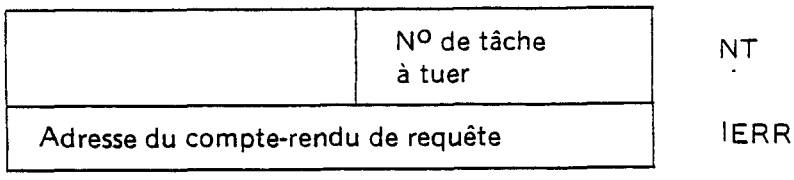
```
IERR = 1 requête acceptée
IERR > 2 requête refusée
```

Syntaxe Assembleur

Format de la requête

```
LAD RPR
SVC KILL
```

Table des paramètres



Erreurs

IERR = 2 . la tâche NT est inconnue du système

IERR = 3 . la tâche NT est une tâche hardware ou une tâche système ou n'appartient pas au même groupe que l'appelant.

IERR = 4 . requête interdite aux tâches hardware

IERR = 5 . la tâche NT est active ou le système a enregistré des appels sur cette tâche.

Demande ou libération d'un pavé utilisateur (GESPAV) :

GESPAV

But

Demandé d'accès à un pavé utilisateur situé dans la zone ZWB.

Les paramètres d'appel se trouvent dans les registres RA et RY.

Syntaxe assembleur

<table>
<tr><th></th><th colspan="2">Demande de pavé</th><th>Libération de pavé</th></tr>
<tr><td rowspan="2">ENTREE</td><td colspan="2">A = 0</td><td rowspan="2">A = 1

Y = @ PAVE</td></tr>
<tr><td>Y = 0

Blocage</td><td>Y = 1
Sans
blocage</td></tr>
<tr><td>SORTIE</td><td colspan="2">A = @ PAVE ou A = 0
si plus de pavé
Y = Taille pavé</td><td>A = 0 @ donnée fausse

A = @ pavé : OK</td></tr>
</table>

Syntaxe Fortran

Demande de pavé

```
CALL DEMPAV (NOBLOQ, ADPAV, TAIPAV, IERR)
```

NOBLOQ : variable ou constante entière nulle si la demande de pavé se fait avec bloquage. égale à un si sans bloquage.

ADPAV : variable entière chargée par l'adresse du pavé si la requête s'est bien effectuée.

TAIPAV : variable entière chargée par la taille du pavé.

IERR

Commentaires

- La zone de pavés étant située en zone maître, la tâche appelante devra obligatoirement être en mode maître pour pouvoir exploiter l'adresse qui lui est rendue (ou à la rigueur en mode privilégié).

- La valeur passée dans le registre RY permet de préciser si l'utilisateur veut être mis en attente ou non dans le cas où il n'y a pas de pavé disponible.

Libération de pavé

CALL LIBPAV (ADPAV, ICR, IERR)

ADPAV : variable entière représentant l'adresse du pavé

ICR : variable entière chargée par le compte rendu de traitement :
si la valeur est nulle, l'adresse donnée est fausse,
sinon ICR est égal à l'adresse du PAVE.

IERR : variable entière chargée par la réponse à la requête
IERR = 1 requête acceptée
IERR = 6 requête refusée

Affectation d'une unité fonctionnelle à une unité symbolique :

SUFU

But

Spécifier l'affectation d'une unité fonctionnelle (FU) à une unité symbolique (SU).

Syntaxe Fortran

CALL SUFU (ISU, IFU, IERR)

ISU : variable ou constante entière égale au rang de la SU

IFU : variable ou constante entière égale au rang de la FU

IERR : variable entièrechargée par la réponse à la requête
- = 1 requête acceptée
- ⩾ 2 requête refusée

Syntaxe Assembleur

Format de la requête

```
LAD RPB
SVC SUFU
```

Table des paramètres

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	
0	Numéro de SU	ISU
1	Adresse de compte rendu de requête	IERR
2	Numéro de FU	IFU

Erreurs

IERR 2 - SU inexistante
- FU inexistante ou non gérée par IOCS

IERR 6 Il existe au moins un paramètre incorrect :
- adresse de RPB
- adresse de compte rendu
- ISU ou IFU n'est pas compris entre 0 et 127

IERR 11 L'affectation est réputée impossible.

Commentaires

1 Les FU de rang compris entre 0 et '39 sont mnémoniques.
Si la FU précisée dans la requête SUFU n'est pas mnémonique, l'affectation est acceptée. Dans le cas où la FU est mnémonique, l'affectation peut être refusée avec un compte rendu IERR = 11. Pour connaître les affectations autorisées, se reporter au § 3 de l'annexe du manuel de référence intitulé : "Unités symboliques et fonctionnelles de IOCS".

2 Si un fichier était auparavant affecté à la SU et à elle seule, il est fermé.

RETOUR

But	Fin d'exécution d'une requête qui s'exécute en mode privilégié (RETOUR).
Paramètre	aucun.
Remarque	La valeur du registre K doit obligatoirement être la même qu'à l'entrée dans le programme privilégié.
	Une requête privilégiée doit obligatoirement se terminer par la SVC Retour (voir exemple dans le manuel d'utilisation).

3.1.7 - Prise en compte du coupleur multifonctions

3.1.7.1 - Introduction

Le coupleur multifonctions offre plusieurs services qui favorisent l'utilisation du SOLAR dans les domaines des automatismes, du contrôle de procédé et de l'instrumentation.

- Fonctions Horloge
  - . Gestion de l'heure et de la date (sauvegardées par batterie)
  - . Synchronisation inter-calculateurs pour assurer une base de temps identique à tous les calculateurs.

- Fonctions Temps Réel
  - . Gestion de 8 réveils programmables
  - . Gestion de 128 temporisations programmables
  - . Gestion d'un chien de garde
  - . Gestion de deux entrées de surveillance "Tout ou Rien".

Ces services sont accessibles par l'intermédiaire de requêtes adressées au moniteur d'entrées-sorties IOCS16, dirigées vers le driver DRVCMF (voir Manuel d'Exploitation du coupleur multifonctions).

D'autre part, sous le système RTES16, leur utilisation est facilitée :

- par la prise en compte des fonctions horloge par le système (initialisation de la date et de l'heure, évolution de la date et de l'heure, évolution de la date et de l'heure indépendamment des arrêts du calculateur et des coupures secteur, édition de la durée d'une coupure secteur),

- par des sous-programmes des bibliothèques BIBRTE permettant d'accéder d'une façon simple aux fonctions temps réel du coupleur, depuis des programmes utilisateurs écrits en langages évolués (PL16, FORTRAN, FORTRAN77, PASCAL).

3.1.7.2 - Fonctions temps réel du coupleur multifonctions

Caractéristiques de la fonction Réveil

La fonction Réveil du CMF permet de gérer des heures d'échéance identifiées par un numéro.

Ce numéro de réveil peut aller :

- de 0 à 3 si l'heure est définie en DELAI (de 20 ms à 1,26 s en multiples de 20 ms).

- de 0 à 7 si l'échéance est définie en HEURE ABSOLUE (de 0 à 24 heures).

Précision des réveils $\leqslant 10\ E{-}4$

Caractéristiques de la fonction Temporisation

La fonction Temporisation du CMF est capable de gérer 128 temporisations identifiées par un numéro de 0 à 127, dont 50 peuvent être armées simultanément.

La durée d'une temporisation, exprimée en multiples d'une période, peut être comprise entre 10 ms et 45 H 30 mn 37,5 s, grâce à la disponibilité d'un large éventail de périodes différentes.

Précision des temporisations ⩽ 10 E-4

Caractéristiques du Chien de Garde

- Armement avec une durée comprise entre 20 ms et 10 mn 55 s 340 ms, exprimée en nombre de périodes de 20 ms.

— Précision ⩽ 10 E-4

- Ouverture d'un contact sur défaut (panne coupleur, équipement hors tension) ou à échéance de la durée.

- Pouvoir de coupure : 60 V max., 200 mA max., 5 Watts max.

— Isolement ⩽ 700 V continu ou crête alternatif

- En option (bandeau de visualisation), sortie du chien de garde sur relais de puissance 220V/1,5A (disponibilité des 2 états "Fermé" et " Ouvert").

Principe de fonctionnement

- Utilisation en chien de garde :

A la mise sous tension de l'équipement ou à l'initialisation du système, le contact est fermé (état repos).
Le logiciel peut armer, réarmer et désarmer le chien de garde dont il a programmé la durée T
Si au bout du temps T après un armement ou un réarmement il n'y a pas eu une nouvelle commande de réarmement ou de désarmement, ou si survient un défaut matériel (panne coupleur, équipement hors tension), le contact s'ouvre (état actif).

Pour éventuellement refermer le contact (état repos), il faut :

. soit remettre l'équipement sous tension,

. soit émettre une commande de désarmement (un armement ne suffit pas).

- Utilisation en sortie Tout ou Rien :

Le chien de garde peut être utilisé comme un relais délivrant une sortie Tout ou Rien :

. Etat Actif : "armement" avec un délai nul (en réalité, le changement d'état survient dans un délai maximum de 20 ms).

. Etat Repos : "désarmement" (immédiat).

Caractéristiques des Entrées de Surveillance

Le coupleur multifonctions dispose de 2 entrées de surveillance Tout ou Rien ETOR1 et ETOR2.

— Isolement ⩽ 700 V crête alternatif ou continu

- Consommation = 8 mA typiques
- Tension d'alimentation 24V ± 20%
- Détection par le coupleur des changements d'état (la lecture de l'état lui-même n'est pas possible).

3.1.7.3 - Utilisation des fonctions temps réel du coupleur multifonctions

- Les services offerts par le driver DRVCMF sur échéance d'un réveil, d'une temporisation, ou sur changement d'état d'une entrée de surveillance sont :

  . armement d'une tâche
  . positionnement d'un événement
  . activation d'un sémaphore privé
  . branchement à une adresse.

  Les services relatifs à la gestion du chien de garde sont :

  . initialisation de la durée avec ou sans armement
  . armement/réarmement
  . désarmement.

  Ces services sont accessibles par des requêtes adressées directement au moniteur d'entrées-sorties IOCS16 (voir Manuel d'Exploitation du coupleur multifonctions pour plus de détails).

- Les services "systèmes" offerts par les bibliothèques BIBRTE sont :

  . le positionnement d'événements sur échéance de réveil, de temporisation ou changement d'état des entrées de surveillance,

  . initialisation avec armement, réarmement, désarmement, pour le chien de garde,

  . "collage", désarmement, pour le chien de garde utilisé en sortie Tout ou Rien.

— L'utilisation des fonctions Réveil, Temporisation, Entrée de surveillance comporte trois phases :

  . une première de DEFINITION qui consiste à affecter un numéro d'événement à un numéro de réveil, de temporisation ou d'entrée de surveillance,

  . une seconde phase d'EXPLOITATION correspondant aux commandes d'initialisation, d'armement des fonctions et d'exploitation des événements positionnés sur les échéances ou les changements d'état,

  . enfin, une phase de LIBERATION des ressources.

— La (ré)initialisation des événements gérés par les fonctions temps réel est laissée à la charge de l'utilisateur.

— Les fonctions Réveil, Temporisation, Entrée de surveillance peuvent être exploitées sous deux environnements :

. environnement BASIC,

. environnement STANDARD.

Le fonctionnement sous un environnement ou l'autre est déterminé dans la phase de DEFINITION.
Le fonctionnement sous l'environnement STANDARD requiert les ressources de façon sélective, au fur et à mesure de leur définition et de leur libération.
Le fonctionnement sous l'environnement BASIC requiert globalement dans la phase de définition un certain nombre de ressources :

. 4 réveils [0, 1, 4, 5]

. 32 temporisations [0 à 31]

. les 2 entrées de surveillance.

3.1.7.4 - Sous-programmes BIBRTE d'interface des fonctions temps réel du coupleur multi-fonctions

Généralités

- Ces sous-programmes ne correspondent pas à des requêtes programmées du système RTES16, mais interfacent directement le moniteur d'entrées-sorties IOCS16. L'unique syntaxe d'appel est donc la syntaxe type "Langages Evolués", dite syntaxe FORTRAN.

. Syntaxe FORTRAN

CALL SP (IPAR1, IPAR2,..., IPARn)

. Syntaxe PL16 dérivée de la syntaxe FORTRAN

CALL SP (@IPARl, @IPAR2,..., @IPARn, RA : = n)

- Compte rendu (IERR) :

. Quand la fonction demandée a pu être exécutée normalement le compte rendu délivré dans la variable entière IERR est égal à 1.

. Quand la fonction demandée n'a pas pu être exécutée (erreur de syntaxe, erreur de paramètres, défaut matériel), le compte rendu IERK est différent de 1.
Aucun message n'est édité sur l'unité symbolique TE, la tâche appelante n'est pas suspendue et reprend le contrôle en séquence. Elle doit donc gérer elle-même le compte rendu d'exécution. Sur une erreur de syntaxe (nombre de paramètres incorrect), aucun compte rendu n'est délivré dans la variable entière IERR. L'utilisateur doit donc l'initialiser avant l'appel avec une valeur différente de celles utilisées par les sous-programmes, pour pouvoir la contrôler (- 1 par exemple).

- Contrainte de génération d'IOCS16

Une partie du traitement des fonctions Réveil, Temporisation et Entrée de surveillance s'effectuant sous niveau matériel, il faut configurer une pile d'au moins 45 mots pour le niveau sous lequel est définie la FU temps réel :

```
%NIVEAU ? KSTOR = 45
```

Description des sous-programmes :

Libération BASIC :

FBASIC

But

Libérer les ressources requises par le fonctionnement sous l'environnement BASIC.

Syntaxe Fortran

CALL FBASIC (IERR)

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

Erreurs

IERR = 2 - FU Temps Réel non générée pour un fonctionnement sous BASIC

IERR = 5 - Tous les réveils utilisés par l'environnement BASIC ne sont pas échus
- Toutes les temporisations utilisées par l'environnement BASIC ne sont pas désarmées.

Commentaires

FBASIC est utilisée quand des programmes utilisateurs fonctionnant sous l'environnement STANDARD et BASIC peuvent s'exécuter concurrentiellement sur le même équipement et que les programmes BASIC ne libèrent pas les ressources temps réel en fin d'exécution.

Définition d'un réveil :

DCLOCK

But

Affecter à un numéro de réveil un numéro d'événement qui sera positionné sur l'échéance du réveil.

Syntaxe Fortran

CALL DCLOCK (NREV, IEV, IERR, NFU)

NREV : variable ou constante entière représentant le numéro de réveil

IEV : variable ou constante entière représentant le numéro d'événement

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisée à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
. Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Réveil non libre (requis par l'environnement BASIC)
. Réveil actif (initialisé non échu)

IERR = 6 . Il existe au moins un paramètre incorrect
— NREV ϵ [0, 7]
— IEV ϵ [0, 255]
— NFU ϵ [1, 125]

Libération d'un réveil :

FCLOCK

But

Libérer un réveil défini sous l'environnement standard

Syntaxe Fortran

CALL FCLOCK (NREV, IERR, NFU)

NREV : variable ou constante entière représentant le numéro de réveil

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
. Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Réveil actif (initialisé non échu)

IERR = 6 . Il existe au moins un paramètre incorrect
— NREV ∈ [0, 7]
— NFU ∈ [1, 125]

Initialisation d'un réveil :

CLOCK

FORME 1

But

Initialisation d'un réveil par une heure d'échéance absolue

Syntaxe Fortran

CALL CLOCK (NREV, IH, IM, IS, IMS. IERR, NFU)

NREV : variable ou constante entière représentant le numéro du réveil (0 à 7)

IH, IM, IS, IMS : variables ou constantes entières représentant l'heure d'échéance exprimée en heures, minutes, secondes et milli-secondes

IERR variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Défaut matériel

IERR = 6 . Il existe au moins un paramètre incorrect

- NREV ϵ [0, 7]
- IH ϵ [0, 23]
- IM ϵ [0, 59]
- IS ϵ [0, 59]
- IMS ϵ [0, 999]
- NFU ϵ [1, 125]

FORME 2

But

Initialisation d'un réveil par une heure d'échéance en délai.

Syntaxe Fortran

CALL CLOCK (NREV, NTOP, IERR, NFU)

NREV : variable ou constante entière représentant le numéro de réveil (0 à 3)

NTOP : variable ou constante entière représentant le délai exprimé en nombre de périodes de 20 ms

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR =2 . Numéro de FU non géré par IOCS16
. Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Défaut matériel

IERR = 6 . Il existe au moins un paramètre incorrect
— NREV ϵ [0, 3]
— NTOP ϵ [0, 63]
— NFU ϵ [1, 125]

Définition d'une temporisation :

DTIMER

But

Affecter à un numéro de temporisation un numéro d'événement qui sera positionné à-l'échéance de la temporisation.

Syntaxe Fortran

CALL DTIMER (NTIMER, IEV, IERR. NFU)

NTIMER — variable ou constante entière représentant le numéro de la temporisation

IEV — variable ou constante entière représentant le numéro d'événement

IERR — variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU — variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisée à la génération d'IOCS16

Erreurs

IERR = 2 — Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 — Temporisation non libre (requise par l'environnement BASIC)
Temporisation active (initialisée non désarmée)

IERR = 6 — Il existe au moins un paramètre incorrect
— NTIMER ϵ [0, 127]
— IEV ϵ [0, 255]
— NFU ϵ [1, 125]

Libération d'une temporisation :

FTIMER

But

Libérer une temporisation définie sous l'environnement standard

Syntaxe Fortran

CALL FTIMER (NTIMER, IERR, NFU)

NTIMER : variable ou constante entière représentant le numéro de la temporisation

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
. Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Temporisation active (initialisée non désarmée)

IERR = 6 . Il existe au moins un paramètre incorrect
— NTIMER ϵ [0, 127]
— NFU ϵ [1, 125]

initialisation (et armement) de temporisation(s) :

TIMER

FORME 1

But

Initialisation et armement d'une temporisation, cyclique ou non

Syntaxe Fortran

CALL TIMER (NTIMER, IPER, NP, IREP, IERR, NFU)

NTIMER : variable ou constante entière représentant le numéro de la temporisation

IPER : variable ou constante entière qui précise la période de base dans laquelle est exprimée la durée (IP*NP) de la temporisation

0 pour	10 ms	4 pour	150 ms	8 pour	500 ms	12 pour	1 s
1	20 ms	5	200 ms	9	600 ms	13	1,5 s
2	50 ms	6	250 ms	10	700 ms	14	2 s
3	100 ms	7	400 ms	11	800 ms	15	2,5 s

NP : variable ou constante entière représentant la durée de la temporisation, exprimée en nombre de périodes de base (0 à 65 535)

IREP : variable ou constante entière représentant la répétitivité de la temporisation

IR = 0 la temporisation est automatiquement réarmée indéfiniment

IR = n , n = [1, 15] la temporisation est automatiquement réarmée n - 1 fois

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement

IERR > 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'lOCS16.

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Défaut matériel

IERR = 6 . Il existe au moins un paramètre incorrect

- NTIMER ϵ [0, 127]
- IPER ϵ [0, 15]
- IREP ϵ [0, 15]
- NFU ϵ [1, 125]

FORME 2

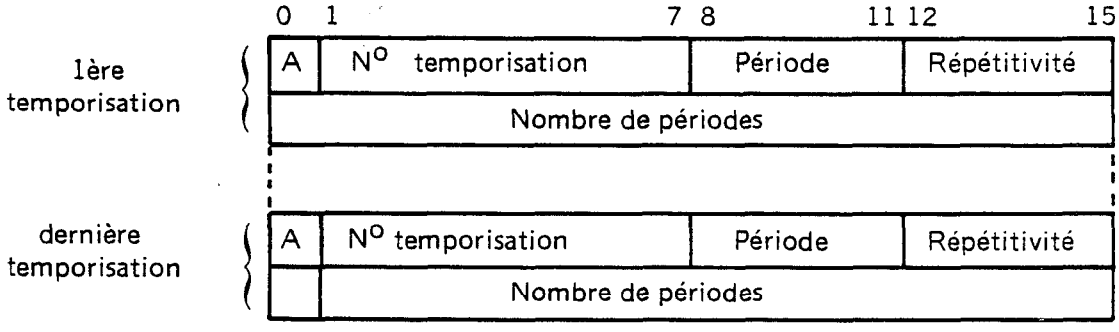
But

Initialisation simultanée, avec ou sans armement, de plusieurs temporisations, cycliques ou non

Syntaxe Fortran

CALL TIMER (ISUF, NOMBRE, IERR, NFU)

IBUF : tableau entier 2*NOMBRE mots contenant les paramètres des temporisations



A: Armement = 1 immédiat
= 0 différé
N° temporisation : voir paramètre NTIMER de FORME1
Période : voir paramètre IPER de FORME1
Répétitivité : voir paramètre IREP de FORME1
Nombre de périodes : voir paramètre NP de FORME1

NOMBRE : variable ou constante entière représentant le nombre de temporisations à initialiser (1 à 128)

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Défaut matériel

IERR = 6 . Il existe au moins un paramètre incorrect
- IBUF n'est pas un tableau
— NOMBRE ϵ [1, 128]
— NFU ϵ [1, 125]

Arrêt, armement ou réarmement d'une temporisation :

RTIMER

But

Arrêter, armer ou réarmer une temporisation initialisée

Syntaxe Fortran

CALL RTIMER (NTIMER. IA. IERR, NFL)

NTIMER : variable ou constante entière représentant le numéro de la temporisation

IA : variable ou constante entière précisant la fonction

IA= 0 arrêt de la temporisation
IA= 1 armement ou réarmement de la temporisation

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Défaut matériel

IERR = 6 . Il existe au moins un paramètre incorrect

- NTIMER ϵ [0, 127]
- IA ϵ [0, 1]
- NFU ϵ [1, 125]

Définition d'une entrée de surveillance :

DETOR

But

Affecter à un numéro d'entrée de surveillance Tout ou Rien un numéro d'événement qui sera positionné sur le changement d'état de l'entrée de surveillance.

Syntaxe Fortran

```
CALL DETOR (NETOR, IEV, IERR, NFU)
```

NETOR : variable ou constante entière représentant le numéro de l'entrée de surveillance

IEV : variable ou constante entière représentant le numéro de l'événement

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 5 . Entrée de surveillance non libre (requise par l'environnement BASIC)

IERR = 6 . Il existe au moins un paramètre incorrect

- NETOR ϵ [0, 1]
- IEV ϵ [0, 256]
- NFU ϵ [1, 125]

Libération d'une entrée de surveillance :

FETOR

But

Libérer une entrée de surveillance Tout ou Rien définie sous l'environnement standard

Syntaxe Fortran

```
CALL FETOR (NETOR, IERR, NFU)
```

NETOR : variable ou constante entière représentant le numéro de l'entrée de surveillance

IERR : variable entière chargée par le compte rendu d'exécution

IERR = 1 fonction exécutée normalement
IERR ⩾ 2 fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
. Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 6 . Il existe au moins un paramètre incorrect
— NETOR ϵ [0, 1]
— NFU ϵ [1, 125]

Initialisation, armement, réarmement et désarmement du chien de garde :

RELAY

But

Initialiser et armer pour une certaine durée, ou réarmer, ou désarmer le chien de garde

Syntaxe Fortran

```
CALL RELAY ([ID,] IERR, NFU)
```

ID : variable ou constante entière précisant la fonction demandée et le délai

- ID = -1 : désarmement (et passage à l'état repos) du chien de garde
- ID = 0 : passage immédiat (dans les 20 ms suivantes) à l'état actif
- ID = 1 à 32767 : initialisation pour la durée exprimée en nombre de tops de 20 ms, et armement
- paramètre ID absent : réarmement du chien de garde

IERR : variable entière chargée par le compte rendu d'exécution

- IERR = 1 : fonction exécutée normalement
- IERR ⩾ 2 : fonction refusée

NFU : variable ou constante entière représentant le numéro de l'unité fonctionnelle Temps Réel précisé à la génération d'IOCS16

Erreurs

IERR = 2 . Numéro de FU non géré par IOCS16
Numéro de FU ne correspondant pas à la FU Temps Réel

IERR = 6 . Il existe au moins un paramètre incorrect

- ID ϵ [-1, 32767]
- NFU ϵ [1, 125]

3.2 - LES COMMANDES DE L'OPERATEUR

3.2.1 - Principe

Lorsque l'opérateur veut communiquer avec RTES16, il le fait par l'intermédiaire de "commandes" qu'il émet depuis l'un des dispositifs périphériques d'entrée affecté à ces commandes : le clavier du téléimprimeur en général.

Une commande est constituée d'un enregistrement lu sur le périphérique associé à l'unité symbolique "DIALOGUE INPUT" (DI).

Cet enregistrement sera en général une ligne de téléimprimeur (délimitée par le caractère "retour chariot" (cr) ou une carte perforée.

Une commande peut demander :

- soit l'affectation d'une unité fonctionnelle à une unité symbolique (cf BOS et IOCS)
- soit la réalisation d'une fonction spécifique du système RTES16 (activation d'une tâche impression de la date et de l'heure...).

Le système indique qu'il est en attente d'une commande en écrivant sur le périphérique associé à l'unité symbolique "DIALOGUE OUTPUT" (DO) le message : (cr) (lf) *

Sur SOLAR 16, le coupleur du téléimprimeur émet un défaut "time-out" lorsque, au bout d'une vingtaine de secondes après l'initialisation d'une opération d'entrée, aucun caractère n'a été frappe. Au niveau du système cela se traduit par l'impression du message : "ERS 08 TIME-OUT". Pour relancer le dialogue, l'utilisateur doit alors appuyer sur le bouton "BREAK" du téléimprimeur (voir paragraphe 3.2.3).

Si les SU DI et DO ne sont pas affectées au même périphérique, la commande est recopiée sur la SU DO.
Les caractères non alphanumériques ne seront éliminés qu'après cette recopie, ce qui permet par exemple d'utiliser les attributs vidéo des consoles.

3.2.2 - Conventions d'écriture des commandes

— Le caractère «flèche arrière» (←) (ou «souligné» (_)) ▲ provoque l'annulation d'un début de commande erronée. Les caractères (↑) (ou intersection (^)) et backspace (BS) servent à annuler le caractère significatif précédent. Cette annulation peut être multiple.

Exemple :

```
LOUL ——— COUCOU   équivaut à COUCOU
ME␣VIO ↑↑ OILA   équivaut à ME␣VOILA
```

- Une seule commande peut se trouver dans un enregistrement. Une commande se présente sous la forme d'une clé composée avec les 4 premiers caractères significatifs de l'enregistrement, éventuellement suivie d'une liste de paramètres, chacun précédé d'un caractère séparateur (,) ou (/) ou espace.

Exemple :

```
TIME              commande = clé
IRUN,14,2         commande = clé, p1, p2
 IRUN  14  2
```

Remarque :

Une suite de plusieurs espaces consécutifs est analysée par le système comme un seul espace. Dans une commande où certains paramètres sont omis, il n'est donc pas possible de les remplacer par une suite d'espaces. Dans ce cas, il faut utiliser la virgule (",") ou le slash ("/").

- La touche DEL ou Rub Out (ASCII 'FF) puis Retour Chariot provoque le réaffichage de la dernière commande rentrée.
L'utilisateur peut alors la modifier avant de la réexécuter. Le caractère DEL doit être le 1er caractère de la commande.

Exemple :

DRUN␣25␣␣5␣␣2 est analysée comme
DRUN␣25␣5␣2

et n'est pas équivalent à :

DRUN, 25, , 5, 2

- Certaines commandes n'ont aucun paramètre : c'est le cas par exemple des commandes d'affectation d'une unité fonctionnelle à une unité symbolique.

- Il est possible d'introduire des commentaires dans une commande, à l'aide du caractère "!". Tous les caractères situés entre le point d'exclamation et le retour chariot de fin de commande seront pris comme étant des commentaires.

Les autres commandes ont des paramètres obligatoires ou optionnels : c'est le cas de certaines commandes permettant la réalisation de fonctions spécifiques de RTES16.

Le lecteur trouvera, dans le descriptif de chacune de ces commandes, un paragraphe «signification des paramètres,) : les paramètres obligatoires y sont mis en relief (NT) ; les autres sont optionnels et chacun d'eux est suivi d'un commentaire également mis en relief qui précise le sens donné à leur absence.

Exemple : on trouvera dans le descriptif de la commande IRUN, le paragraphe suivant :

- Signification des paramètres
- NT : numéro d'appel de tâche
- PAR : paramètre de travail (nul par défaut).

En règle générale, les paramètres associés à une commande doivent être des nombres décimaux ou hexadécimaux, indifféremment.
Les cas particuliers qui s'écartent de cette règle sont spécifiés explicitement.

NT : Dans toutes les commandes où le NT apparaît, (sauf KILL), on peut frapper le nom du fichier disque (sans catalogue) associé à la tâche à la place du numéro de tâche.

Exemple :

IRUN, TOVER, 1

est équivalent à

IRUN, 25, 1

3.2.3 - L'appel opérateur

L'opérateur peut communiquer avec le système même si celui-ci n'est pas en attente d'une commande. Il suffit pour cela qu'il appuie sur la touche "BREAK" du téléimprimeur : l'exécution ultérieure de la requête SVC TAPS dans une tâche de l'application ou dans la tâche la moins prioritaire provoquera l'impression du message :

ERS 03 (cr) (lf)

sur le périphérique associé à l'unité symbolique "SYSTEM ALARMS" (SA) et le système se mettra en attente d'une commande de l'opérateur sur le périphérique associé à l'unité symbolique DI (Dialogue Input).
Cependant, si les programmes ne testent pas les appels de l'opérateur par la requête SVC TAPS, il existe quand même une possibilité de les interrompre en faisant "quatre appels" consécutifs (en appuyant 4 fois de suite sur la touche "BREAK").

Le système imprimera de la même façon le message :

ERS 02 (cr) (lf)

sur le périphérique associé à l'unité symbolique SA et se mettra en attente d'une commande sur Dl.
Dans les deux cas, le programme interrompu ne sera pas suspendu, le jeu des commandes opérateur de RTES16 permettant d'agir sur le système d'une façon aussi précise que possible.
Dans les deux cas également, le système réalisera la réaffectation standard des deux SU : Dl (TK) et DO (TS) avant la demande d'entrée de commande sur Dl.

3.2.4 - Commandes d'affectation aux Unités Symboliques

Ces commandes permettent de spécifier l'association d'une unité fonctionnelle type ou d'un fichier disque à une unité symbolique type. Une telle affectation restera valable jusqu'à la prochaine commande d'affectation ou la prochaine commande de fermeture de fichier portant sur la même unité symbolique. Cependant, dans certains cas (appel opérateur, erreur de commande, fichier affecté à une SU du dialogue devenant inaccessible) le système peut réaliser une affectation standard de certaines unités symboliques.

3.2.4.1 - Commandes d'affectation de périphériques aux unités symboliques types

Ces commandes permettent de spécifier l'association d'un périphérique ou plus précisément d'une unité fonctionnelle type, à une unité symbolique type.
Ces commandes sont toujours composées de la juxtaposition du nom d'unité symbolique type et du nom d'unité fonctionnelle type qu'on veut lui affecter.

SU FU

Si la commande d'affectation est acceptée, le système imprime le message :

(cr) (lf)
*

et passe en attente d'une nouvelle commande.

Commentaire : si un fichier était affecté à l'unité symbolique et à elle seule, il est fermé.

Si la commande n'est pas acceptée, le système imprime sur le périphérique associé.

à l'unité symbolique DØ :

ERC 11 pour indiquer que l'affectation est réputée impossible

ERC 02 si le n° de FU précisé dans la commande est inconnu du système.

L'opérateur et le système disposent de 4 unités symboliques spécialisées pour dialoguer :

- Dl (DIALOGUE INPUT) pour entrée de commandes
- DO (DIALOGUE OUTPUT) pour réponse aux commandes
- SA (SYSTEM ALARMS) pour sortie de messages d'alarme
- TE (TASK ERRORS) pour sortie de messages d'erreur.

Lorsqu'on initialise le système, celui-ci affecte de façon standard les unités fonctionnelles aux unités symboliques.

On donne en annexe un synoptique des affectations standard et des affectations possibles au niveau dialogue opérateur.

3.2.4.2 - Commandes d'affectation de fichiers aux unités symboliques et de fermeture de fichiers

```
SU [ art. ] nomfic [ - catg ] [ , FU / SU' ]
```

But

Affecter [un article ou] un fichier à l'unité symbolique.
Si un autre fichier était auparavant affecté à cette unité symbolique et à elle seule, il est fermé.

Signification des paramètres

SU : nom de l'unité symbolique à laquelle sera affecté l'article ou le fichier

art : désigne le nom d'un article du fichier qui est alors de type indexé (par défaut, le fichier est de type séquentiel)

nomfic : nom du fichier : 6 caractères maximum
(caractères alphanumériques et : < ;)

catg : nom du catalogue : 2 caractères maximum
(par défaut, nom d'un catalogue commun identifié par 2 caractères NUL)

FU / SU' : nom d'unité fonctionnelle ou symbolique désignant le support disque sur lequel se trouve le fichier
Par défaut, c'est la FU disque affectée à l'unité symbolique BI

Commentaires

1) Fichier ou article spécifié dans la commande inexistant :

- si la SU est dédiée aux entrées (SI, BI, CC, EC, PC, DI) le système refuse de lui affecter un fichier ou un article inexistant,
- sinon (SO, BO, LL, LO, EL, DO, TE, SA, U1 à UF)
  . quand le fichier n'existe pas et que la commande ne spécifie pas un nom d'article, le fichier (de type séquentiel) est créé
  . quand le fichier n'existe pas et que la commande spécifie un nom d'article, le fichier (de type indexé) et l'article sont créés
  . quand le fichier existe mais que la commande spécifie un nom d'article inexistant, la commande est acceptée dans la mesure où le fichier est de type indexé, et l'article est créé.

2) Après une commande acceptée, le fichier est ouvert et le pointeur courant est positionné en début de fichier (ou d'article).

Erreurs

ERC 02 : le nom de l'unité fonctionnelle support du fichier (,FU) désigne une FU inconnue du système

ERC 02 '600C : fichier inexistant
ERC 02 '600E : article inexistant
} dans le cas d'une SU dédiée aux entrées

ERC 07 : on ne peut pas enregistrer la commande :
le système est sous-dimensionné :
- zone des descripteurs de fichiers saturée (ZDF)

ERC 08 : syntaxe incorrecte

ERC 13 : le paramètre FU ou SU' précisé dans la commande (ou par défaut l'unité symbolique BI) ne correspond pas à une FU disque gérée par FMS.

ERC 15 'yyyy : demande d'accès au fichier rejetée par FMS avec 'yyyy = compte rendu FMS

ERC 16 'yyyy : erreur FMS lors du close du fichier qui était éventuellement affecté auparavant à la SU.
la nouvelle affectation est effective.

CLOSE SU

But

Fermer le fichier affecté à l'unité symbolique désignée par le paramètre SU (CLOSE)

Signification des paramètres

su : nom de l'unité symbolique

Commentaires

L'unité symbolique retrouve son affectation standard, ainsi que les autres unités symboliques auxquelles pouvait être affecté simultanément le fichier.

Erreurs

ERC 03 : aucun fichier n'est affecté à l'unité symbolique
la commande est ineffective

ERC 16'yyyy : demande rejetée par FMS
avec 'yyyy = compte rendu FMS
l'affectation standard est effective.

3.2.4.3 - Affectation à une unité symbolique du fichier ou de l'unité fonctionnelle affecté à une autre SU

But

Affecter à une unité symbolique le fichier déjà affecté à une autre unité symbolique.

L'affectation d'une unité fonctionnelle affectée à une autre unité symbolique est aussi permise.

Format

```
SU SU'
```

Signification des paramètres

SU : nom de l'unité symbolique pour laquelle est réalisée la nouvelle affectation

SU' : nom d'une unité symbolique dont on veut copier l'affectation.

Commentaires

1) Dans le but de protéger le fonctionnement du système des erreurs d'affectations (exemple : destruction du fichier de commandes affecté à D1 dans le cas où on associerait ce même fichier à une SU dédiée aux sorties de messages) certaines règles d'utilisation sont contrôlées et peuvent conduire à refuser la commande.

Les unités symboliques peuvent être classées selon 2 critères :

- critère Entrée/Sortie, 3 classes :
  SU dédiées aux entrées SI BI CC EC PC DI
  SU dédiées aux sorties SO BO LL LO EL DO TE SA
  SU non dédiées U1 à UF
- critère d'utilisation par le dialogue système, 2 classes :
  DO DI SA TE
  toutes les autres.

Le système refuse les commandes SU SU' :

- amenant à une affectation de FU non permise
  (voir en annexe le synoptique des affectations possibles)

  exemple :
  ```
  U1 TS
  CC U1 (commande refusée)
  ERC 11
  ```
- affectant un fichier simultanément à une SU dédiée aux entrées et à une SU dédiée aux sorties

  exemples :
  ```
  LO TOTO-TO
  CC LO (commande refusée)
  ERC 03
  LO TATA-TA
  U1 LO
  LL U1 (commande permise)
  CC U1 (commande refusée)
  ERC 03
  U1 TOTO-TO
  CC U1 (commande permise)
  EC U1 (commande permise)
  U2 U1 (commande permise)
  LO U2 (commande permise)
  ERC 03
  ```

- affectant un fichier simultanément à une SU non dédiée au sens Entrée/Sortie et à une SU du dialogue

exemple :

```
U 1  T K
D1  U1  (commande  permise)
U2  TOTO-TO
D1  U2  (commande  refusée)
ERC 05

U1  TOTO-TO
EL  U1  (commande  permise)
LL  EL  (commande  permise)
SA  EL  (commande  refusée)
ERC 05
```

2) Si un fichier était affecté à l'unité symbolique bénéficiant de la nouvelle affectation et à elle seule, il est fermé.

Erreurs

ERC 03	affectation à une SU dédiée aux entrées (respectivement Sorties) d'un fichier déjà affecté à une SU dédiée aux Sorties (respectivement Entrées).
ERC 05	affectation simultanée d'un fichier à une SU non dédiée au sens Entrée/Sortie et à une SU du dialogue.
ERC 08	syntaxe incorrecte
ERC 11 :	affectation de FU réputée impossible
ERC 16 'hhhh :	erreur FMS lors du close du fichier qui était éventuellement affecté auparavant à la SU ; la nouvelle affectation est effective.

3.2.4.4 - Utilisation des fichiers affectables aux unités symboliques

1. Caractéristiques des fichiers

Il s'agit de fichiers permanents partageables simultanément, avec autorisation d'écriture, de type séquentiel ou indexé si la commande d'affectation comporte un nom d'article.

2. Gestion des fichiers

L'affectation "SU fichier" accorde au système la gestion complète du fichier associé à l'unité symbolique.

Création : affectation d'un nom de fichier inexistant à une SU dédiée aux sorties (ou non dédiées)

Ouverture : affectation d'un nom de fichier existant à une SU

Fermeture : commande CLOSE SU
ou réaffectation de la SU si le fichier était affecté à elle seule

Lecture-écriture : accès par des requêtes IOCS portant sur la SU. Le système commute ces appels à IOCS, pouvant provenir des tâches système ou des tâches de l'application, en appels à FMS (dans ce cas, la profondeur disponible de la Kstore est celle requise par une requête FMS).

Tout autre accès à un fichier affecté à une unité symbolique est impossible. En effet :

- en phase de création, un fichier permanent n'est pas partageable simultanément et est en mono-accès
- RTES16 ouvre un fichier existant en demandant une autorisation d'écriture, ce qui a pour effet d'interdire tout autre accès à ce fichier.

3. Nombre de fichiers affectés simultanément à des SU sous RTES16

Le nombre est limité : par le nombre de pavés de la ZDF, défini à la configuration du système ou à la génération de FMS si l'option "Mode privilégié" est présente. Pour chaque fichier affecté à une (des) SU, 1 pavé pour le descripteur de fichier et 1 pavé pour l'unité d'accès sont utilisés.
Quand FMS ne dispose plus d'aucune ressource dans la ZDF, la commande "SU fichier" donne lieu à un compte rendu d'erreur ERC 15 '6020.
L'utilisateur peut fermer certains fichiers pour remédier à cette situation et réitérer la commande.

4. Erreurs survenant lors de l'exploitation des fichiers par les tâches du système ou de l'application

Lorsque FMS refuse une lecture ou une écriture sur un fichier affecté à une unité symbolique (requête IOCS portant sur la SU commutée automatiquement en requête FMS).

Le système prend les mesures suivantes :

- réaffectation standard de la (ou des) SU permettant d'accéder au fichier,
- signalisation de l'erreur sur l'unité symbolique SA par un message :

ERS 04 SU = 'hh 'hhhh

où 'hhhh est le compte rendu de FMS.

Si l'erreur survient lors de la lecture d'une commande sur un fichier affecté à l'unité symbolique Dl, il y a en outre réaffectation standard de l'unité DO et signalisation sur DO par un message ERC 10 'hhhh. L'utilisateur peut reprendre le contrôle du dialogue en appuyant sur la touche "appel opérateur" de la console de service.

3.2.5 - Commandes d'intervention ON-LINE sur le système

Les fonctions réalisées par ces commandes sont les suivantes :

- Activation, suppression de demandes d'activation d'une tâche
- Adjonction, suppression d'une tâche
- Suspension, reprise, suppression des demandes d'activation d'une tache
- Impression de l'état d'une tâche
- Initialisation de l'horloge temps réel, impression de la date et de l'heure
- Impression de l'état du système
- Impression de la carte mémoire
- Visualisation ou modification d'une zone mémoire
- Accès aux périphériques (état et commande)
- fin de dialogue opérateur

La réponse du système à une commande se fait sur le périphérique associé à l'unité symbolique DO. La nature de cette réponse est multiple :

1 - acceptation de la commande et attente de la commande suivante ; le message a la forme suivante :

cr lf *

2 - impression d'un message de réponse et attente de la commande suivante ; c'est par exemple le cas de la commande d'impression de la date et de l'heure

```
* TIME
  25/12/77    20H 02M 05S  cr lf
*
```

3 - refus de la commande avec impression d'un message d'erreur et attente de la commande suivante ; le message a la forme suivante :

```
ERC nn
*
```

ou nn est un numéro d'erreur ayant la signification suivante :

nn	Signification
00	
01	. Commande inexistante
02	. L'entité désignée par la commande n'existe pas
03	. La commande est illogique
04	. L'un des paramètres spécifie une entité inaccessible par commande opérateur
05	. La commande est incompatible avec l'état instantané du système
06	. Il existe au moins un paramètre incorrect
07	. On ne peut pas enregistrer la commande : le système est sous-dimensionné
08	. Syntaxe incorrecte
09	. Erreur d'utilisation des partitions
10	. Informations système non valides sur disque . Disque non prêt . Clé de commande syntaxiquement incorrecte
11	. Affectation SU-FU réputée impossible
12	. Mot de passe incorrect (lancement du dialogue)
13	. Affectation BI incorrecte pour commandes TASK et EXEC.
14	. Caractère impair en entrée de commande

INIT

But	Initialiser le système
Format	INIT [, [SI] [, FU]]

Signification des paramètres

SI : le numéro d'un système, ou d'un état de l'application antérieurement sauvé sur disque par une commande SAVE (par défaut, ce numéro sera considéré comme étant égal à celui de la dernière commande SAVE).

FU : désigne la FU bootstrap (Dl par défaut)
la FU précisée doit être une FU initiale d'un disque bootstrapable (voie 0 d'un coupleur disque)

Commentaires

Initialiser le système revient à charger en mémoire un état antérieur de la partie résidente d'une application :

(SI ϵ [0,7])

Lors du lancement par cette commande de l'un des systèmes sauvegardés par SAVE, RTES16 "monte" le disque support du système. Si le montage ne peut être fait, RTES16 édite sur TS le message :

"ERREUR MONT DISQUE SYSTEME NO (a) CR = '(b)

(a) 1 = SVC FMS "delete FU"
2 = SVC IOCS read structure
3 = SVC IOCS DMON
4 = SVC IOCS MONT
5 = SVC IOCS create FU

(b) compte-rendu IOCS ou FMS

c) RTES16 contrôle l'existence du fichier "RTES16". En cas d'erreur il y a édition du message :

- OPEN FICHIER (a) IMPOSSIBLE ('b)

(a) nom du fichier
(b) CR FMS

En cas d'erreur sur le montage du disque système ou à l'ouverture du fichier, RTES16 ne peut être lancé, il édite le message "ERREUR FATALE" et se bloque.

d) Si lors de la configuration du système l'utilisateur a déclaré une tâche à lancer, en cas d'initialisation ou de restart celle-ci est lancée avec un paramètre d'appel égal à FFFF (-1).
Si cette tâche n'est pas connue sous le système rechargé, il y a impression du message d'erreur : ERC 02.

e) Si lors de la configuration du système l'utilisateur a demandé d'affecter Dl à un fichier de commande, celui-ci est lancé.
Ce fichier doit se terminer par Dl TK.
Si ce fichier n'existe pas Dl est affecté à TK.

Erreurs

ERC 02 : numéro de système SI incorrect ou tâche à relancer sur INIT inexistante ou FU inexistante.
ERC 03 : le paramètre FU ne désigne pas une FU disque ou ne pointe pas un disque bootstrapable (VOIE ≠ 0).
ERC 04 : - désignation de la FU bootstrap par un nom de SU interdit
- pas de gestion de volume sur le disque désigné par le paramètre FU.
ERC 08 : syntaxe incorrecte.

Remarque

L'initialisation du système n'est effectivement réalisée que lorsque tous les échanges du système en cours sont terminés.

SAVE

But — Sauvegarder sur disque l'état actuel de la mémoire

Format — Save, [P1] , [SI], [ADK] , [P2] , [P3] [, {D / M}]

Signification des paramètres

P1 : numéro de partition de la zone mémoire inférieure à 64K

SI : numéro de système (Si = 0 ... 7) (par défaut, numéro du système en cours)

ADK : adresse disque (en secteurs). Par défaut, adresse du dernier système sauvegardé sous le numéro SI.

D : la sauvegarde doit être faite, les disques volumes démontés

M : la sauvegarde peut être faite, les disques volumes montés.

Le paramètre pris par défaut est D.

Commentaires

Cette commande permet de sauvegarder soit :

- la zone de mémoire contenant le système ainsi que les tables (ZUEP, ZIOCB, ZDR,...)

  exemple : Save,,2

- la zone système ainsi que les P + 1 premières partitions appartenant à la zone mémoire allant jusqu'à 64 K

  exemple : Save,p,2

- la zone système ainsi que les partitions comprises entre P2 et P3 (bornes incluses) à l'adresse disque 900

  exemple : Save,,2 ,900 , P2 , P3

- la zone système, les pl premières partitions et les partitions comprises entre P2 et P3

  exemple : Save, p , 2 ,, P2 , P3

La sauvegarde se fait sur disque dans l'unité fonctionnelle Bootstrap Dl sous le numéro de référence SI (2 dans les exemples) à l'adresse précisée par le paramètre ADK ou à l'adresse du dernier système sauvegardé sous le numéro SI.
L'application doit être dans un état inerte, toutes les tâches étant inactives (fin de configuration).

Le système vérifie qu'il est possible de sauvegarder à l'adresse secteur précisée, c'est-à-dire qu'on ne risque pas d'écraser un autre système (→ ERC 06).

Dans le cas où le système SI n'existe pas encore dans D1 (1ère sauvegarde), il faut préciser le paramètre ADK.
Si on utilise un système avec Background, il faut conserver de la place au fond de la FU Dl pour le Swapp.

Erreurs

ERC 02 : numéro de partition incorrect

ERC 03 : - sauvegarde interdite après montage de volume
- adresse disque non précisée et non connue implicitement (aucun système n'a été sauvegardé jusqu'alors sous le numéro SI)

ERC 05 : disque non prêt

ERC 06 : - taille de la zone disque requise pour la sauvegarde supérieure à celle disponible pour ce système
- adresse disque < 5 (tentative d'écrasement des 5 premiers secteurs)

ERC 08 : syntaxe incorrecte.

DMON

But : Démonter un volume (DMON)

Format : DMON, {SU / FU}

Signification des paramètres

FU : nom de l'unité fonctionnelle pointant sur l'espace enveloppe (et caractérisant l'unité physique)
SU : nom de l'unité symbolique à laquelle est affectée la FU

Commentaires

Démonter le volume c'est invalider tous les FU pointant sur le volume.

Erreurs

ERC 06 : nom de FU incorrect
ERC 08 : syntaxe incorrecte
ERC 15 'xxxx : erreur de montage IOCS où 'xxxx désigne le compte-rendu d'IOCS
ERC 16 'yyyy : erreur de montage FMS où 'yyyy désigne le compte-rendu de FMS

Remarque

Le démontage n'est effectif que si tous les fichiers sont fermés sur le volume.

Liste des erreurs IOCS

'6000 : erreur de génération
'6001 : pas de gestion de volumes sur ce type de disque
'6002 : aucun volume n'est monté sur l'unité physique
'6003 : erreur de label
'6004 : pas de FU enveloppe
'6005 : IOCS ne peut prendre en compte tous les espaces ; les espaces pris en compte sont ceux imprimés par la commande MONT
'6006 : volume non structuré
'6007 : aucun cylindre valide parmi les 8 premiers

Liste des erreurs FMS

'6003 : sous dimensionnement de la zone réservée à l'unité physique par GENFMS (un ou plusieurs espaces peuvent avoir été pris en compte)
'600C : erreur de label
'6018 : volume mal structuré
'601A : type de disque sans gestion de volume
'601E : il reste des fichiers ouverts sur le volume
'6035 : erreur de génération de BOS16

MON

But — Monter un volume (MON)

Format — MONT , {SU / FU} [, Label]

Signification des paramètres

FU : nom de l'unité fonctionnelle pointant sur l'espace enveloppe (et caractérisant l'unité physique)
su : nom de l'unité symbolique à laquelle est affectée la FU
Label : label du volume monté

Commentaire

Le montage d'un volume sur une unité physique consiste à faire correspondre à chaque espace du volume une FU appartenant à l'unité physique.
La régie d'affectation espace - FU est la suivante : la ième FU déclarée à GENIO sur l'unité physique est affectée à l'espace de numéro i.

La commande MONT sans paramètre permet d'éditer la correspondance N° espace - N° FU du disque support du système.

Exemple

A l'aide de la commande SDEF du processeur FUP4 on a défini sur le volume (label CART1) les espaces suivants :

E0	adresse	cylindre	0	longueur	400
E1	P1		1	"	30
E2	"		31	"	0 (espace invalide)
E3	"		100	"	299

Les FU suivants sont déclarés à GENIO sur l'unité physique : D9, D3, D4, D5, D6.

```
* MONT, D9

  Label   :  CART1
  SPACE  0      FU  D9      ' 29
  SPACE  1      FU  D3      ' 0F
  SPACE  3      FU  D5      ' 11
```

Erreurs

ERC 06 : nom de FU incorrect
ERC 08 : syntaxe incorrecte
ERC 15 ' xxxx : erreur de montage IOCS où ' xxxx désigne le compte-rendu d'IOCS
ERC 16 ' yyyy : erreur de montage FMS où ' yyyy désigne le compte-rendu de FMS.

Remarque

Lorsque le nombre de FU déclarées sur l'unité physique et le nombre d'espaces définis sur le volume ne sont pas égaux, les espaces pris en compte sont ceux imprimés par la commande MONT.
La correspondance FU_i et E_i n'est pas réalisée si l'espace de rang i n'est pas valide.
Pour les erreurs d'IOCS et de FMS voir DMON.

SUSP

But	Affecter à une SU la FU disque pointant sur un espace d'un volume monté
Format	SUSP, Ui, Ei [,FU]

Signification des paramètres

- Ui : nom d'unité symbolique (i = 1 à F)
- Ei : numéro d'espace (Ei = 1 à 15)
- FU : nom de l'unité fonctionnelle pointant sur un volume monte. Par défaut il s'agit du dernier volume monté.

Erreurs

- ERC 03 : pas de volume monté
- ERC 06 : numéro espace supérieur au nombre de FU sur l'unité physique supportant le volume
  nom de FU correspondant à cet espace est incorrect
- ERC 08 : syntaxe incorrecte
- ERC 10 : espace invalide sur disque
- ERC 15' xxxx : erreur IOCS

Remarque

Pour les erreurs d'IOCS et de FMS voir DMON.

BACK

But Activer le moniteur Background

Format BACK, NP [,REQUET] [,N]

Signification du paramètre

NP : numéro de partition de type non résident

REQUET : symbole optionnel spécifiant que, pour cette vacation, les requêtes temps réel sont accessibles aux processeurs background (par défaut, pas d'accès aux requêtes temps réel)

N : symbole optionnel spécifiant que, pour cette vacation, aucune tâche "temps réel" ne provoquera de swapp-out des partitions background : le moniteur BACKM ne transforme plus les E/S en IMOD et IBMOD en E/S en EMOD (augmentation des performances)

La commande imprime alors :

SWAPP SIZE : XX KWORDS

XX correspond à la taille maximum de SWAPP accessible dans la FU D1 en fonction de la taille des partitions allouées au background et de la place disponible en fond de DI. Cette valeur permet de connaître la valeur maximum du paramètre TAILLE de la commande JOB du background.

Erreurs

ERC 01 : option background non gérée
ERC02 : le moniteur background n'est pas intégré
ERC04 : une tâche ayant le numéro système NT = 122 a été intégrée, et n'est pas le moniteur background
ERC05 : vacation background en cours
ERC 09 : numéro de partition incorrect (partition de type résident ou partition inexistante, ou partition du moniteur BACKM).

Commentaires

1 Après intégration du moniteur background (TASK, R, BACKM – : S, Pi) sous RTES-D l'ouverture de la vacation background est réalisée par cette commande spécifique.
Le paramètre NP permet de préciser le numéro de la première (et éventuellement unique) partition allouée aux processeurs du background (ne pas confondre avec le numéro Pi de la partition du moniteur background).

2 L'accès par un processeur background aux requêtes temps réel (RUN, START...., WEVENT ,... REDGET . . . . TCALL . . . . . SEND) doit être utilisé avec circonspection les risques de perturbation de l'application temps réel par un processeur qui n'est pas au point pouvant être importants (par exemple, destruction de la ZDR par une requête REDPUT).

3 L'activation effective de BACKM (impression de (rc) (lf) # et attente de commande opérateur) est réalisée par la première commande GI VE consécutive à l'ouverture de la vacation par la commande BACK.

4 Si le paramètre N a été spécifié dans la commande BACK, la ou les tions utilisées par les processeurs background ne seront libérées qu'en fin de tion background (commande EBOS).

GIVE

But Autoriser au background l'accès à une, ou plusieurs, unité fonctionnelle.

Format GIVE, B1R1, B2R2, --- , B5R5

Signification des paramètres

Bi : unité fonctionnelle rendue accessible au background

Ri : unité fonctionnelle de l'installation qui simule Bi ; par défaut, Ri = Bi

Nombre maximum de Bi [Ri] : 5. (les paramètres additionnels provoquent une erreur de syntaxe au décodage de la commande).

Exemple : GIVE, D2, CR, LPF3, T2T4

Erreurs

ERC 02 : un paramètre au moins correspond à une FU non gérée
ERC 03 : commande émise hors vacation background
ERC 04 : affectation interdite▲
ERC 08 : erreur de syntaxe

Commentaire

La console background, simulée en standard par les unités fonctionnelles F7 (TS) et F8 (TK) peut être en fait simulée par toute autre console de l'installation. Si par exemple l'unité F1 est un dispositif de visualisation alphanumérique, la commande :

GIVE,TKF1.TSF1,CR,LP,D2

permet d'utiliser cette visualisation comme console background, à condition que cette commande GIVE soit la première émise après la commande BACK.

▲ voir tableau des affectations possibles en annexe

TAKE

But Supprimer l'autorisation d'accès à une, ou plusieurs, unité fonctionnelle par le background

Format TAKE, R1, R2, ---, R5

Signification des paramètres

R i : unité fonctionnelle de l'installation
nombre maximum de paramètres Ri : 5

Erreurs

ERC 02 : un paramètre au moins correspond à une FU non gérée
ERC 03 : commande émise hors vacation background
ERC 06 : il reste des fichiers ouverts sur une des FU
ERC 08 : erreur de syntaxe.

Remarque

Cette commande peut être ineffective (cas où la ou les FU précisées ne sont pas accessibles au background). Dans ce cas, il n'y a pas de diagnostic d'erreur.

KEND

But Terminer le dialogue opérateur (KEND)

Format KEND [,MOPASS]

Signification du paramètre

MOPASS : mot de passe constitué de 6 caractères alphanumériques maximum (par défaut, aucune protection sur l'accès au dialogue opérateur).

Commentaires

Lorsque le mot de passe est présent dans la commande, il sera nécessaire de le fournir lors de la prochaine activation du dialogue opérateur (après l'appel pupitre).

Lorsqu'il est absent, toute personne sera habilitée à dialoguer avec le système après avoir appuyé sur le bouton appel du téléimprimeur.

Remarque

Lorsque l'unité symbolique Dl est connectée à l'unité fonctionnelle TK, le système réalise une surimpression sur la feuille du téléimprimeur pour masquer le mot de passe.

Erreurs

ERC 08 . mot de passe incorrect (plus de 6 caractères)
. syntaxe incorrecte.

POFF

But Mettre hors tension le lecteur de cartes

Format POFF

Commentaire Commande toujours acceptée

TASK

But : Intégrer sous le système une nouvelle tâche

Format : TASK, { RESID / NRESID / RP / [N]P }, FICNAM - PW, P1, P2, P3, P4.

Signification des paramètres :

{ RESID / NRESID } : symboles spécifiant l'état de résidente de la tâche intégrée :
- RESID (ou R) : la tâche est résidente en mémoire centrale..
- NRESID (ou N) : la tâche est non résidente (résidente sur disque).
  (par défaut, la tâche est non résidente).
- RP : la tâche est résidente et fonctionne en mode privilégié
- P ou (NP) : la tâche est non résidente et fonctionne en mode privilégié.

FICNAM-PW : nom de fichier (6 caractères maximum) et de catalogue (2 caractères) spécifiant le nom d'un fichier du type image mémoire généré sur disque par le chargeur disque BUILDER de BOS16 (création de tâche OFF-LINE) ou de BACKM (création de tâche ON-LINE en Background).

P1, P2, P3, P4 : Liste des partitions (4 au maximum) accessibles à la tâche intégrée. Pour une tâche résidente, un seul numéro doit être spécifié. (par défaut, le mot 14 de la PST de la tâche est interprété comme numéro de l'unique partition accessible - cf ANNEXE).

Les tâches intégrées sous RTES16 peuvent être situées sur toute FU disque gérée par FMS (et pas seulement sur la FU D2).

Le fichier précisé en paramètre de la commande est lu sur la FU disque affectée à l'unité symbolique BI (D2, en affectation standard).

Exemples :

```
*BI     D3
 *TASK,  RESID,  FTKO1-C2,4
*TASK,  R,  RTKO2-C2,2
 *BI  D8
*TASK,  R,  FTKO3-C3,3
 *TASK,  N,  FTSO6-C3,  7,  10,  11
 *BI     E2
*TASK,,FTSO7-C3
  TASK,NP,TEST-IM,5
```

Commentaires : Cette commande introduit sous RTES16 la tâche contenue dans le fichier spécifié.
Son exécution comprend trois phases :

1 Lecture du descripteur du fichier (cf BUILDER de BOS16) et test de vraisemblance des paramètres qu'il contient et des paramètres de la commande.
2 recopie de l'état initial de la PST en mémoire dans une zone résidente (Zone des Blocs de Contrôle des Tâches de l'application). En cours d'exécution, la tâche pourra réutiliser les 16 mots de la PST initiale en tant que mémoire de travail.
3 si la tâche contenue dans le fichier est déclarée résidente en mémoire, la racine du fichier est chargée en mémoire dans sa partition.

Erreurs

ERC 02 : Fichier spécifié inexistant dans la FU disque affectée à l'unité symbolique BI

ERC 03 : Le fichier précisé n'est pas libre ou n'est pas de type indexé.
. Il existe déjà une tâche de l'application ayant la priorité ou le numéro de la tâche à intégrer.
. Une tâche hardware ne peut pas avoir une structure d'overlay.
. Pas de PST dans la tâche.
. La priorité de la tâche est inconnue (non précisée au niveau BUILDER).
. Nombre d'appels cumulables supérieur à 256.

ERC 05 : Disque non prêt ; FU verrouillée

ERC 07 : On ne peut pas enregistrer la commande : le système est sous-dimensionné :
- enveloppe des tâches insuffisante (ZBCT)
- zone de travail du système saturée (ZGIN)

ERC 08 : erreur de syntaxe.

ERC 09 : Plusieurs partitions sont spécifiées dans la commande pour une tâche résidente.
. Numéro(s) de partition(s) non valide(s) dans la commande (ou dans le descripteur).
. La partition spécifiée pour l'intégration d'une tâche résidente est déjà occupée.
. La taille de l'image mémoire de la tâche intégrée est supérieure à la taille de la (des) partition(s) spécifiée(s).
. Plusieurs partitions sont indiquées dans la commande pour l'intégration d'une tâche en mode maître.
. Pour une tâche en mode maître, l'adresse d'implantation doit être égale à l'adresse de début de la partition spécifiée.

ERC 13 : Lors de l'exécution de la commande TASK, l'unité symbolique BI n'est pas affectée à une FU disque gérée par FMS.

EXEC

But Intégrer sous le système un processeur foreground et l'exécuter (EXEC)

Format EXEC, {RESID / P}, FICNAM-PW, P1

Signification des paramètres

{RESID (ou R) / P} : Ce symbole spécifie l'état de résidence du processeur : un processeur foreground est toujours résident ; le symbole P signifie que le processeur fonctionne en mode privilégié.

FICNAM-PW : Nom de fichier (6 caractères maximum) et de catalogue (2 caractères) spécifiant le nom d'un fichier du type image mémoire généré sur disque par le chargeur disque BUILDER de BOS16 (BACKM)

P1 : Numéro de la partition dans laquelle le processeur doit être chargé. Si on a le paramètre P, la partition peut être une partition de type non résident.

Commentaires

Les processeurs intégrés sous RTES16 peuvent être situés sur toute FU disque gérée par FMS (et pas seulement sur la FU D2).
Le fichier précisé en paramètre de la commande EXEC est lu sur la FU disque affectée à l'unité symbolique BI (D2 en standard)
La commande TASK permet d'intégrer sous RTES16 une tâche de l'application, résidente ou non résidente.
La commande EXEC permet d'intégrer un programme d'intérêt général.- qui n'a pas de priorité propre, et qui, après son chargement dans la partition spécifiées'exécute avec une priorité réservée au système (celle de la tâche de dialogue opérateur).

Erreurs

ERC 02 : fichier spécifié inexistant dans la FU disque affectée à l'unité symbolique BI

ERC 03 : processeur déjà intégré
le fichier précisé n'est pas libre ou n'est pas de type indexé
le programme associé ne doit pas contenir de PST
Un processeur intégré sous RTES16 doit être en mode maître, et ne pas avoir de structure d'overlay.

ERC 05 : disque non prêt ; FU verrouillée

ERC 07 : on ne peut pas enregistrer la commande : le système est sous-dimensionné (zone de travail ZGIN du système saturée).

ERC 08 : erreur de syntaxe

ERC 09 : un seul numéro de partition doit être spécifié dans la commande
numéro de partition incorrect
partition déjà occupée par une tâche résidente
la taille de l'image mémoire du processeur est supérieure à la taille de la partition indiquée
l'adresse d'implantation du processeur doit être égale à l'adresse de début de sa partition

ERC 13 : Lors de l'exécution de la commande EXEC, l'unité symbolique BI n'est pas affectée à une FU disque gérée par FMS.

ROCK-SLOW

But	Etablir un compromis espace-temps sur le chargement des tâches non résidentes.
Format	ROCK, NT SLOW, NT

Signification du paramètre

NT : numéro d'appel de tâche ou nom de la tâche sans catalogue.

Commentaires

Ces commandes permettent à l'utilisateur, en phase de mise au point de l'application, d'établir un compromis occupation mémoire-temps de réponse sur le traitement d'approvisionnement d'une tâche non résidente.

Dans le fonctionnement standard, ce traitement se décompose de la façon suivante :

1 - allocation d'une partition
2 - ouverture du fichier image mémoire : OPEN OLD
3 - lecture de la racine de la tâche : IREAD (ROOT)
4 - activation effective de la tâche
5 - déroulement de la tâche
6 - sur EXIT, libération de la partition et fermeture du fichier : CLOSE.

La durée des opérations 1 + 2 + 3 + 4 + 6 peut en moyenne être divisée par 2 si le fichier est ouvert de façon permanente (suppression des requêtes OPEN OLD et CLOSE dans le traitement ci-dessus).

D'où un gain de temps relativement appréciable. Néanmoins, les informations "système" (1 pavé descripteur de fichier +1 pavé d'accès au fichier, soit 60 mots), allouées dynamiquement en mémoire lors de l'OPEN OLD et désallouées lors du CLOSE, deviennent elles aussi permanentes en mémoire.

La commande : ROCK, NT
permet de passer du mode standard d'approvisionnement au mode "rapide" où le fichier support de la tâche NT est ouvert en permanence (au prochain lancement de NT, la requête OPEN sera réalisée, mais le CLOSE ne le sera plus sur EXIT).

La commande : SLOW, NT
permet de repasser au mode standard (sur le prochain EXIT de NT, le fichier sera fermé et les informations mémoire "système" désallouées).

Erreurs

ERC 02 : NT tâche inexistante
ERC 03 : Commande redondante (ex : 2 fois ROCK)
NT tâche non résidente
ERC 04 : NT tâche hardware ou "système"
ERC 06 : NT incorrect (négatif)
ERC 08 : erreur de syntaxe.

KILL

But Supprimer une tâche, un processeur foreground ou le moniteur background

Format KILL [,NT | ,FICNAM- PW]

Signification des paramètres

NT : la commande KILL ,NT permet de supprimer la tâche de numéro NT et de libérer sa partition si elle l'occupe.

Par défaut (commande "KILL") la tâche supprimée est le moniteur background.

FICNAM - PW : un processeur foreground est identifié par le nom de son fichier image mémoire. Ainsi, le processeur intégré par la commande

EXEC, R, FICNAM - PW, 4

peut être supprimé par la commande :

KILL, FICNAM - PW

Cette commande a pour effet de libérer la partition 4 et de supprimer les éventuelles commandes qu'il a introduites par SVC CAMO.
Les éventuelles requêtes spécifiques introduites par ce processeur (SVC NEWS) devront être préalablement interdites par la commande INEX, NR

Erreurs

ERC 02 : NT tâche inexistante ou FICNAM-PW processeur foreground inexistant
EHC 04 : NT tâche hardware
ERC 05 : NT active (ou background actif) et non suspendue après erreur ou alarme ou non inhibée
ERC 06 : NT incorrect (négatif)
ERC 08 : Erreur de syntaxe.

Commentaires

La commande KILL permet de tuer une tâche inactive (après EXIT pour les tâches temps réel, après EBOS pour le background) ou une tâche suspendue par le système après alarme ou erreur sur requête ou inhibée par requête programmée (INHIB) ou commande opérateur (STOP). Seules les ressources connues du système sont désallouées (partition, PST, appels cumulés, pavé overlay).

INEX

But	Interdire l'accès par programme à une requête utilisateur introduite par SVC NEWS
Format	INEX NR

Signification du paramètre

NR : numéro de requête programmée.

Commentaire

L'usager peut, par intégration d'un processeur foreground. étendre le système par adjonction de nouveaux sous-programmes superviseur (SVC) Avant d'utiliser la commande KILL de suppression d'un tel processeur, il conviendra d'interdire l'accès à ces requêtes par la commande INEX. L'utilisation ultérieure d'une telle requête provoquera l'erreur sur requête ERR 208 : "requête non traitée".

Erreurs

ERC 04 : numéro NR incorrect : NR $\notin$ ⌈0, 139⌋
ERC 05 : la requête NR n'existe pas (non intégrée par NEWS)
ERC 08 : syntaxe incorrecte.

CONT

But — Supprimer les contrôles sur requêtes (CONT)

Format — CONT, N/Y [, p1 [, p2]]

Signification des paramètres

N : Supprime les contrôles.
Y : Rétablit les contrôles.

P1, P2 : priorités des tâches concernées.

Commentaires

- Cette commande a deux effets :

  . Suppression des contrôles concernant les adresses de RPB, de profondeur de kstore, d'adresse de buffer.

  . Suppression de la suspension d'une tâche en cas d'erreur sur requête : la tâche devra traiter elle-même les compte-rendus.

- On peut spécifier :

  . Toutes les tâches (de 0 à 127) :

  CONT, N

  . Une plage de tâches :

  CONT, N, P1, P2

  Toutes les tâches comprises entre P1 et P2 sont alors concernées.

  . Une tâche :

  CONT, N, P1

Erreurs

ERC 03 : priorité de tâche incorrecte.
ERC 08 : syntaxe incorrecte.

PART

But	Modifier l'affectation de partition (s) d'une tâche non résidente
Format	PART, NT, P1, P2,..., PN

Signification des paramètres

NT : numéro d'appel de tâche ou nom de la tâche sans catalogue.
P1, P2, .. PN : numéros de partitions (au minimum, un seul numéro, 4 numéros maximum)

Erreurs

ERC 02 : NT tâche inexistante
ERC 03 : NT est une tâche hardware
. NT est une tâche résidente
. NT est en mode maître
ERC 04 : NT tâche système
ERC 06 : NT incorrect (négatif)
ERC 08 : syntaxe erronée
ERC 09 : une des partitions est trop petite ou est de type résident.

Commentaires

1 On ne peut modifier l'affectation de partition (s) d'une tâche que si elle est résidente sur disque.

2 La liste des partitions comprend au plus 4 numéros représentant les partitions "possibles" pour l'exécution de la tâche.

3 Cette commande peut intervenir alors que la tâche NT est active.

4 Une tâche non résidente n'étant pas systématiquement approvisionnée en mémoire à chaque activation, l'effet de cette commande peut être différé ; la tâche 5 étant active dans la partition 12 par exemple, la commande :

PART, 5, 13

n'entraînera le chargement de la tâche 5 dans la partition 13 qu'au prochain approvisionnement en mémoire (après allocation de la partition 12 à une tâche).

RTY

But — Modifier la priorité d'une tâche

Format — PRTY, NT, PY

Signification des paramètres
NT : numéro d'appel de tâche ou nom de la tâche sans catalogue
PY : numéro de priorité

Erreurs
ERC 02 : NT tâche inexistante
ERC 03 : NT est une tâche hardware
. une tâche de type software ne peut pas devenir une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible par dialogue opérateur
ERC 05 : on ne peut pas modifier la priorité d'une tâche software active.
la priorité à affecter à NT est déjà occupée
ERC 08 : syntaxe incorrecte.

Commentaire — On ne peut modifier la priorité d'une tâche que lorsqu'elle est inactive.

IRUN

But Activer une tâche immédiatement

Format IRUN, NT, PAR

Cette commande permet à l'opérateur d'émettre la requête programmée RUN.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue
PAR : paramètre de travail (nul par défaut).

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible
ERC 05 : le nombre maximum d'appel de NT est déjà atteint
ERC 06 : NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte

Remarque Si deux tâches ont le même nom mais des catalogues différents, c'est la 1ère tâche intégrée qui sera activée.

TACT

But Activer une tâche immédiatement ou jamais

Format TACT, NT, PAR

Cette commande permet à l'opérateur d'émettre la requête programmée RUN si, et seulement si, la tâche NT n'est pas déjà active.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue
PAR : paramètre de travail (nul par défaut).

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible
ERC 05 : la tâche NT est active
ERC 06 : NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte.

Remarque Si deux tâches ont le même nom mais des catalogues différents, c'est la 1ère tâche intégrée qui sera activée.

WRUN

But	Activation d'une tâche et attente de la fin d'exécution
Format	WRUN, NT, PAR

Cette commande permet à l'opérateur d'émettre la commande STARTW.

Signification des paramètres

NT : numéro d'appel de la tâche ou nom sans catalogue
PAR : paramètre de travail (nul par défaut)

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système
ERC 05 : le nombre maximum d'appels est déjà atteint
ERC 06 : NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte.

Remarques

- Cette commande permet en particulier de substituer un dialogue utilisateur au dialogue système sur la console système.

- Pendant le temps d'exécution de la tâche, il n'y a plus de traitement des défauts périphériques.
  II est donc important que l'utilisateur prenne en compte le traitement des défauts.

DRUN

But Activer une tâche après un délai initial

Format DRUN, NT, PAR, ID/IUD. IP/IUP

Cette commande permet à l'opérateur d'émettre la requête programmée START.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue
PAR : paramètre de travail (nul par défaut)
ID : délai spécifié par son unité IUD (nul par défaut)
IUD : unité de temps 0 : top de base
1 : milliseconde
2 : seconde
3 : minute
IP : période spécifiée par son unité IUP (activation unique par défaut)
IUP : unité de temps - même signification que IUD

Exemples : DRUN,12,2,45/3 DRUN,15,,0/0,5/2

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible
ERC 05 : le nombre maximum. d'appel de NT est déjà atteint
ERC 06 : délai ou période incorrect
. NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte.

TRUN

But — Activer une tâche à une heure donnée

Format — TRUN, NT, PAR, HR/MN/SC, IP/IUP

Cette commande permet à l'opérateur d'émettre la requête programmée TRNON.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue
PAR : paramètre de travail (nul par défaut)
HR, MN, SC : heure d'activation $0 \leqslant HR \leqslant 23$, $0 \leqslant MN \leqslant 59, 0 \leqslant SC \leqslant 59$
IP : Période spécifiée par son unité IUP (activation unique par défaut)
IUP : unité de temps 0 : top de base
1 : milliseconde
2 : seconde
3 : minute

Exemples : TRUN, 103,5,8/30/0, 20/2 TRUN,15,,23/10/5

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible
ERC 05 : le nombre maximum d'appel de NT est déjà atteint
ERC 06 : heure ou période incorrecte
. NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte.

TOFF

But — Supprimer des appels à une tâche

Format — TOFF, NT, PAR

Cette commande permet à l'opérateur d'émettre la requête programmée OFF.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue
PAR : paramètre de travail (par défaut, tous appels antérieurs supprimés)

Erreurs

ERC 02 : la tâche NT est inconnue du système
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT est une tâche système inaccessible
ERC 06 : NT négatif
ERC 07 : système sous-dimensionné
ERC 08 : syntaxe incorrecte.

STOP

But — inhiber une tâche dans son état actuel

Format — STOP, NT

Cette commande permet à l'opérateur d'émettre la requête programmée INHIB.

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue

Erreurs

ERC 02 : la tâche NT n'existe pas
ERC 03 : la tâche NT est une tâche hardware
ERC 04 : la tâche NT n'est pas accessible à l'opérateur
ERC 05 : la tâche NT est déjà inhibée
ERC 08 : syntaxe incorrecte

Commentaires

La tâche inhibée est suspendue dans l'état où elle se trouve ; elle conserve ses ressources et en particulier la partition qu'elle occupe si elle est présente en mémoire.
Les demandes d'activation de cette tâche (par requêtes programmées ou commandes opérateur) continuent à être enregistrées pendant la durée de l'inhibition.

VALI

But — Relancer une tâche inhibée

Format — VALI, NT

Signification du paramètre

NT : numéro d'appel de tâche ou nom sans catalogue

Erreurs

ERC 02 : la tâche NT n'existe pas
ERC 03 : la tâche NT n'est pas inhibée
ERC 04 : la tâche NT n'est pas accessible à l'opérateur
ERC 08 : syntaxe incorrecte

Commentaires

La tâche est reprise dans l'état où elle a été inhibée si, pendant l'inhibition, son environnement n'a pas été modifié.
Dans tous les cas, les registres de son contexte n'ont pas évolué.

RSUM

But	Relancer une tâche interrompue sur erreur
Format	RSUM, NT

Signification du paramètre

NT : numéro d'appel de tâche ou nom de la tâche sans catalogue

Commentaires

Une tâche interrompue sur erreur dans l'exécution d'une requête programmée reprend le contrôle, lorsqu'elle est relancée par RSUM, en séquence après l'appel de cette requête.
Elle doit exploiter le compte-rendu de requête qui se trouve dans l'accumulateur (utilisation immédiate du compte-rendu) et dans la mémoire d'adresse spécifiée dans le RPB de la requête (utilisation différée du compte-rendu).

Erreurs

ERC 02 : la tâche NT n'existe pas
ERC 03 : la tâche NT n'est pas suspendue sur erreur
ERC 04 : la tâche NT n'est pas accessible à l'opérateur
ERC 08 : syntaxe incorrecte.

Remarque

Une extension dans l'utilisation de la commande RSUM permet de rendre optionnelle la suspension des tâches software de l'application après une erreur sur requête, et ceci de façon globale, pour toutes les tâches et toutes les requêtes.

- La commande : RSUM (sans paramètre) permet de passer dans un mode de fonctionnement du système où il n'y a plus de suspension après les erreurs sur requêtes.
- La commande : RSUM, FFFF permet de repasser dans le mode initial de suspension après erreur.

Après une erreur sur requête, que la tâche fautive soit suspendue ou non, le message d'erreur imprimé par le système sur le périphérique affecté à l'unité symbolique TE est le même (ERR .... NT= ....) et le compte rendu de la requête est inchangé.

TIME

But	Initialiser ou imprimer la date et l'heure
Format	Initialisation : TIME, [J/M/A][,HR/MN/SC] Impression : TIME

1 initialisation :

Signification des paramètres

J,M,A : date en jour, mois, année (si ces paramètres sont absents la date est inchangée)

HR, MN, SC : heure actuelle - heure, minute, seconde (MN, SC nuls par défaut)
(si ces 3 paramètres sont absents, l'heure est inchangée).

Exemple : TIME, 25/12/73, 20

```
Initialisation de la date et de l'heure :
25/12/73, 20 heures.
```

Erreurs

ERC 03 : Paramètres non vraisemblables (J>31, HR>24....)
ERC 08 : Syntaxe incorrecte.

2 Impression :

Le format de sortie sur le dispositif associé à l'unité symbolique DO (Dialogue Output) est donné sur un exemple :

25/12/75 20H 02M 05S

```
Date : 25/12/75    heure : 20h 2mn 5s
```

Commentaires

RTES16 gère l'heure et la date :

- changement de mois,
- changement d'année,
- années bissextiles.

SYST

But Connaître l'état du système

Format SYST, N

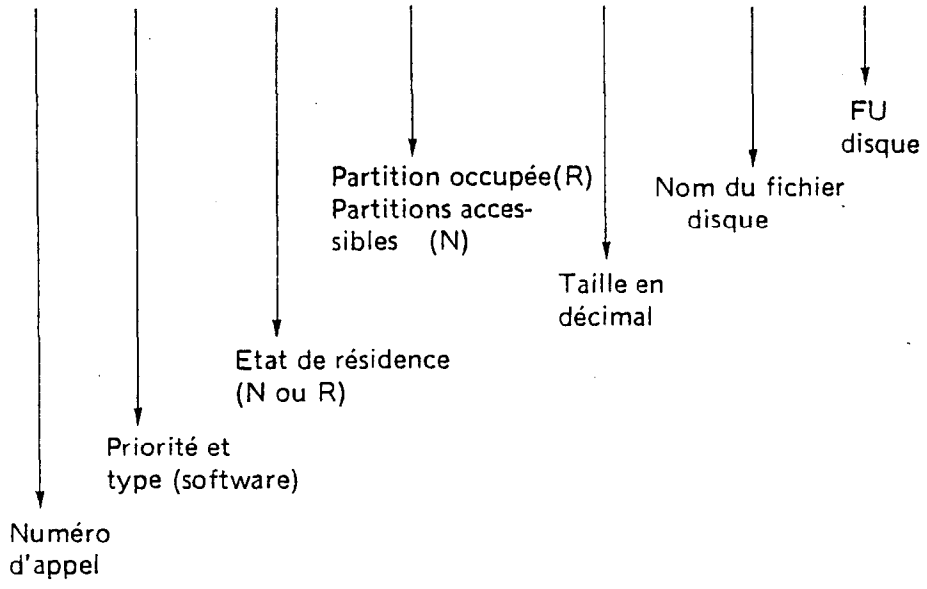
Signification du paramètre

N : un paramètre qui précise le type d'informations système à visualiser.

N = 0

Informations relatives aux tâches software de l'application

```
* SYST,0
NT    TASK     RST  PARTITIONS   SIZE   FICNAM-PW   FU
011   S,011    N      P11        01061  TASK1 - AA  15
017   S,017    N      P16        00211  TASK7 - AA  15
018   S,018    N    P17 P16      00211  TASK8 - AA  15
020   S,020    N      P18        00233  TASK10-AA   15
021   S,021    N      P19        00223  TASK11-AA   15
022   S,022    N      P20        00233  TASK12-AA   15
```

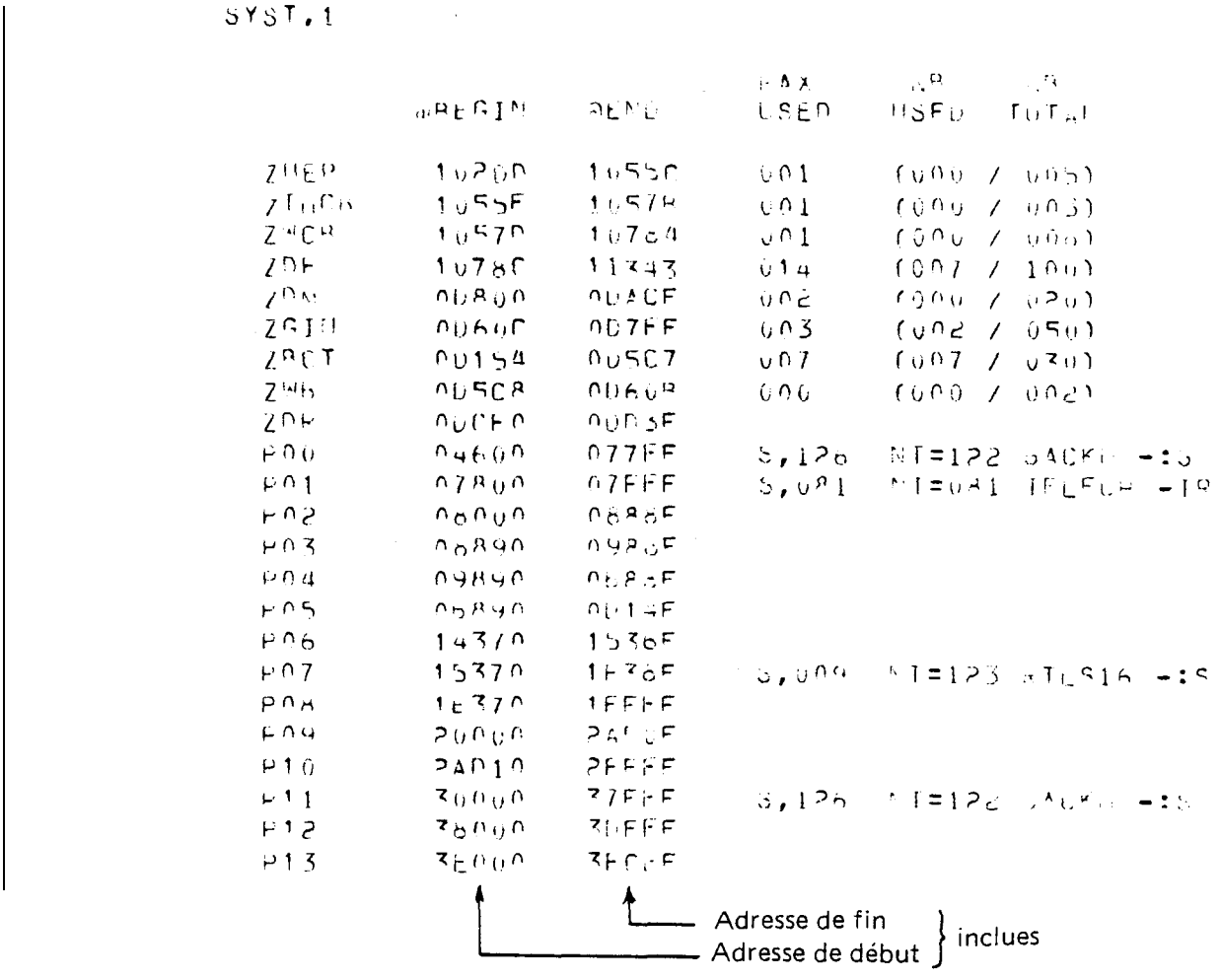


Commentaires Seules les informations système relatives aux tâches software sont visualisées.

N = 1

Informations relatives à la carte de la mémoire vive

SYST,1



L'information donnée sur l'utilisation des zones du système peut être exploitée en phase de mise au point d'une application.

N = 2

Informations relatives aux ressources définies par la requête RESDEF

SYST, 2

NAME	IACCES	RACCES
00011	002	002
00012	002	002
00013	002	002
00600	012	012
00700	047	047

Nombre d'accès résiduels
Nombre initial d'accès
Numéro (ou nom) de la ressource.

STAT

But — Connaître l'état d'une tâche software

Format — STAT, NT

Signification du paramètre — NT : numéro d'appel de la tâche ou nom de la tâche sans catalogue

Commentaires — L'état d'une tâche est indiqué selon le format donné sur un exemple :

```
STAT, 25
   DATE    TASK     RESIDENT      PARTITIONS      SIZE NBCALL OVERLAY
01/01/74   S,003        N      P17 P16 P15 P14    0183 0011/010   YES

ARMED MASKED READY INHIB  USED PARTITION      CALL PARAMETER
  NOT    NOT     NOT   NOT          P14
```

TASK : Priorité de la table software
RESIDENT : résidente (R) ou Non résidente (N) résidente privilégié (RP) ou Non résidente Privilégié (NP)
SIZE : taille de l'image mémoire (en décimal)
NBCALL : nombre d'activations en cours/nombre maximum (précisé dans l'extension de PST lors de l'écriture de la tâche).
USED PARTITION : pour une tâche résidente, le numéro précisé sous ce label est identique au numéro indiqué sous le label PARTITIONS.
Pour une tâche non résidente, ce numéro est l'un des numéros de partitions accessibles, si la tâche est présente en mémoire ; sinon, il n'y a pas de numéro sous ce label.
CALL PARAMETER : pour une tâche active, valeur du paramètre de travail associé à l'activation en cours.

Erreurs
ERC 02 : NT n'est pas une tâche software de l'application
ERC 08 : Erreur de syntaxe
. NT incorrect

DEST

But — Connaître l'état instantané d'un.périphérique

Format — DEST, NFU

Signification du paramètre

NFU : numéro d'unité fonctionnelle rattachée au périphérique (cf § 1.3.2 du manuel d'utilisation d'IOCS : numérotation des unités fonctionnelles)

Commentaires

Cette commande permet de connaître un certain nombre d'informations instantanées relatives à un périphérique de l'installation :
- périphérique super-attaché (ou non)
- périphérique attaché (- - -)
- un échange est en cours sur le périphérique (- - -)
- périphérique en défaut (- - -)
- le périphérique fonctionne en mode canal (- - -)
- mot d'état de ce périphérique.

Exemple :

```
* DEST,11
FU SUPATCH ATTCH BUSY FAULTY CHANNEL STATUS
11    NOT    NOT   NOT   NOT    YES     '0000
```

Lorsque plusieurs unités fonctionnelles sont rattachées à un même périphérique, (ex : TS et TK pour un téléimprimeur) l'utilisation de la commande DEST avec chacune d'entre elles donnera la même information.

Erreurs

ERC 02 : unité fonctionnelle inexistante
ERC 08 : erreur de syntaxe

CFMS

But	Edition des compteurs FMS (CFMS)
Format	CFMS, I / E

Signification des paramètres	I	: Initialisation des compteurs
	E	: Edition des compteurs.

Commentaires

Cette commande permet d'éditer le nombre de primitives adressées à FMS (depuis la dernière initialisation) et le nombre d'accès IOCS qui ont été générés (entre 0 et 65535) par ces primitives.

Erreurs

ERC 08 : erreur de syntaxe.

PRIN

But — Connaître le contenu d'une zone mémoire

Format — PRIN, ADRDEB [, NBMOT] [, NPAG]

Signification des paramètres

ADRDEB : adresse relative, dans la page de début de la zone mémoire à visualiser

NBMOT : longueur (en mots) de la zone mémoire (maximum 50 mots)

NPAG : numéro de la page 64 K à laquelle appartient la zone mémoire (par défaut page 0).

Commentaires

L'opérateur devra interpréter avec circonspection le contenu d'une zone susceptible d'évoluer pendant sa visualisation sur l'unité fonctionnelle affectée à DO.

Erreurs

ERC08 : Syntaxe incorrecte
Paramètre incorrect (adresse inexistante, nombre de mots supérieur à 50, n° de page incorrect).

Exemple : PRIN, ' 0, 5, 1

```
17F2   <-------  visualisation du contenu des 5
4B0A             premiers mots de la page 1
5780
070A
0002
```

DUMP

But — Connaître le contenu d'une zone mémoire (DUMP)

Format — DUMP, ADRI, ADRF [, NPAG]

Signification des paramètres

ADRI, ADRF : adresses relatives dans la page de début et de fin de la zone mémoire.

NPAG : numéro de page de 64 K dans laquelle la zone mémoire est à visualiser.

Commentaires

L'opérateur devra interprêter avec circonspection le contenu d'une zone susceptible d'évoluer entre le début et la fin de visualisation sur l'unité fonctionnelle affectée à DO.

Erreurs

ERC 08 : syntaxe incorrecte

Exemple :

```
ØØ2Ø    3FØØ  8ØØ1  CØØ8  8ØØØ  ØØ3B  8ØØØ  ØØ14  ØØ1Ø
ØØ28    7FFF  Ø2ØØ  23CØ  ØØØØ  ØØØØ  ØØØØ  ØØØØ  ØØØØ
ØØ3Ø    ØØØØ  ØØØ1  DC7F  FFFF  FFFF  FFFF  FFFF  FFFF
ØØ38    FFFF  FFFF  FFCØ  1187  FFFF  FFFF  FFFF  FFFF
ØØ4Ø    FFFF  FFFF  Ø36E  4ØØØ  ØØØØ  ØØØØ  ØØØØ  ØØØØ
```

PAGE

But	Effectuer des sauts de page sur imprimante avec impression d'un en-tête
Format	PAGE, NP

Signification du paramètre

NP : nombre de sauts de page à effectuer.
Par défaut, NP = 1

Commentaire

La commande est ineffective si l'unité symbolique DO n'est pas affectée à l'unité fonctionnelle LP de l'installation.
L'entête comprend la dâte, un titre, et un numéro de page.

Erreurs

ERC 08 : syntaxe incorrecte
. NP supérieur à 10.

MEMO

But — Modifier le contenu d'une zone mémoire.

Format — MEMO, ADR, MASK [[,{AND / OR / EOR}], [,N], [,NPAG]]

Signification des paramètres

ADR : adresse du début de la zone à modifier

MASK : masque de modification du contenu des mémoires de la zone

{AND / OR / EOR} : d'après la valeur de ce paramètre, le contenu modifié de chaque mémoire de la zone est le résultat de l'intersection (AND) ou de l'union (OR) ou de la disjonction (EOR) du contenu initial de la mémoire et du masque.
(si ce paramètre est absent dans la commande, le masque est directement chargé dans les mémoires de la zone à modifier).

N : définit la taille en mots de la zone à modifier.
(si ce paramètre est absent dans la commande, la zone à modifier est considérée comme n'ayant qu'un seul mot)

NPAG : précise le numéro de la page de 64 K sur laquelleporte la commande (par défaut il s'agit de la page 0).

Erreurs — ERC 08 : syntaxe incorrecte.

Exemples

```
Y MEMO, 'CAFE, '8000, OR, 3, 2

`CAFE   := ( `CAFE) OR `8000
`CAFE +1  := (`CAFE +1) OR `8000   } ← dans la page 2
`CAFE +2  := (`CAFE +2) OR `8000

* MEMO, `ABCD, `FADE
  `ABCD  := `FADE                  } ← dans la page 0

* MEMO, `ABCD, `FADE,,2
  `ABCD := `FADE
  `ABCD +1 := `FADE                } ← dans la page 0
```

DBCT

But	Imprimer le contenu du bloc de contrôle d'une tâche de l'application
Format	DBCT,NT

Signification des paramètres

NT : numéro d'appel de tâche ou nom sans catalogue.

Erreurs

ERC 02 : la tâche NT n'existe pas ou nom de la tâche sans catalogue
ERC 04 : la tâche NT n'est pas accessible à l'opérateur
ERC 07 : système sous dimensionné (cette commande utilise un pavé de la zone ZBCT pour prendre un flash du bloc de contrôle spécifié avant de l'imprimer, et aucun pavé de cette zone n'est disponible).
ERC 08 : syntaxe incorrecte

Exemple

```
*DBCT,25
  RA     RB     RX     RY     RC     RL     RW     RK     RP     RS
'007F  '0000  '0085  '0000  '00D4  '00EB  '0000  '4ED4  '379B  '8000

 RSL0   RSLE  NTUSR  TPY    ERSVC  APVLY  FNAM0  FNAM1  FNAM2  PUBW
'4ED0  '569F  '1900  '0119  '0000  '433D  'D4CF  '56C5  'D200  '4141

MTAIL  APFMS  PART1  PART3  MAXAP  PRMAP  DERAP  FUPROG  USRP
'00B7  '0000  '0180  '8080  '000A  '4430  '4430  '000F   '0000
```

Commentaire

1 - Les 12 premiers mots du Bloc de Contrôle de la Tâche constituent sa PST ; les 17 mots suivants correspondent à des informations relatives à la tâche et spécifiques du système RTES16 ; la signification de ces informations est donnée dans le manuel d'utilisation du système.

2 - La commande DBCT fournit un "flash" instantané et cohérent du bloc BCT de la tâche spécifiée.

PAUSE

But	Suspendre l'exécution d'un fichier de commande
Format	PAUSE, texte

Signification des paramètres

texte : texte libre.

Commentaires

Cette commande permet de réaffecter Dl et DO sur les FU TK et TS.
Si le système était en train d'exécuter un fichier de commande, on pourra continuer son exécution par la commande RETU.

Erreurs

ERC 06 : commande PAUSE émise après une 1ère commande PAUSE sans qu'il y ait eu de RETU.

RETU

But	Reprendre l'exécution d'un fichier de commande suspendue par PAUSE
Format	RETU

Commentaires

Permet de continuer l'exécution d'un fichier de commandes interrompue par une commande PAUSE. Dl et DO sont réaffectés aux FU ou aux fichiers auxquels ils étaient affectés.

Erreurs

ERC 06 : il n'y a pas eu de commande pause.

PARD

But	Dumper le contenu d'une partition
Format	P A R D , N P

Signification des paramètres	NP : numéro de partition à dumper.
Commentaires	L'édition est effectuée sur la SU DO.
Erreurs	ERC 06 : pas de partition spécifiée ou numéro de partition incorrect.

TACD

But	Dumper une tâche présente en mémoire
Format	TACD, NT

Signification des paramètres

NT : numéro d'appel de la tâche ou nom du fichier (sans catalogue).

Commentaires

- L'édition du dump est effectuée sur la SU DO.
- La tâche doit être présente en mémoire centrale.
- Dans le cas où elle est active, l'utilisateur devra interpréter avec circonspection le contenu de la partition qui est susceptible d'évoluer pendant la visualisation.
- Toute la partition qui contient la tâche est dumpée.

Erreurs

ERC 03 : la tâche spécifiée n'existe pas, ou n'est pas présente en mémoire.

VIFI

But	Visualiser les fichiers ouverts à un instant donné
Format	VIFI

Commentaires

- Pour chaque fichier ouvert, on édite :

AFAU	: Adresse de la FU
USFN	: USR + FNUM
ABUF	: Adresse buffer de travail
LBUF	: Longueur buffer de travail
SLOB	: SLO buffer travail
ATIX	: Adresse TIX
LTIX	: Longueur TIX
SLOT	: SLO TIX
PP	: Primitive précédente :
S	: Simultané
W	: Ecriture autorisée
RWK	: Clé de partage
ADDF	: Adresse du DF
NOMFIC-PW	: Nom fichier
FU	: FU support
SID	: Type fichier

- Si la commande est émise en même temps qu'il y a des accès d'ouverture ou de fermeture de fichiers les informations affichées peuvent être erronées.

COMP

But Editer l'adresse d'une section du système

Format COMP, label [(± déplacement) (± label)]

Signification des paramètres

label : suite de caractères, donnant le nom d'une section du système
exemple : RTCOM, LSVC, DRVVM...

déplacement : nombre permettant de connaître directement l'adresse d'un mot situé en label + déplacement
exemple : COMP, RTCOM + '52
si RTCOM = 2A50, on aura '2AA2
On peut calculer la différence d'adresses entre 2 sections en remplaçant le déplacement par un autre label.
exemple : COMP, LSVC-RTCOM '00FF
si RTCOM = 2A50 et LSVC = 2B4F

Commentaires On peut également obtenir les adresses d'implantation, de lancement, de fin du système grâce aux commandes :

```
COMP, BEGIN
COMP, RUN
COMP, END
```

Pour que la commande fonctionne, il faut que l'article LABEL du fichier système existe.

Erreurs ERC 06 : l'article label n'existe pas, ou le label précisé n'existe pas.

VITU

But — Visualisation paramètres d'IOCS (TUP)

Format — VITU FU, [LT / LC / LF / ET / EC / EF] , [P / N] , [déplacement] [, nombre / valeur]

Signification des paramètres

FU : nom ou numéro de la FU dont on veut visualiser la TUP

LT : (implicite) lecture de la TUP. Dans ce cas, le paramètre P (implicite) signifie un déplacement positif, N un déplacement négatif.
nombre est le nombre de mots à visualiser

LC : lecture mot de commande. Les autres paramètres sont absents.

LF : lecture de l'adresse de la TUP contenue dans la TBF. Les autres paramètres sont absents.

ET : écriture dans la TUP. Les autres paramètres ont la même signification que pour LT, sauf "nombre" qui représente la valeur à mettre dans le mot adressé.

EC : écriture mot de commande.
les paramètres signe et déplacement sont omis, le paramètre "valeur" est obligatoire et représente la valeur à ranger dans le mot de commande.

EF : écriture de l'adresse de la TUP dans la TBF. Les paramètres sont les même que pour EC.

Commentaires

Cette commande est une commande "pseudo-système" qui est à utiliser avec précautions, en particulier lorsqu'on veut écrire dans une TUP. Aucun contrôle n'est fait sur les valeurs précisées en paramètres.

MAP

But	Visualisation du découpage d'une FU bootstrap
Format	MAP [,FU]

Signification des paramètres	FU : nom ou numéro de FU initiale d'un disque par défaut, on aura la liste de la FU Dl
Erreurs	ERC 06 : la FU est incorrecte

DSYS

But	Suppression d'un système dans la FU bootstrapp
Format	DSYS, N. SU/FU

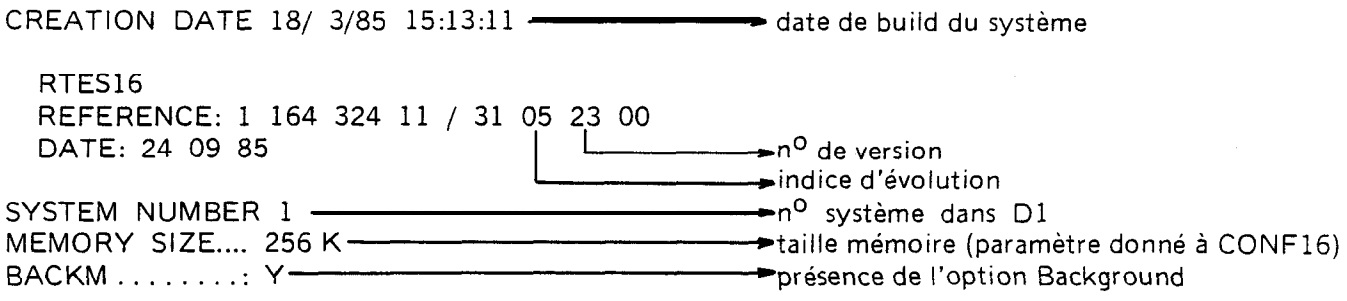
Signification des paramètres	N	Numéro de système à détruire [0,7]. On ne peut pas détruire le numéro de système correspondant au système en cours.
	SU/FU	: Numéro ou nom de FU (ou de SU) disque initiale. Par défaut, on prend Dl.

Erreurs	ERC 06	: paramètre incorrect numéro de système ou FU erronée.

INFOSYS

But	Connaître certaines caractéristiques du système
Format	INFOSYS

Cette commande permet d'éditer sur la SU DO les renseignements suivants :



32.6 - DIALOGUE HELP

Cette commande permet d'obtenir sur l'unité symbolique DO des informations relatives aux commandes des systèmes ou des processeurs disponibles sous RTES16, ainsi que la signification des différents messages d'erreurs émis.

Syntaxe :

— Mode d'emploi de HELP :

? (cr)

Liste des commandes de RTES16

? RTES16 (cr)

- Liste des commandes d'un processeur

? nomproc (cr) par exemple ? FUP2 (cr)

Syntaxe d'une commande

? nomcde (cr) par exemple ? FDES (cr)
ou ? TASK (cr)

— Signification d'un message d'erreur

? libellé erreur (cr) par exemple ?ERP 3 (cr)

— Liste des numéros de SVC

? SVC

— Liste des valeurs numériques des FU et SU

? FU

Commande FUHE

Syntaxe FUHE, FU

FU est une FU disque qui précise l'emplacement des fichiers Help (D2 par défaut)

Les données concernant les commandes RTES16 se trouvent dans le fichier <RTESl6-LP.

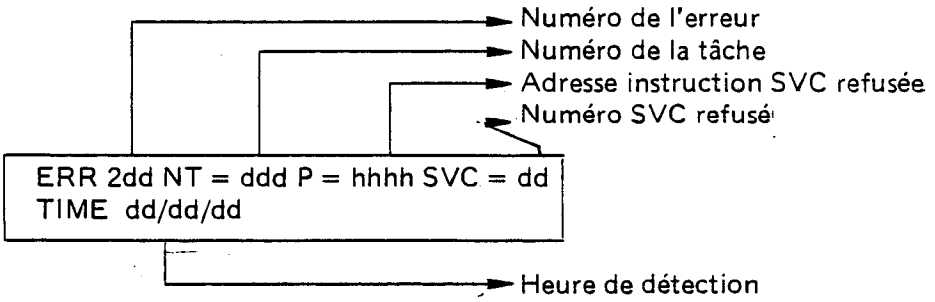
3.3 - DETECTION DES ERREURS

Un effort particulièrement vigilant a été porté dans RTES16 sur la détection des erreurs : l'exécution de chaque requête est sanctionnée par un compte rendu de requête, les commandes erronées sont suivies d'un message d'erreur, les alarmes de fonctionnement sont signalées par un message explicite...
Trois unités symboliques permettent à l'usager de diversifier le support de l'historique de son application selon les dispositifs périphériques dont il dispose.

Le but poursuivi par cette recherche d'un haut degré de détection des erreurs est multiple :
- accélérer la phase de mise au point de l'application
- donner un rôle déterminant à l'opérateur pendant cette phase
- assurer une sécurité maximum sur le système et les tâches "au point".

3.3.1 - Erreurs sur requêtes

Les erreurs signalées dans le descriptif de chaque requête sont des erreurs fatales ; elles provoquent toutes la suspension de la tâche appelante et l'impression d'un message d'erreur sur le dispositif périphérique associé à la SU TE (tasks errors).
Le format du message et sa signification sont les suivants :
(d : chiffre décimal ; h : chiffre hexadécimal)



Le numéro d'erreur (dd) imprimé est identique au compte rendu de requête.
A la suite d'une erreur sur requête, et après impression du message ci-dessus, l'opérateur peut relancer la tâche fautive par la commande RSUM ; la tâche se poursuit en séquence après la requête.
Si la requête erronée a été émise depuis 'une tâche hardware aucun message n'est imprimé, la tâche n'est pas suspendue, seul le compte rendu de requête (RPB) témoigne de l'erreuri

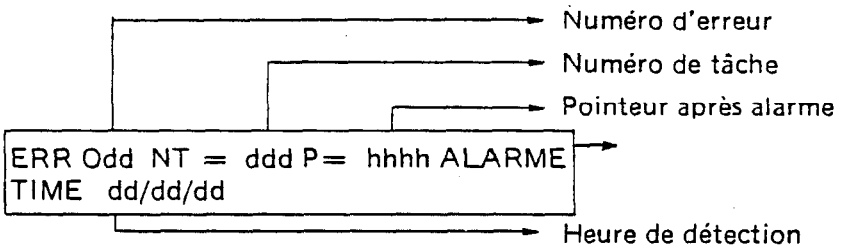
> Il est impératif de tester le compte rendu fourni par le système en retour des requêtes programmées : suite à un refus de requête, et en l'absence de traitement plus spécifique, il conviendra :
> - de libérer les ressources de la tâche (close de fichiers, RRLSE, DETACH...)
> - de terminer la tâche par la requête EXIT.
>
> Ces conditions sont nécessaires pour réaliser le remplacement de la tâche en erreur par une nouvelle tâche (produite en background par exemple).

3.3.2 - Erreurs de programmation

Ce sont les 10 erreurs qui provoquent l'activation de la tâche hardware de priorité 0. Chaque type d'erreur correspond au sous-niveau N de cette tâche.

N	
0	Mémoire inexistante
1	Erreur Protection mémoire DRPS
2	Erreur de parité hors du mode DEBUG
3	Instruction optionnelle inexistante
4	Instruction privilégiée exécutée en mode esclave
5	Instruction RQST, WAIT ou QUIT exécutée sous niveau hardware
6	Appel d'une autre unité de traitement : IPI
7	Exécution illégale de l'instruction STEP
8	Point d'arrêt (mode DEBUG)
9	Exécution illégale de l'instruction ACTD.

Le message d'erreur est également délivré sur le périphérique affecté à la SU TE (task errors)
Il a le format suivant :



L'instruction ACTD peut ainsi être utilisée pour produire une alarme signalée par le système à l'opérateur ; comme pour les erreurs sur requêtes, la tâche fautive est suspendue, et, dans le cas de l'alarme ACTD, pourra être relancée par la commande RSUM. Dans les autres cas, essentiellement les erreurs de type 4 (instruction privilégiée), 0 (mémoire inexistante), et 1 (protection mémoire), la tâche ne pourra pas être relancée : de telles erreurs doivent être décelées lors d'une mise au point préalable du programme en background (avec l'utilitaire DRIP16 par exemple).
Si l'alarme a été provoquée par une tâche hardware aucun message d'erreur n'est imprimé et la tâche n'est pas suspendue (les tâches hardware sont supposées au point).

3.3.3 - Erreurs sur commandes opérateur

La signification de ces erreurs et le traitement associé sont explicités au paragraphe 3.2.5
Le message délivré sur le dispositif affecté à l'unité symbolique DO a le format suivant :

ERC dd ---------- dd : numéro d'erreur

3.3.4 - Alarmes "Système"

On regroupe sous cette rubrique un certain nombre d'incidents ou d'aléas de fonctionnement perçus par le système et signalés sous la forme d'un message à l'opérateur sur le périphérique affecté à l'unité symbolique SA (System Alarms)
Ce message a le format général :

ERS dd [< information complémentaire >]
[TIME dd/dd/dd]

A la suite de tels défauts, et selon leur gravité, l'application continue à fonctionner, éventuellement dans un mode dégradé dont on précise ici les caractéristiques.

ERS 01 SYSTEM RELOAD

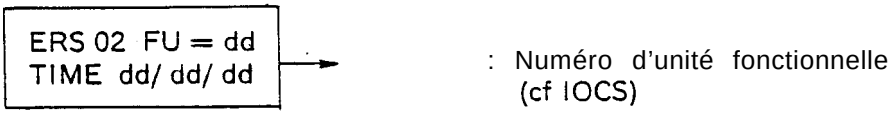
Ce message signale une disparition suivie d'une réapparition du secteur.
Dans RTES16 la séquence activée au moment du restart automatique réalise les actions standard suivantes :

- Sauvegarde en fond de mémoire centrale des informations rémanentes : ZDR, phases d'avancernent. état des événements, date et heure.
- Fermeture de tous les fichiers permanents et destruction des fichiers temporaires (EOJ généralisé)
- Chargement en mémoire de la dernière application sauvegardée par SAVE ou relancée par INIT (soit encore, chargement en mémoire de l'application en cours).
- Récupération des informations rémanentes ci-dessus.
- Activation de la tâche de l'application précisée lors du dialogue de configuration du système ; si cette tâche n'existe pas, ou si aucune tâche n'a été précisée dans ce dialogue, activation du dialogue opérateur.

Dans tous les cas, il y a impression sur le dispositif associé à l'unité SA du message ci-dessus.

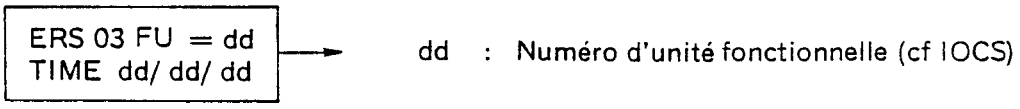
La tâche spécialisée de l'application activée sur restart dispose des informations rémanentes restituées et des fichiers sur disque dans l'état où ils se trouvaient au moment du défaut secteur.

L'heure interne devra être rétablie par dialogue opérateur dès que possible.

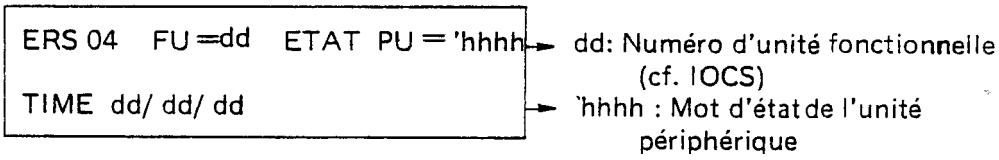


ERS 02 - Ce message signale l'occurence d'un appel "4 coups", c'est à dire de 4 appels opérateur, sur un périphérique tel que le téléimprimeur de service, un téléimprimeur ou un dispositif de visualisation connecté par un coupleur asynchrone 1 voie ou 8 voies.

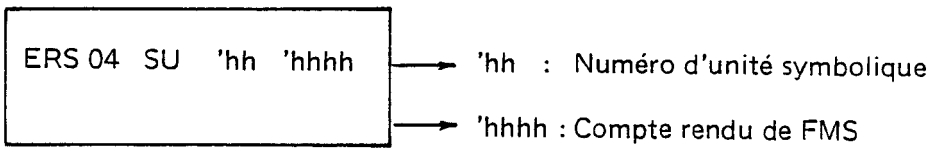
Un tel défaut peut intervenir lorsqu'à la suite du mauvais fonctionnement d'une tâche de l'application qui "boucle"), la tâche système IDLE de plus faible priorité ne peut réaliser sa séquence de scrutation de la table des mots d'état des périphériques et déceler ainsi un appel opérateur: l'occurence du 4ème appel provoque alors une alarme émise par le driver.
Après impression de ce message, le dialogue opérateur de RTES 16 est activé.



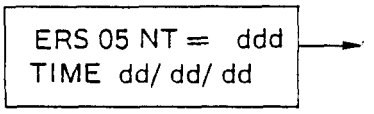
ERS 03 - Ce message signale l'occurence d'un appel opérateur sur un téléimprimeur ou un dispositif de visualisation alphanumérique. Après impression de ce message, le dialogue opérateur de RTES est activé.



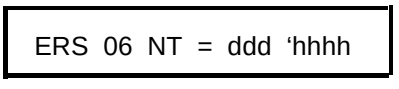
ERS 04 - Ce message signale l'arrivée d'un défaut de périphérique (autre qu'un appel opérateur) et indique le mot d'état de l'unité périphérique associée.
Pour l'exploitation de ce mot d'état, on se reportera au Manuel de Référence d'IOCS.
L'occurence d'un tel défaut ne provoque pas l'activation du dialogue opérateur.



Ce message signale une erreur hors d'un accès à un fichier affecté à une SU, par un appel IOCS commuté en appel FMS.
Pour l'exploitation du compte rendu FMS, se reporter au Manuel de Référence FMS.



ERS 05 - Ce message indique qu'un appel périodique au moins de la tâche NT a été perdu, le traitement réalisé par cette tâche n'étant pas terminé lorsqu'au début d'un cycle le module horloge s'apprête à l'activer.
Ce message constitue simplement un "warning" : le système continue à activer cycliquement la tâche NT.

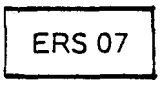


ERS 06 - L'alimentation en mémoire d'une tâche non résidente peut être rendue momentanément impossible pour diverses raisons (défaut disque, saturation des zones du système...)
Dès lors, le message ci-dessus est imprimé, avec :

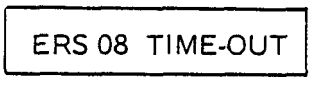
ddd : Numéro de la tâche non résidente
'hhhh : Compte rendu FMS

Exemple : ERS 06 NT= 108 '6020

approvisionnement impossible pour cause de saturation des zones dynamiques du système.
L'activation de la tâche NT est alors perdue.



ERS 07 - Ce message intervient lorsque le système n'est pas en mesure de réaliser le swap du background pour cause de défaut permanent du disque système.
L'activité batch est alors interrompue ; elle ne pourra être relancée qu'après initialisation du système.



ERS 08 - Ce message intervient lorsque, le dialogue opérateur étant en attente de commandes, aucun caractère n'a été frappé depuis plus de 20 secondes.
Une fois ce message imprimé, le dialogue est interrompu jusqu'au prochain appel opérateur (voir paragraphe 3.2.1).

3.4 - REQUETE D'EXECUTION DE COMMANDE DE L'OPERATEUR (SVC COMAND)

Principe :

La requête COMAND permet de demander l'exécution d'une commande opérateur dans les mêmes conditions que si elle était émise par l'opérateur. Elle permet de réaliser automatiquement des commandes comme INIT, MONT, DMONT qui n'existent pas sous forme de requêtes.

Cette requête est paramétrée par le texte de la commande. Le caractère "retour chariot" (rc) sert de délimiteur.

La requête peut être refusée soit parce que le demandeur ne satisfait pas certaines conditions (il n'y a pas d'exécution de commande), soit parce que la commande demandée a été refusée (donnant lieu à une erreur ERC).

Dans le premier cas, les erreurs 4 et 6 sont considérées comme des erreurs sur requêtes et la tâche émettrice est suspendue. Les autres erreurs ne provoquent pas de suspension de la tâche.

Les erreurs ERC sont transmises en compte-rendu sous forme binaire.

Si la commande demandée édite un compte rendu sur DO en retour d'exécution, celui-ci est perdu.

Plusieurs tâches peuvent utiliser la requête COMAND, mais une seule commande est exécutée à la fois.
L'ordre d'exécution est gérée par le processus de gestion des priorités des tâches.

Mise en œuvre

La requête COMAND est une option intégrée au processeur REQCOM.

La partition dans laquelle sera intégrée le processeur REQCOM doit être résidente et comprise entre 0 et 64 K.

Le processeur REQCOM est livré sous forme de binaire translatable dans le fichier REQC16-BT.

1 Build du processeur

```
CALL BUILD
BI REQC16-BT
MLOD  'XXXX = adresse de début de la partition

CATA IM, REQCOM-:S
```

2 Intégration du processeur

```
EXEC, R, REQCOM-:S, P      P= N° de la partition
```

Contraintes d'utilisation

- Utilisation de Dl et DO :

  Le module REQCOM modifie les affectations de Dl et DO pendant l'exécution de la requête COMAND. C'est pourquoi les tâches ne doivent pas faire des entrées et des sorties, sur Dl et DO lorsqu'une tâche exécute la SVC COMAND.

• Utilisation de la FU ME :

La FU ME est, elle aussi, utilisée. Les tâches de l'application ne peuvent donc pas en disposer si le processeur REQCOM est intégré.

• Dialogue inactif :

Le dialogue doit être inactif (tâche DIALOG désarmée) pour que la requête COMAND soit acceptée. Si ce n'est pas le cas, elle est refusée avec un compte rendu d'erreur égal à 15.

• Mot de passe :

Toute requête COMAND supprime la protection d'accès au dialogue opérateur demandée éventuellement dans la dernière commande KEND (mot de passe mentionné).

Description de la requête :

COMAND

But

Demander l'exécution d'une commande opérateur.

Syntaxe Fortran

CALL COMAND (TEXT, IERR)

TEXT : tableau entier contenant le texte de la commande sous forme d'une suite de caractères ASCII.

IERR : variable entière chargée par la réponse à la requête :

1	requête acceptée
⩾ 2	requête refusée

Syntaxe Assembleur

Format de la requête :

```
LAD  RPB
SVC  COMAND
```

Table des paramètres :

Adresse du texte de la commande	TEXT
Adresse de compte rendu de requête	IERR

Numéro de la SVC = 76 ('4C)

Erreurs

IERR = 2, 3, 5 7 à 14
Erreurs provoquées pendant l'exécution de la commande. Elles ont la signification des erreurs ERC de même numéro.

IERR = 4 . Requête interdite aux tâches hardware.
Erreur ERC 04 provoquée par la commande.

IERR = 6 . II existe un paramètre incorrect
- adresse de RPB
- adresse de compte rendu de requête
- adresse du texte de la commande
- erreur ERC 06 provoquée par la commande (paramètre de la commande incorrect).

IERR = 15 . Le dialogue opérateur est actif.

Commentaires

1 Les erreurs 4 et 6 ne correspondant pas à des erreurs ERC détectées lors de l'exécution de la commande sont considérées comme des erreurs sur requêtes (la tâche fautive est suspendue). Les autres ne le sont pas.

2 Si la commande édite sur DO un message en retour d'exécution, celui-ci est perdu.

4 - LE MODE PRIVILEGIE

RTES16 offre sur le 16-70 un mode d'exécution supplémentaire : le mode PRIVILEGIE.

Définition :

Le mode privilégié est un mode maître translaté.
La base de translation est le registre SLO.
On peut donc implanter des programmes privilégiés dans toute la mémoire. Ces programmes ont accès à toutes les instructions de la machine (DIT, EIT, RQST, RLSE,...).
Par contre, les instructions LAR, STAR, RDOE et WOE sont déconseillées, du fait que l'adressage est basé par SLO, en particulier WOE qui modifie brutalement la base.

Intégration à RTES16 :

Pour pouvoir utiliser le mode privilégié sous RTES16, il faut :

- disposer d'un processeur Solar 16-70 (le 16-35 ne gère pas ce mode),
- avoir généré le système avec la macro-instruction %MODPRI (voir manuel d'utilisation), dans ce cas, on aura automatiquement FMS en mode privilégié, ce qui apporte un gain de place important entre 0 et 64K.

Utilisation :

RTES16 peut gérer des tâches et des requêtes en mode privilégié.
Ces programmes seront buildés en mode esclave. On précisera le mode de fonctionnement au moment de l'intégration au système par les commandes TASK et EXEC.

Contraintes :

- Une requête doit obligatoirement retourner au système par la SVC RETOUR qui effectue le retour au mode maître. De plus, la kstore doit être dans le même état qu'à l'entrée dans la requête.

Réseaux
et systèmes
d'information

Bull

ANNEXES TECHNIQUES

1 - Contenu initial de la PST d'une tâche

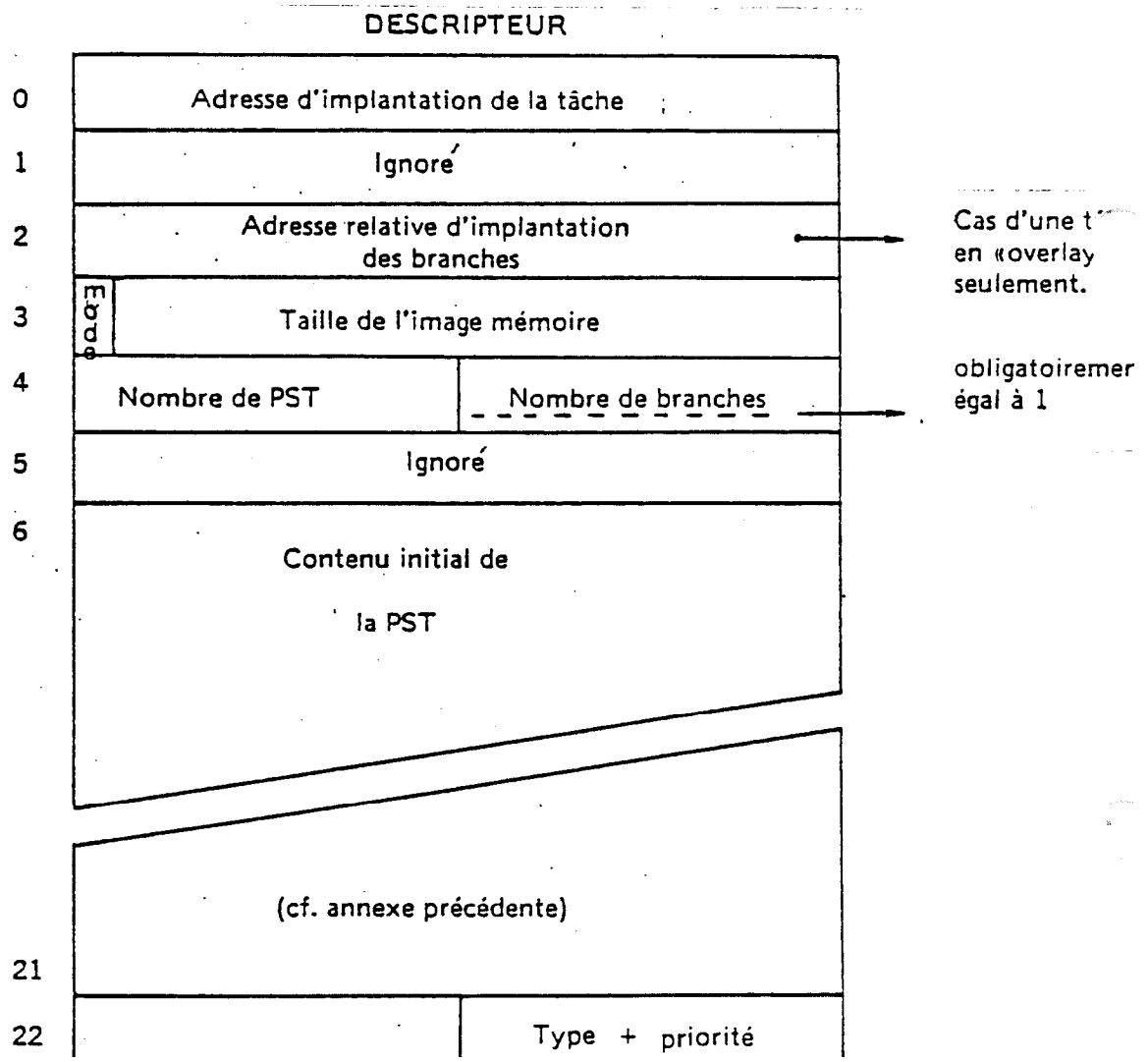
Les 16 mots qui suivent la directive ASSEMBLEUR PSTS (ou PSTH) ou la déclaration de tâche PL16 sont interprêtés, au moment de l'intégration de la tâche sous RTES16 de la façon suivante :

0	Valeur initiale du registre A	
1	- - - - - B	
2	– – – – – X	
3	– – – – – Y	
4	- - - - - C	
5	– – – – – L	
6	- - - - - W	
7	- - - - - K	
8	Adresse de première activation (P)	
9	Valeur initiale de S	
10	ignoré	
11	ignoré	
12	NT : numéro d'appel de tâche	USR : numéro d'usager (FMS)
13	Nombre maximum d'appels cumulables	
14	USRP numéro d'usager public (FMS)	Numéro partition
15	Réserve	

Pour les tâches hardware seuls les mots 4, 7, 8 et 9 sont significatifs.

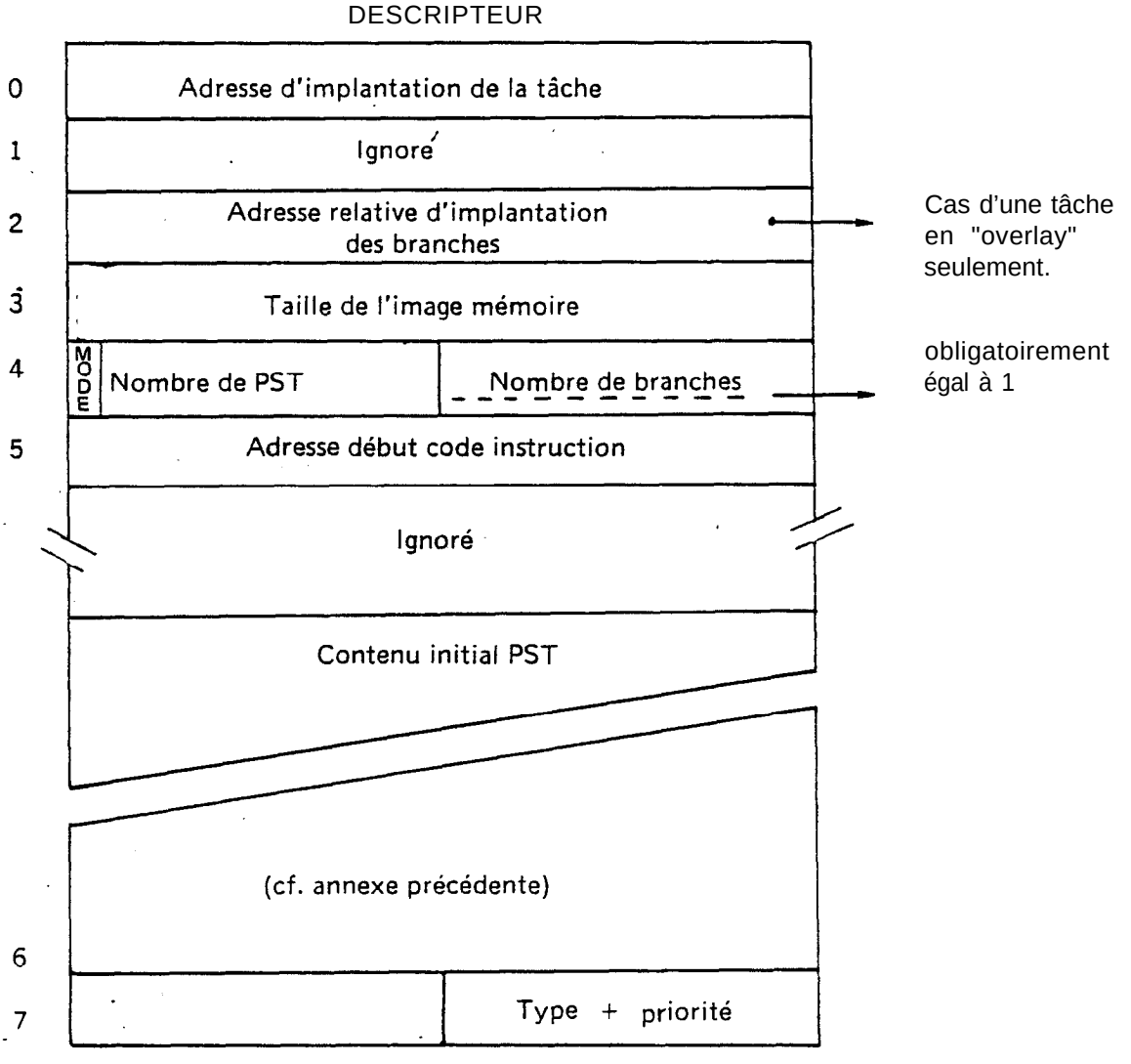
2 - PROFIL DES DESCRIPTEURS DE FICHIER IMAGE MEMOIRE

2.1 - Profil du descripteur d'un fichier image mémoire < 32 K



2.2 - Profil du descripteur d'un fichier image mémoire > 32 K et < 64 K

Le "chargement" d'une tâche sous RTES16 (commande TASK) consiste dans un premier temps à exploiter les informations recueillies par le chargeur disque (BUILDER) et situées dans le descripteur DESC64 du fichier support de la tâche. L'interprétation de ces informations est la suivante :



3 - Unités symboliques et fonctionnelles de IOCS

- Affectations possibles (signe x)
- Affectations standard (signe 0)

Unités symboliques / Unités Fonctionnelles	'80 SO	'81 SI	'82 BO	'83 BI	'84 LL	'85 CC	'86 LO	'87 EC	'88 EL	'89 PC	'8A U1	'8B U2	'8C U3	'8D U4	'8E U5	'8F U6	'90 DO	'91 DI	'92 TE	'93 SA	'94à'9C U7àUF
ME '7F	x	x	x	x							x	x	x	x	x	x					x
ZE '0	x		x				x		x		(x)	(x)	(x)	(x)	(x)	(x)					(
TR '1		x		x		x		x		x	x	x	x	x	x	x		x			x
TS '2	x				(x)		(x)		(x)		x	x	x	x	x	x	(x)		(x)	(x)	x
TK '3		x				(x)		(x)		(x)	x	x	x	x	x	x		(x)			x
TP '4	x		x		x		x		x		x	x	x	x	x	x	x		x	x	x
HR '5		(x)		x		x		x		x	x	x	x	x	x	x		x			x
HP '6	(x)		(x)		x		x		x		x	x	x	x	x	x	x		x	x	x
CR '7		x		x		x		x		x	x	x	x	x	x	x		x			x
LP '8	x				x		x		x		x	x	x	x	x	x	x		x	x	x
T1 à T4 '9 à 'C	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
D1 à D8 'D à '14				x (D2)							x	x	x	x	x	x					x
F1 à FF '15 à '23	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
D9 à DF '24 à '2A				x							x	x	x	x	x	x					x
E1 à EF '2B à '39				x							x	x	x	x	x	x					x

Remarques :

1) - Le nombre des unités fonctionnelles d'une installation peut être supérieur à celui précisé dans le tableau (la seule limitation est celle introduite par GENIO : 125 au maximum)
Les unités fonctionnelles de numéro supérieur à '39, ou 57, ne seront pas accessibles par dialogue opérateur (commandes d'affectation SU FU), ceci à l'exception de la FU ME de numéro 127.

2) - Une installation peut n'inclure qu'un sous-ensemble des unités fonctionnelles du tableau ci-avant ; les FU suivantes sont néanmoins imposées :
- TS, TK
- HR
- D1, D2

et, pour une installation fonctionnant avec le background de RTES16 :
- F7, F8

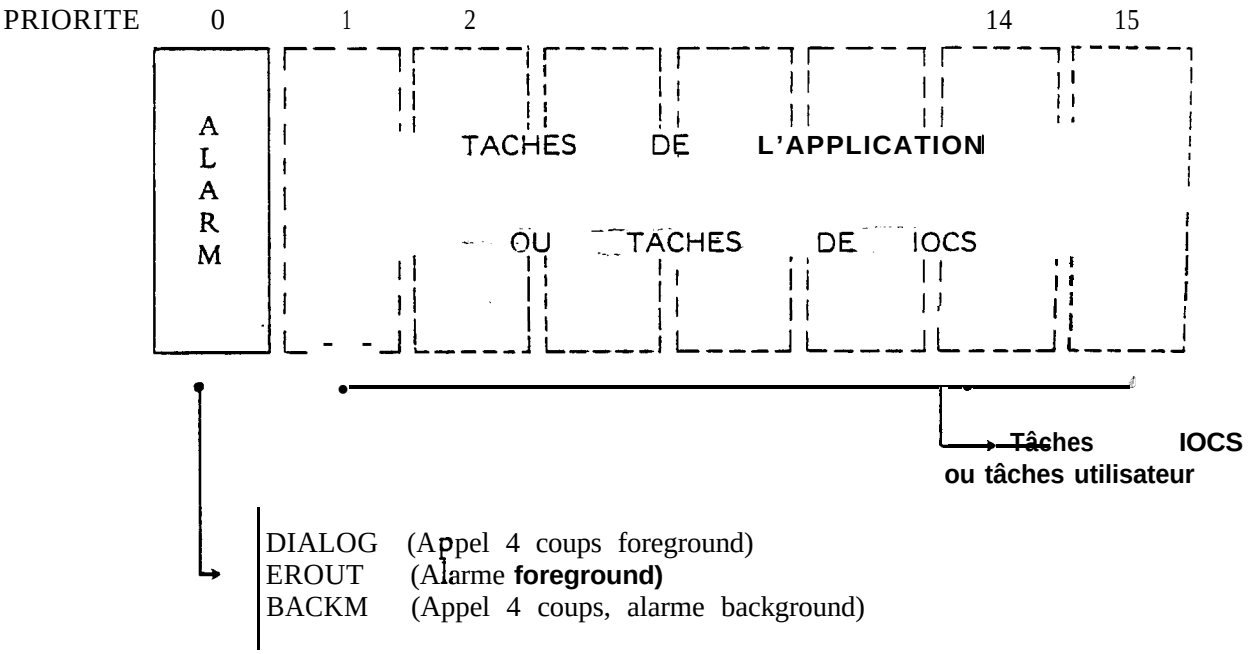
pour la simulation des fonctions TS, TK de la seconde console.

3) - Les unités fonctionnelles disque auront de préférence les numéros correspondant aux mnémoniques :
- D1 à D8 soit 'D à '14
- D9 à DF " '24 à '2A
- E1 à EF " '2B à '39

En effet, seules ces FU peuvent être gérées par FMS d'une part (la FU bootstrap D1 exceptée) d'autre part la commande MONT, utilisée lors d'un changement de cartouche, ne connaît en paramètre que ces seuls mnémoniques.

4 - Taches hardware gérées par RTES16

Le système gère en standard **la priorité hardware 0.**
Les autres priorités sont, soit gérées par IOCS, soit disponibles pour les traitements spécifiques de l'application.



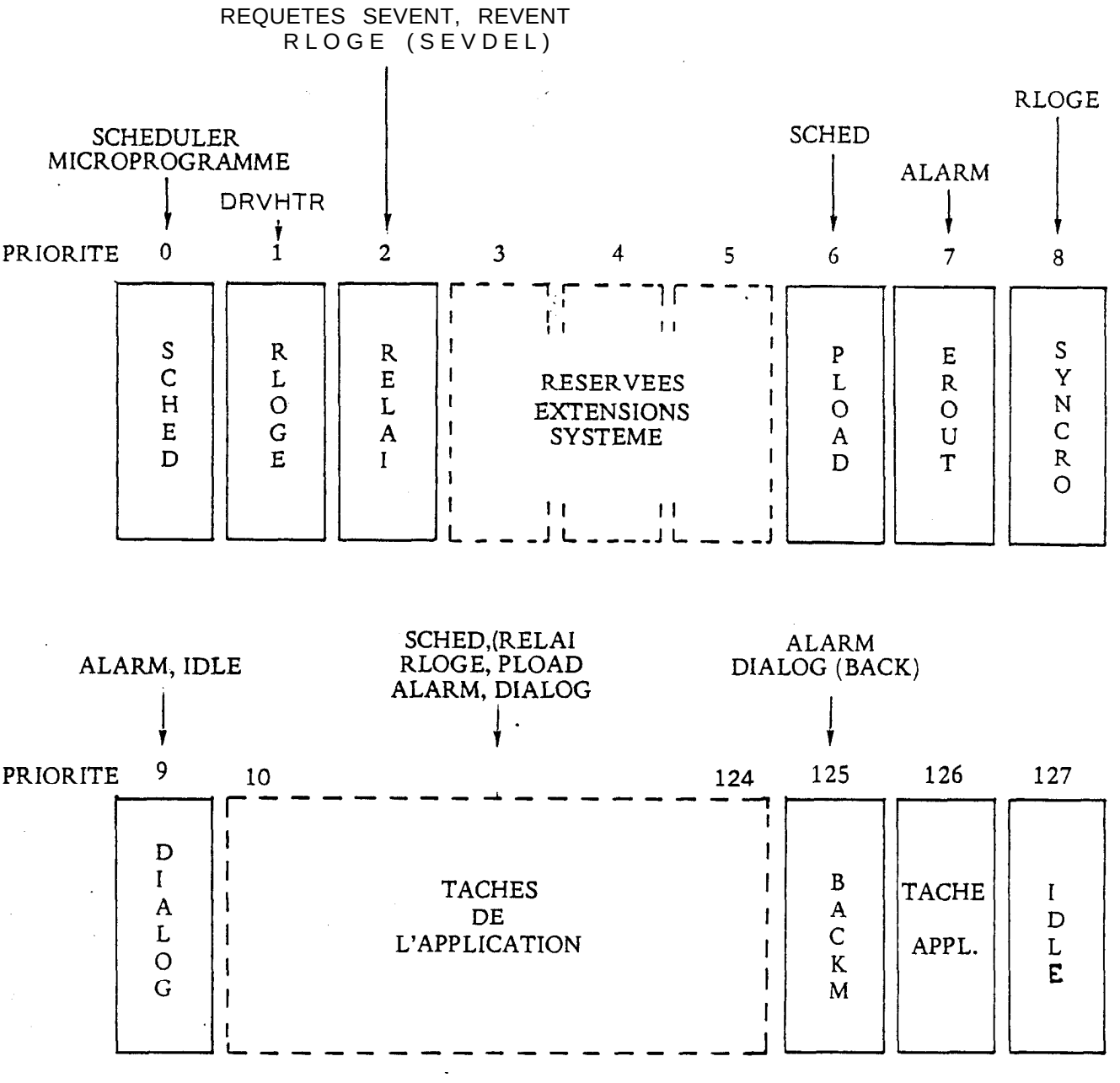
ALARM

- **Tâche-la plus prioritaire de la machine.**
- **Réalise la prise en compte et le pré-traitement des alarmes de programmation (suspension de la tâche fautive).**
- **Active la tâche software EROUT d'impression d'un message d'alarme si la tâche fautive est une tâche de l'application, relance le moniteur background BACKM sur une séquence de traitement d'alarme si le fautif est un processeur background.**
- **Relance la tâche de dialogue DIALOG sur appel 4 coups émis sur le téléimprimeur système, ou le moniteur BACKM sur appel 4 coups émis sur la console background.**

HORLOGE TEMPS REEL

- **L'horloge temps réel gérée par RTES16 a une période multiple de 10 ms (fixée à laconfiguration de IOCS).**
- **Un driver "DRVHTR" intégré au noyau du système RTES16 assure la prise en compte des interruptions provenant de l'horloge.**
- **Le niveau de priorité des interruptions est défini lors de la phase de génération du système (voir manuel d'utilisation de RTES16 - paragraphe 5.2).**

5 - tâches software "SYSTEME") de RTES16 :



SCHED

. Activée par le scheduler microprogrammé ; paramètre d'appel : la tâche NL est non prête.
. Analyse le type de complément de scheduling à réaliser
- traite les inhibitions validations de tâches
- prépare l'alimentation d'une tâche temps réel : allocation d'une partition et appel de la tâche PLOAD pour approvisionnement en mémoire

RLOGE
. Activée parle driver DRVHTR pour effectuer une opération différée
. Réalise cette opération (activation de tâche, réveil après temporisation, création d'événement...)
. Ordonne la file des opérations sur horloge (pour optimiser le temps de réponse de HORL)

RELAI
. Tâche relai des tâches hardware pour certaines requêtes programmées (gestion des événements)
. Réalise ces requêtes programmées (optimisation des temps de réponse des tâches hardware)

NIVEAUX 3, 4, 5 . Niveaux très prioritaires réservés à des extensions système.

PLOAD
. Effectue l'approvisionnement en mémoire d'une tâche (racine du fichier associé)
. Eventuellement, réalise au préalable le swap-out du processeur background
. Effectue le swap-in du processeur background lorsque la partition background est libre

EROUT
. Imprime les messages d'erreurs sur requête et d'alarme de programmation sur le dispositif périphérique associé à l'unité symbolique TE (Task Errors)

SYNCRO
. Réalise l'impression sur le périphérique associé à l'unité symbolique SA (System Alarms) des messages ERS 05 : asynchronisme d'une tâche cyclique avec le module horloge.

DIALOG
. Tâche de dialogue opérateur
. Assure la prise en compte et le traitement des commandes de RTES16
. Utilise les périphériques associés aux unités symboliques Dl (Dialog Input) en entrée et DO (Dialog Output) en sortie.
. Imprime sur le périphérique associé à l'unité symbolique SA (System Alarms) les messages de défauts de périphériques.

BACKM
. Moniteur et processeurs background
. Le moniteur d'enchaînement des travaux BACKM et les processeurs exploités sous BACKM s'exécutent sous une seule et même priorité : celle qui a été préciséedans la commande MLOD du builder lors du chargement sur disque de BACKM. Par défaut, cette priorité est égale à 126. Sur l'exemple, priorité de BACKM = 125.

IDLE
. Tâche butée du scheduler
. Réalise une scrutation de la table des défauts de périphériques et/ou appels opérateur et active DIALOG.
. Affiche sur les voyants du pupitre la charge de l'unité centrale. (Plus le nombre de voyants allumes-est grand, plus la charge est importante).

6 - Synoptique des requêtes programmées

But de la requête	Mnémonique ou symbole valeur	Valeur du symbole	Profondeur dans KSTORE
Réaliser une opération d'E/S ou positionner un périphérique	IOCS	'08 (4) 8	30
Attendre la fin d'une opération d'E/S	WEIO	'09 9	30
Transmettre la table des commandes associées à un processeur	CAMO	'0A 10	30
Tester les appels de l'opérateur et les défauts de périphérique	TAPS	'08 11	15
Transmettre au superviseur l'état de fin d'un processeur	ABOS	'0C 12	30
Retourner au superviseur une adresse de programme traitant les échanges mémoire	EMAD	'0D 13	30
Demander l'état communiqué au superviseur par ABOS	RBOS	'0E 14	15
Communiquer au superviseur l'adresse d'un nouveau sous-programme superviseur	NEWS	'0F 15	10
Demander les adresses des sections COMMON de RTES-D et IOCS	TEST (1)	'10 (4) 16	10
Acquisition de la FU affectée à une SU	AFSU	'14 20	10
Activer une tâche et lui transmettre un paramètre de travail	RUN	'15 (4) 21	30
Faire l'acquisition du paramètre de travail	DATAG	'16 22	30
Terminer une tâche	EXIT (1)	'17 23	30 (2)
Inhiber une tâche	INHIB	'18 24	30

(1) Symbole réservé du langage PL 16. Employer CTEST et CEXIT.

(2) Pour une tâche non résidente avec overlay, la profondeur exigée par EXIT est plus importante : 60 mots (en rapport avec BCHLR).

(4) Les SVC de numéros 0 à 7 et 18 et 19 sont utilisables par SVC NEWS.
La SVC 17 est réservée à l'utilisation du module FDM (Floppy Disk Management Monitor) : ne pas utiliser pour l'intégration d'un autre module.

But de la requête	Mnémonique ou symbole valeur	Valeur du symbole	Profondeur dans KSTORE
Faire l'acquisition du mot d'état d'un d'un périphérique	RMDEF	'19 (4) 25	30
Faire l'acquisition des limites de la zone libre au delà de 32 K	FREM 64	'1A 26	30
Signaler l'arrivée d'un événement	SEVENT	'1B (4) 27	30
Attendre l'arrivée d'un événement	WEVENT	'1C 28	30
Signaler la disparition d'un événement	REVENT	'1D 29	30
Tester l'arrivée d'un événement	TEVENT	'1E 30	30
Temporiser une tâche	WAIT (1)	'1F 31	30
Réactiver une tâche software en attente sur un sémaphore privé	RACT	'20 32	30
Mettre une tâche software en attente sur un sémaphore privé	RWAIT	'21 33	30
Demander un accès à une ressource	RRQST	'22 34	30
Libérer un accès à une ressource	RRLSE	'23 35	30
Lire une zone de données résidentes	REDGET	'24 36	30
Ecrire une zone de données résidentes	REDPUT	'25 37	30
Connaître la phase d'avancement	STAGET	'26 38	30
Définir la phase d'avancement	STADEF	'27 (4) 39	30
Affecter-une unité fonctionnelle à une unité symbolique	SUFU	'31 49	80
Faire l'acquisition des limites de la zone libre en dessous de 32 K	FREEM	'33 51	30
Faire l'acquisition du numéro et des bornes de la partition utilisée	FREEMA	'34 (4) 52	30
Lire la date et l'heure	TIME	'35 (4) 53	30
Initialiser la date et l'heure	WTIME	'36 54	30
Requêtes à FMS du niveau OPEN/ CLOSE	FMS	'38 56	60
Requêtes à FMS du niveau portion d'article			
Requêtes à FMS du niveau article en accès indexé	FMSI	'3A 58	60
Requêtes à FMS du niveau article en accès direct	FMSD	'3B (4) 59	
Charger et lancer une branche	BCHLR	'3D (4) 61	65
Retourner à l'appelant, branche ou racine	BACK	'3E 62	65
Retourner à la racine	BACKR	'3F 63	30

(1) Symboles réservés du langage PL 16. Employer CWAIT par exemple.
(4) Les SVC 40 à 50, 55 et 60 sont réservées pour des extensions futures du système.

But de la requête	Mnémonique ou symbole valeur	Valeur du symbole		Profondeur dans KSTORE
Activer une tâche avec délai, période et paramètre de travail	START (1)	'40	64	30
Activer une tâche avec délai, paramètre de travail et attente d'un compte rendu	STARTW	'41	65	30
Activer une tâche à une heure donnée, avec période et paramètre de travail	TRNON	'42	66	30
Activer une tâche à une heure donnée avec paramètre de travail et attente d'un compte rendu	TRNONW	'43	67	30
Supprimer des appels à une tâche	OFF	'44	68	30
Attendre l'arrivée de tous les événements d'une liste	WEVAND	'45	69	30
Attendre l'arrivée de l'un des événements d'une liste,	WEVOR	'46	70	30
Signaler l'arrivée différée d'un événement	SEVDEL	'47	71	30
Définir une ressource usager	RESDEF	'48	72	30
Supprimer l'arrivée différée d'un événement	OFFDEL	'49	73	30
Activer une tâche fille	TCALL	'4A	74	30
Activer une tâche fille avec attente	TCALLW	'4B	75	30
Exécution d'une commande de l'opérateur	COMAND	'4C	76	30
Envoyer un bloc d'informations dans une zone système	SEND	'4D	77	30
Recevoir un bloc d'information stocké dans le système	RECEIV	'4E	78	30
Créer une tâche	TAS KC	'4F	79	45
Tuer une tâche	KILL (4)	'50	80	30
ZWB	GESPAV	'64		3 0
RETOUR du mode privilégié	RETOUR	'FF	255	8

(1) Symboles réservés du langage PL16 Employer CSTART
(2) Utilisé par le processeur REQCOM
(4) Les SVC de numéros 81, 82, 83, 84 sont utilisées par le moniteur BACKM pour communiquer avec RTES16.

7 - Syntaxe fortran des requêtes de type temps réel

RUN a task	CALL RUN (NT, IERR [,IPAR])
START a task	CALL START (NT, ID, IUD, IERR [,IPAR [,IP, IUP]])
START and Wait	CALL STARTW (NT, ID, IUD. IERR. IPAR, ICR)
TuRNON a task	CALL TRNON (NT, ITIME, IERR [,IPAR [,IP ,IUP]])
TuRNON and Wait	CALL TRNONW (NT, ITIME, IERR, IPAR, ICR)
Task WAIT	CALL WAIT (ID, IUD, IERR)
DATA Get	CALL DATAG (IDATA)
Task EXIT	CALL EXIT [(ICRDU)]
Calls OFF	CALL OFF (NT, IERR, IPAR)
Task INHIBition	CALL INHIB (NT, IERR)
Wait EVENT	CALL WEVENT (IEV, IERR)
Wait EVents : AND	CALL WEVAND (IERR, IA, IB, IC....)
Wait EVents : OR	CALL WEVOR (IERR, ICR, IA, IB, IC...)
Set EVENT	CALL SEVENT (IEV, IERR)
Set EVent with DELay	CALL SEVDEL (IEV, IERR, ID, IUD)
OFF event with DELay	CALL OFFDEL (IEV, IERR)
Reset EVENT	CALL REVENT (IEV, IERR)
Test EVENT	CALL TEVENT (IEV, IERR, ICR)
REsident Data GET	CALL REDGET (IN, IERR, IDEP, IADR)
REsident Data PUT	CALL REDPUT (IN, IERR, IDEP, IADR)
RESource DEFine	CALL RESDEF (IR, IAC, IERR)
ReQueST an access	CALL RRQST (IR, IERR)
ReLeaSE an access	CALL RRLSE (IR, IERR)
SEND a data block	CALL SEND (IADR, IERR, ICR)
RECEIVe a data block	CALL RECEIV (IADR, IERR, ICR)
TASK Création	CALL TASKC (NT, IERR, INAM, ISTAT, IUSR, IAPPMAX, IUSRP, IPRTY)
KILL	CALL KILL (NT, IERR)

STAge DEFine	CALL STADEF (IPH, IERR)
STAge GET	CALL STAGET (NT, ICR, IERR)
get TIME	CALL TIME (IDATIM, IERR)
WAIT for activation	CALL RWAIT (IDEP, IERR)
ACTivate a task	CALL RACT (IDEP, IERR [,IPAR])
Task CALL	CALL TCALL (NT, IFU, IERR, IPAR, NOMFIC)
Task CALL and Wait	CALL TCALLW (NT,IFU,IERR,ICR,IPAR,NOMFIC)
Analog Input SeQuential (1)	CALL AISQ (INV, JADR, KTAB, IERR)
Analog Input SeQuential and Wait	CALL AISQW (INV, JADR, KTAB, IERR)
Analog Input R an Dom	CALL AIRD (INV, JADR, KTAB, IERR)
Analog Input R an D om and Wait	CALL AIRDW (INV, JADR, KTAB, IERR)
Analog Output	CALL AO (INV, JADR, KTAB, IERR)
Analog Output and Wait	CALL AOW (INV, JADR, KTAB, IERR)
Digital Input	CALL DI (INV, JADR, KTAB, IERR)
Digital Input and Wait	CALL DIW (INV, JADR, KTAB, IERR)
Digital Output Latching	CALL DOL (INV, JADR, KTAB, MTAB, IERR)
Digital Output Latching and Wait	CALL DOLW (INV, JADR, KTAB, MTAB, IERR)
Digital Output Momentary	CALL DOM (INV, JADR, KTAB, MTIME, IERR)
Digital Output Momentary and Wait	CALL DOMW (INV, JADR, KTAB, MTIME, IERR)

(1) voir manuel de référence BIBIND pour toutes les requêtes spécifiques aux entrées - sorties industrielles.

8 - Liste des paramètres des requêtes :

NT	Variable ou constante entière représentant un numéro d'appel d'une tâche
IERR	Variable entière chargée par la réponse à la requête ERR = 1 requête acceptée ERR ⩾ 2 requête refusée
IPAR, IDATA	Variable ou constante entière qui représente un paramètre de travail transmis à une tâche.
ID	Variable ou constante entière qui représente un délai en unités spécifiées par IUD, ou en tops de 20 ms
IUD	Variable ou constante entière qui précise l'unité du délai. UD = 0 top de l'horloge de base IUD = 1 milliseconde UD = 2 seconde UD = 3 minute
IP	Variable ou constante entière qui représente une période en unités spécifiées par IUP.
IUP	Variable ou constante entière qui précise l'unité de la période UP = 0 top de l'horloge de base UP = 1 milliseconde UP = 2 seconde UP = 3 minute
ICR, ICRDU	Variable entière représentant le compte rendu d'exécution de la requête
ITIME	Tableau entier de 3 mots qui précise l'heure d' (e) (première) activation d'une tâche TIME (1) : heure ITIME (2) : minute ITIME (3) : seconde
IFU	Variable ou constante entière qui précise un numéro de FU disque,
NOMFIC	Tableau entier de 4 mots précisant un nom de fichier.
IEV	Variable ou constante entière qui précise un numéro d'événement
IA, IB, IC...	Variables ou constantes entières représentant une liste d'événements d'une même classe.
IDEP	Variable ou constante entière représentant un déplacement depuis le Début de la zone de données résidentes;
IN	Variable ou constante entière précisant le nombre de mots à transférer
IADR	Variable entière ou tableau entier précisant le nom d'un buffer.
IR	Variable ou constante entière qui précise le numéro de la ressource.
IAC	Variable ou constante entière qui précise le nombre d'accès à la ressource.
I P H	Variable ou constante entière représentant la phase d'avancement d'une tâche.
INAME	Nom de branche (6 caractères ASCII)
IDATIM	Tableau entier de 7 mots chargé par la date et l'heure.
INV	Variable ou constante entière précisant un nombre de mesures.
JADR	Tableau entier d'adresses physiques
KTAB	Tableau entier de mesures-
MTAB	Tableau de masque de sorties digitales
MTIME	Variable ou constante entière précisant la durée d'une sortie digitale en nombre de tops.

9 - Syntaxe Fortran des fonctions Temps Réel du CMF

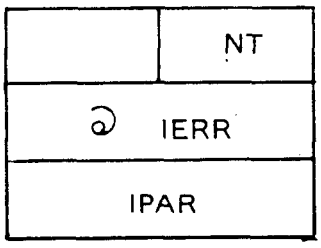
Free BASIC	CALL FBASIC (IERR)
Define CLOCK	CALL DCLOCK (NREV, IEV, IERR, NFU)
Free CLOCK	CALL FCLOCK (NREV, IERR, NFU)
Initialize CLOCK	CALL CLOCK (NREV, IH, IM, IS, IMS, IERR, NFU)
	CALL CLOCK (NREV, NTOP, IERR, NFU)
Define TIMER	CALL DTIMER (NTIMER, IEV, IERR, NFU)
Free TIMER	CALL FTIMER (NTIMER, IERR, NFU)
Initialize TIMER	CALL TIMER (NTIMER, IPER, NP, IREP, IERR, NFU)
	CALL TIMER (IBUF, NOMBRE, IERR, NFU)
Rearm or stop TIMER	CALL RTIMER (NTIMER, IA, IERR NFU)
Define ETOR	CALL DETOR (NETOR, IEV, IERR, NFU)
Free ETOR	CALL FETOR (NETOR, IERR, NFU)
Set (with delay) RELAY or reset	CALL RELAY (ID, IERR, NFU)
re-arm RELAY	CALL RELAY (IERR, NFU)

Liste des paramètres spécifiques aux fonctions Temps Réel du CMF :

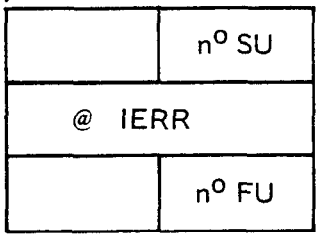
NFU	Variable ou constante entière représentant le numéro de la FU Temps Réel
NREV	Variable ou constante entière représentant un numéro de réveil
NTIMER	Variable ou constante entière représentant un numéro de temporisation
NETOR	Variable ou constante entière représentant un numéro d'entrée de surveillance Tout ou Rien
IBUF	Tableau entier de NOMBRE*2 mots contenant les paramètres d'initialisation des temporisations
IH IM IS IMS	Variables ou constantes entières représentant une heure d'échéance absolue exprimée en heures, minutes, secondes et millisecondes
NTOP	Variable ou constante entière représentant un délai exprimé en nombre de tops de 20 ms
IPER	Variable ou constante entière qui précise une période de base pour exprimer la durée d'une temporisation 0 pour 10 ms | 4 pour 150 ms | 8 pour 500 ms | 12 pour 1 s 1 pour 20 ms | 5 pour 200 ms | 9 pour 600 ms | 13 pour 1,5 s 2 pour 50 ms | 6 pour 250 ms | 10 pour 700 ms | 14 pour 2 s 3 pour 100 ms | 7 pour 400 ms | 11 pour 800 ms | 15 pour 2,5 s
NP	Variable ou constante entière représentant la durée d'une temporisation, exprimée en nombre de périodes de base, précisées par IPER
IREP	Variable ou constante entière représentant la répétitivité de la temporisation. IREP = 0, répétitivité infinie
IA	Variable ou constante entière précisant l'arrêt (IA = 0) ou le réarmement (IA = 1) d'une temporisation

10 - Description des RPB des requêtes

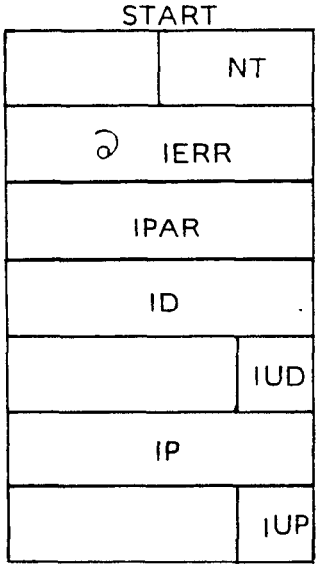
RUN
OFF



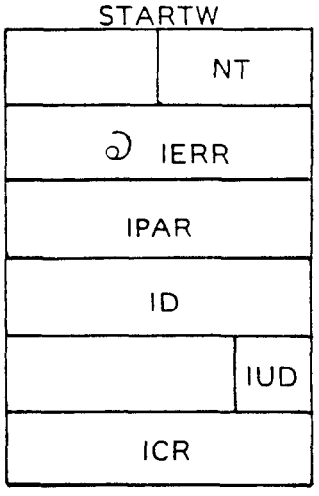
SUFU



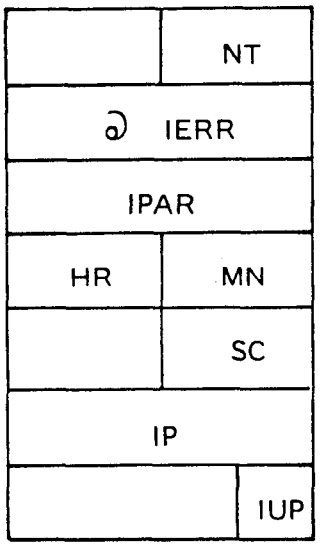
START



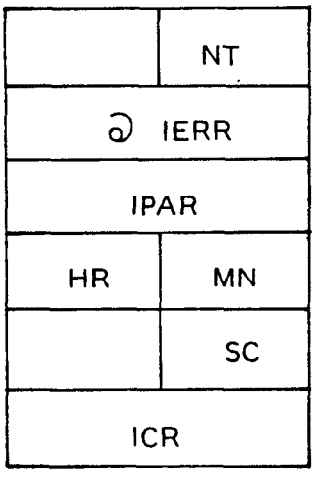
STARTW



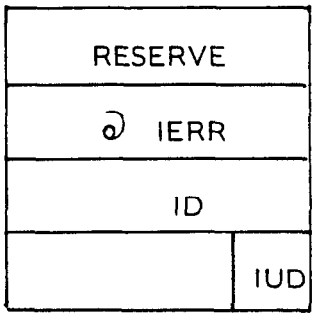
TRNON



TRNONW



WAIT



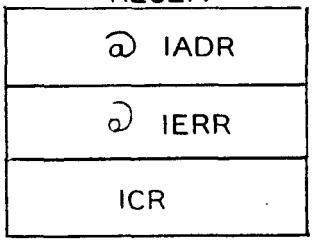
DATAG

pas de table de paramètres

EXIT

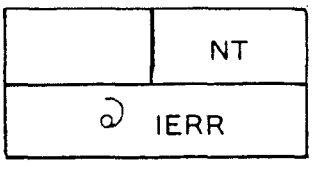
pas de table de paramètres

SEND
RECEIV

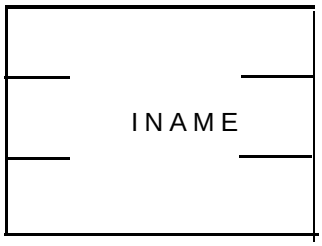


KILL

INHIB



BCHLR



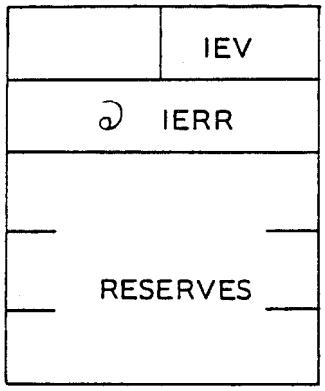
BACK
BACKR

Pas de table de paramètres

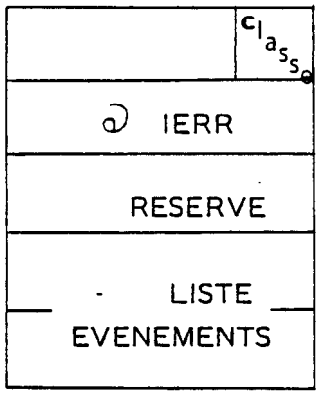
BACKR
FREEM
RMDEF

Pas de table de paramètres

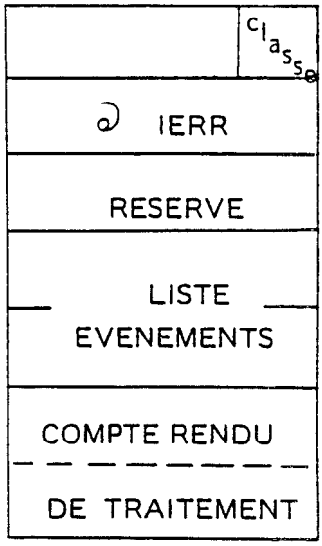
WEVENT



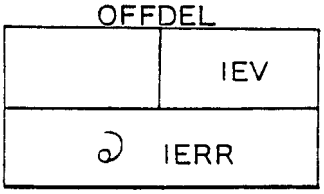
WEVAND



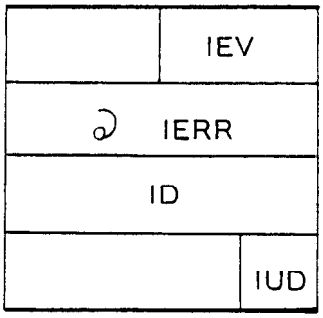
WEVOR



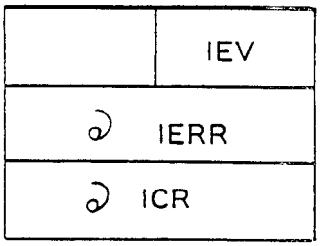
SEVENT
REVENT
OFFDEL



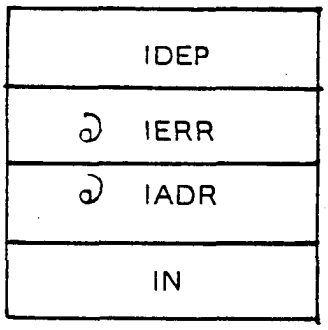
SEVDEL



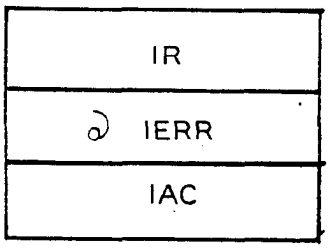
TEVENT



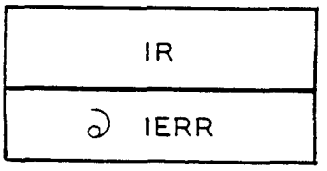
REDPUT
REDGET



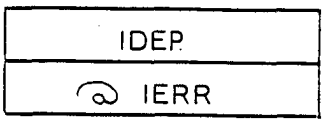
RESDEF



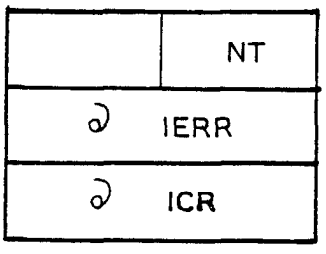
RRQST
RRLSE



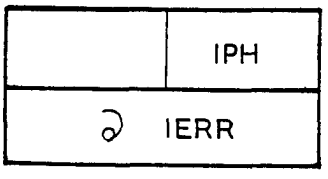
RACT
RWAIT



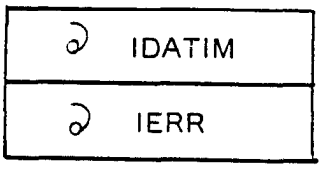
STAGET



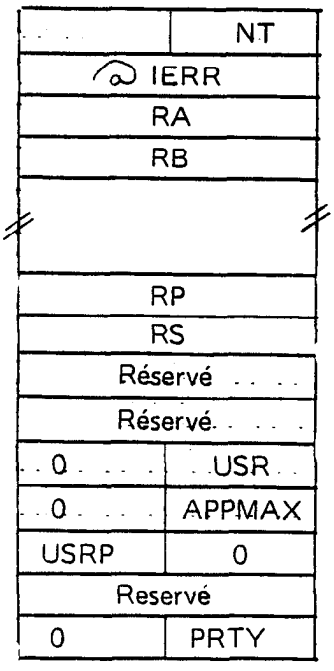
STADEF



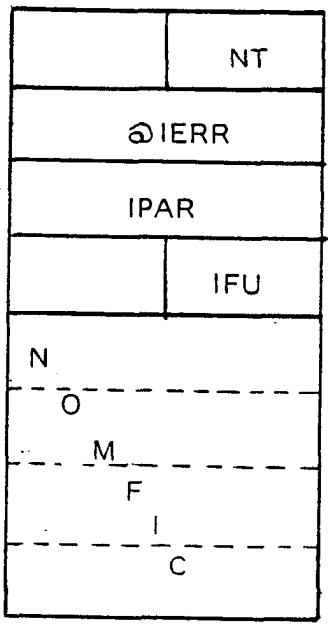
WTIME
TIME



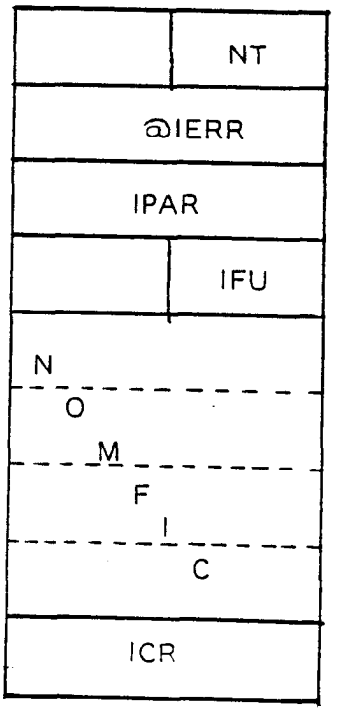
TASKC



TCALL



TCALLW



11 - Compte rendu des requêtes de RTES16 :

REQUETES ADRESSEES A IOCS

→ Se reporter au manuel de référence de IOCS.

REQUETES ADRESSEES A FMS

→ Se reporter au manuel de référence de FMS

Les erreurs logiques de numéro supérieur ou égal à 32 sont considérées comme fatales pour les tâches de l'application dont le numéro d'usager USR de FMS est inférieur à 128 ; le traitement réalisé par le système pour ces erreurs fatales est identique au traitement des erreurs des requêtes de type temps réel.

REQUETES DE TYPE TEMPS REEL

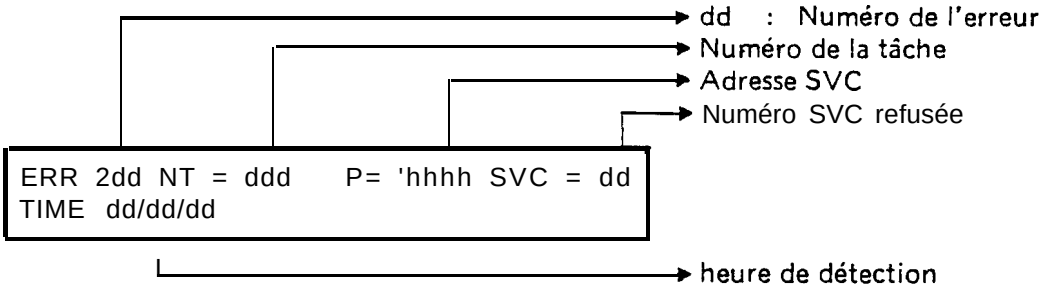
La plage des compte rendus est la même pour toutes ces requêtes ; pour une interprétation plus précise, se reporter au chapitre "Requêtes Programmées" de ce manuel.

1	Requête acceptée - Traitement réalisé	
2	L'entité spécifiée n'existe pas	erreurs fatales.
3	La requête est illogique ou redondante	
4	L'appelant n'a pas accès à la requête	
5	La requête est incompatible avec l'état actuel de l'entité	
6	Il existe au moins un paramètre incorrect	
7	Système sous-dimensionné	
8	Requête non traitée par RTES16 ou valeur incorrecte du registre K (mode esclave)	

TRAITEMENT DES ERREURS

→ La tâche en erreur est suspendue (sauf s'il s'agit d'une tâche de type hardware)

→ Un message est imprimé sur le périphérique associé à l'unité symbolique TE (sauf s'il s'agit d'une tâche de type hardware)



→ La tâche en erreur pourra être relancée par l'opérateur (commande RSUM)

Les tâches de type hardware d'une application sont supposées ne jamais émettre de requêtes erronées.

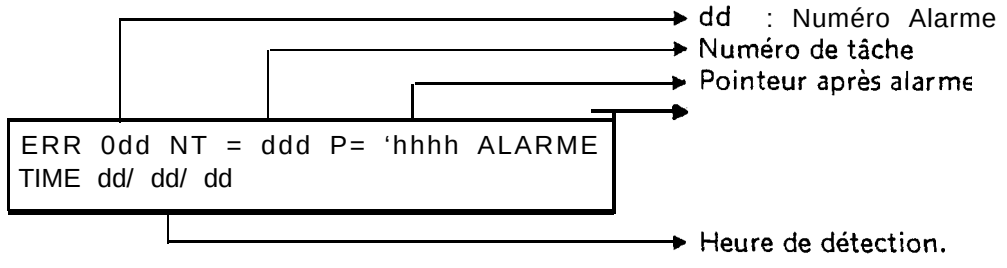
Exemple de message d'erreur :

ERR 240 NT = 25 P = '0256 SVC = 56
erreur '60 28 de FMS (erreur de syntaxe dans le FCB)

TRAITEMENT DES ALARMES DE PROGRAMMATION

→ La tâche fautive est suspendue (sauf s'il s'agit d'une tâche de type hardware)

→ Un message est imprimé sur le périphérique associé à l'unité symbolique TE. (sauf s'il s'agit d'une tâche de type hardware)



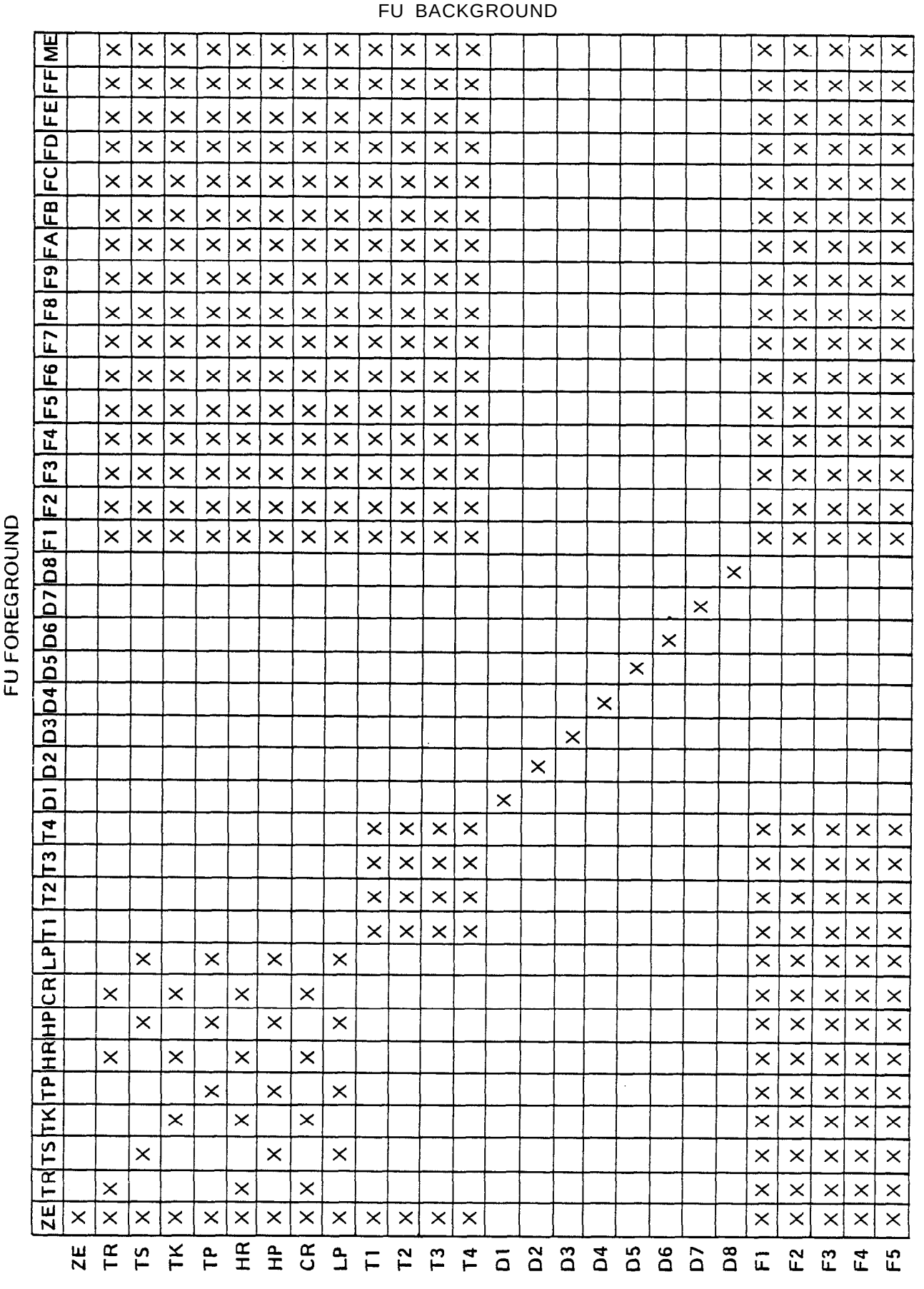
Les tâches de type hardware d'une application sont supposées ne jamais provoquer d'alarmes de programmation.

SU FU	Affecter une unité fonctionnelle à une unité symbolique
IRUN, NT [,PAR]	Activer une tâche immédiatement
TACT, NT [,PAR]	Activer une tâche immédiatement ou jamais
DRUN,NT,[PAR],{ID/IUD}[,IP/IUP]	Activer une tâche après un délai initial et/ou périodiquement
TRUN,NT,[PAR],HR/MN/SC[,IP/IUP]	Activer une tâche à une heure donnée [et périodiquement]
TOFF , NT [,PAR]	Supprimer les activations différées et périodiques d'une tâche
STOP, NT	Inhiber une tâche
VALI, NT	Relancer une tâche inhibée
RSUM, NT	Relancer une tâche interrompue sur erreur
SAVE, [P1] , SI [, P2 , P3]	Sauvegarder une application sur disque
INIT [, [SI] [, FU]]	Lancer un système ou une application sauvegardé sur disque
TASK,{R[P]/N[P]},FICNAM-PW,P1[,P2,P3,P4]	Intégrer une tâche ou le moniteur background
EXEC,R,FICNAM-PW,P1	Intégrer un processeur
KILL[,{NT/FICNAM-PW}]	Tuer une tâche, un processeur ou le background
TIME[,J/M/A, HR/MN/SC]	Imprimer ou initialiser la date et l'heure
PART, NT, P1 [,P2,P3,P4]	Modifier l'affectation de partition (s) pour une tâche

PRTY, NT, PY [,T]	Modifier la priorité d'une tâche
SYST, N	Connaître l'état du système
STAT, NT	Connaître l'état d'une tâche
DEST, NPU	Connaître l'état d'un périphérique
PRIN, ADRDEB [, NBMØT] [, NPAG]	Visualiser une zone de mémoire centrale
DUMP, ADRDEB, ADRFIN [, NPAG]	Visualiser une zone de mémoire centrale
KEND [,MOPASS]	Terminer le dialogue opérateur
INEX, NR	Interdire l'accès à une requête utilisateur
PAGE, NP	Effectuer des sauts de page sur imprimante
ROCK, NT	Passer au mode accéléré de chargement d'une tâche non résidente
SLOW, NT	Passer au mode standard de chargement d'une tâche non résidente
POFF	Mettre hors tension le perforateur rapide et certains modèles de lecteurs rapides
BACK, PI, [REQUET], [N]	Activer le moniteur background
GIVE, B1 [,B2, B3, B4, B5]	Autoriser au background l'accès à une ou plusieurs FU
TAKE, B1 [,B2, B3, B4, B5]	Supprimer l'autorisation d'accès à une ou plusieurs FU par le background
MEMO, ADR, MASK [[[,{AND / OR / EOR}],[N],[NPAG]]]	Modifier le contenu d'une zone de mémoire centrale

CLOSE SU	Fermeture fichier/SU
DMON, {SU / FU}	Démontage disque
MONT, {SU / FU} [, label]	Montage disque
SUSP, U_i, E_i [, FU]	Affectation SU - espace
CONT, {N / Y} [, P1 [, P2]]	Suppression/Rétablissement des contrôles sur erreurs sur requêtes
CFMS, {I / E}	Édition de comptages d'accès FMS
DBCT, NT	Édition de la PST d'une tâche
PAUSE <texte>	Suspension fichier de commandes
RETURN	Reprise fichier de commandes
TACD, NT	Dump d'une tâche
PARD, NP	Dump d'une partition
VIFI	Liste des fichiers ouverts
COMP, label [± déplact / ± label]	Calculs d'adresses
VITU, FU, [ft], [P / N], [dep], [nb]	Visualisation de TUP
MAP [, FU]	Liste FU bootstrap
DSYS, NS	Destruction système dans D1

13 - TABLEAU DES AFFECTATIONS POSSIBLES DES FU “FOREGROUND” AUX FU “BACKGROUND”



14 - PUBLICATION DE L' I.S.A.

DRAFT

"INDUSTRIAL COMPUTER SYSTEM FORTRAN PROCEDURES
FOR EXECUTIVE FUNCTIONS AND PROCESS INPUT-OUTPUT".

Table of contents :

SPONSOR

INSTRUMENT SOCIETY OF AMERICA
400 STANWIX STREET
PITTSBURGH. PA, 15222

PHONE : (412) 281-3171

1 - SCOPE

This standard presents procedure subprogram references for use by industrial computer Control system implementers. These procedure subprogram references permit interfacing with Executive Systems, Process Input and Output functions. and allows manipulation of bit Strings. These are of the form defined by the ANSI X3.9-1966 Standard FORTRAN. No changes in Standard FORTRAN syntax are intended.

2 - EXECUTIVE INTERFACES

2.1 - Control of Program Execution

Executive interfaces allow the implementer the facility to control operations of the program with in the System. Given these subroutine subprogram facilities an implementer may start. stop, or delay execution of any application program within the system. The argument m, shown below, shall be equal to or greater than (2) for all instances in which the request is not accepted by the executive system. Individual implementations may specify unique values of m within the allowable range to designate the specific reason for which the request was rejected.

2.2 - Starting a Program Immediately or After a specified Time Delay

The START call shall, after the expiration of the specified time delay, cause the execution of the designated program. The actual time delay obtainable in a specific industrial computer, object machine, is subject to the resolution of that object machine's real time clock. Execution of the designated program will commence at the program's first executable statement. The form of the call is :

```
CALL START (i, j, k, m)
```

Where :

i specifies the program to be executed

j specifies the length of time, in units as specified by k, to delay before executing the program. If the value is zero or negative the requested program will run as soon as permissable.

k specifies units of time as follows :

0 - Basic counts of the system's clock

1 - Milliseconds

2 - Seconds
3 - Minutes

m is set on return to the calling program to indicate the disposition of the request as follows

0 or less	- Undefined
1	- Request accepted
2 or greater	- Request not accepted

2.3 - Starting a Program at a specified Time

The TRNON call shall cause the designated program to be executed at a specified time of day. Execution of the designated program will commence at the program's first axecutable statement. The form of the call is :

```
CALL TRNON (i, j, m)
```

Where :

specifies the program to be executed.

designates a one-dimensional integer array of length three : the array contains the absolute time the program is to be executed. The elements of the array are as follows :

(1) Hours using a 24-hour clock
(2) - Minutes
(3) - Seconds

is set on return to the calling program to indicate the disposition of the request as follows :

0 or less	- Undefined
1	- Request accepted
2 or greater	- Request not accepted

- Delaying Continuation of a Program

The WAIT call shall provide a means whereby a program in a multiprogramming environment may voluntarily relinquish control of the system for a specified lengh of time. After expiration of the specified delay, the executive is required to resume execution of the requesting program at the first executable statement following the WAIT call. It is further required that the requesting program's resources are returned intact when the execution is resumed.
The form of the call is :

```
CALL WAIT (i, k, m)
```

Where :

i specifies the length of time in units as specified by k, to delay before continuing execution in the requesting program. If the value is zero or nëgative no delay will occur.

k specifies units of time as follows :

0 - Basic counts of the system's clock
1 - Milliseconds
2 - Seconds
3 - Minutes

m is set on return to the calling program to indicate the disposition of the request as follows :

0 or less - Undefined
1 - Request accepted
2 or greater - Request not accepted.

2.5 - Program Termination Accomplished Using Present Standard.

Termination of programs shall be accomplished using the STOP statement from ANSI X3,9-1966 Standard FORTRAN. Operation of the system shall continue performing other task in the multi-programming system.

3 - PROCESS INPUT AND OUTPUT FUNCTION INTERFACES

3.1 - Control of Analog and Digital Sensors and Outputs

Process input and output function interfaces allow the implementer to access specified analog and digital sensors and output. Two forms are required. The first form accommodates executive systems which permit continuation of execution in the requesting program while the process input or output is being accomplished ; the other form accommodates executive systems which suspend execution in the requesting program until the process input or output is completed. The latter case is flagged by the addition of a "W" as the last letter in the subroutine subprogram name.
The argument m, shown below, is to be interrogated by the requesting program in order to determine the status of the request, When a "W" is added, as noted above, m will be set on return to the requesting program ; if "W" is absent the requesting program is required to have provision for periodically testing the state of the request, Individual implementations may specify unique values of m within the allowable range to designate the specific error condition.

3.2 - Analog Inputs in a Sequential Order

These subroutine subprograms are for the inputting of any number of sequential analog points. The forms of the calls are :

CALL AISQ (i. j, k, m) and CALL AISQW (i, j, k, m)

Where :

i specifies the number of analog points to be input

j specifies terminal connection data of the first analog point to be input. Specific information relevant to construction of this argument must be provided by the vendor since various types of analog to digital convertors are available

k designates a one-dimensional array into which the converted values are stored

m indicates the disposition of the request as follows :

0 or less	- Undefined
1	- All data collected
2	- Operation incomplete
3 or greater	- Error condition.

3.3 - Analog Inputs in a Random Order

These subroutine subprograms are for the inputting of analog points in a random order. Array j contro the order and allows the specification of control information on an individual point basis. The forms of the calls are :

CALL AIRD (i, j, k m) and CALL AIRDW (i, j, k, m)

Where

i specifies the number of analog points to be input

j specifies terminal connection data for each analog to be input. Specific information rele: to construction of this argument must be provided by the vendor.

k designates a one-dimensional array into which the converted values are stored

m indicates the disposition of the request as follows :

0 or leess	- Undefined
1	- all date collected
2	- Operation incomplete
3 or greater	- Error condition.

3.4 - Analog Output

These subroutine subprograms are for the outputting of high and low speed analog outputs in a random order. The forms of these calls are :

CALL AO (i, j, k, m) and CALL AOW (i, j, k, m)

Where :

i specifies the number of analog points to output

j specifies terminal connection data of the analog output points. Specific information relevant to construction of this argument must be provided bu the vendor since various types of digital to analog converters are available.

k designates a one-dimensional array from which the analog output values are taken.

m indicates the disposition of the request as follows :

0 or less	-	Undefined
1	-	All data outputted
2	-	Operation incomplete
3 or greater	-	Error condition

3.5 - Digital Input

These subroutine subprograms are for the inputting of digital input strings. The length of the string is related to the vendor's process hardware but shall not exceed the number of bits in the object machine word. The forms of these calls are :

CALL DI (i, j, k, m) and CALL DIW (i, j, k. m)

Where :

i specifies the number of digital words to be input

j specifies terminal connection data for each digital input Word. Specific information releva to construction of this argument must be provide by the vendor since various types of digital inputs are available. The digital input words Will be input in the order that they are listed in j

k designates a one-dimensional integer array into which the requested digital input values are stored. For every closed contact input or logical input, "one" will correspond to a corresponding bit set in the image input.

m indicates the disposition of the request as follows :

0 or less - Undefined
1 - All data collected
2 - Operation incomplete
3 or greater - Error condition

3.6 - Momentary Digital Output

These subroutine subprograms are for the outputting of high and low speed digital outputs requiring a momentary signal. The specific digital outputs are of the type that take up to a full word of bits and output it using an operation causing individual outputs to be set when a corresponding bit in the output word is set. The forms of the call are :

CALL DOM (i, j, k, n, m) and CALL DOMW (i, j, k, n, m)

Where :

i specifies the number of digital output words

j specifies terminal connection data for each word of output, information relevant to the construction of this argument must be provided by the vendor.

k specifies a one-dimensional integer array whose contents are the image words to be output

n specifies the duration measured in basic system clock counts that the outputs are to rema set. If the computer system on which the subroutine is run does not allow selection of du tion, this argument is ignored.

m indicates the disposition of the request as follows :
0 or less - Undefined
1 - All output accomplished
2 - Operation incomplete
3 or greater - Error condition.

3.7 - Latching Digital Output

These subroutine subprograms are the outputting of digital outputs which can be latched in either the set or reset state. The specific digital outputs are of the type that up to a full word of bits and outputs it using an operation causing individual digital outputs to be set when a corresponding bit in the output word is set or causing individual digital outputs to be reset when a corresponding bit in the output word is reset. The forms of the calls are :

CALL DOL (i. j, k_1, k_2, m) and CALL DOLW (i, j, k_1, k_2, m)

Where :

i specifies the number of words of digital output

j designates terminal connection data for each word of digital output. Specific information relevant to the construction of this argument must the provided by the vendor.

k_1 specifies a one-dimensional integer array whose, contents are the image words to be output.

k_2 designates a one-dimensional integer array whose contents define digital outputs which can be changed by the subroutine. A bit set in the k_2 array indicates that the digital output will be changed to state defined by the corresponding bit position in corresponding word in k_1

m indicates the disposition of the request as follows :

0 or less	-	Undefined
1	-	All output accomplished
2	-	Operation incomplete
3 or greater	-	Error condition

4 BIT STRING MANIPULATIONS

4.1 - Types of Manipulations

Two types of bit string manipulations are specified, logical and shifting. These are selected to support the previously specified calls. In order to process words from arrays it is necessary to be able to manipulate information on a bit by bit basis.

4.2 - Logical Operations

These operations are implemented as function subprograms. In the following functions, m and n specify integer variables or array elements. Operations are performed on a full word, bit by bit.

4.2.1 - Inclusive Or

IOR (m, n)

Where m and n designate arguments that are logically added.

4.2.2 - Logical Product

IAND (m, n)

Where m and n designate arguments that are logically multiplied

4.2.3 - Logical Complement

NOT (m)

Where m designates the argument that is logically complemented

4.2.4 - Exclusive Or

IDEOR (m, n)

Where m and n designate arguments that are exclusively added.

4.3 - Shift Operations

The logical shift is implemented as a function subprogram.
A right or left shift can be specified Zeroes are propagated following the shifted value and the arguments sign is not extended.

ISHFT (m. n)

m designates the argument to be shifted

n specifies the number of positions to be shifted and the direction of the shift thus :
n > 0 shift right
n < 0 shift left
n = 0 noshift

The absolue value of n, n, should not exceed the number of bits in the object machine word. If it does the value of the function will be zero.

5 - GLOSSARY

Bit String - A string of binary digits in which each bit position is considered as an independent unit.

Executable statement - Constituant of a program specifying the action of the program, contrasted with a nonexecutable statement which describes the use of the program. the characteristics of the operands, editing information, statement functions, or data arrangement.

Execute - To interpret machine instructions or higher level statements and perform the indicated operations on the operands specified.

Executive System - An integrated collection of service routines for supervising the sequencing of programs by a computer.

Multiprogramming - Pertaining to the concurrent execution of two or more programs by a single computer.

Object Machine - The computer on which the object program is to be executed.

Program - A series of actions proposed in order to achieve a certain result. Also a unit of work for the central processing unit from the standpoint of the executive program.

Real Time Clock - A clock that indicates the passage of actual time, in contrast to a fictitious time set up by a computer program ; such as, the elapsed time in the flight of a missile.

Resource - Any facility of the computing system or operating system required by a job or task. and including main storage, input/output devices, the central processing unit, data sets, and control processing programs.

Réseaux
et systèmes
d'information

Bull

Bull Sems

VOS REMARQUES SUR CE DOCUMENT

TITRE
RTES16 ADDENDUM A

N° DE REFERENCE	DATE
Bull-Sems : 1 164 324 00 036 04	JANVIER 1986

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
38 130 - ECHIROLLES