cii division petits ordinateurs et applications systèmes

mitra 15

COMPAGNIE INTERNATIONALE POUR L'INFORMATIQUE

Moniteur temps réel MTR

manuel d'utilisation

Gamme	:	MITRA 15
Systèmes	:	MTR - MTR.E
Objet	:	Ce manuel décrit le moniteur temps réel MTR disponible sur l'ordinateur MITRA 15 (configuration minimale 8 K mots mémoire et une téléscriptrice). On y trouve la description des commandes, l'organisation des E/S, le traitement des interruptions et déroutements et l'utilisation des modules moniteur.
Remarques	:	Version 5
Nombre de pages	:	111
Date d'édition	:	Février 1975

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

Compagnie Internationale pour l'Informatique
CIDOC 68, route de Versailles 78430 LOUVECIENNES

Référence documen
4111 U4/

I-1
II-1
II-2
III-1
III-2
III-3
III-4
III-5
III-6
III-7
III-8
III-9
III-10
IV-1
IV-2
IV-3
IV-4
IV-5
IV-6
V-1
V-2
V-3
V-4
V-5
V-6
V-7
V-8
V-9
V-10
V-11
V-12
V-13
V-14
V-15
V-16
V-17
V-18
V-19
V-20
V-21
V-22
V-23
V-24
V-25
V-26
V-27
V-28
V-29
V-30
V-31
V-32
V-33
V-34
V-35
V-36
V-37
V-38
V-39
V-40
V-41
V-42
V-43
V-44
V-45
V-46
V-47
V-48
V-49
V-50
V-51
V-52
V-53
V-54
V-55
V-56
V-57
V-58
V-59
V-60
V-61
V-62
V-63
V-64
V-65
V-66
V-67
V-68
V-69
V-70
V-71
V-72
V-73
V-74
V-75
V-76
V-77
V-78
V-79
V-80
V-81
V-82
V-83
V-84
V-85
V-86
V-87
V-88
V-89
V-90
V-91
V-92
V-93
V-94
V-95
V-96
V-97
V-98
V-99
V-100
V-101
V-102
V-103
V-104
V-105
V-106
V-107
V-108
V-109
V-110
V-111
V-112
V-113
V-114
V-115
V-116
V-117
V-118
V-119
V-120
V-121
V-122
V-123
V-124
V-125
V-126
V-127
V-128
V-129
V-130
V-131
V-132
V-133
V-134
V-135
V-136
V-137
V-138
V-139
V-140
V-141
V-142
V-143
V-144
V-145
V-146
V-147
V-148
V-149
V-150
V-151
V-152
V-153
V-154
V-155
V-156
V-157
V-158
V-159
V-160
V-161
V-162
V-163
V-164
V-165
V-166
V-167
V-168
V-169
V-170
V-171
V-172
V-173
V-174
V-175
V-176
V-177
V-178
V-179
V-180
V-181
V-182
V-183
V-184
V-185
V-186
V-187
V-188
V-189
V-190
V-191
V-192
V-193
V-194
V-195
V-196
V-197
V-198
V-199
V-200
V-201
V-202
V-203
V-204
V-205
V-206
V-207
V-208
V-209
V-210
V-211
V-212
V-213
V-214
V-215
V-216
V-217
V-218
V-219
V-220
V-221
V-222
V-223
V-224
V-225
V-226
V-227
V-228
V-229
V-230
V-231
V-232
V-233
V-234
V-235
V-236
V-237
V-238
V-239
V-240
V-241
V-242
V-243
V-244
V-245
V-246
V-247
V-248
V-249
V-250
V-251
V-252
V-253
V-254
V-255
V-256
V-257
V-258
V-259
V-260
V-261
V-262
V-263
V-264
V-265
V-266
V-267
V-268
V-269
V-270
V-271
V-272
V-273
V-274
V-275
V-276
V-277
V-278
V-279
V-280
V-281
V-282
V-283
V-284
V-285
V-286
V-287
V-288
V-289
V-290
V-291
V-292
V-293
V-294
V-295
V-296
V-297
V-298
V-299
V-300
V-301
V-302
V-303
V-304
V-305
V-306
V-307
V-308
V-309
V-310
V-311
V-312
V-313
V-314
V-315
V-316
V-317
V-318
V-319
V-320
V-321
V-322
V-323
V-324
V-325
V-326
V-327
V-328
V-329
V-330
V-331
V-332
V-333
V-334
V-335
V-336
V-337
V-338
V-339
V-340
V-341
V-342
V-343
V-344
V-345
V-346
V-347
V-348
V-349
V-350
V-351
V-352
V-353
V-354
V-355
V-356
V-357
V-358
V-359
V-360
V-361
V-362
V-363
V-364
V-365
V-366
V-367
V-368
V-369
V-370
V-371
V-372
V-373
V-374
V-375
V-376
V-377
V-378
V-379
V-380
V-381
V-382
V-383
V-384
V-385
V-386
V-387
V-388
V-389
V-390
V-391
V-392
V-393
V-394
V-395
V-396
V-397
V-398
V-399
V-400
V-401
V-402
V-403
V-404
V-405
V-406
V-407
V-408
V-409
V-410
V-411
V-412
V-413
V-414
V-415
V-416
V-417
V-418
V-419
V-420
V-421
V-422
V-423
V-424
V-425
V-426
V-427
V-428
V-429
V-430
V-431
V-432
V-433
V-434
V-435
V-436
V-437
V-438
V-439
V-440
V-441
V-442
V-443
V-444
V-445
V-446
V-447
V-448
V-449
V-450
V-451
V-452
V-453
V-454
V-455
V-456
V-457
V-458
V-459
V-460
V-461
V-462
V-463
V-464
V-465
V-466
V-467
V-468
V-469
V-470
V-471
V-472
V-473
V-474
V-475
V-476
V-477
V-478
V-479
V-480
V-481
V-482
V-483
V-484
V-485
V-486
V-487
V-488
V-489
V-490
V-491
V-492
V-493
V-494
V-495
V-496
V-497
V-498
V-499
V-500
V-501
V-502
V-503
V-504
V-505
V-506
V-507
V-508
V-509
V-510
V-511
V-512
V-513
V-514
V-515
V-516
V-517
V-518
V-519
V-520
V-521
V-522
V-523
V-524
V-525
V-526
V-527
V-528
V-529
V-530
V-531
V-532
V-533
V-534
V-535
V-536
V-537
V-538
V-539
V-540
V-541
V-542
V-543
V-544
V-545
V-546
V-547
V-548
V-549
V-550
V-551
V-552
V-553
V-554
V-555
V-556
V-557
V-558
V-559
V-560
V-561
V-562
V-563
V-564
V-565
V-566
V-567
V-568
V-569
V-570
V-571
V-572
V-573
V-574
V-575
V-576
V-577
V-578
V-579
V-580
V-581
V-582
V-583
V-584
V-585
V-586
V-587
V-588
V-589
V-590
V-591
V-592
V-593
V-594
V-595
V-596
V-597
V-598
V-599
V-600
V-601
V-602
V-603
V-604
V-605
V-606
V-607
V-608
V-609
V-610
V-611
V-612
V-613
V-614
V-615
V-616
V-617
V-618
V-619
V-620
V-621
V-622
V-623
V-624
V-625
V-626
V-627
V-628
V-629
V-630
V-631
V-632
V-633
V-634
V-635
V-636
V-637
V-638
V-639
V-640
V-641
V-642
V-643
V-644
V-645
V-646
V-647
V-648
V-649
V-650
V-651
V-652
V-653
V-654
V-655
V-656
V-657
V-658
V-659
V-660
V-661
V-662
V-663
V-664
V-665
V-666
V-667
V-668
V-669
V-670
V-671
V-672
V-673
V-674
V-675
V-676
V-677
V-678
V-679
V-680
V-681
V-682
V-683
V-684
V-685
V-686
V-687
V-688
V-689
V-690
V-691
V-692
V-693
V-694
V-695
V-696
V-697
V-698
V-699
V-700
V-701
V-702
V-703
V-704
V-705
V-706
V-707
V-708
V-709
V-710
V-711
V-712
V-713
V-714
V-715
V-716
V-717
V-718
V-719
V-720
V-721
V-722
V-723
V-724
V-725
V-726
V-727
V-728
V-729
V-730
V-731
V-732
V-733
V-734
V-735
V-736
V-737
V-738
V-739
V-740
V-741
V-742
V-743
V-744
V-745
V-746
V-747
V-748
V-749
V-750
V-751
V-752
V-753
V-754
V-755
V-756
V-757
V-758
V-759
V-760
V-761
V-762
V-763
V-764
V-765
V-766
V-767
V-768
V-769
V-770
V-771
V-772
V-773
V-774
V-775
V-776
V-777
V-778
V-779
V-780
V-781
V-782
V-783
V-784
V-785
V-786
V-787
V-788
V-789
V-790
V-791
V-792
V-793
V-794
V-795
V-796
V-797
V-798
V-799
V-800
V-801
V-802
V-803
V-804
V-805
V-806
V-807
V-808
V-809
V-810
V-811
V-812
V-813
V-814
V-815
V-816
V-817
V-818
V-819
V-820
V-821
V-822
V-823
V-824
V-825
V-826
V-827
V-828
V-829
V-830
V-831
V-832
V-833
V-834
V-835
V-836
V-837
V-838
V-839
V-840
V-841
V-842
V-843
V-844
V-845
V-846
V-847
V-848
V-849
V-850
V-851
V-852
V-853
V-854
V-855
V-856
V-857
V-858
V-859
V-860
V-861
V-862
V-863
V-864
V-865
V-866
V-867
V-868
V-869
V-870
V-871
V-872
V-873
V-874
V-875
V-876
V-877
V-878
V-879
V-880
V-881
V-882
V-883
V-884
V-885
V-886
V-887
V-888
V-889
V-890
V-891
V-892
V-893
V-894
V-895
V-896
V-897
V-898
V-899
V-900
V-901
V-902
V-903
V-904
V-905
V-906
V-907
V-908
V-909
V-910
V-911
V-912
V-913
V-914
V-915
V-916
V-917
V-918
V-919
V-920
V-921
V-922
V-923
V-924
V-925
V-926
V-927
V-928
V-929
V-930
V-931
V-932
V-933
V-934
V-935
V-936
V-937
V-938
V-939
V-940
V-941
V-942
V-943
V-944
V-945
V-946
V-947
V-948
V-949
V-950
V-951
V-952
V-953
V-954
V-955
V-956
V-957
V-958
V-959
V-960
V-961
V-962
V-963
V-964
V-965
V-966
V-967
V-968
V-969
V-970
V-971
V-972
V-973
V-974
V-975
V-976
V-977
V-978
V-979
V-980
V-981
V-982
V-983
V-984
V-985
V-986
V-987
V-988
V-989
V-990
V-991
V-992
V-993
V-994
V-995
V-996
V-997
V-998
V-999
V-1000

mitra 15

compagnie internationale
pour l'informatique

manuel d'utilisation

Moniteur Temps Réel MTR

SOMMAIRE

INTRODUCTION	I-1
STRUCTURE DU SYSTEME - ORGANISATION DE LA MEMOIRE	II-1
Structure du noyau résident	II-1
Organisation de la mémoire	II-1
TRAITEMENT DES ENTREES/SORTIES ET DES DELAIS	III-1
Interface d'Entrée/Sortie	III-1
Module de contrôle d'Entrée/Sortie : le handler	III-2
Appel d'Entrée/Sortie : CSV M:IO	III-3
Attente fin de transfert : CSV M:WAIT	III-4
Etiquettes opérationnelles	III-4
Schéma d'organisation d'une Entrée/Sortie sous MTR	III-5
Traitement des périphériques non prêts	III-7
Traitement des time-out	III-8
Interruption externe en fin de transfert	III-8
Traitement des délais	III-8
INTERRUPTIONS	IV-1
L'interruption pupitre (ou opérateur)	IV-1
L'interruption horloge	IV-1
Les interruptions de coupure et retour secteur	IV-2
Les interruptions des coupleurs (traitées par handler)	IV-2
Autres interruptions	IV-2
Fin d'une tâche	IV-2
DEROUTEMENTS	V-1
Différents types de déroutement	V-1
Simulation des instructions optionnelles	V-1
M:TRAP type 0	V-2
Les autres types de module M:TRAP	V-3

4111 U4/FR
111 pages

SOMMAIRE (suite)

COMMUNICATIONS AVEC L'OPERATEUR - M:OC	VI-1
Commandes opérateur	VI-1
%LOAD commande de chargement	VI-3
%RUN commande de lancement	VI-5
%ABORT commande d'abandon d'un programme	VI-7
%FOREGROUND déplacement de la limite du foreground	VI-8
%DISPLAY édition des éléments foreground et système	VI-9
%DUMP commande d'édition mémoire	VI-9
%ASSIGN commande d'assignation	VI-11
%TIME initialisation de la date	VI-13
%X abandon du background et dump post-mortem	VI-13
%Y abandon du background sans dump post-mortem	VI-14
%MODIFY commande de modification mémoire	VI-14
%DEVICE action particulière sur périphérique	VI-15
%IT armement et/ou validation d'interruption	VI-16
%ACTIVATE excitation d'une interruption	VI-17
%PM protection ou libération de zone mémoire	VI-18
Messages opérateur	VI-19
UTILISATION DES MODULES DU MONITEUR	VII-1
Principes généraux des appels moniteur	VII-1
M:IO appel d'Entrée/Sortie	VII-4
M:WAIT attente d'activation d'évènement	VII-10
M:EXIT fin de programme	VII-11
M:KEY clés et voyants	VII-11
M:DCBN conversion décimal-binaire	VII-12
M:HXBN conversion hexadécimal-binaire	VII-12
M:BNDC conversion binaire-décimal	VII-13
M:BNHX conversion binaire-hexadécimal	VII-13
M:ASEB conversion ASCII-EBCDIC	VII-14
M:EBAS conversion EBCDIC-ASCII	VII-17
M:LOAD chargement d'un module IMT en mémoire	VII-18
M:MOVE déplacement d'une chaîne d'octets	VII-21
M:EDIT édition d'une zone mémoire	VII-22
M:ABRT fin anormale d'une tâche	VII-22
M:DUMP édition d'une zone mémoire	VII-23
M:IT demande d'opérations sur le système d'interruption	VII-24
M:CMPA comparaison arithmétique de deux mots de 16 bits	VII-25
M:CMPS comparaison de deux chaînes d'octets	VII-26
M:CNEC connexion d'un contexte à un niveau	VII-26
M:ZIO demande d'Entrée/Sortie en zone commune	VII-27
M:ZWAT attente de fin de transfert en zone commune	VII-27
M:ASGN assignation dynamique	VII-28
M:DLAY lancement d'un délai	VII-29
M:ZDLY lancement d'un délai, CB en zone commune	VII-31
M:SDLY suppression d'un délai	VII-31
M:ZSDL suppression d'un délai, CB en zone commune	VII-32

SOMMAIRE (suite et fin)

M:TIME demande de l'heure VII-32
M:PM protection/libération d'une zone mémoire VII-32
M:RQST réservation d'une ressource VII-33
M:RLSE libération d'une ressource VII-33
M:TAR test et réservation d'une ressource VII-34
M:KILL meurtre d'une tâche VII-34
M:CSOL passage à l'état actif d'un sous-système VII-35
M:AFF recherche du périphérique associé à une étiquette opérationnelle VII-35
M:ACTV activation d'un événement VII-36
M:ZACT activation d'un événement, CB en zone commune VII-36

MTR-E COMMUNICATION OPERATEUR -MODULES MONITEUR VIII-1
Généralités VIII-1
Communications opérateur - les commandes VIII-1
%RUN commande de lancement VIII-2
%ASSIGN commande d'assignation VIII-4
%DEVICE action particulière sur périphérique VIII-8
%TRACE commande d'édition dynamique des registres VIII-9
%SNAP commande d'édition dynamique de la mémoire VIII-10
%HALT commande d'arrêt sur instruction VIII-12
%NEXT commande d'exécution d'une séquence d'instructions VIII-14
%EXECUTE commande de reprise du background VIII-16
%PERFORM commande de calcul VIII-18
%*/ commande de sous-système VIII-19
Communications opérateur -messages opérateur VIII-20
Modules moniteur du MTR-E VIII-24

LISTE DES INSTRUCTIONS A-1

1. Introduction

Le MTR (Moniteur Temps Réel) a été particulièrement développé pour assurer avec une configuration de 8 K mots mémoire et télé-imprimeur :

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

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

- Gestion d'une zone foreground et d'une zone background.
- Connexion de programmes à un niveau d'interruption.
- Multiprogrammation assurée par le dispositif matériel de gestion de priorité des interruptions.
- Files d'attente.
- Gestion du temps.
- Ressources (MTR-E).
- Modules réentrants.
- Possibilité d'avoir des sous-systèmes (MTR-E).

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

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

Le Moniteur Temps Réel existe sous deux versions principales : le MTR (version minimum) et le MTR-E (version étendue). Le MTR-E possède, un jeu de commandes supplémentaires ainsi que, pour les commandes déjà existantes dans le MTR des options supplémentaires, principalement destinées à la mise au point des programmes. Elles sont décrites dans le chapitre 8 de ce manuel.

Le MTR peut être étendu par ailleurs par :

1) Adjonction de modules IMT par génération

Ceci concerne :

- Les handlers (modules de contrôle des périphériques).
- Les modules simulant les instructions optionnelles (traitement des déroutements). Ces modules sont précisés dans le chapitre 6 de ce manuel.
- Des programmes immédiats utilisateur connectés à une interruption (bien que l'utilisateur ait la possibilité plus souple et plus simple de les connecter au lancement).

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

2) Adjonction de modules BT par édition de liens

Ceci concerne :

- Les modules de bibliothèque standard (version CSV).
- Des modules utilisateur (qui doivent être normalement réentrants, c'est-à-dire travailler dans le TWB de l'appelant).
- Des modules superviseur standard (adjonction des commandes de mise au point par exemple). L'extension "commandes de mise au point" est disponible dans le MTR étendu et est décrite dans le chapitre 8 de ce manuel.

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

Les éléments qui concernent les encombrements et les modes opératoires sont décrits dans les manuels de génération et les manuels opérateur des bibliothèques IMT du MTR.

2. Structure du système Organisation de la mémoire

II-1. STRUCTURE DU NOYAU RESIDENT

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

a) Des données système

- Adresses de communication avec la micro-machine
- Zone de sauvegarde sur déroutement
- Adresses des tables système
- Adresses de communication entre les modules du superviseur
- Constantes système (pointeurs de contexte, table d'Entrée/Sortie, etc...).

b) Sous-programmes communs (en modules superviseur)

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

c) Les handlers

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

Le MTR comprend toujours au moins le handler téléscriptrice. Tous les handlers sont intégrables à la génération et sont disponibles sous forme IMT.

d) Tâches immédiates

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

- Le programme de traitement de l'interruption pupitre
- Le programme de traitement de l'interruption horloge
- Les programmes de traitement des interruptions "coupure secteur" et "retour secteur"
- Les tâches immédiates utilisateur incluses à la génération. Remarquons à ce sujet qu'il est plus normal, au niveau MTR, d'utiliser la zone foreground pour les tâches immédiates utilisateur.

II-2. ORGANISATION DE LA MEMOIRE

- Zone moniteur

- A partir de l'adresse zéro jusqu'à la fin du noyau résident y compris les tâches immédiates d) ci-dessus.
- Les modules moniteur sont en mode maître.

Zone foreground

C'est une zone s'étendant depuis la fin du noyau résident jusqu'au début de la zone background.

Cette zone reçoit normalement les tâches immédiates utilisateur (connectées à un niveau d'interruption).

Les programmes chargés dans cette zone peuvent être en mode maître ou en mode esclave, avoir accès ou non à des zones protégées suivant les options utilisées à l'édition de liens. Ils peuvent être protégés ou non suivant l'option utilisée au chargement. Remarquons cependant que l'utilisation principale du foreground correspond au chargement des programmes immédiats protégés et ayant accès aux zones protégées.

Zone background

Cette zone s'étend depuis la fin de la zone foreground jusqu'au début de la zone commune.

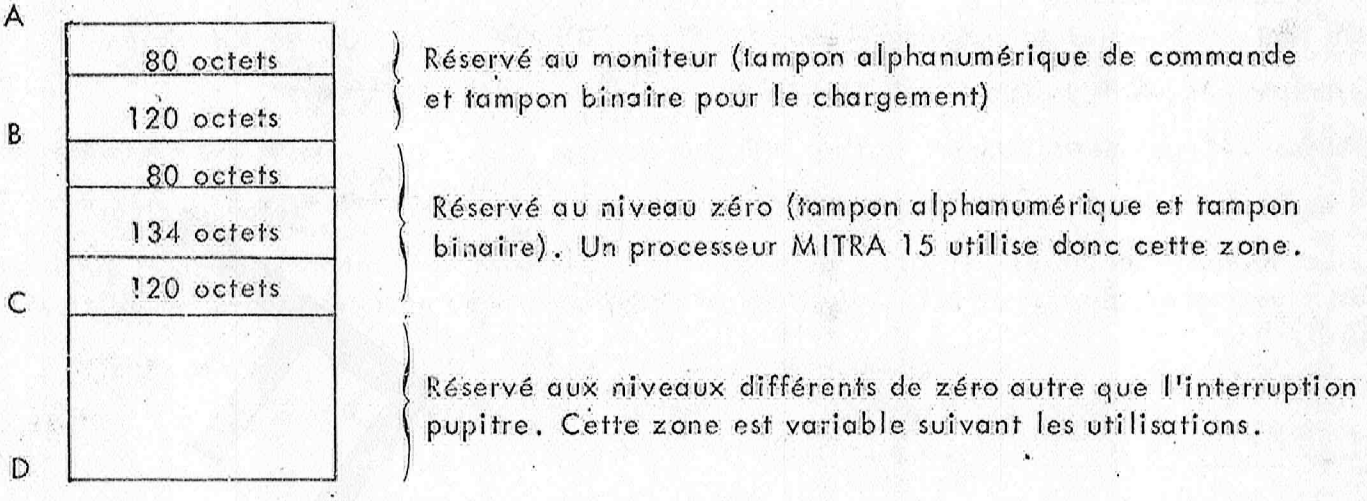
Cette zone est réservée aux programmes associés au niveau zéro (programme utilisateur ou processeur). Il n'y a qu'un programme qui peut être actif dans le background. Ce programme peut être en mode maître ou en mode esclave, avoir accès ou non à des zones mémoire protégées suivant les options utilisées à l'édition de liens. Il peut être protégé ou non suivant l'option utilisée au chargement. Remarquons cependant que l'utilisation principale du background correspond au chargement de programmes non protégés n'ayant pas accès aux zones protégées.

Zone commune

Cette zone s'étend depuis la fin de la zone background jusqu'à la fin de la mémoire.

Elle sert aux communications entre des programmes distincts.

Sa structure est la suivante :



La zone comporte donc au minimum 534 octets. Les moniteurs standard auront donc une zone commune de 534 octets.

La taille de la zone commune peut être fixée de façon différente à la génération.

L'adresse A de la zone commune est disponible dans le quatrième mot de la CDS de chaque

programme utilisateur.

Cette adresse de la zone commune est implantée par le moniteur lors du traitement de la commande %LOAD et est exprimée relativement à la base G du programme concerné.

Lors du traitement de la commande %RUN, le registre X est chargé avec l'adresse relative au début de la zone commune de :

- La zone réservée au niveau zéro s'il s'agit d'un programme background
- La zone réservée aux niveaux différents de zéro s'il s'agit d'un programme foreground.

Ce dispositif assure la compatibilité d'accès à la zone commune sous les différents moniteurs MOB, MTR et MTRD.

Zones mémoire et protection mémoire

L'utilisateur gère librement la protection de ses programmes (%LOAD) et leur accès aux zones protégées (%LINK).

Cependant, en plus des processeurs pouvant tourner au niveau zéro, un système temps-réel opérationnel peut contenir deux types principaux de programmes utilisateur :

- des programmes au point
- des programmes en cours de mise au point

Il faut éviter principalement que les programmes en cours de mise au point puissent modifier les données ou les programmes temps réel au point.

Pour éviter ces interférences, on utilisera la protection mémoire.

On dispose à priori de quatre possibilités :

a) Programme protégé ayant accès aux zones protégées et non protégées.

b) Programme protégé ayant accès seulement aux zones non protégées.

c) Programme non protégé ayant accès aux zones protégées et non protégées.

d) Programme non protégé ayant accès seulement aux zones non protégées.

D'une façon générale, on pourra dire que :

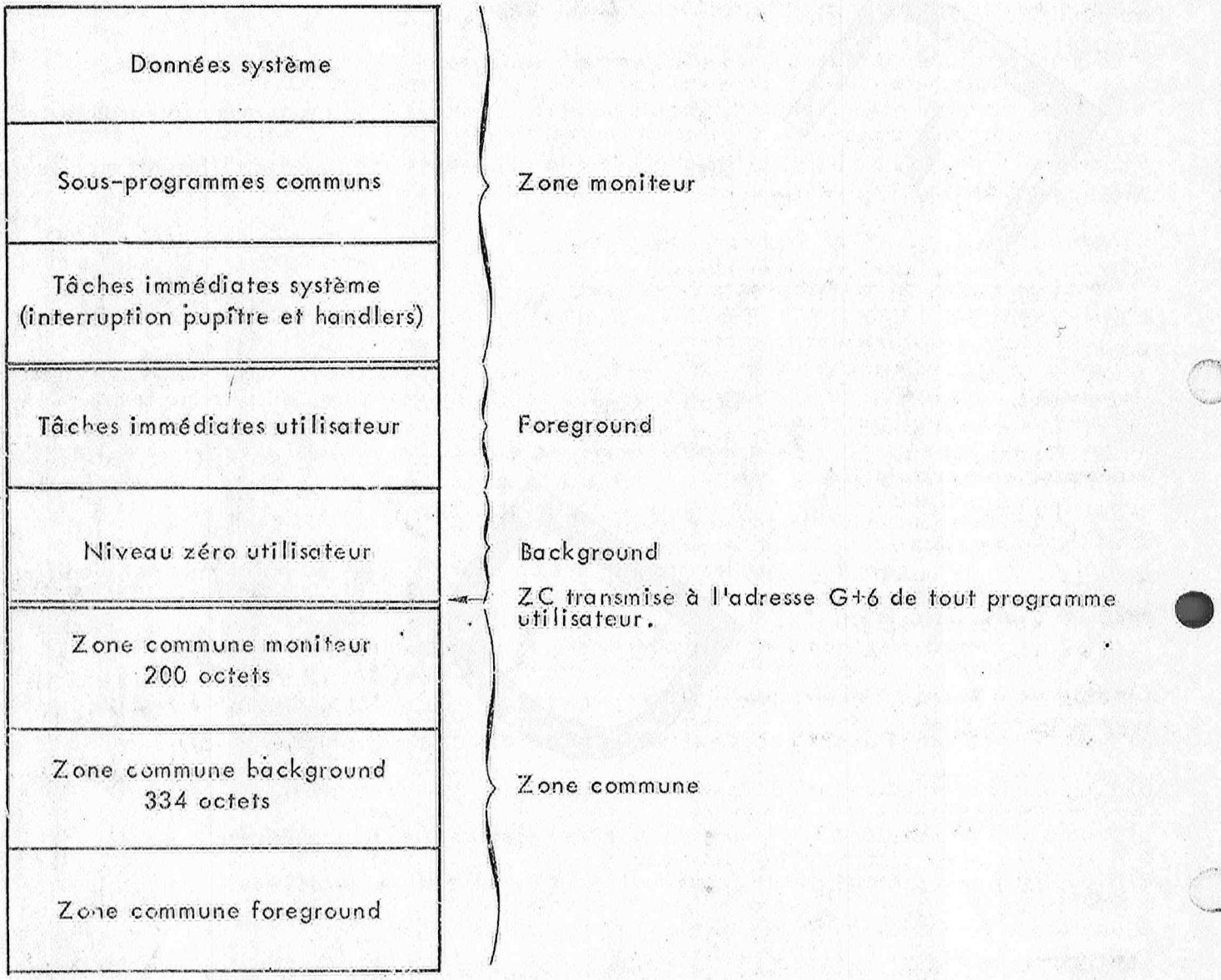
- Le type a) correspond aux tâches immédiates opérationnelles en foreground.
- Le type b) a peu d'intérêt (programme non exécutable)
- Le type c) peut correspondre à certains cas particuliers, par exemple :

. accès par un programme background à des données temps réel protégées, pour lecture

. accès par un programme foreground en cours de mise au point à d'autres données temps-réel.

- Le type d) correspond généralement au background ou à un programme foreground en période probatoire.

Schéma de la mémoire



3. Traitement des entrées/sorties et des délais

Une Entrée/Sortie peut se décomposer en cinq phases :

Phase 1 : Réservation éventuelle d'un tampon.

Phase 2 : Demande de transfert.

Phase 3 : Initialisation du transfert.

Phase 4 : Déroulement du transfert.

Phase 5 : Fin de transfert et contrôle de validité.

La première et la deuxième phase sont à la charge du demandeur d'Entrée/Sortie (appel CSV M:IO).

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

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

Si lors de la demande de transfert, le périphérique est occupé, la demande est mise en file d'attente et le contrôle est rendu au programme appelant.

Pour l'utilisateur tout se passe comme si la phase d'initialisation 3 avait eu lieu : il peut considérer que pour lui, la phase 4 commence, et se mettre éventuellement en attente de la fin de transfert.

Dans ce cas, la phase 4 comprend à la fois l'attente de libération logique du périphérique, l'initialisation et le déroulement effectifs du transfert.

III-1. INTERFACE D'ENTREE/SORTIE

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

- Analyse de la demande de transfert pour un traitement préliminaire éventuel,

- Test d'occupation logique du handler,

- Mise en file d'attente éventuelle de la demande de transfert, et gestion de cette file d'attente,

- Mise en place des paramètres nécessaires à l'initialisation physique et au contrôle du transfert,

- Branchement au module de contrôle du transfert propre au périphérique ("handler"),

- Prise en compte et traitement de certaines erreurs en fin de transfert (seul traitement propre à la phase 5).

C'est l'ensemble de ces traitements communs à tous les transferts, effectués entre l'appel de l'utilisateur et la prise en charge du transfert par un "handler" puis entre la fin du transfert et sa validation pour le compte de l'utilisateur, qui constitue l'interface d'Entrée/Sortie offert par le système d'exploitation.

III-2. MODULE DE CONTROLE D'ENTREE/SORTIE : LE HANDLER

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

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

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

H 1 handler d'initialisation.

H 2 handler de contrôle de transfert.

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

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

a - Transfert par bloc de données,

b - Transfert donnée par donnée.

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

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

III-2.1. Handler 1 : initialisation d'un transfert

Le handler 1 effectue :

a) s'il y a lieu, une analyse de l'état initial du coupleur et/ou du périphérique en ce qui concerne les conditions particulières au type de périphérique, avant de commander le transfert. En cas d'anomalie, le transfert est considéré comme terminé et les conditions d'erreurs sont transmises au superviseur d'Entrée/Sortie.

b) l'envoi d'une commande de transfert de bloc ou une lecture/écriture de donnée, suivant le mode de fonctionnement du coupleur et du handler 2.

c) vérifie la bonne prise en compte de cette initialisation par le coupleur pour les conditions d'anomalies qui ne se traduisent pas par une interruption. En cas d'anomalie, on est ramené au cas a).

d) retour au programme appelant.

Le handler 1 travaille toujours au niveau de priorité du programme appelant

Le programme appelant est généralement le superviseur d'Entrée/Sortie (niveau 1), mais ce peut être aussi le handler 2 en cas de relance automatique des transferts de données.

III-2.2. Handler 2 : contrôle du transfert

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

- La prise en compte sur interruption des phases du transfert, en alternance avec le micro-programme.
- La relance éventuelle du handler 1.
- Le contrôle de la fin du transfert.

A chaque phase, le handler 2 peut déceler des anomalies pouvant entraîner un arrêt du transfert ou la reprise du transfert. Dans ce dernier cas, cette reprise se fera par retour au handler 1.

Le handler 2 travaille toujours au niveau de priorité de l'interruption du couplage

En fin de transfert (anormal ou non), le handler 2 fait un appel au superviseur d'Entrée/Sortie pour lui signaler la fin de transfert et les conditions de validité du transfert, avant de désactiver le niveau d'IT associé.

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

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

Au moment de l'appel, le registre A doit contenir l'adresse du CB. Dans le cas de M:IO, cette adresse est relative à la base G du programme appelant.

Le moniteur associe à la demande de transfert, un évènement dynamique défini par l'adresse du CB; c'est le bit zéro de l'octet zéro du CB :

Reconnaissance de l'appel : évènement désactivé (mise à 1 du bit zéro de l'octet zéro du CB)

Fin de transfert : évènement activé (mise à zéro du bit zéro de l'octet zéro du CB)

Les éléments sont mis en file d'attente (une file d'attente par coupleur).

Le moniteur rend le contrôle au programme utilisateur après la mise en file d'attente. L'initialisation effective du transfert se fait au fur et à mesure de la disponibilité du coupleur concerné.

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

III-4. ATTENTE DE FIN DE TRANSFERT : CSV M:WAIT

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

Le transfert se termine après remise à zéro du bit évènement : le moniteur active l'évènement.

Après l'initialisation du transfert, ou la mise en file d'attente si le coupleur est occupé, l'utilisateur peut faire appel au module M:WAIT.

Le rôle de M:WAIT est de :

- Désactiver l'interruption de l'appelant, si celui-ci n'est pas au niveau zéro, pour permettre aux programmes connectés à un niveau d'interruption inférieur à l'appelant d'être exécutés. L'appelant sera réactivé lors de la fin effective du transfert.

- Effectuer une boucle d'attente d'activation d'évènement (attente fin de transfert) si le programme appelant est au niveau zéro.

- Effectuer le branchement sur "adresse de reprise en cas d'erreur", si celle-ci est spécifiée dans le CB concerné.

III-5. ETIQUETTES OPERATIONNELLES

Une demande d'Entrée/Sortie n'est pas adressée directement à un handler. En effet, les handlers gèrent directement l'environnement physique. Une liaison directe aux handlers rendrait la programmation des Entrées/Sorties toujours tributaire d'un environnement physique précis.

C'est pour se libérer de cette contrainte qu'a été utilisé le système des "étiquettes opérationnelles" de façon qu'un programme travaille sur un environnement logique.

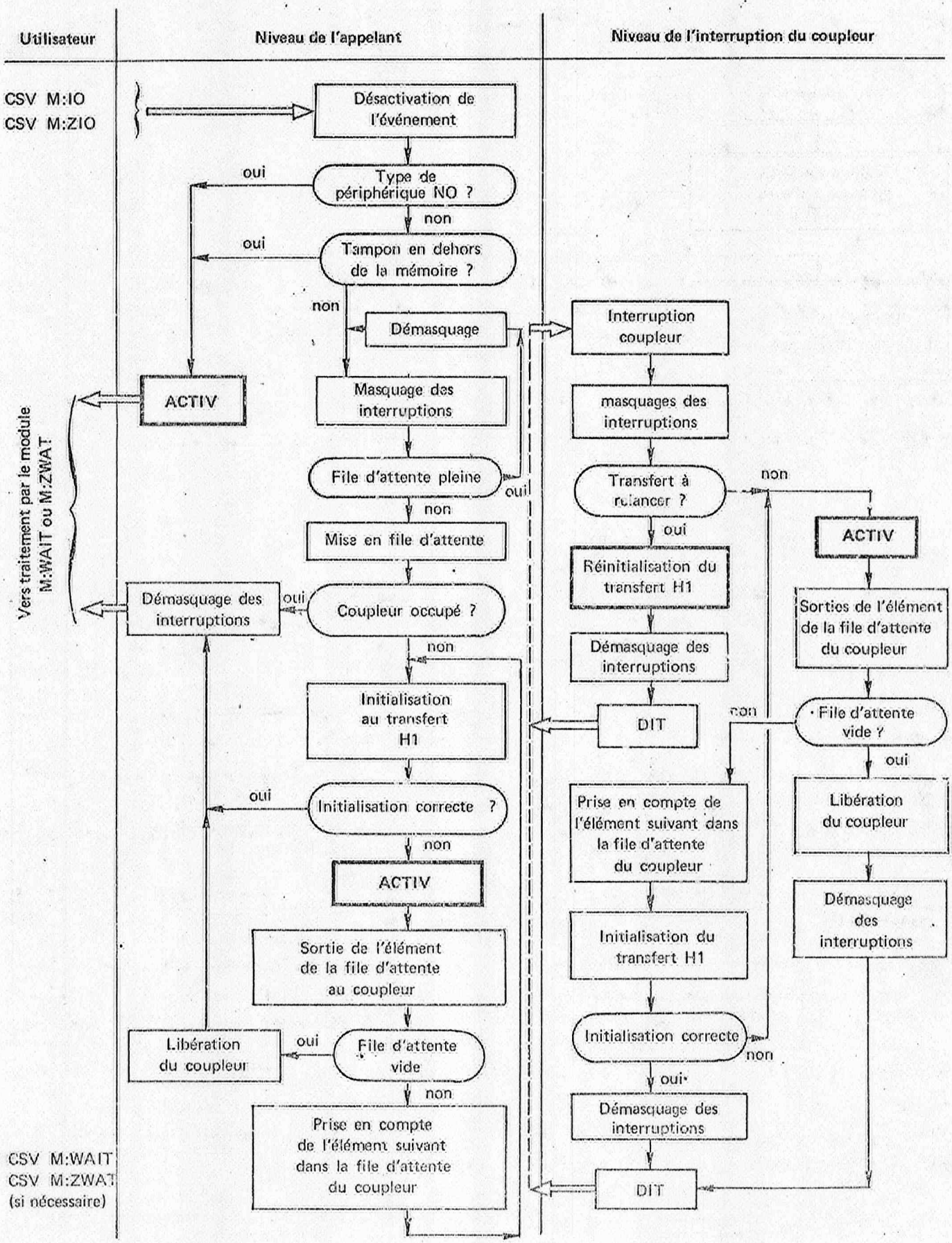
Une Entrée/Sortie sera adressée à une étiquette opérationnelle (voir description du CB d'Entrée/Sortie au chapitre VIII). La correspondance entre étiquette opérationnelle et handler sera faite par le module M:IO. Elle est fixée à la génération du moniteur et modifiable par la commande % ASSIGN.

Les étiquettes opérationnelles, décrites avec le module M:IO et la commande % ASSIGN, représentent des fonctions logiques (Entrée de commande, Sortie de binaire, etc...).

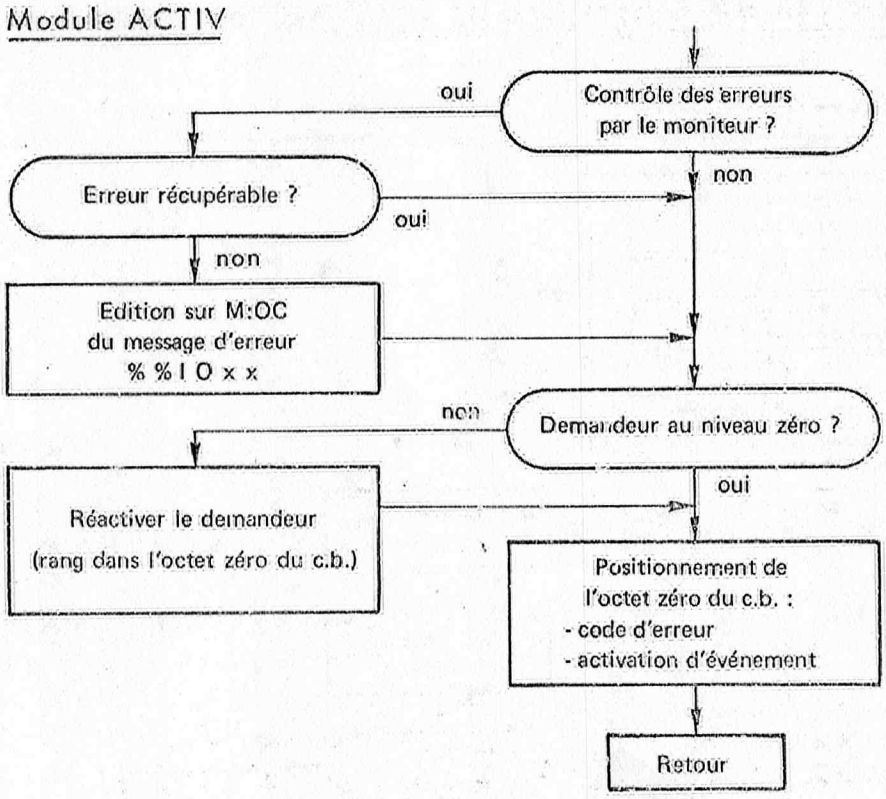
Dans MTR, il existe un seul jeu d'étiquettes opérationnelles utilisées à la fois par les programmes en background (niveau zéro) et les programmes en foreground (niveau différent de zéro). Sous MTR-E, il existe deux jeux d'étiquettes opérationnelles, un réservé au foreground (option F de la commande %ASSIGN), l'autre réservé au background. Cette séparation permet d'assurer une indépendance logique entre le temps réel et le travail de fond.

Il faut remarquer que la fonction %LOAD est considérée comme une fonction temps-réel puisqu'elle s'effectue au niveau de l'interruption opérateur.

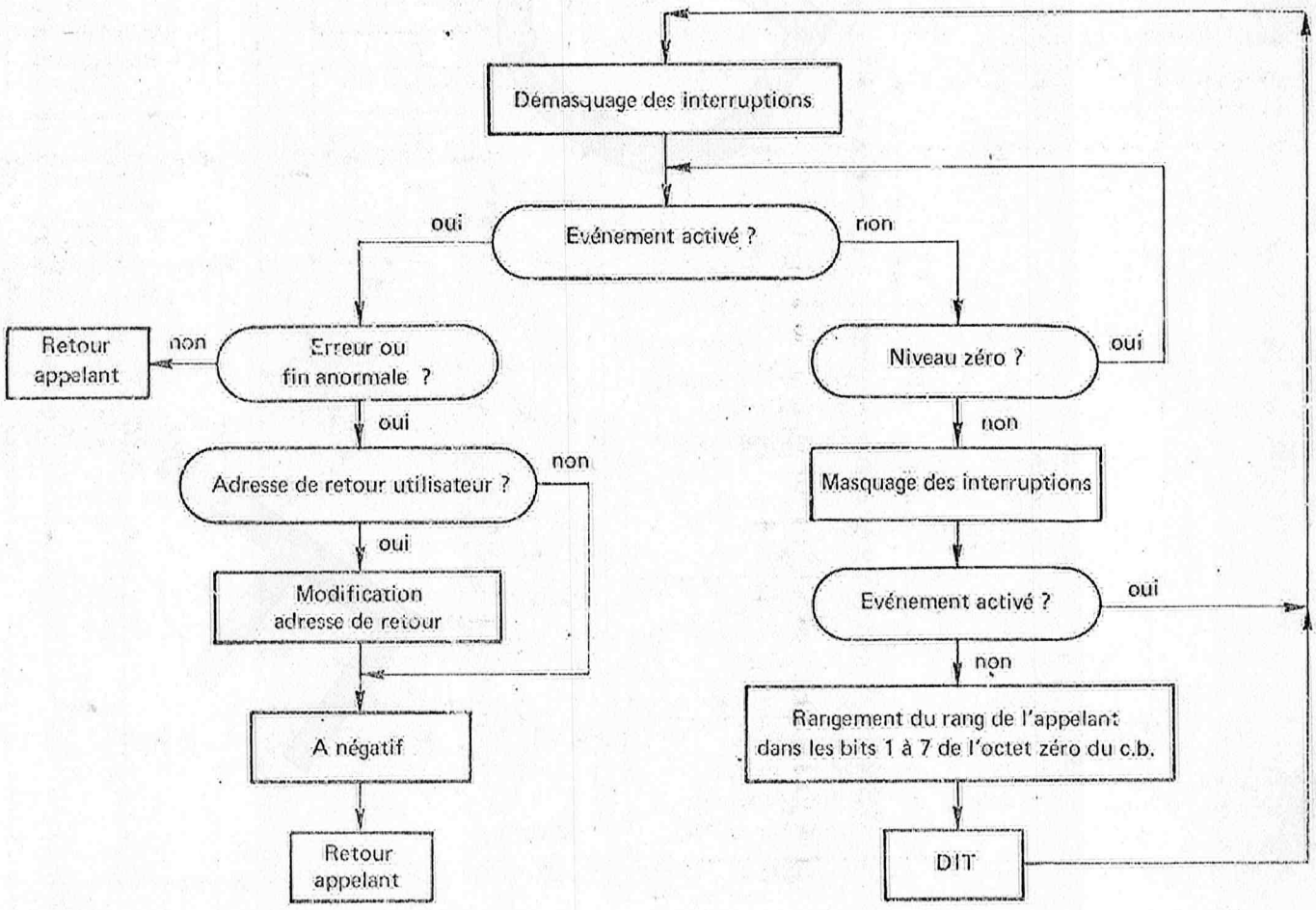
III-6. SCHEMA D'ORGANISATION D'UNE ENTREE/SORTIE SOUS MTR



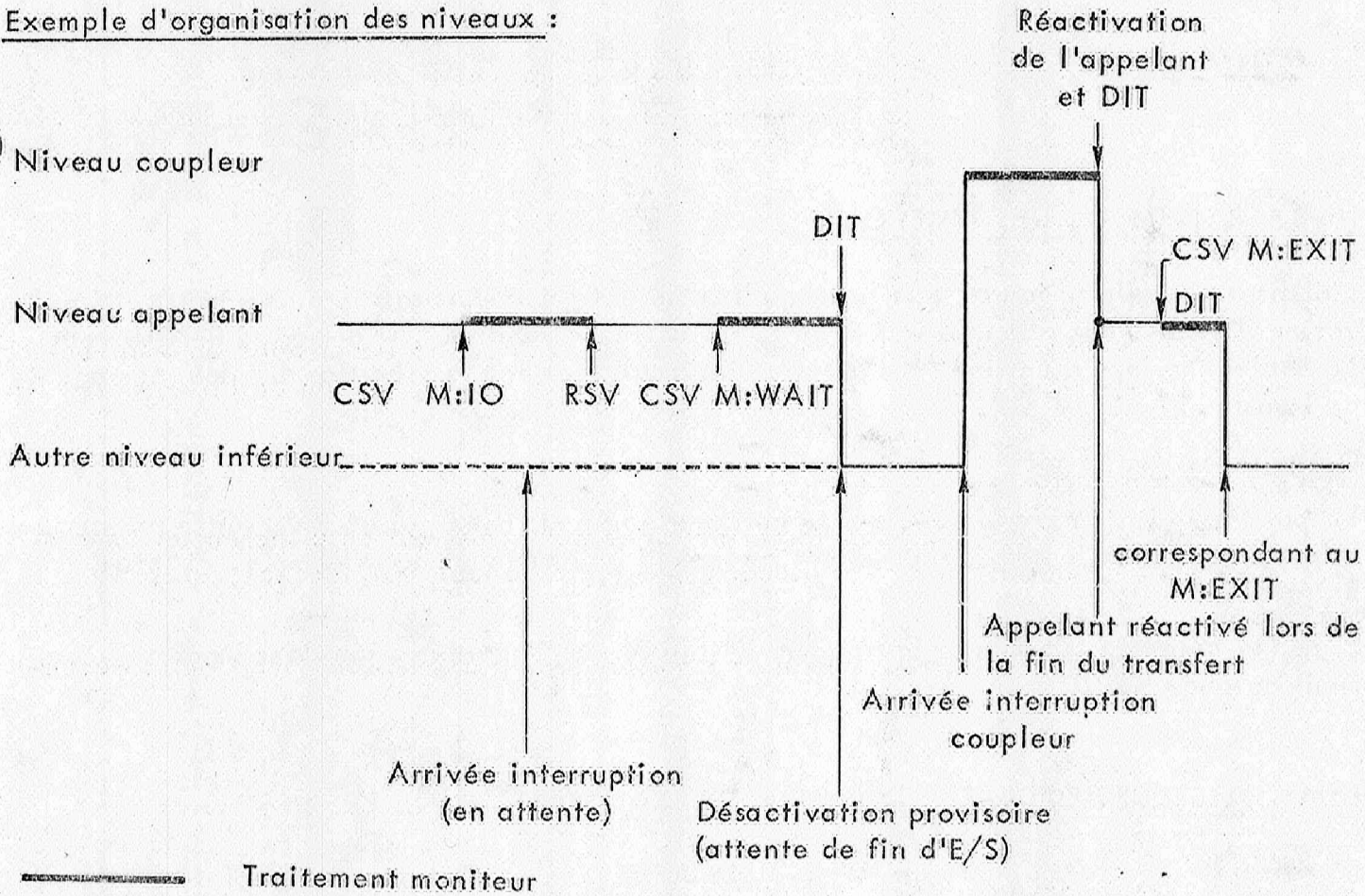
Module ACTIV



Module M:WAIT



Exemple d'organisation des niveaux :



III-7. Traitement des périphériques non-prêts

Il est fréquent qu'un périphérique ne soit pas prêt et que son passage en "prêt" puisse ce faire par une action opérateur.

Il s'agit des incidents suivants : périphérique en "manuel", absence de papier, bourrage, magasin d'alimentation vide, magasin de réception plein. Plus précisément, il s'agit des incidents pour lesquels un handler a renvoyé un code d'erreur supérieur à 7A (hexadécimal).

Ce sont toujours des erreurs physiques à l'initialisation du transfert.

Sur ce type d'erreur, le système effectue une reprise automatique, en retentant le transfert jusqu'à ce que le périphérique devienne "prêt", après avoir édité sur M : CC un message :

%%XXYZ où

XX désigne le périphérique = CR-LP-DC ....etc

Y désigne l'unité de traitement à laquelle est connectée le périphérique :

0	UC
1	UEM1
2	UEM2
3	UEM3

Z désigne le numéro de DEVICE pour un coupleur multipériphérique. $\in$ [0,F]

Exemples :

%%LP0C imprimante non prête

%%9T10 dérouleur de bande 0 sur UEM1 non prêt.

Remarque :

Si le moniteur utilisé possède la gestion du temps, l'utilisateur pourra obtenir l'interruption pupitre même si le périphérique est non opérationnel.

Si le moniteur ne possède pas la gestion du temps, l'utilisateur ne pourra poursuivre ou aborter le programme demandeur, que si le périphérique est passé en "prêt".

III-8. TRAITEMENT DES TIME-OUT

L'utilisateur peut demander que le temps écoulé entre l'initialisation du transfert et l'activation d'évènement soit inférieur à un temps défini à l'avance : il demande un time-out. Le déclenchement d'un time-out entraîne automatiquement l'activation de l'évènement fin de transfert.

Cette activation se fait :

1) par l'envoi d'un ordre STOP sur le périphérique pour ceux qui en disposent et donc la transmission dans l'octet évènement du code &4E (voir manuel de référence, chapitre Entrées/Sorties).

2) par la détection de l'interruption fin de transfert et la transmission dans l'octet évènement du code &30.

III-9. INTERRUPTION EXTERNE EN FIN DE TRANSFERT

L'utilisateur peut demander que l'activation de l'évènement déclenche une interruption externe dont le niveau est spécifié dans le CB associé.

III-10. TRAITEMENT DES DELAIS

Au call superviseur (CSV) pour demande de délai est associé un bloc de commande, le CBD (Control Block of Delay). Ce bloc de commande contient les paramètres définis par l'utilisateur. (Voir description du CBD au paragraphe de description de M:DLAY).

Le moniteur associe à la demande de transfert, un évènement dynamique défini par l'adresse du CB; c'est le bit zéro de l'octet zéro du CB :

- Reconnaissance de l'appel : évènement désactivé (mise à 1 du bit zéro de l'octet zéro du CB).

- Fin de délai : évènement activé (mise à zéro du bit zéro de l'octet zéro du CB).

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

Le rôle de M:WAIT est de :

- Désactiver l'interruption de l'appelant, si celui-ci n'est pas au niveau zéro, pour permettre aux programmes connectés à un niveau d'interruption inférieur à l'appelant d'être exécutés. L'appelant sera réactivé lors de la fin de délai.

- Effectuer une boucle d'attente d'activation d'évènement (attente de fin de délai) si le programme appelant est au niveau zéro.

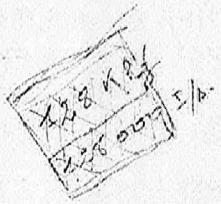
L'utilisateur peut demander que l'activation de l'évènement déclenche une interruption externe dont le niveau est spécifié dans le CBD associé.

4. Interruptions

Le MTR assure le contrôle des interruptions du système. Il contrôle :

- Les interruptions des coupleurs dont les handlers ont été inclus à la génération.
- L'interruption pupître.
- Les interruptions de coupure et retour secteur
- Toute autre interruption à laquelle on voudra connecter une tâche immédiate (par %RUN ou à la génération).

65



IV-1 L'INTERRUPTION PUPITRE (OU OPERATEUR)

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

Le module de programme immédiat lié à cette interruption :

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

IV-2 L'INTERRUPTION HORLOGE

Le système utilise pour la gestion du temps une voie de comptage dont la fréquence de base est 12,8 khz et qui génère une interruption au passage à -1.

Le module de programme immédiat lié à cette interruption permet :

- la gestion de la date
- la gestion de time-out
- la gestion de délais

L'horloge est lancée au chargement du système.

IV-3. LES INTERRUPTIONS DE COUPURE ET RETOUR SECTEUR

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

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

%%CS
SYS READY

La réinitialisation effectuée consiste en :

- réinitialisation du noyau moniteur (en particulier des tables d'Entrée/Sortie),
- réinitialisation des contextes de tous les programmes immédiats (exceptés ceux des niveaux 30 et 31) et des handlers avec le premier élément de leur PRT. Tous les programmes immédiats devront donc être lancés par leur première section et tous les handlers devront avoir comme premier LPS le handler 2.

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

- mise en attente du système (attente d'interruption)
- l'horloge est remise à zéro et relancée.

IV-4. LES INTERRUPTIONS DES COUPLEURS (TRAITES PAR HANDLER)

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

IV-5. AUTRES INTERRUPTIONS

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

Elles seront connectées dynamiquement à une tâche utilisateur par la commande %RUN. Elles pourront alors être armées et/ou validées (commande %IT) et éventuellement excitées par la commande %ACTIVATE. L'utilisateur pourra commander ces diverses fonctions par programme (appel CSV M:IT).

IV-6. FIN D'UNE TACHE

- Fin normale :

Une tâche se termine normalement par un appel moniteur CSV M:EXIT. Le système désactivera alors le niveau de la tâche appelante, le contexte courant de celle-ci étant protégé pour assurer la relance de la tâche si le niveau est activé de nouveau ultérieurement.

Le CSV M:EXIT devra donc normalement être suivi d'un branchement sur une séquence de réinitialisation de la tâche.

- Fin anormale :

L'utilisateur dispose de la commande %ABORT pour abandonner la tâche dont le niveau est indiqué dans la commande (voir chapitre VI).

En cas de déroutement dans un programme immédiat il y a réinitialisation de l'ensemble du système car le programme immédiat fautif à vraisemblablement perturbé l'ensemble du système (voir chapitre V).

5. Déroutements

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

V-1. DIFFERENTS TYPES DE DEROUTEMENT

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

Lorsqu'elle détecte l'incident, la micro-machine :

- Protège dans les octets 4 à 9 de la mémoire, le contenu de L et P et des indicateurs,
- Indique par des bits du mot 2 de la mémoire, qu'elle est la cause du déroutement,
- Réalise l'appel à la section zéro du superviseur (module M:TRAP).

V-2. SIMULATION DES INSTRUCTIONS OPTIONNELLES

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

. Type 0	Pas de simulation
. Type 1	Simulation de DIV, SLLD, SRLD, PTY, NLZ
. Type 2	Simulation de FAD, FSU, FMU, FDV
. Type 3	Simulation de DIV, SLLD, SRLD, PTY, NLZ, FAD, FSU, FMU, FDV
. Type 4	Simulation de DIV
. Type 8	Simulation de DIV, SLLD, SRLD, PTY, NLZ, FAD, FSU, FMU, FDV, MVS, TRS, CPS

V-3. M:TRAP TYPE 0

V-3.1. Déroutement "programme" au niveau zéro

Le module M:TRAP de type 0 provoque :

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

. Mot d'état déroutement

. P absolu
. L absolu
. Indicateurs

} sur l'instruction ayant provoqué le déroutement

. P - G
. L - G
. Indicateurs

} sur le dernier appel au moniteur

%%DR xx

xx = 01 si déroutement dû à un coupleur

xx = 02 si déroutement programme.

- L'ABORT du programme en cours

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

Mot d'état déroutement :

Les différents chiffres binaires de ce mot ont le sens suivant :

Bit 15	PG	(1 déroutement sur programme, 0 déroutement dû à un coupleur),
Bit 14	ES	(Horloge de garde d'Entrée/Sortie),
Bit 13	II	(Instruction inexistante),
Bit 12	PA	(Parité mémoire),
Bit 11	AI	(Adresse inexistante),
Bit 10	PM	(Protection mémoire),
Bit 9	VM	(Violation de mode).

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

V-3.2. Déroutement "coupleur" ou déroutement "programme" à un niveau différent de zéro

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

- réinitialisation du noyau moniteur (en particulier des tables d'Entrée/Sortie),

- réinitialisation des contextes de tous les programmes immédiats (exceptés ceux des niveaux 30 et 31) et des handlers avec le premier élément de leur PRT.

Tous les programmes immédiats devront donc être lancés par leur première section et tous les handlers devront avoir comme premier LPS le handler 2.

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

- édition de :

%%DRxxyy où : xx niveau de la tâche qui a déroutée
SYS READY yy contenu du mot 2 de la mémoire qui donne la nature du déroutement

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

V-4. LES AUTRES TYPES DU MODULE M:TRAP

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

Pour les instructions simulées, le déroutement est transparent pour l'utilisateur. Dans tous les autres cas, le traitement est identique au traitement offert par le module M:TRAP de type 0.

Remarques :

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

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

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

3) Si un déroutement se produit au cours de la simulation d'une instruction inexistante, il y aura édition du message de déroutement et abandon du programme ayant utilisé cette instruction. Les informations éditées concerneront l'instruction simulée et non l'instruction déroutante. Seule en effet, l'instruction simulée intéresse alors l'utilisateur. Rappelons que le traitement des instructions flottantes est réentrant.

4) Si un déroutement se produit au cours du traitement du déroutement la machine sera arrêtée après affichage de :

Voyants adresse : &00FF

Voyants donnée : Mot d'état du deuxième déroutement.

5) Le CSV de la section 0 est autorisé par le hardware. Le résultat en sera le même qu'un déroutement "programme", mais le mot d'état déroutement sera nul.

6. Communications avec l'opérateur M : OC

VI-1. COMMANDES OPERATEUR

Principe

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

Lorsqu'une interruption pupître est prise en compte, MTR édite sur M:OC le caractère % et connecte la téléscriptrice en entrée d'une commande. Sur l'entrée de "retour-chariot", la commande est analysée et exécutée généralement au niveau de l'interruption pupître. Il n'y a pas cependant de blocage du niveau zéro. Par ailleurs, l'impression des DUMP s'effectue au niveau zéro, ce qui rend ceux-ci interruptibles par l'interruption pupître. Une grande souplesse d'utilisation est ainsi assurée.

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

Les commandes reconnues par MTR sont les suivantes :

% LOAD	Chargement d'un IMT
% RUN	Lancement du dernier programme chargé
% ABORT	Abandon d'un programme
% FOREGROUND	Déplacement de la limite du Foreground
% DISPLAY	Edition des éléments système
% DUMP	Edition de mémoire
% ASSIGN	Assignation des étiquettes opérationnelles
% DEVICE	Action particulière sur un périphérique
% TIME	Initialisation de la date
% X	Abandon du background et dump post-mortem
% Y	Abandon du background sans dump post-mortem
% MODIFY	Modification mémoire
% IT	Armement et/ou validation d'interruption
% ACTIVATE	Excitation d'une interruption
% PM	Protection-déprotection

Normes utilisées pour les commandes

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

— Une commande commence par %.

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

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

— Le caractère ":" a deux utilisations :

1) Séparateur : il a le sens général d'affectation numérique. Les éléments qu'il sépare ne sont donc pas syntaxiquement permutables. Il est utilisé dans ce sens dans la commande MODIFY et la commande SNAP (voir extension MTR).

2) Caractère alphabétique : il figure à ce titre dans les éléments du genre étiquette opérationnelle, type de périphérique, etc... sous la forme M : xx, C : zz, D : t.

Exemples :

T : PR M : 5! etc...

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

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

— Le caractère ";" a le sens d'un caractère suite et figure, quand il est utilisé, comme dernier caractère d'une commande.

Dans le MTR, un point virgule comme dernier caractère d'une commande reconnecte la téléscriptrice en entrée pour une nouvelle commande après l'exécution de la précédente sans qu'il soit besoin de refaire une interruption pupitre.

— Le caractère "@" indique que ce qui suit est une adresse. Les adresses spécifiées doivent être paires.

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

Conventions pour la description des commandes et de leurs paramètres

— Les accolades { } indiquent qu'un des termes inclus doit être présent.

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

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

— Deux termes superposés s'excluent mutuellement.

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

Remarque importante :

Toutes les adresses spécifiées dans les commandes opérateur doivent être des adresses-mot (adresses paires).

Exemple :

Une commande étant spécifiée comme suit :

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

Les écritures suivantes sont correctes :

% DUMP

% DU

% DU/M

% DUMP/M

Adresse en hexadécimal ↓ Adresse en décimal ↓

% DUMP/A, @ &xxxx, @ yyyy

% DU/@ &xxxx

Une commande étant spécifiée comme suit :

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

Les écritures suivantes sont incorrectes :

% LOAD/A,F @ xxxx — il manque un séparateur entre F et l'adresse. Sera considéré comme une demande de chargement à l'adresse absolue zéro.

% LOAD/A,F, xxxx — Il manque le caractère @, erreur de syntaxe.

% LOAD commande de chargement

• Définition

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

• Forme

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

• Sens des paramètres

[F]

- Paramètre présent : chargement en zone foreground et vérification du débordement.
- Paramètre absent : chargement en zone background et vérification du débordement.

[P]

- Paramètre présent : le programme sera protégé
- Paramètre absent : le programme sera non protégé

[A]

- Paramètre présent : l'adresse spécifiée est absolue. Dans ce cas il est prévu une possibilité supplémentaire; en spécifiant une adresse absolue, l'utilisateur aura la possibilité de charger un module de données en zone commune foreground (ce qui peut être utile par exemple pour simuler un périphérique temps-réel au cours de la mise au point d'un programme).
- Paramètre absent : l'adresse spécifiée est relative au début de la zone foreground ou au début de la zone background suivant que le paramètre [F] est présent ou non.
- Ce paramètre n'a pas de sens si aucune adresse n'est spécifiée.

[@ [&] xxxx]

- Adresse de chargement. Cette adresse est absolue si le paramètre [A] est présent. Elle est relative au début de la zone foreground si le paramètre [A] est absent et si le paramètre [F] est présent. Elle est relative au début de la zone background si les paramètres [A] et [F] sont absents.
- Quand ce paramètre est absent, l'adresse a pour valeur implicite zéro; c'est-à-dire en début de foreground si le paramètre [F] est présent et au début du background si le paramètre [F] est absent. Si le paramètre [A] est présent il doit y avoir une adresse.

[S] - Edition en fin de chargement sur M:OC du contenu de la PRT du programme sous la forme suivante à raison de huit mots par ligne :

```
xxxx  yyyy      xxxx  yyyy  .....
 :     :         :     :
L-G   P-G       L-G   P-G
Section n       Section n-1 etc...
```

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

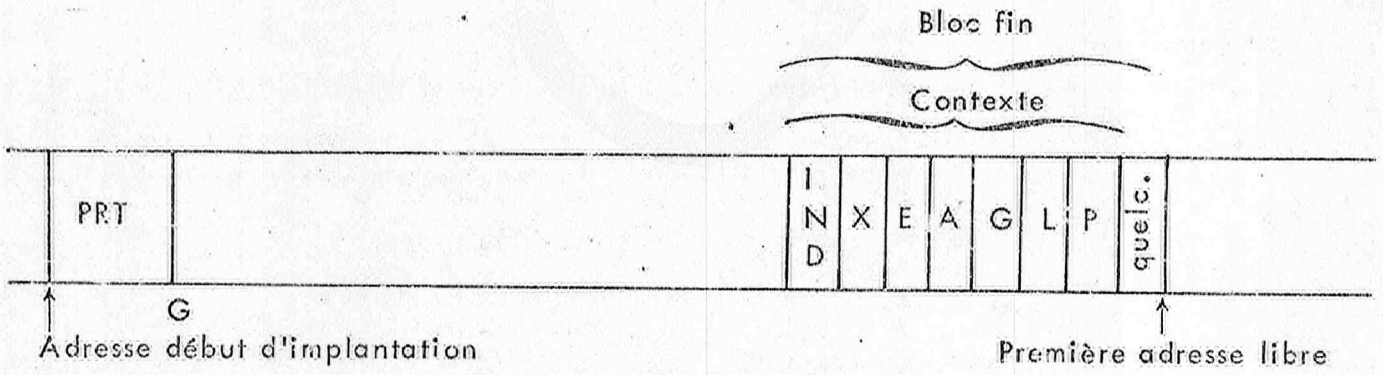
Le chargeur (loader) charge l'IMT du programme y compris les 16 octets du bloc fin (comprenant le contexte) et laisse après chargement, en G+6, l'adresse relative à G de la zone commune.

Il édite sur M:OC (téléscriptrice) les informations suivantes :

AAAAAA Nom du programme (figurant dans l'en-tête de l'IMT)

```
xxxx  yyyy  xxxx  yyyy  ...
 :     :     :     :
```
} PRT à raison de huit mots par ligne si l'option S est présente

```
zzzz  zzzz  zzzz
```
- zzzz (3e) : Première adresse libre après le programme (absolue)
- zzzz (2e) : Valeur de la base G (absolue)
- zzzz (1e) : Adresse de début d'implantation (absolue)



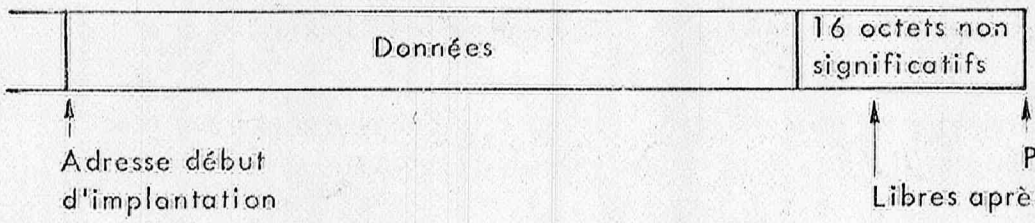
• Effet en cas de chargement d'un module de données

Le chargeur (loader) charge le bloc de données IMT y compris 16 octets non significatifs.

Il édite ensuite sur M:OC (téléscriptrice) les informations suivantes :

AAAAAA Nom du module (figurant dans l'en-tête de l'IMT)

```
zzzz  zzzz  zzzz
```
- zzzz (3e) : Première adresse libre (absolue)
- zzzz (2e) : Taille en octets de l'IMT chargé
- zzzz (1e) : Adresse de début d'implantation (absolue)



Rappelons qu'un module de données est obtenu par édition de liens avec l'option DA et qu'il est possible de charger un module de données non seulement en foreground ou en background (cas usuel) mais aussi en zone commune foreground si l'on spécifie l'option [A] et une adresse absolue correspondante.

Remarques :

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

2) Le tampon d'entrée binaire de 120 octets utilisé en zone commune est réservé au chargement, si bien que le chargement d'un programme en foreground ne perturbe en rien le fonctionnement d'un processeur en background : le déroulement du chargement ne bloque pas le niveau zéro.

Exemples :

% LOAD	- Chargement à la première adresse du background
% LOAD/@ &1000,F	- Chargement à l'adresse &1000 relative au début du foreground
% L/A,@ 2500	- Chargement à l'adresse décimale absolue 2500
% LOAD/A,@ &2FF0,S	- Chargement à l'adresse absolue &2FF0 et édition de la PRT.
% L/P,F	- Chargement à la première adresse du foreground et protection
% L/F,A,@ &2100	- Chargement à l'adresse absolue &2100 et contrôle du débordement de la zone foreground.

% RUN commande de lancement

- Définition

Commande lançant l'exécution du dernier programme chargé.

- Forme

$$\left\{\begin{matrix}\%\ \text{RUN}\\ \%\ \text{R}\end{matrix}\right\} \ / \ \left[\text{N}\ [\&]\ xx\right], \left[\text{S}\ [\&]\ xx\right]$$

- Sens des paramètres

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

- Si ce paramètre est absent, le niveau zéro est pris par défaut.

[S [&]xx] - Si ce paramètre est présent, l'exécution sera lancée sur la section de numéro [&]xx. C'est l'élément de PRT de numéro [&]xx qui sera utilisé pour charger les bases L et P.

- Si ce paramètre est absent, les bases L et P seront chargées avec leur valeur courante (figurant dans le contexte du programme).

• Effet

Chargement du pointeur de contexte du niveau concerné.

Le contexte du programme est chargé avec les valeurs suivantes :

A	Bits 0 à 7 : niveau de la tâche appelante (ici, niveau de l'interruption pupitre) Bits 8 à 15 : niveau de la tâche lancée
E	Chargé avec zéro
X	Adresse relative au début de la zone commune moniteur (ZC) de la zone commune background ou de la zone commune foreground suivant que le programme est lancé au niveau zéro ou non.
G	Base G du programme
L	Base L de la section de lancement
P	Base P de la première instruction à exécuter

• Rappel

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

Remarques :

1) On ne peut faire RUN qu'une fois sur un programme. En particulier, un programme au niveau zéro ayant fait appel à M:EXIT ou ayant été aborté ne peut être relancé.

2) Un programme à un niveau différent de zéro devra se terminer par un appel à M:EXIT suivi d'un branchement sur sa séquence de réinitialisation. C'est l'arrivée des interruptions sur son niveau qui réglera le lancement et le relancement effectif du programme. Ce programme occupera le niveau auquel il a été connecté par %RUN tant qu'il n'aura pas été aborté.

3) On ne peut connecter un programme qu'à un niveau libre.

4) On ne peut connecter au niveau zéro qu'un programme chargé au background. On ne peut connecter à un niveau différent de zéro qu'au programme chargé en foreground.

Exemples :

% R % RUN % RUN/N&00	Lancement au niveau zéro
% RUN/N4	Connexion au niveau 4
% RUN/N&F	Connexion au niveau 15
% R/S01	Lancement au niveau zéro par la section 1
% R/N4,S&0A	Connexion au niveau quatre par la dixième section

% ABORT commande d'abandon d'un programme

- Définition

Commande permettant d'abandonner le programme en cours à un niveau donné ainsi que ses Entrées/Sorties.

- Forme

{ % ABORT / % AB } / [N [&] xx]

- Sens des paramètres

[N [&] xx] - Si ce paramètre est présent, [&] xx indique le niveau d'interruption auquel est connecté le programme à abandonner.

- Si ce paramètre est absent, le niveau zéro est pris par défaut.

- Effet

Le programme est abandonné et le niveau correspondant est connecté à un contexte d'attente :

- DIT
  BRU $-1 si le niveau est différent de zéro
- BRU $ si le niveau est zéro.

Dans tous les cas, édition sur M:OC (téléscriptrice) des informations suivantes sur le programme aborté :

```
xxxx xxxx xxxx
  |    |    |
  |    |    Première adresse libre après le programme (absolue)
  |    Valeur de la base G (absolue)
  Adresse de début d'implantation (absolue)
```

%%Axxx où xxx spécifie le niveau aborté :

000 à 00F pour les 16 premiers niveaux
100 à 10F pour les 16 derniers niveaux

Un programme aborté est éliminé logiquement du système et ses Entrées/Sorties en cours ou en file d'attente abandonnées ainsi que les délais lancés éventuellement par le programme, et libération des ressources occupées.

Exemples :

```
% AB
     1500  150C  1546
%% A000
% AB/N2
     1400  1404  1430
%% A002
% AB/N&15
     14B0  14B4  14D0
%% A105
```

% FOREGROUND déplacement de la limite du foreground

- Définition

Commande permettant d'étendre ou de réduire la zone foreground : adaptation dynamique du système aux besoins utilisateur.

- Forme

$$\left\{\begin{array}{l}\%\ \text{FOREGROUND}\\ \%\ \text{FO}\end{array}\right\} \ / \left[\ @\ [\&]\ \text{xxxx}\ \right]$$

- Sens des paramètres

`[@ [&]xxxx]` - Indique relativement à la fin du moniteur l'adresse de la première adresse libre après le foreground. Il s'agit donc de la taille du foreground en octets.

- Si ce paramètre vaut zéro, le foreground est supprimé et tout l'espace compris entre le moniteur et la zone commune est alloué au background.

- Si ce paramètre est absent, le background est supprimé et tout l'espace compris entre le moniteur et la zône commune est alloué au foreground.

- Effet

Modification de la zone mémoire réservée aux tâches immédiates utilisateur (connectées à un niveau d'interruption).

- La réduction du foreground ne peut se faire qu'après que l'utilisateur ait vérifié qu'aucune tâche immédiate déjà en mémoire et pouvant être activée n'utilisera la zone mémoire réaffectée au background.

- L'extension du foreground ne peut se faire que si la zone background ne contient pas de programme en cours. Si cela était, l'utilisateur devra préalablement aborter (% ABORT) le niveau zéro.

% DISPLAY édition des éléments foreground et système

Définition

Commande de visualisation sur M:OC des éléments d'implantation permettant de gérer le foreground.

Forme

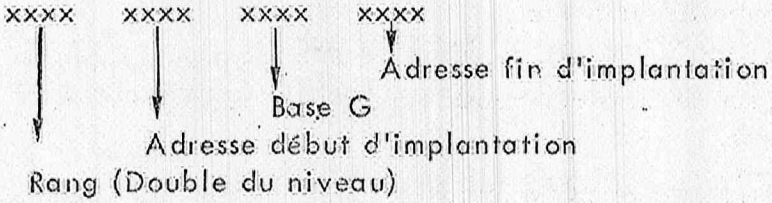
{ % DISPLAY
{ % DI

Effet

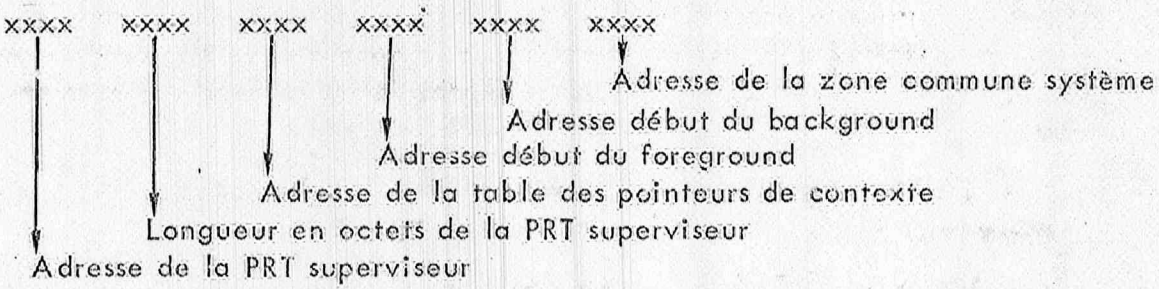
On appellera "niveau actif" un niveau d'interruption auquel est connecté un programme immédiat.

La commande DISPLAY édite sur M:OC en hexadécimal les éléments suivants :

1) Pour chaque niveau actif



2) Pour le système



% DUMP commande d'édition mémoire

Définition

Edition sur M:LO en hexadécimal d'une zone mémoire. L'édition est faite au niveau zéro.

Forme

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

Sens des paramètres

[M] Edition de toute la mémoire

[A] Quand ce paramètre est présent, les adresses spécifiées sont absolues.

Quand ce paramètre est absent, les adresses spécifiées sont relatives à la base G du dernier programme chargé.

[@ [&] xxxx]

Edition de l'élément adressé soit :

1 mot si option OC
1 ligne de dump si option LO

[@ [&] xxxx], [@ [&] yyyy]

Edition de la zone mémoire comprise entre les deux adresses spécifiées. Si le paramètre LO est spécifié, l'édition sera toujours arrondie à un nombre entier de lignes.

Ces adresses sont absolues si le paramètre [A] est présent; sinon elles sont relatives à la base G du dernier programme chargé.

Si aucune adresse n'est spécifiée, édition de la zone mémoire comprise entre les adresses début et fin d'implantation du dernier programme chargé.

[{ OC / LO }]

• Etiquette opérationnelle sur laquelle est effectuée l'édition, le format de l'édition étant un peu différent suivant l'étiquette opérationnelle spécifiée :

- OC : Edition sur M:OC à raison de huit mots par ligne le nombre exact de mots demandés étant édité (sans indication des adresses).

- LO : Edition sur M:LO à raison de 8 ou 16 mots par ligne. Un nombre entier de lignes est toujours édité selon le format suivant : chaque ligne commence par l'adresse absolue du premier octet mémoire édité dans cette ligne. Le nombre de mots édités par ligne est fixé à la génération. Du point de vue standard, seuls les moniteurs comportant le handler imprimante utiliseront le format de 16 mots par ligne.

• Si ce paramètre est absent, la valeur prise par défaut sera LO

Remarques :

Une fois initialisé au niveau de l'interruption pupitre, le DUMP s'exécute au niveau zéro pour que la prise en compte de commandes ne soit pas inhibée pendant tout le DUMP (possibilité en particulier d'aborter un DUMP). Le background sera suspendu jusqu'à la fin du DUMP.

Exemples :

```
% DU/A, @ &1200, OC
  0040
% DU/A, @ &1200, @ &1204, OC
  0040 01B0 0030 0034
% DU/A, @ &1200, LO
  1200 0040 01B0 0030 0034 0000 0000 0000 0001 0002 FF00 0000 0007 FFF0 5F2A ...
% DU/A, @ &1200, @ &1204
  1200 0040 01B0 0030 0034 0000 0000 0000 0001 0002 FF00 0000 FFF0 FFF0 5F2A ...
```

```
% DU/@1200,@1216,OC
 0000 0000 0000 0000 0000 0000 220A 5344
 F729 4242 210A
% DU/@1200,@1216
 2400 0000 0000 0000 0000 0000 0000 220A 5344 F729 4242 210A 0404 F70D 4042 ...
```

% ASSIGN commande d'assignation

• Définition

Commande modifiant les assignations des étiquettes opérationnelles aux périphériques.

• Forme

$$\left\{\begin{matrix}\%\ \text{ASSIGN}\\ \%\ \text{AS}\end{matrix}\right\} / \left\{\begin{matrix}\text{M:}O_1\,O_2\\ \text{U:}[\&]nn\end{matrix}\right\}, \text{T:}t_1t_2, \left[\text{D:}\,[\&]\,d\right], \left[\text{E:}\,[\&]\,e\right], \left[\left\{\begin{matrix}\text{BN}\\ \text{AN}\end{matrix}\right\}\right]$$

• Sens des options

$\left\{\begin{matrix}\text{M:}O_1\,O_2\\ \text{U:}[\&]nn\end{matrix}\right\}$ — Spécifie l'étiquette opérationnelle affectée. C'est une étiquette opérationnelle standard qui doit être une des neuf suivantes : M:BI, M:BO, M:CI, M:EI, M:EO, M:LO, M:LL, M:DO, M:SI.

— U:[&] nn étiquettes utilisateur (1 à 47).

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

$\text{T:}t_1t_2$ Spécifie le type du périphérique à assigner à l'étiquette opérationnelle.

Sous MTR, ce type doit être un des suivants :

T:NO	Annulation d'étiquette. Une Entrée/Sortie demandée sur l'étiquette opérationnelle ainsi assignée ne sera pas effectuée.
T:TY	Clavier téléscriptrice (Entrée et Sortie).
T:PT	Lecteur-perforateur de ruban téléscriptrice.
T:PR	Lecteur de ruban perforé rapide.
T:PP	Perforateur de ruban rapide
T:LP	Imprimante
T:CR	Lecteur de cartes
T:CP	Perforateur de cartes
T:DC	Disque
T:9T	Bandes magnétiques 9 pistes

Ne pourront être spécifiés que des types de périphérique dont le handler a été inclu à la génération (voir fiche opérateur du moniteur MTR).

$\left[\text{D:}\,[\&]\,d\right]$ [&] d est le numéro de périphérique pour un coupleur multipériphérique (par exemple disques amovibles ou bandes).

$0 \leq d \leq F$ en hexadécimal

Par défaut, [&] d vaut zéro.

[E: [&] e] [&]e est le numéro de l'unité de traitement sur laquelle est connecté le périphérique :

e = 0 Unité centrale
e ≠ 0 Numéro de l'unité d'échange

Par défaut E: [&] e vaut E:0

[{BN / AN}] Spécifie si l'Entrée/Sortie sera binaire ou alphanumérique. Ce paramètre n'est utile que pour les étiquettes opérationnelles standard M:EI et M:EO. Les autres étiquettes opérationnelles sont définitivement alphanumériques ou binaires.

L'utilisation d'une étiquette opérationnelle en mode alphanumérique permet d'assurer automatiquement toutes conversions nécessaires pour que le code interne soit le code EBCDIC quel que soit le code utilisé par le périphérique.

Exemples :

% AS/M:BI,T:PT — M:BI assurera des entrées binaires sur le lecteur de ruban de la téléscriptrice.

% ASSIGN/M:EO,T:PP,AN — M:EO assurera des sorties alphanumériques sur le perforateur de ruban rapide.

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

Tableau des étiquettes opérationnelles du MTR

Nom	Numéro binaire		Fonction	Mode
M:BI	1	&01	Entrée binaire (Binary Input)	Binaire
M:BO	2	&02	Sortie binaire (Binary Output)	Binaire
M.CI	3	&03	Entrée de commande (Command input)	Alphanumérique
M:OC	4	&04	Organe de commande (Operator Communication)	Alphanumérique
M:EI	5	&05	Entrée d'éléments (Element Input)	Binaire ou Alphanumérique
M:EO	6	&06	Sortie d'éléments (Element Output)	Binaire ou Alphanumérique
M:LO	7	&07	Sortie de liste (Listing Output)	Alphanumérique
M:LL	8	&08	Journal de bord (Listing Log)	Alphanumérique
M:DO	9	&09	Sortie de diagnostics (Diagnostic Output)	Alphanumérique
M:SI	10	&0A	Entrée de langage source (Source Input)	Alphanumérique

Etiquettes opérationnelles utilisateur U:1 à U:47
correspondant aux valeurs hexadécimales &11 à &3F (décimales : 17 à 64)

%TIME initialisation de la date

• Définition

Commande permettant d'initialiser la date

• Forme

{%TIME / %TI} / [A [&] xxx], [J [&] xx], [H [&] xxx], [M [&] xxx], [S [&] xxx]

• Sens des paramètres

[A [&] xxx] Année
Valeur par défaut : 0

[J [&] xxx] Jour dans l'année
Valeur par défaut : 0

[H [&] xxx] Heure dans le jour $0 \leqslant xxx \leqslant 24$
Valeur par défaut : 0

[M [&] xxx] Minute dans l'heure $0 \leqslant xxx \leqslant 60$
Valeur par défaut : 0

[S [&] xxx] Seconde dans la minute $0 \leqslant xxx \leqslant 60$
Valeur par défaut : 0

Exemples :

%TIME/J204,H13,M4,S10

%TI

%TI/M34

% X abandon du background et dump post-mortem

• Définition

Commande d'abandon du programme background en cours et dump du background.

• Forme

% X

• Effet

Abort du programme background en cours et abandon des Entrées/Sorties correspondantes.

Edition du message d'abort correspondant.

Dump de la zone background sur M:LO.

Remarque :

Un DUMP, executé au niveau zéro, mais demandé par une commande moniteur précédente

(% DUMP, % X) n'est pas considéré comme en background. Une commande % X sera refusée si un DUMP de ce type est en cours. Ce DUMP devra être préalablement aborté.

% Y abandon du background sans dump post-mortem

- Définition

Commande d'abandon du programme background en cours et du dump éventuel demandé par une commande précédente.

- Forme

% Y

- Effet

Abort du programme background en cours et abandon des Entrées/Sorties correspondantes.

Edition du message d'abort correspondant.

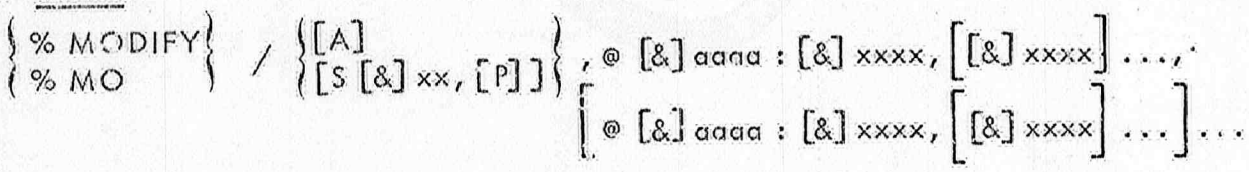
Abandon d'un dump éventuel demandé par une commande moniteur précédente (% DUMP, % X).

% MODIFY commande de modification mémoire

- Définition

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

- Forme



- Sens des paramètres

[A]	Les adresses @ [&] aaaa figurant dans la commande sont absolues
[S [&] xx, [P]]	Les adresses @ [&] aaaa figurant dans la commande sont relatives à l'une des bases de la section de numéro [&] xx. C'est la base P si le paramètre [P] est présent. C'est la base L si le paramètre [P] est absent. Si aucun des paramètres [A] ni [S [&] xx, [P]] n'est présent, les adresses @ [&] aaaa figurant dans la commande sont relatives à la base G du dernier programme chargé.
@ [&] aaaa	Adresse à partir de laquelle on désire modifier la mémoire.

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

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

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

Exemples :

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

Résultat

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

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

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

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

%DEVICE action particulière sur périphérique

• Définition

Commande de positionnement de bandes magnétiques, écriture de marque fin de fichier, sortie de ruban vierge sur perforateur de ruban.

• Forme

{ %DEVICE / %DE } / { M:O_1 O_2 / U:[&] nn } , O:aaa, [N: [&] zz]

• Sens des paramètres

{ M:O_1 O_2 / U:[&] nn } Action sur le périphérique auquel est assignée l'étiquette opérationnelle M:O_1 O_2. C'est une étiquette opérationnelle standard qui doit être une des neuf suivantes : M:BI, M:BO, M:CI, M:EI, M:EO, M:LO, M:LL, M:DO, M:SI.

U:[&] nn est une OL utilisateur comprise entre U:1 et U:47 ou U:&1 et U:&3F.

O:aaa — Action à exécuter :

O:EOD	Ecriture d'une marque de fin de fichier
O:ROW	(REWIND) Rebobinage off-line
O:REW	(REWIND) Rebobinage on-line
O:RSK	(RECORD SKIP) Saut en avant du nombre de blocs spécifié par le paramètre [N: [&] zz] .
O:RBK	(RECORD BACK) Saut en arrière du nombre de blocs spécifié par le paramètre [N: [&] zz] .

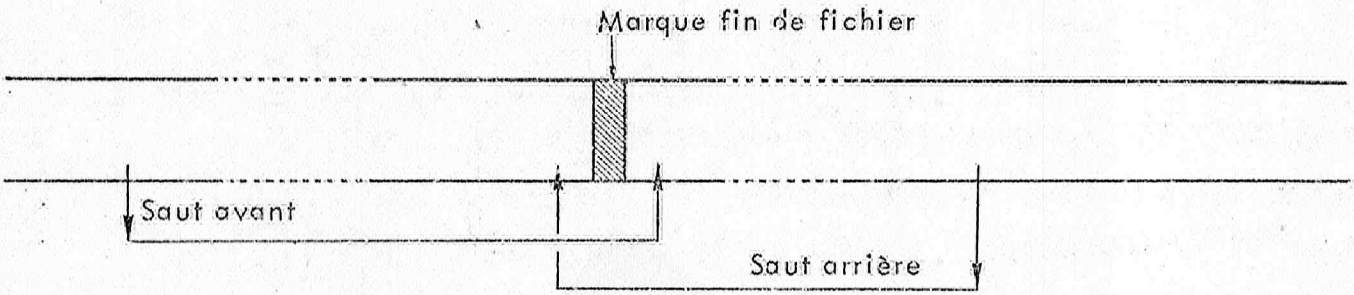
O:FSK (FILE SKIP) Saut en avant du nombre de fichiers spécifié par le paramètre [N: [&] zz] .
O:FBK (FILE BACK) Saut en arrière du nombre de fichiers spécifié par le paramètre [N: [&] zz] .
O:SPK (SPROCKET) Avance ruban perforé

[N [&] zz] Nombre de blocs ou de fichiers à sauter
Valeur par défaut : N1

• Effet

Une commande DEVICE demandée sur un périphérique pour lequel l'action demandée n'existe pas est ineffective.

Les sauts de fichier saut des sauts de marque fin de fichier exécutés selon le schéma suivant :



Exemples :

%DEVICE/M:SI, O:REW

%DEVICE/M:EO, O:F

%DEVICE/M:EI, O:RBK, N:3

%DE/M:EO, O:SPK

%DE/M:BO, O:EOD

% IT armement et/ou validation d'interruption

• Définition

Commande permettant d'armer, de désarmer, de valider ou d'invalider une interruption.

• Forme

% IT /N [&] xx, [{ A / V / D / I / T / R }]

• Sens des paramètres

N [&] xx — Spécifie le niveau de l'interruption que l'on veut manipuler.

[(A / V / D / I / T / R)]

A pour Armement
V pour Validation
D pour Désarmement
I pour Invalidation
T pour Tout (armement et validation)
R pour Rien (désarmement et invalidation)

Valeur par défaut : T (armement et validation)

Cette commande nécessite la présence physique des bascules d'armement et de validation des niveaux que l'on veut manipuler. Rappelons que la bascule d'armement est absente pour les interruptions liées aux coupleurs d'Entrée/Sortie ainsi que pour les interruptions "défaut secteur" et "opérateur" et que la bascule de validation est absente pour les interruptions liées aux coupleurs d'Entrée/Sortie, ainsi que pour les interruptions "défaut secteur".

La commande % IT ne traite pas les interruptions multiplexées.

Exemples :

% IT/N&04,T

% IT/N4

■ % ACTIVATE excitation d'une interruption

• Définition

Commande permettant l'excitation d'une interruption

• Forme

{ % ACTIVATE / % AC } /N [&] xx

• Sens des paramètres

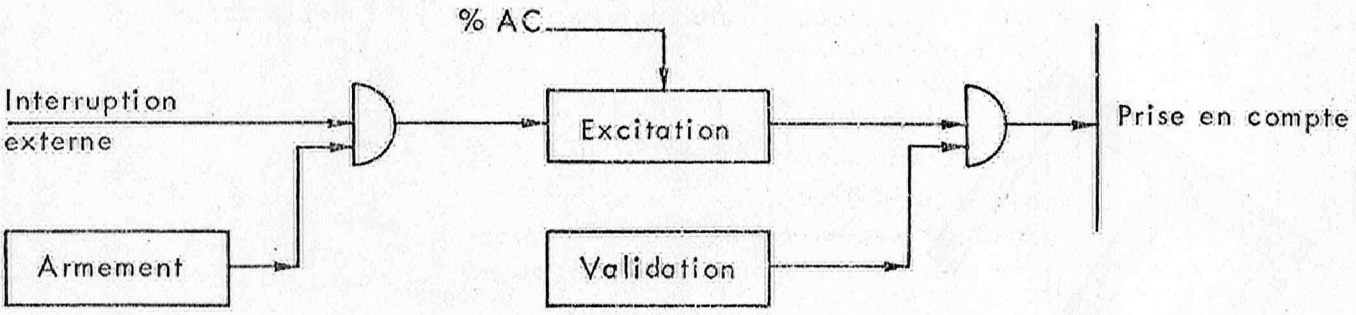
N [&] xx — Spécifie le niveau de l'interruption que l'on veut exciter.

• Effets

Cette commande nécessite la présence physique de la bascule d'excitation des niveaux que l'on veut exciter. Si le niveau n'est pas validé, l'interruption restera en attente. Le fait que le niveau soit armé ou non n'a pas d'influence. Rappelons que la bascule d'excitation (ou de déclenchement) est absente pour les interruptions "défaut secteur".

La commande % AC ne traite pas les interruptions multiplexées.

• Rappel sur la prise en compte d'une interruption



Exemples

% ACTIVATE/N&04

% AC/N04

%PM protection ou libération de zone mémoire

• Définition

Commande permettant la protection ou la libération d'une zone mémoire.

• Forme

%PM/[A], {[YES] / [NO]} , @ [&] xxxx, [@ [&] yyyy]

• Sens des paramètres

[A] Les adresses indiquées sont absolues.
Valeur par défaut : adresses relatives à la base G du dernier programme chargé.

{[YES] / [NO]} YES Protection
NO Libération
Valeur par défaut : YES

@ [&] xxxx Action sur le mot d'adresse [&] xxxx

@ [&] xxxx, [@ [&] yyyy] Action sur la zone comprise entre les deux adresses spécifiées, bornes comprises.

Exemples :

%PM/A, @&4000, @&4200

%PM/@ 200, @210

VI-2. MESSAGES OPERATEUR

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

VI-2.1. Au chargement du système

SYSTEM READY xxxxxx.y	Le système vient d'être chargé en mémoire. Il est en attente d'une interruption pupitre. xxxxxx nom du moniteur (1 à 6 caractères EBCDIC). y numéro de la version.

VI-2.2. A l'analyse des commandes

%	En attente d'entrée de commande
%% AC01	Commande inconnue
%% AC02	Erreur de syntaxe
%% AC03	Commande non autorisée
%% AC04	Argument non autorisé
%% AC05	Débordement dans un nombre
%% AC06	Numéro incorrect (niveau d'interruption, numéro de section)
%% AC07	Numéro de coupleur incorrect. Numéro de périphérique incorrect pour un coupleur multipériphérique.
%% AC08	Etiquette opérationnelle non réassignable
%% AC09	Adresse inexistante

VI-2.3. Au chargement (par LOAD)

%%LD01	Erreur d'entrée/sortie au chargement
%%LD02	Checksum (somme de contrôle) incorrecte
%%LD03	Erreur de séquencement des blocs IMT
%%LD04	Background occupé
%%LD05	Implantation incorrecte
%%LD06	Tentative de charger autre chose qu'un IMT
%%LD08	Rencontre d'une fin de fichier avant la fin normale du module IMT.

VI-2.4. A l'exécution d'une commande

%%EC01	%FOREGROUND	: Adresse foreground impossible à implanter
	%ABORT	: Essai à abort de l'interruption pupître
	%ASSIGN	: Incompatibilité entre l'étiquette opérationnelle et le périphérique physique.
	%PM	: Commande sans adresse
	%DUMP	: Débordement mémoire
	%DEVICE	: Commande incorrecte.
%%EC02	%FOREGROUND	: Agrandissement de la zone foreground impossible. Il faut au préalable aborter le niveau zéro.
	%ABORT	: Essai d'abort d'un niveau inactif
	%RUN	: Il n'y a pas de programme chargé
	%ASSIGN	: Incompatibilité entre le mode demandé (alphanumérique ou binaire) et les modes autorisés.
	%MODIFY	: Il n'y a pas de programme chargé
	%TIME	: Valeurs d'initialisation incorrectes
%%EC03	%RUN	: Tentative de connexion d'un programme background à un niveau d'interruption.
	%MODIFY	: Adresse inexistante
	%DUMP	: Erreur d'entrée/sortie sur M:LO
	%X	: Pas de programme actif au niveau zéro
	%Y	: Pas de programme actif au niveau zéro
	%IT	: Tentative de manipulation du niveau zéro
	%ACTIVATE	: Tentative de manipulation du niveau zéro
	%PM	: Adresse début > Adresse fin
%%EC04	%RUN	: Tentative de connexion d'un programme foreground au niveau zéro.
	%IT	: Tentative de manipulation d'une interruption inexistante.
	%PM	: Adresse hors limite
%%EC05	%RUN	: Tentative de connexion d'un programme à un niveau occupé. Il faut au préalable aborter ce niveau.

%%EC06	%RUN	: Section de lancemant inexistante
	%MODIFY	: Section inexistante
	%DUMP	: Il n'y a pas de programme chargé.
%%EC07	%RUN	: L ou P impossible à implanter

VI-2.5. Erreurs d'Entrée/Sortie

%%IO00	Time-out sur un périphérique ne possédant pas d'ordre STOP
%%iO01	Erreur logique détectée en fin de transfert
%%IO02	Erreur logique détectée à l'initialisation du transfert
%%IO03	Erreur physique détectée en fin de transfert
%%IO04	Erreur physique détectée à l'initialisation du transfert.

VI-2.6. Erreurs diverses

%%ER00	Processeur standard lancé de façon incorrecte
%%ER02	Ressource inexistante
%%ER03	Appel incorrect à M:FLAG et M:ADRS

VI-2.7. Messages de déroutement

- Mot d'état déroutement
- P (en absolu)
- L (en absolu)
- Indicateurs

} sur l'instruction ayant provoqué le déroutement

- P - G
- L - G
- Indicateurs

} sur le dernier appel moniteur

%% DRxx avec
- xx = 01 déroutement dû à un coupleur
- xx = 02 déroutement programme

VI-2.8. Messages d'abort

%% Ayxx

xxxx	xxxx	xxxx
↓	↓	↓
Adresse début d'implantation (absolue)	Valeur de la base G (absolue)	Première adresse libre après le programme (absolue)

avec 000 ≤ yxx ≤ 00 F niveau de 0 à 15
100 ≤ yxx ≤ 10 F niveau de 16 à 31

VI-2.9. Incidents graves

%% CS
SYS READY — Coupure secteur ayant entraîné une réinitialisation du système (voir chapitre V).

%% DR xxyy
SYS READY — Déroutement ayant entraîné une réinitialisation du système (voir chapitre VI).

avec xx : niveau de la tâche qui a déroutée
yy : contenu du mot 2 de la mémoire qui donne la nature du déroutement.

VI-2.10. Périphériques non opérationnels

%% XXYZ où

XX désigne le périphérique non opérationnel

Y désigne l'unité de traitement à laquelle est connecté le périphérique :

0 : UC
1 : UEM1
2 : UEM2
3 : UEM3

Z numéro de périphérique pour un coupleur multipériphérique ($0 \leqslant Z \leqslant F$)

Exemples :

%% LP00 — Imprimante sur UC non prête.
%% 9T10 — Dérouleur de bande 0 sur UEM1 non prêt.

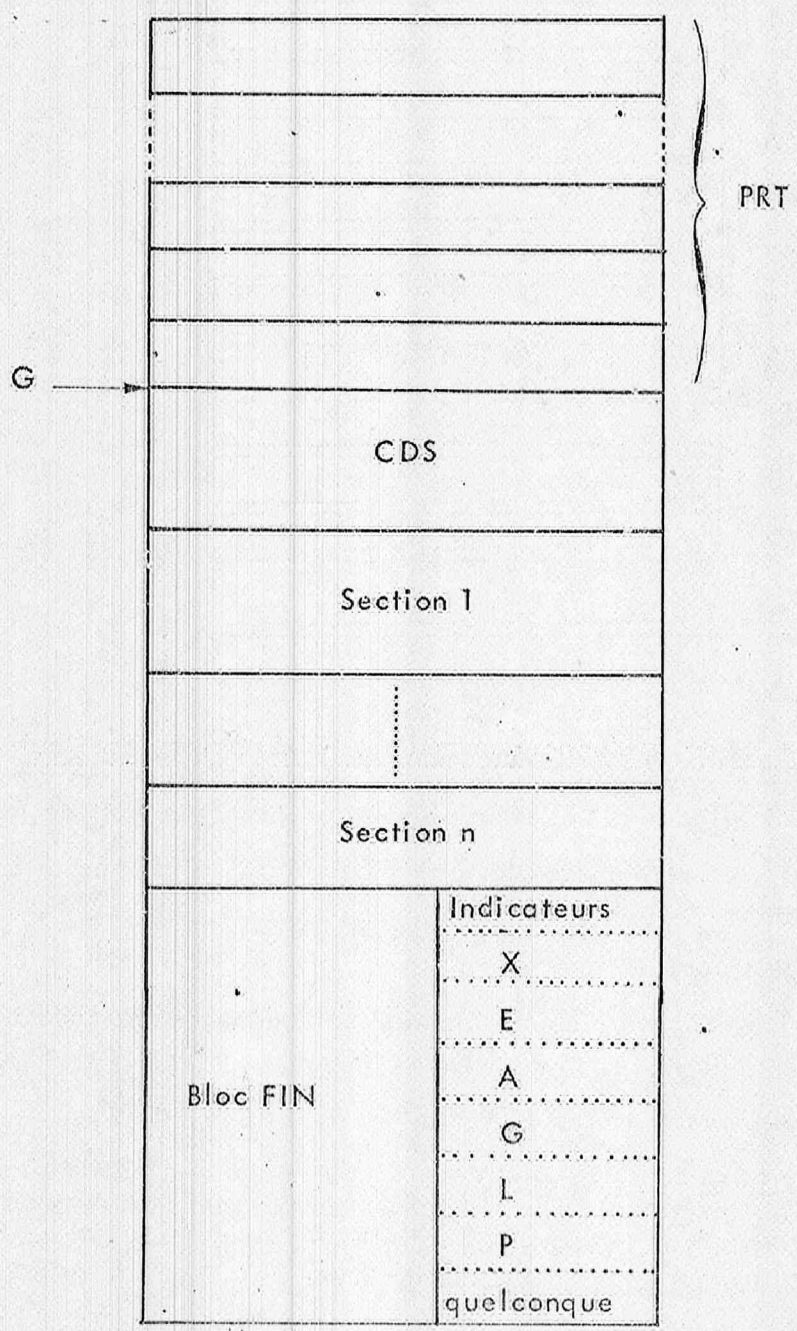
7. Utilisation des modules du moniteur

VII-1. PRINCIPES GENERAUX DES APPELS MONITEUR

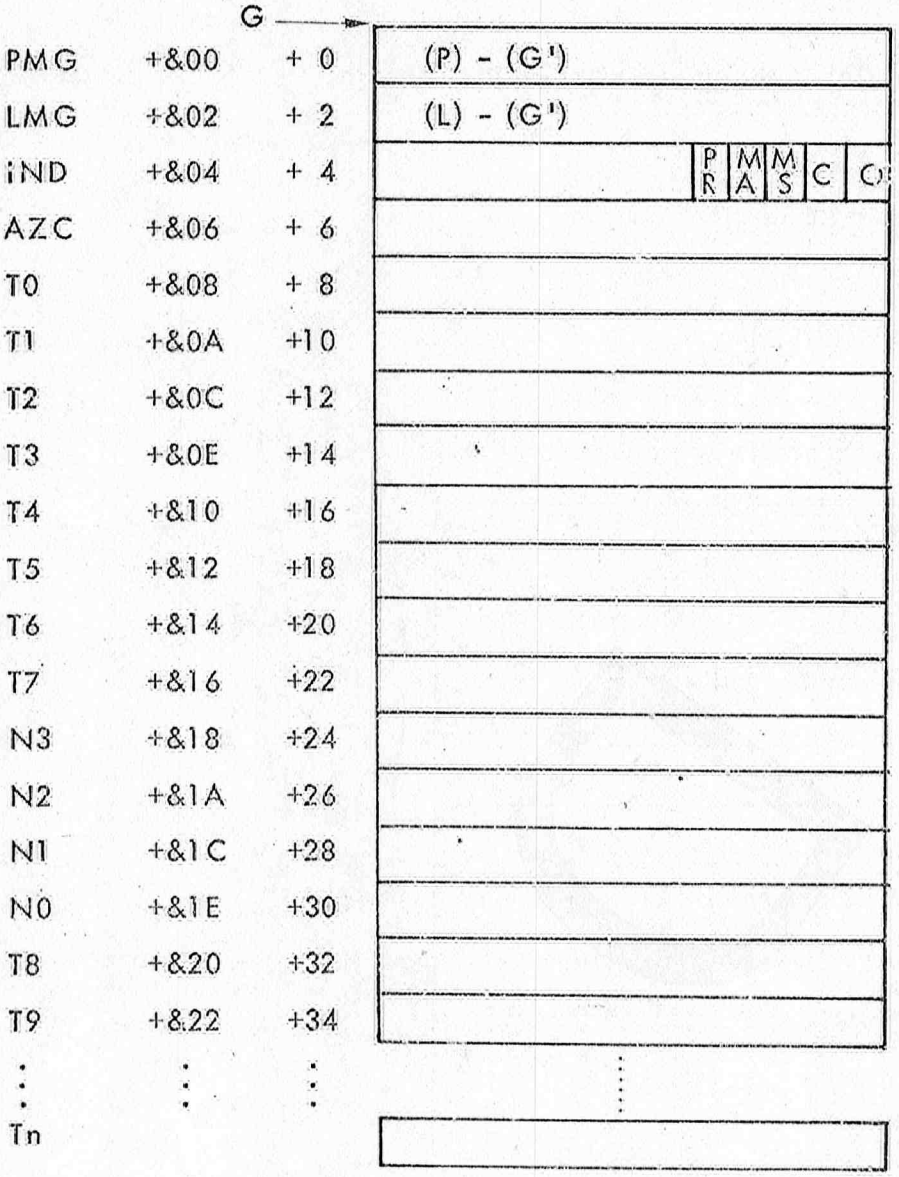
Structure d'un programme

Un programme en mémoire se compose de :

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



Description du TWB (Task Working Block)



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

Le moniteur peut utiliser dans le cas standard, les 16 premiers mots de la CDS (TWB) :

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

- comme mémoires de travail.

Les programmes qui font appel au moniteur, doivent normalement commencer leur CDS par une réservation de 16 mots.

Ces 16 mots sont désignés par le sigle TWB (Task Working Block).

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

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

■ Procédure d'appel d'un module du MTR

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

CSV M : <nom du module>

Remarque importante :

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

Exemple :

Appel d'Entrée/Sortie :

a) Constitution du CB (Control Block).

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

c) CSV M:IO (M:IO interface d'Entrée/Sortie).

Sous-programmes utilisables par appel CSV

M:IO	02	Appel d'Entrée/Sortie
M:WAIT	04	Attente fin de transfert
M:EXIT	01	Fin normale d'un programme
M:KEY		Lecture des clés - Ecriture des voyants du pupître
M:DCBN		Conversion décimal-binaire
M:HXBN		Conversion hexadécimal-binaire
M:BNDC		Conversion binaire-décimal
M:BNHX		Conversion binaire-hexadécimal
M:ASEB	09	Conversion ASCII - EBCDIC.
M:EBAS	0A	Conversion EBCDIC - ASCII
M:LOAD	0B	Chargement en mémoire d'un module IMT (TWB étendu)
M:MOVE		Déplacement d'une chaîne d'octets
M:EDIT		Edition d'une zone mémoire en hexadécimal (TWB étendu)
M:ABRT	12	Fin anormale d'une tâche
M:DUMP	14	Edition d'une zone mémoire (TWB normal)
M:IT	19	Demande d'opérations sur le système d'interruptions
M:CMPA	1B	Comparaison arithmétique de deux mots de 16 bits
M:CMPS	1C	Comparaison de deux chaînes d'octets

M:CNEC	Connexion d'un contexte à un niveau
M:ZIO	Appel d'entrée/sortie, CB en zone commune
M:ZWAT	Attente de fin d'entrée/sortie, CB en zone commune
M:ASGN	Assignation dynamique
M:DLAY	Lancement d'un délai
M:ZDLY	Lancement d'un délai, CB en zone commune
M:SDLY	Suppression d'un délai
M:ZSDL	Suppression d'un délai, CB en zone commune
M:TIME	Demande de l'heure
M:PM	Protection/Libération d'une zone
M:RQST	Réservation d'une ressource
M:RLSE	Libération d'une ressource
M:TAR	Test et réservation d'une ressource
M:KILL	Meutre d'une tâche
M:CSOL	Passage à l'état actif d'un sous-système
M:AFF	Recherche du périphérique associé à une étiquette opérationnelle

VII-2. M:IO APPEL D'ENTREE/SORTIE

- Séquence d'appel

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

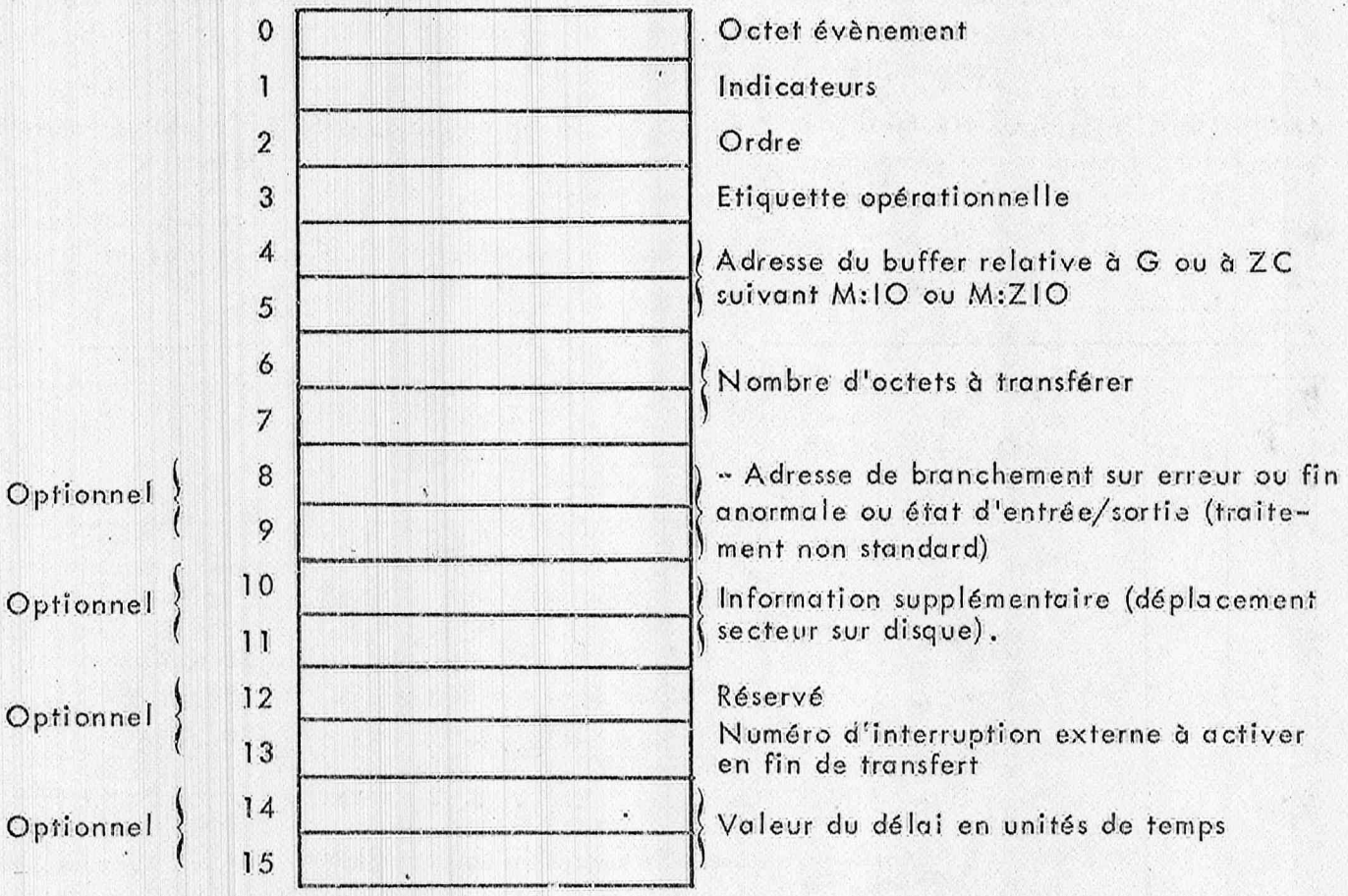
Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du bloc de commande d'Entrée/Sortie (Control Block), le CB étant dans le CDS ou un LDS du programme. Un CB doit toujours avoir une adresse-mot (adresse paire).

- Fonction

Exécution des demandes d'Entrée/Sortie (voir organisation des Entrées/Sorties sous contrôle du MTR).

- Description des paramètres du bloc de commande : CB

Lorsqu'un élément optionnel figure dans un CB, les éléments optionnels qui pourraient le précéder doivent être réservés.



• Octet 0 : octet évènement

Il permet de suivre l'évolution du transfert.

L'octet évènement (octet zéro du CB) est chargé successivement de la façon suivante :

- Par le module M:IO au moment de l'appel d'entrée/sortie :

. bit 0 = 1 Transfert en cours
. bits 1 à 7 = 0

- Par le module M:WAIT au moment de la demande de mise en attente de fin de transfert et si le bit zéro de l'octet évènement est à 1 :

. bit 0 = Inchangé
. bits 1 à 7 = rang de la tâche en attente (double du niveau)

- Par le superviseur en fin de transfert

. bit 0 = 0 Entrée/Sortie terminée

. bit 1 = { 0 Fin normale
1 Erreur ou fin anormale

. bit 2 = { 0 Erreur logique (par exemple : format d'appel incorrect)
1 Erreur physique (par exemple : signalée par le coupleur)

. bit 3 = { 0 Erreur en fin de transfert
1 Erreur à l'initialisation du transfert

Bits 4, 5, 6, 7 Code d'erreur ou de fin anormale. Ces bits permettent de définir 16 codes

pour chacune des combinaisons de bits 2 et 3. Le sens de ces bits peut être général (erreurs détectées par l'interface d'entrée/sortie) ou particulier (erreurs détectées par un handler).

Les bits 2 à 7 n'ont de sens que si le bit 1 est à 1. Cependant le code &30F permet d'indiquer à l'utilisateur que l'évènement qu'il attendait (fin de transfert) a été activé sur déclenchement d'un time-out (fin de délai d'entrée/sortie).

Les tableaux des codes rendus sont donnés dans les paragraphes d'utilisation des handlers.

- Octet 1 : indicateurs de contrôle

Rang du bit	Nom	Traité par	Valeur	Signification
0	U	Moniteur	0	Abandon du programmeur demandeur en cas d'incident irrécupérable et édition sur M:OC de : %%IO01 si erreur logique en fin de transfert %%IO02 si erreur logique à l'initialisation %%IO03 si erreur physique en fin de transfert %%IO04 si erreur physique à l'initialisation
			1	Pas d'abandon du programme demandeur même en cas d'incident irrécupérable. A partir du niveau MTR, reprise automatique de transfert pour les périphériques non opérationnels.
1	E	Moniteur Module M:WAIT	0	Poursuite en séquence en cas d'incident (sauf si abandon du programme).
			1	Branchement en cas d'incident (sauf si abandon du programme) : utilisation du mot d'adresse 8 du CB.
2	S	Handler	0	Mot d'adresse 10 du CB non traité
			1	Mot d'adresse 10 du CB traité. Exemple du handler disque : adresse disque.
3	T	Moniteur	0	Ineffectif
			1	Time-out demandé : utilisation du mot d'adresse 14 du CB.
4	I	Moniteur	0	Ineffectif
			1	Interruption demandée en fin de transfert : utilisation de l'octet d'adresse 12 du CB.
5	F			Utilisation réservée
6	C	Moniteur	0	Ineffectif
			1	Annule le traitement du bit U. Aucun traitement sur incidents, aucune reprise en cas de périphérique non opérationnel.
7		Non affecté		

• Octet 2 : ordre

Valeur (hexadécimal)		Signification
Fonctions générales	Fonctions particulières	
00		Lecture avant avec filtrage
	01	Lecture avant sans filtrage
10		Lecture avant avec format et filtrage
	11	Lecture avant avec format sans filtrage
20		Lecture arrière
30		Rebobinage on-line
40		Saut avant de n blocs
50		Saut avant de n fichiers
60		Saut arrière de n blocs
70		Saut arrière de n fichiers
80		Ecriture
	83	Ecriture sans contrôle
90		Ecriture avec format
	91	Saut de papier suivi d'écriture
	93	Ecriture avec format sans contrôle
A0		Ecriture de tape-mark
B0		Avance
C0		Non affecté
D0		Positionnement de bras
E0		Non affecté
F0		Rebobinage off-line.

Le traitement détaillé effectué sur chacun de ces ordres sera décrit avec chaque handler (voir manuel de référence, chapitre Entrées/Sorties).

Cependant le traitement de ces ordres a été prévu pour assurer une compatibilité maximum. Ainsi :

- Les fonctions particulières sont représentées par les quatre bits de poids faible de l'octet d'ordre. Lorsqu'une fonction particulière n'est pas traitée, elle est assimilée à la fonction principale correspondante (mise à zéro des quatre bits de poids faible).

- Un ordre 90 est traité comme un ordre 80 quand le handler n'attribue pas de sens au "format".

- Un ordre 10 est traité comme un ordre 00 quand le handler n'attribue pas de sens au "format".

- Les ordres sont classés par chaque handler en :

. ordres acceptés : entraînant un transfert physique
. ordres refusés : incompatible avec le type de périphérique (exemple : lecture sur imprimante).
. ordres ineffectifs : n'entraînant pas de transfert physique, mais pouvant nécessiter une intervention de l'utilisateur (exemple : rebobinage sur lecteur de cartes).

. ordres transparents : n'entraînant pas de transfert physique, et ne nécessiant aucune intervention de l'utilisateur (exemple : rebobinage sur imprimante).

• Octet 3 : étiquette opérationnelle

Il contient l'équivalent numérique de l'étiquette opérationnelle (voir commande %ASSIGN).

• Octets 4 et 5 : adresse tampon

Adresse de l'information à transférer de/vers la mémoire contrôle.

Cette adresse sera exprimée relativement à la base G pour les CB correspondant à des appels à M:IO et M/WAIT. Elle pourra être initialisée dynamiquement (instruction LEA) ou statiquement. En mode maître, où les références sont normalement rendues absolues au chargement, une initialisation statique devra se faire :

- en assembleur, en utilisant une directive DATA et en faisant précéder l'étiquette nommant le tampon du caractère "#".

- en LP15, en initialisant le mot avec le nom du tampon précédé du caractère ".".

Cette adresse sera exprimée relativement à l'adresse de la zone commune pour les CB correspondant à des appels à M:ZIO et M:ZWAT. L'adresse de la zone commune est disponible dynamiquement à l'adresse 6 du programme après chargement.

• Octets 6 et 7 : taille tampon

Nombre d'octets à transférer ou bien nombre de blocs ou de fichiers à sauter.

• Octets 8 et 9 (optionnels)

F = 0 et E = 1 : Adresse de branchement en cas d'erreur ou de fin anormale (toujours /G de l'appelant).

F = 1 : Bits d'état fournis par le coupleur (si le bit F est traité par le handler concerné).

• Octets 10 et 11 (optionnels) : information supplémentaire

Information supplémentaire demandée par certains handlers si S = 1. Pour un disque on spécifie ici un déplacement au début du zone disque (voir les paragraphes décrivant l'utilisation des handlers disque).

• Octets 12 et 13 (optionnels) : niveau d'interruption (à partir du niveau MTR)

Octet 13 Niveau de l'interruption à activer en fin de transfert

Octet 12 Réservé

• Octets 14 et 15 (optionnels) : délai (à partir du niveau MTR)

Bit 0 { = 0 Echelle 1 (unité de temps = 100 ms)
= 1 Echelle 2 (unité de temps = 10 s)

Bits 1 à 15 Valeur du délai en unités de temps : Echelle 1 100 ms≤t<2300 s
Echelle 2 10 s≤t≤320000 s

. Contrôle des transferts

On dire qu'il y a fin de transfert lorsque :

- un transfert s'est déroulé jusqu'au bout, correctement ou non, et la fin physique du transfert a été normalement prise en compte sur l'interruption de couplage.

- une erreur a été relevée par le résident, soit à l'initialisation (logique ou physique) soit au cours du transfert. L'erreur peut être détectée.

- par le coupleur auquel cas il excède une interruption de couplage et arrête ce transfert.

- par le superviseur sur examen des caractères reçus.

Le transfert est entièrement contrôlé par le résident qui renvoie à l'utilisateur dans l'octet évènement du CB un code d'anomalie normalisé, facile à interpréter et peut provoquer l'abandon du programme demandeur.

- Si C = 0 et U = 1, le superviseur rend toujours le contrôle à l'utilisateur quelle que soit l'anomalie qu'il a pu relever à l'exception des cas de reprise automatique.

Il y a reprise automatique sur les cas de périphérique non opérationnels (codes rendus dans l'octet évènement supérieurs ou égaux à &7B (hexadécimal)).

- Si C = 0 et U = 0, le superviseur abandonne le programme demandeur dans le cas d'incidents dits irrécupérables (voir paragraphes d'utilisation des handlers).

- Si C = 1 et U quelconque, le superviseur rend toujours le contrôle à l'utilisateur quelle que soit l'anomalie relevée, sans tentative de reprise.

Exemple de sortie sur M:LO

```
EXEC    CDS
        RES     16
CB      DATA    0
        DATA,1  &80
        DATA,1  M:LO
        DATA    # BUFF
        DATA    13
BUFF    TEXT    "SORTIE SUR LO"
        FIN
LOC     LDS
        RES     2
        FIN
```

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

        END     PROG
```

- Eléments communiqués en sortie

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

Les registres E et X sont quelconques

L'octet évènement contient les informations suivantes :

. Erreur (ou non) de transfert,

. Code d'erreur le cas échéant

- Mémoires de TWB utilisées : T0 à N0

- Modules appelés

M:EBAS, Handlers 1, Handlers 2

VII-3. M:WAIT ATTENTE D'ACTIVATION D'EVENEMENT

- Séquence d'appel

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

Au moment de l'appel (CSV) le registre A doit contenir l'adresse relative à G du bloc de commande d'entrée/sortie ou de délai que l'on veut contrôler.

- Fonction

Attente d'activation d'évènement :

- Evènement désactivé (transfert ou délai en cours)

. rangement du rang de la tâche appelante dans l'octet zéro du CB

. si le niveau de l'appelant est nul, attente; si le niveau de l'appelant n'est pas nul, désactivation temporaire de celui-ci (réactivation lors de l'activation d'évènement : fin de transfert, fin de délai, libération des ressources).

- Evènement activé (transfert ou délai terminé)

. reprise du contrôle par l'appelant et dans le cas d'un CB d'entrée/sortie comportant le traitement utilisateur des incidents, branchement éventuel à une adresse de reprise.

C'est le module M:WAIT qui assure le fonctionnement évolué du MTR en permettant la désactivation des programmes en attente et la prise en compte de niveaux moins prioritaires.

• Eléments communiqués en sortie

A<0 en cas d'erreur d'Entrée/Sortie

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

A≥0 aucune erreur.

• Mémoires de TWB utilisées : T0 à N0

VII-4. M:EXIT FIN DE PROGRAMME

• Séquence d'appel

```
CSV    M:EXIT
```

• Fonction

Fin de programme.

- Si le programme est au niveau zéro le contrôle est rendu au moniteur qui se met en attente d'interruption pupître (boucle d'attente au niveau 0 : BRU $). Les Entrées/Sorties en cours ou en attente sont abortées.

- Si le programme n'est pas au niveau zéro, son niveau est désactivé, ce qui permet de prendre en compte tout niveau inférieur. Les Entrées/Sorties en cours ou en attente sur le niveau considéré ne sont pas abortées.

• Mémoires de TWB utilisées : T0 à N0

VII-5. M:KEY CLES ET VOYANTS

• Séquence d'appel

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

CSV    M:KEY
```

• Fonction

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

• Eléments communiqués en sortie

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

• Mémoires de TWB utilisées : Néant

VII-6. M:DCBN CONVERSION DECIMAL-BINAIRE

• Séquence d'appel

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

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

• Fonction

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

• Eléments communiqués en sortie

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

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

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

• Mémoires de TWB utilisées : T0 à N0

VII-7. M:HXBN CONVERSION HEXADECIMAL-BINAIRE

• Séquence d'appel

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

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

• Fonction

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

&0≤n≤&FFFF

• Eléments communiqués en sortie

E et T1	Résultat de la conversion (sur un mot)
X et T0	Adresse relative à G du séparateur de fin de chaîne (sur un mot)
A	≥0 Nombre de caractères de la chaîne s'il n'y a pas eu débordement
	<0 S'il y a eu débordement

• Mémoires de TWB utilisées : T0 à N0

VII-8. M:BNDC CONVERSION BINAIRE-DECIMAL

• Séquence d'appel

LDE	NOMBRE
LEA	CHAINE
CSV	M:BNDC

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

• Fonction

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

• Eléments communiqués en sortie

A, E et X	quelconques
T0	Adresse relative à G du début de la chaîne.

• Mémoires de TWB utilisées : T0 à N0

VII-9. M:BNHX CONVERSION BINAIRE-HEXADECIMAL

• Séquence d'appel

LDE	NOMBRE
LEA	CHAINE
CSV	M:BNHX

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

• Fonction

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

• Eléments communiqués en sortie

A, E et X	quelconques
A	Caractère EBCDIC de plus fort poids
X	&FFFF
T0	Adresse relative à G de la chaîne

• Mémoires de TWB utilisées : T0 à N0

VII-10. M:ASEB CONVERSION ASCII-EBCDIC

• Séquence d'appel

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

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

• Fonction

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

Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	Symbole	Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	Symbole
00 ou 00	00	NUL	1C ou 9C	1C	FS
01 ou 81	01	SOH	1D ou 1D	1D	GS
02 ou 82	02	STX	1E ou 1E	1E	RS
03 ou 03	03	ETX	1F ou 9F	1F	US
04 ou 84	37	EOT	20 ou A0	40	SP
05 ou 05	2D	ENQ	21 ou 21	4F	!
06 ou 06	2E	ACK	22 ou 22	7F	"
07 ou 87	2F	BEL	23 ou A3	7B	#
08 ou 88	16	BS	24 ou 24	5B	$
09 ou 09	05	HT	25 ou A5	6C	%
0A ou 0A	15	LF	26 ou A6	50	&
0B ou 8B	0B	VT	27 ou 27	7D	'
0C ou 0C	0C	FF	28 ou 28	4D	(
0D ou 8D	0D	CR (Retour chariot)	29 ou A9	5D	)
0E ou 8E	0E	SO	2A ou AA	5C	*
0F ou 0F	0F	SI	2B ou 2B	4E	+
10 ou 90	10	DLE	2C ou AC	6B	,
11 ou 11	11	DC1	2D ou 2D	60	-
12 ou 12	12	DC2	2E ou 2E	4B	.
13 ou 93	13	DC3	2F ou AF	61	/
14 ou 14	3C	DC4	30 ou 30	F0	0
15 ou 95	3D	NAK	31 ou B1	F1	1
16 ou 96	32	SYN	32 ou B2	F2	2
17 ou 17	26	ETB	33 ou 33	F3	3
18 ou 18	18	CAN	34 ou B4	F4	4
19 ou 99	19	EM	35 ou 35	F5	5
1A ou 9A	3F	SUB	36 ou 36	F6	6
1B ou 1B	27	ESC	37 ou B7	F7	7

Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	Symbole	Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	Symbole
38 ou B8	F8	8	54 ou D4	E3	T
39 ou B9	F9	9	55 ou 55	E4	U
3A ou 3A	7A	:	56 ou 56	E5	V
3B ou BB	5E	;	57 ou D7	E6	N
3C ou 3C	4C	<	58 ou D8	E7	X
3D ou BD	7E	=	59 ou 59	E8	Y
3E ou BE	6E	>	5A ou 5A	E9	Z
3F ou 3F	6F	?	5B ou DB	4A	[
40 ou C0	7C	@	5C ou 5C	E0	\
41 ou 41	C1	A	5D ou DD	5A	]
42 ou 42	C2	B	5E ou DE	5F	^
43 ou C3	C3	C	5F ou 5F	6D	–
44 ou 44	C4	D	60 ou 60	0A	
45 ou C5	C5	E	61 ou E1	0B	VT
46 ou C6	C6	F	62 ou E2	0C	FF
47 ou 47	C7	G	63 ou 63	CD	
48 ou 48	C8	H	64 ou E4	CE	
49 ou C9	C9	I	65 ou 65	CF	
4A ou CA	D1	J	66 ou 66	86	f
4B ou 4B	D2	K	67 ou E7	87	g
4C ou CC	D3	L	68 ou E8	88	h
4D ou 4D	D4	M	69 ou 69	89	i
4E ou 4E	D5	N	6A ou 6A	91	j
4F ou CF	D6	O	6B ou EB	92	k
50 ou 50	D7	P	6C ou 6C	93	l
51 ou D1	D8	Q	6D ou ED	94	m
52 ou D2	D9	R	6E ou EE	95	n
53 ou 53	E2	S	6F ou 6F	96	o

Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	symbole	Code ASCII hexadécimal (bit zéro à zéro)	Code EBCDIC correspondant (hexadécimal)	symbole
70 ou F0	97	p	78 ou 78	A7	x
71 ou 71	98	q	79 ou F9	A8	y
72 ou 72	99	r	7A ou FA	A9	z
73 ou F3	A2	s	7B ou 7B	C0	{
74 ou 74	A3	t	7C ou FC	6A	|
75 ou F5	A4	u	7D ou 7D	D0	}
76 ou F6	A5	v	7E ou 7E	A1	~
77 ou 77	A6	w	7F ou FF	07	DEL

- Eléments communiqués en sortie

A	zéro
E	Non modifié
X	Zéro
T0	Adresse relative à G au début de la chaîne
T1	Zéro

- Mémoires de TWB utilisées : T0 à N0

VII-11. M:EBAS CONVERSION EBCDIC-ASCII

- Séquence d'appel

LDX	Nombre de caractères de la chaîne
LEA	CHAINE
CSV	M:EBAS

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

- Fonction

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

• Eléments communiqués en sortie

A, E et X quelconques.

• Mémoires de TWB utilisées : T0 à N0

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

• Séquence d'appel

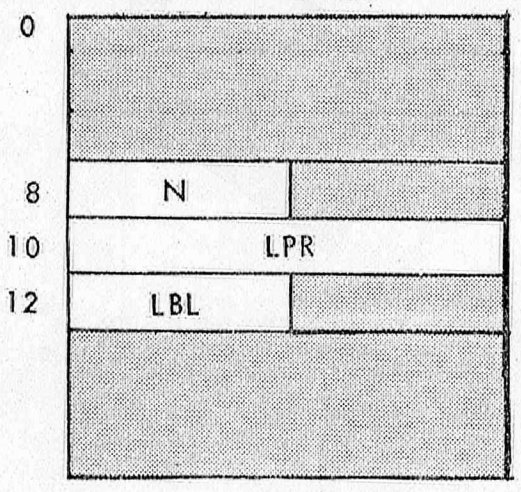
- Lecture de l'en-tête (premier enregistrement)
- Mise en place des paramètres dans le TWB
- CSV M:LOAD

• Fonction

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

- Edition sur M:OC de l'implantation et si demandé de la PRT du programme chargé (voir commande % LOAD).

• Informations de l'en-tête à exploiter



N = 1 programme / 3 données

LPR Longueur du programme y compris la PRT et le contexte

LBL Longueur des enregistrements (120).

• Eléments à communiquer en entrée

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

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

G + 20	T8	CB	INDI
G + 22	T9	Ordre	Etiquette
G + 2A	T13	0	
G + 2C	T14	ADTAMP	
G + 2E	T15	LBL	
G + 30	T16	0	SORPRT
G + 32	T17	N	
G + 38	T20	ADPR	
G + 3A	T21	LPR	

hexadécimal

CB	= 0	Octet évènement
INDI	= 0	Format IMT non disque
Ordre	= 0	Lecture
Etiquette	=	Etiquette opérationnelle (normalement 1 pour M:BI)
ADTAMP	=	Adresse du tampon de lecture par rapport à G de l'appelant
LBL	= 120	Longueur enregistrement (à prendre dans l'en-tête)
SORPRT	bit 7 = 0	Pas d'édition de la PRT
	bit 7 = 1	Edition de la PRT
	bit 6 = 0	Programme non protégé
	bit 6 = 1	Programme protégé
	bit 5 = 0	Pas d'édition des implantations
	bit 5 = 1	Edition des implantations
N	= 1	Programme (à prendre dans l'en-tête)
	= 3	Données (à prendre dans l'en-tête)
ADPR	=	Adresse d'implantation du programme par rapport à G de l'appelant.
LPR	=	Longueur du programme en octets y compris la PRT et le contexte (à prendre dans l'en-tête).

• Eléments communiqués en sortie

Les éléments de TWB utilisés en sortie sont :

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

hexadécimal

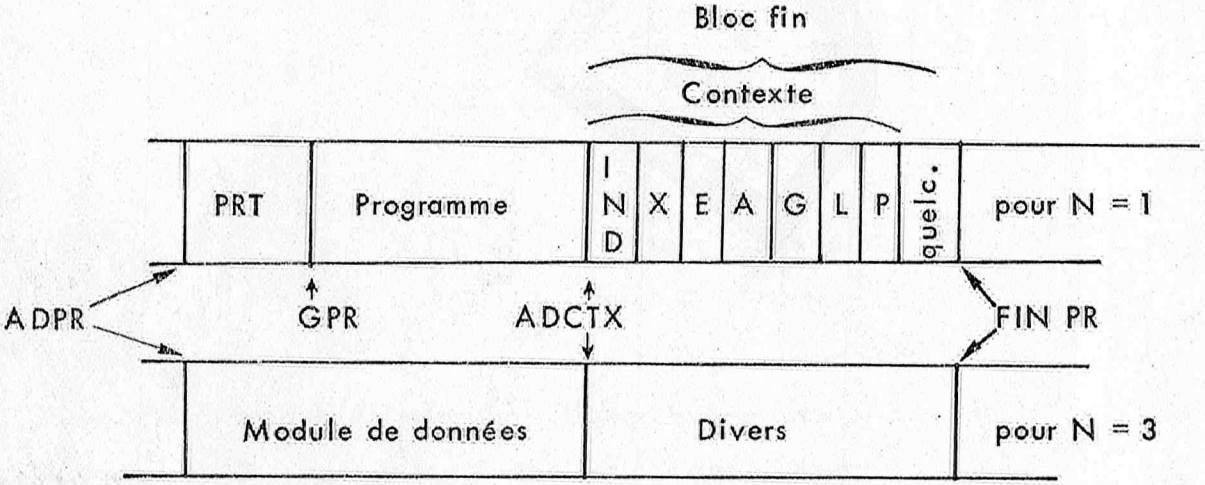
ADPR = Adresse absolue d'implantation

GPR = Base G du programme chargé (en absolu) (pour N = 1)

Longueur de l'IMT chargé pour N = 3 (bloc de données).

FIN PR = Première adresse libre après le module chargé (en absolu)

ADCTX = Adresse relative à la base G de l'appelant du contexte du programme chargé.



Accumulateur : A< 0 Incident en cours de chargement

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

Après chargement d'un module de programme, l'utilisateur le connectera à un niveau par appel au module M:CNEC.

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

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

• Editions sur M:OC

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

PRT ——→ absente pour un module de données

ADPR

GPR ——→ égale à ADPR pour un module de données

FIN PR

• Mémoires de TWB utilisées

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

• Modules appelés

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

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

• Séquence d'appel

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

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

• Fonction

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

• Eléments communiqués en sortie

A — Adresse relative à G de la chaîne émettrice.

E — Adresse relative à G de la chaîne réceptrice.

• Mémoires de TWB utilisées : T0 à N0

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

• Séquence d'appel

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

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse par rapport à G de la zone mémoire à éditer, le registre E doit contenir le nombre d'octets mémoire à éditer et le registre X doit contenir l'adresse par rapport à G du tampon utilisateur nécessaire pour l'édition. Ce tampon sera au moins de 5 n + 1 octets, modulo 41, si l'on désire éditer n octets mémoire par ligne.

• Fonction

Editer en hexadécimal une zone mémoire. Cette édition est effectuée mot par mot à raison de huit mots par ligne sur M:OC (téléscriptrice).

Une erreur d'Entrée/Sortie entraîne l'abandon du module.

Exemple :

Edition demandée de 11 mots :

▽▽xxxx▽xxxx▽xxxx▽xxxx▽xxxx▽xxxx▽xxxx▽xxxx

▽▽xxxx▽xxxx▽xxxx

(▽signifie "espace")

• Mémoires de TWB utilisées

TWB étendue à 56 octets, soit : T0 à T19

• Modules appelés

M:IO, M:WAIT, M:BNHX

VII-15. M:ABRT FIN ANORMALE D'UNE TACHE

• Séquence d'appel

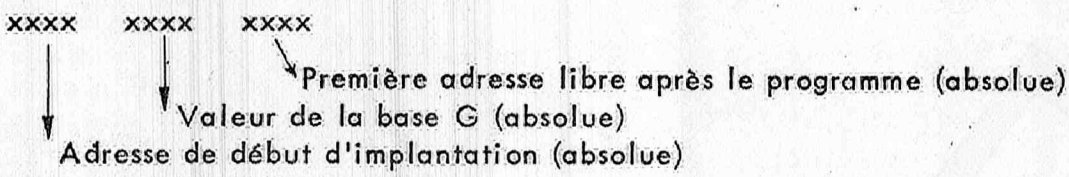
```
CSV  M:ABRT
```

• Fonction

Abandon du programme faisant appel à M:ABRT par connexion à un contexte d'attente :

- DIT BRU $ - 1 si le niveau est différent de zéro
- BRU $ si le niveau est zéro

Dans tous les cas, édition sur M:OC (téléscriptrice) des informations suivantes sur le programme aborté :



%% Ayxx

où yxx spécifie le niveau aborté

000 à 00F pour les 16 premiers niveaux

100 à 10F pour les 16 derniers niveaux

Un programme aborté est éliminé logiquement du système et ses Entrées/Sorties en cours ou en file d'attente abandonnées.

• Mémoires de TWB utilisées

TWB étendu à 56 octets soit T0 à T19

• Modules appelés

M:EDIT, M:MOVE

VII-16. M:DUMP EDITION D'UNE ZONE MEMOIRE

• Séquence d'appel

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

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

• Fonction

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

Le nombre de mots édités par ligne est fixé à la génération. Du point de vue standard, seuls les moniteurs comportant le handler imprimante utiliseront le format de 16 mots par ligne.

• Bloc de commande utilisateur

```
CB    DATA,1     0
      DATA,1     0
      DATA,1     &80
      DATA,1     M:xx      (Etiquette opérationnelle utilisée. En cas d'utilisation de
                           MITRAS 1, c'est le numéro binaire lui-même, qui devra être
                           spécifié).
      DATA       #TAMPON   Adresse du tampon utilisateur
      DATA       54        Taille minimum du tampon fourni par l'utilisateur
```

• Format d'impression (exemple du cas à huit mots par ligne)

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

avec aaa0 ⩽ ADEBUT ⩽ aaaF

zzz0 ⩽ AFIN ⩽ zzzF

• Eléments communiqués en sortie

A<O Erreur d'entrée/sortie
A⩾O Pas d'erreur

E et X quelconques

• Mémoires de TWB utilisées

T0 à N0 (TWB standard), CB et tampon utilisateur

• Modules appelés

M:BNHX, M:IO, M:WAIT

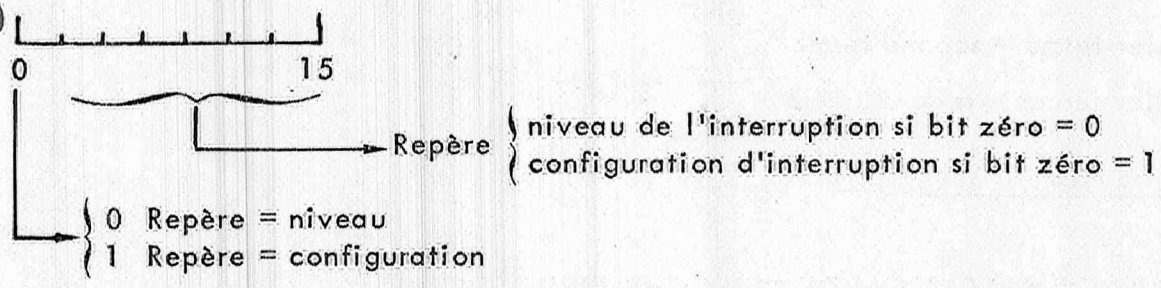
VII-17. M:IT DEMANDE D'OPERATIONS SUR LE SYSTEME D'INTERRUPTION

• Séquence d'appel

```
LDA     = REPERE
LDX     = CODE
CSV     M:IT
```

Au moment de l'exécution du CSV, le registre A doit contenir le repère de l'interruption à manipuler, ce repère étant son niveau ou sa configuration suivant la valeur du bit zéro de A; le registre X doit contenir le code qui spécifie l'opération. Les remarques faites pour les commandes %IT et %AC sont valables pour l'utilisation du module M:IT.

• Détail de A (Repère)



• Détail de X (code)

X =
- 0 Armer
- 1 Désarmer
- 2 Valider
- 3 Invalider
- 4 Armer et valider
- 5 Désarmer et invalider
- 6 Exciter
- 7 Acquiter

• Eléments communiqués en sortie

Accumulateur

A<0 erreur dans la demande

A>0 demande correcte et effectuée

• Mémoires de travail utilisées : T0 à N0

VII-18. M:CMPA COMPARAISON ARITHMETIQUE DE DEUX MOTS DE 16 BITS

• Séquence d'appel

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

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

• Fonction

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

• Eléments communiqués en sortie

A < 0 Premier terme < second terme

A = 0 Premier terme = second terme

A > 0 Premier terme > second terme

• Mémoires de travail utilisées : T0 à N0

VII-19. M:CMPS COMPARAISON DE DEUX CHAINES D'OCTETS

• Séquence d'appel

```
LEA     CHAIN2
XAE
LEA     CHAIN1
LDX     =LONG
CSV     M:CMPS
```

Au moment du CSV, le registre A doit contenir l'adresse relative à G du premier octet de la chaîne 1, le registre E doit contenir l'adresse relative à G du premier octet de la chaîne 2 et le registre X le nombre d'octets à comparer.

• Fonction

La comparaison se fait en considérant chacune des deux chaînes comme un nombre, le premier octet de la chaîne en contenant donc les poids forts.

• Eléments communiqués en sortie

A < 0 Chaîne 1 < chaîne 2

A = 0 Chaîne 1 = chaîne 2

A > 0 Chaîne 1 > chaîne 2

• Mémoires de travail utilisées : T0 à N0

VII-20. M:CNEC CONNEXION D'UN CONTEXTE A UN NIVEAU

• Séquence d'appel

```
LDA     ADCTX     Adresse du contexte relative au G de l'appelant
LDX     = R       Niveau d'interruption
CSV     M:CNEC
```

Au moment du CSV, le registre A doit contenir l'adresse (par rapport à la base G du programme appelant) du contexte à connecter, et le registre X doit contenir le niveau d'interruption auquel on désire connecter le contexte.

• Fonction

Connecter un contexte à un niveau de façon à ce qu'il soit activé par l'arrivée d'une interruption sur ce niveau. Le module M:CNEC s'utilise normalement après le module M:LOAD.

• Mémoires de TWB utilisées : T0 à N0

VII-21. M:ZIO DEMANDE D'ENTREE/SORTIE EN ZONE COMMUNE

• Séquence d'appel

```
LDA     ACBZC     (Adresse bloc de commande)
CSV     M:ZIO
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative au début de la zone commune du bloc de commande d'Entrée/Sortie.

• Fonction

Exécution des demandes d'Entrée/Sortie (voir organisation des Entrées/Sorties sous contrôle du MTR).

• Description du CB (Bloc de Commande)

Le CB utilisé, situé en zone commune, est identique au CB utilisé par M:IO à ceci prêt que l'adresse du tampon utilisateur devra être exprimée relativement au début de la zone commune (et non plus par rapport à G). L'adresse de branchement en cas d'erreurs contrôlées par l'utilisateur (éventuellement spécifiée) restera relative à G.

• Eléments communiqués en sortie

Ils sont identiques à ceux communiqués par M:IO, à ceci près que A contient l'adresse du CB relative à la zone commune.

• Mémoires de TWB utilisées : T0 à N0

• Modules appelés : M:IO

VII-22. M:ZWAT ATTENTE DE FIN DE TRANSFERT EN ZONE COMMUNE

• Séquence d'appel

```
LDA     ACBZC     (Adresse bloc de commande)
CSV     M:ZWAT
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative au début de la zone commune du bloc de commande de l'Entrée/Sortie que l'on veut contrôler.

• Fonction

Elle est identique à celle du module M:WAIT

• Eléments communiqués en sortie

Ils sont identiques à ceux communiqués par M:WAIT

• Mémoires de TWB utilisées : T0 à N0

• Modules appelés : M:WAIT

VII-23. M:ASGN ASSIGNATION DYNAMIQUE

• Séquence d'appel

```
LEA     CTB     (Table de contrôle)
CSV     M:ASGN
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G d'une table de contrôle. La table de contrôle doit avoir une adresse paire et contenir les informations suivantes :

CTB	OL	MODE
CTB + 2	NOM	
CTB + 4	Zéro	D
CTB + 6	UEM	Zéro

OL : Numéro de l'étiquette opérationnelle à assigner

MODE : 0 0 0 1 0 0

- 0 BN
- 1 AN
- 0 Etiquette background
- 1 Etiquette foreground (seulement pour MTRE)

NOM : Nom du périphérique en EBCDIC (2 caractères)

D : Numéro du périphérique
Si le coupleur est monopériphérique, mettre zéro.

UEM : Numéro de l'unité de traitement (0 si unité centrale)

• Fonction

Traitement identique à celui d'une commande %ASSIGN. L'étiquette opérationnelle M:OC n'est pas assignable.

• Eléments communiqués en sortie

X
- 0 : Assignation correcte
- 1 : Tableau CTB erroné
- 2 : Assignation interdite

• Modules appelés : Néant

VII-24. M:DLAY LANCEMENT D'UN DELAI

• Séquence d'appel

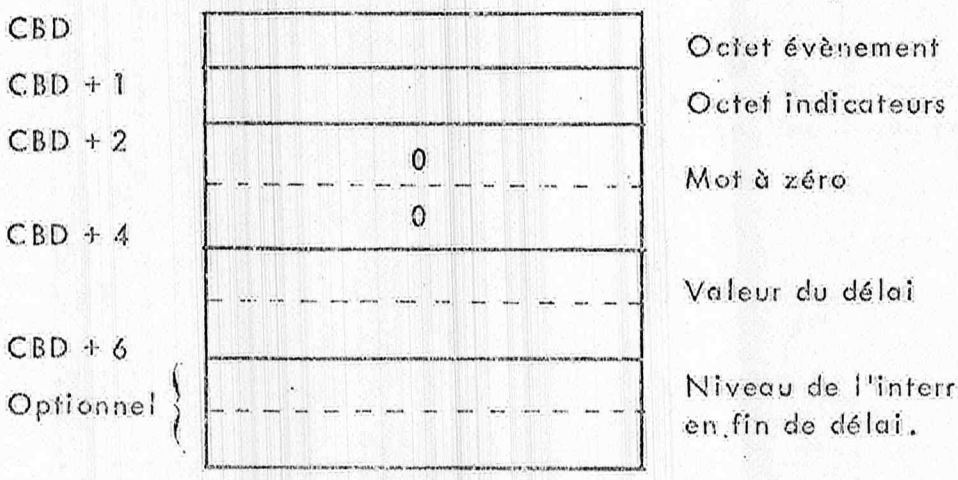
```
LEA     CBD
CSV     M:DLAY
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du bloc de commande de délai (Control Block of Delay), le CBD étant dans la CDS ou un LDS du programme. Un CBD doit toujours avoir une adresse mot (adresse paire).

• Fonction

Exécution de la demande de délai (voir chapitre IV-10).

• Description du CBD



- Octet 0 : octet évènement

Bit 0
- = 1 Un délai associé à un M:DLAY utilisant ce CBD est en cours. L'évènement (fin de délai) est activé.
- = 0 Le délai est terminé. L'évènement (fin de délai) est activé.

Bits 1 à 7 Rang du programme désactivé en attente d'évènement.

- Octet 1 : indicateurs

Bits 0 à 2 Zéro

Bit 3
- = 1 Délai demandé
- = 0 Délai non demandé

} Bit T

Bit 4
- = 1 Interruption demandée en fin de délai
- = 0 Pas d'interruption demandée

} Bit I

Bits 5 à 7 Zéro

- Octets 2 et 3 : zéro

- Octets 4 et 5 : valeur du délai

Bit 0
- = 0 Echelle 1 (unité de temps = 100 ms)
- = 1 Echelle 2 (unité de temps = 10 s)

Bits 1 à 15 Valeur du délai en unités de temps

Echelle 1 100 ms $\leq$ délai $\leq$ 2300 s
Echelle 2 10 s $\leq$ délai $\leq$ 320000 s (3 jours 16 heures)

- Octet 6 : réservé

- Octet 7 : niveau de l'interruption à activer en fin de délai

• Eléments communiqués en sortie

A Adresse du CBD relative à G

X
- 0 Délai pris en compte
- 1 Délai non pris en compte

Un délai n'est pas pris en compte lorsque :

- le bit T est à zéro dans l'octet indicateurs
- la valeur du délai est hors des bornes autorisées
- la table des délais est saturée.

• Mémoires de TWB utilisées : T0 à N0

• Modules appelés : M:ZDLY

VII-25. M:ZDLY LANCEMENT D'UN DELAI, CB EN ZONE COMMUNE

- Séquence d'appel

```
LDA     ACBZC   (Adresse bloc de commande)
CSV     M:ZDLY
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative au début de la zone commune du bloc de commande de délai.

- Fonction

Exécution de la demande de délai (voir chapitre IV-10).

- Description du CB (Bloc de commande de délai)

Le CB utilisé en zone commune est identique au CB utilisé par M:DLAY

- Eléments communiqués en sortie

Ils sont identiques à ceux communiqués par M:DLAY à ceci près que A contient l'adresse du CB relative à la zone commune.

- Mémoires de TWB utilisées : T0 à N0

VII-26. M:SDLY SUPPRESSION DE DELAI

- Séquence d'appel

```
LEA     CB
CSV     M:SDLY
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à C du bloc de commande de délai à supprimer (demandé par M:DLAY).

- Fonction

Suppression de l'évènement associé au CB spécifié.

- Eléments communiqués en sortie

A { 0 Suppression effectuée
  { <0 Demande incorrecte

- Mémoires de TWB utilisées : T0 à N0

- Modules appelés : M:ZSDL

VII-27. M:ZSDL SUPPRESSION D'UN DELAI, CB EN ZONE COMMUNE

- Séquence d'appel

```
LDA     ACBZC   (Adresse bloc de commande)
CSV     M:ZSDL
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative au début de la zone commune du bloc de commande du délai à supprimer (demandé par M:ZDLY).

- Fonction

Suppression de l'évènement associé au CB spécifié.

- Eléments communiqués en sortie

A { 0 Suppression effectuée
  { <0 Demande incorrecte

- Mémoires de TWB utilisées : T0 à N0

VII-28. M:TIME DEMANDE DE L'HEURE

- Séquence d'appel

```
CSV     M:TIME
```

Fonction

Demande de l'heure. Cette heure dépend de l'initialisation faite par la commande %TIME.

- Eléments communiqués en sortie

T0 = Année
X = Jour dans l'année
E = Heure dans le jour
A = Instant dans l'heure exprimé en multiples de 100 ms.

VII-29. M:PM PROTECTION/LIBERATION D'UNE ZONE MEMOIRE

- Séquence d'appel

```
LDE     AFIN
LDA     ADEB
LDX     CODE
CSV     M:PM
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse du début de la zone dont on veut manipuler le bit de protection, le registre E doit contenir l'adresse de la fin de cette zone et le registre X le code indiquant la nature de la manipulation et par rapport à quoi sont spécifiées les adresses.

Détail de X (code)

Bit 15 { 0 Adresses relatives à G
1 Adresses relatives au début de la zone commune

Bit 14 { 0 Libération
1 Protection

Eléments communiqués en sortie

A ⩾ 0 Accepté

A < 0 Refusé (Appel par un programme au niveau zéro Adresse début > Adresse fin)

Mémoires de TWB utilisées

T0 à N0

Modules appelés

M:CMPA

VII-30. M : RQST — Réservation d'une ressource

Séquence d'appel

```
LDA     NRES
CSV     M : RQST
```

Au moment de l'exécution du CSV, le registre A doit contenir le numéro de la ressource que l'on veut réserver : les ressources utilisateur commencent au numéro 10. Les ressources 1 à 9 étant réservées au Système.

Fonction

— La réservation d'une ressource transmet la propriété de la ressource à la tâche réservante.

— Si la tâche possède déjà la ressource, la demande de réservation est passante et ineffective.

— Si la ressource n'est pas libre, la tâche est mise en file d'attente.

— Lors de l'abort d'une tâche possèdant une ressource, cette ressource sera libérée ; de même lors d'un EXIT (niveau 0).

Elément communiqué en sortie

A non significatif.

Mémoires de TWB utilisées

T0 à N0

VII-31. M : RLSE — Libération d'une ressource

Séquence d'appel

```
LDA     NRES
CSV     M : RLSE
```

Au moment de l'exécution du CSV, le registre A doit contenir le numéro de la ressource à libérer. Les ressources utilisateur commencent au numéro 10, les ressources 1 à 9 étant réservées au Système.

Fonction

— Libération d'une ressource, réservée jusqu'alors par une tâche.

— La libération d'une ressource non réservée est ineffective.

Elément communiqué en sortie

A < 0 la ressource n'a jamais été réservée.

A ⩾ 0 la libération s'est exécutée.

Mémoires de TWB utilisées

T0 à N0

VII-32. M : TAR — Test de l'état d'une ressource et réservation

Séquence d'appel

```
LDA   NRES
CSV   M : TAR
```

Au moment de l'exécution du CSV, le registre A doit contenir le numéro de la ressource dont on veut tester l'état.

Fonction

— Test de l'état de la ressource (libre ou occupée).

— Réservation si la ressource est libre.

Elément communiqué en sortie

A < 0 ressource occupée.

A = 0 réservation effective.

Mémoires de TWB utilisées

T0 à N0

VII-33. M : KILL — Meurtre d'une tâche

Séquence d'appel

```
LDA   NIV
CSV   M : KILL
```

Au moment de l'exécution du CSV, le registre A doit contenir le niveau de la tâche que l'on veut tuer.

Fonction

— Ce module force la tâche dont le niveau est spécifié dans le registre A, à se suicider.

— Cette élimination de la tâche s'accompagne d'un nettoyage des tables systèmes en ce qui concerne les entrées/sorties, les délais et les ressources que possèdait éventuellement, la tâche tuée.

Eléments communiqués en sortie

A, E, X } non significatifs

Mémoires de TWB utilisées

T0 à T19 soit une TWB étendue à 56 octets.

VII-34. M : CSOL — Rendre actif un sous-système

Séquence d'appel

```
LEA    CB
CSV    M : CSOL
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse par rapport à G de l'appelant du CB utilisé pour cette action.

Format du CB

0	0
2	0
4	Ad Buf
6	CO
8	X X

AD.BUF : adresse/G du buffer dans lequel le sous-système désire récupérer les commandes opérateur qui lui sont destinées.

CO : nombre de caractères qui lui seront transmis.

XX : code reconnaissance du sous-système (caractère EBCDIC).

Fonction

— Cet appel permet d'avertir la tâche IT pupitre que le sous-système (SS) est prêt à recevoir des commandes opérateur.

— Tant qu'un SS n'as pas effectué un CSV M : CSOL, l'IT pupitre le considérera au repos et refusera toute commande MTR-E %*/).

— Après le CSV M : CSOL, le SS pourra se mettre en attente d'une commande par un WAIT ou un WLST sur son CB.

Remarque : Le nom des SS sont mis dans la table des noms de SS (dans la LDS de l'IT pupitre) à la génération.

Eléments communiqués en sortie

A = 0	demande acceptée
A < 0	demande erronée — SS inconnu SS déjà connecté sur un autre niveau
X	adresse du buffer dans la ZC destiné au sous système appelant.

Mémoires de TWB utilisées

T0 à N0

VII-35. M : AFF — Obtention du périphérique auquel est connecté une étiquette opérationnelle donnée

Séquence d'appel

```
LDA    NETI
CSV    M : AFF
```

Au moment de l'exécution du CSV, le registre A doit contenir le numéro de l'étiquette opérationnelle pour laquelle on veut connaitre l'affectatation.

Fonction

Pour une étiquette opérationnelle, ce module donne son affectation, c'est-à-dire le nom EBCDIC du périphérique qui lui est connecté, à un instant donné ainsi que son élément d'OLT.

Eléments communiqués en sortie

A nom EBCDIC du périphérique

E élément de OLT

X = 0 demande effectuée

≠ 0 demande erronée, étiquette opérationnelle inexistante

Remarque : Le niveau de l'appelant permet le choix entre les étiquettes background et les étiquettes temps réel.

Mémoires de TWB utilisées

T0 à N0

VII-36. M : ACTV — activation d'un événement (CB/G)

Séquence d'appel

```
LEA   CB
CSV   M : ACTV
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse relative à G du CB dont on veut activer le bit événement.

Fonction

— Activation du bit événement du CB (bit 0 de l'octet 0).

— S'il y a une tâche en attente, réveil de cette tâche.

— Activation d'une éventuelle IT secondaire si le bit 1 de l'octet 1 est présent.

— Transfert d'un code d'erreur dans l'octet événement.

Eléments communiqués en sortie

A = 0 demande satisfaite

A < 0 événement déjà activé ou IT secondaire non valide

Mémoires de TWB utilisées

T0 à N0

VII-37. M : ZACT — activation d'un événement (CB/ZC)

Séquence d'appel

```
LEA   ZCB
CSV   M : ZACT
```

Au moment de l'exécution du CSV, le registre A doit contenir l'adresse par rapport à ZC du CB dont on veut activer le bit événement.

Le traitement est identique à M : ACTV.

8.MTR-E : Communication opérateur modules moniteur

VIII-1. GENERALITES

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

En standard, cette extension est disponible d'un seul tenant, mais compte tenu de la modularité du software MITRA 15, des extensions (ou réductions) partielles sont possibles au moment de la génération du moniteur.

Les commandes possédant de nouvelles options par rapport à la version de base et les nouvelles commandes seront décrites complètement dans ce chapitre. Le tableau des messages opérateur sera complètement repris.

VIII-2. COMMUNICATIONS OPERATEUR - LES COMMANDES

VIII-2.1. Principe

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

Lorsqu'une interruption pupître est prise en compte, MTR édite sur M:OC le caractère % et connecte la téléscriptrice en entrée d'une commande. Sur l'entrée de "Retour-chariot", la commande est analysée et exécutée au niveau de l'interruption pupître (excepté le DUMP qui s'effectue au niveau zéro).

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

Les commandes reconnues par MTR-E sont listées ci-dessous. Les commandes soulignées sont complètement décrites dans ce chapitre. Ce sont soit celles qui n'existent pas sous MTR, soit celles qui possèdent des options supplémentaires sous MTR-E.

% LOAD	Chargement d'un IMT
% RUN	Lancement du dernier programme chargé
% ABORT	Abandon d'un programme
% FOREGROUND	Déplacement de la limite du Foreground
% DISPLAY	Edition des éléments système
% DUMP	Edition de mémoire
% ASSIGN	Assignation des étiquettes opérationnelles
% DEVICE	Action particulière sur un périphérique
% TIME	Initialisation de la date
% MODIFY	Modification mémoire
% X	Abandon du background et dump post-mortem

% Y	Abandon du niveau zéro sans dump post-mortem
% IT	Armement et/ou validation d'interruption
% ACTIVATE	Excitation d'une interruption
% PM	Protection-déprotection
% TRACE	Edition des registres (en cours de programme)
% SNAP	Edition de mémoire (en cours de programme)
% HALT	Arrêt sur instruction
% NEXT	Exécution d'un pas
% EXECUTE	Reprise du programme utilisateur
% PERFORM	Calcul en hexadécimal
% */	Commande de sous-système

VIII-2.2. Normes utilisées pour les commandes

Ces normes sont identiques à celles de la version de base (voir chapitre IV).

VIII-2.3. %RUN commande de lancement

- Définition

Commande lançant l'exécution du dernier programme chargé.

- Forme

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

- Sens des paramètres

[N [&] xx] Si ce paramètre est présent, [&] xx indique le niveau d'interruption auquel sera connecté le programme. Le programme sera effectivement lancé quand le niveau correspondant sera actif (quand l'interruption correspondante sera activée si le niveau n'est pas nul ou si le niveau est nul, quand plus aucune interruption ne sera active).

Si ce paramètre est absent, le niveau zéro est pris par défaut.

[S [&] xx] Ce paramètre précise par rapport à quoi sont exprimées les valeurs spécifiées dans les paramètres [&] xxxx] et [P [&] xxxx].

Si ce paramètre est présent, la valeur éventuellement indiquée par le paramètre [&] xxxx] est exprimée relativement à la base P de la section de numéro [&] xx et la valeur éventuellement indiquée par le paramètre [L [&] xxxx] est exprimée relativement à la base L de la section de numéro [&] xx. Ce sont les éléments de PRT de la section de numéro [&] xx qui sont utilisés à cet effet.

Si ce paramètre est absent, les valeurs éventuellement spécifiées dans les paramètres [L [&] xxxx] et [P [&] xxxx] sont exprimées relativement à la base G du dernier programme chargé.

[L [&] xxxx]

Si ce paramètre est présent, le registre de base locale est chargé avec la valeur spécifiée. Deux cas sont possibles :

- Paramètre [S [&] xx] présent : la valeur est exprimée relativement à la base L de la section de numéro [&] xx (élément de PRT).

- Paramètre [S [&] xx] absent : la valeur est exprimée relativement à G.

Si ce paramètre est absent, deux cas sont possibles :

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

- Paramètre [S [&] xx] absent : le registre de base locale est chargé avec la valeur courante (figurant dans le contexte du programme). Si le programme est lancé pour la première fois, cette valeur est la base L de la section initiale. Si le programme n'est pas lancé pour la première fois, il s'agit de la valeur de la base L au moment où le programme a été interrompu.

[P [&] xxxx]

Si ce paramètre est présent, le registre programme est chargé avec la valeur spécifiée. Deux cas sont possibles :

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

- Paramètre [S [&] xx] absent : la valeur est exprimée relativement à G.

Si ce paramètre est absent, deux cas sont possibles :

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

- Paramètre [S [&] xx] absent : le registre programme est chargé avec la valeur courante (figurant dans le contexte du programme). Si le programme est lancé pour la première fois, cette valeur est la base L de la section initiale. Si le programme n'est pas lancé pour la première fois, il s'agit de la base L au moment où le programme a été interrompu.

[A [&] xxxx]

Si ce paramètre est présent, le registre A est chargé avec la valeur spécifiée.

Si ce parapètre est absent, le registre A est chargé avec la valeur suivante :

- bits 0 à 7 : niveau de la tâche appelante (ici, niveau de l'interruption pupître).

- bits 8 à 15 : niveau de la tâche appelée.

[E [&] xxxx]

Si ce paramètre est présent, le registre E est chargé avec la valeur spécifiée.

Si ce paramètre est absent, le registre E est chargé avec zéro.

[X [&] xxxx] Si ce paramètre est présent, le registre X est chargé avec la valeur spécifiée.

Si ce paramètre est absent, le registre X est chargé avec l'adresse relative au début de la zone commune moniteur (ZC) de la zone commune background ou de la zone commune foreground suivant que le programme est lancé au niveau zéro ou non.

• Rappel

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

Exemples :

% R

% RUN

% RUN/N&00

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

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

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

% RUN/N4

% RUN/N&F

% R/S01

% R/N4,S&0A

VIII-2.4. %ASSIGN commande d'assignation

• Définition

Commande modifiant les assignations des étiquettes opérationnelles aux périphériques.

• Forme

{% ASSIGN / % AS} / [F], {M:O1 O2 / U: [&] nn}, T:t1 t2, [D: [&] d], [E: [&] e], [{BN / AN}]

• Sens des paramètres

[F] Si ce paramètre est présent, l'étiquette opérationnelle assignée appartient au foreground.

Si ce paramètre est absent, l'étiquette opérationnelle assignée appartient au background.

{M:O1 O2 / U: [&] nn} Spécifie l'étiquette opérationnelle affectée.
M:O1 O2 est une étiquette opérationnelle standard qui doit être une des neuf suivantes :

M:BI, M:BO, M:CI, M:EI, M:EO, M:LO, M:LL, M:DO, M:SI

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

U: [&] nn est une étiquette opérationnelle utilisateur comprise entre U:1 et U:47 ou bien U:&1 et U:&3F

T:t_1 t_2

Spécifie le type du périphérique à assigner à l'étiquette opérationnelle. Sous MTRE, ce type doit être un des suivants :

T:NO	Annulation d'étiquette. Une Entrée/Sortie demandée sur l'étiquette opérationnelle ainsi assignée ne sera pas effectuée.
T:TY	Clavier téléscriptrice (Entrée et Sortie).
T:PT	Lecteur-perforateur de ruban téléscriptrice.
T:PR	Lecteur de ruban perforé rapide.
T:PP	Perforateur de ruban rapide.
T:LP	Imprimante.
T:CR	Lecteur de cartes.
T:CP	Perforateur de cartes.
T:DC	Disque fixe
T:9T	Bandes magnétiques 9 pistes.

Ne pourront être spécifiés que des types de périphérique dont le handler a été inclu à la génération (voir fiche opérateur du moniteur MTR-E).

[D: [&] d]

[&] d est le numéro de périphérique pour un coupleur multipériphérique.

$0 \leq d \leq F$ en hexadécimal

Par défaut, [&] d vaut zéro.

[E: [&] e]

[&] e est le numéro de l'unité de traitement sur laquelle est connecté le périphérique :

e = 0 Unité centrale
e ≠ 0 Numéro de l'unité d'échange

Par défaut E: [&] e vaut E:0

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

Remarques :

L'utilisation d'une étiquette opérationnelle en mode alphanumérique permet d'assurer automatiquement toutes conversions nécessaires pour que le code interne soit le code EBCDIC quel que soit le code utilisé par le périphérique.

Il existe deux jeux d'étiquettes opérationnelles :

- les étiquettes opérationnelles foreground,
- les étiquettes opérationnelles background.

Il existe ainsi par exemple deux étiquettes opérationnelles M:LO. Chacune de ces deux étiquettes peut être assignée à un périphérique différent. C'est l'option [F] qui permet de préciser laquelle des deux on manipule.

Ce double jeu d'étiquettes opérationnelles permet de dissocier complètement l'exploitation des Entrées/Sorties background de celle des Entrées/Sorties foreground. C'est le niveau du programme appelant M:IO (d'après R8) qui indique si l'Entrée/Sortie demandée est foreground ou background.

Ainsi, le chargement d'un programme par la commande % LOAD s'effectuant au niveau de l'interruption pupître sur M:BI, c'est l'étiquette opérationnelle M:BI foreground qui sera utilisée. Egalement l'action effectuée par la commande %DEVICE utilisera une étiquette opérationnelle foreground.

Exemples :

% AS/M:BI,T:PT	M:BI assurera des entrées binaires sur le lecteur de ruban de la téléscriptrice pour les programmes background.
% ASSIGN/M:EO,T:PP,AN	M:EO assurera des sorties alphanumériques sur le perforateur de ruban rapide pour les programmes background.
% AS/M:EI,T:9T,D:4,BN	M:EI assurera des entrées binaires sur le dérouleur de bandes magnétiques numéro quatre pour les programmes background.
% AS/U:F3,T:CR,AN,F	U:F3 assurera des entrées alphanumériques sur le lecteur de cartes pour les programmes background.
% AS/F,U:F1,T:LP,C:2,AN	U:F1 assurera des sorties alphanumériques sur l'imprimante numéro 2 pour les programmes foreground.
% ASSIGN/F,M:EO,T:9T,D:1,AN	M:EO assurera des sorties alphanumériques sur le dérouleur de bandes magnétiques numéro 1 pour les programmes foreground.

Tableau des étiquettes opérationnelles du MTR étendu

Nom	Numéro binaire		Fonction	Mode
M:BI	1	&01	Entrée binaire (Binary Input)	Binaire
M:BO	2	&02	Sortie binaire (Binary Output)	Binaire
M:CI	3	&03	Entrée de commande (Command Input)	Alphanumérique
M:OC	4	&04	Organe de commande (Operator Communication)	Alphanumérique
M:EI	5	&05	Entrée d'éléments (Element Input)	Binaire ou Alphanumérique
M:EO	6	&06	Sortie d'élément (Element Output)	Binaire ou Alphanumérique
M:LO	7	&07	Sortie de liste (Listing Output)	Alphanumérique
M:LL	8	&08	Journal de bord (Listing Log)	Alphanumérique
M:DO	9	&09	Sortie de diagnostics (Diagnostic Output)	Alphanumérique
M:SI	10	&0A	Entrée de langage source (Source Input)	Alphanumérique

Etiquettes opérationnelles utilisateur :

U:1 à U:47 correspondant aux valeurs hexadécimales &11 à &3F, décimales 17 à 64

%DEVICE action particulière sur périphérique

• Définition

Commande de positionnement de bandes magnétiques, écriture de marque fin de fichier, sortie de ruban vierge sur perforateur de ruban.

• Forme

$$\left\{\begin{matrix}\%DEVICE\\ \%DE\end{matrix}\right\} / \left\{\begin{matrix}M:O_1\ O_2\\ U:\ [\&]\ nn\end{matrix}\right\},\ O:aaa,\ \left[N:\ [\&]\ zz\right]$$

• Sens des paramètres

M:O_1 O_2 — Action sur le périphérique auquel est assignée l'étiquette opérationnelle foreground M:O_1 O_2. C'est une étiquette opérationnelle standard qui doit être une des neuf suivantes : M:BI, M:BO, M:CI, M:EI, M:EO, M:LO, M:LL, M:DO, M:SI.

U: [&] nn — Action sur le périphérique auquel est assignée l'étiquette opérationnelle foreground U: & nn.

O:aaa — Action à exécuter :

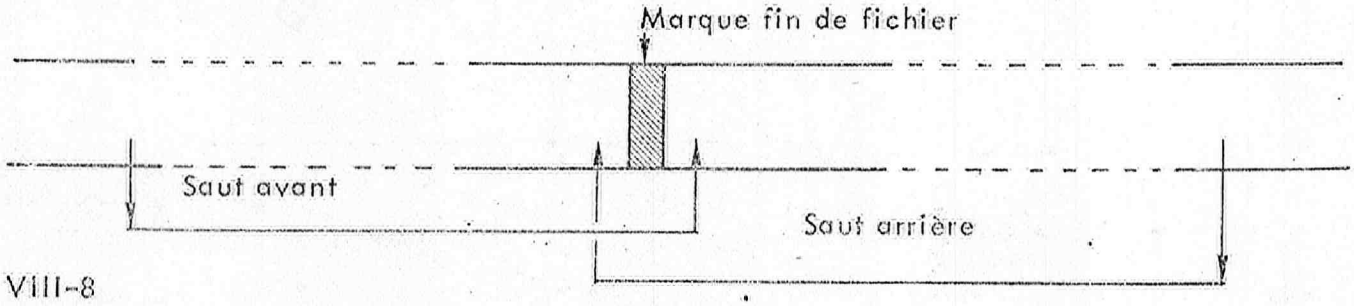
O:EOD	Ecriture d'une marque fin de fichier
O:REW	(REWIND) rebobinage on-line
O:RSK	(RECORD SKIP) Saut en avant du nombre de blocs spécifié par le paramètre [N: [&] zz] .
O:RBK	(RECORD BACK) Saut en arrière du nombre de blocs spécifié par le paramètre [N: [&] zz] .
O:FSK	(FILE SKIP) Saut en avant du nombre de fichiers spécifié par le paramètre [N: [&] zz] .
O:FBK	(FILE BACK) Saut en arrière du nombre de fichiers spécifié par le paramètre [N: [&] zz] .
O:SPK	(SPROCKET) Avance ruban perforé.
O:ROW	(REWIND) rebobinage off-line

[N: [&] zz] — Nombre de blocs ou de fichiers à sauter
Valeur par défaut : N1

• Effet

Une commande DEVICE demandée sur un périphérique pour lequel l'action demandée n'existe pas est ineffective.

Les sauts de fichier sont des sauts de marque fin de fichier exécutés selon le schéma suivant :



Exemples :

%DEVICE/M:SI, O:REW

%DEVICE/M:EO, O:F

%DEVICE/M:EI, O:RBK, N:3

%DE/M:EO, O:SPK

%DE/M:BO, O:EOD

VIII-2.5 % TRACE commande d'édition dynamique des registres

• Définition

Edition dynamique sur M:LO des six premiers registres, des indicateurs et de l'instruction courante (après exécution de celle-ci). Toute nouvelle commande % TRACE annule la précédente.

• Forme

$$\left\{\begin{matrix}\%\ \text{TRACE}\\ \%\ \text{TR}\end{matrix}\right\} / \left\{\begin{matrix}[A]\\ [S\ [\&]\ xx]\end{matrix}\right\}\ ,@\ [\&]\ xxxx,\ \left[@\ [\&]\ yyyy\right]$$

• Sens des paramètres

[A] — Si ce paramètre est présent, les adresses figurant dans la commande seront absolues.

[S [&] xx] — Si ce paramètre est présent, les adresses figurant dans la commande seront relatives à la base P de la section de numéro [&] xx du dernier programme chargé. (Elément de PRT).

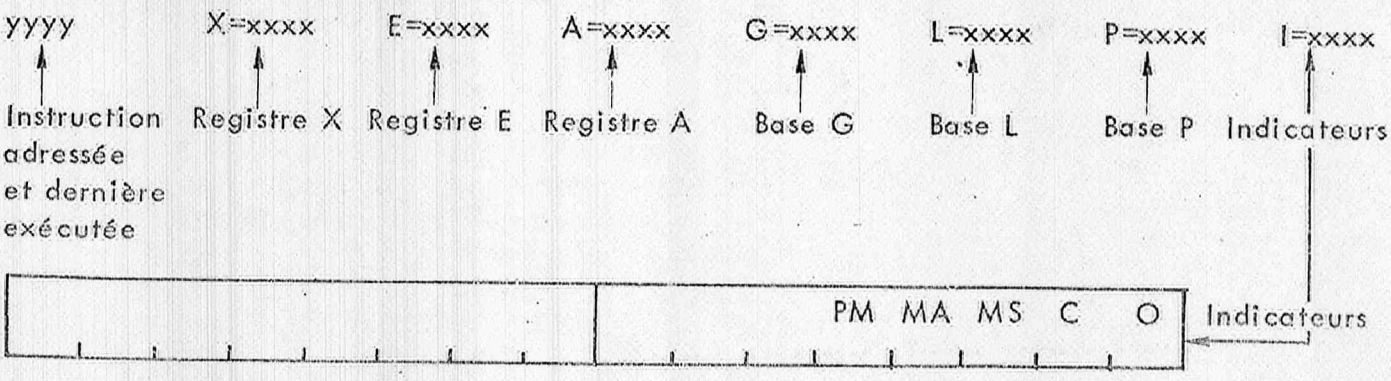
Si aucun des deux paramètres [A] ou [S [&] xx] n'est présent, toutes les adresses figurant dans la commande sont relatives à la base G du dernier programme chargé.

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

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

Si ce paramètre est absent, seule l'instruction d'adresse [@ [&] xxxx] sera tracée.

• Format de l'édition (hexadécimal)



Remarques :

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

On ne peut interrompre une TRACE en cours d'exécution pour spécifier une nouvelle commande % TRACE.

Si aucune TRACE n'est en cours, une nouvelle commande % TRACE annule et remplace la précédente.

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

- après chargement, avant le lancement
- après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

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

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

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

Exemples :

% TRACE/A, @ &2000

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

% TR/@ &2000,A

VIII-2.6.%SNAP commande d'édition dynamique de la mémoire

• Définition

Edition dynamique sur M:LO en hexadécimal d'une zone mémoire suivant le format du DUMP.

Edition de la dernière instruction exécutée et des registres avant l'édition mémoire suivant le format du TRACE.

• Forme

{% SNAP / % SN} / {[A] / [S [&] xx]} , @ [&] zzzz : [{[M] / [A] / [S [&] xx], [P]} ,[@ [&] xxxx] , [@ [&] yyyy]]

• Sens des paramètres

Les paramètres de la commande % SNAP se divisent en deux groupes séparés par ":". Le premier est relatif à l'adresse programme au passage de laquelle sera effectuée l'édition, le deuxième est relatif à la zone mémoire à éditer.

Premier groupe : adresse programme de l'instruction après laquelle sera effectuée l'édition mémoire

[A] — Si ce paramètre est présent, l'adresse programme @ [&] zzzz sera indiquée en absolu.

[S [&] xx] — Si ce paramètre est présent, l'adresse programme @ [&] zzzz sera relative à la base P de la section de numéro [&] xx (élément de PRT) du dernier programme chargé.

Si aucun des deux paramètres [A] ou [S [&] xx] n'est présente, l'adresse programme @ [&] zzzz sera relative à la base G du dernier programme chargé.

@ [&] zzzz — Spécifie l'adresse de l'instruction au passage de laquelle l'édition mémoire sera exécutée.

Cette adresse sera absolue, relative à la base P d'une section ou de la base G du dernier programme chargé suivant les paramètres [A] et [S [&] xx]

Deuxième groupe : zone mémoire à éditer

[M] — Edition de toute la mémoire.

[A] — Si ce paramètre est présent, les adresses éventuelles [@ [&] xxxx] et [@ [&] yyyy] de la zone mémoire à éditer seront en absolu.

[S [&] xx, [P]] — Les adresses éventuelles [@ [&] xxxx] et [@ [&] yyyy] de la zone mémoire à éditer seront relatives à l'une des bases de la section de numéro [&] xx du dernier programme chargé (élément de PRT).

Ce sera la base P si le paramètre [P] est présente.

Ce sera la base L si le paramètre [P] est absente.

Si aucun des paramètres [A] ni [S [&] xx, [P]] n'est présente, les adresses éventuelles [@ [&] xxxx] et [@ [&] yyyy] de la zone mémoire à éditer seront relatives à la base G du dernier programme chargé.

[@ [&] xxxx] — Edition d'une ligne de dump incluant le mot d'adresse [@ [&] xxxx]

[@ [&] xxxx], [@ [&] yyyy] — Edition d'un nombre entier de lignes de dump incluant la zone comprise entre l'adresse [@ [&] xxxx] et [@ [&] yyyy]

Si la zone à éditer n'est pas spécifiée (absence des paramètres [M], [@ [&] xxx] et [@ [&] yyyy]), il y aura édition de la zone mémoire comprise entre les adresses début et fin d'implantation du dernier programme chargé.

Remarques :

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

On ne peut interrompre un SNAP en cours d'exécution pour spécifier une nouvelle commande % SNAP.

Si aucun SNAP n'est en cours, une nouvelle commande % SNAP annule et remplace la précédente.

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

- après chargement, avant lancement
- après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

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

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

Exemples :

% SNAP/S01, @ &24	Dump à l'adresse &24 (ou 36) du LPS de la section 01 du dernier programme chargé.
% SN/A, @ 2014:A, @ &1007, @ &3004	Dump à l'adresse absolue 2014 de la zone mémoire comprise entre les adresses absolues &1000 et &300F.
% SN/S&02, @ &30:S&03, @ &0000, @ &0100	Dump à l'adresse &30 du LPS de la section &02 de la première page du LDS de la section &03.

VIII-2.7. %HALT commande d'arrêt sur instruction

- Définition

Arrêt sur adresse pour entrée de commandes dynamiques, après édition du M:OC, suivant le format du % TRACE, des six premiers registres, des indicateurs et de l'instruction adressée (après exécution de celle-ci).

L'exécution d'une commande % HALT précède normalement l'entrée des commande % SNAP, % TRACE et % NEXT, la reprise se faisant par la commande % EXECUTE.

- Forme

$$\left\{\begin{matrix}\%\ \text{HALT}\\ \%\ \text{HA}\end{matrix}\right\} / \left[\left\{\begin{matrix}[\text{A}]\\ [\text{S}\ [\&]\ xx]\end{matrix}\right\} , @\ [\&]\ aaaa\right]$$

- Sens des paramètres

[A] L'adresse indiquée est absolue.

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

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

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

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

Absence des paramètres

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

Remarques :

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

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

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

- après chargement, avant lancement
- après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

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

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

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

Exemples :

% HALT/A, @ &2000

% HA/@ &2000,A

% HA/@ 24

% HA

% HALT

% HA/S02, @ &F0

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

VIII-2.8.% NEXT commande d'exécution d'une séquence d'instructions

• Définition

Implantation d'une "HALT" à l'adresse courante augmentée d'un nombre de mots (nombre d'instructions). L'exécution sera lancée par % EXECUTE. Au passage de l'adresse où est implantée la "HALT", il y aura édition sur M:OC des six premiers registres, des indicateurs et de l'instruction adressée (après exécution de celle-ci).

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

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

• Forme

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

• Sens des paramètres

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

Valeur par défaut : N1

Remarques :

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

On ne peut interrompre un NEXT en cours d'exécution pour spécifier une nouvelle commande % NEXT.

Si aucun NEXT n'est en cours, une nouvelle commande % NEXT annule et remplace la précédente.

Une commande % NEXT n'a de sens que si au moins une commande % HALT a déjà été exécutée. Une commande % NEXT ne se spécifie donc normalement qu'après prise de contrôle à la suite de l'exécution d'une commande % HALT ou % NEXT.

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

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

Si, au cours de l'exécution d'une commande % NEXT, on passe par CSV ou un CLS, les instructions de la section appelée ne seront pas comptées (voir commande % TRACE, remarques).

Exemples 1 :

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

Exemples 2':

```
% HA/A, @&278A
% R
  X=0049 E=0008 A=18A0 G=1B12 L=2692 P=278C I=0002 F005 (SLLS=5)
         Halte en 278A
         Après exécution de SLLS=5, P=278C
% NE;
% EXE
  X=0049 E=0008 A=18A0 G=1B12 L=2692 P=2792 I=0002 C003 (BE $+3)
         Equivalent à % HA/A, @ &278C
         Après exécution de BE $+3, P=2792 (branchement satisfait)
% NE/N3;
% EXE
  X=0049 E=0008 A=18A0 G=1B12 L=2692 P=2794 I=0002 C403 (BAN $+3)
         Equivalent à % HA/A, @ &2792
         Après exécution de BAN $+3, P=2794 (branchement non satisfait)
% NE/N3;
% EXE
  X=0049 E=0008 A=0000 G=1B12 L=2692 P=279A I=0002 0E1B (LBR)
         Equivalent à %HA/A, @ &2798
         Après exécution de LBR, P=279A
% NE;
% EXE
  X=0049 E=0008 A=0000 G=1B12 L=2692 P=279C I=0002 030C (EOR)
         Equivalent à % HA/A, @ &279A
         Après exécution de EOR, P=279C
```

VIII-2.9.% EXECUTE commande de reprise du background

- Définition

Un programme background suspendu par l'exécution d'une commande % HALT ou % NEXT sera relancé par % EXECUTE.

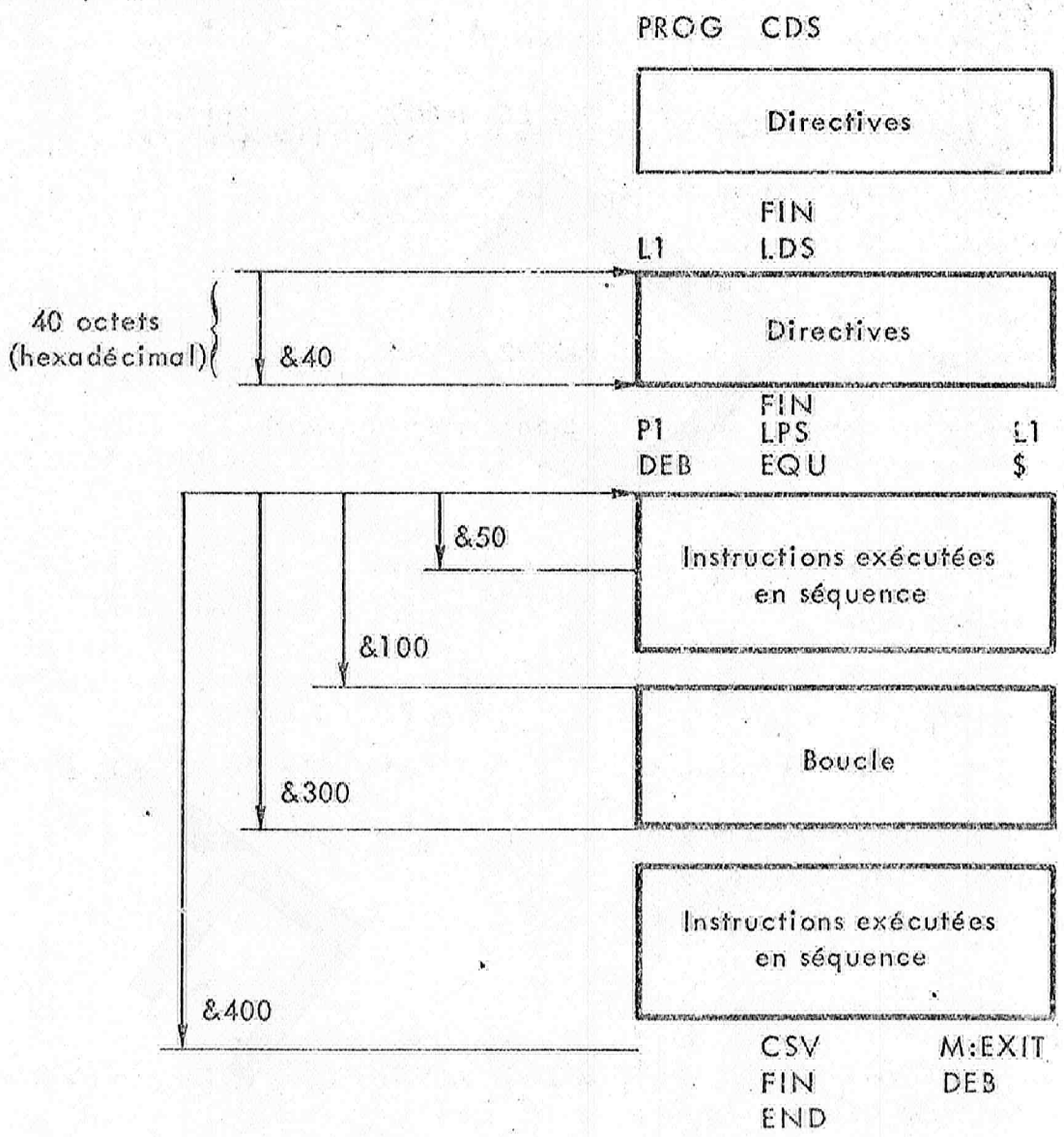
Une commande % EXECUTE ne peut remplacer % RUN et réciproquement, une commande % RUN ne peut remplacer une commande % EXECUTE.

- Forme

{ % EXECUTE
{ % EX

- Fonction et exemple

Soit le programme suivant :



L'analyse de l'exécution peut se faire de la façon suivante :

% LOAD

% HALT/S01, @ &20

% RUN

⋮ Déroulement

Exécution de l'instruction adressée dans le HALT précédent.

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

Passage en entrée de commande

% SNAP/S01, @ &100:S01, @ &00, @ &40

% HALT/S01, @ &300

% EX

⋮ Relance et déroulement du programme

Exécution du SNAP et éditions

⋮ Poursuite du déroulement, premier passage dans la boucle

Exécution de l'instruction adressée dans le HALT précédent.

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

Passage en entrée de commande

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

% SNAP/S01, @ &230:S01, @ &00, @ &40

% HALT

% EX

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

Exécution du TRACE et éditions

Exécution du SNAP et éditions

⋮ Poursuite du déroulement, deuxième passage dans la boucle

Exécution de l'instruction adressée dans le HALT précédent (réactivation du premier HALT)

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

Passage en entrée de commande

% TRACE/S01, @ &4FC

% SNAP/S01, @ &400

% EX

⋮ Relance du programme sans arrêt par halte, les SNAP et TRACE permettant de mettre en évidence le retour au moniteur

Dans le cas ou une erreur de programmation aurait fait que l'on reste dans la boucle sans pouvoir en sortir, on peut demander les actions suivantes au choix :

% DUMP/LO
⋮ Déroulement du DUMP du programme sur M:LO

% Y Abort du background sans DUMP

% X Abort du background et DUMP du background

VIII-2.10.% PERFORM commande de calcul

• Définition

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

• Forme :

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

• Sens des paramètres

X	Valeur courante du registre X (Index)
E	Valeur courante du registre E (Extension)
A	Valeur courante du registre A (Accumulateur)
G	Valeur courante de la base G (Base Générale)
L	Valeur courante de la base L (Base Locale)
P	Valeur courante de la base P (Base Programme)
[&]aaaa	Valeur numérique premier terme
+	L'opération est une addition
-	L'opération est une soustraction
[&]bbbb	Valeur numérique deuxième terme

Exemples :

% PERFORM/L+&00F0

% PE/P-24

VIII-2.11. %*/ — commande de sous-système

— Un sous-système est une fonction indépendante connectée sur un niveau d'IT et nécessitant un dialogue avec l'opérateur par l'intermédiaire de la console et du bouton IT pupitre.

— Pour cette raison, l'IT pupitre a besoin d'un certain nombre d'informations concernant les différents sous-systèmes potentiels ou actifs dans le système.

Forme

%*/xx/ paramètres pour le sous-système

xx code de reconnaissance du sous-système (2 caractères EBCDIC).

— Un sous-système est chargé comme un programme par % L et est activé (% AC).

— Pour qu'un sous-système soit considéré comme actif, il devra tout d'abord faire appel au module CSOL (voir CSV M : CSOL), dans le cas contraire, il sera considéré comme au repos et l'IT pupitre (la tâche) refusera toute commande adressée à ce sous-système.

VIII-3. COMMUNICATIONS OPERATEUR - MESSAGES OPERATEUR

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

VIII-3.1. Au chargement du système

SYSTEM xxxxxx.y READY où : xxxxxx = nom du moniteur (1 à 6 caractères EBCDIC)
y = numéro de la version

Le système vient d'être chargé en mémoire. Il est en attente d'une interruption pupître.

VIII-3.2. A l'analyse des commandes

%	En attente d'entrée de commande
%% AC01	Commande inconnue
%% AC02	Erreur de syntaxe
%% AC03	Commande non autorisée
%% AC04	Argument non autorisé
%% AC05	Débordement dans un nombre
%% AC06	Numéro incorrect (niveau d'interruption, numéro de section ou étiquette opérationnelle utilisateur).
%% AC07	Numéro du coupleur incorrect Numéro de périphérique incorrect pour un coupleur multipériphérique.
%% AC08	Etiquette opérationnelle non réassignable
%% AC09	Adresse inexistante

VIII-3.3. Au chargement (par LOAD)

%% LD01	Erreur d'entrée-sortie au chargement
%% LD02	Checksum (somme de contrôle) incorrecte
%% LD03	Erreur de séquencement des blocs IMT
%% LD04	Background occupé
%% LD05	Implantation incorrecte
%% LD06	Tentative de charger autre chose qu'un IMT
%% LD08	Rencontre d'une fin de fichier avant la fin normale du module IMT

VIII-3.4. A l'exécution d'une commande

%% EC01	% FOREGROUND : Adresse foreground impossible à implanter % ABORT : Essai d'abort de l'interruption pupître

	% ASSIGN	: Incompatibilité entre l'étiquette opérationnelle et le périphérique physique.
	% DUMP	: Débordement mémoire
	% SNAP	: Il manque une adresse
	% HALT	: Il manque une adresse et aucun HALT n'a été spécifié auparavant.
	% NEXT	: Il n'y a pas eu de HALT précédent
	% TRACE	: Il manque une adresse
	% DEVICE	: Commande incorrecte
	% PM	: Commande sans adresses
%% EC02	% FOREGROUND	: Agrandissement de la zone foreground impossible. Il faut au préalable aborter le niveau zéro.
	% ABORT	: Essai d'abort d'un niveau inactif
	% RUN	: Il n'y a pas de programme chargé
	% ASSIGN	: Incompatibilité entre le mode demandé (alphanumérique ou binaire) et les modes autorisés.
	% MODIFY	: Il n'y a pas de programme chargé
	% SNAP	: Adresse de SNAP en dehors des limites du dernier programme chargé.
	% HALT	: Adresse de HALT en dehors des limites du dernier programme chargé.
	% NEXT	: Le NEXT fait sortir des limites du dernier programme chargé.
	% TRACE	: Adresse de TRACE en dehors des limites du dernier programme chargé.
	% TIME	: Valeurs d'initialisation incorrectes
%% EC03	% X	: Niveau zéro inactif
	% Y	: Niveau zéro inactif
	% RUN	: Tentative de connexion d'un programme background à un niveau d'interruption.
	% IT	: Tentative de manipulation du niveau zéro
	% ACTIVATE	: Tentative de manipulation du niveau zéro
	% MODIFY	: Adresse inexistante
	% DUMP	: Erreur d'entrée-sortie sur M:LO
	% SNAP	: Il n'y a pas de programme chargé

	% HALT	: Il n'y a pas de programme chargé
	% NEXT	: Il n'y a pas de programme chargé
	% EXECUTE	: Il n'y a pas de programme chargé
	% TRACE	: Il n'y a pas de programme chargé
	% PM	: Adresse début > adresse fin
%% EC04	% RUN	: Tentative de connexion d'un programme foreground au niveau zéro.
	% IT	: Tentative de manipulation d'une IT inexistante
	% PM	: Adresse hors limite
	% SNAP	: Le dernier programme chargé l'a été en foreground
	% HALT	: Le dernier programme chargé l'a été en foreground
	% NEXT	: Le dernier programme chargé l'a été en foreground
	% EXECUTE	: Le dernier programme chargé l'a été en foreground
	% TRACE	: Le dernier programme chargé l'a été en foreground
%% EC05	% RUN	: Tentative de connexion d'un programme à un niveau occupé. Il faut au préalable aborter ce niveau.
%% EC06	% RUN	: Section de lancement inexistante
	% MODIFY	: Section inexistante
	% DUMP	: Il n'y a pas de programme chargé
	% SNAP	: Section inexistante
	% HALT	: Section inexistante
	% TRACE	: Section inexistante
%% EC07	% RUN	: L ou P impossible à implanter

VIII-3.5. Erreurs d'Entrée/Sortie

%% 1000	Time-out sur un périphérique ne possédant pas d'ordre STOP
%% 1001	Erreur logique détectée en fin de transfert
%% 1002	Erreur logique détectée à l'initialisation du transfert
%% 1003	Erreur physique détectée en fin de transfert
%% 1004	Erreur physique détectée à l'initialisation du transfert

VIII-3.6. Erreurs diverses

%% ER00	Processeur standard lancé de façon incorrecte
%% ER02	Ressource inexistante
%% ER03	Appel incorrect à M:FLAG et M:ADRS

VIII-3.7. Messages de déroutement

- Mot d'état déroutement
- P (en absolu)
- L (en absolu)
- Indicateurs

} sur l'instruction ayant provoqué le déroutement

- P - G
- L - G
- Indicateurs

} sur le dernier appel moniteur

%% DRxx avec { xx = 01 déroutement du à un coupleur
xx = 02 déroutement programme

VIII-3.8. Messages d'abort

%% Ayxx

xxxx	xxxx	xxxx
↓	↓	↓
Adresse de début d'implantation (absolue)	Valeur de la base G (absolue)	Première adresse libre après le programme (absolue)

avec 000 ⩽ yxx ⩽ 00 F niveau de zéro à quinze
100 ⩽ yxx ⩽ 10 F niveau de seize à trente et un

VIII-3.9. Incidents graves

%% CS
SYS READY

} Coupure secteur ayant entraîné une réinitialisation du système (voir chapitre V)

%%DRxxyy
SYS READY

} Déroutement ayant entraîné une réinitialisation du système (voir chapitre VI)

xx = numéro de la tâche qui a déroutée
yy = contenu du mot 2 de la mémoire qui donne la nature du déroutement

VIII-3.10. Périphériques non-opérationnels

%% XXYZ

XX : désigne le périphérique non opérationnel

Y : désigne l'unité de traitement à laquelle est connecté le périphérique.

0 : UC
1 : UEM1
2 : UEM2
3 : UEM3

Z : numéro de DEVICE pour un coupleur multipériphérique.

Exemples :

%% LP00	Imprimante sur UC non prête.
%% 9T10	Dérouleur de bande 0 sur UEM1 non prêt.

Action :

Rendre le périphérique opérationnel (mettre sous-tension, mettre ON-LINE, alimenter en papier, vider les magasins etc...).

VIII-4. MODULES MONITEUR DU MTR ETENDU

La liste des modules moniteur est identique à celle du MTR.

Annexe - Liste des instructions

Instr.	Classe	Code suivant adressage							
		P	DL	IL	ILX	DG	IGX	RP	RM
ADD	0	25	05	65	A5	45	85	-	-
ADM	0'	-	17	77	B7	57	97	-	-
AND	0	29	09	69	A9	49	89	-	-
BAN	2	-	-	D4	-	-	DC	C4	CC
BAZ	2	-	-	D5	-	-	DD	C5	CD
BCF	2	-	-	D3	-	-	DB	C3	CB
BCT	2	-	-	D0	-	-	D8	C0	C8
BOF	2	-	-	D6	-	-	DE	C6	CE
BOT	2	-	-	D2	-	-	DA	C2	CA
BRU	2	-	-	D7	-	-	DF	C7	CF
BRX	2	-	-	D1	-	-	D9	C1	C9
• CLM	1	F400	-	-	-	-	-	-	-
CLS	1	F8	38	-	-	-	-	E8 ∘	-
CMP	0	2B	0B	6B	AB	4B	8B	-	-
* CPS	0	2A	0A	6A	AA	4A	8A	-	-
CSV	1	F7	37	-	-	-	-	E7 ∘	-
DCL	1	F6	36	-	-	-	-	E6 ∘	-
DCX	1	F3	33	-	-	-	-	E3 ∘	-
• DIT	1	F401	-	-	-	-	-	-	-
* DIV	0	28	08	68	A8	48	88	-	-
DLD	0'	-	10	70	B0	50	90	-	-
DST	0'	-	16	76	B6	56	96	-	-
EOR	0	23	03	63	A3	43	83	-	-
* FAD	0'	-	1A	7A	BA	5A	9A	-	-
* FDV	0'	-	1D	7D	BD	5D	9D	-	-
* FMU	0'	-	1C	7C	BC	5C	9C	-	-
* FSU	0'	-	1B	7B	BB	5B	9B	-	-
ICL	1	F5	35	-	-	-	-	E5 ∘	-
ICX	1	F2	32	-	-	-	-	E2 ∘	-
IOR	0	27	07	67	A7	47	87	-	-

Instr.	Classe	Code suivant adressage							
		P	DL	IL	ILX	DG	IGX	RP	RM
LBL	0	2D	0D	6D	AD	4D	8D	-	-
LBR	0	2E	0E	6E	AE	4E	8E	-	-
LBX	0	2F	0F	6F	AF	4F	8F	-	-
LDA	0	20	00	60	A0	40	80	-	-
LDE	0	21	01	61	A1	41	81	-	-
** LDP	1	FB	3B	-	-	-	-	-	-
• LDR	1	F9	39	-	-	-	-	-	-
LDX	0	22	02	62	A2	42	82	-	-
LEA	0	-	04	64	A4	44	84	-	-
MUL	0	2C	0C	6C	AC	4C	8C	-	-
* MVS	0'	-	1F	7F	BF	5F	9F	-	-
• RD	1	F402	-	-	-	-	-	-	-
RSV	1	F1	-	-	-	-	-	-	-
RTS	1	F100	-	-	-	-	-	-	-
SBL	0'	-	14	74	B4	54	94	-	-
SBR	0'	-	15	75	B5	55	95	-	-
* SHC	1	FC	3C	-	-	-	-	∘EC	-
SHR	1	F0	30	-	-	-	-	∘E0	-
SPA	0'	-	18	78	B8	58	98	-	-
SRG	1	F1	31	-	-	-	-	∘F1	-
STA	0'	-	11	71	B1	51	91	-	-
STE	0'	-	12	72	B2	52	92	-	-
• STM	1	F408	-	-	-	-	-	-	-
• STR	1	FA	3A	-	-	-	-	∘EA	-
STS	0'	-	19	79	B9	59	99	-	-
STX	0'	-	13	73	B3	53	93	-	-
SUB	0	26	06	66	A6	46	86	-	-
* TES	1	FD	3D	-	-	-	-	∘ED	-
* TRS	0'	-	1E	7E	BE	5E	9E	-	-
• WD	1	F403	-	-	-	-	-	-	-

Nota :

• : instruction privilégiée

* : instruction optionnelle

∘ : adressage ?X

Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.
00	LDA	0	DL	20	LDA	0	P	40	LDA	0	DG	60	LDA	0	IL
01	LDE	0	-	21	LDE	0	-	41	LDE	0	-	61	LDE	0	-
02	LDX	0	-	22	LDX	0	-	42	LDX	0	-	62	LDX	0	-
03	EOR	0	-	23	EOR	0	-	43	EOR	0	-	63	EOR	0	-
04	LEA	0	-	24	LEA	0	-	44	LEA	0	-	64	LEA	0	-
05	ADD	0	-	25	ADD	0	-	45	ADD	0	-	65	ADD	0	-
06	SUB	0	-	26	SUB	0	-	46	SUB	0	-	66	SUB	0	-
07	IOR	0	-	27	IOR	0	-	47	IOR	0	-	67	IOR	0	-
08	* DIV	0	-	28	* DIV	0	-	48	* DIV	0	-	68	* DIV	0	-
09	AND	0	-	29	AND	0	-	49	AND	0	-	69	AND	0	-
0A	* CPS	0	-	2A	* CPS	0	-	4A	* CPS	0	-	6A	* CPS	0	-
0B	CMP	0	-	2B	CMP	0	-	4B	CMP	0	-	6B	CMP	0	-
0C	MUL	0	-	2C	MUL	0	-	4C	MUL	0	-	6C	MUL	0	-
0D	LBL	0	-	2D	LBL	0	-	4D	LBL	0	-	6D	LBL	0	-
0E	LBR	0	-	2E	LBR	0	-	4E	LBR	0	-	6E	LBR	0	-
0F	LBX	0	-	2F	LBX	0	-	4F	LBX	0	-	6F	LBX	0	-
10	DLD	0'	-	30	SHR	1	DL	50	DLD	0'	-	70	DLD	0'	-
11	STA	0'	-	31	SRG	1	-	51	STA	0'	-	71	STA	0'	-
12	STE	0'	-	32	ICX	1	-	52	STE	0'	-	72	STE	0'	-
13	STX	0'	-	33	DCX	1	-	53	STX	0'	-	73	STX	0'	-
14	SBL	0'	-	34				54	SBL	0'	-	74	SBL	0'	-
15	SBR	0'	-	35	ICL	1	-	55	SBR	0'	-	75	SBR	0'	-
16	DST	0'	-	36	DCL	1	-	56	DST	0'	-	76	DST	0'	-
17	ADM	0'	-	37	CSV	1	-	57	ADM	0'	-	77	ADM	0'	-
18	SPA	0'	-	38	CLS	1	-	58	SPA	0'	-	78	SPA	0'	-
19	STS	0'	-	39	• LDR	1	-	59	STS	0'	-	79	STS	0'	-
1A	* FAD	0'	-	3A	• STR	1	-	5A	* FAD	0'	-	7A	* FAD	0'	-
1B	* FSU	0'	-	3B	* • LDP	1	-	5B	* FSU	0'	-	7B	* FSU	0'	-
1C	* FMU	0'	-	3C	* SHC	1	-	5C	* FMU	0'	-	7C	* FMU	0'	-
1D	* FDV	0'	-	3D	* TES	1	-	5D	* FDV	0'	-	7D	* FDV	0'	-
1E	* TRS	0'	-	3E				5E	* TRS	0'	-	7E	* TRS	0'	-
1F	* MVS	0'	-	3F				5F	* MVS	0'	-	7F	* MVS	0'	-

Nota :

* : instruction en option
• : instruction privilégiée

Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.	Code	Instruc.	Classe	Adr.
80	LDA	0	IGX	A0	LDA	0	ILX	C0	BCT	2	RP	E0	SHR	1	PX
81	LDE	0	-	A1	LDE	0	-	C1	BRX	2	-	E1	SRG	1	-
82	LDX	0	-	A2	LDX	0	-	C2	BOT	2	-	E2	ICX	1	-
83	EOR	0	-	A3	EOR	0	-	C3	BCF	2	-	E3	DCX	1	-
84	LEA	0	-	A4	LEA	0	-	C4	BAN	2	-	E4			
85	ADD	0	-	A5	ADD	0	-	C5	BAZ	2	-	E5	ICL	1	-
86	SUB	0	-	A6	SUB	0	-	C6	BOF	2	-	E6	DCL	1	-
87	IOR	0	-	A7	IOR	0	-	C7	BRU	2	-	E7	CSV	1	-
88	*DIV	0	-	A8	*DIV	0	-	C8	BCT	2	RM	E8	CLS	1	-
89	AND	0	-	A9	AND	0	-	C9	BRX	2	-	E9	• LDR	1	-
8A	*CPS	0	-	AA	*CPS	0	-	CA	BOT	2	-	EA	• STR	1	-
8B	CMP	0	-	AB	CMP	0	-	CB	BCF	2	-	EB	••LDP	1	-
8C	MUL	0	-	AC	MUL	0	-	CC	BAN	2	-	EC	* SHC	1	-
8D	LBL	0	-	AD	LBL	0	-	CD	BAZ	2	-	ED	* TES	1	-
8E	LBR	0	-	AE	LBR	0	-	CE	BOF	2	-	EE			
8F	LBX	0	-	AF	LBX	0	-	CF	BRU	2	-	EF			
90	DLD	0'	-	B0	DLD	0'	-	D0	BCT	2	IL	F0	SHR	1	P
91	STA	0'	-	B1	STA	0'	-	D1	BRX	2	-	F1	SRG	1	-
92	STE	0'	-	B2	STE	0'	-	D2	BOT	2	-	F2	ICX	1	-
93	STX	0'	-	B3	STX	0'	-	D3	BCF	2	-	F3	DCX	1	-
94	SBL	0'	-	B4	SBL	0'	-	D4	BAN	2	-	F4	• SYS(1)	1	-
95	SBR	0'	-	B5	SBR	0'	-	D5	BAZ	2	-	F5	ICL	1	-
96	DST	0'	-	B6	DST	0'	-	D6	BOF	2	-	F6	DCL	1	-
97	ADM	0'	-	B7	ADM	0'	-	D7	BRU	2	-	F7	CSV	1	-
98	SPA	0'	-	B8	SPA	0'	-	D8	BCT	2	IG	F8	CLS	1	-
99	STS	0'	-	B9	STS	0'	-	D9	BRX	2	-	F9	• LDR	1	-
9A	*FAD	0'	-	BA	*FAD	0'	-	DA	BOT	2	-	FA	• STR	1	-
9B	*FSU	0'	-	BB	*FSU	0'	-	DB	BCF	2	-	FB	••LDP	1	-
9C	*FMU	0'	-	BC	*FMU	0'	-	DC	BAN	2	-	FC	* SHC	1	-
9D	*FDV	0'	-	BD	*FDV	0'	-	DD	BAZ	2	-	FD	* TES	1	-
9E	*TRS	0'	-	BE	*TRS	0'	-	DE	BOF	2	-	FE			
9F	*MVS	0'	-	BF	*MSV	0'	-	DF	BRU	2	-	FF			

Nota :

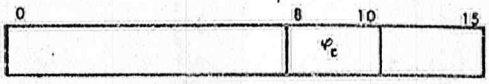
* : instruction en option
• : instruction privilégiée

(1) SYS : ce mnémonique n'est pas reconnu par l'assembleur

Instruction SRG

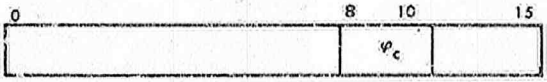
Instruction	Fonction	Code
AAE	(A) ∩ (E) ⟶ (A)	F118
ACE	(E) + C ⟶ (E)	F10E
AEE	(A) ⊕ (E) ⟶ (A)	F112
AIE	(A) ∪ (E) ⟶ (A)	F116
CCA	$(\overline{A})$ ⟶ (A)	F110
CCE	$(\overline{E})$ ⟶ (E)	F10A
CHX	Décalage arithmétique droit X 1 pas	F11E
CNA	-A ⟶ (A)	F11C
CNX	-X ⟶ (X)	F114
LNE	-1 ⟶ (E)	F11A
RTS	Retour section	F100
RSV	Retour superviseur	F10C
XAA	A_{0-7} ⟷ A_{8-15}	F108
XAE	(A) ⟷ (E)	F102
XAX	(A) ⟷ (X)	F104
XEX	(E) ⟷ (X)	F106

Instruction SHR



Fonction	φ_c	Code
Logique gauche A	0	SLLS
Circulaire droit A	1	SRCS
Arithmétique droit E,A	2	SAD
Circulaire gauche E,A	3	SLCD
Circulaire gauche A	4	SLCS
Arithmétique droit A	5	SAS
Logique droit A	6	SRLS
Circulaire droit E,A	7	SRCD

Instruction SHC



Fonction	φ_c	Code
Décalage logique gauche E,A	0	SLLD
Calcul de parité	2	PTY
Décalage logique droit E,A	4	SRLD
Normalisation E,A	6	NLZ
Désactivation de l'IT rapide	1, 3, 5, 7	DITR

Instruction SYS[1]

Fonction	Classe	Instruction	Code
Démasquage IT	1	CLM	F400
Désactivation IT	1	DIT	F401
Lecture	1	RD	F402
Ecriture	1	WD	F403
Masquage IT	1	SIM	F408

(1) SYS : ce mnémonique n'est pas reconnu par l'assembleur

Adressage :

Classe 0		Classe 0'		Classe 1		Classe 2	
P	Y = D	DL	Y = D + (L)	P	N = D	RP	Y = (P) + 2D
DL	Y = D + (L)	IL	Y = (D +(L))+G'	PX	N = D + (X)	RM	Y = (P) - 2D
IL	Y = (D +(L))+G'	ILX	Y = (D +(L))+G'+(X)	DL	N = (D + (L))	DL	Y = (D +(L))+G'
ILX	Y = (D +(L))+G'+(X)	DG	Y = D + (G)			DG	Y = (D +(G))+G'
DG	Y = D + (G)	IGX	Y = (D +(G))+(G)+(X)				
IGX	Y = (D +(G))+(G)+(X)						