Réseaux
et systèmes
d'information

Bull

MANUEL DE RÉFÉRENCE

SOLAR

UNITÉS CENTRALES

Caractéristiques générales
Description des instructions

Bull

1 160 900 00 036

AVANT PROPOS

UNITÉS CENTRALES

MANUEL DE RÉFÉRENCE

Ce Manuel de Référence décrit les unités centrales de la série SOLAR ; il comprend la description de toutes les instructions de la série sauf celles des options DAP16, VSS16 et FFM16.
Il ne donne que les renseignements utiles à la programmation de ces unités centrales.

La programmation des entrées-sorties se fait normalement par l'intermédiaire du programme IOCS (cf. Manuel de Référence IOCS, Manuels de Référence des systèmes, Handbook SOLAR 16). Les entrées-sorties sont traitées ici sans tenir compte des particularités relatives aux différents périphériques. Si l'on n'utilise pas IOCS, le présent manuel est donc insuffisant pour permettre la programmation des entrées-sorties.

Ce manuel sera utilisé en complément du Handbook SOLAR16, lorsque des précisions supplémentaires seront nécessaires. En raison de son organisation et de la nécessité de fournir le maximum de détails sur chacun des points abordés il n'est pas adapté à une première étude de la série SOLAR.

Ⓧ : s'applique uniquement au processeur X

⊗ : ne s'applique pas au processeur X

Sauf spécifications contraires, les caractéristiques des processeurs suivants sont identiques :

- 16-04 identique au 16-05
- 16-30 et 16-35 identiques au 16-40
- 16-70 et 16-75 identiques qu 16-65.

Le modèle 85 est un modèle commercial mono en bi-processeur constitué d'unités de traitement 75P.
Le modèle 90 est un modèle commercial mono en bi-processeur constitué d'unités de traitement 16-70.

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

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

TABLES DES MATIERES

1 - REPRÉSENTATION DES NOMBRES

1.1 – NOMBRES ENTIERS ALGEBRIQUES

Les instructions suivantes : additions (AD, ADR, ADRI, ADCR, IC), soustractions (SB, SBR SBCR, DC), multiplications (MP), divisions (DV), changement de signe (NGR), comparaisons (CP, CPZ, CPR CPZR, CPI), décalages numériques (SARS, SARD), permettent de faire des calculs sur des nombres binaires positifs et négatifs. Ces nombres, représentés sous la forme complément à deux, sont telS que :

. Les nombres positifs ont toujours le bit 0 à zéro, et les nombres négatifs ont toujours le bit 0 à un. Le **bit 0 doit être interprété comme ayant le poids -2^{15} et les 15 bits poids faible comme ayant un poids** positif.

Par exemple :

1 111 1111 1111 1111 = $(-2^{15}) + (2^{15} - 1) = -1$

. On change le signe d'un nombre en laissent inchangés tous les bits à partir de la droite jusqu'au premier 1 rencontré inclus, puis en inversant tous les autres bits.

Par exemple :

= 0000 0000 0000 0100
= 1111 1111 1111 1100

. **Sur 16 bits ces nombres sont compris entre :**

– 32 768 et 32 767
soit : -2^{15} et $2^{15} - 1$
soit : `8000 et `7FFF en hexadécimal

. **Sur 32 bits ces nombres sont compris entre :**

– 4 294 967 296 et 4 294 967 295
soit : -2^{31} et $-2^{31} - 1$
soit : '8000 0000 et '7FFF FFFF en hexadécimal

. **L'opposé du nombre négatif -2^{15} sur 16 bits et l'opposé du nombre -2^{31} sur 32 bits ne sont pas représentables.**

. Le zéro n'a qu'une seule représentation.

. Pour attendre sur n + p bits un nombre de n bits, on le complète à gauche par p bits identiques au bit poids fort du nombre initial.

Par exemple :

'40 s'écrit '0040 sur 16 bits
'8000 s'écrit 'FFFF 8000 sur 32 bits

. Pour réduire à n bits un nombre de n + p bits, il faut que les p bits éliminés soient tous identiques au bit poids fort du résultat

Par exemple :

'FFFF 8000 peut s'écrire '8000 sur 16 bits
'FF00 ne peut pas s'écrire sur 8 bits

. Un nombre de n bits peut être considéré comme la somme de deux nombres : un nombre algébrique constitue par p bits poids fort suivis de n - p zéro, et un nombre positif constitué par les n - p bits poids faible.

Par exemple, en double longueur

`FFFF FFFF = `0000 FFFF + `FFFF 0000
= 32 767 + (– 1 x 2^{15})
= – 1

1.1.1 - SIGNIFICATION DU DÉBORDEMENT POUR LES NOMBRES ALGEBRIQUES

Lorsque les opérandes et le résultat d'une addition, d'une soustraction ou d'un changement de signe sont considérés comme des nombres algébriques, le débordement signifie que le résultat est supérieur à 32 767 ($2^{15} - 1$) ou inférieur à – 32 768 (– 2^{15}), donc n'est pas représentable sur 16 bits.

Le registre destination contient, après le déroulement de l'instruction, 16 bits qui sont les 16 bits poids faible du résultat algébrique correct.

1.1.2 - SIGNIFICATION DU REPORT POUR LES NOMBRES ALGÉBRIQUES

Le report est sans signification pour des opérandes algébriques.

1.1.3 – DÉBORDEMENT DANS LE CAS D'UNE ADDITION DE NOMBRES ALGÉBRIQUES

Pour AD, ADR, ADRI, ADCR le tableau suivant indique les cas où l'indicateur V (bit 6 de S) est positionné a 1.

1er opérande	2d opérande	résultat obtenu	indicateur V	le résultat algébrique correct serait
⩾ 0	⩾ 0	⩾ 0	0	
⩾ 0	⩾ 0	< 0	1	> 32 767
⩾ 0	< 0	⩾ 0	0	
⩾ 0	< 0	< 0	0	
< 0	⩾ 0	⩾ 0	0	
< 0	⩾ 0	< 0	0	
< 0	< 0	⩾ 0	1	<– 32 768
< 0	< 0	< 0	0	

1.1.4 – DÉBORDEMENT DANS LE CAS D'UNE SOUSTRACTION DE NOMBRES ALGÉBRIQUES

Pour SB, SBR, SBCR le tableau suivant indique les œs où l'indicateur V (bit 6 de S) est positionné à 1 (opérande 1 - opérande 2).

1er opérande	2d opérande	résultat obtenu	indicateur V	le résultat algébrique correct serait
⩾ 0	⩾ 0	⩾ 0	0	
⩾ 0	⩾ 0	< 0	0	
⩾ 0	< 0	⩾ 0	0	
⩾ 0	< 0	< 0	1	> 32 767
< 0	⩾ 0	⩾ 0	1	< – 32 768
< 0	⩾ 0	< 0	0	
< 0	< 0	⩾ 0	0	
< 0	< 0	< 0	0	

1.1.5 – DÉBORDEMENT DANS LE CAS DE CHANGEMENT DE SIGNE

Pour NGR : il y a débordement si l'opérande et le résultat obtenu sont de même signe, c'est-à-dire si l'opérande **est égal à – 32 768 (– 2^{16})**

1.1.6 – REPORT DANS LE CAS D'UN CHANGEMENT DE SIGNE

Pour NGR : il y a report dans tous les cas, sauf celui où l'opérande est nul

1.2 – NOMBRES ENTIERS POSITIFS

Alors que les opérations arithmétiques de multiplication, division, comparaisons ne peuvent être utilisées qu'avec des nombres algébriques, les instructions d'addition (AD, ADR ADRI ADCR) et de soustraction (SB, SBR, SBCR) permettent de faire aussi des calculs sur des entiers positifs. Le bit 0 ayant le poids **2^{15}, ces nombres sont compris entre 0 et 65 535 (2^{16} – 1)**.

Les adresses mémoire et la partie poids faible des nombres en double longueur sont des nombres absolus.

1.2.1 – SIGNIFICATION DU DEBORDEMENT POUR LES NOMBRES ENTIERS POSITIFS

Le débordement est sans signification pour des entiers positifs.

1.2.2 – SIGNIFICATION DU REPORT POUR DES NOMBRES ENTIERS POSITIFS

Lorsque les opérandes et le résultat d'une addition ou d'une soustraction sont considérés comme des entiers **positifs, le report signifie que le résultat est supérieur à 65 535 (2^{16} - 1) dans le cas de l'addition, ou inférieur** à 0 dans le cas de la soustraction, donc n'est pas représentable sur 16 bits. Le registre destination contient après le déroulement de l'instruction 16 bits qui sont les 16 bits poids faible du résultat correct.

1.2.3 – REPORT DANS LE CAS D'UNE ADDITION DE NOMBRES ENTIERS POSITIFS

Pour AD, ADR, ADRI, ADCR le tableau suivant indique les cas où l'indicateur C (bit 7 de S) est positionné à 1

bit 0 du 1er opérande	bit 0 du 2d opérande	bit 0 du résultat obtenu	Indicateur C	le résultat absolu correct serait
0	0	0	0	
0	0	1	0	
0	1	0	1	> 65 535
0	1	1	0	
1	0	0	1	> 65 535
1	0	1	0	
1	1	0	1	> 65 535
1	1	1	1	> 65 535

1.2.4 – REPORT DANS LE CAS D'UNE SOUSTRACTION DE NOMBRES ENTIERS POSITIFS

Pour SB, SBR, SBCR le tableau suivant indique les cas où l'indicateur C (bit 7 de S) est positionné à 1 (opérande 1 - opérande 2).

bit 0 du 1er opérande	bit 0 du 2d opérande	bit 0 du résultat obtenu	indicateur C	résultat absolu correct
0	0	0		
0	0	1		< o
0	1	0		< o
0	1	1		< o
1	0	0		
1	0	1		
1	1	0		
1	1	1		< o

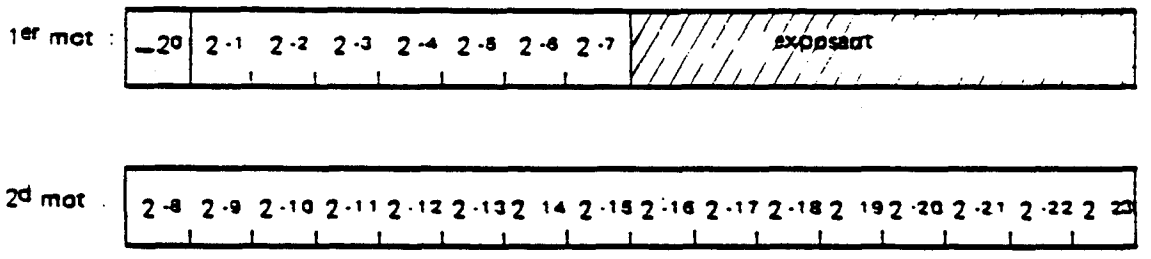
1.3 - NOMBRES FLOTTANTS

1.3.1 - NOMBRES FLOTTANTS SIMPLE PRECISION

L'arithmétique en virgule flottante simple précision permet de traiter des nombres algébriques dont la valeur absolue est comprise entre 10^{-37} et 10^{37} avec, selon les cas, 6 ou 7 chiffres significatifs.

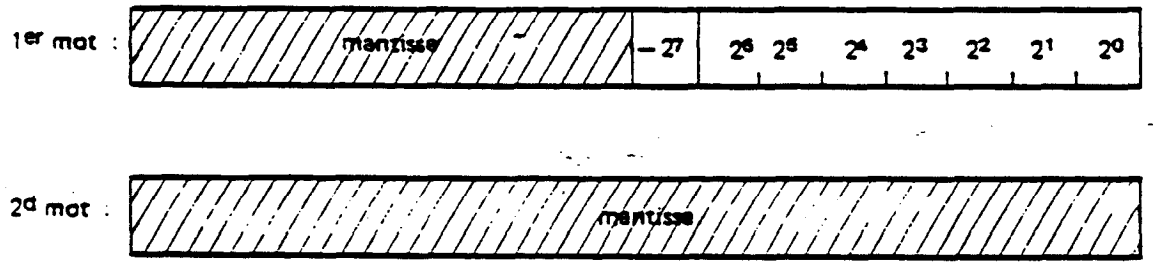
La représentation choisie pour ces nombres nécessite 32 bits. Ils occupent deux mots mémoire consécutifs ou deux registres du calculateur. Un nombre flottant est constitué d'une mantisse m de 24 bits et d'un exposant e de 8 bits. Il est interprété comme ayant la valeur $0,m \times 2^e$. Si le nombre est négatif la mantisse m est négative, elle est alors représentée en complément à deux.

Le bit 0 de la mantisse doit être interprété comme ayant le poids -2^0 et les bits suivants les poids $+2^{-1} + 2^{-2}, \ldots, + 2^{-23}$.



Si le nombre est inférieur à 1/2 en valeur absolue l'exposant est négatif, il est alors représenté en complément à deux.

Le bit 0 de l'exposant (bit 8 du premier mot) doit être interprété comme ayant le poids -2^7 et les sept bits suivants les poids $+2^6, +2^5, \ldots, +2^0$



Les nombres positifs ont toujours le bit 0 de la mantisse à zéro, les nombres négatifs ont toujours le bit 0 de la mantisse à 1.

Les nombres flottants sont toujours normalisés, c'est-à-dire qu'ils sont tels que les deux premiers bits de la mantisse sont différents, sauf pour les nombres négatifs dont la valeur absolue est une puissance de 2. Dans ce cas, par convention, les deux premiers bits de la mentisse sont à 1. La représentation unique utilisée pour le zéro comporte une mantisse nulle et l'exposant minimum.

- Nombre positif :

0 1 _ _ _ _ _ _ _ _ [_ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _]

- Nombre négatif (sauf puissance de 2) :

1 0 _ _ _ _ _ _ _ _ [_ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _]

- Nombre négatif puissance de 2

1 1 0 0 0 0 0 0 [_ _ _ _ _ _ _ _
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

- Zéro (seule représentation utilisée)

0 0 0 0 0 0 0 0 [1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

Nombre flottant + 1,250 :

0 1 0 1 0 0 0 0 [0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

$(2^{-1} + 2^{-3}) \times 2^{+1} = (0,5 + 0,125) \times 2$

$= 0,625 \times 2 = +1,250$

- Nombre flottant = 0,3125

1 0 1 1 0 0 0 0 [1 1 1 1 1 1 1 1]
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

$(-2^{0} + 2^{-2} + 2^{-3}) \times 2^{-1}$

$= (-1 + 0,25 + 0,125) \times 2^{-1}$

$= -0,625 \times \frac{1}{2} = -0,3125$

Pour prendre l'opposé d'un nombre (changement de signe) il suffit, dans tous les cas, de prendre l'opposé de la mantisse.

Ex

nombre de départ :

0 1 1 0 1 0 0 0 [0 0 0 0 1 0 0 0]
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

résultat :

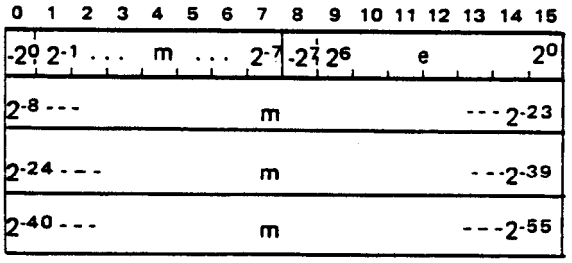
1 0 0 1 0 1 1 1 [0 0 0 0 1 0 0 0]
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1.3.2 - NOMBRES FLOTTANTS DOUBLE PRECISION

L'arithmétique en virgule flottante double précision permet de manipuler des nombres algébriques dont la **valeur absolue est comprise entre 10^{-37} et 10^{37}** avec, selon le cas, 15 ou 16 chiffres significatifs.

Ces nombres sont représentés sur 64 bits et occupent 4 mots mémoire consécutifs.
Le format est le suivant :

- mantisse signée sur 56 bits
- exposant signé sur 8 bits



L'exposant représente la puissance de 2

$Nb = 0,m \times 2^{e}$

Lorsque la mantisse est négative, elle est représentée en complément à 2, de même pour l'exposant.
Les règles de normalisation sont les même qu'en simple précision.
Le zéro a également une seule représentation : mantisse nulle et exposant minimum

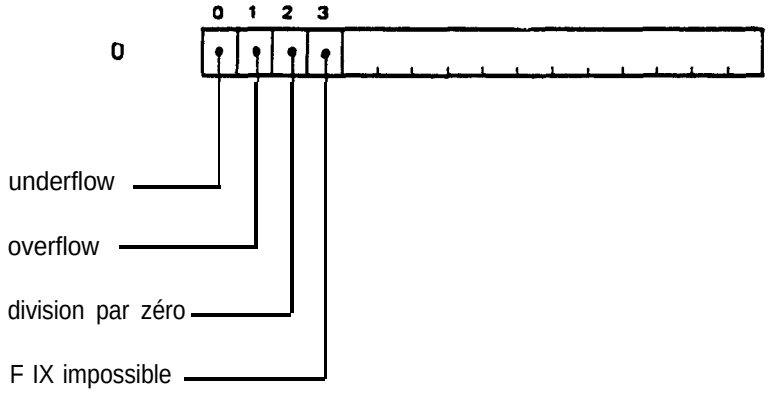
1.33 • DEBORDEMENT DES NOMBRES FLOTTANTS (OVERFLOW ET UNDERFLOW)

Au cours d'une addition, d'une soustraction, d'une multiplication, d'une division ou d'une normalisation, il peut se produire que l'exposant du résultat ne soit pas représentable sur 8 bits. II y a overflow Si l'opération conduit à un résultat dont l'exposant est supérieur à 127 (nombre trop grand en valeur absolue). Dans ce cas l'indicateur V est positionné à 1

II y a underflow si l'opération conduit à un résultat dont l'exposant est inférieur à - 128 (nombre trop petit en valeur absolue). Dans ce cas le résultat fourni est zéro et l'indicateur C est positionne à 1.

En plus de ces deux indicateurs le premier mot du COMMON (mot dont l'adresse est donnée par le contenu du registre C diminué de 128) est utilise pour mémoriser les underflows et les overflows
le bit 0 de ce mot est mis à 1 lorsqu'il y a underflow, le bit 1 lorsqu'il y a overflow, le bit 2 lorsqu'il y a division par zéro. et le bit 3 lorsque l'opération FIX est impossible

Ces indicateurs sont rémanents, les bits correspondants n'étant jamais remis à zéro par les opérations en virgule flottante, et de cette manière il est possible de ne pas tester les indicateurs V et C après chacune des opérations en virgule flottante d'un programme.



2 - REGISTRES ACCESSIBLES PAR PROGRAMME

2 - REGISTRES ACCESSIBLES PAR PROGRAMME

A - REGISTRE ACCUMULATEUR

A est le registre destination de la plupart des instructions ayant un mot mémoire pour opérande

A peut être chargé à partir de la mémoire ou rangé en mémoire.

A est un des registres opérandes possibles des Instructions registre-registre

A est le registre qui sert aux échanges avec les périphériques.

A est sauvegardé et restaure lors des changements de contextes de tâches différées.

B - EXTENSION DE L'ACCUMULATEUR

B sert d'extension à l'accumulateur et constitue avec celui-ci un registre 32 bits

- pour des opérandes arithmétiques en double longueur (multiplication, division, décalage numérique)
- pour des opérandes logiques en double longueur (décalages logiques et circulaires, opérations sur bits)
- pour les opérandes en virgule flottante

B peut être charge à partir de la mémoire ou range en mémoire

B est un des registres opérandes possibles des instructions registre-registre.

B est sauvegarde et restauré lors des changements de contextes de tâches différées.

X - REGISTRE INDEX

X a le rôle d'index dans l'adressage indirect des opérandes mémoire : lorsqu'une constante adresse comporte le bit 0 à 1 les 16 bits de l'index sont ajoutés aux 15 bits poids faible de la constante adresse ; le contenu de X est alors considéré comme un entier algébrique.
X permet d'indexer les instructions de décalage, d'opération sur bit et l'instruction ARM : son contenu est ajouté aux bits 11 à 15 (9 à 15 pour ARM) de l'instruction.

X peut être chargé à partir de la mémoire et rangé en mémoire.

X est un des registres opérandes possibles des instructions registre-registre.

X est sauvegarde et restauré lors des changements de contextes de tâches différées.

Y - REGISTRE INTERMEDIAIRE

Y peut être chargé à partir de la mémoire et rangé en mémoire

Y est un des opérandes possibles des instructions registre-registre.

De préférence aux registres A B et X qui ont des rôles spécifiques et qu'il est souvent intéressant de ne pas modifier, Y est destiné à servir de registre intermédiaire.

Par exemple l'utilisation de Y est conseillée pour :

- charger à partir de la mémoire ou ranger en mémoire C, L, W ou K ;

- transférer le contenu d'une mémoire dans une autre mémoire
- faire des opérations arithmétiques, logiques, de comparaison entre la mémoire et un des registres B, X, C, L, W, K sans modifier l'accumulateur.

Y est sauvegardé et restauré lors des changements de contextes différés.

C - BASE COMMUNE

L - BASE LOCALE

W - BASE DE TRAVAIL

C, W et L ont un fonctionnement identique. Le choix des appellations respectives de ces registres (qui correspond à l'utilisation qui en est faite dans l'assembleur et PL16) est arbitraire.

C, L et W sont utilisés en tant que bases dans l'adressage de la mémoire, les 16 bits de ces registres permettant aux instructions à référence mémoire (qui comportent l'indication d'une base et d'un déplacement sur 6 bits) d'accéder à trois zones mémoire de 256 mots chacunes.

C, L et W sont des opérandes possibles pour les instructions registre-registre.

C, L et W sont sauvegardés et restaurés lors des changements de contextes différées. Seul C est sauvegarde et restauré lors des changements partiels des tâches immédiates.

K - POINTEUR DE PILE

Les 16 bits de K sont utilisés pour pointer sur une zone mémoire fonctionnant en pile. Les instructions BSR et SVC utilisent cette pile pour sauvegarder le registre P, les instructions RSR et RSV pour le restaurer. De même l'instruction PSR utilise cette pile pour sauvegarder les registres A, B, X, Y, C, L, W, K et l'instruction PLR pour le restaurer.

Les instructions de sauvegarde incrémentent K avant chaque rangement et les instructions de restauration le décrémentent après chaque chargement, de telle manière qu'il pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande. Si la pile est vide K pointe sur la mémoire précédant immédiatement la pile.

K est un des opérandes possibles des instructions registre-registre.

K est sauvegardé et restauré lors des changements de contextes immédiats et différés.

Il doit y avoir une pile pour chaque tâche immédiate ou différée ; en particulier, les tâches différées peuvent être mises en attente et ne se terminent donc pas toujours dans l'ordre hiérarchique des priorités.

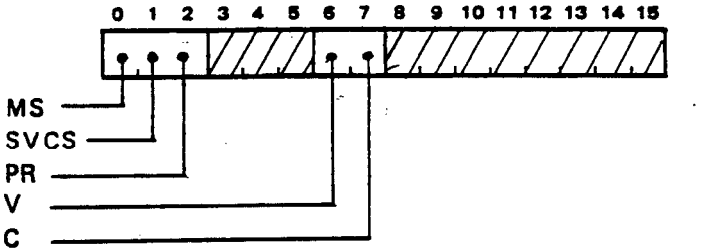
P - POINTEUR D'INSTRUCTION

Les 16 bits de ce registre indiquent l'adresse de la prochaine instruction à exécuter.

P est sauvegardé et restauré lors des changements de contextes de tâches immédiates et différées.

S - REGISTRE D'ETAT DE TACHE

S est en fait le sous-ensemble des bits de ST qui appartiennent au contexte d'une tâche ; ils participent donc aux changements de contextes de tâches immédiates et différées.



Les bits MS, SVCS et PR déterminent le mode de fonctionnement de la machine.

MS	SVCS	PR	Mode	Disponible sur SOLAR 16 04/05	30/35/40	65/75	70
0	0	0	Esclave	non	oui	oui	oui
0	0	1	Privilégié	non	non	non	oui
1	X	X	Maître	oui	oui	oui	oui

Dans le cas où le mode privilégié est inexistant (Solar autre que 16-70), la position du bit PR est non significative.

Le mécanisme DRPS est :

- inexistant sur SOLAR 16-04 et 16-05
- optionnel sur SOLAR 16-40, 16-65 et 16-75
- standard sur SOLAR 16-30, 16-35 et 16-70.

Mode Esclave :

Certaines instructions sont interdites dans ce mode.
Leur exécution provoque une alarme.
Si l'option DRPS est présente, le mécanisme de translation et de protection mémoire est activé.

Mode Privilégié :

Pratiquement toutes les instructions dites privilégiées sont exécutables dans ce mode.
Si l'option DRPS est présente, le mécanisme de translation et de protection mémoire est activée.

Mode Maître :

Toutes les instructions sont exécutables dans ce mode.
Pour l'adressage normal le mécanisme de translation et de protection mémoire est ineffectif.
Dans ce mode, le bit SVCS permet d'activer le mécanisme de translation et protection mémoire uniquement pour les empilements et dépilements à travers K par les instructions appropriées.
Les sous-programmes superviseur peuvent ainsi accéder à la pile de l'appelant (réentrance) tout en conservant les pr propriétés de translation et de protection DRPS.
Dans ce mode, le bit PR permet de savoir si l'appelant est un esclave (PR = 0) ou un privilégie (PR = 1).

Indicateur V et indicateur C :

Ces deux indicateurs sont positionnés par certaines instructions, avec des significations différentes selon l'instruction, et peuvent ensuite être testés par les jumps conditionnels. (Les opérations arithmétiques, les comparaisons, les décalages, certaines instructions sur bits, etc... positionnent ces deux indicateurs).

- après une instruction arithmétique :

  V = 1 si débordement (entiers algébriques)
  C = 1 si report (entiers arithmétiques)

- après instruction de comparaison

  V = 1 si les deux nombres sont égaux
  C = 1 si le 1er terme est algébriquement inférieur au second.

L'exécution de l'instruction SCY (qui positionne l'indicateur C à 1 sans modifier l'indicateur V) est le seul ces où un des indicateurs est modifie sans que l'autre le soit.
Dans tous les autres ces ou bien l'état de chacun des deux indicateurs a une signification, ou bien seul l'un des indicateurs a une signification et l'autre est remis à zéro.

SLO - ORIGINE DE LA PROTECTION MEMOIRE ET DE LA TRANSLATION

SLE - FIN DE LA PROTECTION MEMOIRE

Les registres SLO et SLE font partie de l'option DRPS (donc n'existent pas sur 05)

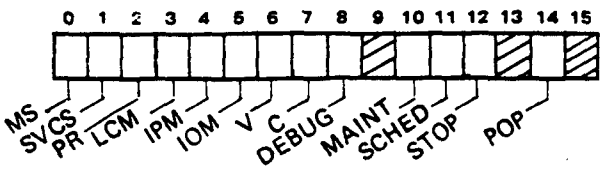
Ces registres contiennent les 16 bits poids fort d'adresses mémoire (20 bits) qui doivent être multiples de 16.

SLO contient l'adresse physique de l'origine (adresse logique 0) d'une tâche en mode esclave.

SLE définit l'adresse physique de la fin d'une tâche en mode esclave. Vu le format de cette adresse, il pointe en fait le premier des 16 derniers mots de la tâche.
De plus, la taille d'une tâche (représentée par la différence SLE - SLO) ne doit pas excéder 64 Kmots. Les registres SLO et SLE sont accessibles par les instructions RDOE et WOE. Ils sont sauvegardés et restaurés lors d'un changement de Contexte de tâches différées.
Si la différence SLE - SLO ne respecte pas la condition exposée cidessus, les 4 bits poids fort de SLE sont modifiés par le calculateur.

ST - REGISTRE D'ETAT DU PROCESSEUR DE TRAITEMENT

Ce registre regroupe les différents bits qui définissent l'état du processeur à chaque instant. Il comprend d'une part des bits appartenant au Contexte d'une tâche (registre S) d'autre part des bits qui ne font pas partie du contexte des tâches et ne participent donc pas aux changements de contexte ni immédiat, ni différé.



- les bits non explicités ne sont utilisables que par la micromachine et ne doivent jamais être modifies ni au pupitre ni par programme.

- les bits MS, SVCS, PR, V et C font partie du registre S et sont décrits à cette rubrique :

- LCM : masque des interruptions du canal LDC (1 = interruptions masquées).

- IPM : masque des interruptions interprocesseurs 05 (1 = interruptions masquées).

- IOM : masque global des interruptions de programme y compris du défaut secteur (1 = interruptions masquées).

- DEBUG : positionné à 1, le processeur est en mode mise au point. Les erreurs de parité mémoire ne sont plus interprétées comme telles mais comme des points d'arrêt. De plus la vitesse du processeur est altérée dans un rapport variant de 1 à 3 environ.

- MAINT : ce bit est normalement à 0 ; s'il a la valeur 1, le processeur passe dans un mode spécial réservé à la maintenance. Dans ce mode les erreurs de parité ne provoquent plus ni d'alarmes ni de points d'arrêt.

- SCHED : s'il est à 1, ce bit signifie que le processeur a été interrompu au cours de son passage dans le scheduler. Le bit SVCS prend pendant ce passage une autre signification :

  1 = la sauvegarde du contexte de la tâche différée abandonnée n'est pas faite
  0 = la sauvegarde a été faite

  Cette signification est temporaire ; elle ne peut être vue au pupitre mais seulement à l'examen du contexte réduit de la tâche immédiate interrompante (registre S).

- S T O P :

  = 0 le processeur est en marche programme et exécute donc des instructions.

  = 1 le processeur est à l'arrêt programme et ne traite que les appels canaux éventuels. Ce bit peut être forcé à 1 par programme (instruction SST) mais l'arrêt n'est pas instantané et le processeur continue à exécuter les instructions suivantes pendant environ 2 millisecondes.

- POP : Ce bit n'a de sens que si le processeur est en mode mise au point (DEBUG = 1) :
  - 1 : mode mise au point au pupitre. Les points d'arrêt provoquent l'arrêt programme.
  - 0 : mode mise au point programme. Les points d'arrêt provoquent des alarmes à traiter par programme.

A l'initialisation du calculateur,

- MS = 1 : mode maître
- LCM = IPM = IOM = 1 : toute interruption masquée
- STOP = 1 : arrêt programme (STOP = 0 en cas de RESTART).
- Les autres indicateurs sont à 0.

IM - MASQUE SELECTIF DES INTERRUPTIONS

Les bits 1 à 15 de ce registre permettent, quand ils sont à 1 de masquer sélectivement les niveaux d'interruptions de programme 1 à 15. Les appels masqués sont mis en attente et seront pris en compte dès leur démasquage. Par contre si un niveau est masqué au cours de son traitement, soit dans sa propre tâche immédiate soit dans une tâche de niveau supérieur, ce traitement continue jusqu'à l'acquittement de sa tâche (ACQ).

Le bit 0 de IM ne masque pas la tâche alarme ; par contre il permet de masquer le pupitre opérateur. Ce masquage n'est actif qu'en marche programme ; le pupitre opérateur est toujours opérationnel quand le calculateur est à l'arrêt.

Les masques IOM et IM agissent indépendamment l'un de l'autre ; une interruption de rang i est masquée :

- **soit si IOM = 1 quelle que soit la valeur de IM_i (masquage global),**
- **soit si IM_i** = 1 quelle que soit la valeur de IOM (masquage sélectif).

IM ne participe à aucun changement de contexte
Le masque IM peut être modifié par l'instruction XIMR
A l'initialisation du calculateur, IM est à 0.

HV - REGISTRE VECTEUR DES TACHES HARDWARE

Ce registre recense les différentes demandes d'interruption qui ont été prises en compte et dont le traitement n'est pas encore terminé ; à chaque tâche immédiate dans cet état correspond un bit à 1 dans HV ayant pour rang le numéro de priorité de cette tâche.

Parmi ces tâches, seule celle de plus forte priorité (rang dans HV le plus faible) est en cours d'exécution ; les autres sont suspendues jusqu'à la fin du traitement de la plus prioritaire.

HV peut être lu par programme (RDHV) mais ne peut être modifié.
HV ne fait pas partie du contexte des tâches et ne participe donc à aucun changement de contexte.
HV est à zéro à l'initialisation du calculateur.

3 - CLASSES D'INSTRUCTION, ADRESSAGE

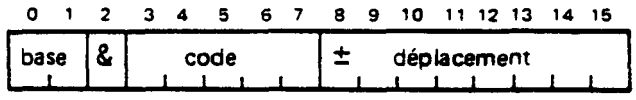
3.1 • FORMAT DES INSTRUCTIONS

Les instructions peuvent être, classées, selon les différents champs qui les composent, en 9 formats différents (à l'exception de PSR, PLR, SVC et ARM qui utilisent chacune un format propre).

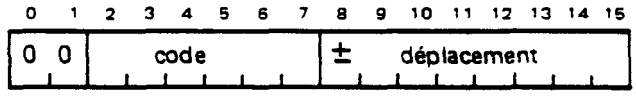
A chacun des différents formats correspond un ou plusieurs modes d'adressage spécifiques.

Notations
- base = base C, L ou W
- & = indirection
- Rsce = registre source
- Rdest = registre destination

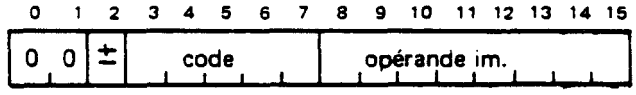
. Instructions avec opérande en mémoire (mot ou octet)
adressage base direct, indirect, indirect post-indexé



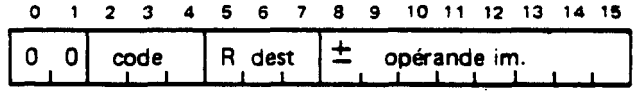
. Instructions de saut
adressage par rapport à P



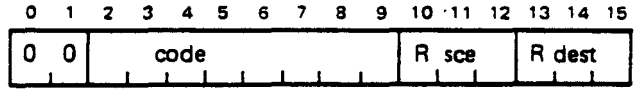
. Instructions avec opérande immédiat 9 bits
adressage immédiat



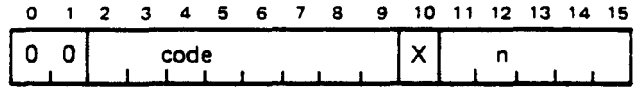
. Instructions avec opérande immédiat 8 bits
adressage immédiat



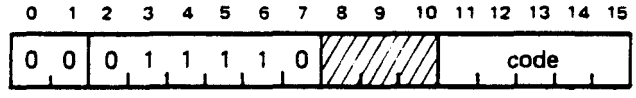
. Instructions registre-registre
8 registres adressables en tant que registres banalises



. Décalages et opérations sur bits non indexés ou indexés

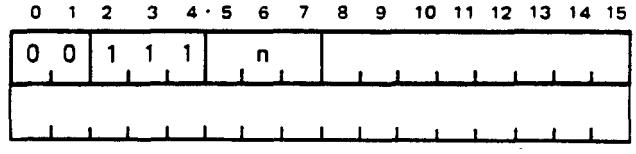


. Instructions sans indication d'opérande



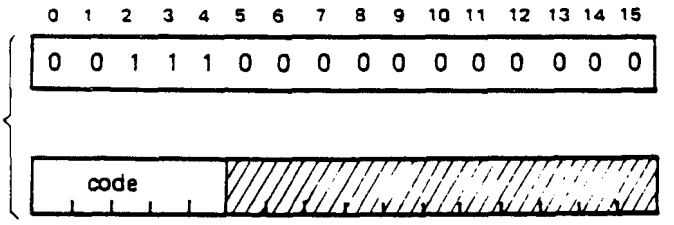
Instructions extensions (codes réservés)
exemples :
- flottant simple n = 0
- double précision n = 1
- scheduler CDA n = 7

(voir p. 4.22)



• Instructions virgule flottante sans opérande de mémoire

Ces instructions occupent deux mots de 16 bits, le premier mot identifiant l'instruction comme faisant partie de l'extension flottant



3.2 - INSTRUCTIONS AVEC OPERANDE EN MEMOIRE

Les bits 0 et 1 comportent l'indication d'une base avec le codage indiqué ci-contre. Le bit 2, lorsqu'il est à 1, indique l'adressage indirect.

base C	01
L	10
W	11

Les bits 8 à 15 comportent un deplacement compris entre - 128 et + 127. Certaines de ces instructions ont pour opérande un mot, d'autres, un octet. Elles utilisent l'adressage direct, indirect et indirect post-indexé, avec dans tous les cas l'adressage base, dont le fonctionnement est le suivant : le deplacement est additionné au contenu du registre de base indiqué par l'instruction, ce qui permet d'avoir accès à chacun des 256 mots dont l'adresse est comprise entre "valeur de la base – 128" et "valeur de la base + 127."

l Adressage direct de mot : l'opérande est le mot mémoire adresse par la base et le déplacement.

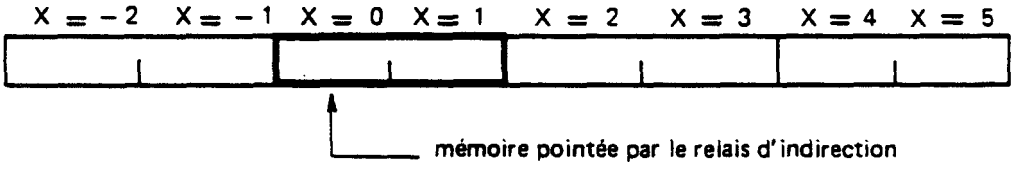
l Adressage indirect de mot : l'opérande est le mot mémoire dont l'adresse est donnée par les 15 bits poids faible d'un relais d'indirection, ce relais d'indirection étant le mot mémoire adresse par la base et le déplacement. Dans ce cas le bit 0 du relais d'indirection vaut 0. Ce mode d'adressage ne donne accès qu'a 32 K mots (adresses comprises entre SLO et SLO + 32 767).

• Adressage indirect post-indexé de mot : l'opérande est le mot mémoire dont l'adresse est donnée par la somme des 16 bits du registre X, et des 15 bits poids faible du relais d'indirection adresse par la base et le déplacement. Dans ce cas, le bit 0 du relais d'indirection vaut 1.

• Adressage direct d'octet : l'opérande est obligatoirement l'octet gauche du mot mémoire adresse par la base et le déplacement. Ce mode d'adressage ne peut être utilise que dans des cas particuliers.

• Adressage indirect d'octet : l'opérande est obligatoirement l'octet gauche du mot mémoire dont l'adresse est donnée par les 15 bits poids faible du relais d'indirection adresse par la base et le déplacement. Dans ce cas le bit 0 du relais d'indirection vaut 0. Ce mode d'adressage ne donne accès qu'aux octets gauche des mots mémoire dont l'adresse est comprise entre SLO et SLO + 32 767. Ce mode d'adressage ne peut être utilise que dans des cas particuliers.

• Adressage indirect post-indexé d'octet : dans ce cas le bit 0 du relais d'indirection vaut 1. L'opérande est l'octet gauche du mot mémoire si la valeur du registre X est paire, l'octet droit si la valeur du registre X est impaire. L'adresse du mot est donnée par la somme du registre X décalé algébriquement à droite d'une position, et des 15 bits poids faible du relais d'indirection adressé par la base et le déplacement En faisant varier l'index par pas de 1 on peut ainsi avoir accès à des octets rangés successivement en mémoire



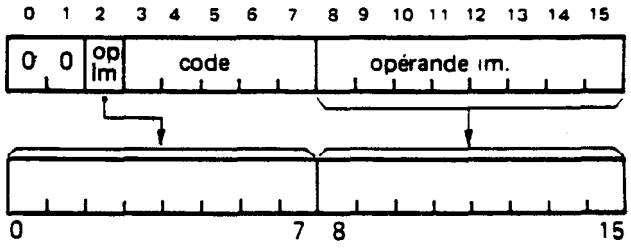
3.3 - INSTRUCTIONS DE SAUT

Ces instructions utilisent l'adressage par rapport à P ; les bits 8 à 15 comportent un déplacement qui est additionné au registre P pour obtenir l'adresse vers laquelle le saut est effectué. Au moment de l'exécution, P pointe sur l'instruction de saut ; on peut atteindre de cette manière les 128 instructions qui la précédent et les 127 instructions qui la suivent.

3.4 . INSTRUCTIONS AVEC OPÉRANDE IMMEDIAT 9 BITS

L'opérande effectif comporte 16 bits obtenus de la manière suivante :

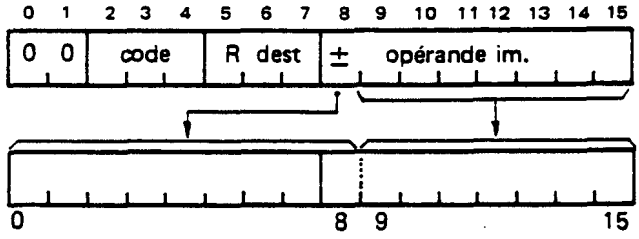
les 8 bits poids fort sont identiques au bit 2 de l'instruction, les 8 bits poids faible aux bits 8 - 15 de l'instruction



3.5 - INSTRUCTION AVEC OPERANDE IMMEDIAT 8 BITS

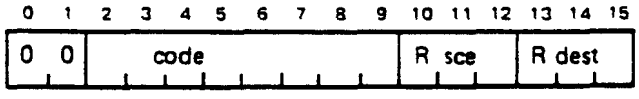
L'opérande effectif comporte 16 bits obtenus de la manière suivante :

les 9 bits poids fort sont identiques au bit 8 de l'instruction, les 7 bits poids faible aux bits 9 - 15 de l'instruction.



Le codage du registre concerné est celui utilisé dans les instructions registre-registre.

3.6 - INSTRUCTIONS REGISTRE-REGISTRE



Le codage du registre source et du registre destination est le suivant :

A	B	X	Y	C	L	W	K
000	001	010	011	100	101	110	111

Pour certaines instructions, le registre source ou le registre destination est implicite ; dans ce cas le contenu des bits correspondants de l'instruction est sans signification.

L'opération est effectuée entre les contenus du registre source et du registre destination et le résultat rangé dans le registre destination.

Il est toujours possible de spécifier le même registre comme registre source et destination.

3.7 - DÉCALAGES ET OPERATIONS SUR BITS

Les 5 bits poids faible de l'instruction indiquent un nombre de pas de décalage, ou un numéro de bit (de 0 pour A0 à 31 pour B_{15}).

Lorsque l'instruction est indexée (bit 10 = 1) les 5 bits poids faible de l'instruction sont additionnés au registre X pour donner sur 5 bits le nombre de pas ou le n° de bit effectif. Cette opération s'effectue modulo 32.

3.8 - EXTENSIONS

3.8.1 - GENERALITES

Ce format correspond à des codes ouverts pour des instructions optionnelles (flottant simple, double précision, VSS 16, CDA) ou des instructions spécifiques qui peuvent être soit programmées, soit microprogrammées.

En l'absence du microprogramme et/ou du matériel associé(s) à ces instructions, l'unité centrale simule une instruction SVC de rang 0 à 7 selon le paramètre de cette instruction (voir description détaillée chapitre 4).

En général les instructions extensions sont codées sur deux mots :

- le premier mot détermine :
  . la famille par le paramètre n sur les bits 5 à 7,
  . éventuellement une sous-famille ou une instruction déterminée sur les bits 8 à 15,
- le deuxième détermine :
  . l'opérande suivant l'un des formats standard (en particulier le format référence mémoire),
  . éventuellement le code de l'instruction comme dans un format standard.

On se référera aux manuels spécifiques pour le codage et la description de ces instructions. En particulier les familles :

- flottant simple (FFP 16),
- CDA,

sont décrites dans ce manuel.

3.8.2 - VIRGULE FLOTTANTE SIMPLE PRECISION

Le premier mot est identique pour toutes les instructions en virgule flottante; il comporte le code '3800 et sert à distinguer des autres instructions (voir codes réservés aux extensions § 4). S'il s'agit d'une instruction avec opérande en mémoire les bits 0 et 1 du second mot comportent l'indication de la base. le bit 2 indique l'adressage indirect, les bits 3 à 7 comportent le code opération et les bits 8 à 15 un déplacement.

S'il s'agit d'une instruction dont le ou les opérandes sont dans des registres, le second mot ne comporte que le code opération.

En l'absence de l'opérateur câblé, les instructions FLD, FST, FABS, FNEG, FIX, FCAZ, FCMZ sont exécutées en standard par microprogramme sur certains modèles d'unité centrale (40) et (65).

Selon l'option choisie ces instructions sont soit exécutées directement par le calculateur (option flottant hardware) soit exécutées par un ensemble de sous-programmes appelés par le calculateur lorsqu'il rencontre une instruction virgule flottante (option flottant programmé).

L'ensemble des sous-programmes occupe environ 500 mots de mémoire.

Les routines sont réentrantes, de telle manière qu'une tâche effectuant des calculs en virgule flottante puisse être interrompue par une autre tâche effectuant elle aussi des calculs en virgule flottante.

II n'y a donc pas à modifier les programmes lorsqu'on utilise l'une ou l'autre option. Cependant avec l'option flottant programmé on doit s'assurer que la pile pointée par le registre K comporte suffisamment de mots mémoire (23 mots) pour permettre le déroulement des sous-programmes.

Les opérations sont faites d'une manière interne sur 32 bits pour l'option programmée et sur 27 bits (addition-soustraction) ou 48 bits (multiplication division) pour l'option câblée. Comme il est tenu compte de l'arrondi sur ces bits supplémentaires pour le résultat des opérations arithmétiques il peut y avoir une différence non significative, parce qu'inférieure à la précision des calculs, entre les résultats obtenus avec l'une et l'autre option.

4.1 - DESCRIPTION DES INSTRUCTIONS STANDARD

Dans les pages suivantes on trouvera la description de l'ensemble des instructions SOLAR 16.
L'existence ou le fonctionnement de certaines instructions peut dépendre de la présence de certaines "options".

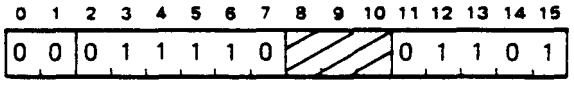
Type de SOLAR 16

Options	04/05	30	35	40	65	70	75
DRPS	0	1	1	X	1	1	1
Scheduler	X	1	1	X	1	1	1
FFP16	0	X	X	X	X	X	X
DAP16	0	X	X	X	X	X	X
VSS16	0	0	0	0	X	X	X
FFM16	0	0	0	0	X	1	X
CDA16	0	0	?	0	1	1	1
ISP16	0	0	?	0	0	1	0

0 - inexistant
1 - standard
X - optionnel
? - voir remarques

Remarques :

Sur SOLAR 16-35 les instructions RCDA et WCDA sont disponibles en standard. Les autres instructions de l'option CDA sont inexistantes.
Sur SOLAR 16-35 les instructions de l'option ISP16 sont disponibles en standard à l'exception de l'instruction XCTX.
Les instructions des options DAP16, VSS16 et FFM16 sont décrites dans les notices correspondantes à ces modules.
Les instructions de l'option ISP16 sont décrites dans le chapitre des instructions spéciales (par. 4.2).
Toutes les instructions privilégiées exécutables en mode maître sont aussi exécutables en mode privilégié sur SOLAR 16-70, sauf l'instruction XCTX.
Pour ce qui est du mécanisme DRPS, le mode privilégié est assimilable au mode esclave (MS = 0, SVCS = 0) dans le fonctionnement des instructions sur pile K.



code : 1 E 0 D

Opération effectuée : permet à la tâche qui exécute cette instruction de reconnaître le sous-niveau le plus prioritaire en attente :
- l'indicateur V est mis à 1 si un sous-niveau de type normal est en attente,
- l'indicateur C est mis à 1 si seuls des sous-niveaux de type exception sont en attente,
- le numéro du plus prioritaire des sous-niveaux en attente est chargé dans le registre X :
  - 0 à 15 pour un sous-niveau normal
  - 0 à 47 pour un sous-niveau exception.

Lorsqu'il n'y a pas d'interruption en attente :
- l'indicateur V est positionné à 0
- l'indicateur C est positionné à 0
- le registre X n'est pas modifié.

Dans une tâche différée cette instruction a comme seul effet de remettre à 0 les indicateurs V et C.

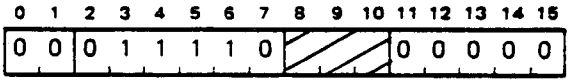
Sont modifiés : X, ST

Indicateurs :

V	C	
0	0	pas de sous-niveau
1	0	sous-niveau normal
0	1	sous-niveau exception

Alarmas possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : ACK ne doit pas être utilisée dans la tache alarme, ni dans la tâche défaut secteur.



Mode : maître

Code : 1 E 0 0

Opération effectuée : L'instruction ACQ annule la tâche immédiate en cours en remettant à zéro le bit du registre HV correspondant à cette tâche et provoque le lancement de la tâche interruption ou de la tâche différée la plus prioritaire en attente.

Les masques IOM et IPM sont mis à zéro.

La commutation de tâche se fait par un changement de contexte partiel (C, K, P, S) qui sauvegarde le contexte de la tâche abandonnée et restitue le contexte de la tâche reprise.

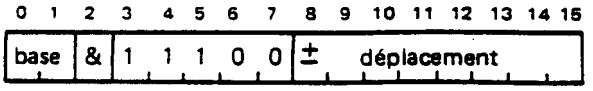
ACQ peut être aussi utilisée dans une tâche différée (dans ce cas le registre HV n'est pas modifié) pour provoquer un changement de contexte après modification par programme des files ASTF, ESTF et RSTF (mais surtout pas de NS I).

En l'absence de l'option scheduler, cette instruction ne doit pas être utilisée dans une tâche différée.

Sont modifiés :
- la PST de la tâche exécutant l'ACQ,
- le contexte partiel C, K, P, S (ACQ dans une tâche hardware)
- le contexte complet (ACQ dans une tâche différée)
- HV, ST.

Indicateurs : ceux du nouveau contexte, IOM, IPM

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction privilégiée



Option : scheduler

Mode : maître

Code :

	Direct	Indirect
base C	5C__	7C__
base L	9C__	BC__
base W	DC__	FC__

Adressage : mémoire (1er mot du sémaphore)

Opération effectuée : la seule différence entre l'opération effectuée par ACT sur un sémaphore de synchronisation (bit 0 du premier mot du sémaphore à zéro) et sur un sémaphore d'appel (bit 0 du premier mot du sémaphore à un) est le traitement de la file paramètre :

dans le cas d'un sémaphore d'appel, le bit de la file paramètre dont le rang ($\geqslant$ 0 et $\leqslant$ 32 767) est égal à la valeur du registre Y est mis à 1. Ensuite, dans les deux cas le compteur du sémaphore est augmenté de 1 :

- si après cette modification le compteur a une valeur positive la tâche qui effectue l'instruction ACT se poursuit ;
- si après cette modification le compteur à une valeur nulle ou négative :

  le bit de la file ESTF dont le rang est égal au n° de la tâche en attente sur le sémaphore est mis à 1 (la tache est démasquée),
- si la tâche qui effectue l'instruction ACT est plus prioritaire elle se poursuit normalement, sinon le scheduler effectue un changement de contexte au prof it de la tâche démasquée.

Les indicateurs IOM, IPM sont remis à 0 (avant passage éventuel dans le scheduler).

I merviennent : le sémaphore adressé
NS

Sont modifiés : le sémaphore adressé
NS et tous les registres s'il y a changement de contexte
ST

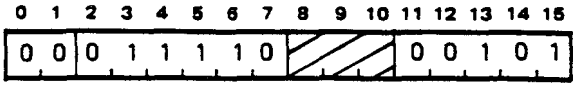
Indicateurs : IOM, IPM, plus le changement de contexte éventuel

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (06)
- parité
- instruction privilégiée
- instruction optionnelle inexistante

ATTENTION : le rang du bit mis à 1 dans la file paramètre d'un sémaphore d'appel n'est pas égal au n° de la tâche qui effectue l'instruction ACT, mais au contenu du registre Y, et peut avoir une valeur supérieure à 128.

activate debug (ACTD)

ACTD



Code : 1E05

Opération effectuée : permet de déclencher l'alarme numéro 9. Cette alarme provoque le lancement de la tâche immédiate. 0 comme toutes les alarmes.

On peut ainsi appeler un programme de mise au point (AID par exemple).

Sont modifiés : les registres du contexte partiel (C, K, P, S)

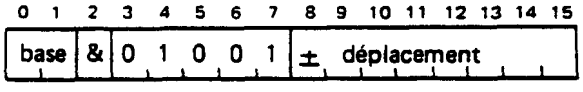
Indicateurs : ceux du nouveau contexte

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

add (AD)

AD



Code :

	direct	indirect
base C	49	69
base L	89	A9
base W	C9	E9

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adressé est additionné au registre A.
Les indicateurs V et C sont positionnés de la manière indiquée aux § 1.1 et 1.2

Interviennent : . le mot mémoire adressé
. A

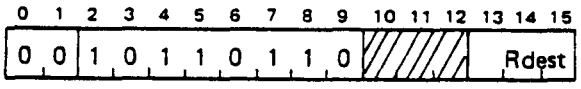
Sont modifiés : . A
. ST

Indicateurs : V : débordement addition
C : report addition

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

add carry to register (ADCR)

ADCR



Code : 2D8_

Adressage : registre - registre

Opérations effectuée : la valeur de l'indicateur C est additionnée au registre destination.

Les Indicateurs V et C sont position& de la manière indiquée aux § 1.1 et 1.2. On peut ainsi effectuer des additions en double longueur, le report de l'addition des poids faibles étant additionné aux poids forts.

Intervient : un des registres A B X Y C LWK

Sont modifiés : . un des registres A B X Y C LWK.

. ST

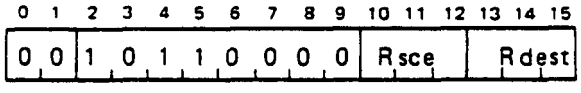
Indicateurs : V : débordement addition
C : report addition

Alarmas possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

add registers (ADR)

ADR



Code : 2C0_

Adressage : registre - registre

Opération effectuée : le contenu du registre source est additionné au registre destination.

Les indicateurs C et V sont positionnés de la manière indiquée aux § 1.1 et 1.2. On peut ainsi :

- additionner un quelconque des huit registres A B X Y C L W K à un autre de ces registres,
- doubler le contenu d'un de ces registres en précisant le même registre comme source et destination.

Interviennent : deux des registres A B X Y C L W K.

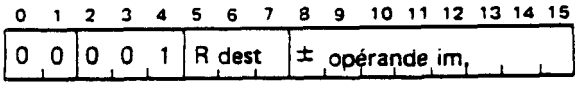
Sont modifiés : • un des registres A B X Y C L W K
• ST

Indicateurs : V : débordement addition
C : rapport addition

Alarmes possibles : -protection mémoire (05)
-mémoire inexistante (05)
-parité

add register immediate (ADRI)

ADRI



Code : 08__ pour A
09__ pour B
0A__ pour X
0B__ pour Y
0C__ pour C
0D__ pour L
0E__ pour W
0F__ pour K

Opération effectuée : la valeur de l'opérande immédiat est additionnée au registre destination.
L'opérande immédiat est étendu sur 16 bits, ses 9 bits poids fort étant identiques au bit 8 de l'instruction, ses 7 bits poids faible étant identiques aux bits 9 à 15 de l'instruction. Les indicateurs C et V sont positionnés de la manière indiquée aux § 1.1 et 1.2. On peut ainsi additionner à l'un des huit registres A B X Y C L W K une valeur comprise entre - 128 et + 127

Interviennent : . l'opérande immédiat (sur 16 bits)
. le registre précisé dans l'instruction

Sont modifiés : . le registre précisé dans l'instruction
. ST

Indicateurs : .V débordement addition
. C report addition

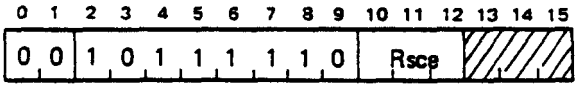
Alarmas possibles : - protection mémoire (05)
-mémoire inexistante (05)
- parité

ATTENTION : cette instruction est la seule qui utilise un opérande immédiat sur 8 bits (compris entre + 127 et - 128).

---

add register to P (ADRP)

ADRP



Code : 2F8_

Adressage : registre - registre

Opération effectuée : le contenu du registre source est additionne au registre P (qui contient l'adresse de l'instruction ADRP elle même).

On peut ainsi :

- réaliser un aiguillage en fonction d'un indice
- remplacer l'instruction BR dans des programmes translatables.

Interviennent : un des registres A B X Y C L W K
P

Est modifié : P

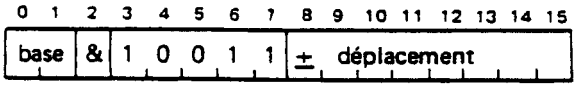
Indicateurs : non modifies

Alarmas possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : la valeur de P additionnée est l'adresse de l'instruction ADRP elle-même.

and (AND)

AND



Code :

	direct	indirect
base C	53__	73__
base L	93__	B3__
base W	D3__	F3__

Adressage : mémoire (mot)

Opération effectuée : le résultat de l'intersection logique des 16 bits du registre A et du mot mémoire adressé est rangé dans le registre A.
(Intersection : l'intersection de 1 et 1 vaut 1, les Intersections de 1 et 0, 0 et 1, 0 et 0 valent 0).

Interviennent : . le mot mémoire adressé
. A

Est modifié : A

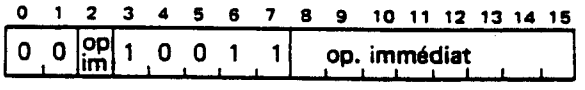
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

and immediate (ANDI)

ANDI



Code :

bit 2 = 0	13__
bit 2 = 1	33__

Adressage : immédiat (9 bits)

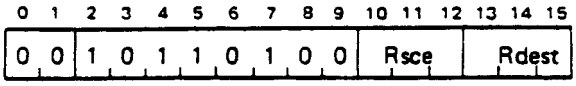
Opération effectuée : le résultat de l'intersection des 16 bits du registre A et de l'opérande immédiat est rangé dans le registre A. L'opérande immédiat est étendu sur 16 bits, ses huit bits poids fort étant identiques au bit 2 de l'instruction, ses huit bits poids faible étant identiques aux bits 8 à 15 de l'instruction.
On peut ainsi effectuer une intersection entre A et un opérande immédiat compris entre '0000 et '00FF ou entre 'FF00 et 'FFFF.

Interviennent : . l'opérande immédiat (sur 16 bits)
. A

Est modifié : A

Indicateurs : non modifiés

Alarmes possibles :-protection mémoire (05)
-mémoire inexistante (05)
-parité



Code : 2D9_

Adressage : registre - registre

Opération effectuée : le résultat de l'intersection logique des 16 bits du registre source et du registre destination est rangé dans le registre destination.

On peut ainsi faire l'intersection logique de deux quelconques des huit registres A B X Y C L W K.

Interviennent : deux des registres A B X Y C L W K.

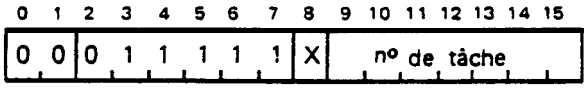
Est modifié : un des registres A B X Y C L W K.

Indicateurs : non modifiés

Alarmas possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ARM

arm (ARM)



Option : scheduler

Mode : maître

Code : 1F__

Adressage : indexation possible

Opération effectuée : le bit de la file ASTF correspondant au numéro de la tâche indiqué par l'instruction est mis à 1. Si l'instruction n'est pas indexée le numéro de la tâche armée est contenu dans les bits 9 à 15 de l'instruction. Si l'instruction est indexée ce numéro est la somme des bits 9 à 15 de l'instruction et des 16 bits du registre X. Les masques IOM et IPM sont remis à zéro.

Lorsque ARM est rencontré dans une tâche immédiate cette tâche immédiate se poursuit.
Lorsqu'une tâche différée arme une tâche différée plus prioritaire, le scheduler provoque un changement de contexte pour lancer la tâche différée qui vient d'être armée.

Intervient : le n° de tâche indiqué dans l'instruction

Sont modifiés : - la file ASTF
- NS et tous les registres s'il y a changement de contexte
- S T

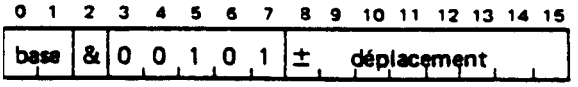
Indicateurs : IOM, IPM, plus le changement de contexte éventuel

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction privilégiée
- instruction optionnelle inexistante

ATTENTION : lorsqu'il y a indexation, le numéro de tâche est pris modulo 128

branch (BR)

BR



Code :

	direct	indirect
base C	45__	65__
base L	85__	A5__
base W	C5__	E5__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adressé est chargé dans le registre P.

On peut ainsi en chargeant dans P un mot mémoire qui contient l'adresse vers laquelle le branch est effectué avoir accès à des instructions qu'on ne peut pas atteindre par JMP.

Intervient : le mot mémoire adressé

Est modifié : P

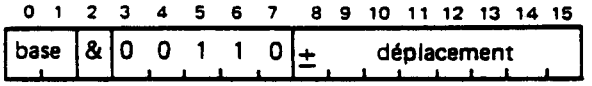
Indicateurs : non modifies

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (08)
- parité

ATTENTION : une instruction BR indirecte fait intervenir deux constantes adresse : la première (éventuellement indexée) fournit l'adresse de la seconde qui est chargée dans P.

branch subroutine (BSR)

BSR



Code :

	direct	indirect
base C	46__	66__
base L	86__	A6__
base W	C6__	E6__

Adressage : mémoire (mot)

Opération effectuée : le contenu du registre K est augmenté de 1, puis l'adresse de l'instruction qui suit le BSR est rangée dans le mot mémoire pointé par K.

Enfin le registre P est chargé par le contenu du mot mémoire adressé par l'instruction BSR.

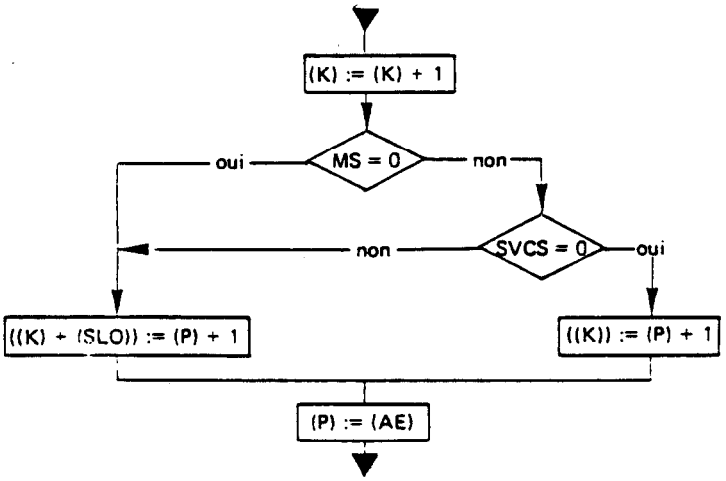
On peut ainsi appeler un sous-programme en sauvegardant le point de retour :

- le mot mémoire chargé dans P contient l'adresse du sous-programme appelé
- l'adresse de retour rangée en mémoire sera reprise à la fin du sous-programme par une instruction RSR (la fonction de RSR est complémentaire de celle de BSR).
- les appels imbriqués de sous-programmes sont possibles grâce à l'addition de 1 au registre K effectuée par l'instruction BSR, cette évolution de K pour la sauvegarde de l'adresse de retour étant de plus homogène avec l'évolution de K pour les sauve gardes effectuées par SVC et PSR.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :

- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue.
- si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.

De plus, si l'instruction est exécutée en mode esclave, l'adresse de retour qui est empilée est également une adresse relative à SLO.



Interviennent : le mot mémoire adressé
K, ST

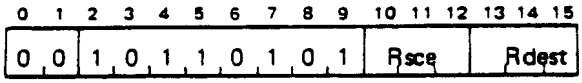
Sont modifiés : P
K
le mot mémoire pointé par K

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : - K pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande : ici le dernier point de retour rangé.
- le programme peut passer en mode esclave à la suite d'une alarme.

clear selective registers (CLSR) — CLSR



Code : 2D4_

Adressage : registre - registre

Opération effectuée : le résultat de l'intersection logique des 16 bits du registre destination et du complément logique du contenu du registre source est rangé dans le registre destination.

On peut ainsi mettre à zéro sélectivement les bits du registre destination correspondant aux bits à un du registre source. Le registre source et le registre destination sont deux des huit registres A B X Y C L W K.

Interviennent : deux des registres A B X Y C L W K

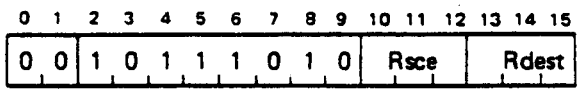
Est modifié : un des registres A B X Y C L W K

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

complement register (CMR) — CMR



Code : 2E8_

A d - : registre - registre

Opération effectuée : le registre destination est chargé par le complément logique du contenu du registre source (complément logique : chaque bit valant 0 devient 1, chaque bit valant 1 devient 0).

On peut ainsi :

- effectuer un transfert avec complémentation logique entre deux quelconques des huit registres A B X Y C L W K.
- effectuer la complémentation logique d'un de ces huit registres, le registre destination étant alors le registre source.

Intervient : un des registres A B X Y C L W K

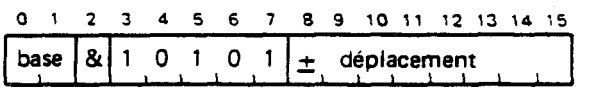
Est modifié : un des registres A B X Y C L W K

Indicateurs : non modifies

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

compare (CP)

CP



Code

	direct	indirect
base C	55__	75__
base L	95__	B5__
base W	D5__	F5__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adressé est comparé algébriquement avec le contenu du registre A, et les indicateurs V et C sont positionnés en fonction de cette comparaison. On peut tester le résultat de la comparaison de la manière suivante

JL	saut si	A	<	mémoire
JE	saut si	A	=	mémoire
JG	saut si	A	>	mémoire
JGE	saut si	A	≥	mémoire
JNE	saut si	A	≠	mémoire
JLE	saut si	A	≤	mémoire

Interviennent : . A
. le mot mémoire adresse

Est modifié : ST

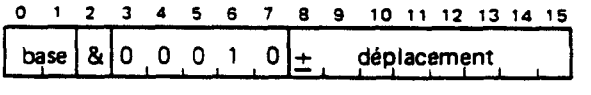
Indicateurs :

V	C	
0	1	A < mot mémoire
1	0	A = mot mémoire
0	0	A > mot mémoire

Alarmes possibles: - protection mémoire (05)
- mémoire inexistante
- parité

compare byte (CPBY)

CPBY



Code

	direct	indirect
base C	42__	62__
base L	82__	A2__
base W	C2__	E2__

Adressage : mémoire (octet)

Opération effectuée : le contenu de l'octet mémoire adresse (sans extension du signe) est comparé algébriquement avec les 16 bits du registre A ; les indicateurs V et C sont positionnés en fonction de cette comparaison. On peut tester le résultat de la comparaison qui n'a de sens que si l'octet gauche de A est nul, de la manière suivante :

JL	saut si	A	<	octet
JE	saut si	A	=	octet
JG	saut si	A	>	octet
JGE	saut si	A	≥	octet
JNE	saut si	A	≠	octet
JLE	saut si	A	≤	octet

L'octet gauche de A étant nul la comparaison doit être interprétée comme une comparaison absolue.

Interviennent : . A
. l'octet mémoire adresse

Est modifié : ST

Indicateurs :

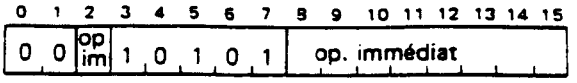
V	C	
0	1	A < octet mémoire
1	0	A = octet mémoire
0	0	A > octet mémoire

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION
- pour que la comparaison ait une signification, l'octet gauche de A doit être nul
- s'il n'y a pas indexation, l'octet adressé est l'octet gauche d'un mot.

compare immediate (CPI)

CPI



Code .	bit 2 = 0	15__
	bit 2 = 1	35__

Adressage : immédiat (9 bits)

Opération effectuée : l'opérande Immédiat est comparé algébriquement avec le contenu du registre A et les indicateurs V et C sont positionnés en fonction de cette comparaison. L'opérande immédiat est étendu sur 16 bits, ses 8 bits poids fort étant identiques au bit 2 de l'instruction, ses 8 bits poids faible étant Identiques aux bits 8 à 15 de l'instruction On peut ainsi comparer le registre A à une valeur algébrique comprise entre - 256 et + 255 ou à un caractère ASCII 8 bits et tester le résultat de la comparaison de la manière suivante :

JL saut si A < op. im.
JE saut si A = op. im.
JG saut si A > op. im.
JGE saut si A ≥ op. im.
JNE saut si A ≠ op. im.
JLE saut si A ≤ op. im.

Interviennent : . l'opérande immédiat (sur 16 bits)
. A

Est modifié : ST

Indicateurs :

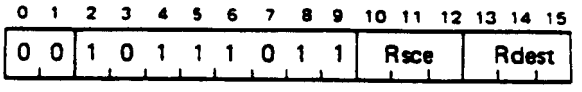
v	c	
0	1	A < opérande immédiat
1	0	A = opérande immédiat
0	0	A > opérande immédiat

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

compare registers (CPR)

CPR



Code : 2E__

Adressage : registre registre

Opération effectuée : le contenu du second registre est comparé algébriquement au contenu du premier registre et les indicateurs V et C sont positionnés en fonction de cette comparaison.
On peut ainsi comparer deux des huit registres A B X Y C L W K et tester le résultat de la comparaison de la manière suivante :

JL saut si 2d reg. < 1er reg.
JE saut si 2d reg. = 1er reg.
JG saut si 2d reg. > 1er reg.
JGE saut si 2d reg. ≥ 1er reg.
JNE saut si 2d reg. ≠ 1er reg.
JLE saut si 2d reg. ≤ 1er reg.

interviennent deux des registres A B X Y C L W K

Est modifié ST

indicateurs

v	vc	
0	1	2d reg. < 1er reg.
1	0	2d reg. = 1er reg.
0	0	2d reg. > 1er reg.

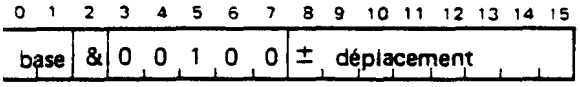
Alarmes possibles .-protection mémoire (05)
-mémoire inexistante (05)
-parité

ATTENTION : mettre comme second registre le premier terme de la comparaison

Exemple :
CPR B,A
JG ASUPB (A > B)

compare to zero (CPZ)

CPZ



Code :

	direct	Indirect
base C	44__	64__
base L	84__	A4__
base W	C4__	E4__

Adressage : mémoire (mot)

Opération effectuée : les indicateurs V et C sont positionnés selon le résultat de la comparaison algébrique de la mémoire adressée et de zéro.

On peut ainsi comparer la valeur d'une mémoire par rapport à zéro sans avoir à passer par les registres, et tester le résultat de la manière suivante

JL	saut si	mémoire C 0
JE	saut si	mémoire = 0
JG	saut si	mémoire > 0
JGE	saut si	mémoire ⩾ 0
JNE	saut si	mémoire ≠ 0
JLE	saut si	mémoire ⩽ 0

Intervient : le mot mémoire adressé

Est modifié :ST

Indicateurs :

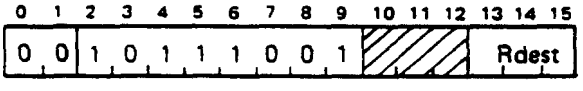
v	c	
0	1	mot mémoire < 0
1	0	mot mémoire = 0
0	0	mot mémoire > 0

Alarmes possibles :-protection mémoire (05)
-mémoire inexistante (05)
-parité

---

compare register to zero (CPZR)

CPZR



Code : 2E4_

Adressage : registre . registre

Opération effectuée : le contenu du registre est comparé algébriquement à zéro et les indicateurs V et C sont positionnés en fonction de cette comparaison.

On peut tester le résultat de cette comparason de la manière suivante

JL	saut si	reg. < 0
JE	saut si	reg. = 0
JG	saut si	reg. > 0
JGE	saut si	reg. ⩾ 0
JNE	saut si	reg. ≠ 0
JLE	saut si	reg. ⩽ 0

Intervient : un des registres A B X Y C L W K

Est modifié : ST

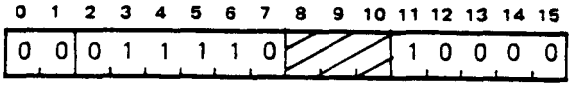
Indicateurs :

v	C	
0	1	reg. < 0
1	0	reg. = 0
0	0	reg. > 0

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : le registre indiqué est le premier terme de la comparaison

Exemple 0
CPZR B
JG BPOS (B > 0)



Mode : maître et mise au point

Code : 1 E 1 0

Opération effectuée : recherche les points d'arrêt (mots en parité inverse) en décrémentant les adresses depuis l'adresse contenue dans Y jusqu'à 0.

Si l'option DRPS est présente, elle est activée. Y contient alors une adresse relative à SLO et peut provoquer une alarme protection mémoire.

L'instruction se termine à la rencontre d'une parité inverse ou d'une mémoire inexistante (05) avec positionnement des indicateurs et avec dans Y l'adresse de ce mot.

A défaut, l'instruction se termine avec le registre Y et les indicateurs à 0.

Interviennent : Y, SLO, SLE, le segment mémoire [(SLO), (Y) + (SLO)]

Sont modifiés : Y, ST

Indicateurs :
V = 1 mémoire inexistante 05
C = I parité mémoire

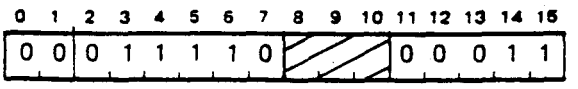
Alarmes possibles :
- mémoire inexistante 05
- protection mémoire [(Y) + (SLO) > (SLE)] 05
- petite mémoire
- instruction privilégiée

Interruptibilité : cette instruction est interruptible après chaque traitement de mot.

ATTENTION :
- exécutée hors du mode mise au point, les points d'arrêt provoquent des alarmes parité.
- le programme peut passer en mode esclave à la suite d'une alarme.

discover bit (DBT)

DBT



Code . 1E03

Opération effectuée : s'il existe au moins un bit à 1 dans les registres A et B, considérés comme un registre unique dont les bits sont numérotés de 0 à 31, le rang du bit à 1 le plus à gauche est charge dans le registre X.

S'il n'y a aucun bit à 1 le registre X n'est pas modifié et l'indicateur C est positionné à 1, alors qu'il est positionné à 0 dans le cas contraire.

Interviennent : A, B

Sont modifiés : • X (si un des bits de AB est à 1)
• ST

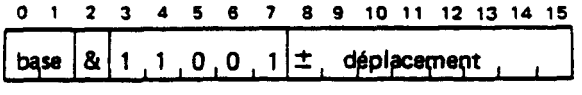
Indicateurs :

V	C	
0	0	un des bits de AB est à 1
0	1	aucun bit de AB n'est à 1

Alarmes possibles: - protection mémoire (05)
- mémoire inexistante (08)
- parité

decrement (DC)

DC



code :

	direct	indirect
base C	59__	79 __
base L	99__	B9 __
base W	D9__	F9 __

Adressage : mémoire (mot)

Opération effectuée : la valeur 1 est soustraite à la mémoire adressée. Les indicateurs V et C sont positionnés selon le résultat de la comparaison de la valeur et de zéro.
On peut ainsi diminuer de 1 la valeur d'une mémoire sans avoir à passer par les registres, et tester ensuite le résultat obtenu sans avoir à faire de comparaison :

JL saut si mémoire < 0
JE saut si mémoire = 0
JG saut si mémoire > 0
JGE saut si **mémoire** ≥ 0
JNE saut si **mémoire** ≠ 0
JLE saut si **mémoire** ≤ 0

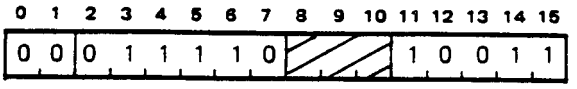
Intervient : le mot mémoire adressé

Sont modifiés : - le mot mémoire adressé
- ST

Indicateurs :

V	C	
0	1	mot mémoire < 0
1	0	mot mémoire = 0
0	0	mot mémoire > 0

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Mode : maître

Code : 1E13

Opération effectuée : inhibe les interruptions de programme y compris le défaut secteur mais à l'exclusion de l'interruption inter-processeurs.

Les interruptions suivantes seront mémorisées et prises en compte par le démasquage correspondant (cf. EIT).
Le masque sélectif IM n'est pas modifié. Ce masque n'affecte pas les canaux.

Est modifié : ST

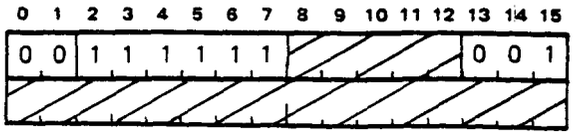
Indicateurs : IOM

Alarmes possibles :
- mémoire inexistante
- protection mémoire
- parité mémoire
- instruction privilégiée

Nota : Sur les modèles 16-65P, 16-75P ou 16-70 DIT inhibe aussi l'interruption inter-processeurs (IPM = 1).

discover and reset bit in memory (DRBM)

DRBM



Option : CDA16 (65) et (70)

Code : premier mot 3F01
deuxième mot 0000

Opération effectuée : recherche le rang du premier bit à 1 d'une file de bits en mémoire

l'accumulateur contient l'adresse du mot origine de la file (bits 0 à 15 de la chaîne).
le registre X contient le rang initial à partir duquel la recherche sera faite. Le contenu de X peut être négatif ; la recherche commencera alors à une adresse inférieure au contenu de l'accumulateur.
le registre Y contient le rang du premier bit n'appartenant plus à la file, à partir duquel la recherche est arrêtée.
la recherche est faite par adresses croissantes, des poids forts vers les poids faibles.
si la recherche est positive, le registre X est chargé par le rang du 1er bit à 1 rencontré, le bit correspondant est mis à zéro et les indicateurs V et C sont mis à zéro.
si la recherche est négative ou si initialement (X) ⩾ (Y), le registre X est chargé par Y
l'indicateur C est mis à 1 et V à 0.
l'instruction est interruptible entre chaque mot y compris par le défaut secteur.

En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

Interviennent : A, X, Y, la file de bits pointée par A

Sont modifiés : X, ST, le bit de rang (X) de la file

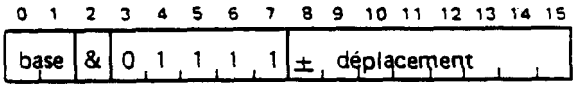
Indicateurs : V = 0 et C = 0 bit à 1 rencontré
V = 0 et C = l bit à 1 non rencontré

Alarmer possibles : - mémoire inexistante
- protection mémoire
- parité mémoire

Interruptibilité : l'instruction est interruptible entre chaque mot y compris par le défaut secteur.

divide (DV)

DV



Option — cette instruction est soit microprogrammée, soit câblée.

Code

	direct	indirect
base C	4F__	6F__
base L	8F__	AF__
base W	CF__	EF__

Adressage — mémoire (mot)

Opération effectuée : le contenu des registres A et B est divisé par le contenu de la mémoire adressée.

Avant la division, le dividende est sur 32 bits sous forme algébrique, les poids forts étant dans A et les poids faibles dans B.

Après la division le quotient occupe les 16 bits de A sous forme algébrique, et le reste les 16 bits de B sous forme algébrique. La division est faite de telle manière que le reste, s'il n'est pas nul. est de même signe que le dividende.

Quand les valeurs relatives du dividende et du diviseur sont telles que le quotient ne peut pas s'exprimer sous la forme d'un nombre de 16 bits en complément à deux (division impossible) l'indicateur V est positionné à 1 ; le contenu des registres A et B est alors sans signification et dépend de la valeur des opérandes et de l'option division.

On peut ainsi :

- diviser un nombre algébrique de 16 bits, après l'avoir mis en place dans A et B, par un nombre algébrique de 16 bits, et obtenir un résultat sur 16 bits.
- faire suivre immédiatement une multiplication par une division, le résultat de la multiplication étant alors en place dans A et B pour la division.
- diviser un nombre algébrique en double longueur par un nombre algébrique de 16 bits et obtenir la partie entière du résultat puis les parties fractionnaires en redivisant les restes successifs par le même quotient. La division étant algébrique, on ne pourra obtenir que 15 bits supplémentaires à chaque fois pour éviter les résultats non significatifs dûs à des débordements.

Interviennent :
. A
. B
le mot mémoire adressé

Sont modifiés : A, B, ST

Indicateurs :

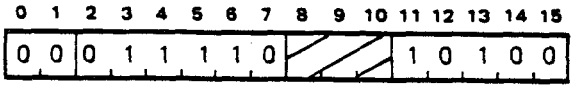
V	C	
0	0	division correcte
1	0	division impossibl

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : dans le cas d'un dividende de 16 bits, charger ce dividende dans A puis faire un décalage arithmétique (SARD 16) pour le mettre en place dans A et B en complément à deux (avec extension dans les 16 bits de A du bit 0 de B).

enable interrupt (EIT)

EIT



Mode : maître

Code : 1 E 1 4

Opération effectuée : réactivation des interruptions de programme y compris le défaut secteur mais à l'exclusion de l'interruption inter-processeurs.

Les interruptions en attente et non masquées par IM sont alors prises en compte
Le masque IM n'est pas modifié.

Est modifié : ST

Indicateurs : IOM

Alarmes possibles : - mémoire inexistante (05)
- protection mémoire (03)
- parité mémoire
- instruction privilégiée

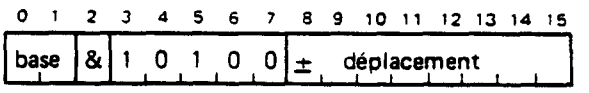
ATTENTION : les instructions ACQ. ARM, QUIT, RQST, RLSE, ACT, WAIT exécutent implicitement les mêmes fonctions de démasquage que EIT, l'indicateur IOM est également remis à zéro.

- IOM est mis à 1 au passage dans la tâche défaut secteur et ne doit pas y être remis à zéro.
- Note : sur les modèles 16-65P, 16-75P ou 16-70, EIT démasque l'interruption interprocesseur IPM = 0.

---

exclusive or (EOR)

EOR



Code :

	direct	indirect
base C	54__	74__
base L	94__	B4__
base W	D4__	F4__

Adressage : mémoire (mot)

Opération effectuée : le résultat de la disjonction des 16 bits du registre A et du mot mémoire adressé est rangé dans le registre A (disjonction la disjonction de 1 et 0, 0 et 1 vaut 1, la disjonction de 0 et 0, 1 et 1 vaut O).

Interviennent : . le mot mémoire adresse
. A

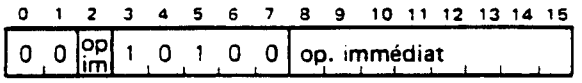
Est modifié : A

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (03)
- mémoire inexistante (05)
- parité

exclusive or immediate (EORI)

EORI



Code

bit 2 = 0	14__
bit 2 = 1	34__

Adressage : Immédiat (9 bits)

Opération effectuée : le résultat de la disjonction des 16 bits du registre A et de l'opérande immédiat est rangé dans le registre A. L'opérande immédiat est étendu sur 16 bits, ses huit bits poids fort étant identiques au bit 2 de l'instruction, ses huit bits poids faible étant identiques aux bits 8 à 15 de l'instruction.

On peut ainsi effectuer une disjonction entre A et un opérande immédiat compris entre '0000 et '00FF ou entre 'FF00 et 'FFFF.

Interviennent : . l'opérande immédiat (sur 16 bits)
. A

Est modifié : A

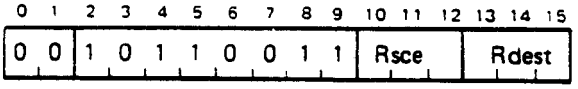
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

exclusive or registers (EORR)

EORR



Code : 2C$_$

Adressage : registre registre

Opération effectuée : le résultat de la disjonction des 16 bits du registre source et du registre destination est chargé dans le registre destination.

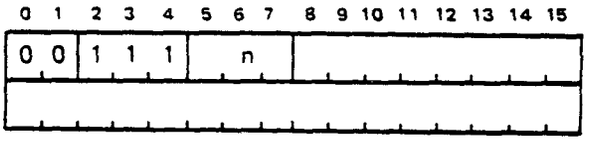
On peut ainsi faire la disjonction de deux quelconques des huit registres A B X Y C L W K.

interviennent : deux des registres A B X Y C L W K

Est modifié : un des registres A B X Y C L W K.

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Codes : (pour le premier des deux mots) :

38__
39__
3A__
3B__
3C__
3D__
3E__
3F__

Opération effectuée : huit codes ont été réservés de manière à pouvoir coder sur deux mots de 16 bits huit groupes d'instructions optionnelles.

Ces groupes d'instructions sont réalisés soit par microprogrammation soit par programmation.

Si un groupe d'instructions est microprogrammé les instructions correspondantes sont exécutées directement par le calculateur.

Si un groupe d'instructions n'est pas microprogrammé le code correspondant à ce groupe d'instructions provoque l'appel du programme qui assure les mêmes fonctions.

Cet appel est fait de la manière suivante

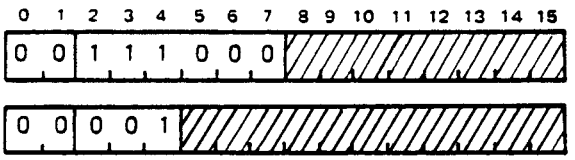
- le registre K est augmenté de 1 puis le contenu du registre X est rangé dans le mot mémoire pointé par K ;
- le registre P est augmenté de manière à pointer sur le mot mémoire qui suit les deux mots de 16 bits occupés par l'instruction ;
- les opérations équivalentes à l'exécution d'une instruction "SVC n" sont effectuées n étant pris dans les bits 5 à 7 du premier mot de l'instruction.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue
- si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.

ATTENTION : Des groupes sont utilisés par des options :
- n = 0 utilisé par l'option virgule flottante câblée ou programmée,
- n = 1 utilisé par l'option double précision (DAP 16),
- n = 2 utilisé par l'option VSS16
- n = 4 utilisé par l'option ISP16
- n = 6 utilisé par l'option FFM16
- n = 7 utilisé par l'option CDA16

floating point absolute value (FABS)

FABS



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38__

second mot 0800

Opération effectuée : le nombre flottant contenu dans les registres A et B est mis en valeur absolue

Interviennent : A et B

Sont modifiés : A, B, ST

K (option programmée)

Indicateurs :

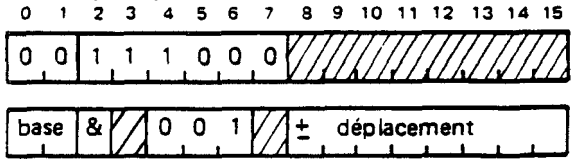
V	C	
0	0	quel que soit le nombre

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 06
- parité

ATTENTION : dans le cas de l'option programmée 26 mots libres sont nécessaires dans la pile pointée par K

floating point add (FAD)

FAD



Option : **virgule flottante câblée (05) ou programmée**

Code : premier mot 38__

second mot

	direct	indirect
base C	42__	62__
base L	82__	A2__
base W	C2__	E2__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant pris en mémoire à l'adresse indiquée par l'instruction est additionné au nombre flottant contenu dans les registres A et B, le résultat se trouve dans les registres A et B.

Les indicateurs V et C et le premier mot du commun sont positionnés de la manière indiquée § 1.3.3

Interviennent : - A et B
- les 2 mots mémoire contenant le nombre flottant
- K (option programmée)

Sont modifiés : - A, B, ST

- le 1er mot du COMMON

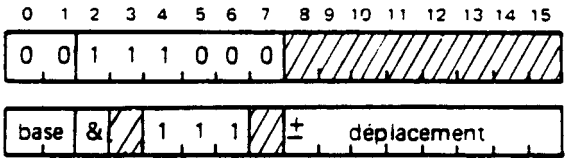
Indicateurs :

V	C	
1	0	overflow
0	1	underflow
0	0	résultat correct

Alarmes possibles
- protection mémoire 05
- mémoire inexistante 06
- parité

ATTENTION :
- dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.
- dans le cas de l'option câblée, l'utilisation d'opérandes non normalisés ou une panne de l'opérateur câblé peuvent provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.
- dans le cas d'un débordement, le contenu des registres A et B est sans signification et dépend de l'option.

floating point compare accumulator to memory (FCAM)



Option . virgule flottante câblée (05) ou programmée

Code . premier mot : 38__

second mot

	direct	indirect
base C	4E__	6E__
base L	8E__	AE__
base W	CE__	EE__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant contenu dans les registres A et B est comparé au nombre flottant pris en mémoire à l'adresse Indiquée par l'instruction, et les Indicateurs V et C sont positionnés selon les résultats de cette comparaison.

on peut tester le résultat de la comparaison de la manière suivante

JL	saut si	AB	<	nombre en mémoire
JE	saut si	AB	=	nombre en mémoire
JG	saut si	AB	>	nombre en mémoire
JGE	saut si	AB	≥	nombre en mémoire
JNE	saut si	AB	≠	nombre en mémoire
JLE	saut si	AB	≤	nombre en mémoire

Interviennent :- A et B
- les 2 mots mémoire contenant le nombre flottant
K (option programmée)

Est modifié : ST

Indicateurs :

V	C	
0	1	AB < mémoire
1	0	AB = mémoire
0	0	AB > mémoire

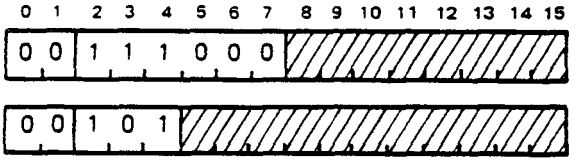
Alarmes possibles : - protection mémoire 05
- mémoire inexistante 06
- parité

ATTENTION : – dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

– dans le cas de l'option câblée, l'utilisation d'opérandes non normalisés ou une panne de l'opérateur câblé peuvent provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

FCAZ

floating point compare accumulator to zero (FCAZ)



Option : **virgule flottante câblée (05) ou programmée**

Code : premier mot 38__
second mot 2800

Opération effectuée : le nombre flottant contenu dans les registres A et B est comparé à zéro, et les indicateurs V et C sont positionnés selon le résultat de cette comparaison.

on peut tester le résultat de la comparaison de la manière suivante

JL	saut si	AB	<	0
JE	saut si	AB	=	0
JG	saut si	AB	>	0
JGE	saut si	AB	⩾	0
JNE	saut si	AB	≠	0
JLE	saut SI	AB	⩽	0

Interviennent : - A et B
K (option programmée)

Est modifié : ST

Indicateurs :

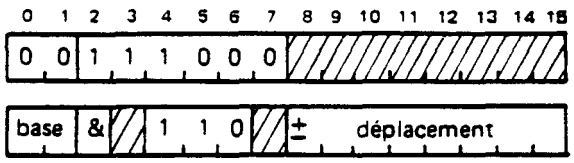
V	C	
0	1	AB < 0
1	0	AB = 0
0	0	AB > 0

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : - dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

FCMZ

floating point compare memory to zero (FCMZ)



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38__

second mot

	direct	indirect
base C	4C__	6C__
base L	8C__	AC__
base W	CC__	EC__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant pris en mémoire à l'adresse indiquée par l'instruction est comparé à zéro et les indicateurs V et C sont positionnés selon les résultats de cette comparaison

on peut tester le résultat de la comparaison de la manière suivante

JL	saut si	nombre en mémoire	< 0
JE	saut si	nombre en mémoire	= 0
JG	saut si	nombre en mémoire	> 0
JGE	saut si	nombre en mémoire	⩾ 0
JNE	saut si	nombre en mémoire	≠ 0
JLE	saut si	nombre en mémoire	⩽ 0

Interviennent : - les 2 mots mémoire contenant le nombre flottant
K (option programmée)

Est modifié : ST

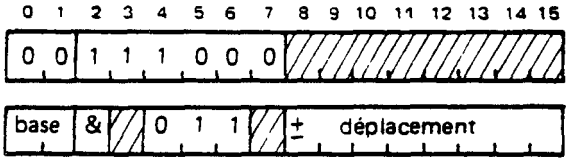
Indicateurs :

V	C	
0	1	mémoire < 0
1	0	mémoire = 0
0	0	mémoire > 0

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 08
- parité

ATTENTION – dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

floating point divide (FDV)



option — virgule flottante câblée (05) ou programmée

Code — premier mot 38__

second mot

	direct	indirect
base C	46__	66__
base L	86__	A6__
base W	C6__	E6__

Adressage — mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant contenu dans les registres A et B est divisé par le nombre flottant pris en mémoire à l'adresse indiquée par l'instruction ; le quotient se trouve dans les registres A et B

les indicateurs V et C et le premier mot du COMMON sont positionnés de la manière indiquée § 1.3.3

Interviennent : - A et B
- les 2 mots mémoire contenant le nombre flottant
K (option programmée)

Sont modifiés : A, B, ST
le premier mot du COMMON

Indicateurs :

V	C	
1	0	overflow division par zéro
0	1	underflow
0	0	résultat correct

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

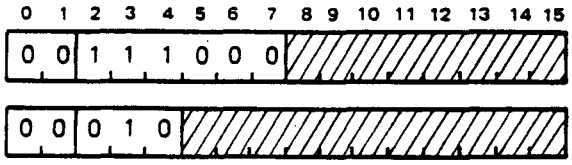
ATTENTION : - dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

- dans le cas de l'option câblée, l'utilisation d'opérandes non normalisés ou une panne de l'opérateur câblé peuvent provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

- dans le cas d'un débordement, le contenu des registres A et B est sans signification et dépend de l'option.

FIX

fix (FIX)



Option : virgule flottante câblée (05) ou programmée

Code
premier mot 38__
second mot 1000

Opération effectuée : le registre A prend comme valeur celle de la partie entière [avec troncature, et de telle manière que FIX (A) = - FIX (- A)] du nombre flottant contenu dans les registres A et B.

si ce nombre flottant est < - 32 768 ou est > 32 767 l'opération est impossible et l'indicateur V est positionné à 1.

le registre B est mis à 0.

Interviennent : - A et B
K (option programmée)

Sont modifiés A, B, ST

- la première mémoire du COMMON (cf § 1.3.3)

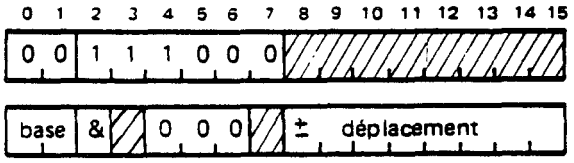
Indicateurs :

V	C	
1	0	nombre flottant trop grand
0	0	résultat correct

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION
- dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.
- dans le cas d'un débordement, le contenu des registres A et B est sans signification et dépend de l'option.

floating point load (FLD)



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38__

second mot

	direct	indirect
base C	40__	60__
base L	80__	A0__
base W	C0__	E0__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le contenu du mot mémoire adressé est chargé dans le registre A, le contenu du mot mémoire suivant est chargé dans le registre B.

On peut ainsi en une seule instruction charger dans les registres A et B un nombre flottant qui se trouve en mémoire dans deux mots consécutifs. Le même relais d'indirection sert pour les 2 mots dans le cas d'adressage indirect.

Interviennent : _ les 2 mots mémoire contenant le nombre flottant
- K (option programmée)

Sont modifiés : A, B, ST

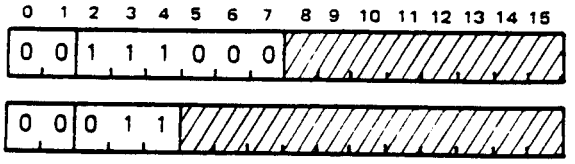
Indicateurs

V	C	
0	0	quel que soit le nombre chargé

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : – dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

float (FLT)



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38--
second mot 1800

Opération effectuée : le nombre entier pris dans le registre A est converti en flottant et rangé dans les registres A et B.

Interviennent : A
K (option programmée)

Sont modifiés : A. B, ST

Indicateurs :

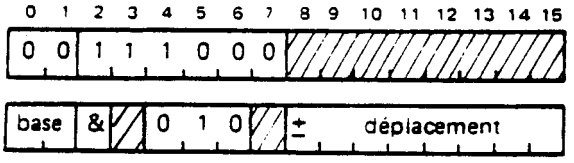
V	C	
0	0	dans tous les cas

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : – dans le cas de l'option programmée 26 mots libres sont nécessaires dans la pile pointée par K,

– dans le cas de l'option câblée une panne de l'opérateur peut provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

floating point multiply (FMP)



option : **virgule flottante câblée (05) ou programmée**

Code : premier mot 38__

second mot

	direct	indirect
base C	44__	64__
base L	84__	A4__
base W	C4__	E4__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant contenu dans les registres A et B est multiplié par le nombre flottant pris en mémoire à l'adresse indiquée par l'instruction, le résultat se trouve dans les registres A et B.

les indicateurs V et C et le premier mot du COMMON sont positionnés de la manière indiquée § 1.3.3

Interviennent : - A et B
- les 2 mots mémoire contenant le nombre flottant
- K (option programmée)

Sont modifiés A, B, ST

le premier mot du COMMON

Indicateurs :

V	C	
1	0	overflow
0	1	underflow
0	0	résultat correct

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
-parité

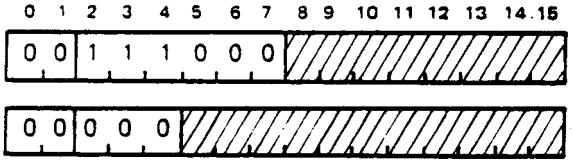
ATTENTION : – dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

– dans le cas de l'option câblée, l'utilisation d'opérandes non normalisés ou une panne de l'opérateur câblé peuvent provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

– dans le cas d'un débordement, le contenu des registres A et B est sans signification et dépend de l'option.

floating point negate (FNEG)

FNEG



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38--
second mot 0000

Opération effectuée : le nombre flottant contenu dans les registres A et B est changé de signe.

Interviennent : A et B
K (option programmée)

Sont modifiés : A, B, ST

Indicateurs :

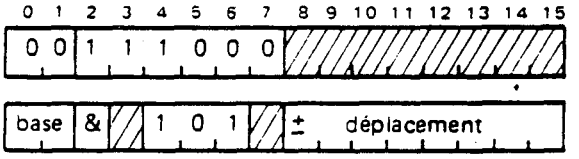
v	c	
0	0	quel que soit le nombre

Alarmes possibles :
- protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.

FSB

floating point substract (FSB)



Option : **virgule flottante câblée (05) ou programmée**

Code premier mot 38__

second mot	direct	indirect
base C	4A__	6A__
base L	8A__	AA__
base W	CA__	EA__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le nombre flottant pris en mémoire à l'adresse indiquée par l'instruction est soustrait au nombre flottant contenu dans les registres A et B, le résultat se trouve dans les registres A et B

Les indicateurs V et C et les premiers mots du COMMON sont positionnés de la manière indiquée § 1.3.3

Interviennent : - A et B
- les 2 mots mémoire contenant le nombre flottant
- K (option programmée)

Sont modifiés : - A, B, ST
- le premier mot du COMMON

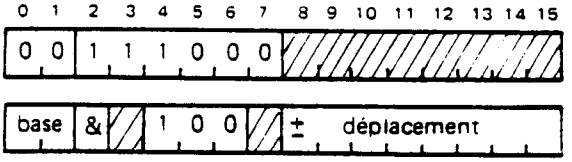
Indicateurs :

V	C	
1	0	overflow
0	1	underflow
0	0	résultat correct

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION : - dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.
- dans le cas de l'option câblée, l'utilisation d'opérandes non normalisés ou une panne de l'opérateur câblé peuvent provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

floating point store (FST)



Option : virgule flottante câblée (05) ou programmée

Code : premier mot 38__

second mot

	direct	indirect
base C	48__	68__
base L	88__	A8__
Base W	C8__	E8__

Adressage : mémoire (1er des 2 mots du nombre flottant)

Opération effectuée : le contenu du registre A est rangé dans le mot mémoire adressé, le contenu du registre B est rangé dans le mot mémoire suivant.

On peut ainsi, en une seule instruction, ranger dans deux mots mémoire consécutifs un nombre flottant qui se trouve dans les registres A et B. Le même relais d'indirection sert pour les 2 mots dans le cas d'adressage indirect.

Interviennent : - A et B
- K (option programmée)

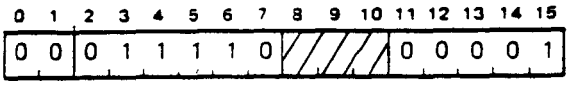
Sont modifiés : - les 2 mots mémoire où l'on range le nombre flottant
- ST

Indicateurs :

V	C	
0	0	quel que soit le nombre rangé

Alarmes possibles
- protection mémoire 05
- mémoire inexistante 05
- parité

ATTENTION
- dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K.



Code : 1E01

Opération effectuée : le calculateur cesse d'exécuter des instructions jusqu'à l'épuisement du nombre de périodes indiqué dans X, ou jusqu'à la prise en compte d'une interruption de programme, d'un défaut secteur ou d'un appel LDC. L'instruction HALT se termine alors et X contient le nombre de périodes qui restaient à écouler.

On peut ainsi attendre les interruptions sans que l'unité centrale effectue de cycle mémoire, ce qui permet des entrées/sorties sur les périphériques rapides à la cadence maximale de ces périphériques.

HALT doit être la dernière instruction de la séquence de prise en compte des défauts secteur avec un nombre de périodes suffisant.

- le contenu de X est considéré comme un nombre arithmétique compris entre 1 et 65536 (0 étant interprété comme 2^{16}).
- la période correspond à 16 cycles de base soit 2 microsecondes pour une horloge de base réglée à 8 MHz.
- les différents canaux intégrés au processeur interrompent le décomptage des périodes, lequel correspond donc strictement au temps passé par le processeur à ne rien faire.

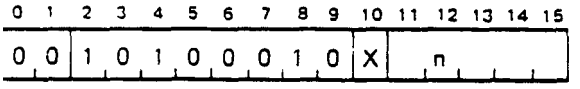
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

Interruptibilité : l'instruction se termine sur une interruption de programme, sur un appel LDC et sur un défaut secteur.

inverse bit (IBT)

IBT



Code : 28 8_

Adressage : indexation possible

Opération effectuée : les registres A et B étant considérés comme un registre unique de 32 bits, le bit dont le rang est n si l'instruction n'est pas indexée. n + x modulo 32 si l'instruction est indexée, est inversé.

Intervient : un bit de A ou un bit de B

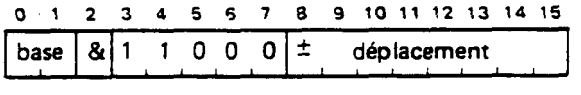
Est modifié : un bit de A ou un bit de B

Indicateurs : non modifiés

Alarmes possibles
- protection mémoire (05)
- mémoire inexistante (05)
- parité

increment (IC)

IC



Code :

	direct	indirect
base C	58__	78__
base L	98__	B8__
base W	D8__	F8__

Adressage : mémoire (mot)

Opération effectuée : la valeur 1 est additionnée à la mémoire adressée. Les indicateurs V et C sont positionnés selon le résultat de la comparaison de la valeur obtenue et de zéro.
On peut ainsi augmenter de 1 la valeur d'une mémoire sans avoir à passer par les registres, et tester ensuite le résultat obtenu sans avoir à faire de comparaison :

JL	Saut si	mémoire < 0
JE	saut si	mémoire = 0
JG	saut si	mémoire > 0
JGE	saut si	mémoire $\geqslant$ 0
JNE	saut si	mémoire $\neq$ 0
JLE	saut si	mémoire $\leqslant$ 0

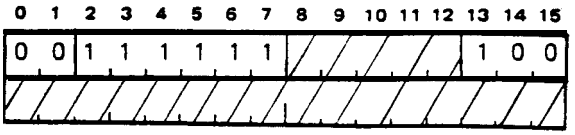
Intervient : le mot mémoire adressé

Sont modifiés : . le mot mémoire adressé
. ST

Indicateurs :

V	C	
0	1	mot mémoire < 0
1	0	mot mémoire = 0
0	0	mot mémoire > 0

Alarmes possibles
- protection mémoire (05)
- mémoire inexistante (05)
- parité



Option : CDA16 (65) et (70)

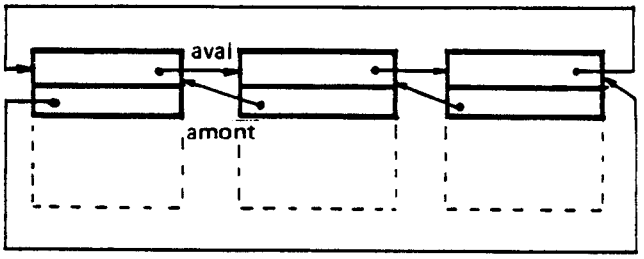
Code : premier mot 3F04
deuxième mot 0000

Opération effectuée : insère dans une liste l'élément dont l'adresse est dans l'accumulateur à la suite de l'élément dont l'adresse est dans le registre B.

La liste est une suite de double mots chaînés dans les deux sens :

le premier mot contient l'adresse du double mot suivant (chaînage aval),
le second mot contient l'adresse du double mot précédent (chaînage amont),

A chaque élément de la liste on peut associer un ou plusieurs mots qui constituent l'information propre à cet élément. Ces informations ne sont pas modifiées par l'instruction.



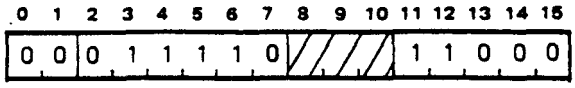
En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

Interviennent : A, B, l'élément pointé par B.

Sont modifiés : l'élément pointé par A, l'élément pointé par B et son successeur

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante
- protection mémoire
- parité mémoire



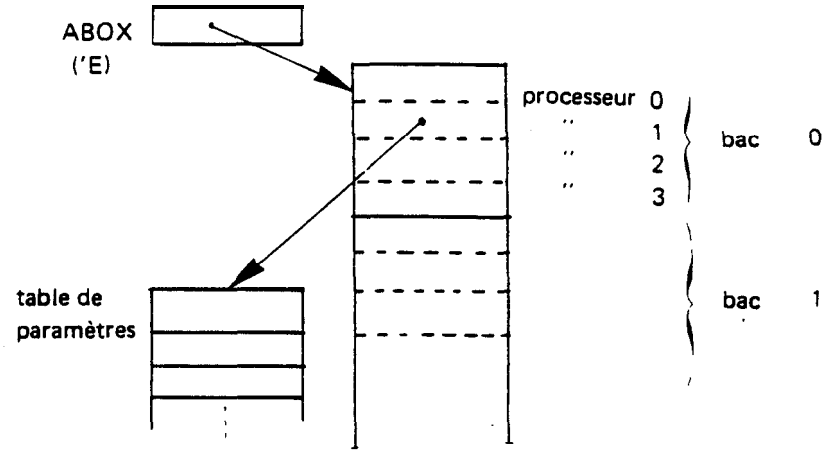
Mode : maître

Code : 1E18

Opération effectuée : Cette instruction permet :

- soit de provoquer une interruption interprocesseurs sous la forme de l'alarme numéro 6 : réveil interprocesseurs (05)
- soit d'initialiser ou de terminer un échange canal.
  L'adressage du ou des processeurs concernés et la communication d'informations se font à travers une zone mémoire (dite boîte aux lettres) de la manière suivante :

  la boite aux lettres est constituée d'une table dont l'adresse est contenue par le mot mémoire d'adresse débanalisée 'E. Chaque processeur y possède un mot indexé par son numéro compte tenu de son numéro de bac :



  Cette boite aux lettres doit être en mémoire commune.

- L'interruption inter-processeurs est reçue par tous les processeurs (y compris le processeur émetteur). Chacun examine alors le mot de la boite aux lettres qui lui est réservé.
  Si ce mot est nul, le processeur correspondant n'est pas concerné, sinon ce mot contient l'adresse d'une table de paramètres qui décrit le travail sous-traité à ce processeur.
  Deux cas standard se présentent :

  . l'initialisation et la libération des canaux,
  . le réveil inter-processeurs.

- Pour une initialisation (ou une libération) de canal la table de paramètres est alors une table d'échange (CCB). Le processeur concerné prend en compte le travail qui lui est demandé puis remet à zéro sa boite aux lettres. On se reportera à la description des canaux pour les détails sur les tables d'échanges et leur programmation.
- Pour un réveil inter-processeurs, la table de paramètres se résume au mot suivent :

  '8000

  Le processeur concerné remet alors sa boîte aux lettres à zéro et reflète cette interruption au programme sous la forme d'une alarme de numéro 6. C'est le programme qui doit alors retrouver en mémoire commune les informations complémentaires

liées à cet appel.

Dans le cas où le processeur destinataire est déjà dans la tâche alarme au moment de l'interruption, celle-ci est ignorée.
Dans un système multiprocesseur (on exclut ici le cas des processeurs d'entrées-sorties), l'utilisation du réveil interprocesseurs impose donc de terminer les tâches alarme par une séquence de test des informations complémentaires en mémoire commune pour ne pas perdre d'appel. De plus, pour éviter tout aléa, cette séquence doit être masquée de cette interruption (masque IPM) et se terminer par l'instruction ACQ qui la redémasque.
Le réveil inter-processeur ne doit pas être utilisé avec un processeur d'entrées-sorties (IOP16-M,...) sauf microprogrammation spécifique complémentaire.

Interviennent : la boite aux lettres, la table de paramètres

Est modifié : la boite aux lettres, changement de contexte éventuel

Indicateurs : non modifiés, sauf changement de contexte éventuel

Alarmes possibles :
- mémoire inexistante 06
- protection mémoire 05
- parité mémoire
- réveil inter-processeurs 05

Modèles 65P et 75P : Sur ces modèles un nouveau mécanisme d'IPI réveil de processeurs est implémenté.
Une mémoire débanalisée ABOX2 doit pointer sur une adresse multiple de 16 mots (adresse donnée en format "SLO").

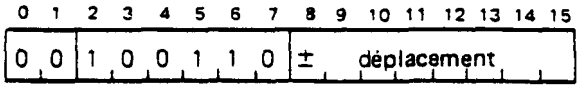
L'adresse pointée par ABOX2 et indexée par le numéro de processeur est une boite aux lettres limitée à la fonction réveil de processeur.
Pour réveiller le processeur i, mettre '8000 dans le mot pointé par (ABOX2) + i et exécuter IPI.

Ce mécanisme ne fonctionne que si ABOX2 est différent de 0.
ABOX2 est la mémoire débanalisée d'adresse '1E, elle est remise à 0 sur mise sous tension ou initialisation manuelle, mais pas sur retour secteur.

La boîte aux lettres est remise à 0 après prise en compte de l'IPI.

jump on A equal zero (JAE)

JAE



Code : 26__

Adressage : par rapport à P

Opération effectuée : si le contenu du registre A est égal à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JAE elle-même).

On peut ainsi, selon la valeur de A par rapport à zéro sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

Intervient : A

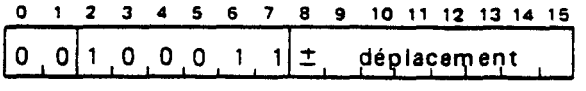
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

jump on A greater zero (JAG)

JAG



Code : 23__

Adressage : par rapport à P

Opération effectuée : si le contenu du registre A est supérieur à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JAG elle-même).

On peut ainsi, selon la valeur de A par rapport à zéro, sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

Intervient : A

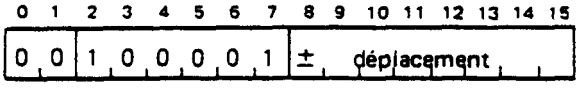
Est modifié : P

Indicateurs non modifiés

Alarmes possibles :- protection mémoire (05)
- mémoire inexistante (05)
- parité

JAGE

jump on A greater or equal zero (JAGE)



Code : 21__

Adressage : par rapport à P

Opération effectuée : si le contenu du registre A est plus grand ou égal à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JAGE elle-même.

On peut ainsi, selon la valeur de A par rapport à zéro sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

Intervient : A

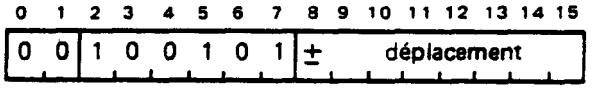
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : -protection mémoire 05
- mémoire inexistante 05
- parité

JAL

jump on A less zero (JAL)



Code : 25__

Adressage : par rapport à P

Opération effectuée : si le contenu du registre A est inférieur à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JAL elle-même). On peut ainsi, selon la valeur de A par rapport à zéro, sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

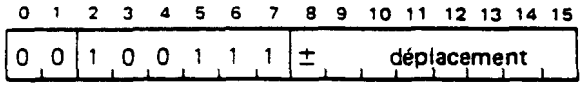
Intervient : A

Est modifié : P

Indicateurs non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

jump on A less or equal zero (JALE) — JALE



Code : 27__

Adressage : par rapport à P

Opération effectuée : Si le contenu du registre A est inférieur ou égal à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JALE elle-même).

On peut ainsi, selon la valeur de A par rapport à zéro sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

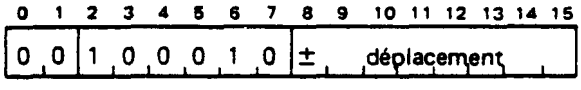
Intervient : A

Est modifié : P

Indicateurs : non modifiés

Alarmes possibles: - protection mémoire (05)
- mémoire Inexistante (05)
- parité

jump on A non equal zero (JANE) — JANE



Code : 22__

Adressage : par rapport à P

Opération effectuée : si le contenu du registre A est différent de zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JANE elle-même)

On peut ainsi, selon la valeur de A par rapport à zéro, sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, sans avoir à faire de comparaison.

Intervient : A

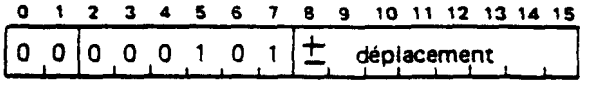
Est modifié : P

indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

jump on C indicator (JC)

JC



Code : 0 5 _ _

Adressage : par rapport à P

Opération effectuée : si l'indicateur C vaut 1, le déplacement est additionné au registre P (qui pointe sur l'instruction JC elle-même).

On peut ainsi, selon la valeur de l'indicateur C, sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent.

Intervient : Indicateur C

Est modifié : P

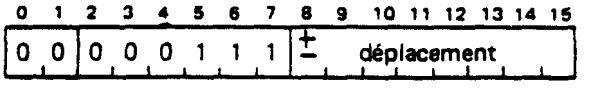
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

jump on C or V indicator (JCV)

JCV



Code : 0 7 _ _

A d - : par rapport à P

Opération effectuée : si l'un au moins des deux indicateurs C et V vaut 1, le déplacement est additionné au registre P (qui pointe sur l'instruction JCV elle-même).

On peut ainsi, selon la valeur de l'intersection logique des deux indicateurs C et V, sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent.

Interviennent : indicateurs C et V

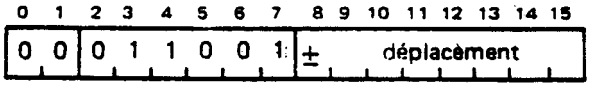
Est modifié P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

jump decrementing X (JDX)

JDX



Code : 19__

Adressage : par rapport à P

Opération effectuée : la valeur 1 est soustraite au registre X. Si la nouvelle valeur de X est supérieure à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JDX elle-même).

On peut ainsi réaliser une boucle de programme à l'aide d'un JDX placé à la fin de la séquence et effectuant à la fois la variation de l'index et le saut conditionnel. Si l'index a une valeur initiale positive n, la séquence sera exécutée n fois

Intervient : X

Sont modifiés : - X
- P

Indicateurs : non modifiés

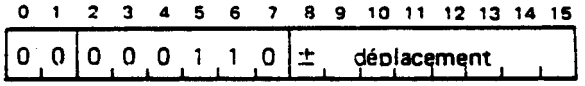
Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : si X = 0 pas de saut

---

jump on equal (JE)

JE



Code : 06__

Adressage : par rapport à P

Opération effectuée : JE est un autre code mnémonique de l'instruction JV. Cette instruction permet de sauter vers une des 128 instructions qui précédent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée

CP	A	=	mémoire
CPI	A	=	op. im.
CPBY	A	=	octet
CPR	2d reg.	=	1er reg.
IC	mem	=	0
DC	mem	=	0
CPZ	mem	=	0
CPZR	reg.	=	0

Intervient : indicateur V

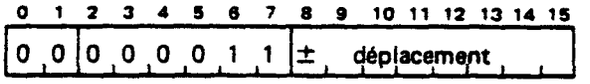
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

JG

jump on greater (JG)



Code : 03__

Adressage : par rapport à P

Opération effectuée : JG est un autre code mnémonique de l'instruction JNCV. Cette instruction permet de sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée.

CP	A	>	mémoire
CPI	A	>	op. im.
CPBY	A	>	octet
CPR	2d reg.	>	1er reg.
IC	mem	>	0
DC	mem	>	0
CPZ	mem	>	0
CPZR	reg.	>	0

Interviennent : indicateurs C et V

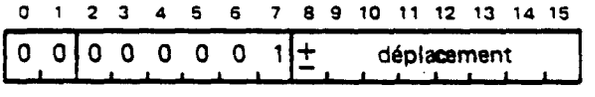
Est modifié : P

Indicateurs : non modifies

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (08)
- parité

JGE

jump on greater or equal (JGE)



Code : 01__

Adressage : par rapport à P

Opération effectuée : JGE est un autre code mnémonique de l'instruction JNC. Cette instruction permet de sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée :

CP	A	≥	mémoire
CPI	A	≥	op. im.
CPBY	A	≥	octet
CPR	2d reg.	≥	1er reg.
IC	mem	≥	0
DC	mem	≥	0
CPZ	mem	≥	0
CPZR	reg.	≥	0

Intervient : indicateur C

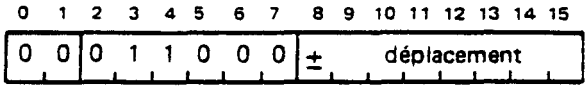
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

jump incrementing X (JIX)

JIX



Code : 18__

Adressage : par rapport à P

Opération effectuée : la valeur 1 est additionnée au registre X, puis si la nouvelle valeur de X est inférieure à zéro, le déplacement est additionné au registre P (qui pointe sur l'instruction JIX elle-même).

On peut ainsi réaliser une boucle de programme à l'aide d'un JIX placé à la fin de la séquence et effectuant à la fois la progression de l'index et le saut conditionnel.

Si l'index a une valeur initiale négative - n, la séquence sera exécutée n fois.

Intervient : x

Sont modifiés : - x
- P

Indicateurs : non modifiés

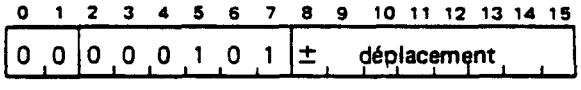
Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : si X = 0 pas de saut

---

jump on less (JL)

JL



Code : 05__

Adressage : par rapport à P

Opération effectuée : JL est un autre code mnémonique de l'instruction JC. Cette instruction permet de sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée.

CP	A	< mémoire
CPI	A	< op. im.
CPBY	A	< octet
CPR	2d reg.	< 1er reg.
IC	mem	< 0
DC	mem	< 0
CPZ	mem	< 0
CPZR	reg	< 0

Intervient : indicateur C

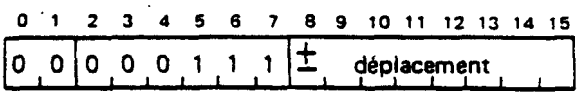
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

jump on less or equal (JLE)

JLE



Code : 07__

Adressage : par rapport à P

Opération effectuée : JLE est un autre code mnémonique de l'instruction JCV. Cette instruction permet de sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée.

CP	A	≤	mémoire
CPI	A	≤	op. im.
CPBY	A	≤	octet
CPR	2d reg.	≤	1er reg.
IC	mem	≤	0
DC	mem	≤	0
CPZ	mem	≤	0
CPZR	reg.	≤	0

Interviennent : indicateurs C et V

Est modifié : P

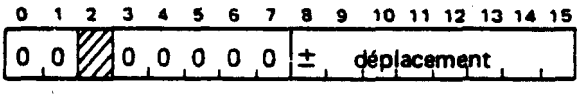
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

---

jump (JMP)

J M P



codes :00__ et 20__

Adressage : par rapport à P

Opération effectuée : le déplacement est additionné au registre P (qui pointe sur l'instruction JMP elle-même).

On peut ainsi sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent.

Est modifié : P

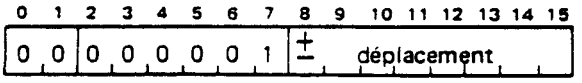
Indicateurs non modifiés

Alarmes possibles : protection mémoire (05)
mémoire inexistante (05)
parité

ATTENTION : les codes 00-- et 20-- provoquent tous les deux un saut inconditionnel, mais leurs temps d'exécution diffèrent selon les processeurs.

jump on not C indicator (JNC)

JNC



Code 0 1 _ _

Adressage par rapport à P

Opération effectuée : si l'indicateur C vaut 0 le déplacement est additionné au registre P (qui pointe sur l'instruction JNC elle-même).

On peut ainsi, selon la valeur de l'indicateur C, sauter vers une des 128 instructions qui précèdent ou des 127 qui suivent.

Intervient : indicateur C

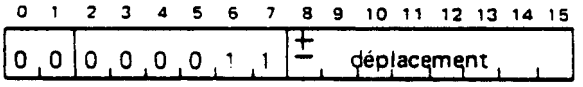
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire Inexistante 05
- parité

jump on not C and V indicators (JNCV)

JNCV



Code : 03__

Adressage : par rapport à P

Opération effectuée : si l'indicateur C vaut 0 et l'indicateur V vaut 0 le déplacement est additionné au registre P (qui pointe sur l'instruction JNCV elle-même).

On peut ainsi, selon la valeur de l'intersection logique des deux indicateurs C et V, sauter vers une des 128 Instructions qui précèdent ou des 127 instructions qui suivent.

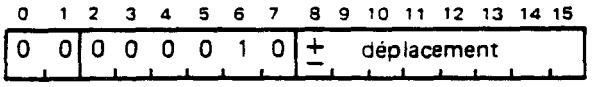
Interviennent : indicateurs C et V

Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

jump on not equal (JNE) — JNE



Code : 02__

Adressage : par rapport à P

Opération effectuée : JNE est un autre code mnémonique de l'instruction JNV. Cette instruction permet de sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent, lorsqu'elle est placée à la suite des instructions qui figurent dans le tableau suivant et que la condition indiquée est réalisée.

CP	A	≠	mémoire
CPI	A	≠	op. im.
CPBY	A	≠	octet
CPR	2d reg.	≠	1er reg.
IC	mem	≠	0
DC	mem	≠	0
CPZ	mem	≠	0
CPZR	reg.	≠	0

Intervient : indicateur V

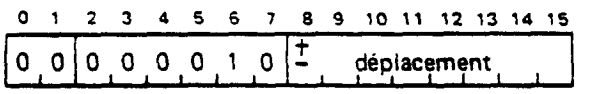
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

---

jump on not V indicator (JNV) — JNV



code : 02__

Adressage : par rapport à P

Opération effectuée : si l'indicateur V vaut 0 le déplacement est additionné au registre P (qui pointe sur l'instruction JNV elle-même)

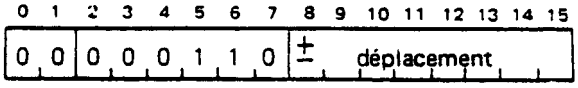
On peut ainsi, selon la valeur de l'indicateur V sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent.

Intervient : indicateur V

Est modifié : P

indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité



Code : 06__

Adressage : par rapport à P

Opération effectuée : si l'indicateur V vaut 1 le déplacement est additionné au registre P (qui pointe sur l'instruction JV elle-même).

On peut ainsi, selon la valeur de l'indicateur V sauter vers une des 128 instructions qui précèdent ou des 127 instructions qui suivent.

Intervient : indicateur V

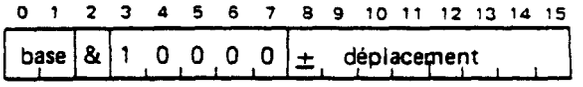
Est modifié : P

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité

load A (LA)

LA



Code

base C	50__	70__
base L	90__	B0__
base W	D0__	F0__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire est chargé dans le registre A.

Intervient : le mot mémoire adressé

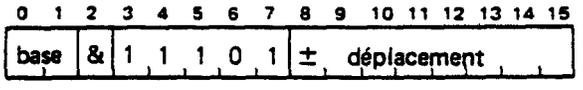
Est modifié : A

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

load address (LAD)

LAD



Code :

	direct	indirect
base C	5D ––	7D ––
base L	9D ––	BD ––
base W	DD ––	FD ––

Adressage : mémoire (mot)

Opération effectuée : le registre A est chargé par l'adresse précisée dans l'instruction (et non par le contenu du mot mémoire situé à cette adresse).

On peut ainsi :

- charger le registre A par la somme de la base et du déplacement (adressage direct)
- charger le registre A par le contenu du relais d'indirection (adressage indirect)
- charger le registre A par la somme du relais d'indirection et du registre X (adressage indirect post indexé).

Interviennent : la base, le déplacement éventuellement le relais d'indirection, X

Est modifié : A

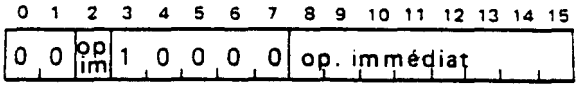
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : en mode esclave SLO n'est pas ajouté à l'adresse chargée dans A, de manière à ce qu'un programme exécutant un LAD dans ce mode puisse utiliser normalement l'adresse obtenue.

load A immediate (LAI)

LAI



Code:

bit 2 = 0	10__
bit 2 = 1	30__

Adressage : immédiat (9 bits)

Opération effectuée : les bits 8 à 15 de l'instruction sont chargés dans les bits 8 à 15 du registre A.
Le bit 2 de l'instruction est répété dans les bits 0 à 7 du registre A.
On peut ainsi

- remettre à zéro le registre A
- charger le registre A par une valeur algébrique comprise entre - 256 et + 255
- charger le registre A par une valeur logique comprise entre '0000 et '00FF ou comprise entre 'FF00 et 'FFFF
- charger le registre A par un caractère ASCII 8 bits.

Intervient : l'opérande immédiat (sur 16 bits)

Est modifié : . A

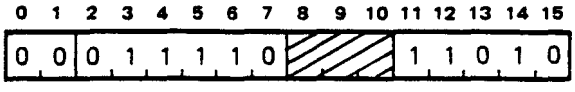
indicateurs : . non modifiés

Alarmes possibles : - protection mémoire (08)
- mémoire inexistante (09)
- parité

---

load accumulator relative (LAR)

LAR



Mode : maître

Code : 1 E 1 A

Opération effectuée : chargement de l'accumulateur par le mot mémoire dont l'adresse relative à SLO est contenue par le registre Y
La protection d'accès (SLE) est activée.
En l'absence de l'option DRPS, le registre Y contient une adresse absolue.

Intervient : le mot mémoire pointé par Y, SLO, SLE

Est modifié : A

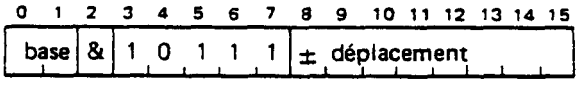
Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

ATTENTION : le programme peut passer en mode esclave à la suite d'une alarme.

load B (LB)

LB



Code :

	direct	indirect
base C	57__	77__
base L	97__	B7__
base W	D7__	F7__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adresse est chargé dans le registre B.

Intervient : le mot mémoire adresse

Est modifié : B

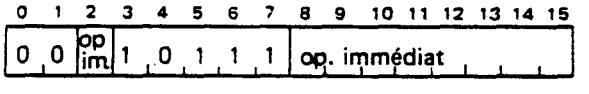
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

load B immediate (LBI)

LBI



Code:

bit 2 = 0	17__
bit 2 = 1	37__

Adressage : immédiat (9 bits)

Opération effectuée : les bits 8 à 15 de l'instruction sont chargés dans les bits 8 à 15 du registre B.
Le bit 2 de l'instruction est répété dans les bits 0 à 7 du registre B.
On peut ainsi :
- remettre à zéro le registre B
- charger le registre B par une valeur algébrique comprise entre - 256 et + 255.
- charger le registre B par une valeur logique comprise entre '0000 et '00FF ou entre 'FF00 et 'FFFF
- charger le registre B par un caractère ASCII 8 bits

Intervient : l'opérande immédiat (sur 16 bits)

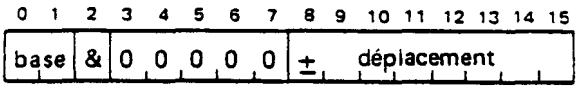
Est modifié : B

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

load byte (LBY)

LBY



		direct	indirect
Code	base C	40__	60__
	base L	80__	A0__
	base W	C0__	E0__

Adressage : mémoire (octet)

Opération effectuée : les bits 8 à 15 du registre A sont chargés par l'octet mémoire adressé.

Les bits 0 à 7 du registre A sont remis à zéro.

Intervient : l'octet mémoire adressé.

Est modifié : A

Indicateurs : non modifies

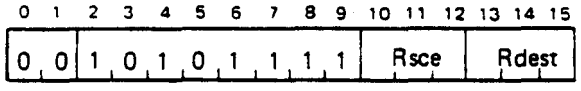
Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : les bits 0 à 7 de A sont mis à zéro.
s'il n'y a pas indexation, l'octet adressé est l'octet gauche d'un mot

---

load register (LR)

LR



Code 28__

Adressage registre registre

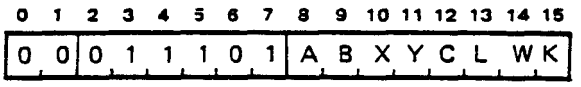
Opération effectuée le contenu du registre source est chargé dans le registre destination.
On peut ainsi effectuer un transfert entre deux quelconques des huit registres A B X Y C L W K.

Intervient : un des registres A B X Y C L W K.

Est modifié : un des registres A B X Y C L W K.

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Code ‘10__

Opération effectuée : les registres précisés dans l'instruction sont chargés successivement dans l'ordre A, B, X, Y, C, L, W, K, par les mots mémoire qui suivent l'instruction (adresses (P) + 1, (P) + 2, ...).

En fin d'opération, le pointeur P est augmenté de n + 1, si n est le nombre de registres précisés.

On peut ainsi initialiser les registres et en particulier les bases au début des programmes, sous-programmes, tâches d'interruption, ...

Interviennent : les mémoires d'adresses (P) + 1, (P) + 2,...

Sont modifiés : P, les registres précisés dans l'instruction

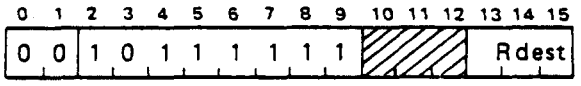
Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante (05)
protection mémoire (05)
parité mémoire

ATTENTION : cette instruction n'est interruptible ni par le canal LDC, ni par les interruptions de programme et le chargement de plus de quatre registres peut provoquer une dégradation de leurs performances.

load register with P (LRP)

LRP



Code : 2FC_

Adressage : registre - registre

Opération effectuée : le contenu du registre P (c'est-à-dire l'adresse de l'instruction LRP elle-même) est chargé dans le registre destination.
On peut ainsi : initialiser les bases C, L, W et le registre K dans un programme écrit sous forme translatable.

Intervient : P

Est modifié : un des registres A B X Y C L W K.

Indicateurs : non modifiés

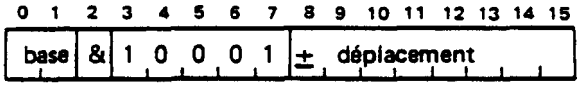
Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

ATTENTION : la valeur de P chargée est l'adresse de l'instruction LRP elle-même.

---

load X (LX)

LX



Code :

	direct	indirect
base C	51__	71__
base L	91__	B1__
base W	D1__	F1__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adresse est chargé dans le registre X.

Intervient : le mot mémoire adressé

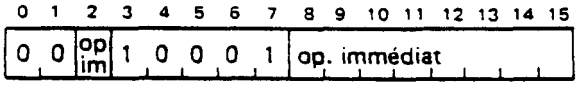
Est modifié : x

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

load X immediate (LXI)

LXI



Code:

bit 2 = 0	11__
bit 2 = 1	31__

Adressage : immédiat (9 bits)

Opération effectuée : les bits 8 à 15 de l'instruction sont chargés dans les bits 8 à 15 du registre X.
Le bit 2 de l'instruction est répété dans les bits 0 à 7 du registre X.
On peut ainsi :
- remettre à zéro le registre X
- charger le registre X par une valeur algébrique comprise entre - 256 et + 255
- charger le registre X par une valeur algébrique comprise entre '0000 et '00FF ou entre 'FF00 et 'FFFF
- charger le registre X par un caractère ASCII 8 bits

Intervient : l'opérande immédiat (sur 16 bits)

Est modifie : X

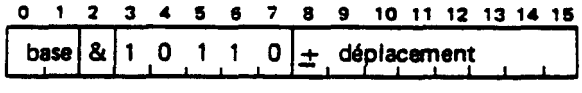
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

load Y (LY)

LY



Code :

	direct	indirect
base C	56__	76__
base L	96__	B6__
base W	D6__	F6__

Adressage : mémoire (mot)

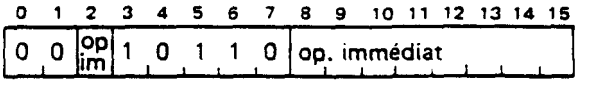
Opération effectuée : le contenu du mot mémoire adressé est chargé dans le registre Y.

Intervient : le mot mémoire adressé

Est modifié : Y

Indicateurs : non modifiés

Alarmes possibles :-protection mémoire (05)
-mémoire inexistante (05)
-parité



Code :

bit 2 = 0	16 _ _
bit 2 = 1	36 _ _

Adressage : immédiat (9 bits)

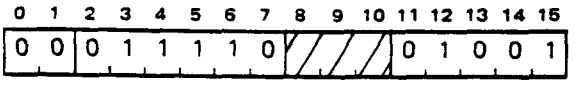
Opération effectuée : les bits 8 à 15 de l'instruction sont chargés dans les bits 8 à 15 du registre Y.
Le bit 2 de l'instruction est répété dans les bits 0 à 7 du registre Y.
On peut ainsi
- remettre à zéro le registre Y
- charger le registre Y par une valeur algébrique comprise entre - 256 et + 255
- charger le registre Y par une valeur logique comprise entre '0000 et '00FF ou antre 'FF00 et 'FFFF
- charger le registre Y par un caractère ASCII 8 bits

Intervient : l'opérande immédiat (sur 16 bits)

Est modifié : Y

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (06)
- mémoire inexistante (05)
- parité



Code : 1 E 0 9

Opération effectuée : transfert du contenu d'une zone mémoire vers une autre zone mémoire :
- le registre A contient l'adresse du 1er mot de la zone source,
- le registre B contient l'adresse du 1er mot de la zone destination,
- le registre X contient le nombre de mots à transférer.

Le transfert s'effectue mot par mot par adresses décroissantes et les zones peuvent se recouvrir.

A chaque transfert, le registre X est diminué de 1 et les interruptions (ainsi que le défaut secteur) sont prises en compte.

A la fin du transfert, le contenu de X est nul.

L'opération est ineffective si le contenu de X est négatif après décrémentation.

On peut transférer au maximum 32768 mots.

Interviennent : A (adresse zone source)
B (adresse zone destination)
X (nombre de mots à transférer)
les mots mémoire pointés par A

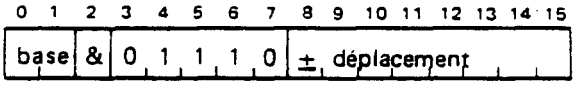
Sont modifiés : x
les mots mémoire pointés par B

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
-mémoire inexistante 05
-parité

Interruptibilité : l'instruction est interruptible entre chaque transfert d'un mot, y compris par le défaut secteur.

Remarque : les adresses passées dans A et B sont prises sur 16 bits (0 à 64 K), relativement à SLO en mode esclave avec l'option DRPS.



Option : cette instruction est soit microprogrammée, soit câblée.

Code :

	direct	indirect
base C	4E__	6E__
base L	8E__	AE__
base W	CE__	EE__

Adressage : mémoire (mot)

Opération effectuée : le contenu du registre A est multiplié par le contenu de la mémoire adressée. On obtient dans tous les cas un résultat algébrique sur 32 bits, les poids forts étant dans A et les poids faibles dans B.

On peut ainsi :

- utiliser le résultat obtenu en double longueur,
- faire suivre immédiatement une multiplication par une division, le résultat de la multiplication étant alors correctement en place dans A et B pour la division
- ne conserver que les 16 bits poids faible du résultat, en s'assurant que le résultat de la multiplication effectuée tient sur 16 bits (les 16 bits de A sont identiques au bit 0 du B), par la séquence suivante :

```
TBT     16      < CARRY := BIT 0 DE B
ADCR    A       < A := A + BIT 0 DE B
JANE    ØVFLØW  < SI A ≠ 0    DEBORDEMENT
```

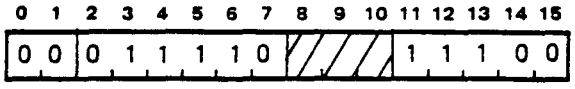
Interviennent : . A
. le mot mémoire adressé

Sont modifiés : . A, B, ST

Indicateurs :

v	c	
0	0	dans tous les cas

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Mode : maître

Code : 1E1C

Opération effectuée : transfert du contenu d'une zone mémoire d'adresse relative à SLD, vers une zone mémoire d'adresse absolue :

- le registre A contient l'adresse relative à SLO du 1er mot de la zone source,
- le registre B contient l'adresse absolue du 1er mot de la zone destination,
- le registre X contient le nombre de mots à transférer.

Le transfert s'effectue mot par mot par adresses décroissantes et les zones peuvent se recouvrir.

A chaque transfert, le registre X est diminué de 1 et les interruptions (ainsi que le défaut secteur) sont prises en compte.
A la fin du transfert, le contenu de X est nul.

L'opération est ineffective si le contenu de X est négatif après décrémentation.

On peut ainsi transférer au maximum 32768 mots, en s'affranchissant des frontières de 64 Kmots.

La protection de l'option DRPS est activée à chaque lecture de la zone source.
En l'absence de l'option DRPS, le contenu du registre A devient une adresse absolue.

Interviennent : A, B, X, SLO, SLE, la zone mémoire pointée par A

Sont modifiés : X, la zone mémoire pointée par B

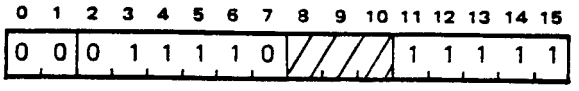
Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

Interruptibilité : l'instruction est interruptible à chaque transfert de mots y compris par le défaut secteur.

Remarque : les adresses passées dans A et B sont prises sur 16 bits (0 à 64 K), relativement à SLO en mode esclave avec l'option DRPS.

ATTENTION : le programme peut passer en mode esclave à la suite d'une alarme.



Mode : maître

Code : 1 E 1 F

Opération effectuée : transfert du contenu d'une zone mémoire d'adresse absolue vers une zone mémoire d'adresse relative à SLO :

- le registre A contient l'adresse absolue du 1er mot de la zone source,
- le registre B contient l'adresse relative à SLO du 1er mot de la zone destination,
- le registre X contient le nombre de mot à transférer.

Le transfert s'effectue mot par mot par adresses décroissantes et les zones peuvent se recouvrir.

A chaque transfert, le registre X est diminué de 1 et les interruptions (ainsi que le défaut secteur) sont prises en compte.
A la fin du transfert, le registre X est nul.
L'opération est ineffective si le contenu de X est négatif après décrémentation.

On peut ainsi transférer au maximum 32768 mots en s'affranchissant des frontières de 64 Kmots.

La protection de l'option DRPS est activée à chaque écriture dans la zone destination. En l'absence de l'option DRPS, le contenu du registre B devient une adresse absolue.

Interviennent : A, B, X, SLO, SLE, la zone mémoire pointée par A

Sont modifiés : X, la zone mémoire pointée par B

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

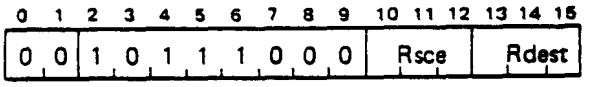
Interruptibilité : l'instruction est interruptible à chaque transfert de mots y compris par le défaut secteur.

Remarque : les adresses passées en A et B sont prises sur 16 bits (0 à 64 K), relativement à SLO en mode esclave avec l'option DRPS.

ATTENTION : le programme peut passer en mode esclave à la suite d'une alarme.

negate register (NGR)

NGR



Code : 2E0_

Adressage : registre - registre

Opération effectuées : le registre destination est chargé par l'opposé du contenu du registre source (il s'agit du complément à deux du nombre : même valeur, signe inverse).
On peut ainsi :
- effectuer un transfert avec changement de signe entre deux quelconques des huit registres A B X Y C L W K.
- changer le signe du contenu d'un de ces huit registres, le registre destination étant alors le registre source.

Interviennent : . deux des registres A B X Y C L W K
. indicateur C

Sont modifiés : . un des registres A B X Y C L W K
. ST

Indicateurs :

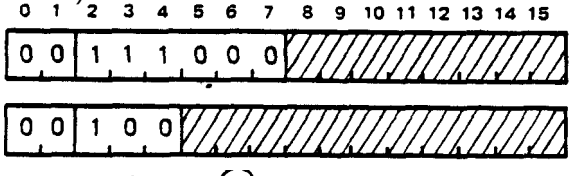
V	C	
1	1	Rsce = – 32 768
0	0	Rsce = 0
0	1	autres cas

Alarme possible : aucune

Alarmespossibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

NORM

normalize (NORM)



Option : virgule flottante câblée ((05)) ou programmée

Code : premier mot : 38__
second mot : 2000

Opération effectuée : le nombre flottant contenu dans les registres A et B est normalisé (voir § 1.3.1)
Les indicateurs V et C sont positionnés de la manière indiquée § 1.3.3

Interviennent : - A et B
- K (option programmée)

Sont modifiés : - ST
- la première mémoire du COMMON

Indicateurs :

V	C	
1	0	overflow
0	1	underflow
0	0	résultat correct

Alarmes possibles- : - protection mémoire (05)
- mémoire inexistante (05)
- parité

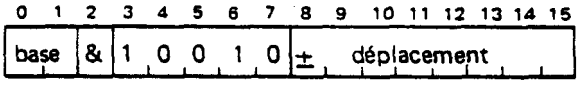
ATTENTION :
- dans le cas de l'option programmée, 26 mots libres sont nécessaires dans la pile pointée par K

- dans le cas de l'option câblée une panne de l'opérateur câblé peut provoquer l'appel du flottant programmé (SVC 0) comme en l'absence de l'opérateur.

- dans le cas d'un débordement, le contenu des registres A et B est sans signification et dépend de l'option.

or (Or)

OR



Code

	direct	indirect
base C	52__	72__
base L	92__	B2__
base W	D2__	F2__

Adressage : mémoire (mot)

Opération effectuée : le résultat de l'union logique des 16 bits du registre A et du mot mémoire adresse est rangé dans le registre A (union: l'union de 1 et 0, 0 et 1, 1 et 1 vaut 1, l'union de 0 et 0 vaut O).

Interviennent : . le mot mémoire adresse
. A

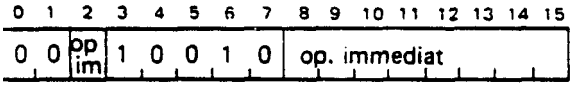
Est modifie : A

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

or immediate (ORI)

ORI



Code :

bit 2 = 0	12__
bit 2 = 1	32__

Adressage : immédiat (9 bits)

Opération effectuée : le résultat de l'union logique des 16 bits du registre A et de l'opérande immédiat est rangé dans le registre A. L'opérande immédiat est étendu sur 16 bits, ses huit bits poids fort étant identiques au bit 2 de l'instruction, ses huit bits poids faible étant identiques aux bits 8 à 15 de l'instruction.

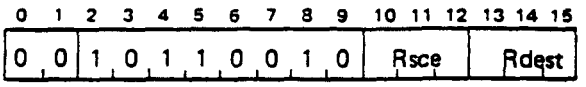
On peut ainsi effectuer une union logique entre A et un opérande immédiat compris entre '0000 et '00FF ou entre 'FF00 et 'FFFF.

Interviennent : . l'opérande immédiat (sur 16 bits)
. A

Est modifie : A

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Code : 2C8_

Adressage : registre - registre

Opération effectuée : le résultat de l'union logique des 16 bits du registre source et du registre destination est chargé dans le registre destination.

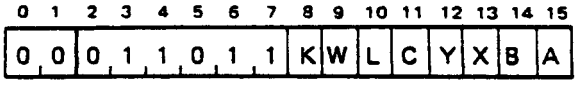
On peut ainsi faire l'union logique entre deux quelconques des huit registres A B X Y C L W K.

Interviennent : deux des registres A B X Y C L W K.

Est modifié : un des registres A B X Y C L W K.

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 03
- parité



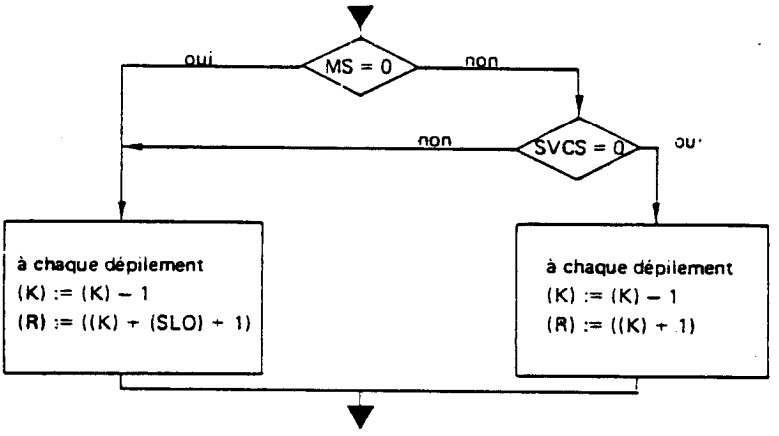
Code : 1 B _ _

Opération effectuée : les registres précisés dans l'instruction sont chargés successivement, dans l'ordre K W L C Y X B A, les deux opérations suivantes étant répétées pour chacun d'eux :
- diminution de 1 de la valeur de K
- chargement du registre par le mot suivant le mot pointé par K.

K doit pointer avant l'exécution de l'instruction sur le mot mémoire où se trouve le contenu du premier registre à restaurer. A la fin de l'exécution de l'instruction la valeur de K a diminué du nombre de registres chargés.

On peut ainsi : restaurer à l'aide d'une seule instruction le ou les registres sauvegardés par une instruction PSR.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue
- si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.



Interviennent : K, ST
les mémoires pointées par K

Sont modifiés : K
les registres précisés dans l'instruction

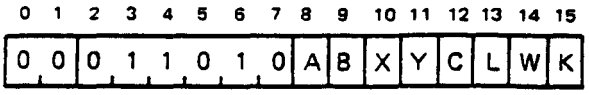
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : – le chargement de plus de 4 registres par une seule instruction PLR peut causer une dégradation des performances du canal microprogrammé (LDC) intégré au processeur, ainsi que de la réponse en interruption du processeur car cette instruction n'est pas interruptible par ces appels.

– K pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande : ici le prochain registre à charger.

– le programme peut passer en mode esclave à la suite d'une alarme.



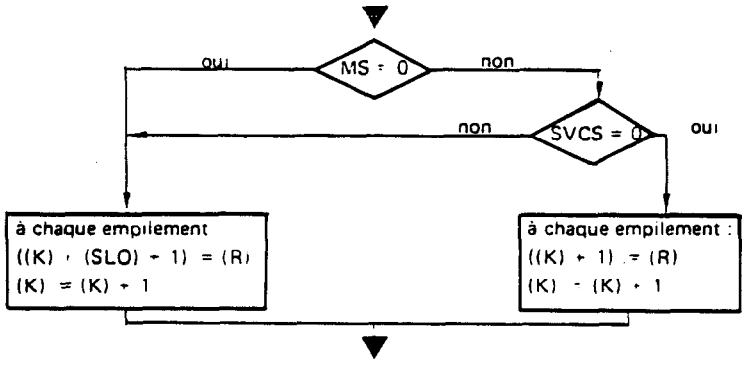
Code : 1 A _ _

Opération effectuée : les registres précisés dans l'instruction sont rangés successivement, dans l'ordre A B X Y C L W K, les deux opérations suivantes étant répétées pour chacun d'eux :
- rangement du contenu du registre dans le mot suivant le mot pointé par K
- augmentation de 1 de la valeur de K

A la fin de l'exécution de l'instruction la valeur de K a progressé du nombre de registres rangés, K pointe sur le mot mémoire contenant le dernier registre rangé.

On peut ainsi : sauvegarder à l'aide d'une seule instruction un ou plusieurs registres. Ces registres seront ensuite restaures à l'aide de l'instruction PLR dont le fonctionnement est symétrique.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue
- si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.



Interviennent : K, ST
les registres précisés dans l'instruction

Sont modifiés : K
les mémoires pointées par K

Indicateurs : non modifiés

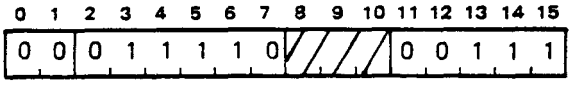
Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

ATTENTION : – le rangement de plus de 4 registres par une seule instruction PSR peut causer une dégradation des performances du canal microprogrammé (LDC) intégré au processeur, ainsi que de la réponse en interruption du processeur car cette instruction n'est pas interruptible par ces appels.

– lorsque le registre K est sauvegardé par une instruction PSR c'est le contenu de K diminué de 1 qui est rangé (si K est rangé en '1000, '1000 contiendra '0FFF). L'instruction PLR, en augmentant de 1 le contenu de K, en assurera une restauration correcte.

– K pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande : ici le dernier registre rangé.

– le programme peut passer en mode esclave à la suite d'une alarme.



Code : 1E07

Opération effectuée : si l'octet droit du registre A comporte un nombre pair de bits à 1, l'indicateur C est mis à 0. S'il contient un nombre impair de bit à 1, l'indicateur C est mis à 1.

On peut ainsi tester la parité d'un caractère ASCII 8 bits.

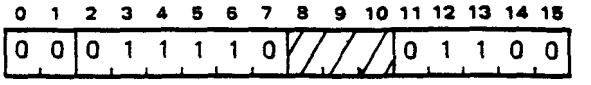
Intervient : A (bits 8 a 15)

Est modifié : ST

Indicateurs :

V	C	
0	0	octet pair
0	1	octet impair

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

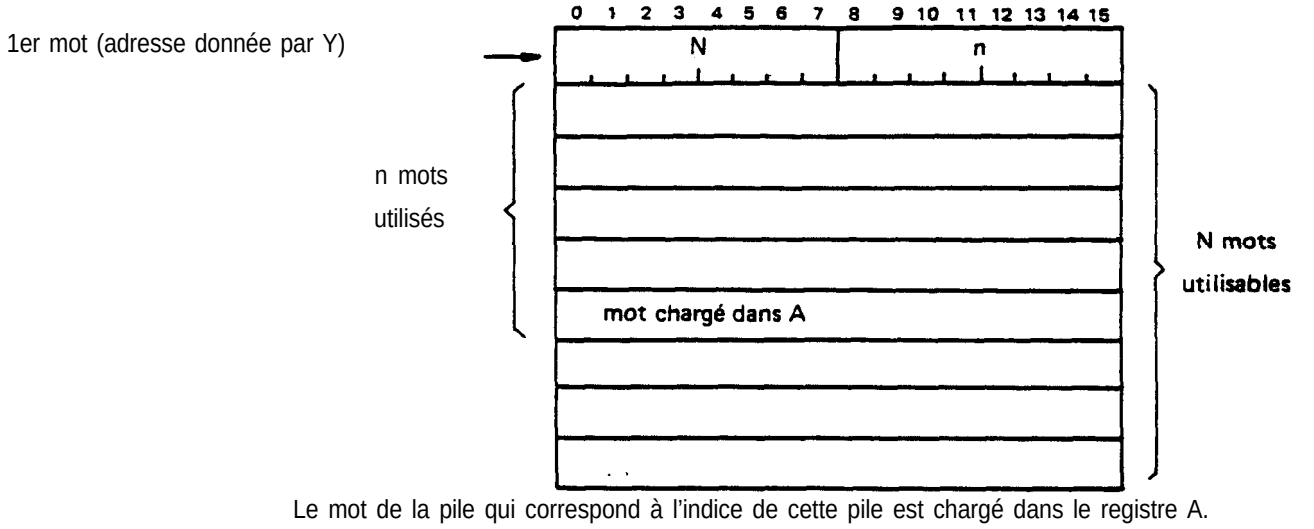


Option : **Scheduler sur (05), standard sur (40) et (65)**

Code : 1E0C

Opération effectué : Cette instruction porte sur une pile organisée de la manière suivante :
L'octet poids fort du premier mot contient le nombre maximum N de mots utilisables dans la pile (le premier mot n'étant pas compté), l'octet poids faible **contient l'indice n de la pile (tel que 1 ⩽ n ⩽ N, si la pile est utilisée correctement)**

Les informations sont prises dans les N mots qui suivent ce premier mot.



Le mot de la pile qui correspond à l'indice de cette pile est chargé dans le registre A. puis l'indice est diminué de 1.

L'adresse de la pile est donnée par le registre Y qui pointe sur le premier mot qu'elle occupe en mémoire.

Si, après le chargement de A par PULL, n'est égal à 0 l'indicateur C est positionné à 1.

Si, avant dépilement, n vaut 0, l'indicateur V est mis à 1, la pile et son pointeur restent inchangés.

Intervient : la pile pointée par le registre Y

Sont modifiés : A, ST
la pile pointée par le registre Y

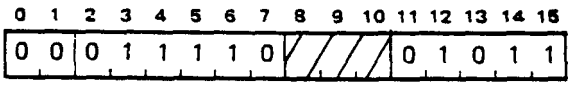
Indicateurs :

V	C	
0	0	pile non vide
0	1	pile vide
1	0	pile déjà vide

Alarmes possibles : - protection mémoire (03)
- mémoire inexistante (04)
- parité
- **instruction optionnelle inexistante sur (05) sans scheduler.**

PUSH

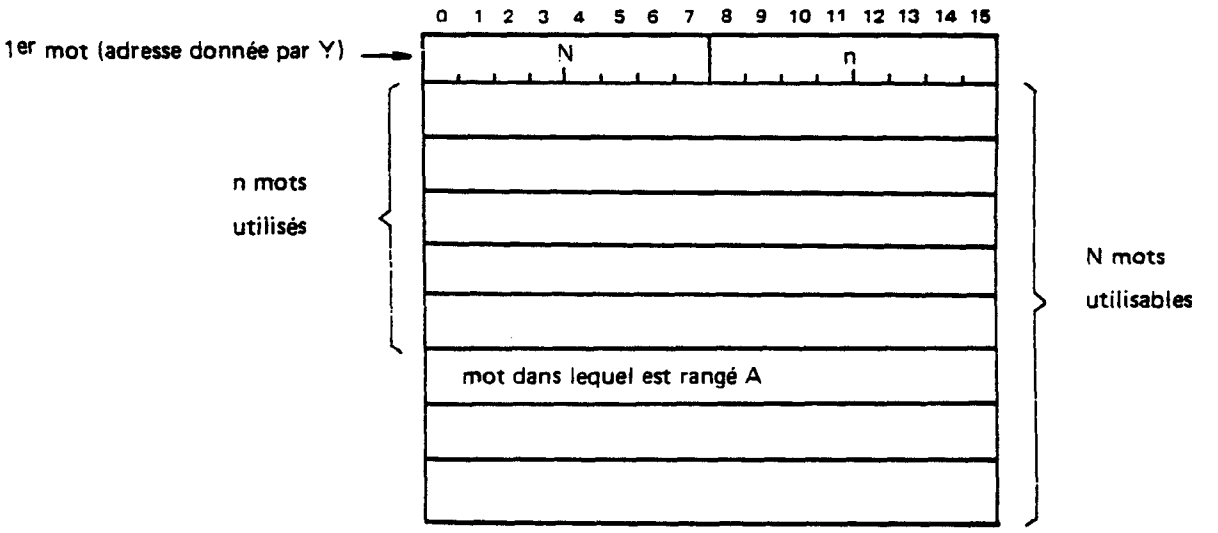
push A (PUSH)



Option : Scheduler sur (05), standard sur (40) et (65)

Code : 1 E 0 B

Opération effectuée : Cette instruction porte sur une pile organisée de la manière suivante : l'octet poids fort du premier mot contient le nombre maximum N de mots utilisables dans la pile (le premier mot n'étant pas compté), l'octet poids faible contient l'indice n de la pile (tel que $1 \leqslant n \leqslant N$ si la pile est utilisée correctement). Les informations sont rangées dans les N mots qui suivent ce premier mot.



L'indice de la pile est augmenté de 1, puis le contenu du registre A est rangé dans la pile à l'adresse correspondant à la valeur de l'indice. L'adresse de la pile est donnée par le registre Y qui pointe sur le premier mot qu'elle occupe en mémoire.

Si, après le rangement dans la pile par PUSH, n est égal à N, l'indicateur C est positionné à 1.

Si, avant rangement, n est supérieur ou égal à N, l'indicateur V est mis à 1, la pile et son pointeur restent inchangés.

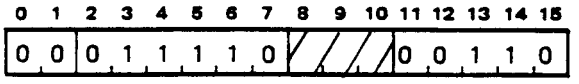
Interviennent : la pile pointée par le registre Y
A

Sont modifiés : la pile pointée par le registre Y, ST

Indicateurs :

V	C	
0	0	pile non saturée
0	1	pile saturée
1	0	pile déjà saturée

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction optionnelle inexistante sur (05) sans scheduler.



Option : scheduler

Code : 1E06

Intervient : NS

Opération effectuée : l'instruction QUIT désarme la tâche différée en cours en remettant à zéro le bit de la file ASTF qui correspond au niveau de cette tâche et provoque le lancement d'une tâche différée moins prioritaire.

Les indicateurs IOM et IPM sont remis à 0.

Comme dans tout changement de contexte, le contexte de la tâche qui effectue le QUIT est sauvegardé dans la PST correspondante.

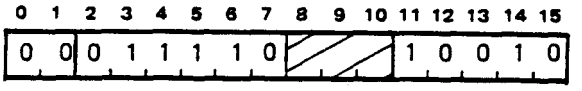
Sont modifiés : - NS et tous les registres (changement de contexte)
– ST

Indicateurs : IOM, IPM et ceux du nouveau contexte

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (06)
- parité
- instruction optionnelle inexistante
- exécution interdite sous niveau hardware

reset breakpoint (RBP)

RBP



Mode : maître et mise au point

Code : 1E12

Opération effectuée : efface le point d'arrêt dont l'adresse est contenue dans Y en réécrivant le mot correspondant avec une parité normale quelle que soit sa parité antérieure.

Si l'option DRPS est présente, elle est activée : Y contient alors une adresse relative à SLO et peut provoquer une alarme protection mémoire.

Interviennent : Y, SLO, SLE

Est modifiée : la parité du mot pointé par Y

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante 05
- protection mémoire 05
- parité mémoire
- instruction privilégiée

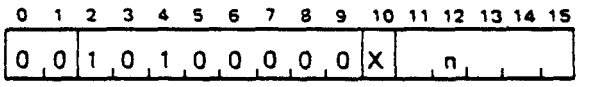
ATTENTION :

- cette instruction doit normalement être exécutée en mode mise au point ; le point d'arrêt antérieur est alors ignoré ; sinon il provoque une alarme parité.
- le programme peut passer en mode esclave à la suite d'une alarme.

---

reset bit (RBT)

RBT



Code : 280

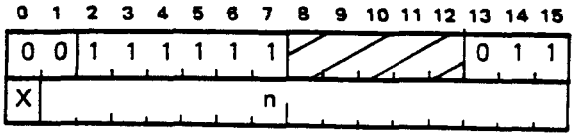
Adressage : indexation possible

Opération effectuée : les registres A et B étant considérés comme un registre unique de 32 bits, le bit dont le rang est n si l'instruction n'est pas indexée, n + x modulo 32 si l'instruction est indexée, est mis a zéro.

Est modifié : un bit de A ou un bit de B

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 05
- parité



option : **CDA16** (65) **et** (70)

Code : premier mot 3F03
deuxième mot n plus le bit 0 si indexation

Adressage : indexation possible sur le rang du bit.

Opération effectuée : mise à zéro du bit spécifié dans une file de bits en mémoire

- le registre A contient l'adresse du mot origine de la file (bits 0 à 15 de la chaîne).
- le rang du bit modifié est indiqué sur les bits 1 à 15 du deuxième mot de l'instruction.
- ce rang est indexé par X si le bit 0 du deuxième mot de l'instruction est à un. Le rang spécifié peut alors devenir négatif.

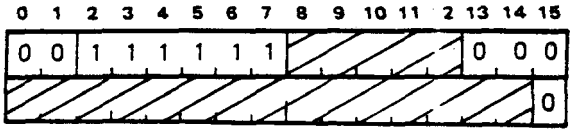
En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

Interviennent : A, X, la file de bits pointée par A...

Est modifié : le bit spécifié dans la file

Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante
- protection mémoire
- parité mémoire



Option : CDA16 (65), (70) et (35)

Code : premier mot 3F00
deuxième mot 0000

Opération effectuée : transfert d'un segment de mémoire situé dans la zone de données communes (CDA) vers une zone mémoire relative à la tâche en cours :

- le registre A contient l'adresse, relative au début de la zone de données communes, du 1er mot source.
- le registre B contient l'adresse du 1er mot destination. Cette adresse est absolue ou relative à SLO selon l'état maître ou esclave du programme.
- le registre X contient le nombre de mots à transférer.

Les adresses début et fin de la zone CDA sont contenues dans les mots d'adresses absolues '18 et '19. Ces mots ont le même format que SLO : ils contiennent les 16 bits poids fort d'une adresse sur 20 bit, sachant que la zone CDA ne peut s'implanter qu'à des adresses multiples de 16 et ne peut chevaucher une frontière de 64 Kmots.

Le transfert s'effectue mot par mot par adresses décroissantes et les zones peuvent se recouvrir.
A chaque transfert, le registre X est diminué de 1 et les interruptions (ainsi que le défaut secteur) sont prises en compte.
A la fin du transfert, le registre X est nul.
L'opération est ineffective si le contenu du registre X est négatif après décrémentation.

On peut ainsi lire au maximum 32768 mots d'une zone de données, de 64 Kmots au maximum, commune à toutes les tâches d'un système.

La protection d'accès à la zone CDA est effectuée globalement avant tout transfert et, le cas échéant, provoque une alarme protection mémoire.
En l'absence de l'option DRPS, les mots d'adresse '18 et '19 continuent à définir la zone CDA mais leurs quatre bits de poids fort doivent être nuls.

En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

Interviennent : A, B, X, SLO, SLE, les mots mémoire d'adresse '18 et '19, les zones pointées par A et B.

Sont modifiés : X, la zone pointée par B

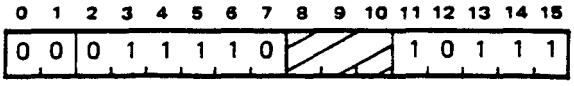
Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante
- protection mémoire
- parité mémoire

Interruptibilité : l'instruction est interruptible à chaque transfert de mot y compris par le défaut secteur.

read HV (RDHV)

RDHV



Mode : maître

Code : 1 E 1 7

Opération effectuée : charge dans l'accumulateur le contenu du registre HV. On peut ainsi connaître la liste des tâches immédiates en cours de traitement (commencées et encore inachevées).

Intervient : HV

Est modifié : A

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante 05
- protection mémoire 05
- parité mémoire
- instruction privilégiée

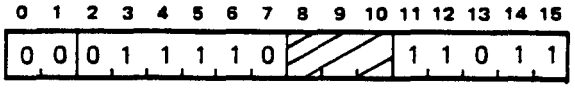
ATTENTION : le contenu du registre HV n'est pas identique à la liste des interruptions en attente :

- certaines interruptions ont déjà pu être traitées et ne sont plus en attente mais la tâche immédiate correspondante n'est pas terminée,
- certaines interruptions moins prioritaires ou masquées ont pu apparaître sans que leur traitement ait encore commencé.

---

read SLO - SLE (RDOE)

RDOE



Mode : maître

Code : 1 E 1 B

Opération effectuée : transfert du contenu des registres SLO et SLE, respectivement dans les registres A et B.

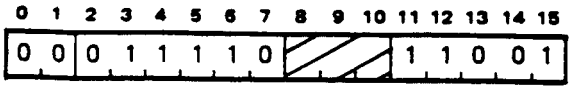
En l'absence de l'option DRPS, les registres A et B sont remis à 0.

Interviennent : SLO, SLE

Sont modifiés : A, B

Indicateurs : non modifiés

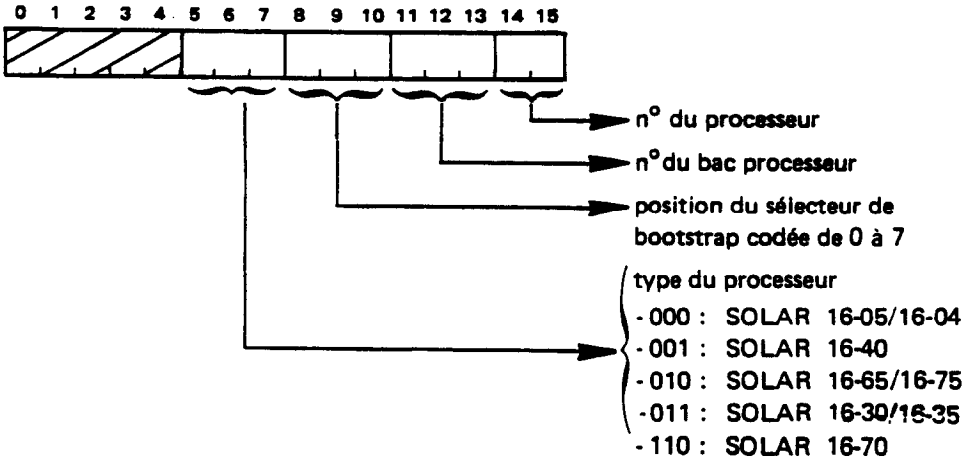
Alarmes possibles : - mémoire inexistante 05
- protection mémoire 05
- parité mémoire
- instruction privilégiée



Mode : **maître**

Code : **1 E 1 9**

Opération effectuée : charge dans l'accumulateur des informations relatives à l'identité du processeur suivant le format suivant :



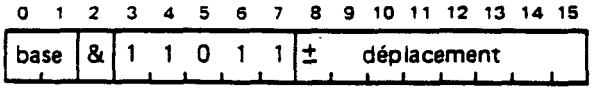
Est modifie : A

Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

Note : Les types 4 à 7 sont réservés pour usage futur.

release (RLSE)



Option : scheduler

Mode : maître

Code :

	direct	indirect
base C	5B__	7B__
base L	9B__	BB__
base W	DB__	FB__

Adressage : mémoire (1er mot du sémaphore)

Opération effectuée : le compteur du sémaphore d'exclusion mutuelle adressé par l'instruction est augmenté de 1.

- si après cette modification le compteur a une valeur positive la tâche qui effectue l'instruction RQST se poursuit
- si après cette modification le compteur a une valeur nulle ou négative :

  le bit a 1 de la file du sémaphore dont le rang n est égal au n° de la tâche la plus prioritaire est remis à 0 ;

  le bit de la file ESTF de même rang n est mis à 1 (la tâche n est démasquée) ;
- si la tâche qui effectue l'instruction RQST est plus prioritaire, elle se poursuit normalement, sinon le scheduler effectue un changement de contexte au profit de la tâche démasquée.

Les indicateurs IOM et IPM sont remis à 0 (avant passage éventuel dans le scheduler).

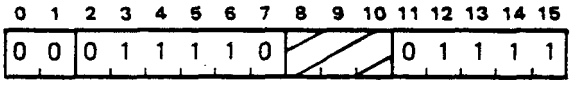
Interviennent :
- le sémaphore adressé
- NS

Sont modifiés :
- le sémaphore adressé
- NS et tous les registres s'il y a-changement de contexte
- ST

Indicateurs : IOM, IPM, plus le changement de contexte éventuel

Alarmes possibles :
- protection mémoire (03)
- mémoire inexistante (05)
- parité
- instruction privilégiée
- instruction optionnelle inexistante

ATTENTION : L'octet gauche du sémaphore est remis à 0.



Option : microprogramme non standard

Mode : maître

Code : 1E0F

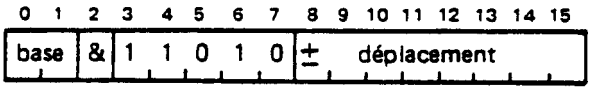
Opération effectuée : cette instruction déclenche l'exécution du microprogramme, la première micro-instruction effectuée étant à l'adresse indiquée par le contenu du registre Y.

On peut ainsi lorsqu'on dispose d'une mémoire morte comportant des microprogrammes spécifiques d'une application déclencher par programme l'exécution de ces micropro-grammes.

Intervient : Y

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction privilégiée

ATTENTION : L'exécution d'une instruction ROMB à une adresse de microprogramme quelconque, située dans le microprogramme standard ou à une adresse inexistante, peut faire croire à un mauvais fonctionnement du calculateur.



Option : scheduler

Mode : maître

Code :

	direct	indirect
base C	5A__	7A__
base L	9A__	BA__
base W	DA__	FA__

Opération effectuée : le compteur du sémaphore d'exclusion mutuelle adressé par l'instruction est diminué de 1.

- si après cette modification le compteur a une valeur positive ou nulle, la tâche qui effectue l'instruction RQST se poursuit.
- si après cette modification le compteur a une valeur négative :

  le bit de la file du sémaphore dont le rang n ($0 \leq n \leq 127$) est égal au n° de la tâche qui effectue l'instruction RQST est mis à 1 ;

  le bit de la file ESTF dont le rang est égal au n° de la tâche qui effectue l'instruction RQST est mis à 0 (la tâche est masquée) ;

  le contexte de cette tâche est sauvegardé et une tâche moins prioritaire est lancée par le scheduler.

Les indicateurs IOM et IPM sont remis à 0 (avant passage éventuel dans le scheduler).

Adressage : mémoire (1er mot du sémaphore)

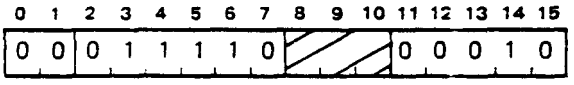
Interviennent :
- le semaphore adresse
- NS

Sont modifiés :
- le sémaphore adressé
- NS et tous les registres s'il y a changement de contexte
- ST

Indicateurs : IOM, IPM, plus le changement de contexte éventuel

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (06)
- parité
- instruction privilégiée
- instruction optionnelle inexistante
- exécution interdite sous niveau immédiat

ATTENTION : L'octet gauche du sémaphore est remis à 0.



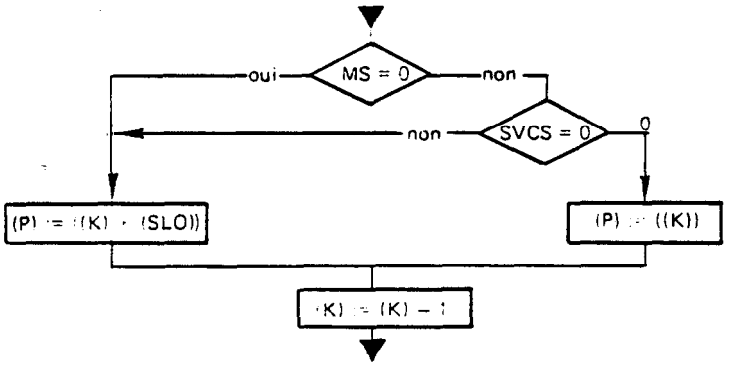
Code : 1 E 0 2

Opération effectuée : le mot mémoire pointé par K est chargé dans le registre P, puis le contenu du registre K est diminué de 1.

On peut ainsi retourner à la fin d'un sous-programme vers l'instruction suivant le BSR qui a appelé ce sous-programme.

Les appels imbriqués de sous-programme sont possibles grâce à la soustraction de 1 au registre K effectuée par l'instruction RSR, cette évolution de K pour la restauration de l'adresse de retour étant de plus homogène avec l'évolution de K pour les restaurations effectuées par RSV et PLR.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
– si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue.
– si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.



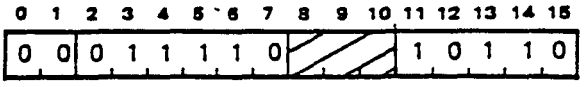
Interviennent : la mémoire pointée par K
K, ST

Sont modifiés : P
K

Indicateurs : non modifiés

Alarmes possibles : – protection mémoire (05)
– mémoire inexistante (03)
– parité

ATTENTION : – K pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande : ici le prochain point de retour.
– le programme peut passer en mode esclave à la suite d'une alarme.



Mode : martre

Code : 1 E 1 6

Opération effectuée : force à zéro les bits du registre ST spécifiés dans l'accumulateur et charge la nouvelle valeur de ST dans l'accumulateur.

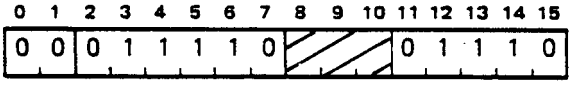
Interviennent : A, ST

Sont modifiés : A, ST

Indicateurs : modifiés selon le contenu de A

Alarmes possibles : - mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

ATTENTION : la modification de ST est une opération extrêmement délicate ; toute erreur peut avoir des conséquences très importantes voire imprévisibles.
Dans la pratique on se limitera à la manipulation de LCM et IPM.
La remise à zéro du bit IOM par cette instruction n'est pas équivalente à l'exécution de l'instruction EIT car les interruptions en attente ne sont alors pas testées systématiquement.
Le forçage à zéro de certains bits est ineffectif.



Code : 1 E 0 E

Opération effectuée : Le mot mémoire pointé par K est chargé dans le registre P, puis le registre K est diminué de 1 et les indicateurs MS et SVCS sont remis à 0.

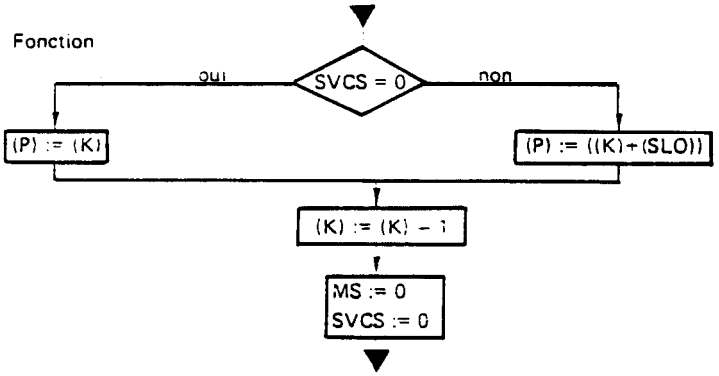
Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue,
- si l'instruction est exécutée en mode esclave (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur (cf SVC) appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.

On peut ainsi retourner à la fin d'une requête superviseur vers l'instruction suivant le SVC qui a appelé ce sous-programme.

On peut aussi appeler un programme en mode esclave depuis un programme maître (MS = 1, SVCS = 0) en ayant empilé l'adresse relative à SLO du programme appelé à travers K encore absolu.

Cette instruction est interdite sur (05) où le mode esclave n'existe pas.

L'évolution du registre K pour la restauration de l'adresse de retour est homogène avec l'évolution de K pour les restaurations effectuées par RSR et PLR.



Interviennent : K, ST
la mémoire pointée par K

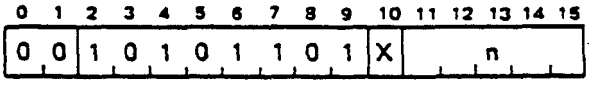
Sont modifiés : K, P, ST

Indicateurs : MS et SVCS

Alarmes possibles : - protection mémoire (06)
- mémoire inexistante (05)
- parité
- instruction inexistante **sur** (05)

ATTENTION : – K pointe toujours sur la mémoire occupée de la pile d'adresse la plus élevée : ici le prochain point de retour.
– Le programme peut passer en mode esclave à la suite d'une alarme..

shift arithmetic right double (SARD) — SARD



Code : 2 B 4_

Adressage : indexation possible

Opération effectuée : les 32 bits des registres A et B considérés comme un registre unique sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + x modulo 32 si l'instruction est indexée.

Les p bits de droite sont perdus
Les p + 1 bits de gauche prennent la valeur du bit 0
L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 32 - p.

On peut ainsi :

- charger un nombre algébrique de 16 bits dans les registres A et B considérés comme un registre unique de 32 bits (SARD 16).

Si p = 0, l'indicateur C est remis à zéro.
On peut ainsi :

- charger un nombre algébrique de 16 bits dans les registres A et B considérés comme un registre unique de 32 bits (SARD 16).
- diviser algébriquement le contenu de A B par 2^p.

Interviennent : . A
. B

Sont modifiés : . A, B, ST

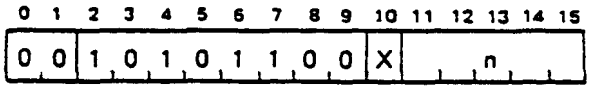
Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

shift arithmetic right single (SARS) — SARS



Code : 2 B 0_

Adressage : indexation possible

Opération effectuée : les 16 bits du registre A sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + x modulo 32 si l'instruction est indexée.

Les p bits de droite sont perdus.
Les p + 1 bits de gauche prennent la valeur du bit 0.
L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 16 - p.

Si p = 0, l'indicateur est remis à zéro.
Si p > 16, l'indicateur prend la valeur du signe de A.

On peut ainsi diviser algébriquement le contenu de A par 2P

Intervient : A

Sont modifiés : . A, ST

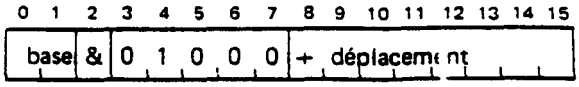
Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

subtract (SB)

SB



Code :

	direct	indirect
base C	48__	68__
base L	88__	A8__
base W	C8__	E8__

Adressage : mémoire (mot)

Opération effectuée : le contenu du mot mémoire adressé est soustrait au registre A.
Les indicateurs V et C sont positionnés de la manière indiquée aux § 1.1 et 1.2.

Interviennent : . A
. le mot mémoire adresse

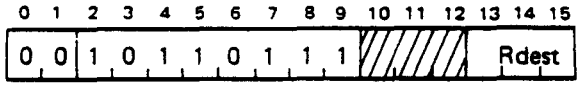
Est modifié : A, ST

Indicateurs : V : débordement soustraction
C : report soustraction

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

subtract carry from register (SBCR)

SBCR



Code : 2DC_

Adressage : registre - registre

Opération effectuée : la valeur de l'indicateur C est soustraite au registre destination.
Les indicateurs V et C sont positionnés de la manière indiquée aux § 1.1 et 1.2. On peut ainsi effectuer des soustractions en double longueur, le report de la soustraction des poids faibles étant soustrait aux forts.

Interviennent : . un des registres A B X Y C L W K
. indicateur C

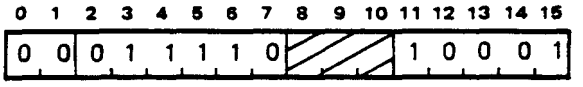
Sont modifiés : . un des registres A B X Y C L W K
. ST

Indicateurs : V : débordement soustraction
C : report soustraction

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

set breakpoint (SBP)

SBP



Mode : maître et mise au point

Code : 1E11

Opération effectuée : crée un point d'arrêt en inversant la parité normale du mot mémoire dont l'adresse est contenue dans Y.

Si l'option DRPS est présente, elle est activée : Y contient alors une adresse relative à SLO et peut provoquer une alarme protection mémoire.

Interviennent : Y, SLO, SLE

Est modifiée : la parité du mot pointé par Y

Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante (05)
protection mémoire (05)
parité mémoire
instruction privilégiée

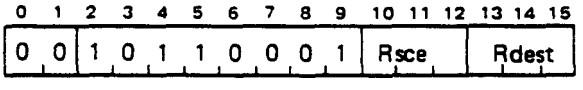
ATTENTION :

- cette instruction doit être exécutée en mode mise au point si l'on veut ignorer le point d'arrêt antérieur éventuel.
- le programme peut passer en mode esclave à la suite d'une alarme.

---

subtract registers (SBR)

SBR



Code : 2C4_

Adressage : registre - registre

Opération effectuée : le contenu du registre source est soustrait du registre destination.
Les indicateurs C et V sont positionnés de la manière indiquée aux § 1.1 et 1.2. On peut ainsi soustraire un quelconque des huit registres A B X Y C L W K d'un autre de ces registres.

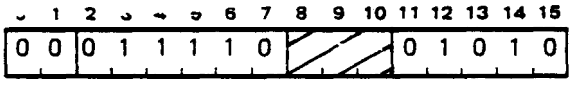
Interviennent : deux des registres A B X Y C L W K

Sont modifiés : . un des registres A B X Y C L W K
. ST

Indicateurs : V : débordement soustraction
C : report soustraction

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

$$R_d := R_d - R_s$$



Option : scheduler sur (05), standard sur (40) et (65)

Code : 1 E 0 A

Opération effectuée : les 16 bits du registre A sont comparés aux 8 bits de l'octet mémoire situé dans la chaîne d'octets dont le premier mot est pointé par le registre B, et qui a comme rang dans cette chaîne la valeur du registre X (le rang étant compté à partir de zéro)

S'il y a égalité, l'indicateur C est mis à zéro et l'opération est terminée.
S'il n'y a pas égalité, le registre X est augmenté de 1, si X <Y, l'exécution de l'instruction est répétée, sinon (X ⩾Y) l'indicateur C est mis à un et l'opération est terminée.

On peut ainsi :

- rechercher un octet ayant une configuration donnée dans une chaîne d'octets, les bits B à 15 de A contenant l'octet cherché (avec 0 sur les bits 0 à 7), B l'adresse début de la chaîne, X le rang du 1er octet à examiner, Y le rang du 1er octet non examiné.
- en faisant progresser X de 1 sans modifier A, B, Y après une recherche qui a abouti, effectuer la recherche du même octet dans la suite de la chaîne.

Interviennent : A (octet cherché)
B (adresse début de la chaîne)
X (rang du 1er octet à examiner)
Y (rang du 1er octet non examiné)

Sont modifié : X, ST

indicateurs :

V	C	
0	0	l'octet a été trouvé
0	1	l'octet n'a pas été trouvé

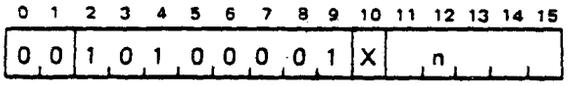
Alarmes possibles : protection mémoire (05)
mémoire inexistante (05)
parité
instruction optionnelle inexistante (05) sans scheduler

Interruptibilité : l'instruction est interruptible entre chaque comparaison d'un octet.

ATTENTION : - pour que la recherche ait une signification l'octet gauche de A doit être nul
- si X ⩾Y aucune recherche n'est effectuée et l'indicateur C est mis à un.
- ne pas utiliser X < 0.

set bit (SBT)

SBT



Code : 284_

Adressage : indexation possible

Opération effectuée : les registres A et B étant considérés comme un registre unique de 32 bits, le bit dont le rang est n si l'instruction n'est pas indexée, n + x modulo 32 si l'instruction est indexée, est mis à un.

Est modifié : un bit de A ou un bit de B.

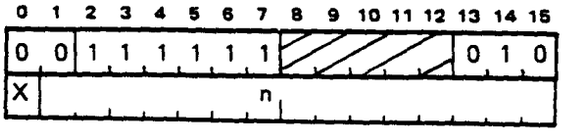
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

---

set bit in memory (SBTM)

SBTM



Option : **CDA16** (65) **et** (70)

Code : premier mot 3F02
deuxième mot n plus le bit 0 si indexation

Adressage : indexation possible sur le rang du bit.

Opération effectuée : mise à un du bit spécifié dans une file de bits en mémoire
- le registre A contient l'adresse du mot d'origine de la file (bits 0 à 15 de la chaîne).
- le rang du bit modifié est indiqué sur les bits 1 à 15 du deuxième mot de l'instruction.
- ce rang est indexé par X si le bit 0 deuxième mot de l'instruction est à un. Le rang spécifié peut alors devenir négatif.

Interviennent : A, X, la file de bits pointée par A.

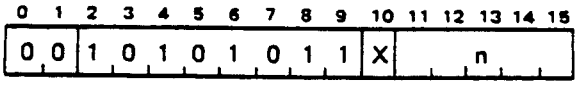
Est modifié : le bit spécifié de la file

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante
- protection mémoire
- parité mémoire

shift circular left double (SCLD)

SCLD



Code : 2 A C_

Adressage : indexation possible

Opération effectuée : les 32 bits des registres A et B considérés comme un registre unique sont décalés à gauche d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n +X modulo 32 si l'instruction est indexée.

Les p bits de droite sont remplacés par les bits sortis à gauche.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position p - 1.

Si p = 0 l'indicateur est mis à zéro.

Interviennent : . A
. B

Sont modifiés : . A, B, ST

Indicateurs :

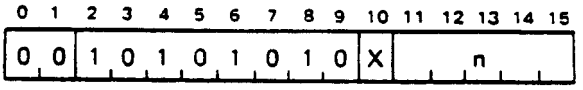
V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 0

Alarmes possibles :- protection mémoire (05)
- mémoire inexistante (05)
- parité

---

shift circular left single (SCLS)

SCLS



Code : 2 A 8_

Adressage : indexation possible

Opération effectuées : les 16 bits du registre A sont décalés à gauche d'un nombre p de positions, égal à n si l'instruction n'est pas indexée, à n + X modulo 32 si l'instruction est indexée.

Les p bits de droite sont remplacés par les bits sortis à gauche.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position p - 1.

Si p = 0 l'indicateur est mis à zéro.

Intervient : A

Sont modifiés: : A, ST

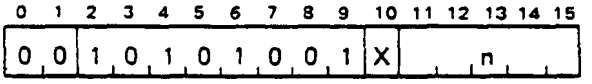
Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

shift circular right double (SCRD)

SCRD



Code : 2 A 4_

Adressage : indexation possible

Opération effectuée : les 32 bits des registres A et B considérés comme un registre unique sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + X modulo 32 si l'instruction est indexée.

Les p bits de gauche sont remplacés par les bits sortis à droite.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 32 - p. Si p = 0 l'indicateur est remis à zéro.

Interviennent : . A
. B

Sont modifiés : . A, B, ST

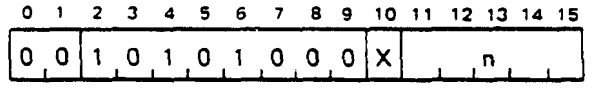
indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

shift circular right single (SCRS)

SCRS



Code : 2 A 0_

Adressage : indexation possible

Opération effectuée : les 16 bits du registre A sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + X modulo 32 si l'instruction est indexée.

Les p bits de gauche sont remplacés par les bits sortis à droite.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 16 - p.

Si p = 0 l'indicateur est mis à zéro.

Intervient : A

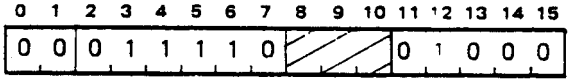
Sont modifiés : . A, ST

Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : -protection mémoire (05)
- mémoire inexistante (05)
- parité

set carry (SCY)



Code : 1E08

Opération effectuée : l'indicateur C est mis à 1. SCY est la seule instruction qui modifie un des indicateurs en laissant inchangé l'autre indicateur.

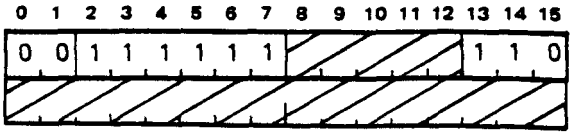
on peut ainsi positionner les indicateurs d'une manière simple :

```
V = 0   C = 0 : ADRI   0,A
V = 0   C = 1 : ADRI   0,A
                SCY
V = 1   C = 0 : CPR    A,A
V = 1   C = 1 : CPR    A,A
                SCY
```

Est modifié : ST

Indicateurs : V inchangé
C mis à 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Option : CDA16 (65) et (70)

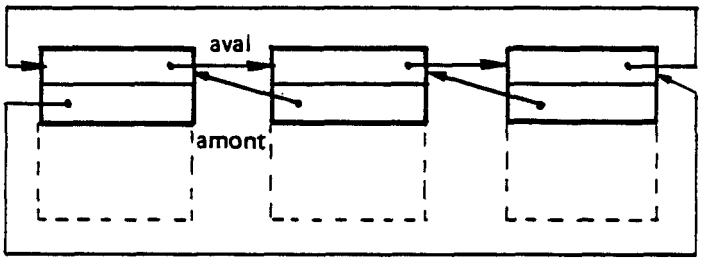
Code : premier mot 3F06
deuxième mot 0000

Opération effectuée : supprime dans une liste l'élément qui précède celui dont l'adresse est dans l'accumulateur et en donne l'adresse dans l'accumulateur.
L'élément supprimé n'est pas modifié.
Si l'élément supprimé est seul dans la liste, l'opération est ineffective.

La liste est une suite de doubles mots chaînés dans les deux sens :

- le premier mot contient l'adresse du double mot suivant (chaînage aval),
- le second mot contient l'adresse du double mot précédent (chainage amont),

A chaque élément de la liste on peut associer un ou plusieurs mots qui constituent l'information propre à cet élément. Ces informations ne sont pas modifiées par l'instruction.



On peut ainsi gérer une liste en "premier entré - premier sorti" (FIFO). L'élément pointé par A constituera une "tête de liste". Une liste vide se résumera à sa "tête". On insérera un nouvel élément à la suite de la "tête" par l'instruction INSQ. On sortira le premier élément entré par l'instruction SFQ. Aucun masquage n'est alors nécessaire en multitraitement sur un monoprocesseur.

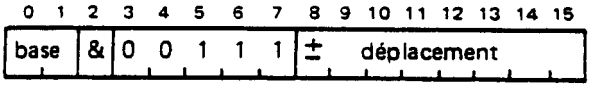
En l'absence de l'option scheduler, cette instruction est'interprétée comme une instruction SVC 7 (cf codes extension).

Interviennent : A, l'élément pointé par A et son prédécesseur.

Sont modifiés : A, l'élément pointe par A et le prédécesseur de l'élément supprimé.

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante
- parité mémoire
- protection mémoire



Mode : maître

Code :

	direct	indirect
base C	47__	67__
base L	87__	A7__
base W	C7__	E7__

Adressage : mémoire (mot)

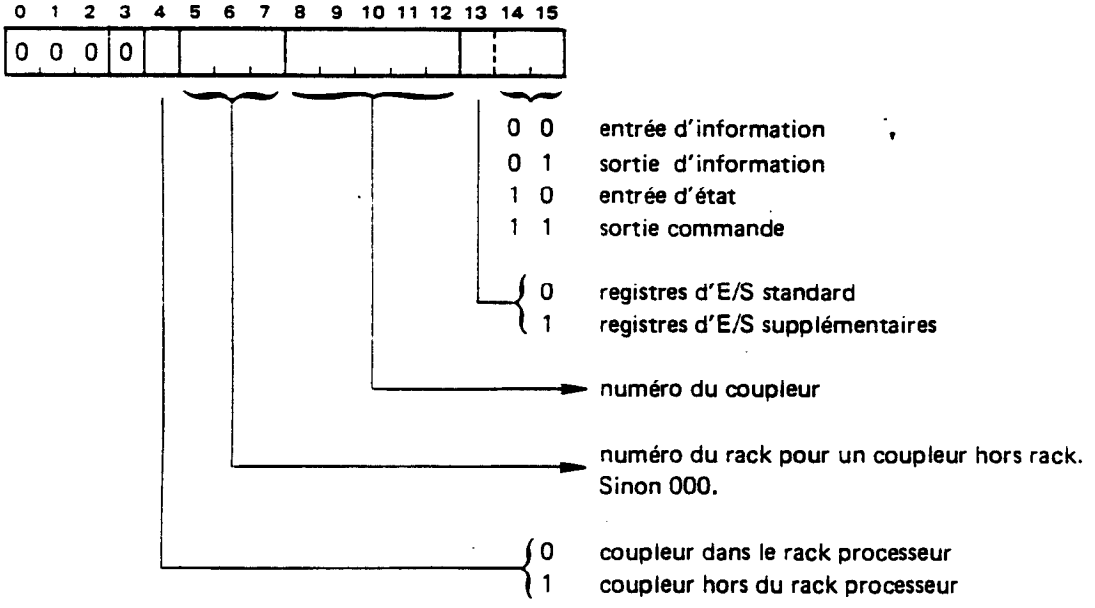
Opération effectuée : cette instruction déclenche les différentes opérations d'entrée/sortie possibles sur SOLAR (lectures, écritures, envoi de commandes aux périphériques, entrées de mots d'état) pour tous les périphériques qui y sont connectables, et ceci dans chacun des modes d'échange possibles (mode programmé simple, mode programmé avec gestion par interruption, mode canal).

Dans le cas d'une entrée, le registre A reçoit l'information ou l'état provenant du coupleur de périphérique. Si l'information est un octet, cet octet est chargé en poids faibles et les poids forts sont remis à 0.

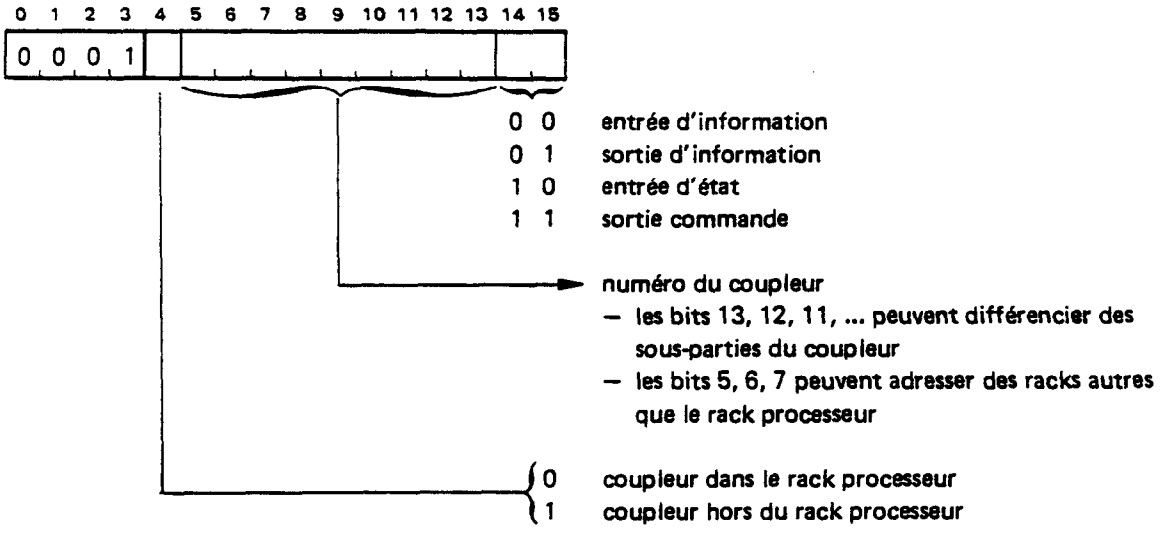
Dans le cas d'une sortie, l'information ou la commande émise vers le coupleur est issue du registre A. Si l'information est un octet, cet octet est pris en poids faibles de A et les poids forts peuvent être quelconques.

C'est dans l'opérande de l'instruction que sont codées : l'opération à effectuer et l'adresse du périphérique concerné.
On trouvera ci-après les opérandes correspondant aux différentes opérations d'entrée/sortie possibles.

Format court :



Format long :



Remarque :

La programmation d'un périphérique utilise plusieurs opérandes. On appelle adresse du coupleur (ou du périphérique) l'opérande correspondant à l'entrée d'information (de la voie de rang 0 dans le cas d'un coupleur multiplexé).

Interviennent : le mot mémoire adressé, A pour une sortie

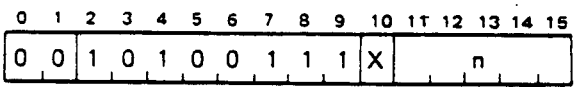
Est modifié : A pour une entrée

Indicateurs : non modifies

Alarmes possibles : – protection mémoire (05)
– mémoire inexistante (05)
– parité
- instruction privilégiée

shift logical left double (SLLD)

SLLD



Code : 29C_

Adressage : indexation possible

Opération effectuée : les 32 bits des registres A et 6; considérés comme un registre unique, sont décalés à gauche d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n +X modulo 32 si l'instruction est indexée.

Les p bits de droite sont remplacés par des zéros.

Les p bits de gauche sont perdus.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position p - 1.

Si p = 0 l'indicateur est remis à zéro.

Interviennent : . A
. B

Sont modifiés . A, B, ST

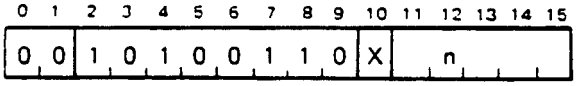
Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

shift logical left single (SLLS)

SLLS



Code : 298_

Adressage : indexation possible

Opération effectuée : les 16 bits du registre A sont décalés à gauche d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + X modulo 32 si l'instruction est indexée.

Les p bits de droite sont remplacés par des zéros.

Les p bits de gauche sont perdus.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position p - 1.

Si p = 0 ou p > 16 l'indicateur est mis à zéro.

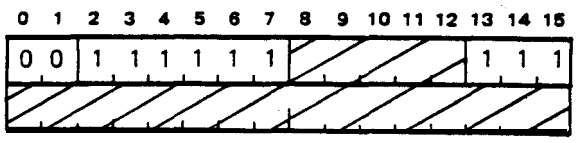
Intervient : A

Sont modifiés : . A, ST

Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Option : CDA16 (65) et (70)

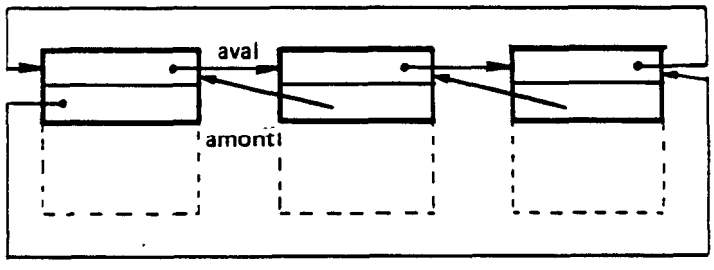
Code : premier mot 3F07
deuxième mot 0000

Opération effectuée : supprime le successeur de l'élément dont l'adresse est dans l'accumulateur et en donne l'adresse dans l'accumulateur.
L'élément supprimé n'est pas modifié.
Si l'élément supprimé est seul dans la liste, l'opération est ineffective.

La liste est une suite de doubles mots chaînés dans les deux sens :

. le premier mot contient l'adresse du double mot suivant (chaînage aval),
. le second mot contient l'adresse du double mot précédent (chaînage amont),

A chaque élément de la liste on peut associer un ou plusieurs mots qui constituent l'information propre à cet élément. Ces informations ne sont pas modifiées par l'instruction.



On peut ainsi gérer une liste en "dernier entré - premier sorti" (pile LIFO). L'élément pointé par A constituera une "tête de liste". Une liste vide se résumera à sa "tête". On insérera un nouvel élément à la suite de la tête par l'instruction INSQ. On sortira le dernier élément entré par l'instruction SLQ. Aucun masquage n'est alors nécessaire en multitraitement sur un monoprocesseur.

En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

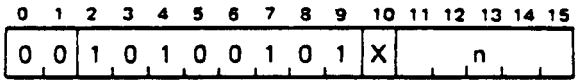
Interviennent : A, l'élément pointé par A et son successeur

Sont modifiés : A, l'élément pointé par A et le successeur de l'élément supprimé

Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante
- parité mémoire
- protection mémoire

shift logical right double (SLRD) — SLRD



Code : 294

Adressage : Indexation possible

Opération effectuée : les 32 bits des registres A et B, considérés comme un registre unique, sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + x modulo 32 si l'instruction est indexée.

Les p bits de gauche sont remplacés par des zéros.

Les p bits de droite sont perdus.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 32 - p.

Si p = 0 l'indicateur est mis à zéro.

Interviennent: . A
. B

Sont modifiés: . A, B, ST

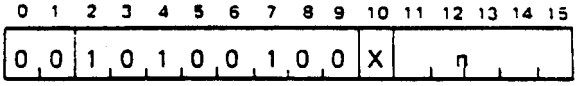
Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

---

shift logical right single (SLRS) — SLRS



Code : 290

Adressage : indexation possible

Opération effectuée : les 16 bits du registre A sont décalés à droite d'un nombre p de positions égal à n si l'instruction n'est pas indexée, à n + x modulo 32 si l'instruction est indexée.

Les p bits de gauche sont remplacés par des zéros.
Les p bits de droite sont perdus.

L'indicateur C prend la valeur du dernier bit sorti, bit qui occupait la position 16 - p. Si p = 0 ou p > 16 l'indicateur est mis à zéro.

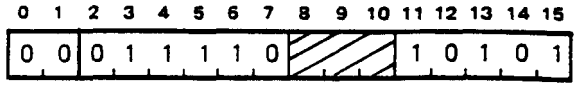
Intervient : A

Sont modifiés : . A, ST

Indicateurs :

V	C	
0	0	le dernier bit sorti vaut 0
0	1	le dernier bit sorti vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité



Mode : maître

Code : 1E15

Opération effectuée : force à un les bits du registre ST spécifiés dans l'accumulateur et charge la nouvelle valeur de ST dans l'accumulateur.

Interviennent : A, ST

Sont modifiés : A, ST

Indicateurs : modifiés selon le contenu de A

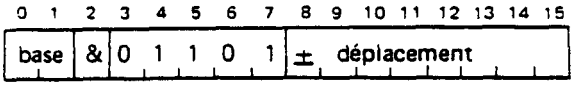
Alarmes possibles :
- mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

ATTENTION : la modification de ST est une opération extrêmement délicate ; toute erreur peut avoir des conséquences très importantes voire imprévisibles.

Dans la pratique on se limitera à la manipulation des bits LCM et IPM.
Le bit STOP peut être forcé mais sa prise en compte par le processeur n'est effective qu'après environ 2 millisecondes.
Le forçage de certains bits est ineffectif

store A (STA)

STA



Code :

	direct	indirect
base C	4D__	6D__
base L	8D__	AD__
base W	CD__	ED__

Adressage : mémoire (mot)

Opération effectuée : le contenu du registre A est rangé dans le mot mémoire adressé.

Intervient : A

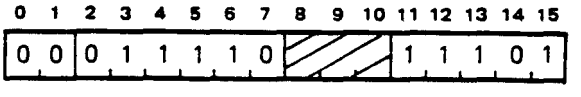
Est modifié : le mot mémoire adressé

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

store accumulator relative (STAR)

STAR



Mode : maître

Code : 1 E 1 D

Opération effectuée : le contenu de l'accumulateur est rangé dans le mot mémoire dont l'adresse relative à SLO est contenue par le registre Y.
La protection d'accès (SLE) est activée.
En l'absence de l'option DRPS, le registre Y contient une adresse absolue.

Intervient : A, SLO, SLE

Est modifié : le mot mémoire pointé par Y

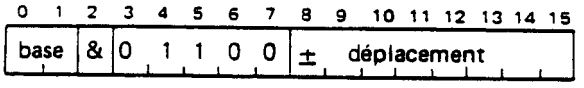
Indicateurs : non modifiés

Alarmes possibles : - mémoire inexistante (05)
- protection mémoire (05)
- parité mémoire
- instruction privilégiée

ATTENTION : le programme peut passer en mode esclave à la suite d'une alarme.

Store B (STB)

STB



Code

	direct	indirect
base C	4C__	6C__
base L	8C__	AC__
base W	CC__	EC__

Adressage : mémoire (mot)

Opération effectuée : le contenu du registre B est rangé dans le mot mémoire adressé.

Intervient : B

Est modifié : le mot mémoire adressé

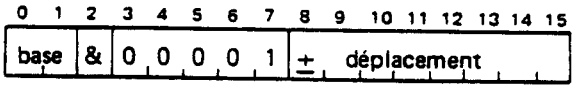
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

---

store byte (STBY)

STBY



Code :

	direct	indirect
base C	41__	61__
base L	81__	A1__
base W	C1__	E1__

Adressage : mémoire (octet)

Opération effectuée : les bits 8 à 15 du registre A sont rangés dans l'octet mémoire adressé.

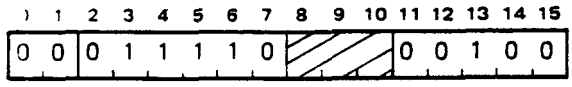
Intervient : A (bits 8 à 15)

Est modifié : l'octet mémoire adressé

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

ATTENTION : s'il n'y a pas indexation, l'octet adressé est l'octet gauche d'un mot.



Mode : maître

Code : 1 E 0 4

Opération effectuée : comme l'instruction ACQ. cette instruction provoque la remise à zéro du bit du registre HV correspondant à la tâche immédiate dans laquelle elle est rencontrée (s'il s'agit d'une tâche immédiate provoque un changement de contexte, et provoque le lancement d'une tâche immédiate ou différée suivant la hiérarchie habituelle.

Mais une seule instruction de cette tâche est exécutée. Après l'exécution de cette instruction, l'alarme de numéro 7 est déclenchée.

Les masques IOM et IPM sont mis à 0.

Cette instruction ne doit pas être utilisée dans une tâche différée.

On peut ainsi exécuter des programmes, sous le contrôle d'un programme de mise au point (AID par exemple), en "pas à pas" ou en "exécution contrôlée".

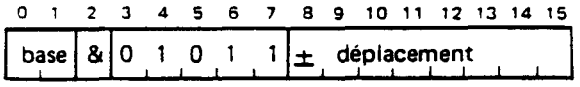
Sont modifiés . les registres du contexte partiel (C, K, P, S), ST, HV

Indicateurs : ceux du nouveau contexte, IOM, IPM

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction privilégiée

store X (STX)

STX



Code :

	direct	indirect
base C	48__	68__
base L	88__	A8__
base W	C8__	E8__

Adressage : mémoire (mot)

Opération effectuée : le contenu du registre X est rangé dans le mot mémoire adressé.

Intervient : x

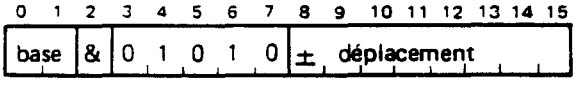
Est modifié : le mot mémoire adressé

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité

store Y (STY)

STY



Code :

	direct	indirect
base C	4A__	6A__
base L	8A__	AA__
base W	CA__	EA__

Adressage : mémoire (mot)

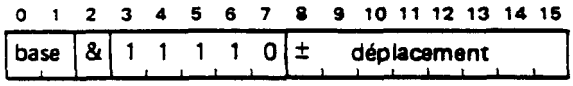
Opération effectuée : le contenu du registre Y est rangé dans le mot mémoire adressé.

Intervient : Y

Est modifié : le mot mémoire adresse

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Code :

	direct	indirect
base C	5E__	7E__
base L	9E__	BE__
base W	DE__	FE__

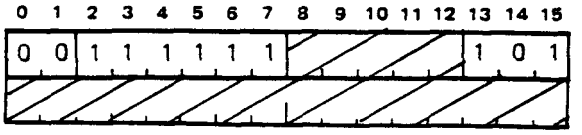
Adressage : mémoire (mot)

Opération effectuée : le mot mémoire adressé est remis à zéro.

Est modifié : le mot mémoire adressé

Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (05)
- parité



Option CDA16 (65) et (70)

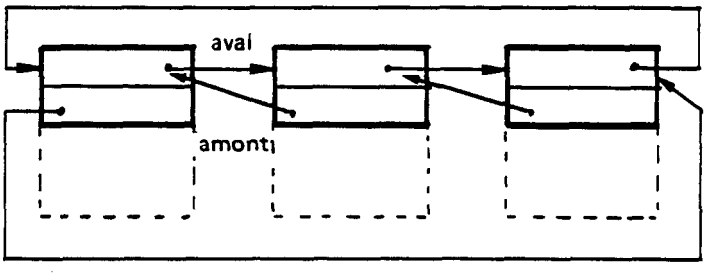
Code premier mot 3F05
deuxième mot 0000

Opération effectuée : supprime dans une liste l'élément dont l'adresse est donnée dans l'accumulateur. L'élément supprimé n'est pas modifié.
Si l'élément supprimé est seul dans la liste l'opération est ineffective.

La liste est une suite de doublesmots chaînés dans les deux sens :

- le premier mot contient l'adresse du double mot suivant (chaînage aval).
- le second mot contient l'adresse du double mot précédent (chaînage amont),

A chaque élément de la liste on peut associer un ou plusieurs mots qui constituent l'information propre à cet élément. Ces informations ne sont pas modifiées par l'instruction.



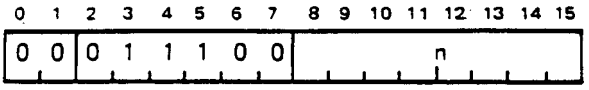
En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

Intervient : A, l'élément de la liste pointé par l'accumulateur, le précédent et le suivant.

Sont modifiés : les éléments précédents et suivants

Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante
- protection mémoire
- parité mémoire



code : 1C__

Opération effectuée : L'octet poids fort du registre X est remis à zéro et l'octet poids faible du registre est chargé par les bits 8 à 15 de l'instruction.

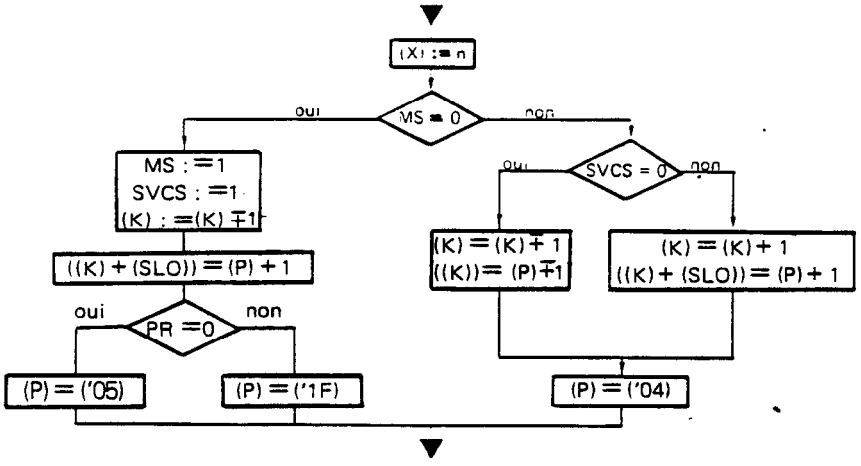
Quand l'instruction est exécutée en mode maître :
- le contenu de la mémoire d'adresse absolue 4 est chargé dans le registre P,
- MS et SVCS ne sont pas modifiés,
- le contenu du registre K est augmenté de 1, puis l'adresse de l'instruction qui suit le SVC est rangée dans le mot mémoire pointé par K.

Quand l'instruction est exécutée en mode esclave :
- le contenu de la mémoire d'adresse absolue 5 est chargé dans le registre P,
- les bits MS et SVCS sont mis à 1,
- le contenu du registre K est augmenté de 1, puis l'adresse de l'instruction qui suit le SVC est rangée dans le mot mémoire pointé par K.

Quand l'instruction est exécutée en mode privilégié :
- le contenu de la mémoire d'adresse '1F est chargé dans le registre P
- les bits MS et SVCS sont mis à 1
- le contenu du registre K est augmente de 1, puis l'adresse de l'instruction qui suit le SVCS est rangé dans le mot mémoire pointé par K.

Si l'option DRPS est présente, le fonctionnement est identique mais K contient soit une adresse absolue, soit une adresse relative à SLO selon les indicateurs MS et SVCS :
- si l'instruction est exécutée en mode maître (MS = 1 et SVCS = 0), K contient une adresse absolue,
- si l'instruction est exécutée en mode esclave ou privilégié (MS = 0 et SVCS = 0) ou en mode maître mais dans une requête superviseur appelée en mode esclave (MS = 1 et SVCS = 1), K contient une adresse relative à SLO.

De plus, si l'instruction est exécutée en mode esclave ou privilégié, l'adresse de retour qui est empilée est également une adresse relative à SLO.



On peut ainsi appeler des sous-programmes devant être exécutés en mode maître, sans que l'appelant ait à connaître l'adresse de ce sous-programme (requête superviseur) :

- l'adresse de retour est rangée en mémoire puis reprise à la fin du sous-programme par une instruction RSV, la fonction de RSV étant complémentaire de celles de SVC. (L'évolution de K pour la sauvegarde de l'adresse de retour est homogène avec l'évolution de K pour les sauvegardes effectuées par BSR et PSR).
- les bits 8 à 15 de l'instruction, chargés dans le registre X, indiquent le numéro du sous-programme à appeler, et la mémoire débanalisée d'adresse 5 doit pointer sur une séquence qui, en fonction de cette valeur de X, assure l'aiguillage vers le sous-programme appelé.

- l'appel d'un tel sous-programme par un SVC étant lui-même exécute en mode maître est possible : la mémoire d'adresse 4 doit pointer sur une séquence assurant d'une part l'appel du sous-programme par un BSR, d'autre part le retour derrière le SVC d'appel par un RSR.
- l'adressage de K relativement à SLO dans le cas de l'option DRPS permet de communiquer une zone mémoire propre à l'appelant, qui sert à la réentrance des requêtes superviseur.

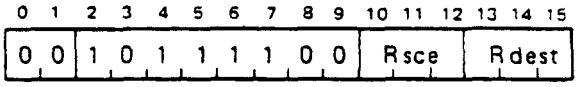
Sont modifiés : X, K, P, ST
la mémoire pointée par K

Indicateurs : MS et SVCS

Alarmes possibles : – protection mémoire 05
- mémoire inexistante 06
- parité

ATTENTION : – K pointe toujours sur la mémoire occupée de la pile d'adresse la plus grande : ici le dernier point de retour rangé.
– Le retour correspondant à un SVC exécuté en mode maître se fait par un RSR et non par un RSV.
– Le programme peut passer en mode esclave à la suite d'une alarme.

swap bytes register (SWBR)



Code : 2F0_

Adressage : registre - registre

Opération effectuée : les bits 8 à 15 du registre source sont chargés dans les bits 0 à 7 du registre destination, les bits 0 à 7 du registre source sont chargés dans les bits 8 à 15 du registre destination (le registre source n'est pas modifié).

On peut ainsi :

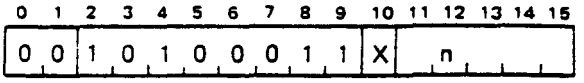
- échanger l'octet gauche et l'octet droit d'un des registres A B X Y C L W K, en précisant le même registre comme source et destination.
- charger un des registres A B X Y C, L W K par les deux octets intervertis d'un autre de ces registres.

Intervient : un des registres A B X Y C L W K (le registre source)

Est modifié : un des registres A B X Y C L W K (le registre destination)

Indicateurs : non modifiés

Alarmes possibles :- protection mémoire 05
- mémoire inexistante 05
- parité



Code : 28C

Adressage : Indexation possible

Opération effectuée : les registres A et B étant considérés comme un registre unique de 32 bits, la valeur du bit dont le rang est n si l'instruction n'est pas indexée, n + x modulo 32 si l'instruction est indexée, est chargée dans l'indicateur C.

On peut ainsi tester grâce aux instructions JC et JNC, un bit de A ou de B sans détruire le registre.

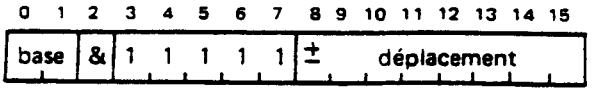
Intervient : un bit de A ou un bit de B

Est modifié : ST

Indicateurs :

: V	C	
0	0	le bit vaut 0
0	1	le bit vaut 1

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité



Option : scheduler

Mode : maître

Code :

	direct	indirect
base C	5F__	7F__
base L	9F__	BF__
base W	DF__	FF__

Adressage : mémoire (1er mot du sémaphore)

Opération effectuée : que le sémaphore adressé par l'instruction soit un sémaphore de synchronisation ou un sémaphore d'appel, l'opération effectuée est identique :

le compteur du sémaphore est diminué de un ;

le n° de la tâche qui effectue l'instruction WAIT est rangé dans les bits 1 à 7 du premier mot du sémaphore.

- si la nouvelle valeur du compteur est nulle ou positive la tâche qui effectue l'instruction WAIT se poursuit normalement,
- si la nouvelle valeur du compteur est négative :

  le bit de la file ESTF dont le rang correspond au n° de la tâche qui effectue l'instruction WAIT est mis à 0 (la tâche est masquée) ;

  le contexte de cette tâche est sauvegardé et une tâche moins prioritaire est lancée par le scheduler.

Les indicateurs IOM, IPM sont remis à 0 (avant passage éventuel dans le scheduler).

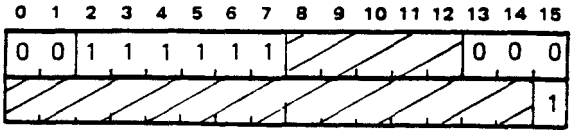
Interviennent : le sémaphore adressé
NS

Sont modifiés : le sémaphore adressé
NS et tous les registres s'il y a changement de contexte
ST

Indicateurs : IOM, IPM, plus le changement de contexte éventuel

Alarmes possibles : - protection mémoire 05
- mémoire inexistante 03
- parité
- instruction privilégiée
- instruction optionnelle inexistante
exécution interdite sous tâche hardware

ATTENTION : WAIT ne prend pas en compte et ne modifie pas la file paramètre. C'est la tâche où se trouve le WAIT qui doit remettre les bits de cette file à zéro. Pour ce faire, la tâche doit se protéger en marquant les interruptions par DIT ou en utilisant un sémaphore d'exclusion mutuelle.



Option : **CDA16** (65), (70) et (35)

Code : premier mot 3F00
deuxième mot 0001

Opération effectuée : transfert d'un segment de mémoire relatif à la tâche en cours vers la zone de données communes (CDA),

- le registre B contient l'adresse du 1er mot de la zone source. Cette adresse est absolue ou relative à SLO selon que le programme est en mode maître ou esclave.
- le registre A contient l'adresse, relative au début de la zone CDA, du 1er mot destination.
- le registre X contient le nombre de mots à transférer.

Les adresses début et fin de la zone CDA sont contenues dans les mots d'adresse '18 et '19. Ces mots ont le même format que SLO : ils contiennent les 16 bits poids forts d'une adresse sur 20 bits, sachant que la zone CDA ne peut s'implanter qu'à des adresses multiples de 16, et ne peut chevaucher une frontière de 64 Kmots.

Le transfert s'effectue mot par mot par adresses décroissantes et les zones peuvent se recouvrir.
A chaque transfert, le registre X est diminué de 1 et les interruptions (ainsi que le défaut secteur) sont prises en compte.
A la fin du transfert, le registre X est nul.
L'opération est ineffective si le contenu du registre X est négatif après décrémentation.

On peut ainsi écrire au maximum 32768 mots d'une zone de données, de 64 Kmots au maximum, commune à toutes les tâches d'un système.

La protection d'accès à la zone CDA est effectuée globalement avant tout transfert et le cas échéant, provoque une alarme protection mémoire.
En l'absence de l'option DRPS, les mots d'adresse '18 et '19 continuent à définir la zone CDA mais leurs quatre bits de poids forts doivent être nuls.

En l'absence de l'option scheduler, cette instruction est interprétée comme une instruction SVC 7 (cf codes extension).

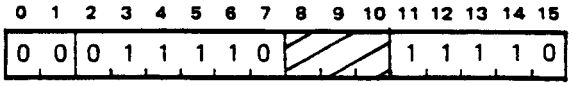
Interviennent : A, B, X, SLO, SLE, les mots mémoire d'adresse '18 et '19, les zones pointées par A et B.

Sont modifiés : X, la zone pointée par A.

Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante
- protection mémoire
- parité mémoire

Interruptibilité : l'instruction est interruptible à chaque transfert de mot y compris par le défaut secteur.



Mode : maître

Code : 1 E 1 E

Opération effectuée : transfert du contenu des registres A et B respectivement dans les registres SLO et SLE.
En l'absence de l'option DRPS, cette instruction est ineffective.

Interviennent : A, B

Sont modifiés : SLO et SLE

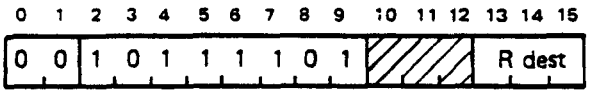
Indicateurs : non modifiés

Alarmes possibles :
- mémoire inexistante (05)
- protection mémoire (06)
- parité mémoire
- instruction privilégiée

ATTENTION : la différence (SLE) - (SLO) doit toujours correspondre à une taille mémoire inférieure à 64 K. Pour cette raison, vu le cadrage des registres SLO, SLE, les quatre bits poids forts de B sont ignorés et la relecture de SLE (RDOE) peut conduire à des valeurs modifiées si la condition précédente n'est pas satisfaite.

exchange interrupt mask with register (XIMR)

XIMR



Mode : maître

Code : 2 F 4 _

Adressage : registre registre

Opération effectuée : les contenus des registres IM et destination de l'instruction sont échangés. Les éventuelles interruptions préalablement en attente et démasquées par l'instruction sont immédiatement prises en compte si IOM est à 0.

Interviennent :
- un des registres A B X Y C L W K
- IM

Sont modifiés :
- un des registres A B X Y C L W K
- IM

indicateurs : non modifies

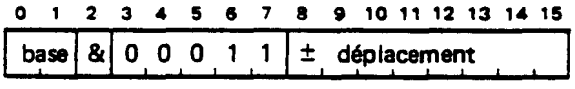
Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité
- instruction privilégiée

ATTENTION : une tâche immédiate interrompue par une tâche immédiate plus prioritaire est reprise même si, pendant qu'elle est suspendue, le niveau correspondant est masqué par cette tâche immédiate plus prioritaire.

---

exchange memory with A (XM)

XM



Code :

	direct	indirect
base C	43 __	63 __
base L	83 __	A3 __
base W	C3 __	E3 __

Adressage : mémoire (mot)

Opération effectuée : les contenus du registre A et du mot adressé sont échangés.

Interviennent : A, mot adressé

Sont modifiés : A, mot adressé

Indicateurs : non modifiés

Alarmes possibles :
- protection mémoire (05)
- mémoire inexistante (05)
- parité

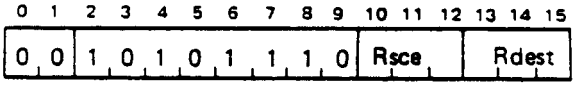
Remarque : cette instruction permet d'écrire des séquences interverrouillées dans des configurations à plusieurs processeurs de traitement (primitive du type "Test and Set Lock") :

```
LAI    -1
XM     VERROU
JANE   $ - '
```

En se basant sur un verrou nul quand il est libre, on peut ainsi s'assurer de l'exclusivité d'une ressource.

exchange registers (XR)

XR



Code : 2B8_

Adressage : registre - registre

Opération effectuée : échange du contenu des deux registres précisés par l'instruction.
On peut ainsi : effectuer un échange entre deux quelconques des huit registres A B X Y C L W K.

Interviennent : deux des registres A B X Y C L W K

sont modifiés : deux des registres A B X Y C L W K

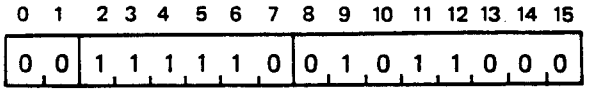
Indicateurs : non modifiés

Alarmes possibles : - protection mémoire (05)
- mémoire inexistante (06)
- parité

---

exchange memory relative (XMR)

XMR



Code : 3E58

Opération effectuée : le contenu du registre A est échangé par un cycle mémoire indivisible, avec le mot mémoire adressé par Y + SLO.

Interviennent : A, mot mémoire adresse

Alarmes possibles : - protection mémoire
- mémoire inexistante
- parité

Nota importante : cette instruction n'existe que sur les modèles 16-65P ou 16-75P.

4.2 - INSTRUCTIONS SPECIALES ISP16

En vue d'améliorer les performances, un nouveau jeu d'instructions a été créé sur les processeurs 16-35 et 16-70. Elles se classent en trois groupes :

- Les instructions basées :

  Elles sont exécutables en mode maître et privilégié et permettent d'atteindre les zones mémoire au-delà de 64K sans utiliser le registre SLO courant. Ceci est obtenu en donnant en paramètre de l'instruction une base en format SLO qui pointe sur la page de 64K mots à atteindre.

- Les instructions sur CDA :

  Elles utilisent le système standard de la CDA (zone de mémoire commune) définie par ses limites contenues dans les mémoires OCDA ('0018) et ECDA ('0019) en format SLO. Ces instructions permettent d'atteindre un octet, un mot ou un double mot avec une meilleure performance que les instructions à mots multiples RCDA et WCDA. On rappelle que la CDA est limitée à 64K mots et ne doit pas chevaucher une frontière physique de 64K.

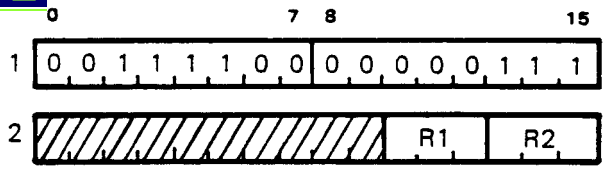
- Les instructions diverses :

  XCTX permet sur 16-70 de commuter les contextes appelant du superviseur.
  XENT et XSOR facilitent les entrées et sorties de procédures langages.

based discover bit in memory (BDBTM)

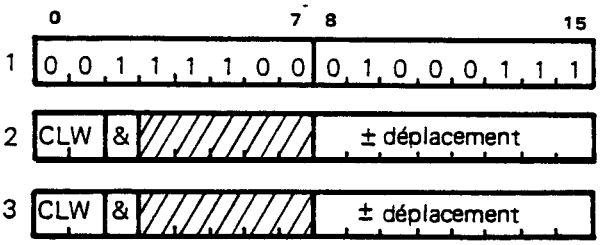
BDBTM

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits du début de la file de bits est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + paramètre 2
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
On peut ainsi atteindre une page de 64K mots à partir de l'adresse donnée.
Le rang du bit est relatif au bit 0 du premier mot de la file.

Opération : recherche à partir du rang (X) jusqu'au rang (Y) - 1 du premier bit à 1 dans une file de bits en mémoire.
L'instruction est interruptible entre chaque mot.
Recherche positive :
V = 0, C = 0, le bit trouvé est au rang (X)
Recherche négative :
V = 0, C = 1, le registre X est chargé par Y
Si (X) ⩾ (Y) la recherche sera négative.

Interviennent : X, Y et la file de bits.

Sont modifiés : X et ST.

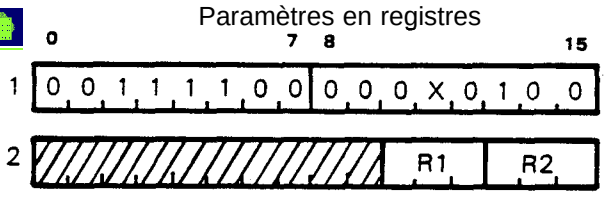
Indicateurs
V = 0 et C = 0, bit à 1 rencontré
V = 0 et C = 1, bit à 1 non rencontré

Alarmes possibles :
mémoire inexistante
protection mémoire
parité
instruction privilégiée

BDLD

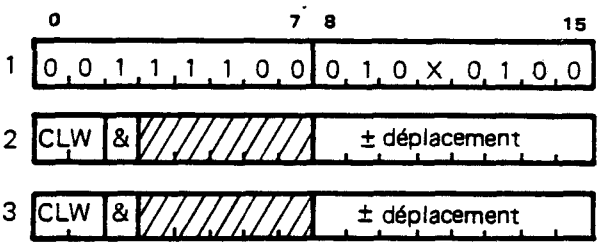
based double load (BDLD)

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Las paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + (paramètre 2 [+ x])
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (paramètre 2 + x) est effective sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : le contenu du mot adressé est chargé dans le registre A, le contenu du mot suivant est chargé dans le registre 8.

Interviennent : les deux mots mémoire adressés.

Sont modifiés : A et B.

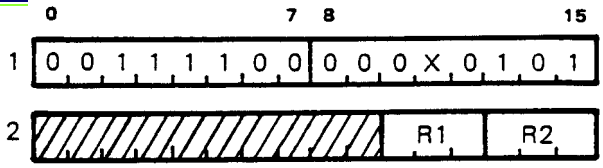
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée

based double store (BDST)

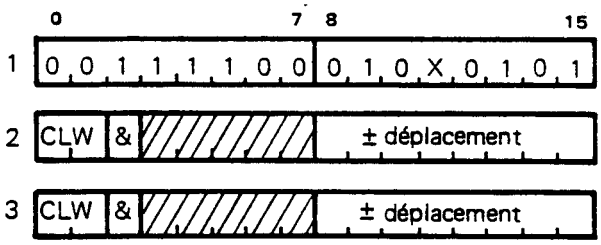
BDST

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + (paramètre 2 [+ X])
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 18 bits
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (paramètre 2 = X) est effective sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : le contenu du registre A est rangé dans le mot adressé, le contenu du registre B est rangé dans le mot suivant.

Interviennent : A et B.

Sont modifiées : les deux mots mémoire adressés.

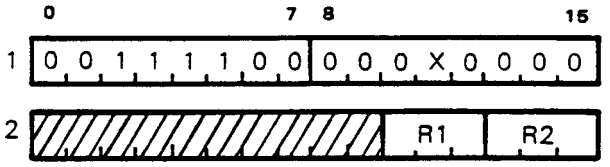
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée

BLA

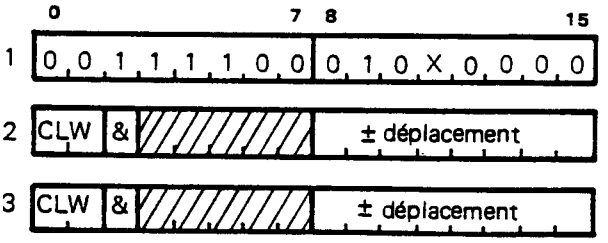
based load A (BLA)

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + (paramètre 2 [+X])
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (paramètre 2 + X) est effectuée sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : le contenu du mot mémoire adressé est chargé dans le registre A.

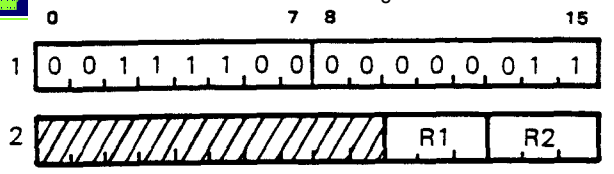
Intervient : le mot mémoire adressé.

Est modifiée : A

Indicateurs : non modifiés.

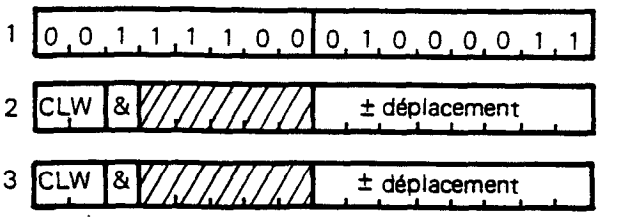
Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP18 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits du mot mémoire qui contient l'opérande octet est calculée de la manière suivante :

AE20 = paramètre 1 * 16 + (paramètre 2 + $X/_2$)

avec paramètre 1 = base en format SLO
paramètre 2= déplacement sur 16 bits
X = registre index.

L'octet concerné est l'octet gauche si X est pair, ou l'octet droit si X est impair.

L'opération (paramètre 2 + $X/_2$) est effectuée sur 16 bits.

On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : les bits 8 à 15 du registre A sont chargés par l'octet mémoire adresse, les bits 0 à 7 du registre A sont mis à zéro.

Intervient : l'octet mémoire adresse.

Est modifié : A

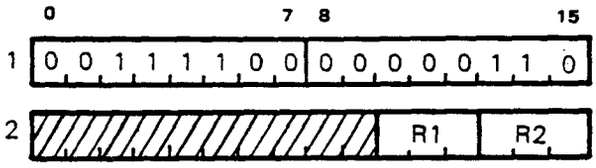
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

BMOVE

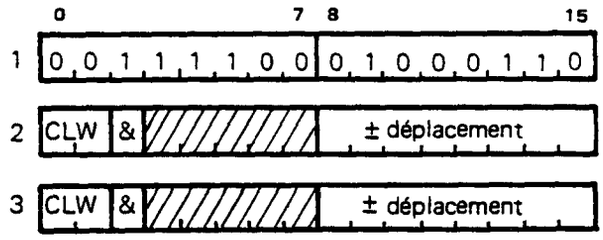
based move (BMOVE)

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Opération : transfert de X mots dans le sens des adresses décroissantes pour X de 1 à 32768.
L'opération est ineffective si le contenu du registre X est négatif après décrémentation.
L'instruction est interruptible entre chaque mot.
A la fin du transfert le contenu de X est nul.

Adressage : zone origine : paramètre 1 + 16 + A
zone destination : paramètre 2 * 16 + B
avec paramètre 1 et paramètre 2 = bases en format SLO.
Les opérations (A + X - 1) et (B + X - 1) sont exécutées sur 16 bits.
On peut ainsi atteindre deux pages de 64K mots à partir des bases données.

Interviennent : A, B, X et la zone mémoire pointée par A.

Sont modifiés : X et la zone mémoire pointée par B.

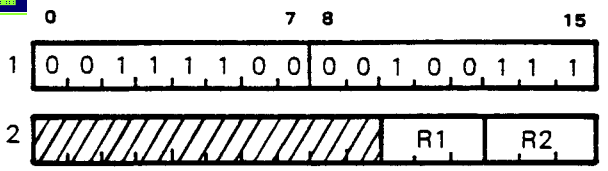
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

based reset bit in memory (BRBTM)

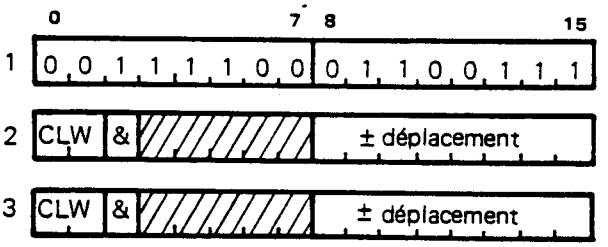
BRBTM

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits du début de la file de bits est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + paramètre 2
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de l'adresse donnée.
Le rang du bit est relatif au bit 0 du premier mot de la file.

Opération : Mise à zéro du bit de rang (X) dans une file de bits en mémoire.

Interviennent : X et la file de bits.

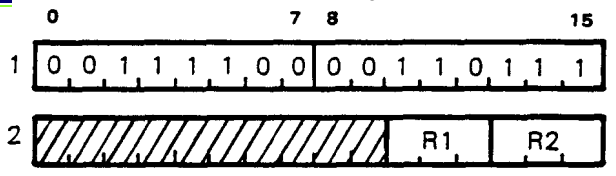
Est modifie : le bit spécifié dans la file.

Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

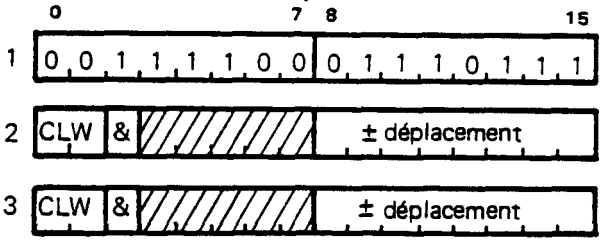
based set bit in memory (BSBTM) B S B T M

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramétras en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits du début de la file de bits est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + paramètre 2
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
On peut ainsi atteindre une page de 64K mots à partir de l'adresse donnée.
Le rang du bit est relatif au bit 0 du premier mot de la file.

Opération : mise à un du bit de rang (X) dans une file de bits en mémoire.

interviennent : X et la file de bits

Est modifié : le bit spécifié dans la file

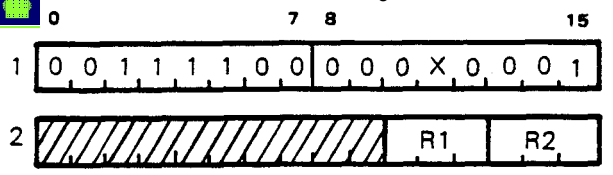
Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

based store A (BSTA)

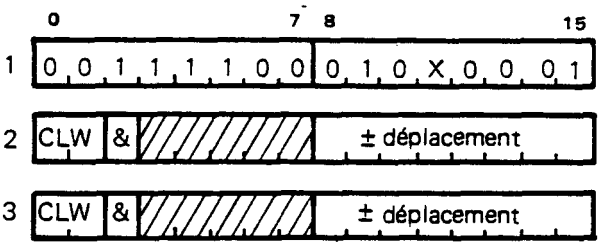
BSTA

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + (paramètre 2 [+ X])
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (paramètre 2 + X) est effectuée sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : le contenu du registre A est rangé dans le mot mémoire adressé.

Intervient : A

Est modifié : le mot mémoire adressé.

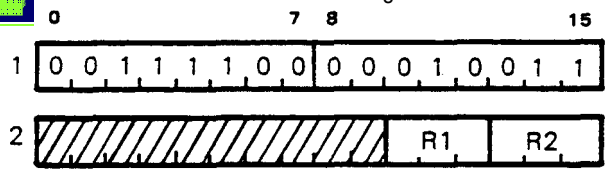
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

BSTBY

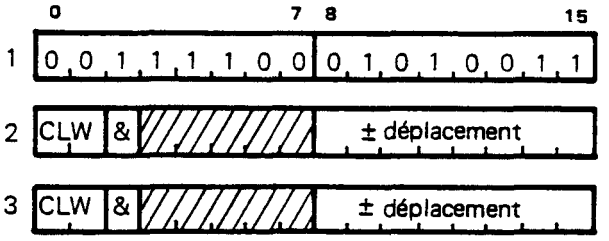
based store byte (BSTBY)

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits du mot mémoire qui contient l'opérande octet est calculée de la manière suivante :

AE20 = paramètre 1 * 16 + (paramètre 2 + $X/_2$)

avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
X = registre index.

L'octet concerné est l'octet gauche si X est pair, ou l'octet droit si X est impair.

L'opération (paramètre 2 + $X/_2$) **est effectuée sur 16 bits.**

On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : les bits 8 à 15 du registre A sont rangés dans l'octet mémoire adresse.

Intervient : A (bit 8 à 15).

Est modifie : l'octet mémoire adresse

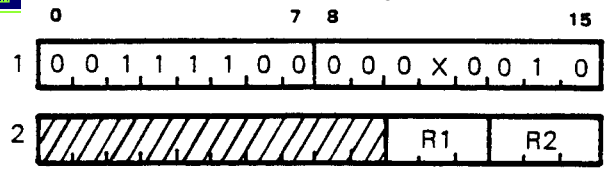
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

BXM

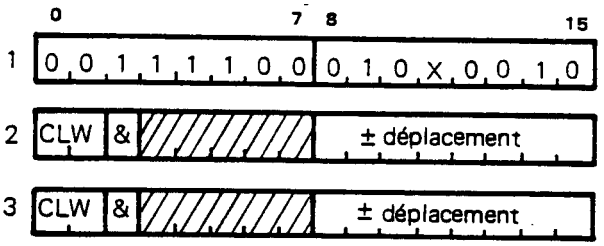
based exchange memory with A (BXM)

Paramètres en registres



Les paramètres 1 et 2 sont dans les registres R1 et R2 désignés par le mot 2 de l'instruction.

Paramètres en mémoires



Les paramètres 1 et 2 sont dans les mots mémoire pointés respectivement par les mots 2 et 3 de l'instruction.

Option : ISP16 Mode : Maître ou Privilégié

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = paramètre 1 * 16 + (paramètre 2 [+ X])
avec paramètre 1 = base en format SLO
paramètre 2 = déplacement sur 16 bits
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (paramètre 2 + X) est effectuée sur 16 bits.
On peut ainsi atteindre une page de 64K mots à partir de la base donnée.

Opération : les contenus du registre A et du mot adressé sont échangés en un cycle mémoire ininterruptible.

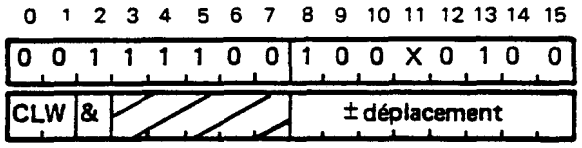
Interviennent : A, mot adressé

Sont modifiés : A, mot adressé

Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante
protection mémoire
parité
instruction privilégiée.

CDA double load (CDLD)



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16 [+ X])
avec OCDA = origine de la CDA en format SL0
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (D16 + X) est effectuée sur 16 bits.

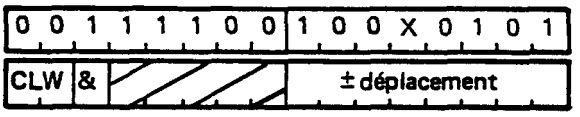
Opération : le contenu du mot adressé est chargé dans le registre A, le contenu du mot suivant est chargé dans le registre 8.

Interviennent : OCDA, ECDA et les deux mots mémoire adressés.

Sont modifiés : A et B.

Indicateurs : non modifiés.

Alarme possible : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16 [+ X])
avec OCDA = origine de la CDA en format SLO
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (D16 + X) est effectuée sur 16 bits.

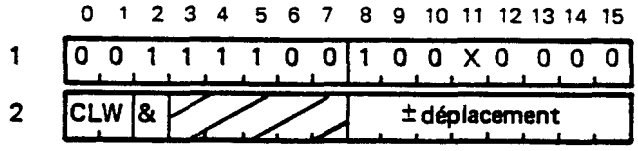
Opération : le contenu du registre A est rangé dans le mot adressé, le contenu du registre B est rangé dans le mot suivant.

Interviennent : OCDA, ECDA, A et B

Sont modifiés : les deux mots mémoires adressés.

indicateurs : non modifiés.

Alarme possible : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16 [+X])
avec OCDA = origine de la CDA en format SLO
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (D16 + X) est effectuée sur 16 bits.

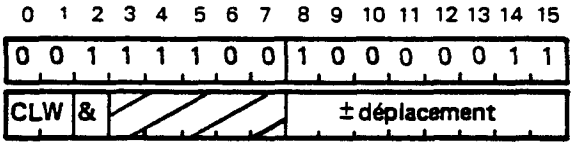
Opération : le contenu du mot mémoire adressé est chargé dans le registre A.

Interviennent : OCDA, ECDA et le mot mémoire adressé.

Est modifié : A

Indicateurs : non modifiés.

Alarme possible : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :

AE20 = OCDA + (D16 [+ $X/_2$])

avec OCDA = origine de la CDA en format SLO

D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction

X = registre index

L'octet concerné est l'octet gauche si X est pair, ou l'octet droit si X est impair.

L'opération (D16 + $X/_2$) est effective sur 16 bits.

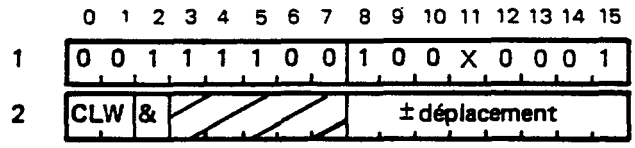
Opération : les bits 8 à 15 du registre A sont chargés par l'octet mémoire adressé, les bits 0 à 7 du registre A sont mis à zéro.

Interviennent : OCDA, ECDA et l'octet mémoire adressé.

Est modifié : A

Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16 [+ X])
avec OCDA = origine de la CDA en format SLO
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (D16 + X) est effectuée sur 16 bits.

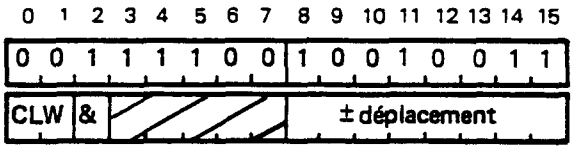
Opération : le contenu du registre A est rangé dans le mot mémoire adressé.

Interviennent : OCDA, ECDA et A

Est modifié : le mot mémoire adressé

Indicateurs : non modifiés

Alarme possible : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16.[+ $X/_2$])
avec OCDA = origine de la CDA en format SLO
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index
L'octet concerné est l'octet gauche si X est pair, ou l'octet droit si X est impair.
L'opération (D16 + $X/_2$) est effectuée sur 16 bits.

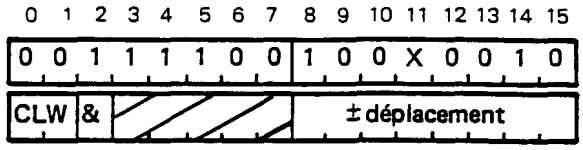
Opération : les bits 8 à 15 du registre A sont rangés dans l'octet mémoire adressé.

Interviennent : OCDA, ECDA et les bits 8 à 15 de A.

Est modifié : l'octet mémoire adressé.

Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16

Adressage : l'adresse effective sur 20 bits de l'opérande est calculée de la manière suivante :
AE20 = OCDA + (D16[+X])
avec OCDA = origine de la CDA en format SLO
D16 = déplacement sur 16 bits contenu dans le mot mémoire pointé par le mot 2 de l'instruction
X = registre index si l'instruction est indexée.
L'instruction est indexée si le bit 11 du premier mot du code est à 1.
L'opération (D16 + X) est effectuée sur 16 bits.

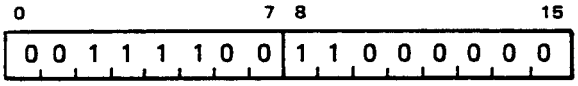
Opération : les contenus du registre A et du mot adressé sont échangés en un cycle mémoire ininterruptible.

Interviennent : OCDA, ECDA, A et le mot adressé.

Sont modifiés : A et le mot adressé.

Indicateurs : non modifiés.

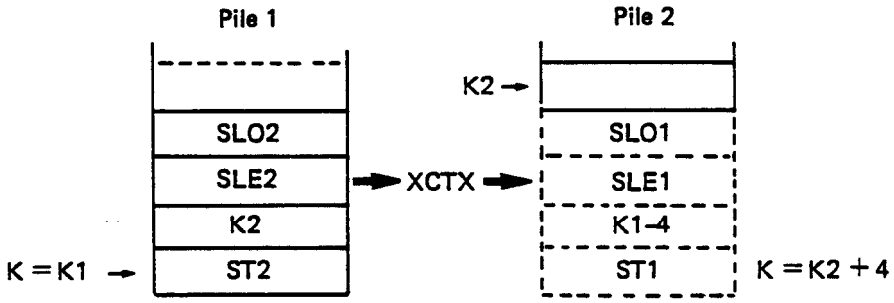
Alarme possible : mémoire inexistante
protection mémoire relative à SLO/SLE ou OCDA/ECDA
parité mémoire.



Option : ISP16 sur 16-70 seulement

Mode : maître seulement

Opération : dépilement de 4 mots de la pile K courante : c'est le nouveau contexte.
Echange de ces 4 mots avec SLO, SLE, K et les bits 1 et 2 de ST : c'est la commutation.
Empilement de ces 4 mots dans la nouvelle pile K : c'est l'ancien contexte.

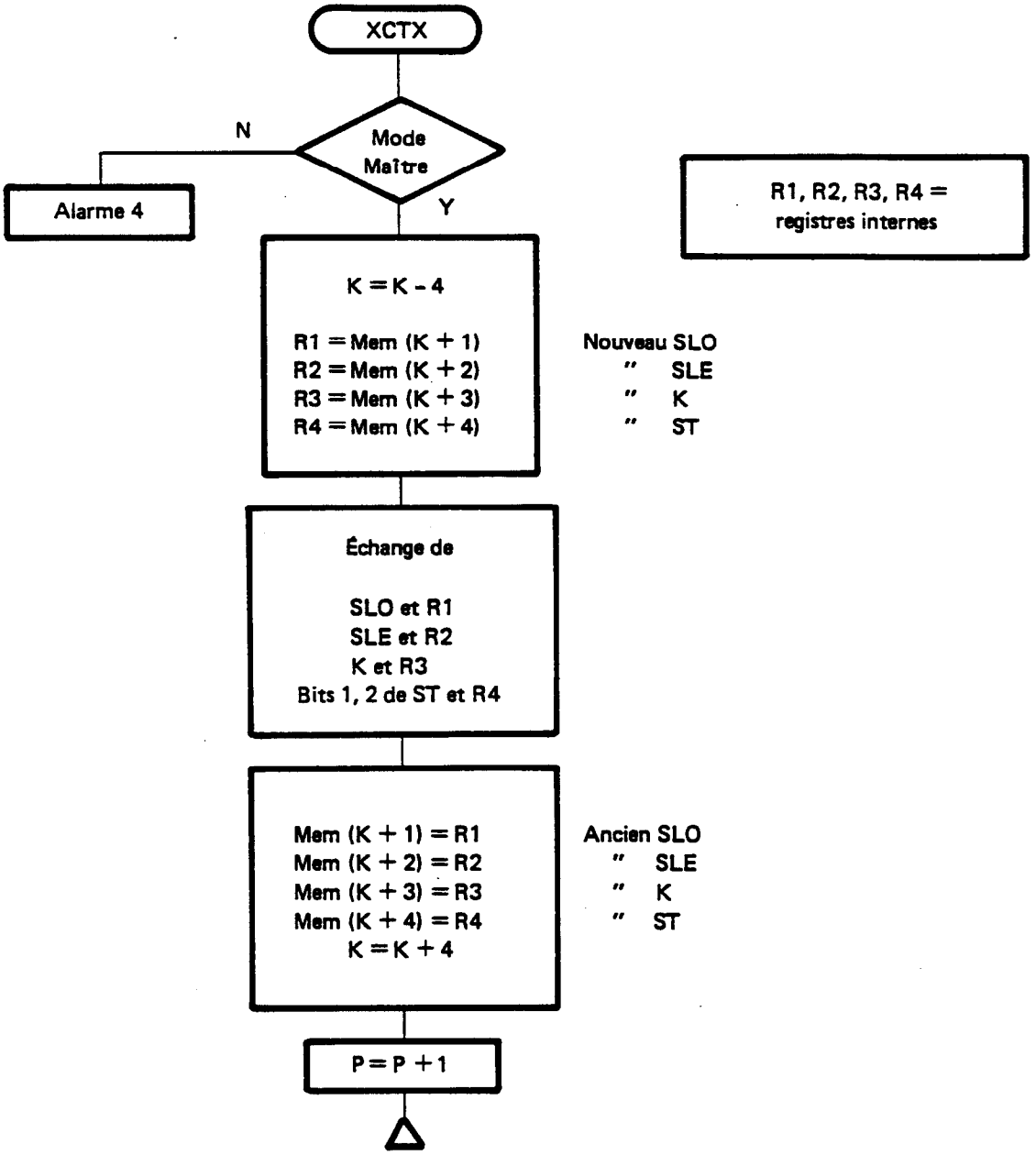


Interviennent : SLO, SLE, K, les bits 1 et 2 de ST et les piles.

Sont modifiés : SLO, SLE. K, les bits 1 et 2 de ST et les piles.

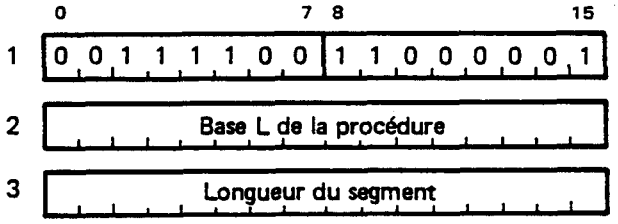
Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité mémoire
instruction privilégiée.


XCTX
Mode Maître
N
Y
Alarme 4
R1, R2, R3, R4 = registres internes
K = K - 4
R1 = Mem (K + 1)
R2 = Mem (K + 2)
R3 = Mem (K + 3)
R4 = Mem (K + 4)
Nouveau SLO
" SLE
" K
" ST
Échange de
SLO et R1
SLE et R2
K et R3
Bits 1, 2 de ST et R4
Mem (K + 1) = R1
Mem (K + 2) = R2
Mem (K + 3) = R3
Mem (K + 4) = R4
K = K + 4
Ancien SLO
" SLE
" K
" ST
P = P + 1

entrée de procédure (XENT)

XENT



Option : ISP16

Fonction : les instructions XENT et XSOR sont réservés à l'usage des compilateurs de langages pour gérer les entrées et sorties de procédures.
Pour cela elles utilisent une structure débanalisée de la zone COMMON pointée par la base c :

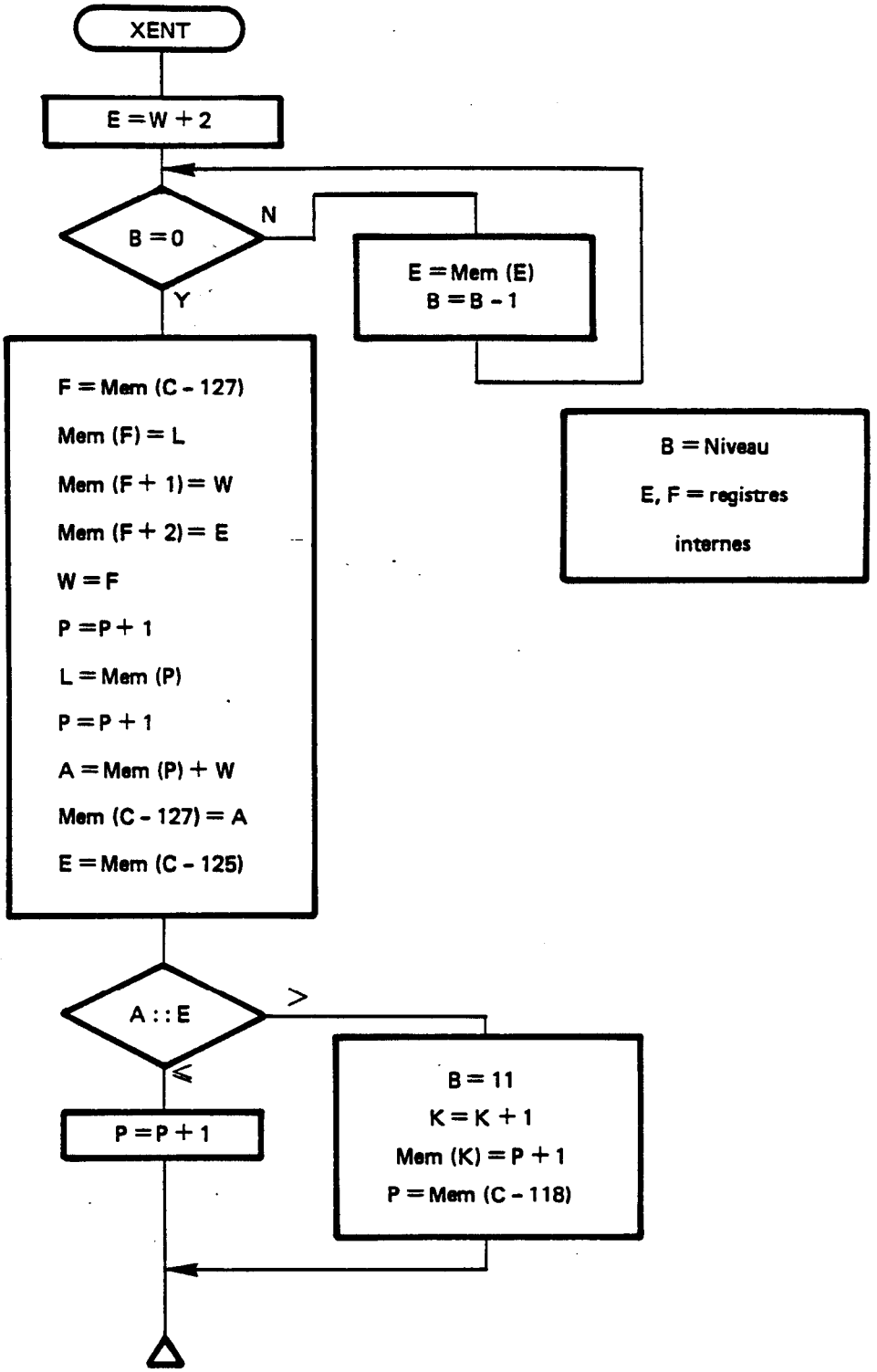
C-127 = Lien statique
C-125 = Lien dynamique
C-118 = Sous-programme d'erreur

Opération : entrée de procédure - voir l'organigramme.

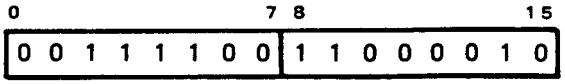
Sont modifiés : A, B, L, W, le lien statique, la pile et éventuellement K.

Indicateurs : non modifiés.

Alarmes possibles : mémoire inexistante
protection mémoire
parité mémoire.


XENT
E = W + 2
B = 0
N
Y
E = Mem (E)
B = B - 1
F = Mem (C - 127)
Mem (F) = L
Mem (F + 1) = W
Mem (F + 2) = E
W = F
P = P + 1
L = Mem (P)
P = P + 1
A = Mem (P) + W
Mem (C - 127) = A
E = Mem (C - 125)
B = Niveau
E, F = registres
internes
A : : E
>
≤
P = P + 1
B = 11
K = K + 1
Mem (K) = P + 1
P = Mem (C - 118)

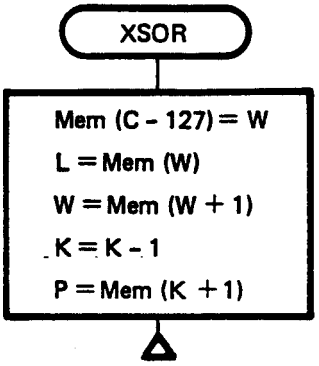
sortie de procédure (XSOR)



Option : ISP16

Fonction : les instructions XENT et XSOR sont réservées à l'usage des compilateurs de langages pour gérer les entrées et sorties de procédures.

Opération : sortie de procédure



Sont modifiés : L, W, K, le lien statique

Indicateurs : non modifiés

Alarmes possibles : mémoire inexistante
protection mémoire
parité mémoire.

5 - TACHES IMMÉDIATES (interruptions et alarmes)

5 - TACHES IMMÉDIATES

On appelle tâche immédiate le traitement qui est associé :

- aux interruptions de programme issues du système d'interruption, y compris le réveil interprocesseurs (IPI),
- aux alarmes machine et de programmation,
- au défaut secteur.

5.1 - MÉCANISMES GÉNERAUX

5.1.1 - Hiérarchie

Les tâches immédiates sont hiérarchisées par priorité décroissante dans l'ordre suivant :

- tâche défaut secteur,
- tâche immédiate 0 (alarmes)
- tâche immédiate 1 (interruptions de niveau 1)
- tâche immédiate 2 (interruptions de niveau 2)

- tâche immédiate 15 (interruptions de niveau 15)

L'ensemble des tâches immédiates est plus prioritaire que l'ensemble des tâches différées (cf option scheduler).

Dans le cas des tâches immédiates. (no 1 à 15) seul le niveau de l'interruption intervient pour la hiérarchie de ces tâches : Chaque tâche doit ensuite identifier le sous-niveau appelant le plus prioritaire par l'instruction ACK.

Les interruptions de niveau 0 ne sont utilisées que pour la gestion du pupitre opérateur. Ces interruptions sont traitées par microprogramme et ne sont pas reflétées au niveau programme. Ces interruptions peuvent être masquées par IM en marche programme, ce qui n'est pas le cas pour les alarmes.

5.1.2 - Election

Le lancement d'une tâche immédiate ne peut avoir lieu que :
- si aucune tâche plus prioritaire n'est déjà en cours de traitement,
- et si la tâche n'est pas masquée (cf masquage paragraphe 5.1.6)

Chaque demande d'interruption est mémorisée au niveau du coupleur correspondant jusqu'à son acquittement par SIO (cf chapitre entrées-sorties).

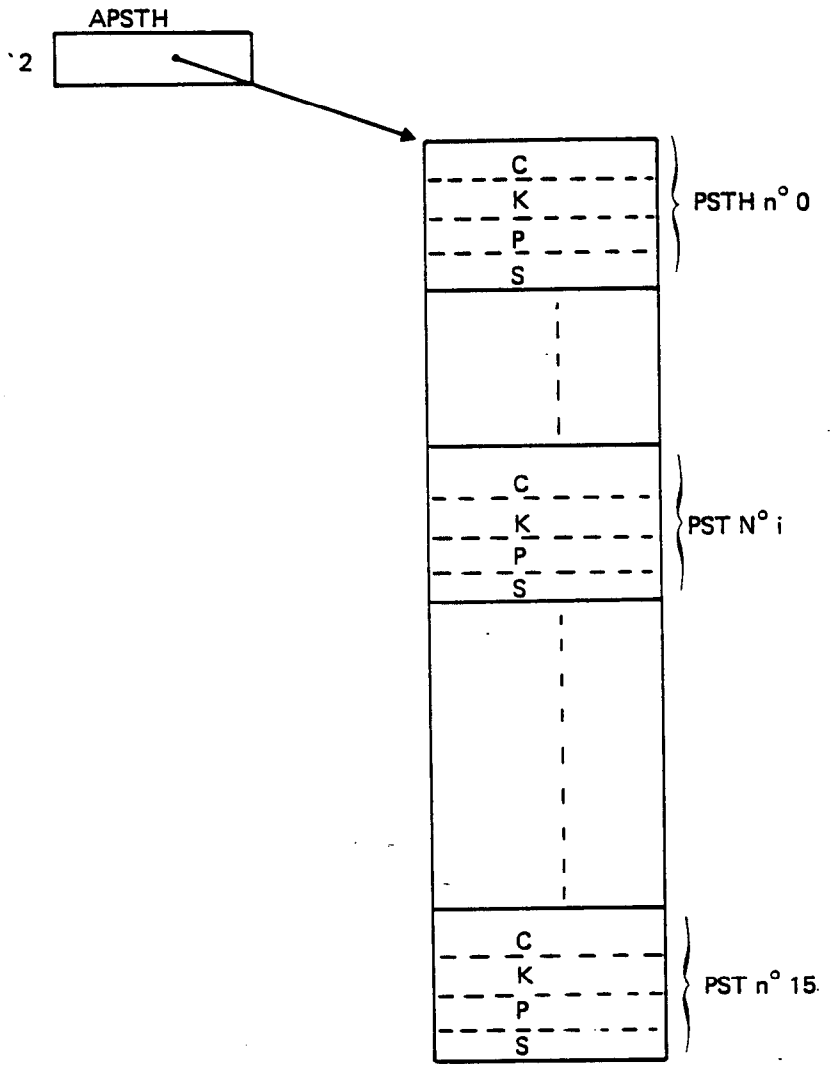
L'ensemble des demandes d'interruption peut être testé par le processeur au cours d'un cycle d'interrogation des interruptions sur le bus d'entrées-sorties (polling). Ce mécanisme permet de prendre en compte les interruptions mises en attente pendant le traitement des tâches plus prioritaires (cf ACQ paragraphe 5.1.5).

5.1.3 - Changement de contexte : PSTH

Physiquement une tâche est déterminée par son numéro de priorité et son contexte. Le contexte (ou PSTH : Program Status Table Hardware) fixe les valeurs initiales de certains registres et détermine en particulier le programme associé à la tâche.

Dans le cas des tâches immédiates ce contexte est qualifié de partiel car il se réduit aux registres C, K, P, S. Les autres registres utilisés par la tâche doivent donc être sauvegardés et restaurés par programme.

Les différents contextes de tâche immédiate sont regroupés dans une table pointée par le contenu de la mémoire débanalisée d'adresse '2 (APSTH) :



Au lancement d'une tâche immédiate de rang i, le contenu des registres C, K, P, S et celui des 4 mots du contexte de rang i sont échangés de sorte que simultanément on restaure le contexte réduit de la tâche interrompante et on sauvegarde celui de la tâche interrompue dans le contexte de la tâche interrompante.

Inversement, en fin de tâche immédiate (cf ACQ paragraphe 5.1.5), les registres C, K, P et S sont échangés avec le contexte de la tâche. Ainsi, simultanément, on sauvegarde les registres de la tâche abandonnée et on restaure les registres de la tâche qui avait été interrompue.

Par contre, un contexte n'a aucune signification pendant l'exécution de sa tâche associée.

Le registre S ne comporte que cinq bits qui participent donc seuls aux changements de contexte, les autres bits de ST n'étant pas modifiés. Par contre les autres bits du mot S des contextes immédiats sont modifiés et sont sans signification sauf dans le cas de la tâche alarme. Dans ce cas, les huit bits poids faibles donnent le numéro de l'alarme.

5.1.4 - Registre HV

Le registre HV mémorise en permanence le rang des différentes demandes d'interruption qui ont été prises en compte et dont les tâches sont donc en cours de traitement :
- le bit de rang i de HV correspond à la tâche immédiate no i.

La tâche défaut secteur est un cas particulier et n'apparaît pas dans HV.
Le registre n'est accessible qu'en lecture avec l'instruction RDHV.

Une tâche apparaît dans HV lors de sa prise en compte effective (élection et changement de contexte). Inversement elle disparaît de HV lors de son acquittement (cf ACQ paragraphe 5.1.5).

Si plusieurs tâches apparaissent dans HV, seule la tâche la plus prioritaire est effectivement en cours d'exécution. Les autres tâches ont été interrompues et sont en attente. Leur exécution reprendra quand le traitement des tâches plus prioritaires sera terminé (cf ACQ paragraphe 5.1.5).

Si une demande d'interruption arrive pendant l'exécution d'une tâche plus prioritaire, elle est mise en attente et donc n'apparaît pas immédiatement dans HV. Elle n'y apparaîtra qu'à sa prise en compte en fin de traitement des tâches plus prioritaires (cf ACQ paragraphe 5.1.5).

5.1.5 - Acquittement des tâches immédiates : ACQ

Toute tâche immédiate sauf la tâche défaut secteur) doit se terminer par l'instruction ACQ
Cette instruction a pour rôle :
- de restituer le contexte partiel (C, K, P, S) de la tâche interrompue,
- de remettre à zéro le bit correspondant de HV,
- de prendre en compte les interruptions en attente si elles sont plus prioritaires que la tâche interrompue,
- de relancer la tâche interrompue.

De plus cette instruction remet à zéro les masques IOM et IPM.

Sur cette instruction le contexte sauvegarde pour la tâche en cours sera le contexte utilisé au prochain appel de cette tâche. II contient en particulier comme pointeur l'adresse de l'instruction qui suit l'instruction ACQ. On doit donc en général y trouver une instruction de rebouclage en début de tâche, d'où la notion de tâche bouclée.

Si la tâche interrompue était une tâche immédiate cette tâche sera reprise même si entre temps une tâche plus prioritaire l'a rendue masquée à travers le registre IM.

Si la tâche interrompue était une tâche différée et si l'option scheduler est présente, on effectue une élection de tâche différée avant de relancer l'exécution. II peut donc en résulter un changement de tâche différée.

5.1.6 - Masquage

On peut masquer les tâches immédiates à l'aide des masques IM, IOM, IPM. Toutefois la tâche alarme ne peut jamais être masquée et on ne peut masquer une tâche interrompue ou en cours de traitement.

Le registre IOM masque à la fois :
- les interruptions de programme,
- le défaut secteur,
- le pupitre opérateur en marche programme.

Ce masque est programmable par les instructions DIT, EIT, RST, SST. Toutefois, le démasquage de IOM par l'instruction RST ne provoque pas la prise en compte immédiate des interruptions en attente ; ceci contrairement à l'instruction EIT.

Le registre IM masque sélectivement les différents niveaux d'interruptions de programme ; ce registre est programmable par l'instruction XIMR. Une interruption masquée est mise en attente et sera immédiatement prise en compte à son démasquage par XIMR ou EIT. (Le bit IMo ne masque pas les alarmes, mais le pupitre opérateur en marche programme).

Le registre IPM masque l'interruption interprocesseurs, donc l'initialisation des canaux intégrés au processeur, mais aussi l'alarme "réveil interprocesseurs". Cette interruption est mémorisée par le masquage et prise en compte immédiatement à son démasquage par l'instruction RST.

De plus les deux registres IOM et IPM sont remis à zéro par l'instruction ACQ et les différentes instructions de l'option scheduler (RQST : RLSE, ...).

5.1.7 - Interruptibilité

L'aptitude du calculateur à répondre aux interruptions est mesurée par son temps de réponse. Il correspond au temps maximal au bout duquel une interruption prioritaire et non masquée provoquera un changement de contexte quel que soit le traitement en cours.
Les points d' interruptibilité se situent :

- entre chaque instruction,
- au cours de certaines instructions dites longues.

A titre d'exemple le temps de réponse sur SOLAR 16-65 est de l'ordre de 7 μs maximum mais en moyenne inférieur à 2 μs vu la durée des instructions.

Parmi les instructions longues on distingue deux classes :

- à réexécution complète telles MP/DV microprogrammées et les décalages,...
- à exécution poursuivie telles MOVE, SBS, ...

On doit ajouter au temps de réponse du calculateur le temps de prise en compte de l'interruption qui correspond **en fait au changement de contexte partiel (soit environ 7 μs sur SOLAR 16-65).**

Enfin les canaux intégrés à un calculateur étant traités en priorité sur les différents traitements programmés, il est bien évident que ces notions n'ont un sens absolu qu'en l'absence de tels canaux.

5.2. - TACHES D'INTERRUPTION (niveau 1 à 15)

5.2.1. - Sous-niveaux : ACK

A chaque source d'interruption de programme est attaché un niveau et un sous-niveau. Seul le niveau intervient dans la hiérarchie de prise en compte des interruptions et donc de lancement des tâches immédiates. Ainsi l'arrivée d'un nouveau sous-niveau du niveau en cours de traitement ne provoque pas l'interruption de la tâche correspondante même si ce sous-niveau est plus prioritaire que les sous-niveaux déjà présents.
Ces sous-niveaux ne servent donc qu'à identifier les différents appels qui sont regroupés sur un niveau.
Cette identification est réalisée par l'instruction ACK qui effectue une interrogation (polling) des différents sous niveaux du niveau en cours de traitement et donne dans X et les indicateurs le rang du sous-niveau le plus prioritaire ainsi que son type.

Un sous-niveau est de type :

- normal si l'appel correspond à un échange d'information,
- exception dans les autres cas (défauts,...).

Cette instruction permet aussi de vérifier qu'aucun sous-niveau ne reste en attente avant de procéder à l'acquittement de la tâche. En effet, si des sous-niveaux sont encore présents à l'exécution de l'instruction ACQ, ils provoqueront la relance immédiate de la tâche quittée et donc un changement de contexte supplémentaire.

L'ensemble des sous-niveaux en attente sur un niveau peut être lu par des instructions SIO spéciales dites de polling (voir paragraphe entrées-sorties).

L'instruction ACK ne doit pas être utilisée dans les tâches alarme et défaut secteur.

5.2.2 - Acquittement de l'interruption

L'acquittement de l'interruption correspond à la remise à zéro de la cause de l'appel.
Cette opération n'est pas effectuée par l'instruction ACK qui ne fait qu'identifier l'appel ni par l'instruction ACQ qui détermine la fin du traitement d'une tâche, mais par une instruction SIO appropriée :

- SIO entrée ou sortie de donnée pour un sous-niveau normal,
- SIO entrée du mot d'état pour un sous-niveau exception.

Faute de cette action, le sous-niveau reste actif et la tâche immédiate correspondante ne pourra être acquittée puisqu'elle sera systématiquement relancée sur l'instruction ACQ pour traiter ce niveau.

5.2.3 - Structure d'une tâche d'interruption

Compte tenu des paragraphes précédents, une tâche d'interruption doit donc être constituée séquentiellement de :

- sauvegarde complémentaire des registres,
- reconnaissance du sous-niveau,
- traitement et acquittement du sous-niveau,
- test des autres sous-niveaux,
- acquittement de la tâche,
- rebouclage.

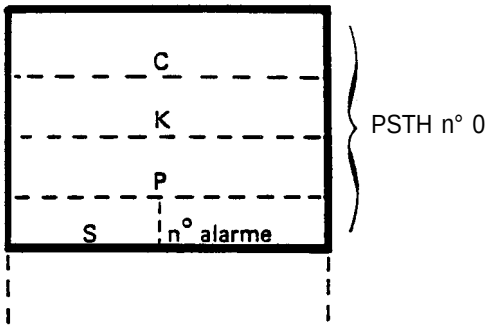
Soit en assembleur, le squelette de programme suivant :

```
DEBUT : PSR      A, X      <<sauvegarde complémentaire minimale
        ACK                <<reconnaissance du sous-niveau
SNIV:   BR       &TABAD    <<aiguillage suivant le sous-niveau
        ≀
        SIO      PERIF     <<traitement et acquittement du sous niveau d'interruption
        ≀
        ACK                <<test des autres sous-niveaux
        JCV      SNIV
        PLR      A, X      <<restitution du contexte complémentaire
        ACQ                <<acquittement de la tâche
        JMP      DEBUT     <<  rebouclage
```

5.3. - TACHE ALARME

5.3.1 - Généralités

Les alarmes sont dues soit à des erreurs de programmation, soit à des défauts du matériel. On y rattache aussi des pseudo-alarmes générales par programmation par exemple pour l'aide à la mise au point.
Dans tous les cas l'incident provoque le lancement de la tâche immédiate n° 0. Cette tâche ne peut être masquée.
Les alarmes sont numérotées et le numéro de cette alarme est placé dans l'octet droit du mot S du contexte de la tâche immédiate 0.



L'instruction ACK ne doit pas être utilisée sous cette tâche.
Par contre la tâche alarme se termine également par l'instruction ACQ.

Dans le contexte de la tâche responsable de l'alarme, la valeur des registres modifiés par l'instruction en faute n'a pas de signification. En particulier le pointeur n'a pas nécessairement été incrémenté. De plus certaines instructions (PSR, LAR, ...) peuvent laisser la tâche dans un état maître ou esclave anormal.
Ces restrictions n'existent évidemment pas dans le cas des pseudo-alarmes programmables.

Si l'exécution de la tâche alarme provoque une nouvelle alarme alors le calculateur passe à l'arrêt en allumant le voyant alarme du pupitre de commande. Seule une réinitialisation au pupitre sortira le calculateur de cet état.

5.3.2 - Liste des alarmes

Mémoire inexistante (n° 0, 05)
résulte d'une tentative d'accès à une adresse de mémoire n'existant pas dans la configuration.

Protection mémoire (n° 1, 05)
est causée par une tentative d'accès hors des limites définies par le contenu des registres SLO et SLE lorsque l'option DRPS est présente.
Cette alarme survient généralement en mode esclave, mais certaines instructions (PSR. LAR,...) peuvent également la provoquer en mode maître.
Enfin, les instructions RCDA et WCDA peuvent causer cette alarme en cas d'accès hors des limites de la zone de données commune (CDA).
Dans le cas d'une écriture en mémoire, l'écriture est alors inhibée.

Erreur de parité (n° 2)
est rencontrée lorsque la parité d'un mot mémoire recalculée par le processeur est différente de la parité lue.
Cette alarme ne peut arriver qu'en dehors du mode DEBUG, sinon on obtient l'alarme n° 8.

Instruction optionnelle inexistante
résulte de la tentative d'exécution d'une instruction appartenant à une option absente sur la configuration (scheduler). Les instructions flottantes n'entrent pas dans cette catégorie puisqu'en cas d'absence de l'opérateur, elles sont transformées en SVC.

Instruction privilégiée (n° 4, 05)
Cette alarme correspond à l'exécution d'une instruction privilégiée en mode esclave :
- SIO, IPI, ROMB, XIMR, ACQ, RQST, LAR,...

Instruction interdite sous tâche immédiate (n°5)
causée par les instructions RQST, WAIT, QUIT qui sont interdites dans une tâche immédiate.

Réveil interprocesseurs (n° 6, 05)

pseudo-alarme créée par la réception d'une interruption IPI de réveil interprocesseurs (opérande '8000). Contrairement aux autres alarmes, si cette interruption arrive pendant le déroulement de la tâche alarme, l'interruption est ignorée. Par contre, si le masque IPM est à 1, l'interruption reste mémorisée et sera prise en compte en fin de tâche (ACQ).

Mode pas à pas (n° 7)

pseudo-alarme engendrée par l'exécution de l'instruction STEP. Ceci permet d'exécuter un programme en pas à pas en surveillant son comportement depuis la tâche alarme qui est ainsi relancée à chaque instruction.

Point d'arrêt (n° 8)

pseudo-alarme créée par une erreur de parité mémoire survenue alors que le processeur est en mode DEBUG. Dans ca mode les écritures mémoire deviennent des échanges et toute erreur de parité est interprêtée comme un point d'arrêt. Dans le cas d'une écriture le point d'arrêt est simultanément effacé.
Ce point d'arrêt met le calculateur à l'arrêt si le bit POP de ST est à 1 (mode pupitre), ce qui est le cas si les points d'arrêts ont été positionnés au pupitre. Dans le cas contraire (POP = 0), le calculateur ne passe pas à l'arrêt, mais exécute une alarme n° 8.
Si le point d'arrêt est positionné sur une instruction, l'alarme a lieu avant exécution de cette instruction ; si le point d'arrêt est positionné sur un opérande, l'alarme a lieu après exécution complète de l'instruction.
Les points d'arrêt rencontrés dans la tâche alarme sont ignorés.

Alarme ACTD (n° 9)

pseudo-alarme créée par l'exécution de l'instruction ACTD.
Cette instruction permet à toute tâche de provoquer l'exécution de la tâche alarme.

5.4 - TACHE DÉFAUT SECTEUR ET RELANCE AUTOMATIQUE

L'alimentation des calculateurs détecte les défaillances du secteur et prévient l'unité centrale dès qu'un défaut dure plus de 20 à 40 ms (en 220V 50Hz). Dans ce cas si le masque IOM est à 0, l'unité centrale :

- inhibe les canaux,
- inhibe les interruptions de programme,
- positionne les bits IOM, IPM et LCM du registre ST à 1,
- échange le contenu des registres C, K, P et S avec celui du contexte réduit de la tâche défaut secteur ; ce contexte est défini par les quatre mots pointés par le mot débanalisé SECT (adresse 6),
- passe le contrôle au programme ainsi défini :

La prise en compte du défaut secteur est inhibée si le masque IOM est à 1, mais reste mémorisée et est activée dés que ce masque repasse à 0. De plus ce mécanisme n'a lieu que si le calculateur est en marche programme (RUN).

La tâche défaut secteur dispose au minimum de 10 ms pour sauvegarder les informations nécessaires et en particulier les registres programmables, mais on doit tenir compte du fait que les processeurs sont ralentis dans un rapport de 2 à 3 pendant ce traitement.

Cette tâche doit en particulier :

- sauvegarder les informations nécessaires dans une zone mémoire secourue,
- être située elle-même dans une zone mémoire secourue et réinitialiser son propre contexte pour être disponible en cas de défaut secteur pendant la phase de réinitialisation,
- préparer la réinitialisation du calculateur lors de la réapparition du secteur (en particulier la mémoire INI),
- prévenir les processeurs d'entrées-sorties IOP16-M du défaut secteur en leur envoyant un CCB spécial (se réduisant à '4000).
- se terminer par une instruction HALT afin d'éviter tout cycle mémoire incorrect au moment de la disparition définitive de la tension.

De plus la tâche défaut secteur doit laisser le bit IOM à 1 et ne pas modifier le bit DEBUG.
Une alarme quelconque intervenant pendant la tâche défaut secteur provoque le passage du calculateur à l'arrêt, inhibant ainsi le mécanisme de relance automatique à la réapparition du secteur.

Le dispositif de relance automatique effectue une séquence identique à la mise sous tension avec lancement immédiat du programme dû au fait que le calculateur est resté en marche programme (RUN) soit en particulier :

- initialisation générale du système,
- lancement en mode maître, tous niveaux masqués, du programme à l'adresse contenue dans le mot mémoire débanalisé INI (adresse '8).

Les périphériques, ayant des alimentations autonomes, ont une détection des défauts secteur indépendante et spécifique de chacun d'entre eux. Dans la pratique, leur détection est plus rapide que celle de l'unité centrale et ils peuvent voir des défauts secteur qui passeront inaperçus pour celle-ci.

6 — ENTRÉES - SORTIES

6.1. - GENÉRALITÉS

6.1.1 - Avertissement

La programmation des entrées/sorties sans utiliser IOCS nécessite, outre la connaissance du fonctionnement des instructions SIO, IPI et du système d'interruption, une connaissance détaillée du fonctionnement des périphériques. La description de ces périphériques ne pouvant être abordée dans le présent manuel on s'est limité à donner une vue générale, très approximative, de la programmation des entrées/sorties sans IOCS.
En particulier ce chapitre introduit la notion de coupleur standard dont la périphérie SOLAR offre de nombreux exemples, mais aussi un certain nombre d'exceptions.

On rappelle, d'autre part que, sauf cas particulier, le programmeur n'a pas intérêt à entreprendre la réalisation de programmes ou sous-programmes d'entrée/sortie réalisant les mêmes fonctions qu'IOCS, et que s'il est amené à apporter des extensions à ce système (écriture de nouveaux drivers) il est nécessaire qu'il se conforme aux conventions utilisées dans celui-ci et aux manuels décrivant le fonctionnement du périphérique :

- manuel de référence d'IOCS,
- manuel d'utilisation d'IOCS,
- manuel d'exploitation du périphérique.

6.1.2 - Types de périphériques

Un périphérique constitue pour l'unité centrale soit un moyen de stockage des données soit un moyen de communication avec l'extérieur. La première catégorie comporte par exemple des lecteurs et perforateurs de bande, des disques et des bandes magnétiques. La seconde catégorie comporte des périphériques dont les principes de fonctionnement sont très différents les uns des autres, tels que console de visualisation alphanumérique, entrée tout ou rien, sorties analogiques, chaînes de mesures, matériel de télécommunication, etc...

Pour pouvoir donner une vue générale de leur programmation on a classé les périphériques plus ou moins arbitrairement, selon les trois modes de fonctionnement suivants :

- périphériques, par exemple, entrée ou sortie tout ou rien, sur lesquels une écriture ou une lecture s'effectue immédiatement,
- périphériques asynchrones, par exemple lecteur de bande perforée, sur lequel chaque caractère n'est lu que lorsque le programme en fait la commande au lecteur,
- périphériques synchrones, par exemple lecteur de bande magnétique, sur lequel les caractères sont lus à la vitesse de défilement de la bande devant la tête de lecture.

A ce sujet on notera que les transmissions sur lignes asynchrones se comportent comme des périphériques synchrones au cours d'une réception puisque le calculateur doit s'asservir à l'émetteur.
Ce n'est évidemment pas le cas en émission.

Dans ce qui suit on ne s'intéressera qu'à ce qui concerne directement la programmation et on appellera indistinctement périphérique l'ensemble des dispositifs qui assurent l'entrée et la sortie des données (sans distinguer entre coupleur, partie mécanique et partie électronique du périphérique, câble, etc...)

On appellera écriture ou sortie d'information l'envoi de données de l'unité centrale vers un périphérique, lecture ou entrée d'information la réception de données par l'unité centrale.

Suivant les périphériques, ces informations sont échangées par caractères (8 bits en parallèle) ou par mots (16 bits en parallèle).

6.1.3 - Modes d'échanges

Selon les périphériques on peut utiliser plusieurs modes d'échanges auxquels correspondent des programmations différentes :

– le mode programme simple permet de faire une lecture ou une écriture par l'intermédiaire du registre A, sans simultanéité entre les entrées/sorties et le déroulement du programme. Pour cette raison ce mode est très peu utilisé (essentiellement dans les utilitaires : chargeur,...),

– le mode programmé prioritaire ou gestion par interruption contraint lui aussi le programme à effectuer les échanges caractère par caractère (ou mot par mot) par l'intermédiaire du registre A, mais avec simultanéité entre ces échanges et le programme. Chaque caractère (ou mot) génère une interruption de programme qui provoque la suspension du programme courant au profil de la tâche immédiate attachée à l'interruption.

– le mode canal (HDC, MDC ou LDC) permet l'échange de blocs de caractères (ou de mots) directement entre la mémoire et le périphérique, avec simultanéité entre ces échanges et le programme. Le programme n'intervient qu'au début et à la fin de l'échange ; toutefois l'échange consomme du temps d'unité centrale, sauf si le canal est supporté par le processeur IOP16-M.

6.2. - NOTION DE COUPLEUR STANDARD

Physiquement un coupleur se présente sur une carte au format 1/2 (parfois 1/1). Usuellement une carte gère un seul périphérique, mais certaines multiplexent en fait, plusieurs périphériques.

Plusieurs types de multiplexages existent :
– multiplexage vrai, soit de périphériques différents (lecteur et perforateur de bande papier,...), soit de périphériques identiques (lignes de télétransmission,...)
– multiplexage partiel (ex : disques et bandes magnétiques sur formateur). Dans ce cas le même coupleur gère plusieurs périphériques vrais, un seul est en échange à la fois.
Dans tous les cas on peut en première approximation considérer que chaque périphérique est géré par un coupleur standard monopériphérique.

Du point de vue de la programmation, un coupleur se compose :
- de registres d'entrées-sorties de 16 bits (usuellement 3 ou 4).
- d'un système d'interruption.

Registres

L'information échangée en lecture ou en Ecriture est mémorisée dans les registres d'entrées ou de sorties du coupleur.
Ces registres sont adressés par l'opérande de l'instruction SIO.
On distingue essentiellement 4 types de registres dont le codage porte sur les bits 14 et 15 de leurs adresses :

	bit 14	bit 15
- registre d'entrée d'information	0	0
- registre de sortie d'information	0	1
- registre d'entrée du mot d'état	1	0
- registre de sortie du mot de commande	1	1

Le bit 15 code toujours le sens de l'échange (0 : entrée, 1 : sortie).

L'accès à un registre inexistant conduit :
- en sortie à une opération ineffective
- en entrée à l'acquisition d'un mot nul.

Interruptions

Un coupleur comporte usuellement deux sources d'interruption :
– interruption "normale" associée à un échange d'information en lecture ou en écriture,
– interruption "exceptionnelle" associée aux défauts et événements divers.

6.2.1 - Registres d'information

L'information est soit un mot, soit un octet.
Lors d'une entrée-sortie entre un coupleur et un processeur, l'information échangée est toujours un mot de 16 bits, mais si l'information utile est un octet :
- en lecture l'octet utile est en poids faibles et les poids forts sont à zéro.
- en écriture l'octet utile est également en poids faibles, mais l'octet poids forts peut être quelconque.

En particulier l'instruction SIO réalise :
- en lecture le transfert du registre d'entrée adressé dans l'accumulateur,
- en écriture le transfert de l'accumulateur dans le registre de sortie adressé.

6.2.2 - Registre d'état

Les SIO d'entrée du mot d'état d'un périphérique sont utilisées soit en cas de mauvais fonctionnement (défauts tels que : périphérique hors tension, erreur sur bande magnétique, erreur de cadence sur un lecteur de cartes, etc...), soit pour tout autre événement détecté au niveau d'un périphérique (appel manuel sur un téléimprimeur, fin de fichier sur bande magnétique, etc...), soit pour connaître l'état d'un périphérique (périphérique sous tension ou hors tension, unité de bande magnétique fonctionnant en haute ou en basse densité, etc...).

Le mot d'état est envoyé par le périphérique dans le registre A pendant l'exécution de l'instruction SIO et permet de connaître la cause du défaut ou l'événement qui est survenu.

Les SIO d'entrée du mot d'état d'un périphérique ont un fonctionnement indépendant du mode d'échange utilisé pour les lectures ou les écritures.
L'entrée du mot d'état se fait toujours, que le périphérique soit ou non occupé.

Le contenu des mots d'État dépend des périphériques concernés. Toutefois il existe un standard que l'on retrouve pratiquement sur tous les coupleurs :

- bit 0 regroupement des défauts réels. Les événements (fin d'échange,...) ou états ne participent pas dans ce bit,
- bit 1 erreur de cadence en entrée ou sortie,
- bit 2 information incorrecte (parité, checksum,...),
- bit 3 tentative de viol (protection écriture,...)
- bit 5 chien de garde,
- bit 6 appel opérateur (break,...)
- bit 7 fin de bloc en sortie (voie Full-Duplex seulement),
- bit 8 en sortie le registre information peut recevoir la donnée suivante (voie Full-Duplex seulement),
- bit 13 fin de bloc en entrée ou sortie (en entrée seulement sur une voie Full-Duplex)
- bit 14 . en entrée le registre information contient la donnée suivante
  . en sortie le registre information peut recevoir la donnée suivante (sauf sur une voie Full-Duplex).
- bit 15 le coupleur ou la voie est opérationnel (présent, sous-tension,...)

Certains coupleurs nécessitent un autre mot d'état qui est alors spécifique du périphérique (disques,...)

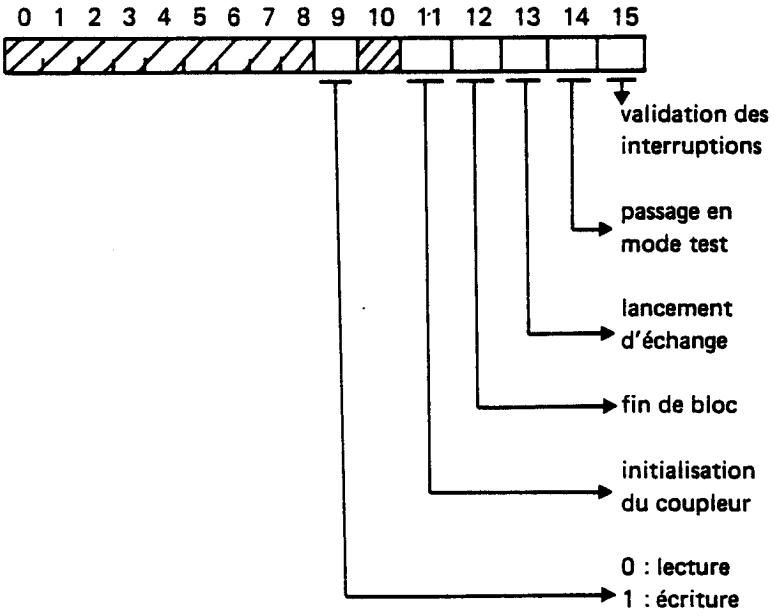
6.2.3 - Registre de commande

Les SIO de commande sont utilisées pour faire effectuer aux périphériques les opérations qui ne comportent pas de transfert de données, telle que : mettre sous tension un téléimprimeur, rebobiner une bande magnétique, déclencher la lecture ou l'écriture d'un bloc sur une bande magnétique, inhiber ou valider les demandes d'interruption de fin d'opération sur un périphérique.

Les SIO de commandes ont un fonctionnement indépendant du mode d'échange utilisé pour les lectures et les écritures. Les opérations à effectuer par un périphérique lors d'une SIO de commande sont déterminées par le mot de commande. Ce mot de commande, qui doit être chargé dans le registre A avant d'effectuer la SIO, dépend du périphérique concerné.

On distingue plusieurs types de commandes :
- les commandes immédiates qui n'engendrent donc pas d'occupation sur le coupleur (validation des interruptions,...)
- les commandes longues sans échange qui engendrent une occupation du coupleur et/ou du périphérique sans être associées à des transferts d'information (rebobinage de bandes magnétiques, déplacement de bras sur un disque,...). La fin d'occupation sera alors signalée par la montée du bit correspondant à l'événement dans le mot d'état ; si les interruptions sont validées on aura simultanément l'émission d'une interruption "exception" spécifique du coupleur.
- les commandes de lancement d'échange qui initialisent le transfert effectif des informations. En particulier, si les interruptions sont validées, ces commandes provoquent la montée d'une interruption normale spécifique du coupleur pour chaque entrée ou sortie d'information.

La séquence et le type de commandes à exécuter sur un coupleur donné dépendent de celui-ci. On se reportera au manuel d'exploitation du périphérique pour en connaître le détail.
Toutefois un certain nombre de commandes ont un caractère universel et sont standardisées :



6.2.4 - interruption normale

Les interruptions de type normal sont liées aux transferts effectifs donc aux registres d'informations.
Quand les interruptions sont validées sur un coupleur, celui-ci génère une interruption normale :
- en entrée lorsque la donnée est valide dans le registre d'information et peut donc être acquise par une SIO d'entrée,
- en sortie lorsque le registre d'information est libre et qu'il doit donc être chargé par une SIO de sortie.

L'interruption reste active et donc identifiable (par "polling") jusqu'à son acquittement. Celui-ci est réalisé par l'exécution de la SIO d'entrée (ou de sortie) d'information sur le registre d'entrée (ou de sortie) associé à l'interruption.

Dans le cas d'un couplage en mode canal, le fonctionnement est strictement semblable. Les appels d'interruption ne sont plus aiguillés vers le programme (bit LDC, MDC ou HDC du trap word). L'acquisition des informations et l'acquittement de l'interruption se font également par accès aux registres d'information, les cycles d'entrées-sorties étant dans ce cas lancés par microprogramme.

6.2.5 - Interruption exception

Ces interruptions signalent des événements autres que les transferts réels d'information :
- les défauts intervenant en cours d'échange (erreur de parité, erreur de cadence,...)
- événements normaux hors échange (fin de rebobinage, de déplacement de bras,...)
- fin d'échange (ou fin de bloc).

L'acquittement d'une interruption exception se fait par acquisition du mot d'état qui y est associé. L'analyse de ce mot d'état permet en outre de connaître la cause de cette interruption.

Dans le cas d'un couplage en mode canal l'événement "fin d'échange" est particulièrement important. En effet cet événement est généré par les canaux qui, lors du transfert de la dernière donnée d'un échange, provoquent cette interruption en envoyant une commande "fin de bloc" au coupleur. C'est donc par cette interruption que le programme est averti de la fin de l'échange qu'il a initialisé.

6.3.1 - Mode programmé simple

Ce mode d'échange n'est normalement pas utilisé car il ne permet pas la simultanéité entre le déroulement des entrées-sorties et l'exécution du programme.
II est toutefois utilisé avec certains coupleurs (entrées-sorties tout ou rien) et par quelques programmes utilitaires (chargeurs, configurateurs,...)

En lecture, comme en écriture, les échanges se font mot par mot (ou octet par octet) entre l'accumulateur et le coupleur.

Avant d'accéder au registre d'information, il faut normalement tester la validité de la donnée en entrée ou l'occupation du registre en sortie soit :

```
ENTREE :    SIO     STATUS      <<entrée du mot d'état
            TBT     14          <<test de validité
            JNC     $ - 2       <<attente de validité
            SIO     ADIN        <<entrée de la donnée
            STA     BUF         <<rangement

SORTIE :    SIO     STATUS      <<entrée du mot d'état
            TBT     8           <<test d'occupation
            JNC     $ - 2       <<attente de libération
            LA      BUF         <<chargement de la donnée
            SIO     ADOUT       <<sortie de la donnée
```

Ces séquences supposent l'exécution préalable de ou des SIO commandes appropriées au périphérique. De plus les interruptions ne doivent pas être validées ou doivent être masquées.

6.3.2 - Mode programmé prioritaire

Ce mode d'échange est le mode qui est normalement utilisé pour un périphérique qui n'est pas connecté en mode canal. II permet d'obtenir la simultanéité entre le déroulement des échanges et l'exécution du programme.

L'échange doit être initialise par les SIO commandes appropriées avec validation des interruptions ; celles-ci doivent par ailleurs être démasquées (IOM = 0, IMi = 0).

Lorsqu'une donnée est valide en entrée ou lorsque le registre est libre en sortie le coupleur émet une interruption de type normal. Celle-ci interrompt le programme en cours au profit de la tâche immédiate attachée à cette interruption. Après l'analyse du sous-niveau d'interruption, si celui-ci est de type normal, la tâche immédiate donc accéder au registre d'information (SIO d'entrée ou de sortie) ce qui a simultanément pour effet d'acquitter cette demande d'interruption (voir paragraphe 5.2.3).
De même si le coupleur détecte un défaut (parité, cadence,...) celui-ci émet une interruption de type exception que la tâche immédiate correspondante doit acquitter par lecture du mot d'état.

6.3.3 - Mode canal

Ce mode d'échange est impératif pour les périphériques à cadence élevées (au-delà de quelques K octets/s).
II offre de plus l'intérêt de réduire la charge du processeur au cours de l'entretien de l'échange, ceci à fortiori si on utilise l'IOP16-M.

Performances en K mots/s

	04/05	40	65/75	30/35	70	IOP16M
HDC		470	590	370	1000	390
MDC		290		220		290
LDC	40	44	60/70	40	285	50

Ces valeurs définissent la charge maximale admissible en saturant le type de canal considéré. La saturation peut s'obtenir avec deux ou plusieurs coupleurs ou une conception spécifique d'un coupleur.

Les canaux intégrés aux processeurs de traitement y consomment de la puissance proportionnellement au débit des périphériques connectés ; l'exécution des programmes s'en trouve donc ralentie d'autant (ex : environ 20 % pour un disque à têtes mobiles sur 65).
Quand cette charge devient prohibitive, il convient de passer les périphériques sur un processeur d'entrées-sorties IOP16-M qui assurera alors les échanges en perturbant au minimum l'unité centrale. Il peut toutefois subsister des conflits d'accès aux blocs mémoire que l'on peut négliger en première approximation.

Le processeur 16-70 peut être utilisé en processeur d'entrée-sortie rapide. Dans ce mode on dispose des canaux HDC et LDC avec les mêmes performances que pour le 16-70.

a) mécanisme général
Dans une liaison en mode canal, les interruptions de type normal sont émises vers le canal et ne provoquent pas d'interruption de programme. Ces interruptions correspondent à l'entretien de l'échange proprement dit.
Par contre les interruptions de type exception ne sont pas traitées par les canaux mais par les tâches immédiates Ces interruptions correspondent soit à des défauts soit à la fin de l'échange, et ces événements doivent être traités par programme.

Un échange en mode canal doit comporter les phases suivantes dans l'ordre décrit :

– initialisation du canal : cette opération consiste à passer au canal les paramètres de l'échange (adresse mémoire, compte de mots,...) par exécution d'une instruction IPI (voir paragraphe b).

– lancement de l'échange : le canal étant prêt, on peut maintenant lancer l'échange, c'est-à-dire initialiser le coupleur. Cette opération se fait comme en programmé prioritaire en exécutant la ou les SIO commandes appropriées sur le coupleur ; on doit en particulier valider les interruptions du coupleur.

– entretien d'échange
Cette phase est entièrement microprogrammée et se passe à l'insu du programme. Toutefois dans le cas de canaux intégrés à l'unité centrale, celle-ci utilise un certain pourcentage de son temps pour ces entretiens d'échanges (1 à 25 µs par caractère selon le type de canal et le modèle d'unité centrale). Par contre l'unité centrale ne participe pas du tout à l'entretien d'échange si le canal est supporté par un processeur d'entrée-sortie parallèle (IOP16-M).

– fin d'échange
La fin de l'échange est signalée par une interruption de programme de type exception. L'analyse du mot d'état permet de distinguer les différentes fins d'échange possibles :
. défaut détecté par le coupleur,
. fin normale détectée par le canal ou le coupleur (code d'arrêt, fin d'enregistrement magnétique,...)
En particulier, sur détection du compte de caractères nul, le canal avertit le coupleur par une SIO commande "fin de bloc" qui provoque une interruption exception avec le bit "fin de bloc" dans le mot d'état.

- Compte rendu et libération

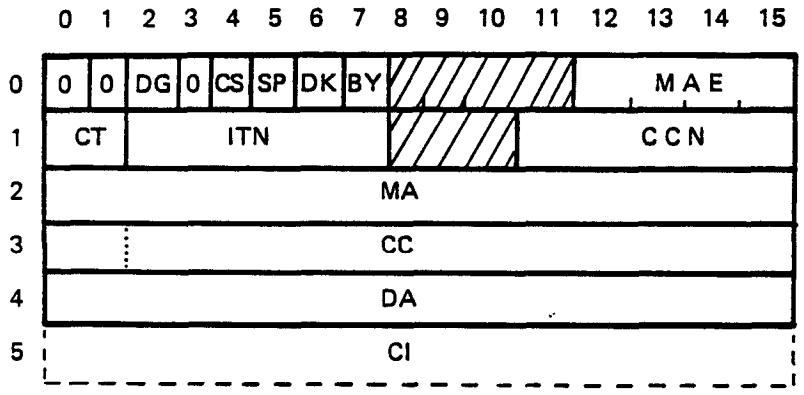
Cette opération de compte rendu est également réalisée par l'instruction IPI, mais avec un CCB différent (voir paragraphe C). Elle est nécessaire à chaque fin d'échange canal pour :

- connaître le nombre de mots ou d'octets effectivement échangés en particulier en cas de défaut,
- effectuer le rangement en mémoire du dernier caractère dans le cas d'une entrée,
- libérer la voie du canal (masquage impératif en particulier sur HDC).

b) initialisation du canal

Le passage des paramètres au canal se fait par une instruction IPI où le paramètre passé dans la boîte aux lettres est précisément l'adresse de la table d'échange qui regroupe ces paramètres (voir l'instruction IPI chapitre 4).

La table d'échange appelée CCB : (Channel Control Block) a la structure suivante :



DG = 1 05 : n'est opérant que pour les processeurs IOP16-M. Si ce bit est à 1 le processeur passe en mode

DEBUG et ignorera les points d'arrêt (imparité mémoire) jusqu'à la remise à zéro de ce bit par un nouveau CCB comportant DG à 0.

CS = 0 : initialisation d'échange
CS = 1 : demande de compte rendu et libération

SP = 0 : canal standard
SP = 1 : le canal est géré par un microprogramme spécial dont l'adresse de lancement est donnée dans le mot 5 (CI). C'est en particulier le cas de l'option EDC 64 et des échanges sur code d'arrêt (voir paragraphe d).

DK = 1 (05) : l'échange a lieu sur un disque à têtes mobiles

BY = 0 : échange sur mots
BY = 1 : échange sur octets

MAE (05) : 4 bits de poids forts de l'adresse mémoire de la table à échanger.

CT = 00 : échange sur canal LDC
CT = 10 : échange sur canal HDC
CT = 11 : échange sur canal MDC

ITN : niveau d'interruption normale utilisé par le coupleur.

	05	40	65/75	IOP16-M	30/35	70
HDC		implicite (0 à 1)	0 à 7	0 à 7	0 à 7	0 à 7
MDC		0 à 4*		0 à 15	0 à 2	
LDC	0 à 15	0 à 63	0 à 63	0 à 63	0 à 63	0 à 63

* 0 à 5 si pas de coupleur canal HDC connecté

CCN : numéro du jeu de registres (ou contexte) canal utilisé par l'échange.

	05	40	65/75	IOP16-M	30/35	70
HDC		implicite	0 ou 1	0 ou 1	implicite	implicite
MDC		0 à 4*		0 à 15	0 à 2	
LDC	implicite	implicite	implicite	implicite	implicite	implicite

* 0 à 5 s'il n'y a pas de coupleur connecté en HDC.

Attention : en mode canal HDC sur (65), et IOP16-M ce contexte doit être considéré comme une ressource. En effet on ne dispose que de 2 contextes pour 8 périphériques connectables ; on ne peut donc avoir que 2 échanges simultanés sur ces périphériques. Il en est de même pour le contexte du (30).

MA : 16 bits poids faibles de l'adresse mémoire de la table à échanger.

CC : compte de caractères à échanger (octets ou mots suivant les périphériques). Ce compte de caractères doit normalement être compris entre 0 et 16 K bornes exclues.

DA : adresse du registre concerné par l'échange sur le coupleur de périphérique. Cette adresse est dans le même format que les opérandes de SIO. De plus elle définit le type de l'échange en fonction des bits 14 et 15 :

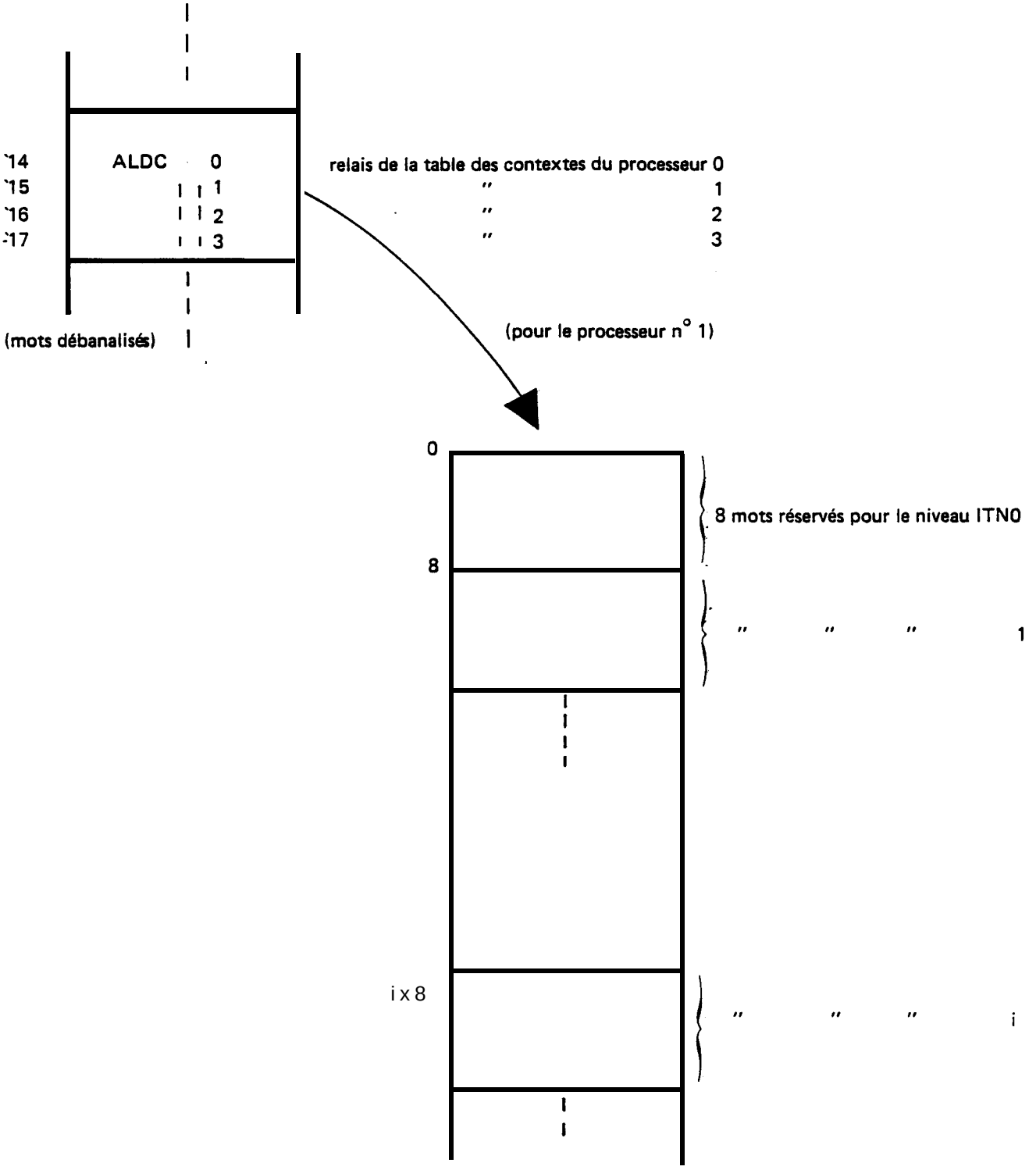
14	15	
0	0	entrée d'information
0	1	sortie d'information
1	0	lecture de contrôle sur disque

CI : information optionnelle
si DK = 1 CI contient l'adresse du premier secteur disque concerné par l'échange
si SP = 1 CI contient l'adresse du microprogramme spécifique d'échange.

Pendant la communication du CCB par IPI, le coupleur doit être au repos et l'unité centrale doit attendre la remise à zéro de la boîte aux lettres avant de lancer l'échange.
De plus dans le cas d'un canal intégré à l'unité centrale on doit au préalable démasquer les interruptions IPI (IPM = 0) ainsi que LDC le cas échéant (LCM = 0).

Les paramètres adresse mémoire et compte de caractères doivent être de telle façon que la table d'échange ne chevauche pas une frontière physique de 64K mots.

Enfin en LDC les processeurs autres que le SOLAR 16-70 utilisent des contextes de travail en mémoire vive. Ces zones mémoire, qui doivent être réservées par programme, mais n'ont pas à être initialisées, ont la structure suivante :



Exemple d'initialisation

```
RBOX :  WORD    BOX, X      <relais vers le 1er mot de la boîte aux lettres.
          .                 <définition du CCB
          .                 <appropriation des ressources (périphérique, canal,...)
          .                 <démasquage de IPI et du canal
        LAD     CCB
        LXI     NPROC       <n° du processeur d'E/S
        DIT                 <appropriation de la boîte aux lettres
        STA     &RBOX
        IPI
        CPZ     &RBOX       <attente accusé de réception
        JNE     $-1
        EIT                 <libération de la boîte aux lettres
          .
          .
          .
        SIO     COMAND      <lancement du périphérique
```

Remarque :
Dans le cas de configuration à plusieurs processeurs de traitement l'appropriation de la boîte aux lettres doit se faire de façon exclusive, par exemple en utilisant l'instruction XM (cf chapitre 4).

c) Compte rendu d'échange

Une demande de compte rendu d'échange se fait comme une initialisation canal avec un CCB différent :

CCB0 : le bit CS doit être à 1, les informations SP, DK, MAE sont ineffectives.
CCB1 : doit contenir les informations CT, ITN, CCN.
CCB2 à CCB5 sont ineffectifs.

Après la prise en compte de la demande de compte rendu (boîte aux lettres à zéro), le mot CCB3 contient le nombre de caractères (mots ou octets) qui restait à échanger au moment de la fin d'échange (soit en général 0). Les autres mots du CCB ne sont pas modifiés.

De plus, dans le cas d'un IOP16-M, les 2 bits de poids forts du compte rendu (CCB3) ont la signification suivante :

0 0	état normal
0 1	état après INI ou défaut secteur
1 0	alarme parité mémoire
1 1	alarme mémoire inexistante.

Si l'IOP16-M n'est pas dans un état normal, tous ses échanges sont alors arrêtés jusqu'à la prochaine initialisation d'échange.

La demande de compte rendu doit être faite à la fin de chaque échange, car elle a les effets secondaires suivants :
— en entrée, rangement du dernier mot ou octet en mémoire car l'acquisition sur le périphérique et le rangement en mémoire étant déphasés, le dernier caractère reste dans le tampon du canal lors du dernier échange.
— libération du canal. Ceci correspond sur HDC en particulier à un masquage de la voie spécifiée par CCN, masquage qui est impératif puisque l'affectation d'un contexte à un niveau d'interruption y est dynamique.

d) Code d'arrêt

Ce type d'échange est un cas particulier d'utilisation des microprogrammes d'entrées-sorties spécifiques.
Ce mode d'échange est utilisable sur tous les processeurs :

- en standard sur IOP-16M, 16-30, 35, 65, 75 et 70
- associé à l'option scheduler sur 16-40
- associé à l'option pupitre opérateur sur 05

Ce canal permet d'effectuer un échange en l'interrompant soit sur la rencontre d'un caractère déterminé (code d'arrêt) soit au bout d'un compte de caractères précisé, ceci avec les restrictions suivantes :

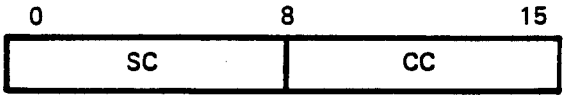
- échanges en entrée seulement,.
- échanges sur octet seulement,
- le code d'arrêt est programmable mais unique pour un échange donné,
- le compte de caractères maximal est de 255 octets.

Pour le fonctionnement avec code d'arrêt il faut positionner le bit 2 à 1 dans les adresses de coupleur qui ne sont pas en format court dans le rack de base. Les adresses ainsi modifiées restent compatibles avec les canaux standard.

Initialisation :

Elle se fait comme pour un canal standard par communication d'un CCB qui comporte alors les différences suivantes :

- CCBO : SP = 1, BY = 1, DK ineffectif
- CCB3 :



CC : compte de caractères (1 à 255)
SC : code d'arrêt pour l'échange

- CCB5 : doit contenir l'adresse du micro programme spécifique soit :
  `7F5 sur 65, 75 et 70
  `DC0 sur 40
  `4DA sur 05
  `2C0 sur IOP16-M
  `DCO sur 30 et 35

Compte rendu

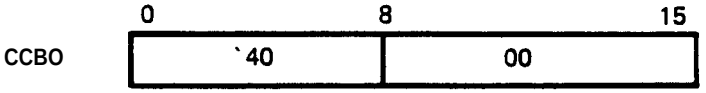
Il se fait comme un compte rendu normal (bit CS = 1) et restitue dans CCB3

- le code d'arrêt utilisé (SC) bits 0 à 7
- le compte de caractère restant à échanger (CC) bits 8 à 15.

e) Défaut secteur

Après la prise en compte du défaut secteur par un processeur 05, 40 ou 65 les différents canaux sont inhibés avant leur fin d'échange.

Par contre le processeur IOP1 6-M ne "voit" pas directement les défauts secteurs, mais on peut également y arrêter les échanges en cours en lui communiquant le CCB spécial suivant :



Il n'y a pas de remise à zéro de la boîte aux lettres dans ce cas particulier.

f) Alarmes

Si un échange canal provoque une alarme mémoire inexistante ou parité :

- sur un processeur de calcul le canal n'est pas inhibé et l'alarme est reflétée à la tâche alarme.
- sur un processeur d'entrées-sorties, tous les échanges sont arrêtés et la cause de l'alarme est mentionnée dans les comptes rendus jusqu'à la prochaine réinitialisation d'échange.

La protection mémoire n'étant pas mise en œuvre dans les échanges canaux, les vérifications d'adresse doivent être faites avant le lancement de l'échange.

Si le processeur qui effectue l'échange est en mode debug, les alarmes parité provoquées par les échanges sont ignorées.

g) Code d'arrêt étendu

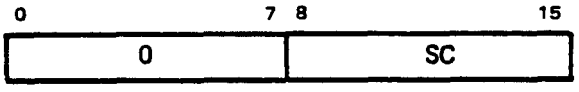
Ce type d'échange est un cas particulier des micro-programmes d'entrée-sortie spécifiques.

Ce mode d'échange est utilisable :
- en standard sur 30, 35 et 70
- associé à l'option EDC 64 sur IOP16M.

Ce canal permet d'effectuer un échange en l'interrompant sur la rencontre d'un caractère déterminé sans limitation du compte de caractères, ceci avec les restrictions suivantes :
- échange en entrée sur octet seulement
- code d'arrêt programmable mais unique pour un échange donné.

Initialisation :

Elle se fait comme pour un canal standard par communication d'un CCB qui comporte alors les différences suivantes :
- CCB0 : SP = 1, BY = 1
- CCB3 :



. SC code d'arrêt pour l'échange
- CCB5 : adresse du micro programme spécifique soit
  '0AF0 pour 30 et 35
  '07EE pour 70
  '0540 pour IOP16-M

Compte rendu :

Il se fait comme un compte rendu normal (bit CS = 1) et restitue dans CCB3 le compte de caractères échangés modulo 64K.
Si le compte de caractère dépasse 16K, il peut y avoir confusion avec l'état anormal d'IOP-16M.

h) Accès direct mémoire par octet

Ce type d'échange est un cas particulier d'utilisation des microprogrammes d'entrées-sorties spécifiques.

Ce mode d'échange est utilisable :
- en standard sur 30, 35 et 70
- associé à l'option EDC64 sur IOP16M.

Ce canal permet de faire des échanges octet par octet en entrée ou en sortie entre la mémoire et un périphérique.
Le coupleur ayant "l'initiative du choix" de l'adresse mémoire octet et du sens de l'échange.

A chaque interruption le microprogramme effectue un échange d'un octet soit en entrée, soit en sortie en fonction du sens de l'échange donné par le coupleur, l'adresse mémoire octet étant elle aussi donnée par le coupleur.

Initialisation :

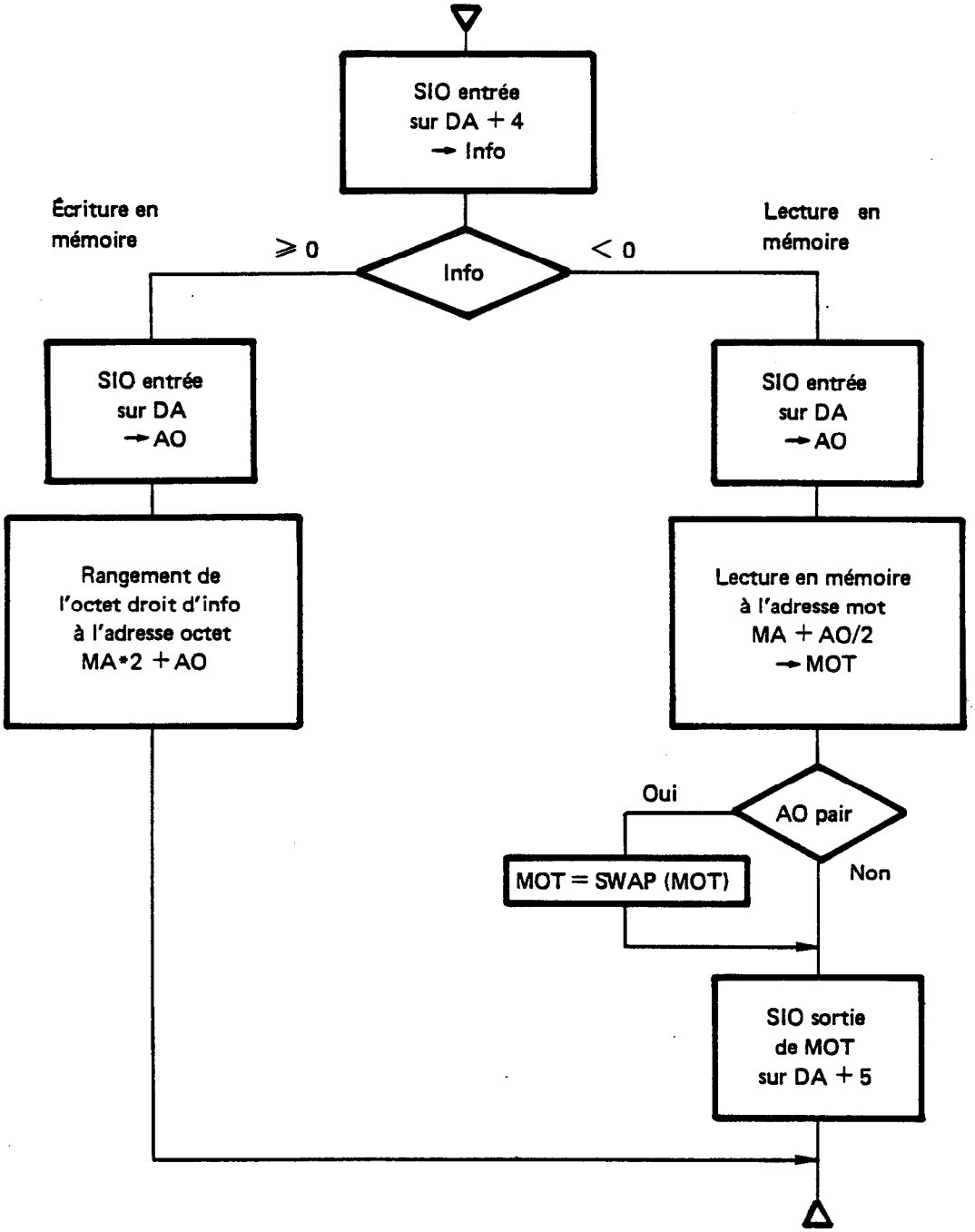
Elle se fait comme pour un canal standard par communication d'un CCB qui comporte les différences suivantes :
- CCB0 : SP = 1, BY = 1
- MAE, MA Adresse du buffer de 64K octets dans lequel s'effectuent les échanges.
- CCB3 : non utilisé
- CCB5 : adresse du microprogramme spécifique soit :
  '0DFC pour 30 et 35
  '07EB pour 70
  '0500 pour IOP16M

Libération :

La libération ne sert qu'à inhiber le canal HDC et ne fournit aucune information utile.

Entretien :

A chaque interruption le microprogramme effectue la séquence suivante :



DA = Adresse périphérique
MA = Adresse buffer 32 Kmots
AO = Adresse octet donnée par le coupleur

7 - MISE AU POINT (mode DEBUG)

7 - MISE AU POINT (mode DEBUG)

7.1 - PRINCIPES GENERAUX

Outre les processeurs de mise au point utilisables en conversationnel ou en mode train de travaux (AID, DRIP16, ...), les unités centrales SOLAR donnent à l'utilisateur les outils de mise au point suivants :

- visualisation et chargement des mots mémoire,
  " " des registres programmables,
- exécution en pas à pas,
- points d'arrêts sur instruction,
- " " sur opérandes,
- récupération des alarmes de programmation

Ce type de mise au point est accessible, soit au pupitre opérateur (POP), soit par programme à l'aide des instructions spécialisées.

Dans les deux cas, on utilise le mode "DEBUG" des processeurs pour définir les points d'arrêt.

Un processeur est dans le mode "DEBUG" dès que le bit 8 de son registre d'état ST est à 1.
Dans cet état, les défauts de parité mémoire ne provoquent plus d'alarme, mais sont considérés comme des points d'arrêt. De plus les cycles d'écriture mémoire sont transformés en cycles d'échange au niveau du bus mémoire de sorte que les modifications d'opérandes provoquent également des points d'arrêt ; mais dans ce cas, l'imparité mémoire et donc le point d'arrêt est effacé.
Ce mécanisme n'existe sur DPX 05 que sur les modules mémoire avec bit de parité.

Dans le cas d'un point d'arrêt sur instruction, le programme est arrêté avant l'exécution de l'instruction.
Dans le cas d'un point d'arrêt sur opérande, le programme est arrêté après exécution qui accède à cet opérande.

En mode DEBUG, les processeurs sont ralentis dans un rapport allant de 1 à 3 environ selon les modèles et les instructions.

Au pupitre comme par programme, il est possible de définir simultanément plusieurs points d'arrêt sur instructions comme sur opérandes.
Par contre, les entrées-sorties en mode canal ignorent et, en entrée, effacent les points d'arrêts définis dans les buffers ; les points d'arrêts ne sont donc pas rémanents dans les zones de "swap" et d'"overlay".

7.2 - PUPITRE OPERATEUR

Les touches du pupitre DB, RB et REC (voir le manuel d'exploitation MFI) permettent de positionner et d'effacer les points d'arrêt.

La définition du premier point d'arrêt par la touche DB passe le calculateur dans l'état "DEBUG". De plus le bit POP du registre ST passe à 1, signifiant ainsi que le calculateur passera en STOP à la rencontre du premier point d'arrêt.
Le calculateur sortira normalement du mode DEBUG par utilisation de la touche REC. Le bit DEBUG peut être effacé du status par écriture de ST mais alors les points d'arrêt restants provoqueront une alarme parité mémoire.

En présence de l'option DRPS, le fonctionnement du pupitre est modifié en fonction de l'état maître ou esclave du processeur :

- en mode maître les différentes adresses affichées aux clés sont considérées comme des adresses absolues de 0 à 64 K,
- en mode esclave les adresses sont considérées comme relatives au contenu du registre SLO. De plus, elles sont comparées au registre SLE et peuvent donc provoquer une alarme protection mémoire.

Le pupitre opérateur peut être utilisé lorsque le calculateur est en marche programme (RUN) s'il n'est pas masqué ni par le bit IOM de ST ni par le bit 0 de IM.
Toutefois dans ce cas le pupitre ne permet d'accéder qu'aux adresses absolues de 0 à 64 K et ceci indépendamment de l'état instantané maître ou esclave du programme et de la présence ou non de l'option DRPS.
Dans cette utilisation, la lecture et à fortiori l'écriture des registres est généralement à proscrire puisque l'on ignore quelle tâche est active au moment précis de l'opération.

Les alarmes mémoire inexistante, protection mémoire, parité mémoire créées au pupitre par les touches de visualisations ou d'écriture, provoquent l'activation de la tâche alarme avec affichage du pointeur de cette tâche. Toutefois les alarmes parité et mémoire inexistante sont ignorées si le calculateur est en mode DEBUG.

7.3 - UTILISATION PROGRAMMEE

C'est en particulier le cas du processeur de mise au point AID.
La mise en oeuvre des points d'arrêts se fait à l'aide des instructions :

- SBP, RBP, DBP,
- SST, RST pour accéder aux bits DEBUG et POP.

De plus l'instruction STEP permet d'exécuter un programme en pas à pas en le contrôlant depuis la tâche alarme, et l'instruction ACTD permet de forcer l'exécution de la tâche alarme.

Lorsque le calculateur est en mode DEBUG avec le bit POP de ST à 0, les erreurs de parité sont interprétées comme des points d'arrêt programmés et provoquent l'alarme "point d'arrêt" (numéro 8) :

- avant exécution de l'instruction dans le cas d'un point d'arrêt sur instruction,
- après exécution de l'instruction dans le cas d'un point d'arrêt sur opérande.

Dans ce mode, les points d'arrêt rencontrés dans la tâche alarme sont ignorés.

7.4 - CANAUX

Les canaux intégrés à une unité centrale ignorent les défauts de parité rencontrés dans des buffers d'entrées-sorties si le calculateur est en mode DEBUG. De plus, dans le cas d'une entrée, les points d'arrêt sont effacés. II n'est donc pas possible d'utiliser des points d'arrêt dans les zones de "swap" ou d'"overlay".

Dans le cas des canaux parallèles sur IOP 16-M, il faut passer l'état DEBUG à l'IOP 16-M au moment des initialisations et des comptes rendus si l'on veut que l'IOP 16-M ignore les défauts de parité rencontrés (voir § 6).

8 - OPTION DRPS (mode esclave)

8 - OPTION DRPS (mode esclave)

8.1 - PRESENTATION GENERALE

Le dispositif DRPS (Dynamic Relocation and Protection System) est utilisable :

- en option sur le processeur 16-40
- en standard sur les processeurs 16-65, 75, 30, 35 et 70.

Sans aucune modification des performances il permet :

- d'utiliser des tailles mémoire dépassant 64K jusqu'à concurrence de :

  . 1024 K sur 65, 75 et 70
  . 512 K sur 40
  . 256 K sur 35
  . 128 K sur 30.

- d'effectuer la translation dynamique des programmes,
- de réaliser une protection efficace entre les tâches d'une application

Le mécanisme de translation et de protection utilise deux registres SLO et SLE et n'est activé que quand le processeur est en mode esclave.
Les registres SLO et SLE font partie du contexte des tâches différées exclusivement. Les tâches immédiates devant toujours s'exécuter en mode maître ne peuvent utiliser l'option DRPS.
De plus, le mode esclave interdit l'utilisation des instructions privilégiées, ainsi avec l'option DRPS, les taches en mode esclave sont protégées entre elles et ne peuvent perturber le fonctionnement du système.

8.2 - MODE ESCLAVE ET PRIVILEGIE

Ces modes sont déterminés par les bits MS (bit 0) et PR (bit 2) du registre d'état ST :

- MS = 1 mode maître
- MS = 0, PR = 0 mode esclave
- MS = 0, PR = 1 mode privilégiée

Le mode privilégié n'existe que sur SOLAR 16-70 ; pour les autres processeurs le bit PR est non significatif.
A l'initialisation le calculateur est toujours en mode maître ; par contre une tâche différée peut être crée en mode maître, esclave ou privilégiée par le jeu des bits MS et PR des contextes. De plus une tâche peut changer de mode par exécution des instructions SVC et RSV. La modification des bits MS et PR de ST par les instructions RST et SST est à prohiber sauf cas très spéciaux.

Outre l'activation de l'option DRPS, qui est détaillée dans les paragraphes suivants, le passage en mode esclave interdit l'utilisation des instructions suivantes appelées instructions privilégiées :

- ACQ, SIO, IPI, DIT, EIT, XIMR, RDHV,
- SST, RST, RDSI, ROMB,
- ACT, WAIT, RQST, RLSE, ARM,
- SBP, RBP, DBP, STEP,
- LAR, STAR, MVTM, MVTS, ROE, WOE.
- BLA, BSTA, BXM, BDLD, BDST, BLBY, BSTBY,
- BMOVE, BRBTM, BSBTM, BDBTM, XCTX.

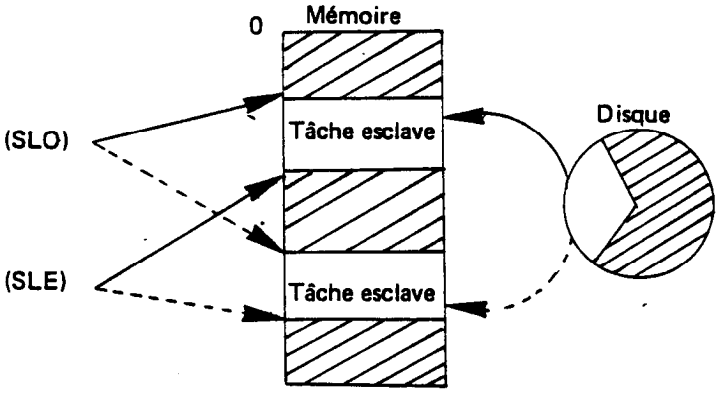
Leur exécution en mode esclave provoque l'alarme n° 4.
En l'absence de l'option DRPS, la notion de mode esclave et d'instruction privilégiée subsiste (05) mais présente peu d'intérêt, vu l'absence de protection mémoire.

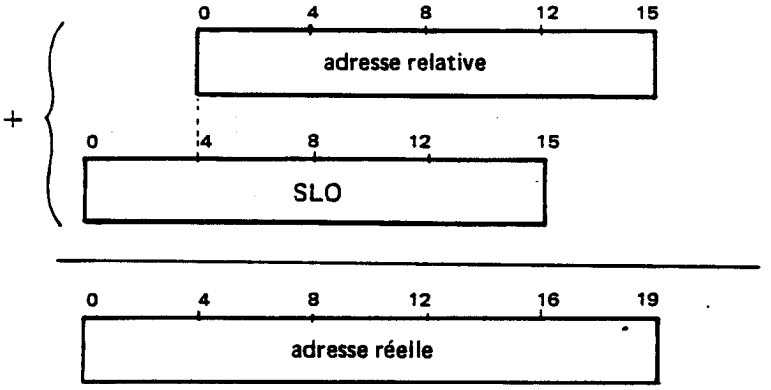
83 - TRANSLATION (SLO : Slave Origin)

Le registre SLO contient une valeur 16 bits représentant une adresse 20 bits (1 M mot) dont les 4 bits poids faible sont nuls. Ce registre définit l'adresse d'implantation où l'on a implanté la tâche associée.

Quand un programme est exécuté en mode esclave ou privilégié avec l'option DRPS, toutes les adresses utilisées par ce programme deviennent des adresses relatives à la base d'implantation représentée par le contenu du registre SLO. Un même programme en binaire absolu peut ainsi s'exécuter à toute adresse multiple de 16 de 0 à 1024K.

La tâche correspondante peut en particulier être alimentée en mémoire (swap) à des adresses différentes entre deux activations successives.





La translation d'adresse des programmes et de leurs données ne subit aucune exception au passage des adresses mémoire multiples de 64K mots.

8.4 - PROTECTION (SLE : slave end)

Le registre SLE contient une valeur 16 bits représentant une adresse 20 bits (1 M mot) dont les quatre bits poids faible sont supposés avoir la valeur 1. Cette adresse définit l'adresse du dernier mot accessible par la tâche en cours. SLO et SLE pointent donc deux blocs de 16 mots qui sont entièrement inclus dans la zone accessible.

Les valeurs contenues dans SLO et SLE doivent être telles que l'espace qu'elles définissent ne soit pas supérieur à 64 K mots. Si ce n'est pas le cas SLE est modifié en prenant la différence SLE - SLO module 64 K mots. Par contre, la zone définie par SLO et SLE peut chevaucher les frontières physiques de 64 K mots.

Une tâche en mode esclave ou privilégié sa trouve ainsi confinée dans la zone mémoire définie par les registres SLO et SLE et ne peut donc ni lire ni modifier l'espace mémoire appartenant au superviseur ou aux autres tâches de l'application.
Ce mécanisme associé aux instructions privilégiées permet d'assurer aux applications un maximum de sécurité.

Par contre rien n'interdit d'implanter plusieurs tâches esclaves ou privilégiés à la suite les unes des autres dans un même espace translatable (même couple SLO, SLE) par exemple pour résoudre des problèmes de communication.

8.5 - COMMUNICATION

8.5.1 - ESCLAVE OU PRIVILEGIE VERS SUPERVISEUR

Une tâche esclave ou privilégiée ne peut communiquer avec le superviseur que par l'utilisation de l'instruction SVC. Cette instruction permet d'accéder aux services du superviseur par une interface unique entièrement contrôlée par le système.
La plupart des services systèmes étant réentrant, l'utilisation de la pile pointée par K doit rester réentrante, c'est la raison pour laquelle après une instruction SVC les accès à cette pile par les instructions PLR, PSR, BSR, RSR, SVC, RSV se font toujours relativement à la base SLO.
Cet état maître spécial est mémorisé dans le mot d'état ST et fait partie du contexte des tâches ; c'est le bit SVCS (bit 1 de ST). Les différents états possibles sont :

ST_0	ST_1	
1	0	état maître normal (initialisation)
1	1	état maître après une instruction SVC
0	0	état esclave ou privilégié

Le quatrième état est interdit et une tâche ne doit pas être initialisée (par sa PST) dans cet état.

8.5.2 - SUPERVISEUR VERS ESCLAVE OU PRIVILÉGIÉ

Outre l'accès à la pile K de l'appelant (voir § précédent) qui permet de sauvegarder la réentrance des sous-programmes, d'imbriquer des appels au superviseur, et éventuellement de passer des paramètres, les instructions privilégiées suivantes permettent de réactiver la translation et la protection associées aux registres SLO, SLE pour accéder ou modifier des mots OU des zones de mémoire appartenant à la tâche esclave ou privilégiée appelante :

- LAR, STAR, MVTM, MVTS.

II en est de même pour les instructions liées à la mise au point :

- SBP, RBP, DBP.

8.5.3 - ESCLAVE VERS ESCLAVE

Par définition du mécanisme de protection, cette communication est impossible.

Toutefois plusieurs tâches peuvent avoir en commun tout ou partie de leur espace mémoire ; elles communiqueront alors entre elles sans mettre en danger le reste du système et en restant globalement translatables.

8.5.4 - SUPERVISEUR OU PRIVILEGIE VERS PRIVILEGIE OU ESCLAVE

Les instructions basées de l'option ISP16 (BLA, BSTA, BXM, BDLD, BDST, BLBY, BSTBY, BMOVE, BRBTM, BSBTM, BDBTM) permettent la communication d'une tâche superviseur ou privilégiée vers une tâche privilégiée ou esclave, indépendamment des registres SLO et SLE.

Enfin les instructions :

- RCDA et WCDA de l'option CDA16,
- CLA, CSTA, CXM, CDLD, CDST, CLBY, CSTBY de l'option ISP16

permettent à toutes les tâches esclaves, privilégiées ou maître d'accéder à un segment de données communes défini par le système.

Bull

extension de l'adressage à 1024 K mots impose de définir les adresses des buffers d'entrées-sorties sur 20 bits.

En mode programmé il faut donc réactiver le mécanisme DRPS par l'utilisation des instructions spéciales (LAR, STAR, MVTM, MVTS) après avoir rechargé les registres SLO, SLE par les valeurs associées au buffer.

En mode canal, l'adresse du buffer doit être calculée sur 20 bits et passée sous cette forme au canal par l'intermédiaire de la table d'échange (CCB : voir chapitre entrées-sorties). Dans ce mode les buffers d'entrées-sorties ne doivent pas chevaucher les frontières physiques de 64 K mots dans la mémoire.

Dans ce dernier cas en particulier les tests de validité des paramètres de l'échange (en particulier adresse et taille du buffer) doivent être faits par le système avant le lancement de l'échange.

II va de soi que le moniteur d'entrées-sorties IOCS prend de lui-même en charge toutes ces opérations de translation et de contrôle.

9 - OPTION MTS 16 : scheduler et sémaphores

9. - OPTION MTS 16 : Scheduler et Sémaphores

9.1 - TACHE DIFFEREE

9.1.1 - NOTION

A la suite de sollicitations extérieures (interruptions), le calculateur doit exécuter un certain nombre d'actions qui peuvent être de 2 types :

- les actions immédiates qui doivent être exécutées très rapidement,
- les actions différées dont le temps de réponse peut être plus long.

Les actions immédiates sont confiées aux tâches immédiates (cf. chapitre 5) et donc directement liées et hiérarchisées par le système d'interruption.

Les actions différées ne pouvant être exécutées simultanément par le calculateur, il est nécessaire :

- de les répartir en tâches nommées "tâches différée"
- de les hiérarchiser par un indice de priorité tâche différée,
- de disposer d'un système d'exploitation qui, en fonction de l'état du système et des événements extérieurs, gère cet ensemble de tâches.

Les microprogrammes spécialisés qui constituent l'option MTS16 facilitent l'organisation d'un programme en tâches immédiates et différées et fournissent les fonctions principales d'un moniteur.

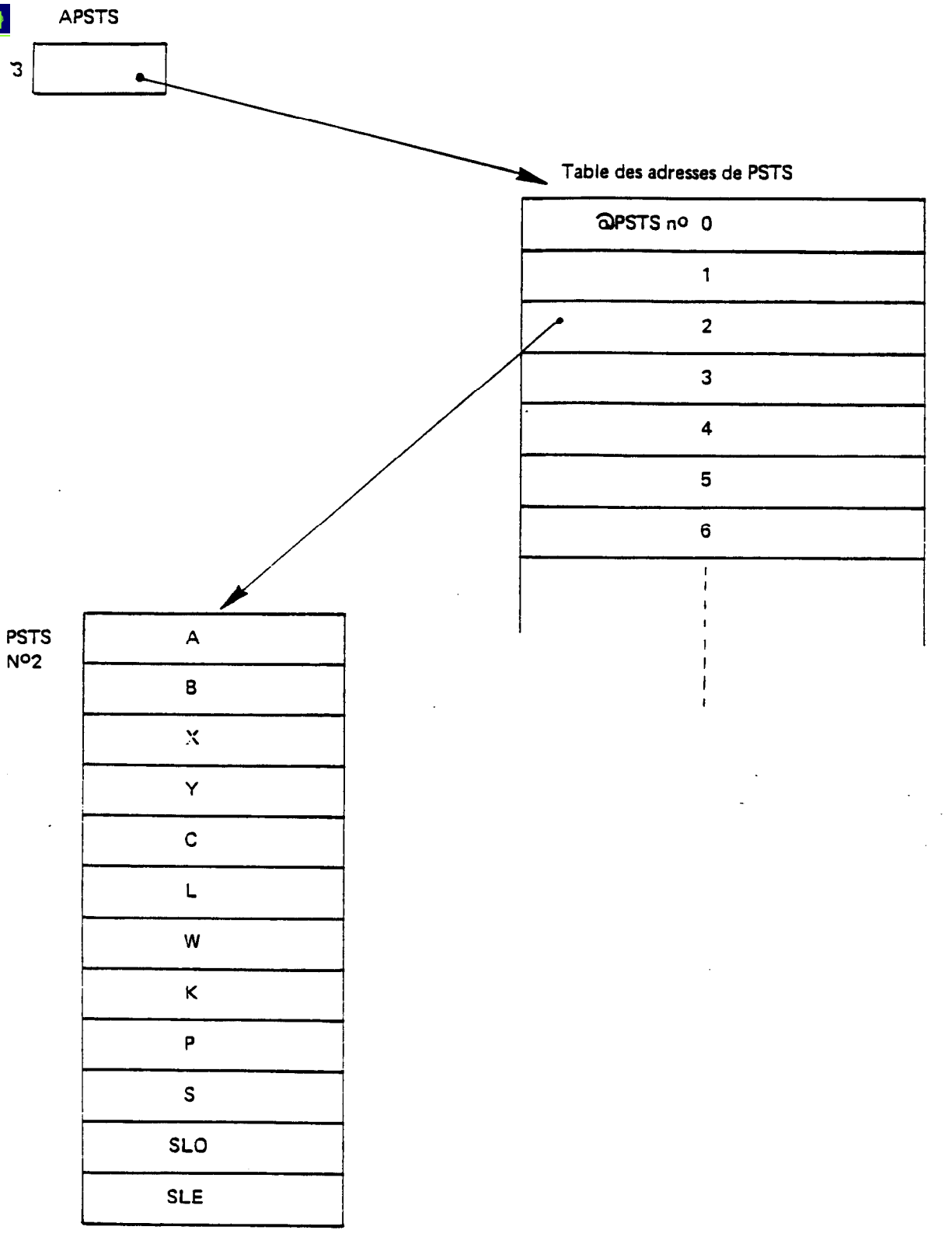
L'option MTS16 permet de gérer jusqu'à 128 niveaux de priorité pour les tâches différées et offre des primitives de gestion de ressources et d'événements (par sémaphores).

9.1.2 - Contexte (PSTS)

Physiquement une tâche immédiate est déterminée par son numéro de priorité et son contexte.
Dans le cas des tâches différées ce contexte est qualifié de complet car il inclut tous les registres programmables :
A, B, X, Y, C, L, W, K, P, S, SLO, SLE.

A chacune des tâches différées est associée une zone de 12 mémoires consécutives appelée PSTS (program Status Table différée) ; cette zone sert à conserver le contexte de la tâche correspondante lorsqu'elle n'est pas active.
A chaque fois que l'on passe le contrôle d'une tâche différée à une autre, quelle que soit la tâche abandonnée, quelle que soit la tâche lancée, et quelle que soit la raison du changement de tâche, les registres sont rangés dans la PSTS de la tâche abandonnée puis rechargés par le contenu de la PSTS de la tâche lancée. Pour que les changements de contextes puissent se faire correctement il est nécessaire qu'il existe une PSTS par niveau de priorité différée utilisé. Par contre lorsque l'on passe d'une tâche différée à une tâcheimmédiate le contenu de la PSTS de la tâche quittée est en général sans signification de même si l'on examine par programme ou au pupitre, le contenu de la PSTS de la tâche en cours d'exécution.

L'organisation suivante permet à l'unité centrale de gérer les différents PSTS :



Les relais représentés ci-dessus permettent d'implanter librement les PSTS et la table de leurs relais dans les premiers 64 K mots de la mémoire.

Lorsque l'option DRPS n'existe pas sur le calculateur, il reste nécessaire de réserver 12 mots par PSTS malgré l'inexistance des registres SLO et SLE.

9.2 - SCHEDULER

Lorsque l'option scheduler existe sur le calculateur celui-ci prend en charge non seulement le lancement et la suspension des tâches immédiates en fonction des demandes d'interruptions, mais aussi d'une manière analogue le lancement et la suspension des tâches différées en fonction du déroulement du programme. C'est le calculateur qui assure en permanence le choix de la tâche différée active, déclenche automatiquement les changements de contexte nécessaires et tient à jour dans la mémoire débanalisée NS (adresse '0000) le numéro de cette tâche.

9.2.1 - HIERARCHIE DES TACHES DIFFEREES

L'ensemble des tâches différées a une priorité inférieure à l'ensemble des tâches immédiates, et une tâche différée peut se dérouler que s'il n'y a plus d'interruption non masquée en attente, ni de tâche immédiate en cours ou suspendue.

Une tâche différée ne peut donc se dérouler que si tous les bits du registre HV sont à zéro. Le choix de la tâche différée active est fait en fonction de trois éléments :

1) Une tâche ne peut être lancée que si elle a été armée par programme.

   C'est la file ASTF qui indique en permanence l'état armé ou non armé de chacune des tâches différées. Cette file occupe 8 mots consécutifs de la zone mémoire débanalisée à partir de l'adresse '20. le bit de rang n de cette file (avec $0 \leqslant n \leqslant 127$) est à 1 lorsque la tâche de niveau n est armée.

   Une tâche différée est armée soit par le chargement initial de la file ASTF, soit par l'instruction ARM qui met à 1 le bit de la file ASTF correspondant au numéro de la tâche indiqué dans l'instruction.

   Une tâche différée se désarme grâce à l'instruction QUIT : cette instruction permet à une tâche de se terminer en remettant à 0 le bit de la file ASTF correspondant à son niveau.

2) Une tâche ne peut être lancée que si elle n'a pas provoqué sa propre mise en attente en demandant par l'intermédiaire d'un sémaphore l'attribution d'une ressource non disponible ou sa synchronisation sur signal envoyé par une autre tâche.

   C'est la file ESTF, par l'intermédiaire de laquelle l'état des sémaphores est répercuté sur le scheduler, qui indique en permanence l'état masqué ou non masqué de chacune des tâches différées. Cette file occupe 8 mots consécutifs de la zone mémoire débanalisée à partir de l'adresse `28. Le bit de rang n de cette file (avec $0 \leqslant n \leqslant 127$) est à 1 lorsque la tâche n n'est pas masquée.

   Dans le cas général la file ESTF est initialisée à 1 de manière à ce que toutes les tâches différées soient non masquées dans l'état initial du programme.

   Une tâche est masquée lorsqu'elle exécute une instruction RQST ou une instruction WAIT portant sur un sémaphore dont l'état est tel qu'il entraîne alors la suspension de la tâche.

   Une tâche différée est démasquée lorsque l'état de ce même semaphore, modifié par une autre tâche exécutant une instruction RLSE ou ACT, ne nécessite plus la suspension de la tâche.

3) Chacune des 128 tâches différées possibles a un niveau de priorité allant de 0 à 127 dans l'ordre décroissant des priorités, qui sert à identifier cette tâche et correspond au rang de l'adresse de sa PST dans la table des adresses de PST différées ainsi qu'au rang des bits qui lui correspondent dans les files ASTF, ESTF et RSTF.

   Parmi les tâches armées et non masquées, la tâche active est toujours la tâche la plus prioritaire. Le rôle de scheduler est précisément de déterminer quelle est cette tâche et de provoquer les changements de contexte nécessaires. Il intervient à chaque fois que l'état d'une tâche différée est modifié. c'est-à-dire, après l'exécution des instructions ARM, QUIT, des instructions portant sur les sémaphores (RQST, RLSE, WAIT, ACT) et à chaque fois que les tâches immédiates sont abandonnées (ACQ). De plus, une tâche différée peut activer le scheduler en exécutant l'instruction ACQ.

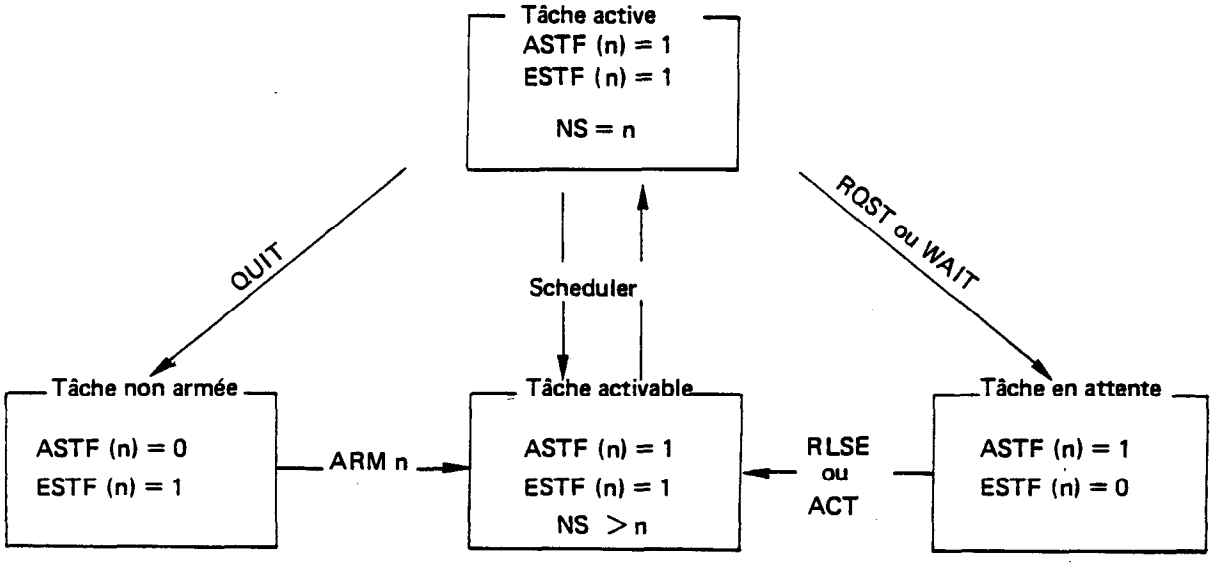
Pour que le scheduler puisse fonctionner correctement il est nécessaire qu'il y ait toujours une tâche différée en état d'être lancée (comportant ASTF = 1, ESTF = 1, RSTF = 1).

La manière la plus simple et la plus sûre de remplir cette condition est d'inclure dans chaque programme une tâche différée ayant la plus faible priorité et ayant pour fonction d'attendre qu'une autre tâche soit en état d'être lancée. Une telle tâche (appelée tâche "idle") pourra se borner aux deux instructions suivantes :

```
HALT
JPM      $ - 1
```

Les différents états d'une tâche peuvent être résumés dans le schéma ci-dessous, à propos duquel on notera que :

- une tâche différée est toujours armée par une autre tâche immédiate ou différée (ARM),
- une tâche différée est toujours démasquée par une autre tâche immédiate ou différée (RLSE OU ACT portant sur le semaphore qui a entraîné le masquage).
- une tâche différée est toujours désarmée par elle même (QUIT)
- une tâche différée est toujours masquée par elle même (RQST ou WAIT portant sur un sémaphore).



Pour ne pas compliquer inutilement ce schéma on n'y a pas fait intervenir la file RSTF

9.2.2 - FILE RSTF, TACHE DIFFEREE ZÉRO

Il est possible de compléter et éventuellement d'altérer le fonctionnement du scheduler en faisant jouer à la tâche différée zéro (tâche différée la plus prioritaire) un rôle particulier.

On utilise pour cela la file RSTF qui occupe 8 mots consécutifs de la zone mémoire débanalisée à partir de l'adresse '0030 et indique les tâches différées pour lesquelles la tâche zéro doit intervenir.

Si le bit de rang n de cette file ($0 \leqslant n \leqslant 127$) est à 1 la tâche n correspondante est lancée normalement lorsqu'elle devient la plus prioritaire des tâches différées en attente.

Par contre si ce bit est à 0, c'est la tâche différée zéro qui est lancée à la place de la tâche n, de la manière suivante :

- la tâche zéro est armée (le bit zéro de la file ASTF est mis à 1),
- les registres sont chargés par le contenu de la PSTS de la tâche zéro,
- la mémoire NS (numéro de la tâche différée en cours, adresse '0000) est chargée par zéro,
- la mémoire NL (mémoire débanalisée, adresse '0001) est chargée par le numéro n de la tâche qui aurait dû être lancée.

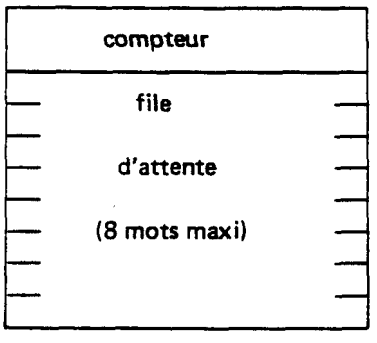
- Dans cette utilisation de la tâche zéro les bits correspondants à cette tâche dans les files ASTF, ESTF, et RSTF doivent être initialisés de la manière suivante :

  bit 0 de ASTF initialisé à zéro (non armée)
  bit 0 de ESTF initialisé à un (non masquée)
  bit 0 de RSTF initialisé à un

- Si tous les bits de la file RSTF sont à 1 le fonctionnement du scheduler n'est pas modifié et la tâche différée zéro ne joue aucun rôle particulier.

9.3 - SEMAPHORES D'EXCLUSION MUTUELLE : RQST, RLSE

Dans un contexte de multitraitement, plusieurs tâches différées indépendantes peuvent entrer en concurrence parce qu'elles doivent s'approprier des périphériques, des buffers ou des zones de mémoire libres, modifier des données communes ou des fichiers communs, appeler des sous-programmes non réentrants, etc... D'une manière générale on considère qu'il s'agit de gérer les accès à des ressources, un seul accès (ex : un sous-programme non réentrant) ou plusieurs accès (ex : un pool de buffers) étant possibles simultanément pour une même ressource. Pour cela on attache à une ressource donnée un sémaphore d'exclusion mutuelle constitué par un compteur et une file d'attente placée en mémoire à une adresse quelconque.



Le compteur occupe le premier mot du sémaphore, la file qui comporte au plus 128 bits occupe au maximum les 8 mots suivants.

Quand le compteur est positif ou nul, sa valeur indique le nombre d'accès disponibles à la ressource. Dans ce cas tous les bits de la file sont à zéro.

Quand le compteur est négatif il n'y a plus d'accès disponibles à la ressource gérée par le sémaphore et sa valeur absolue indique le nombre de tâches qui attendent un accès à cette ressource.

Dans ce cas la file mémorise le niveau de chacune de ces tâches : le bit de rang n (avec $0 \leqslant n \leqslant 127$) est à 1lorsque la tâche de niveau n est en attente.

Le compteur doit être initialisé par le nombre d'accès possibles simultanément pour la ressource à laquelle il est attaché, et la file d'attente doit être initialisée à zéro. Le nombre de mots réservés pour la file du sémaphore doit correspondre au niveau de la tâche la moins prioritaire utilisant le sémaphore.

Une tâche différée désirant un accès à une ressource exécute pour l'obtenir une instruction RQST portant sur le sémaphore attaché à cette ressource, puis une instruction RLSE portant sur le même sémaphore lorsque l'accès à la ressource ne lui est plus nécessaire.

Lors de l'instruction RQST deux cas peuvent se présenter suivant qu'un accès est disponible (compteur du sémaphore > 0) ou n'est pas disponible (compteur du sémaphore ≤ 0) :

- Si l'accès est disponible la tâche se poursuit normalement après l'instruction RQST, le compteur étant diminué de un (un accès de moins est disponible). Lors de l'instruction RLSE signalant que l'accès à la ressource n'est plus nécessaire à la tâche, le compteur est augmenté de un (un accès de plus est disponible).

- Si l'accès n'est pas disponible la tâche est mise en attente : le bit correspondant à la tâche dans la file d'attente du sémaphore est mis à 1 et le compteur est diminué de 1 (une tâche de plus dans la file d'attente du sémaphore), la tâche est masquée (le bit correspondant à la tâche dans la file ESTF est mis à 0), le contexte de la tâche est sauvegardé dans sa PST et une tâche moins prioritaire est lancée.

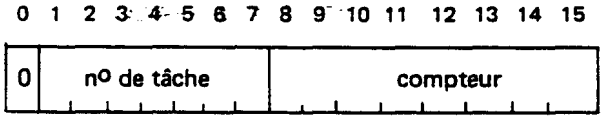
La tâche demeure en attente jusqu'à ce qu'elle soit la plus prioritaire des tâches en attente d'un accès à la ressource (elle correspond alors au bit à 1 le plus à gauche dans la file d'attente du sémaphore) et qu'une autre tâche exécute une instruction RLSE sur ce sémaphore. L'accès à la ressource libéré par l'instruction RLSE est alors attribuée à cette tâche et provoque la remise à 0 du bit correspondant à la tâche dans la file du sémaphore, l'addition de un au compteur du sémaphore (une tâche de moins dans la file d'attente), le démasquage de la tâche en attente (le bit correspondant à la tâche est mis à 1 dans la file ESTF) et, dans le cas où la tâche qui a exécuté RLSE est moins prioritaire un changement de contexte immédiat au profit de la tâche en attente.

9.4 • SEMAPHORES DE SYNCHRONISATION (PRIVES SANS PARAMETRES) : ACT, WAIT

Ce type de sémaphore permet à une tâche différée de se synchroniser sur une autre tâche T, immédiate ou différée c'est-à-dire de provoquer sa propre mise en attente si le signal de synchronisation n'a pas été envoyé par la tâche T.

Pour cela on utilise un sémaphore constitué par un seul mot placé en mémoire à une adresse quelconque.

Ce sémaphore se distingue des sémaphores privés avec file paramètre (voir paragraphe suivant) par le fait que le bit 0 a toujours la valeur zéro.



Le compteur doit être initialisé à 0, il vaut + 1 lorsque le signal de synchronisation arrive avant la demande de synchronisation il vaut - 1 lorsque la demande de synchronisation arrive avant le signal de synchronisation.
Le compteur ne doit jamais prendre une autre valeur que - 1,0 ou + 1.

Les bits 1 à 7 du sémaphore servent à mémoriser le numéro de la tâche qui est en attente du signal de synchronisation.

A un instant donné il ne peut y avoir qu'une seule tâche différée en attente pour un sémaphore donné, mais le n° de cette tâche est quelconque et n'a pas à être connu au moment de la définition du sémaphore.

Une tâche différée désirant se synchroniser sur une autre tâche T (immédiate ou différée) exécute une instruction WAIT portant sur un sémaphore. La tâche T effectue une instruction ACT portant sur le même sémaphore pour envoyer le signal de synchronisation.

Deux cas peuvent se présenter suivant que lors de l'instruction WAIT le signal de synchronisation a déjà été envoyé (compteur = 1) ou n'a pas encore été envoyé (compteur = 0) :

- Si le signal a déjà été envoyé la tâche se poursuit normalement après l'instruction WAIT, le compteur étant diminué de un et reprenant alors l'état initial 0.

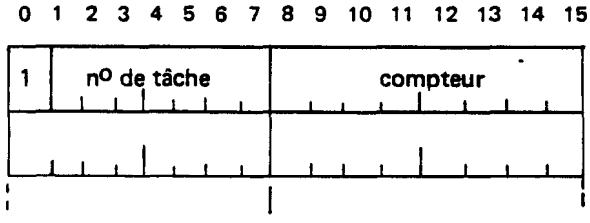
- Si le signal n'a pas été envoyé la tâche est mise en attente : le numéro de cette tâche est mémorisé dans les bits 1 à 7 du sémaphore, le compteur étant diminué de un prend la valeur - 1, la tâche est masquée (le bit correspondant à la tâche dans la file ESTF est mis à 0), le contexte est sauvegardé et une tâche moins prioritaire est lancée.
  La tâche demeure en attente jusqu'à ce que le signal de synchronisation soit envoyé par l'instruction ACT.
  Cette instruction ACT provoque : l'addition de un au compteur du sémaphore qui reprend alors l'état initial 0, le démasquage de la tâche en attente (le bit correspondant est mis à 1 dans la file ESTF) et, dans le cas où la tâche qui a exécuté ACT est moins prioritaire, un changement de contexte immédiat au profit de la tâche en attente.

  Un même sémaphore ne peut servir à plusieurs tâches se synchronisant sur une tâche T donnée que si son utilisation est régie par un semaphore d'exclusion mutuelle (le sémaphore de synchronisation constitue alors la ressource gérée par le semaphore d'exclusion mutuelle).

9.5 - SEMAPHORES D'APPELS (PRIVES AVEC PARAMETRES) : ACT, WAIT

Ce type de sémaphores permet l'appel par des tâches immédiates ou différées d'une tâche différée faisant appel à des programmes non réentrants : il permet à ce programme de servir les demandes qui lui sont faites les unes après les autres (et non pas simultanément comme un programme réentrant qui peut être interrompu et réappelé par plusieurs tâches). Il permet la transmission de paramètres entre les tâches appelantes et la tâche appelée.

Le sémaphore placé en mémoire à une adresse quelconque, est associé à la tâche appelée. Il est constitué par un premier mot comportant un compteur et le numéro de la tâche appelée, suivi d'une file occupant un nombre variable de mots et servant à la transmission des paramètres.



Il se distingue des sémaphores de synchronisation (sémaphores privés sans paramètre) par le fait que le bit 0 a toujours la valeur un.

Le compteur doit être initialisé à - 1, il vaut 0, 1, 2,... n - 1 lorsqu'il y a 1, 2, 3,... n demandes d'exécution non encore satisfaites.

Les bits 1 à 7 doivent être initialisés par le numéro de la tâche à laquelle est attachée le sémaphore.

La file-paramètre comporte un nombre de mots fixés par le programmeur. Elle doit être initialisée à zéro.
Un bit de cette file est mis à 1 à chacun des appels de la tâche, le rang de ce bit n'étant pas égal au numéro de la tâche appelante mais égal à la valeur du registre Y au moment de l'instruction d'appel (ACT). Une même tâche peut ainsi effectuer plusieurs appels différents sur le même sémaphore avec des valeurs de Y différentes.

La position de ce bit dans la file et sa signification dépendent de la convention choisie par le programmeur.
Cette position pouvant par exemple correspondre au rang qu'occupent dans une table les paramètres transmis à la tâche appelée.

Une tâche gérée par un sémaphore d'appel doit être initialisée avec le bit de la file ASTF qui lui correspond à 1 (tâche armée) et le bit de la file ESTF qui lui correspond à 0 (tâche masquée).

Une tâche immédiate ou différée désirant lancer une tâche gérée par un sémaphore d'appel exécute une instruction ACT portant sur ce sémaphore après avoir chargé le registre Y. Le compteur du sémaphore est augmenté de un, et le bit de la file-paramètre correspondant à la valeur de Y est mis à 1.

Deux cas peuvent se présenter suivant que lors de l'instruction ACT il y a des demandes antérieures d'exécution **non satisfaites (compteur ⩾ 0) ou qu'il n'y en a pas (compteur < 0) :**

- S'il n'y a pas de demande antérieure d'exécution non satisfaite, la tâche appelée est démasquée (le bit correspondant à cette tâche dans la file ESTF est mis à 1) et dans le cas où cette tâche est plus prioritaire que la tâche appelante, un changement de contexte immédiat est effectué à son profit.

- S'il y a des demandes antérieures d'exécution non satisfaites, la tâche appelante se poursuit normalement après l'instruction ACT.

  Lorsque la tâche appelée a satisfait une demande d'exécution elle exécute une instruction WAIT après avoir retiré le bit correspondant de la file paramètre. Cette instruction WAIT diminue de un le compteur du sémaphore :

  - Si le compteur devient < 0 (il n'y a plus de demande d'exécution en attente) la tâche est masquée (le bit correspondant à la tâche dans la file ESTF est mis à 0) et une autre tâche est lancée.

  - **Si le compteur reste ⩾ 0 (il reste des demandes d'exécution en attente) la tâche n'est pas masquée mais se déroule** à nouveau pour satisfaire une de ces demandes.

    On notera que l'instruction WAIT ne prend pas en compte et ne modifie pas l'état de la file-paramètre : c'est à la tâche appelée de déterminer l'ordre dans lequel sont servies les demandes et de remettre à 0 les bits de cette file. La séquence qui remet à 0 les bits de cette file ne doit pas être interrompue. Elle doit se protéger soit en masquant toutes les interruptions par une instruction DIT, soit en utilisant un sémaphore d'exclusion mutuelle.

    Une instruction WAIT portant sur le sémaphore d'appel ne peut pas être utilisée par la tâche appelante pour attendre la fin de l'exécution qu'elle a demandé (si cette attente est nécessaire on utilisera un sémaphore de synchronisation passé en paramètre).

Remarque :

Dans le cas où l'on veut gérer les appels à une tâche différée par sémaphore sans transmettre de paramètres à cette tâche on peut utiliser un sémaphore de synchronisation (sans file-paramètre) et les instructions ACT et WAIT comme il est indiqué ci-dessus.

CODES INSTRUCTIONS PAR ORDRE NUMERIQUE (hexadécimal)

0

00 —	JMP	
01 —	JNC	JGE
02 —	JNV	JNE
03 —	JNCV	JG
04 —	NOP	
05 —	JC	JL
06 —	JV	JE
07 —	JCV	JLE
08 —	ADRI	A
09 —	ADRI	B
0A —	ADRI	X
0B —	ADRI	Y
0C —	ADRI	C
0D —	ADRI	L
0E —	ADRI	W
0F —	ADRI	K

1

10 —	LAI	+ n
11 —	LXI	+ n
12 —	ORI	+ n
13 —	ANDI	+ n
14 —	EORI	+ n
15 —	CPI	+ n
16 —	LYI	+ n
17 —	LBI	+ n
18 —	JIX	
19 —	JDX	
1A —	PSR	
1B —	PLR	
1C —	SVC	
1D —	LRM	
1E 00	ACQ	
1E 01	HALT	
1E 02	RSR	
1E 03	DBT	
1E 04	STEP	
1E 05	ACTD	
1E 06	QUIT	
1E 07	PTY	
1E 08	SCY	
1E 09	MOVE	
1E 0A	SBS	
1E 0B	PUSH	
1E 0C	PULL	
1E 0D	ACK	
1E 0E	RSV	
1E 0F	ROMB	
1E 10	DBP	
1E 11	SBP	
1E 12	RBP	
1E 13	DIT	
1E 14	EIT	
1E 15	SST	
1E 16	RST	
1E 17	RDHV	
1E 18	IPI	
1E 19	RDSI	
1E 1A	LAR	
1E 1B	RDOE	
1E 1C	MVTM	
1E 1D	STAR	
1E 1E	WOE	
1E 1F	MVTS	
1F —	ARM	

2

20 —	JMP	
21 —	JAGE	
22 —	JANE	
23 —	JAG	
24 —	NOP	
25 —	JAL	
26 —	JAE	
27 —	JALE	
28 0—	RBT	n
28 1—	RBT	16 + n
28 2—	RBT	X, n
28 3—	RBT	X, 16 + n
28 4—	SBT	n
28 5—	SBT	16 + n
28 6—	SBT	X, n
28 7—	SBT	X, 16 + n
28 8—	IBT	n
28 9—	IBT	16 + n
28 A—	IBT	X, n
28 B—	IBT	X, 16 + n
28 C—	TBT	n
28 D—	TBT	16 + n
28 E—	TBT	X, n
28 F—	TBT	X, 16 + n
29 0—	SLRS	n
29 1—	SLRS	16 + n
29 2—	SLRS	X, n
29 3—	SLRS	X, 16 + n
29 4—	SLRD	n
29 5—	SLRD	16 + n
29 6—	SLRD	X, n
29 7—	SLRD	X, 16 + n
29 8—	SLLS	n
29 9—	SLLS	16 + n
29 A—	SLLS	X, n
29 B—	SLLS	X, 16 + n
29 C—	SLLD	n
29 D—	SLLD	16 + n
29 E—	SLLD	X, n
29 F—	SLLD	X, 16 + n
2A 0—	SCRS	n
2A 1—	SCRS	16 + n
2A 2—	SCRS	X, n
2A 3—	SCRS	X, 16 + n
2A 4—	SCRD	n
2A 5—	SCRD	16 + n
2A 6—	SCRD	X, n
2A 7—	SCRD	X, 16 + n
2A 8—	SCLS	n
2A 9—	SCLS	16 + n
2A A—	X, n	
2A B—	X, 16 + n	
2A C—	SCLD	n
2A D—	SCLD	16 + n
2A E—	SCLD	X, n
2A F—	SCLD	X, 16 + n
2B 0—	SARS	n
2B 1—	SARS	16 + n
2B 2—	SARS	X, n
2B 3—	SARS	X, 16 + n
2B 4—	SARD	n
2B 5—	SARD	16 + n
2B 6—	SARD	X, n
2B 7—	SARD	X, 16 + n
2B 8—	XR	
2B 9—	XR	
2B A—	XR	
2B B—	XR	
2B C—	LR	
2B D—	LR	
2B E—	LR	
2B F—	LR	
2V 0—	ADR	
2C 1—	ADR	
2C 2—	ADR	
2C 3—	ADR	
2C 4—	SBR	
2C 5—	SBR	
2C 6—	SBR	
2C 7—	SBR	
2C 8—	ORR	
2C 9—	ORR	
2C A—	ORR	
2C B—	ORR	
2C C—	EORR	
2C D—	EORR	
2C E—	EORR	
2C F—	EORR	
2D 0—	ANDR	
2D 1—	ANDR	
2D 2—	ANDR	
2D 3—	ANDR	
2D 4—	CLSR	
2D 5—	CLSR	
2D 6—	CLSR	
2D 7—	CLSR	
2D 8—	ADCR	
2D 9—	ADCR	
2D A—	ADCR	
2D B—	ADCR	
2D C—	SBCR	
2D D—	SBCR	
2D E—	SBCR	
2D F—	SBCR	
2E 0—	NGR	
2E 1—	NGR	
2E 2—	NGR	
2E 3—	NGR	
2E 4—	CPZR	
2E 5—	CPZR	
2E 6—	CPZR	
2E 7—	CPZR	
2E 8—	CMR	
2E 9—	CMR	
2E A—	CMR	
2E B—	CMR	
2E C—	CPR	
2E D—	CPR	
2E E—	CPR	
2E F—	CPR	
2F 0—	SWBR	
2F 1—	SWBR	
2F 2—	SWBR	
2F 3—	SWBR	
2F 4—	XIMR	
2F 5—	XIMR	
2F 6—	XIMR	
2F 7—	XIMR	
2F 8—	ADRP	
2F 9—	ADRP	
2F A—	ADRP	
2F B—	ADRP	
2F C—	LRP	
2F D—	LRP	
2F E—	LRP	
2F F—	LRP	

3

30 —	LAI	— n
31 —	LXI	— n
32 —	ORI	— n
33 —	ANDI	— n
34 —	EORI	— n
35 —	CPI	— n
36 —	LYI	— n
37 —	LBI	— n

38 — extension (option FFP16)
39 — extension (" DAP 16)
3A — extension (" VSS 16)
3B — extension
3C — extension (" ISP16)
3D — extension (" FFM16)
3E — extension
3F — extension (option CDA)

base C		base L		base W		
direct	indirect	direct	indirect	direct	indirect	
4	6	8	A	C	E	
40—	60—	80—	A0—	C0—	E0—	LBY
41—	61—	81—	A1—	C1—	E1—	STBY
42—	62—	82—	A2—	C2—	E2—	CPBY
43—	63—	83—	A3—	C3—	E3—	XM
44—	64—	84—	A4—	C4—	E4—	CPZ
45—	65—	85—	A5—	C5—	E5—	BR
46—	66—	86—	A6—	C6—	E6—	BSR
47—	67—	87—	A7—	C7—	E7—	SIO
48—	68—	88—	A8—	C8—	E8—	SB
49—	69—	89—	A9—	C9—	E9—	AD
4A—	6A—	8A—	AA—	CA—	EA—	STY
4B—	6B—	8B—	AB—	CB—	EB—	STX
4C—	6C—	8C—	AC—	CC—	EC—	STB
4D—	6D—	8D—	AD—	CD—	ED—	STA
4E—	6E—	8E—	AE—	CE—	EE—	MP
4F—	6F—	8F—	AF—	CF—	EF—	DV
5	7	9	B	D	F	
50—	70—	90—	B0—	D0—	F0—	LA
51—	71—	91—	B1—	D1—	F1—	LX
52—	72—	92—	B2—	D2—	F2—	OR
53—	73—	93—	B3—	D3—	F3—	AND
54—	74—	94—	B4—	D4—	F4—	EOR
55—	75—	95—	B5—	D5—	F5—	CP
56—	76—	96—	B6—	D6—	F6—	LY
57—	77—	97—	B7—	D7—	F7—	LB
58—	78—	98—	B8—	D8—	F8—	IC
59—	79—	99—	B9—	D9—	F9—	DC
5A—	7A—	9A—	BA—	DA—	FA—	RQST
5B—	7B—	9B—	BB—	DB—	FB—	RLSE
5C—	7C—	9C—	BC—	DC—	FC—	ACT
5D—	7D—	9D—	BD—	DD—	FD—	LAD
5E—	7E—	9E—	BE—	DE—	FE—	STZ
5F—	7F—	9F—	BF—	DF—	FF—	WAIT

DECODAGE DE L'OCTET DROIT DES INSTRUCTIONS REGISTRE - REGISTRE

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0– 4– 8– C–	A, A	A,B	A, X	A, Y	A, C	A, L	A, W	A, K	B, A	B, B	B, X	B, Y	B, C	B, L	B, W	B, K
1– 5– 9– D–	X, A	X, B	X, X	X, Y	X, C	X, L	X, W	X, K	Y, A	Y, B	Y, X	Y, Y	Y, C	Y, L	Y, W	Y, K
2– 6– A– E–	C, A	C, B	C, X	C, Y	C, C	C, L	C, W	C, K	L, A	L, B	L, X	L, Y	L, C	L, L	L, W	L, K
3– 7– B– F–	W, A	W, B	W, X	W, Y	W, C	W, L	W, W	W, K	K, A	K, B	K, X	K, Y	K, C	K, L	K, W	K, K

ex : '2BC2 se décode LR A, X. On trouve A, X dans la ligne C- et la colonne - 2

CODES INSTRUCTIONS PAR ORDRE ALPHABETIQUE

Chaque instruction registre-registre renvoie à l'un des quatre tableaux suivants pour le codage de son octet droit.

(a)

registre source	registre destination A	B	X	Y	C	L	W	K
A	00	01	02	03	04	05	06	07
B	08	09	0A	0B	0C	0D	0E	0F
X	10	11	12	13	14	15	16	17
Y	18	19	1A	1B	1C	1D	1E	1F
C	20	21	22	23	24	25	26	27
L	28	29	2A	2B	2C	2D	2E	2F
W	30	31	32	33	34	35	36	37
K	38	39	3A	3B	3C	3D	3E	3F

(b)

registre source	registre destination A	B	X	Y	C	L	W	K
A	40	41	42	43	44	45	46	47
B	48	49	4A	4B	4C	4D	4E	4F
X	50	51	52	53	54	55	56	57
Y	58	59	5A	5B	5C	5D	5E	5F
C	60	61	62	83	64	65	66	67
L	68	69	6A	6B	6C	6D	6E	6F
W	70	71	72	73	74	75	76	77
K	78	79	7A	7B	7C	7D	7E	7F

(c)

registre source	registre destination A	B	X	Y	C	L	W	K
A	80	81	82	83	84	85	86	87
B	88	89	8A	8B	8C	8D	8E	8F
X	90	91	92	93	94	95	96	97
Y	98	99	9A	9B	9C	9D	9E	9F
C	A0	A1	A2	A3	A4	A5	A6	A7
L	A8	A9	AA	AB	AC	AD	AE	AF
W	B0	B1	B2	B3	B4	B5	B6	B7
K	B8	8B9	BA	BB	BC	BD	BE	BF

(d)

registre source	registre destination A	B	X	Y	C	L	W	K
A	C0	C1	C2	C3	C4	C5	C6	C7
B	C8	C9	CA	CB	CC	CD	CE	CF
X	D0	D1	D2	D3	D4	D5	D6	D7
Y	D8	D9	DA	DB	DC	DD	DE	DF
C	E0	E1	E2	E3	E4	E5	E6	E7
L	E8	E9	EA	EB	EC	ED	EE	EF
W	F0	F1	F2	F3	F4	F5	F6	F7
K	F8	F9	FA	FB	FC	FD	FE	FF

A

ACK	1E0D	
ACQ	1E00	
ACT	5C--	C
	7C--	& C
	9C--	L
	BC--	& L
	DC--	W
	FC--	& W
ACTD	1E05	
AD	49--	C
	69--	& C
	89--	L
	A9--	& L
	C9--	W
	E9--	& W
ADCR	2D(c)	
ADR	2C(a)	
ADRI	08--	A
	09--	B
	0A--	X
	0B--	Y
	0C--	C
	0D--	L
	0E--	W
	0F--	K
ADRP	2F(c)	
AND	53--	C
	73--	& C
	93--	L
	B3--	& L
	D3--	W
	F3--	& W
ANDI	13--	+ n
	33--	- n
ANDR	2D(a)	
ARM	1F--	

B

BR	45--	C
	65--	&C
	85--	L
	A5--	&L
	C5--	W
	E5--	&W
BSR	46--	C
	66--	&C
	86--	L
	A6--	&L
	C6--	W
	E6--	&W

C

CLSR	2D(b)	
CMR	2E(c)	
CP	55--	C
	75--	& C
	95--	L
	B5--	& L
	D5--	W
	F5--	& W
CPBY	42--	C
	62--	& C
	82--	L
	A2--	& L
	C2--	W
	E2--	& W
CPI	15--	+ n
	35--	- n
CPR	2E(d)	
CPZ	44--	C
	64--	& C
	84--	L

	A4——	& L
	C4——	W
	E4——	& W
CPZR	2 EC (b)	

D

DBP	1E10	
DBT	1E03	
DC	59——	C
	79——	& C
	99——	L
	B9——	& L
	D9——	W
	F9——	& W
DIT	1E13	
DV	4F——	C
	6F——	& C
	8F——	L
	AF——	& L
	CF——	W
	EF——	& W

E

EIT	1E14	
EOR	54——	C
	74——	& C
	94——	L
	B4——	& L
	D4——	W
	F4——	& W
EORI	14——	+ n
	34——	− n
EORR	2C(d)	

F

FABS	3800	
	0800	
FAD	3800	
	42——	C
	62——	& C
	82——	L
	A2——	& L
	C2——	W
	E2——	& W
FCAM	3800	
	4E——	C
	6E——	& C
	8E——	L
	AE——	& L
	CE——	W
	EE——	& W
FCAZ	3800	
	2800	
FCMZ	3800	
	4C——	C
	6C——	& C
	8C——	L
	AC——	& L
	CC——	W
	EC——	& W
FDV	3800	
	46——	C
	66——	& C
	86——	L
	A6——	& L
	C6——	W
	E6——	& W
FIX	3800	
	1000	
FLD	3800	
	40——	C
	60——	& C
	80——	L
	A0——	& L
	C0——	W
	E0——	& W
FLT	3800	
	1800	
FMP	3800	
	44——	C
	64——	& C
	84——	L
	A4——	& L
	C4——	W
	E4——	& W
FNEG	3800	
	0000	
FSB	3800	
	4A——	C
	6A——	& C
	8A——	L
	AA——	& L
	CA——	W
	EA——	& W
FST	3800	
	48——	C
	68——	& C
	88——	L
	A8——	& L
	C8——	W
	E8——	& W

H

HALT	1E01	

I

IBT	288—	n
	289—	16 + n
	28A—	X,n
	28B—	X, 16+n
IC	58——	C
	78——	& C
	98——	L
	B8——	& L
	D8——	W
	F8——	& W
IPI	1E18	

J

JAE	26——	
JAG	23——	
JAGE	21——	
JAL	25——	
JALE	27——	
JANE	22——	
JC	05——	
JCV	07——	
JDX	19——	
JE	06——	
JG	03——	
JGE	01——	
JIX	18——	
JL	05——	
JLE	07——	
JMP	00——	
JNC	01——	
JNCV	03——	
JNE	02——	
JNV	02——	
JV	06——	

L

LA	50——	C
	70——	& C
	90——	L
	B0——	& L
	D0——	W
	F0——	& W
LAD	5D——	C
	7D——	& C
	9D——	L
	BD——	& L
	DD——	W
	FD——	& W
LAI	10——	+n
	30——	− n
LAR	1E14	
LB	57——	C
	77——	& C
	97——	L
	B7——	& L
	D7——	W
	F7——	7 W
LBI	17——	+ n
	37——	− n
LBY	40——	C
	60——	& C
	80——	L
	A0——	& L
	C0——	W
	E0——	& W
LR	2B (d)	
LRM	1D——	
LRP	2F(d)	

LX	51——	C
	71——	& C
	91——	L
	B1——	& L
	D1——	W
	F1——	& W
LXI	11——	+ n
	31——	− n
LY	56——	C
	76——	& C
	96——	L
	B6——	& L
	D6——	W
	F6——	& W
LYI	16——	+ n
	36——	− n

M

MOVE	1E09	
MP	4E——	C
	6E——	& C
	8E——	L
	AE——	& L
	CE——	W
	EE——	& W
MVTM	1E1C	
MVTS	1E1F	

N

NGR	2E(a)	
NOP	04——	
NORM	3800	
	2000	

O

OR	52——	C
	72——	& C
	92——	L
	B2——	& L
	D2——	W
	F2——	& W
ORI	12——	
	32——	
ORR	2C(c)	

P

PLR	1B——	
PSR	1A——	
PTY	1E07	
PULL	1E0C	
PUSH	1E0B	

Q

QUIT	1E06	

R

RBP	1E12	
RBT	280—	n
	281—	n + 16
	282—	X,n
	283—	X, n +16
RDHV	1E17	
RDOE	1E1B	
RDSI	1E19	
RLSE	5B——	C
	7B——	& C
	9B——	L
	BB——	& L
	DB——	W
	FB——	& W
ROMB	1E0F	
RQST	5A——	C
	7A——	& C
	9A——	L
	BA——	& L
	DA——	W
	FA——	& W
RSR	1E02	
RST	1E16	
RSV	1E0E	

S

SARD	2B4—	n
	2B5—	n + 16
	2B6—	X, n
	2B7—	X, n +16
SARS	2B0—	n
	2B1—	n +16
	2B2—	X, n
	2B3—	X, n + 16
SB	48——	C
	68——	& C
	88——	L
	A8——	& L
	C8——	W
	E8——	& W
SBCR	2D(d)	
SBP	1E11	
SBR	2C(b)	
SBS	1E0A	
SBT	284—	n
	285—	n + 16
	286—	X, n
	287—	X, n + 16
SCLD	2AC—	n
	2AD—	n + 16
	2AE—	X, n
	2AF—	X, n + 16
SCLS	2A8—	n
	2A9—	n + 16
	2AA—	X, n
	2AB—	X, n + 16
SCRD	2A4—	n
	2A5—	n + 16
	2A6—	X, n
	2A7—	X, n + 16
SCRS	2A0—	n
	2A1—	n + 16
	2A2—	X, n
	2A3—	X, n + 16
SCY	1E08	
SIO	47——	C
	67——	& C
	87——	L
	A7——	& L
	C7——	W
	E7——	& W
SLLD	29C—	n
	29D—	n + 16
	29E—	X, n
	29F—	X, n + 16
SLLS	298—	n
	299—	n + 16
	29A—	X, n
	29B—	X, n + 1
SLRD	294—	n
	295—	16 + n
	296—	X, n
	297—	X, 16 +
SLRS	290—	n
	291—	16 +n
	292—	X, n
	293—	X, 16 +
SST	1E15	
STA	4D——	C
	6D——	& C
	8D——	L
	AD——	& L
	CD——	W
	ED——	& W
STAR	1E1D	
STB	4C——	C
	6C——	& C
	8C——	L
	AC——	& L
	CC——	W
	EC——	& W
STBY	41——	C
	61——	& C
	81——	L
	A1——	& L
	C1——	W
	E1——	& W
STEP	1E04	
STX	4B——	C
	6B——	& C
	8B——	L
	AB——	& L
	CB——	W
	EB——	& W
STY	4A——	C
	6A——	& C
	8A——	L
	AA——	& L

	EA——	W
	8A——	& W
STZ	5E——	C
	7E——	& C
	9E——	L
	BE——	& L
	DE——	W
	FE——	& W
SVC	1C——	
SWBR	2F(a)	

T

TBT	28C–	n
	28D–	16 +n
	28E–	X, n
	28F–	X, 16 + n

W

WAIT	5F——	C
	7F——	& C
	9F——	L
	BF——	& L
	DF——	W
	FF——	& W
WOE	1E1E	

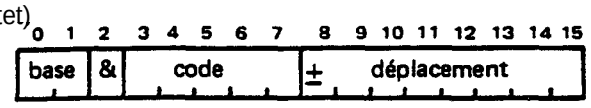
X

XIMR	2F(b)	
XM	43——	C
	63——	& C
	83——	L
	A3——	& L
	C3——	W
	E3——	& W
XR	2B(c)	

FORMAT DES INSTRUCTIONS, MODES D'ADRESSAGE

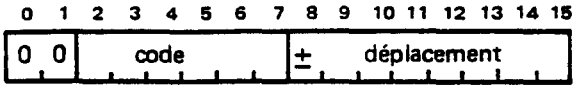
• Instructions avec opérande en mémoire (mot ou octet)

Opérandes mémoires 16 bits ou 8 bits
Déplacement compris entre - 128 et + 127
Adressage basé par rapport à C, L, W
Adressage direct, indirect, indirect index&



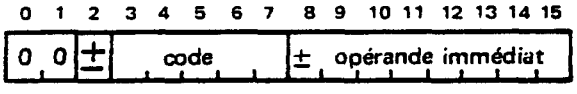
• Instructions de saut :

Adressage basé par rapport à P
Déplacement compris entre - 128 et + 127



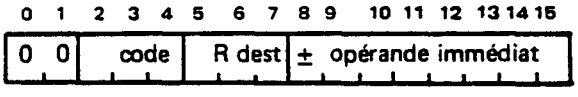
• Instructions avec opérande immédiat 9 bits :

Opérande effectif de 16 bits : les 8 bits poids forts sont identiques au bit 2 de l'instruction, les 8 bits poids faibles sont identiques aux bits 8 à 15 de l'instruction



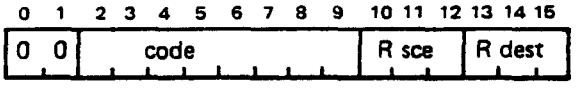
• Instruction avec opérande immédiat 8 bits :

Opérande effectif de 16 bits : les 9 bits poids forts sont identiques au bit 8 de l'instruction, les 7 bits poids faibles sont identiques aux bits 9 à 15 de l'instruction



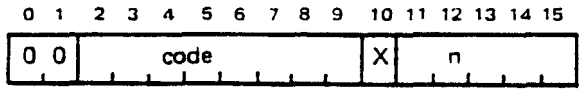
• Instructions registre-registre

Le résultat de l'opération effectuée entre les contenus des registres source et destination est rangé dans le registre destination.



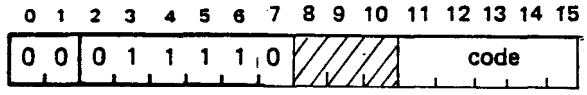
• Décalages et opérations sur bit :

Le nombre n est un nombre de pas de décalages ou un nombre de bits
Si l'instruction est indexée (bit 10 à 1) l'index est ajouté à n modulo 32



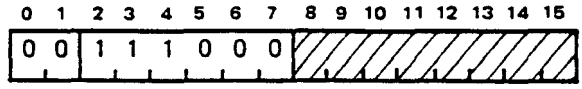
• Instructions sans indication d'opérande :

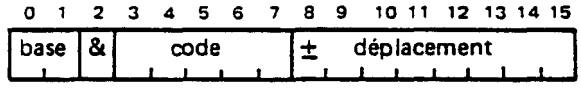
Dans une instruction de ce format les opérandes sont indiqués par le contenu de certains registres (tels que A, B, X, Y) selon l'instruction



• Instruction virgule flottante avec opérande mémoire :

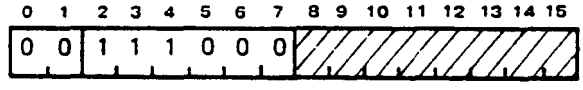
Ces instructions occupent deux mots de 16 bits
Opérande mémoire : nombre en virgule flottante sur 2 mots de 16 bits.
Déplacement compris entre - 128 et + 127
Adressage basé par rapport à C, L, W
Adressage direct, indirect, indirect indexé.

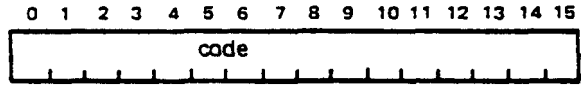




• Instruction virgule flottant sans opérande mémoire :

Ces instructions occupent deux mots de 16 bits
Opérande : nombre en virgule flottante dans les registres A et B





CHARGEMENTS

LA	MEM	Chargement de la mémoire dans le registre A adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
LB	MEM	Chargement de la mémoire dans le registre B adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
LX	MEM	Chargement de la mémoire dans le registre X adressage : direct, indirect, indirect indexe déplacement : compris entre - 128 et + 127 bases : C, L, W
LY	MEM	Chargement de la mémoire dans le registre Y adressage : direct, indirect, indirect indexe déplacement : compris entre - 128 et + 127 bases : C, L, W
LAI	± 255	Chargement de l'opérande immédiat dans le registre A opérande immédiat : compris entre - 256 et + 255
LBI	± 255	Chargement de l'opérande immédiat dans le registre B opérande immédiat : compris entre - 256 et + 255
LXI	± 255	Chargement de l'opérande immédiat dans le registre X opérande immédiat : compris entre - 256 et + 255
LYI	± 255	Chargement de l'opérande immédiat dans le registre Y opérande immédiat : compris entre - 256 et + 255
LR	Rs, Rd	Chargement du registre source dans le registre destination registres source et destination : A B X Y C L W K
LRP	Rd	Chargement du registre P dans le registre destination registre destination : A B X Y C L W K ATTENTION : P contient l'adresse de l'instruction LRP
LAD		Voir instructions diverses
LBY		Voir opérations sur octets
FLD		Voir flottant
PLR		Voir instructions diverses
PULL		Voir instructions diverses
LRM		Voir instructions diverses
LAR		Voir instructions diverses

RANGEMENTS

STA	MEM	Rangement du registre A dans la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases, : C, L, W
STB	MEM	Rangement du registre B dans la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
STX	MEM	Rangement du registre X dans la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
STY	MEM	Rangement du registre Y dans la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
STZ		Voir instructions diverses
STBY		Voir opérations sur octets
FST		Voir flottant
PSR		Voir instructions diverses
PULL		Voir instructions diverses
STAR		Voir instructions diverses

ECHANGES

XM	MEM	Echange du registre A et du mot mémoire
XR	Rs-d, R s-d	Echange du contenu des deux registres registres source destination : A B X Y C L W K
XIMR		Voir interruptions
SWBR		Voir opérations sur octets

OPERATIONS ARITHMETIQUES

additions

AD	MEM	Addition de la mémoire au registre A indicateurs : débordement (V), report (C) adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
ADRI	± 127 , Rd	Addition de l'opérande immédiat au registre destination indicateurs : débordement (V), report (C) opérande immédiat : compris entre - 128 et + 127 registre destination : A B X Y C L W K
ADR	Rs [, Rd]	Addition du registre source au registre destination indicateurs : débordement (V) report (C) registres source et destination : A B X Y C L W K
ADCR	Rd	Addition du report au registre destination indicateurs : débordement (V), report (C) registre destination : A B X Y C L W K
IC		Voir instructions diverses
ADRP		Voir instructions diverses
FAD		Voir flottant

soustractions

SB	MEM	Soustraction de la mémoire au registre A indicateurs : débordement (V), report (C) adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
SBR	Rs [, Rd]	Soustraction du registre source au registre destination indicateurs : débordement (V), report (C) registres source et destination : A B X Y C L W K
SBCR	Rd	Soustraction du report au registre destination indicateurs : débordement (V), report (C) registres source et destination : A B X Y C L W K
DC		Voir instructions diverses
FSB		Voir flottant

OPERATIONS ARITHMETIQUES (suite)

changement de signe

NGR	Rs [, Rd]	L'opposé du registre source est chargé dans le registre destination indicateurs : débordement (V), report (C) registres source et destination : A B X Y C L W K
FNEG		Voir flottant

multiplication

MP	MEM	Multiplication du registre A par la mémoire avec résultat dans les registres A (poids forts) et B (poids faibles) indicateurs : remise à zéro de V et C adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
FMP		Voir flottant

division

DV	MEM	Division des registres A et B par la mémoire avec résultat dans le registre A et reste dans le registre B indicateurs : division impossible (V) adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W ATTENTION : si le dividende est algébrique sur 16 bits charger le dividende dans A puis faire SARD 16
FDV		Voir flottant

COMPARAISONS

CP	MEM	Comparaison algébrique du registre A à la mémoire adressage : direct, indirect, indirect indexé déplacement : compris en - 128 et + 127 bases: C, L, W indicateurs : V et C sont positionnés selon le résultat de la comparaison (1)
CPI	± 255	Comparaison algébrique du registre A à l'opérande immédiat opérande immédiat : compris en - 256 et + 255 indicateurs : V et C sont positionnés selon le résultat de la comparaison (1)
CPR	Rs [, Rd]	Comparaison algébrique du second registre avec le premier registre registres : A B X Y C L W K indicateurs : V et C sont positionnés selon le résultat de la comparaison (1) ATTENTION : le 1er terme de la comparaison est le registre destination
CPZR	Rd	Comparaison algébrique du registre avec zéro registres : A B X Y C L W K indicateurs : V et C sont positionnés selon le résultat de la comparaison (1) ATTENTION : le 1er terme de la comparaison est le registre destination
CPZ	MEM	Comparaison algébrique de la mémoire à zéro adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : V et C sont positionnés salon la résultat de la comparaison (1)
CPBY		Voir opérations sur octets
FCAM		Voir flottant
FCAZ		Voir flottant
FCMZ		Voir flottant

(1)	JL	JE	JG	JGE	JNE	JLE
CP	A < mem	A = mem	A > mem	A ≥ mem	A ≠ mem	A ≤ mem
CPI	A < op. imm	A = op. imm	A > op. imm	A ≥ op. imm	A ≠ op. imm	A ≤ op. imm
CPR	2d reg < 1er reg	2d reg = 1er reg	2d reg > 1er reg	2d reg ≥ 1er reg	2d reg ≠ 1er reg	2d reg ≤ 1er reg
CPZR	reg < 0	reg = 0	reg > 0	reg ≥ 0	reg ≠ 0	reg ≤ 0
CPZ	mem < 0	mem = 0	mem > 0	mem ≥ 0	mem ≠ 0	mem ≤ 0

OPERATIONS LOGIQUES

intersections

AND	MEM	Intersection logique du registre A avec la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
ANDI	± 255	Intersection logique du registre A avec l'opérande immédiat opérande immédiat : compris entre - 256 et + 255
ANDR	Rs [, Rd]	Intersection logique du registre source avec le registre destination registres source et destination : A B X Y C L W K

unions

OR	MEM	Union logique du registre A avec la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L, W
ORI	± 255	Union logique du registre A avec l'opérande immédiat opérande immédiat : compris entre - 256 et + 255
ORR	Rs [, Rd]	Union logique du registre source avec le registre destination registres source et destination : A B X Y C L W K

disjonctions

EOR	MEM	Disjonction du registre A avec la mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
EORI	± 255	Disjonction de registre A avec l'opérande immédiat opérande immédiat : compris entre - 256 et + 255
EORR	Rs [, Rd]	Disjonction du registre source avec le registre destination registres source et destination : A B X Y C L W K

et non

CLSR	Rs [, Rd]	Remise à zéro des bits du registre destination correspondants aux bits à un du registre source registres source et destination : A B X Y C L W K

complément

CMR	**Rs**	[, **Rd**]	Chargement du registre destination par le complément logique du registre source registres source et destination : A B X Y C L W K

DÉCALAGES

logiques

SLRS	31	Décalage logique droit du registre A nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SLRD	31	Décalage logique droit des registres A et B nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SLLS	31	Décalage logique gauche du registre A nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SLLD	31	Décalage logique gauche des registres A et B nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti

circulaires

SCRS	31	Décalage circulaire droit du registre A nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SCRD	31	Décalage circulaire droit des registres A et B nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SCLS	31	Décalage circulaire gauche du registre A nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SCLD	31	Décalage circulaire gauche des registres A et B nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti

DÉCALAGES (suite)

arithmétiques

SARS	31	Décalage arithmétique droit du registre A nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti
SARD	31	Décalage arithmétique droit des registres A et B nombre de décalages : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du dernier bit sorti

OPÉRATIONS SUR BITS

RBT	31	Mise à zéro d'un bit du registre A ou du registre B numéro de bit : compris entre 0 et 31, indexation possible
SBT	31	Mise à un d'un bit du registre A ou du registre B numéro de bit : compris entre 0 et 31, indexation possible
IBT	31	Inversion d'un bit du registre A ou du registre B numéro de bit : compris entre 0 et 31, indexation possible
TBT	31	Test d'un bit du registre A ou du registre B numéro de bit : compris entre 0 et 31, indexation possible indicateurs : C prend la valeur du bit testé
DBT		Chargement du registre X par le rang du bit le plus à gauche des registres A et B indicateurs : C est mis à 1 si tous les bits de A et B sont à 0
RBTM (option scheduler 65)		Mise à zéro d'un bit dans une file de bits en mémoire. A contient l'adresse du premier mot de la file. Rang du bit de 0 à 32767 avec indexation possible
SBTM (option scheduler 65)		Mise à un d'un bit dans une file de bits en mémoire. A contient l'adresse du premier mot de la file. Rang du bit de 0 à 32767 avec indexation possible.
DRBM (option scheduler 65)		Recherche du premier bit à un dans une file de bits en mémoire. A contient l'adresse du premier mot de la file. X contient le rang initial de la recherche. Y contient le rang du bit à partir duquel la recherche sera arrêtée. L'indicateur V est remis à 0. L'indicateur C est mis à 0 si la recherche est positive sinon il est mis à 1.

OPÉRATIONS SUR OCTETS

LBY	MEM	Chargement de l'octet dans les 8 bits poids faible du registre A adressage : direct et indirect (octet gauche du mot mémoire) indirect indexé (octet gauche ou octet droit) déplacement : compris entre - 128 et + 127 bases : C, L, W ATTENTION : les 8 bits poids fort de A sont remis à zéro
STBY	MEM	Rangement des 8 bits poids faible du registre A dans l'octet mémoire adressage : direct et indirect (octet gauche du mot mémoire, indirect indexé (octet gauche ou octet droit) deplacement : compris entre - 128 et + 127 bases : C, L, W
CPBY	MEM	Comparaison algébrique du registre A et de l'octet mémoire adressage : direct et indirect (octet gauche du mot mémoire) indirect indexé (octet gauche ou octet droit) déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : V et C sont positionnés selon le résultat de la comparaison (1) ATTENTION : les 16 bits de A sont comparés aux 8 bits de l'octet
SWBR	**Rs** [, **Rd**]	Les deux octets du registre source sont intervertis et placés dans le registre destination registres source et destination : A B X Y C L W K
SBS **(option scheduler sur** (05) **)** (standard sur (40) **et** (65) **)**		Recherche d'un octet identique au registre A, dans la chaîne dont l'adresse est donnée par le registre B et la longueur par le registre Y, à partir de l'octet dont le rang est donné par le registre X **indicateurs :** C est mis à 1 quand l'octet n'a pas été trouvé **ATTENTION** : les 16 bits de A sont comparés aux 8 bits de l'octet si X > Y aucune recherche n'est faite
PTY		Test de la parité des 8 bits poids faible du registre A indicateurs : C est mis à 1 quand l'octet est impair

(1)	JL	JE	JG	JGE	JNE	JLE
CPBY	A < octet	A = octet	A > octet	A ≥ octet	A ≠ octet	A ≤ octet

STZ	MEM	Remise à zéro d'une mémoire adressage : direct, indirect, indirect indexe déplacement : compris entre - 128 et + 127 bases : C, L, W
IC	MEM	Addition de un à la mémoire et comparaison du résultat par rapport à zéro adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L, W indicateurs : V et C sont positionnés selon le résultat de la comparaison (1)
DC	MEM	Soustraction de un à la mémoire et comparaison du résultat par rapport à zéro adressage : direct, indirect, indirect indexe bases : c, L, W indicateurs : V et C sont positionnes selon le résultat de la comparaison (1)
LAD	MEM	Chargement du registre A par l'adresse de l'opérande adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
ADRP	Rs	Addition du registre source au registre P registre source : A B X Y C L W K ATTENTION : C'est l'adresse de l'instruction ADRP elle-même qui est additionnée
PSR	Rs [,Rs, . . .]	Rangement en mémoire du ou des registres source en utilisant le registre K comme pointeur de pile registres source : A B X Y C L W K
PLR	Rd [, Rd, . . .]	Chargement du ou des registres destination à partir de la mémoire en utilisant le registre K comme pointeur de pile registres destination : A B X Y C L W K
LRM	Rd [, Rd, . . .]	Chargement du ou des registres destination à partir des mots mémoire d'adresse P + 1, P + 2,... Registres destination : A, B, X, Y, C, L, W, K.

(1)	JL	JE	JG	JGE	JNE	JLE
IC, DC, CPZ	mem < 0	mem = 0	mem > 0	mem ≥ 0	mem ≠ 0	mem ≤ 0

OPÉRATIONS DIVERSES (suite)

PUSH (option scheduler sur 05) (standard sur 40 et 65)	Rangement du registre A dans la pile dont l'adresse est donnée par le registre Y indicateurs : C est mis à 1 quand le rangement sature la pile
PULL (option scheduler sur 05) (standard sur 40 et 65)	Chargement du registre A à partir de la pile dont l'adresse est donnée par le registre Y indicateurs : C est mis à 1 quand le rangement vide la pile
MOVE	Transport du contenu d'une zone mémoire, dont l'adresse est donnée par le registre A et la longueur dans le registre X, dans une zone mémoire dont l'adresse est donnée par le registre B
SCY	Mise à 1 de l'indicateur C sans modification de V
MVTM (mode maître)	Transfert d'une zone mémoire dont l'adresse relative à SLO est donnée dans le registre A et la longueur dans le registre X, vers la zone mémoire dont l'adresse absolue est donnée dans le registre B.
MVTS (mode maître)	Transfert d'une zone mémoire dont l'adresse absolue est donnée dans le registre A et la longueur dans le registre X, vers la zone mémoire dont l'adresse relative à SLO est donnée par le registre B.
LAR (mode maître)	Chargement du registre A par le mot mémoire dont l'adresse relative à SLO est donnée par le registre Y.
STAR (mode maître)	Rangement du registre A dans le mot mémoire dont l'adresse relative à SLO est donnée par le registre Y.
RDOE (mode maître)	Lecture dans les registres A et B des registres SLO et SLE.
WOE (mode maître)	Chargement des registres SLO et SLE à partir des registres A et B.
RCDA (option scheduler 65)	Transfert d'un segment de mémoire situé dans la zone de données communes (CDA) vers une zone mémoire relative à la tâche en cours : . le registre A contient l'adresse, relative au début de la zone de données communes, du 1er mot source, . le registre B contient l'adresse du 1er mot destination ; cette adresse est absolue ou relative à SLO selon l'état maître ou esclave du programme, . le registre X contient le nombre de mots à transférer.

WCDA (option scheduler 65)	Transfert d'un segment de mémoire relatif à la tâche en cours vers la zone de données communes (CDA) : . le registre B contient l'adresse du 1er mot de la zone source ; cette adresse est absolue ou relative à SLO selon que le programme est en mode maître ou esclave, . le registre A contient l'adresse, relative au début de la zone CDA, du 1er mot destination, . le registre X contient le nombre de mots à transférer.

BRANCHEMENTS ET SAUTS

inconditionnel

BR	MEM	Branchement inconditionnel à l'adresse donnée par le contenu de la mémoire adressage : direct, indirect, indirect indexé deplacement : compris entre - 128 et + 127 bases : C, L, W
JMP	$ ± 127	Saut inconditionnel déplacement : compris entre - 128 et + 127 base : P
ADRP		Voir opérations diverses

sous-programme

BSR	MEM	Sauvegarde du point de retour à l'adresse donnée par le registre K et branchement inconditionnel adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
RSR		Retour à l'adresse sauvegardée par un BSR et donnée par le registre K

BRANCHEMENTS ET SAUTS (suite)

superviseur

SVC	255	Appel d'une routine superviseur avec sauvegarde du point de retour à l'adresse donnée par le registre K
RSV		Retour à l'adresse sauvegardée par un SVC et donnée par le registre K

modification de l'index et saut

JIX	$ ± 127	Addition de 1 au registre X puis saut si X < 0 déplacement : compris entre - 128 et + 127 base : P
JDX	$ ± 127	Soustraction de 1 au registre X puis saut si X > 0 déplacement : compris entre - 128 et + 127 base : P

test accumulateur

JAL	$ ± 127	Saut si A < 0 déplacement : compris entre - 128 et + 127 base : P
JAE	$ ± 127	Saut si A = 0 déplacement compris entre - 128 et + 127 base : P
JAG	$ ± 127	Saut si A > 0 déplacement : compris entre - 128 et + 127 base : P
JAGE	$ ± 127	Saut si A ⩾ 0 déplacement : compris entre - 128 et + 127 base : P
JANE	$ ± 127	Saut si A ≠ 0 déplacement : compris entre - 128 et + 127 base : P
JALE	$ ± 127	Saut si A ⩽ 0 déplacement : compris entre - 128 et + 127 base : P

BRANCHEMENTS ET SAUTS (suite)

tests comparaisons

JL	$ ± 127	Saut si inférieur déplacement : compris entre - 128 et + 127 base : P
JE	$ ± 127	Saut si égal déplacement : compris entre - 128 et + 127 base : P
JG	$ ± 127	Saut si supérieur déplacement : compris entre - 128 et + 127 base : P
JGE	$ ± 127	Saut si supérieur ou égal déplacement : compris entre - 128 et + 127 base : P
JNE	$ ± 127	Saut si différent déplacement : compris entre - 128 et + 127 base : P
JLE	$ ± 127	Saut si inférieur ou égal déplacement : compris entre - 128 et + 127 base : P

tests indicateurs

JC	$ ± 127	Saut si l'indicateur C vaut 1 déplacement : compris entre - 128 et + 127 base : P
JV	$ ± 127	Saut si l'indicateur V vaut 1 déplacement : compris entre - 128 et + 127 base : P
JNC	$ ± 127	Saut si l'indicateur C vaut 0 déplacement : compris entre - 128 et + 127 base : P
JNV	$ ± 127	Saut si l'indicateur V vaut 0 déplacement : compris entre - 128 et + 127 base : P
JCV	$ ± 127	Saut si l'indicateur C vaut 1 OU l'indicateur V vaut 1 déplacement : compris entre - 128 et + 127 base : P
JNCV	$ ± 127	Saut si l'indicateur C vaut 0 et l'indicateur V vaut 0 déplacement : compris entre - 128 et + 127 base : P

ENTRÉES/SORTIES

SIO MEM	Lancement d'une opération d'entrée sortie avec un périphérique adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 + 127 bases : C, L, W
IPI	Voir interruption

INTERRUPTIONS

ACQ (mode maître)	Fin de tâche hardware
ACK	Reconnaissance du sous-niveau d'interruption le plus prioritaire sur le niveau en cours de traitement. X est chargé par le rang du sous-niveau. L'indicateur V est mis à 1 si un sous-niveau normal est en attente (x =0 à 15). L'indicateur C est mis à 1 si un sous-niveau exception est en attente (X = 0 à 47).
XIMR Rsd (mode maître)	Échange du contenu du registre IM et du registre destination de l'instruction. Registre : A B X Y C L W K
HALT	Le calculateur cesse d'exécuter des instructions jusqu'à l'épuisement du nombre de périodes indiqué dans X ou jusqu'à la prise en compte de la prochaine interruption.
DIT (mode maître)	Inhibition générale des interruptions de programme, y compris du défaut secteur mais à l'exclusion de l'interruption inter-processeur.
EIT (mode maître)	Désinhibition des interruptions de programme et du défaut secteur à l'exclusion de l'interruption inter-processeur.
IPI (mode maître)	Génération d'une interruption inter-processeur permettant en particulier d'initialiser les échanges en mode canal.
RDHV (mode maître)	Lecture du registre HV dans l'accumulateur.

ACTD	Appel du programme de mise au point (tâche hard 0, sous niveau 2)
STEP (mode maître)	Appel du programme de mise au point (tâche hard 0, sous niveau 2) (tâche hard 0, sous niveau 3)
ROMB (option : micro-programme non standard) (mode maître)	Déclenchement de l'exécution d'un microprogramme spécifique d'une application implanté en mémoire morte.
SST (mode maître)	Force à un les bits du registre ST spécifiés dans l'accumulateur et charge la nouvelle valeur de ST dans l'accumulateur.
RST (mode maître)	Force à 0 les bits du registre ST spécifiés dans l'accumulateur et charge la nouvelle valeur de ST dans l'accumulateur.
SBP (mode maître)	Crée un point d'arrêt en inversant la parité normale du mot mémoire dont l'adresse relative à SLO est donnée par le registre Y.
RBP (mode maître)	Efface un point d'arrêt éventuel en redonnant une parité normale au mot mémoire dont l'adresse relative à SLO est donnée par le registre Y
DBP (mode maître)	Recherche les points d'arrêt (mots en parité inverse) en décrémentant les adresses depuis l'adresse contenue dans le registre Y jusqu'à 0 (ceci relativement à SLO). L'indicateur C est mis à 1 à la rencontre d'un point d'arrêt. L'indicateur V est mis à 1 à la rencontre d'une adresse inexistante (05).
RDSI (mode maître)	Charge dans l'accumulateur des informations sur l'identité du processeur (type, numéro de processeur, n° de bac,...).

FLOTTANT SIMPLE PRECISION (OPTION FFP16)

FLD MEM (option virgule flottante)	Chargement de deux mémoires consécutives dans les registres A et B adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L, W indicateur; : V = 0 C = 0
FST MEM (option virgule flottante)	Rangement des registres A et B dans deux mémoires consécutives adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L, W indicateurs : V = 0 C = 0
FAD MEM (option virgule flottante)	Addition du nombre flottant mémoire aux registres A et B adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : overflow : V = 1 C = 0 underflow : V = 0 C = 1 si non : C = 0 C = 0
FSB MEM (option virgule flottante)	Soustraction du nombre flottant mémoire aux registres A et B adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : overflow : V = 1 C = 0 underflow : V = 0 C = 1 si non : V = 0 C = 0
FNEG (option virgule flottante)	Inversion du signe du nombre flottant contenu dans les registres A et B indicateurs : V = 0 C = 0
FABS (option virgule flottant)	Calcul de la valeur absolue du nombre flottant contenu dans les registres A et B indicateurs : V = 0 C = 0
FMP MEM (option virgule flottante)	Multiplication des registres A et B par le nombre flottant mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L. w indicateurs : overflow : V = 1 C = 0 underflow : V = 0 C = 1 si non : V = 0 C = 0
FDV MEM (option virgule flottante)	Division des registres A et B par le nombre flottant mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : c, L, W indicateurs : overflow et division par zéro : V = 1 C = 0 underflow V = 0 C = 1 si non V = 0 C = 0

FLOTTANT SIMPLE PRECISION (suite)

FCAM MEM (option virgule flottante)	Comparaison des registres A et B avec le nombre flottant. mémoire adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : V et C sont positionnés selon le résultat de la comparaison (1)
FCAZ (option virgule flottante)	Comparaison des registres A et B avec le nombre flottant zéro indicateurs : V et C sont positionnés selon le résultat de la comparaison (1)
FCMZ MEM (option virgule flottant)	Comparaison du nombre flottant mémoire avec le nombre flottant zéro adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W indicateurs : V et C sont position& selon le résultat de la comparaison (1)
FIX (option virgule flottante)	Conversion en entier dans le registre de la partie entière (tronquée) du nombre flottant contenu dans les registres A et B. indicateurs : nombre flottant trop grand : V = 1 C = 0 si non V = 0 C = 0
FLT (option virgule flottante)	Conversion en flottant dans les registres A et B du nombre entier contenu dans le registre A indicateurs : C = 0 V = 0
NORM (option virgule flottante)	Normalisation du nombre flottant contenu dans les registres A et B overflow : V = 1 C = 0 indicateurs : overflow : V = 1 C = 0 underflow : V = 0 C = 1 si non : V = 0 C = 0

(1)	JL	JE	JG	JGE	JNE	JLE
FCAM	AB < MEM	AB = MEM	AB > MEM	AB ≥ MEM	AB ≠ MEM	AB ≤ MEM
FCAZ	AB < 0	AB = 0	AB > 0	AB ≥ 0	AB ≠ 0	AB ≤ 0
FCMZ	MEM < 0	MEM = 0	MEM > 0	MEM ≥ 0	MEM ≠ 0	MEM ≤ 0

SCHEDULER, SEMAPHORES (OPTION MTS 16)

ARM 127 (option scheduler-sémaphores) (mode maître)	Armement d'une tâche différée numéro de tâche : compris entre 0 et 127, indexation possible
QUIT (option scheduler-sémaphores)	Fin d'une tâche différée
RQST MEM (option scheduler-sémaphores) (mode maître) (tâche différée seulement)	Demande d'accès à une ressource gérée par un sémaphore d'exclusion mutuelle adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 et + 127 bases : C, L, W
RLSE MEM (option -sémaphores) (mode maître)	Restitution d'un accès à une ressource gérée par un sémaphore d'exclusion mutuelle adressage : direct, indirect, indirect indexe deplacement : compris entre - 128 et + 127 bases : c. L. W
WAIT MEM (option scheduler-sémaphores) (mode maître) (tâche software seulement)	Sémaphore de synchronisation (sans files) : demande de synchronisation Sémaphore d'appel (avec files) : fin de la tâche différée gérée par le sémaphore adressage : direct, indirect, indirect indexé déplacement : compris entre - 128 + 127 bases : C, L, W
ACT MEM (option scheduler-sémaphores) (mode maître)	Semaphore de synchronisation (sans files) : réponse de synchronisation Sémaphore d'appel (avec files) : appel de la tâche différée gérée par le semaphore adressage : direct, indirect, indirect indexe déplacement : compris entre – 128 et + 127 bases : C, L, W

OPERATIONS SUR LISTES (OPTION SCHEDULER 65)

INSQ	Insère dans une liste l'élément dont l'adresse est dans l'accumulateur à la suite de l'élément dont l'adresse est dans le registre B.
SUPQ	Supprime dans une liste l'élément dont l'adresse est donnée dans l'accumulateur.
SFQ	Supprime dans une liste l'élément qui précède celui dont l'adresse est dans l'accumulateur et en donne l'adresse dans l'accumulateur.
SLQ	Supprime le successeur de l'élément dont l'adresse est dans l'accumulateur et en donne l'adresse dans l'accumulateur.

INSTRUCTIONS PRIVILEGIEES (non exécutables en mode esclave)

ACQ	acquit
ACT	activate
ARM	arm
BLA	based load A
BSTA	based store A
BXM	based exchange memory with A
BLBY	based load byte
BSTBY	based store byte
BDLD	based double load
BDST	based double store
BMOVE	based move
BDBTM	based discover bit in memory
BRBTM	based reset bit in memory
BSBTM	based set bit in memory
DBP	discover breakpoint
DIT	disable interrupt
EIT	enable interrupt
IPI	inter-processor interrupt
LAR	load A relative
MVTM	move to master locations
MVTS	move to slave locations
RBP	reset breakpoint
RDHV	read HV
RDSI	read system identification
RLSE	release
RDOE	read SLO SLE
ROMB	read-only memory branch
RQST	request
RST	read ST
SBP	set breakpoint
SIO	start IO
SST	set ST
STAR	store A relative
STEP	step
WAIT	wait
WOE	write SLO SLE
XCTX (1)	commutation de contexte appelant
XIMR	exchange IM with register

(1) Exécutable uniquement en mode maître

INSTRUCTIONS DE L'OPTION SCHEDULER-SEMAPHORES (MTS 16)

ACT	activate
ARM	arm
PULL (1)	pull
PUSH (1)	push
QUIT	quit
RLSE	release
RQST	request
SBS (1)	search byte string
WAIT	wait
DRBM (2)	discover reset bit in memory
INSQ (2)	insert in queue
RBTM (2)	reset bit in memory
RCDA (2)	read common data area
SBTM (2)	set bit in memory
SFQ (2)	suppress first in queue
SLQ (2)	suppress last in queue
SUPQ (2)	suppress from queue
WCDA (2)	write common data aera

(1) standard sur 40 et 65

(2) 65 seulement (option dite CDA)

INSTRUCTIONS DE L'OPTION VIRGULE FLOTTANTE SIMPLE PRECISION (FFP16)

FABS	floating point absolute value
FAD	floating point add
FCAM	floating point compare accumulator to memory
FCAZ	floating point compare accumulator to zero
FCMZ	floating point compare memory to zero
FDV	floating point divide
FIX	fix
FLD	floating point load
FLT	float
FMP	floating point multiply
FNEG	floating point negate
FSB	floating point substract
FST	floating point store
NORM	floating point normalize

INSTRUCTIONS DE L'OPTION ISP16

a) Instructions basées

BLA	based load A
BSTA	based store A
BXM	based exchange memory with A
BDLD	based double load
BDST	based double store
BLBY	based load byte
BSTBY	based store byte
BMOVE	based move
BRBTM	based reset bit in memory
BSBTM	based set bit in memory
BDBTM	based discover bit in memory

b) Instructions sur CDA

CLA	CDA load A
CSTA	CDA store A
CXM	CDA exchange memory with A
CDLD	CDA double load
CDST	CDA double store
CLBY	CDA load byte
CSTBY	CDA store byte

c) Instructions diverses

XCTX	commutation de contexte appelant
XENT	entrée de procédure langage
XSOR	sortie de procédure langage

INSTRUCTIONS DIFFERENTES ENTRE SOLAR ET T 1600

a) instructions nouvelles sur SOLAR

DBP	discover breakpoint
DRBM	discover and reset bit in memory
DIT	disable interrupt
EIT	enable interrupt
INSQ	insert in queue
IPI	interprocessor interrupt
LAR	load A relative
LRM	load register multiple
MVTM	move to master locations
MVTS	move to slave locations
RBP	reset breakpoint
RBTM	reset bit in memory
RCDA	read common data area
RDHV	read HV
RDSI	read system identification
ROE	read SLO SLE
RST	read ST
SBP	set breakpoint
SBTM	set bit in memory
SFQ	suppress first in queue
SLQ	" last in queue
SST	set ST
STAR	store A relative
SUPQ	suppress from queue
WCDA	write common data area
WOE	write SLO SLE
XM	exchange memory whith A

b) Instructions supprimées sur SOLAR

BIO	bus input/output
NOP (codé '1Dxx)	no operation

cl Instructions modifiées

ACK	acknowledge
ACQ	acquit
ACT	activate
ARM	arm
BSR	branch subroutine
HALT	halt
MOVE	move
PLR	pu ll registers
PSR	push registers
PULL	pull A
PUSH	push A
QUIT	quit
RLSE	release
RQST	request
RSR	return from subroutine
RSV	return from supervisor
SBS	search byte string
SIO	start input/output
SVC	supervisor-call
WAIT	wait
XIMR	exchange IM with registor

MÉMOIRES DEBANALISÉES

adresse	mnémonique	contenu	exploité par micro	exploité par machine
'0000	NS	**numéro de la tâche différée en cours : n_s**		
1	NL	**numéro de la tâche différée non prête : n_l**		X
2	APSTH	adresse de début des PSTH		X
3	APSTS	adresse des relais des PSTS		X
4	ASVCM	adresse d'appel en mode maître des sousprogrammes superviseurs		X
5	ASVCS	adresse d'appel en mode esclave des sous-programmes superviseurs		X
6	SECT	adresse de la PSTH défaut secteur		X
7	ENTBOS	adresse de retour à BQS (utilisée par AID)		
8	INI	adresse de démarrage sur initialisation ou relance automatique		X
9				
A	MAXMEM	adresse maximum x 16 de l'espace mémoire		
B	FIRSTM	adresse de la première mémoire libre		
C	SYSMEM	adresse maximum x 16 de la zone système		
D		mot 29 de BOS-D		
E	ABOX	adresse de la boîte aux lettres d'adresse BOX		X
F	WSCHED	mémoire de travail du scheduler		X
'0010	WPOPO	utilisé par le pupitre opérateur UT n° 0		X
1	WPOP1	utilisé par le pupitre opérateur UT n° 1		X
2	WPOP2	utilisé par le pupitre opérateur UT n° 2		X
3	WPOP3	utilisé par le pupitre opérateur UT n° 3		X
4	ALDC0	adresse des zones de sauvegarde des contextes LDC du processeur n° 0		X
5	ALDC1	adresse des zones de sauvegarde des contextes LDC du processeur n° 1		X
6	ALDC2	adresse des zones de sauvegarde des contextes LDC du processeur n° 2		X
7	ALDC3	adresse des zones de sauvegarde des contextes LDC du processeur n° 3		X
8	OCDA	adresse de début de la CDA x 16		X
9	ECDA	adresse de fin de la CDA x 16		X
A	ATVSS	adresse de la table des paramètres de VSS		X
:				
E	ABOX 2	adresse de la boite aux lettres réveil interprocesseur 65/75 P.		X
F	ASVCP	adresse d'appel en mode privilégié des sous-programmes superviseurs		X
0020	file ASTF	file des tâches différées armées		X
7				
0028				
9				
:	file ESTF	file des tâches différées éligibles		X
D				
E				
F				
'0030				
1				
2		file des tâches différées prêtes		X
:	file RSTF			
6				
7				
'0038		début du bootstrap		
:				
'005F		fin du bootstrap		X
'0060				
:				
'0137		fin du bootstrap disque		

LISTE DES ALARMES

'00	mémoire inexistante	05
‘01	protection DRPS ou CDA	05
‘02	erreur de parité hors du mode DEBUG	
‘03	instruction optionnelle inexistante	
‘04	instruction privilégiée	05
‘05	RQST, WAIT, QUIT sousniveau hard (option scheduler)	
‘06	appel d’une autre UT : réveil interprocesseur (IPI)	05
‘07	STEP	
‘08	ACTD	

Ce numéro est chargé dans l’octet droit du 4e mot de la PSTH alarme, lors du changement de contexte partiel qui précède le lancement de la tâche alarme.

TEMPS DES INSTRUCTIONS

– Les temps fournis correspondent à la valeur nominale du cycle de base de chaque unité centrale.

– Ces temps incluent l'influence du "refresh" pour le cas des mémoires dynamiques.

– Ces temps peuvent être altérés par les phénomènes suivants :

. fonctionnement simultané de canaux intégrés ou sur IOP,
. utilisation du module mémoire cache,
. utilisation de mémoire vue à travers un CBM,
. utilisation du mode DEBUG,
. valeur des opérandes pour certaines instructions.

– Certains temps sont donnes par paires d'instructions :

. ACTD + ACQ : fournit sensiblement le coût de prise en compte et d'acquittement d'une interruption, soit un double changement de contexte "hard" terminé par un scheduling conservant la priorité logicielle courante.

. ACT + WAIT, RQST + RLSE, ARM + QUIT fournissent la somme des temps d'exécution de ces instructions dans le cas où elles conduisent à un changement de la tache différée en cours, ces temps incluent donc deux changements de contexte.

– Les notations PI et E utilisées pour certains opérandes ont pour valeur :

. PI = 3,14159
. E = 2,71828

– Le temps de MP est donné pour le calcul suivant : '5555 * '5555 = '1C718E39

– Le temps de DV est donné pour le calcul suivant : . 1C718E39 / '5555 = '5555

– Les temps de MP et DV sont donnés avec option FFP16 sur 16-75, 65 et 40 et sans option sur 16-05 et 04.

Le tableau ci-dessous donne les temps de MP et DV dans tous les cas d'options

	Mémoire	Option FFP16		Option MP DV		Sans option	
	(cycles)	MP	DV	MP	DV	MP	DV
16-75	3	2.94	3.37	3.95	4.64	20.4	43.1
16-65	5	3.39	3.83	4.4	4.85	20.5	43.8
16-65	6	3.40	4.15	4.8	5.4	20.5	44.0
16-40	5	5.94	6.20	7.4	7.45	63.2	61.2
16-40	6	6.35	6.50	7.4	7.7	63.1	60.8
16-05	–	*	*	9.6	11.3	98.4	101
16-04	–	*	*	9.9	11.3	98.1	102
16-30	5	6.35	6.23	7.9	8.0	67.5	64.9

temps en microsecondes pour les opérandes définis précédemment
* option n'existant pas sur ce modèle

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-75 — MEMOIRE 3 CYCLES

ACK		5.95	JAGE	>=0	0.84	ORI	+	0.56	SCRU	7	7.02
ACQ+ACTD		30.14	JAGE	A<0	1.12	ORI	-	0.56	SCRU	8	5.93
ACT		4.63	JAL	A<0	0.84	ORR		0.56	SCRU	15	7.87
ACT+WAIT		77.07	JAL	A>=0	1.12	PLR	1REG	2.10	SCRU	16	4.07
ACTD+ACQ		30.14	JALE	<=0	0.84	PLR	2REG	3.23	SCRU	23	7.87
AD		0.84	JALE	A>0	1.12	PLR	4REG	5.47	SCRU	24	4.78
ADCR		0.84	JANE	A#0	0.84	PSR	1REG	2.52	SCRU	31	8.73
ADR		0.56	JANE	A=0	1.12	PSR	2REG	3.51	SCRS	1	5.93
ADRI	+	0.56	JC	C=1	0.98	PSR	4REG	4.49	SCRS	7	8.40
ADRI	-	0.84	JC	C=0	1.26	PTY		1.83	SCRS	8	5.79
ADRP		0.84	JCV	CV=1	1.12	PULL		4.64	SCRS	15	8.74
AND		0.84	JCV	CV=0	1.41	PUSH		4.64	SCY		1.97
ANDI	+	0.56	JDX	X>0	0.84	QUIT+ARM		77.84	SFG		4.08
ANDI	-	0.56	JDX	X<=0	1.12	RBP		4.08	SIO	C	5.51
ANDR		0.56	JE	(EQ)	1.12	RBT	(A)	0.84	SIO	L	2.07
ARM		6.74	JE	(NE)	1.41	RBT	(B)	0.84	SLLU	1	8.73
ARM+QUIT		77.84	JG	(GT)	1.12	RBT	(AX)	0.84	SLLU	8	4.78
BR		0.85	JG	(LE)	1.41	RBT	(BX)	0.84	SLLU	9	7.87
BSR		1.26	JGE	(GE)	0.98	RBTM		3.94	SLLU	10	5.93
CLSR		0.56	JGE	(LT)	1.26	RCDA	1	10.24	SLLU	17	8.45
CMR		0.70	JIX	X<0	0.84	RCDA	2	12.37	SLLU	24	4.50
CP		0.84	JIX	X>=0	1.12	RCDA	5	18.69	SLLU	25	7.80
CPBY		1.12	JL	(LT)	0.98	RDHV		1.97	SLLU	31	4.21
CPI	+	0.56	JL	(GE)	1.26	RDOE		2.11	SLLS	1	8.59
CPI	-	0.56	JLE	(LE)	1.12	RDSI		2.39	SLLS	8	5.05
CPR		0.56	JLE	(GT)	1.41	RLSE		4.22	SLLS	9	8.32
CPZ		0.84	JMP		0.56	RLSERQST		79.75	SLLS	16	5.37
CPZR		0.56	JNC	C=0	0.98	RQST		4.22	SLW		4.08
DBP	Y=1	8.44	JNC	C=1	1.26	RQSTRLSE		79.75	SLRU	1	5.51
DBP	Y=2	12.65	JNCV	NCV	1.12	RSR		2.25	SLRU	7	8.09
DBT	(A)	1.97	JNCV	VC	1.41	RST		2.53	SLRU	8	5.79
DBT	(B)	2.39	JNE	(NE)	1.12	RSV		2.06	SLRU	15	7.75
DC		1.41	JNE	(EQ)	1.41	SARD	1	5.93	SLRU	16	5.93
DIT		2.23	JNV	V=0	1.12	SARD	7	7.32	SLRU	23	7.75
DRBM	M1	4.21	JNV	V=1	1.40	SARD	8	4.21	SLRU	24	4.64
DRBM	M2	5.76	JV	V=1	1.12	SARD	15	8.15	SLRU	31	8.59
DRBM	M3	7.30	JV	V=0	1.40	SARD	16	4.35	SLRS	1	5.65
DV		3.37	LA		0.84	SARD	23	8.15	SLRS	7	8.19
EIT		3.82	LAD		0.84	SARD	24	5.06	SLRS	8	5.51
EOR		0.84	LAI	+	0.56	SARD	31	9.02	SLRS	15	8.45
EORI	+	0.56	LAI	-	0.56	SARS	1	4.21	SST		2.39
EORI	-	0.56	LAR		3.66	SARS	7	6.74	STA		0.98
EORR		0.56	LB		0.84	SARS	8	4.08	STAR		5.51
FABS	+	2.24	LBI	+	0.56	SARS	15	7.02	STB		0.98
FABS	-	3.09	LBI	-	0.56	SB		0.84	STBY		1.26
FAD	PI+E	5.05	LBY	1	0.98	SBCR		0.84	STEP		16.15
FCAM		4.22	LR		0.56	SBP		5.30	STX		0.98
FCAZ		2.24	LRM	1REG	2.81	SBR		0.56	STY		0.98
FCMZ		2.53	LRM	2REG	3.94	SBS	1	3.65	STZ		0.98
FDV	PI/E	7.02	LRM	4REG	6.18	SBS	2	4.77	SUPG		3.35
FIX	0	2.66	LRP		0.70	SBS	3	7.45	SVC		2.10
FIX	PI	6.18	LX		0.84	SBS	4	8.57	SWBR		0.70
FLD		2.53	LXI	+	0.56	SBT	(A)	0.84	TBT	(A)	0.84
FLT		4.64	LXI	-	0.56	SBT	(B)	0.84	TBT	(B)	0.84
FMP	PI*E	6.05	LY		0.84	SBT	(AX)	0.84	TBT	(AX)	0.84
FNEG		2.95	LYI	+	0.56	SBT	(BX)	0.84	TBT	(BX)	0.84
FSB	PI-E	5.05	LYI	-	0.56	SBTM		3.93	WAIT		3.47
FST		3.23	MOVE	1	3.94	SBTM	(X)	3.79	WAIT+ACT		77.10
HALT	1	2.10	MOVE	2	5.19	SCLD	1	8.73	WCDA	1	10.10
HALT	2	4.35	MOVE	5	9.00	SCLD	8	4.78	WCDA	2	12.50
IBT	(A)	0.84	MP		2.94	SCLD	9	7.87	WOE		1.97
IBT	(B)	0.84	MVTM	1	5.62	SCLD	16	3.93	XIMR		3.37
IBT	(AX)	0.84	MVTM	2	8.71	SCLD	17	8.45	XM		1.12
IBT	(BX)	0.84	MVTM	5	18.00	SCLD	24	4.50	XR		0.84
IC		1.40	MVTS	1	5.48	SCLD	25	7.80	&		0.42
INSQ		3.36	MVTS	2	8.44	SCLD	31	4.21	&X		0.42
IPI		5.51	MVTS	5	17.29	SCLS	1	6.59	& FLUTAN		0.42
JAE	A=0	0.84	NGR		0.70	SCLS	8	3.65	&XFLUTAN		0.42
JAE	A#0	1.12	NOP		0.84	SCLS	9	6.32			
JAG	A>0	0.84	NORM		4.64	SCLS	16	3.37			
JAG	A<=0	1.12	OR		0.84	SCRD	1	5.66			

Réseaux et systèmes d'information

Bull

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-65 MEMOIRE 5 CYCLES

ACK		5.20	JAGE	>=0	0.87	URI	+	0.73	SCRD	7	7.33
ACQ+ACTD		34.19	JAGE	A<0	1.61	URI	-	0.73	SCRD	8	4.23
ACT		5.13	JAL	A<0	0.87	URR		0.73	SCRD	15	8.23
ACT+WAIT		90.12	JAL	A>=0	1.61	PLR	1REG	2.12	SCRD	16	4.58
ACTD+ACQ		34.19	JALE	<=0	0.87	PLR	2REG	3.36	SCRD	23	8.23
AD		1.47	JALE	A>0	1.61	PLR	4REG	5.33	SCRD	24	5.08
ADCR		0.83	JANE	A#0	0.87	PSR	1REG	2.85	SCRD	31	9.08
ADR		0.73	JANE	A=0	1.61	PSR	2REG	3.90	SCRS	1	4.23
ADRI	+	0.73	JC	C=1	1.01	PSR	4REG	4.81	SCRS	7	8.78
ADRI	-	0.83	JC	C=0	1.62	PTY		2.11	SCRS	8	4.09
ADRP		1.15	JCV	CV=1	1.13	PULL		5.45	SCRS	15	7.07
AND		1.47	JCV	CV=0	1.75	PUSH		5.62	SCY		2.26
ANDI	+	0.73	JDX	X>0	0.87	QUIT+ARM		90.70	SFG		5.77
ANDI	-	0.73	JDX	X<=0	1.61	RBP		4.74	SIO	C	5.92
ANDR		0.70	JE	(EQ)	1.13	RBT	(A)	0.84	SIO	L	5.08
ARM		7.69	JE	(NE)	1.75	RBT	(B)	0.83	SLLD	1	9.04
ARM+QUIT		90.65	JG	(GT)	1.13	RBT	(AX)	0.86	SLLD	8	5.08
BR		1.45	JG	(LE)	1.75	RBT	(BX)	0.87	SLLD	9	8.23
BSR		2.36	JGE	(GE)	1.00	RBTM		4.42	SLLD	16	4.23
CLSR		0.73	JGE	(LT)	1.61	RCDA	1	10.94	SLLD	17	8.77
CMR		0.73	JIX	X<0	0.87	RCDA	2	13.22	SLLD	24	4.81
CP		1.47	JIX	X>=0	1.61	RCDA	5	19.61	SLLD	25	7.86
CPBY		1.47	JL	(LT)	1.01	RDMV		2.28	SLLD	31	4.49
CPI	+	0.70	JL	(GE)	1.62	RDOE		2.28	SLLS	1	8.92
CPI	-	0.70	JLE	(LE)	1.13	RDSI		3.10	SLLS	8	3.95
CPR		0.73	JLE	(GT)	1.75	RLSE		5.05	SLLS	9	8.05
CPZ		1.47	JMP		0.73	RLSERQST		95.70	SLLS	16	3.87
CPZR		0.73	JNC	C=0	1.00	RQST		4.65	SLW		5.77
DBP	Y=1	9.68	JNC	C=1	1.61	RQSTRLSE		95.70	SLRD	1	3.02
DBP	Y=2	14.45	JNCV	NCV	1.13	RSR		2.62	SLRD	7	7.21
DBT	(A)	2.14	JNCV	VC	1.75	RST		2.89	SLRD	8	4.09
DBT	(B)	2.40	JNE	(NE)	1.13	RSV		3.04	SLRD	15	8.04
DC		2.49	JNE	(EQ)	1.75	SARD	1	4.23	SLRD	16	4.23
DIT		2.55	JNV	V=0	1.13	SARD	7	7.61	SLRD	23	8.04
DRBM	M1	4.73	JNV	V=1	1.75	SARD	8	4.49	SLRD	24	4.93
DRBM	M2	6.55	JV	V=1	1.13	SARD	15	8.47	SLRD	31	8.88
DRBM	M3	8.32	JV	V=0	1.75	SARD	16	4.65	SLRS	1	3.95
DV		3.83	LA		1.47	SARD	23	8.47	SLRS	7	8.49
EIT		4.23	LAD		1.15	SARD	24	5.37	SLRS	8	5.82
EOR		1.47	LAI	+	0.73	SARD	31	9.33	SLRS	15	8.75
EORI	+	0.70	LAI	-	0.73	SARS	1	4.49	SST		2.68
EORI	-	0.70	LAR		4.25	SARS	7	7.07	STA		1.75
EORR		0.73	LB		1.46	SARS	8	4.38	STAR		3.65
FABS	+	2.41	LBI	+	0.73	SARS	15	7.33	STB		1.75
FABS	-	3.11	LBI	-	0.73	SB		1.46	STBY		2.54
FAD	PI+E	5.47	LBY	1	1.46	SBCR		0.84	STEP		19.93
FCAM		5.07	LR		0.73	SBP		6.11	STX		1.75
FCAZ		2.41	LRM	1REG	2.88	SBR		0.73	STY		1.75
FCMZ		3.30	LRM	2REG	4.12	SBS	1	4.19	STZ		1.75
FDV	PI/E	7.38	LRM	4REG	6.44	SBS	2	5.16	SUPG		4.57
FIX	0	2.94	LRP		1.02	SBS	3	8.27	SVC		3.12
FIX	PI	6.17	LX		1.46	SBS	4	9.45	SWBR		0.73
FLD		3.31	LXI	+	0.73	SBT	(A)	0.84	TBI	(A)	0.87
FLT		4.84	LXI	-	0.73	SBT	(B)	0.84	TBT	(B)	0.87
FMP	PI*E	6.36	LY		1.46	SBT	(AX)	0.86	TBT	(AX)	0.87
FNEG		2.99	LYI	+	0.73	SBT	(BX)	0.86	TBT	(BX)	0.87
FSB	PI-E	5.47	LYI	-	0.73	SBTM		4.40	WAIT		5.79
FST		5.13	MOVE	1	4.32	SBTM	(X)	4.21	WAIT+ACT		90.18
HALT	1	2.38	MOVE	2	5.88	SCLD	1	9.05	WCDA	1	10.70
HALT	2	4.67	MOVE	5	10.25	SCLD	8	5.08	WCDA	2	15.27
IBT	(A)	0.87	MP		3.39	SCLD	9	8.28	WOE		2.29
IBT	(B)	0.87	MVTM	1	5.70	SCLD	16	4.23	XIMR		3.53
IBT	(AX)	0.87	MVTM	2	9.11	SCLD	17	8.78	XM		1.75
IBT	(BX)	0.87	MVTM	5	18.46	SCLD	24	4.81	XR		0.64
IC		2.49	MVTS	1	5.82	SCLD	25	7.91	&		0.73
INSQ		4.84	MVTS	2	8.84	SCLD	31	4.49	&&		0.73
IPI		7.33	MVTS	5	17.85	SCLS	1	6.92	& FLOTTANT		0.08
JAE	A=0	0.87	NGR		0.73	SCLS	8	3.95	&& FLOTTANT		0.08
JAE	A#0	1.61	NOP		1.15	SCLS	9	6.65			
JAG	A>0	0.87	NORM		4.73	SCLS	16	3.69			
JAG	A<=0	1.61	OR		1.47	SCRD	1	3.95			

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-65 — MEMOIRE 6 CYCLES

ACK		6.28	JAGE	>=0	0.85	ORI	+	0.86	SCRD	7	7.48
ACQ+ACTD		36.95	JAGE	A<0	1.85	ORI	-	0.86	SCRD	8	4.37
ACT		5.38	JAL	A<0	0.85	ORR		0.85	SCRD	15	8.30
ACT+WAIT		99.09	JAL	A>=0	1.85	PLR	1REG	2.29	SCRD	16	4.52
ACTD+ACQ		36.95	JALE	<=0	0.85	PLR	2REG	3.58	SCRD	23	8.29
AD		1.71	JALE	A>0	1.85	PLR	4REG	5.83	SCRD	24	5.22
ADCR		0.86	JANE	A#0	0.85	PSR	1REG	2.98	SCRD	31	9.16
ADR		0.86	JANE	A=0	1.85	PSR	2REG	4.01	SCRS	1	4.37
ADRI	+	0.86	JC	C=1	1.01	PSR	4REG	4.94	SCRS	7	8.93
ADRI	-	0.86	JC	C=0	1.84	PTY		2.27	SCRS	8	4.23
ADRP		1.27	JCV	CV=1	1.13	PULL		5.82	SCRS	15	7.20
AND		1.72	JCV	CV=0	1.84	PUSH		6.22	SCY		2.39
ANDI	+	0.86	JDX	X>0	0.85	QUIT+ARM		100.47	SFG		8.52
ANDI	-	0.86	JDX	X<=0	1.84	RBP		4.94	SIO	C	4.10
ANDR		0.85	JE	(EQ)	1.14	RBT	(A)	0.86	SIO	L	3.42
ARM		8.50	JE	(NE)	1.84	RBT	(B)	0.86	SLLD	1	9.16
ARM+QUIT		100.47	JG	(GT)	1.14	RBT	(AX)	0.85	SLLD	8	5.22
BR		1.73	JG	(LE)	1.84	RBT	(BX)	0.85	SLLD	9	8.30
BSR		2.55	JGE	(GE)	0.99	RBTM		5.23	SLLD	16	4.37
CLSR		0.85	JGE	(LT)	1.84	RCDA	1	11.27	SLLD	17	8.87
CMR		0.85	JIX	X<0	0.85	RCDA	2	13.54	SLLD	24	4.92
CP		1.70	JIX	X>=0	1.84	RCDA	5	20.28	SLLD	25	8.05
CPBY		1.70	JL	(LT)	1.01	RDHV		2.40	SLLD	31	4.06
CPI	+	0.85	JL	(GE)	1.84	RDOE		2.40	SLLS	1	7.04
CPI	-	0.85	JLE	(LE)	1.13	RDSI		3.22	SLLS	8	4.08
CPR		0.84	JLE	(GT)	1.84	RLSE		5.03	SLLS	9	8.75
CPZ		1.71	JMP		0.85	RLSERQST		103.88	SLLS	16	5.80
CPZR		0.84	JNC	C=0	0.99	RQST		4.79	SLG		8.52
DBP	Y=1	10.11	JNC	C=1	1.84	RQSTRLSE		103.88	SLRD	1	3.95
DBP	Y=2	15.21	JNCV	NCV	1.14	RSR		3.12	SLRD	7	7.32
DBT	(A)	2.25	JNCV	VC	1.84	RST		2.94	SLRD	8	4.23
DBT	(B)	2.40	JNE	(NE)	1.14	RSV		3.42	SLRD	15	8.15
DC		2.67	JNE	(EQ)	1.84	SARD	1	4.37	SLRD	16	4.37
DIT		2.69	JNV	V=0	1.14	SARD	7	7.73	SLRD	23	8.15
DRBM	M1	4.72	JNV	V=1	1.84	SARD	8	4.65	SLRD	24	5.08
DRBM	M2	6.81	JV	V=1	1.14	SARD	15	8.60	SLRD	31	9.04
DRBM	M3	8.78	JV	V=0	1.84	SARD	16	4.79	SLRS	1	4.08
DV		4.15	LA		1.71	SARD	23	8.60	SLRS	7	8.05
EIT		4.24	LAD		1.27	SARD	24	5.51	SLRS	8	3.95
EOR		1.71	LAI	+	0.85	SARD	31	9.45	SLRS	15	8.93
EORI	+	0.85	LAI	-	0.85	SARS	1	4.65	SSI		2.78
EORI	-	0.85	LAR		4.51	SARS	7	7.19	STA		1.84
EORR		0.86	LB		1.71	SARS	8	4.52	STAR		3.95
FABS	+	2.74	LBI	+	0.85	SARS	15	7.48	STO		1.84
FABS	-	3.24	LBI	-	0.85	SB		1.71	STBY		2.56
FAD	PI+E	5.87	LBY	1	1.71	SBCR		0.85	STEP		21.49
FCAM		5.23	LR		0.85	SBP		6.78	STX		1.84
FCAZ		2.74	LRM	1REG	3.10	SBR		0.85	STY		1.84
FCMZ		3.84	LRM	2REG	4.25	SBS	1	4.56	STZ		1.84
FDV	PI/E	7.52	LRM	4REG	6.46	SBS	2	5.61	SUPG		5.52
FIX	0	2.77	LRP		1.12	SBS	3	8.80	SVC		3.34
FIX	PI	6.40	LX		1.71	SBS	4	9.88	SWBR		0.85
FLD		3.76	LXI	+	0.85	SBT	(A)	0.85	TBT	(A)	0.83
FLT		5.10	LXI	-	0.85	SBT	(B)	0.85	TBT	(B)	0.83
FMP	PI*E	7.03	LY		1.71	SBT	(AX)	0.85	TBT	(AX)	0.83
FNEG		3.06	LYI	+	0.85	SBT	(BX)	0.85	TBT	(BX)	0.83
FSB	PI-E	5.87	LYI	-	0.85	SBTM		5.21	WAIT		0.12
FST		5.58	MOVE	1	4.07	SBTM	(X)	5.07	WAIT+ACT		99.22
HALT	1	2.54	MOVE	2	6.29	SCLD	1	9.16	WCDA	1	11.13
HALT	2	4.76	MOVE	5	10.96	SCLD	8	5.22	WCDA	2	13.05
IBT	(A)	0.85	MP		3.40	SCLD	9	8.29	WOE		2.41
IBT	(B)	0.85	MVTM	1	6.06	SCLD	16	4.37	XIMR		4.03
IBT	(AX)	0.85	MVTM	2	9.10	SCLD	17	8.87	XM		1.99
IBT	(BX)	0.85	MVTM	5	18.46	SCLD	24	4.95	XR		0.85
IC		2.71	MVTS	1	5.87	SCLD	25	8.05	&		0.86
INSQ		5.43	MVTS	2	8.87	SCLD	31	4.65	&X		0.86
IPI		7.39	MVTS	5	17.09	SCLS	1	7.04	& FLUTAN		0.86
JAE	A=0	0.85	NGR		0.85	SCLS	8	4.06	&XFLUTAN		0.86
JAE	A#0	1.85	NOP		1.27	SCLS	9	6.75			
JAG	A>0	0.85	NORM		4.99	SCLS	16	3.61			
JAG	A<=0	1.84	OR		1.71	SCRD	1	4.06			

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-40 — MEMOIRE 5 CYCLES

ACK		12.60	JAGE	>=0	2.29	ORI	+	2.67	SCRD	7	25.14
ACQ+ACTD		56.00	JAGE	A<0	2.29	ORI	-	2.53	SCRD	8	25.54
ACT		10.09	JAL	A<0	2.00	ORR		5.37	SCRD	15	51.45
ACT+WAIT		171.23	JAL	A>=0	2.55	PLR	1REG	9.11	SCRD	16	9.59
ACTD+ACQ		56.00	JALE	<=0	2.29	PLR	2REG	12.48	SCRD	23	20.38
AD		2.32	JALE	A>0	2.29	PLR	4REG	19.51	SCRD	24	20.82
ADCR		6.20	JANE	A#0	2.30	PSR	1REG	9.11	SCRD	31	32.08
ADR		5.53	JANE	A=0	2.30	PSR	2REG	12.48	SCRS	1	10.48
ADRI	+	4.13	JC	C=1	3.17	PSR	4REG	15.96	SCRS	7	13.10
ADRI	-	4.13	JC	C=0	3.13	PTY		5.82	SCRS	8	13.50
ADRP		5.94	JCV	CV=1	3.13	PULL		8.58	SCRS	15	10.39
AND		2.32	JCV	CV=0	3.13	PUSH		8.29	SCY		3.28
ANDI	+	2.69	JDX	X>0	2.07	QUIT+ARM		168.70	SFG		INEX
ANDI	-	2.54	JDX	X<=0	3.26	RBP		6.28	SIO	C	6.80
ANDR		5.34	JE	(EQ)	3.09	RBT	(A)	5.76	SIO	L	5.78
ARM		14.53	JE	(NE)	3.13	RBT	(B)	5.76	SLLD	1	52.08
ARM+QUIT		168.70	JG	(GT)	3.15	RBT	(AX)	6.06	SLLD	8	20.38
BR		2.67	JG	(LE)	3.40	RBT	(BX)	6.06	SLLD	9	25.96
BSR		5.09	JGE	(GE)	3.13	RBTM		INEX	SLLD	16	8.67
CLSR		5.37	JGE	(LT)	3.40	RCDA	1	INEX	SLLD	17	25.96
CMR		5.26	JIX	X<0	2.08	RCDA	2	INEX	SLLD	24	22.95
CP		2.43	JIX	X>=0	3.26	RCDA	5	INEX	SLLD	25	22.94
CPBY		2.81	JL	(LT)	3.17	RDHV		2.98	SLLD	31	20.23
CPI	+	2.66	JL	(GE)	3.15	RDOE		4.30	SLLS	1	14.96
CPI	-	2.52	JLE	(LE)	3.17	RDSI		2.98	SLLS	8	10.97
CPR		5.52	JLE	(GT)	3.13	RLSE		8.51	SLLS	9	10.77
CPZ		2.33	JMP		1.72	RLSERQST		179.44	SLLS	16	8.08
CPZR		5.67	JNC	C=0	3.15	RQST		9.01	SLG		INEX
DBP	Y=1	13.17	JNC	C=1	3.40	RQSTRLSE		179.44	SLRD	1	19.70
DBP	Y=2	19.38	JNCV	NCV	3.13	RSR		5.73	SLRD	7	24.75
DBT	(A)	2.96	JNCV	VC	3.40	RST		3.95	SLRD	8	25.14
DBT	(B)	3.83	JNE	(NE)	3.13	RSV		5.37	SLRD	15	51.45
DC		4.06	JNE	(EQ)	3.38	SARD	1	14.96	SLRD	16	9.24
DIT		3.31	JNV	V=0	3.13	SARD	7	25.14	SLRD	23	17.58
DRBM	M1	INEX	JNV	V=1	3.38	SARD	8	25.55	SLRD	24	17.80
DRBM	M2	INEX	JV	V=1	3.09	SARD	15	31.46	SLRD	31	20.78
DRBM	M3	INEX	JV	V=0	3.13	SARD	16	9.71	SLRS	1	13.78
DV		6.10	LA		2.31	SARD	23	17.81	SLRS	7	10.39
EIT		6.84	LAD		2.33	SARD	24	18.25	SLRS	8	10.58
EOR		2.32	LAI	+	2.68	SARD	31	21.07	SLRS	15	14.70
EORI	+	2.66	LAI	-	2.55	SARS	1	14.52	SST		3.09
EORI	-	2.52	LAR		4.15	SARS	7	16.97	STA		2.29
EORR		5.53	LB		2.33	SARS	8	17.38	STAR		4.16
FABS	+	4.33	LBI	+	2.68	SARS	15	20.23	STB		2.29
FABS	-	6.99	LBI	-	2.53	SB		2.54	STBY		3.43
FAD	PI+E	14.00	LBY	1	2.57	SBCR		6.23	STEP		31.99
FCAM		14.00	LR		4.95	SBP		11.61	STX		2.30
FCAZ		5.62	LRM	1REG	9.00	SBR		5.68	STY		2.30
FCMZ		8.97	LRM	2REG	12.34	SBS	1	7.14	STZ		2.30
FDV	PI/E	15.67	LRM	4REG	18.69	SBS	2	11.50	SUPG		INEX
FIX	0	5.25	LRP		5.79	SBS	3	16.06	SVC		6.07
FIX	PI	20.87	LX		2.31	SBS	4	20.51	SWBR		5.25
FLD		9.62	LXI	+	2.68	SBT	(A)	5.67	TBT	(A)	0.34
FLT		11.13	LXI	-	2.53	SBT	(B)	5.67	TBT	(B)	0.34
FMP	PI*E	14.00	LY		2.33	SBT	(AX)	5.90	TBT	(AX)	0.05
FNEG		6.01	LYI	+	2.68	SBT	(BX)	5.90	TBT	(BX)	0.05
FSB	PI-E	13.86	LYI	-	2.53	SBTM		INEX	WAIT		14.00
FST		10.13	MOVE	1	8.02	SBTM	(X)	INEX	WAIT+ACT		171.41
HALT	1	4.64	MOVE	2	11.46	SCLD	1	32.66	WCDA	1	INEX
HALT	2	6.65	MOVE	5	21.53	SCLD	8	26.82	WCDA	2	INEX
IBT	(A)	5.63	MP		5.44	SCLD	9	26.38	WUE		2.99
IBT	(B)	5.63	MVTM	1	10.80	SCLD	16	8.80	XIMR		10.03
IBT	(AX)	5.94	MVTM	2	15.52	SCLD	17	33.33	XM		3.26
IBT	(BX)	5.94	MVTM	5	30.63	SCLD	24	27.73	XR		0.07
IC		4.22	MVTS	1	10.92	SCLD	25	26.82	&		0.97
INSQ		INEX	MVTS	2	15.67	SCLD	31	21.97	&X		0.97
IPI		10.23	MVTS	5	30.87	SCLS	1	17.17	& FLOTAN		0.55
JAE	A=0	1.99	NGR		5.76	SCLS	8	14.21	&XFLOTAN		0.55
JAE	A#0	2.55	NOP		2.00	SCLS	9	13.78			
JAG	A>0	2.00	NORM		10.05	SCLS	16	8.67			
JAG	A<=0	2.55	OR		2.31	SCRD	1	19.96			

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-40 MEMOIRE 6 CYCLES

ACK	12.64	JAGE >=0	2.42	URI +	2.70	SCRD 7	25.04
ACQ+ACTD	58.50	JAGE A<0	2.42	URI -	2.55	SCRD 8	25.86
ACT	10.35	JAL A<0	2.13	URR	5.36	SCRD 15	31.96
ACT+WAIT	177.12	JAL A>=0	2.70	PLR 1REG	9.31	SCRD 16	9.53
ACTD+ACQ	58.50	JALE <=0	2.42	PLR 2REG	12.56	SCRD 23	20.29
AD	2.42	JALE A>0	2.42	PLR 4REG	19.36	SCRD 24	27.18
ADCR	6.25	JANE A#0	2.42	PSR 1REG	9.31	SCRD 31	33.24
ADR	5.57	JANE A=0	2.42	PSR 2REG	12.56	SCRS 1	10.07
ADRI +	4.27	JC C=1	3.25	PSR 4REG	16.05	SCRS 7	13.27
ADRI -	4.27	JC C=0	3.28	PTY	5.97	SCRS 8	13.68
ADRP	6.09	JCV CV=1	3.25	PULL	8.83	SCRS 15	16.88
AND	2.42	JCV CV=0	3.28	PUSH	8.38	SCY	5.41
ANDI +	2.70	JDX X>0	2.84	QUIT+ARM	172.08	SFW	INEX
ANDI -	2.56	JDX X<=0	3.41	RBP	6.55	SIO C	0.98
ANDR	5.41	JE (EQ)	3.26	RBT (A)	5.80	SIO L	5.72
ARM	15.05	JE (NE)	3.28	RBT (B)	5.80	SLLD 1	32.59
ARM+QUIT	172.08	JG (GT)	3.28	RBT (AX)	6.10	SLLD 8	20.73
BR	2.72	JG (LE)	3.56	RBT (BX)	6.10	SLLD 9	25.86
BSR	5.13	JGE (GE)	3.28	RBIM	INEX	SLLD 16	8.78
CLSR	5.40	JGE (LT)	3.56	RCDA 1	INEX	SLLD 17	25.86
CMR	5.25	JIX X<0	2.84	RCDA 2	INEX	SLLD 24	26.19
CP	2.43	JIX X>=0	3.41	RCDA 5	INEX	SLLD 25	22.85
CPBY	2.85	JL (LT)	3.25	RDMV	3.14	SLLD 31	20.40
CPI +	2.70	JL (GE)	3.28	RDOE	4.39	SLLS 1	20.13
CPI -	2.56	JLE (LE)	3.25	RDSI	3.14	SLLS 8	17.28
CPR	5.52	JLE (GT)	3.28	RLSE	8.00	SLLS 9	10.88
CPZ	2.43	JMP	1.85	RLSERQST	183.11	SLLS 16	8.24
CPZR	5.68	JNC C=0	3.28	RQST	8.98	SLW	INEX
DBP Y=1	13.64	JNC C=1	3.56	RQSTRLSE	183.11	SLRD 1	19.66
DBP Y=2	20.34	JNCV NCV	3.28	RSR	6.17	SLRD 7	25.04
DBT (A)	2.98	JNCV VC	3.54	RST	3.84	SLRD 8	25.45
DBT (B)	3.84	JNE (NE)	3.28	RSV	5.60	SLRD 15	31.36
DC	4.28	JNE (EQ)	3.55	SARD 1	20.13	SLRD 16	9.58
DIT	3.42	JNV V=0	3.28	SARD 7	25.45	SLRD 23	17.71
DRBM M1	INEX	JNV V=1	3.55	SARD 8	25.86	SLRD 24	17.93
DRBM M2	INEX	JV V=1	3.26	SARD 15	31.96	SLRD 31	20.97
DRBM M3	INEX	JV V=0	3.28	SARD 16	9.78	SLRS 1	13.97
DV	6.50	LA	2.42	SARD 23	18.15	SLRS 7	16.48
EIT	7.23	LAD	2.41	SARD 24	18.38	SLRS 8	16.88
EOR	2.43	LAI +	2.70	SARD 31	21.27	SLRS 15	19.86
EORI +	2.70	LAI -	2.57	SARS 1	14.73	SST	3.61
EORI -	2.56	LAR	4.42	SARS 7	17.28	STA	2.42
EORR	5.53	LB	2.43	SARS 8	17.49	STAR	4.25
FABS +	4.49	LBI +	2.70	SARS 15	20.40	STB	2.42
FABS -	7.03	LBI -	2.55	SB	2.61	STBY	3.41
FAD PI+E	14.11	LBY 1	2.60	SBCR	6.22	STEP	34.40
FCAM	14.11	LR	4.97	SBP	12.38	STX	2.42
FCAZ	5.47	LRM 1REG	9.43	SBR	5.65	STY	2.42
FCMZ	8.91	LRM 2REG	12.46	SBS 1	7.48	STZ	2.42
FDV PI/E	15.88	LRM 4REG	18.83	SBS 2	12.00	SUPQ	INEX
FIX 0	5.27	LRP	5.80	SBS 3	16.81	SVC	0.19
FIX PI	20.87	LX	2.43	SBS 4	21.16	SWBR	5.24
FLD	10.04	LXI +	2.70	SBT (A)	5.69	TBT (A)	0.50
FLT	11.39	LXI -	2.55	SBT (B)	5.69	TBT (B)	0.56
FMP PI*E	14.11	LY	2.43	SBT (AX)	5.93	TBT (AX)	0.07
FNEG	5.88	LYI +	2.70	SBT (BX)	5.93	TBT (BX)	0.07
FSB PI-E	14.11	LYI -	2.55	SBTM	INEX	WAIT	14.98
FST	10.38	MOVE 1	8.16	SBTM (X)	INEX	WAIT+ACT	177.49
HALT 1	4.54	MOVE 2	11.73	SCLD 1	33.24	WCDA 1	INEX
HALT 2	6.85	MOVE 5	22.24	SCLD 8	27.18	WCDA 2	INEX
IBT (A)	5.69	MP	6.35	SCLD 9	26.29	WOE	3.11
IBT (B)	5.69	MVTM 1	10.44	SCLD 16	9.00	XIMR	11.04
IBT (AX)	5.97	MVTM 2	15.57	SCLD 17	33.91	XM	3.38
IBT (BX)	5.97	MVTM 5	30.99	SCLD 24	27.64	XR	0.09
IC	4.43	MVTS 1	10.57	SCLD 25	27.18	&	0.99
INSQ	INEX	MVTS 2	15.72	SCLD 31	22.19	&X	1.00
IPI	10.86	MVTS 5	30.99	SCLS 1	17.28	& FLOTAN	0.70
JAE A=0	2.13	NGR	5.81	SCLS 8	14.42	&XFLOTAN	0.65
JAE A#0	2.70	NOP	2.14	SCLS 9	13.97		
JAG A>0	2.13	NORM	10.19	SCLS 16	8.44		
JAG A<=0	2.70	OR	2.43	SCRD 1	20.13		

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-30 — MEMOIRE 5 CYCLES

ACR		13.45	JAGE >=0	2.46	ORI +	3.01	SCRD 7	24.77
ACQ+ACTD		64.11	JAGE A<0	2.45	ORI -	2.85	SCRD 8	24.77
ACT		11.02	JAL A<0	2.14	ORR	5.91	SCRD 15	26.91
ACT+WAIT		191.89	JAL A>=0	2.75	PLR 1REG	10.98	SCRD 16	10.30
ACTD+ACQ		63.86	JALE <=0	2.45	PLR 2REG	14.65	SCRD 23	26.02
AD		2.59	JALE A>0	2.45	PLR 4REG	22.33	SCRD 24	26.02
ADCR		6.81	JANE A#0	2.46	PSR 1REG	10.80	SCRD 31	28.35
ADR		6.08	JANE A=0	2.46	PSR 2REG	15.65	SCRS 1	11.87
ADRI	+	4.36	JC C=1	3.47	PSR 4REG	19.27	SCRS 7	12.73
ADRI	-	4.36	JC C=0	3.49	PTY	6.66	SCRS 8	12.86
ADRP		6.51	JCV CV=1	3.52	PULL	9.50	SCRS 15	14.00
AND		2.42	JCV CV=0	3.49	PUSH	9.32	SCY	3.45
ANDI	+	3.01	JDX X>0	2.95	QUIT+ARM	187.89	SFQ	INEX
ANDI	-	2.85	JDX X<=0	3.51	RBP	7.19	SIO FC	6.65
ANDR		5.95	JE (EQ)	3.44	RBT (A)	6.65	SIO FL	6.05
ARM		16.41	JE (NE)	3.49	RBT (B)	6.65	SLLD 1	27.85
ARM+QUIT		187.89	JG (GT)	3.49	RBT (AX)	7.03	SLLD 8	25.59
BR		2.47	JG (LE)	3.73	RBT (BX)	7.03	SLLD 9	25.59
BSR		6.22	JGE (GE)	3.49	RBTM	INEX	SLLD 16	9.59
CLSR		5.92	JGE (LT)	3.73	RCDA 1	INEX	SLLD 17	24.77
CMR		5.74	JIX X<0	2.95	RCDA 2	INEX	SLLD 24	23.63
CP		2.78	JIX X>=0	3.50	RCDA 5	INEX	SLLD 25	23.63
CPBY		3.23	JL (LT)	3.47	RDMV	3.18	SLLD 31	22.92
CPI	+	3.01	JL (GE)	3.49	RDQE	4.24	SLLS 1	18.34
CPI	-	2.97	JLE (LE)	3.47	RDSI	3.18	SLLS 8	17.44
CPR		6.10	JLE (GT)	3.49	RLSE	9.02	SLLS 9	17.44
CPZ		2.42	JMP	1.77	RLSERQST	198.64	SLLS 16	9.34
CPZR		6.25	JNC C=0	3.49	RQST	9.58	SLQ	INEX
DBP	Y=1	13.58	JNC C=1	3.73	RQSTRLSE	198.64	SLRD 1	22.58
DBP	Y=2	20.73	JNCV NCV	3.49	RSR	6.36	SLRD 7	24.38
DBT	(A)	3.20	JNCV VC	3.79	RST	3.48	SLRD 8	24.38
DBT	(B)	4.04	JNE (NE)	3.49	RSV	6.13	SLRD 15	26.46
DC		4.57	JNE (EQ)	3.82	SARD 1	22.92	SLRD 16	10.21
DIT		3.51	JNV V=0	3.49	SARD 7	24.77	SLRD 23	17.88
DRBM	M1	INEX	JNV V=1	3.82	SARD 8	24.77	SLRD 24	17.88
DRBM	M2	INEX	JV V=1	3.44	SARD 15	26.91	SLRD 31	18.83
DRBM	M3	INEX	JV V=0	3.49	SARD 16	10.59	SLRS 1	15.65
DV		6.23	LA	2.48	SARD 23	18.34	SLRS 7	16.60
EIT		7.65	LAD	2.38	SARD 24	18.34	SLRS 8	16.60
EOR		2.42	LAI +	3.01	SARD 31	19.33	SLRS 15	17.66
EORI	+	3.00	LAI -	2.85	SARS 1	16.41	SST	3.06
EORI	-	2.97	LAR	4.42	SARS 7	17.44	STA	2.42
EORR		6.08	LB	2.42	SARS 8	17.44	STAR	4.60
FABS	+	4.84	LBI +	3.01	SARS 15	18.34	STB	2.42
FABS	-	8.34	LBI -	2.85	SB	2.95	STBY	4.24
FAD	PI+E	15.59	LBY	2.95	SBCR	6.81	STEP	35.02
FCAM		15.76	LR	5.53	SBP	12.96	STX	2.42
FCAZ		6.11	LRM 1REG	10.59	SBR	6.20	STY	2.42
FCMZ		9.64	LRM 2REG	13.89	SBS 1	8.12	STZ	2.42
FDV	PI/E	17.33	LRM 4REG	21.29	SBS 2	12.57	SUPQ	INEX
FIX	0	5.87	LRP	6.36	SBS 3	17.41	SVC	7.41
FIX	PI	20.94	LX	2.42	SBS 4	21.94	SWBR	5.74
FLD		10.80	LXI +	3.01	SBT (A)	6.55	TBT (A)	7.30
FLT		12.67	LXI -	2.85	SBT (B)	6.55	TBT (B)	7.31
FMP	PI*E	15.76	LY	2.42	SBT (AX)	6.87	TBT (AX)	7.63
FNEG		6.83	LYI +	3.01	SBT (BX)	6.87	TBT (BX)	7.63
FSB	PI-E	15.76	LYI -	2.85	SBTM	INEX	WAIT	15.87
FST		11.36	MOVE 1	8.25	SBTM (X)	INEX	WAIT+ACT	191.89
HALT	1	4.37	MOVE 2	12.00	SCLD 1	28.35	WCDA 1	INEX
HALT	2	6.65	MOVE 5	22.50	SCLD 8	20.02	WCDA 2	INEX
IBT	(A)	6.56	MP	6.35	SCLD 9	20.02	WOE	3.19
IBT	(B)	6.56	MVTM 1	10.98	SCLD 16	9.76	XIMR	12.01
IBT	(AX)	6.87	MVTM 2	16.88	SCLD 17	28.86	XM	3.52
IBT	(BX)	6.86	MVTM 5	33.69	SCLD 24	26.91	XR	6.65
IC		4.56	MVTS 1	11.12	SCLD 25	26.91	&	1.08
INSQ		INEX	MVTS 2	16.88	SCLD 31	25.18	&X	1.08
IPI		9.59	MVTS 5	33.69	SCLS 1	14.79	& FLUTAN	0.89
JAE	A=0	2.14	NGR	6.35	SCLS 8	13.70	&XFLUTAN	0.89
JAE	A#0	2.75	NOP	2.11	SCLS 9	13.55		
JAG	A>0	2.13	NORM	12.22	SCLS 16	9.51		
JAG	A<=0	2.75	OR	2.48	SCRD 1	22.92		

OPTION : FFP16

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-30 — MEMOIRE 5 CYCLES

ACK		13.45	JAGE	>=0	2.46	ORI	+	3.01	SCRD	7	24.77
ACQ+ACTD		64.11	JAGE	A<0	2.46	ORI	-	2.85	SCRD	8	24.77
ACT		11.02	JAL	A<0	2.13	ORR		5.97	SCRD	15	26.91
ACT+WAIT		191.89	JAL	A>=0	2.75	PLR	1REG	10.98	SCRD	16	10.30
ACTD+ACQ		63.86	JALE	<=0	2.45	PLR	2REG	14.65	SCRD	23	26.02
AD		2.59	JALE	A>0	2.46	PLR	4REG	22.58	SCRD	24	26.02
ADCR		6.81	JANE	A#0	2.45	PSR	1REG	10.80	SCRD	31	28.35
ADR		6.04	JANE	A=0	2.46	PSR	2REG	15.85	SCRS	1	11.87
ADRI	+	4.36	JC	C=1	3.47	PSR	4REG	19.27	SCRS	7	12.73
ADRI	-	4.36	JC	C=0	3.49	PTY		6.65	SCRS	8	12.86
ADRP		6.51	JCV	CV=1	3.52	PULL		9.50	SCRS	15	14.00
AND		2.42	JCV	CV=0	3.49	PUSH		9.32	SCY		3.45
ANDI	+	3.01	JDX	X>0	2.95	QUIT+ARM		187.89	SFQ		INEX
ANDI	-	2.85	JDX	X<=0	3.51	RBP		7.19	SIO	FC	6.65
ANUR		5.96	JE	(EQ)	3.44	RBT	(A)	6.65	SIO	FL	5.86
ARM		16.41	JE	(NE)	3.49	RBT	(B)	6.65	SLLD	1	27.85
ARM+QUIT		187.89	JG	(GT)	3.49	RBT	(AX)	7.03	SLLD	8	25.59
BR		2.47	JG	(LE)	3.73	RBT	(BX)	7.03	SLLD	9	25.59
BSR		6.22	JGE	(GE)	3.49	RBTM		INEX	SLLD	16	9.59
CLSR		5.91	JGE	(LT)	3.73	RCDA	1	INEX	SLLD	17	24.77
CMR		5.73	JIX	X<0	2.96	RCDA	2	INEX	SLLD	24	23.63
CP		2.78	JIX	X>=0	3.50	RCDA	5	INEX	SLLD	25	23.63
CPBY		3.23	JL	(LT)	3.47	RDHV		3.18	SLLD	31	22.92
CPI	+	3.02	JL	(GE)	3.49	RDOE		4.24	SLLS	1	18.34
CPI	-	2.98	JLE	(LE)	3.47	RDSI		3.18	SLLS	8	17.44
CPR		6.11	JLE	(GT)	3.49	RLSE		9.02	SLLS	9	17.44
CPZ		2.42	JMP		1.77	RLSERQST		198.64	SLLS	16	9.34
CPZR		6.25	JNC	C=0	3.49	RQST		9.38	SLQ		INEX
DBP	Y=1	13.58	JNC	C=1	3.73	RQSTRLSE		198.64	SLRD	1	22.58
DBP	Y=2	20.73	JNCV	NCV	3.49	RSR		6.30	SLRD	7	24.38
DBT	(A)	3.20	JNCV	VC	3.79	RST		3.48	SLRD	8	24.38
DBT	(B)	4.04	JNE	(NE)	3.49	RSV		5.93	SLRD	15	26.46
DC		4.57	JNE	(EQ)	3.82	SARD	1	22.92	SLRD	16	10.21
DIT		3.51	JNV	V=0	3.49	SARD	7	24.77	SLRD	23	17.88
DRBM	M1	INEX	JNV	V=1	3.82	SARD	8	24.77	SLRD	24	17.88
DRBM	M2	INEX	JV	V=1	3.44	SARD	15	26.91	SLRD	31	18.83
DRBM	M3	INEX	JV	V=0	3.49	SARD	16	10.59	SLRS	1	15.65
DV		64.97	LA		2.47	SARD	23	18.34	SLRS	7	16.60
EIT		7.65	LAD		2.38	SARD	24	18.34	SLRS	8	16.60
EOR		2.42	LAI	+	3.01	SARD	31	19.33	SLRS	15	17.66
EORI	+	3.02	LAI	-	2.85	SARS	1	16.41	SST		3.06
EORI	-	2.98	LAR		4.42	SARS	7	17.44	STA		2.42
EORR		6.08	LB		2.42	SARS	8	17.44	STAR		4.61
FABS	+	4.84	LBI	+	3.01	SARS	15	18.34	STB		2.42
FABS	-	8.34	LBI	-	2.85	SB		2.95	STBY		4.24
FAD	PI+E	84.84	LBY		2.95	SBCR		6.81	STEP		35.02
FCAM		46.73	LR		5.49	SBP		12.96	STX		2.42
FCAZ		6.11	LRM	1REG	10.49	SBR		6.25	STY		2.42
FCMZ		9.64	LRM	2REG	13.89	SBS	1	8.12	STZ		2.42
FDV	PI/E	582.45	LRM	4REG	21.14	SBS	2	12.57	SUPQ		INEX
FIX	0	5.87	LRP		6.36	SBS	3	17.41	SVC		7.41
FIX	PI	20.94	LX		2.42	SBS	4	21.94	SWBR		5.74
FLD		10.80	LXI	+	3.01	SBT	(A)	6.55	TBT	(A)	7.31
FLT		51.59	LXI	-	2.85	SBT	(B)	6.55	TBT	(B)	7.31
FMP	PI*E	457.45	LY		3.42	SBT	(AX)	6.87	TBT	(AX)	7.63
FNEG		6.83	LYI	+	3.01	SBT	(BX)	6.87	TBT	(BX)	7.63
FSB	PI-E	104.51	LYI	-	2.85	SBTM		INEX	WAIT		15.87
FST		11.36	MOVE	1	8.25	SBTM	(X)	INEX	WAIT+ACT		191.89
HALT	1	4.37	MOVE	2	12.00	SCLD	1	28.35	WCDA	1	INEX
HALT	2	6.65	MOVE	5	22.50	SCLD	8	26.02	WCDA	2	INEX
IBT	(A)	6.56	MP		67.58	SCLD	9	20.02	WUE		3.20
IBT	(B)	6.56	MVTM	1	10.98	SCLD	16	9.76	XIMR		12.01
IBT	(AX)	6.87	MVTM	2	16.88	SCLD	17	28.86	XM		3.52
IBT	(BX)	6.87	MVTM	5	33.69	SCLD	24	26.91	XR		6.65
IC		4.56	MVTS	1	11.12	SCLD	25	26.91	&		1.08
INSQ		INEX	MVTS	2	16.88	SCLD	31	25.18	&X		1.09
IPI		9.59	MVTS	5	33.98	SCLS	1	14.79	& FLUTAN		0.89
JAE	A=0	2.14	NGR		6.34	SCLS	8	13.70	&XFLUTAN		0.89
JAE	A#0	2.75	NOP		2.11	SCLS	9	13.55			
JAG	A>0	2.14	NORM		277.46	SCLS	16	9.51			
JAG	A<=0	2.75	OR		2.49	SCRD	1	22.92			

OPTION : FLOTTANT MICROPROGRAMME

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-05

MEMOIRE 8 CYCLES

ACK		21.95	JAGE	>=0	5.99	ORI	+	7.36	SCRD	7	20.46
ACQ+ACTD		125.46	JAGE	A<0	6.13	ORI	-	7.36	SCRD	8	23.01
ACT		17.32	JAL	A<0	6.35	ORR		7.07	SCRD	15	20.99
ACT+WAIT		259.87	JAL	A>=0	5.85	PLR	1REG	17.49	SCRD	16	25.95
ACTD+ACQ		125.46	JALE	<=0	6.51	PLR	2REG	21.79	SCRD	23	31.24
AD		6.40	JALE	A>0	6.13	PLR	4REG	30.25	SCRD	24	28.16
ADCR		7.81	JANE	A#0	6.13	PSR	1REG	11.66	SCRD	31	31.24
ADR		7.63	JANE	A=0	6.28	PSR	2REG	17.49	SCRS	1	16.95
ADRI	+	5.83	JC	C=1	5.63	PSR	4REG	21.23	SCRS	7	16.01
ADRI	-	5.83	JC	C=0	5.58	PTY		8.99	SCRS	8	16.43
ADRP		8.58	JCV	CV=1	5.89	PULL		15.09	SCRS	15	16.01
AND		6.25	JCV	CV=0	5.38	PUSH		17.01	SCY		8.00
ANDI	+	7.36	JDX	X>0	6.74	QUIT+ARM		254.58	SFG		INEX
ANDI	-	7.36	JDX	X<=0	6.90	RBP		9.71	SIO	FC	10.44
ANDR		7.69	JE	(EQ)	5.58	RBT	(A)	11.44	SIO	FL	9.11
ARM		20.52	JE	(NE)	5.56	RBT	(B)	11.59	SLLD	1	20.99
ARM+QUIT		254.58	JG	(GT)	5.58	RBT	(AX)	11.92	SLLD	8	24.05
BR		5.93	JG	(LE)	5.63	RBT	(BX)	12.06	SLLD	9	20.99
BSR		7.84	JGE	(GE)	5.51	RBTM		INEX	SLLD	16	25.95
CLSR		7.67	JGE	(LT)	5.48	RCDA	1	INEX	SLLD	17	25.95
CMR		7.09	JIX	X<0	6.66	RCDA	2	INEX	SLLD	24	22.77
CP		6.35	JIX	X>=0	6.74	RCDA	5	INEX	SLLD	25	25.45
CPBY		7.24	JL	(LT)	5.63	RDMV		7.27	SLLD	31	23.19
CPI	+	7.51	JL	(GE)	5.56	RDOE		7.54	SLLS	1	20.52
CPI	-	7.51	JLE	(LE)	5.87	RDSI		8.80	SLLS	8	19.20
CPR		7.87	JLE	(GT)	5.36	RLSE		13.93	SLLS	9	20.32
CPZ		6.86	JMP		5.25	RLSERQST		289.50	SLLS	16	19.84
CPZR		7.87	JNC	C=0	5.51	RQST		15.78	SLQ		INEX
DBP	Y=1	22.82	JNC	C=1	5.48	RQSTRLSE		289.50	SLRD	1	24.05
DBP	Y=2	34.99	JNCV	NCV	5.63	RSR		9.18	SLRD	7	20.46
DBT	(A)	12.00	JNCV	VC	5.57	RST		12.11	SLRD	8	23.01
DBT	(B)	13.49	JNE	(NE)	5.51	RSV		INEX	SLRD	15	20.99
DC		8.19	JNE	(EQ)	5.51	SARD	1	25.95	SLRD	16	25.95
DIT		8.45	JNV	V=0	5.49	SARD	7	28.09	SLRD	23	20.46
DRBM	M1	INEX	JNV	V=1	5.51	SARD	8	25.45	SLRD	24	23.01
DRBM	M2	INEX	JV	V=1	5.65	SARD	15	28.69	SLRD	31	20.46
DRBM	M3	INEX	JV	V=0	5.63	SARD	16	27.54	SLRS	1	18.59
DV		101.09	LA		5.85	SARD	23	28.10	SLRS	7	19.84
EIT		11.30	LAD		5.97	SARD	24	25.45	SLRS	8	18.50
EOR		6.25	LAI	+	7.02	SARD	31	28.69	SLRS	15	19.84
EORI	+	7.42	LAI	-	7.02	SARS	1	20.52	SST		13.50
EORI	-	7.42	LAR		8.79	SARS	7	21.60	STA		8.03
EORR		7.63	LB		5.83	SARS	8	20.18	STAR		9.57
FABS	+	INEX	LBI	+	7.02	SARS	15	21.60	STB		8.63
FABS	-	INEX	LBI	-	7.02	SB		7.02	STBY		7.72
FAD	PI+E	INEX	LBY		8.47	SBCR		7.81	STEP		71.00
FCAM		INEX	LR		8.78	SBP		9.34	STX		8.65
FCAZ		INEX	LRM	1REG	15.56	SBR		7.67	STY		8.03
FCMZ		INEX	LRM	2REG	19.85	SBS	1	15.01	STZ		7.19
FDV	PI/E	INEX	LRM	4REG	28.61	SBS	2	20.87	SUPG		INEX
FIX	0	INEX	LRP		8.86	SBS	3	29.25	SVC		9.58
FIX	PI	INEX	LX		5.83	SBS	4	34.66	SWBR		7.27
FLD		INEX	LXI	+	7.02	SBT	(A)	11.44	TBT	(A)	12.71
FLT		INEX	LXI	-	7.02	SBT	(B)	11.59	TBT	(B)	12.91
FMP	PI*E	INEX	LY		5.83	SBT	(AX)	11.92	TBT	(AX)	13.11
FNEG		INEX	LYI	+	7.02	SBT	(BX)	12.06	TBT	(BX)	13.32
FSB	PI-E	INEX	LYI	-	7.02	SBTM		INEX	WAIT		22.72
FST		INEX	MOVE	1	17.12	SBTM	(X)	INEX	WAIT+ACT		259.87
HALT	1	9.56	MOVE	2	21.99	SCLD	1	28.10	WCDA	1	INEX
HALT	2	11.53	MOVE	5	36.00	SCLD	8	24.97	WCDA	2	INEX
IBT	(A)	11.47	MP		98.34	SCLD	9	28.10	WDE		8.47
IBT	(B)	11.63	MVTM	1	17.12	SCLD	16	26.99	XIMR		12.34
IBT	(AX)	11.94	MVTM	2	21.70	SCLD	17	31.24	XM		8.03
IBT	(BX)	12.10	MVTM	5	36.00	SCLD	24	28.10	XR		8.39
IC		8.14	MVTS	1	17.12	SCLD	25	31.24	&		2.01
INSQ		INEX	MVTS	2	21.70	SCLD	31	28.09	&X		3.16
IPI		36.76	MVTS	5	36.00	SCLS	1	16.89	& FLUTAN		INEX
JAE	A=0	6.51	NGR		7.07	SCLS	8	17.73	&XFLUTAN		INEX
JAE	A#0	5.99	NOP		4.02	SCLS	9	16.89			
JAG	A>0	6.28	NORM		INEX	SCLS	16	16.01			
JAG	A<=0	6.35	OR		6.28	SCRD	1	24.05			

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-04 — MEMOIRE 8 CYCLES

ACK		23.19	JAGE	>=0	0.13	ORI	+	7.54	SCRO	7	20.02
ACQ+ACTD		128.00	JAGE	A<0	0.29	ORI	-	7.54	SCRO	8	23.08
ACT		18.06	JAL	A<0	0.52	ORR		7.84	SCRO	15	20.82
ACT+WAIT		271.02	JAL	A>=0	5.99	PLR	1REG	17.59	SCRO	16	20.29
ACTD+ACW		128.00	JALE	<=0	0.00	PLR	2REG	22.12	SCRO	23	31.07
AD		0.09	JALE	A>0	0.29	PLR	4REG	31.07	SCRO	24	20.51
ADCR		7.92	JANE	A#0	0.29	PSR	1REG	11.94	SCRO	31	31.77
ADR		7.82	JANE	A=0	0.44	PSR	2REG	17.59	SCRS	1	17.02
ADRI	+	0.01	JC	C=1	5.09	PSR	4REG	22.12	SCRS	7	18.13
ADRI	-	0.01	JC	C=0	5.79	PTY		9.25	SCRS	8	10.76
ADRP		8.79	JCV	CV=1	0.00	PULL		14.87	SCRS	15	18.13
AND		0.53	JCV	CV=0	5.31	PUSH		17.09	SCY		3.21
ANDI	+	7.54	JDX	X>0	0.94	QUIT+ARM		259.47	SFG		INEX
ANDI	-	7.54	JDX	X<=0	7.02	RBP		9.01	SIO	FC	10.01
ANDR		7.85	JE	(EQ)	5.81	RBT	(A)	11.73	SIO	FL	9.20
ARM		20.70	JE	(NE)	5.79	RBT	(B)	11.73	SLLD	1	27.36
ARM+QUIT		258.92	JG	(GT)	5.72	RBT	(AX)	12.21	SLLD	8	24.33
BR		8.31	JG	(LE)	5.09	RBT	(BX)	12.21	SLLD	9	27.36
BSR		8.31	JGE	(GE)	5.05	RBTM		INEX	SLLD	16	28.29
CLSR		7.84	JGE	(LT)	5.81	RCDA	1	INEX	SLLD	17	25.78
CMR		7.25	JIX	X<0	0.77	RCDA	2	INEX	SLLD	24	23.02
CP		6.69	JIX	X>=0	6.94	RCDA	5	INEX	SLLD	25	25.78
CPBY		7.56	JL	(LT)	5.09	RDHV		7.45	SLLD	31	23.44
CPI	+	7.66	JL	(GE)	5.79	RDUE		7.73	SLLS	1	20.70
CPI	-	7.66	JLE	(LE)	6.02	RDSI		8.90	SLLS	8	19.35
CPR		8.06	JLE	(GT)	5.51	RLSE		14.85	SLLS	9	20.70
CPZ		7.19	JMP		5.33	RLSERQST		295.17	SLLS	16	20.00
CPZR		8.08	JNC	C=0	5.65	RQST		10.43	SLG		INEX
DBP	Y=1	22.62	JNC	C=1	5.01	RQSTRLSE		295.17	SLRO	1	24.33
DBP	Y=2	34.34	JNCV	NCV	5.72	RSR		10.27	SLRO	7	20.82
DBT	(A)	12.11	JNCV	VC	5.75	RST		12.16	SLRO	8	23.08
DBT	(B)	13.65	JNE	(NE)	5.65	RSV		INEX	SLRO	15	20.82
DC		8.63	JNE	(EQ)	5.65	SARD	1	20.29	SLRO	16	20.29
DIT		8.60	JNV	V=0	5.65	SARD	7	28.51	SLRO	23	20.82
DRBM	M1	INEX	JNV	V=1	5.65	SARD	8	25.76	SLRO	24	23.08
DRBM	M2	INEX	JV	V=1	5.79	SARD	15	29.12	SLRO	31	20.02
DRBM	M3	INEX	JV	V=0	5.79	SARD	16	27.93	SLRS	1	18.72
DV		101.35	LA		0.13	SARD	23	28.51	SLRS	7	20.00
EIT		11.42	LAD		0.50	SARD	24	25.76	SLRS	8	18.42
EOR		6.53	LAI	+	7.19	SARD	31	28.51	SLRS	15	20.00
EORI	+	7.56	LAI	-	7.19	SARS	1	20.70	SST		15.05
EORI	-	7.56	LAR		8.90	SARS	7	21.81	STA		0.33
EORR		7.82	LB		6.15	SARS	8	20.35	STAR		4.25
FABS	+	INEX	LBI	+	7.19	SARS	15	21.81	STB		0.33
FABS	-	INEX	LBI	-	7.19	SB		7.27	STBY		7.54
FAD	PI+E	INEX	LBY		6.77	SBCR		7.92	STEP		70.11
FCAM		INEX	LR		0.93	SBP		9.11	STX		0.33
FCAZ		INEX	LRM	1REG	10.00	SBR		7.84	STY		0.33
FCMZ		INEX	LRM	2REG	20.02	SBS	1	16.14	STZ		7.10
FDV	PI/E	INEX	LRM	4REG	29.19	SBS	2	21.15	SUPG		INEX
FIX	U	INEX	LRP		7.10	SBS	3	29.32	SVC		4.39
FIX	PI	INEX	LX		0.15	SBS	4	34.01	SWBR		7.45
FLD		INEX	LXI	+	7.19	SBT	(A)	11.73	TBI	(A)	12.02
FLT		INEX	LXI	-	7.19	SBT	(B)	11.73	TBI	(B)	13.03
FMP	PI*E	INEX	LY		6.15	SBT	(AX)	12.21	TBT	(AX)	13.24
FNEG		INEX	LYI	+	7.19	SBT	(BX)	12.21	TBT	(BX)	13.45
FSB	PI-E	INEX	LYI	-	7.19	SBTM		INEX	WAIT		25.36
FST		INEX	MOVE	1	17.57	SBTM	(X)	INEX	WAIT+ACT		271.02
HALT	1	9.96	MOVE	2	22.99	SCLD	1	28.51	XCDA	1	INEX
HALT	2	12.07	MOVE	5	38.00	SCLD	8	25.20	XCDA	2	INEX
IBT	(A)	11.72	MP		98.09	SCLD	9	27.93	XUE		0.07
IBT	(B)	11.72	MVTM	1	17.57	SCLD	16	27.36	XTMR		12.55
IBT	(AX)	12.21	MVTM	2	22.99	SCLD	17	31.77	XM		0.33
IBT	(BX)	12.21	MVTM	5	38.00	SCLD	24	28.51	XR		0.03
IC		8.35	MVTS	1	17.57	SCLD	25	31.07	X		2.22
INSW		INEX	MVTS	2	22.99	SCLD	31	29.12	XX		3.48
IPI		37.51	MVTS	5	38.00	SCLS	1	19.03	X FLOTAN		INEX
JAE	A=0	0.00	NGR		7.84	SCLS	8	17.84	XXFLOTAN		INEX
JAE	A#0	0.13	NOP		4.50	SCLS	9	19.03			
JAG	A>0	0.44	NORM		INEX	SCLS	16	18.42			
JAG	A<=0	0.52	OR		0.52	SCRD	1	24.33			

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-90 HORLOGE 140 NS MEMOIRE 3 CYCLES

ACK		5.46	JAGE	>=0	0.34	CRI	+	0.42	SCRD	7	6.75								
ACQ+ACTD		26.83	JAGE	A<0	0.98	CRI	-	0.42	SCRD	8	6.55								
ACT		3.93	JAL	A<0	0.98	CRR		0.56	SCRD	15	7.63								
ACT+WAIT		65.43	JAL	A>=0	0.98	PLR	1REG	1.97	SCRD	16	7.65								
ACTD+ACQ		26.83	JALE	<=0	0.84	PLR	2REG	2.95	SCRD	23	7.44								
AD		0.84	JALE	A>0	0.98	PLR	4REG	4.91	SCRD	24	7.43								
ADCR		0.84	JANE	A#0	0.84	PSR	1REG	2.24	SCRD	31	8.22								
ADR		0.56	JANE	A=0	0.98	PSR	2REG	3.22	SCRS	1	3.79								
ADRI	+	0.42	JC	C=1	0.84	PSR	4REG	4.21	SCRS	7	6.31								
ADRI	-	0.42	JC	C=0	0.98	PTY		1.68	SCRS	8	6.65								
ADRP		0.84	JCV	CV=1	0.84	PULL		4.21	SCRS	15	6.63								
AND		0.84	JCV	CV=0	0.98	PUSH		4.07	SCY		1.68								
ANDI	+	0.42	JDX	X>0	0.84	QUIT+ARM		65.59	SFBR		3.23								
ANDI	-	0.42	JDX	X<=0	0.98	RBP		3.23	SHLT	FC	2.55								
ANDR		0.56	JE	(EQ)	0.84	RBT	(A)	0.84	SHLT	FL	2.55								
ARM		5.49	JE	(NE)	0.98	RBT	(B)	0.84	SLLD	1	4.63								
ARM+QUIT		65.59	JG	(GT)	0.94	RBT	(AX)	0.84	SLLD	8	7.74								
BR		0.84	JGE	(GE)	0.84	RBT	(BX)	0.84	SLLD	9	7.74								
BSR		1.26	JGE	(GE)	0.84	RBTM		3.37	SLLD	16	3.79								
CLSR		0.56	JGE	(LT)	0.98	RCDA	1	7.72	SLLD	17	3.79								
CMR		0.42	JIX	X<0	0.84	RCDA	2	9.54	SLLD	24	4.35								
CP		0.84	JIX	X>=0	0.98	RCDA	5	15.00	SLLD	25	7.45								
CPBY		1.12	JL	(LT)	0.84	RDHV		1.97	SLLD	31	4.07								
CPPI	+	0.42	JL	(GE)	0.84	RDCE		1.82	SLLS	1	6.33								
CPPI	-	0.42	JLE	(LE)	0.84	RDSI		2.38	SLLS	8	3.07								
CPR		0.56	JLE	(GT)	0.98	RLSS		3.51	SLLS	9	6.04								
CPZ		0.84	JMP		0.42	RLSSERQST		67.80	SLLS	16	3.08								
CPZR		0.42	JNC	C=0	0.84	RQST		3.93	SLRD		3.08								
DBP	Y=1	5.05	JNC	C=1	0.98	RQSTRLSE		67.80	SLRD	1	3.37								
DBP	Y=2	6.87	JNCV	NCV	0.84	RSR		2.10	SLRD	7	6.75								
DBT	(A)	1.96	JNCV	VC	0.98	RST		2.10	SLRD	8	3.65								
DBT	(B)	2.24	JNE	(NE)	0.84	RSV		2.24	SLRD	15	7.63								
DC		1.26	JNE	(EQ)	0.98	SARD	1	3.65	SLRD	16	3.65								
DIT		1.94	JNV	V=0	0.84	SARD	7	7.02	SLRD	23	7.44								
DR8M	M1	3.51	JNV	V=1	0.98	SARD	8	3.92	SLRD	24	4.35								
DR8M	M2	4.91	JV	V=1	0.84	SARD	15	7.85	SLRD	31	8.22								
DR8M	M3	6.31	JV	V=0	0.98	SARD	16	3.92	SLRS	1	3.51								
DV		3.09	LA		0.84	SARD	23	7.74	SLRS	7	6.51								
EIT		3.37	LAD		0.84	SARD	24	4.63	SLRS	8	3.31								
EOR		0.84	LAI	+	0.42	SARD	31	8.55	SLRS	15	6.33								
EORI	+	0.42	LAI	-	0.42	SARS	1	4.07	SST		1.96								
EORI	-	0.42	LAR		2.24	SARS	7	6.51	STA		0.98								
EORR		0.56	LB		0.84	SARS	8	3.92	STAR		2.25								
FABS	+	1.96	LBH	+	0.42	SARS	15	6.37	STB		0.98								
FABS	-	2.81	LBH	-	0.42	SBR	CR	0.84	STBY		1.26								
FADD	PI+E	4.77	LBY		0.98	SBP		4.09	STEP	16	1.08								
FCAM		3.93	LCR		0.42	SBR		0.56	STX		0.98								
FCAZ		1.83	LCRM	1REG	2.24	SBS		3.51	STY		0.98								
FCMZ		2.25	LCRM	2REG	2.94	SBS	1	4.49	STZ		0.98								
FDV	PI/E	6.74	LCRM	4REG	4.35	SBS	2	7.00	SUPQ		3.06								
FIX	0	2.11	LCRP		0.70	SBS	3	7.99	SVC		2.10								
FIX	PI	5.61	LCX		0.34	SBS	4	7.99	SWBR		0.42								
FLD		2.24	LXI	+	0.42	SBT	(A)	0.84	TBT	(A)	0.84								
FLT		3.93	LXI	-	0.42	SBT	(B)	0.84	TBT	(B)	0.84								
FMP	PI*E	5.75	LY		0.84	SBT	(AX)	0.84	TBT	(AX)	0.84								
FNEG		2.53	LYI	+	0.42	SBT	(BX)	0.84	TBT	(BX)	0.84								
FSB	PI-E	4.77	LYI	-	0.42	SBTM		3.36	WAIT		4.77								
FST		3.09	MCVE		3.23	SBTM	(X)	3.23	WAIT+ACT	65.45	5.92								
HALT	1	2.11	MCVE		4.35	SCLD	1	3.55	WCDA	1	7.92								
HALT	2	4.35	MCVE	5	7.75	SCLD	8	4.63	WCDA	2	9.82								
IBT	(A)	0.84	MP		2.55	SCLD	16	7.74	WCDE		2.25								
IBT	(B)	0.84	MVTM	1	3.37	SCLD	17	8.79	XMR		1.12								
IBT	(AX)	0.84	MVTM	2	4.49	SCLD	24	8.29	XM		1.12								
IBT	(BX)	0.84	MVTM	5	7.87	SCLD	25	4.35	XR		0.70								
IC		1.25	MVTS	1	3.35	SCLD	31	7.45	X		0.42								
INSQ		3.06	MVTS	2	4.48	SCLS	1	4.07	XX		0.42								
IPI		3.79	MVTS	5	7.87	SCLS	8	6.31	XFLOTAN		0.42								
JAE	A=0	0.84	NGR		0.56	SCLS	9	3.37	XXFLOTAN		0.42								
JAE	A#0	0.98	NOP		0.70	SCLS	16	6.04											
JAG	A>0	0.84	NORM		3.93	SCLS	16	3.08											
JAG	A<=0	0.98	OR		0.84	SCRD	1	3.37											

TEMPS MOYEN D'EXECUTION DES INSTRUCTIONS (MICROSECONDES)

SOLAR 16-70 HORLOGE 140 NS MEMOIRE 5 CYCLES

ACK	6.30	JAGE >=0	0.86	ORI +	0.74	SCRD 7	6.76				
ACQ+ACTD	31.30	JAGE A<0	1.45	ORI -	0.74	SCRD 8	3.97				
ACT	4.64	JAL A<0	0.86	ORR	0.73	SCRD 15	7.60				
ACT+WAIT	80.76	JAL A>=0	1.45	PLR 1REG	2.53	SCRD 16	3.97				
ACTD+ACQ	31.26	JALE <=0	0.86	PLR 2REGG	3.80	SCRD 23	7.48				
AD	1.47	JALE A>0	1.45	PLR 4REGG	5.37	SCRD 24	4.65				
ADCR	0.87	JANE A#0	0.86	PSR 1REGG	2.53	SCRD 31	8.30				
ADR	0.74	JANE A=0	1.45	PSR 2REGG	3.60	SCRRS 1	3.85				
ADRI +	0.74	JC C=1	0.86	PSR 4REG	4.58	SCRRS 7	6.35				
ADRI -	0.74	JC C=0	1.47	PTY	1.98	SCRRS 8	3.97				
ADRP	1.16	JCV CV=1	0.86	PULL	5.00	SCRRS 15	6.62				
AND	1.46	JCV CV=0	1.47	PUSH	5.45	SCRY	2.01				
ANDI +	0.74	JDX X>0	0.86	QUIT+ARM	81.09	SFBO	5.65				
ANDI -	0.74	JDX X<=0	1.45	RBP	3.86	SIID EC	3.62				
ANDR	0.73	JEE (EQ)	0.88	RBT (A)	0.87	SIID EL	2.85				
ARM	6.82	JE (NE)	1.47	RBT (B)	0.86	SLLDD 1	8.59				
ARM+QUIT	81.13	JG (GT)	0.88	RBT (AX)	0.86	SLLDD 8	4.94				
BR	1.54	JGE (LE)	1.45	RBT (BX)	0.86	SLLDD 9	7.77				
BSR	2.22	JGE (GE)	0.88	RBTM	4.52	SLLDD 16	4.06				
CLSR	0.73	JGE (LT)	1.45	RCDA 1	8.87	SLLDD 17	8.30				
CMR	0.73	JIX X<0	0.86	RCDA 2	10.85	SLLDD 24	4.65				
CP	1.45	JIX X>=0	1.45	RCDA 5	17.31	SLLDD 25	7.48				
CPBY	1.45	JL (LT)	0.86	RDHV	2.26	SLLDD 31	4.06				
CPI +	0.73	JL (GE)	1.47	RDCE	2.15	SLLSS 1	6.35				
CPI -	0.73	JLE (LE)	0.86	RDSS	2.41	SLLSS 8	3.68				
CPR	0.73	JLE (GT)	1.47	RLSS	4.17	SLLSS 9	6.03				
CPZ	1.47	JMP	0.74	RLSS RQST	85.36	SLLS 16	3.39				
CPZR	0.73	JNC C=0	0.88	RQST	4.52	SLRD	6.68				
DBP Y=1	5.95	JNC C=1	1.45	RQSTRLSE	85.36	SLRRDD 1	3.38				
DBP Y=2	8.09	JNCV NCV	0.88	RSR	2.64	SLRRDD 7	6.76				
DBT (A)	1.98	JNCV VC	1.45	RST	2.43	SLRRDD 8	3.97				
DBT (B)	2.26	JNE (NE)	0.88	RSV	2.75	SLRRDD 15	7.60				
DC	2.33	JNE (EQ)	1.47	SARD 1	3.67	SLRRDD 16	3.97				
DIT	2.13	JNV V=0	0.88	SARD 7	7.03	SLRRDD 23	7.48				
DRBM M1	3.95	JNV V=1	1.47	SARD 8	4.25	SLRRDD 24	4.65				
DRBM M2	5.69	JV V=1	0.88	SARD 15	7.89	SLRRDD 31	8.30				
DRBM M3	7.34	JV V=0	1.47	SARD 16	4.26	SLRRS 1	3.56				
DV	3.71	LA	1.45	SARD 23	7.72	SLRRS 7	6.03				
EIT	3.84	LAD	1.15	SARD 24	4.94	SLRRS 8	3.68				
EOR	1.47	LAI +	0.74	SARD 31	8.59	SLRRS 15	6.34				
EORI +	0.73	LAI -	0.74	SARS 1	4.07	SST	2.25				
EORI -	0.73	LAR	2.86	SARS 7	6.62	SSTA	1.75				
EORR	0.74	LBS	1.47	SARS 8	4.26	SSTAR	2.97				
FABS +	2.26	LBBI +	0.74	SARS 15	6.89	SSTBY	1.75				
FABS -	3.02	LBBI -	0.74	SBBCR	1.47	SSTEP	2.33				
FAD PI+E	5.45	LBBY	1.47	SBBP	0.87	SSTEP	17.47				
FCAM	4.52	LBR	0.74	SBBR	5.10	SSTX	1.75				
FCAZ	2.22	LBRM 1REG	2.73	SBBR	0.73	SSTY	1.75				
FCMZ	3.25	LBRM 2REGG	3.43	SBBS 1	4.15	SSTZ	1.75				
FDV PI/E	7.39	LBRM 4REG	4.87	SBBS 2	5.14	SSUPQ	4.94				
FIX 0	2.30	LBRP	1.03	SBBS 3	7.98	SSVC	3.15				
FIX PI	5.74	LBX	1.47	SBBS 4	9.12	SSWBR	0.74				
FLD	3.20	LBXI +	0.74	SBBT (A)	0.87	TBBT (A)	0.86				
FLT	4.07	LBXI -	0.74	SBBT (B)	0.87	TBBT (B)	0.86				
FMP PI#E	6.39	LBY	1.47	SBBT (AX)	0.86	TBBT (AX)	0.86				
FNEG	2.71	LBYI +	0.74	SBBT (BX)	0.86	TBBT (BX)	0.85				
FSB PI-E	5.45	LBYI -	0.74	SBBTM	4.58	TWAIT	5.42				
FST	4.96	LCVB 1	3.74	SBBTM (X)	4.35	TWAIT+ACT	80.76				
HALT 1	2.43	MCVB 2	5.03	SCLD 1	3.59	WCCDA 1	3.15				
HALT 2	4.69	MCVB 5	9.12	SCLD 3	4.94	WCCDA 2	11.13				
HBT (A)	0.86	MP	3.38	SCLD 9	7.77	WDBMR	2.43				
HBT (B)	0.86	MVTM 1	3.93	SCLD 16	4.06	XHM	3.27				
HBT (AX)	0.86	MVTM 2	5.17	SCLD 17	8.30	XM	1.75				
HBT (BX)	0.86	MVTM 5	9.16	SCLD 24	4.65	XR	1.75				
IC	2.33	MVTS 1	3.93	SCLD 25	7.48	SX	0.73				
INSC	5.03	MVTS 2	5.17	SCLD 31	4.06	SX	0.73				
IPI	6.44	MVTS 5	9.16	SCLS 1	6.34	SX FLOTAN	0.73				
JAE A=0	0.86	NGR	0.73	SCLS 8	3.69	SXFLOTAN	0.75				
JAE A#0	1.45	NOP	1.03	SCLS 9	6.06						
JAG A>0	0.86	NORM	4.19	SCLS 16	3.38						
JAG A<=0	1.45	OR	1.45	SCRD 1	3.38						

TEMPS DES INSTRUCTIONS ISP16

(en microsecondes)

Sur SOLAR 16-70, Horloge 140 ns, Mémoire 3 cycles

a - Instructions basées

	Adressage	
	Registre	Mémoire
BLA	4,20	5,46
BSTA	4,06	5,32
BXM	4,20	5,46
BLBY	4,48	5,74
BSTBY	4,62	5,88
BDLD	4,62	5,88
BDST	4,48	5,74
BMOVE 1 mot	4,34	5,60
mot sup.	1,96	1,96
BRBTM	5,32	6,58
BSBTM	5,18	6,44
BDBTM 1 mot	6,02	7,28
mot sup.	2,10	2,10
Indexation	0,14	0,14
Indirection 1° op.	X	0,42
" 2° op.	X	0,28
" les 2	X	0,70

b - Instructions sur CDA

CLA	5,18
CSTA	5,04
CXM	5,18
CLBY	5,60
CSTBY	5,74
CDLD	5.60
CDST	5,46
RCDA 1 mot	7,70
mot sup.	1,82
WCDA 1 mot	7,98
mot sup.	1,82
Indexation	0,14
Indirection	0,14

c - Instructions diverses

XENT 1° niveau	5,46
niveau sup.	1,12
XSOR	3,36
XCTX	5,88

TEMPS DES INSTRUCTIONS ISP16

(en microsecondes)

Sur SOLAR 16-35, Horloge 150 ns, Mémoire 6 cycles

a - Instructions basées

	Adressage Registre	Adressage Mémoire
BLA	28,65	36,45
BSTA	28,05	35,85
BXM	28,65	36,45
BLBY	31,5	39,3
BSTBY	33,75	41,55
BDLD	30,15	37,85
BDST	28,8	36,6
BMOVE 1 mot	29,7	37,5
mot sup.	30,3	38,1
BRBTM	43,2	51
BSBTM	43,35	51,15
BDBTM 1 mot	43,05	50,85
mot sup.	10,05	10,05
Indexation	1,35	1,35
Indirection 1° op.	X	0,9
" 2° op.	X	0,6
" les 2	x	1,5

b - Instructions sur CDA

CLA	30,9
CSTA	30,3
CXM	31,65
CLBY	34,05
CSTBY	36,15
CDLD	31,95
CDST	30,75
RCDA 1 mot	25,86
mot sup.	8,50
WCDA 1 mot	26,65
mot sup.	8,70
Indexation	1,2
Indirection	1,35

c - Instructions diverses

XENT 1° niveau	**23,10**
niveau sup.	**3,90**
XSOR	**18,45**
XCTX	**inexistant**

TABLES DE CONVERSIONS HEXADECIMAL <-> DECIMAL

Cette table permet des conversions directes des nombres hexadécimaux compris entre '000 et 'FFF, et des nombres décimaux compris entre 0 et 4095.

Pour les nombres qui sont hors des limites de la table, additionner les valeurs suivantes aux nombres de la table.

HEXADECIMAL	DECIMAL
4000	16384
5000	20484
6000	24576
7000	28672
8000	32768
9000	36864
A000	40960
B000	45056
C000	49152
D000	53248
E000	57344
F000	61440

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
010	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
020	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
030	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
040	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
050	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
060	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
070	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
080	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
090	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A0	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B0	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C0	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D0	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E0	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F0	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255
100	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
110	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
120	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
130	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
140	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
150	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
160	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
170	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
180	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
190	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A0	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B0	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C0	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D0	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E0	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F0	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
210	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
220	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
230	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
240	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
250	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
260	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
270	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
280	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
290	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A0	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B0	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C0	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D0	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E0	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F0	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
300	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
310	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
320	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
330	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
340	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
350	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
360	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
370	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
380	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
390	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A0	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B0	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C0	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D0	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E0	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F0	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
400	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
410	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
420	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
430	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
440	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
450	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
460	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
470	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
480	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
490	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A0	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B0	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C0	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D0	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E0	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F0	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
500	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
510	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
520	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
530	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
540	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
550	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
560	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
570	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
580	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
590	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A0	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B0	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C0	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D0	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E0	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F0	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
600	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
610	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
620	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
630	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
640	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
650	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
660	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
670	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
680	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
690	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A0	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B0	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C0	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D0	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E0	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F0	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791
700	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
710	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
720	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
730	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
740	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
750	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
760	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
770	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
780	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
790	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A0	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B0	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C0	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D0	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E0	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F0	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
800	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
810	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
820	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
830	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
840	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
860	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
870	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
880	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
890	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B0	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C0	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D0	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E0	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
900	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
910	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
920	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
930	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
950	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
960	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
970	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
980	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
990	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A0	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B0	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C0	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D0	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F0	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

Bull	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
[illegible]	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A20	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A40	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A50	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A60	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A70	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A90	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA0	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB0	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC0	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE0	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF0	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B00	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B10	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B30	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B40	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B50	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B60	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
[illegible]	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
[illegible]	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B90	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA0	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB0	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD0	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE0	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF0	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C10	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C20	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C30	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C40	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C60	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C70	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C80	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C90	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
[illegible]	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
[illegible]	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE0	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D00	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D10	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D20	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D30	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D50	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D60	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D70	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D80	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB0	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC0	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD0	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF0	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E00	3584	3585	3586	3587	3583	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E20	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E40	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E50	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E60	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E70	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E90	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA0	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB0	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC0	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE0	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF0	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F00	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F10	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F30	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F40	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F50	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F60	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F80	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F90	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA0	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB0	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD0	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE0	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF0	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

TABLE DES PUISSANCES DE DEUX

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25

CODAGE ASCII

hexa-décimal	code	code bande 8 7 6 5 4 3 2 1	touches du clavier TTY	remarques
00	NUL	0 0 0 0 0 . 0 0 0	CTRL SHIFT @	null
81	SØH	● 0 0 0 0 . 0 0 ●	CTRL A	début d'en-tête
82	STX	● 0 0 0 0 . 0 ● 0	CTRL B	début de texte
03	ETX	0 0 0 0 0 . 0 ● ●	CTRL C	fin de texte
84	EØT	● 0 0 0 0 . ● 0 0	CTRL D	fin de transmission
05	ENQ	0 0 0 0 0 . ● 0 ●	CTRL E	demande
06	ACK	0 0 0 0 0 . ● ● 0	CTRL F	accusé de réception
87	BEL	● 0 0 0 0 . ● ● ●	CTRL G	sonnerie
88	BS	● 0 0 0 ● . 0 0 0	CTRL H	espace arrière
09	HT	0 0 0 0 ● . 0 0 ●	CTRL I	tabulation horizontale
0A	LF	0 0 0 0 ● . 0 ● 0	line feed	interligne
8B	VT	● 0 0 0 ● . 0 ● ●	CTRL K	tabulation verticale
0C	FF	0 0 0 0 ● . ● 0 0	CTRL L	présentation de formule
8D	RC	● 0 0 0 ● . ● 0 ●	return	retour chariot
8E	SØ	● 0 0 0 ● . ● ● 0	CTRL N	hors code
0F	SI	0 0 0 0 ● . ● ● ●	CTRL Ø (lettre o)	en code
90	DLE	● 0 0 ● 0 . 0 0 0	CTRL P	échappement transmission
11	DC1	0 0 0 ● 0 . 0 0 ●	CTRL Q	marche lecteur
12	DC2	0 0 0 ● 0 . 0 ● 0	CTRL R	embrayage perfo.
93	DC3	● 0 0 ● 0 . 0 ● ●	CTRL S	arrêt lecteur
14	DC4	0 0 0 ● 0 . ● 0 0	CTRL T	débrayage perfo.
95	NAK	● 0 0 ● 0 . ● 0 ●	CTRL U	accusé de réception négatif
96	SYN	● 0 0 ● 0 . ● ● 0	CTRL V	synchronisation
17	ETB	0 0 0 ● 0 . ● ● ●	CTRL W	fin de bloc de transmission
18	CAN	0 0 0 ● ● . 0 0 0	CTRL X	annulation
99	EM	● 0 0 ● ● . 0 0 ●	CTRL Y	fin de support
9A	SUB	● 0 0 ● ● . 0 ● 0	CTRL Z	substitution
1B	ESC	0 0 0 ● ● . 0 ● ●	ESC	échappement
9C	FS	● 0 0 ● ● . ● 0 0	CTRL SHIFT L	séparateur de fichier
1D	GS	0 0 0 ● ● . ● 0 ●	CTRL SHIFT M	séparateur de groupe
1E	RS	0 0 0 ● ● . ● ● 0	CTRL SHIFT N	séparateur d'article
9F	US	● 0 0 ● ● . ● ● ●	CTRL SHIFT O	séparateur de sous-article

hexa-décimal	code	code bande 8 7 6 5 4 3 2 1	touches du clavier TTY	remarques
A0	SP	● 0 ● 0 0 . 0 0 0	barre d'espacement	espace
21	!	0 0 ● 0 0 . 0 0 ●	SHIFT (!)	
22	"	0 0 ● 0 0 . 0 ● 0	SHIFT (")	double apostrophe
A3	#	● 0 ● 0 0 . 0 ● ●	SHIFT (#)	
24	$	0 0 ● 0 0 . ● 0 0	SHIFT ($)	
A5	%	● 0 ● 0 0 : ● 0 ●	SHIFT (%)	
A6	&	● 0 ● 0 0 . ● ● 0	SHIFT (&)	
27	'	0 0 ● 0 0 . ● ● ●	SHIFT (')	apostrophe
28	(	0 0 ● 0 ● . 0 0 0	SHIFT (()	
A9	)	● 0 ● 0 ● . 0 0 ●	SHIFT ())	
AA	*	● 0 ● 0 ● . 0 ● 0	SHIFT (*)	
2B	+	0 0 ● 0 ● . 0 ● ●	SHIFT (+)	
AC	,	● 0 ● 0 ● . ● 0 0	(,)	
2D	−	0 0 ● 0 ● . ● 0 ●	(−)	signe moins
2E	.	0 0 ● 0 ● . ● ● 0	(.)	
AF	/	● 0 ● 0 ● . ● ● ●	(/)	signe "diviser"
30	0	0 0 ● ● 0 . 0 0 0	(0)	(chiffre 0)
B1	1	● 0 ● ● 0 . 0 0 ●	(1)	
B2	2	● 0 ● ● 0 . 0 ● 0	(2)	
33	3	0 0 ● ● 0 . 0 ● ●	(3)	
B4	4	● 0 ● ● 0 . ● 0 0	(4)	
35	5	0 0 ● ● 0 . ● 0 ●	(5)	
36	6	0 0 ● ● 0 . ● ● 0	(6)	
B7	7	● 0 ● ● 0 . ● ● ●	(7)	
B8	8	● 0 ● ● ● . 0 0 0	(8)	
39	9	0 0 ● ● ● . 0 0 ●	(9)	
3A	:	0 0 ● ● ● . 0 ● 0	(:)	
BB	;	● 0 ● ● ● . 0 ● ●	(;)	
3C	<	0 0 ● ● ● . ● 0 0	SHIFT (<)	
BD	=	● 0 ● ● ● . ● 0 ●	SHIFT (=)	
BE	>	● 0 ● ● ● . ● ● 0	SHIFT (>)	
3F	?	0 0 ● ● ● . ● ● ●	SHIFT (?)	

hexa-decimal	code	code bande 8 7 6 5 4 3 2 1	touches du clavier TTY	remarques
C0	@	● ● 0 0 0 . 0 0 0	SHIFT @	
41	A	0 ● 0 0 0 . 0 0 ●	A	
42	B	0 ● 0 0 0 . 0 ● 0	B	
C3	C	● ● 0 0 0 . 0 ● ●	C	
44	D	0 ● 0 0 0 . ● 0 0	D	
C5	E	● ● 0 0 0 . ● 0 ●	E	
C6	F	● ● 0 0 0 . ● ● 0	F	
47	G	0 ● 0 0 0 . ● ● ●	G	
48	H	0 ● 0 0 ● . 0 0 0	H	
C9	I	● ● 0 0 ● . 0 0 ●	I	
CA	J	● ● 0 0 ● . 0 ● 0	J	
4B	K	0 ● 0 0 ● . 0 ● ●	K	
CC	L	● ● 0 0 ● . ● 0 0	L	
4D	M	0 ● 0 0 ● . ● 0 ●	M	
4E	N	0 ● 0 0 ● . ● ● 0	N	
CF	Ø	● ● 0 0 ● . ● ● ●	O	(lettre o)
50	P	0 ● 0 ● 0 . 0 0 0	P	
D1	Q	● ● 0 ● 0 . 0 0 ●	Q	
D2	R	● ● 0 ● 0 . 0 ● 0	R	
53	S	0 ● 0 ● 0 . 0 ● ●	S	
D4	T	● ● 0 ● 0 . ● 0 0	T	
55	U	0 ● 0 ● 0 . ● 0 ●	U	
56	V	0 ● 0 ● 0 . ● ● 0	V	
D7	W	● ● 0 ● 0 . ● ● ●	W	
D8	X	● ● 0 ● ● . 0 0 0	X	
59	Y	0 ● 0 ● ● . 0 0 ●	Y	
5A	Z	0 ● 0 ● ● . 0 ● 0	Z	
DB	[	● ● 0 ● ● . 0 ● ●	SHIFT K	
5C	\	0 ● 0 ● ● . ● 0 0	SHIFT L	different du code /
DD	]	● ● 0 ● ● . ● 0 ●	SHIFT M	
DE	↑	● ● 0 ● ● . ● ● 0	SHIFT ↑	
5F	←	0 ● 0 ● ● . ● ● ●	SHIFT ←	

hexa-decimal	code	code bande 8 7 6 5 4 3 2 1	touches du clavier TTY	remarques	
60	\	0 ● ● 0 0 . 0 0 0		sur TTY s'imprime :	@
E1	a	● ● ● 0 0 . 0 0 ●		"	A
E2	b	● ● ● 0 0 . 0 ● 0		"	B
63	c	0 ● ● 0 0 . 0 ● ●		"	C
E4	d	● ● ● 0 0 . ● 0 0		"	D
65	e	0 ● ● 0 0 . ● 0 ●		"	E
66	f	0 ● ● 0 0 . ● ● 0		"	F
E7	g	● ● ● 0 0 . ● ● ●		"	G
E8	h	● ● ● 0 ● . 0 0 0		"	H
69	i	0 ● ● 0 ● . 0 0 ●		"	I
6A	j	0 ● ● 0 ● . 0 ● 0		"	J
EB	k	● ● ● 0 ● . 0 ● ●		"	K
6C	l	0 ● ● 0 ● . ● 0 0		"	L
ED	m	● ● ● 0 ● . ● 0 ●		"	M
EE	n	● ● ● 0 ● . ● ● 0		"	N
6F	o	0 ● ● 0 ● . ● ● ●		"	O
F0	p	● ● ● ● 0 . 0 0 0		"	P
71	q	0 ● ● ● 0 . 0 0 ●		"	Q
72	r	0 ● ● ● 0 . 0 ● 0		"	R
F3	s	● ● ● ● 0 . 0 ● ●		"	S
74	t	0 ● ● ● 0 . ● 0 0		"	T
F5	u	● ● ● ● 0 . ● 0 ●		"	U
F6	v	● ● ● ● 0 . ● ● 0		"	V
77	w	0 ● ● ● 0 . ● ● ●		"	W
78	x	0 ● ● ● ● . 0 0 0		"	X
F9	y	● ● ● ● ● . 0 0 ●		"	Y
FA	z	● ● ● ● ● . 0 ● 0		"	Z
7B	{	0 ● ● ● ● . 0 ● ●			
FC	|	● ● ● ● ● . ● 0 0			
7D	}	0 ● ● ● ● . ● 0 ●			
7E	~	0 ● ● ● ● . ● ● 0			
FF	DEL	● ● ● ● ● . ● ● ●	rub out	oblitération	

CODAGE ASCII PAR ORDRE NUMERIQUE

0		1		2		3	
00	NULL	11	DC1	21	!	30	0
03	ETX	12	DC2 (perfo on)	22	"	33	3
05	ENQ	14	DC4 (perfo off)	24	$	35	5
06	ACK	17	ETB	27	'	36	6
09	HT	18	CAN	28	(	39	9
0A	LF (line feed)	1B	ESC	2B	+	3A	:
0C	FF	1D	GS	2D	−	3C	<
0F	SI	1E	RS	2E	.	3F	?
4		**5**		**6**		**7**	
41	A	50	P	60	`	71	q
42	B	53	S	63	c	72	r
44	D	55	U	65	e	74	t
47	G	56	V	66	f	77	w
48	H	59	Y	69	i	78	x
4B	K	5A	Z	6A	j	7B	
4D	M	5C	\ (diff. de /)	6C	l	7D	
4E	N	5F	←	6F	o	7E	
8		**9**		**A**		**B**	
81	SOH	90	DLE	A0	SP (espace)	B1	1
82	STX	93	DC3	A3	#	B2	2
84	EOT	95	NAK	A5	%	B4	4
87	BEL (sonnerie)	96	SYN	A6	&	B7	7
88	BS	99	EM	A9	)	B8	8
8B	VT	9A	SUB	AA	*	BB	;
8D	RC (retour chariot)	9C	FS	AC	,	BD	=
8E	SO	9F	US	AF	/	BE	>
C		**D**		**E**		**F**	
C0	@	D1	Q	E1	a	F0	p
C3	C	D2	R	E2	b	F3	s
C5	E	D4	T	E4	d	F5	u
C6	F	D7	W	E7	g	F6	v
C9	I	D8	X	E8	h	F9	y
CA	J	DB	[	EB	k	FA	z
CC	L	DD	]	ED	m	FC	
CF	O	DE	↑	EE	n	FF	rub out