

CLEFS POUR APPLE II GS

Nicole Bréaud - Pouliquen



compete GSDemo.PAS

link qsdemo heap = qsdemo

CLEFS POUR APPLE II GS

Connaissez-vous la collection Apple II chez P.S.I. ?

* Disquette disponible par correspondance

Apple IIe et IIc

Basic

- 102 programmes pour Apple II* - Jacques Deconchat
- Super jeux Apple* - Jean-François Sehan
- Basic Plus, 80 routines sur Apple II* - Michel Martin

Applications

- Apple, modems et serveurs* - Alain Mariatte
- Apple, logique et systèmes experts* - René Descamps
- Création et animations graphiques sur Apple II* - Gilles Fouchard et Jean-Yves Corre
- Appletworks au travail* - Jean-Michel Jégo et Jean-Michel Gargadennec

Système

- Les ressources de l'Apple IIc - Nicole Bréaud-Pouliquen
 - Assembleur de l'Apple II - Nicole Bréaud-Pouliquen et Daniel-Jean David
 - Clefs pour Apple IIc - Nicole Bréaud-Pouliquen
 - Programmation système de l'Apple II - Marcel Cottini
 - ProDOS sur Apple - Marcel Cottini
-

Apple II GS

A paraître :

- Assembleur de l'Apple II GS - Jean-Pierre Lagrange
- La boîte à outils de l'Apple II GS - Jean-Pierre Curcio
- Programmation système de l'Apple II GS - Marcel Cottini
- Super jeux Apple II GS* - Jean-François Sehan

Pour tout problème rencontré dans les ouvrages P.S.I.
vous pouvez nous contacter au numéro ci-dessous :

Numéro Vert/Appel Gratuit en France

05 21 22 01

(Composer tous les chiffres, même en région parisienne)

La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective », et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1^{er} de l'article 40).

Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal.

MEMENTO

CLEFS POUR APPLE II GS

Nicole Bréaud - Pouliquen



**Éditions du P.S.I.
1986**

Présentation de l'auteur :

Nicole Bréaud-Pouliquen est ingénieur-conseil en informatique individuelle. Ingénieur de l'Ecole polytechnique féminine et Docteur-ingénieur en Automatique de l'Université de Grenoble, elle a enseigné quelques années à l'étranger (Brésil, Maroc), avant de se spécialiser dans les micro-ordinateurs Apple.

Auteur de nombreux ouvrages d'approfondissement de la programmation aux Editions du P.S.I., elle se consacre parallèlement à la recherche de modes d'utilisation conviviaux des ordinateurs individuels, plus particulièrement dans les domaines artistiques.

Ouvrages conseillés par l'auteur :

- **Programmer votre Macintosh** - A. Andrieux et C. Droulers (*McGraw Hill*).
- **Mastering the Macintosh Toolbox** - David B. Peatroy et Datatech Publications (*Osborne et McGraw Hill*).
- **Inside Macintosh** - Apple USA et Addison Wesley.
- **La documentation destinée aux développeurs**, fournie par Apple Seedrin.

Nous remercions Apple Computer France pour son aide matérielle et technique.

Apple IIgs est une marque déposée de Apple Computer Inc.

SOMMAIRE

PRESENTATION	11
CHAPITRE I : MATERIEL APPLE IIgs	13
Architecture de la carte-mère	13
Circuits intégrés de la carte-mère	14
Brochage des connecteurs	15
Connecteurs d'entrée/sortie	15
Connecteur pour carte d'extension de mémoire	15
Connecteur de manettes de jeux	15
Port série	16
Manettes de jeux	16
Lecteur de disquette	16
Vidéo R V B	16
Apple Desk Bus	16
Signaux des connecteurs	17
Broches des connecteurs pour cartes d'interface	18
Microprocesseur 65C816	21
255 codes d'instructions	21
Registres du 65C816	50
Mémoires de l'Apple IIgs	53
Utilisation de l'espace mémoire : vive, morte, extension	53
Occupation de la mémoire : système, utilisateur, graphique	53
Shadowing-ombre portée	54

Ressources graphiques	56
Deux prises vidéo	56
Modes vidéo	56
Couleurs des palettes	57
SCB ou Scan line Control Byte	57
Pixels	58
Registre \$C029	58
Entrées/sorties	60
Deux ports série	60
Interface AppleTalk	60
Port disque	60
Port manette de jeux	61
Bus Apple Desktop	61
Sorties vidéo	62
Sorties haut-parleur	62
Connecteurs d'E/S	62
Horloge	62
Interruptions	63
Liste par ordre de priorité	63
Vecteurs d'interruptions	63
Indicateurs des sources d'IRQ	65
Etats des registres après un BRK	66
Autorisation ou inhibition des interruptions	66
File d'attente du Hearbeat	67
Registres d'état	68
\$C029 : Vidéo Select Register	68
\$C02B : Langage Select Register	68
\$C02D : Slot ROM Register	69
\$C036 : Configure your Apple Register - sélecteur de configuration	69
\$C068 : registre d'état des commutateurs logiciels	70
Tableau de bord	71

CHAPITRE II : LOGICIELS DE DEVELOPPEMENT 73

Moniteur	73
Commandes à valider par return	73
<i>Examiner les registres</i>	73
<i>Examiner la mémoire</i>	73
<i>Modifier les registres</i>	74
<i>Modifier la mémoire</i>	74
<i>Lister un programme</i>	75
<i>Exécuter un programme</i>	75
<i>Rechercher une chaîne de caractères</i>	75
<i>Afficher en inverse</i>	76
<i>Revenir à l'écran-texte</i>	76
<i>Regler le jour et l'heure</i>	76

<i>Convertir et calculer</i>	76
<i>Rediriger les entrées/sorties</i>	76
<i>Sauter à un programme par une seule commande</i>	76
<i>Sortir du moniteur</i>	76
<i>Appeler une fonction d'un outil</i>	77
<i>Appel du mini-assembleur</i>	77
Listings	
<i>Pratique du moniteur</i>	77
<i>Analyse d'un outil</i>	78
CPW	
Editeur CPW	86
Modes	86
Taquets de tabulation	86
Retour à la ligne	88
Frappe au km	88
Recherche et remplacement	88
Fin d'édition	89
Fichier des défauts	89
Création de commandes personnalisées	89
Effacer complètement le texte	90
Commandes et utilisation du CPW	90
Liste par ordre alphabétique	90
Paramètres optionnels	90
Caractères JOKER	94
Redirection de entrées/sorties	94
Procédure d'assemblage en vue d'une exécution sous Pro Dos/16	95
Codes des fichiers	95
Nouvelles commandes	97
Listings	97
Catalogue principal de la disquette CPW	97
Catalogue de la disquette CPW en cours	97
Macro assembleur ORCA/M 4.0 inclus dans CPW	98
Choix du jeu d'instructions	101
Choix de la longueur des registres	101
Choix dans la présentation du code généré	101
Connaître ou non le temps d'exécution	102
Fixer ou non l'adresse d'implantation du code	102
Délimiter et nommer des segments	102
Données	102
Explotation de fichiers	103
Listing : exercice de présentation	104
Macros sous CPW	105
Utilisation des macros	108
Création de macros	108
Exemple d'une macro	110
Listing : catalogue des macro-système	111
Langage LINKED et segmentation	112
Commandes	113
	114

Outils	
Ensemble (TOOL SET) en mémoire morte	116
Structure des ensembles d'outils	116
Liste des fonctions outil par outil	116
<i>Tool Locator</i>	117
<i>Memory Manager</i>	117
<i>Miscellaneous Tool</i>	118
<i>Quick Draw II</i>	118
<i>Desk Manager</i>	119
<i>Event Manager</i>	124
<i>Scheduler</i>	124
<i>Sound</i>	125
<i>ADB</i>	125
<i>Sane</i>	125
<i>Integer Maths</i>	125
<i>Text Tools</i>	126
<i>Window Manager</i>	127
<i>Menu Manager</i>	128
<i>Control Manager</i>	129
<i>System Loader</i>	130
<i>Line Edit</i>	131
<i>Dialog Manager</i>	132
<i>Scrap Manager</i>	132
Liste alphabétique des fonctions	133
Memory Manager	134
Quick Draw	154
Event Manager	156
Utilisation du menu-Manager	159
Window Manager	162
Desk Manager	167
	173

CHAPITRE III - EXEMPLE 175

Montre extra-plate de votre bureau 175

CONSEILS DE LECTURE 187

PRESENTATION

C'est une "formule 1" que vous allez avoir entre les mains, et la meilleure école de pilotage pour l'aborder est celle du Macintosh.

Tableau et instruments de bord sont en effet les mêmes. Mais, en plus, avec le GS, le paysage est en couleur ; et en prêtant l'oreille, on peut entendre les cloches sonner, l'orage tonner et un concert de rock débiter à 2 heures du matin !

Les logiciels de cette nouvelle machine sont superbes. GS/Paint, fidèle reproduction de MacPaint, mais en couleur, fait en plus de l'animation graphique. Le traitement de texte GS/Write en couleur lui aussi, met les mots en évidence comme avec un feutre fluo, et raye sans l'effacer la ligne à supprimer. Superbes, on vous dit !

Mais telles ou tels qu'on vous connaît, vous avez déjà le pied au plancher et vous voilà partis dans la programmation d'un logiciel personnel, hors des sentiers battus.

Ce n'est pas un plan galère, c'est presque un plan d'enfer. C'est pourquoi, il m'a paru utile de baliser la piste avant vous.

A travers ce livre, qui résume l'ensemble infiniment riche des fonctions du système Apple IIGS, je vous propose des stands de ravitaillement où vous trouverez une description détaillée du microprocesseur 65816, des mémoires, du système de développement CPW... ainsi que l'indispensable boîte à outils (software tools).

En un mot, tout ce qui est nécessaire pour réaliser des logiciels accessibles et compréhensibles au commun des mortels, vos futurs clients !

Et maintenant, bonne route !

L'autrice.

MATERIEL APPLE II GS

ARCHITECTURE DE LA CARTE-MERE

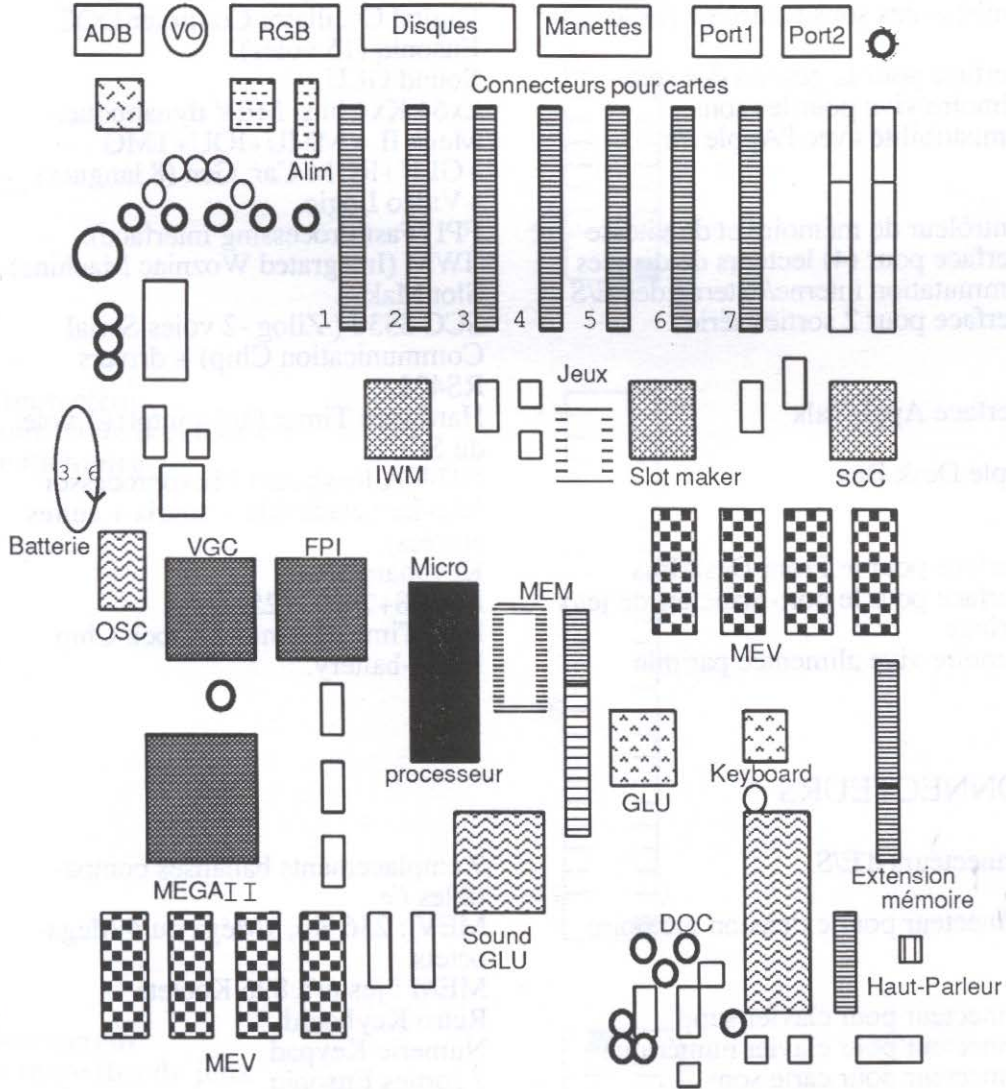


Schéma de la carte-mère

CIRCUITS INTEGRES DE LA CARTE-MERE

Fonctions

Microprocesseur

Mémoire vive ou MEV
Mémoires mortes ou MEM

Affichage vidéo et interruptions

Générateur de signaux vidéo RGB
Synthèse des sons contrôlée par le

Interface pour la gestion des sons
Mémoire vive pour les sons
Compatibilité avec l'Apple //e

Contrôleur de mémoire et de vitesse
Interface pour (4) lecteurs de disques
Commutation interne/externe des E/S
Interface pour 2 sorties série

Interface AppleTalk

Apple Desk Bus

Interface pour le Front Desk Bus
Interface pour le port-manettes de jeux
Horloge
Mémoire vive alimentée par pile

CONNECTEURS

Connecteurs d'E/S

Connecteur pour extension mémoire

Connecteur pour clavier supp
Connecteur pour clavier numérique
Connecteur pour carte son

Circuits

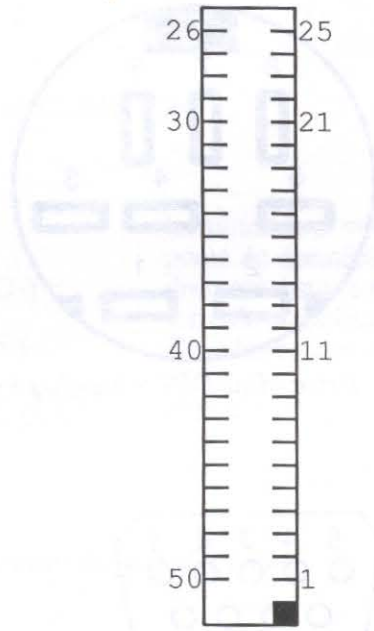
65SC816-16 bits - 8 registres-mode
émulation 65C02 - 2 vitesses : 1Mhz
ou 2.5 Mhz. **32 bits d'adressage.**
2x4 C.I.x64 Kx4bits=256 Koctets.
4 C.I x 32 Koctets=128 Koctets.

contrôlés par le Video Graphics
Controller VGC et le Mega II.
Video Hybrid + MC1377.
Digital Oscillator Controller DOC
Ensoniq (15 voies).
Sound GLU.
2x64 Kx4 bits MEV dynamique.
Mega II = MMU+IOU+TMG
+GLU+ROM Car. Gen (8 langues)
+Vidéo Logic.
FPI (Fast Processing Interface).
IWM (Integrated Wozniac Machine)
Slot Maker.
SCC 8530 (Zilog -2 voies Serial
Communication Chip) + drivers
RS422.
Hardware Timer (1/4 s intpt)+1 voie
du SCC.
50740A Keyboard Microprocessor
(clavier détachable + souris + autres
entrées).
Keyboard GLU.
NE558+74HCT251.
Real Time Macintosh Clock Chip.
RAM-battery.

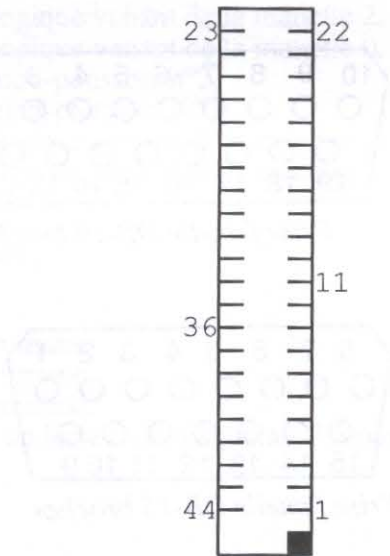
7 emplacements banalisés compa-
tibles //e.
MEV : 256 Ko, 1Méga ou 4 Mega-
octets.
MEM : jusqu'à 896 Koctets.
Retro Keyboard.
Numeric Keypad.
7 sorties Ensoniq.

BROCHAGE DES CONNECTEURS

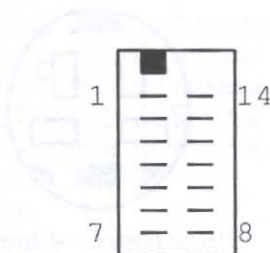
Connecteurs
d'entrée/sortie 1 à 7



Connecteur
pour carte d'extension
de mémoire

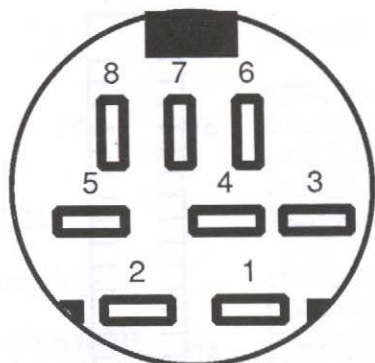


Connecteur
de manettes de jeux



BROCHAGE DES CONNECTEURS

Port série 1 ou 2



Prise Mini-DIN 8 broches



1 = 20

3 =

3 =

4 = 7

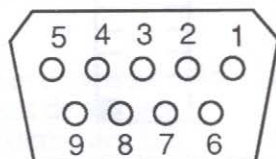
5 =

6 =

7 =

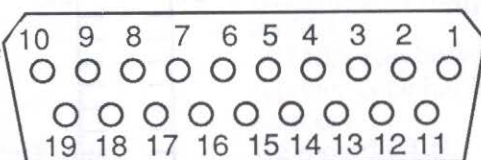
8 =

Manettes de jeu



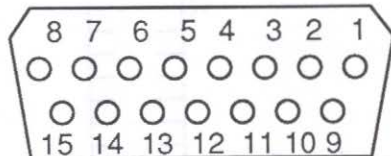
Prise femelle DB-9 broches

Lecteur de disquette



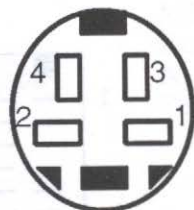
Prise femelle DB-19 broches

Vidéo RVB



Prise femelle DB-15 broches

Apple Desk Bus



Prise femelle 4 broches

BROCHAGE DES CONNECTEURS

Signaux des connecteurs

Ports Série: Norme RS-232C ou CCITT V.24

N°			CCIT V.24
1	Handshake Out ou DTR	terminal de données prêt	108.2
2	Handshake In ou DSR	poste de données prêt	107
3	Transmit Data Minus (TxD-)	données émises en sortie	103
4	Signal Ground (SG)	terre de signalisation	102
5	Receive Data Minus (RxD-)	données reçues en entrée	104
6	Transmit Data Plus (TxD+)	cf 3.	-
7	Vers l'entrée DCD du SCC (GPi)	détection de porteuse	109
8	Receive Data Plus (RxD+)	cf 5.	-
	Entourage	terre physique de protection	101

Connecteur externe pour manettes de jeux d'arcade

N°		
1	PB1	état du bouton-poussoir n° 1.
2	+5 Volts.	
3	GND	terre.
4	PDL2	entrée analogique venant de la manette 2.
5	PDL0	entrée analogique venant de la manette 0.
6	PB2	état du bouton-poussoir n°2.
7	PB0	état du bouton-poussoir N°0.
8	PDL1	entrée analogique venant de la manette 1.
9	PDL3	entrée analogique venant de la manette 3.

Port pour lecteur de disquette

N°		
1	GND	terre commune.
2	GND	terre commune.
3	GND	terre commune.
4	EN3.5	autorise un lecteur de disquettes 3.5 pouces.
5	-12 volts.	
6	+5 volts.	
7	+12 volts.	
8	+12 volts.	
9	EXTINT	interruption externe.
10	WRPROT	entrée indiquant que la disquette est protégée en écriture.
11	Phi 0	signal phase 0 de contrôle du moteur.
12	Phi 1	signal phase 1 de contrôle du moteur.
13	Phi 2	signal phase 2 de contrôle du moteur.
14	Phi 3	signal phase 3 de contrôle du moteur.
15	WRREQ'	demande d'écriture.
16	HDSEL	sélection de la tête.
17	DR1'	sélection du lecteur n° 1.

BROCHAGE DES CONNECTEURS

- 18 RDDATA entrée des données lues sur disquette.
19 WRDATA sortie des données à écrire.

Connecteur vidéo R V B

N°		
1	GND	terre.
2	Red Video and Sync	signal vidéo rouge incluant le signal de Sync Composite.
3	Composite Sync	
4	N.C.	non connecté.
5	Green Video and Sync	signal vidéo vert incluant le signal de Sync Composite.
6	GND	terre.
7	-5 volts	
8	+12 volts	
9	Blue Video and Sync	signal vidéo bleu incluant le signal de Sync Composite.
10	N.C.	non connecté.
11	Sound	signal audio.
12	Composite Video	signal composite (identique à celui sortant de la prise VC).
13	GND	terre.
14	N.C.	non connecté.
15	N.C.	non connecté.

Front Desk Bus

N°	
1	Données.
2	Réservée.
3	Alimentation (+5v , 500mA).
4	Retour.

Broches des connecteurs pour cartes d'interface

N°	(l'apostrophe après le nom indique que le signal est actif à son niveau bas).
1	IOSEL' informe la carte que les adresses CnXX sont sélectionnées.
2	A0
3	A1
4	A2
5	A3
6	A4

BROCHAGE DES CONNECTEURS

7	A5	
8	A6	
9	A7	
10	A8	
11	A9	
12	A10	
13	A11	
14	A12	
15	A13	
16	A14	
17	A15	
18	R / W' Read/Write	signal d'écriture/lecture venant du micro- processeur.
19	N.C.	non connecté.
20	IOSTROBE	informe la carte quand les adresses C800 à CFFF sont sélectionnées.
21	RDY	signal d'entrée dans le microprocesseur pour le stopper.
22	DMA (Direct Memory Access)	bloque l'accès au bus d'adresses par le microprocesseur pour l'autoriser à la carte d'interface.
23	INT OUT	(non connecté sur le connecteur d'E/S n°7) sortie interruption.
24	DMA OUT	(non connecté sur le connecteur d'E/S n°7) sortie DMA.
25	+ 5 volts	
26	GND	terre commune du système.
27	DMA IN	(non connecté sur le connecteur d'E/S n°1) entrée DMA.
28	INT IN	(non connecté sur le connecteur d'E/S n°1) entrée interruption.
29	NMI'	signal d'interruption non masquable envoyé au microprocesseur.
30	IRQ'	signal d'interruption masquable avec l'indicateur I du registre P.
31	RESET'	signal donnant lieu à une procédure de redémarrage.
32	INH'	inhibition de la mémoire principale au profit de celles transmises par l'interface.
33	-12 volts	
34	-5 volts	
35	N.C.	non connecté (signal CREF : 3.58 Mhz vidéo sur le connecteur d'E/S n°7).
36	7M	signal d'horloge du système.
37	Q3	signal asymétrique d'horloge à 2 Mhz
38	PH1	phase 1 du timing du microprocesseur.
39	M2SEL	informe qu'une adresse valide des banques E0 et E1 est présente.
40	PH0	phase 0 du timing du microprocesseur.

BROCHAGE DES CONNECTEURS

41 DEVSEL

informe que les adresses C0nX sont présentes sur le bus d'adresses (n est le numéro du connecteur+8).

42 D7

43 D6

44 D5

45 D4

46 D3

47 D2

48 D1

49 D0

50 +12 volts

bus de données sur 8 bits D7 à D0.

255 codes d'instructions (ordre alphabétique)

ADC

Addition avec retenue (ADd with Carry)

$$A \leftarrow A + M + c$$

$$\text{nv} \text{m} \text{x} \text{d} \text{i} \text{z} \text{c}$$

$$\sim \sim \dots \sim \sim$$

On ajoute à l'accumulateur le mot spécifié plus le bit de retenue. On opère en mode binaire ou décimal. (En cas de résultat nul en décimal, l'indicateur z n'est pas positionné).

Modes d'adressage	Assembleur	Code	Oct..	Cycles min +
Immédiat émulé (m=1)	ADC £donnée	69 donnée	2	2
Immédiat natif (m=0)	ADC £donnée	69 donL donH	3	3 (1)
Absolu	ADC adr ou adr	6D adL adH	3	4 (1)
Absolu long	ADC adr ou >adr	6F adL adH Badr	4	5 (1)
Absolu indexé par X	ADC adr,X	7D adL adH	3	4 (1,3)
Absolu indexé par Y	ADC adr,Y	79 adL adH	3	4 (1,3)
Absolu long indexé	ADC adr,X ou >adr,X	7F adL adH Badr	4	5 (1)
Direct page zéro	ADC adr ou <adr	65 adr	2	3 (1,2)
Direct indirect	ADC (adr)	72 adr	2	5 (1,2)
Direct indirect indexé	ADC (adr),Y	71 adr	2	5 (1,2,3)
Direct indirect long	ADC [adr]	67 adr	2	6 (1,2)
Direct indirect long indexé	ADC [adr],Y	77 adr	2	6 (1,2)
Direct indexé indirect	ADC (adr,X)	61 adr	2	6 (1,2)
Direct indexé par X	ADC adr,X	75 adr	2	4 (1,2)
Relatif à la pile	ADC adr,S	63 adr	2	4 (1)
Relatif à la pile indirect indexé	ADC(adr,S),Y	73 adr	2	7 (1)

AND

ET logique (AND)

$$A \leftarrow A \& M$$

$$\text{nv} \text{m} \text{x} \text{d} \text{i} \text{z} \text{c}$$

$$\sim \dots \sim$$

On effectue le ET logique bit à bit entre l'accumulateur et la mémoire conformément à la table de vérité :

0&0=0
0&1=0
1&0=0
1&1=1

Modes d'adressage	Code	Oct	Cycles+
Immédiat émulé (m=1)	AND £donnée	29 donnée	2 2
Immédiat natif (m=0)	AND £donnée	29 donL donH	3 3

MICROPROCESSEUR 65C816

Absolu	AND adr ou adr	2D	adL adH	3	4	(1)
Absolu long	AND adr ou >adr	2F	adL adH Badr	4	5	(1)
Absolu indexé par X	AND adr,X	3D	adL adH	3	4	(1,3)
Absolu indexé par Y	AND adr,Y	39	adL adH	3	4	(1,3)
Absolu long indexé	AND adr,X ou >adr,X	3F	adL adH Badr	4	5	(1)
Direct page zéro	AND adr ou <adr	25	adr	2	3	(1,2)
Direct indirect	AND (adr)	32	adr	2	5	(1,2)
Direct indirect indexé	AND (adr),Y	31	adr	2	5	(1,2,3)
Direct indirect long	AND [adr]	27	adr	2	6	(1,2)
Direct indirect long indexé	AND [adr],Y	27	adr	2	6	(1,2)
Direct indexé indirect	AND (adr,X)	21	adr	2	6	(1,2)
Direct indexé par x	AND adr,X	35	adr	2	4	(1,2)
Relatif à la pile	AND adr,S	23	adr	2	4	(1)
Relatif à la pile indirect indexé	AND(adr,S),Y	33	adr	2	7	(1)

ASL

Décalage à gauche (Arithmetic Shift Left)

Natif (m=0)	c<-b ₁₅ ; b ₁₅ <-b ₁₄ ;...; b ₁ <-b ₀ ; b ₀ <-0	nvmxdizc
Emulation (m=1)	c<-b ₈ ; b ₈ <-b ₇ ;...; b ₁ <-b ₀ ; b ₀ <-0	~.....~

On décale à gauche (d'un bit) l'accumulateur ou une mémoire. Un zéro entre à droite, tandis que le bit sortant à gauche tombe dans la retenue.

Modes d'adressage		Code	Oct.	Cycles +
Accumulateur	ASL A	0A	1	2
Absolu	ASL adr ou adr	0E adL adH	3	6 (5)
Absolu indexé par X	ASL adr,X	1E adL adH	2	7 (5)
Direct	ASL adr ou <adr	06 adr	2	5 (2,5)
Direct indexé par X	ASL adr,X	16 adr	2	6 (2,5)

Exemple

Le décalage d'un bit vers la gauche correspond à une multiplication par 2.

BCC

Branchement si pas de retenue (Branch on Carry Clear)

	nvmxdizc
si c=0 alors PC<-PC+dep

Mode d'adressage		Code	Oct.	Cycles +
Relatif	BCC adr	90 dep	2	2 (7,8)

BCS

Branchement si retenue (Branch on Carry Set)

si c=1 alors PC<-PC+dep

nvmxdizc
.....

Mode d'adressage

Relatif

BCS adr

Code

B0 dep

Oct. Cycles +

2 2 (7,8)

BEQ

Branchement si zéro (Branch on Equal)

si z=0 alors PC<-PC+dep

nvmxdizc
.....

Mode d'adressage

Relatif

BEQ adr

Code

F0 dep

Oct. Cycles +

2 2 (7,8)

Exemple

Après comparaison de l'accumulateur et d'un opérande, le bit z est à 1 s'ils sont égaux et à 0 s'ils sont différents ; d'où l'usage de BEQ après CMP si on attend l'égalité pour sauter à une action déterminée.

BIT

Test de bits (BIT Test)

$z < \overline{\sum A_i \& m_i}$; $n < M_7$; $v < M_6$

n v mxdizc
M7 M6~.

On effectue le ET virtuel (c'est-à-dire que le résultat n'est pas remis dans A qui reste inchangé) entre l'accumulateur et la mémoire spécifiée ; l'indicateur z est positionné en conséquence. En outre, les bits 6 et 7 ou les bits 14 et 15 de la mémoire sont copiés respectivement dans v et n avant l'exécution de BIT. En mode d'adressage immédiat les indicateurs n et v ne sont pas affectés.

Modes d'adressage

Immédiat émulé (m=1) BIT £donnée
Immédiat natif (m=0) BIT £donnée
Absolu BIT adr
Absolu indexé par X BIT adr,X
Direct BIT adr ou <adr
Direct indexé par X BIT adr,X

Code

89 donnée
89 donL donH
2C adL adH
3C adL adH
24 adr
34 adr

Oct. Cycles +

2 2
3 3
3 4 (1)
3 4 (1,3)
2 3 (1,2)
2 4 (1,2)

BMI

Branchement si négatif (Branch on Minus)

si n=1 alors PC<-PC+dep

nvmxdizc
.....

Si le dernier résultat est négatif, on saute à l'adresse indiquée ; sinon, on continue en séquence.

MICROPROCESSEUR 65C816

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles +</i>
Relatif	BMI adr	30 dep	2 2 (7,8)

BNE

Branchement si non égal (Branch if Not Equal)

si z=0 alors PC<-PC+dep
nvmxdizc
.....

Si le dernier résultat est différent de zéro ou si la dernière comparaison n'a pas donné l'égalité, on saute à l'adresse indiquée ; sinon, on continue en séquence.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles +</i>
Relatif	BNE adr	D0 dep	2 2 (7,8)

BPL

Branchement si positif ou nul (Branch if Plus)

si n=0 alors PC<-PC+dep
nvmxdizc
.....

Si le dernier résultat est ≥ 0 , on saute à l'adresse indiquée ; sinon, on continue en séquence.

Puisque le bit n est à 1 pour les nombres négatifs représentés par leur complément à 2, on continue en séquence si le dernier résultat est un nombre négatif.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles +</i>
Relatif	BPL adr	10 dep	2 2 (7,8)

BRA

Branchement inconditionnel (BRanch Always)

PC<- PC+dep
nvmxdizc
.....

On saute à l'adresse indiquée sans condition.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Relatif	BRA adr	80 dep	2 2

Exemple

Cette instruction est identique à JMP mais n'utilise que 2 octets au lieu de 3. L'opérande étant un déplacement, cette instruction est relogeable alors que JMP ne l'est pas, mais l'adresse de destination ne peut se trouver à plus de 128 octets avant ou après le premier octet suivant l'instruction.

Boucle jsr ceci
 jsr cela
 bra Boucle

BRK

Interruption logicielle (BReaK)

Empile PC(+2) et P; nv.bdizc
...1....
 Mode Emulation(e=1) PCL <- (\$00FFFE); PCH <- (\$00FFFF)
 Mode Natif (e=0) PCL <- (\$00FFE7); PCH <- (\$00FFE6)

Force une interruption en mettant le bit b à 1 (mode émulation e=1) ce qui simule une interruption ; le PC + 2 est empilé ainsi que le registre d'état P, on saute indirectement à l'adresse contenue dans le vecteur d'interruption.

Le bit i d'inhibition d'interruption n'a pas d'effet sur BRK.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Inhérent	BRK donnée	00 donnée	2 7 (9)

BRL

Branchement inconditionnel long (BRanch Long)

PC <- PC + ldep nvmxdizc
.....

On saute à l'adresse indiquée sans condition.

ldep est un déplacement relatif sur 16bits.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Relatif	BRL adr	82 depL depH	3 4

Exemple

BRL ne permet un branchement inconditionnel qu'à l'intérieur du banc de programme courant (modulo \$10000), car le registre de banc de programme PB ne reçoit pas la retenue de propagation du PC quand le déplacement est ajouté au PC.

BVC

Branchement si pas de débordement (Branch on oVerflow Clear)

si v=0 alors PC <- PC + dep nvmxdizc
.....

Si le bit de débordement du registre P est à 0, on saute à l'adresse indiquée ; sinon on continue en séquence.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles +</i>
Relatif	BVC adr	50 dep	2 2 (7,8)

MICROPROCESSEUR 65C816

Direct indirect long	CMP [adr]	C7 adr	2	6 (1,2)
Direct indirect long indexé	CMP [adr], Y	D7 adr	2	6 (1,2)
Direct indexé indirect	CMP (adr, X)	C1 adr	2	6 (1,2)
Direct indexé par X	CMP adr, X	D5 adr	2	4 (1,2)
Relatif à la pile	CMP adr, S	C3 adr	2	4 (1)
Relatif à la pile indirect indexé	CMP(adr, S), Y	D3 adr	2	7 (1)

COP

Autorise un Co-processeur

(CO-Processor enable) Empile PC(+2) et P;

Mode Emulation (e=1) PCL<-(\$FFF4) ; PCH<-(\$FFF5); PB<-00

Mode Natif (e=0) PCL<-(\$FFE4) ; PCH<-(\$FFE5); PB<-00

Il s'agit d'une interruption logicielle du même type que BRK, mais avec un vecteur différent de traitement de l'interruption. COP permet à un processeur de nombres flottants ou un processeur graphique d'exécuter une de ses fonctions.

Mode d'adressage	Code	Oct. Cycles
Inhérent	COP donnée	02 donnée 2 7 (9)

La donnée n'est pas un opérande mais elle permet à l'instruction d'occuper 2 octets pour que le retour d'interruption se fasse correctement.

CPX

Comparaison avec X (ComPare X register)

X - M => n, z, c	X > M	~	z c
	X = M	0	11
	X < M	~	00

On effectue la soustraction virtuelle (c'est-à-dire que le résultat n'est pas remis dans X, qui reste inchangé) Registre X - Mémoire et on positionne les indicateurs n, z et c ; z est mis à 1 s'il y a égalité ; c est mis à 1 si $X \geq M$ (les nombres sont considérés comme sans signe). Notez que c'est c qui est le plus déterminant. Pour prévoir l'état de n, faire X+complément de M ; n sera correct s'il n'y a pas de débordement ; v est inchangé.

La caractéristique la plus importante de l'instruction est que X reste inchangé, d'où la possibilité de comparaison en cascade.

Modes d'adressage	Code	Oct. Cycles
Immédiat émulé (x=1)	CPX £donnée	E0 don 2 2
Immédiat natif (x=0)	CPX £donnée	E0 donL donH 3 3
Absolu	CPX adr	EC adL adH 3 4 (10)
Direct	CPX adr	E4 adr 2 3 (2,10)

CPY

Comparaison avec Y (ComPare Y register)

Y - M =>n,z,c	Y>M	nvmxdi	zc
	Y=M	~.....	01
	Y<M	0.....	11
		~.....	00

On effectue la soustraction virtuelle(c'est-à-dire que le résultat n'est pas remis dans Y, qui reste inchangé) RegistreY - Mémoire et on positionne les indicateurs n, z et c ; z est mis à 1 s'il y a égalité ; c est mis à 1 si $Y \geq M$ (les nombres sont considérés comme sans signe). Notez que c'est c qui est le plus déterminant. Pour prévoir l'état de n, faire Y+complément de M ; n sera correct s'il n'y a pas de débordement ; v est inchangé.

La caractéristique la plus importante de l'instruction est que Y reste inchangé, d'où la possibilité de comparaison en cascade.

<i>Modes d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Immédiat emul (x=1)	CPY £donnée	C0 donnée	2 2
Immédiat natif (x=0)	CPY £donnée	C0 donL donH	3 3
Absolu	CPY adr	CC adL adH	3 4 (10)
Direct	CPY adr	C4 adr	2 3 (2,10)

DEC

Décrémentation mémoire (DECrement memory)

M<-M - 1	nvmxdizc
	~.....~.

On diminue de 1 le contenu de la mémoire indiquée.

<i>Modes d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Accumulateur	DEC A	3A	1 2
Absolu	DEC adr	CE adL adH	3 6 (5)
Absolu indexé par X	DEC adr,X	DE adL adH	3 7 (5)
Direct	DEC adr ou <adr	C6 adr	2 5 (2,5)
Direct indexé par X	DEC adr,X	D6 adr	2 6 (2,5)

DEX

Décrémentation de X (DECrement X register)

X<-X - 1	nvmxdizc
	~.....~.

On diminue de 1 le contenu du registre d'index X.

<i>Mode d'adressage</i>		<i>Code</i>	<i>Oct. Cycles</i>
Inhérent	DEX	CA	1 2

MICROPROCESSEUR 65C816

DEY

Décrémentation de Y (DEcrement Y register)

$$Y < Y - 1$$

On diminue de 1 le contenu du registre d'index Y.

nvmxdizc

~.....~.

Mode d'adressage

Inhérent

DEY

Code

88

Oct. Cycles

1 2

EOR

OU Exclusif (Exclusive OR)

$$A < A \text{ XOR } M$$

nvmxdizc

~.....~.

On effectue le OU exclusif bit à bit entre l'accumulateur et la mémoire conformément à la table de vérité suivante :

$$0 \text{ XOR } 0 = 0$$

$$0 \text{ XOR } 1 = 1$$

$$1 \text{ XOR } 0 = 1$$

$$1 \text{ XOR } 1 = 0$$

Mode d'adressage

Immédiat émulé(m=1)

EOR fdonnée

Code

49 donnée

Oct. Cycles

2 2

Immédiat natif(m=0)

EOR fdonnée

49 donL donH

3 3

Absolu

EOR adr ou |adr

4D adL adH

3 4 (1)

Absolu long

EOR adr ou >adr

4F adL adH Badr

4 5 (1)

Absolu indexé par X

EOR adr,X

5D adL adH

3 4 (1,3)

Absolu indexé par Y

EOR adr,Y

59 adL adH

3 4 (1,3)

Absolu long indexé

EOR adr,X

5F adL adH Badr

4 5 (1)

Direct page zéro

EOR adr ou <adr

45 adr

2 3 (1,2)

Direct indirect

EOR (adr)

52 adr

2 5 (1,2)

Direct indirect indexé

EOR (adr),Y

51 adr

2 5 (1,2,3)

Direct indirect long

EOR [adr]

47 adr

2 6 (1,2)

Direct indirect long

EOR [adr],Y

57 adr

2 6 (1,2)

Direct indexé indirect

EOR (adr,X)

41 adr

2 6 (1,2)

Direct indexé par X

EOR adr,X

55 adr

2 4 (1,2)

Relatif à la pile

EOR adr,S

43 adr

2 4 (1)

Relatif à la pile indirect

EOR (adr,S),Y

53 adr

2 7 (1)

indexé

INC

Incrémentation (INcrement)

$$M < M + 1$$

On augmente de 1 le contenu de la mémoire indiquée.

nvmxdizc

~.....~.

MICROPROCESSEUR 65C816

Modes d'adressage		Code	Oct.	Cycles
Accumulateur	INC A	1A	1	2
Absolu	INC adr	EE adL adH	3	6 (5)
Absolu indexé par X	INC adr,X	FE adL adH	3	7 (5)
Direct page zéro	INC adr	E6 adr	2	5 (2,5)
Direct indexé par X	INC adr,X	F6 adr	2	6 (2,5)

INX

Incrémentation de X (INcrement X register)

X <- X + 1		nvmxdizc	~.....~.	
On augmente de 1 le contenu du registre d'index X.				
Mode d'adressage		Code	Oct.	Cycles
Inhérent	INX	E8	1	2

INY

Incrémentation de Y (INcrement Y register)

Y <- Y - 1		nvmxdizc	~.....~.	
On diminue de 1 le contenu du registre d'index Y.				
Mode d'adressage		Code	Oct.	Cycles
Inhérent	INY	C8	1	2

JML

Saut inconditionnel long (JUMp Long)

PC <- (adresse) PB <- Badr		nvmxdizc	
On saute à l'adresse longue contenue dans celle indiquée comme opérande.				
Mode d'adressage		Code	Oct.	Cycles
Absolu indirect	JML (adr)	DC adL adH	3	6

Cette instruction force une nouvelle valeur du registre de banc de programme. Ce qui ne se produit pas avec JMP (adr) qui maintient le banc de programme à sa valeur courante.

JMP

Saut inconditionnel (JuMP to new location)

PC <- adresse		nvmxdizc	
On saute à l'adresse indiquée				
Modes d'adressage		Code	Oct.	Cycles
Absolu	JMP adr ou JMP adr	4C adL adH	3	3
Absolu long	JMP adr ou JMP >adr	5C adL adH Badr	4	4
Absolu indirect	JMP (adr) ou JMP(adr)	6C adL adH	3	5
Absolu indirect indexé	JMP (adr,X)	7C adL adH	3	6

MICROPROCESSEUR 65C816

Exemple

Le mode absolu renvoie à une adresse fixe du banc courant. Si cette adresse est relative, il faut utiliser l'instruction BRA adr qui, elle, est relogeable.

Le mode absolu long permet d'accéder à n'importe quelle adresse de l'espace 16 Mo.

JSL

Appel d'un sous-programme d'un autre banc (Jump to Sub-routine Long)

```
nvmdizc
PILE<-PB; S<-S-1
PILE<-PC;S<-S-2
PC<-adresse; PB<-Badr
```

On sauve le PB et la valeur du PC de l'instruction +2, dans la pile pour constituer l'adresse de retour, puis on saute à l'adresse indiquée.

Mode d'adressage

Mode d'adressage	JSL adr	Code	Oct.	Cycles
Absolu long		22 adrL adrH Badr	4	8

JSR

Appel d'un sous-programme (Jump to SubRoutine)

```
nvmdizc
PILE<-PC;S<-S-2
PC<-adresse
```

On sauve la valeur du PC de l'instruction +2, dans la pile pour constituer l'adresse de retour, puis on saute à l'adresse indiquée, en restant dans le banc courant.

Mode d'adressage

Mode d'adressage	JSR adr	Code	Oct.	Cycles
Absolu		20 adL adH	3	6
Absolu indexé indirect	JSR (adr,X)	FC adL adH	3	6

(L'adresse du sous-programme appelé est contenue dans l'adresse adr+X du banc courant).

LDA

Chargement de l'accumulateur (Load Accumulator)

```
nvmdizc
A<-M
~.....~
```

On met dans l'accumulateur le contenu de la mémoire indiquée (la mémoire n'est pas altérée).

Modes d'adressage

Modes d'adressage	LDA £donnée	Code	Oct	Cycles
Immédiat émulé(m=1)		A9 donnée	2	2
Immédiat natif(m=0)		A9 donL donH	3	3
Absolu	LDA adr ou adr	AD adL adH	3	4 (1)

MICROPROCESSEUR 65C816

Absolu long	LDA adr ou >adr	AF	adL adH Badr	4	5 (1)
Absolu indexé par X	LDA adr,X	BD	adL adH	3	4 (1,3)
Absolu indexé par Y	LDA adr,Y	B9	adL adH	3	4 (1,3)
Absolu long indexé	LDA adr,X ou >adr,X	BF	adL adH Badr	4	5 (1)
Direct page zéro	LDA adr ou <adr	A5	adr	2	3 (1,2)
Direct indirect	LDA (adr)	B2	adr	2	5 (1,2)
Direct indirect indexé	LDA (adr),Y	B1	adr	2	5 (1,2,3)
Direct indirect long	LDA [adr]	A7	adr	2	6 (1,2)
Direct indirect long indexé	LDA [adr],Y	B7	adr	2	6 (1,2)
Direct indexé indirect	LDA (adr,X)	A1	adr	2	6 (1,2)
Direct indexé par X	LDA adr,X	B5	adr	2	4 (1,2)
Relatif à la pile	LDA adr,S	A3	adr	2	4 (1)
Relatif à la pile indirect indexé	LDA (adr,S),Y	B3	adr	2	7 (1)

LDX

Chargement de X (LoaD X register)

X<-M

nvmxdizc

~.....~.

On met dans le registre d'index X le contenu de la mémoire indiquée (la mémoire n'est pas altérée).

Modes d'adressage

Modes d'adressage	Code	Oct.	Cycles +
Immédiat emul(x=1)	LDX £donnée	A2	donnée 2 2
Immédiat natif(x=0)	LDX £don	A2	donL donH 3 3
Absolu	LDX adr	AE	adL adH 3 4 (10)
Absolu indexé par Y	LDX adr,Y	BE	adL adH 3 4 (3,10)
Direct	LDX adr	A6	adr 2 3 (2,10)
Direct indexé par Y	LDX adr,Y	B6	adr 2 4 (2,10)

LDY

Chargement de Y (LoaD Y register)

Y<-M

nvmxdizc

~.....~.

On met dans le registre d'index Y le contenu de la mémoire indiquée (la mémoire n'est pas altérée).

Modes d'adressage

Modes d'adressage	Code	Oct.	Cycles +
Immédiat emul(x=1)	LDY £donnée	A0	donnée 2 2
Immédiat natif(x=0)	LDY £donnée	A0	donL donH 3 3
Absolu	LDY adr	AC	adL adH 3 4 (10)
Absolu indexé par X	LDY adr,X	BC	adL adH 3 4 (1,10)
Direct	LDY adr	A4	adr 2 3 (2,10)
Direct indexé par X	LDY adr,X	B4	adr 2 4 (2,10)

MICROPROCESSEUR 65C816

LSR

Décalage à droite (Arithmetic Shift Right)

Natif (m=0)	0->b ₁₅ ;b ₁₅ ->b ₁₄ ;...;b ₁ ->b ₀ ;b ₀ ->c	nvmxdizc
Emulation(m=1)	0->b ₈ ;b ₈ ->b ₇ ;...;b ₁ ->b ₀ ;b ₀ ->c	~.....~

On décale à droite (d'un bit) l'accumulateur ou une mémoire. Un zéro entre à gauche, tandis que le bit sortant à droite tombe dans la retenue.

Modes d'adressage		Code	Oct.	Cycles +
Accumulateur	LSR A	4A	1	2
Absolu	LSR adr ou adr	4E adL adH	3	6 (5)
Absolu indexé par X	LSR adr,X	5E adL adH	2	6 (5)
Direct	LSR adr ou <adr	46 adr	2	5 (2,5)
Direct indexé par X	LSR adr,X	56 adr	2	6 (2,5)

MVN

Déplacement de bloc par adresse croissante (block MoVe Negative)

dest<-source

DB<-banc de l'adresse de destination:BKD

On copie les octets du bloc source dans le bloc destination en commençant par les adresses les plus basses.

Format d'instruction	Code	Oct.	Cycles
MVN adrD,adrS	54 BKD,BKS	3	7

par octet

Exemple d'utilisation

Le second octet de l'instruction contient les 8 bits de poids les plus forts de l'adresse de destination. Le registre Y contient les 16 bits de poids faibles de l'adresse de destination. Le troisième octet de l'instruction contient les 8 bits de poids les plus forts de l'adresse-source, les autres 16 bits sont à mettre dans le registre X. L'accumulateur doit contenir le nombre d'octets à transférer. L'incrémention de X et Y est réalisée automatiquement par MVN qui décrémente aussi l'accumulateur à chaque octet copié.

MVP

Déplacement de bloc par adresses décroissantes (block MoVe Positive)

dest<-source

DB<-banc de l'adresse de destination ou BKD

Un bloc est délimité par ses deux adresses longues de début et de fin de bloc. On copie les octets du bloc source dans le bloc destination en commençant par les adresses de fin. Le registre DB de bancs de données est chargé de la valeur du banc de l'adresse de destination.

MICROPROCESSEUR 65C816

<i>Format d'instruction</i>	<i>Code</i>	<i>Oct. Cycles</i>
MVP adrD,adrS	54 BKD,BKS	3 7 par octet

Exemple d'utilisation

Le second octet de l'instruction contient les 8 bits de poids les plus forts de l'adresse de destination. Le registre Y doit contenir les 16 bits de poids faibles de l'adresse de destination. Le troisième octet de l'instruction contient les 8 bits de poids les plus forts de l'adresse-source, les autres 16 bits sont à mettre dans le registre X. Le nombre d'octets à déplacer est à mettre dans le registre Accumulateur. La décrémentation de X et Y est réalisée automatiquement par MVP qui décrémente aussi l'accumulateur à chaque octet copié. En assembleur ORCA/M, la syntaxe est la suivante :

MVP ou MVN

sans opérande si les bancs sont les bancs actuels, sinon :

adrD EQU \$50000

adrS EQU \$60000

MVP adrD,adrS

NOP

Pas d'opération (No Operation)

PC<-PC+1

nvmxdizc

.....

Instruction muette : on ne fait aucune action. La durée est de 2 cycles. Celle-ci est utilisée soit pour remplacer des instructions supprimées lors de corrections de programme, soit pour allonger des boucles de délai.

<i>Mode d'adressage</i>	<i>Code</i>	<i>Oct. Cycles</i>
Inhérent	NOP EA	1 2

ORA

Ou inclusif (OR Accumulator)

A<-AvM

nvmxdizc

~.....~.

On effectue le OU inclusif bit à bit entre l'accumulateur et la mémoire conformément aux relations suivantes :

0v0=0

0v1=1

1v0=1

1v1=1

<i>Modes d'adressage</i>	<i>Code</i>	<i>Oct. Cycles +</i>
Immédiat émulé	ORA £donnée	09 donnée 2 2
Immédiat natif	ORA £donnée	09 donL donH 3 3

MICROPROCESSEUR 65C816

Absolu	ORA adr ou adr	0D	adL adH	3	4 (1)
Absolu long	ORA adr ou >adr	0F	adL adH Badr	4	5 (1)
Absolu indexé par X	ORA adr,X	1D	adL adH	3	4 (1,3)
Absolu indexé par Y	ORA adr,Y	19	adL adH	3	4 (1,3)
Absolu long indexé	ORA adr,X ou >adr,X	1F	adL adH Badr	4	5 (1)
Direct page zéro	ORA adr ou <adr	05	adr	2	3 (1,2)
Direct indirect	ORA (adr)	12	adr	2	5 (1,2)
Direct indirect indexé	ORA (adr),Y	11	adr	2	5 (1,2,3)
Direct indirect long	ORA [adr]	07	adr	2	6 (1,2)
Direct indirect long indexé	ORA [adr],Y	17	adr	2	6 (1,2)
Direct indexé indirect	ORA (adr,X)	01	adr	2	6 (1,2)
Direct indexé par X	ORA adr,X	15	adr	2	4 (1,2)
Relatif à la pile	ORA adr,S	03	adr	2	4 (1)
Relatif à la pile Indirect indexé	ORA(adr,S),Y	13	adr	2	6 (1,2)

PEA

Empiler des données immédiates (Push Effective Absolute address on stack)

PILE<-donH ;S<-S - 1

PILE<-donL ;S<-S - 1

On place le troisième octet de l'instruction au sommet de la pile, puis on met à jour le pointeur de pile, ensuite on place le deuxième octet de l'instruction au sommet de la pile et on met à jour le pointeur de pile.

Mode d'adressage		Code	Oct.	Cycles
Immédiat	PEA don	F4 donL donH	3	5

Exemple

Empiler un pointeur de 4 octets sur la pile

PEA \$00E1 F4 E1 00

PEA \$1700 F4 00 17

PEI

Empiler un mot de la page zéro (Push Effective Indirect address on stack)

PILE<-M+1 ;S<-S-1

PILE<-M ;S<-S - 1

Le contenu de la mémoire suivante, puis le contenu de la mémoire indiquée, sont empilés au sommet de la pile. Le pointeur de pile est mis à jour en conséquence.

Mode d'adressage		Code	Oct.	Cycles
Direct	PEI adr	D4 adr	2	6

PER

Empiler la somme de la donnée et du registre compteur ordinal PC (Push Effective program counter Relative address on stack)

PILE<-PC+don+1 ;S<-S - 1

PILE<-PC +don ;S<-S - 1

La valeur empilée est la somme obtenue en ajoutant le contenu du registre PC à la donnée trouvée en opérande.

Mode d'adressage		Code	Oct.	Cycles
Immédiat	PER £don	62 donL donH	3	6

PHA

Empiler A (PusH Accumulator)

Emulation(m=1)

PILE<-AL ;S<-S - 1

nvmxdizc

Natif (m=0)

PILE<-A ; S<-S - 2

.....

On place le contenu de l'accumulateur au sommet de la pile et on met à jour le pointeur de pile. A reste intact.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHA	48	1	3

PHB

Empiler le registre banc de données DB (PusH data Bank register on stack)

PILE<-DB ;S<-S - 1

nvmxdizc

.....

On place le contenu du registre DB (1 octet) sur la pile et on met à jour le pointeur de pile.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHB	8B	1	3

Exemple d'utilisation détournée

Cette instruction permet de décrémenter le pointeur de pile d'une position quand on travaille en mode natif.

PHD

Empiler le registre Direct DR (PusH Direct register on stack)

PILE<-DR ;S<-S - 2

nvmxdizc

.....

On place le contenu du registre DR(2 octets) sur la pile et on met à jour le pointeur de pile.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHD	0B	1	4

MICROPROCESSEUR 65C816

PHK

Empiler le registre PB (PusH program banK register on stack)

nvmdizc
.....
PILE<-PB; S<-S-1

On place le contenu du registre de banc de programme PB, appelé K par le monitor, au sommet de la pile et on met à jour le pointeur de pile. PB reste intact.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHK	4B	1	3

PHP

Empiler P (PusH Processor status register)

nvmdizc
.....
PILE<-P; S<-S-1

On place le contenu du registre d'état au sommet de la pile et on met à jour le pointeur de pile. P reste intact.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHP	08	1	3

PHX

Empiler X (PusH X register)

nvmdizc
.....
Mode émulation(x=1) PILE<-XL; S<-S-1
Mode natif (x=0) PILE<-X; S<-S-2

On place le contenu du registre X au sommet de la pile et on met à jour le pointeur de pile. X reste intact.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHX	DA	1	3 (1,10)

PHY

Empiler Y (PusH Y register)

nvmdizc
.....
Mode émulation(x=1) PILE<-YL; S<-S-1
Mode natif (x=0) PILE<-Y; S<-S-2

On place le contenu du registre Y au sommet de la pile et on met à jour le pointeur de pile. Y reste intact.

Mode d'adressage		Code	Oct.	Cycles
Inhérent	PHY	5A	1	3 (1,10)

PLA

Dépiler vers A (Pull Accumulator)

nvmxdizc

Mode émulation (m=1) AL<-PILE ; S<-S+1

~.....~.

Mode natif (m=0) A<-PILE ; S<-S+2

On transfère vers A le contenu du sommet de la pile et on met à jour le pointeur de pile.

Mode d'adressage

Inhérent

PLA

Code

68

Oct. Cycles

1 4 (1)

PLB

Dépiler vers DB (Pull data Bank register from stack)

nvmxdizc

DB<-PILE ; S<-S+1

~.....~.

On transfère vers le registre de banc de données DB, appelé B par le Monitor, le contenu du sommet de la pile et on met à jour le pointeur de pile.

Mode d'adressage

Inhérent

PLB

Code

AB

Oct. Cycles

1 4

PLD

Dépiler vers DR (Pull Direct register from stack)

nvmxdizc

DR<-PILE ; S<-S+2

~.....~.

On transfère vers le registre Direct page zéro DR, appelé D par le monitor, le contenu du sommet de la pile et on met à jour le pointeur de pile.

Mode d'adressage

Inhérent

PLD

Code

2B

Oct. Cycles

1 5

PLP

Dépiler vers P (Pull Processor status from stack)

nvmxdi zc

P<-PILE ; S<-S+1

~~~~~ ~.

On transfère vers P le contenu du sommet de la pile et on met à jour le pointeur de pile.

*Mode d'adressage*

Inhérent

PLP

*Code*

28

*Oct. Cycles*

1 4

## PLX

Dépiler vers X (Pull X register)

nvmxdizc

Mode émulation (x=1) XL<-PILE ; S<-S+1

~.....~.

Mode natif (x=0) X<-PILE ; S<-S+2



## MICROPROCESSEUR 65C816

On transfère vers X le contenu du sommet de la pile et on met à jour le pointeur de pile.

| Mode d'adressage |     | Code | Oct. Cycles |
|------------------|-----|------|-------------|
| Inhérent         | PLX | FA   | 1 4 (10)    |

### PLY

Dépiler vers Y (Pull Y register)

| Mode émulation (x=1) | Y ← -PILE ; S ← -S+1 | nvmxdizc |
|----------------------|----------------------|----------|
| Mode natif (x=0)     | Y ← -PILE ; S ← -S+2 | ~.....~. |

On transfère vers Y le contenu du sommet de la pile et on met à jour le pointeur de pile.

| Mode d'adressage |     | Code | Oct. Cycles |
|------------------|-----|------|-------------|
| Inhérent         | PLY | 7A   | 1 4 (10)    |

### REP

Mise à zéro des bits de P (REset Processor status bits)

|            |           |
|------------|-----------|
| P ← ¬O & P | nvmxd izc |
|            | ~~~~~ ~~~ |

On effectue le ET entre le complément de l'opérande O et le registre d'état P en reportant le résultat dans P ; on provoque une mise à zéro des bits de P si les bits correspondants de l'opérande sont à 1.

| Mode d'adressage |          | code   | Oct. Cycles |
|------------------|----------|--------|-------------|
| Immédiat         | REP £don | C2 don | 2 3         |

### Exemple

Commutation du microprocesseur du mode natif mixte au mode natif pur (16 bits)

REP £30

### ROL

Rotation à gauche (ROtate Left)

|                 |                                                                                                                         |          |
|-----------------|-------------------------------------------------------------------------------------------------------------------------|----------|
| Natif (m=0)     | c ← -b <sub>15</sub> ; b <sub>15</sub> ← -b <sub>14</sub> ; ...; b <sub>1</sub> ← -b <sub>0</sub> ; b <sub>0</sub> ← -c | nvmxdizc |
| Emulation (m=1) | c ← -b <sub>8</sub> ; b <sub>8</sub> ← -b <sub>7</sub> ; ...; b <sub>1</sub> ← -b <sub>0</sub> ; b <sub>0</sub> ← -c    | ~.....~  |

On décale à gauche (d'un bit) l'accumulateur ou une mémoire. L'ancienne valeur du bit de retenue entre à droite, tandis que le bit sortant à gauche tombe dans la retenue. Il s'agit donc d'une rotation sur 9 bits.

| Modes d'adressage |                 | Code       | Oct. Cycles + |
|-------------------|-----------------|------------|---------------|
| Accumulateur      | ROL A           | 2A         | 1 2           |
| Absolu            | ROL adr ou  adr | 2E adL adH | 3 6 (5)       |

# MICROPROCESSEUR 65C816

|                     |                 |            |   |         |
|---------------------|-----------------|------------|---|---------|
| Absolu indexé par X | ROL adr,X       | 3E adL adH | 3 | 7 (5)   |
| Direct              | ROL adr ou <adr | 26 adr     | 2 | 5 (2,5) |
| Direct indexé par X | ROL adr,X       | 36 adr     | 2 | 6 (2,5) |

## ROR

Rotation à droite (ROtate Right)

|                |                                                                                                                |          |
|----------------|----------------------------------------------------------------------------------------------------------------|----------|
| Natif (m=0)    | c->b <sub>15</sub> ;b <sub>15</sub> ->b <sub>14</sub> ;...;b <sub>1</sub> ->b <sub>0</sub> ;b <sub>0</sub> ->c | nvmxdizc |
| Emulation(m=1) | c->b <sub>8</sub> ;b <sub>8</sub> ->b <sub>7</sub> ;...;b <sub>1</sub> ->b <sub>0</sub> ;b <sub>0</sub> ->c    | ~.....~  |

On décale à droite (d'un bit) l'accumulateur ou une mémoire. L'ancienne valeur du bit de retenue entre à gauche, tandis que le bit sortant à droite tombe dans la retenue.

| Mode d'adressage    |                 | code       | Oct. | Cycles + |
|---------------------|-----------------|------------|------|----------|
| Accumulateur        | ROR A           | 6A         | 1    | 2        |
| Absolu              | ROR adr ou  adr | 6E adL adH | 3    | 6 (5)    |
| Absolu indexé par X | ROR adr,X       | 7E adL adH | 2    | 6 (5)    |
| Direct              | ROR adr ou <adr | 66 adr     | 2    | 5 (2,5)  |
| Direct indexé par X | ROR adr,X       | 76 adr     | 2    | 6 (2,5)  |

## RTI

Retour d'interruption (ReTurn from Interrupt)

|                      |                                   |                       |
|----------------------|-----------------------------------|-----------------------|
|                      |                                   | nvmxdizc              |
| Mode émulation (e=1) | P<-PILE;S<-S+1<br>PC<-PILE;S<-S+2 | ~~~~~ (tous modifiés) |
| Mode natif (e=0)     | idem plus:<br>PB<-PILE;S<-S+1     |                       |

Retour de routine d'interruption : on récupère sur la pile PC et P qui avaient été sauvés par le mécanisme d'interruption et on met à jour le pointeur de pile. On reprend l'exécution où on en était lors de l'interruption.

| Mode d'adressage |     | Code | Oct. | Cycles |
|------------------|-----|------|------|--------|
| Inhérent         | RTI | 40   | 1    | 6 (9)  |

## RTL

Retour long de sous-programme (ReTurn from subroutine Long)

|                 |          |
|-----------------|----------|
|                 | nvmxdizc |
| PC<-PILE;S<-S+2 | .....    |
| PC<-PC+1        |          |
| PB<-PILE;S<-S+1 |          |

On récupère sur la pile le PC et le PB qui avaient été sauvés par le dernier JSL. On reprend donc l'exécution derrière l'instruction d'appel du sous-programme.

| Mode d'adressage |     | Code | Oct. | Cycles |
|------------------|-----|------|------|--------|
| Inhérent         | RTL | 6B   | 1    | 6      |



# MICROPROCESSEUR 65C816

## RTS

Retour de sous-programme (ReTurn from Subroutine)

PC<-PILE; S<-S+2  
PC<-PC+1

nvmxdizc  
.....

On récupère sur la pile le PC qui avait été sauvé par le dernier JSR. On reprend donc l'exécution derrière l'instruction d'appel du sous-programme.

|                         |             |                    |
|-------------------------|-------------|--------------------|
| <i>Mode d'adressage</i> | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | RTS 60      | 1 6                |

## SBC

Soustraction avec retenue (SuBstract with Carry)

A<-A-M-~~c~~

nvmxdizc  
~~.....

On soustrait de l'accumulateur le contenu de la mémoire indiquée et aussi l'opposé de la retenue (c'est-à-dire l'emprunt). On opère en mode binaire ou décimal. En cas de résultat nul en mode décimal, l'indicateur z n'est pas positionné.

| <i>Modes d'adressage</i>          |                        | <i>code</i>     | <i>Oct. Cycles</i> |
|-----------------------------------|------------------------|-----------------|--------------------|
| Immédiat émulé                    | SBC £donnée            | E9 donnée       | 2 2                |
| Immédiat natif                    | SBC £donnée            | E9 donL donH    | 3 3                |
| Absolu                            | SBC adr ou  adr        | ED adL adH      | 3 4 (1)            |
| Absolu long                       | SBC adr ou >adr        | EF adL adH Badr | 4 5 (1)            |
| Absolu indexé par X               | SBC adr,X              | FD adL adH      | 3 4 (1,3)          |
| Absolu indexé par Y               | SBC adr,Y              | F9 adL adH      | 3 4 (1,3)          |
| Absolu long indexé                | SBC adr,X<br>ou >adr,X | FF adL adH Badr | 4 5 (1)            |
| Direct page zéro                  | SBC adr ou <adr        | E5 adr          | 2 3 (1,2)          |
| Direct indirect                   | SBC (adr)              | F2 adr          | 2 5 (1,2)          |
| Direct indirect indexé            | SBC (adr),Y            | F1 adr          | 2 5 (1,2,3)        |
| Direct indirect long              | SBC [adr]              | E7 adr          | 2 6 (1,2)          |
| Direct indirect long indexé       | SBC[adr],Y             | F7 adr          | 2 6 (1,2)          |
| Direct indexé indirect            | SBC (adr,X)            | E1 adr          | 2 6 (1,2)          |
| Direct indexé par x               | SBC adr,X              | F5 adr          | 2 4 (1,2)          |
| Relatif à la pile                 | SBC adr,S              | E3 adr          | 2 4 (1)            |
| Relatif à la pile indirect indexé | SBC(adr,S),Y           | F3 adr          | 2 7 (1)            |

## SEC

Mise à un de la retenue (SEt Carry)

c<-1

nvmxdizc  
.....1

On force à 1 le bit de retenue c (sert en particulier avant SBC pour faire une soustraction sans retenue).



# MICROPROCESSEUR 65C816

| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
|-------------------------|-----|-------------|--------------------|
| Inhérent                | SEC | 38          | 1 2                |

## Exemple

Pour passer en mode émulation :

```
SEC      ;c<1
XCE     ;e<->c
```

## SED

Mise en mode décimal (SEt Decimal mode)

```
d<-1      nvmxdizc
          ....1...
```

On force à 1 le bit d pour mettre l'unité arithmétique en mode décimal (en vue de ADC et SBC).

| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
|-------------------------|-----|-------------|--------------------|
| Inhérent                | SED | F8          | 1 2                |

## SEI

Inhibition des interruptions (SEt interrupt Inhibit flag)

```
i<-1      nvmxdizc
          .....1..
```

On force à 1 le bit d'inhibition des interruptions IRQ, donc on **masque** ces interruptions. Si la demande d'interruption est maintenue (broche  $\rightarrow$  IRQ maintenue à 0), l'interruption sera prise en compte dès que le bit sera mis à 0. Cette instruction est nécessaire dès le début d'une séquence critique d'instructions pendant laquelle les interruptions doivent être inhibées (par exemple pendant qu'on change le vecteur d'interruption).

| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
|-------------------------|-----|-------------|--------------------|
| Inhérent                | SEI | 78          | 1 2                |

## SEP

Mise à 1 des bits de P (SEt Processor status bits)

```
P <- O v P      nvmxd i zc
                ~~~~~ ~ ~
```

On effectue le OU entre le registre d'état P et l'opérande en reportant le résultat dans P. On provoque ainsi une mise à 1 des bits de P.

| <i>Modes d'adressage</i> |          | <i>Code</i> | <i>Oct. Cycles</i> |
|--------------------------|----------|-------------|--------------------|
| Immédiat                 | SEP £don | E2 don      | 2 3                |

# MICROPROCESSEUR 65C816

## Exemple

Commutation du processeur en mode natif mixte 8 bits :

```
SEC
XCE ;e<-0
SEP £30 ;m=1,x=1,XH=0,YH=0
```

## STA

Rangement de l'accumulateur (STore Accumulator)

```
M<-A nvmxdizc
 (aucune action)
```

On transfère le contenu de l'accumulateur A dans la mémoire indiquée. A reste inchangé.

| Modes d'adressage                 |                        | Code            | Oct. | Cycles + |
|-----------------------------------|------------------------|-----------------|------|----------|
| Absolu                            | STA adr ou  adr        | 8D adL adH      | 3    | 4        |
| Absolu long                       | STA adr ou >adr        | 8F adL adH Badr | 4    | 5        |
| Absolu indexé par X               | STA adr,X              | 9D adL adH      | 3    | 5 (1)    |
| Absolu indexé par Y               | STA adr,Y              | 99 adL adH      | 3    | 5 (1)    |
| Absolu long indexé                | STA adr,X<br>ou >adr,X | 9F adL adH Badr | 4    | 5 (1)    |
| Direct page zéro                  | STA adr ou <adr        | 85 adr          | 2    | 3 (1,2)  |
| Direct indirect                   | STA (adr)              | 92 adr          | 2    | 5 (1,2)  |
| Direct indirect indexé            | STA (adr),Y            | 91 adr          | 2    | 6 (1,2)  |
| Direct indirect long              | STA [adr]              | 87 adr          | 2    | 6 (1,2)  |
| Direct indirect long indexé       | STA[adr],Y             | 97 adr          | 2    | 6 (1,2)  |
| Direct indexé indirect            | STA (adr,X)            | 81 adr          | 2    | 6 (1,2)  |
| Direct indexé par x               | STA adr,X              | 95 adr          | 2    | 4 (1,2)  |
| Relatif à la pile                 | STA adr,S              | 83 adr          | 2    | 4 (1)    |
| Relatif à la pile indirect indexé | STA(adr,S),Y           | 93 adr          | 2    | 7 (1)    |

## STP

Arrêt de l'horloge (SToP the clock)

```
phase2 (OUT)<- 1 n v m x d i z c

```

Le signal d'horloge en sortie du microprocesseur est maintenu au niveau haut durant cette instruction. La reprise ne peut être assurée qu'au prochain front descendant du signal RES.

| Mode d'adressage |     | Code | Oct. | Cycles + |
|------------------|-----|------|------|----------|
| Inhérent         | STP | DB   | 1    | 3 (14)   |

# MICROPROCESSEUR 65C816

## STX

Rangement de X (STore X register)

nvmxdizc

M<-X

On transfère le contenu du registre d'index X dans la mémoire indiquée. X reste inchangé.

### Modes d'adressage

| Modes d'adressage   |           | Code       | Oct. | Cycles + |
|---------------------|-----------|------------|------|----------|
| Absolu              | STX adr   | 8E adL adH | 3    | 4 (10)   |
| Direct              | STX adr   | 86 adr     | 2    | 3 (2,10) |
| Direct indexé par Y | STX adr,Y | 96 adr     | 2    | 4 (2,10) |

## STY

Rangement de Y (STore Y register)

nvmxdizc

M<-Y

On transfère le contenu du registre d'index Y dans la mémoire indiquée. Y reste inchangé.

### Modes d'adressage

| Modes d'adressage   |                 | Code       | Oct. | Cycles + |
|---------------------|-----------------|------------|------|----------|
| Absolu              | STY adr ou  adr | 8C adL adH | 3    | 4 (10)   |
| Direct              | STY adr         | 84 adr     | 2    | 3 (2,10) |
| Direct indexé par X | STY adr,X       | 94 adr     | 2    | 4 (2,10) |

## STZ

Mise à zéro d'une mémoire (STore Zero)

nvmxdizc

M<-0

On transfère la valeur 0 dans la mémoire indiquée.

### Modes d'adressage

| Modes d'adressage   |           | Code       | Oct. | Cycles + |
|---------------------|-----------|------------|------|----------|
| Absolu              | STZ adr   | 9C adL adH | 3    | 4 (1)    |
| Absolu indexé par X | STZ adr,X | 9E adL adH | 3    | 5 (1)    |
| Direct page zéro    | STZ adr   | 64 adr     | 2    | 3 (1,2)  |
| Direct indexé par X | STZ adr,X | 74 adr     | 2    | 4 (1,2)  |

## TAX

Transfert de A dans X (Transfer A to X)

nvmxdizc

Mode émulation (x=1) XL<-AL ~.....~.  
 Mode natif (x=0) X<-A

On copie le contenu de l'accumulateur A dans le registre index X. A reste inchangé.

### Modes d'adressage

| Modes d'adressage |     | Code | Oct. | Cycles |
|-------------------|-----|------|------|--------|
| Inhérent          | TAX | AA   | 1    | 2      |



# MICROPROCESSEUR 65C816

## TAY

Transfert de A dans Y (Transfer A to Y)

Mode émulation (x=1) YL<-AL ~.....~.  
 Mode natif (x=0) Y<-A

On copie le contenu de l'accumulateur A dans le registre d'index Y. A reste inchangé.

### Modes d'adressage

|          |     |            |                    |
|----------|-----|------------|--------------------|
| Inhérent | TAY | Code<br>A8 | Oct. Cycles<br>1 2 |
|----------|-----|------------|--------------------|

## TCD

Transfert de C dans D (Transfer C to D)

D<-A  
 ~.....~.

On copie les 16 bits de l'accumulateur dans le registre direct D, registre contenant l'adresse de début de la page Zéro.

### Mode d'adressage

|          |     |            |                    |
|----------|-----|------------|--------------------|
| Inhérent | TCD | Code<br>5B | Oct. Cycles<br>1 2 |
|----------|-----|------------|--------------------|

## TCS

Transfert de C dans S (Transfer C to S)

S<-A  
 ~.....~.

On copie les 16 bits de l'accumulateur dans le registre S, pointeur de pile, registre contenant l'adresse du haut de la pile.

### Modes d'adressage

|          |     |            |                    |
|----------|-----|------------|--------------------|
| Inhérent | TDC | Code<br>1B | Oct. Cycles<br>1 2 |
|----------|-----|------------|--------------------|

### Exemple

Si on veut utiliser une partie de la pile comme page zéro :

```
TSC ;A<-S
SEC
SBC £$000A ;A<-A-10
TCS ;S<-A ;pointeur de pile décalé de
10
PHD ;ancienne page Zéro sauvegardée
TCD ;D<-A :nouvelle page Zéro
```

## TDC

Transfert de D dans C (Transfer D to C)

C<-D  
 ~.....~.

On copie le registre D dans l'accumulateur.

# MICROPROCESSEUR 65C816

| Modes d'adressage |     | Code | Oct. Cycles |
|-------------------|-----|------|-------------|
| Inhérent          | TDC | 7B   | 1 2         |

## TRB

Test et mise à zéro de bits (Test and Reset memory Bits with accumulator)

|                                                                  | n              | v              | mx   | diz | c |
|------------------------------------------------------------------|----------------|----------------|------|-----|---|
| $n \leftarrow M_7 ; v \leftarrow M_6 ; M \leftarrow A \& \neg M$ | M <sub>7</sub> | M <sub>6</sub> | .... | ~.  | . |

On copie les bits 6 et 7 de la mémoire dans les bits v et n. On effectue le ET entre le complément de l'accumulateur et la mémoire spécifiée, puis on reporte le résultat dans la mémoire : on provoque ainsi une mise à zéro des bits de la mémoire si les bits correspondants de l'accumulateur sont à 1. L'indicateur z est positionné en conséquence.

| Modes d'adressage |                 | Code       | Oct. Cycles + |
|-------------------|-----------------|------------|---------------|
| Absolu            | TRB adr ou  adr | 1C adL adH | 3 6 (5)       |
| Direct page zéro  | TRB adr         | 14         | 2 5 (2,5)     |

## TSB

Test et mise à 1 de bits (Test and Set memory Bits with accumulator)

|                                                               | n              | v              | mx   | diz | c |
|---------------------------------------------------------------|----------------|----------------|------|-----|---|
| $n \leftarrow M_7 ; v \leftarrow M_6 ; M \leftarrow A \vee M$ | M <sub>7</sub> | M <sub>6</sub> | .... | ~.  | . |

On copie les bits 6 et 7 de la mémoire dans les bits v et n. On effectue le OU entre le complément de l'accumulateur et la mémoire spécifiée, puis on reporte le résultat dans la mémoire : on provoque ainsi une mise à 1 des bits de la mémoire si les bits correspondants de l'accumulateur sont à 1. L'indicateur z est positionné en conséquence.

| Modes d'adressage |                 | Code       | Oct. Cycles + |
|-------------------|-----------------|------------|---------------|
| Absolu            | TSB adr ou  adr | 0C adL adH | 3 6 (5)       |
| Direct Page Zéro  | TSB adr         | 04 adr     | 2 5 (2,5)     |

## TSC

Transfert de S dans C (Transfer S to C)

|                  | n | v | mx | diz | c |
|------------------|---|---|----|-----|---|
| $A \leftarrow S$ | ~ | ~ | ~  | ~   | ~ |

On copie le registre S dans l'accumulateur 16 bits.

| Mode d'adressage |     | Code | Oct. Cycles |
|------------------|-----|------|-------------|
| Inhérent         | TSC | 3B   | 1 2         |

## TSX

Transfert de S dans X (Transfer S to X)

|                                         | n | v | mx | diz | c |
|-----------------------------------------|---|---|----|-----|---|
| Mode émulation (x=1) $XL \leftarrow SL$ | ~ | ~ | ~  | ~   | ~ |

## MICROPROCESSEUR 65C816

On copie les poids faibles du registre S dans les poids faibles du registre d'index X :

Mode natif (x=0) X<-S

On copie les 16 bits de S dans X.

|                         |     |             |                    |
|-------------------------|-----|-------------|--------------------|
| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | TSX | BA          | 1 2 2              |

### TXA

Transfert de X dans A (Transfer X to A)

|                      |        |          |
|----------------------|--------|----------|
|                      |        | nvmxdizc |
| Mode émulation (m=1) | AL<-XL | ~.....~. |
| Mode natif (m=0)     | A <- X |          |

On copie le contenu du registre X dans l'accumulateur.

|                         |     |             |                    |
|-------------------------|-----|-------------|--------------------|
| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | TXA | 8A          | 1 2                |

### TXS

Transfert de X dans S (Transfer X to S)

|                      |                |          |
|----------------------|----------------|----------|
|                      |                | nvmxdizc |
| Mode émulation (e=1) | SH<-01 ;SL<-XL | .....    |
| Mode natif (e=0)     | S<-X           |          |

|                         |     |             |                    |
|-------------------------|-----|-------------|--------------------|
| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | TXS | 9A          | 1 2                |

### TXY

Transfert de X dans Y (Transfer X to Y)

|                      |        |          |
|----------------------|--------|----------|
|                      |        | nvmxdizc |
| Mode émulation (x=1) | YL<-XL | ~.....~. |
| Mode natif (x=0)     | X<- Y  |          |

|                         |     |             |                    |
|-------------------------|-----|-------------|--------------------|
| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | TXY | 9B          | 1 2                |

### TYA

Transfert de Y dans A (Transfer Y to A)

|                      |        |          |
|----------------------|--------|----------|
|                      |        | nvmxdizc |
| Mode émulation (m=1) | AL<-YL | ~.....~. |
| Mode natif (m=0)     | A<-Y   |          |

On copie le registre Y dans l'accumulateur A.

|                         |     |             |                    |
|-------------------------|-----|-------------|--------------------|
| <i>Mode d'adressage</i> |     | <i>Code</i> | <i>Oct. Cycles</i> |
| Inhérent                | TYA | 98          | 1 2                |



## TYX

Transfert de Y dans X (Transfer Y to X)

n v m x d i z c

Mode émulation (x=1) XL<-YL

~ . . . . . ~ .

Mode natif (x=0) X<-Y

On copie le registre Y dans le registre X.

Mode d'adressage

Code

Oct. Cycles

Inhérent

TYX

BB

1 2

## WAI

Attente d'un signal d'interruption (WAit for Interrupt)

n v m x d i z c

RDY<-0

. . . . . . . .

Cette instruction fait passer le signal RDY au niveau logique bas, ce qui met le microprocesseur en "bas régime" dont il sortira à l'apparition d'une interruption externe (qui remettra RDY à 1). Si les interruptions sont inhibées avec i=1, alors un signal IRQ déclenchera l'exécution de l'instruction suivante.

Mode d'adressage

Code

Oct. Cycles

Inhérent

WAI

CB

1 3

Exemple

On l'utilise pour rendre minimum le temps de latence d'une interruption.

## XBA

Echange entre A (ou AH) et B (ou AL) (eXchange B and A)

n v m x d i z c

A <-> B

~ . . . . . ~ .

Les 8 bits de poids forts sont échangés avec les 8 bits de poids faibles de l'accumulateur. Les indicateurs n et z reflètent l'état de la valeur finale de AL. En mode émulation (8 bits), XBA permet d'avoir accès à B (ou AH).

Mode d'adressage

Code

Oct. Cycles

Inhérent

XBA

EB

1 3

## XCE

Echange de la retenue et du bit d'émulation (eXchange Carry and E bits)

n v m x d i z c

e <-> c

. . . . . . . ~

Cette instruction permet de changer le mode du microprocesseur, e=0 est le mode natif, e=1 est le mode émulation du 6502. Si e=1, alors XH=0, YH=0, SH=1, m=1, x=1.

Mode d'adressage

Code

Oct. Cycles

Inhérent

XCE

FB

1 2

# MICROPROCESSEUR 65C816

## Exemple

en mode natif:

```
CLC ;c<-0
XCE ;e<-0 (Natif)
```

Il faut sauvegarder le pointeur de pile avant de passer en mode natif.

## Notes sur le nombre de cycles :

### N° Modification

- 1 +1 si m=0.
- 2 +1 si DL, les poids faibles du registre Direct sont différents de zéro.
- 3 +1 si l'addition du registre d'index fait changer de page.
- 5 +2 si m=0.
- 7 +1 si le branchement a lieu.
- 8 +1 si le branchement conduit à changer de page, en mode émulation (e=1).
- 9 +1 si e=0 (mode natif).
- 10 +1 si x=0.
- 14 +cycles nécessaires au redémarrage par RES.

*Remarque* : les notes 4 et 6 n'apparaissent pas ici comme dans les spécifications officielles puisque le microprocesseur concerné est le 65816 et non les microprocesseurs compatibles comme le 6502 ou le 65C02 qui ont dans certaines instructions des temps d'exécution différents de ceux du 65816.

## Registres du 65C816

### Registre Accumulateur A (16 bits ou 8 bits)

Il sert de registre de travail pour les opérations arithmétiques et logiques. Les 8 bits de poids forts ou AH constituent le registre B en mode Natif.

### Registres d'index X et Y (16 bits ou 8 bits)

Le contenu des registres d'index est susceptible de s'ajouter à une adresse.

### Registre D ou registre Direct (16 bits)

Le contenu du registre D est ajouté aux adresses spécifiées dans les instructions en mode Direct. Ce registre contient une adresse de base d'une zone de données dénommée la "page zéro" qui s'implantera n'importe où dans le banc de mémoire \$00.

### Registre DBR ou Data Bank Register ou registre Banc de Données (8 bits)

Ses 8 bits constituent ceux des poids les plus forts (b16 à b24) de l'adresse d'une donnée choisie parmi les 256\*64K positions adressables par le 65816.

### Registre PB ou Program Bank ou registre Banc de Programme (8 bits)

Ses 8 bits constituent ceux des poids les plus forts de l'adresse de



l'instruction à exécuter, instruction implantée dans un des 256 bancs de mémoire adressables par le 65816.

## Registre PC ou Program Counter ou registre Compteur Ordinal (16 bits)

Le compteur Ordinal contient à tout instant l'adresse de la prochaine instruction à exécuter. En mode Natif, les 8 bits de plus forts poids du compteur Ordinal sont ceux du registre PBR.

## Registre S ou Stack Pointer ou Pointeur de Pile (16 bits)

La Pile est une zone de données accessibles suivant l'ordre "dernière entrée-première sortie" : le registre S contient à tout instant l'adresse de la première adresse disponible au-dessus de la pile. Par convention la pile se remplit par adresses décroissantes donc après chaque opération de rangement sur la pile (EMPILEMENT ou PUSH on stack) le registre S est automatiquement diminué de 2 ou de 1 pour pointer sur la prochaine adresse disponible en haut de la pile. Pour récupérer la dernière donnée entrée sur la pile, on donne un ordre de DESEMPILEMENT (ou PULL from stack) qui provoque automatiquement l'augmentation de 2 ou de 1 du registre S et le déchargement de la donnée vers un autre registre.

## Registre P ou Processor Register ou registre D'Etat du processeur (8 bits)

Sont regroupés dans ce registre les indicateurs du dernier résultat obtenu ainsi que les modes opératoires :

- b7 =n signe du résultat : 1 si négatif. Avec BIT, valeur de b7 de l'opérande.
- b6 =v débordement : 1 si le résultat est trop grand. Avec BIT, b6 de l'opérande.
- b5 =m longueur du registre accumulateur : 1 si A est sur 8 bits, 0 si 16bits.
- b4 =x longueur des registres X et Y : 1 si X et Y ont 8 bits de long, 0 si 16.
- =b en mode émulation seulement pour signaler l'exécution d'un BRK.
- b3 =d mode de calcul : 1 si en décimal codé binaire, 0 en binaire.
- b2 =i inhibition des interruptions : 1 si les interruptions sont masquées.
- b1 =Z résultat nul si 1.
- b0 =C retenue due a un addition ou à une soustraction ou à un décalage.
- e émulation du 6502.

### Modes de calcul

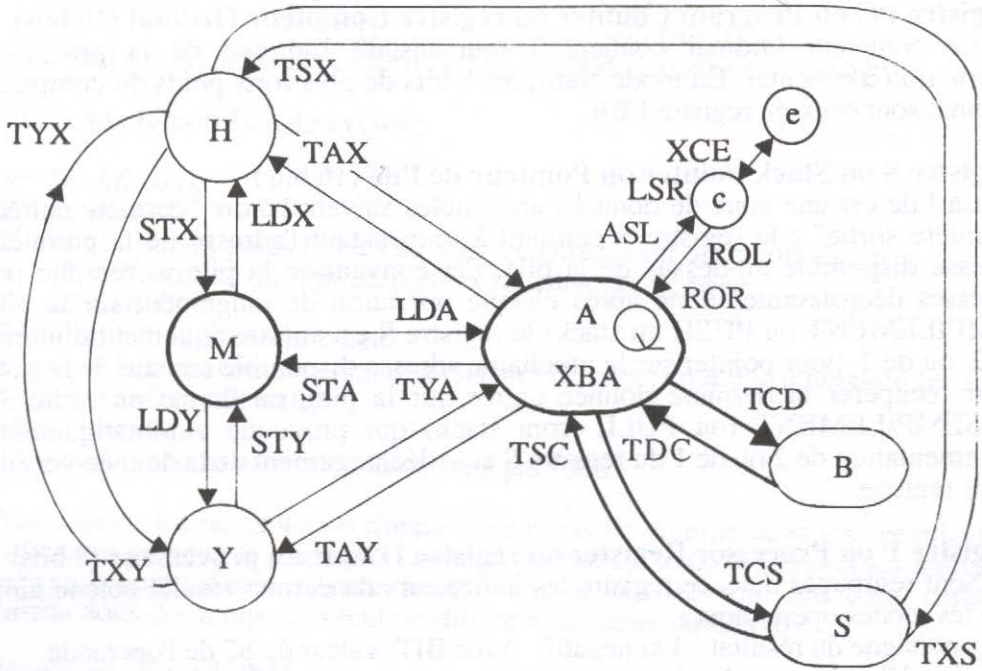
#### emx modes

- 000 NATIF PUR 65816
- 001 NATIF MIXTE 1(X et Y sur 8 bits et A sur 16 bits)
- 010 NATIF MIXTE 2(A sur 8 bits, X et Y sur 16 bits)
- 011 NATIF MIXTE 3(X et Y et A sur 8 bits)
- 111 EMULATION DU 6502
- 101 impossible
- 110 impossible
- 100 impossible



# MICROPROCESSEUR 65C816

## Echanges entre registres



1 banc = 1 apple IIe = 64 Ko

\$ xx 0000 → \$ xx FFFF

## MEMOIRES DE L'APPLE IIes

### Utilisation de l'espace-mémoire : vive, morte, extension

↳ (adressage banc)

La mémoire est divisée en bancs de 64 Koctets ; ces bancs sont adressés par les registres de bancs de données : DB, et de banc de programme : PB de 8bits.

|       |           |                                               |        |
|-------|-----------|-----------------------------------------------|--------|
| Bancs | \$00-\$01 | Mémoire vive rapide de la carte-mère          | 128 K  |
|       | \$02-\$05 | Mémoire vive rapide de la carte d'extension : | 256 K  |
| Bancs | \$02-\$12 | Mémoire vive rapide de la carte d'extension : | 1 Mega |
|       | \$02-\$3F | Mémoire vive rapide de la carte d'extension : | 4 Méga |
| Bancs | \$E0-\$E1 | Mémoire vive lente (affichage vidéo+ystème)   | 128 K  |
| Bancs | \$F0-\$FD | Mémoire morte de la carte d'extension         | 896 K  |
| Bancs | \$FE-\$FF | Mémoire morte de la carte-mère                | 128 K  |

### Occupation de la mémoire : système, utilisateur, graphique

↳ (carte mère)

#### Banc adresse

#### Mémoire rapide principale (carte mère) circuit

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| \$00 / 0000-01FF | Piles et pages-zéros demandées par l'utilisateur. (?)                 |
| \$00 / 0200-02FF | Zone de mémoire-tampon pour le clavier. (2)                           |
| \$00 / 0300-03FF | Libre.                                                                |
| \$00 / 03D0-03FF | Vecteurs et routines utilisés par ProDOS.                             |
| \$00 / 0400-07FF | Ecran-texte et le même en E0 /0400-07FF(MEV lente).                   |
| \$00 / 0800-7FFF | Piles et pages-zéros demandées par l'utilisateur.                     |
| \$00 / 8000-BFFF | Pile et système ProDOS -16.                                           |
| \$00 / C000-CFFF | Adresses d'E/S et les mêmes en \$01, \$E0 et \$E1. (1)                |
| \$00 / D000-DFFF | 2 bancs de 2 K octets de MEV (pour récupérer les 4 K précédents). (2) |
| \$00 / E000-FFFF | 12 K+4 K précédents=Language Card (LC)=16 K MEV. (6)                  |

#### Mémoire rapide auxiliaire (carte mère) circuit

|                  |                                           |
|------------------|-------------------------------------------|
| \$01 / 0000-03FF | Utilisateur.                              |
| \$01 / 0400-07FF | Ecran-text 80 colonnes (colonnes paires). |
| \$01 / 0800-BFFF | Utilisateur.                              |

#### Mémoire lente principale (carte mère) circuit

|                  |                                                                   |
|------------------|-------------------------------------------------------------------|
| \$E0 / 0000-03FF | Variation du système.                                             |
| \$E0 / 0400-07FF | Ecran-texte et certaines variables d'E/S dans les trous d'écrans. |
| \$E0 / 0800-1FFF | Texte Page 2 et buffer E/S                                        |
| \$E0 / 2000-3FFF | Ecran graphique HGR (page 1) et double haute résolution.          |
| \$E0 / 4000-5FFF | Ecran graphique HGR2 (page 2).                                    |
| \$E0 / 6000-BFFF | Utilisateur.                                                      |
| \$E0 / C000-CFFF | Adresses d'E/S et interruptions.                                  |
| \$E0 / D000-FFFF | 16 K MEV (espace 'carte-langage) réservés au système.             |



## MEMOIRES DE L'APPLE IIgs

Mémoire lente auxiliaire *contient : au bit 4.*

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| \$E1 / 0000-03FF | Vecteurs et variables-système.                            |
| \$E1 / 0400-07FF | Ecran-texte 80 colonnes (col. paires).                    |
| \$E1 / 0800-1FFF | Texte Page 2 et buffers E/S                               |
| \$E1 / 2000-3FFF | Graphique double HGR ou bien (32 pages)                   |
| \$E1 / 2000-9FFF | Graphique super haute résolution (écran quick-draw).      |
| \$E1 / A000-BFFF | Utilisateur (112 pages)                                   |
| \$E1 / C000-CFFF | Adresses réservées aux entrées/sorties.                   |
| \$E1 / D000-FFFF | 16 K MEV (espace 'carte-langage') réservés à l'AppleTalk. |

### Shadowing-ombre portée

La mémoire vive rapide (bancs \$00-\$01) a la possibilité de porter son ombre sur certaines zones de la mémoire vive lente (bancs \$E0-\$E1) avec l'option 'Shadowing' autorisée pour ces zones : il s'agit des zones-tampons d'affichage des textes et des graphiques.

Cette option, une fois autorisée, provoque l'écriture automatique et simultanée dans les bancs \$E0 et \$E1 de ce qui doit être écrit en \$00 et \$01 pour les zones choisies, aux mêmes adresses et à vitesse lente.

Il est possible d'inhiber l'option d'ombre portée sur les zones suivantes, séparément :

|                       |                                                             |                   |
|-----------------------|-------------------------------------------------------------|-------------------|
| \$E0 \$E1 / 0400-07FF | Text-40 col et 80 colonnes-Mev Principale et Mev Auxiliaire | (b <sub>0</sub> ) |
| \$E0 / 2000-3FFF      | Haute résolution graphique -page 1                          | (b <sub>1</sub> ) |
| \$E0 / 4000-5FFF      | Haute résolution graphique -page 2                          | (b <sub>2</sub> ) |
| \$E1 / 2000-9FFF      | Zone-tampon de super haute résolution graphique             | (b <sub>3</sub> ) |
| \$E1 / 2000-3FFF      | Double haute résolution                                     | (b <sub>4</sub> ) |

Les bits b<sub>0</sub> à b<sub>4</sub>, indiqués entre parenthèses, sont à mettre à 0 pour autoriser l'écriture en dédoublement et à mettre à 1 pour l'interdire ; ces bits appartiennent à un registre spécial, le shadow register (\$C035).

Le bit b<sub>6</sub> du shadow register contrôle l'usage des adresses \$Cxxx, soit en espace d'entrée/sortie, soit en mémoire vive : ce bit est appelé IOLC (input/output and Language Card) :

|        |                                                                                                                                                                                                                                                                                                    |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IOLC=1 | inhibition des adresses d'E/S et de l'espace équivalent d'adresses alloué à la carte-langage. L'espace d'adressage de la MEV est total : \$000000 à \$FFFFFF.                                                                                                                                      |
| IOLC=0 | autorisation de l'espace d'E/S et aussi de l'espace 'carte-langage' de MEV du banc2 entre \$D000 et \$DFFF (permettant à la carte-langage de disposer de 16 k d'adressage). Cet état IOLC=0 est appelé IOLC Shadowing. Il est expressément recommandé de mettre le système toujours dans cet état. |

Les bits 5 et 7 sont réservés : ils doivent contenir 0.



Par exemple pour émuler l'Apple //e, puisque les circuits d'affichage video reçoivent leurs données des bancs de mémoire lente (\$E0 et \$E1) et puisque les programmes écrits en mode émulation Apple //e, sont chargés naturellement sur le banc \$00, il faudra mettre  $b0=0, b1=0, b2=0, b3=0, b4=0, b5=0, b6=0, b7=0$  (ShadowON)=\$00.

Pour utiliser au mieux les ressources du système, avec son environnement de type "bureau électronique", c'est-à-dire l'affichage graphique en super haute résolution, et sous contrôle du memory manager, l'état du registre shadow est le suivant :

$b0=0, b1=1, b2=1, b3=1, b4=1, b5=0, b6=0, b7=0$  : (ShadowOFF)=\$1E.

Le IOLC (b6) est bien à 0 pour autoriser l'ombrage des adresses d'E/S des bancs de mémoire rapide (\$00,\$01) sur les bancs de mémoire vive lente (\$E0,\$E1).

Le bit b0 est aussi à 0 pour réaliser l'ombrage de la zone d'affichage de texte.

Toute la zone graphique super-haute résolution est gérée directement sur son propre espace d'adresses, celui de la mémoire vive lente (\$E1) grâce à  $b3=0$  qui la protège de ce qui est écrit dans le banc de mémoire vive rapide (\$01).

# RESSOURCES GRAPHIQUES

## Deux prises vidéo

- La sortie vidéo série ou vidéo composite (NTSC) pour l'affichage des textes monochromes et des graphiques où les couleurs seront remplacées par des nuances de gris, à l'aide d'un moniteur vidéo monochrome (N&B).
- Les signaux RVB (rouge vert bleu) analogiques pour l'affichage des textes en couleur et des graphiques en couleur avec un moniteur vidéo couleur (RVB).

## Modes vidéo

- texte 40 ou 80 colonnes de 24 lignes (16 couleurs) ;
- graphique basse-résolution 40 X 48 pixels (16 couleurs) ;
- graphique haute-résolution 280 X 192 pixels (8 couleurs) ;
- mixte 4 lignes de texte plus graphique basse ou haute résolution ;
- graphique double haute-résolution 560 X 192 pixels (8 couleurs) ; 140 X 192 (16 coul) ;
- graphique super haute résolution 640 X 200 pixels ou 320 X 200 pixels.

*Seize couleurs du texte, du fond de l'écran, du bord de l'écran*

|               |     |              |     |
|---------------|-----|--------------|-----|
| Noir          | \$0 | Marron       | \$8 |
| Rouge profond | \$1 | Orange       | \$9 |
| Bleu foncé    | \$2 | Gris clair   | \$A |
| Mauve         | \$3 | Rose foncé   | \$B |
| Vert foncé    | \$4 | Vert clair   | \$C |
| Gris foncé    | \$5 | Jaune        | \$D |
| Bleu moyen    | \$6 | Aigue-marine | \$E |
| Bleu clair    | \$7 | Blanc        | \$F |

Les numéros des couleurs sont les valeurs enregistrées dans les registres spécifiques :

- registre des couleurs du texte et du fond : \$C022 ;
- registre de la couleur du bord : bits 3, 2, 1, 0 de \$C034.

## *Basse-résolution, haute et double haute résolution*

Ce sont les mêmes modes que pour les modèles Apple IIe et IIc.

## *Deux nouveaux modes super haute résolution*

Résolution verticale : 200 lignes.  
Mode 0 : résolution horizontale de 320 pixels.  
Mode 1 : résolution horizontale de 640 pixels.



## RESSOURCES GRAPHIQUES

Choix des couleurs :

- dans 1 palette de 16 couleurs pour chacune des 200 lignes

Choix des palettes :

- dans 1 table de 16 palettes de 16 couleurs chacune.

La zone de mémoire utilisée occupe 32 K octets de la manière suivante :

\$E12000 à \$E19CFF : 200 X 80 octets pour les pixels (4 ou 2 par octet).

\$E19D00 à \$E19DC7 : 200 octets pour les SCB (Scan line Control Block) des lignes.

\$E19E00 à \$9FFF : 512 octets pour les 16 palettes (32 octets par palette).

### Couleurs des palettes (2 octets par couleur)

- Octet impair : bits 7, 6, 5, 4 à zéro ;  
bit 3, 2, 1, 0 : niveau de rouge (1 parmi 16).
- Octet pair : bits 7, 6, 5, 4 : niveau de vert (1 parmi 16) ;  
bits 3, 2, 1, 0 : niveau de bleu (1 parmi 16).

### Exemple des palettes standard

| N° de couleur | En mode 320 pixels/ligne |            | En mode 640 pixels/ligne |          |
|---------------|--------------------------|------------|--------------------------|----------|
|               | Contenus                 | Couleurs   | Contenus                 | Couleurs |
|               | RVB                      |            | RVB                      |          |
| 0             | 0000                     | Noir       | 0000                     | Noir 00  |
| 1             | 0777                     | Gris       | 0F00                     | Rouge 01 |
| 2             | 0841                     | Marron     | 00F0                     | Vert 10  |
| 3             | 072C                     | Mauve      | 0FFF                     | Blanc 11 |
| 4             | 000F                     | Bleu foncé | 0000                     | Noir 00  |
| 5             | 0080                     | Vert       | 000F                     | Bleu 01  |
| 6             | 0F70                     | Orange     | 0FF0                     | Jaune 10 |
| 7             | 0D00                     | Rouge      | 0FFF                     | Blanc 11 |
| 8             | 0FA9                     | Rose       | 0000                     | Noir 00  |
| 9             | 0FF0                     | Jaune      | 0F00                     | Rouge 01 |
| A             | 00E0                     | Vert clair | 00F0                     | Vert 10  |
| B             | 04DF                     | Bleu clair | 0FFF                     | Blanc 11 |
| C             | 0DAF                     | Lilas      | 0000                     | Noir 00  |
| D             | 078F                     | Bleu       | 000F                     | Bleu 01  |
| E             | 0CCC                     | Gris clair | 0FF0                     | Jaune 10 |
| F             | 0FFF                     | Blanc      | 0FF0                     | Noir 11  |

### SCB ou Scan line Control Byte

Chaque ligne est caractérisée par son SCB de la manière suivante :

Bit 7                      résolution (0=340, 1=640) ;

## RESSOURCES GRAPHIQUES

|                 |                                                                                                                                                                 |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bit 6           | interruption ;                                                                                                                                                  |
| Bit 5           | fill (en 320 seulement,) si 1 : la couleur \$0 indique qu'il faut utiliser la couleur du pixel précédent, ce qui permet un remplissage de ligne (15 couleurs) ; |
| Bit 4           | réservé; contient 0 ;                                                                                                                                           |
| Bits 3, 2, 1, 0 | n° de la palette choisie pour cette ligne.                                                                                                                      |

### Pixels

En mode 320 pixels par ligne, un pixel est codé dans un 1/2 octet, soit 4 bits dont la valeur de 0 à F est le n° d'une couleur de la palette associée à la ligne de ce pixel.

En mode 640 pixels par ligne, un pixel est codé dans 1/4 d'octet, soit 2 bits.

La valeur de couleur correspondante dépend de la position du pixel dans l'octet.

La palette est divisée en 4 classes de 4 couleurs ; la position du pixel détermine sa classe (1, 2, 3 ou 4) ; la valeur du pixel (00, 01, 10, 11) détermine la couleur dans la classe.

### Registre \$C029

|                 |                                                                                                                                                                 |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bit 7           | inhibe les modes vidéo Apple II, affiche la super- haute-résolution et rend linéaire l'utilisation des adresses graphiques (bit 7=1).                           |
| Bit 6           | linéarise les adresses de la MEV de \$2000 à \$9FFF (bit 6=1).<br>Si ce bit est à 0, on retrouve le système non linéaire des adresses graphiques.               |
| Bit 5           | met en N&B la double haute résolution avec une résolution 560 X 192 (bit 5=1).<br>Si ce bit est à 0, la double haute résolution est en 16 couleurs (140 X 192). |
| Bits 4, 3, 2, 1 | réservés ; contiennent 0.                                                                                                                                       |
| Bit 0           | Enable Bank Latch. (concerne l'accès à la MEV auxiliaire sans utiliser les commutateurs logiciels).                                                             |

*Exemple : (depuis le Moniteur-Système)*

|           |                                                                                                                                   |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------|
| *C029: A1 | affiche l'écran graphique super haute résolution.                                                                                 |
| *C029: 41 | affiche l'écran texte on tape en 'aveugle' pour y parvenir ;<br>(le bit 6 doit être à 1 pour que les adresses soient conformes ). |



# RESSOURCES GRAPHIQUES

## LISTING 1 - Exercice d'application des graphiques couleur.

\*0029:41

\*E1/9E00:00 00

\*E1/9E20:0F 00

\*E1/9E40:0E 00

\*E1/9E60:0D 00

\*E1/9E80:0C 00

\*E1/9EA0:0B 00

\*E1/9EC0:0A 00

\*01<E1/9D00.9D1F2

\*02<E1/9D20.9D3F2

\*03<E1/9D40.9D5F2

\*04<E1/9D60.9D7F2

\*05<E1/9D80.9D9F2

\*06<E1/9DA0.9DBF2

\*08<E1/9DC0.9DCF2

\*00<E1/2000.9CFF2

\*0029:A1

\*0029:41

## ENTREES/SORTIES

### Deux ports série

Ils sont utilisés pour brancher une imprimante, un modem ou un autre périphérique à transmission série, ou le relier à d'autres Apple grâce au système AppleTalk.

Le port 1 est affecté en standard à une imprimante ; le port 2 est affecté à un modem.

L'AppleTalk prend place sur un des deux ports et neutralise donc soit l'imprimante soit le modem lorsqu'il est en action.

Les fonctions intégrées dans la MEM de commande et de contrôle de ses ports sont telles qu'elles émulent les fonctions de la carte super série et celles du port Série de l'Apple //c.

La MEM assure sur commande le "buffering" en entrée ou en sortie : la zone de stockage des données est de 128 octets par défaut, mais sa taille est programmable ; ceci permet d'imprimer un texte ou une image simultanément à d'autres tâches sollicitées par l'utilisateur (Background Printing), ou bien de taper une commande au clavier alors que les opérations en cours ne sont pas terminées.

Pour mettre en service ce stockage des entrées ou des sorties, le tableau de bord propose, dans le menu options, l'option KeyBoard Buffering (Yes) et dans le menu Printer Port, l'option Buffering (Yes).

### Interface AppleTalk

C'est un réseau local pour les ordinateurs Apple II et Macintosh.

L'Apple IIGS contrôle avec son propre microprocesseur les informations transmises par les circuits de communication série, par le biais d'une routine d'interruption spécialement conçue. De plus, une interruption est générée tous les 1/4 de seconde pour que la MEM de l'AppleTalk puisse effectuer des fonctions spécifiques.

### Port disque

Il sert à contrôler des lecteurs de disquettes 5 pouces 1/4 (UniDisk ou DuoDisk) et des lecteurs de disquettes 800 K de 3 pouces 1/2 (UniDisk 3.5) reliés par une chaîne d'au plus 4 lecteurs.

Le contrôleur de ce port est le circuit intégré IWM (Integrated Woz Machine) qui était déjà sur les autres modèles Apple. Il gère aussi les connecteurs 5 et 6



sur lesquels peuvent se brancher une carte d'interface pour contrôler 2 lecteurs 5"1/4 supplémentaires, ou encore un disque dur ProFile, ou encore un disque virtuel sur une carte d'extension de MEV ou une carte ROMDisk.

La gestion en MEM des informations transmises par ces divers lecteurs, appelés périphériques d'E/S par blocs (512 octets par bloc), est assurée par le convertisseur de protocole "**Smart Port**™". Il s'agit d'un ensemble de routines dont les fonctions fondamentales sont mises à la disposition de l'application.

Le Smart Port est chargé du contrôle des lecteurs branchés en chaîne sur le port 5, ainsi que du disque virtuel /RAM5 et éventuellement d'un ROMDisk.

Les lecteurs de disquettes doivent être branchés en chaîne dans l'ordre suivant à partir de l'Apple IIGS :

- lecteurs non intelligents Sony 3.5 appelés aussi Unified Disk 3.5 (2 au plus) ;
- lecteurs intelligents Unidisk 3.5 ;
- lecteur de disquettes 5"1/4 de type DiskII ou UniDisk 5"1/4 ou DuoDisk.

En standard, au moment du démarrage, le système explore les lecteurs depuis le bout de la chaîne vers le début pour trouver une disquette d'amorçage du système d'exploitation. Mais, grâce au tableau de bord, avec l'option Startup Slot du menu Slots, le lecteur d'amorçage du système peut être fixé.

*Par exemple* : un lecteur 3"1/2 et un lecteur UniDisk 5"1/4 sont branchés en chaîne : pour amorcer avec la disquette 3"1/2, on donne la valeur 5 à Startup Slot.

Pour créer un disque virtuel sur la MEV d'extension, il suffit d'appeler l'option RAM Disk du tableau de bord et de définir sa taille par modules de 32 K. On tape 'return' pour que cette taille soit enregistrée, puis on choisit l'option Quit du tableau de bord.

Les volumes en lignes après redémarrage de ce système seront :

- disquette 5"1/4, lecteur D1, Slot 6 .D1 (pour CPW)
- disquette 3"1/2, lecteur D1, Slot 5 .D3 (pour CPW)
- disque virtuel /RAM5, lecteur D2, Slot 5 .D4 (pour CPW)

### Port manette de jeux

### Bus Apple Desktop

Il sert à brancher le clavier détachable, la souris et d'autres périphériques de saisie comme un crayon-optique ou une tablette graphique.

M  
A  
T  
E  
R  
I  
E  
L  
  
A  
P  
P  
L  
E  
  
I  
I  
G  
S

## ENTREES/SORTIES

Le clavier détachable comporte un bloc de touches numériques et est décodé par une MEM qui reconnaît 8 jeux de caractères différents.

La souris est gérée par la MEM interne en utilisant les mêmes protocoles que ceux pratiqués sur l'Apple //c, mais ses mouvements sont détectés par le microprocesseur du Bus Apple Desktop de manière autonome par rapport à l'unité centrale (sans nécessairement l'interrompre).

### Sorties vidéo

Elles sont de deux types :

- la prise RVB pour brancher un moniteur couleur à entrées RVB analogiques ;
- la prise vidéo composite pour brancher un moniteur N&B.

### Sortie haut-parleur

Grâce à son synthétiseur 15 voies et 64 K de MEV, toutes les sonorités sont possibles (en plus du simple signal sonore monovoie "beep" dont on règle aisément la fréquence et le volume sur le tableau de bord).

### Connecteurs d'E/S

Ce sont les 7 connecteurs traditionnels de l'Apple //e.

### Horloge

Elle se trouve sur la carte-mère et est alimentée par une pile longue-durée.



## Liste par ordre de priorité

Type      Source

- RESET** mise sous tension  
CTRL-PO-RESET (démarrage à froid)  
CTRL-RESET (démarrage à chaud)  
- une carte d'interface de périphérique envoie un signal de Reset.
- NMI** - une carte d'interface de périphérique envoie un signal NMI. (interruption non masquable).
- ABORT** - une carte d'extension de mémoire sur un connecteur envoie un signal d'abort sur le microprocesseur.
- COP** - une instruction COP (coprocesseur).
- BRK** - une instruction BRK.
- IRQ** - AppleTalk (la plus haute priorité) ;  
- les ports série ;  
- balayage de ligne vidéo (dès que le compteur horizontal passe à 0) ;  
- Ensoniq : 1 source pour chacun des 32 oscillateurs ;  
- VBL (tous les 20ms), Vertical Blanking à chaque fois que le faisceau passe du bas à droite au haut à gauche de l'écran ;  
- souris ; seulement si le mode interruptible a été choisi, pour les mouvements ; le bouton-poussoir ou VBL ;  
- timer au 1/4 de seconde ; AppleTalk s'en sert ;  
- clavier : dès qu'une touche est enfoncée ;  
- réponse : due au microprocesseur du clavier ;  
- SRQ : si un périphérique branché sur le Bus Apple Desktop demande la main ;  
- CTRL-PO-Esc appelle le gestionnaire d'accessoires de bureau ;  
- CTRL-PO-Delete remet à zéro le buffer du clavier (permettant de taper une commande à l'avance) en envoyant une commande Flush à d'autres claviers et en créant une interruption ;  
- Micro Abort envoyée par le microprocesseur-clavier en cas d'erreur fatale ;  
- horloge : toutes les secondes ;  
- EXTINT issu d'un dispositif connecté au Video Graphic Controller ;  
- cartes externes.

## Vecteurs d'interruptions

Deux vecteurs d'interruptions sont prévus pour chacune de ces sources suivant le mode de fonctionnement du microprocesseur :

## INTERRUPTIONS

| Type  | Natif         | Emulation     |
|-------|---------------|---------------|
| IRQ   | \$FFEE.\$FFEF | \$FFFE.\$FFFF |
| RESET | \$FFFC.\$FFFD | \$FFFC.\$FFFD |
| NMI   | \$FFEA.\$FFEB | \$FFFA.\$FFFB |
| ABORT | \$FFE8.\$FFE9 | \$FFF8.\$FFF9 |
| COP   | \$FFE4.\$FFE5 | \$FFF4.\$FFF5 |

En ayant autorisé le shadowing pour C000-CFFF (IOLC=0), on s'assure que ces vecteurs seront ceux situés en MEM.

Dans le vecteur FF/FFEE.FFEF:74 C0  
et à partir de \$C074: B8  
\$C075: 5C 10 00 E1

CLV  
JMP E10010 ; saut au gestionnaire d'IRQ.

Les différents sous-programmes de traitement des interruptions ont donc un premier point d'entrée dans l'espace \$C0xx qui renvoie à un autre vecteur en MEV, lequel renvoie à nouveau en MEM en standard (ou en MEV si l'application le désire).

\$E1/0010:5C 73 B6 FF      JMP FFB673

Pour connaître les vecteurs d'interruptions associés à chaque source, il faut se servir de l'outil MISCELLANEOUS, (n°3), et de sa fonction (n°\$11) GetVector.

Chaque gestionnaire d'interruption a un numéro de référence qu'il faut entrer dans GetVector pour récupérer l'adresse en MEM de ce gestionnaire :

| N°     | Gestionnaire d'interruption ou vecteur |
|--------|----------------------------------------|
| \$0000 | Tool Locator 1.                        |
| \$0001 | Tool Locator 2.                        |
| \$0002 | Tool Locator de l'utilisateur n°1.     |
| \$0003 | Tool Locator de l'utilisateur n°2.     |
| \$0004 | IRQ.                                   |
| \$0005 | COP.                                   |
| \$0006 | Abort.                                 |
| \$0007 | Erreur fatale.                         |
| \$0008 | AppleTalk.                             |
| \$0009 | Communication série.                   |
| \$000A | Balayage ligne.                        |
| \$000B | Son.                                   |
| \$000C | VBL.                                   |
| \$000D | Souris.                                |
| \$000E | Timer 1/4s.                            |
| \$000F | Clavier.                               |
| \$0010 | Réponse du Bus Apple Desktop.          |
| \$0011 | SRQ.                                   |
| \$0012 | Accessoires de bureau.                 |
| \$0013 | Mise à zéro du buffer du clavier.      |



## INTERRUPTIONS

|               |                                                                |
|---------------|----------------------------------------------------------------|
| \$0014        | Erreur fatale du micro du clavier.                             |
| \$0015        | 1 seconde.                                                     |
| \$0016        | VGC externe.                                                   |
| \$0017        | Autre.                                                         |
| \$0018        | Curseur.                                                       |
| \$0019        | Incrémentation de l'indicateur d'occupation pour le Scheduler. |
| \$001A        | Décrémentation de l'indicateur d'occupation pour le Scheduler. |
| \$001B        | Beep.                                                          |
| \$001C        | Brk du Debugger.                                               |
| \$001D        | Trace.                                                         |
| \$001E        | Step.                                                          |
| \$001F-\$0027 | Réservés.                                                      |
| \$0028        | CTRL-Y.                                                        |
| \$0029        | Réservé.                                                       |
| \$002A        | MLI ProDOS16.                                                  |
| \$002B        | OS.                                                            |
| \$002C        | MSGPointer.                                                    |

Pour tester rapidement la fonction GetVector à partir du moniteur :

|                       |                                             |
|-----------------------|---------------------------------------------|
| *\6 4 0 0 0 0 4 11 3U | la commande U d'appel d'un outil            |
| Tool error-> 0000     | 6 octets à empiler, dont 4 pour le résultat |
| 73 B6 FF 00           | et 0 4 pour le N° de référence.             |

\*

### Indicateurs des sources d'IRQ

| N° Ref | Indicateur       |
|--------|------------------|
| \$0000 | IRQ.INTFLAG      |
| \$0001 | IRQ.DATAREG      |
| \$0002 | IRQ.SERIAL1      |
| \$0003 | IRQ.SERIAL2      |
| \$0004 | IRQ.APLTLKHI     |
| \$0005 | compteur de tops |
| \$0006 | IRQ.VOLUME       |
| \$0007 | IRQ.ACTIVE       |
| \$0008 | IRQ.SOUNDDATA    |

Ce numéro de référence est passé comme paramètre d'entrée à la fonction GetAddr (\$16) de l'outil Miscellaneous (\$03) pour obtenir, sur le dessus de la pile, l'indicateur demandé.

**IRQ.INTFLAG** informe sur les sources suivantes :

|       |                                       |
|-------|---------------------------------------|
| bit 7 | Bouton de la souris enfoncé.          |
| bit 6 | Bouton enfoncé à la dernière lecture. |
| bit 5 | Etat de l'entrée AN3.                 |
| bit 4 | Interruption du 1/4 de seconde.       |

## INTERRUPTIONS

|       |                                              |
|-------|----------------------------------------------|
| bit 3 | VBL.                                         |
| bit 2 | Bouton de la souris (en mode non passif).    |
| bit 1 | Mouvement de la souris (en mode non passif). |
| bit 0 | La ligne IRQ a changé d'état.                |

**IRQ.DATAREG** informe sur les autres sources :

|            |                                                       |
|------------|-------------------------------------------------------|
| bit 7      | 1 (octet de réponse), 0 (octet d'état).               |
| bit 6      | Abort.                                                |
| bit 5      | CTRL-PO-Esc (accessoires de bureau demandés).         |
| bit 4      | CTRL-PO-Delete.                                       |
| bit 3      | SRQ.                                                  |
| bits 0 à 2 | 0 (pas de données ADB) ou nombre d'octets valides -1. |

### Etat des registres après un BRK

La sauvegarde des registres A, X, Y, S, D, P, B, K, PC, Etat, Shadow, CYA, Mslot, après l'exécution de l'instruction BRK, est effectuée dans une zone de 20 octets dont l'adresse est : GetAddr (\$0009).

### Autorisation ou inhibition des interruptions

| <i>N° Ref</i> | <i>Source</i>     | <i>Résultat</i> |
|---------------|-------------------|-----------------|
| 0000          | Clavier           | Autorisée       |
| 0001          | Clavier           | Inhibée         |
| 0002          | VBL               | Autorisée       |
| 0003          | VBL               | Inhibée         |
| 0004          | Timer 1/4 s       | Autorisée       |
| 0005          | Timer 1/4 s       | Inhibée         |
| 0006          | Timer 1 s         | Autorisée       |
| 0007          | Timer 1 s         | Inhibée         |
| 000A          | Apple Bus Desktop | Autorisée       |
| 000B          | Apple Bus Desktop | Inhibée         |
| 000C          | Balayage de ligne | Autorisée       |
| 000D          | Balayage de ligne | Inhibée         |
| 000E          | VGC externe       | Autorisée       |
| 000F          | VGC externe       | Inhibée         |

Ce numéro de référence est à passer comme paramètre d'entrée à la fonction IntSource de l'outil Miscellaneous (fonction n°\$23, outil n°\$03) afin d'obtenir le résultat indiqué.



## File d'attente du HEARTBEAT (battement de coeur)

Le signal VBL, qui se produit toutes les 20 ms (1/50 s), est le battement de coeur qui va déclencher des tâches en attente.

Une tâche est installée dans la file du Heartbeat grâce à la fonction SetHeartBeat (n° 12) de l'outil Miscellaneous (\$00). Le paramètre à passer est le pointeur de l'en-tête de cette tâche (task header).

### *Structure d'un Task header*

- 4 octets destinés au pointeur de la tâche suivante (\$00000000 si c'est la dernière) ;
- 2 octets pour le nombre de VBL à compter avant de déclencher cette tâche ;
- 2 octets de signature de la tâche.

La tâche proprement dite devra réinitialiser le compteur, elle s'exécutera en mode natif avec  $m=1$  et  $x=1$ , et devra se terminer par un RTL.

La fonction DelHeartBeat (\$13) enlève la tâche spécifiée par son pointeur, de la file d'attente du HeartBeat.

La fonction ClrHeatBeat (\$14) supprime toutes les tâches de la file.

## REGISTRES D'ETAT

|        |                              |                                            |
|--------|------------------------------|--------------------------------------------|
| \$C029 | Video Select Register        | Sélecteur de mode vidéo                    |
| \$C02B | Language Select Register     | Sélecteur de caractères internationaux     |
| \$C02D | Slot ROM Register            | Sélecteur de MEM d'interface               |
| \$C02E | Vertical Count Register      | Compteur vidéo vertical                    |
| \$C02F | Horizontal Count Register    | Compteur vidéo horizontal                  |
| \$C035 | Shadow Register              | Sélecteur de zones "ombrées"               |
| \$C036 | CYA (Configuration) Register | Sélecteur de configuration                 |
| \$C037 | DMA Bank Register            | Sélecteur de banc à accès direct           |
| \$C068 | State Register               | Registre d'état des commutateurs logiciels |

L'écriture ou la lecture sur ces registres n'est possible que si IOLC est à 0, bit 6 = 0 dans \$C035, c'est-à-dire si l'espace Cxxx est réservé aux E/S et non à la MEV, et que les adresses D000-FFFF donnent accès à la carte-langage de 16 K de MEV.

### \$C029 : Video Select Register

- Bit 7** inhibe les modes vidéo Apple II, affiche la super-haute-résolution et rend linéaire l'utilisation des adresses graphiques (bit 7=1).
- Bit 6** linéarise les adresses de la MEV de \$2000 à \$9FFF (bit 6=1). Si ce bit est à 0, on retrouve le système non linéaire des adresses graphiques.
- Bit 5** met en N&B la double haute résolution avec une résolution 560 X 192 (bit 5 =1). Si ce bit est à 0, la double haute résolution est en 16 couleurs (140 X 192).
- Bits 4, 3, 2, 1** réservés ; contiennent 0.
- Bit 0** Enable Bank Latch. (concerne l'accès à la MEV auxiliaire sans utiliser les commutateurs logiciels).

### \$C02B : Language Select Register

|                     |                            |
|---------------------|----------------------------|
| <b>Bits 7, 6, 5</b> | choix du jeu de caractères |
| 000                 | Américain                  |
| 001                 | Anglais                    |
| 010                 | Français                   |
| 011                 | Danois                     |
| 100                 | Espagnol                   |
| 101                 | Italien                    |
| 110                 | Allemand                   |
| 111                 | Suédois                    |



- Bit 4** choix de la fréquence de balayage (0 : 50 HZ, 1 : 60 HZ) pour les moniteurs Vidéo NTSC ou PAL.
- Bit 3** équivaut à l'interrupteur AZERTY/QWERTY, le mettre à 1 pour choisir les jeux de caractères internationaux.
- Bits 2 à 0** réservés et doivent rester à 0.

## \$C02D : Slot ROM Register

Les bits mis à 1 font accéder aux mémoires mortes des cartes d'interface suivantes :

- Bit 7** la carte d'interface placée sur le connecteur n° 7 : (MEM C700-C7FF, MEV C0F0-C0FF).
- Bit 6** la carte d'interface placée dans le connecteur n° 6 : (MEM C600-C6FF, MEV C0E0-C0EF).
- Bit 5** la carte d'interface placée dans le connecteur n° 5 : (MEM C500-C5FF, MEV C0D0-C0DF).
- Bit 4** la carte d'interface placée dans le connecteur n° 4 : (MEM C400-C4FF, MEV C0B0-C0BF).
- Bit 3** non utilisé (doit rester à 0).
- Bit 2** la carte d'interface placée dans le connecteur n° 2 : (MEM C200-C2FF, MEV C0A0-C0AF).
- Bit 1** la carte d'interface placée dans le connecteur n° 1 : (MEM C100-C1FF, MEV C090-C09F).

Si un de ces bits est mis à 0, c'est la MEM interne associée qui est en ligne. Pour le connecteur n° 3, c'est le commutateur logiciel SLOT3ROM qui assure l'utilisation de l'espace d'adresse C300-C3FF.

## \$C036 : Configure Your Apple Register - sélecteur de configuration

- Bit 7 Vitesse** soit 1.024 MHz (bit 7 = 0) pour le mode normal AppleII ; soit 2.8 MHz (bit 7 = 1) pour le mode rapide GS.
- Bit 6-bit 5** bits réservés à mettre à 0.
- Bit 4 Ombrage** (Shadowing Enable in all RAM Banks) : autorise les données écrites dans les espaces d'adresses ci-dessus des bancs de MEV de numéro pair ou impair, à être écrites automatiquement sur \$E0 ou \$E1 (bit 4 = 1).
 

|                              |            |
|------------------------------|------------|
| \$400-7FF                    | Text Page1 |
| \$2000-5FFF                  | HGR1,HGR2  |
| \$2000-9FFF (\$01 ou impair) | SuperHires |
| \$C000-CFFF                  | E/S        |
- bit 3** détecteur d'accès à l'adresse \$C0F8 (mise en marche du lecteur du connecteur 7).

## REGISTRES D'ETAT

|       |                                                                                   |
|-------|-----------------------------------------------------------------------------------|
| bit 2 | détecteur d'accès à l'adresse \$C0E8 (mise en marche du lecteur du connecteur 6). |
| bit 1 | détecteur d'accès à l'adresse \$C0D9 (mise-en-marche du lecteur du connecteur 5). |
| bit 0 | détecteur d'accès à l'adresse \$C0C9 (mise en marche du lecteur du connecteur 4). |

Les lecteurs de disque ou de disquettes fonctionnent à la cadence de 1.024 Mhz, ce qui oblige à un ralentissement du microprocesseur pendant les phases où les moteurs d'entraînement sont en marche. Ces phases sont reproduites dans le registre CYA et, dès qu'un de ces bits 0 à 3 est à 1, la vitesse d'exécution passe de 2.8 MHz à 1.024 MHz et, dès qu'ils passent à zéro, celle-ci redevient rapide.

Sous le Moniteur, le bit 7 du registre CYA indiquant la vitesse d'exécution est lisible ou modifiable par l'intermédiaire du registre Q. Le bit 7 de Q correspond au bit 7 du CYA (Q=80 indique une vitesse rapide).

### \$C068 : Registre d'état des commutateurs logiciels

|       |          |          |                                                          |
|-------|----------|----------|----------------------------------------------------------|
| Bit 7 | ALTZP    | (\$C016) | Page zéro en mémoire auxiliaire et carte-langage active. |
| Bit 6 | PAGE2    | (\$C01C) | TextPage2 actif.                                         |
| Bit 5 | RAMRD    | (\$C013) | Lecture en MEV principale (\$00 ou n° pair).             |
| Bit 4 | RAMWRT   | (\$C014) | Ecriture en MEV principale (\$00 ou n° pair).            |
| Bit 3 | RDRAM    | (\$C012) | Lecture seulement, sur la carte-langage.                 |
| Bit 2 | BANK2    | (\$C011) | Banc 2 de l'espace D000-DFFF de LC actif.                |
| Bit 1 | ROMBANK  | (\$C028) | Banc de MEM actif.                                       |
| Bit 0 | INTCXROM | (\$C015) | MEM interne Cs active.                                   |

L'état de chacun de ces commutateurs logiciels peut être lu ou modifié à l'aide de \$C068.

Lorsque le Moniteur est en service, l'état de ces bits est disponible dans le registre M (Machine-state) : en mode normal M=0C, c'est-à-dire :

bit 7 = 0, bit 6 = 0, bit 5 = 0, bit 4 = 0, bit 3 = 1, bit 2 = 1, bit 1 = 0, bit 0 = 0

- l'espace carte-langage n'est accessible qu'en lecture, et le banc 2 est sélectionné
- la MEM n'est pas en ligne.

Le registre L du Moniteur correspond au bit 2 de \$C068 indiquant quel banc est en ligne dans l'espace d'adresse D000 à DFFF. En demandant un désassemblage, la première ligne en haut de l'écran affiche les états de m, x et celui de L sous la forme :

0=LCBank (0/1)

1=LCBank (0/1)

si le banc 1 est sélectionné par L= 0 ;

si le banc 2 a été sélectionné par L= 1.



## TABLEAU DE BORD

Ce programme se trouve en MEM et fait partie de l'ensemble des accessoires de bureau dont une application peut avoir besoin à tout moment.

L'apparition du tableau de bord est obtenue de deux façons :

- Soit en faisant un redémarrage ; en appuyant simultanément sur OPTION-CTRL- RESET, on obtient l'écran suivant :

|                                   |                                                                              |
|-----------------------------------|------------------------------------------------------------------------------|
| 1-Enter the Control Panel.        | 1-Tableau de bord.                                                           |
| 2-Set system standards and 60 Hz. | 2-Mettre en 60 Hz pour le moniteur vidéo et attribuer les valeurs standards. |
| 3-Set system standards and 50 Hz. | 3-Mettre en 50 Hz pour le moniteur vidéo et attribuer les valeurs standards. |
| 4-Continue restarting the system. | 4-Redémarrer le système.                                                     |

Le choix à faire est 3 pour un affichage correct, puis à nouveau OPTION-CTRL\_RESET pour pouvoir choisir l'option 1 du tableau de bord.

- Soit en tapant PO-CTRL-Esc (si les interruptions n'ont pas été masquées) ; au cours de l'exécution d'un programme, ce dernier reprendra son déroulement après qu'on ait quitté le tableau de bord.

- Le menu Desk Accessories est affiché :

|                        |                                     |
|------------------------|-------------------------------------|
| Control Panel          | Tableau de bord.                    |
| Alternate Display Mode | Texte page 2 ombré ou non sur \$E0. |
| <u>Quit</u>            | <u>Quitter.</u>                     |

- En tapant flèche-en-bas puis 'return', le tableau de bord est ouvert.

- Le menu Control Panel est affiché (ainsi que l'heure et la date) :

|              |                                              |
|--------------|----------------------------------------------|
| Display      | mode et couleurs de l'écran texte.           |
| Sound        | volume et fréquence du beep sonore.          |
| System Speed | vitesse d'exécution lente ou rapide.         |
| Clock        | réglage de l'heure et de la date.            |
| Options      | options diverses du clavier et de la souris. |
| Slots        | utilisation des connecteurs d'E/S.           |
|              | choix du lecteur d'amorçage du système.      |
| Printer Port | réglages des paramètres d'impression.        |
| Modem Port   | réglages des paramètres de transmission.     |
| RAM Disk     | utilisation de la MEV en disque virtuel.     |

- Après avoir sélectionné une option, on valide par 'return' et apparaît le menu secondaire correspondant à l'option choisie.
- Pour revenir au menu Control Panel, on tape Esc.
- Pour valider une modification de paramètres, on tape 'return'.

## TABLEAU DE BORD

Les paramètres présentés sur le tableau de bord sont enregistrés dans une MEV alimentée sur pile afin de pouvoir retrouver une certaine configuration en remettant le système sous tension. Les changements demandés ne seront pris en compte que lors d'un redémarrage de la machine, sauf pour l'affichage des couleurs de l'écran texte qui se modifie sur le moment.

Les chaînes de caractères affichées par le tableau de bord sont en MEM, (à partir de FF/8AB2), mais cette adresse est référencée dans le pointeur MSGPOINTER du banc \$E1 de MEV, d'où la possibilité d'avoir une version traduite des textes du tableau de bord implantée en MEV.



## MONITEUR

CALL -151      pour y entrer  
CTRL-C        pour en sortir

### Commandes (à valider par return)

\*                      Attente d'une commande

#### - Examiner les registres

\*CTRL-E            Affiche le contenu des registres et des indicateurs sur 2 lignes si 40 colonnes, ou 1 seule ligne sur 80 colonnes.  
 A=0000 X=0000 Y=0000 S=01F4 D=0000 P=00  
 B=00 K=00 M=08 Q=80 L=1 m=1 x=1 e=1  
 Le registre D est le registre direct contenant l'adresse de début de la page zéro. Le registre B est le registre de banc de données. Le registre P est le registre de banc de programme.

#### - Examiner la mémoire

\*adresse            Affiche le contenu de l'octet d'adresse indiquée, sous 2 formes séparées par un tiret: les 2 chiffres hexadécimaux et le caractère dont le code ASCII est égal à cette valeur.  
 \*0 return  
 00/0000:4C-L  
 \*  
 (le caractère . est affiché pour tous les codes \$00 à \$1F et \$80 à \$9F qui sont ceux des caractères de contrôle).  
 Si l'adresse n'est pas spécifiée, 8 ou 16 octets sont quand même affichés, l'adresse utilisée par défaut est la dernière entrée+1 et est prise comme début de zone d'octets à afficher.

\*adr1.adr2        Affiche les contenus des octets délimités par ces 2 adresses  
 \*300.306 return  
 00/0300 : C2 CF CE CA CF D5 D2-BONJOUR.  
 \*

# MONITEUR

- si le texte est en 40 colonnes, les lignes sont de 8 octets
- si le texte est en 80 colonnes, les lignes sont de 16 octets.

A partir de la 2de ligne, l'adresse se termine toujours par 0 ou 8.

Pour interrompre cet affichage, on tape CTRL- X.

## - Modifier les registres

\*val=Nom du registre (A,X,Y,S,D,P,B,K,M,Q,L,m,x,e)  
\*FF=A  
\*CTRL-E  
A=00FF X=0000 etc

## - Modifier la mémoire

\*adresse : valeur      Ecrit la valeur indiquée à l'adresse spécifiée  
\*300 : 80 return  
\*300 return  
00 / 0300: 80-.

Les **types de valeur** sont les suivants :

- valeur hexadécimale : 2 chiffres en hexa (0 à F)
- valeur littérale : chaîne de caractères entre guillemets (max.240)
- valeur littérale retournée : chaîne de caractères entre apostrophes (longueur max. : 4) enregistrée de droite à gauche.  
\*300:"BONJOUR" return  
\*

\*adr : val1 val2 val3      Ecrit les valeurs données (séparées par un espace) dans les adresses successives depuis celle spécifiée.  
\*300:00 00 'return'  
\*300.301 'return'  
00 / 0300:00 00-..  
\*

Les 3 types de valeurs sont autorisés (hexa, littérale, retournée).

\*adr1<adr2.adr3M      Recopie les octets situés entre adr2 et adr3 à partir de l'adr1.

\*val <adr1.adr2 Z      Remplit tous les octets situés entre adr1 et adr2 de la même val. Attention de ne pas taper M au lieu de Z, ce qui modifierait une zone d'adresse val.



*- Lister un programme*

\*adr L Désassemble 20 lignes d'instructions à partir de l'adresse spécifiée en utilisant les modes actuels pour m et x.  
 \*300L'return'  
 1=m 1=x 1=LCbank (0/1) (haut de l'écran)  
 00/0300: 00 00 BRK 00  
 00/0302: CE CA CF DEC CFCA  
 00/0304: D5 D2 CMP D2,X  
 etc....(jusqu'au bas de l'écran).

*- Exécuter un programme*

\*adr G Exécute les instructions à partir de l'adresse indiquée en utilisant les valeurs courantes de m, x et e, par un saut JSR.  
 L'adresse doit se trouver dans le banc \$00  
 \*300G  
 00/0300: 00 00 BRK 00  
 A=0000 X=0000 Y=0000 S=01DD D=0000 P=30  
 B=00 K=00 M=0C Q=80 L=1 m=1 x=1 e=1  
 \*  
 (le déroulement a été interrompu dès la 1ère instruction qui était un BRK, les registres sont affichés à cause de ce BRK).

\*adr X Exécute les instructions à partir de l'adresse indiquée en utilisant les valeurs courantes de m, x et e, et par un saut JSL adr.  
 L'adresse n'est pas dans le banc \$00.  
 Le sous-programme appelé doit se terminer par une instruction RTL.

\*adr R Exécute les instructions à partir de l'adresse indiquée en utilisant les valeurs courantes de m, x, e, et par un saut JMP adr.

*- Rechercher une chaîne de caractères*

\*\val \<adr1.adr2P La valeur val à rechercher peut être des 3 types (hexa, littérale, renversée). Dès que cette valeur est trouvée dans l'espace d'adresses, l'adresse est affichée suivie d'un retour à la ligne, puis le reste de l'espace est à nouveau analysé.  
 \*"check"\<FF/0000.FFFFFP  
 FF / 8A6D:  
 \*

## MONITEUR

- *Afficher avec la couleur du fond sur un fond de la couleur du texte*

\*I Affichage inversé  
\*N Affichage normal

- *Revenir à l'écran-texte depuis l'écran graphique*

\*CTRL - T

- *Quelle heure est-il, quel jour sommes-nous ?*

\*=T Time= 8/28/86 12:06:51 AM

- *Régler le jour et l'heure*

\*=T=mm/jj/aa hh:nn:ss

- *Convertir et Calculer*

\*valhexa= Renvoie la valeur décimale.  
\*=valdec Renvoie la valeur hexadécimale.  
\*val1Opval2 Op est une des 4 opérations arithmétiques (+ - \* \_).  
Les valeurs sont comprises entre \$0000000 et \$FFFFFFF.

- *Rediriger les entrées/sorties*

\*s CTRL-K Les entrées seront saisies sur le port s (0 à 7).  
\*s CTRL-P Les caractères seront sortis à travers le port s (0 à 7).

- *Sauter à un programme par une seule commande*

\*CTRL-Y L'adresse du programme est à enregistrer au préalable en \$3FE.

- *Sortir du moniteur*

\*Q Saut indirect à l'adresse \$3D0.  
\*CTRL-C Retour à l'interpréteur Basic.



## - Appeler une fonction d'un outil

\*\P1 P2 0..0 val1 val2 ... FN \U

L'outil est caractérisé par N (par ex : \$02, Memory Manager).

La fonction est caractérisée par F (par exemple \$04, Version).

Les paramètres de la fonction sont donnés dans l'ordre d'empilement sur la pile: 0..0 sont les P2 octets empilés pour le résultat, val1 val2 sont les valeurs des paramètres d'entrée. Le nombre total d'octets empilés est P1.

Tous les appels donnent le code de l'erreur (\$0000 : pas d'erreur) et affichent les valeurs de sortie telles qu'elles sont dans la pile après l'appel d'une fonction.

\*\ 2 2 0 0 4 2 \U La fonction n°4 de l'outil n° 2

Tool error -> \$0000

00 01

Les 2 octets du résultat.

## - Appel du mini-assembleur

\*!  
!adresse:Instruction Appel du mini-assembleur.  
demande d'assemblage à l'adresse donnée

!00/300 : BRK 00

00/300 : 00 00 BRK 00

!espace'Instruction Assemble à l'adresse suivante.

! adresse : : val1 val2 Ecrit les valeurs hexadécimales en mémoire.

!adresse : "valeur littérale"

Ecrit les codes ASCII des caractères de la chaîne.

\* Fin du mini-assembleur (par Return).

## LISTING 2 - Pratique du moniteur.

!CALL-151

\*FFFFFF=

Decimal-> 16777215 (+16777215)

\*010000=

Decimal-> 65536 (+65536)

\*020000=

Decimal-> 131072 (+131072)

\*030000=

Decimal-> 196608 (+196608)

\*040000=

Decimal-> 262144 (+262144)

## MONITEUR

\*090000=

Decimal -> 589824 (+589824)

\*0A0000=

Decimal -> 655360 (+655360)

\*0B0000=

Decimal -> 720896 (+720896)

\*0C0000=

Decimal -> 786432 (+786432)

\*0D0000=

Decimal -> 851968 (+851968)

\*0E0000=

Decimal -> 917504 (+917504)

\*0F0000=

Decimal -> 983040 (+983040)

\*100000=

Decimal -> 1048576 (+1048576)

\*400000=

Decimal -> 4194304 (+4194304)

\*0

### LISTING 3 - Analyse d'un outil.

JCALL-151

\*E1/0000L

1=m 1=x 1=LCbank (0/1)

```
E1/0000: 5C B1 00 FE JMP FE00B1
E1/0004: 5C 9E 00 FE JMP FE009E
E1/0008: 5C 68 00 FE JMP FE0068
E1/000C: 5C 55 00 FE JMP FE0055
E1/0010: 5C 73 B6 FF JMP FFB673
E1/0014: 5C 65 B6 FF JMP FFB665
E1/0018: 5C 65 B6 FF JMP FFB665
E1/001C: 5C 74 A4 FF JMP FFA474
E1/0020: 5C 98 B4 FF JMP FFB498
E1/0024: 5C B8 53 FF JMP FF53B8
E1/0028: 5C 98 B4 FF JMP FFB498
E1/002C: 5C 98 B4 FF JMP FFB498
E1/0030: 5C 98 B4 FF JMP FFB498
```



```

E1/0034: 5C 98 B4 FF JMP FFB498
E1/0038: 5C 98 B4 FF JMP FFB498
E1/003C: 5C 98 B4 FF JMP FFB498
E1/0040: 5C 92 84 FF JMP FF8492
E1/0044: 5C 92 81 FF JMP FF8192
E1/0048: 5C F2 AD FE JMP FEADF2
E1/004C: 5C 98 B4 FF JMP FFB498

```

\*FE/00B1L

1=m 1=x 1=LCbank (0/1)

```

FE/00B1: 18 CLC
FE/00B2: FB XCE
FE/00B3: C2 30 REP #30
FE/00B5: B0 6B BCS 0122 (+6B)
FE/00B7: 3B TSC
FE/00B8: 38 SEC
FE/00B9: E9 0A SBC #0A
FE/00BB: 00 1B BRK 1B
FE/00BD: 0B PHD
FE/00BE: 5B TCD
FE/00BF: A9 36 LDA #36
FE/00C1: 01 85 ORA (85,X)
FE/00C3: 08 PHP
FE/00C4: A9 01 LDA #01
FE/00C6: FE 85 09 INC 0985,X
FE/00C9: AF C0 03 E1 LDA E103C0
FE/00CD: 85 05 STA 05
FE/00CF: AF C1 03 E1 LDA E103C1
FE/00D3: 85 06 STA 06
FE/00D5: AF C8 03 E1 LDA E103C8

```

\*0=m

\*0=x

\*0=e

\*B1L

0=m 0=x 1=LCbank (0/1)

```

FE/00B1: 18 CLC
FE/00B2: FB XCE
FE/00B3: C2 30 REP #30
FE/00B5: B0 6B BCS 0122 (+6B)
FE/00B7: 3B TSC
FE/00B8: 38 SEC
FE/00B9: E9 0A 00 SBC #000A
FE/00BC: 1B TCS

```





```

FE/0202: 00 3B BRK 3B
FE/0204: 38 SEC
FE/0205: E9 08 00 SBC #0008
FE/0208: 1B TCS
FE/0209: 0B PHD
FE/020A: 5B TCD
FE/020B: A2 02 01 LDX #0102
FE/020E: 22 00 00 E1 JSL E10000
FE/0212: 90 03 BCC 0217 (+03)
FE/0214: 4C A4 02 JMP 02A4
FE/0217: 22 80 16 E1 JSL E11680
FE/021B: AF C0 03 E1 LDA E103C0
FE/021F: 85 01 STA 01
FE/0221: AF C2 03 E1 LDA E103C2
FE/0225: 85 03 STA 03
FE/0227: A7 01 LDA [01]
FE/0229: 0A ASL
FE/022A: 0A ASL
FE/022B: F4 00 00 PEA 0000
FE/022E: F4 00 00 PEA 0000

```

\*FE/0203L

0=m 0=x 1=LCbank (0/1)

```

FE/0203: 3B TSC
FE/0204: 38 SEC
FE/0205: E9 08 00 SBC #0008
FE/0208: 1B TCS
FE/0209: 0B PHD
FE/020A: 5B TCD
FE/020B: A2 02 01 LDX #0102
FE/020E: 22 00 00 E1 JSL E10000
FE/0212: 90 03 BCC 0217 (+03)
FE/0214: 4C A4 02 JMP 02A4
FE/0217: 22 80 16 E1 JSL E11680
FE/021B: AF C0 03 E1 LDA E103C0
FE/021F: 85 01 STA 01
FE/0221: AF C2 03 E1 LDA E103C2
FE/0225: 85 03 STA 03
FE/0227: A7 01 LDA [01]
FE/0229: 0A ASL
FE/022A: 0A ASL
FE/022B: F4 00 00 PEA 0000
FE/022E: F4 00 00 PEA 0000

```

# MONITEUR

\*L

0=m 0=x 1=LCbank (0/1)

```

FE/0231: F4 00 00 PEA 0000
FE/0234: 48 PHA
FE/0235: F4 00 90 PEA 9000
FE/0238: F4 08 80 PEA 8008
FE/023B: F4 00 00 PEA 0000
FE/023E: F4 00 00 PEA 0000
FE/0241: A2 02 09 LDX #0902
FE/0244: 22 00 00 E1 JSL E10000
FE/0248: 90 05 BCC 024F (+05)
FE/024A: FA PLX
FE/024B: FA PLX
FE/024C: 4C A4 02 JMP 02A4
FE/024F: 68 PLA
FE/0250: 85 05 STA 05
FE/0252: 68 PLA
FE/0253: 85 07 STA 07
FE/0255: A7 05 LDA [05]
FE/0257: AA TAX
FE/0258: A0 02 00 LDY #0002
FE/025B: B7 05 LDA [05],Y

```

\*L

0=m 0=x 1=LCbank (0/1)

```

FE/025D: 85 07 STA 07
FE/025F: 8F CA 03 E1 STA E103CA
FE/0263: 86 05 STX 05
FE/0265: 8A TXA
FE/0266: 8F C8 03 E1 STA E103C8
FE/026A: A7 01 LDA [01]
FE/026C: 0A ASL
FE/026D: 0A ASL
FE/026E: A8 TAY
FE/026F: 88 DEY
FE/0270: 88 DEY
FE/0271: A9 00 00 LDA #0000
FE/0274: 97 05 STA [05],Y
FE/0276: 88 DEY
FE/0277: 88 DEY
FE/0278: 10 FA BPL 0274 (-06)
FE/027A: 20 76 03 JSR 0376
FE/027D: 20 B7 02 JSR 02B7
FE/0280: B0 22 BCS 02A4 (+22)
FE/0282: F4 03 01 PEA 0103

```



\*L

0=m 0=x 1=LCbank (0/1)

```

FE/0285: A3 01 LDA 01,S
FE/0287: AA TAX
FE/0288: 22 00 00 E1 JSL E10000
FE/028C: B0 07 BCS 0295 (+07)
FE/028E: A3 01 LDA 01,S
FE/0290: 1A INC
FE/0291: 83 01 STA 01,S
FE/0293: 80 F0 BRA 0285 (-10)
FE/0295: C9 02 00 CMP #0002
FE/0298: F0 F4 BEQ 028E (-0C)
FE/029A: A2 00 00 LDX #0000
FE/029D: C9 01 00 CMP #0001
FE/02A0: F0 01 BEQ 02A3 (+01)
FE/02A2: AA TAX
FE/02A3: 68 PLA
FE/02A4: 2B PLD
FE/02A5: 3B TSC
FE/02A6: 18 CLC
FE/02A7: 69 08 00 ADC #0008
FE/02AA: 1B TCS

```

\*L

0=m 0=x 1=LCbank (0/1)

```

FE/02AB: 8A TXA
FE/02AC: C9 01 00 CMP #0001
FE/02AF: B0 05 BCS 02B6 (+05)
FE/02B1: 22 84 16 E1 JSL E11684
FE/02B5: 18 CLC
FE/02B6: 6B RTL
FE/02B7: F4 00 00 PEA 0000
FE/02BA: F4 00 00 PEA 0000
FE/02BD: F4 00 00 PEA 0000
FE/02C0: F4 04 00 PEA 0004
FE/02C3: F4 00 90 PEA 9000
FE/02C6: F4 08 80 PEA 8008
FE/02C9: F4 00 00 PEA 0000
FE/02CC: F4 00 00 PEA 0000
FE/02CF: A2 02 09 LDX #0902
FE/02D2: 22 00 00 E1 JSL E10000
FE/02D6: 90 03 BCC 02DB (+03)
FE/02D8: FA PLX
FE/02D9: FA PLX
FE/02DA: 60 RTS

```

# MONITEUR

\*\8 4 0 0 0 0 0 0 0 1 9 1\U

Tool error-> 0000

C3 01 FE 00

\*\8 4 0 0 0 0 0 0 1 011 9 1\U

Tool error-> 0000

C3 01 FE 00

\*\C 4 0 0 0 0 0 0 0 1 0 0 0 1 9 1\U

Tool error-> 0000

C3 01 FE 00

FF/0001: 00 00 BRK 00

A=80A0 X=0000 Y=0000 S=01DE D=0000 P=73

B=00 K=FF M=0D Q=80 L=1 m=1 x=1 e=0

\*\8 4 0 0 0 0 0 1 0 1 B 1\U

Tool error-> 0012

00 00 00 00

\*\C 4 0 0 0 0 0 0 0 1 0 0 0 1 B 1\U

Tool error-> 0012

01 00 00 00

FF/0001: 00 00 BRK 00

A=80A0 X=0012 Y=0000 S=01C7 D=0000 P=73

B=00 K=FF M=09 Q=80 L=0 m=1 x=1 e=0

\*\4 2 0 0 4 1\U

Tool error-> 0001

01 04 00 00

\*\2 2 0 0 4 1\U

00/A770: 00 00 BRK 00

A=0000 X=0000 Y=0000 S=01B0 D=0000 P=80

B=00 K=00 M=09 Q=80 L=0 m=0 x=0 e=0

\*

]

ICALL-151

\*\2 2 0 0 4 3\U

Tool error-> 0000

00 01

\*\2 2 0 0 4 1\U



```
00/A770: 00 00 BRK 00
A=0000 X=0000 Y=0000 S=0195 D=0000 P=80
B=00 K=00 M=09 Q=80 L=0 m=0 x=0 e=0
*\2 2 0 0 6 3\U
```

```
Tool error-> 0000
FF FF
*\2 2 0 0 6 2\U
```

```
Tool error-> 0000
FF FF
*\2 2 0 0 6 1\U
```

```
Tool error-> 0000
FF 00
*\2 2 0 0 4 3\U
```

```
Tool error-> 0000
00 01
*\2 2 0 0 4 2\U
```

```
Tool error-> 0000
00 01
*\2 2 0 0 4 1\U
```

```
00/A770: 00 00 BRK 00
A=0000 X=0000 Y=0000 S=017E D=0000 P=80
B=00 K=00 M=09 Q=80 L=0 m=0 x=0 e=0
*
```

```
A=0000 X=0000 Y=0000 S=017E D=0000 P=80
B=00 K=00 M=09 Q=80 L=0 m=0 x=0 e=0
```

## CPW

Ce logiciel de développement intègre un éditeur, un langage de commandes et le macro-assembleur ORCA/M.

### Editeur CPW

**NEW** : nouveau texte à éditer  
(version classique - écran texte)  
**EDIT** : entrée dans l'éditeur  
**Ctrl-Q** : sortie de l'éditeur

L'écran affiche 22 lignes de texte sur 80 colonnes ainsi que 2 lignes en bas affichées sur fond blanc pour entrer des commandes et connaître le mode courant, la position des taquets de tabulation, les numéros de ligne et de colonne où se trouve le curseur, et le pourcentage de mémoire utilisée.

Les 4 touches flèches (gauche-droit-bas-haut) servent à positionner le curseur n'importe où sur l'écran.

Les touches associées aux touches spéciales (PO, CTRL, ESC) produisent l'effet indiqué qu'elles soient en majuscules ou en minuscules.

#### Modes

- **EDIT** : édition par substitution

Tout caractère tapé, excepté avec les touches spéciales, est affiché en recouvrant le caractère se trouvant sous le curseur. Le curseur est avancé d'une colonne sur la droite sauf en fin de ligne où il reste en colonne 80. Pour effacer le caractère se trouvant à gauche du curseur, il faut utiliser la touche Delete. La commande **PO-Z** fait réapparaître le caractère précédemment effacé. Pour passer en mode EDIT INSERT, il faut appuyer sur les 2 touches **PO** et **E** (voir plus loin).

- **ESC** : défilement ou effacement *insertion ligne*

Le texte va défiler de page en page ou de plusieurs lignes à la fois grâce à un facteur de répétition n (un nombre entre 1 et 32767) suivi de la lettre correspondante au **déplacement** :

**ESC nX** n écrans plus loin

**ESC nC** défilement de n lignes vers le bas (qui font augmenter le numéro de ligne où se trouve le curseur).

**ESC nW** n écrans plus haut

**ESC nE** défilement de n lignes vers le haut (qui font diminuer le numéro de ligne).

Le mode ESC permet aussi l'effacement et l'insertion :

**ESC nY** **effacement** de n lignes y compris celle où se trouve le curseur.

**ESC nB** **insertion** de n lignes d'espaces sans déplacement du curseur.

**ESC nH** **insertion** de n espaces sans déplacement du curseur.

**ESC nG** **effacement** de n caractères y compris celui où se trouvait le curseur.

*insertion = 1 caractère à droite du curseur*

*effacement = y sans plus curseur*



**ESC Delete** effacement des caractères précédant le curseur jusqu'au premier espace. En tapant PO-Z on annule cet effacement.

Tout caractère différent d'un chiffre ou des lettres X, C, W, E, Y, B, H, G ou Delete fait revenir au mode **EDIT**, comme par exemple ESC.

### - **EDIT INSERT** : insertion

Tout caractère tapé excepté les touches spéciales, est affiché là où se trouvait le curseur et est inséré dans la ligne, les caractères suivant sont décalés sur la droite automatiquement. Le curseur est avancé à la colonne suivante sauf en fin de ligne où il reste en colonne 80. La touche Delete sert à effacer le caractère précédant le curseur. La commande PO-Z fait revenir le caractère précédemment effacé.

Pour revenir au mode **EDIT**, il faut appuyer sur **PO-E**.

### - **ESC INSERT**

Le mode est le même que ESC, sauf que toute touche autre que celles de défilement ou d'insertion ou de suppression, conduit au mode **EDIT INSERT**.

### - **ESC ou EDIT SELECT** :

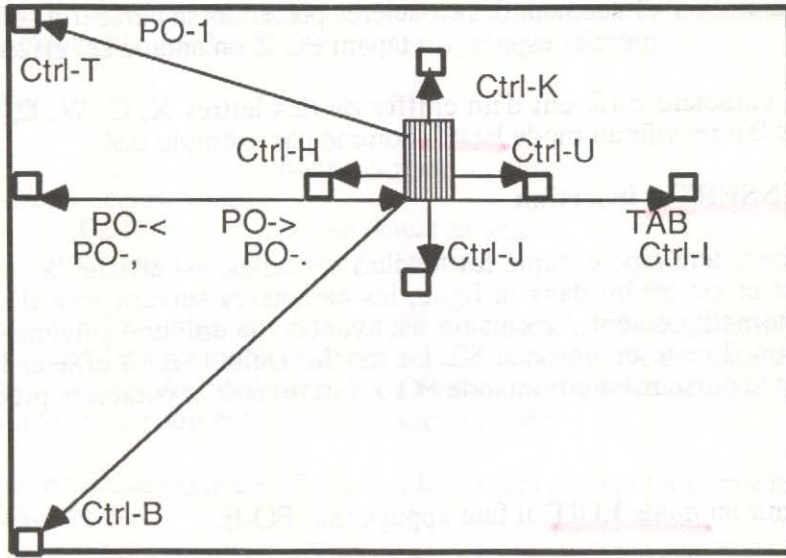
Il existe le moyen de **COPIER**, **COUPER** ou **COLLER** un bloc de lignes grâce à la touche PO (pomme) :

- PO-C** le bloc de texte sélectionné avec les flèches reste sur l'écran mais il est recopié dans le tiroir-buffer appelé **SYSTEMP** sur disquette (**COPIER**).
- PO-V** le bloc de texte préalablement recopié est collé à partir de la position du curseur (**COLLER**).
- PO-X** le bloc de texte sélectionné est **découpé**, c'est-à-dire effacé et recopié dans le tiroir-buffer **SYSTEMP** pour pouvoir le recoller autant que l'on veut et où l'on veut (**COUPER**).

Pour que la sélection ne se fasse pas par lignes mais par caractères, il faut envoyer la commande **Ctrl-PO-X** avant de couper ou copier ou coller un bloc de caractères.

- PO-Delete** effacement d'un bloc sélectionnable par les flèches.
- PO-Y** effacement depuis la position du curseur jusqu'à la fin de la ligne.
- PO-R** Ces effacements sont récupérables grâce à la commande **PO-Z**. effacement des lignes d'espaces adjacentes si le curseur est sur une telle ligne.

Dans tous les modes, on peut déplacer le curseur sur l'écran plus rapidement qu'avec les touches flèches, mais il faudra appuyer sur 2 touches.



Déplacement du curseur dans l'écran de textes

*Taquets de tabulation*

- TAB ou Ctrl-I déplace le curseur jusqu'au prochain taquet en mode d'édition par substitution.
- TAB ou Ctrl-I insère des espaces avant le prochain taquet en mode INSERT.
- PO-A ou Ctrl-A recule le curseur au taquet précédent.
- PO-I ou Ctrl-I pose un taquet sur la colonne où se trouve actuellement le curseur ou bien l'enlève s'il existe déjà. Par défaut, l'Editeur se sert des positions de taquets enregistrés dans le fichier SYSTABS et contenant les tabulations de 10 langages.

*Retour à la ligne*

(\*)

- Déplace le curseur sur le début de la ligne suivante en mode ESC.
- Insère un retour-chariot en mode EDIT INSERT.
- Fait passer le curseur à la ligne suivante en mode EDIT par recouvrement.

Après un Return, le curseur se trouve dans la colonne 1 à moins d'en avoir modifié l'effet en tapant Ctrl-PO-M, qui fait revenir le curseur sur le 1er caractère non-espace du texte de la ligne suivante ; pour revenir à la situation normale, il suffit de retaper Ctrl-PO-M.

*Frappe au km*

En tapant Ctrl-PO-W, le mode WRAP est déclenché : il reporte un mot automatiquement sur la ligne suivante s'il ne tient pas en fin de ligne. En retapant Ctrl-PO-W, le mode EDIT réapparaît.



## Fonctions de recherche et de remplacement

|           |                                                                                                                                                                                                                            |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ctrl-PO-H | positionne le curseur sur le premier caractère non-espace du mot précédent.                                                                                                                                                |
| Ctrl-PO-K | positionne le curseur sur la première ligne de texte visible sur l'écran.                                                                                                                                                  |
| Ctrl-PO-J | positionne le curseur sur la dernière ligne visible de texte sur l'écran.                                                                                                                                                  |
| PO-J      | recherche dans la suite du texte la chaîne de caractères sollicitée par Search : et la remplace par celle saisie à la demande de Replace. L'opération est faite soit en pas à pas, soit automatiquement sur tout le texte. |
| PO-H      | recherche dans la partie du texte précédant le curseur et remplace éventuellement le mot trouvé.                                                                                                                           |
| PO-K      | recherche dans la partie du texte précédant le curseur.                                                                                                                                                                    |
| PO-L      | recherche dans la partie du texte suivant le curseur et positionne au début du mot trouvé.                                                                                                                                 |

## Fin de l'édition de textes

Ctrl-Q est la commande de fin d'édition qui renvoie le menu suivant :

Editor V4.1 Phase 3 B3

Version 4.1 Phase 3B3 de l'Editeur

File name: CLEFS

Nom du fichier: CLEFS

<R> Return to editor

<R> Revenir à l'édition

<S> Save to the same name

<S> Sauvegarder sous le même nom

<N> Save to a new name

<N> Sauvegarder sous un autre nom

<L> Load another file

<L> Charger un autre fichier-texte

<E> Exit without updating

<E> Quitter l'éditeur sans rien mettre à jour

Enter selection: \_

Choisissez : \_

En répondant E, pour quitter l'éditeur sans avoir fait N ou S pour sauvegarder d'abord, une phrase vous demande de confirmer votre choix :

"About to lose changes. Are-you sure?" :

on répond Y pour le quitter ou N pour sauver le texte.

## Fichier des défauts SYSTABS

Suivant le langage choisi avant d'éditer un texte, des options par défaut sont prises par l'éditeur à partir du fichier SYSTABS présent dans le répertoire /CPW/SYSTEM.

Ce fichier SYSTABS est de type SRC, il est donc modifiable.

La structure de SYSTABS est la suivante pour chaque langage :

- la première ligne fixe les valeurs par défaut des modes

## CPW

- le 1er bit est associé à l'effet du Return
- le 2nd bit est associé au mode SELECT  
*(copy, etc...)*
- le 3e bit est associé au mode WRAP
- en début de ligne 0
- au 1er caractère non-espace de la ligne suivante 1 ;
- ligne par ligne 0
- mot par mot 1 ;
- arrêt du curseur en fin de ligne 0
- insertion d'un RC et report du mot sur la ligne suivante : 1 ;

□ la seconde ligne explicite où sont placés les taquets de tabulation par défaut: un bit à 1 indique qu'il y a un taquet, le chiffre 2 indique où se trouve la fin de la ligne.

### Création de commandes personnalisées ou macros de l'éditeur

L'utilisation de la touche FUNCT (ou OPTION) permet d'ajouter des commandes personnalisées. Dans la configuration standard, les macros commandes ont été conçues pour faciliter la présentation des listings-source :

- OPTION-A** affiche une ligne d'astérisques sur 65 colonnes.
- OPTION-B** affiche un cadre pour définir un programme.
- OPTION-C** insère 3 ; et place le curseur à la hauteur du second.
- OPTION-S** affiche un cadre de présentation d'un programme suivi de START et END.

Ces quelques exemples montrent les possibilités de ce système de type glossaire.

La procédure de modification des mots-clés et des textes envoyés débute par **PO-esc** : Editor Macro Entry est affiché ainsi que chaque commande

Une commande est modifiée en tapant sa lettre d'appel puis le nouveau texte puis **OPTION-esc** pour achever sa définition.

Pour revenir à l'éditeur de textes, il faut taper seulement **OPTION**. Les commandes ainsi modifiées peuvent être sauveées sur disquette.

**Effacer complètement le texte : PO-1, PO-Delete, PO-9, Return**

### Commandes et utilisation du CPW

£

£->

Attente d'une commande.

La touche flèche-à-droite fait apparaître la première commande ( par ordre alphabétique ), puis la touche flèche-vers-le bas la suivante, ou bien la



nota £ = " sans CPW "  
qd prog. SRC ... indiquer seul prog.  
0BS

CPW

£HELP

touche flèche-vers-le haut, la commande précédente. En tapant les premières lettres d'une commande puis flèche-à-droite, la commande cherchée apparaît ou bien, successivement, celles qui commencent par ces lettres en appuyant sur la touche flèche-en-bas sont proposées.

Liste toutes les commandes simplement par leur nom.

£ALINK

Exécute les commandes d'un fichier SRC de subtype LINKED.

£APPEND fich1 fich2

Le fichier 1 est copié à la fin du fichier 2.

£ASM65816

Le format choisi pour éditer est celui de l'assembleur 65816, et le fichier édité sera sauvé avec : SRC comme type, ASM65816 comme subtype (indique à l'éditeur le langage d'écriture)

£ASML prog.SRC

Assemble et Lie un programme-source (en \$2000).

£ASMLG prog.SRC

Assemble, Lie et Exécute le code-objet du prog.SRC.

£ASSEMBLE prog.SRC

Assemble et produit un module-objet (OMF).

£CAT

Affiche le catalogue du volume courant.

£CATALOG

Affiche le catalogue du volume courant.

£CHANGE fichier lang.

Change le langage(subtype) d'un fichier éditable.

£CMPL prog.SRC

Compile un programme-source, comme le fait ASML.

£CMPLG prog.SRC

Compile, lie et exécute un programme-source, id à ASMLG.

£COMMANDS

Permet la modification aisée de l'ensemble des Commandes.

£COMPARE fich1 fich2

Compare octet par octet 2 fichiers et affiche les différences.

£COMPILE prog.SRC

Compile un programme-source comme le fait ASSEMBLE.

£COMPRESS A|C|AC

Tri Alphabétique ou Compression du répertoire courant.

£COPY fich repertoire

Recopie le fichier indiqué dans le répertoire spécifié.

£CREATE repertoire

Créer un sous-volume sous le nom de répertoire indiqué.

£CRUNCH prog.OBJ

Combine tous les modules objets partiellement assemblés, en un seul prog.OBJ.A (non opérationnel).

£DELETE fichier

Efface le fichier indiqué.

£DCOPY vol1 vol2

Recopie tout le volume 1 dans le volume 2.

£DISABLE D fichier

Le fichier ne pourra pas être effacé.

£DISABLE N fichier

Le fichier ne pourra pas changer de nom.

£DISABLE W fichier

L'écriture sur le fichier sera impossible.

£DISABLE R fichier

La lecture du fichier sera impossible.

£DISABLE.RAM

Le disque virtuel /RAM est rendu inutilisable.

£DIV fichier taille

Division d'un fichier en sous-fichiers de la taille indiquée.



## CPW

- £DUMPOBJ prog.OBJ** Affiche le contenu d'un module-objet en format OMF ou en format désassemblé (+D) ou en hexadécimal (+H) avec son en-tête ou non (-H), avec vérification du type de fichier ou non (-F), seulement certains segments (s1, s2..).
- £EDIT fichier** Appel de l'Editeur de textes et chargement du fichier indiqué. Faire NEW pour éditer un nouveau texte.
- £ENABLE D fichier** Autorise l'effacement du fichier.
- £ENABLE N fichier** Autorise le changement de nom.
- £ENABLE B fichier** Autorise l'indication qu'une copie soit requise.
- £ENABLE W fichier** Autorise la réécriture sur le fichier.
- £ENABLE R fichier** Autorise la lecture du fichier.
- £EXEC** Le langage choisi pour éditer un fichier est EXEC, c'est-à-dire le langage des commandes CPW.; ce fichier une fois édité sera exécutable simplement en tapant son nom après £.
- £FILETYPE fichier type** Change le type du fichier spécifié en un nouveau type (code hexa \$xx, ou 3 lettres).
- £HELP Commande** Affiche le mode d'emploi de la commande donnée.
- £INIT lecteur nom** Formatage ProDOS de la disquette vierge, présente dans le lecteur spécifié, sous le nom de volume indiqué.
- £JOIN src1 src2 prog.SRC** Met dans le fichier prog.SRC, les deux fichiers indiqués.
- £LINK prog.OBJ** Appel de l'éditeur de liens, le LINKER qui va générer le code-objet à partir des modules-objets : prog.OBJ.ROOT et prog.OBJ.A issus de l'assemblage d'un programme-source prog.SRC. Le code généré par LINK est le type EXE (binaire relogeable et chargeable).
- £LINKED** Le langage choisi pour éditer un texte est le LINKED: c-a-d le langage des commandes à fournir au LINKER. Le fichier après édition sera sauvé avec SRC comme type et LINKED dans la colonne subtype du catalogue, et sera exécutable par l'ordre ALINK fichier.
- £MACGEN prog.SRC ... fichier fich. MACROS** Recherche toutes les macros utilisées dans le programme-source et génère un 'fichier' de macros sur mesure à partir de fich. MACROS où se trouvent toutes les macros du Système.
- £MAKE.REFS fich >ficref** Analyse le fichier 'fich' généré par la commande DUMPOBJ fich, en générant une directive de déclaration de référence hardware de r'proc' pour chaque procédure de nom 'proc'. Ces directives peuvent être envoyées sur un fichier 'ficref' à ajouter à un programme-source à assembler.



nota : en PRODOS.

prefix / name / → fixe un nouveau  
prefix / → vide  
prefix → affiche le contenu.

CPW

- £NEW** Appel de l'éditeur de textes pour la création d'un nouveau texte.
- £PEEK lecteur ou volume** Appel de l'éditeur de blocs et affichage en hexa ou en ASCII (esc), lecture (CTRL-R) ou écriture (CTRL-W) d'un bloc donné, déplacement dans le bloc avec les flèches et sortie de l'éditeur (CTRL-Q).
- £PREFIX repertoire** Fixe comme préfixe courant le nom du repertoire donné.
- £PRODOS** Le langage prédéfini sous l'éditeur sera du PRODOS -Texte ; un fichier édité avec ce langage aura comme type TXT au lieu de SRC (qui est du CPW-Texte).
- £QUIT** Appel de la fonction QUIT du système ProDOS qui sollicite un préfixe de volume et le nom d'une application de type SYS.
- £RENAME fich1 fich2** Le fichier1 prend le nom fich2.
- £RUN prog.SRC** Réalise l'assemblage d'un programme-source et l'édition de liens des modules-objets puis exécute le code-objet résultant (type EXE) en restant dans l'environnement CPW.
- £SET repertoire prefix** Change le nom d'un des répertoires du système CPW en lui attribuant un nouveau préfixe (comme avec SYSGEN). Cette commande peut ainsi faire partie d'un fichier EXECutable.
- £SHOW LANGUAGE** Affiche le langage courant des textes à éditer.
- £SHOW LANGUAGES** Affiche la liste de tous les langages disponibles.
- £SHOW PREFIX** Affiche le préfixe courant.
- £SHOW TIME** Affiche la date et l'heure.
- £SHOW UNITS** Affiche la liste des volumes en ligne.
- £SWITCH fich1 fich2** Echange l'ordre d'apparition des fichiers1 et 2 sur leur repertoire.
- £SYSGEN** Appelle l'utilitaire SYSTEM GENERATION de reconfiguration de CPW (caractéristiques du système, préfixes du système, caractéristiques de l'imprimante).
- £TEXT** Le langage choisi pour éditer un texte est TEXT.
- £TYPE fichier** Affiche le contenu d'un fichier TXT ou SRC. Les lignes peuvent être numérotées (+N avant le nom du fichier) et l'affichage limité entre 2 numéros de lignes(n1,n2 après le nom).
- £XREF fichier** Génère la table des références croisées des symboles du fichier ; avec l'option +F compte la fréquence des codes-opération du programme. Cet utilitaire nécessite la présence du fichier XREF.ASM65816 dans le sous-volume /CPW/UTILITIES.(la version du 30.05.86 est inopérante).

## CPW

Paramètres optionnels des commandes ASML, ASMLG, ASSEMBLE, LINK, ALINK, COMPILE, CMPL, CMPLG :

### *Mot clef et syntaxe*

**LIST ON**

**LIST OFF**

**SYMBOL ON**

**SYMBOL OFF**

**ORG=valeur**

**KEEP=fichier**

**NAMES=(sp1 sp2 ...)**

### *Utilisation*

Affiche le listing du programme-source (SRC ou LINKED).

N'affiche pas le listing.

Affiche la table des symboles.

N'affiche pas la table des symboles.

Fixe une adresse absolue d'origine au code assemblé.

Sauvegarde le fichier résultant de l'assemblage, de la compilation ou de l'édition de liens, sous le nom de fichier spécifié après le signe =.

Ne fait l'assemblage que des segments spécifiés par leur nom écrit entre des parenthèses après le signe =.

Ces commandes tapées du clavier sont prioritaires sur les directives similaires incluses dans le fichier-source.

### Caractères JOKER dans les noms des fichiers pour globaliser une commande

= Remplace n'importe quelle chaîne de caractères.

? Remplace n'importe quelle chaîne de caractères mais n'exécute la commande correspondante qu'avec une réponse Y à l'affichage du nom du fichier.

### *Exemples*

**DELETE =  
DELETE ?**

Efface TOUS les fichiers ayant le préfixe courant. Interroge à chaque fichier avant de supprimer éventuellement un ou plusieurs fichiers.

**CATALOG /CPW/= .MACROS** Affiche un extrait du répertoire où les noms de fichiers se terminent par .MACROS.

**COPY /CPW/LANGUAGES/? /RAM5** Recopie dans le volume /RAM5 certains des langages du répertoire.



## Redirection des entrées/sorties

|           |                                                |
|-----------|------------------------------------------------|
| >.PRINTER | Sortie sur Imprimante.                         |
| >fichier  | Destine le résultat au fichier spécifié.       |
| <fichier  | Les commandes d'entrée proviennent du fichier. |
| <.CONSOLE | Les commandes sont tapées au clavier.          |

*Exemples*

**CATALOG >.PRINTER** Imprime le catalogue du volume courant. Pour envoyer des codes de réglage de l'imprimante, il faut taper SYSGEN et choisir l'option PRINTER CHARACTERISTICS et entrer les codes d'impression désirés.

**ASSEMBLE PROG.SRC LIST OFF >ESSAI** Le fichier PROG.SRC est assemblé sans afficher les instructions mais en écrivant tous les messages émis pendant le déroulement de l'assemblage dans le fichier ESSAI qui gardera ainsi la trace de cet assemblage.

**TYPE >.PRINTER ESSAI** Imprime le fichier ESSAI de type SRC.

## Procédure d'assemblage en vue d'une exécution sous ProDOS/16

*Ecrire un programme-source*

£ASM65816

£NEW

éditeur de textes : écrire le texte du programme

avec KEEP PROG.OBJ

et MCOPY PROG.MACROS

CTRL-Q pour sauvegarder le texte.

N pour sauver le fichier édité sous le nom PROG.SRC (par exemple) sous le type SRC et le subtype ASM65816.

E pour quitter l'éditeur.

*Générer le fichier des MACROS nécessaires à ce programme*

£MACGEN PROG.SRC PROG.MACROS

## CPW

*Créer un fichier de commandes à faire exécuter par l'ADVANCED LINKER en temps utiles :*

£LINKED

£NEW

éditeur de textes : texte à taper :

KEEP PROG.SYS16

SYMBOL OFF

LINK/ALL PROG.OBJ

ctrl-Q

N sauvegarde sous le nom PROG.LINK ,son type est SRC et le subtype LINKED

E sortie de l'éditeur de textes.

*Créer un fichier de commandes à faire exécuter par CPW*

£EXEC

£NEW

sous l'éditeur de textes taper les lignes suivantes :

ASSEMBLE PROG.SRC

ALINK PROG.LINK

FILETYPE PROG.SYS16 \$B3

ctrl-Q

N sauvegarder ce fichier sous le nom PROG.ASS(par exemple),son type est SRC et le sous-type EXEC

E sortie de l'éditeur de textes

Taper enfin :

£PROG.ASS

L'assembleur entre en opération, il fait 2 passes par segment de programme, puis s'il n'y a pas d'erreur, l'Advanced Linker s'exécute en rassemblant les modules-objets commençant par PROG.OBJ pour en faire un code-objet chargeable et relogeable PROG.SYS16. de type EXE dont la liste et la longueur des segments est affichée.

La dernière ligne de commande de PROG.ASS modifie ce type pour en faire un segment chargeable par le System Loader sous ProDOS 16.

Vérification :

| CATALOG       | Type | Subtype  |
|---------------|------|----------|
| PROG.SRC      | SRC  | ASM65816 |
| PROG.ASS      | SRC  | EXEC     |
| PROG.MACROS   | SRC  |          |
| PROG.LINK     | SRC  | LINKED   |
| PROG.OBJ.ROOT | OBJ  |          |
| PROG.OBJ.A    | OBJ  |          |
| PROG.SYS16    | \$B3 | A=\$0000 |



## Codes des fichiers

| <i>Sous CPW</i> | <i>Sous ProDOS</i> | <i>Type</i>                                             |
|-----------------|--------------------|---------------------------------------------------------|
| SRC             | \$B0               | fichier-source (de divers langages).                    |
| TXT             | \$03               | fichier Texte ASCII.                                    |
| EXE             | \$B5               | fichier exécutable sous le SHELL de CPW.                |
| OBJ             | \$B1               | module-objet.                                           |
| OBJ             | \$B2               | fichier de bibliothèque.                                |
| \$B3            | \$16               | fichier chargeable par le System Loader sous ProDOS 16. |

de fin/subtype

**Nouvelles commandes du CPW en mode bureau électronique**  
(menus déroulants, fenêtres multiples, défilement horizontal et vertical)  
Phase 4-

|                                |                                                                                                                   |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>C</b>                       | choix du langage C pour éditer un programme-source.                                                               |
| <b>DEBUG</b>                   | appel de l'utilitaire DEBUGGER.                                                                                   |
| <b>MAKELIB prog.A prog.lib</b> | produit un fichier de bibliothèque (LIBRARY file) à partir d'un module-objet (Il faudra effacer le module .ROOT). |
| <b>MOVE fich1 répertoire</b>   | déplace le fichier 1 dans un autre volume.                                                                        |
| <b>PEDIT</b>                   | change les valeurs par défaut de l'éditeur de textes.                                                             |
| <b>PRINTER</b>                 | utilitaire de configuration d'imprimante.                                                                         |

Les paramètres optionnels des commandes d'assemblage deviennent :

|                                 |                                                                          |
|---------------------------------|--------------------------------------------------------------------------|
| <b>+L</b>                       | LIST ON.                                                                 |
| <b>-L</b>                       | LIST OFF.                                                                |
| <b>+S</b>                       | SYMBOL ON.                                                               |
| <b>-S</b>                       | SYMBOL OFF.                                                              |
| <b>langage1=(option...)....</b> | permet de fixer les options du langage1 puis du langage2.                |
| <b>langage2=(option ..)</b>     | si les modules sont générés par des compilateurs de différents langages. |

## LISTING 4 - Catalogue principal de la disquette CPW

/CPW/=

| Name          | Type | Blocks | Modified        | Created         | Access | Subtype  |
|---------------|------|--------|-----------------|-----------------|--------|----------|
| PRODOS        | SYS  | 31     | 3 APR 86 15:15  | 17 JUN 86 11:25 | DNEWR  |          |
| ORCA.SYSTEM   | SYS  | 5      | 17 JUN 86 10:34 | 17 JUN 86 10:34 | DNEWR  |          |
| ORCA.HOST     | BIN  | 13     | 18 AUG 86 13:03 | 30 MAY 84 15:41 | DNEWR  | A=\$0800 |
| RELEASE.NOTES | TXT  | 5      | 17 JUN 86 11:19 | 17 JUN 86 11:19 | DNEWR  |          |
| SYSTEM        | DIR  | 1      | 17 JUN 86 11:20 | 10 MAY 86       | DNEWR  |          |

# CPW

|                |     |   |    |        |       |    |        |       |       |
|----------------|-----|---|----|--------|-------|----|--------|-------|-------|
| LANGUAGES      | DIR | 1 | 17 | JUN 86 | 11:06 | 10 | MAY 86 |       | DNBWR |
| UTILITIES      | DIR | 2 | 17 | JUN 86 | 10:47 | 10 | MAY 86 |       | DNBWR |
| LIBRARIES      | DIR | 1 | 6  | JUN 86 | 19:31 | 10 | MAY 86 |       | DNBWR |
| CPW.MACROS     | DIR | 1 | 10 | MAY 86 |       | 10 | MAY 86 |       | DNBWR |
| SYSTEM.MACROS  | DIR | 2 | 17 | JUN 86 | 10:57 | 17 | JUN 86 | 10:52 | DNBWR |
| SYSTEM.EQUATES | DIR | 1 | 17 | JUN 86 | 10:59 | 17 | JUN 86 | 10:52 | DNBWR |
| SANE.MACROS    | DIR | 1 | 17 | JUN 86 | 11:04 | 17 | JUN 86 | 10:52 | DNBWR |
| UTILITY.MACROS | DIR | 1 | 17 | JUN 86 | 11:02 | 17 | JUN 86 | 10:52 | DNBWR |
| HELLO          | TXT | 3 | 17 | JUN 86 | 11:25 | 17 | JUN 86 | 11:25 | DNBWR |

Blocks Free: 311      Blocks Used: 1289      Total Blocks: 1600

## LISTING 5 - Catalogue de la disquette CPW en cours

/CPW/=

| Name           | Type | Blocks | Modified  | Created         | Access | Subtype        |
|----------------|------|--------|-----------|-----------------|--------|----------------|
| PRODDS         | SYS  | 31     | 3 APR 86  | 15:15 17 JUN 86 | 11:25  | DNBWR          |
| ORCA.SYSTEM    | SYS  | 5      | 17 JUN 86 | 10:34 17 JUN 86 | 10:34  | DNBWR          |
| ORCA.HOST      | BIN  | 13     | 18 AUG 86 | 13:03 30 MAY 84 | 15:41  | DNBWR A=\$0800 |
| RELEASE.NOTES  | TXT  | 5      | 17 JUN 86 | 11:19 17 JUN 86 | 11:19  | DNBWR          |
| SYSTEM         | DIR  | 1      | 18 AUG 86 | 13:19 10 MAY 86 |        | DNBWR          |
| LANGUAGES      | DIR  | 1      | 17 JUN 86 | 11:06 10 MAY 86 |        | DNBWR          |
| UTILITIES      | DIR  | 2      | 17 JUN 86 | 10:47 10 MAY 86 |        | DNBWR          |
| LIBRARIES      | DIR  | 1      | 6 JUN 86  | 19:31 10 MAY 86 |        | DNBWR          |
| CPW.MACROS     | DIR  | 1      | 10 MAY 86 | 10 MAY 86       |        | DNBWR          |
| SYSTEM.MACROS  | DIR  | 2      | 17 JUN 86 | 10:57 17 JUN 86 | 10:52  | DNBWR          |
| SYSTEM.EQUATES | DIR  | 1      | 17 JUN 86 | 10:59 17 JUN 86 | 10:52  | DNBWR          |
| SANE.MACROS    | DIR  | 1      | 17 JUN 86 | 11:04 17 JUN 86 | 10:52  | DNBWR          |
| UTILITY.MACROS | DIR  | 1      | 17 JUN 86 | 11:02 17 JUN 86 | 10:52  | DNBWR          |
| HELLO          | TXT  | 3      | 17 JUN 86 | 11:25 17 JUN 86 | 11:25  | DNBWR          |
| SYSTEMP        | TXT  | 1      | 18 AUG 86 | 13:37 18 AUG 86 | 13:37  | DNBWR          |
| IMP.CAT        | SRC  | 1      | 18 AUG 86 | 13:48 18 AUG 86 | 13:48  | DNBWR EXEC     |

Blocks Free: 309      Blocks Used: 1291      Total Blocks: 1600

/CPW/SYSTEM/=

| Name    | Type | Blocks | Modified  | Created         | Access | Subtype        |
|---------|------|--------|-----------|-----------------|--------|----------------|
| MONITOR | BIN  | 32     | 27 MAY 86 | 30 MAY 84 15:57 |        | DNBWR A=\$2000 |
| EDITOR  | BIN  | 30     | 6 JUN 86  | 16:59 6 JUN 86  | 18:52  | DNBWR A=\$2000 |
| SYSEMAC | BIN  | 8      | 2 FEB 86  | 10 MAY 86       |        | DNBWR A=\$0000 |
| SYSTABS | SRC  | 3      | 29 JAN 86 | 10 MAY 86       |        | DNBWR TEXT     |
| LOGIN   | SRC  | 1      | 21 APR 86 | 17 JUN 86       | 11:20  | DNBWR EXEC     |

Blocks Free: 309      Blocks Used: 1291      Total Blocks: 1600



/CPW/LANGUAGES/=

| Name     | Type | Blocks | Modified       | Created         | Access | Subtype  |
|----------|------|--------|----------------|-----------------|--------|----------|
| LINKED   | BIN  | 53     | 13 JUN 86      | 17 JUN 86 11:06 | DNBWR  | A=\$2000 |
| ASM65816 | BIN  | 66     | 6 JUN 86 9:11  | 6 JUN 86 18:52  | DNBWR  | A=\$2000 |
| LINKER   | BIN  | 31     | 6 JUN 86 10:52 | 6 JUN 86 18:53  | DNBWR  | A=\$2000 |

Blocks Free: 309      Blocks Used: 1291      Total Blocks: 1600

/CPW/UTILITIES/=

| Name          | Type | Blocks | Modified        | Created         | Access | Subtype  |
|---------------|------|--------|-----------------|-----------------|--------|----------|
| HELP          | DIR  | 4      | 17 JUN 86 10:45 | 10 MAY 86       | DNBWR  |          |
| COMMANDS      | BIN  | 10     | 1 JAN 85 12:00  | 10 MAY 86       | DNBWR  | A=\$2000 |
| COMPRESS      | BIN  | 13     | 15 MAR 85 12:00 | 10 MAY 86       | DNBWR  | A=\$2000 |
| DUMPOBJ       | BIN  | 40     | 6 JUN 86 15:56  | 6 JUN 86 18:55  | DNBWR  | A=\$2000 |
| DCOPY         | BIN  | 8      | 15 MAR 85 12:00 | 10 MAY 86       | DNBWR  | A=\$2000 |
| DISASM        | BIN  | 25     | 1 SEP 85 12:00  | 10 MAY 86       | DNBWR  | A=\$2000 |
| INIT          | BIN  | 35     | 27 MAY 86       | 30 MAY 84 15:55 | DNBWR  | A=\$2000 |
| MAGGEN        | BIN  | 20     | 6 JUN 86 9:45   | 6 JUN 86 18:56  | DNBWR  | A=\$2000 |
| PEEK          | BIN  | 10     | 24 85 20:53     | 10 MAY 86       | DNBWR  | A=\$2000 |
| SWITCH        | BIN  | 16     | 15 MAR 85 12:00 | 10 MAY 86       | DNBWR  | A=\$2000 |
| SYSGEN        | BIN  | 26     | 15 MAR 85 12:00 | 10 MAY 86       | DNBWR  | A=\$2000 |
| XREF          | BIN  | 15     | 21 MAY 86       | 30 MAY 84 15:56 | DNBWR  | A=\$3000 |
| XREF.ASM65816 | BIN  | 8      | 1 SEP 85 10:27  | 30 MAY 84 15:56 | DNBWR  | A=\$2000 |
| INIT.BOOT     | BIN  | 3      | 7 NOV 85 11:59  | 10 MAY 86       | DNBWR  | A=\$0000 |
| SET           | BIN  | 3      | 8 NOV 85 12:45  | 20 MAY 86       | DNBWR  | A=\$2000 |
| CRUNCH        | BIN  | 11     | 23 MAY 86       | 30 MAY 84 15:56 | DNBWR  | A=\$2000 |
| JOIN          | BIN  | 6      | 12 JAN 86       | 17 JUN 86 10:41 | DNBWR  | A=\$2000 |
| COMPARE       | BIN  | 7      | 27 JAN 86 8:11  | 17 JUN 86 10:41 | DNBWR  | A=\$2000 |
| DIV           | BIN  | 8      | 15 JAN 86       | 17 JUN 86 10:41 | DNBWR  | A=\$2000 |
| APPEND        | BIN  | 5      | 15 JAN 86 16:34 | 17 JUN 86 10:41 | DNBWR  | A=\$2000 |
| MAKE.REFS     | BIN  | 5      | 16 JAN 86 10:40 | 17 JUN 86 10:41 | DNBWR  | A=\$2000 |
| DISABLE.RAM   | BIN  | 1      | 4 MAR 86        | 17 JUN 86 10:47 | DNBWR  | A=\$2000 |

Blocks Free: 309      Blocks Used: 1291      Total Blocks: 1600

LOGICIELS DE  
 ASW  
 VE  
 PLE

# CPW

Le fichier /CPW/IBM.CAT a été composé au préalable et il contient de quoi imprimer automatiquement tous les répertoires :

Type /CPW/IMP.CAT >.PRINTER

```
PREFIX /CPW
CATALOG >.PRINTER
PREFIX /CPW/SYSTEM
CATALOG >.PRINTER
PREFIX /CPW/LANGUAGES
CATALOG >.PRINTER
PREFIX /CPW/UTILITIES
CATALOG >.PRINTER
PREFIX /CPW
```



## Macro assembleur ORCA/M 4.0 inclus dans CPW

En plus des instructions mnémoniques du microprocesseur 65816, ce langage d'assemblage reconnaît des instructions qui lui sont propres : ce sont les directives d'assemblage.

↳ p. etc. ent + des directives du pascal -

## Choix du jeu d'instructions

|                  |                                                                                     |
|------------------|-------------------------------------------------------------------------------------|
| 65816 on  off    | Permet de programmer le 65816 si ON ou un microprocesseur 6502 avec l'option OFF.   |
| 65C02 on  off    | Permet de programmer en 65C02 au lieu du 6502 (compatibilité avec //c ou //e).      |
| RENAME mné1 mné2 | L'ancien mnémorique 1 est remplacé par le nouveau mnémorique 2 d'un code-opération. |

## Choix de la longueur des registres

|                |                                                                                                                                         |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| LONGA on   off | Informe l'assembleur de la longueur du registre Accumulateur et des mots exploités en mémoire :<br>16bits = ON (m=0) ; 8bits=OFF (m=1). |
| LONGI on off   | Informe l'assembleur de la longueur des registres d'Index X et Y :<br>16bits=ON (x=0) ; 8bits=OFF (x=1).                                |

## Choix dans la présentation du code généré

|                  |                                                                                                                                                                                                                                               |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ABSADDR on   off | La colonne des adresses est sur 6 chiffres hexa pour avoir sur le listing d'assemblage les adresses absolues (ON). Par défaut, cette colonne n'en fait figurer que 4 (OFF).                                                                   |
| LIST on   off    | Affiche ou non les instructions assemblées suivantes et les erreurs éventuelles (ON par défaut).                                                                                                                                              |
| EXPAND on  off   | Fait apparaître (ON) sur le listing tous les octets (64 MAX) générés par les directives DC ou seulement les 4 premiers (OFF).                                                                                                                 |
| SETCOM colonne   | Fixe à partir de quelle colonne l'assembleur doit considérer les caractères comme des commentaires et ne plus rechercher de mnémorique (40 par défaut) ; le point-virgule s'avère inutile d'ailleurs pour débiter cette zone de commentaires. |
| PRINTER on   off | Produit ou non un listing sur imprimante (OFF par défaut).                                                                                                                                                                                    |
| TITLE 'le titre' | Affiche systématiquement le titre spécifié en tête de chaque nouvelle page du listing avec le numéro de la page. L'opérande, s'il existe, doit être une                                                                                       |

## CPW

|                      |                                                                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                      | chaîne de caractères entre apostrophes si elle contient des espaces.                                                                                                    |
| EJECT                | Provoque un saut de page sur l'imprimante.                                                                                                                              |
| SYMBOL on   off      | Affiche ou non la table des symboles (ON par défaut).                                                                                                                   |
| TRACE on   off       | Affiche ou non les directives incluses dans les macros (OFF par défaut).                                                                                                |
| ERR on   off         | Affiche (ON) ou non (OFF) les erreurs d'assemblage là où elles ont lieu, si LIST est à OFF (ON par défaut). Le nombre d'erreurs trouvées est affiché dans tous les cas. |
| MERR niveau d'erreur | Fixe le niveau d'erreur maximum toléré pour entreprendre l'édition de liens après un assemblage par ASML ou ASMLG (0 par défaut).                                       |

### *Connaître ou non le temps d'exécution*

|                  |                                                                                                                                                                                                                                                        |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| INSTIME on   off | Une colonne supplémentaire indique le nombre de cycles de chaque instruction et une marque pour ajouter des cycles si une page est franchie (*), si un branchement a lieu (') ou si le temps dépend du nombre d'octets à déplacer (+). OFF par défaut. |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### *Fixer ou non l'adresse d'implantation du code*

|                      |                                                                                                                                                                                                                             |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ORG=adresse          | Fixe l'adresse absolue d'origine du code assemblé.                                                                                                                                                                          |
| ALIGN 2 <sup>n</sup> | Impose à la prochaine instruction d'être assemblée sur un début de page en insérant des octets de valeur 0 avant.                                                                                                           |
| MEM adr1, adr2       | Cette directive réserve de l'espace-mémoire en adresses absolues et est donnée à l'éditeur de liens LINK pour qu'il n'implante pas des sous-programmes qui s'étaleraient sur cette zone (par exemple les pages graphiques). |

### *Délimiter et nommer des segments*

|       |                                                                                                                                                                   |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| START | Marque le début d'un segment de programme (code). Doit nécessairement être étiquetée par un label qui sera le nom de ce segment. C'est son point d'entrée unique. |
|       | PROGRAMME START                                                                                                                                                   |
| DATA  | Marque le début d'un segment de données (data). Doit nécessairement être étiquetée par un label qui sera le nom de ce segment.                                    |
|       | DONNEES START                                                                                                                                                     |



|                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                            |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
|-----------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|---------|-----------------------------------------------------------------------------------------------------------------------------------|-------|-----------|-------|--|------------|--|-----|
| END                                                                                                                               | Marque la fin logique d'un segment de programme ou de données. Obligatoire :                                                                                                                                                                                                                                                                                                               |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| USING                                                                                                                             | <table border="0"> <tr> <td>FIN</td> <td>END</td> </tr> <tr> <td colspan="2">Définit quel segment spécifique de données est à utiliser dans le segment de programme dans lequel se trouve USING. Par exemple :</td> </tr> <tr> <td>FAIRECECI</td> <td>START</td> </tr> <tr> <td></td> <td>USING CELA</td> </tr> </table>                                                                   | FIN  | END     | Définit quel segment spécifique de données est à utiliser dans le segment de programme dans lequel se trouve USING. Par exemple : |       | FAIRECECI | START |  | USING CELA |  |     |
| FIN                                                                                                                               | END                                                                                                                                                                                                                                                                                                                                                                                        |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| Définit quel segment spécifique de données est à utiliser dans le segment de programme dans lequel se trouve USING. Par exemple : |                                                                                                                                                                                                                                                                                                                                                                                            |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| FAIRECECI                                                                                                                         | START                                                                                                                                                                                                                                                                                                                                                                                      |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
|                                                                                                                                   | USING CELA                                                                                                                                                                                                                                                                                                                                                                                 |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| PRIVATE                                                                                                                           | Le segment commençant par cette directive (qui porte obligatoirement une étiquette) est déclaré inaccessible depuis l'extérieur du module-objet dans lequel il a été créé. (cas de modules de bibliothèque ou de modules compilés séparément).                                                                                                                                             |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| PRIVDATA                                                                                                                          | Définit un segment de données privé, c'est-à-dire inaccessible depuis l'extérieur du module-objet dont il fait partie.                                                                                                                                                                                                                                                                     |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| ENTRY                                                                                                                             | Définit un point d'entrée dans un sous-programme différent de celui situé à l'emplacement de la directive START.                                                                                                                                                                                                                                                                           |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| CASE ON   OFF                                                                                                                     | <p>Les minuscules des noms de segments ou de tout autre label sont différenciées des majuscules avec l'option ON :</p> <table border="0"> <tr> <td>PROG</td> <td>CASE ON</td> </tr> <tr> <td></td> <td>START</td> </tr> <tr> <td>Prog</td> <td>END</td> </tr> <tr> <td></td> <td>START</td> </tr> <tr> <td></td> <td>END</td> </tr> </table> <p>est constitué de 2 segments distincts.</p> | PROG | CASE ON |                                                                                                                                   | START | Prog      | END   |  | START      |  | END |
| PROG                                                                                                                              | CASE ON                                                                                                                                                                                                                                                                                                                                                                                    |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
|                                                                                                                                   | START                                                                                                                                                                                                                                                                                                                                                                                      |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| Prog                                                                                                                              | END                                                                                                                                                                                                                                                                                                                                                                                        |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
|                                                                                                                                   | START                                                                                                                                                                                                                                                                                                                                                                                      |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
|                                                                                                                                   | END                                                                                                                                                                                                                                                                                                                                                                                        |      |         |                                                                                                                                   |       |           |       |  |            |  |     |
| OBJCASE ON OFF                                                                                                                    | Cette directive est donnée pour différencier (ON) les minuscules des majuscules dans les noms des modules-objets devant être reconnus à l'extérieur de l'assemblage.                                                                                                                                                                                                                       |      |         |                                                                                                                                   |       |           |       |  |            |  |     |

*Données*

- Constantes utilisables par tous les segments de programme :
 

|                                                                                                                 |                                                                                                                          |
|-----------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| GEQU                                                                                                            | ceci est un "equate" global ; le label de cette directive est le nom de la constante et l'opérande constitue sa valeur : |
| VRAI                                                                                                            | GEQU %10000000                                                                                                           |
| Les constantes sont à définir avant de les utiliser, aussi il est d'usage de les déclarer dans le 1er segment : |                                                                                                                          |
| DEBUT                                                                                                           | START                                                                                                                    |
|                                                                                                                 | BRA PRET                                                                                                                 |
| VRAI                                                                                                            | GEQU \$8000                                                                                                              |
|                                                                                                                 | END                                                                                                                      |
| PRET                                                                                                            | START                                                                                                                    |
|                                                                                                                 | etc..                                                                                                                    |

LOGICIELS DE DEVELOPPEMENT

## CPW

- Adresses absolues ou adresses en page zéro :

GEQU

Le label sera le nom attribué à cette adresse et l'opérande l'adresse correspondante:

|               |      |        |
|---------------|------|--------|
| VideoRegister | GEQU | \$C029 |
| H1            | GEQU | \$10   |
| H2            | GEQU | H1+4   |

- Déclaration de types de données et réservation de place en mémoire :

DC

Déclare une constante en lui affectant un type et une valeur initiale, laquelle doit être précédée et suivie d'apostrophes. Son nom étant le label de la directive DC. Les types de constantes sont :

I entier sur 2 octets.

Ix entier sur x octets.

A adresse sur 2 octets (idem à I2).

R référence d'une adresse (aucune place).

S référence logicielle d'une adresse (2 octets).

H constante hexadécimale.

B constante binaire.

C chaîne de caractères.

F nombre flottant (4 octets).

D nombre flottant en double précision (8 octets).

E nombre flottant en précision étendue utilisant 80 bits de longueur (cf SANE).

La même directive DC peut servir à la déclaration de plusieurs constantes, soit de même type et de même valeur initiale (faire précéder le type d'un facteur de répétition), soit de valeurs différentes (les séparer par des virgules).

RECT DC I4'RECTANGLE'

RECTANGLE DC I'40,20,100,180'

ATTENTION DC 20C'!',H'0D'

Les constantes de type I2 ou A sont implémentées avec l'octet le moins significatif en premier.

DS taille

Réserve de la place en mémoire dont la longueur est celle indiquée dans l'opérande en nombre d'octets. L'assembleur leur attribue la valeur \$00

ZONE DS 256

IEEE on | off

Définit quel format les nombres déclarés par les directives de F (flottant) et de D (double précision) utilisent. L'option OFF correspond au format Applesoft. (OFF par défaut).

### Exploitation de fichiers

KEEP= fichier

Sauvegarde le programme-objet décomposé en 2 modules-objets : fichier. ROOT et fichier.A, sur la



|                |                                                                                                                                                                                      |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                | disquette au fur et à mesure de l'assemblage.<br>ATTENTION ce nom de fichier ne doit pas dépasser 10 caractères.                                                                     |
| COPY fichier   | Insère le segment venant de la disquette dont le nom est spécifié, à la place de la directive COPY dans le programme-source pour y être assemblé à cet endroit du programme.         |
| APPEND fichier | Ajoute au programme-source courant la suite du programme encore sur disquette (cas où l'éditeur n'aurait pas assez de place pour tout le programme).                                 |
| MCOPY fichier  | Indique sur quel fichier sont sauvegardés les macros-instructions utiles au programme. Ce fichier sur mesure de macros est inclus dans la liste des bibliothèques de macros actives. |
| MDROP fichier  | Enlève le fichier de macros spécifié par son nom d'accès, de la liste des bibliothèques, liste qui ne peut contenir que 4 éléments.                                                  |
| MLOAD fichier  | Vérification que le fichier de macros indiqué, est bien dans la liste, sinon il est chargé comme le ferait MCOPY.                                                                    |

## LISTING 6 - Exercice de présentation

```

;EXERCICE DE PRESENTATION
 TITLE BONJOUR
 SYMBOL ON
 KEEP /CC/PROG.OBJ
 ABSADDR ON
 INSTIME ON
PROG START
 BRA PRET
VIDEO GEQU $E0C029
SUPER GEQU $A1
MASQUE GEQU $41
 END
PRET START
 CLC
 XCE
 SEP E%00110000
 LONGA OFF
 LDA ESUPER
 ORA VIDEO
 STA VIDEO
 REP E%00110000
 BRK 0
 END

```

LOGICIELS DE DEVELOPPEMENT

1st/2nd

3rd/4th

5th/6th

7th/8th

9th/10th

# CPW

ORCA/M ASM65816 V4.1 Phase 3 D6

30 Jul 86 23:03

```
0001 0000 ;EXERCICE DE PRESENTATION
0002 0000 TITLE BONJOUR
0003 0000 SYMBOL ON
0004 0000 KEEP /CC/PROG.OBJ
0005 002000 0000 ABSADDR ON
0006 002000 0000 INSTIME ON
0007 002000 0000 PROG START
0008 002000 0000 80 FE 3 BRA PRET
0009 002002 0002 VIDEO GEQU $E0C029
0010 002002 0002 SUPER GEQU $A1
0011 002002 0002 MASQUE GEQU $41
0012 002002 0002 END
```

Page 2 BONJOUR

```
0013 002002 0000 PRET START
0014 002002 0000 18 2 CLC
0015 002003 0001 FB 2 XCE
0016 002004 0002 E2 30 3 SEP £%00110000
0017 002006 0004 LONGA OFF
0018 002006 0004 A9 A1 2 LDA £SUPER
0019 002008 0006 0F 29 C0 E0 5 ORA VIDEO
0020 00200C 000A 8F 29 C0 E0 5 STA VIDEO
0021 002010 000E C2 30 3 REP £%00110000
0022 002012 0010 00 00 7 BRK 0
0023 002014 0012 END
```

Page 3 BONJOUR

Global Symbols

```
000041 MASQUE 0000A1 SUPER E0C029 VIDEO
```

23 source lines  
0 macros expanded  
0 lines generated

Linker 4.1 Phase 3 B4

```
00002000 00000002 Code: PROG
00002002 00000012 Code: PRET
```



Global symbol table:

```
00000041 00 MASQUE 00002002 00 PRET 00002000 00 PROG
000000A1 00 SUPER 00E0C029 00 VIDEO
```

Program starts at \$00002000 and is \$00000014 bytes long.

\$BLOAD PROG.OBJ

\$CALL-151

```
*
A=0000 X=0000 Y=0000 S=01DE D=0000 P=00 B=00 K=00 M=0C Q=80 L=1 m=1 x=1 e=1
*0=e
```

\*00/2000L

1=m 1=x 1=LCbank (0/1)

```
00/2000: 80 00 BRA 2002 é+00e
00/2002: 18 CLC
00/2003: FB XCE
00/2004: E2 30 SEP £30
00/2006: A9 A1 LDA £A1
00/2008: 0F 29 C0 E0 ORA E0C029
00/200C: 8F 29 C0 E0 STA E0C029
00/2010: C2 30 REP £30
00/2012: 00 00 BRK 00
00/2014: 00 00 BRK 00
00/2016: 00 00 BRK 00
00/2018: 00 00 BRK 00
00/201A: 00 00 BRK 00
00/201C: 00 00 BRK 00
00/201E: 00 00 BRK 00
00/2020: 00 00 BRK 00
00/2022: 00 00 BRK 00
00/2024: 00 00 BRK 00
00/2026: 00 00 BRK 00
00/2028: 00 00 BRK 00
```

# CPW

## Macros sous CPW

### Utilisation des macros

Les fichiers de macros inclus dans la disquette système du CPW donnent des moyens de programmation aisée : il n'est plus nécessaire de programmer des routines d'usage courant : elles existent déjà sous forme de macros. Les paramètres de ces macros ont la même syntaxe que celle des opérandes en assembleur :

- une constante est écrite précédée de £ ( ou bien #), comme l'adressage immédiat ;
- une adresse indexée s'écrit ADR,X ;
- une adresse indirecte s'écrit [ADR],X où ADR est dans la page zéro ;
- en l'absence de paramètre, l'accumulateur est utilisé si nécessaire ;

Par exemple, le fichier UTILITY.MACROS qui contient les routines suivantes :

#### - Manipulation de la pile

|              |                                                     |
|--------------|-----------------------------------------------------|
| pullword adr | dépile 2 octets et les range à l'adresse spécifiée. |
| pullong adr  | dépile 4 octets et les range à l'adresse spécifiée. |
| pull3 adr    | dépile 3 octets et les range à l'adresse spécifiée. |
| pullxy adr   | dépile 4 octets en les faisant passer par X et Y.   |
| pullay adr   | dépile 4 octets en les faisant passer par A et Y.   |
| pullx adr    | dépile 2 octets en les faisant passer par X.        |
| pushword adr | empile 2 octets venant de l'adresse indiquée.       |
| pushlong adr | empile 4 octets venant de l'adresse indiquée.       |
| push1 adr    | empile 1 octet venant de l'adresse indiquée.        |
| push3 adr    | empile 3 octets venant de l'adresse indiquée.       |
| push4 adr    | empile 4 octets venant de l'adresse indiquée.       |
| pushxy       | empile 4 octets venant de X et Y.                   |
| pushay       | empile 4 octets venant de A et Y.                   |

#### - Transferts entre mémoire et les registres A et Y

|          |                                                         |
|----------|---------------------------------------------------------|
| lday adr | met 4 octets dans A et Y venant de l'adresse spécifiée. |
| stay adr | range, dans l'adresse indiquée, les 4 octets de A et Y. |

#### - Opérations arithmétiques

|                     |                                                                                                                                                                             |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| add adr1, adr2,adr3 | adr3<-(adr1)+(adr2):mettre à l'adresse adr3 le résultat de l'addition sur 2 octets des contenus de adr1 et adr2.<br>Lorsqu'une adresse n'est pas spécifiée, il s'agit de A. |
| add4 adr1,adr2,adr3 | addition sur 4 octets : adr3<-(adr1)+(adr2).                                                                                                                                |
| sub adr1, adr2,adr3 | soustraction sur 2 octets.                                                                                                                                                  |
| sub4 adr1,adr2,adr3 | soustraction sur 4 octets.                                                                                                                                                  |





## CPW

Le répertoire /CPW/SYSTEM.MACROS contient les appels aux fonctions des outils.

Le fichier MAKE.ALL, qui fait partie de ce répertoire, permet de rassembler dans un seul fichier toutes les macros du SYSTEM.MACROS, fichier appelé ALL.MACROS et occupant 95 blocs. Pour le consulter, taper TYPE ALL.MACROS.

Le fichier HANDY.STUFF y est inclus : il contient les macros utilitaires similaires à celles du fichier UTILITY.MACROS ; ainsi que la macro permettant l'affichage de messages d'erreur éventuelle après l'appel d'un outil :

ErrorDeath 'texte'      teste le bit de retenue; si c=0 alors saut à l'instruction qui suit la macro ; si c=1 alors affichage du 'texte' et BRK.  
La chaîne de caractères du texte est enregistrée dans le corps de la macro avec la directive de c'texte'.

L'utilisation des macros se fait ainsi :

- inclure dans le programme-source la directive MCOPY fichier.MAC (par exemple) ;
- utiliser les noms des macros précédés ou non du caractère \_ pour que l'assembleur les insère à la place de leur nom ;
- une fois le programme-source enregistré sous le nom de fichier Prog.SRC, créer le fichier sur mesure des macros de ce programme, appelé fich.MAC en utilisant la commande MACGEN de CPW : £MACGEN Prog.SRC fich.MAC SYSTEM.MACROS/ALL.MACROS ;
- s'affichera la liste des macros à inclure dans fich.MAC.

### Création de macros

|                          |                                                                                                                                                                                                             |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MACRO                    | directive définissant le début de la définition d'une macro.                                                                                                                                                |
| MEND                     | directive définissant la fin d'une macro.                                                                                                                                                                   |
| MNOTE 'texte'            | le texte est affiché à l'assemblage (commentaire).                                                                                                                                                          |
| MEXIT                    | marque la fin de la définition d'une macro en cas d'assemblage conditionnel.                                                                                                                                |
| LCLA &nbre               | déclaration d'une variable arithmétique locale.                                                                                                                                                             |
| LCLB &booléen            | déclaration d'une variable booléenne locale.                                                                                                                                                                |
| LCLC &chaîne             | déclaration d'une variable locale chaîne de caractères.                                                                                                                                                     |
| GBLA &nbre               |                                                                                                                                                                                                             |
| GBLB &booléen            | idem, mais globales.                                                                                                                                                                                        |
| GBLC 1&chaîne<br>&syscnt | est la variable permanente qui vaut 1 au début et est incrémentée à chaque nouvelle définition de macro. Elle doit être concaténée aux étiquettes dans les macros pour éviter une duplication d'étiquettes. |





# CPW

Voici la macro sous sa forme expansée dans le listing-source avec GEN ON :

```
Salut str 'bonjour'
+Salut dc il'L:sys2'
+sys2 dc c 'Bonjour'
```

Les lignes commençant par + sont les instructions générées.

La variable sys2 est locale dans ce segment.

Voici le code généré :

```
07
42 6F 6E 6A
6F 75 72
```

## LISTING 7 - Catalogue des macros-système

/CPW/= .MACROS

| Name           | Type | Blocks | Modified        | Created         | Access | Subtype |
|----------------|------|--------|-----------------|-----------------|--------|---------|
| CPW.MACROS     | DIR  | 1      | 10 MAY 86       | 10 MAY 86       | DNBWR  |         |
| SYSTEM.MACROS  | DIR  | 2      | 17 JUN 86 10:57 | 17 JUN 86 10:52 | DNBWR  |         |
| SANE.MACROS    | DIR  | 1      | 21 JUL 86 1:02  | 17 JUN 86 10:52 | DNBWR  |         |
| UTILITY.MACROS | DIR  | 1      | 17 JUN 86 11:02 | 17 JUN 86 10:52 | DNBWR  |         |

Blocks Free: 67      Blocks Used: 1533      Total Blocks: 1600

/CPW/CPW.MACROS/=

| Name            | Type | Blocks | Modified      | Created   | Access | Subtype  |
|-----------------|------|--------|---------------|-----------|--------|----------|
| M65816.I.O      | SRC  | 34     | 5 NOV 85 5:42 | 10 MAY 86 | DNBWR  | ASM65816 |
| M65816.INT2MATH | SRC  | 21     | 9 NOV 85 6:16 | 10 MAY 86 | DNBWR  | ASM65816 |
| M65816.LONGMATH | SRC  | 16     | 9 NOV 85 5:41 | 10 MAY 86 | DNBWR  | ASM65816 |
| M65816.MSC      | SRC  | 32     | 11 JAN 86     | 10 MAY 86 | DNBWR  | ASM65816 |

Blocks Free: 67      Blocks Used: 1533      Total Blocks: 1600

/CPW/SYSTEM.MACROS/=

| Name        | Type | Blocks | Modified        | Created         | Access | Subtype |
|-------------|------|--------|-----------------|-----------------|--------|---------|
| HANDY.STUFF | TXT  | 17     | 15 JUN 86 13:45 | 17 JUN 86 10:53 | DNBWR  |         |
| TL.MACROS   | TXT  | 3      | 3 JUN 86 10:34  | 17 JUN 86 10:53 | DNBWR  |         |
| MM.MACROS   | TXT  | 6      | 29 MAY 86 15:48 | 17 JUN 86 10:53 | DNBWR  |         |
| MT.MACROS   | TXT  | 7      | 29 MAY 86 15:48 | 17 JUN 86 10:53 | DNBWR  |         |



|                 |     |    |    |     |    |       |    |     |    |       |       |      |
|-----------------|-----|----|----|-----|----|-------|----|-----|----|-------|-------|------|
| GD.MACROS       | TXT | 28 | 29 | MAY | 86 | 15:48 | 17 | JUN | 86 | 10:54 | DNEWR |      |
| EM.MACROS       | TXT | 4  | 29 | MAY | 86 | 15:48 | 17 | JUN | 86 | 10:54 | DNEWR |      |
| TEXT.MACROS     | TXT | 5  | 29 | MAY | 86 | 15:48 | 17 | JUN | 86 | 10:54 | DNEWR |      |
| INT.MACROS      | TXT | 4  | 29 | MAY | 86 | 15:48 | 17 | JUN | 86 | 10:54 | DNEWR |      |
| MAKE.ALL        | SRC | 1  | 17 | JUN | 86 | 16:59 | 17 | JUN | 86 | 10:54 | DNEWR | EXEC |
| PRODOS8.MACROS  | TXT | 12 | 29 | MAY | 86 | 15:48 | 17 | JUN | 86 | 10:54 | DNEWR |      |
| DESK.MACROS     | TXT | 3  | 12 | JUN | 86 | 14:54 | 17 | JUN | 86 | 10:55 | DNEWR |      |
| ALL.MACROS      | TXT | 95 | 18 | JUN | 86 | 9:02  | 17 | JUN | 86 | 10:55 | DNEWR |      |
| RELEASE.NOTES   | TXT | 4  | 18 | JUN | 86 | 9:16  | 17 | JUN | 86 | 10:55 | DNEWR |      |
| FM.MACROS       | TXT | 1  | 29 | MAY | 86 | 15:49 | 17 | JUN | 86 | 10:56 | DNEWR |      |
| PRODOS16.MACROS | TXT | 6  | 29 | MAY | 86 | 15:49 | 17 | JUN | 86 | 10:56 | DNEWR |      |
| LOADER.MACROS   | TXT | 3  | 6  | JUN | 86 | 10:12 | 17 | JUN | 86 | 10:56 | DNEWR |      |
| SOUND.MACROS    | TXT | 4  | 6  | JUN | 86 | 8:29  | 17 | JUN | 86 | 10:56 | DNEWR |      |
| WIND.MACROS     | TXT | 9  | 16 | JUN | 86 | 19:53 | 17 | JUN | 86 | 10:57 | DNEWR |      |
| MENU.MACROS     | SRC | 8  | 16 | JUN | 86 | 19:15 | 17 | JUN | 86 | 10:57 | DNEWR |      |
| CTRL.MACROS     | SRC | 5  | 17 | JUN | 86 | 10:00 | 17 | JUN | 86 | 10:57 | DNEWR |      |
| LE.MACROS       | TXT | 5  | 17 | JUN | 86 | 17:07 | 17 | JUN | 86 | 10:57 | DNEWR |      |

Blocks Free: 67      Blocks Used: 1533      Total Blocks: 1600

/CPW/SANE.MACROS/=

| Name        | Type | Blocks | Modified        | Created        | Access | Subtype |
|-------------|------|--------|-----------------|----------------|--------|---------|
| SANE.EGUS   | TXT  | 15     | 20 JUL 86 13:03 | 21 JUL 86 1:01 | DNEWR  |         |
| SANE.MACROS | TXT  | 39     | 20 JUL 86 13:03 | 21 JUL 86 1:00 | DNEWR  |         |

Blocks Free: 67      Blocks Used: 1533      Total Blocks: 1600

/CPW/UTILITY.MACROS/=

| Name      | Type | Blocks | Modified      | Created         | Access | Subtype  |
|-----------|------|--------|---------------|-----------------|--------|----------|
| UTILITY.M | SRC  | 52     | 3 JUN 86 9:30 | 17 JUN 86 11:02 | DNEWR  | ASM65816 |

Blocks Free: 67      Blocks Used: 1533      Total Blocks: 1600

## Langage LINKED et segmentation sous CPW

Le LINKED est donc un langage au même titre que l'ASM65816, il est sur la disquette CPW dans le sous-répertoire LANGUAGES/.

Un fichier écrit dans ce langage est un fichier TXT qui sera exécutable par la commande ALINK, ou bien la commande ASSEMBLE, ou bien COMPILE.

Chaque commande de ce fichier en LINKED utilise une ligne séparée.

Une ligne commençant par \* ou par ; est considérée comme un commentaire, de même qu'une ligne d'espaces.

LOGICIELS DEVE

## CPW

Commandes du LINKED (les options sont indiquées entre crochets [ ])

|                                               |                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| APPEND fichier                                | ajoute le fichier désigné au fichier-source.                                                                                                                                                                                                                                                                                                                                                                  |
| COPY fichier                                  | Copie un fichier-source et exécute ses instructions puis retourne à l'instruction du fichier-source implantée après COPY.                                                                                                                                                                                                                                                                                     |
| EJECT                                         | saut de page si l'imprimante est utilisée.                                                                                                                                                                                                                                                                                                                                                                    |
| KEEP= fichier                                 | ouvre le fichier désigné comme fichier de sortie pour y sauvegarder tous les segments successivement générés par l'éditeur de liens.<br>Les segments ainsi générés sont relogeables dans toute zone de mémoire libre et peuvent être chargés seulement au moment où ils sont nécessaires (segments dynamiques).<br>Le fichier de sortie est un module chargeable et relogeable prêt à être chargé et exécuté. |
| LIBRARY [ /REPEAT ] fichier de bibliothèque   | le fichier indiqué est une bibliothèque de routines qui devra être parcourue pour y trouver la référence non résolue par l'assembleur (par exemple le nom d'une routine-système) d'un nom de segment. L'option/REPEAT permet de balayer la bibliothèque plusieurs fois. Si * remplace le fichier, la bibliothèque-système est parcourue.                                                                      |
| LINK [ /ALL ] nom de fichier                  | le fichier désigné est un module-objet à inclure dans l'édition finale. Avec l'option /ALL (écrite concaténée à LINK sans espace), tous les modules commençant par "nom" sont pris en charge (nom.ROOT, nom.A, nom.B, etc).                                                                                                                                                                                   |
| LIST ON / OFF                                 | affiche ou non la liste de tous les sous-programmes par leur nom.                                                                                                                                                                                                                                                                                                                                             |
| OBJ valeur                                    | donne comme adresse de départ du compteur d'instructions du programme, la valeur indiquée, qui n'est pas l'adresse physique d'implantation.                                                                                                                                                                                                                                                                   |
| OBJEND                                        | donne comme adresse de départ du compteur d'instructions du programme, l'adresse physique actuelle. Ces 2 valeurs coïncident tant qu'une instruction OBJ n'a pas été introduite.                                                                                                                                                                                                                              |
| ORG valeur                                    | donne comme adresse de départ du compteur d'instructions du programme, la nouvelle valeur indiquée. Si cette instruction est en tête du premier segment de programme, cette valeur servira d'adresse fixe à partir de laquelle se fera l'édition du module et son exécution.                                                                                                                                  |
| PRINTER ON /OFF                               | l'imprimante est sélectionnée ou non pour faire le listing des segments et de la table des symboles produite par l'éditeur de liens.                                                                                                                                                                                                                                                                          |
| SELECT [ /SCAN] nom de fichier (nom1 [,nom2]) | seuls les segments référencés sont à inclure dans le                                                                                                                                                                                                                                                                                                                                                          |



fichier désigné, et ils le seront dans l'ordre indiqué.

L'option /SCAN ne permet de les référencer que par le début de leur nom (seuls les caractères précédant le point).

SEGMENT [/STATIC ] nom de segment

création dans le module courant d'un nouveau segment chargeable. Tous les segments, excepté le 1er, sont considérés comme dynamiques par l'éditeur de lien, sauf si l'option /STATIC a été imposée par cette instruction à ce nouveau segment. À moins que l'instruction ORG n'ait été donnée, tout segment statique ou dynamique est relogeable.

SOURCE ON/OFF

affiche ou non toutes les lignes d'instructions du programme-source.

SYMBOL ON/OFF

affiche ou non la table des symboles et la table d'implantation établie par l'éditeur de liens : cette table donne pour chaque segment son adresse de départ et sa longueur.

LOGICIELS DE DEVELOPPEMENT

1. /t /root

pub/xip

M

I

oub

P

E

## OUTILS

### Ensembles ( TOOL SET ) en mémoire morte

| Numéro | Nom de l'ensemble   |                           |
|--------|---------------------|---------------------------|
| \$01   | Tool locator        | Positionneur d'outils     |
| \$02   | Mémory Manager      | Gestionnaire de mémoire   |
| \$03   | Miscellaneous Tools | Outils drivers            |
| \$04   | Quicdraw            | Quicdraw (graphiques)     |
| \$05   | Desk Manager        | Gestionnaire du bureau    |
| \$06   | Event Manager       | Gestionnaire d'événements |
| \$07   | Scheduler           | Séquenceur                |
| \$08   | Sound Manager       | Gestionnaire du son       |
| \$09   | ADB Tools           | Outils de gestion du ADB  |
| \$0A   | SANE                | Arithmétique étendue      |
| \$0BC  | Text Tools          |                           |
| \$0B   | Integer Maths.      |                           |

### Structure des ensembles d'outils

Grace à l'outil de positionnement (TOOL LOCATOR), les outils n'ont plus besoin d'être à des positions fixes de MEM ou de MEY. Toutes les fonctions qu'ils contiennent sont définies chacune par un n° d'ensemble et un n° de fonction.

L'outil de positionnement utilise le numéro de l'ensemble comme clé d'entrée dans la table des pointeurs des outils TPT.

Chaque élément ensemble d'outils a une FPT contenant les pointeurs vers les fonctions individuelles de chaque ensemble. Chaque outil en MEM a sa table de pointeurs des fonctions de cet outil, une FPT en MEM. Il y a aussi une TPT en MEM pointant sur toutes les FPT. Une adresse fixe en MEV est utilisée pour pointer sur cette TPT, elle est initialisée au démarrage.

#### Structure de la TPT.

|          |          |                          |
|----------|----------|--------------------------|
| Nombre   | 4 octets | combien d'ensembles -1   |
| Pointeur | 4 octets | vers le premier ensemble |
| Pointeur | 4 octets | vers le 2nd ensemble     |
| etc.     |          |                          |

#### Structure d'une FPT

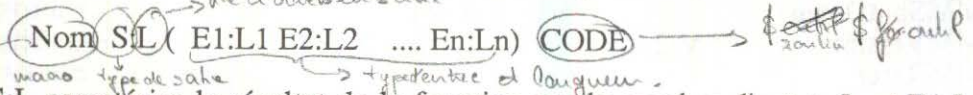
|                      |          |                         |
|----------------------|----------|-------------------------|
| Nombre               | 4 octets | combien de fonctions -1 |
| (Pointeur vers F1)-1 |          | vers la 1ère fonction   |
| (Pointeur vers F2)-1 |          | vers la 2nd fonction    |
| etc.                 |          |                         |



Chaque outil dispose d'une zone de travail dont l'adresse est donnée par la table des pointeurs de zones de travail WAPT.  
 Voir listing 3 de la page précédente.

Liste des fonctions outil par outil

Chaque fonction est donnée par le nom de la macro correspondante, puis la variable de sortie, puis la liste des variables d'entrée ; le nombre d'octets de chaque variable est donné après le caractère deux-points :



où S:L caractérise le résultat de la fonction par le nombre d'octets L et E1:L1 caractérise le premier paramètre d'entrée et sa longueur.

Le CODE est la juxtaposition du numéro de la fonction et du numéro de l'outil en hexadécimal.

Une fonction sans variables d'entrée et de sortie n'est donnée que par son nom.

Par exemple :

**TLVersion V:2 0401**

renvoie une valeur V sur 2 octets (la version actuelle du Tool Locator) sans qu'il y ait besoin de variable d'entrée.

Cette manière d'énumérer les variables est en correspondance directe avec la manière dont les paramètres doivent être empilés avant l'appel de la fonction.

Voici ce qu'il faut programmer pour utiliser la fonction TLVersion :

- PEA 0000 réserve seulement 2 octets sur le haut de la pile pour le résultat V:2 ;
- TLVersion appelle la fonction (le code d'instructions est celui de la Macro) ;
- PLA désempile les 2 octets vers l'accumulateur qui contiendra le résultat.

Prenons l'exemple plus complexe de la fonction NewHandle :

**NewHandle H:4 (LG:4 ID:2 AT:2 AD:4) 0902**

Un pointeur H d'adresse de bloc est obtenu en donnant en entrée la longueur LG du bloc, le numéro ID d'identification de l'application qui en a besoin, les attributs de ce bloc codés dans les 2 octets AT et l'adresse de début d'implantation de ce bloc. Cette fonction fait partie de l'outil n° 02, c'est-à-dire le MM ou Memory Manager.

Fonctions de l'outil N° 01 : TOOL LOCATOR (où sont les outils?)

|            |      |
|------------|------|
| TLBootInit | 0101 |
| TLStartup  | 0201 |
| TLShutDown | 0301 |

LOGICIELS DE DEVELOPPEMENT

# OUTILS

*en sahe si faut*      *en cube*

|            |       |                 |      |
|------------|-------|-----------------|------|
| TLVersion  | V:2   |                 | 0401 |
| TLReset    |       |                 | 0501 |
| GetTSPtr   | FPT:4 | (U:2 N:2)       | 0901 |
| SetTSPtr   |       | (U:2 N:2 FPT:4) | 0A01 |
| GetFuncPtr | P:4   | (U:2 F:1 N:1)   | 0B01 |
| GetWAP     | Z:4   | (U:2 N:2)       | 0C01 |
| SetWAP     |       | (U:2 N:2 Z:4)   | 0D01 |
| LoadTools  |       | (P:4)           | 0E01 |

## Fonctions de l'outil N° 02 : MEMORY MANAGER (où sont les blocs réservés?)

|               |      |                           |      |
|---------------|------|---------------------------|------|
| MMBootInit    |      |                           | 0102 |
| MMStartup     | ID:2 |                           | 0202 |
| MMShutDown    |      |                           | 0302 |
| MMVersion     | V:2  |                           | 0402 |
| MMReset       |      |                           | 0502 |
| MMStatus      | S:2  |                           | 0602 |
| NewHandle     | H:4  | (LG:4 ID:2 AT:2 AD:4)     | 0902 |
| ReallocHandle |      | (H:4 LG:4 ID:2 AT:2 AD:4) | 0A02 |
| RestoreHandle |      |                           | 0B02 |
| DisposeHandle |      | (H:4)                     | 1002 |
| DisposeAll    |      | (ID:2)                    | 1102 |
| PurgeHandle   |      | (H:4)                     | 1202 |
| PurgeAll      |      | (ID:2)                    | 1302 |
| GetHandleSize | LG:4 | (H:4)                     | 1802 |
| SetHandleSize |      | (LG:4 H:4)                | 1902 |
| FindHandle    | H:4  | (A:4)                     | 1A02 |
| FreeMem       | LG:4 |                           | 1B02 |
| MaxBlock      | LG:4 |                           | 1C02 |
| TotalMem      | LG:4 |                           | 1D02 |
| VerifyHandle  |      |                           | 1E02 |
| CompactMem    |      |                           | 1F02 |
| Hlock         |      | (H:4)                     | 2002 |
| HlockAll      |      | (ID:2)                    | 2102 |
| HUnlock       |      | (H:4)                     | 2202 |
| HUnlockAll    |      | (ID:2)                    | 2302 |
| SetPurge      |      | (N:2 H:4)                 | 2402 |
| SetPurgeAll   |      | (N:2 ID:2)                | 2502 |
| PtrToHand     |      |                           | 2802 |
| HandToPtr     |      |                           | 2902 |
| HandToHand    |      |                           | 2A02 |
| BlockMove     |      | (PS:4 PD:4 LG:4)          | 2B02 |

*disque un bloc  
in table, in a attributs  
mais localise  
vance*

*→ Purge bloc & block*

*→ Purge de bloc  
en conservant le  
bande*

*AD =  
adresse  
pour les  
attributs partiales*

*→ adresse  
pointe / plein main  
retourne le bande  
pour tout ce plein main*

*H = 4 octets*

*designant  
bande*

*→ un de purge (à impédit)*

*(PS H LG) → source/ptr → dest/bande + LG*

*H PD LG*

*H H LG*

*= copie sauve*

## Fonctions de l'outil N° 03 : MISCELLANEOUS TOOL (outils divers)

|            |     |       |      |
|------------|-----|-------|------|
| MTBootInit |     |       | 0103 |
| MTStartup  |     |       | 0203 |
| MTShutDown |     |       | 0303 |
| MTVersion  | V:2 |       | 0403 |
| MTRreset   |     |       | 0503 |
| MTStatus   | B:2 |       | 0603 |
| WriteBRam  |     | (P:4) | 0903 |



|               |                     |                                      |      |
|---------------|---------------------|--------------------------------------|------|
| ReadBRam      |                     | (P:4)                                | 0A03 |
| WriteBParam   |                     | (D:2 N:2)                            | 0B03 |
| RdBParam      | D:2                 | (N:2)                                | 0C03 |
| ReadTimeHex   | JS:2 M:1 J:1        | AN:1 HH:1 MN:1 SC:1                  | 0D03 |
| WriteTimeHex  |                     | (M:1 J:1 AN:1 HH:1 MN:1 SC:1)        | 0E03 |
| ReadAsciiTime |                     | (P:4)                                | 0F03 |
| SetVector     |                     | (N:2 P:4)                            | 1003 |
| GetVector     | P:4                 | (N:2)                                | 1103 |
| SetHeartBeat  |                     | (P:4)                                | 1203 |
| DelHeartBeat  |                     | (P:4)                                | 1303 |
| ClrHeartBeat  |                     |                                      | 1403 |
| SetDeathMgr   |                     | (E:2 P:4)                            | 1503 |
| GetAddr       | P:4                 | (N:2)                                | 1603 |
| ReadMouse     | X:2 Y:2 S:1 M:1     |                                      | 1703 |
| InitMouse     |                     | (C:2)                                | 1803 |
| SetMouse      |                     | (M:2)                                | 1903 |
| HomeMouse     |                     |                                      | 1A03 |
| ClearMouse    |                     |                                      | 1B03 |
| ClampMouse    |                     | (Xm:2 XM:2 Ym:2 YM:2)                | 1C03 |
| GetMouseClamp | Xm:2 XM:2 Ym:2 YM:2 |                                      | 1D03 |
| PosMouse      |                     | (X:2 Y:2)                            | 1E03 |
| ServeMouse    | I:2                 |                                      | 1F03 |
| GetNewID      | ID:2                | (T:2)                                | 2003 |
| DeleteID      |                     | (ID:2)                               | 2103 |
| StatusID      |                     | (ID:2)                               | 2203 |
| Intsource     |                     | (N:2)                                | 2304 |
| FWentry       | P:2 Acc:2 X:2 Y:2   | (Acc:2 X:2 Y:2 AD:2)                 | 2403 |
| GetTick       | CT:4                |                                      | 2503 |
| PackBytes     | CT:2                | (PS:4 LS:4 PD:4 LD:2)                | 2603 |
| UnPackBytes   | CT:2                | (PS:4 LS:2 PD:4)                     | 2703 |
| Munger        | S:2                 | (PR:4 LR:4 PT:4 LT:2 PP:4 LP:2 PA:4) | 2803 |
| GetIRQenbl    | I:2                 | (:0)                                 | 2903 |
| SetAbsClamp   |                     | (Xm:2 XM:2 Ym:2 YM:2)                | 2A03 |
| GetAbsClamp   | Xm:2 XM:2 Ym:2 YM:2 |                                      | 2B03 |

Fonctions de l'outil N° 04 : QUICK DRAW II (outil graphique)

|                  |       |                                      |      |
|------------------|-------|--------------------------------------|------|
| QDBootInit       |       |                                      | 0104 |
| × QDStartup      |       | (Z:2 SCB:2 LG:2 ID:2)                | 0204 |
| × QDShutDown     |       | <i>adresse de la page de l'outil</i> | 0304 |
| × QDVersion      | V:2   |                                      | 0404 |
| × QDReset        |       |                                      | 0504 |
| QDStatus         | B:2   |                                      | 0604 |
| Grafon           |       |                                      | 0A04 |
| Graffoff         |       |                                      | 0B04 |
| × GetStandardSCB | SCB:2 |                                      | 0C04 |
| × InitColorTable |       | (PA:4)                               | 0D04 |
| × SetColorTable  |       | (PA:4)                               | 0E04 |
| × GetColorTable  |       | (N:2 PA:4)                           | 0F04 |
| SetColorEntry    |       | (N:2 C:2 D:2)                        | 1004 |

*pointeur sur une palette (420=32 octets)*

*usage de la table palette = 0 → 15  
\$00 → \$0F*

LOGICIELS DEVELOPPE

## OUTILS

|                 |       |             |      |
|-----------------|-------|-------------|------|
| GetColorEntry   | D:2   | (N:2 C:2)   | 1104 |
| SetSCB          |       | (L:2 SCB:2) | 1204 |
| GetSCB          | SCB:2 | (L:2)       | 1304 |
| SetAllSCB       |       | (SCB:2)     | 1404 |
| ClearScreen     |       | (D:2)       | 1504 |
| ↗ SetMasterSCB  |       | (SCB:2)     | 1604 |
| ↘ GetMasterSCB  | SCB:2 |             | 1704 |
| OpenPort        |       | (PO:4)      | 1804 |
| InitPort        |       | (PO:4)      | 1904 |
| ClosePort       |       | (PO:4)      | 1A04 |
| SetPort         |       | (PO:4)      | 1B04 |
| GetPort         | PO:4  |             | 1C04 |
| SetPortLoc      |       | (PL:4)      | 1D04 |
| GetPortLoc      |       | (PL:4)      | 1E04 |
| SetPortRect     |       | (RCT:4)     | 1F04 |
| GetPortRect     |       | (RCT:4)     | 2004 |
| SetPortSize     |       | (l:2 h:2)   | 2104 |
| MovePortTo      |       | (H:2 V:2)   | 2204 |
| SetOrigin       |       | (H:2 V:2)   | 2304 |
| SetClip         |       | (HRG:4)     | 2404 |
| GetClip         |       | (HRG:4)     | 2504 |
| ClipRect        |       | (RCT:4)     | 2604 |
| HidePen         |       |             | 2704 |
| ShowPen         |       |             | 2804 |
| GetPen          | PP:4  |             | 2904 |
| SetPenState     |       | (PS:4)      | 2A04 |
| GetPenState     | PS:4  |             | 2B04 |
| SetPenSize      |       | (l:2 h:2)   | 2C04 |
| GetPenSize      | PP:4  |             | 2D04 |
| SetPenMode      |       | (M:2)       | 2E04 |
| GetPenMode      | M:2   |             | 2F04 |
| SetPenPat       |       | (PM:4)      | 3004 |
| GetPenPat       |       | (PM:4)      | 3104 |
| SetDrawMask     |       | (PD:4)      | 3204 |
| GetDrawMask     |       | (PD:4)      | 3304 |
| SetBackPat      |       | (PM:4)      | 3404 |
| GetBackPat      |       | (PM:4)      | 3504 |
| PenNormal       |       |             | 3604 |
| SetSolidPenPat  |       | (C:2)       | 3704 |
| SetSolidBackPat |       | (C:2)       | 3804 |
| SolidPattern    |       | (PM:4 C:2)  | 3904 |
| Moveto          |       | (H:2 V:2)   | 3A04 |
| Move            |       | (DH:2 DV:2) | 3B04 |
| LineTo          |       | (H:2 V:2)   | 3C04 |
| Line            |       | (DH:2 DV:2) | 3D04 |
| SetPicSave      |       | (val:4)     | 3E04 |
| GetPicSave      | val:4 |             | 3F04 |
| SetRgnSave      |       | (val:4)     | 4004 |
| GetRgnSave      | val:4 |             | 4104 |
| SetPolySave     |       | (val:4)     | 4204 |



|               |       |                         |      |
|---------------|-------|-------------------------|------|
| GetPolySave   | val:4 |                         | 4304 |
| SetGrafsProcs |       | (PGP:4)                 | 4404 |
| GetGrafsProcs | PGP:4 |                         | 4504 |
| SetUserField  |       | (val:4)                 | 4604 |
| GetUserField  | val:4 |                         | 4704 |
| SetSysField   |       | (val:4)                 | 4804 |
| GetSysField   | val:4 |                         | 4904 |
| SetRect       |       | (RCT:4 g:2 h:2 d:2 b:2) | 4A04 |
| OffsetRect    |       | (RCT:4 DH:2 DV:2)       | 4B04 |
| InsetRect     |       | (RCT:4 DH:2 DV:2)       | 4C04 |
| SectRect      | B:2   | (RC1:4 RC2:4 RC3:4)     | 4D04 |
| UnionRect     |       | (RC1:4 RC2:4 RC3:4)     | 4E04 |
| PtInRect      | B:2   | (PPT:4 RCT:4)           | 4F04 |
| Pt2Rect       |       | (P1:4 P2:4 RCT:4)       | 5004 |
| EqualRect     | B:2   | (RCT1:4 RCT2:4)         | 5104 |
| EmptyRect     | B:2   | (RCT:4)                 | 5204 |
| FrameRect     |       | (RCT:4)                 | 5304 |
| PaintRect     |       | (RCT:4)                 | 5404 |
| EraseRect     |       | (RCT:4)                 | 5504 |
| InvertRect    |       | (RCT:4)                 | 5604 |
| FillRect      |       | (RCT:4)                 | 5704 |
| FrameOval     |       | (OV:4)                  | 5804 |
| PaintOval     |       | (OV:4)                  | 5904 |
| EraseOval     |       | (OV:4)                  | 5A04 |
| InvertOval    |       | (OV:4)                  | 5B04 |
| FillOval      |       | (OV:4)                  | 5C04 |
| FrameRRect    |       | (RCT:4)                 | 5D04 |
| PaintRRect    |       | (RCT:4)                 | 5E04 |
| EraseRRect    |       | (RCT:4)                 | 5F04 |
| InvertRRect   |       | (RCT:4)                 | 6004 |
| FillRRect     |       | (RCT:4)                 | 6104 |
| FrameArc      |       | (ARC:4)                 | 6204 |
| PaintArc      |       | (ARC:4)                 | 6304 |
| EraseArc      |       | (ARC:4)                 | 6404 |
| InvertArc     |       | (ARC:4)                 | 6504 |
| FillArc       |       | (ARC:4)                 | 6604 |
| NewRgn        | HRG:4 |                         | 6704 |
| DisposeRgn    |       | (HRG:4)                 | 6804 |
| CopyRgn       |       | (HRGS:4 HRGD:0)         | 6904 |
| SetEmptyRgn   |       | (HRG:4)                 | 6A04 |
| SetRctRgn     |       | (HRG:4 g:2 h:2 d:2 b:2) | 6B04 |
| RectRgn       |       | (HRG:4 RCT:4)           | 6C04 |
| OpenRgn       | HRG:4 |                         | 6D04 |
| CloseRgn      |       | (HRG:4)                 | 6E04 |
| OffsetRgn     |       | (HRG:4 DH:2 DV:2)       | 6F04 |
| InsetRgn      |       | (HRG:4 DH:2 DV:2)       | 7004 |
| SectRgn       |       | (HRG1:4 HRG2:4 HRG3:4)  | 7104 |
| UnionRgn      |       | (HRG1:4 HRG2:4 HRG3:4)  | 7204 |
| DiffRgn       |       | (HRG1:4 HRG2:4 HRG3:4)  | 7304 |
| XorRgn        |       | (HRG1:4 HRG2:4 HRG3:4)  | 7404 |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                |       |                         |      |
|----------------|-------|-------------------------|------|
| PtInRgn        | B:2   | (PPT:4 HRG:4)           | 7504 |
| RectInRgn      | B:2   | (RCT:4 HRG:4)           | 7604 |
| EqualRgn       | B:2   | (HRG1:4 HRG2:4)         | 7704 |
| EmptyRgn       | B:2   | (HRG:4)                 | 7804 |
| FrameRgn       |       | (HRG:4)                 | 7904 |
| PaintRgn       |       | (HRG:4)                 | 7A04 |
| EraseRgn       |       | (HRG:4)                 | 7B04 |
| InvertRgn      |       | (HRG:4)                 | 7C04 |
| FillRgn        |       | (HRG:4)                 | 7D04 |
| ScrollRect     |       | (RCT:4 DH:2 DV:2 HRG:4) | 7E04 |
| PaintPixels    |       | (PBP:4)                 | 7F04 |
| AddPt          |       | (PPTS:4 PPTD:4)         | 8004 |
| SubPt          |       | (PPTS:4 PPTD:4)         | 8104 |
| SetPt          |       | (PPTS:4 H:2 V:4)        | 8204 |
| EqualPt        | B:2   | (PPTS:4 PPTD:4)         | 8304 |
| LocalToGlobal  |       | (PPT:4)                 | 8404 |
| GlobalToLocal  |       | (PPT:4)                 | 8504 |
| Random         | val:2 |                         | 8604 |
| SetRandSeed    |       | (val:4)                 | 8704 |
| GetPixel       | val:2 | (H:2 V:2)               | 8804 |
| ScalePt        |       | (PPT:4 RCT1:4 RCT2:4)   | 8904 |
| MapPt          |       | (PPT:4 RCT1:4 RCT2:4)   | 8A04 |
| MapRect        |       | (RCT:4 RCT1:4 RCT2:4)   | 8B04 |
| MapRgn         |       | (HRG:4 RCT1:4 RCT2:4)   | 8C04 |
| SetStdProc     |       | (PGR:4)                 | 8D04 |
| SetCursor      |       | (PC:4)                  | 8E04 |
| GetCursorAdr   | PC:4  |                         | 8F04 |
| HideCursor     |       |                         | 9004 |
| ShowCursor     |       |                         | 9104 |
| ObscureCursor  |       |                         | 9204 |
| SetMouseLoc    |       | (X:2 Y:2)               | 9304 |
| SetFont        |       | (HF:4)                  | 9404 |
| GetFont        | HF:4  |                         | 9504 |
| GetFontInfo    |       | (PF:4)                  | 9604 |
| GetFontGlobals | PFG:4 |                         | 9704 |
| SetFontFlags   |       | (val:2)                 | 9804 |
| GetFontFlags   | val:2 |                         | 9904 |
| SetTextFace    |       | (TF:2)                  | 9A04 |
| GetTextFace    | TF:2  |                         | 9B04 |
| SetTextMode    |       | (TM:2)                  | 9C04 |
| GetTextMode    | TM:2  |                         | 9D04 |
| SetSpaceExtra  |       | (val:4)                 | 9E04 |
| GetSpaceExtra  | val:4 |                         | 9F04 |
| SetForeColor   |       | (C:2)                   | A004 |
| GetForeColor   | C:2   |                         | A104 |
| SetBackColor   |       | (C:2)                   | A204 |
| GetBackColor   | C:2   |                         | A304 |
| DrawChar       |       | (Car:2)                 | A404 |
| DrawString     |       | (PC:4)                  | A504 |
| DrawCString    |       | (PC:4)                  | A604 |



# OUTILS

|                |       |                        |      |
|----------------|-------|------------------------|------|
| DrawText       |       | (PC:4 LG:2)            | A704 |
| CharWidth      | L:2   | (Car:2)                | A804 |
| StringWidth    | L:2   | (PC:4)                 | A904 |
| CStringWidth   | L:2   | (PC:4)                 | AA04 |
| TextWidth      | L:2   | (PC:4 LG:2)            | AB04 |
| CharBounds     |       | (Car:2 RCT:4)          | AC04 |
| StringBounds   |       | (PC:4 RCT:4)           | AD04 |
| CStringBounds  |       | (PC:4 RCT:4)           | AE04 |
| TextBounds     |       | (PC:4 LG:2 RCT:4)      | AF04 |
| SetArcRot      |       | (ARC:4 R:2)            | B004 |
| GetArcRot      | R:2   | (ARC:4)                | B104 |
| SetSysFont     |       | (HF:4)                 | B204 |
| GetSysFont     | HF:4  |                        | B304 |
| SetVisRgn      |       | (HR:4)                 | B404 |
| GetVisRgn      |       | (HR:4)                 | B504 |
| SetIntUse      |       | (I:2)                  | B604 |
| OpenPicture    | H:4   | (RCT:4)                | B704 |
| PicComment     |       | (T:2 LG:4 HC:4)        | B804 |
| ClosePicture   |       |                        | B904 |
| DrawPicture    |       | (H:4 RCT:4)            | BA04 |
| KillPicture    |       | (H:4)                  | BB04 |
| FramePoly      |       | (HP:4)                 | BC04 |
| PaintPoly      |       | (HP:4)                 | BD04 |
| ErasePoly      |       | (HP:4)                 | BE04 |
| InvertPoly     |       | (HP:4)                 | BF04 |
| FillPoly       |       | (HP:4 PA:4)            | C004 |
| OpenPoly       | HP:4  |                        | C104 |
| ClosePoly      |       |                        | C204 |
| KillPoly       |       | (HP:4)                 | C304 |
| OffsetPoly     |       | (HP:4 DH:2 DV:2)       | C404 |
| MapPoly        |       | (HP:4 RCT1:4 RCT2:4)   | C504 |
| SetClipHandle  |       | (HRG:4)                | C604 |
| GetClipHandle  | HRG:4 |                        | C704 |
| SetVisHandle   |       | (HRG:4)                | C804 |
| GetVisHandle   | HRG:4 |                        | C904 |
| InitCursor     |       |                        | CA04 |
| SetBufDims     |       | (LG:2 hF:2 eF:2)       | CB04 |
| ForceBufDims   |       | (LG:2 hF:2 eF:2)       | CC04 |
| SaveBufDims    |       | (PLB:4)                | CD04 |
| RestoreBufDims |       | (LG:2 hF:2 eF:2)       | CE04 |
| GetFGSize      | LG:2  |                        | CF04 |
| SetFontID      |       | (val:4)                | D004 |
| GetFontID      | val:4 |                        | D104 |
| SetTextSize    |       |                        | D204 |
| GetTextSize    |       |                        | D304 |
| SetCharExtra   |       | (val:4)                | D404 |
| GetCharExtra   | val:4 |                        | D504 |
| PPToPort       |       | (PBP:4 RECT:4 X:2 Y:2) | D604 |

LOGICIELS DEVELOPPEMENT

## OUTILS

### Fonctions de l'outil N° 05 : DESK MANAGER (quels accessoires de bureau ?)

|                  |       |             |      |
|------------------|-------|-------------|------|
| DeskBootInit     |       |             | 0105 |
| DeskStartup      |       |             | 0205 |
| DeskShutDown     |       |             | 0305 |
| DeskVersion      | V:2   |             | 0405 |
| DeskReset        |       |             | 0505 |
| DeskStatus       | B:2   |             | 0605 |
| savescrn         |       |             | 0905 |
| restscrn         |       |             | 0A05 |
| saveall          |       |             | 0B05 |
| restall          |       |             | 0C05 |
| InstallNDA       |       | (HDA:4)     | 0D05 |
| InstallCDA       |       | (HCA:4)     | 0F05 |
| choosecda        |       |             | 1105 |
| setdastring      |       | (ID:2 PC:4) | 1305 |
| getdastring      | PC:4  | (ID:2)      | 1405 |
| OpenNDA          | Ref:2 | (IDI:2)     | 1505 |
| CloseNDA         |       | (Ref:2)     | 1605 |
| SystemClick      |       |             | 1705 |
| SystemTask       |       |             | 1905 |
| GetNumNDAs       | N:2   |             | 1B05 |
| CloseNDAbyWinPtr |       | (PW:4)      | 1C05 |
| CloseAllNDAs     |       |             | 1D05 |
| FixAppleMenu     |       | (ID:2)00,0  | 1E05 |

### Fonctions de l'outil N° 06: EVENT MANAGER (que s'est-il passé?)

|              |      |                                     |      |
|--------------|------|-------------------------------------|------|
| EMBootInit   |      |                                     | 0106 |
| EMStartup    |      | (Z:2 LQ:2 Xm:2 XM:2 Ym:2 YM:2 ID:2) | 0206 |
| EMShutDown   |      |                                     | 0306 |
| EMVersion    | V:2  |                                     | 0406 |
| EMReset      |      |                                     | 0506 |
| EMActive     | B:2  |                                     | 0606 |
| DoWindows    | Z:2  |                                     | 0906 |
| GetNextEvent | B:2  | (ME:2 PEV:4)                        | 0A06 |
| EventAvail   | B:2  | (ME:2 PEV:4)                        | 0B06 |
| GetMouse     |      | (PM:4)                              | 0C06 |
| Button       | B:2  | (N:2)                               | 0D06 |
| StillDown    | B:2  | (N:2)                               | 0E06 |
| WaitMouseUp  | B:2  | (N:2)                               | 0F06 |
| TickCount    | CT:4 |                                     | 1006 |
| GetDblTime   | CT:4 |                                     | 1106 |
| GetCaretTime | CT:4 |                                     | 1206 |
| SetSwitch    |      |                                     | 1306 |
| PostEvent    | B:2  | (TE:2 AE:4)                         | 1406 |
| FlushEvent   | TE:2 | (ME:2 MS:2)                         | 1506 |
| GetOSEvent   | B:2  | (ME:2 PEV:4)                        | 1606 |
| OsEventAvail | B:2  | (ME:2 PEV:4)                        | 1706 |
| SetEventMask |      | (ME:2)                              | 1806 |



Fonctions de l'outil N° 07 : SCHEDULER (chaque chose en son temps)

|             |        |      |
|-------------|--------|------|
| SCHBootInit |        | 0107 |
| SCHStartup  |        | 0207 |
| SCHShutDown |        | 0307 |
| SCHVersion  | V:2    | 0407 |
| SCHReset    |        | 0507 |
| SCHActive   | B:2    | 0607 |
| SCHAddTask  | (PT:4) | 0907 |
| SCHFlush    |        | 0A07 |

Fonctions de l'outil N° 08 : SOUND (musique)

|                   |                    |      |
|-------------------|--------------------|------|
| SoundBootInit     |                    | 0108 |
| SoundStartup      | (Z:2)              | 0208 |
| SoundShutDown     |                    | 0308 |
| SoundVersion      | V:2                | 0408 |
| SoundReset        |                    | 0508 |
| SoundToolStatus   | B:2                | 0608 |
| WriteRamBlock     | (PS:4 DOCB:2 LG:2) | 0908 |
| ReadRamBlock      | (PD:4 DOCB:2 LG:2) | 0A08 |
| GetTableAddress   | TA:4               | 0B08 |
| GetSoundVolume    | V:2 (N:2)          | 0C08 |
| SetSoundVolume    | (V:2 N:2)          | 0D08 |
| FFStartSound      | (N:2 PPRM:4)       | 0E08 |
| FFStopSound       | (MG:2)             | 0F08 |
| FFSoundStatus     | SG:2               | 1008 |
| FFGeneratorStatus | GCB:2 (N:2)        | 1108 |
| SetSoundMIRQV     | (VCT:4)            | 1208 |
| SetUserSoundIRQV  | OVCT:4 (VCT:4)     | 1308 |
| FFSoundDoneStatus | B:2 (N:2)          | 1408 |

Fonctions de l'outil N° 09 : ADB (AppleFront Desk Bus)

|             |     |      |
|-------------|-----|------|
| ADDBootInit |     | 0109 |
| ADBStartup  |     | 0209 |
| ADBShutDown |     | 0309 |
| ADBVersion  | V:2 | 0409 |
| ADBReset    |     | 0509 |
| ADBStatus   | B:2 | 0609 |

Fonctions de l'outil N° \$0A : SANE (Calculs sur des nombres flottants)

|               |                                |      |
|---------------|--------------------------------|------|
| SANEBootInit  |                                | 010A |
| SANESStartup  |                                | 020A |
| SANESShutDown |                                | 030A |
| SANEVersion   | V:2                            | 040A |
| SANEReset     |                                | 050A |
| SANESStatus   | B:2                            | 060A |
| FOPRF         | COP:2 (arithmétique flottante) | 090A |
| FOPRD         | COP:2 (analyse et formatage)   | 0A0A |
| FOPRE         | COP:2 (fonctions élémentaires) | 0B0A |

Les MACROS SANE: les opérandes doivent être empilées avant l'appel ainsi que l'adresse de rangement du résultat.

LOGICIELS DE DEVELOPPEMENT

fab/10/10

outils

P L E

## OUTILS

Les formats d'opérandes sont les suivants :

**X** : extended            flottant précision étendue (80 bits)  
**D** : double                flottant double précision (64bits)  
**S** : single                flottant précision simplé (32bits)  
**C** : comp                 entier utilisé en comptabilité (64 bits)  
**I** : integer                entier (16 bits)  
**L** : longinteger          entier long (32 bits)  
FADDX, FADDD, FADDS, FADDC, FADDI, FADDL  
FSUBX, FSUBD, FSUBS, FSUBC, FSUBI, FSUBL  
FMULX, FMULD, FMULS, FMULC, FMULI, FMULL  
FDIVX, FDIVD, FDIVS, FDIVC, FDIVI, FDIVL  
FSQRTX  
FRINTX  
FTINTX  
FREMX, FREMD, FREMC, FREMI, FREML  
FLOGBX  
FSCALBX  
FCPYSGNX, FCPYSGND, FCPYSGNS, FCPYSGNC, FCPYSGNI,  
FCPYSGNL  
FNEGX  
FABSX  
FNEXTS, FNEXTD, FNEXTX  
FX2X, FD2X, FS2X, FI2X, FL2X, FC2X  
FX2D, FX2S, FX2I, FX2L, FX2C  
FX2DEC, FD2DEC, FS2DEC, FC2DEC, FI2DEC, FL2DEC  
FDEC2X, FDEC2D, FDEC2S, FDEC2C, FDEC2I, FDEC2L  
FCMPX, FCMPD, FCMPs, FCMPc, FCMPi, FCMPl  
FCPXX, FCPXD, FCPXS, FCPXC, FCPXI, FCPXL  
FBEQ, FBLT, FBLE, FBGT, FBGE, FBULT, FBULE, FBUGT, FBUGE,  
FBU, FBO  
FBNE, FBUE, FBLG  
FCLASS, FCLASSD, FCLASSX, FCLASSC, FCLASSI, FCLASSL  
FBSNAN, FBQANAN, FBINF, FBZERO, FBNORM, FBDENORM,  
FBNZENUM, FBNUM,  
FBMINUS, FBPLUS  
FTESTXCP, FSETXCP  
FPROCENTRY, FPROCEXIT  
FGETHV, FSETHV  
FLNX, FLOG2X, FLN1X, FLOG21X, FEXPX, FEXP2X, FEXP1X,  
FEXP21X  
FXPWRI, FXPWRY, FCOMPOUND, FANNUIITY, FATANX, FSINX,  
FCOSX  
FCOSX, FTANX, FRANDX  
FPSTR2DEC, FCSTR2DEX, FDEC2STR

---

### Fonctions de l'outil N° \$0B : INTEGER MATHS

(Calculs sur des nombres entiers)

|             |      |
|-------------|------|
| IntBootInit | 010B |
| IntStartup  | 020B |
| IntShutDown | 030B |



|                                                           |           |                      |      |
|-----------------------------------------------------------|-----------|----------------------|------|
| IntMVersion                                               | V:2       |                      | 040B |
| IntMReset                                                 |           |                      | 050B |
| IntMStatus                                                | B:2       |                      | 060B |
| Les 4 types de nombres suivants sont manipulés            |           |                      |      |
| - Entier sur 2 octets ou Int2                             |           |                      |      |
| - Entier Long sur 4 octets ou Int4                        |           |                      |      |
| - Fixe sur 4 octets, avec signe et 16 bits de fraction    |           |                      |      |
| - Fraction sur 4 octets avec signe et 30 bits de fraction |           |                      |      |
| Multiply                                                  | ML:4      | (M1:2 M2:2)          | 090B |
| SDivide                                                   | R:2 Q:2   | (M1:2 M2:2)          | 0A0B |
| UDivide                                                   | R:2 Q:2   | (M1:2 M2:2)          | 0B0B |
| LongMul                                                   | MLF:4 MLf | (ML1:4 ML2:4)        | 0C0B |
| LongDivide                                                | R:4 Q:4   | (ML1:4 ML2:4)        | 0D0B |
| FixRatio                                                  | RF:4      | (M1:2 M2:2)          | 0E0B |
| FixMul                                                    | MF:4      | (MF1:4 MF2:4)        | 0F0B |
| Int2Hex                                                   |           | (M:2 PC:4 LG:2)      | 220B |
| Long2Hex                                                  |           | (ML:4 PC:4 LG:2)     | 230B |
| Hex2Int                                                   | M:2       | (PC:4 LG:2)          | 240B |
| Hex2Long                                                  | ML:4      | (PC:4 LG:2)          | 250B |
| Int2Dec                                                   |           | (M:2 PC:4 LG:2 S:2)  | 260B |
| Long2Dec                                                  |           | (ML:4 PC:4 LG:2 S:2) | 270B |
| Dec2Int                                                   | M:2       | (PC:4 LG:2 S:2)      | 280B |
| Dec2Long                                                  | ML:4      | (PC:4 LG:2 S:2)      | 290B |
| HexIt                                                     | CH:4      | (M:2)                | 2A0B |

-----  
Fonctions de l'outil N° \$0C : TEXT TOOLS (Mode texte style AppleII)

|                 |                |                  |      |
|-----------------|----------------|------------------|------|
| TextBootInit    |                |                  | 010C |
| TextStartup     |                |                  | 020C |
| TextShutDown    |                |                  | 030C |
| TextVersion     | V:2            |                  | 040C |
| TextReset       |                |                  | 050C |
| TextStatus      | B:2            |                  | 060C |
| SetInGlobals    |                | (MAND:2 MOR:2)   | 090C |
| SetOutGlobals   |                | (MAND:2 MOR:2)   | 0A0C |
| SetErrGlobals   |                | (MAND:2 MOR:2)   | 0B0C |
| GetInGlobals    | MAND:2 MOR:2   |                  | 0C0C |
| GetOutGlobals   | MAND:2 MOR:2   |                  | 0D0C |
| GetErrGlobals   | MAND:2 MOR:2   |                  | 0E0C |
| SetInputDevice  |                | (Type:2 Pslot:4) | 0F0C |
| SetOutputDevice |                | (Type:2 Pslot:4) | 100C |
| SetErrDevice    |                | (Type:2 Pslot:4) | 110C |
| GetInputDevice  | Type:2 Pslot:4 |                  | 120C |
| GetOutputDevice | Type:2 Pslot:4 |                  | 130C |
| GetErrorDevice  | Type:2 Pslot:4 |                  | 140C |
| InitTextDev     |                | (D:2)            | 150C |
| CtrlTextDev     |                | (D:2 Cctrl:2)    | 160C |
| StatusTDev      |                | (D:2 S:2)        | 170C |
| WriteChar       |                | (car:2)          | 180C |
| ErrWriteChar    |                | (car:2)          | 190C |
| WriteLine       |                | (PC:4)           | 1A0C |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                 |       |                          |      |
|-----------------|-------|--------------------------|------|
| ErrWriteLine    |       | (PC:4)                   | 1B0C |
| WriteString     |       | (PC:4)                   | 1C0C |
| ErrWriteString  |       | (PC:4)                   | 1D0C |
| WriteBlock      |       | (PC:4 Dep:2 LG:2)        | 1E0C |
| ErrWriteBlock   |       | (PC:4 Dep:2 LG:2)        | 1F0C |
| WriteCString    |       | (PCC:4)                  | 200C |
| ErrWriteCstring |       | (PCC:4)                  | 210C |
| ReadChar        | car:2 | (echo:2)                 | 220C |
| ReadBlock       |       | (PB:4 dep:2 LG:2 echo:2) | 230C |
| ReadLine        | N:2   | (PB:4 LM:2 EOL:2 echo:2) | 240C |

---

### Fonctions de l'outil N° \$0D : Réservé au Driver RAMDisk

---

### Fonctions de l'outil N° \$0E : WINDOW MANAGER (Superposition de fenêtres)

TOOL014

|               |           |                              |      |
|---------------|-----------|------------------------------|------|
| WindBootInit  |           |                              | 010E |
| WindStartup   |           | (ID:2)                       | 020E |
| WindShutDown  |           |                              | 030E |
| WindVersion   | V:2       |                              | 040E |
| WindReset     |           |                              | 050E |
| WindStatus    | S:2       |                              | 060E |
| NewWindow     | PW:4      | (PPRM:4)                     | 090E |
| CheckUpDate   | BU:2      | (PEV:4)                      | 0A0E |
| CloseWindow   |           | (PW:4)                       | 0B0E |
| Desktop       | RP:4      | (OP:2 P:4)                   | 0C0E |
| SetWTitle     |           | (PC:4 PW:4)                  | 0D0E |
| GetWTitle     | PC:4      | (PW:4)                       | 0E0E |
| SetFrameColor |           | (PPA:4 PW:4)                 | 0F0E |
| GetFrameColor |           | (PPA:4 PW:4)                 | 100E |
| SelectWindow  |           | (PW:4)                       | 110E |
| HideWindow    |           | (PW:4)                       | 120E |
| ShowWindow    |           | (PW:4)                       | 130E |
| SendBehind    |           | (PWR:4 PW:4)                 | 140E |
| FrontWindow   | PW:4      |                              | 150E |
| SetInfoDraw   |           | (PW:4)                       | 160E |
| FindWindow    | OU:2      | (APW:4 POINT:4)              | 170E |
| TrackGoAway   | B:2       | (PD:4 PW:4)                  | 180E |
| MoveWindow    |           | (X:2 Y:2 PW:4)               | 190E |
| DragWindow    |           | (G:2 X0:2 Y0:2 g RCT:4 PW:4) | 1A0E |
| GrowWindow    | Nl:2 Nh:2 | (ml:2 mh:2 X0:2 Y0:2 PW:4)   | 1B0E |
| SizeWindow    |           | (Nl:2 Nh:2 PW:4)             | 1C0E |
| TaskMaster    | C:2       | (ME:2 EVTE:4)                | 1D0E |
| BeginUpdate   |           | (PW:4)                       | 1E0E |
| EndUpDate     |           | (PW:4)                       | 1F0E |
| GetWmgrPort   | PW:4      |                              | 200E |
| PinRect       | POINT:4   | (RCT:4 X:2 Y:2)              | 210E |
| HiliteWindow  |           | (FH:2 PW:4)                  | 220E |
| ShowHide      |           | (F:2 PW:4)                   | 230E |
| BringToFront  |           | (PW:4)                       | 240E |
| WNewRes       |           |                              | 250E |



# OUTILS

|               |           |                  |      |
|---------------|-----------|------------------|------|
| TrackZoom     | B:2       | (PD:4 PW:4)      | 260E |
| ZoomWindow    |           | (PW:4)           | 270E |
| SetWRefCon    |           | (V:4 PW:4)       | 280E |
| GetWRefCon    | V:4       | (PW:4)           | 290E |
| GetNextWindow | PW:4      | (PW:4)           | 2A0E |
| GetWKind      | B:2       | (PW:4)           | 2B0E |
| GetWFrame     | F:2       | (PW:4)           | 2C0E |
| SetWFrame     | F:2       | (PW:4)           | 2D0E |
| GetStructRgn  | HSW:4     | (PW:4)           | 2E0E |
| GetContRgn    | HCW:4     | (PW:4)           | 2F0E |
| GetUpdateRgn  | HSW:4     | (PW:4)           | 300E |
| GetDefProc    | HDP:4     | (PW:4)           | 310E |
| SetDefProc    |           | (HSW:4 PW:4)     | 320E |
| GetWControls  | ACL:4     | (PW:4)           | 330E |
| GetFControls  | ACF:4     | (PW:4)           | 340E |
| GetInfoText   | PT:4      | (PW:4)           | 350E |
| SetInfoText   |           | (PT:4 PW:4)      | 360E |
| GetFullRect   | RCT:4     | (PW:4)           | 370E |
| SetFullRect   |           | (RCT:4 PW:4)     | 380E |
| Refresh       |           |                  | 390E |
| InvalRect     |           | (RCT:4)          | 3A0E |
| InvalRgn      |           | (HR:4)           | 3B0E |
| ValidRect     |           | (RCT:4)          | 3C0E |
| ValidRgn      |           | (HR:4)           | 3D0E |
| GetCOrigin    | X:4 Y:4   | (PW:4)           | 3E0E |
| SetCOrigin    |           | (PW:4)           | 3F0E |
| GetDataSize   | ML:2 MH:2 | (PW:4)           | 400E |
| SetDataSize   |           | (ML:2 MH:2 PW:4) | 410E |
| GetMaxGrow    | ML:2 MH:2 | (PW:4)           | 420E |
| SetMaxGrow    |           | (T:4 PW:4)       | 430E |
| GetScroll     | SVH:4     | (PW:4)           | 440E |
| SetScroll     |           | (SVH:4 PW:4)     | 450E |
| GetPage       | VH:4      | (PW:4)           | 460E |
| SetPage       |           | (V:2 H:2 PW:4)   | 470E |
| GetCDraw      | ADR:4     | (PW:4)           | 480E |
| SetCDraw      |           | (ADR:4 PW:4)     | 490E |
| GetInfoDraw   | ADRI:4    | (PW:4)           | 4A0E |
| SetSysWindow  |           |                  | 4B0E |
| GetSysWFlag   |           |                  | 4C0E |
| StartDrawing  |           | (PW:4)           | 4D0E |

Fonctions de l'outil N° \$0F MENU MANAGER (Sélection par menus déroulants) TOOL015

|              |     |                       |      |
|--------------|-----|-----------------------|------|
| MenuBootInit |     |                       | 010F |
| MenuStartup  |     | (ID:2 Z:2) _InitMenus | 020F |
| MenuShutDown |     |                       | 030F |
| MenuVersion  | V:2 |                       | 040F |
| MenuReset    |     |                       | 050F |
| MenuStatus   | B:2 |                       | 060F |
| MenuKey      |     | (PER:4 HMB:4)         | 090F |

## OUTILS

|               |            |                     |      |
|---------------|------------|---------------------|------|
| GetMenuBar    | HMB:4      |                     | 0A0F |
| MenuRefresh   |            | (ADR:4)             | 0B0F |
| FlashMenuBar  |            |                     | 0C0F |
| InsertMenu    |            | (HM:4 IDM:2)        | 0D0F |
| DeleteMenu    |            | (IDM:2)             | 0E0F |
| InsertItem    |            | (PDI:4 IDI:2 IDM:2) | 0F0F |
| DeleteItem    |            | (IDI:2)             | 100F |
| GetSysBar     | HMB:4      |                     | 110F |
| SetSysBar     |            | (HMB:4)             | 120F |
| FixMenuBar    | Mh:2       |                     | 130F |
| CountMItems   | NI:2       | (IDM:2)             | 140F |
| NewMenuBar    | HMB:4      | (PPO:4)             | 150F |
| SetBarColors  |            | (CN:2 CI:2 CC:2)    | 160F |
| GetBarColors  | CM:4       |                     | 170F |
| SetTitleStart |            | (XT:2)              | 180F |
| GetTitleStart | XT:2       |                     | 190F |
| GetMenuPtr    | PM:4       | (QM:2 IDM:2)        | 1A0F |
| CalcMenuSize  |            | (LT:2 Mh:2 IDM:2)   | 1B0F |
| SetTitleWidth |            | (LT:2 IDM:2)        | 1C0F |
| GetTitleWidth | LT:2       | (IDM:2)             | 1D0F |
| SetMenuFlag   |            | (NE:2 MM:2 IDM:2)   | 1E0F |
| GetMenuFlag   | NE:2       | (IDM:2)             | 200F |
| SetMenuTitle  |            | (PT:4 IDM:2)        | 210F |
| GetMenuTitle  | PT:4       | (IDM:2)             | 220F |
| GetItemPtr    | PI:4       | (QI:2 IDI:2)        | 230F |
| SetItem       |            | (PT:4 IDI:2)        | 240F |
| GetItem       | PT:4       | (IDI:2)             | 250F |
| SetItemFlag   | IM:2       | (SI:2 IDM:2)        | 260F |
| GetItemFlag   | SS:2 XOR:2 | (IDI:2)             | 270F |
| SetItemBlink  |            | (CT:2)              | 280F |
| MNewRes       |            |                     | 290F |
| DrawMenuBar   |            |                     | 2A0F |
| MenuSelect    |            | (TAR:4 HMB:4)       | 2B0F |
| HiliteMenu    |            | (BC:2 IDM:2)        | 2C0F |
| NewMenu       | HM:4       | (PMS:4)             | 2D0F |
| DisposeMenu   |            | (HM:4)              | 2E0F |
| InitPalette   |            |                     | 2F0F |
| EnableItem    |            | (IDI:2)             | 300F |
| DisableItem   |            | (IDI:2)             | 310F |
| CheckItem     |            | (BI:2 IDI:2)        | 320F |
| SetItemMark   |            | (MK:2 IDI:2)        | 330F |
| GetItemMark   | MK:2       | (IDI:2)             | 340F |
| SetItemStyle  |            | (St :2 IDI:2)       | 350F |
| GetItemStyle  | St :2      | (IDI:2)             | 360F |
| SetMenuID     |            | (ID:2 IDM:2)        | 370F |
| SetItemID     |            | (ID:2 IDI:2)        | 380F |
| SetMenuBar    |            | (HMB:4)             | 390F |

Fonctions de l'outil N° \$10 : CONTROL MANAGER (Gestionnaire de commandes souris non-standards dans une fenêtre) TOOL016



|                |       |                                                       |      |
|----------------|-------|-------------------------------------------------------|------|
| CtrlBootInit   |       |                                                       | 0110 |
| CtrlStartup    |       | (ID:2 Z:2)                                            | 0210 |
| CtrlShutDown   |       |                                                       | 0310 |
| CtrlVersion    | V:2   |                                                       | 0410 |
| CtrlReset      |       |                                                       | 0510 |
| CtrlStatus     | B:2   |                                                       | 0610 |
| NewControl     | HCL:4 | (PW:4 RCT:4 PT:4 F:2 V:2 V1:2<br>V2:2 AP:4 AV:4 PA:4) | 0910 |
| DisposeControl |       | (HCL:4)                                               | 0A10 |
| KillControl    |       | (PW:4)                                                | 0B10 |
| SetCTitle      |       | (PT:4 HCL:4)                                          | 0C10 |
| GetCTitle      | PT:4  | (HCL:4)                                               | 0D10 |
| HideControl    |       | (HCL:4)                                               | 0E10 |
| ShowControl    |       | (HCL:4)                                               | 0F10 |
| DrawnControls  |       | (PW:4)                                                | 1010 |
| HiliteControl  |       | (CS:2 HLE:4)                                          | 1110 |
| CtrlNewRes     |       |                                                       | 1210 |
| FindControl    | C:2   | (HFC:4 XG:2 YG:2 PW:4)                                | 1310 |
| TestControl    | C:2   | (XL:2 YL:2 HCL:4)                                     | 1410 |
| TrackControl   | C:2   | (XD:2 YD:2 ADR:4 HCL:4)                               | 1510 |
| MoveControl    |       | (NX:2 NY:2 HCL:4)                                     | 1610 |
| DragControl    |       | (XD:2 YD:2 RCTL:4 RCTS:4<br>AX:2 HCL:4)               | 1710 |
| SizeControl    |       | (NI:2 Nh:2 HCL:4)                                     | 1810 |
| SetCtlValue    |       | (V:2 HCL:4)                                           | 1910 |
| GetCtlValue    | V:2   | (HCL:4)                                               | 1A10 |
| SetCtlParams   |       | (V2:2 V1:2 HCL:4)                                     | 1B10 |
| GetCtlParams   | V12:4 | (HCL:4)                                               | 1C10 |
| DragRect       | DYX:4 | (ADR:4 PN:4 XD:2 YD:2 RCTD:4<br>RCTL:4 RCTS:4 AX:2)   | 1D10 |
| GrowSize       |       |                                                       | 1E10 |
| GetCtrlzpage   | Z:2   |                                                       | 1F10 |

Fonctions de l'outil N° \$11 : SYSTEM LOADER (Chargeur de segments)

|                |                    |                        |      |
|----------------|--------------------|------------------------|------|
| TOOL017        |                    |                        |      |
| LoaderBootInit |                    |                        | 0111 |
| LoaderStartup  |                    |                        | 0211 |
| LoaderShutDown |                    |                        | 0311 |
| LoaderVersion  | V:2                |                        | 0411 |
| LoaderReset    |                    |                        | 0511 |
| LoaderStatus   | B:2                |                        | 0611 |
| InitialLoad    | ID:2 AD:4 Z:2 LZ:2 | (ID:2 AN:4 IS:2)       | 0911 |
| Restart        | ID:2 AD:4 Z:2 LZ:2 | (ID:2)                 | 0A11 |
| LoadSegNum     | HS:4               | (ID:2 FN:2 SN:2)       | 0B11 |
| UnLoadSegnum   |                    | (ID:2 FN:2 SN:2)       | 0C11 |
| LoadSegName    | HS:4 FN:2 SN:2     | (ID:2 AN:4 AS:4)       | 0D11 |
| GetUserId      | ID:2               | (AN:4)                 | 0E11 |
| GetLoadSegInfo |                    | (ID:2 FN:2 SN:2 ADU:4) | 0F11 |
| LockSeg        |                    | (FN:2 SN:2)            | 1011 |
| UnlockSeg      |                    | (FN:2 SN:2)            | 1111 |

LOGICIELS DEVELOPPEURS

## OUTILS

UserShutDown ID:2 (ID:2) 1211

Fonctions de l'outil N° \$12 : **High Level Printer Driver** (imprimer) TOOL018  
 Fonctions de l'outil N° \$13 : **Low Level Printer Driver** (gestion d'imprimante)  
 TOOL019

Fonctions de l'outil N° \$14 : **LINE EDIT** (Editeur de lignes) TOOL020

|               |       |                       |      |
|---------------|-------|-----------------------|------|
| LEBootInit    |       |                       | 0114 |
| LEStartup     |       | (Z:2 ID:2)            | 0214 |
| LEShutDown    |       |                       | 0314 |
| LEVersion     | V:2   |                       | 0414 |
| LEReset       |       |                       | 0514 |
| LEActive      | B:2   |                       | 0614 |
| LENew         | HLE:4 | (RCTD:4 RCTV:4 LG:2)  | 0914 |
| LEDispose     |       | (HLE:4)               | 0A14 |
| LESetText     |       | (PT:4 LG:2 HLE:4)     | 0B14 |
| LEIdle        |       | (HLE:4)               | 0C14 |
| LEClick       |       | (EVT:4 HLE:4)         | 0D14 |
| LESetSelect   |       | (début:2 fin:2 HLE:4) | 0E14 |
| LEActivate    |       | (HLE:4)               | 0F14 |
| LEDeactivate  |       | (HLE:4)               | 1014 |
| LEKey         |       | (T:2 M:2 HLE:4)       | 1114 |
| LECut         |       | (HLE:4)               | 1214 |
| LECopy        |       | (HLE:4)               | 1314 |
| LEPaste       |       | (HLE:4)               | 1414 |
| LEDelete      |       | (HLE:4)               | 1514 |
| LEInsert      |       | (PT:4 LG:2 HLE:4)     | 1614 |
| LEUpdate      |       | (HLE:4)               | 1714 |
| LETextBox     |       | (PT:4 LG:2 RCT:4 J:2) | 1814 |
| LEFromScrap   |       |                       | 1914 |
| LEToScrap     |       |                       | 1A14 |
| LEScrapHandle | HSC:4 |                       | 1B14 |
| LEGetScrapLen | LG:2  |                       | 1C15 |
| LESetScrapLen | LG:2  |                       | 1D15 |
| LESetHilite   |       | (PAC:4 HLE:4)         | 1E14 |
| LESetCaret    |       | (PCA:4 HLE:4)         | 1F14 |

Fonctions de l'outil N° \$15 : **DIALOG MANAGER** (dialoguons) TOOL021

|                   |       |                                 |      |
|-------------------|-------|---------------------------------|------|
| DialogBootInit    |       |                                 | 0115 |
| DialogStartup     |       | (ID:2)                          | 0215 |
| DialogShutDown    |       |                                 | 0315 |
| DialogVersion     | V:2   |                                 | 0415 |
| DialogReset       | B:2   |                                 | 0515 |
| DialogStatus      |       |                                 | 0615 |
| ErrorSound        |       | (PSon:4)                        | 0715 |
| SetDAFont         |       | (HF:4)                          | 0815 |
| NewDialog         |       |                                 | 0915 |
| NewModalDialog    | PDL:4 | (RCTL:4 Vs:2 val:2)             | 0A15 |
| NewModelessDialog | PDL:4 | (RCTL:4 PFD:4 C:2 val:2 RECT:4) | 0B15 |
| CloseDialog       | PDL:4 |                                 | 0C15 |



|                   |         |                                           |       |
|-------------------|---------|-------------------------------------------|-------|
| NewDItem          |         | (PDL:4 ID:2 RCT:4 Type:2 Pdsc:4 Vs:4 C:4) | 0D15  |
| RemoveItem        |         | (PDL:4 ID:2)                              | 0E15  |
| ModalDialog       | ID:2    | (Pfiltrage:4)                             | 0F15  |
| IsDialogEvent     | B:2     | (EVT:4)                                   | 1015* |
| DialogSelect      | B:2     | (EVT:4 PDL:4 Pitem:4)                     | 1115* |
| DlgCut            |         | (PDL:4)                                   | 1215  |
| DlgCopy           |         | (PDL:4)                                   | 1315  |
| DlgPaste          |         | (PDL:4)                                   | 1415  |
| DlgDelete         |         | (PDL:4)                                   | 1515  |
| DrawDialog        |         | (PDL:4)                                   | 1615  |
| Alert             | ID:2    | (PPRM:4 Pfiltrage:4)                      | 1715  |
| StopAlert         | ID:2    | (PPRM:4 Pfiltrage:4)                      | 1815* |
| NoteAlert         | ID:2    | (PPRM:4 Pfiltrage:4)                      | 1915* |
| CautionAlert      | ID:2    | (PPRM:4 Pfiltrage:4)                      | 1A15* |
| ParamText         |         | (PC1:4 PC2:4 PC3:4 PC4:4)                 | 1B15* |
| TalkAlert         | ID:2    | (PPRM:4 Pfiltrage:4)                      | 1C15* |
| QuickAlert        | ID:2    | (PPT:4 Pmessage:4 Pfiltrage:4)            | 1D15* |
| GetControlItem    | PCTRL:4 | (PDL:4 ID:2)                              | 1E15  |
| GetIText          |         | (PDL:4 ID:2 PC:4)                         | 1F15  |
| SetIText          |         | (PDL:4 ID:2 PC:4)                         | 2015  |
| SellText          |         | (PDL:4 ID:2 debut:2 fin:2)                | 2115  |
| HideDitem         |         |                                           | 2215  |
| ShowDitem         |         |                                           | 2315  |
| FindDItem         |         |                                           | 2415  |
| UpdtDialog        |         |                                           | 2515  |
| GetItemType       | Type:2  | (PDL:4 ID:2)                              | 2615  |
| SetItemType       |         | (Type:2 PDL:4 ID:2)                       | 2715  |
| GetItemBox        |         | (PDL:4 ID:2 RCT:4)                        | 2815  |
| SetItemBox        |         | (PDL:4 ID:2 RCT:4)                        | 2915  |
| GetFirstItem      | ID:2    | (PDL:4)                                   | 2A15  |
| GetNextItem       | ID:2    | (PDL:4 ID:2)                              | 2B15  |
| GetItemFlag       | val:2   | (PDL:4 ID:2)                              | 2C15  |
| SetItemFlag       |         | (val:2 PDL:4 ID:2)                        | 2D15  |
| GetItemValue      | val:2   | (PDL:4 ID:2)                              | 2E15  |
| SetItemValue      |         | (val:2 PDL:4 ID:2)                        | 2F15  |
| GetItemColor      | CT:4    | (PDL:4 ID:2)                              | 3015  |
| SetItemColor      |         | (CT:4 PDL:4 ID:2)                         | 3115  |
| GetNewModalDialog | PDL:4   | (PPRM:4)                                  | 3215  |
| GetNewDItem       |         | (PDL:4 PPRM:4)                            | 3315  |

-----  
 Fonctions de l'outil N° \$16 : SCRAP MANAGER (couper, copier, coller)  
 TOOL022

|               |     |      |
|---------------|-----|------|
| ScrapBootInit |     | 0116 |
| ScrapStartup  |     | 0216 |
| ScrapShutdown |     | 0316 |
| ScrapVersion  | V:2 | 0416 |
| ScrapReset    |     | 0516 |
| ScrapStatus   |     | 0616 |
| UnloadScrap   |     | 0916 |

LOGICIELS DEVELOP

## OUTILS

|           |      |
|-----------|------|
| LoadScrap | 0A16 |
| ZeroScrap | 0B16 |
| PutScrap  | 0C16 |
| GetScrap  | 0D16 |

## Liste alphabétique des fonctions

Les paramètres sont explicités dans une autre liste : la liste outil par outil.  
Si la fonction ramène un résultat et s'il n'y a pas d'erreur (c=0), le résultat est disponible au-dessus de la pile.

| <i>Nom de la Macro</i>  | <i>N°outil</i> | <i>Fonction</i>                                                                             |
|-------------------------|----------------|---------------------------------------------------------------------------------------------|
| <b>AddPt</b>            | 04             | Somme de 2 points dans Quic Draw.                                                           |
| <b>Alert</b>            | 15             | Crée et affiche une fenêtre d'alerte donnée.                                                |
| <b>BeginUpDate</b>      | 0E             | Appelé pour traiter une remise en état de fenêtre.                                          |
| <b>BlockMove</b>        | 02             | Déplace un bloc.                                                                            |
| <b>Button</b>           | 06             | Renvoie la valeur vraie si le bouton est appuyé.                                            |
| <b>BringToFront</b>     | 0E             | Appelé par SelectWindow pour mettre la fenêtre au-dessus.                                   |
| <b>CalcMenuSize</b>     | 0E             | Fixe les dimensions d'un menu ou bien les calcule.                                          |
| <b>Caution</b>          | 15             | Crée et affiche une fenêtre d'alerte avec l'icône attention.                                |
| <b>CharBounds</b>       | 04             | Remplit un rectangle avec un caractère.                                                     |
| <b>CharWidth</b>        | 04             | Renvoie la largeur du caractère.                                                            |
| <b>CheckItem</b>        | 0F             | Marque ou non l'item donné.                                                                 |
| <b>CheckUpDate</b>      | 0E             | Appelé par l'Event Manager pour tester les fenêtres à remettre en état.                     |
| <b>Choosecda</b>        | 05             | Active le Desk Manager et affiche le menu des CDA.                                          |
| <b>ClampMouse</b>       | 03             | Fixe les valeurs limites dans lesquelles évolueront les coordonnées de la souris.           |
| <b>ClearMouse</b>       | 03             | Fixe à 0 ou au minimum positif, les axes X et Y dans lesquels se déplace le curseur-souris. |
| <b>ClearScreen</b>      | 04             | Fixe une valeur unique pour tous les pixels.                                                |
| <b>ClipRect</b>         | 04             | Change la clip région courante avec un rectangle donné.                                     |
| <b>CloseAllNDAs</b>     | 05             | Referme tous les accessoires.                                                               |
| <b>CloseDialog</b>      | 15             | Enlève le dialogue de l'écran et de la liste des fenêtres et libère de la mémoire.          |
| <b>CloseNDA</b>         | 05             | Referme l'accessoire donné par son n°.                                                      |
| <b>CloseNDAbyWinPtr</b> | 05             | Referme l'accessoire donné par son pointeur de fenêtre.                                     |
| <b>ClosePicture</b>     | 04             | Fin de traitement d'une image.                                                              |
| <b>ClosePoly</b>        | 04             | Fin de traitement d'un polygone.                                                            |



|                |    |                                                                       |
|----------------|----|-----------------------------------------------------------------------|
| ClosePort      | 04 | Désalloue la mémoire utilisée par le port.                            |
| CloseRgn       | 04 | Fin de traitement d'une région.                                       |
| CloseWindow    | 0E | Enlève la fenêtre donnée par son pointeur, du bureau.                 |
| CopyRgn        | 04 | Recopie le contenu d'une région dans une autre.                       |
| CountMItems    | 0F | Renvoie le nombre d'items d'un menu donné.                            |
| CStringBounds  | 04 | Remplit le rectangle d'une chaîne-C donnée.                           |
| CStringWidth   | 04 | Renvoie la largeur d'une chaîne-C.                                    |
| CtrlBootInit   | 10 | Initialisation du Control Manager.                                    |
| CtrlHeartBeat  | 03 | Efface toutes les tâches de la file d'attente du HeartBeat.           |
| CtrlTextDev    | 0C | Transmet un caractère de contrôle en entrée, sortie ou erreur.        |
| CtrlReset      | 10 | Mise à zéro du Control Manager.                                       |
| CtrlShutDown   | 10 | Fin d'utilisation du Control Manager, donc libération.                |
| CtrlStatus     | 10 | Renvoie l'état d'activité du Control Manager.                         |
| CtrlStartup    | 10 | Mise en service du Control Manager pour l'application ID.             |
| CtrlVersion    | 10 | Renvoie le numéro de version de l'outil Control Manager.              |
| Dec2Int        | 0B | Convertit une chaîne de chiffres décimaux en entier.                  |
| Dec2Long       | 0B | Convertit une chaîne de chiffres décimaux en entier long.             |
| DeleteID       | 03 | Efface toutes les références d'un type d'ID donné.                    |
| DeleteItem     | 0F | Efface un item donné par son ID.                                      |
| DeleteMenu     | 0F | Supprime un menu de la barre sans libérer de mémoire.                 |
| DelHeartBeat   | 03 | Efface une tâche de la file de celles déclenchées par le HeartBeat.   |
| DeskBootInit   | 05 | Initialisation du Desk Accessory Manager.                             |
| DeskReset      | 05 | Mise à zéro du Desk Accessory Manager.                                |
| DeskShutDown   | 05 | Fin d'utilisation du gestionnaire des accessoires.                    |
| DeskStatus     | 05 | Renvoie l'état d'activité de l'outil.                                 |
| DeskStartup    | 05 | Mise en service des accessoires de bureau.                            |
| DeskTop        | 0E | Demande l'exécution d'opérations de rangement du bureau électronique. |
| DeskVersion    | 05 | Renvoie le numéro de version de cet outil.                            |
| DialogBootInit | 15 | Initialisation du Dialog Manager.                                     |
| DialogReset    | 15 | Remet les valeurs standards.                                          |
| DialogShutDown | 15 | Fin d'utilisation et libération de la mémoire.                        |
| DialogStatus   | 15 | Renvoie vrai si le Dialog Manager a été mis en service.               |
| DialogStartup  | 15 | Mise en service du Dialog Manager pour l'application ID.              |
| DialogVersion  | 15 | Renvoie le n° de version du Dialog Manager.                           |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                       |    |                                                               |
|-----------------------|----|---------------------------------------------------------------|
| <b>DisposeControl</b> | 10 | Supprime le contrôle de l'écran et libère la mémoire.         |
| <b>DiffRgn</b>        | 04 | Calcule la différence entre 2 régions.                        |
| <b>DisableItem</b>    | 0F | Empêche la sélection d'un item donné et l'estompe.            |
| <b>DisposeALL</b>     | 02 | Libère tous les Handles.                                      |
| <b>DisposeHandle</b>  | 02 | Libère le Handle spécifié.                                    |
| <b>DisposeMenu</b>    | 0F | Libère la mémoire allouée avec NewMenu.                       |
| <b>DisposeRgn</b>     | 04 | Libère l'espace de la région spécifiée par son Handle.        |
| <b>DlgCopy</b>        | 15 | Si le dialogue contient des lignes à copier, alors LECopy.    |
| <b>DlgCut</b>         | 15 | Si le dialogue contient des lignes à couper, alors LECut.     |
| <b>DlgDelete</b>      | 15 | Si le dialogue contient des lignes à effacer, alors LEDelete. |
| <b>DlgPaste</b>       | 15 | Si le dialogue contient des lignes à coller, alors LEPaste.   |
| <b>DoWindows</b>      | 05 | Appelé par le Window Manager à son initialisation.            |
| <b>DragControl</b>    | 10 | Fait glisser le contrôle d'après les mouvements souris.       |
| <b>DragRect</b>       | 10 | Déplace un rectangle en pointillé en suivant la souris.       |
| <b>DragWindow</b>     | 0E | Déplace la fenêtre sous contrôle de la souris.                |
| <b>DrawChar</b>       | 04 | Trace un caractère en Quick Draw.                             |
| <b>DrawControl</b>    | 10 | Dessine tous les contrôles d'une fenêtre donnée.              |
| <b>DrawCString</b>    | 04 | Dessine la chaîne de caractères de type C.                    |
| <b>DrawDialog</b>     | 15 | Dessine le contenu d'un dialogue donné.                       |
| <b>DrawMenuBar</b>    | 0F | Dessine la barre des menus courante.                          |
| <b>DrawPicture</b>    | 04 | Dessine l'image.                                              |
| <b>DrawString</b>     | 04 | Dessine la chaîne indiquée.                                   |
| <b>DrawText</b>       | 04 | Dessine le texte indiqué.                                     |
| <b>EMActive</b>       | 06 | Renvoie l'état d'activité de l'Event Manager.                 |
| <b>EMBootInit</b>     | 06 | Initialisation de l'Event Manager.                            |
| <b>EMShutdown</b>     | 06 | Fin d'utilisation de l'Event Manager.                         |
| <b>EMStartup</b>      | 06 | Mise en service de l'Event Manager avec une page zéro.        |
| <b>EMVersion</b>      | 06 | Renvoie le numéro de version de l'outil Event Manager.        |
| <b>EmptyRect</b>      | 04 | Teste qu'un rectangle est vide ( $H1 > H2$ ou $V1 > V2$ ).    |
| <b>EmptyRgn</b>       | 04 | Teste qu'une région est vide.                                 |
| <b>EnableItem</b>     | 0F | Autorise la sélection de l'item donné.                        |
| <b>EndUpDate</b>      | 0E | Appelé si un BeginUpDate précède, pour restaurer visRgn.      |
| <b>EqualPt</b>        | 04 | Teste si 2 points sont identiques.                            |
| <b>EqualRect</b>      | 04 | Teste l'égalité de 2 rectangles.                              |
| <b>EqualRgn</b>       | 04 | Teste l'égalité de 2 régions.                                 |



|                          |    |                                                                                         |
|--------------------------|----|-----------------------------------------------------------------------------------------|
| <b>EraseArc</b>          | 04 | Remplit l'arc inscrit dans un rectangle donné avec le fond.                             |
| <b>EraseOval</b>         | 04 | Remplit l'ellipse avec le motif de fond.                                                |
| <b>ErasePoly</b>         | 04 | Remplit le polygone.                                                                    |
| <b>EraseRect</b>         | 04 | Remplit le rectangle avec le motif de fond.                                             |
| <b>EraseRgn</b>          | 04 | Remplit la région avec le motif du fond.                                                |
| <b>EraseRREct</b>        | 04 | Remplit le rectangle arrondi avec le motif de fond.                                     |
| <b>ErrorSound</b>        | 15 | Fixe une procédure d'émission de sons pour les alertes. (0 pour la procédure standard). |
| <b>ErrWriteLine</b>      | 0C | Transmet une ligne vers le dispositif d'affichage d'erreur.                             |
| <b>ErrWriteBlock</b>     | 0C | Transmet un bloc vers le dispositif d'affichage d'erreur.                               |
| <b>ErrWriteCString</b>   | 0C | Transmet une chaîne-C au dispositif d'affichage d'erreur.                               |
| <b>EventAvail</b>        | 06 | Renvoie le dernier événement en le laissant en attente.                                 |
| <b>FFGeneratorStatus</b> | 08 | Renvoie l'état d'un générateur de sons donné.                                           |
| <b>FFSoundDoneStatus</b> | 08 | Etat de fin de génération de sons.                                                      |
| <b>FFSoundStatus</b>     | 08 | Renvoie l'état des 15 oscillateurs de sons.                                             |
| <b>FFStartSound</b>      | 08 | Autorise le DOC à débiter la génération de sons d'un osc.                               |
| <b>FFStopSound</b>       | 08 | Arrête la génération des sons.                                                          |
| <b>FillArc</b>           | 04 | Remplit l'intérieur d'un arc inscrit dans un rectangle donné.                           |
| <b>FillOval</b>          | 04 | Remplit l'ellipse avec un motif donné.                                                  |
| <b>FillPoly</b>          | 04 | Remplit le polygone.                                                                    |
| <b>FillRect</b>          | 04 | Remplit le rectangle avec un motif donné.                                               |
| <b>FillRgn</b>           | 04 | Remplit la région avec un motif donné.                                                  |
| <b>FindControl</b>       | 10 | Détecte sur quel contrôle se trouvait la souris au moment où le bouton a été enfoncé.   |
| <b>FindHandle</b>        | 02 | Renvoie le Handle du bloc incluant l'adresse spécifiée.                                 |
| <b>FindWindow</b>        | 0E | Renvoie la région de la fenêtre où se trouve le curseur.                                |
| <b>FixAppleMenu</b>      | 05 | Ajoute les noms des accessoires dans le menu "Pomme".                                   |
| <b>FixMenuBar</b>        | 0F | Renvoie la hauteur calculée de la barre des menus.                                      |
| <b>FixMul</b>            | 0B | Fait le produit de 2 entiers longs en 32 bits, virgule fixe.                            |
| <b>FixRatio</b>          | 0B | Fait le rapport de 2 entiers relatifs en 32 bits, virgule fixe.                         |
| <b>FlashMenuBar</b>      | 0F | Fait clignoter la barre des menus.                                                      |
| <b>FlushEvents</b>       | 06 | Supprime tous les événements en attente dans la file.                                   |
| <b>ForceBufDims</b>      | 04 | Force la taille des buffers de textes et de clip.                                       |
| <b>FrameArc</b>          | 04 | Dessine le contour d'un arc inscrit dans un rectangle donné.                            |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                        |    |                                                                                                                                       |
|------------------------|----|---------------------------------------------------------------------------------------------------------------------------------------|
| <b>FrameOval</b>       | 04 | Dessine le contour d'une ellipse avec le crayon courant.                                                                              |
| <b>FramePoly</b>       | 04 | Dessine le contour d'un polygone.                                                                                                     |
| <b>FrameRect</b>       | 04 | Dessine le contour d'un rectangle avec le crayon courant.                                                                             |
| <b>FrameRgn</b>        | 04 | Dessine le contour d'une région.                                                                                                      |
| <b>FrameRRect</b>      | 04 | Dessine le contour d'un rectangle arrondi.                                                                                            |
| <b>FreeMem</b>         | 02 | Renvoie le nombre d'octets libres.                                                                                                    |
| <b>FrontWindow</b>     | 0E | Renvoie le pointeur de la fenêtre active.                                                                                             |
| <b>FWEntry</b>         | 03 | Permet l'accès à des routines-système Apple II, depuis un programme en mode natif.                                                    |
| <b>GetAbsClamp</b>     | 03 | Renvoie les valeurs limites des coordonnées souris.                                                                                   |
| <b>GetArcRot</b>       | 04 | Renvoie la courbure d'un arc.                                                                                                         |
| <b>GetAddr</b>         | 03 | Renvoie une adresse d'une variable donnée du Système.                                                                                 |
| <b>GetBackColor</b>    | 04 | Renvoie la couleur de fond.                                                                                                           |
| <b>GetBackPat</b>      | 04 | Charge la valeur du motif de fond à une adresse donnée.                                                                               |
| <b>GetBarColors</b>    | 0F | Renvoie sur 16 bits les 3 couleurs de la barre.                                                                                       |
| <b>GetCaretTime</b>    | 06 | Temps de clignotement du curseur-pointeur de texte.                                                                                   |
| <b>GetCDraw</b>        | 0E | Renvoie l'adresse de la routine de dessin du contenu.                                                                                 |
| <b>GetCharExtra</b>    | 04 | Renvoie la valeur du paramètre CharExtra.                                                                                             |
| <b>GetClip</b>         | 04 | Charge une région donnée avec la valeur de ClipRegion.                                                                                |
| <b>GetClipHandle</b>   | 04 | Renvoie le handle de la ClipRgn.                                                                                                      |
| <b>GetColorEntry</b>   | 04 | Renvoie une couleur d'une palette.                                                                                                    |
| <b>GetColorTable</b>   | 04 | Affecte les couleurs d'une palette à une autre palette.                                                                               |
| <b>GetCOrigin</b>      | 0E | Renvoie les coordonnées de l'origine de la fenêtre dans la zone des données pour calculer les ascenseurs.                             |
| <b>GetControlItem</b>  | 15 | Renvoie le Handle du contrôle de l'item donné.                                                                                        |
| <b>GetContRgn</b>      | 0E | Renvoie le pointeur sur la région de contenu de la fenêtre.                                                                           |
| <b>GetCTitle</b>       | 10 | Renvoie le pointeur sur le titre du contrôle.                                                                                         |
| <b>GetCtlValue</b>     | 10 | Renvoie la valeur courante affectée au contrôle.                                                                                      |
| <b>GetCtlParams</b>    | 10 | Renvoie les valeurs des paramètres additionnels du contrôle.                                                                          |
| <b>GetCtrlzpage</b>    | 10 | Renvoie l'adresse de la page zéro du Control Manager.                                                                                 |
| <b>GetCursorAdr</b>    | 04 | Renvoie le pointeur du curseur courant.                                                                                               |
| <b>GetCurrentClamp</b> | 03 | Renvoie les valeurs limites des coordonnées souris                                                                                    |
| <b>GetDAstring</b>     | 05 | Renvoie le nom de l'accessoire.                                                                                                       |
| <b>GetDataSize</b>     | 0E | Renvoie les dimensions de la zone de données qui pourra être visualisée dans la fenêtre grâce au défilement et au contrôle de taille. |



|                       |    |                                                                                                    |
|-----------------------|----|----------------------------------------------------------------------------------------------------|
| <b>GetDbtTime</b>     | 06 | Renvoie l'intervalle de temps maxi d'un double click.                                              |
| <b>GetDefProc</b>     | 0E | Renvoie le pointeur de la procédure de définition de la fenêtre.                                   |
| <b>GetErrGlobals</b>  | 0C | Renvoie les masques AND et OR des messages d'erreur.                                               |
| <b>GetErrorDevice</b> | 0C | Renvoie le numéro du port d'affichage des erreurs.                                                 |
| <b>GetFControls</b>   | 0E | Renvoie le 1er contrôle de la liste des contrôles du contour.                                      |
| <b>GetFGSize</b>      | 04 | Renvoie la taille du font globals record.                                                          |
| <b>GetFirstItem</b>   | 15 | Renvoie le numéro d'ID du 1er item d'un dialogue.                                                  |
| <b>GetFont</b>        | 04 | Renvoie le Handle du jeu courant.                                                                  |
| <b>GetFontID</b>      | 04 | Renvoie l'ID du jeu.                                                                               |
| <b>GetFontFlags</b>   | 04 | Renvoie les indicateurs du jeu courant.                                                            |
| <b>GetFontInfo</b>    | 04 | Renvoie le pointeur sur les informations du jeu courant.                                           |
| <b>GetFontGlobals</b> | 04 | Renvoie le pointeur sur les info globales du jeu.                                                  |
| <b>GetForeColor</b>   | 04 | Renvoie la couleur de 1er plan.                                                                    |
| <b>GetFullRect</b>    | 0E | Renvoie le pointeur du rectangle donnant la taille maxi.                                           |
| <b>GetFuncPtr</b>     | 01 | Renvoie l'adresse de début d'une fonction d'un outil.                                              |
| <b>GetFrameColor</b>  | 0E | Charge la table des couleurs du contour de la fenêtre à l'adresse donnée.                          |
| <b>GetGrafProcs</b>   | 04 | Charge le pointeur du champ GrafProcs à l'adresse donnée.                                          |
| <b>GetHandleSize</b>  | 02 | Renvoie la taille du bloc pointé par le Handle donné.                                              |
| <b>GetInfoDraw</b>    | 0E | Renvoie l'adresse de la routine de tracé des info.                                                 |
| <b>GetInfoText</b>    | 0E | Renvoie la valeur passée dans la routine de tracé de la barre d'informations d'une fenêtre donnée. |
| <b>GetInGlobals</b>   | 0C | Renvoie les valeurs des masques AND et OR des caractères/entrés.                                   |
| <b>GetInputDevice</b> | 0C | Renvoie le type et l'adresse du port d'entrée de caractères.                                       |
| <b>GetIRQenbl</b>     | 03 | Renvoie l'état d'inhibition ou non des interruptions.                                              |
| <b>GetItem</b>        | 0F | Renvoie le pointeur sur le titre de l'item donné par son ID.                                       |
| <b>GetItemBox</b>     | 15 | Met le pointeur du rectangle d'affichage d'un item de dialogue à l'adresse spécifiée.              |
| <b>GetItemColor</b>   | 15 | Renvoie le pointeur de la palette de couleurs d'un item.                                           |
| <b>GetItemFlag</b>    | 0F | Renvoie l'état d'affichage d'un item de menu.                                                      |
| <b>GetItemFlag</b>    | 15 | Renvoie l'état d'un item de dialogue donné.                                                        |
| <b>GetItemMark</b>    | 0F | Renvoie le caractère qui sert de marque aux items.                                                 |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                          |    |                                                                                                                                       |
|--------------------------|----|---------------------------------------------------------------------------------------------------------------------------------------|
| <b>GetItemStyle</b>      | 0F | Renvoie le code du style de caractères d'item.                                                                                        |
| <b>GetItemType</b>       | 15 | Renvoie le type d'item (bouton, bouton-radio, case de ctrl).                                                                          |
| <b>GetIText</b>          | 15 | Met le pointeur du texte de l'item donné à l'adresse spécifiée.                                                                       |
| <b>GetLoadSegInfo</b>    | 11 | Met à une adresse donnée la valeur, correspondante au Load Segment donné, trouvée dans la table des segments.                         |
| <b>GetMasterSCB</b>      | 04 | Renvoie la valeur du Master SCB.                                                                                                      |
| <b>GetMaxGrow</b>        | 0E | Renvoie la largeur et la hauteur maxi d'un fenêtre.                                                                                   |
| <b>GetMenuBar</b>        | 0F | Renvoie le Handle de la barre des menus actuelle.                                                                                     |
| <b>GetMenuFlag</b>       | 0F | Renvoie l'état du menu donné.                                                                                                         |
| <b>GetMenuTitle</b>      | 0F | Renvoie le pointeur du titre du menu donné.                                                                                           |
| <b>GetMouse</b>          | 06 | Renvoie la position de la souris.                                                                                                     |
| <b>GetNextEvent</b>      | 06 | Renvoie dans l'Event Record le dernier événement.                                                                                     |
| <b>GetNextItem</b>       | 15 | Renvoie le numéro d'ID de l'item suivant celui donné.                                                                                 |
| <b>GetNextWindow</b>     | 0E | Renvoie le pointeur de la fenêtre suivante dans la liste.                                                                             |
| <b>GetNewDItem</b>       | 15 | Ajoute un item à la liste des dialogues en prenant les paramètres dans une zone appropriée.                                           |
| <b>GetNewID</b>          | 03 | Demande un nouveau numéro d'ID pour un type d'application donné.                                                                      |
| <b>GetNewModalDialog</b> | 15 | Crée un dialogue en "mode" et renvoie un pointeur sur le port du nouveau dialogue en prenant ses paramètres dans une zone appropriée. |
| <b>GetNumNDAs</b>        | 05 | Envoie le numéro de l'accessoire en cours.                                                                                            |
| <b>GetOutGlobals</b>     | 0C | Renvoie les masques AND et OR des caractères sortis.                                                                                  |
| <b>GetOutputDevice</b>   | 0C | Renvoie le type et l'adresse du port de sortie.                                                                                       |
| <b>GetOSEvent</b>        | 06 | Renvoie le dernier événement-système.                                                                                                 |
| <b>GetPage</b>           | 0E | Renvoie le nombre de pixels de défilement d'une page.                                                                                 |
| <b>GetPen</b>            | 04 | Renvoie la position du crayon.                                                                                                        |
| <b>GetPenMask</b>        | 04 | Charge la valeur du masque du crayon à l'adresse donnée.                                                                              |
| <b>GetPenMode</b>        | 04 | Charge la valeur de mode du crayon à l'adresse donnée.                                                                                |
| <b>GetPenPat</b>         | 04 | Charge la valeur du motif du crayon à l'adresse donnée.                                                                               |
| <b>GetPenSize</b>        | 04 | Charge la valeur de la taille du crayon à l'adresse donnée.                                                                           |
| <b>GetPenState</b>       | 04 | Charge l'état du crayon pris dans le GrafPort à l'adresse spécifiée.                                                                  |
| <b>GetPicSave</b>        | 04 | Renvoie la valeur du champ picsave du GrafPort.                                                                                       |



|                        |    |                                                                                                                   |
|------------------------|----|-------------------------------------------------------------------------------------------------------------------|
| <b>GetPixel</b>        | 04 | Renvoie la valeur d'un pixel.                                                                                     |
| <b>GetPolySave</b>     | 04 | Renvoie la valeur du champ PolySave.                                                                              |
| <b>GetPort</b>         | 04 | Renvoie dans la pile le pointeur du port courant.                                                                 |
| <b>GetPortLoc</b>      | 04 | Charge la Map Info courante à une adresse donnée.                                                                 |
| <b>GetPortRect</b>     | 04 | Renvoie la valeur du port rectangle courant.                                                                      |
| <b>GetRgnSave</b>      | 04 | Renvoie la valeur du champ RgnSave.                                                                               |
| <b>GetSCB</b>          | 04 | Renvoie le SCB d'une ligne donnée.                                                                                |
| <b>GetScroll</b>       | 0E | Renvoie le nombre de pixels de défilement par les flèches.                                                        |
| <b>GetSoundVolume</b>  | 08 | Renvoie la valeur du volume d'un générateur donné.                                                                |
| <b>GetSpaceExtra</b>   | 04 | Renvoie la valeur du SpaceExtra.                                                                                  |
| <b>GetStandardSCB</b>  | 04 | Renvoie la valeur du Scan Line Control Byte standard (0).                                                         |
| <b>GetStructRgn</b>    | 0E | Renvoie le handle de la région de la structure de la fenêtre.                                                     |
| <b>GetSysBar</b>       | 0F | Renvoie le handle de la barre de menus-système.                                                                   |
| <b>GetSysField</b>     | 04 | Renvoie la valeur du champ SysField du GrafPorts.                                                                 |
| <b>GetSysWFlag</b>     | 0E |                                                                                                                   |
| <b>GetTableAdresse</b> | 08 | Renvoie l'adresse de la table des routines rapides de Sound Manager.                                              |
| <b>GetTextFace</b>     | 04 | Renvoie la valeur du style.                                                                                       |
| <b>GetTextMode</b>     | 04 | Renvoie la valeur du mode.                                                                                        |
| <b>GetTextSize</b>     | 04 | Renvoie la hauteur du texte.                                                                                      |
| <b>GetTitleStart</b>   | 0F | Renvoie la position du début des titres de la barre.                                                              |
| <b>GetTitileWidth</b>  | 0F | Renvoie la largeur d'un titre de la barre.                                                                        |
| <b>GetTSPtr</b>        | 01 | Renvoie l'adresse de la table des adresses des fonctions d'un outil donné.                                        |
| <b>GetTick</b>         | 03 | Renvoie la valeur du compteur de Tops.                                                                            |
| <b>GetUpdateRgn</b>    | 0E | Renvoie le handle de la région de structure de la fenêtre.                                                        |
| <b>GetUserID</b>       | 11 | Renvoie le numéro d'identification à partir du nom du fichier.                                                    |
| <b>GetUserField</b>    | 04 | Renvoie la valeur du champ Userfield.                                                                             |
| <b>GetVector</b>       | 03 | Renvoie l'adresse du vecteur d'interruption d'un S/P d'interruption spécifique donné par son numéro de référence. |
| <b>GetVisHandle</b>    | 04 | Renvoie le handle de la VisRgn.                                                                                   |
| <b>GetWap</b>          | 01 | Renvoie l'adresse de la page zéro ou zone de travail d'un outil donné.                                            |
| <b>GetWControls</b>    | 0E | Renvoie l'adresse du 1er contrôle de la liste des contrôles.                                                      |
| <b>GetWFrame</b>       | 0E | Renvoie les paramètres de contour de fenêtre sur 16 bits.                                                         |
| <b>GetWKind</b>        | 0E | Renvoie le type de fenêtre (application ou système).                                                              |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                       |    |                                                                                                                                                                                                                            |
|-----------------------|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>GetWmgrPort</b>    | 0E | Renvoie le pointeur du port du Window Manager.                                                                                                                                                                             |
| <b>GetWRefCon</b>     | 0E | Renvoie la valeur de référence du port de la fenêtre.                                                                                                                                                                      |
| <b>GetWTitle</b>      | 0E | Renvoie le pointeur du titre de la fenêtre.                                                                                                                                                                                |
| <b>GlobalToLocal</b>  | 04 | Convertit un point en coordonnées locales.                                                                                                                                                                                 |
| <b>Graffoff</b>       | 04 | Met en mode texte 80 colonnes et non linéaire.                                                                                                                                                                             |
| <b>Grafon</b>         | 04 | Met en mode super haute-résolution.                                                                                                                                                                                        |
| <b>GrowWindow</b>     | 0E | Agrandit ou rétrécit la fenêtre sous contrôle de la souris.                                                                                                                                                                |
| <b>Hex2Int</b>        | 0B | Renvoie un entier non signé égal à la chaîne de chiffres H.                                                                                                                                                                |
| <b>Hex2Long</b>       | 0B | Renvoie un entier long à partir de ses chiffres hexa.                                                                                                                                                                      |
| <b>HexIt</b>          | 0B | Renvoie les 4 chiffres hexa d'un entier sans signe.                                                                                                                                                                        |
| <b>HideContrôle</b>   | 10 | Rend invisible le contrôle donné.                                                                                                                                                                                          |
| <b>HideCursor</b>     | 04 | Diminue de 1 le niveau de visibilité du curseur.                                                                                                                                                                           |
| <b>HidePen</b>        | 04 | Diminue de 1 le niveau de visibilité du crayon.                                                                                                                                                                            |
| <b>HideWindow</b>     | 0E | Rend la fenêtre invisible.                                                                                                                                                                                                 |
| <b>HiliteControl</b>  | 10 | Change le type de rehaussement du contrôle.                                                                                                                                                                                |
| <b>HiliteMenu</b>     | 0F | Rehausse ou non l'affichage du menu donné par son ID.                                                                                                                                                                      |
| <b>HiliteWindow</b>   | 0E | Appelé par SelectWindow pour rehausser le contour.                                                                                                                                                                         |
| <b>Hlock</b>          | 02 | Verrouille un Handle.                                                                                                                                                                                                      |
| <b>HlockAll</b>       | 02 | Verrouille tous les Handle.                                                                                                                                                                                                |
| <b>HomeMouse</b>      | 03 | Positionne le curseur-souris aux valeurs limites minima.                                                                                                                                                                   |
| <b>InsertMenu</b>     | 0F | Ajoute un menu dans la barre après celui indiqué.                                                                                                                                                                          |
| <b>Int2Dec</b>        | 0B | Convertit un entier relatif en chaîne de chiffres décimaux.                                                                                                                                                                |
| <b>InitColorTable</b> | 04 | Recopie la palette standard dans la palette spécifiée.                                                                                                                                                                     |
| <b>InitCursor</b>     | 04 | Réinitialise le curseur.                                                                                                                                                                                                   |
| <b>InitialLoad</b>    | 11 | Appelé par un programme-contrôleur pour charger un fichier de type Load File (\$B3-\$BF) ; renvoie l'adresse de chargement et l'adresse et la taille de la page zéro et de la pile de ce fichier relogeable et exécutable. |
| <b>Initmouse</b>      | 03 | Initialise les valeurs limites, le mode et l'état de la souris.                                                                                                                                                            |
| <b>InitPalette</b>    | 0F | Recrée la palette du logo Apple si les palettes ont changé.                                                                                                                                                                |
| <b>InitPort</b>       | 04 | Initialisation d'un port graphique comme port standard.                                                                                                                                                                    |
| <b>InitTextDevice</b> | 0C | Initialisation d'un des 3 ports (entrée, sortie ou erreur).                                                                                                                                                                |
| <b>InsertItem</b>     | 0F | Insère un item de menu après celui donné.                                                                                                                                                                                  |



|                      |    |                                                                                                                                                                       |
|----------------------|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>InsetRect</b>     | 04 | Agrandit ou réduit un rectangle d'un écart donné.                                                                                                                     |
| <b>InsetRgn</b>      | 04 | Agrandit ou réduit une région.                                                                                                                                        |
| <b>InstallCDA</b>    | 05 | Installe un nouveau CDA (accessoire mode texte).                                                                                                                      |
| <b>InstallNDA</b>    | 05 | Installe un nouveau NDA (accessoire style Macintosh).                                                                                                                 |
| <b>Int2Dec</b>       | 0B | Convertit un entier en chaîne de chiffres décimaux.                                                                                                                   |
| <b>Int2Hex</b>       | 0B | Convertit un entier en chaîne de chiffres hexadécimaux.                                                                                                               |
| <b>IntSource</b>     | 03 | Autorise ou inhibe certaines sources d'interruption.                                                                                                                  |
| <b>InvalRect</b>     | 0E | Change le rectangle dans lequel la fenêtre est redessinée.                                                                                                            |
| <b>InvalRgn</b>      | 0E | Change la région dans laquelle la fenêtre est redessinée.                                                                                                             |
| <b>InvertArc</b>     | 04 | Inverse les pixels à l'intérieur d'un arc inscrit dans un rectangle donné.                                                                                            |
| <b>InvertOval</b>    | 04 | Inverse les pixels à l'intérieur d'une ellipse.                                                                                                                       |
| <b>InvertPoly</b>    | 04 | Inverse les pixels à l'intérieur d'un polygone.                                                                                                                       |
| <b>InvertRect</b>    | 04 | Inverse les pixels à l'intérieur d'un rectangle.                                                                                                                      |
| <b>InvertRgn</b>     | 04 | Inverse les pixels à l'intérieur d'une région.                                                                                                                        |
| <b>InvertRRest</b>   | 04 | Inverse les pixels à l'intérieur d'un rectangle arrondi.                                                                                                              |
| <b>KillControl</b>   | 10 | Enlève tous les contrôles d'une fenêtre donnée.                                                                                                                       |
| <b>KillPicture</b>   | 04 | Libère la mémoire occupée par cette image.                                                                                                                            |
| <b>KillPoly</b>      | 04 | Libère la mémoire occupée par ce polygone.                                                                                                                            |
| <b>LEActivate</b>    | 14 | Le fragment de texte sélectionné est contrasté (rehaussé).                                                                                                            |
| <b>LEActive</b>      | 14 | Renvoie l'état d'activité de Line Edit (0 si désactivé).                                                                                                              |
| <b>LEBootInit</b>    | 14 | Initialisation de l'outil Line Edit , éditeur de ligne.                                                                                                               |
| <b>LEClick</b>       | 14 | A appeler dès que le bouton est enfoncé dans le View rectangle de l'Edit Record pour sélectionner un fragment, ou un mot (double-click), ou une ligne (triple-click). |
| <b>LECut</b>         | 14 | Découpe le fragment et le met dans le Scrap.                                                                                                                          |
| <b>LEDelete</b>      | 14 | Efface le fragment sélectionné.                                                                                                                                       |
| <b>LEDesactivate</b> | 14 | Désactive le fragment de texte sélectionné.                                                                                                                           |
| <b>LEDispose</b>     | 14 | Libère la mémoire utilisée par l'Edit Record spécifié.                                                                                                                |
| <b>LEFromScrap</b>   | 14 | Copie le Scrap du bureau (le Presse-papiers). dans le Scrap de Line Edit.                                                                                             |
| <b>LEIdle</b>        | 14 | Répéter cet appel pour faire clignoter le curseur d'insertion . L'intervalle minimum est réglé sur le tableau de bord par l'utilisateur.                              |
| <b>LEInsert</b>      | 14 | Insère le texte spécifié juste avant la zone de sélection.                                                                                                            |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                       |    |                                                                                                                                                                     |
|-----------------------|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>LEUpdate</b>       | 14 | A appeler dès qu'un événement de mise à jour se produit dans une fenêtre avec de l'édition de ligne.                                                                |
| <b>LEKey</b>          | 14 | Remplace le fragment sélectionné par le caractère entré.<br>Traite le caractère Backspace, Ctrl-F, Ctrl-X, CTRL-Y, flèche.                                          |
| <b>LENew</b>          | 14 | Crée un Edit Record et renvoie son handle.                                                                                                                          |
| <b>LEPaste</b>        | 14 | Colle le contenu du Scrap à l'emplacement du curseur.                                                                                                               |
| <b>LEReset</b>        | 14 | Renvoie une erreur si LE est actif, sinon ne fait rien.                                                                                                             |
| <b>LEScrapHandle</b>  | 14 | Renvoie le handle du Scrap.                                                                                                                                         |
| <b>LEScrapLen</b>     | 14 | Renvoie la longueur du Scrap.                                                                                                                                       |
| <b>LESetCaret</b>     | 14 | Fixe le champ CaretHook avec l'adresse de la routine qui dessine le curseur.                                                                                        |
| <b>LESetSelect</b>    | 14 | Fixe le domaine de sélection entre les 2 paramètres entrés.                                                                                                         |
| <b>LESetHilite</b>    | 14 | Fixe le champ HiliteHook avec l'adresse d'une routine qui affiche le texte en contrasté.                                                                            |
| <b>LESetText</b>      | 14 | Copie dans l'Edit Record le texte donné par son pointeur.                                                                                                           |
| <b>LEShutDown</b>     | 14 | Libère la mémoire utilisée par Line Edit.                                                                                                                           |
| <b>LEStartup</b>      | 14 | Met Line Edit en service et alloue un handle au Scrap ou zone de stockage provisoire d'un fragment de texte.                                                        |
| <b>LETextBox</b>      | 14 | Dessine le texte dans le rectangle spécifié avec justification et tient compte des retour-chariot.                                                                  |
| <b>LEToScrap</b>      | 14 | Copie le Scrap dans le Presse-papiers.                                                                                                                              |
| <b>LEVersion</b>      | 14 | Renvoie le numéro de version de l'outil Line Edit.                                                                                                                  |
| <b>Line</b>           | 04 | Dessine une ligne en suivant le déplacement donné.                                                                                                                  |
| <b>LineTo</b>         | 04 | Dessine une ligne depuis la position courante jusqu'à celle spécifiée.                                                                                              |
| <b>LoaderBootInit</b> | 11 | Initialisation de l'outil SYSTEM LOADER.                                                                                                                            |
| <b>LoaderReset</b>    | 11 | Appel sans effet.                                                                                                                                                   |
| <b>LoadSegNum</b>     | 11 | Chargement d'un Load Segment par son numéro.                                                                                                                        |
| <b>LoadSegName</b>    | 11 | Chargement d'un Load Segment par son nom.                                                                                                                           |
| <b>LoaderShutDown</b> | 11 | Appel sans effet.                                                                                                                                                   |
| <b>LoaderStatus</b>   | 11 | Renvoie l'état d'activation du System Loader (toujours vrai).                                                                                                       |
| <b>LoaderStartup</b>  | 11 | Appel sans effet.                                                                                                                                                   |
| <b>LoadTools</b>      | 01 | Charge en MEV les outils présents sur la disquette sous le préfixe SYSTEM.TOOLS, et sélectionnés dans une table par leur numéro et leur version minimum acceptable. |
| <b>LoadVersion</b>    | 11 | Renvoie le numéro de version de l'outil System Loader.                                                                                                              |
| <b>LocalToGlobal</b>  | 04 | Convertit un point en coordonnées globales.                                                                                                                         |
| <b>LockSeg</b>        | 11 | Verrouille le segment spécifié.                                                                                                                                     |



|              |    |                                                                                                |
|--------------|----|------------------------------------------------------------------------------------------------|
| Long2Dec     | 0B | Convertit un entier long en chaîne de chiffres décimaux.                                       |
| Long2Hex     | 0B | Convertit un entier long en chaîne de chiffres hexa.                                           |
| LongDivide   | 0B | Divise deux entiers longs.                                                                     |
| LongMul      | 0B | Multiplie deux entiers longs.                                                                  |
| MapPoly      | 04 | Application d'un polygone dans un autre cadre.                                                 |
| MapPt        | 04 | Application d'un point d'un cadre à l'autre.                                                   |
| MapRct       | 04 | Application d'un rectangle.                                                                    |
| MapRgn       | 04 | Application d'une région.                                                                      |
| MaxBlock     | 02 | Renvoie la taille du plus grand bloc de mémoire libre.                                         |
| MenuBootInit | 0F | Initialisation du menu Manager.                                                                |
| MenuKey      | 0F | Réalise la correspondance entre touche enfoncée et item.                                       |
| MenuRefresh  |    |                                                                                                |
| MenuReset    | 0F | Mise à zéro des paramètres de menu Manager.                                                    |
| MenuSelect   | 0F | Appelé dès que le bouton est enfoncé dans la barre.                                            |
| MenuShutdown | 0F | Fin d'utilisation et libération des mémoires des menus.                                        |
| MenuStatus   | 0F | Renvoie l'état d'activation du menu Manager.                                                   |
| MenuStartup  | 0F | Début de mise en place de la barre des menus.                                                  |
| MenuVersion  | 0F | Renvoie le numéro de version du menu Manager.                                                  |
| MMBootinit   | 02 | Initialisation du Memoy Manager effectuée par TLStartup.                                       |
| MMReset      | 02 | Remise à zéro du Memory Manager.                                                               |
| MMShutDown   | 02 | Fin d'utilisation du Memory Manager par l'application en cours.                                |
| MMStartup    | 02 | Début d'utilisation du Memory Manager par l'application spécifiée par son n° d'identification. |
| MMVersion    | 02 | Renvoie le numéro de version du Memory Manager.                                                |
| MNewRes      | 0F | Mise en place des menus avec le nouveau mode de résolution.                                    |
| ModalDialog  | 15 | Intercepte et traite les événements attendus dans le dialogue placé sur le dessus du bureau.   |
| NoteAlert    | 15 | Crée et affiche une fenêtre d'alerte avec l'icône note.                                        |
| Move         | 04 | Déplace le crayon d'une translation donnée.                                                    |
| MoveControl  | 10 | Déplace le contrôle jusqu'à une nouvelle position dans la fenêtre.                             |
| MovePortTo   | 04 | Change l'adresse du port rect. courant.                                                        |
| MoveTo       | 04 | Déplace le crayon jusqu'à la position indiquée.                                                |
| MoveWindow   | 0E | Déplace la fenêtre sans changer sa taille.                                                     |
| MTBootInit   | 03 | Initialisation du Miscellaneous Tool.                                                          |
| MTReset      | 03 | Remise à zéro du Miscellaneous Tool.                                                           |
| MTShutdown   | 03 | Fin d'utilisation du Miscellaneous Tool.                                                       |
| MTStartup    | 03 | Début d'utilisation du Miscellaneous Tool.                                                     |
| Multiply     | 0B | Multiplie 2 entiers.                                                                           |



## OUTILS

|                          |    |                                                                                                                                                                         |
|--------------------------|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Munger</b>            | 03 | Traitement de chaînes de caractères.                                                                                                                                    |
| <b>NewControl</b>        | 10 | Crée un contrôle et l'ajoute au début de la liste des ctrls.                                                                                                            |
| <b>NewDItem</b>          | 15 | Ajoute un nouvel item à la liste des dialogues.                                                                                                                         |
| <b>NewHandle</b>         | 02 | Demande l'allocation d'un bloc en mémoire et son Handle.                                                                                                                |
| <b>NewModalDialog</b>    | 15 | Crée un dialogue de type "mode" spécifié par 2 paramètres, alloue l'espace nécessaire et renvoie un handle sur le port de dialogue.                                     |
| <b>NewModelessDialog</b> | 15 | Crée un dialogue sans "mode" et renvoie le handle correspondant.                                                                                                        |
| <b>NewMenu</b>           | 0F | Alloue de la mémoire pour un menu et renvoie un handle qu'il faudra passer à InsertMenu pour l'insérer dans la liste des menus dont le pointeur est donné dans NewMenu. |
| <b>NewMenuBar</b>        | 0F | Crée une barre de menus et renvoie le handle associé.                                                                                                                   |
| <b>NewRgn</b>            | 04 | Alloue l'espace d'une région et renvoie son Handle.                                                                                                                     |
| <b>NewWindow</b>         | 0E | Crée une fenêtre donnée par le pointeur sur ses paramètres et insère cette fenêtre dans la liste des fenêtres.                                                          |
| <b>ObscureCursor</b>     | 04 | Cache le curseur jusqu'à ce que la souris bouge                                                                                                                         |
| <b>OffsetPoly</b>        | 04 | Déplace un polygone.                                                                                                                                                    |
| <b>OPenNDA</b>           | 05 | Ouvre l'accessoire dont le numéro est spécifié.                                                                                                                         |
| <b>OpenPicture</b>       | 04 | Renvoie un handle d'une nouvelle image.                                                                                                                                 |
| <b>OpenPoly</b>          | 04 | Renvoie un handle d'un polygone.                                                                                                                                        |
| <b>OpenPort</b>          | 04 | Ouvre un port graphique et l'utilise comme port standard.                                                                                                               |
| <b>OpenRgn</b>           | 04 | Début de gestion de l'espace d'une région.                                                                                                                              |
| <b>OSEventAvail</b>      | 06 | Renvoie le dernier événement-système en le laissant en attente.                                                                                                         |
| <b>PackBytes</b>         | 03 | Compacte des octets.                                                                                                                                                    |
| <b>PaintArc</b>          | 04 | Colorie l'intérieur d'un arc inscrit dans un rectangle donné.                                                                                                           |
| <b>PaintOval</b>         | 04 | Colorie l'ellipse avec le motif courant du crayon.                                                                                                                      |
| <b>PaintPixels</b>       | 04 | Transfert de pixels.                                                                                                                                                    |
| <b>PaintPoly</b>         | 04 | Colorie le polygone.                                                                                                                                                    |
| <b>PaintRect</b>         | 04 | Colorie le rectangle avec le motif courant du crayon.                                                                                                                   |
| <b>PaintRgn</b>          | 04 | Colorie la région.                                                                                                                                                      |
| <b>PaintRRect</b>        | 04 | Colorie le rectangle arrondi.                                                                                                                                           |
| <b>PicComment</b>        | 04 | Insère un commentaire dans une image donnée.                                                                                                                            |
| <b>PinRect</b>           | 0E | Renvoie le point dans le rectangle donné, le plus près du pixel donné par ses coordonnées.                                                                              |
| <b>PtInRect</b>          | 04 | Détecte si un point fait partie d'un rectangle donné.                                                                                                                   |



|               |    |                                                               |
|---------------|----|---------------------------------------------------------------|
| PtInRgn       | 04 | Détecte si un point fait partie d'une région donnée.          |
| Pt2Rect       | 04 | Copie les points extrêmes de 2 rectangles.                    |
| PenNormal     | 04 | Fixe l'état du crayon avec les valeurs standards.             |
| PosMouse      | 03 | Affecte des coordonnées à la souris.                          |
| PostEvent     | 06 | Place un événement dans la file.                              |
| PPToPort      | 04 | Transfert de pixels.                                          |
| PurgeAll      | 02 | Vidange de tous les Handle.                                   |
| PurgeHandle   | 02 | Vidange d'un Handle donné.                                    |
| QDBootInit    | 04 | Initialisation de Quick Draw II.                              |
| QDReset       | 04 | Mise à zéro de l'outil Quick Draw.                            |
| QDShutdown    | 04 | Fin d'utilisation de Quick Draw par l'application en cours.   |
| QDStatus      | 04 | Renvoie l'état d'activité de l'outil Quick Draw.              |
| QDStartup     | 04 | Début d'utilisation du Quick Draw avec entrées de paramètres. |
| QDVersion     | 04 | Renvoie la version de l'outil Quick Draw en service.          |
| OffsetRect    | 04 | Déplace un rectangle donné à une distance donnée.             |
| OffsetRgn     | 04 | Déplace une région.                                           |
| Random        | 04 | Renvoie un nombre aléatoire entre -32768 et 32767.            |
| ReadBParam    | 03 | Charge en MEV la valeur d'un paramètre de la MEV/pile.        |
| ReadBRAM      | 03 | Charge en MEV les données de la MEV alimentée par pile.       |
| ReadBlock     | 0C | Charge en mémoire un bloc saisi en entrée.                    |
| ReadChar      | 0C | Renvoie le caractère saisi en entrée avec ou sans écho.       |
| ReadLine      | 0C | Charge en mémoire une ligne de caractères entrés.             |
| ReadMouse     | 03 | Renvoie les coordonnées et l'état de la souris.               |
| ReadRamBlock  | 08 | Lit les données musicales.                                    |
| ReadTimeHex   | 03 | Renvoie l'heure en hexa.                                      |
| ReallocHandle | 02 | Réallocation d'un Handle à un bloc déjà alloué.               |
| RectInRgn     | 04 | Détecte si un rectangle coupe une région.                     |
| RectRgn       | 04 | Détruit une région en la remplaçant par le rectangle.         |
| Refresh       | 0E | Redessine tout le bureau et les fenêtres.                     |
| RemoveItem    | 15 | Enlève l'item de la liste et de l'écran.                      |
| RestAll       | 05 | Restaure les variables après l'appel d'un accessoire.         |
| Restart       | 11 | Redémarrage d'une application donnée par son ID.              |
| RestoreBufDim | 04 | Restaure la taille des buffers.                               |
| Restscrn      | 05 | Restaure l'écran texte 80 colonnes sauvé par Savescrn.        |

## OUTILS

|                |    |                                                                                                           |
|----------------|----|-----------------------------------------------------------------------------------------------------------|
| SaveAll        | 05 | Sauve le contexte pendant l'affichage d'un accessoire.                                                    |
| SaveBufDims    | 04 | Sauvegarde la taille des buffers.                                                                         |
| SaveScrn       | 05 | Sauve l'écran texte 80 colonnes.                                                                          |
| ScalePt        | 04 | Mise à l'échelle d'un point.                                                                              |
| SCHAddTask     | 07 | Ajoute une tâche à gérer par le Scheduler.                                                                |
| SCHActive      | 07 | Revoie l'état d'activité du Scheduler.                                                                    |
| SCHBootInit    | 07 | Initialisation du Scheduler.                                                                              |
| SCHedulerFlush | 07 | Enlève toutes les tâches à gérer par le Scheduler.                                                        |
| SCHReset       | 07 | Mise à zéro du Scheduler.                                                                                 |
| SCHShutDown    | 07 | Fin d'utilisation du Scheduler.                                                                           |
| SCHStartup     | 07 | Mise en service du Scheduler.                                                                             |
| SCHVersion     | 07 | Revoie le numéro de version du Scheduler.                                                                 |
| ScrollRect     | 04 | Défilement dans un rectangle de Quick Draw.                                                               |
| SDivide        | 0B | Calcule le quotient et le reste de la division de 2 entiers.                                              |
| SelIText       | 15 | Sélectionne un fragment de texte d'un item de dialogue.                                                   |
| SetBarColors   | 0F | Fixe les couleurs normale, inversée, rehaussée de la barre.                                               |
| SetCorigin     | 0E | Fixe l'origine de la zone de contenu d'une fenêtre.                                                       |
| SetDAFont      | 15 | Change le jeu de caractères des fenêtres de dialogue.                                                     |
| SetItemBox     | 15 | Fixe un nouveau rectangle d'affichage pour un item donné.                                                 |
| SetItemFlag    | 15 | Fixe un nouvel état à un item de dialogue donné.                                                          |
| SetItemType    | 15 | Change le type d'un item de dialogue donné.                                                               |
| SectRect       | 04 | Calcule l'intersection de 2 rectangles donnés.                                                            |
| SectRgn        | 04 | Calcule l'intersection de 2 régions.                                                                      |
| SendBehind     | 0E | Met la fenêtre derrière celle spécifiée en premier.                                                       |
| SelectWindow   | 0E | Active la fenêtre en la mettant au-dessus des autres.                                                     |
| ServeMouse     | 03 | Revoie l'état d'interruption déclenchée par la souris.                                                    |
| SetAbsClamp    | 03 | Fixe les valeurs limites du curseur-souris.                                                               |
| SetAllSCB      | 04 | Fixe la même valeur de SCB pour toutes les lignes.                                                        |
| SetArcRot      | 04 | Fixe la courbure de l'arc.                                                                                |
| SetBackColor   | 04 | Fixe la couleur de fond.                                                                                  |
| SetBackPat     | 04 | Fixe une valeur au motif de fond.                                                                         |
| SetCdraw       | 0E | Fixe l'adresse de la routine de dessin du contenu d'une fenêtre spécifiée par le pointeur de de son port. |
| SetCharExtra   | 04 | Fixe la valeur du paramètre CharExtra.                                                                    |
| SetClip        | 04 | Fixe la région passée avec CopyRgn à la clip région.                                                      |



|                          |    |                                                                                                                |
|--------------------------|----|----------------------------------------------------------------------------------------------------------------|
| <b>SetClipHandle</b>     | 04 | Fixe le handle de la clip région.                                                                              |
| <b>SetColorTable</b>     | 04 | Fixe les couleurs d'une palette.                                                                               |
| <b>SetColorEntry</b>     | 04 | Fixe une couleur dans une palette.                                                                             |
| <b>SetCtlParams</b>      | 10 | Fixe les paramètres additionnels d'un contrôle.                                                                |
| <b>SetCtlValue</b>       | 10 | Fixe une valeur courante au contrôle.                                                                          |
| <b>SetCOrigin</b>        | 0E | Fixe l'origine relative de la fenêtre dans la zone de données.                                                 |
| <b>SetCTitle</b>         | 10 | Fixe un nouveau titre au contrôle spécifié par son pointeur.                                                   |
| <b>SetCursor</b>         | 04 | Fixe le curseur à une nouvelle valeur.                                                                         |
| <b>SetCurrentPort</b>    | 04 | Fixe le port courant avec le port spécifié.                                                                    |
| <b>SetDAstring</b>       | 05 | Change le nom d'un accessoire.                                                                                 |
| <b>SetDataSize</b>       | 0E | Fixe la taille de la zone des données d'une fenêtre.                                                           |
| <b>SetDefProcs</b>       | 0E | Fixe l'adresse de tracé de la barre d'info d'une fenêtre.                                                      |
| <b>SetEmptyRgn</b>       | 04 | Détruit la région en la rendant vide.                                                                          |
| <b>SetEventMask</b>      | 06 | Fixe le masque des événements-système.                                                                         |
| <b>SetErrGlobals</b>     | 0C | Fixe les masques AND et OR des messages d'erreur.                                                              |
| <b>SetErrDevice</b>      | 0C | Fixe le type et l'adresse du port d'affichage des erreurs.                                                     |
| <b>SetFont</b>           | 04 | Fixe un jeu de caractères.                                                                                     |
| <b>SetFontID</b>         | 04 | Fixe le numéro ID à un jeu de caractères.                                                                      |
| <b>SetFontFlags</b>      | 04 | Fixe les indicateurs du jeu.                                                                                   |
| <b>SetFontForeColor</b>  | 04 | Fixe la couleur de 1er plan.                                                                                   |
| <b>SetFontFrameColor</b> | 0E | Fixe la couleur du contour d'une fenêtre (\$0 en standard).                                                    |
| <b>SetFullRect</b>       | 0E | Fixe le pointeur du rectangle à utiliser comme taille maxi.                                                    |
| <b>SetGrafProcs</b>      | 04 | Fixe la valeur du champ GrafProcs.                                                                             |
| <b>SetHandleSize</b>     | 02 | Affecte une taille à un bloc pointé par un handle donné.                                                       |
| <b>SetHeartBeat</b>      | 03 | Installe une tâche dans la file de prise en charge des interruptions déclenchées par le HeartBeat (1/50e sec). |
| <b>SetInfoDraw</b>       | 0E | Fixe l'adresse de la routine de tracé de la barre d'info.                                                      |
| <b>SetInfoText</b>       | 0E | Fixe la valeur à faire passer dans la routine de tracé d'info.                                                 |
| <b>SetIntUse</b>         | 04 | Utilisation ou non de l'interruption de balayage de ligne.                                                     |
| <b>SetInGlobals</b>      | 0C | Fixe les masques AND et OR des caractères à saisir.                                                            |
| <b>SetInputDevice</b>    | 0C | Fixe le type et l'adresse du port d'entrée de caractères.                                                      |
| <b>SetItem</b>           | 0F | Fixe un nouveau nom à un item de menu.                                                                         |
| <b>SetitemBlink</b>      | 0F | Fixe le temps de clignotement d'un item sélectionné.                                                           |
| <b>SetItemColor</b>      | 15 | Fixe une nouvelle palette de couleurs pour l'item donné.                                                       |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                        |    |                                                                                                                      |
|------------------------|----|----------------------------------------------------------------------------------------------------------------------|
| <b>SetItemFlag</b>     | 0F | Fixe un nouvel état à un item de menu donné.                                                                         |
| <b>SetItemFlag</b>     | 15 | Fixe un nouvel état à un item de dialogue donné.                                                                     |
| <b>SetitemID</b>       | 0F | Fixe une nouvelle valeur d'ID à l'item donné.                                                                        |
| <b>SetItemMark</b>     | 0F | Fixe le caractère affiché comme marque d'item.                                                                       |
| <b>SetitemStyle</b>    | 0F | Fixe le style d'affichage de l'item (standard, gras, italique).                                                      |
| <b>SetItemValue</b>    | 15 | Fixe une nouvelle valeur à un item de dialogue.                                                                      |
| <b>SetIText</b>        | 15 | Place le texte dans un item donné d'un dialogue.                                                                     |
| <b>SetMasterSCB</b>    | 04 | Fixe la valeur du MasterSCB.                                                                                         |
| <b>SetMaxGrow</b>      | 0E | Fixe la largeur et la hauteur maxi d'une fenêtre.                                                                    |
| <b>SetMenuBar</b>      | 0F | Fixe la barre de menus courante (\$0 barre-système).                                                                 |
| <b>SetMenuFlag</b>     | 0F | Fixe le nouvel état (autorisé/inhibé, normal/inv) d'un menu.                                                         |
| <b>SetMenuID</b>       | 0F | Fixe un nouveau ID au menu donné.                                                                                    |
| <b>SetMenuTitle</b>    | 0F | Fixe le nouveau titre du menu donné par son ID.                                                                      |
| <b>SetMouse</b>        | 03 | Attribue un mode opératoire à la souris.                                                                             |
| <b>SetMouseLoc</b>     | 04 | Fixe la position de la souris                                                                                        |
| <b>SetOutGlobals</b>   | 0C | Fixe les masques AND et OR des caractères à sortir.                                                                  |
| <b>SetOutputDevice</b> | 0C | Fixe le type et le numéro du port de sortie de caractères.                                                           |
| <b>SetOrigin</b>       | 04 | Ajustement du contenu du port rect et du Bounds Rect pour que le coin supérieur gauche du port ait la valeur donnée. |
| <b>SetPage</b>         | 0E | Fixe le nombre de pixels de défilement d'une page.                                                                   |
| <b>SetPenMask</b>      | 04 | Fixe une valeur pour le masque du crayon.                                                                            |
| <b>SetPenMode</b>      | 04 | Fixe une nouvelle valeur au mode de crayon courant.                                                                  |
| <b>SetPenPat</b>       | 04 | Fixe une nouvelle valeur au motif du crayon courant.                                                                 |
| <b>SetPenSize.</b>     | 04 | Fixe une nouvelle valeur à la taille du crayon courant.                                                              |
| <b>SetPenState</b>     | 04 | Fixe la nouvelle valeur de l'état du crayon dans le GrafPort.                                                        |
| <b>SetPicSave</b>      | 04 | Fixe la valeur du champ PicSave.                                                                                     |
| <b>SetPolySave</b>     | 04 | Fixe la valeur du champ PolySave.                                                                                    |
| <b>SetPortLoc</b>      | 04 | Fixe la valeur de la Map information courante.                                                                       |
| <b>SetPortRect</b>     | 04 | Fixe la valeur du port de rectangle courant.                                                                         |
| <b>SetPortSize</b>     | 04 | Fixe la taille du port rectangle à une nouvelle valeur.                                                              |
| <b>SetPt</b>           | 04 | Fixe un point.                                                                                                       |
| <b>SetPurge</b>        | 02 | Affecte un attribut à un bloc.                                                                                       |
| <b>SetPurgeAll</b>     | 02 | Affecte un attribut à tous les blocs d'une application donnée.                                                       |
| <b>SetRandSeed</b>     | 04 | Amorce le générateur de nombres aléatoires.                                                                          |



|                  |    |                                                                             |
|------------------|----|-----------------------------------------------------------------------------|
| SetRctRgn        | 04 | Détruit une région en la remplaçant par un rectangle.                       |
| SetRect          | 04 | Définit les limites d'un rectangle donné par son pointeur.                  |
| SetRgnSave       | 04 | Fixe la valeur du champ RgnSave.                                            |
| SetSCB           | 04 | Fixe la valeur du SCB d'une ligne donnée.                                   |
| SetScroll        | 0E | Fixe le nombre de pixels à faire défiler par les flèches.                   |
| SetSoundMIRQV    | 08 | Fixe le point d'entrée de la routine de service des int.                    |
| SetSolidBackPat  | 04 | Fixe le motif du fond à la couleur donnée.                                  |
| SetSolidPenPat   | 04 | Fixe le motif du crayon à la couleur donnée.                                |
| SetSoundVolume   | 04 | Fixe la valeur de volume d'un oscillateur de sons.                          |
| SetSpaceExtra    | 04 | Fixe une valeur au SpaceExtra.                                              |
| SetStdProcs      | 04 | Fixe les procédures standards de dessin.                                    |
| SetSysWindow     | 0E |                                                                             |
| SetUserSoundIRQV | 08 | Fixe le point d'entrée d'une routine-utilisateur de traitement d'IRQ.SOUND. |
| SetSwitch        | 06 | Appelé par le Control Manager.                                              |
| SetSysBar        | 0F | Fixe une nouvelle barre de menus donnée par son handle.                     |
| SetSysFont       | 04 | Fixe le jeu de caractères.                                                  |
| SetTextFace      | 04 | Fixe le style des caractères.                                               |
| SetTextMode      | 04 | Fixe le Mode des caractères.                                                |
| SetTextSize      | 04 | Fixe la taille des caractères.                                              |
| SetTitleWidth    | 0F | Fixe la largeur d'un titre et du menu correspondant.                        |
| SetTitleStart    | 0F | Fixe une nouvelle position pour le début des titres.                        |
| SetTSPtr         | 01 | Fixe l'adresse de la table des adresses des fonctions d'un outil spécifié.  |
| SetUserField     | 04 | Fixe la valeur du champ Userfield du GrafPort.                              |
| SetVector        | 03 | Affecte une adresse de vecteur d'interruption.                              |
| SetVisHandle     | 04 | Fixe le champ de handle de région à une valeur donnée.                      |
| SetWap           | 01 | Affecte un pointeur de page zéro à un outil.                                |
| SetWFrame        | 0E | Fixe les 16 bits de type de contour de la fenêtre.                          |
| SetWRefCon       | 0E | Fixe la valeur de référence dans le port de la fenêtre.                     |
| SetWTitle        | 0E | Fixe le titre de la fenêtre donnée.                                         |
| ShowControl      | 10 | Rend visible le contrôle.                                                   |
| ShowCursor       | 04 | Augmente de 1 la visibilité du curseur.                                     |
| ShowHide         | 0E | Montre ou cache la fenêtre.                                                 |
| ShowPen          | 04 | Augmente de 1 la visibilité du crayon.                                      |
| ShowWindow       | 0E | Rend visible une fenêtre sans changer la disposition.                       |
| SizeControl      | 10 | Change les dimensions d'un contrôle donné.                                  |
| SizeWindow       | 0E | Agrandit ou rétrécit la fenêtre aux dimensions données.                     |

LOGICIELS DE DEVELOPPEMENT

## OUTILS

|                        |    |                                                                                            |
|------------------------|----|--------------------------------------------------------------------------------------------|
| <b>SolidPattern</b>    | 04 | Fixe une couleur donnée à un motif donné.                                                  |
| <b>SoundBootInit</b>   | 08 | Initialisation de l'outil de production de sons.                                           |
| <b>SoundReset</b>      | 08 | Mise à zéro de l'outil Sound.                                                              |
| <b>SoundShutDown</b>   | 08 | Fin d'utilisation de l'outil Sound.                                                        |
| <b>SoundStartup</b>    | 08 | Mise à l'oeuvre de l'outil Sound.                                                          |
| <b>SoundToolStatus</b> | 08 | Renvoie l'état d'activité de l'outil Sound.                                                |
| <b>SoundVersion</b>    | 08 | Renvoie le numéro de version de l'outil Sound.                                             |
| <b>StartDrawing</b>    | 0E | Dessine la fenêtre indépendamment des événements de remise en état (Update Events).        |
| <b>StatusTDev</b>      | 0C | Fait un appel de Status à un dispositif d'entrée, de sortie ou d'affichage d'erreur.       |
| <b>StatusID</b>        | 03 | Fait savoir si un certain type d'ID est déjà actif.                                        |
| <b>StillDown</b>       | 06 | Renvoie la valeur vraie si le bouton est encore appuyé.                                    |
| <b>StopAlert</b>       | 15 | Crée et affiche une fenêtre d'alerte avec une icône stop.                                  |
| <b>StringBounds</b>    | 04 | Remplit un rectangle d'une chaîne de caractères.                                           |
| <b>StringWidth</b>     | 04 | Renvoie la largeur de la chaîne donnée.                                                    |
| <b>SubPt</b>           | 04 | Soustrait 2 points.                                                                        |
| <b>SysDeathMgr</b>     | 03 | Saut par le vecteur d'erreur fatale Système.                                               |
| <b>SystemClick</b>     | 05 | Appelé par un click de la souris.                                                          |
| <b>SystemTask</b>      | 05 | Appelé pour traiter les opérations systématiques d'un accès.                               |
| <b>TalkAlert</b>       | 15 | Crée et affiche une fenêtre d'alerte avec l'icône parlante.                                |
| <b>TaskMaster</b>      | 0E | Appelé pour traiter les événements touche et bouton enfoncés.                              |
| <b>TestControl</b>     | 10 | Appelé par FindControl pour détecter quelle partie de contrôle contient le point spécifié. |
| <b>TextBootInit</b>    | 0C | Initialisation du Text Tools pour gérer les E/S classiques.                                |
| <b>TextBounds</b>      | 04 | Remplit un rectangle d'un texte donné dans Quick Draw.                                     |
| <b>TextReset</b>       | 0C | Fixe les valeurs standards des paramètres de Text Tools.                                   |
| <b>TextShutDown</b>    | 0C | Fin d'utilisation du Text Tools.                                                           |
| <b>TextStartup</b>     | 0C | Mise en service du Text Tools.                                                             |
| <b>TextWidth</b>       | 04 | Renvoie la largeur du texte donné dans Quick Draw.                                         |
| <b>TickCount</b>       | 06 | Renvoie le nombre de tops depuis le démarrage.                                             |
| <b>TLBootInit</b>      | 01 | Initialisation du Tool Locator qui initialise tous les outils implantés en MEM.            |
| <b>TLReset</b>         | 01 | Remise à zéro du Tool Locator et de tous les autres.                                       |
| <b>TLShutdown</b>      | 01 | Fin d'utilisation du Tool Locator.                                                         |
| <b>TLStartup</b>       | 01 | Mise en marche du Tool Locator.                                                            |
| <b>TLVersion</b>       | 01 | Renvoie le n° de version du ToolLocator.                                                   |
| <b>TotalMem</b>        | 02 | Renvoie le nombre d'octets total de mémoire.                                               |



|                      |    |                                                                                                       |
|----------------------|----|-------------------------------------------------------------------------------------------------------|
| <b>TrackGoAway</b>   | 0E | Teste si la position donnée du curseur au moment où il est relâché est celle de la case de fermeture. |
| <b>TrackControl</b>  | 10 | Suit les mouvements de la souris dans un contrôle.                                                    |
| <b>TrackZoom</b>     | 0E | Teste si la position donnée du curseur au moment où il est relâché est celle de la case zoom.         |
| <b>UDivide</b>       | 0B | Calcule le quotient avec signe et le reste de la division.                                            |
| <b>UnionRect</b>     | 04 | Calcule l'union de 2 rectangles.                                                                      |
| <b>UnionRgn</b>      | 04 | Calcule l'union de 2 régions.                                                                         |
| <b>UnPackBytes</b>   | 03 | Décompacte les octets compactés par PackBytes.                                                        |
| <b>UnloadSegment</b> | 11 | Libère la mémoire occupée par le segment donné.                                                       |
| <b>UnlockSeg</b>     | 11 | Déverrouille le segment donné.                                                                        |
| <b>UserShutDown</b>  | 11 | Ferme l'application n° ID.                                                                            |
| <b>ValidRect</b>     | 0E | Enlève le rectangle donné de la région de remise en état.                                             |
| <b>ValidRgn</b>      | 0E | Enlève la région donnée de la région de remise en état.                                               |
| <b>WaitMouseUp</b>   | 06 | Permet de reconnaître un double-click.                                                                |
| <b>WindBootInit</b>  | 0E | Initialisation du Window Manager pour gérer les fenêtres.                                             |
| <b>WNewRes</b>       | 0E | Prend en compte le nouveau mode de résolution.                                                        |
| <b>WriteBlock</b>    | 0C | Transmet un bloc au port de sortie.                                                                   |
| <b>WriteBPARAM</b>   | 03 | Ecrit dans la MEV, alimentée par pile, un paramètre donné.                                            |
| <b>WriteBRAM</b>     | 03 | Ecrit dans la MEV, alimentée par pile, les données en MEV.                                            |
| <b>WriteChar</b>     | 0C | Transmet un caractère au port de sortie.                                                              |
| <b>WriteCString</b>  | 0C | Transmet une chaîne de type C au port de sortie.                                                      |
| <b>WriteLine</b>     | 0C | Transmet une ligne de caractères en sortie.                                                           |
| <b>WriteRAMBlock</b> | 08 | Ecrit des données musicales dans la MEV musicale.                                                     |
| <b>WindReset</b>     | 0E | Mise à zéro de l'outil Window Manager.                                                                |
| <b>WindShutDown</b>  | 0E | Fin d'utilisation du Window Manager et libération de l'espace mémoire correspondant.                  |
| <b>WindStartup</b>   | 0E | Mise en service du Window Manager pour l'ID donné.                                                    |
| <b>WindStatus</b>    | 0E | Renvoie l'état d'activité du Window Manager.                                                          |
| <b>WindVersion</b>   | 0E | Renvoie le numéro de version de l'outil Window Manager.                                               |
| <b>WriteTimeHex</b>  | 03 | Met l'horloge à l'heure donnée.                                                                       |
| <b>XorRgn</b>        | 04 | Calcule la différence entre l'union et l'intersection de 2 régions.                                   |
| <b>ZoomWindow</b>    | 0E | Change la taille de la fenêtre entre les 2 extrêmes.                                                  |

LOGICIELS DE DEVELOPPEMENT

*outils*

PLF

## OUTILS

### Memory Manager

Seize bits du numéro d'identification : ID

b<sub>15</sub> b<sub>14</sub> b<sub>13</sub> b<sub>12</sub> Type (\$0 à \$A).  
b<sub>11</sub> b<sub>10</sub> b<sub>9</sub> b<sub>8</sub> Numéro Auxiliaire défini par l'utilisateur (\$0 à \$F).  
b<sub>7</sub> b<sub>6</sub> b<sub>5</sub> b<sub>4</sub> b<sub>3</sub> b<sub>2</sub> b<sub>1</sub> Numéro attribué par le Memory Manager (\$01 à \$FF).

#### Type

Il caractérise un segment de mémoire d'après la catégorie de programme qui a besoin de ce segment :

| Type | Programme                       |
|------|---------------------------------|
| \$0  | Memory Manager                  |
| \$1  | Application                     |
| \$2  | Programme contrôleur            |
| \$3  | ProDOS                          |
| \$4  | Outil du système                |
| \$5  | Accessoire de bureau            |
| \$6  | Bibliothèque de routines-objets |
| \$7  | Chargeur-système                |
| \$8  | Fonction-système                |
| \$9  | Localisateur d'outil            |
| \$A  | Fichier de type SETUP           |

#### Exemples de n° d'ID

|            |                                                                        |
|------------|------------------------------------------------------------------------|
| \$1001     | Segment utilisé par le programme d'application n°1 (chargé le premier) |
| \$3000     | Segment utilisé par ProDOS                                             |
| \$4100     | Segment utilisé par l'outil Miscellaneous                              |
| \$7001/2/3 | Segment utilisé par le chargeur-système                                |

#### Analyse des Handles de segments avec le MANGLER

Cet accessoire de bureau permet d'utiliser les fonctions de gestion du Memory Manager "en direct". Après l'appel du tableau de bord avec Esc-CTRL-PO. et la sélection de Mangler, l'écran se présente ainsi :

Memory Mangler 1.0 by Steven Glass  
May 28, 1986

%



Liste des commandes

Les commandes disponibles sont obtenues en tapant :

%COMMANDS

| Commande au Mangler | Paramètres                 | Signification                                                         |
|---------------------|----------------------------|-----------------------------------------------------------------------|
| LIST                |                            | Liste de tous les Handles (voir format plus bas).                     |
| NEWHANDLE           | Taille ID attribut adresse | Allocation d'un segment avec attribution d'un nouvel Handle.          |
| REALLOCHANDLE       | Taille ID attribut adresse | Reallocation de segment.                                              |
| DISPOSEHANDLE       | Handle                     | Libération du Handle.                                                 |
| DISPOSEALL          | ID                         | Libération de tous les Handles associés à ID.                         |
| PURGEHANDLE         | Handle                     | Vidange du Handle.                                                    |
| PURGEALL            | ID                         | Vidange des Handle associés à ce numéro d'ID.                         |
| GETHANDLESIZE       | Handle                     | Renvoie la taille du segment pointé par ce Handle.                    |
| SETHANDLESIZE       | Taille Handle              | Fixe la taille du segment.                                            |
| FINDHANDLE          | adresse                    | Renvoie le Handle qui pointe sur le segment comprenant cette adresse. |
| FREEMEM             |                            | Renvoie la quantité de MEV libre.                                     |
| MAXBLOCK            |                            | Renvoie la taille du plus grand segment.                              |
| TOTALMEM            |                            | Renvoie la taille totale de MEV.                                      |
| VERIFYHANDLE        | Handle                     | Vérifie le Handle.                                                    |
| COMPACTMEM          |                            | Compacte des segments épars.                                          |
| HLOCK               | Handle                     | Verrouille le segment pointé par ce Handle.                           |
| HLOCKALL            | ID                         | Verrouille tous les segments dont se sert le programme ID.            |
| HUNLOCK             | Handle                     | Déverrouille le segment.                                              |
| HLUNLOCKALL         | ID                         | Tous ceux associés à ID.                                              |
| SETPURGE            | Niveau Handle              | Fixe le niveau de purge.                                              |
| SETPURGEALL         | Niveau ID                  | Fixe le niveau de purge à tous les segments d'une application.        |
| COMMANDS            |                            | Liste des commandes.                                                  |
| RESTOREHANDLE       | Handle                     | Remet ce Handle à sa valeur initiale.                                 |
| BRK                 |                            | Passage en mode Monitor.                                              |

outils

## OUTILS

### PRINT

Listing sur imprimante de tous les Handles.

#### *Format du listing*

N : n° du Handle, HDL : adresse du Handle, ADR : contenu du Handle, ATT : attributs du segment, ID : numéro d'identification du programme, LG : longueur du segment, PREC : adresse du Handle précédent, SUIV : adresse du Handle suivant.

#### *Exemple*

Le ProDOS16 et un programme d'application sont chargés :

| N  | HDL    | ADR    | ATT  | ID   | LG       | PREC   | SUIV   |
|----|--------|--------|------|------|----------|--------|--------|
| 01 | E11700 | 000000 | C000 | 0000 | 00000800 | 000000 | E118E0 |
| 02 | E118E0 | 000800 | C115 | 1001 | 00000400 | E11700 | E117C8 |
| 03 | E117C8 | 009500 | C013 | 3000 | 00000248 | E118E0 | E117DC |
| 04 | E117DC | 009748 | C013 | 3000 | 000027B7 | E117C8 | E1171A |
| 05 | E1171A | 00C000 | C000 | 0000 | 00004800 | E117DC | E118A4 |

.....

### Quick draw

#### *Les variables des fonctions du Quick Draw*

**Z** est l'adresse de début de la zone dite page zéro dont l'outil a besoin dans le banc \$00 pour réaliser ses fonctions. Le programme a dû réserver cette zone et celle des autres outils en appelant la fonction **NewHandle**. Quick Draw demande une page zéro de \$300 octets de longueur.

**SCB** (poids faibles) est la valeur d'un Scan line Control Block ; valeur attribuée à chacune des lignes de l'écran super haute résolution pour fixer la résolution et la palette d'une ligne.

- Dans la fonction QDStartup, cette valeur initialise tous les SCB (\$0080 en mode 640 pixels par ligne, \$0000 en mode 320 pixels par ligne).

**LG** est un nombre d'octets.

- Dans la fonction QDStartup, il s'agit de la largeur maximum d'une ligne graphique (\$00A0 pour toute la largeur de l'écran).

**ID** est le n° d'identification du programme en cours ; ce n° est fourni par MMStartup.

**V** est le n° de version de l'outil en service.



**B** est une valeur booléenne (\$FFFF = actif ou vrai, \$0000 = désactivé ou faux).

**PA** est un pointeur sur une palette ou table de 16 couleurs (\$20 octets).

**N** est un n° d'ordre ; par exemple le n° d'une palette parmi les 16 enregistrées à partir de l'adresse \$E19E00.

**C** est le n° d'ordre d'une couleur (1 parmi 16) dans une palette.

**D** est la valeur d'une couleur dans ses composantes RVB (pds forts OR, pds faibles VB).

**L** est un n° de ligne de 0 à 199 (\$0000 à \$00C7).

**PO** est un pointeur sur un port, c'est-à-dire une structure de données caractéristique de l'environnement graphique courant : (\$AA ou 170 octets de longueur) :

- quelle résolution (SCB:2), où en mémoire (:4), quelle largeur (LG:2) de ligne, quel rectangle-limite (Hm:2,Vm:2,HM:2,VM:2) ou PortInf ou PortLoc.

Les valeurs standards sont :

00 00 00 20 E1 00 A0 00 00 00 00 00 C8 00 40 01 (octet par octet en hexa) ;  
 \$0000,\$00E12000,\$00A0,\$0000,\$0000,\$00C8,\$0140 (valeurs correspondantes) ;

- quelle zone active (H1:2,V1:2,H2:2,V2:2) ou PortRect ;
- le pointeur de ClipRgn (PCR:4) ;
- le pointeur de VisRgn (PVR:4) ;
- le motif de fond en cas d'effacement : 8x8 pixels (\$20 octets, 2 ou 4 pixels par octet) ;
- les coordonnées du crayon (H:2,V:2) ;
- la taille du crayon (h:2,v:2), le mode de tracé (M:2), le motif du crayon (8x8 pixels:\$20 octets), le masque du crayon (64bits=\$08 octets.) ; les valeurs standards sont respectivement \$0001, \$0001, \$0000, 20x\$FF, 8x\$FF ;
- le niveau de visibilité du crayon (:2) ;
- le pointeur de jeu ou FONT de caractères (PF:4) ;
- la présentation des caractères : FontID:4, FontFlags:2, TxSize:2, TxFace:2, TxMode:2, SpExtra:4, ChExtra:4 ;
- la couleur de premier plan : 2, la couleur de fond : 2 ;
- le champ de sauvegarde d'image : 4 ;
- le champ de sauvegarde de région : 4 ;
- le champ de sauvegarde de polygone : 4 ;
- le pointeur sur le GrafProcs : 4 ;
- la valeur de rotation d'arc : 2 ;
- le champ réservé à l'utilisateur : 4 ;
- le champ réservé au système : 4.

## OUTILS

**HPO** est un handle contenant un pointeur de port graphique.

**HRG** est un handle contenant un pointeur de région.

**RCT** est un pointeur sur les coordonnées définissant un rectangle.

**PP** est un pointeur sur les coordonnées du crayon.

**PS** est un pointeur sur l'état du crayon ( coordonnées, taille, mode, motif, masque).

**M** est une valeur définissant le mode de tracé (\$0000 = copie, \$0001 = transparence, \$0002 = XOR, \$0003 = effacement).

**PM** est un pointeur de motif de crayon ou de motif de fond.

**PD** est un pointeur de masque de dessin.

**DH, DV** sont des valeurs de déplacements Horizontal et Vertical.

**PGP** est un pointeur sur Grafprocs, les procédures de tracés standards.

**PPT** est un pointeur sur les coordonnées d'un Point.

**PC** est un pointeur sur les données définissant un Curseur à savoir :

- sa hauteur en nombre de lignes (h:2) ;
- sa largeur en nombre de mots par ligne (l:2) ;
- l'image du curseur ligne par ligne (h x l :2) ;
- l'image -masque du curseur (h x l :2) ;
- Y : 2 ;
- X : 2.

**HF** est un handle contenant un pointeur de jeu de caractères.

**PF** est un pointeur sur les informations du jeu courant.

**PFG** est un pointeur sur les données du jeu global (la longueur de ces données est renvoyée par GetFGSize).

**TF** est le code du style de caractères (\$0000 = standard, \$0001 = gras, \$0002 = souligné).

**TM** est le code du mode de transfert du texte.

**PBP** est un pointeur vers le bloc de paramètres de la fonction PaintPixels.



Event Manager

Objectifs de l'Event Manager

- Prendre en charge la détection des mouvements de la souris, de l'état des boutons-poussoirs et des touches du clavier ;
- fournir aussi le temps écoulé depuis le démarrage ;
- enregistrer aussi des événements internes provoqués par le Window Manager et le Control Manager ou encore des événements externes générés par des périphériques.

Ces événements sont rangés en file d'attente et sont accessibles par des fonctions de lecture de la file d'attente.

Variables de l'Event Manager

**Z** est l'adresse de début de la page zéro dont a besoin l'outil, adresse qu'il transmet au Window Manager par DoWindows.

**LQ** est la longueur maximale de la file d'attente ou queue, en nombre d'événements :

- en mettant LQ=0, la valeur par défaut est enregistrée, c'est-à-dire : 20 ;
- la valeur maximale est 3639.

**Xm, XM, Ym, YM** sont les valeurs limites des coordonnées de la souris (Clamp).

**ID** est le numéro de l'application ou du programme qui utilise EM.

**ME** est le masque d'événement qui permet de sélectionner certains types d'événements.

**PEV** est le pointeur d'une zone d'enregistrement, appelée Event Record, où figureront les données de l'événement en attente.

**PM** est le pointeur sur la zone de réception des valeurs des coordonnées locales de la souris dans la fenêtre active.

**N** est le n° du bouton ayant été stimulé (0 ou 1).

**CT** est un compteur de 1/60ème de seconde.

**TE** est le type d'événement que l'on doit fournir à la fonction d'enregistrement PostEvent.

**AE** est l'information auxiliaire de l'événement posté.

⊖ : valeur booléenne \$FFFF  
 \$0000

LOGICIELS DE DEVELOPPEMENT

## OUTILS

MS est le masque de stop qui sélectionne les types d'événements à ne pas enlever lors de la fonction FlushEvents.

### Event Record

= variable pointée par POFV:4 par le 1<sup>er</sup> évènement.

Chaque événement est caractérisé par les paramètres suivants regroupés en **Event Record** de 16 octets.

#### - Le type ou what (2 octets)

- 0 pas d'événement ;
- 1 bouton de la souris enfoncé ;
- 2 bouton de la souris relâché ;
- 3 touche enfoncée (sauf les touches spéciales ou modificatrices) ;
- 4 indéterminé ;
- 5 auto-key (touche maintenue enfoncée au moins pendant un certain temps qui dépend du paramètre 'repeat speed' réglable par le Control Panel) ;
- 6 update (contenu d'une fenêtre à dessiner ou à redessiner par suite d'un rangement sur le bureau électronique) ;
- 7 indéterminé ;
- 8 fenêtre activée ou désactivée ;
- 9 switch (généralisé par le Control Manager si le bouton de la souris a été enfoncé alors qu'elle se trouvait dans le Switch Control) ;
- 10 accessoire de bureau (si une touche spéciale a été enfoncée pour appeler un accessoire de bureau classique) ;
- 11 contrôleur de périphérique ;
- 12 -14 définis par l'application.

- Le **type auxiliaire**, ou **Event message**, qui complète les informations précédentes suivant leur type :

| Type                  | Event Message (4 octets).                               |
|-----------------------|---------------------------------------------------------|
| Touche enfoncée       | code ASCII du caractère dans l'octet des poids faibles. |
| Auto-Key              | code ASCII du caractère dans l'octet des poids faibles. |
| Activée ou non        | pointeur sur la fenêtre.                                |
| Update                | pointeur sur la fenêtre.                                |
| Bouton enfoncé        | n° du bouton (0 ou 1) dans l'octet des poids faibles.   |
| Bouton relâché        | n° du bouton.                                           |
| Contrôleur de périph. | défini par le contrôleur.                               |
| Application           | défini par l'application.                               |
| Switch                | indéterminé.                                            |
| Accessoire Bureau     | indéterminé.                                            |

- Le **moment** où il s'est produit ou **when** (4 octets).

Nombre de tops depuis le démarrage.

- L'**endroit** où il s'est produit ou **where** (4 octets).

Les coordonnées globales de la souris au moment où l'événement s'est produit.



- L'état des touches spéciales et des activations ou **modificateurs** (2 octets).

Les bits suivants sont à 1 si la touche correspondante est enfoncée :

|            |                        |
|------------|------------------------|
| <i>Bit</i> | <i>Touches</i>         |
| 13         | clavier numérique      |
| 12         | Ctrl                   |
| 11         | option (ou Funct)      |
| 10         | blocage des majuscules |
| 09         | Shift                  |
| 08         | pomme                  |

Les bits suivants sont à 1 si le bouton correspondant est relâché :

|            |                      |
|------------|----------------------|
| <i>Bit</i> | <i>Bouton-souris</i> |
| 07         | état du bouton 0     |
| 06         | état du bouton 1     |

Les bits suivants caractérisent l'activation des fenêtres :

|            |                                                                                                              |
|------------|--------------------------------------------------------------------------------------------------------------|
| <i>Bit</i> | <i>Etat d'activation</i>                                                                                     |
| 1          | changement de fenêtre active entre fenêtre d'application et fenêtre du système (ex fen alerte) = <i>flag</i> |
| 0          | activée (1), désactivée (0)                                                                                  |

Les bits du **masque** d'événement sont attribués de la manière suivante :

|    |                            |
|----|----------------------------|
| 15 | définis par l'application  |
| 12 |                            |
| 11 | contrôleur de périphérique |
| 10 | accessoire de bureau       |
| 9  | switch                     |
| 8  | activation de fenêtre      |
| 7  | non utilisé                |
| 6  | update                     |
| 5  | auto-key                   |
| 4  | non utilisé                |
| 3  | touche enfoncée            |
| 2  | bouton relâché             |
| 1  | bouton enfoncé             |
| 0  | <i>event nul</i>           |

*dans le champs task mask*

*) de la sous*

Le bit doit prendre la valeur 0 pour neutraliser le type d'événement correspondant.

Utilisation de l'Event Manager

*empilement de l'ache avant d'appeler EM Startup.*

|       |          |                                                            |
|-------|----------|------------------------------------------------------------|
| Début | PEA 0B00 | page Zéro au-dessus de celle de QuickDraw (\$0800+\$0300). |
|       | PEA 0014 | LG max de la file d'attente.                               |
|       | PEA 0000 | minimum horizontal du curseur.                             |
|       | PEA 0140 | Max X=640 colonnes.                                        |
|       | PEA 0000 | Y minimum du curseur.                                      |
|       | PEA 00C8 | Max Y=200 lignes.                                          |
|       | LDA ID   |                                                            |

## OUTILS

|       |             |                                                               |
|-------|-------------|---------------------------------------------------------------|
|       | PHA         |                                                               |
|       | EMStartup   | Demande d'utilisation par l'app.                              |
|       | BCC Suite   |                                                               |
|       | JMP Erreur  |                                                               |
| Suite | _ShowCursor | le curseur va se déplacer en restant dans les limites fixées. |

Après cette phase initiale, deux modes de gestion des événements sont envisageables, soit en utilisant `GetNextEvent`, soit, plus simplement, en travaillant avec le `Task Master`, une fonction disponible dans le `Window Manager` qui saura dispatcher les actions à effectuer suivant le type d'événements de la file.

### *Exemple du `_GetNextEvent`*

|             |                                                  |                                                      |
|-------------|--------------------------------------------------|------------------------------------------------------|
|             | PHA                                              | le résultat est un octet 1 si événement, 0 si aucun. |
|             | PEA %0000111101101110                            | le Masque d'Événements.                              |
|             | PushPtr Event                                    | adresse de l'Event Record.                           |
|             | _GetNextEvent                                    | saisie de l'événement suivant.                       |
|             | PLA                                              | a-t-il eu lieu?                                      |
|             | BEQ Boucle                                       | non, alors reboucler.                                |
|             | Dispatching sur les diverses actions à exécuter. |                                                      |
|             | END                                              |                                                      |
|             | DATA                                             |                                                      |
| EventRecord | Anop                                             |                                                      |
| what        | DS 2                                             |                                                      |
| message     | DS 4                                             |                                                      |
| when        | DS 4                                             |                                                      |
| where       | DS 4                                             |                                                      |
| Modifiers   | DS 2                                             |                                                      |

### **Utilisation du Menu-Manager**

Avec cet outil nous disposons principalement de fonctions d'affichage de menus déroulants ; l'usage de la couleur permet de réaliser un bureau électronique très attrayant. Ce "bureau électronique" a pour objectif de laisser à l'utilisateur toute liberté d'action : il choisira à tout moment l'opération à exécuter dans un des menus déroulants proposés dans la barre des menus .

#### *Variables des fonctions du Menu Manager*

- Z** est l'adresse du début de la page zéro dont il a besoin.
- ID** est le n° d'identification de l'application qui s'en sert.



- PER** est le pointeur qui conduit à l'Event Record contenant le caractère à tester.
- PBM** est un pointeur qui conduit à une barre de menu.
- ADR** est l'adresse d'un sous-programme de rafraîchissement de la partie de l'écran cachée par le menu ; la fonction MenuRefresh ne s'applique que si le Window Manager n'est pas en service.
- PDM** est un pointeur sur la chaîne de caractères définissant le menu.
- ILM** est la position d'insertion dans la liste des menus (0 en tête de liste).
- ILI** est la position d'insertion dans la liste des items d'un menu.
- IDM** est le n° d'identification d'un menu.
- IDI** est le n° d'identification d'un item de menu.
- PDI** est le pointeur sur la chaîne de caractères définissant un item de menu.
- Mh** est la hauteur de la barre-système.
- NI** est le nombre d'items d'un menu.

*Couleurs des menus*

- CN** est la couleur normale de la barre : bits7-4 : fond ; bits3-0 : texte.
- CI** est la couleur d'un item sélectionné ou couleur inverse : bits7-4 : fond ; bits3-0 : texte.
- CC** est la couleur du contour : bits 7-4.
- CM** contient les couleurs du menu courant : bits23-16 : contour ; bits15-8 : inverse.

*Titres des menus*

- XT** est la position en nombre de pixels du premier titre à partir du bord gauche.
- LT** est la largeur du titre du menu en nombre de pixels.
- PT** est un pointeur vers la chaîne de caractères d'un titre de menu ou d'item.

*Présentation des menus*

- NE** est le nouvel état désiré pour un menu.

## OUTILS

- MM** est le masque à utiliser pour la présentation du menu.
- SI** est un contrôle du soulignement et du XOR.
- SS** est l'état souligné ou non d'un item.
- XOR** est l'état d'un item (rehaussé ou non).

### Programme type

|           |                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Démarrage | PushWord ID<br>PEA 0D00<br>_MenuStartup                                                                                                                                                       | le n° de l'application<br>le début de la page zéro<br>allouée à cet outil                                                                                                                                                                                                                                                                                                              |
| Affichage | PushLong £0<br>PushLong £MenuSuper<br>_NewMenu<br>PEA 0<br>_InsertMenu<br>PushLong £0<br>PushLong £MenuPomme<br>_NewMenu<br>PEA 0<br>_InsertMenu<br>PHA<br>_FixMenuBar<br>PLA<br>_DrawMenuBar | déclare le menu par un<br>pointeur vers le Menu Record<br>renvoie un Handle qui est<br>fourni à la fonction d'insertion<br>dans la liste des menus ainsi<br>que la position désirée dans<br>cette liste .Menu suivant vers<br>la gauche<br>le Handle est passé à la<br>fonction d'insertion.<br>avec les textes des menus<br>calcul de la hauteur de la<br>barre<br>dessin de la barre |
| Sélection | LDA TaskData<br>AND £\$00FF<br>ASL A<br>TAX<br>LDA Table,X<br>PHA<br>RTS                                                                                                                      | quel n° item a été choisi<br>l'octet poids faibles<br>:*2<br>index dans la Table<br>adresse de la routine-1<br>rangée sur la pile<br>saut à la routine                                                                                                                                                                                                                                 |
| Table     | anop<br>dc i'Apropos-1'<br>dc i'ignore-1'<br>dc i'Quit-1'<br>dc i'Son-1'<br>dc i'Graph-1'<br>END                                                                                              | routines d'actions à prendre<br>dans l'ordre des menus et<br>des items                                                                                                                                                                                                                                                                                                                 |

Les menus sont définis comme des chaînes de caractères de type C commençant par > et contenant des caractères spéciaux pour moduler l'affichage des textes :

MenuPomme dc c'>àçXN1',i1'13'

Titre =Pomme Colorée



```

dc c'A propos de cet exempleçN256',i1'13'
dc c'N.B.P.S.I - 1986çN257',i1'13'
MenuFichier dc c'> Fichier çN2',i1'13'
dc c'QuitterçN258',i1'13'
MenuSuper dc c'>Super çN3',i1'13'
dc c'SonoritésçN259',i1'13'
dc c'GraphiquesçN260',i1'13'
FinMenus dc c.'
END

```

Les caractères spéciaux de définition des titres et des items :

|             |                                                                                                                                   |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------|
| >           | précède le titre.                                                                                                                 |
| 13 (Return) | séparateur de ligne d'item.                                                                                                       |
| à           | le logo d'Apple qui doit être précédé de > et suivi de <del>13</del> sans espace. <i>set</i>                                      |
| ç (ou \)    | début des caractères spéciaux (ne peut être remplacé).                                                                            |
| X           | le rehaussement est dessiné en ombre colorée.                                                                                     |
| N           | précède le n° d'ID en décimal, compris entre 256 et 65534 pour les applications et entre 1 et 255 pour les accessoires de bureau. |
| H           | précède le n° d'ID en hexadécimal.                                                                                                |
| *           | précède le nom de la touche équivalente.                                                                                          |
| C           | précède le caractère utilisé comme marqueur d'item.                                                                               |
| B           | met le texte en gras.                                                                                                             |
| I           | met le texte en italique.                                                                                                         |
| U           | souligne le texte.                                                                                                                |
| V           | trace une ligne de séparation sous l'item.                                                                                        |
| D           | item non sélectionnable, affiché estompé.                                                                                         |

Les menus en couleur :

a-p symboles des 16 couleurs de code 0 à 15 à placer comme 1er caractère d'item.

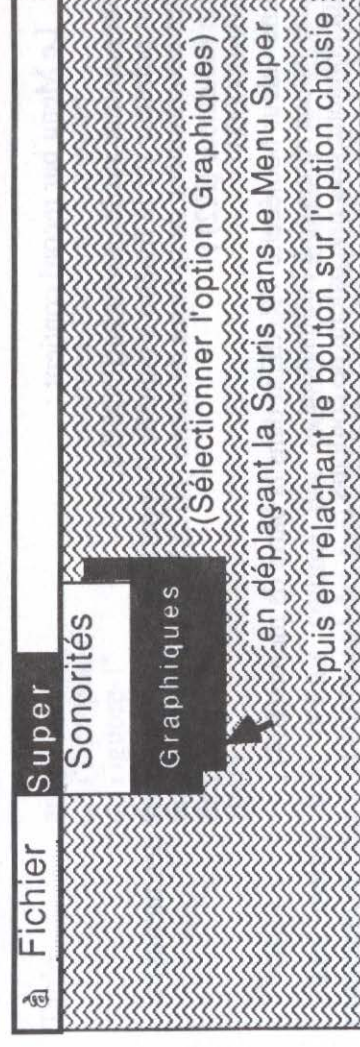
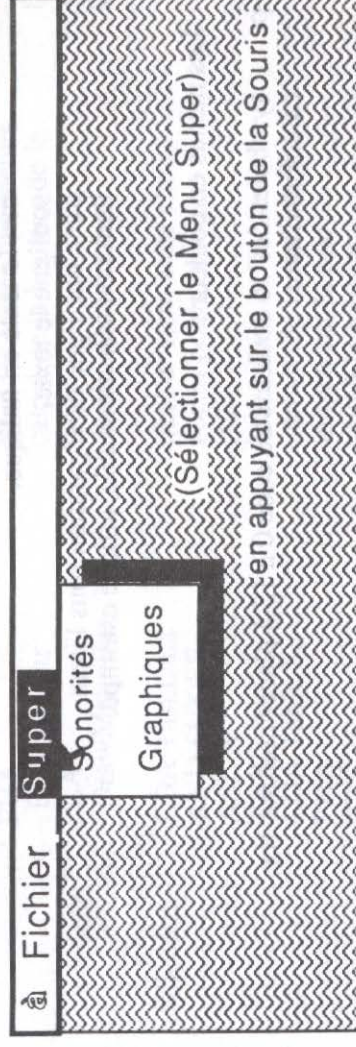
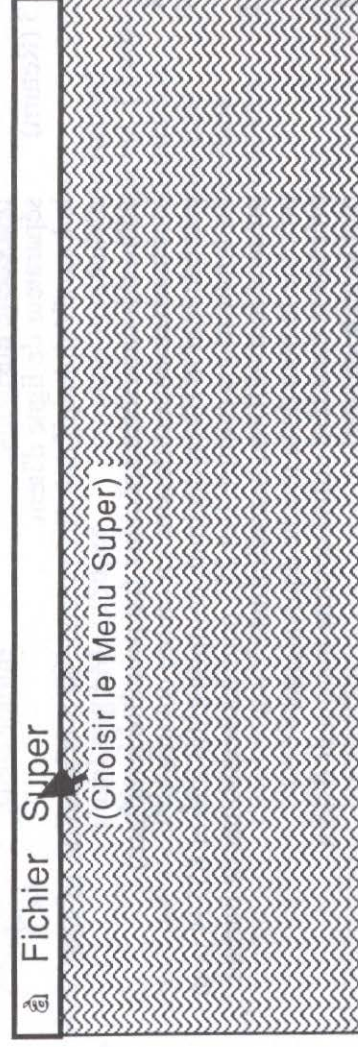
Le Menu bar record contient :

- un pointeur sur le prochain (PNXT:4) ;
- un pointeur sur la fenêtre à laquelle il appartient (\$0 dans la barre des menus) ;
- les coordonnées de la barre de menu (H1:2,V1:2,H2:2,V2:2) ;
- un indicateur (F:2) ;
- \$0A000000 ;
- une valeur booléenne VRAIE si dans la barre ds menus(B:4) ;
- une valeur réservée à l'utilisateur ;
- un pointeur vers une palette de couleurs ;
- la liste des handles de Menus terminée par 0.



## OUTILS

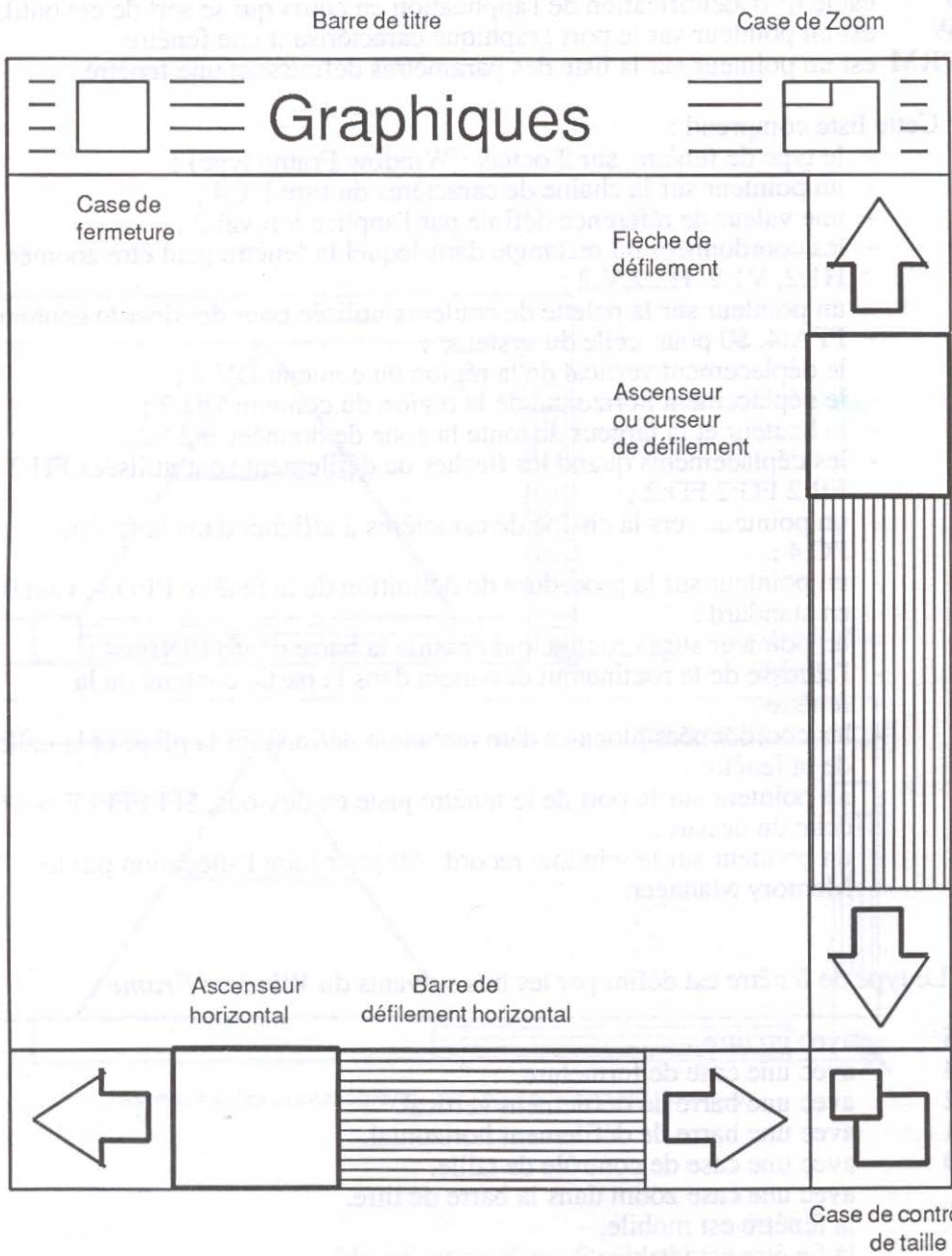
menu  
Pomme



*Les étapes de sélection d'un menu.*



Window manager



Une fenêtre

## OUTILS

### *Variables des fonctions du Window Manager*

**ID** est le n° d'identification de l'application en cours qui se sert de cet outil.

**PW** est un pointeur sur le port graphique caractérisant une fenêtre.

**PPRM** est un pointeur sur la liste des paramètres définissant une fenêtre.

Cette liste comprend :

- le type de fenêtre sur 2 octets :(Window Frame type) ;
- un pointeur sur la chaîne de caractères du titre PT:4 ;
- une valeur de référence définie par l'application val:2 ;
- les coordonnées du rectangle dans lequel la fenêtre peut être zoomée H1:2, V1:2 H2:2,V:2 ;
- un pointeur sur la palette de couleurs utilisée pour dessiner le contour PPA:4, \$0 pour celle du système ;
- le déplacement vertical de la région du contenu DV:2 ;
- le déplacement horizontal de la région du contenu DH:2 ;
- la hauteur et la largeur de toute la zone de données h:2 l:2 ;
- les déplacements quand les flèches de défilement sont utilisées FH:2 FB:2 FG:2 FD:2 ;
- un pointeur vers la chaîne de caractères à afficher dans la fenêtre PC:4 ;
- un pointeur sur la procédure de définition de la fenêtre PPD:4, vaut 0 en standard ;
- un pointeur sur la routine qui dessine la barre d'info PINFo:4 ;
- l'adresse de la routine qui dessinera dans la partie contenu de la fenêtre ;
- les coordonnées globales d'un rectangle définissant la place et la taille de la fenêtre ;
- un pointeur sur le port de la fenêtre juste en dessous, \$FFFFFF pour celle du dessus ;
- un pointeur sur le window record , \$0 pour faire l'allocation par le Memory Manager.

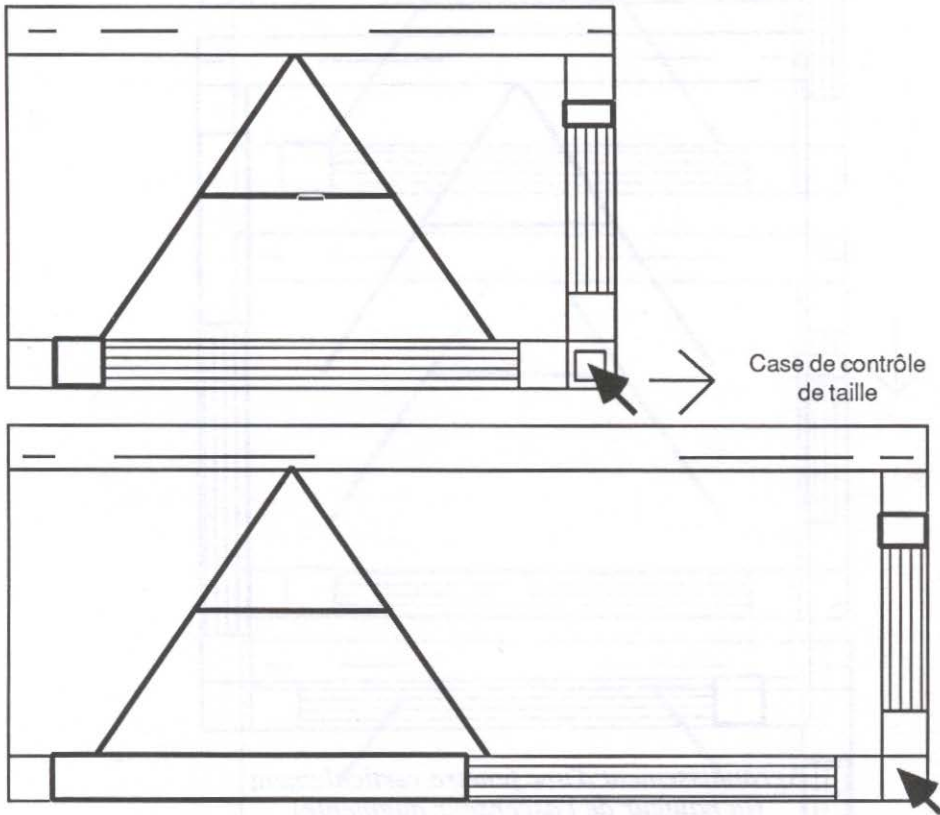
Le type de fenêtre est défini par les bits suivants du *Window Frame* :

- b15** avec un titre.
- b14** avec une case de fermeture.
- b12** avec une barre de défilement vertical.
- b11** avec une barre de défilement horizontal.
- b10** avec une case de contrôle de taille.
- b8** avec une case zoom dans la barre de titre.
- b7** la fenêtre est mobile.
- b5** la fenêtre est visible.
- b4** avec une barre d'informations.

OP est le code d'opération dans la fonction Desktop :

- retrait=0, ajout=1, getDesktop=2, setDesktop=3, getDeskPat=4, SetDeskPat=5
- GetVisDeskTop=6 (retrait de toutes les fenêtres du bureau)



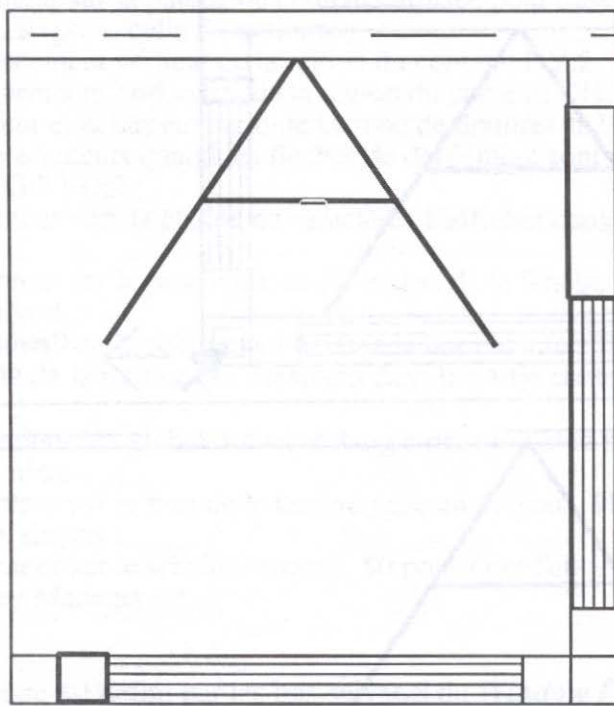
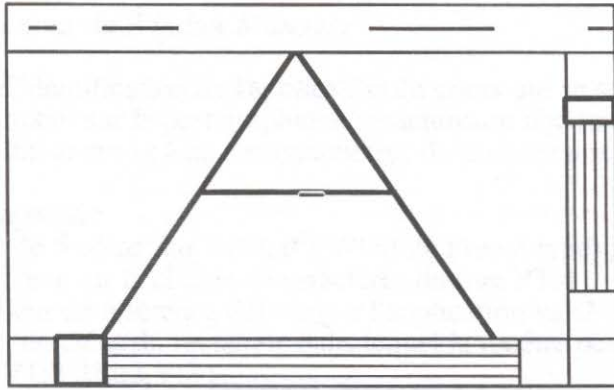


Case de contrôle de taille

Ascenseur horizontal agrandi par l'agrandissement du cadre

*Modification d'une fenêtre*

## OUTILS



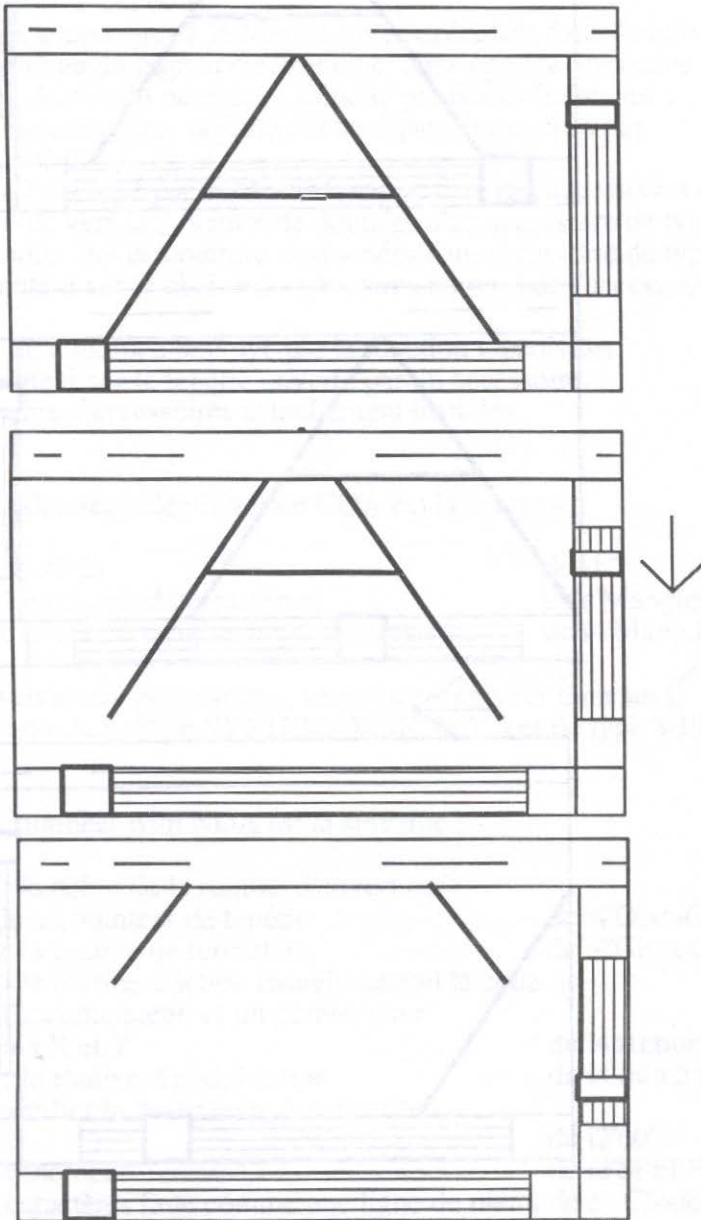
*Agrandissement d'une fenêtre verticalement  
(la hauteur de l'ascenseur augmente)*

La fonction **FindWindow** renvoie la variable OU dont les valeurs sont les suivantes :

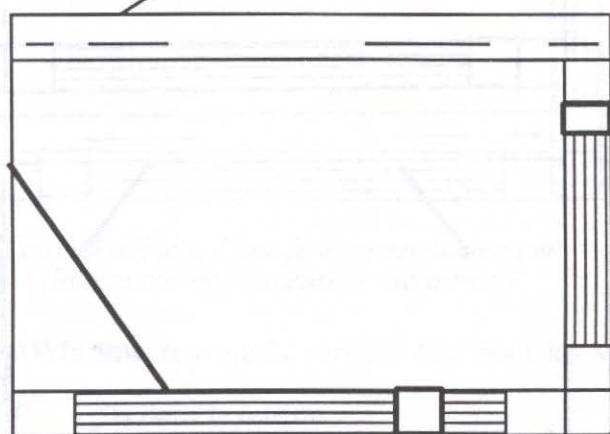
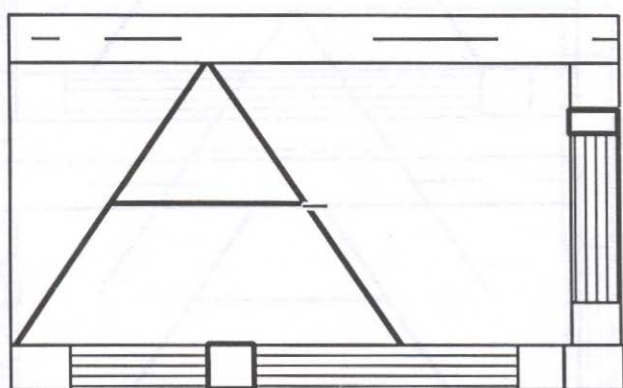
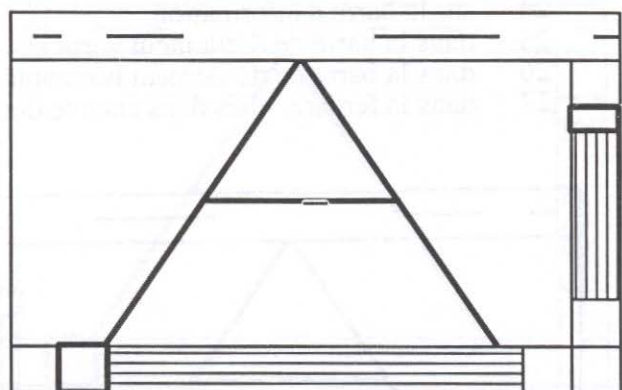
|              |    |                                                  |
|--------------|----|--------------------------------------------------|
| wNoHit       | 0  | pas dans la fenêtre.                             |
| wInDesk      | 1  | sur le bureau électronique.                      |
| wInMenuBar   | 17 | dans la barre des menus.                         |
| wInSysWindow | 18 | dans la fenêtre-système.                         |
| wInContent   | 19 | dans la zone du contenu de la fenêtre.           |
| wInDrag      | 20 | dans la région de déplacement (barre des menus). |
| wInGrow      | 21 | sur la case de contrôle de taille.               |
| wInGoAway    | 22 | sur la case de fermeture.                        |
| wInZoom      | 23 | sur la case zoom.                                |



- wInInfo 24 sur la barre d'information.
- wInVscroll 25 dans la barre de défilement vertical.
- wInHscroll 26 dans la barre de défilement horizontal.
- wInFrame 27 dans la fenêtre, mais dans aucune des zones ci-dessus.



*Défilement du A en fonction du mouvement de l'ascenseur vertical*



*Défilement du A en fonction du mouvement de l'ascenseur horizontal*



Desk manager

*Variables du Desk Accessory Manager*

- V est la version de cet outil.
- NDA qualifie les accessoires à la Macintosh , s'exécutant dans l'environnement graphique du bureau électronique, dans une fenêtre active si elle est sur le dessus du bureau, et sollicité par des événements.
- CDA qualifie les accessoires de bureau classiques déclenchés par PO-CTRL-Esc.
- ID est le n° ID renvoyé par le MenuManager, d'un des accessoires choisis.
- HDA est un handle vers la structure de données d'un accessoire de type NDA.
- HCA est un handle vers la structure de données d'un accessoire de type CDA.
- PC est un pointeur sur la chaîne de caractères du nom de l'accessoire donné par ID.
- N est un n° de référence renvoyé par la fonction OpenNDA.
- PFN est un pointeur sur la fenêtre ouverte par un accessoire.
- NA est le nombre d'accessoires actuellement installés.

La structure des données identifiant un CDA est la suivante :

- longueur du nom (LG:2) dc i1'7'
- nom de l'accessoire (chaîne de caractères) dc c'Mangler'
- un pointeur sur le début du programme de l'accessoire dc i4'Mang.PG'

Ces données, suivies du programme, seront à enregistrer dans un fichier de la disquette-système sous le préfixe SYSTEM/DESK.ACCS et de type \$B9.

La structure de données d'un NDA est la suivante :

- un pointeur sur le début de la routine d'ouverture, laquelle renvoie un pointeur de fenêtre dc i4'OpenClock'
- un pointeur sur la routine de fermeture dc i4'CloseClock'
- un pointeur sur la routine d'action laquelle attend le code d'action dans l'accumulateur, et un pointeur sur l'événement dans X et Y dc i4'ActionClock'
- un pointeur sur la routine d'initialisation dc i4'InitClock'
- un mot pour spécifier la durée entre 2 exécutions (période en s) dc i2'60'
- un masque d'événement dc i2'\$FFFF'
- une chaîne de caractères faite comme une ligne de menu mais avec des espaces avant le nom, et \H\*\* après le nom dc c' Clock\H\*\*'
- nom dc i1'13'

La routine d'ouverture doit être écrite de telle sorte qu'elle renvoie dans la pile le pointeur de la fenêtre : le Desk Accessory Manager empile 4 octets à cet effet juste avant d'appeler cette routine d'ouverture.

## OUTILS

Quand la routine d'action est appelée, un code est présent dans l'accumulateur, il représente l'un des événements suivants :

| <i>Code d'action</i> | <i>Événement</i>                                                                       |
|----------------------|----------------------------------------------------------------------------------------|
| 1                    | bouton enfoncé ou relâché, touche appuyée, ou maintenue.<br>update ou fenêtre activée. |
| 2                    | le moment est venu pour l'exécution (d'après la période<br>fixée).                     |
| 3                    | le curseur est dans la fenêtre.                                                        |
| 4                    | un menu de la barre des menus a été sélectionné.                                       |
| 5                    | l'item Undo ou annuler du menu d'édition a été sélectionné.                            |
| 6                    | l'item Cut ou couper du menu d'édition a été sélectionné.                              |
| 7                    | l'item Copy ou copier du menu d'édition a été sélectionné.                             |
| 8                    | l'item Paste ou coller du menu d'édition a été sélectionné.                            |
| 9                    | l'item Clear ou effacer du menu d'édition a été sélectionné.                           |

Les registres X et Y sont porteurs d'informations, aussi à l'appel de la routine d'action : ils contiennent un pointeur vers l'event record.

La routine d'initialisation est déclenchée par DeskStartup

La structure de données et du programme correspondant à l'accessoire constitueront un fichier à enregistrer sous le prefixe SYSTEM / DESK.ACCS avec le type \$B8.

### *Mise en oeuvre dans l'application*

|                     |                                                  |
|---------------------|--------------------------------------------------|
| <u>DeskStartup</u>  | met le DA Manager en service.                    |
| <u>PEA \$0001</u>   | l'ID n°1 est affecté au premier accessoire.      |
| <u>FixAppleMenu</u> | met la liste des Accessoires dans le menu Pomme. |

Grâce à TaskMaster, les événements concernant les accessoires sont transmis automatiquement au programme de l'accessoire qui les traitera suivant leur nature par l'une des 4 routines (OPEN, CLOSE, ACTION, INIT) qu'il contient.

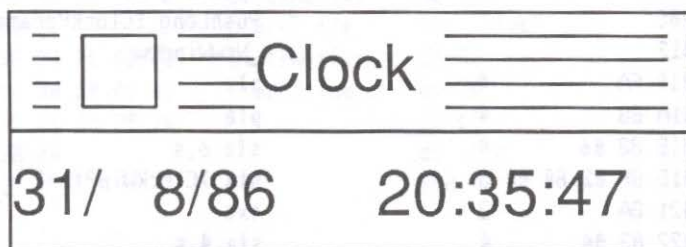
DeskShutDown referme le DA Manager

*Voir l'exemple Clock page 175.*



# EXEMPLE

## MONTRE EXTRA-PLATE DE VOTRE BUREAU



*Image clock*

### Listing du programme-source en macro-assembleur

LISTING D'ASSEMBLAGE DU PROGRAMME-SOURCE  
RESULTANT DE LA COMMANDE : assemble clock.src

\*\*\*\*\*

ORCA/M ASM65816 V4.1 Phase 3 D6

11 Sep 86 17:23

```
0001 0000 list on
0002 002000 0000 absaddr on
0003 002000 0000 instime on
0004 002000 0000 GEN off
0005 002000 0000 SYMBOL on
0006 002000 0000 KEEP clock
0007 002000 0000 mcopy Clock.macros
0008 002000 0000 ClockDA START
0009 002000 0000 END

0010 002000 0000 IDSection START
0011 002000 0000 00 80 00 00 dc i4'OpenClock'
0012 002004 0004 00 80 00 00 dc i4'CloseClock'
0013 002008 0008 00 80 00 00 dc i4'ActionClock'
0014 00200C 000C 00 80 00 00 dc i4'InitClock'
0015 002010 0010 3C 00 dc i2'60'
```

# MONTRE EXTRA-PLATE DE VOTRE BUREAU

```

0016 002012 0012 FF FF dc i2'$FFFF'
0017 002014 0014 20 43 6C 6F dc c' ClockçH**'
0018 00201E 001E 0D dc i1'13'
0019 00201F 001F END

0020 00201F 0000 OpenClock START
0021 00201F 0000 using ClockData
0022 00201F 0000 AF 00 00 00 5 lda >ClockActive
0023 002023 0004 D0 3A 2' bne Ignore
0024 002025 0006 PushLong £0
0025 00202B 000C PushLong £ClockParams
0026 002031 0012 _NewWindow
0027 002038 0019 FA 4 plx
0028 002039 001A 68 4 pla
0029 00203A 001B 83 06 4 sta 6,s
0030 00203C 001D 8F 02 00 00 5 sta >ClockWinPtr+2
0031 002040 0021 8A 2 txa
0032 002041 0022 83 04 4 sta 4,s
0033 002043 0024 8F 00 00 00 5 sta >ClockWinPtr
0034 002047 0028 AF 02 00 00 5 lda >ClockWinPtr+2
0035 00204B 002C 48 3 pha
0036 00204C 002D AF 00 00 00 5 lda >ClockWinPtr
0037 002050 0031 48 3 pha
0038 002051 0032 _SetSysWindow
0039 002058 0039 A9 00 00 2 lda £$0000
0040 00205B 003C 8F 00 00 00 5 sta >ClockActive
0041 00205F 0040 6B 6 Ignore rti

0042 002060 0041 ClockParams anop
0043 002060 0041 A0 C0 dc i2'%1100000010100000'
0044 002062 0043 00 00 00 00 dc i4'ClockTitle'
0045 002066 0047 00 00 00 00 dc i4'0'
0046 00206A 004B 00 00 00 00 dc i2'0,0,0,0'
0047 002072 0053 00 00 00 00 dc i4'0'
0048 002076 0057 00 00 dc i2'0'
0049 002078 0059 00 00 dc i2'0'
0050 00207A 005B 00 00 dc i2'0'
0051 00207C 005D 00 00 dc i2'0'
0052 00207E 005F 00 00 dc i2'0'
0053 002080 0061 00 00 dc i2'0'
0054 002082 0063 00 00 dc i2'0'
0055 002084 0065 00 00 dc i2'0'
0056 002086 0067 00 00 dc i2'0'
0057 002088 0069 00 00 dc i2'0'
0058 00208A 006B 00 00 00 00 dc i4'0'
0059 00208E 006F 00 00 00 00 dc i4'0'
0060 002092 0073 00 00 00 00 dc i4'0'
0061 002096 0077 00 00 00 00 dc i4'0'
0062 00209A 007B 32 00 32 00 dc i'50,50,62,200'
0063 0020A2 0083 FF FF FF FF dc i4'$FFFFFFFF'

```



# MONTRE EXTRA-PLATE DE VOTRE BUREAU

```
0064 0020A6 0087 00 00 00 00 dc i4'0'
0065 0020AA 008B ;
0066 0020AA 008B
0067 0020AA 008B END
```

## Local Symbols

```
000041 ClockParams 000040 Ignore
```

```
0068 0020AA 0000 ClockData DATA
0069 0020AA 0000 00 00 ClockActive dc i'0'
0070 0020AC 0002 ClockTitle str 'Clock'
0071 0020B2 0008 00 00 00 00 ClockWinPtr ds 4
0072 0020B6 000C 00 00 00 00 TimeString ds 20
0073 0020CA 0020 20 20 20 20 dc c'
0074 0020D2 0028 00 dc i1'0'
0075 0020D3 0029 END
```

## Local Symbols

```
000000 ClockActive 000002 ClockTitle 000008 ClockWinPtr 00000C TimeString
000003 sys2
```

```
0076 0020D3 0000 CloseClock Start
0077 0020D3 0000 using ClockData
0078 0020D3 0000 AF 00 00 00 5 lda >ClockActive
0079 0020D7 0004 F0 18 2' beq Ignore
0080 0020D9 0006 PushLong >ClockWinPtr
0081 0020E3 0010 _CloseWindow
0082 0020EA 0017 A9 00 00 2 lda f0
0083 0020ED 001A 8F 00 00 00 5 sta >ClockActive
0084 0020F1 001E 6B 6 rti
0085 0020F2 001F END
```

## Local Symbols

```
00001E Ignore
```

```
0086 0020F2 0000 ActionClock START
0087 0020F2 0000 using ClockData
0088 0020F2 0000 5A 3 phy
0089 0020F3 0001 DA 3 phx
0090 0020F4 0002 0A 2 asl a
0091 0020F5 0003 AA 2 tax
0092 0020F6 0004 FC 0A 00 6 jsr (ActionTable,x)
0093 0020F9 0007 68 4 pla
0094 0020FA 0008 68 4 pla
0095 0020FB 0009 6B 6 rti
0096 0020FC 000A ActionTable anop
0097 0020FC 000A 1E 00 dc i'ignore'
0098 0020FE 000C 4A 00 dc i'ActionEvent'
0099 002100 000E 1F 00 dc i'ActionRun'
0100 002102 0010 1E 00 dc i'ActionCursor'
```

# MONTRE EXTRA-PLATE DE VOTRE BUREAU

|      |        |      |    |    |    |              |                      |
|------|--------|------|----|----|----|--------------|----------------------|
| 0101 | 002104 | 0012 | 1E | 00 |    |              | dc i'ActionMenu'     |
| 0102 | 002106 | 0014 | 1E | 00 |    |              | dc i'ActionUndo'     |
| 0103 | 002108 | 0016 | 1E | 00 |    |              | dc i'ActionCut'      |
| 0104 | 00210A | 0018 | 1E | 00 |    |              | dc i'ActionCopy'     |
| 0105 | 00210C | 001A | 1E | 00 |    |              | dc i'ActionPaste'    |
| 0106 | 00210E | 001C | 1E | 00 |    |              | dc i'ActionClear'    |
| 0107 | 002110 | 001E |    |    |    | ActionCursor | anop                 |
| 0108 | 002110 | 001E |    |    |    | ActionMenu   | anop                 |
| 0109 | 002110 | 001E |    |    |    | ActionUndo   | anop                 |
| 0110 | 002110 | 001E |    |    |    | ActionCut    | anop                 |
| 0111 | 002110 | 001E |    |    |    | ActionCopy   | anop                 |
| 0112 | 002110 | 001E |    |    |    | ActionPaste  | anop                 |
| 0113 | 002110 | 001E |    |    |    | ActionClear  | anop                 |
| 0114 | 002110 | 001E |    |    |    | Ignore       | anop                 |
| 0115 | 002110 | 001E | 60 |    | 6  |              | rts                  |
| 0116 | 002111 | 001F |    |    |    | ActionRun    | anop                 |
| 0117 | 002111 | 001F | 8B |    | 3  |              | phb                  |
| 0118 | 002112 | 0020 | 4B |    | 3  |              | phk                  |
| 0119 | 002113 | 0021 | AB |    | 4  |              | plb                  |
| 0120 | 002114 | 0022 |    |    |    |              | PushLong #0          |
| 0121 | 00211A | 0028 |    |    |    |              | _GetPort             |
| 0122 | 002121 | 002F |    |    |    |              | PushLong ClockWinPtr |
| 0123 | 002129 | 0037 |    |    |    |              | _SetPort             |
| 0124 | 002130 | 003E | 20 | 00 | 80 | 6            | jsr DrawTime         |
| 0125 | 002133 | 0041 |    |    |    |              | _SetPort             |
| 0126 | 00213A | 0048 | AB |    | 4  |              | plb                  |
| 0127 | 00213B | 0049 | 60 |    | 6  |              | rts                  |
| 0128 | 00213C | 004A |    |    |    | ActionEvent  | anop                 |
| 0129 | 00213C | 004A |    |    |    | origd        | equ 1                |
| 0130 | 00213C | 004A |    |    |    | rtsaddr      | equ 3                |
| 0131 | 00213C | 004A |    |    |    | evtptr       | equ 5                |
| 0132 | 00213C | 004A | 0B |    | 4  |              | phd                  |
| 0133 | 00213D | 004B | 3B |    | 2  |              | tsc                  |
| 0134 | 00213E | 004C | 5B |    | 2  |              | tcd                  |
| 0135 | 00213F | 004D | A7 | 05 | 6  |              | lda "evtptr\$        |
| 0136 | 002141 | 004F | 0A |    | 2  |              | asl a                |
| 0137 | 002142 | 0050 | AA |    | 2  |              | tax                  |
| 0138 | 002143 | 0051 | FC | 56 | 00 | 6            | jsr (EventTable,x)   |
| 0139 | 002146 | 0054 | 2B |    | 5  |              | pld                  |
| 0140 | 002147 | 0055 | 60 |    | 6  |              | rts                  |
| 0141 | 002148 | 0056 |    |    |    | EventTable   | anop                 |
| 0142 | 002148 | 0056 | 1E | 00 |    |              | dc i'ignore'         |
| 0143 | 00214A | 0058 | 1E | 00 |    |              | dc i'ignore'         |
| 0144 | 00214C | 005A | 1E | 00 |    |              | dc i'ignore'         |
| 0145 | 00214E | 005C | 00 | 80 |    |              | dc i'keyclock'       |
| 0146 | 002150 | 005E | 1E | 00 |    |              | dc i'ignore'         |
| 0147 | 002152 | 0060 | 00 | 80 |    |              | dc i'keyclock'       |
| 0148 | 002154 | 0062 | 00 | 80 |    |              | dc i'updateclock'    |



# MONTRE EXTRA-PLATE DE VOTRE BUREAU

```

0149 002156 0064 1E 00 dc i'ignore'
0150 002158 0066 1E 00 dc i'ignore'
0151 00215A 0068 1E 00 dc i'ignore'
0152 00215C 006A END

```

## Local Symbols

```

00001E ActionClear 00001E ActionCopy 00001E ActionCursor
00001E ActionCut 00004A ActionEvent 00001E ActionMenu 00001E ActionPaste
00001F ActionRun 00000A ActionTable 00001E ActionUndo 000056 EventTable
00001E Ignore 000005 evtptr 000001 origd 000003 rtsaddr

```

```

0153 00215C 0000 KeyClock START
0154 00215C 0000 60 6 rts
0155 00215D 0001 END

```

```

0156 00215D 0000 UpdateClock START
0157 00215D 0000 using ClockData
0158 00215D 0000 8B 3 phb
0159 00215E 0001 4B 3 phk
0160 00215F 0002 AB 4 plb
0161 002160 0003 PushLong ClockWinPtr
0162 002168 000B _BeginUpdate
0163 00216F 0012 20 00 80 6 jsr DrawTime
0164 002172 0015 PushLong ClockWinPtr
0165 00217A 001D _EndUpdate
0166 002181 0024 AB 4 plb
0167 002182 0025 60 6 rts
0168 002183 0026 END

```

```

0169 002183 0000 DrawTime Start
0170 002183 0000 using ClockData
0171 002183 0000 PushLong £TimeString
0172 002189 0006 _ReadAsciiTime
0173 002190 000D E2 20 3 sep £%00100000
0174 002192 000F longa off
0175 002192 000F A2 13 00 2 ldx £19
0176 002195 0012 BD 00 80 4* loop lda TimeString,x
0177 002198 0015 29 7F 2 and £$7F
0178 00219A 0017 9D 00 80 5 sta TimeString,x
0179 00219D 001A CA 2 dex
0180 00219E 001B 10 F5 2 bpl Loop
0181 0021A0 001D C2 20 3 rep £%00100000
0182 0021A2 001F longa on
0183 0021A2 001F PushWord £7
0184 0021A5 0022 PushWord £10
0185 0021A8 0025 _MoveTo
0186 0021AF 002C PushLong £TimeString
0187 0021B5 0032 _DrawCString
0188 0021BC 0039 60 6 rts
0189 0021BD 003A END

```

# MONTRE EXTRA-PLATE DE VOTRE BUREAU

Local Symbols  
000012 loop

```
0190 0021BD 0000 InitClock START
0191 0021BD 0000 6B 6 rti
0192 0021BE 0001 END
```

192 source lines  
23 macros expanded  
104 lines generated

## Listing des macros

\*\*\*\*\*

\*Fichier CLOCK.MACROS des seules Macros nécessaires:

\*\*\*\*\*

```
MACRO
&LAB PUSHWORD &WHATTOPUSH
LCLC &CHAR
&CHAR AMID &WHATTOPUSH,1,1
AIF "&CHAR"="E",.IMMEDIATE
&LAB LDA &WHATTOPUSH
PHA
MEXIT
IMMEDIATE
&CHAR AMID &WHATTOPUSH,2,100
&LAB DC I1'$F4'
DC I2'&CHAR'
MEND
```

```
MACRO
&LAB PUSHLONG &WHATTOPUSH
LCLC &CHAR
&CHAR AMID &WHATTOPUSH,1,1
AIF "&CHAR"="E",.IMMEDIATE
&LAB LDA &WHATTOPUSH+2
```



## MONTRE EXTRA-PLATE DE VOTRE BUREAU

```
PHA
LDA &WHATTOPUSH
PHA
MEXIT
.IMMEDIATE
&CHAR AMID &WHATTOPUSH,2,100
&LAB DC 11'#$F4'
DC 12'(&CHAR)0-16'
DC 11'#$F4'
DC 12'&CHAR'
MEND
```

```
MACRO
&LAB STR &STUFF
&LAB DC 11'L:sys&SYSQNT'
sys&SYSQNT DC C"&STUFF"
MEND
```

```
MACRO
&lab _ReadAsciiTime
&lab ldx #0F03
jsl #E10000
MEND
```

```
MACRO
&lab _SetPort
&lab ldx #4+256*27
jsl #E10000
MEND
```

```
MACRO
&lab _GetPort.
&lab ldx #4+256*28
jsl #E10000
MEND
```

```
MACRO
&lab _MoveTo
&lab ldx #4+256*58
jsl #E10000
MEND
```

```
MACRO
&lab _DrawCString
&lab ldx #4+256*166
jsl #E10000
MEND
```

## MONTRE EXTRA-PLATE DE VOTRE BUREAU

```
MACRO
&lab _NewWindow
&lab idx f14+256*9
jsl $E10000
MEND
```

```
MACRO
&lab _CloseWindow
&lab idx f14+256*11
jsl $E10000
MEND
```

```
MACRO
&lab _BeginUpdate
&lab idx f14+256*30
jsl $E10000
MEND
```

```
MACRO
&lab _EndUpdate
&lab idx f14+256*31
jsl $E10000
MEND
```

```
MACRO
&lab _SetSysWindow
&lab idx f14+256*75
jsl $E10000
MEND
```

### Listing du BUILD d'exécution des commandes d'assemblage

```

```

```
*Fichier BUILD d'élaboration automatique, à partir du pg-source,
*du segment chargeable CLOCK de type $B8:
```

```

```

```
assemble clock.src
run clock.link
filetype clock $B8
enable dnwr clock
```



## MONTRE EXTRA-PLATE DE VOTRE BUREAU

Listing du fichier d'exécution des commandes à l'éditeur de liens

\*\*\*\*\*

\*Fichier CLOCK.LINK de commandes au LINKER, l'éditeur de liens.

\*\*\*\*\*

```
Keep clock
segment main $20
link/all clock
```

\*\*\*\*\*

```
*L'ASSEMBLAGE DU PROGRAMME-SOURCE A PRODUIT DEUX MODULES-OBJETS
*clock.root
*clock.A
```

\*\*\*\*\*

```
*LISTING D'EDITION DE LIENS DES MODULES-OBJETS
*RESULTANT DE LA COMMANDE run clock.link
*APPLIQUEE AU FICHIER clock.link ECRIT EN LINKED
```

\*\*\*\*\*

Advanced Linker 4.1 Phase 3 B7.1

```
1 keep clock
2 segment main $20
3 link/all clock
```

0 errors found in source file.

# MONTRE EXTRA-PLATE DE VOTRE BUREAU

Segment: MAIN

```
00000000 00000000 Code: CLOCKDA
00000000 0000001F Code: IDSECTION
0000001F 0000008B Code: OPENCLOCK
000000AA 00000029 Data: CLOCKDATA
000000D3 0000001F Code: CLOSECLOCK
000000F2 0000006A Code: ACTIONCLOCK
0000015C 00000001 Code: KEYCLOCK
0000015D 00000026 Code: UPDATECLOCK
00000183 0000003A Code: DRAWTIME
000001BD 00000001 Code: INITCLOCK
```

Global symbol table:

```
ACTIONCLOC 000000F2 00 CLOCKACTIV 000000AA 01 CLOCKDA 00000000 00
CLOCKDATA 000000AA 01 CLOCKTITLE 000000AC 01 CLOCKWINPT 000000B2 01
CLOSECLOCK 000000D3 00 DRAWTIME 00000183 00 IDSECTION 00000000 00
INITCLOCK 000001BD 00 KEYCLOCK 0000015C 00 OPENCLOCK 0000001F 00
SYS2 000000AD 01 TIMESTRING 000000B6 01 UPDATECLOC 0000015D 00
```

Last segment starts at \$00000000 and is \$000001BE bytes l



# MONTRE EXTRA-PLATE DE VOTRE BUREAU

## Catalogue final

\*\*\*\*\*

\*L'EDITION DE LIENS A PRODUIT UN SEGMENT CHARGEABLE DE TYPE EXE  
\*QU'IL FAUT CONVERTIR EN TYPE \*B8 POUR QU'IL SOIT CHARGEABLE ET  
\*EXECUTABLE SOUS PRODOS 16 EN TANT QU'ACCESSOIRE DE BUREAU.

\*\*\*\*\*

\*REPERTOIRE DES FICHIERS PREPARES (type SRC) ET GENERES :

\*\*\*\*\*

/RAM5/CLOCK/=

| Name         | Type | Blocks | Modified     | Created         | Access | Subtype  |
|--------------|------|--------|--------------|-----------------|--------|----------|
| CLOCK.SRC    | SRC  | 11 11  | SEP 86 17:19 | 11 SEP 86 17:19 | DNBWR  | ASM65816 |
| CLOCK.MACROS | SRC  | 4 8    | JUL 86 8:25  | 11 SEP 86 17:15 | DNBWR  | TEXT     |
| CLOCK.ROOT   | OBJ  | 1 11   | SEP 86 17:23 | 11 SEP 86 17:23 | DNBWR  |          |
| CLOCK.A      | OBJ  | 10 11  | SEP 86 17:23 | 11 SEP 86 17:23 | DNBWR  |          |
| CLOCK.LINK   | SRC  | 1 11   | SEP 86 17:23 | 11 SEP 86 17:23 | DNBWR  | LINKED   |
| CLOCK        | *B8  | 4 11   | SEP 86 17:24 | 11 SEP 86 17:24 | DNBWR  |          |
| BUILD        | SRC  | 1 11   | SEP 86 17:22 | 11 SEP 86 17:22 | DNBWR  | EXEC     |

Blocks Free: 22      Blocks Used: 42      Total Blocks: 64

## Installation dans la disquette-système Pro-DOS 16

\*\*\*\*\*

\*REPERTOIRE ET SOUS-REPERTOIRES D'UNE DISQUETTE-SYSTEME  
\*CONTENANT UNE APPLICATION SOUS ProDOS 16

\*\*\*\*\*

/PD16/=

| Name       | Type | Blocks | Modified     | Created         | Access | Subtype |
|------------|------|--------|--------------|-----------------|--------|---------|
| PRODOS     | SYS  | 41 21  | JUL 86 14:31 | 11 SEP 86 16:00 | DNBWR  |         |
| SYSTEM     | DIR  | 1 12   | SEP 86 1:03  | 17 JUN 86 16:23 | DNBWR  |         |
| LOAD.SYS16 | S16  | 26 15  | JUL 86 10:33 | 11 SEP 86 15:47 | DNBWR  | A=*0640 |

Blocks Free: 1129      Blocks Used: 471      Total Blocks: 1600

# MONTRE EXTRA-PLATE DE VOTRE BUREAU

/PD16/SYSTEM/=

| Name         | Type | Blocks | Modified     | Created         | Access | Subtype |
|--------------|------|--------|--------------|-----------------|--------|---------|
| P8           | SYS  | 31 21  | JUL 86 14:31 | 11 SEP 86 16:09 | DNBWR  |         |
| P16          | #F9  | 44 21  | JUL 86 14:31 | 11 SEP 86 16:08 | DNBWR  |         |
| DESK.ACCS    | DIR  | 1 12   | SEP 86 1:03  | 11 AUG 86       | DNBWR  |         |
| SYSTEM.SETUP | DIR  | 1 11   | SEP 86 16:22 | 11 AUG 86       | DNBWR  |         |
| TOOLS        | DIR  | 1 11   | SEP 86 16:40 | 11 AUG 86       | DNBWR  |         |
| LOADER1      | BIN  | 13 21  | JUL 86 14:31 | 11 SEP 86 16:08 | DNBWR  | A=#D000 |
| LOADER2      | BIN  | 3 21   | JUL 86 14:31 | 11 SEP 86 16:09 | DNBWR  | A=#D000 |

Blocks Free: 1129      Blocks Used: 471      Total Blocks: 1600

/PD16/SYSTEM/SYSTEM.SETUP/=

| Name       | Type | Blocks | Modified     | Created         | Access | Subtype |
|------------|------|--------|--------------|-----------------|--------|---------|
| TOOL.SETUP | STR  | 24 21  | JUL 86 14:31 | 11 SEP 86 16:21 | DNBWR  | A=#0000 |

Blocks Free: 1129      Blocks Used: 471      Total Blocks: 1600

/PD16/SYSTEM/DESK.ACCS/=

| Name       | Type | Blocks | Modified     | Created         | Access | Subtype |
|------------|------|--------|--------------|-----------------|--------|---------|
| MANGLER.DA | #B9  | 21 16  | JUN 86 15:51 | 11 SEP 86 16:26 | DNBWR  |         |
| CLOCK      | #B8  | 4 12   | SEP 86 0:59  | 12 SEP 86 1:03  | DNBWR  |         |

Blocks Free: 1129      Blocks Used: 471      Total Blocks: 1600

/PD16/SYSTEM/TOOLS/=

| Name    | Type | Blocks | Modified     | Created         | Access | Subtype |
|---------|------|--------|--------------|-----------------|--------|---------|
| TOOL015 | #BA  | 24 21  | JUL 86 14:34 | 11 SEP 86 16:37 | DNBWR  |         |
| TOOL014 | #BA  | 38 21  | JUL 86 14:32 | 11 SEP 86 16:35 | DNBWR  |         |
| TOOL016 | #BA  | 24 21  | JUL 86 14:33 | 11 SEP 86 16:35 | DNBWR  |         |
| TOOL020 | #BA  | 16 21  | JUL 86 14:33 | 11 SEP 86 16:36 | DNBWR  |         |
| TOOL024 | EXE  | 45 21  | JUL 86 14:33 | 11 SEP 86 16:36 | DNBWR  | A=#0100 |
| TOOL018 | #BA  | 40 21  | JUL 86 14:33 | 11 SEP 86 16:36 | DNBWR  |         |
| TOOL023 | EXE  | 38 21  | JUL 86 14:33 | 11 SEP 86 16:37 | DNBWR  | A=#0100 |
| TOOL021 | #BA  | 25 21  | JUL 86 14:33 | 11 SEP 86 16:37 | DNBWR  |         |
| TOOL022 | #BA  | 11 21  | JUL 86 14:34 | 11 SEP 86 16:39 | DNBWR  |         |

Blocks Free: 1129      Blocks Used: 471      Total Blocks: 1600



# CONSEILS DE LECTURE

Pour maîtriser le système de l'Apple IIGs 65816, et mieux connaître le mode Apple II de votre ordinateur, P.S.I. vous propose une palette d'ouvrages utiles.

## POUR MAITRISER LE SYSTEME DE BASE DE L'APPLE IIGs

A paraître :

- **Assembleur de l'Apple IIGs** - Jean-Pierre Lagrange (*Editions du P.S.I.*)

Une initiation claire et complète à l'assembleur 65816, illustrée par de nombreux programmes-exemples. Ce livre est particulièrement recommandé aux débutants en assembleur.

- **La boîte à outils de l'Apple IIGs** - Jean-Pierre Curcio (*Editions du P.S.I.*)

Cet ouvrage s'adresse à tous ceux qui souhaitent programmer dans l'esprit du GS : ce n'est pas un cours sur l'assembleur ou le C, mais une étude de la boîte à outils intégrée au système. Deux exemples concrets, un MiniPaint et un programme d'interface, illustrent l'utilisation des managers.

## POUR MIEUX CONNAITRE LE MODE APPLE II DE VOTRE ORDINATEUR

- **102 programmes pour Apple II** - Jacques Deconchat (*Editions du P.S.I.*)

Spécialement destinés aux débutants, ces 102 jeux en Basic présentent les instructions Applesoft selon cinq niveaux de difficulté croissante. Chaque programme couvre une page de livre au maximum, et est commenté ligne à ligne.

- **Programmation système de l'Apple II** - Marcel Cottini (*Editions du P.S.I.*)

Pour programmeurs chevronnés sur Apple IIe ou IIc, cet ouvrage présente les microprocesseurs 6502 et 65C02, et de nombreuses astuces inédites de programmation.

- **Introduction à ProDos sur Apple** - Francis Versheure (*Editions du P.S.I.*)  
Initiation au système d'exploitation des Apple IIe et IIc, comprenant notamment la gestion des supports, des catalogues et des fichiers, l'étude d'un système de conversion de DOS à ProDOS, et l'utilisation d'Applesoft sous ProDOS.

- **Système ProDos sur Apple II** - Marcel Cottini (*Editions du P.S.I.*)  
Pour programmeur averti, ce livre présente l'organisation complète de ce système d'exploitation avec de nombreux exemples d'application.



Achévé d'imprimer en octobre 1986  
sur les presses de l'imprimerie Laballery  
58500 Clamecy  
Dépôt légal : octobre 1986

N° d'impression : 609075  
N° d'édition : 86595-368.1  
ISBN : 2-86595-368.8

**Votre avis nous intéresse**

Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques sur le présent ouvrage.

— *Ce livre vous donne-t-il toute satisfaction ?*

.....  
 .....  
 .....

— *Y a-t-il un aspect du problème que vous auriez aimé voir abordé ?*

.....  
 .....  
 .....

Si vous souhaitez des éclaircissements techniques, écrivez-nous, nous ne manquerons pas de vous répondre directement.

**Où avez-vous acheté ce livre ?**

- cadeau                       librairie                       autres  
 exposition                       boutique micro

**Comment en avez-vous eu connaissance ?**

- publicité                       catalogue                       autres  
 exposition                       conseils d'un ami

**Avez-vous déjà acquis des livres P.S.I. ?**

Lesquels ? .....

qu'en pensez-vous ? .....

.....  
 .....

Nom ..... Prénom ..... Age.....

Adresse .....

Profession .....

Centre d'intérêt .....



## CATALOGUE GRATUIT

Vous pouvez obtenir un catalogue complet des ouvrages PSI, sur simple demande, ou en retournant cette page remplie à votre libraire, à votre boutique micro ou aux

**Editions du PSI**  
**BP 86**  
**77402 Lagny-sur-Marne Cedex**

# CLEFS POUR APPLE II GS

Ce mémento s'adresse aux programmeurs en assembleur, C et Basic de l'Apple II GS.

Il offre en effet une synthèse des spécificités du matériel et des logiciels de développement.

Vous disposez ainsi des informations fondamentales concernant l'architecture interne, les brochages, le jeu d'instructions du 65816, les mémoires, les ressources graphiques et les entrées-sorties.

Le système CPW, avec son moniteur, son éditeur et son macro-assembleur, est décrit en détail. L'ensemble des outils du bureau électronique (en particulier QuickDraw, Window Manager, Menu Manager...) est répertorié, fonction par fonction.

À la fin de l'ouvrage, un programme montre comment ajouter un accessoire de bureau à une application, et résume ainsi les possibilités graphiques de l'Apple II GS.



ISBN 2 - 86595 - 368 - 8

250 FF

PHOTO THIERRY LAYANI

prix éditeur  
99/6425000  
566/23750  
prix fnac