

UNE MERVEILLE DE SIMPLICITÉ



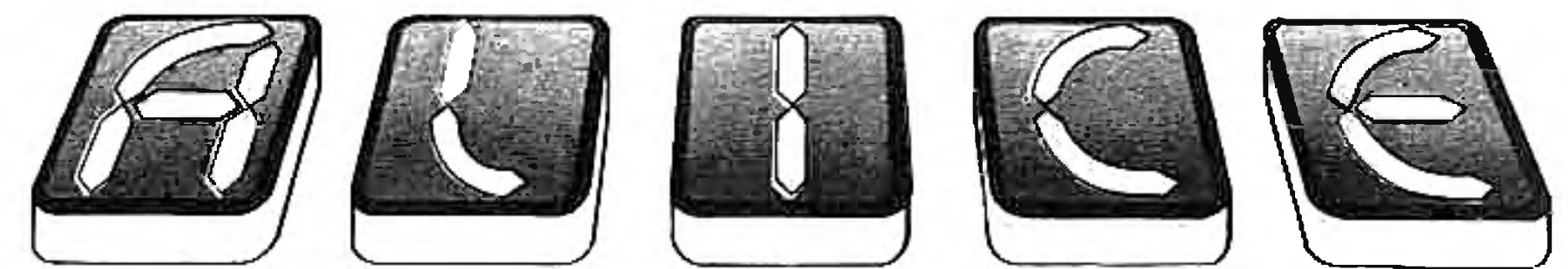
GUIDE D'INSTRUCTIONS
DE L'ÉDITEUR ASSEMBLEUR



MATRA ET HACHETTE

ALICE

Ce guide a été réalisé
par la société European Media Business
avec le concours de Monsieur GALLET (MATRA).
L'illustration de couverture est l'œuvre de Moebius



GUIDE D'INSTRUCTIONS DE L'ÉDITEUR-ASSEMBLEUR

MATRA ET HACHETTE

Sommaire

Avant-propos	7
1. Aller plus loin avec Alice	8
2. Les commandes de l'Editeur	10
3. Le format et les directives de l'Assembleur	14
4. Mode d'adressage et instructions du MC 6803	16
Transfert de données.....	18
Addition, multiplication et soustraction	20
Inversion arithmétique.....	22
Décalages et rotations.....	22
Opérations logiques.....	24
Positionnement des bits de P.....	26
Incrémentation et décrémentation	26
Remise à zéro	26
Utilisation de la pile	28
Comparaison et tests.....	28
Sauts, branchements, retour de routine.....	30
Interruptions	32
Instruction muette	32
 Annexes	
1. Modifications du registre d'état P	33
2. Tableau des mnémoniques.....	35
3. Pour ceux qui débutent	38
4. Quelques routines utiles	40
5. Connecter une imprimante.....	41
6. Présentation du listing	42
7. Un exemple de programme	43
8. Codes erreur.....	45

Avant-propos

Le BASIC d'Alice vous a permis de vous familiariser avec les caractéristiques de votre micro-ordinateur. Et vous avez sans doute déjà réalisé des programmes très élaborés, reflétant vos capacités de création et d'imagination.

Grâce à l'Éditeur-Assembleur Alice, vous pourrez aller encore plus loin dans la conception de vos programmes qui gagneront en vitesse d'exécution et en richesse graphique.

Cet ouvrage, contrairement au Guide DÉCOUVREZ LE BASIC, s'adresse aux programmeurs déjà initiés au code machine. Il constitue le Guide d'utilisation de l'Éditeur-Assembleur d'Alice.

Chapitre 1

Aller plus loin avec ALICE

I CE QU'IL FAUT SAVOIR

UN PROGICIEL EN ROM

Ne cherchez pas, dans la liste du matériel fourni avec Alice, une cassette portant le titre "Éditeur-Assembleur" ; vous ne la trouverez pas.
Comme l'interpréteur BASIC Microsoft, l'Éditeur-Assembleur d'ALICE est en effet stocké de manière permanente en ROM, de façon à être toujours disponible.

PÉRIPHÉRIQUES ET EXTENSIONS : LA CONFIGURATION IDÉALE

Un magnétophone à cassette vous sera nécessaire pour sauvegarder :

- vos fichiers écrits en Assembleur (**fichiers sources**),
- vos programmes exécutables, une fois assemblés en code machine (**fichiers objets**).

L'utilisation d'une imprimante – permettant le tirage de listings – est vivement recommandée pour la mise au point et la correction des fichiers sources.

Pour savoir quel type de magnétophone et d'imprimante choisir, reportez-vous au guide DÉCOUVREZ LE BASIC et à l'annexe 5 à la fin de ce guide.

Pour le modèle de base, le module d'extension de 16 K permettra la création de programmes en Assembleur plus longs. Sans extension, vous disposerez de **7 K RAM** pour l'Éditeur-Assembleur, autorisant la création de fichiers source de quelque **300 lignes de programme** et la génération de fichiers objet d'environ **1 K**. Avec l'extension, vous bénéficierez d'une place en mémoire suffisante pour 1 000 lignes sources générant jusqu'à 4 K de code machine.

ATTENTION A LA SAUVEGARDE DES PROGRAMMES ET DONNÉES BASIC

Avant de mettre Alice en mode Éditeur-Assembleur, n'oubliez pas de sauvegarder sur cassette les données et programmes Basic stockés en RAM puis de faire NEW pour remettre à zéro les pointeurs mémoire.

II COMMENT ENTRER EN MODE ÉDITEUR-ASSEMBLEUR

A LA PROCÉDURE NORMALE

Les lignes de programme en Assembleur que vous allez écrire seront stockées par l'Éditeur dans une zone mémoire que vous devrez préciser au préalable à l'aide de l'instruction :

CLEAR LN,AD **ENTER**

LN doit toujours avoir une valeur supérieure à 60.

AD est l'adresse en RAM à partir de laquelle sera stocké le fichier source. AD ne pourra prendre que des valeurs comprises dans les intervalles suivants :

ALICE sans extension mémoire :	13250 à 20480
ALICE avec extension 16 K :	13250 à 36864
ALICE 90 :	13250 à 45056

Pour déterminer la valeur de AD, il vous faudra tenir compte des éléments suivants :

- AD doit toujours être inférieure à l'adresse mémoire RAM maximum autorisée (20480 sur Alice sans extension mémoire).
- il ne devra y avoir aucune possibilité de recouvrement du fichier source par le fichier objet.

Appeler l'Éditeur-Assembleur

L'étape suivante consiste à entrer en mode Éditeur-Assembleur en tapant :

& **ENTER**

A la suite de cette commande, le rectangle d'affichage de l'écran s'efface, le fond vert devient bleu et le curseur clignotant indique que le programme attend vos instructions.

B LA PROCÉDURE APRÈS ERREUR

Si, durant l'utilisation de l'Éditeur-Assembleur, vous commettez une erreur et revenez involontairement au BASIC, vous avez la faculté, dans la majorité des cas, de "récupérer" votre fichier source en appelant à nouveau l'Éditeur-Assembleur avec la commande :

% **ENTER**

Cette possibilité ne vous est toutefois offerte que si vous n'avez tapé aucune commande BASIC entre le moment où vous êtes sorti de l'Éditeur-Assembleur et celui où vous tapez : **%** **ENTER**.

Chapitre 2

Les commandes de l'Éditeur

L'Éditeur est du type "pleine page" : vous disposez de tout l'écran – à l'exception de la dernière ligne réservée au dialogue entre vous et la machine – pour l'écriture et la correction de votre programme en Assembleur.

REVENIR AU BASIC : BREAK ou INIT

Deux pressions consécutives sur la touche BREAK permettent de revenir au BASIC. Si vous n'avez appuyé qu'une seule fois sur cette touche, vous pourrez rester dans l'Éditeur en pressant n'importe quelle autre touche.

La touche INIT ne sera à utiliser qu'en cas d'erreur fatale ayant provoqué un "plantage" de l'Éditeur (ex. : demande d'impression sans imprimante connectée).

CHANGER DE MODE

L'Éditeur-Assembleur fonctionne indifféremment sur 40 ou sur 80 colonnes. Pour passer d'une mode à l'autre, utilisez :

[CTRL] [8]

DÉPLACER LE CURSEUR

Vous pouvez déplacer le curseur sur tout l'écran (à l'exception de la dernière ligne) en faisant :

[CTRL] [Z] (flèche vers le haut)	passage à la ligne du dessus et retour à la marge gauche.
[CTRL] [W] (flèche vers le bas)	passage à la ligne du dessous et retour à la marge gauche.
[ENTER]	comme [CTRL] [W]
[CTRL] [Q] (flèche gauche)	Déplacement d'un caractère vers la gauche.
[CTRL] [S] (flèche droite)	Déplacement d'un caractère vers la droite.
[CTRL] [T]	Déplacement de 6 caractères à la fois vers la droite.

Remarque : Si vous possédez un ALICE 90, vous pouvez utiliser simplement les touches de direction situées à droite du clavier.

VALIDER UNE LIGNE

Après avoir écrit ou corrigé une ligne, vous devez la valider pour qu'elle soit prise en compte et mise en mémoire.

Cette opération s'effectue simplement en quittant la ligne à l'aide des commandes de déplacement vertical du curseur.

[ENTER] ,

[CTRL] [Z] (ou flèche vers le haut),

[CTRL] [W] (ou flèche vers le bas).

EFFACER UNE LIGNE OU UN CARACTÈRE

Pour effacer toute une ligne, amenez le curseur sur n'importe quel caractère de cette ligne et tapez :

[CTRL] [A]

Pour effacer un caractère, il suffit d'amener le curseur sur celui-ci et de taper :

[CTRL] [D]

INSÉRER UNE LIGNE OU DES CARACTÈRES

Vous pouvez insérer une ligne "blanche" dans un programme, au-dessus de la position occupée par le curseur, en utilisant la commande :

[CTRL] [I]

Pour insérer un ou plusieurs caractères dans une ligne existante, placez le curseur sur le caractère qui suivra immédiatement l'insertion et faites :

[CTRL] [J]

Tapez ensuite normalement le ou les caractères à insérer.

La sortie de ce mode s'obtient soit en changeant de ligne grâce aux commandes de déplacement vertical du curseur, soit à l'aide de **[CTRL] [J]**.

Attention : A chaque insertion d'un caractère, la fin de la ligne est repoussée d'une position vers la droite. Or, lorsque la ligne devient trop longue, les derniers caractères disparaissent par le bord droit de l'écran et sont définitivement perdus.

RÉPÉTITION AUTOMATIQUE DES TOUCHES

Il existe une commande qui vous évite d'avoir à appuyer à plusieurs reprises sur une même touche du clavier pour répéter un caractère. Vous obtiendrez la répétition automatique et le retour au mode normal en appuyant sur :

[CTRL] [R]

RECHERCHER UNE LIGNE OU UNE CHAÎNE DE CARACTÈRES

Pour faciliter la mise au point et la correction de vos programmes, vous disposez de deux commandes de recherche de texte dans le fichier source :

CTRL **L** Numéro de la ligne à chercher **ENTER**

Affiche en haut de l'écran la ligne trouvée. Si vous avez tapé **CTRL** **L** **0** **ENTER**, le curseur se placera automatiquement sur la dernière ligne de votre programme.

CTRL **F** Textes et symboles à rechercher **ENTER**

Affiche en haut de l'écran la ligne de programme contenant la chaîne de caractères recherchée et place le curseur au début de celle-ci.

Si vous êtes en mode 40 colonnes et que la chaîne cherchée se trouve au-delà de la 40^e position, vous passez automatiquement en mode 80 colonnes.

Cette commande ne concerne que les lignes comprises entre la ligne suivant celle où se trouve le curseur et la fin du fichier.

*Si vous faites **CTRL** **W** ou **ENTER** après la dernière ligne, vous ajoutez des lignes vides qui seront prises en compte lors de l'exécution de commandes du type **CTRL** **L** **0**.*

SAUVEGARDER UN FICHIER SOURCE SUR CASSETTE ET LE VÉRIFIER

La sauvegarde d'un fichier source sur cassette est aussi simple qu'en BASIC. Il suffit de taper :

CTRL **3** Nom du Fichier **ENTER**

Pour vérifier la qualité de l'enregistrement qui vient d'être réalisé, il suffit de rebobiner suffisamment la cassette du magnétophone, puis de faire appel à la commande :

CTRL **V** Nom du Fichier **ENTER**

qui est analogue à la commande SKIPF du BASIC.

Attention : il est conseillé, lors des opérations de sauvegarde, de débrancher – le cas échéant – l'imprimante connectée à ALICE.

CHARGER EN MÉMOIRE UN FICHIER SOURCE ENREGISTRÉ SUR CASSETTE.

Pour charger, à partir du magnétophone à cassette, un nouveau programme source tout en écrasant celui qui est en mémoire, utilisez la commande :

CTRL **4** Nom du Fichier à charger **ENTER**

Pour ajouter à la fin d'un fichier source déjà en mémoire, un autre fichier (Merging) à partir du magnétophone, tapez :

CTRL **M** Nom du Fichier à charger **ENTER**

COPIER SUR IMPRIMANTE

Vous obtiendrez une copie intégrale du fichier source sur imprimante grâce à la commande :

CTRL **6**

Une simple copie d'écran est réalisée avec la commande :

CTRL **H**

Attention : Si aucune imprimante n'est connectée à ALICE lorsque vous demandez l'exécution de ces deux commandes, vous devrez appuyer sur la touche INIT (Retour au BASIC) pour reprendre le contrôle de votre ALICE.

ASSEMBLER LE PROGRAMME SOURCE

Après avoir écrit tout votre programme source et l'avoir sauvegardé, vous demandez sa traduction en code machine, exécutable par l'ordinateur, en frappant :

CTRL **I**

L'éditeur vous demande alors quel support de listing vous souhaitez. Vous pouvez répondre ainsi :

E **ENTER** : Visualisation sur l'écran seul du listing d'assemblage,

I **ENTER** : Copie du listing d'assemblage sur imprimante,

ENTER : Visualisation sur écran réduite aux messages d'erreur.

Vous devrez ensuite indiquer sous quel nom vous voulez sauvegarder sur cassette le programme en code machine produit par l'Assembleur en tapant :

Nom **ENTER**

Si vous ne souhaitez pas de sauvegarde du fichier objet, appuyez sur :

BREAK

Enfin vous aurez la possibilité d'exécuter directement le programme assemblé en répondant OUI à la demande de l'Éditeur (ou NON dans le cas contraire...). Après exécution, ALICE repassera en mode BASIC.

Annuler une commande :

*Lorsque l'Éditeur vous demande une réponse sur la dernière ligne de l'écran, vous pouvez annuler l'ensemble de la commande en tapant simplement **BREAK**.*

Corriger une réponse :

*Pour corriger une réponse erronée, tapez simplement **CTRL** **A** à la place de **ENTER**.*

Chapitre 3

Le format et les directives de l'Assembleur

I LE FORMAT

Pour être comprises et traduites par l'Assembleur ALICE, les lignes des fichiers source doivent toujours avoir le format suivant :

Étiquette (Label)

Elle doit toujours être placée en première colonne (contre le bord gauche de l'écran) et ne doit pas comporter plus de 5 caractères.

Commentaire

Il doit être précédé et marqué obligatoirement par un point-virgule. Il peut être introduit à n'importe quel endroit d'une ligne, car tout ce qui se trouve derrière le point-virgule est ignoré par l'Assembleur.

Mnémonique et directive

Un mnémonique (représentant le code d'une instruction du microprocesseur) ou une directive à l'Assembleur ne peut être placé que dans une colonne de rang supérieur à 1 (colonne 2 à 40 ou 80 suivant le mode).

Valeur

Les valeurs numériques et adresses doivent être exprimées en hexadécimal et commencer par le signe "\$" (dollar).

Attention : Toutes les composantes d'une ligne (étiquette, mnémonique d'une instruction ou directive, adresse ou donnée, commentaire) doivent être séparées par au moins un espace.

EXEMPLES DE FORMAT DE LIGNES SOURCES

Zone Étiquette (colonne 0)	Zone mnémonique ou directive	Zone donnée ou opérande
DATA1	ORG EXEC =	\$8000 : Directive + adresse DEBUT : Directive + adresse \$01 : Étiquette + Directive ; + valeur numérique
DEBUT	LDX	#DATA1 : Étiquette + mnémonique ; + donnée immédiate

II LES DIRECTIVES A L'ASSEMBLEUR

Pour faciliter votre programmation, l'Assembleur ALICE met à votre disposition un certain nombre de directives :

AFFECTER UNE VALEUR A UNE ÉTIQUETTE

Le signal "=" (égal) permet de réaliser cette opération.

Exemple : DATA5 = \$1234

INTRODUIRE UNE CHAÎNE DE TEXTE ASCII

Le signe ' (apostrophe) placé devant du texte ASCII indique à l'Assembleur qu'il doit placer ce dernier en mémoire à l'adresse courante.

Exemple : TEXT5 'CECI EST UNE CHAÎNE DE CARACTERES ASCII

RÉSERVER UN ESPACE EN MÉMOIRE

BLC réserve un bloc pouvant contenir jusqu'à 255 octets.

Exemple : (Étiquette facultative) BLC \$ FF

STOCKER UN OU DEUX OCTETS

DFO et DFD réservent respectivement un ou deux octets en mémoire et y attribuent la valeur qui suit immédiatement la directive.

Exemples : (Étiquette facultative) DFO \$ 56
(Étiquette facultative) DFD \$ 3241

DÉFINIR L'ADRESSE D'IMPLANTATION DU CODE MACHINE EN MÉMOIRE

Un fichier source doit toujours commencer par la directive ORG qui indique à l'Assembleur à partir de quelle adresse en RAM il doit stocker le code machine.

Exemple : ORG \$7500

VALEURS POSSIBLES POUR ORG	
ALICE sans extension mémoire	de \$3000 à \$4FFF
ALICE avec extension 16 K	de \$3000 à \$8FFF
ALICE 90	de \$3000 à \$AFFF

Attention : Lorsque vous fixerez la valeur de ORG, prenez garde à ce que votre fichier objet (le programme en code machine) ne vienne écraser une partie du fichier source.

DÉFINIR LE POINT D'ENTRÉE DU PROGRAMME

EXC définit l'adresse de la première instruction de votre programme qui devra être exécutée. Sa syntaxe est la même que celle de la directive ORG.

Chapitre 4

Mode d'adressage et instructions du MC 6803

I L'ADRESSAGE

Comme le 6800, le micro-processeur 6803 d'Alice n'accepte pas le mode adressage indirect; mais l'indexation par le registre double IX autorise des transferts de données sur les 64 K de mémoire.

ADRESSAGE IMMÉDIAT

La donnée servant d'opérande suit l'instruction. Ce mode est marqué dans le programme source par le signe “#” (dièse) placé devant l'opérande.

Exemple : LDAA # \$34

ADRESSAGE DIRECT EN PAGE ZÉRO

L'adresse de la donnée opérande est fournie sur l'octet suivant l'instruction, précédé du signe “<” (inférieur à).

Exemple : LDX < SEA

ADRESSAGE DIRECT ÉTENDU

L'adresse de l'opérande est fournie sur deux octets suivant l'instruction. Ne fait appel à aucun code particulier.

Exemple : TST \$A723

ADRESSAGE INDEXÉ

L'adresse de l'opérande est obtenue en ajoutant la valeur de l'octet qui suit l'instruction au contenu du registre IX. Ce mode se code ainsi :

Mnémonique 1 seul octet, X

Exemple : ADDB \$7E, X

ADRESSAGE INHÉRENT

Dans ce cas, on ne fait aucune référence à une adresse mémoire.

Exemple : NOP

ADRESSAGE RELATIF

Mode utilisé uniquement pour les branchements à une adresse, dans un programme, contenant une instruction à exécuter. Ce type d'adressage autorise des sauts de - 126 à + 128 octets.

Exemple :

BRA ROUTE
Lignes de programmes

ROUTE LDX # \$5B

ROUTE aura une valeur supérieure à 0 et inférieure à 128 (déplacement positif).

II LE JEU D'INSTRUCTIONS DU MC6803

Explications des abréviations utilisées.

Adressage :

IM1 = Valeur immédiate sur un octet,
IM2 = Valeur immédiate sur deux octets,
DR = Adresse sur un ou deux octets,
DR1 = Adresse sur un octet,
DR2 = Adresse sur deux octets,
DEP = Déplacement (valeur comprise entre - 128 et + 127)

Registres internes :

A = Registre simple, accumulateur A,
B = Registre simple, accumulateur B,
P = Registre simple, registre d'état,
D = Registre double composé de A et B (Poids fort : A),
IX = Registre double, registre d'indexation,
SP = Registre double, pointeur de pile.

Bits indicateurs d'état du registre P :

C = Retenue,
V = Débordement,
Z = Mise à zéro,
N = Signe,
I = Interruption,
H = Demi-retenue.

TRANSFERT DE DONNÉES

Les opérations de transfert affectent les bits du registre d'état (P) de la manière suivante :

- Z est mis à 1 si la valeur transférée est égale à 0.
- N est mis à 1 si la valeur transférée est négative (si le bit 7 ou le bit 15 de la valeur hexadécimale est à 1).

Chargement de registres

REGISTRE	MODE	INSTRUCTION	RAPPEL
A	Immédiat	LDAA IM1	charge le registre avec la donnée IM1 charge le registre avec la valeur stockée en DR charge le registre avec la valeur stockée en DRI + IX
A	Direct	LDAA DR	
A	Indexé	LDAA DRI,X	
B	Immédiat	LDAB IM	
B	Direct	LDAB DR	
B	Indexé	LDAB DRI,X	
D	Immédiat	LDD IM2	
D	Direct	LDD DR	
D	Indexé	LDD DRI,X	
IX	Immédiat	LDX IM2	
IX	Direct	LDX DR	
IX	Indexé	LDX DRI,X	
SP	Immédiat	LDS IM2	
SP	Direct	LDS DR	
SP	Indexé	LDS DRI,X	

Transfert entre registres

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A et B	Inhérent	TAB	charge dans le second registre le contenu du premier les bits du registre P sont tous "écrasés"
A et P	Inhérent	TAP	
B et A	Inhérent	TBA	
P et A	Inhérent	TPA	
IX et SP	Inhérent	TXS	transfert de valeurs sur 2 octets entre registres doubles
SP et IX	Inhérent	TSX	

Chargement d'emplacements en mémoire à partir de registres

REGISTRE	MODE	INSTRUCTION	RAPPEL
A	Direct	STAA DR	charge DR avec le contenu du registre charge DRI + IX avec le contenu du registre
A	Indexé	STAA DRI,X	
B	Direct	STAB DR	
B	Indexé	STAB DRI,X	
D	Direct	STD DR	
D	Indexé	STD DRI,X	
IX	Direct	STX DR	
IX	Indexé	STX DRI,X	
SP	Direct	STS DR	
SP	Indexé	STS DRI,X	

ADDITION, MULTIPLICATION ET SOUSTRACTION

Les bits d'état de P sont affectés de la manière suivante :

- Z est mis à 1 si le résultat est nul,
- N est mis à 1 si le résultat est négatif,
- V est mis à 1 si une retenue du bit 6 sur le bit 7 a été générée, lors d'une addition.
- H est positionné dans des conditions similaires à V mais pour les bits 3 et 4.
- C est mis à 1 si le résultat est supérieur à \$FF.

Attention : Sauf pour la multiplication, c'est le registre spécifié dans la première colonne du tableau qui stocke le résultat de l'opération.

Additions

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A	Inhérent	DAA	ajuste le registre pour obtenir une valeur en DCB
A et B	Inhérent	ABA	additionne A et B, résultat en A
IX et B	Inhérent	ABX	additionne X et B, résultat dans IX
A	Immédiat	ADDA IM1	additionne IM1 et le contenu du registre
A	Direct	ADDA DR	additionne le contenu du registre et de DR
A	Indexé	ADDA DR1,X	additionne le contenu du registre et de DR1 + IX
B	Immédiat	ADDB IM1	additionne IM1, le contenu du registre et la retenue C
B	Direct	ADDB DR	
B	Indexé	ADDB DR1,X	
A	Immédiat	ADCA IM1	
A	Direct	ADCA DR	
A	Indexé	ADCA DR1,X	
B	Immédiat	ADCB IM1	additionne le contenu du registre, de DR et C
B	Direct	ADCB DR	
B	Indexé	ADCB DR1,X	
D	Immédiat	ADDD IM2	
D	Direct	ADDD DR	
D	Indexé	ADDD DR1,X	

Multiplcation

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A et B	Inhérent	MUL	multiplie A par B, résultat dans D

Soustractions

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A et B	Inhérent	SBA	soustrait B à A, résultat en A
A	Immédiat	SUBA IM1	soustrait IM1 au contenu du registre
A	Direct	SUBA DR	soustrait le contenu de DR à celui du registre
A	Indexé	SUBA DR1,X	soustrait le contenu de DR1 + IX à celui du registre
B	Immédiat	SUBB IM1	soustrait IM1 et la retenue au contenu du registre
B	Direct	SUBB DR	
B	Indexé	SUBB DR1,X	
D	Immédiat	SUBD IM2	
D	Direct	SUBD DR	
D	Indexé	SUBD DR1,X	
A	Immédiat	SBCA IM1	soustrait le contenu de DR et C au registre
A	Direct	SBCA DR	
A	Indexé	SBCA DR1,X	soustrait le contenu de DR1 + IX et C au registre
B	Immédiat	SBCB IM1	
B	Direct	SBCB DR	
B	Indexé	SBCB DR1,X	

INVERSION ARITHMÉTIQUE (COMPLÉMENTATION À 2)

Effets sur le registre d'état :

- N et Z sont modifiés en fonction des résultats,
- V est mis à 1 si le bit 7 de la donnée complémentée était à 1 ; dans le cas contraire, il est mis à 0,
- C est mis à 0 si l'octet était nul, et à 1 dans le cas contraire.

DÉCALAGES ET ROTATIONS

Effets sur le registre d'état :

- Z est positionné à 1 si le contenu du registre ou de l'adresse concerné est mis à zéro,
- C prend la valeur du bit sortant par décalage ou rotation, tandis que V résulte d'un OU exclusif entre N et C,
- N est mis à 1 ou 0 en fonction du bit 7 de l'octet ayant subi un décalage ou une rotation.

Attention : ASL pourra être remplacé par LSL dans les programmes.

Inversion arithmétique

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A B	Inhérent Inhérent Direct Indexé	NEGA NEGB NEG DR2 NEG DR1,X	inversion arithmétique du registre A inversion arithmétique du registre B inversion arithmétique du contenu de l'adresse DR2 inversion arithmétique du contenu de l'adresse DR1 + IX

Décalages

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A B D	Inhérent Inhérent Inhérent Direct Indexé	ASLA ASLB ASLD ASL DR2 ASL DR1,X	décalage à gauche du contenu du registre décalage à gauche du contenu d'adresse DR décalage à gauche du contenu d'adresse DR1 + IX
A	Inhérent	ASRA	décalage à droite du contenu du registre avec restauration du bit 7 à sa valeur initiale
B	Inhérent Direct Indexé	ASRB ASR DR2 ASR DR1,X	
A B D	Inhérent Inhérent Inhérent Direct Indexé	LSRA LSRB LSRD LSR DR2 LSR DR1,X	décalage à droite du contenu du registre décalage à droite du contenu d'adresse DR décalage à droite du contenu d'adresse DR1 + IX

Rotations

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A B	Direct Indexé	ROLA ROLB ROL DR2 ROL DR1,X	rotation à gauche du contenu du registre rotation à gauche du contenu d'adresse DR rotation à gauche du contenu d'adresse DR1 + IX
A B	Direct Indexé	RORA RORB ROR DR2 ROR DR1,X	rotation à droite du contenu du registre rotation à droite du contenu d'adresse DR rotation à droite du contenu d'adresse DR1 + IX

OPÉRATIONS LOGIQUES

Les opérations logiques affectent les bits du registre d'état (P) de la manière suivante :

- Z est mis à 1 si la valeur finale obtenue est égale à 0,
- N est mis à 1 si cette valeur est négative (si le bit 7 de la valeur hexadécimale est à 1),
- dans le cas de l'inversion logique, C est toujours positionné à 1.

Attention : C'est le registre spécifié dans la première colonne du tableau qui stocke le résultat de l'opération logique.

ET logique

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A A	Immédiat Direct	ANDA IM1 ANDA DR	ET logique entre registre et IM1 ET logique entre registre et contenu d'adresse DR ET logique entre registre et contenu d'adresse DR1 + IX
A	Indexé	ANDA DR1,X	
B B B	Immédiat Direct Indexé	ANDB IM1 ANDB DR ANDB DR1,X	

OU inclusif

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A A	Immédiat Direct	ORAA IM1 ORAA DR	OU inclusif entre registre et IM1 OU inclusif entre registre et contenu d'adresse DR OU inclusif entre registre et contenu d'adresse DR1 + IX
A	Indexé	ORAA DR1,X	
B B B	Immédiat Direct Indexé	ORAB IM1 ORAB DR ORAB DR1,X	

OU exclusif

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A A	Immédiat Direct	EORA IM1 EORA DR	OU exclusif entre registre et IM1 OU exclusif entre registre et contenu d'adresse DR OU exclusif entre registre et contenu d'adresse DR1 + IX
A	Indexé	EORA DR1,X	
B B B	Immédiat Direct Indexé	EORB IM1 EORB DR EORB DR1,X	

Inversion logique des bits d'un octet

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A B	Inhérent Inhérent Direct	COMA COMB COM DR2	inversion logique du registre A inversion logique du registre B inversion logique du contenu de l'adresse DR2
	Indexé	COM DR1,X	

POSITIONNEMENT DES BITS DE P

Ces instructions n’affectent que les bits qu’elles positionnent.

INCRÉMENTATION ET DÉCRÉMENTATION

Ces opérations affectent les bits du registre d’état N, Z et V du registre P de la manière suivante :

- Z est mis à 1 si un registre ou un contenu en mémoire est mis à zéro,
- N est mis à 1 si la valeur résultant des opérations est négative (c’est-à-dire si le bit 7 ou le bit 15 de la valeur hexadécimale est à 1),
- V est mis à 1 lors d’une incrémentation si la valeur de départ était inférieure à \$7F, ou lors d’une décrémentation si cette valeur était supérieure ou égale à \$80.

REMISE À ZÉRO

Effet sur les bits du registre d’état :

- N, V et C sont modifiés,
- Z est mis à 1.

Positionnement des bits de P

BIT	MODE	INSTRUCTION	RAPPEL OU REMARQUE
C	Inhérent	CLC	mise à 0 du BIT
I	Inhérent	CLI	
V	Inhérent	CLV	
C	Inhérent	SEC	mise à 1 du BIT
I	Inhérent	SEI	
V	Inhérent	SEV	

Incrémentation

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A	Indexé Direct	INCA	incrémente le registre simple
B		INCB	incrémente le registre double IX, Z est seul affecté
IX		INX	
SP		INS	incrémente le registre double SP
		INC DR1,X	incrémente le contenu de DR1 + IX
		INC DR	DR sera obligatoirement une adresse sur 2 octets

Décrémentation

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A	Indexé Direct	DECA	décrémente le registre simple
B		DECB	décrémente le registre double IX, Z est seul affecté
IX		DEX	
SP		DES	décrémente le registre double SP
		DEC DR1,X	décrémente le contenu de DR1 + IX
		DEC DR	DR sera obligatoirement une adresse sur 2 octets

Remise à zéro

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A	Inhérent Inhérent Direct	CLRA	remise à zéro du registre A
B		CLRB	remise à zéro du registre B
		CLR DR2	remise à zéro du contenu de l'adresse DR2
	Indexé	CLR DR1,X	remise à zéro du contenu de l'adresse DR1 + IX

UTILISATION DE LA PILE

Les opérations d'empilage/dépilage n'affectent pas les indicateurs d'état.

Empilage, dépilage

REGISTRE	MODE	INSTRUCTION	RAPPEL
A B IX A B IX	Inhérent Inhérent Inhérent Inhérent Inhérent Inhérent	PSHA PSHB PSHX PULA PULB PULX	empile le contenu du registre charge le registre avec le sommet de la pile

COMPARAISON ET TESTS

Les comparaisons s'effectuent en soustrayant la donnée en mémoire de celle servant de référence et contenue dans le registre spécifié.

Avec CBA, CPX et CMP, les indicateurs N, Z et V sont positionnés en fonction du résultat de la soustraction. C est mis à 1, si, en notation non signée, la donnée à comparer est supérieure à la référence.

Les tests positionnent N et Z en fonction des résultats. BIT autorise le test des bits d'un octet en mémoire effectuant un ET logique avec le registre A. TST sert seulement à vérifier si une valeur est égale à 0 ou négative.

Comparaisons

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A et B A A A B B B IX IX IX	Inhérent Immédiat Direct Indexé Immédiat Direct Indexé Immédiat Direct Indexé	CBA CMPA IM CMPA DR CMPA DRI,X CMPB IM CMPB DR CMPB DRI,X CPX IM2 CPX DR CPX DRI,X	compare le contenu de B à A compare le registre à la donnée IM compare le registre au contenu d'adresse DR compare le registre au contenu d'adresse DRI + IX

Tests

REGISTRE	MODE	INSTRUCTION	RAPPEL OU REMARQUE
A A A B B B A B	Immédiat Direct Indexé Immédiat Direct Indexé Inhérent Inhérent Immédiat Direct Indexé	BITA IM1 BITA DR BITA DRI,X BITB IM1 BITB DR BITB DRI,X TSTA TSTB TST IM TST DR TST DRI,X	ET logique entre le registre et la donnée IM1

SAUTS, BRANCHEMENTS, RETOUR DE ROUTINES

Ces instructions n'affectent aucun bit du registre P.

Attention : BHS et BLO pourront être employés à la place, respectivement, de BCC et BCS.

Saut

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
	Direct Indexé Direct	JMP DR2 JMP DR1,X JSR DR2	saut inconditionnel à l'adresse DR2 saut inconditionnel à l'adresse DR1 + IX saut inconditionnel vers un sous-programme d'adresse DR2
	Indexé	JSR DR1,X	saut inconditionnel vers un sous-programme d'adresse DR1 + IX

Non Branchement

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
		BRN DEP	effet équivalent à NOP

Branchement inconditionnel

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
	Relatif	BRA DEP	branchement inconditionnel à l'adresse courante + DEP

Branchements conditionnels
(* = ou inclusif; ° = ou exclusif)

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
C = 0	Relatif	BSR DEP	branchement inconditionnel vers un sous-programme à l'adresse courante + DEP branchement conditionnel à l'adresse courante + DEP
C = 1	Relatif	BCC DEP	
C°Z = 0	Relatif	BCS DEP	
C°Z = 1	Relatif	BHI DEP	
N = 1	Relatif	BLS DEP	
N = 0	Relatif	BMI DEP	
N*V = 0	Relatif	BPL DEP	
N*V = 1	Relatif	BGE DEP	
N°(N*V) = 0	Relatif	BLT DEP	
V = 0	Relatif	BGT DEP	
V = 1	Relatif	BVC DEP	
Z = 0	Relatif	BVS DEP	
Z = 1	Relatif	BNE DEP	
Z°(N*V) = 1	Relatif	BEQ DEP	
	Relatif	BLE DEP	

Retours de sous-programmes

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
	Inhérent Inhérent	RTI RTS	retour d'interruption retour après JSR ou BSR

INTERRUPTIONS

Le vecteur de l'interruption hardware non masquable est en FFFC-FFFD ; le vecteur de l'interruption masquable en FF8-FF9.

Interruptions

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
	Inhérent	SWI	interruption programmée (le vecteur d'interruption se trouve en FFFA et FFFB) attente d'une interruption
	Inhérent	WAI	

INSTRUCTION MUETTE

Cette instruction est utilisée pour permettre une temporisation dans le programme.

Temporisation

CONDITION	MODE	INSTRUCTION	RAPPEL OU REMARQUE
	Inhérent	NOP	

ANNEXES

Annexe 1
Modifications
du registre d'état P

Explication des codes utilisés :

H = demi-retenuc	C = retenue
I = interruption (inhibition)	R = mise à 0
N = signe négatif	S = mise à 1
Z = zéro	‡ = bit affecté
V = débordement (overflow)	• = bit non affecté

Tableau 1: Opérations sur les pointeurs IX et SP

MNEM	H	I	N	Z	V	C
CPX	•	•	‡	‡	‡	‡
DEX	•	•	•	‡	•	•
DES	•	•	•	•	•	•
INX	•	•	•	‡	•	•
INS	•	•	•	•	•	•
LDX	•	•	‡	‡	R	•
LDS	•	•	‡	‡	R	•

Tableau 2: Opérations directes sur bits

MNEM	H	I	N	Z	V	C
CLC	•	•	•	•	•	R
CLI	•	R	•	•	•	•
CLV	•	•	•	•	R	•
SEC	•	•	•	•	•	S
SEI	•	S	•	•	•	•
SEV	•	•	•	•	S	•

Annexe 2

Tableau des mnémoniques

Tableau 3 : Opérations sur accumulateurs ou positions mémoire

MNEM	H	I	N	Z	V	C
ABA	↓	•	↓	↓	↓	↓
ABX	•	•	•	•	•	•
ADCA	↓	•	↓	↓	↓	↓
ADCB	↓	•	↓	↓	↓	↓
ADDA	↓	•	↓	↓	↓	↓
ADDDB	↓	•	↓	↓	↓	↓
ADDD	•	•	↓	↓	↓	↓
ANDA	•	•	↓	↓	R	•
ANDB	•	•	↓	↓	R	•
ASL	•	•	↓	↓	↓	↓
ASLA	•	•	↓	↓	↓	↓
ASLB	•	•	↓	↓	↓	↓
ASLD	•	•	↓	↓	↓	↓
ASR	•	•	↓	↓	↓	↓
ASRA	•	•	↓	↓	↓	↓
ASRB	•	•	↓	↓	↓	↓
BITA	•	•	↓	↓	R	•
BITB	•	•	↓	↓	R	•
CBA	•	•	↓	↓	↓	↓
CLR	•	•	R	S	R	R
CLRA	•	•	R	S	R	R
CLRB	•	•	R	S	R	R
CPMA	•	•	↓	↓	↓	↓
CMPB	•	•	↓	↓	↓	↓
COM	•	•	↓	↓	R	S
COMA	•	•	↓	↓	R	S
COMB	•	•	↓	↓	R	S

MNEM	H	I	N	Z	V	C
DAA	•	•	↓	↓	↓	↓
DEC	•	•	↓	↓	↓	•
DECA	•	•	↓	↓	↓	•
DECB	•	•	↓	↓	↓	•
EORA	•	•	↓	↓	R	•
EORB	•	•	↓	↓	R	•
INC	•	•	↓	↓	↓	•
INCA	•	•	↓	↓	↓	•
INCB	•	•	↓	↓	↓	•
LDAA	•	•	↓	↓	R	•
LDAB	•	•	↓	↓	R	•
LDD	•	•	↓	↓	R	•
LSL	•	•	↓	↓	↓	↓
LSLA	•	•	↓	↓	↓	↓
LSLB	•	•	↓	↓	↓	↓
LSLD	•	•	↓	↓	↓	↓
LSR	•	•	R	↓	↓	↓
LSRA	•	•	R	↓	↓	↓
LSRB	•	•	R	↓	↓	↓
LSRD	•	•	R	↓	↓	↓
MUL	•	•	•	•	•	↓
NEG	•	•	↓	↓	↓	↓
NEGA	•	•	↓	↓	↓	↓
NEGB	•	•	↓	↓	↓	↓
NOP	•	•	•	•	•	•
ORAA	•	•	↓	↓	R	•
ORAB	•	•	↓	↓	R	•

MNEM	H	I	N	Z	V	C
PSHA	•	•	•	•	•	•
PSHB	•	•	•	•	•	•
PULA	•	•	•	•	•	•
PULB	•	•	•	•	•	•
ROL	•	•	↓	↓	↓	↓
ROLA	•	•	↓	↓	↓	↓
ROLB	•	•	↓	↓	↓	↓
ROR	•	•	↓	↓	↓	↓
RORA	•	•	↓	↓	↓	↓
RORB	•	•	↓	↓	↓	↓
SBA	•	•	↓	↓	↓	↓
SBCA	•	•	↓	↓	↓	↓
SBCB	•	•	↓	↓	↓	↓
STAA	•	•	↓	↓	R	•
STAB	•	•	↓	↓	R	•
STD	•	•	↓	↓	R	•
SUBA	•	•	↓	↓	↓	↓
SUBB	•	•	↓	↓	↓	↓
SUBD	•	•	↓	↓	↓	↓
TAB	•	•	↓	↓	R	•
TBA	•	•	↓	↓	R	•
TAP	↓	↓	↓	↓	↓	↓
TPA	•	•	•	•	•	•
TST	•	•	↓	↓	R	R
TSTA	•	•	↓	↓	R	R
TSTB	•	•	↓	↓	R	R

Explication des abréviations utilisées :

HX = code machine traduit en valeur hexadécimale
 MNEM = mnémonique de l'ASSEMBLEUR d'ALICE
 ADR = mode d'adressage
 # = nombre total d'octets (instruction, adresse ou donnée comprise)
 Adressage :
 DIR = direct (adresse page zéro sur un octet)
 ETE = étendu (adresse sur 2 octets)
 IMM = immédiat (sur 1 octet : opération sur un registre simple ; sur 2 octets : opération sur un registre double)
 IND = indexé (indexation par IX)
 INH = inhérent
 REL = relatif (déplacement sur un octet signé)

HX MNEM ADR #	HX MNEM ADR #	HX MNEM ADR #	HX MNEM ADR #
00	10 SBA INH 1	20 BRA REL 2	30 TSX INH 1
01 NOP INH 1	11 CBA INH 1	21 BRN REL 2	31 INS INH 1
02	12	22 BHI REL 2	32 PULA INH 1
03	13	23 BLS REL 2	33 PULB INH 1
04 LSRD INH 1	14	24 BCC REL 2	34 DES INH 1
05 ASLD INH 1	15	25 BCS REL 2	35 TXS INH 1
06 TAP INH 1	16 TAB INH 1	26 BNE REL 2	36 PSHA INH 1
07 TPA INH 1	17 TBA INH 1	27 BEQ REL 2	37 PSHB INH 1
08 INX INH 1	18	28 BVC REL 2	38 PULX INH 1
09 DEX INH 1	19 DAA INH 1	29 BVS REL 2	39 RTS INH 1
0A CLV INH 1	1A	2A BPL REL 2	3A ABX INH 1
0B SEV INH 1	1B ABA INH 1	2B BMI REL 2	3B RTI INH 1
0C CLC INH 1	1C	2C BGE REL 2	3C PSHX INH 1
0D SEC INH 1	1D	2D BLT REL 2	3D MUL INH 1
0E CLI INH 1	1E	2E BGT REL 2	3E WAI INH 1
0F SEI INH 1	1F	2F BLE REL 2	3F SWI INH 1
40 NEGA INH 1	50 NEGB INH 1	60 NEG IND 2	70 NEG ETE 3
41	51	61	71
42	52	62	72
43 COMAINH 1	53 COMBINH 1	63 COM IND 2	73 COM ETE 3
44 LSRA INH 1	54 LSRB INH 1	64 LSR IND 2	74 LSR ETE 3
45	55	65	75
46 RORA INH 1	56 RORB INH 1	66 ROR IND 2	76 ROR ETE 3
47 ASRA INH 1	57 ASRB INH 1	67 ASR IND 2	77 ASR ETE 3
48 ASLA INH 1	58 ASLB INH 1	68 ASL IND 2	78 ASL ETE 3
49 ROLA INH 1	59 ROLB INH 1	69 ROL IND 2	79 ROL ETE 3
4A DECA INH 1	5A DECB INH 1	6A DEC IND 2	7A DEC ETE 3
4B	5B	6B	7B
4C INCA INH 1	5C INCB INH 1	6C INC IND 2	7C INC ETE 3
4D TSTA INH 1	5D TSTB INH 1	6D TST IND 2	7D TST ETE 3
4E	5E	6E JMP IND 2	7E JMP ETE 3
4F CLRA INH 1	5F CLRB INH 1	6F CLR IND 2	7F CLR ETE 3

HX MNEM ADR #	HX MNEM ADR #	HX MNEM ADR #	HX MNEM ADR #
80 SUBA IMM 2	90 SUBA DIR 2	A0 SUBA IND 2	B0 SUBA ETE 3
81 CMPA IMM 2	91 CMPA DIR 2	A1 CMPA IND 2	B1 CMPA ETE 3
82 SBCA IMM 2	92 SBCA DIR 2	A2 SBCA IND 2	B2 SBCA ETE 3
83 SUBD IMM 3	93 SUBD DIR 2	A3 SUBD IND 2	B3 SUBD ETE 3
84 ANDA IMM 2	94 ANDA DIR 2	A4 ANDA IND 2	B4 ANDA ETE 3
85 BITA IMM 2	95 BITA DIR 2	A5 BITA IND 2	B5 BITA ETE 3
86 LDAA IMM 2	96 LDAA DIR 2	A6 LDAA IND 2	B6 LDAA ETE 3
87	97 STAA DIR 2	A7 STAA IND 2	B7 STAA ETE 3
88 EORA IMM 2	98 EORA DIR 2	A8 EORA IND 2	B8 EORA ETE 3
89 ADCA IMM 2	99 ADCA DIR 2	A9 ADCA IND 2	B9 ADCA ETE 3
8A ORAA IMM 2	9A ORAA DIR 2	AA ORAA IND 2	BA ORAA ETE 3
8B ADDA IMM 2	9B ADDA DIR 2	AB ADDA IND 2	BB ADDA ETE 3
8C CPX IMM 3	9C CPX DIR 2	AC CPX IND 2	BC CPX ETE 3
8D BSR REL 2	9D JSR DIR 2	AD JSR IND 2	BD JSR ETE 3
8E LDS IMM 3	9E LDS DIR 2	AE LDS IND 2	BE LDS ETE 3
8F	9F STS DIR 2	AF STS IND 2	BF STS ETE 3
C0 SUBB IMM 2	D0 SUBB DIR 2	E0 SUBB IND 2	F0 SUBB ETE 3
C1 CMPB IMM 2	D1 CMPB DIR 2	E1 CMPB IND 2	F1 CMPB ETE 3
C2 SBCB IMM 2	D2 SBCB DIR 2	E2 SBCB IND 2	F2 SBCB ETE 3
C3 ADDD IMM 3	D3 ADDD DIR 2	E3 ADDD IND 2	F3 ADDD ETE 3
C4 ANDB IMM 2	D4 ANDB DIR 2	E4 ANDB IND 2	F4 ANDB ETE 3
C5 BITB IMM 2	D5 BITB DIR 2	E5 BITB IND 2	F5 BITB ETE 3
C6 LDAB IMM 2	D6 LDAB DIR 2	E6 LDAB IND 2	F6 LDAB ETE 3
C7	D7 STAB DIR 2	E7 STAB IND 2	F7 STAB ETE 3
C8 EORB IMM 2	D8 EORB DIR 2	E8 EORB IND 2	F8 EORB ETE 3
C9 ADCB IMM 2	D9 ADCB DIR 2	E9 ADCB IND 2	F9 ADCB ETE 3
CA ORAB IMM 2	DA ORAB DIR 2	EA ORAB IND 2	FA ORAB ETE 3
CB ADDB IMM 2	DB ADDB DIR 2	EB ADDB IND 2	FB ADDB ETE 3
CC LDD IMM 3	DC LDD DIR 2	EC LDD IND 2	FC LDD ETE 3
CD	DD STD DIR 2	ED STD IND 2	FD STD ETE 3
CE LDX IMM 3	DE LDX DIR 2	EE LDX IND 2	FE LDX ETE 3
CF	DF STX DIR 2	EF STX IND 2	FF STX ETE 3

Il est possible aussi d'utiliser les mnémoniques suivants :

- LSL est équivalent à ASL.
- BHS est équivalent à BCC.
- BLO est équivalent à BCS.

Annexe 3

Pour ceux qui débutent

Programmer en ASSEMBLEUR n'est pas plus compliqué que programmer en BASIC. C'est simplement plus long et plus fastidieux. L'ASSEMBLEUR est le langage évolué le plus proche de celui du microprocesseur, qui est en quelque sorte le cœur d'ALICE : à chaque instruction (ou **mnémonique**) correspond très exactement une seule opération complexe ou simple réalisable par la machine.

Le microprocesseur d'ALICE sait reconnaître un peu plus d'une **centaine d'instructions élémentaires** qui devront lui être données sous forme de suites d'octets, occupant chacun une adresse mémoire. Le rôle d'un ASSEMBLEUR est de traduire les représentations mnémoniques de ces instructions en "code machine".

Par exemple, l'instruction :

LDAA #\$10 sera traduite en : \$86 \$10.

Elle signifie : "mettre dans la case A du microprocesseur la donnée ayant la valeur \$10".

LDAA constitue l'instruction ; il s'agit du **mnémonique** de l'expression américaine "Load A".

Le symbole # indique à l'Assembleur que l'octet à mettre dans A est placé immédiatement après. Le symbole \$ signale que la valeur est exprimée en utilisant la **notation hexadécimale**.

Dans ce système, les nombres sont composés de chiffres qui ne vont pas de 0 à 9 mais de 0 à 15 (on utilise les 6 premières lettres de l'alphabet pour compléter la liste des chiffres : reportez-vous au tableau de la page 39).

Un octet (huit bits) permet de coder des valeurs de 0 à 255 ou \$00 à \$FF, deux octets des valeurs de 0 à (256 x 256) - 1 = 65 535 ou \$00 à \$FFF.

Lorsqu'une donnée numérique (adresse ou valeur) exprimée sur deux octets, par exemple \$F10A, doit être conservée en RAM, elle est stockée en mémoire dans deux adresses consécutives, selon l'ordre suivant :

- le dernier octet (l'octet de poids faible : \$0A) est stocké en premier ;
- le premier octet (l'octet de poids fort : \$F1) est mis à l'adresse du dessus.

L'ensemble des opérations élémentaires que sait faire un microprocesseur sur un ou deux octets à la fois se réduit à :

- des **transferts de données** entre la mémoire et des "cases" du microprocesseur appelées registres, (une position mémoire ne peut stocker qu'un octet, un registre peut être simple ou double : c'est-à-dire contenir soit un soit deux octets),
- des **comparaisons**,
- des **branchements** et des **sauts** dans le programme (comme GOTO, GOSUB et RETURN en BASIC), des décalages ou des rotations (les 8 bits d'un octet contenus dans un registre ou une position mémoire sont poussés d'un cran vers la droite ou vers la gauche... le bit qui est "éjecté" à l'extérieur passe dans le bit C du registre P.

Contrairement au BASIC, l'ASSEMBLEUR ne convertit pas immédiatement le programme que vous écrivez en **code machine** directement exécutable. Vous devez, dans un premier temps, entrer tout votre ensemble d'instructions à l'aide de l'**Editeur** ; puis, dans un second temps, en demander l'**Assemblage** (la conversion). Le résultat peut être sauvegardé sur cassette pour être rechargé et exécuté ultérieurement.

Tout programme en Assembleur doit comporter, en première ligne, l'indication de son **implantation en mémoire** (directive ORG) et de son **adresse d'exécution** (directive EXC).

Exemples de conversion décimal-hexadécimal

Décimal	Hexadécimal	Décimal	Hexadécimal
0	\$00	16	\$10
1	\$01	17	\$11
2	\$02	18	\$12
3	\$03	19	\$13
4	\$04	20	\$14
5	\$05	31	\$1F
6	\$06	32	\$20
7	\$07	33	\$21
8	\$08	34	\$22
9	\$09	48	\$30
10	\$0A	64	\$40
11	\$0B	80	\$50
12	\$0C	96	\$60
13	\$0D	255	\$FF
14	\$0E	256	\$100
15	\$0F	257	\$101

Annexe 4

Quelques routines utiles

Voici quelques adresses qui peuvent vous être utiles pour simplifier la rédaction de vos programmes en ASSEMBLEUR.

Scrutation du clavier :

- adresse F883
- pas de paramètre d'entrée
- sortie : registre A = 0 : pas de touche enfoncée ; sinon A = valeur ASCII de la touche.

Emission d'un caractère :

- adresse : F9C6
- entrée : E8 = 0 : émission sur l'écran
E8 = 1 : émission sur l'imprimante
A = caractère en ASCII
si E8 = 0, position sur l'écran en 3280 (numéro de rangée) et 3281 (numéro de colonne).
- pas de paramètre de sortie.

Emission d'une chaîne de caractères :

- adresse : E7A8
- entrées : comme ci-dessus + registre X = pointeur sur la chaîne de caractères – 1
La chaîne de caractères doit se terminer par 0.

Effacement de l'écran :

- adresse : FBD4.

Transfert d'un tableau dans un autre tableau :

- adresse : F7AD
- entrée : X = pointeur sur le tableau à transférer. Le premier mot du tableau indique sa taille en octets.
D = pointeur sur le tableau destination.

Emission d'un son :

- adresse : FFAB
- entrées : A = fréquence (1 à FF)
B = durée (1 à FF)
- pas de paramètre de sortie.

Annexe 5

Connecter une imprimante

Nous vous conseillons d'utiliser l'imprimante spécialement conçue pour ALICE, qui vous garantira une compatibilité et une fiabilité totales (voir le GUIDE D'ALICE page 17).

A défaut, vous pouvez utiliser une imprimante série correspondant aux normes logicielles spécifiées page 177 du guide DÉCOUVREZ LE BASIC, par exemple une OKI MICROLINE 82 A ou une SEIKOSHA GP 100.

Voici le schéma du câble que vous utiliserez pour raccorder ALICE et l'imprimante :

Côté ALICE :	Côté imprimante:
Prise DIN 4 broches	Prise CANON 25 broches
1 (non utilisé)	
2 détection porteuse	11
3 terre	7
4 transmission de données	3
	relier 6 et 20.

Annexe 6

Présentation du listing

Si vous demandez l'édition d'un listing, voici quel sera son format :

1. l'**en-tête** rappelant l'usage de l'Editeur-Assembleur ;
2. les **erreurs** détectées pendant la première passe (voir page 45, le format d'une ligne d'erreur) ;
3. le **listing** comprenant le fichier source, et des indications sur le fichier objet ;
4. le **nombre d'erreurs** détectées au cours de chacune des 2 passes ;
5. la **liste des symboles** utilisés, avec leur valeur en hexadécimal (cette liste est établie selon leur ordre d'apparition) ;
6. un **saut de page**, si le support du listing est l'imprimante.

Une ligne de listing se présente de la manière suivante :

1. n° de la ligne ;
2. s'il y a une erreur :
le mot **ERREUR** suivi d'une lettre (voir la signification de cette lettre page 45) ;
s'il n'y a pas d'erreur :
 - l'adresse de l'instruction,
 - la valeur en hexadécimal de l'instruction (jusqu'à 6 caractères) ;
3. la ligne du programme source.

*Remarque : Vous pouvez demander l'arrêt temporaire du défilement du listing en appuyant sur **SHIFT** et **@**. Le listing redémarre en appuyant sur n'importe quelle touche du clavier.*

Annexe 7

Un exemple de programme

ASSEMBLEUR ALICE REV 1.00%
COPYRIGHT MATRA, 1984

```
1      ; ! ----- !
2      ; !
3      ; !   Programme d'essai   !
4      ; !   pour l'Assembleur. !
5      ; !
6      ; !   Ce programme permet d'emettre   !
7      ; !   des sons, impossibles a         !
8      ; !   obtenir en BASIC, pour          !
9      ; !   animer des jeux video.          !
10     ; !
11     ; !   Attention: ce programme boucle   !
12     ; ! ----- !
13
14             ORG   $4800
15             EXC   INIT
16
17 4800 CE0001 INIT   LDX   #$0001
18 4803 FF4827       STX   COMPT
19
20 4806 4F          DEBUT CLRA
21 4807 C630       LDAB  #$30
22
23 4809 FE4827 BRUIT  LDX   COMPT
24 480C 8880       EORA  #$80
25 480E B7BFFF     STAA  $BFFF ; ADRESSE QUI FAIT DU
                                BRUIT
```

Annexe 8

Codes erreur

I CODE ERREUR DE L'ÉDITEUR

Certaines commandes de l'ÉDITEUR provoquent l'affichage, sur la dernière ligne, d'un message d'erreur, accompagné d'un petit "bip" sonore. Le message se compose du mot "ERREUR", suivi d'un chiffre dont voici la signification :

ERREUR 0 : – Type du fichier incorrect. Le fichier que vous avez lu n'a pas été créé par l'ÉDITEUR (c'est peut-être un fichier en BASIC, un fichier de données, ou un programme exécutable).

– Les fonctions concernées par cette erreur sont :
`CTRL M`, `CTRL V` et `CTRL 4`.

ERREUR 1 : – Erreur de lecture du fichier sur cassette. Une donnée erronée a été lue.

– Les fonctions concernées par cette erreur sont :
`CTRL M`, `CTRL V` et `CTRL 4`.

ERREUR 2 : – Erreur de lecture du fichier sur cassette. Une erreur a été détectée dans la mémoire d'ALICE.

– Les fonctions concernées par cette erreur sont :
`CTRL M`, `CTRL V` et `CTRL 4`.

ERREUR 3 : – Erreur de lecture du fichier sur cassette. La taille du fichier à lire excède la taille de la mémoire de votre ALICE.

– Les fonctions concernées par cette erreur sont :
`CTRL M`, `CTRL V` et `CTRL 4`.

ERREUR 4 : – Vous avez atteint la limite de la mémoire disponible sur votre ALICE.

– Les fonctions concernées par cette erreur sont :
`CTRL W`, `CTRL Z` et `ENTER`.

ERREUR 5 : – La commande que vous venez de demander est interdite car vous n'avez pas validé la ligne en cours. Les modifications relatives à cette ligne sont perdues.

– Les fonctions concernées par cette erreur sont :
`CTRL A`, `CTRL F`, `CTRL H`, `CTRL I`, `CTRL L`,
`CTRL M`, `CTRL V`, `CTRL 1`, `CTRL 3`, `CTRL 4`,
`CTRL 6` et `CTRL 8`.

```

26
27 4811 09      BOUCL DEX
28 4812 26FD      BNE BOUCL
29 4814 5A      DECB
30 4815 26F2      BNE BRUIT
31
32              CHANG              ; CHANGEMENT DE
                                   FREQUENCE

33 4817 FE4827      LDX  COMPT
34 481A C605      LDAB #$5
35 481C 3A      ABX
36 481D FF4827      STX  COMPT
37 4820 8C0200      CPX  #$200
38 4823 2EDB      BGT  INIT
39 4825 20DF      BRA  DÉBUT
40
41              ; DONNEES DU PROGRAMME
42
43 4827 0000      COMPT DFD  $0
44
45              ; FIN
  
```

0 ERREUR(S) PASSE 1

0 ERREUR(S) PASSE 2

SYMBLES:

INIT = 4800 DEBUT = 4806 BRUIT = 4809 BOUCL = 4811
 CHANG = 4817 COMPT = 4827

ERREUR 6 : – La chaîne de caractères que vous recherchez est inexistante dans la suite du fichier.
– La fonction concernée par cette erreur est :
CTRL F.

ERREUR 7 : – Le chiffre entré est incorrect.
– La fonction concernée par cette erreur est :
CTRL L.

II CODE ERREUR DE L'ASSEMBLEUR

Lorsque l'ASSEMBLEUR rencontre une erreur, la ligne erronée est affichée sur l'écran (si vous n'avez pas demandé de listing, ou si vous l'avez demandé sur l'écran), ou sur l'imprimante (si vous avez demandé le listing sur l'imprimante).

L'erreur détectée est identifiée par une lettre dont voici la signification :

ERREUR A : plus de place pour ajouter de nouveaux symboles.

ERREUR B : symbole déjà défini.

ERREUR C : valeur supérieure à 255 (adressage direct...).

ERREUR D : expression incomplète.

ERREUR E : expression inexistante.

ERREUR F : expression erronée.

ERREUR G : symbole inexistant.

ERREUR H : instruction inconnue.

ERREUR I : registre inconnu.

ERREUR J : adressage indexé incorrect.

ERREUR K : adressage relatif incorrect (overflow).

ERREUR L : ORG déjà défini.

ERREUR M : plus de place pour le programme objet.

ERREUR N : compteur ordinal indéfini (pas de directive ORG).

ERREUR O : instruction incomplète.

ERREUR P : instruction inexistante (STAA #...).