

Manuel de l'assembleur 6809 du T07/T07.70

Michel Weissgerber

cedic/nathan

Illustrations : Annicka Boyriven

Ce volume porte la référence
ISBN 2-7124-0566-8

Toute reproduction, même partielle, de cet ouvrage est interdite. Une copie ou reproduction par quelque procédé que ce soit, photographie, photocopie, microfilm, bande magnétique, disque ou autre, constitue une contrefaçon passible des peines prévues par la loi du 11 mars 1957 sur la protection des droits d'auteur.

© CEDIC 1984
CEDIC, 32, boulevard Saint-Germain, 75005 - PARIS

SOMMAIRE

Avant-propos	7
1 - Organisation de la cartouche Assembleur	9
1.1 - Architecture	10
1.2 - Menu	11
1.3 - Descripteur de fichier	12
Nom de fichier	12
2 - L'éditeur	13
Accès à l'éditeur	15
Écriture d'un programme source	16
Les programmes d'essai	21
Exemple : « EQUATES »	22
Exemple : « MOIRAGE »	24
2.1 - Commandes d'édition	26
Déplacement du curseur (vers le haut)	26
(vers le bas)	27
(vers la gauche)	28
(vers la droite)	28
Espace	29
Entrée	29
Insertion d'un caractère	30
Effacement d'un caractère	31
Descente d'une page	32
Remontée d'une page	33
Effacement d'une ligne	34
Positionnement du curseur (en fin de ligne)	35
(au début de la ligne)	35
Caractères spéciaux	36
Affichage d'un espace en vidéo inverse	36
Commande de changement d'état (mode édition-commande éditeur)	37

2.2 - Commandes éditeur	38
TOP (début du programme source résidant dans l'éditeur)	38
BOTTOM (fin du programme source résidant dans l'éditeur)	39
OFF (suppression de la tabulation automatique)	39
ZONE (sélectionne une zone mémoire dans l'éditeur)	40
COPY (sauvegarde une partie du programme source)	41
INSERT (insertion dans le programme source)	42
DELETE (effacement d'une ligne ou de la zone)	43
FIND (recherche d'une chaîne de caractères)	44
REPLACE (remplacement d'une chaîne de caractères)	46
PRINT (impression sur l'imprimante)	49
NEW (effacement)	50
EXIT (Passage sous contrôle du moniteur)	51
QUIT (retour sous contrôle de menu)	52
2.3 - Commandes de gestion de fichiers sous éditeur	53
LOAD (chargement dans l'éditeur d'un programme source)	53
SAVE (sauvegarde le programme source résidant)	55
MERGE (fusionne un fichier avec le programme source)	56
VERIFY (vérification du contenu de l'éditeur)	57
END OF FILE (positionnement du LEP)	58
3 - Assemblage	59
ASSEMBLAGE (assemblage du programme source)	61
3.1 - Options d'assemblage	64
/WE (arrête l'assemblage si une erreur est détectée)	64
/NL (supprime l'affichage des listings)	65
/LP (commutation de l'imprimante)	66
/NO (suppression du programme objet en mémoire)	66
/NS (suppression de la table des symboles)	67
/SS (édition sur des lignes séparées)	67
3.2 - Directives d'assemblage	68
EQU (affectation d'une valeur à un symbole)	68
SET (affectation temporaire d'une valeur à un symbole)	69
RMB (réservation d'octets en mémoire)	70
ORG (initialisation du compteur de programme)	71
FCB (définition d'une constante d'un octet)	73
FDB (définition d'une constante de deux octets)	74
FCC (définition d'une constante chaîne de caractères)	75
SETDP (positionnement de la page directe « Ø »)	76
PAGE (saut de page)	78

TITLE (définition d'un titre)	78
INCLUD (inclusion d'un programme source)	79
END (fin du programme source)	80
Où est situé le programme objet ?	
Quelle est la fin du programme source ?	81
4 - Le moniteur	86
Accès au moniteur	86
4.1 - Commandes moniteur	88
INPUT (entrée)	88
(ADRESSE) / (examen du contenu d'un emplacement mémoire) ...	89
MODE NUMÉRIQUE	91
OUTPUT (sortie)	92
MODE ASCII	93
MODE MNÉMONIQUE	94
INDIRECTION (introduit un niveau d'indirection dans l'examen des mémoires)	95
MODE CALCULATEUR (évalue une expression)	97
GO (lance l'exécution d'un programme)	98
BREAK (positionne un point d'arrêt)	99
CONTINUE (continue l'exécution d'un programme)	100
BREAK POINT (affiche les points d'arrêt)	101
REGISTER (affiche le contenu des registres)	102
YANK (suppression des points d'arrêt)	103
DUMP (visualise un bloc de mémoire)	104
TRACE (visualise l'état des registres et l'instruction en cours) ...	105
WRITE (visualise l'état des registres et l'instruction en cours du programme principal)	107
PRINT (commute l'affichage sur l'imprimante)	108
SAVE (sauvegarde du programme objet)	109
LOAD (chargement d'un programme objet)	110
VERIFY (compare le contenu d'un fichier avec la mémoire)	111
QUIT (retour sous contrôle de menu)	112
EXIT (passage sous contrôle de l'éditeur)	113
5 - Gestion de fichiers	114
DIRECTORY (catalogue d'une disquette)	114
COPY (copie d'un fichier)	116
RENAME (change le nom d'un fichier)	117
KILL (suppression d'un fichier)	118
FORMAT (initialisation d'une disquette)	119

PRINTER COLUMNS (déclare le nombre de colonnes de l'imprimante)	120
Format d'un fichier objet	121
Annexe A : Points d'entrée du moniteur TO7 et TO7-70	122
Annexe B : Microprocesseur 6809 (instructions et adressage) ..	152
Annexe C : Table ASCII	175
Annexe D : Messages d'erreur	177
Annexe E : Programmes types et utilisation des banques mémoires du TO7-70	180

AVANT-PROPOS

Vous venez d'acquérir ou de vous faire offrir une cartouche ASSEMBLEUR TO7/TO7-70.

Ce module est un nouvel atout pour votre ordinateur.

Avec cette cartouche "Module Langage 6809" vous allez pouvoir écrire des programmes dans un langage aussi près que possible de la machine.

Avant d'aller plus avant, il faut que vous compreniez que le « cœur » ou le « cerveau » de votre TO7 ou TO7-70 est un microprocesseur 6809.

Le 6809 ne comprend qu'un seul langage, le sien, c'est le langage machine.

Le langage machine est constitué d'une succession de codes binaires, exprimés en hexadécimal, dont le regroupement, par paquets de un, deux, trois, quatre ou cinq valeurs, est appelé instruction. La succession d'instructions constitue le programme machine ou programme objet.

Les programmes écrits en langages évolués (BASIC, FORTH, LOGO...) pour être exécutables, c'est-à-dire compréhensibles par le 6809, doivent être traduits en programme machine.

Pour cette tâche on peut retenir deux solutions :

1 - Traduire le programme évolué, en totalité, avant de l'exécuter en machine. Ce travail est effectué par un autre programme que l'on appelle un compilateur. On dit alors que le programme est compilé puis exécuté.

2 - Traduire le programme ligne par ligne ou instruction par instruction avant l'exécution en machine des codes binaires correspondants.

On dit alors que le langage évolué est interprété.

Le BASIC, le FORTH, le LOGO du TO7 sont des langages interprétés.

La première solution implique un traitement de traduction avant l'exécution du programme. Mais une fois ce travail effectué, une nouvelle exécution ne nécessite plus de compilation.

La seconde solution nécessite une traduction de chaque instruction avant son exécution, d'où une vitesse d'exécution réduite.

C'est principalement l'amélioration de ce paramètre que vous propose votre **ASSEMBLEUR**.

La vitesse d'exécution d'une instruction est un paramètre important. L'augmentation de la vitesse d'exécution va vous permettre d'écrire des programmes (d'animation par exemple) qui n'auraient aucun sens dans un langage interprété car leur exécution serait trop lente. Le rôle de l'**ASSEMBLEUR** est de traduire la représentation mnémonique des instructions en leur équivalent binaire qui peut comporter entre un et cinq octets. Un octet est un mot de huit bits. Ce code binaire résultant, appelé **programme objet**, est directement exécutable par le microprocesseur. Le programme écrit en mnémonique est appelé **programme source**.

Cette tâche, que l'on appelle « assemblage », est réalisée par l'**ASSEMBLEUR**.

La cartouche « MEMO7 » contient également un **EDITEUR** destiné à écrire les programmes source, ainsi qu'un **MONITEUR** qui vous permettra de vérifier le fonctionnement du programme objet avant de l'exécuter ou de l'appeler, éventuellement sous contrôle d'un autre langage.

Le langage assembleur reste très près de la machine. Il dépend des limites du microprocesseur employé. C'est pourquoi nous pensons qu'il est nécessaire de bien comprendre la structure interne ainsi que le mode d'adressage du microprocesseur utilisé pour programmer de façon efficace en assembleur.

Cet ouvrage, qui est un manuel de base, et non pas un cours sur l'assembleur, décrit d'une façon détaillée les diverses commandes à votre disposition.

Chaque commande est appliquée, à titre d'exemple, sur un programme type utilisé dans l'ensemble du livre.

En annexe vous trouvez les points d'entrée du moniteur système, les instructions et le mode d'adressage du 6809.

Enfin un conseil : soyez patient ! L'écriture d'un programme en assembleur est un peu plus compliquée qu'en BASIC, mais va vous ouvrir des possibilités que vous ne soupçonnez pas.

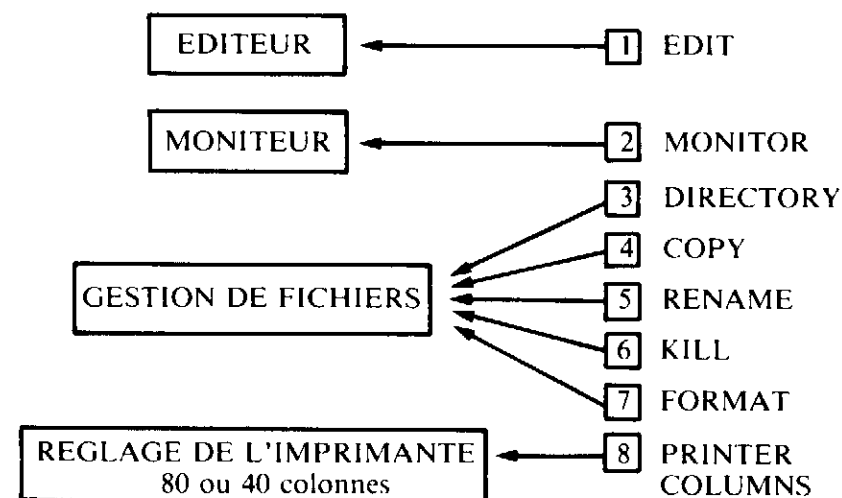
BONNE PROGRAMMATION (en assembleur !!).

CHAPITRE 1 ORGANISATION DE LA CARTOUCHE ASSEMBLEUR

La cartouche Assembleur traite les programmes source écrits en langage d'assemblage 6809. Elle traduit les instructions source en un programme objet compatible avec le microprocesseur 6809.

Le MENU donne accès à quatre blocs de programmes qui sont : **ÉDITEUR**, **MONITEUR**, **GESTION DE FICHIERS**, **RÉGLAGE IMPRIMANTE**.

Leurs points d'entrée par le clavier sont les suivants :



La ligne de commande **EDIT** permet l'accès à l'éditeur et à la fonction **ASSEMBLEUR** (résidant dans l'éditeur).

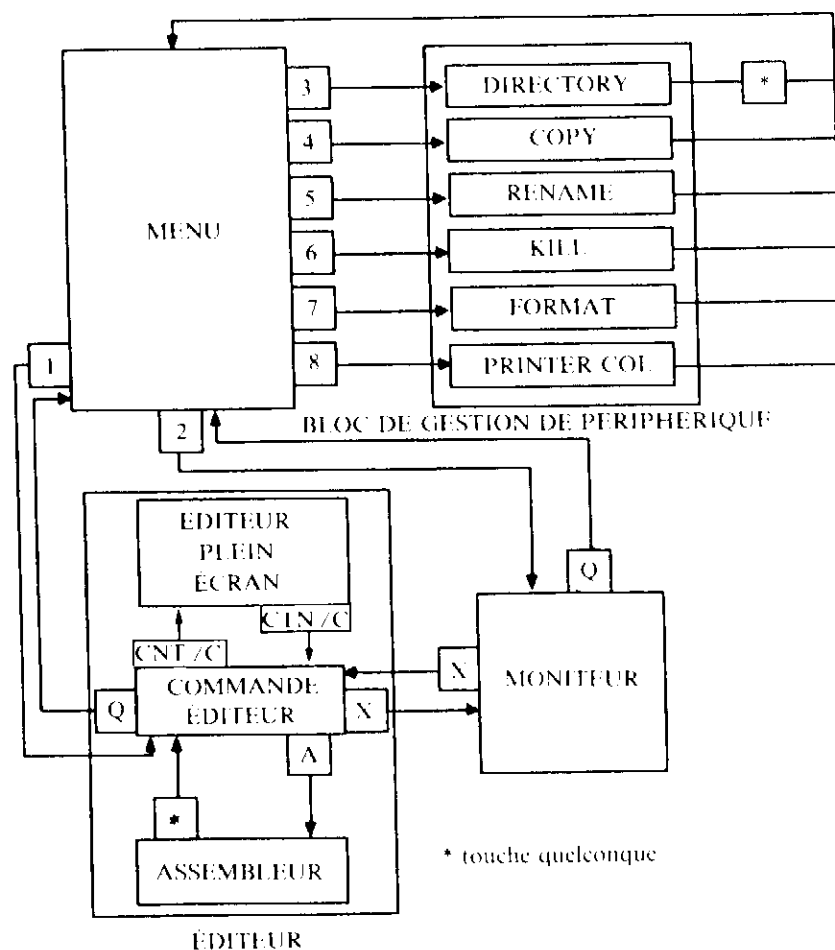
La ligne de commandes **MONITOR** permet l'accès au moniteur.

Le bloc **GESTION DE FICHIERS** est constitué de cinq routines accessibles par le MENU.

- **DIRECTORY** (catalogue du disque)
- **COPY** (copie d'un fichier)
- **RENAME** (renomme un fichier)
- **KILL** (efface un fichier)
- **FORMAT** (initialise un disque)

La ligne de commande **PRINTER COLUMNS** permet de déclarer la largeur du listing sur l'imprimante.

1.1 Architecture



1.2 Menu

Le choix dans MENU est le suivant :

- | | |
|-------------|-------------------|
| 1 EDIT | 5 RENAME |
| 2 MONITOR | 6 KILL |
| 3 DIRECTORY | 7 FORMAT |
| 4 COPY | 8 PRINTER COLUMNS |

L'appel d'une commande de MENU se fait en tapant le numéro correspondant.

En bas de l'écran, la ligne rouge est la zone commentaires de MENU, destinée à recevoir les messages d'erreur ou les questions à destination de l'utilisateur.

Exemple : "Are you sure Y/N?"

(Êtes-vous sûr O/N ?)

(Pour valider l'effacement d'un fichier)

ENTREE valide les commandes de MENU.

CNT C permet l'annulation d'une commande.

COPY, **RENAME**, **KILL** nécessitent, au moins, un descripteur de fichier avant la validation de la commande. La présence d'un descripteur n'est pas obligatoire pour entrer dans **MONITOR** et **EDIT**.

Le chiffre correspondant à votre choix sur MENU fait apparaître un carré bleu (curseur), au début d'une zone jaune délimitant le format (c'est-à-dire la longueur) du descripteur de fichier.

EDIT et **MONITOR** appellent leur écran spécifique.

EDIT copie le descripteur de fichier entré (complété éventuellement du suffixe et du périphérique) sur la ligne commande éditeur.

1.3 Descripteur de fichier

La syntaxe générale des descripteurs de fichiers est la suivante :

<descripteur de fichier> = <périphérique> : <nom du fichier>

Comme <périphérique> on peut avoir :

- C : pour le lecteur-enregistreur de programmes (LEP)
- 0 : pour le premier lecteur de disquette
- 1 : pour le second lecteur de disquettes
- 2 : pour le troisième lecteur de disquettes
- 3 : pour le quatrième lecteur de disquettes

Par défaut, (si <périphérique> est absent dans le <descripteur de fichier>) le système prend le LEP comme périphérique, si aucun lecteur de disquettes n'est présent.

Si un lecteur de disquettes est présent, le lecteur 0 est pris par défaut.

Nom de fichier : le <nom de fichier> est constitué du nom lui-même et d'un suffixe optionnel.

<nom de fichier> = <nom> . <suffixe>

<nom> est composé de un à huit caractères au maximum.

<suffixe> est optionnel et comporte un à trois caractères.

<nom> et <suffixe> doivent être séparés par un point.

Par défaut (en l'absence d'un autre suffixe), le système ajoute

.ASM pour les fichiers source

.BIN pour les fichiers objet

Exemples de descripteurs de fichier :

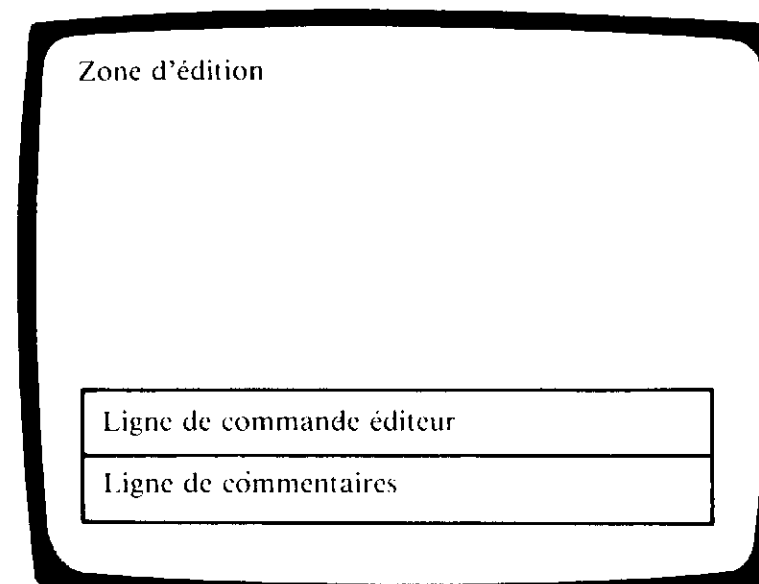
2 : ASSEMBLE.BIN	1 : ESSAI.T07
0 : TRAIN.BUS	C : PARIS.V

CHAPITRE 2 L'ÉDITEUR

L'éditeur permet l'écriture et l'assemblage d'un programme source. Il permet, en phase d'édition, la correction des erreurs, l'insertion de nouvelles lignes, la recherche d'étiquettes, le transfert de blocs de programme d'un/vers un périphérique etc.

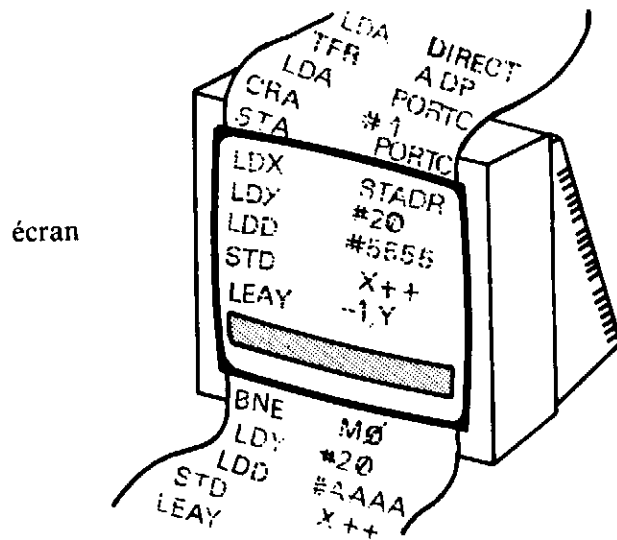
L'éditeur utilise l'écran dans sa totalité et accepte, en entrée, des lignes de programmes source ou des commandes. Il génère en sortie, après assemblage, le programme objet. Les lignes de programmes sont entrées en mode ÉDITION. Les commandes sont entrées en mode COMMANDES ÉDITEUR.

L'écran sous contrôle de l'éditeur est divisé en trois zones :



A - Zone d'ÉDITION : où l'éditeur plein écran est disponible (de la ligne 0 à la ligne 23, caractères bleus sur fond noir). La partie de programme affichée à l'écran est une fenêtre sur le programme source résidant dans la mémoire.

PROGRAMME SOURCE



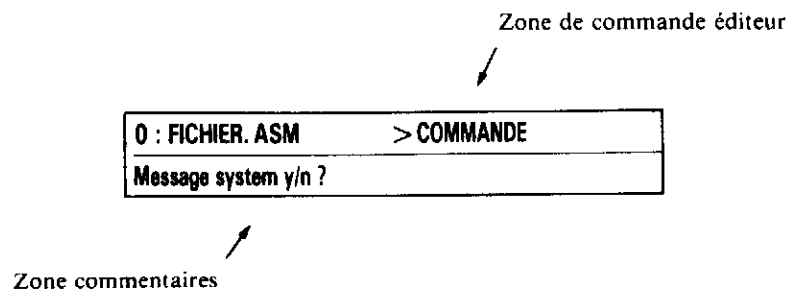
B - Zone de COMMANDE ÉDITEUR : (ligne 24 - Fond jaune, texte bleu). La zone de commande éditeur est divisée en deux parties par « > ».

— La partie gauche affiche le descripteur de fichier en cours de traitement.

— La partie droite reçoit les commandes éditeur

C - Zone COMMENTAIRES : (ligne 25 - Fond rouge, écriture noire). La zone commentaires est destinée à recevoir les questions à destination de l'utilisateur ou les messages d'erreur.

Exemple : "Printer ready Y/N" ou "Bad command".



Accès à l'éditeur : la commande de MENU **1** active le bloc ÉDITEUR.

A - Sans descripteur de fichier : édition d'un nouveau programme ou d'un programme déjà résidant en mémoire.

1 EDIT ENTREE

L'écran de l'ÉDITEUR est généré et les vingt-trois premières lignes du programme source, éventuellement résidant en mémoire, sont affichées. Le curseur se place en 0,0 (en haut et à gauche). Le système attend une entrée sur la ligne de commande éditeur.

B - Avec descripteur de fichier : chargement et édition d'un programme.

1 EDIT ENTREE

ou **1** EDIT ENTREE

ou **1** EDIT ENTREE

Note : C et . ASM sont facultatifs et sont pris par défaut (si on travaille avec un LEP).

L'écran de l'éditeur est généré comme ci-dessus.

Le descripteur de fichier, complété éventuellement du périphérique et du suffixe, est affiché dans la partie gauche de la ligne de commande éditeur.

Exemple :

Le programme se charge, le curseur se place en 0,0 et les vingt-trois premières lignes du programme source s'affichent sur l'écran. Une entrée est attendue sur la ligne de commande éditeur.

Q **ENTREE** Redonne la main à MENU
X **ENTREE** Donne accès au moniteur

CNT C Permet le passage du mode ÉDITION en mode COMMANDE ÉDITEUR et réciproquement.

Note : seuls les fichiers source peuvent être chargés par l'éditeur. Un appel de fichier binaire sous éditeur provoque l'erreur "File format error".

Écriture d'un programme source : l'écriture d'un programme source est réalisée sous contrôle de l'éditeur, en mode ÉDITION.

Pour placer le système sous contrôle de l'éditeur, vous disposez de deux commandes.

— A partir de MENU : Tapez **1** puis **ENTREE**

— A partir du MONITEUR : Tapez **X** puis **ENTREE**


Vous êtes maintenant sous contrôle de l'éditeur.

Un carré bleu (curseur) dans la zone de commande éditeur indique que vous êtes en mode COMMANDE ÉDITEUR.

La commande **CNT C** permet le passage du mode COMMANDE ÉDITEUR au mode ÉDITION

Une nouvelle commande **CNT C** redonne la main au mode COMMANDE ÉDITEUR.

Nous vous invitons à essayer cette commande.

CNT C	>	ÉDITION
CNT C	> 	COMMANDE ÉDITEUR
CNT C	>	ÉDITION

Remarquez que le carré bleu (curseur) disparaît de la ligne de commande en mode ÉDITION. Cela vous permettra de repérer « d'un coup d'œil » le mode en cours dans l'éditeur.

Placez-vous en mode ÉDITION (carré bleu absent) : le curseur est actif dans la zone d'édition.

Chaque ligne de programme source est formée de quatre champs : Une étiquette (ou le caractère (*)) pour les lignes de commentaires), un code opération, un opérande et un commentaire.

[Étiquette]	[Code opération]	[Opérande]	[Commentaire]
0 6	7 13	14 20	22 39

Le champ étiquette

Le champ étiquette est le premier champ d'une ligne de programme source. Il est défini entre les colonnes 0 et 6 et il peut prendre l'une des formes suivantes :

1. Un astérisque (*) placé en tête du champ étiquette signifie que le reste de la ligne du programme source est un commentaire. Les commentaires sont ignorés par l'assembleur, et ne sont présents dans le programme source que pour informer l'utilisateur.
2. Un caractère en tête de la ligne signifie que celle-ci possède une étiquette. Les caractères sont les lettres majuscules de A à Z et les chiffres de 0 à 9. Les caractères spéciaux : point (.), dollar (\$) etc. ne sont pas admis. Les étiquettes se composent de un à six caractères. Le premier doit être obligatoirement un caractère alphabétique. Certaines étiquettes sont réservées à l'assembleur et ne doivent pas être utilisées dans le champ étiquette. Dans le cas où on ne respecterait pas cette règle, une erreur serait générée. Les étiquettes réservées sont A, B, D, X, Y, CC, DP, PC, S, U qui représentent les registres du 6809.
3. Un espace en tête de la ligne source indique que le champ étiquette est vide. La ligne n'a alors pas d'étiquette et ne peut être un commentaire.

Une étiquette ne peut être présente qu'une seule fois dans le champ étiquette, sauf lorsqu'elle est utilisée avec la directive d'assemblage SET. Si une étiquette est utilisée plusieurs fois, chaque passage sur cette étiquette sera signalé par l'erreur "Multiply defined symbol".

Le champ code opération

Le champ code opération se trouve après le champ étiquette entre les colonnes 7 et 13. Il doit être précédé d'au moins un espace et contenir le mnémotique d'une instruction 6809 ou une directive d'assemblage. Il est composé uniquement des lettres majuscules de A à Z. Le contenu du champ code opération peut donc être de deux types :

- Mnémotiques : correspondant aux instructions machine du microprocesseur 6809.
- Directives d'assemblage : des codes opération reconnus par l'assembleur, agissant plutôt sur le processus d'assemblage et qui ne sont pas traduites en programme objet.

Le champ opérande

Le champ opérande suit le champ code opération. Il se situe entre les colonnes 14 et 20 mais peut être étendu jusqu'à la colonne 39. Dans tous les cas, un espace doit être utilisé comme séparateur entre le champ opérande et le champ commentaire. L'interprétation du champ opérande dépend du contenu du champ code opération. Il peut contenir une étiquette, un symbole, une expression, un nombre ou une combinaison des quatre. Il sert également à spécifier le mode d'adressage quand le champ code opération contient un mnémonique.

Les étiquettes et les symboles

Les étiquettes et les symboles sont constitués de six caractères au plus et commencent par une lettre. Une valeur entière comprise entre 0 et 65535 (seize bits) leur est associée et vient les remplacer dans l'évaluation d'une expression qui contient les étiquettes ou les symboles spécifiés.

Certains symboles sont spécifiques et ne peuvent être utilisés que dans les expressions. Ce sont :

(*) (astérisque) ou (.) (point), qui représente la valeur courante du compteur de programme (PC).

ENDMEM, qui définit la fin de la mémoire utilisateur.

Les expressions

Les expressions sont composées d'une suite de symboles, de nombres, d'opérateurs logiques ou arithmétiques et de parenthèses. Les expressions dans le champ opérande servent à spécifier une valeur. Les règles de la logique et de l'arithmétique s'appliquent aux expressions.

Exemple :

ENDMEM - \$4000 DEPART + \$30 * < -8

... sont des expressions

Les opérateurs

Les opérateurs dans une expression peuvent être de deux types : logiques ou arithmétiques, et ils sont hiérarchisés de un à cinq. Les opérations d'un niveau supérieur seront effectuées avant les opérations de niveau inférieur. Les opérations de même niveau hiérarchique seront évaluées de la gauche vers la droite. Les résultats intermédiaires et de l'expression sont arrondis à une valeur entière. Les opérateurs peuvent agir sur les nombres et les symboles.

OPÉRATEUR	SYMBOLES	NIVEAU HIÉRARCHIQUE
Les opérateurs arithmétiques		
DIVISION	.DIV. ou /	5
MULTIPLICATION	*	5
ADDITION	+	2
SOUSTRACTION	-	2
MODULO	.MOD.	5
Les opérateurs logiques		
ET logique	.AND. ou &	4
OU inclusif	.OR. ou !	3
Ou exclusif	.XOR.	3
Décalage à droite de n	< - n	5
Décalage à gauche de n	< [+] n	5
EGALITE*	.EQU. ou =	1
INEGALITE**	.NEQ.	1

* La condition vraie rend - 1 (\$FFFF ou \$FF).

** La condition fausse rend 0.

Les nombres

Les nombres représentent des données qui ne varient pas avec l'exécution du programme. Les nombres peuvent être représentés dans trois bases : décimale (base 10), octale (base 8), hexadécimale (base 16). La représentation binaire n'est pas admise. Un nombre sera reconnu comme étant écrit dans une base donnée par la présence d'un préfixe ou d'un suffixe (voir la table ci-dessous). **Sans préfixe ni suffixe, un nombre sera interprété comme étant un nombre décimal par l'assembleur.**

BASE	PRÉFIXE	SUFFIXE	PLAGE
OCTALE	@	Q ou O	0 à 17777
DÉCIMALE	&	T	0 à 65535
HEXADÉCIMALE	\$	H	0 à FFFF


Exemples : ~~2000~~ H = 8192T &5000 = \$1388

Le mode d'adressage

ADRESSAGE	SYNTAXE
Implicite ou inhérent	Pas d'opérande
Direct	< < expression >
Étendu	> < expression >
Immédiat	# < expression >
Étendu indirect	[< expression >]
Indexé	(expression), R
Indexé (8 bits)	< < expression >, R
Indexé (16 bits)	> < expression >, R
Indexé indirect	[< expression >]
Indexé indirect (8 bits)	< [< expression >, R]
Indexé indirect (16 bits)	> [< expression >, R]
Post-incrémenté de 1	, R +
Post-incrémenté de 2	, R + +
Post-incrémenté indirect	[, R + +]
Pré-décrémenté de 1	, - R
Pré-décrémenté de 2	, - - R
Pré-décrémenté indirect	[, - - R]

Le champ commentaire

Le champ commentaire est le dernier d'une ligne de programme source. Il peut commencer à partir du premier espace après le champ opérande. C'est un champ optionnel : il sert à la documentation du programme et est seulement imprimé sur le listing mais n'est pas traduit en langage machine.

Le passage d'un champ à l'autre se fait automatiquement par  (barre d'espacement), si vous n'êtes pas dans le champ commentaire.

Dans le champ commentaire, l'espace reprend sa fonction normale. Sur les champs étiquette, code opération, opérande, les caractères minuscules sont automatiquement remplacés par les majuscules équivalentes. Ainsi vous pouvez taper indifféremment en majuscules ou en minuscules.

Le passage en colonne 37 active la note « DO » pour signaler à l'utilisateur que la fin de la ligne approche (col. 39).

Les programmes d'essai










Nous vous proposons d'écrire deux programmes d'essai :

« MOIRAGE » et « EQUATES »

qui nous serviront de références dans la suite de cet ouvrage. Pour écrire ces programmes vous devrez utiliser les commandes d'édition qui sont des « outils » destinés à écrire et corriger les programmes source. Nous vous invitons à essayer ces « outils » en écrivant les programmes d'essai ci-dessous.

En mode ÉDITION, vous disposez d'un certain nombre de commandes qui sont accessibles par les touches de contrôle du clavier du TO7-70 ou TO7.

Les touches suivantes sont utilisables :

- | | | |
|---------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|------------------------------------------------------------------------|
|  |  | Décalage en haut ou en bas du curseur dans le texte édité. |
|  |  | Décalage à droite ou à gauche du curseur sur une ligne de texte édité. |
|  |  | Insertion ou effacement d'un caractère dans une ligne de texte. |
|  | | Décalage d'une page vers le bas ou vers le haut. |
|  | | Insertion d'une nouvelle ligne dans le texte. |
|  | | Tabulation automatique ou espace. |

Programme « EQUATES »

« EQUATES » est un programme de définition des points d'entrée du moniteur système (T07-70, T07), des adresses et des codes ASCII nécessaires au fonctionnement du programme « MOIRAGE ».

```
*
* POINTS D'ENTREE DU MONITEUR T07
*
INIT EQU $E900 Initialisation.
PUTCH EQU $E903 Affichage.
GETCH EQU $E906 Clavier.
KTST EQU $E909 Test clavier.
DRAW EQU $E90C Ligne.
PLOT EQU $E90F Point.
RSCONT EQU $E912 RS-232.
K7CONT EQU $E915 Lecteur de K7.
GETLP EQU $E918 Crayon optique.
LPINT EQU $E91B Interrupteur.
NOTE EQU $E91E Musique.
GETPT EQU $E921 Lecture point.
GETSC EQU $E924 lecture caractere.
JOYSTK EQU $E927 Manche a balais.
DKCONT EQU $E92A Controleur disque.
```

```
*
* ADRESSES PHYSIQUES
*
```

```
PORTC EQU $E7C3 Le bit 0 du port C
* controle la memoire ecran accedee :
* 1 = points ; 0 = couleur.
```

```
STADR EQU $4000 Debut de l'ecran.
ENDADR EQU $5F40 Fin de l'ecran + 1
```

```
*
* CODES ASCII
*
```

```
US EQU $1F UNIT SEPARATOR
FF EQU $C FORM FEED
EOT EQU 4 END OF TRANSM
ESC EQU $1B ESCAPE
CR EQU $D CARRIAGE RETURN
LF EQU $A LINE FEED
```

Sauvegarde d'« EQUATES »

La sauvegarde d'« EQUATES » est obtenue en mode COMMANDE ÉDITEUR par l'utilisation de la commande S (SAVE) :

> S0 : EQUATES.ASM

pour la sauvegarde sur la disquette qui se trouve dans l'unité 0.

> SC : EQUATES.ASM

pour la sauvegarde sur le lecteur de cassettes.
Le système vous pose la question :

Cassette Ready Y/N?

Tapez ☒

EQUATES est sauvegardé et le nom du fichier créé est affiché sur la ligne de commande.

C : EQUATES .ASM >

La vérification du fichier sauvegardé sur la cassette peut être obtenue avec la commande V (VERIFY)

Pour cela procédez comme suit :

— Entrez la commande V

C : EQUATES .ASM > V

Le système vous répond en vous posant la question :

Cassette Ready Y/N?

— Rembobinez la cassette

— Répondez ☒

Si « EQUATES » est sauvegardé avec des erreurs le système affichera :

Verification Error

Dans ce cas il faut refaire la sauvegarde d'« EQUATES ».
La commande N (NEW)

<descripteur>	> N
Are you sure Y/N ?	

efface le programme édité (« EQUATES »)

Programme « MOIRAGE »

« MOIRAGE » est un programme d'écriture, en damier, des groupes de points ligne (GPL) en mode forme. Notez que « MOIRAGE » n'intervient pas sur la mémoire couleur qui reste ce qu'elle était.

```
*****MOIRAGE*****
* Programme de moirage de la memoire
* point
      ORG      ENOMEM-$400 1K Reserve
DIRECT EQU    *K-8      Page 0
      SETDP    DIRECT
      TITLE    Balayage Ecran
      INCLUD    EQUATES Fichier* contenant
* les principales adresses d'entree du
* Moniteur TD7
      PAGE
START PSHS     A,B,X,Y,U,DP Sauvegarde
      JSR      INIT      Initialisation
      LDA      #DIRECT Page 0
      TFR      A,DP
      LDA      PORTC      Mise en memoire
* points: Par mise a 1 du bit 0 du port C
      ORA      #1
      STA      PORTC
      LDX      #STADR      Adresse de debut
* de l'ecran
M2     LDY      #20      Compteur colonne:
* On affiche 20 fois 2 octets, soit
* 40 octets par ligne
      LDD      #$5555      Moirage:
* alternance de 1 et 0 sur la ligne
M0     STD      ,X++      Charger l'ecran
      LEAY      -1,Y
      BNE      M0      Repete 20 fois
      LDY      #20      Ligne suivante
      LDD      #$AAAA      Le motif est inver
* se pour decaler les 1 et les 0 d'une
* ligne a l'autre et obtenir un moirage
```

```
M1     STD      ,X++
      LEAY      -1,Y
      BNE      M1      Toujours 20 fois
      CMPX      #ENDADR      Fin d'ecran
      BLS      M2      Sinon on recom
* mence 2 lignes a motifs alteres
      SWI      Retour au moniteur
      END      START
```

* La commande INCLUD ne fonctionne qu'en version disquette. Avec un lecteur de cassette, il faut retaper à la place le contenu du fichier "EQUATES".

Sauvegarde de « MOIRAGE »


La sauvegarde de « MOIRAGE » est obtenue par :


ou	> S0 : MOIRAGE.ASM
	SC : MOIRAGE.ASM
puis (facultatif)	
ou	0 : MOIRAGE.ASM > V
	C : MOIRAGE.ASM > V

2.1 Commandes d'édition

Les commandes d'édition sont des commandes actives en mode ÉDITION dans le champ du programme.

Déplacement du curseur d'une position vers le haut


Commande : La touche 

Effet : La commande  déplace le curseur d'une ligne vers le haut de l'écran. Si le curseur est positionné en ligne 0, le reste du texte visible à l'écran est repoussé d'une position vers le bas, et la ligne qui précédait est affichée en haut de l'écran. Si la ligne 0 est la première du programme source résidant dans l'éditeur, une ligne blanche est ajoutée au début du programme.


Exemple :


```
******MOIRAGE*****  
* Programme de moirage de la memoire  
* point  
  
LDA    PORTC Mise en memoire  
* points:Par mise a 1 du bit 0 du portC  
ORA    #1
```

Tapez 


```
  
*****MOIRAGE*****  
* Programme de moirage de la memoire  
* point  
  
LDA    PORTC Mise en memoire  
* points:Par mise a 1 du bit 0 du portC
```

Déplacement du curseur d'une position vers le bas


Commande : La touche 

Effet : La commande  déplace la curseur d'une ligne vers le bas de l'écran. Le curseur positionné en ligne 23 provoque un défilement de la zone d'édition vers le haut de l'écran et affiche la ligne suivante en ligne 23. Si le curseur est positionné sur la dernière ligne du programme source résidant, après le déplacement le curseur pointera sur une ligne blanche qui ne sera pas ajoutée au texte du programme.

Exemple :

```
STA    PORTC  
LDX    #STADR Adresse de debut  
* de l'ecran  
M2     LDY    #20      Compteur colonne  
* On affiche 20 fois 2 octets,soit  
  
CMPX    #ENDADR Fin d'ecran  
BLS     M2      Sinon on recom  
* mence 2 lignes a motifs alternes  
SWI     Retour au moniteur  
END    START
```

Tapez : 

```
LDX    #STADR Adresse de debut  
* de l'ecran  
M2     LDY    #20      Compteur colonne  
* On affiche 20 fois 2 octets,soit  
* 40 octets par ligne  
  
BLS     M2      Sinon on recom  
* mence 2 lignes a motifs alternes  
SWI     Retour au moniteur  
END    START
```

Déplacement du curseur d'une position vers la gauche

Commande : La touche **←**

Effet : La commande **←** déplace le curseur d'un caractère vers la gauche de l'écran. Si le curseur est positionné sur la colonne 0, une commande **←** le localisera à la fin de la ligne précédente. Si le curseur est situé en ligne 0, un déplacement sur la colonne 0 visualisera une nouvelle ligne en haut de l'écran et positionnera le curseur en ligne 0 et colonne 39.

Exemple :

```
* de l'écran
M2      LC  #20      Compteur colonne:
```

Tapez **←**

```
* de l'écran
M2      LY  #20      Compteur colonne:
```

Déplacement du curseur d'une position vers la droite

Commande : La touche **→**

Effet : La commande **→** déplace le curseur d'un caractère vers la droite de l'écran. Si le curseur est positionné sur la colonne 39, une commande **→** le localisera au début de la ligne suivante. Si le curseur est situé en ligne 23, un déplacement sur la colonne 39 visualisera une nouvelle ligne en bas de l'écran et positionnera le curseur en ligne 23 et colonne 0.

Exemple :

```
INCLUD EQUATES Fichier contenant
* les principales adresses d'entree du
```

Tapez **→**

```
INCLUD EQUATES Fichier contenant
* les principales adresses d'entree du
```

Espace

Commande : La touche (barre d'espace)

Effet : La commande (espace) a plusieurs fonctions :
— Tabulation automatique dans l'écriture d'une ligne de programme source, sauf sur le champ commentaire.

— Un espace dans les zones suivantes :

* ligne de commande éditeur

* lignes de commentaires

* champ commentaire d'une ligne de programme source.

Exemple :

```

DIRECT EQU    *K-8    Page 0
Tapez        DIRECT EQU    *K-8    Page 0
Tapez        DIRECT EQU    *K-8    Page 0
Tapez        DIRECT EQU    *K-8    Page 0
Tapez        DIRECT EQU    *K-8    Page 0
Tapez        DIRECT EQU    *K-8    Page 0
```

Entrée

Commande : A partir du clavier, **ENTREE**

Effet : La commande **ENTREE**, activée dans la zone d'ÉDITION, provoque un saut au début de la ligne suivante en insérant une nouvelle ligne et décale le reste du texte vers le bas. En ligne 23 le texte est poussé vers le haut et le curseur reste positionné au début de la ligne 23. Sur la ligne de commande éditeur la commande : **ENTREE** valide la/les commandes précédentes.

Exemple :

```
* les principales adresses d'entree du
* Moniteur T07
```

Tapez **ENTREE**

```
* les principales adresses d'entree du
* Moniteur T07
```