memento

CLEFS POUR VG 5000

François Normant

Editions du P.S.I.

CLEFS POUR VG 5000

Autres ouvrages relatifs au VG 5000

— 102 programmes pour VG 5000 — Jacques Deconchat
— VG 5000 pour tous — Jean-Michel Jégo

Ouvrages d'initiation

— L'ordinateur apprivoisé — François Picard et Danielle Shaw
— Mon ordinateur — Jean-Claude Barbance
— Visa pour l'informatique — Jean-Michel Jégo
— Visa pour le Basic — Jean-Michel Jégo
— Le logotron informatique — Jean-Pierre Petit
— Minitel mode d'emploi — Dominique Schraen et Sibylle Rochas
— Les mots de la micro — Alan Freedman, adapté par Bernard Sauteur

ISBN : 2-86595-259-2

CLEFS POUR VG 5000

par François Normant

Éditions du PSI
1985

Présentation de l'auteur :

François NORMANT est étudiant en mathématiques spéciales. Autodidacte, il s'est orienté vers l'informatique système et les télécommunications. Il est auteur de logiciels utilitaires et d'ouvrages techniques sur ORIC, MO5, TO7/70 et VG 5000.

Du même auteur, aux éditions du P.S.I. :

- Assembleur et périphériques des MO5 et TO7-70.

SOMMAIRE

ANNEXES

INTRODUCTION

Ce mémento va devenir votre outil de travail privilégié et vous sera bientôt indispensable pour exploiter le VG 5000 au maximum de ses possibilités.

Vous y trouverez rapidement toutes les informations dont vous pourrez avoir besoin pour programmer efficacement, aussi bien en Basic qu'en langage machine.

Chaque renseignement est donné de façon concise, le but de ce livre étant l'accès quasi immédiat à l'information recherchée. Cet ouvrage n'est donc en aucun cas un cours de Basic ou de langage machine, mais plutôt un aide-mémoire.

Vous trouverez à la fin de ce mémento une liste de "Trucs et astuces" : Les comment ?, destinée à mieux utiliser le VG 5000 et ses périphériques.

REPERTOIRE DES INSTRUCTIONS ET DES OPERATEURS BASIC

Mot-clé	*Définitions-exemples*

Opérateurs arithmétiques et logiques

& "xxxx" Permet d'évaluer une chaîne en tant que variable hexadécimale. L'opérande doit être compris entre 0 et $FFFF.
Ex : PRINT &"1A" affiche 26.

ABS (A) Donne la valeur absolue de l'opérande A.
Ex : PRINT ABS(-38.3) affiche 38.3.

A AND B Effectue un "et logique" entre les opérandes A et B ou bien vérifie si deux conditions sont vraies.
Ex : PRINT 96 AND 60 affiche 32.
IF C=0 AND D=1 THEN INIT6,6 efface l'écran si C=0 et D=1.

ATN (A) Donne l'arc tangente de l'opérande A.
Ex : PRINT ATN(0) affiche 0.

COS (A) Donne le cosinus de l'opérande A qui doit être en radian.
Ex : PRINT COS(0) affiche 1.

DEFFN A(X) Permet de définir une fonction numérique ayant pour variable X et pour nom A.
La fonction ainsi définie s'utilise avec l'instruction FN A(X).
Ex : tapez le programme suivant :

```
10 CH(X)=(EXP(X)+EXP(-X))/2
20 SH(X)=(EXP(X)-EXP(-X))/2
```

On a ainsi défini les fonctions cosinus et sinus hyperbolique.

EXP (A) Donne l'exponentielle de l'opérande A, celui-ci doit être inférieur ou égal à 87.3365.
Ex : PRINT EXP(1) affiche 2.71828.

Mot-clé	*Définitions-exemples*
FN A(X)	Permet d'utiliser une fonction prédéfinie par l'instruction DEFFN A(X). **Ex** : en utilisant les fonctions précédemment définies, on obtient : PRINT FN CH(2) affiche 3.7622 PRINT FN SH(2) affiche 3.62686
INT (A)	Donne la partie entière de l'opérande A, c'est-à-dire le plus grand entier inférieur ou égal à celui-ci. **Ex** : PRINT INT (-0.5) affiche -1.
LOG (A)	Donne le logarithme népérien de l'opérande A, celui-ci doit être supérieur à zéro. **Ex** : PRINT LOG(1) affiche 0.
NOT A	Donne le complément à 1 d'un nombre ou d'une expression.
A OR B	Effectue un "ou logique" entre les opérandes A et B ou bien vérifie si l'une au moins des conditions d'un test est vraie. **Ex** : PRINT 96 OR 60 affiche 124. IF C=1 OR D=1 OR E=1 THEN INIT6,6 efface l'écran si l'une des variables est à 1.
RND (A)	Donne un nombre aléatoire compris entre 0 et 1. Si A<0, le nombre tiré sera le même que le précédent, si A>0, le nombre tiré sera le suivant dans la séquence. **Ex** : PRINT RND (1).
SGN (A)	Donne 1 si l'opérande A est positif, -1 si négatif et 0 si nul.
SIN (A)	Donne le sinus de l'opérande A, celui-ci doit être exprimé en radian. **Ex** : PRINT SIN (0) donne 0.
SQR (A)	Donne la racine carrée de l'opérande A, celui-ci doit être positif. **Ex** : PRINT SQR (2) donne 1.41421.
TAN (A)	Donne la tangente de l'opérande A, celui-ci doit être exprimé en radian. **Ex** : PRINT TAN (0) donne 0.

Opérateurs vidéo

CURSORX A	Positionne le curseur à l'abscisse spécifiée par A (colonne).

Mot-clé	*Définitions-exemples*
CURSORY A	Positionne le curseur à l'ordonnée spécifiée par A (ligne).
DELIM A,B,C	Met un message en relief en changeant la couleur du délimiteur (opérande A) et celle du fond (opérande B) et en le soulignant (opérande C). Les valeurs des opérandes A et B correspondent au tableau des codes de couleurs, l'opérande C doit avoir une valeur impaire pour un souligné et une valeur paire pour un non souligné. **Ex** : tapez le programme suivant : 10 INIT 6,6 20 DELIM 1,2,0 30 PRINT "Bonjour à tous"; 40 DELIM 3,6,0 Faites "RUN" et vous voyez un carré rouge suivi de "Bonjour à tous" sur fond vert, suivi d'un carré jaune.
DISPLAY A	Permet de modifier la vitesse d'affichage des caractères. L'opérande A doit être compris entre 0 et 255. Le délai d'affichage entre deux caractères est 20*A en ms. L'opérande est facultatif et vaut 20 par défaut. **Ex** : tapez le programme suivant : 10 A$="ESSAI DE LA COMMANDE DISPLAY" 20 FORI=1TOLEN(A$) 30 PRINT MID$(A$,I,1); 40 SOUND 200,1 50 DISPLAY 60 NEXT I Exécutez le programme, puis supprimez la ligne 50 ; vous apprécierez alors l'intérêt de DISPLAY.
EG A,B,C	Initialise le mode graphique spécial (caractères définis par l'utilisateur). L'opérande A correspond à la couleur de l'encre, l'opérande B à la couleur du fond. Voir le tableau des codes de couleurs. L'opérande C définit le clignotement (0 = stable, 1 = clignotant). **Ex** : tapez le programme suivant : 10 CURSORX 10 : CURSORY 10 20 GR 3,4,1 Ce programme définit la ligne 10 à partir de la colonne 10, comme zone graphique clignotante encre jaune, fond bleu.

Mot-clé	*Définitions-exemples*
ET A,B,C	Initialise le mode texte spécial. L'opérande A correspond à la couleur. L'opérande B correspond à l'attribut des caractères (voir tableau), l'opérande C au clignotement (0 = stable, 1 = clignotant).
GR A,B,C	Initialise le mode graphique normal. Les opérandes A, B, C sont les mêmes que pour EG.
INIT A,B	Efface l'écran et change la couleur de l'encre (opérande A) et du fond (opérande B).
LPOS (A)	Donne l'abscisse (colonne) du dernier caractère imprimé (sur l'imprimante).
LPRINT	Permet l'impression d'une variable ou d'une chaîne de caractères sur l'imprimante.
PAGE	Permet de geler le scrolling de l'écran.
POS (A)	Donne l'abscisse (colonne) du dernier caractère affiché à l'écran.
PRINT	Permet l'impression d'une variable ou d'une chaîne de caractères à l'écran.
SCREEN	Permet d'afficher à l'écran une image qui a été réalisée après une commande STORE.
SCROLL	Permet le scrolling de l'écran après que celui-ci ait été gelé par la commande PAGE.
SETEG A,"BBCCDDEEFFGGHHIIKK"	Permet de définir un caractère graphique spécial. L'opérande A définit le caractère (code compris entre 32 et 127), tandis que l'opérande entre guillemets définit la nouvelle matrice du caractère, chaque ligne étant représentée par une valeur hexadécimale, soit deux caractères.

Ex : matrice du caractère A :

```
00000000 soit 00 en hexadécimal
00111000 soit 38 en hexadécimal
01000100 soit 44 en hexadécimal
01000100 soit 44 en hexadécimal
01000100 soit 44 en hexadécimal
01111100 soit 7C en hexadécimal
01000100 soit 44 en hexadécimal
01000100 soit 44 en hexadécimal
00000000 soit 00 en hexadécimal
00000000 soit 00 en hexadécimal
```

On peut donc redéfinir le A par :

```
SETEG 65, "00384444447C44440000".
```

Mot-clé	*Définitions-exemples*

SETET A,"BBCCDDEEFFGGHHIIJJKK" Commande identique à SETEG, mais elle permet de définir un caractère de texte spécial.

SPC (A) Affiche A espace(s) à la position courante du curseur. L'opérande A doit être compris entre 0 et 255.

STORE Permet de concevoir une image sans que celle-ci n'apparaisse à l'écran lors de sa fabrication. On évite ainsi la désagréable apparition de motifs successifs.
Pour faire afficher l'image ainsi conçue, on utilise SCREEN.
Nota : DISPLAY, END, STOP, un scrolling d'écran, une erreur, font aussi apparaître l'image.
Ex : reprenez le programme donné pour DISPLAY en supprimant la ligne 50 et en rajoutant les lignes :

```
 5 STORE
70 SCREEN
```

Remarque : la ligne 70 est facultative puisque le programme s'achève derrière (=END). Elle a uniquement un but pédagogique.

TAB (A) Permet d'effectuer des tabulations aussi bien sur l'écran que sur l'imprimante. L'opérande A doit être compris entre 1 et 39.

TX A,B,C Initialise le mode texte normal. Les opérandes A, B, C sont les mêmes que pour ET.

Opérateurs de variables numériques et alphanumériques

ASC (A$) Donne le code ASCII du premier caractère de la chaîne A$ (voir tableau des codes ASCII).
Ex : PRINT ASC ("A") affiche 65.

CHR$ (A) Donne le caractère correspondant au code ASCII défini par l'opérande A.
Ex : PRINT CHR$ (65) affiche un A.

CLEAR A,B CLEAR efface toutes les variables. Les opérandes A et B sont optionnels. Le premier permet de définir la totalité de l'espace occupé par les chaînes (valeur comprise entre -32768 et 32767), le second définit l'adresse maximale jusqu'à

Mot-clé	*Définitions-exemples*
	laquelle le Basic peut s'implanter (valeur inférieure à 32767 pour VG 5000 16K ou 42864 avec l'extension mémoire). Cette commande sera utilisée pour réserver de la place pour un programme en langage machine.
DATA	Permet de définir une liste de données qui seront lues par l'instruction READ. Ex : DATA POMME,POIRE,BANANE
DIM variable	Permet de dimensionner un tableau de variables numériques ou alphanumériques. Ex : DIM A$(10) DIM A(10,10)
LEFT$ (A$,B)	Donne une chaîne composée des B premiers caractères de A$. Ex : PRINT LEFT$ ("BONJOUR",3) affiche 'BON'.
LEN (A$)	Donne la longueur de la chaîne A$. Ex : PRINT LEN ("COUCOU") affiche 6.
LET	Affecte une valeur à une variable. Pratiquement inutilisée, cette commande est optionnelle. Ex : LET A=2 <=> A=2 LET B$="Toto" <=> B$="Toto"
MID$ (A$,B,C)	Donne une chaîne composée de C caractères de A$ à partir du B ième. Ex : PRINT MID$("COUCOU",3,2) affiche 'UC'.
READ variable	Permet de lire des données définies par l'instruction DATA. Ex : READ A$
RESTORE A	Positionne le pointeur de lecture DATA sur la ligne A. On peut ainsi accéder à des données par rapport au numéro de ligne. Ex : RESTORE 10:READ A
RIGHT$ (A$,B)	Donne une chaîne composée des B derniers caractères de A$. Ex : PRINT RIGHT$ ("BONJOUR",4) affiche 'JOUR'.
STR$ (A)	Interprète une variable numérique comme une chaîne de caractères. Ex : A$ = STR$(12)
VAL (A$)	Interprète une chaîne de caractères comme une variable numérique (si la chaîne est composée de caractères autres que des chiffres, elle vaut 0). Ex : PRINT VAL ("12") affiche 12.

Mot-clé	*Définitions-exemples*

Opérateurs de branchements et de boucles

CONT Permet de continuer l'exécution d'un programme après que celui-ci a été interrompu par un STOP.

END Achève l'exécution d'un programme.

FOR..TO..NEXT..STEP Permet de faire une boucle, c'est-à-dire de répéter une suite d'instructions.
FOR I= A TO B STEP C: suite d'instructions :
NEXT I a pour effet d'effectuer à chaque passage sur le NEXT I, un incrément de C du pointeur I jusqu'à ce que I atteigne B. L'incrément C est optionnel et vaut par défaut 1.
Ex : tapez le programme suivant :

```
10 FOR I=32 TO 127
20 PRINT CHR$(I);
30 NEXT I
```

Faites RUN, le jeu de caractères du VG 5000 apparaît.

GOSUB A Effectue un appel à un sous-programme situé à la ligne A. Lorsqu'une instruction RETURN sera rencontrée, l'exécution du programme reprendra à la ligne suivant le GOSUB A.

GOTO A Effectue un saut à la ligne A.

IF...THEN Effectue un test. Si celui-ci est vrai, alors l'action suivant le THEN est exécutée, sinon la ligne suivante est directement exécutée.
On peut trouver, derrière le THEN, soit un numéro de ligne, soit une suite d'instructions.
Ex : IF A=0 THEN PRINT "COUCOU"

IF...GOTO Identique à IF...THEN, mais le GOTO peut être uniquement suivi d'un numéro de ligne.

ON A GOSUB... Effectue un appel à différents sous-programmes suivant la valeur de A.
Le branchement s'effectue au A ième numéro de ligne.
Ex : ON A GOSUB 500,600,700,800
Si A=1, le branchement s'effectue en 500.
Si A=2, le branchement s'effectue en 600.
...

ON A GOTO Identique à ON GOSUB, mais effectue un saut au lieu d'un appel à un sous-programme.

Mot-clé	*Définitions-exemples*
RETURN	Permet de terminer un sous-programme et renvoie au programme principal.
RUN A	Exécute un programme à partir de la ligne A en effaçant les variables. Le numéro de ligne est facultatif ; s'il n'est pas spécifié, le programme s'exécute à partir de la première ligne.
STOP	Achève l'exécution d'un programme et affiche le numéro de la dernière ligne exécutée

Opérateurs concernant l'éditeur

AUTO A,B	Permet la numérotation automatique des lignes lorsque l'on écrit un programme. La première ligne est A et l'incrément B. A et B sont pris par défaut lorsqu'ils ne sont pas spécifiés à 10. Pour sortir du mode AUTO, on doit, après avoir validé la dernière ligne, faire reculer le curseur d'une case et appuyer sur RET. Lorsqu'une ligne existe déjà, un "carreau" est affiché à côté du numéro de ligne, ce caractère est transparent et ne sert que d'indicateur.
FRE (A)	Donne le nombre d'octets disponibles pour l'utilisateur. **Ex** : PRINT FRE (A) donne 13758 à l'allumage (sur un VG 5000 16K).
FRE (A$)	Donne le nombre d'octets disponibles pour les chaînes (mis à jour par CLEAR A). **Ex** : PRINT FRE (A$) donne 50 à l'allumage.
LIST A	Liste à l'écran le programme en mémoire à partir de la ligne A. L'opérande A est facultatif ; s'il est omis, le programme est listé à partir de la première ligne.
LLIST A	Identique à LIST, mais liste le programme sur l'imprimante.
NEW A	Permet d'effacer partiellement un programme depuis la ligne A jusqu'à la fin. Si l'opérande A n'est pas précisé, le programme est effacé totalement.
REM	Permet d'insérer des remarques à l'intérieur d'un programme. Celles-ci seront totalement ignorées par le VG 5000 lors de l'exécution du programme.

Mot-clé	*Définitions-exemples*
RENUM A,B,C	Renumérote avec l'incrément C le programme, à partir de la ligne B, en lui donnant le numéro A. Les paramètres A, B, C sont facultatifs et ont pour valeur 10 par défaut.

Opérateurs concernant le clavier

INPUT Acquisition de données au clavier et stockage dans la variable spécifiée. Celle-ci peut être autant numérique qu'alphanumérique. Cette instruction permet aussi d'afficher un message. La fin de la saisie doit être validée par RET.

Ex : tapez le programme suivant :

```
10 INPUT "Votre prenom";A$
20 PRINT "Bonjour";A$
```

KEY (0) Donne la valeur ASCII de la touche enfoncée au moment où l'ordinateur exécute l'instruction. Pour les lettres, la valeur renvoyée est celle correspondant à la lettre minuscule.

Ex : tapez le programme suivant :

```
10 A=KEY(0)
20 PRINT CHR$(A)
30 GOTO 10
```

Opérateurs concernant le langage machine

Toutes les adresses doivent être comprises entre -32768 et 32767.

CALL A Permet d'appeler un programme en langage machine ayant A pour adresse d'exécution. Les codes se trouvant à l'adresse A doivent être évidemment cohérents, sinon on risque un "plantage" intempestif du VG 5000.

PEEK (A) Donne la valeur de l'octet ayant A pour adresse.
Ex : PRINT PEEK(&"0500")

POKE A,B Affecte la valeur B (comprise entre 0 et 255) à l'octet ayant A pour adresse.
Ex : POKE &"4890",1

Mot-clé	*Définitions-exemples*
USR (A)	Permet un appel avec passage de paramètres à une routine utilisateur ; l'adresse a été définie auparavant.

Remarque : certaines de ces instructions seront étudiées en détail dans la rubrique "Les Comment...".

Opérateurs musicaux

PLAY A$

Permet de jouer une séquence de notes de musique. Cette séquence s'organise sous forme d'une chaîne de caractères composée des notes elles-mêmes, leur durée, de leur octave, de leur tempo...). Les notes sont représentées par les lettres A à G (A=la, B=si,..., G=sol).
Le dièse est codé par un + et le bémol par un -, **Ex** : le dièse est représenté A+.
Chaque note peut être suivie de sa durée (facultative, par défaut 16). Celle-ci doit être comprise entre 1 et 99, ex A65.
O donne l'octave. Celle-ci va de 1 à 4, ex O2 est que
T donne le tempo. Celui-ci va de 1 à 255 (255 le tempo le plus lent). **Ex**. T1ØØ.
R correspond à la pause. Sa durée est la même celle d'une note, ex R1Ø.
Ex : PLAY "TO3ØO3CC+DD+EFG-GG+RARB"

SOUND A,B,C

Permet d'émettre un son suivant les opérandes A, B et C, chacun compris entre Ø et 255.
A définit la fréquence du son.
B définit la durée du son.
C est un paramètre facultatif qui définit le rapport cyclique de l'onde. Il vaut Ø par défaut, qui correspond à une onde pure.
Ex : SOUND 11Ø,3Ø,7

Opérateurs concernant le magnétophone

Pour toutes ces instructions, "XXXX" représente un nom de fichier qui ne doit pas excéder six caractères (pour toutes les opérations de chargement, ce nom est facultatif ; s'il est omis, le premier fichier rencontré sera chargé).

Pour toutes les fonctions de sauvegarde, la vitesse lente (1200 bauds) est prise par défaut ; on peut cependant spécifier

Mot-clé *Définitions-exemples*

vitesse rapide (2400 bauds) en faisant suivre le CSAVE ou SAVE de (2).
Ex : CSAVE(2),"TOTO"

Toutes les adresses doivent être comprises entre -32768 et 32767.

CLOAD "XXXX",ligne Charge dans le VG 5000 le programme "XXXX" et l'exécute à partir de la ligne spécifiée. Le numéro de ligne est facultatif.
Le programme Basic qui se trouvait dans la mémoire est effacé.
Cette commande permet aussi bien de charger les programmes Basic (sauvés par CSAVE), les programmes en langage machine (sauvés par CSAVEM), et les écrans (sauvés par CSAVES).

CLOAD "XXXX",chaîne Charge la chaîne spécifiée. Elle devra être préalablement déclarée.

CLOAD*"XXXX",tableau Charge le tableau précédemment sauvé sous le même nom. Celui-ci devra être dimensionné avant le chargement.

CLOAD? "XXXX" Compare le programme présent dans la mémoire et celui sur cassette. S'il y a une différence, le message "mauvais fichier" est affiché.

CLOADA "XXXX",ligne Charge le programme "XXXX" en le concaténant à celui qui réside déjà en mémoire (à condition que les numéros de ligne soient différents), et l'exécute éventuellement à partir de la ligne spécifiée.

CSAVE "XXXX",ligne Sauve le programme résidant en mémoire sous le nom "XXXX" en spécifiant une éventuelle ligne d'exécution au chargement.

CSAVE* "XXXX"tableau Sauve le tableau numérique ou alphanumérique sous le nom "XXXX".

CSAVEL Laisse tourner le moteur du magnétophone de manière à dépasser la bande amorce.

CSAVEM "XXXX",A,B,C Sauve B octet(s) à partir de l'adresse A et spécifie C comme éventuelle adresse d'exécution.

CSAVES "XXXX" Sauve l'écran sur la cassette.

CSAVEX "XXXX"chaîne Sauve la chaîne spécifiée sous le nom "XXXX".

Mot-clé	*Définitions-exemples*
LOAD A,B	Charge un programme Basic sauvé en ASCII à partir de la ligne A jusqu'à la ligne B, et effectue un éventuel mixage avec le programme existant déjà dans le VG 5000.
SAVE A,B	Sauve en ASCII un programme Basic de la ligne A jusqu'à la ligne B. Cette opération efface ensuite le programme de la mémoire.

Opérateurs concernant les poignées de jeux

Voici à quoi correspondent les valeurs de A :

A=0 : poignée de droite ;
A=1 : poignée de gauche ;
A=2 : touches du clavier.

ACTION (A)	0 : aucun bouton enfoncé ; 1 : bouton action 1 enfoncé ; 2 : bouton action 2 enfoncé ; 3 : boutons action 1 et 2 enfoncés.
STICKX (A)	Indique si les poignées de jeux (ou les flèches du clavier) sont orientées à droite ou à gauche. "repos" = 0 "droite" = 1 "gauche" = 255.
STICKY (A)	Identique à STICKX, mais avec haut et bas. "repos" = 0 "bas" = 1 "haut" = 255.

LISTE DES MOTS-CLEFS BASIC AVEC LEUR TOKEN

Instruction	*Token Dec*	*Token Hex*	*Instruction*	*Token Dec*	*Token Hex*
END	128	80	MODEM	176	B0
FOR	129	81	NEW	177	B1
NEXT	130	82	TAB(	178	B2
DATA	131	83	TO	179	B3
INPUT	132	84	FN	180	B4
DIM	133	85	SPC (	181	B5
READ	134	86	THEN	182	B6
LET	135	87	NOT	183	B7
GOTO	136	88	STEP	184	B8
RUN	137	89	+	185	B9
IF	138	8A	-	186	BA
RESTORE	139	8B	*	187	BB
GOSUB	140	8C	/	188	BC
RETURN	141	8D	^	189	BD
REM	142	8E	AND	190	BE
STOP	143	8F	OR	191	BF
ON	144	90	>	192	C0
LPRINT	145	91	=	193	C1
DEF	146	92	<	194	C2
POKE	147	93	SGN	195	C3
PRINT	148	94	INT	196	C4
CONT	149	95	ABS	197	C5
LIST	150	96	USR	198	C6
LLIST	151	97	FRE	199	C7
CLEAR	152	98	LPOS	200	C8
RENUM	153	99	POS	201	C9
AUTO	154	9A	SQR	202	CA
LOAD	155	9B	RND	203	CB
SAVE	156	9C	LOG	204	CC
CLOAD	157	9D	EXP	205	CD
CSAVE	158	9E	COS	206	CE
CALL	159	9F	SIN	207	CF
INIT	160	A0	TAN	208	E0
SOUND	161	A1	ATN	209	E1
PLAY	162	A2	PEEK	210	E2
TX	163	A3	LEN	211	E3
GR	164	A4	STR$	212	E4
SCREEN	165	A5	VAL	213	E5
DISPLAY	166	A6	ASC	214	E6
STORE	167	A7	STICKX	215	E7
SCROLL	168	A8	STICKY	216	E8
PAGE	169	A9	ACTION	217	E9
DELIM	170	AA	KEY	218	EA
SETE	171	AB	LPEN	219	EB
ET	172	AC	CHR$	220	EC
EG	173	AD	LEFT$	221	ED
CURSOR	174	AE	RIGHT$	222	EE
DISK	175	AF	MID$	223	EF

Les messages d'erreur sont de la forme :

<message d'erreur> en <No de ligne>

- Annulé-repositionner la bande

Le chargement a été interrompu à cause d'une mauvaise manipulation. Il faut alors recommencer le chargement.

- Appel de fonction incorrect

Utilisation d'une fonction avec un paramètre hors-intervalle.

- Arrêt en ligne nnnn

Intervient lorsque la commande ou la touche STOP a été utilisée. Le numéro de la ligne exécutée avant l'interruption est affiché.

- Chaîne trop longue

Tentative de déclaration d'une chaîne de longueur supérieure à 255 caractères.

- Dépassement de capacité

Le résultat d'une opération arithmétique est supérieur ou égal à 10E38.

- Dernière information ignorée

Apparaît lorsque l'on demande l'édition d'une donnée non définie.

- Division par zéro

Tentative d'effectuer une division par zéro.

- Données épuisées

Intervient lorsqu'une instruction READ est exécutée alors qu'il ne reste plus de données à lire dans les lignes DATA.

- Espace-chaîne épuisé

Tentative de déclaration d'une chaîne dépassant la capacité mémoire allouée par un CLEAR (50 octets par défaut).

- Erreur de syntaxe

Une instruction est mal orthographiée ou utilisée avec de mauvais arguments.

- **Fichier non correspondant**

  Erreur survenue durant la vérification d'un fichier sur cassette.

- **Fonction utilisateur non définie**

  L'instruction FN a été utilisée sans que la fonction ait été définie par DEFFN.

- **Formule chaîne trop complexe**

  L'expression chaîne est trop complexe pour être interprétée.

- **Impossible de continuer**

  Impossibilité de reprendre l'exécution d'un programme par CONT (notamment après une erreur ou un arrêt par CNTL Delta).

- **Imprimante pas prête**

  L'imprimante ne répond pas à un appel du VG 5000.

- **Incorrect en direct**

  Tentative d'exécuter une commande interdite en mode direct. C'est le cas de INPUT et DEFFN.

- **Indice hors des limites**

  Tentative d'utiliser un élément de tableau inexistant.

- **Ligne non définie**

  Tentative de référencer une ligne inexistante.

- **Mauvais fichier**

  Un défaut est survenu durant le chargement ; le programme n'est pas exécutable.

- **NEXT sans FOR**

  L'instruction NEXT a été rencontrée alors que l'instruction FOR correspondante n'existe pas ou est mal placée.

- **Non reconnue**

  Erreur détectée mais non répertoriée.

- **Opérande mal adapté**

  Valeur alphanumérique affectée à une variable numérique ou inversement.

- **Opérande manquant**

Il manque un opérande dans une expression numérique.

- **Passé : "nom de programme"**

Indique le nom du programme que le VG 5000 rencontre et qui ne correspond pas au nom précisé dans la commande de chargement.

- **Pause...**

S'affiche lors de l'interprétation d'un fichier ASCII par le VG 5000 durant une commande LOAD.

- **Périphérique non connecté**

Le périphérique appelé est absent.

- **Recommencez au début**

Une quantité alphanumérique a été introduite lors d'un INPUT alors qu'il s'agissait d'une demande de type numérique.

Il faut répondre à nouveau à l'INPUT.

- **Retour sans GOSUB**

Une instruction RETURN a été rencontrée sans qu'il y ait eu de GOSUB.

- **Sortie de mémoire**

La mémoire allouée au programme ou aux variables est saturée ; survient également lorsque trop de boucles FOR...NEXT ou de GOSUB ont été imbriquées.

- **Tableau redimensionné**

Tentative de dimensionner un tableau qui a déjà été déclaré.

- **Trouvé : "nom de programme"**

Indique le nom du programme que le VG 5000 est en train de charger.

JEU DE CARACTERES

Caractère	Code Dec	Code Hex	Caractère	Code Dec	Code Hex
!	33	21	F	70	46
"	34	22	G	71	47
#	35	23	H	72	48
$	36	24	I	73	49
%	37	25	J	74	4A
&	38	26	K	75	4B
'	39	27	L	76	4C
(	40	28	M	77	4D
)	41	29	N	78	4E
*	42	2A	O	79	4F
+	43	2B	P	80	50
,	44	2C	Q	81	51
-	45	2D	R	82	52
.	46	2E	S	83	53
/	47	2F	T	84	54
Ø	48	30	U	85	55
1	49	31	V	86	56
2	50	32	W	87	57
3	51	33	X	88	58
4	52	34	Y	89	59
5	53	35	Z	90	5A
6	54	36	[	91	5B
7	55	37	\	92	5C
8	56	38	]	93	5D
9	57	39	^	94	5E
:	58	3A	_	95	5F
;	59	3B	`	96	60
<	60	3C	a	97	61
=	61	3D	b	98	62
>	62	3E	c	99	63
?	63	3F	d	100	64
@	64	40	e	101	65
A	65	41	f	102	66
B	66	42	g	103	67
C	67	43	h	104	68
D	68	44	i	105	69
E	69	45	j	106	6A

Caractère	*Code Dec*	*Code Hex*	*Caractère*	*Code Dec*	*Code Hex*
k	107	6B	v	118	76
l	108	6C	w	119	77
m	109	6D	x	120	78
n	110	6E	y	121	79
o	111	6F	z	122	7A
p	112	70	{	123	7B
q	113	71	¦	124	7C
r	114	72	}	125	7D
s	115	73	~	126	7E
t	116	74	▲	127	7F
u	117	75			

Minuscules accentuées

Caractère	*Code Dec*	*Code Hex*	*Caractère*	*Code Dec*	*Code Hex*
î	17	11	â	24	18
é	18	12	è	25	19
ù	19	13	ô	26	1A
ï	20	14	ê	27	1B
ç	21	15	£	28	1C
û	22	16	½	29	1D
à	23	17			

0	Noir
1	Rouge
2	Vert
3	Jaune
4	Bleu
5	Violet
6	Cyan
7	Blanc

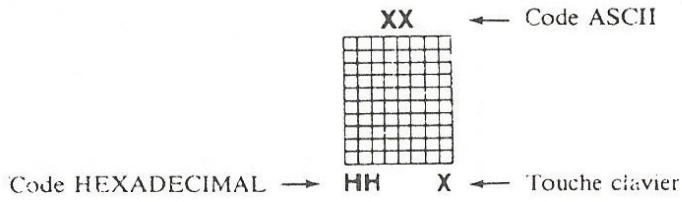
ATTRIBUTS DES CARACTERES

0	Normal
1	Double hauteur
2	Double largeur
3	Double hauteur et double largeur
4	Inversion vidéo
5	Inversion vidéo et double hauteur
6	Inversion vidéo et double largeur
7	Inversion vidéo, double hauteur et double largeur

Code Hex	Code Dec	Clavier	Fonction
00	0		Pas d'effet.
01	1		
02	2	EFFC	Effacement du caractère gauche.
03	3	RET	Retour chariot et saut de ligne.
04	4	EFFL	Effacement depuis la position du curseur jusqu'à la fin de la ligne.
05	5	INSC	Insertion d'un caractère à la position du curseur.
06	6	INSL	Insertion d'une ligne.
07	7	Flèche droite	Déplacement du curseur d'un caractère vers la droite.
08	8	Flèche gauche	Déplacement du curseur d'un caractère vers la gauche.
09	9	Flèche haut	Déplacement du curseur d'un caractère vers le haut.
0A	10	Flèche bas	Déplacement du curseur d'un caractère vers le bas.
0B	11		Pas d'effet.
0C	12		Déplace le curseur en haut à gauche de l'écran.
0D	13	RET	Idem code 3.
0E	14		Autorise la visualisation.
0F	15		Pas d'effet.

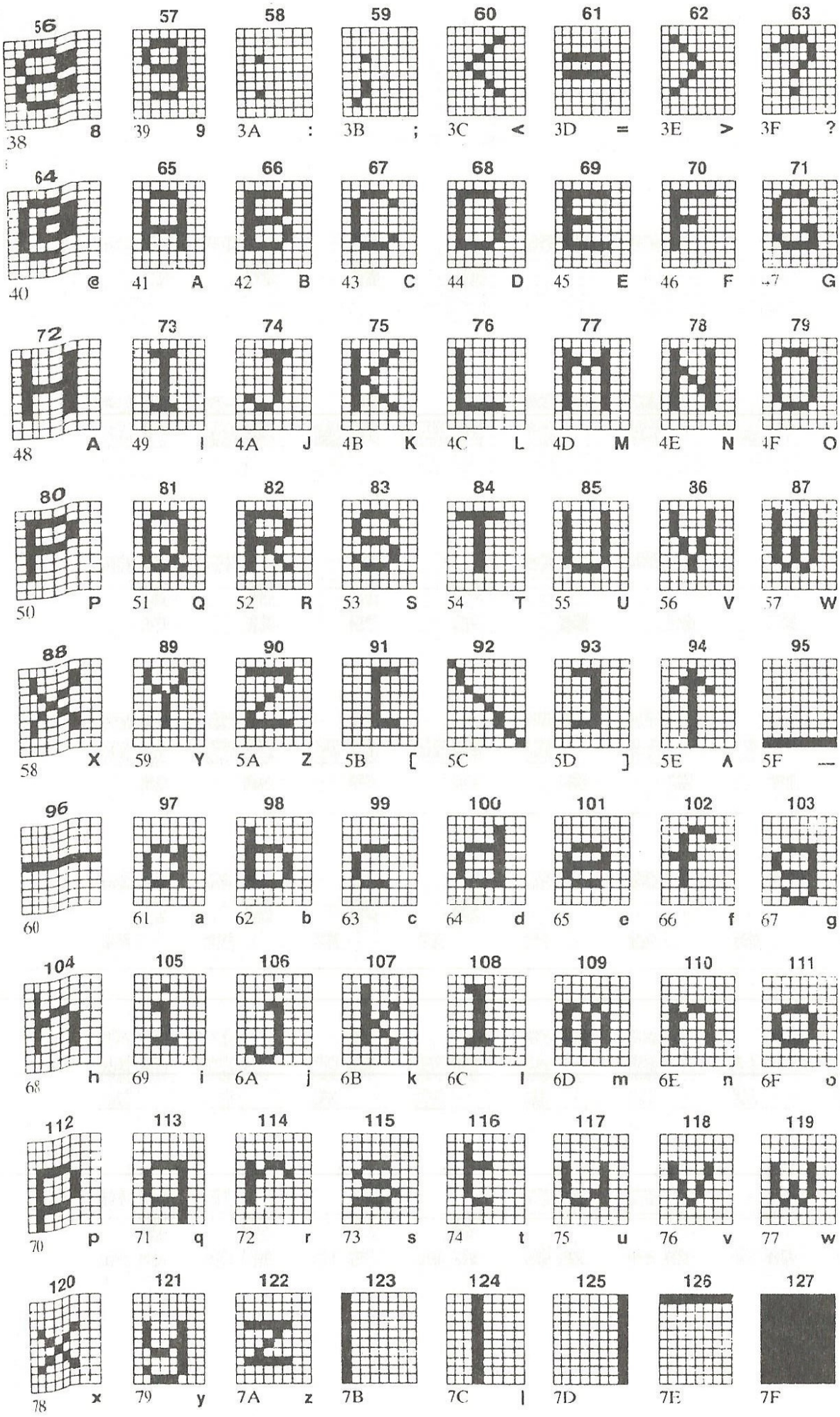
MATRICES DES CARACTERES EN MODE TEXTE

La mention **CS** signifie qu'il s'agit d'un code de contrôle.

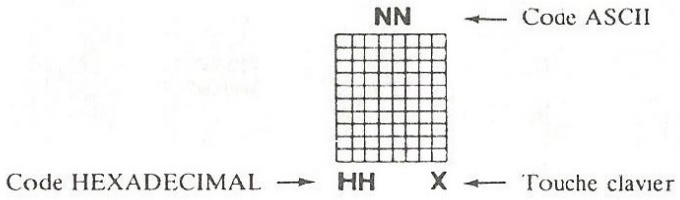


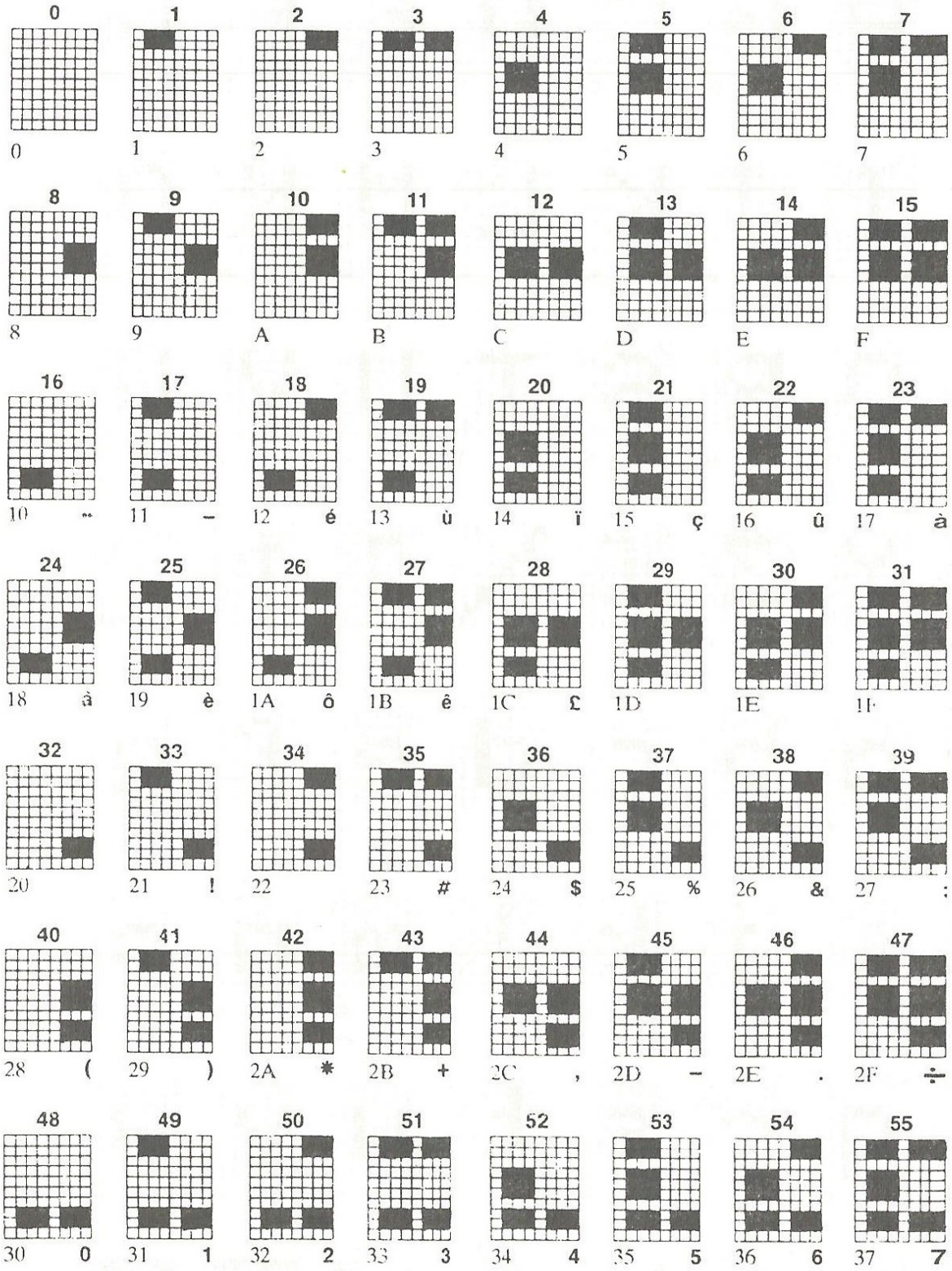
0	1	2	3	4	5	6	7
CS	CS	CS	CS	CS	CS	CS	CS
6	1	2	3	4	5	6	7
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
CS	CS	CS	CS	CS	CS	CS	CS
8	9	A	B	C	D	E	F
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
10 ¨	11 ~	12 é	13 ù	14 ï	15 ç	16 û	17 à
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
â	19 è	1A ô	1B ê	1C £	1D	1E CS	1F CS
32	**33**	**34**	**35**	**36**	**37**	**38**	**39**
20	21 !	22	23 #	24 $	25 %	26 &	27 ;
40	**41**	**42**	**43**	**44**	**45**	**46**	**47**
28 (	29)	2A *	2B +	2C ,	2D −	2E .	2F ÷
48	**49**	**50**	**51**	**52**	**53**	**54**	**55**
30 **0**	31 **1**	32 **2**	33 **3**	34 **4**	35 **5**	36 **6**	37 **7**

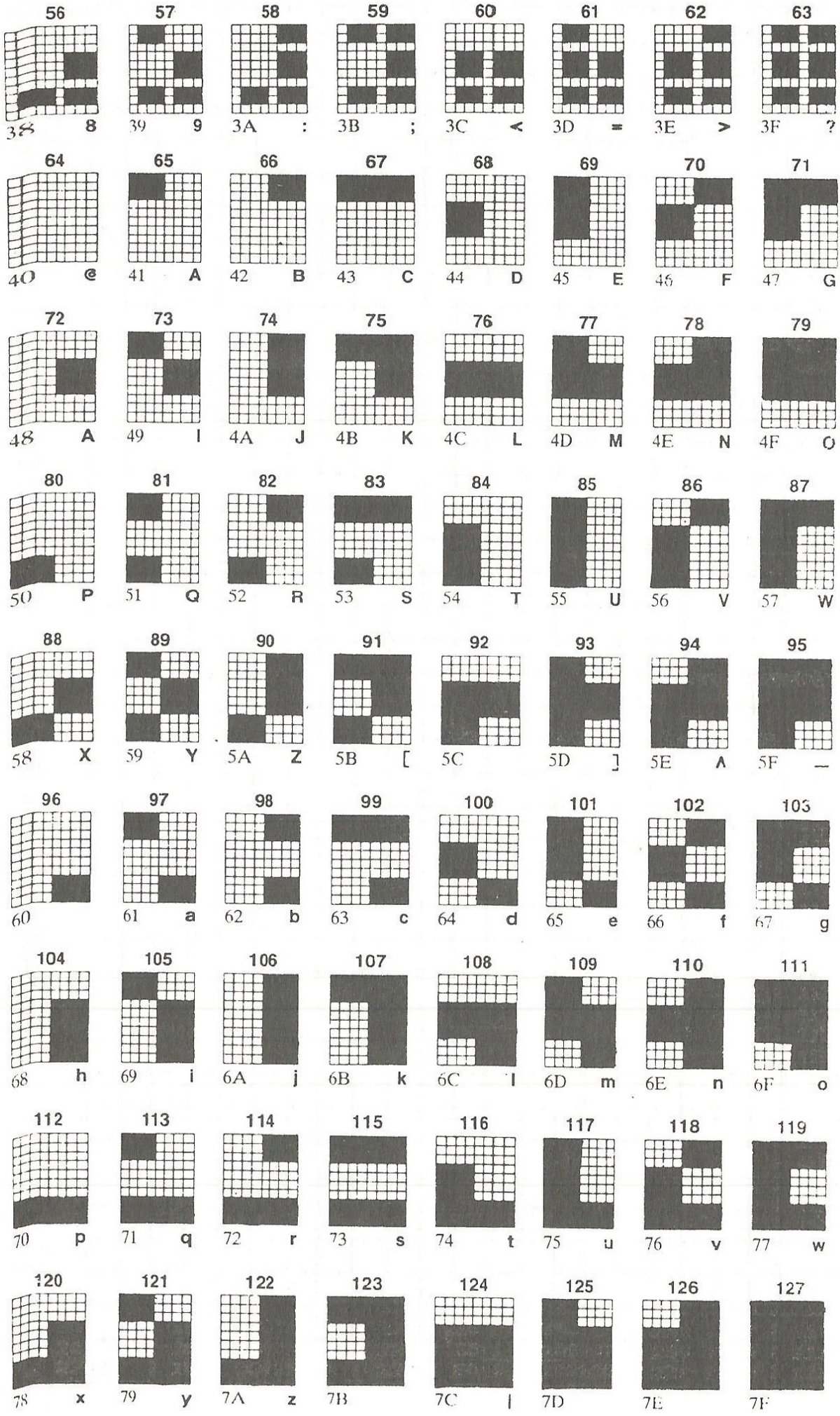
MATRICES DES CARACTERES EN MODE TEXTE



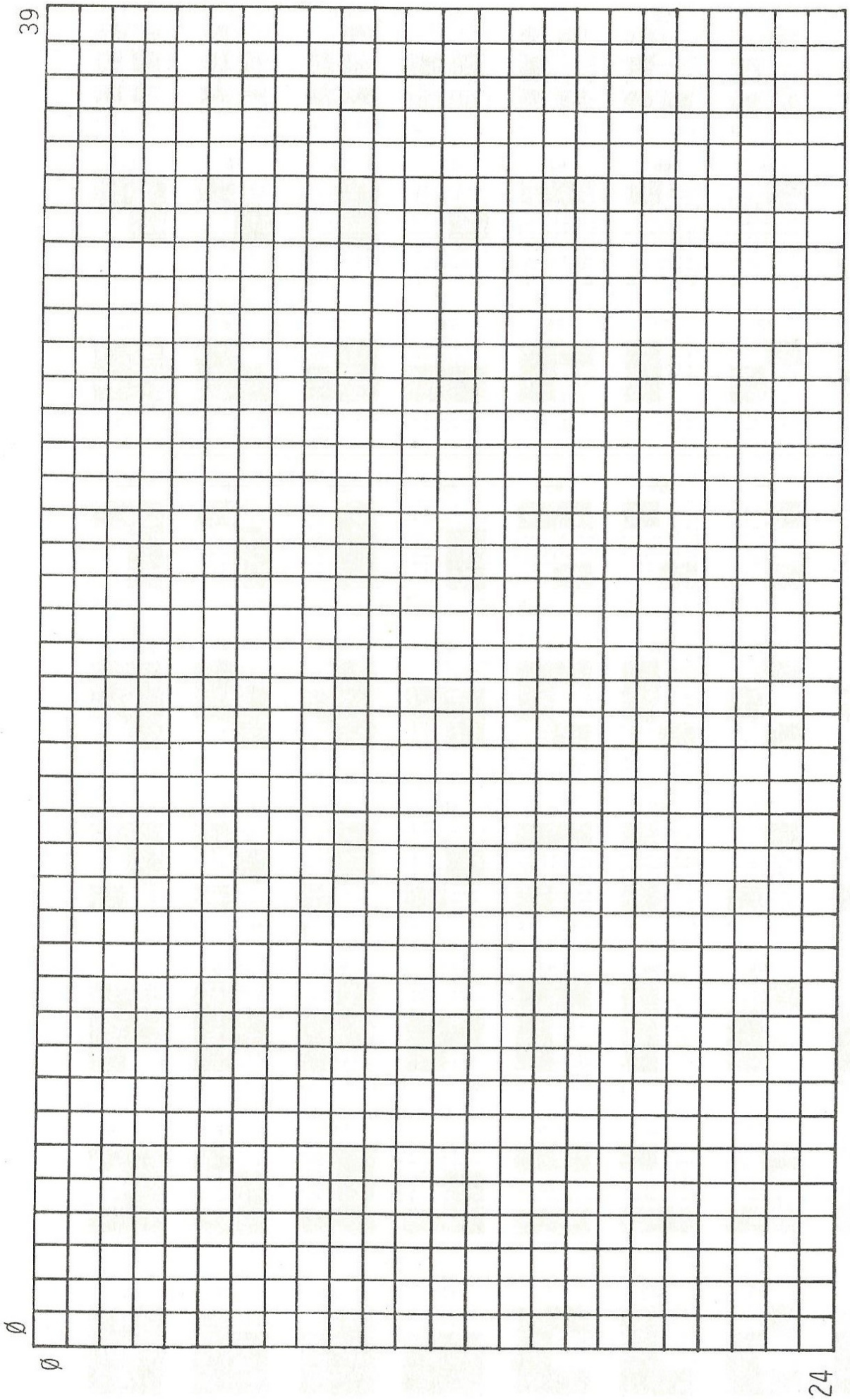
MATRICES DES CARACTERES EN MODE GRAPHIQUE



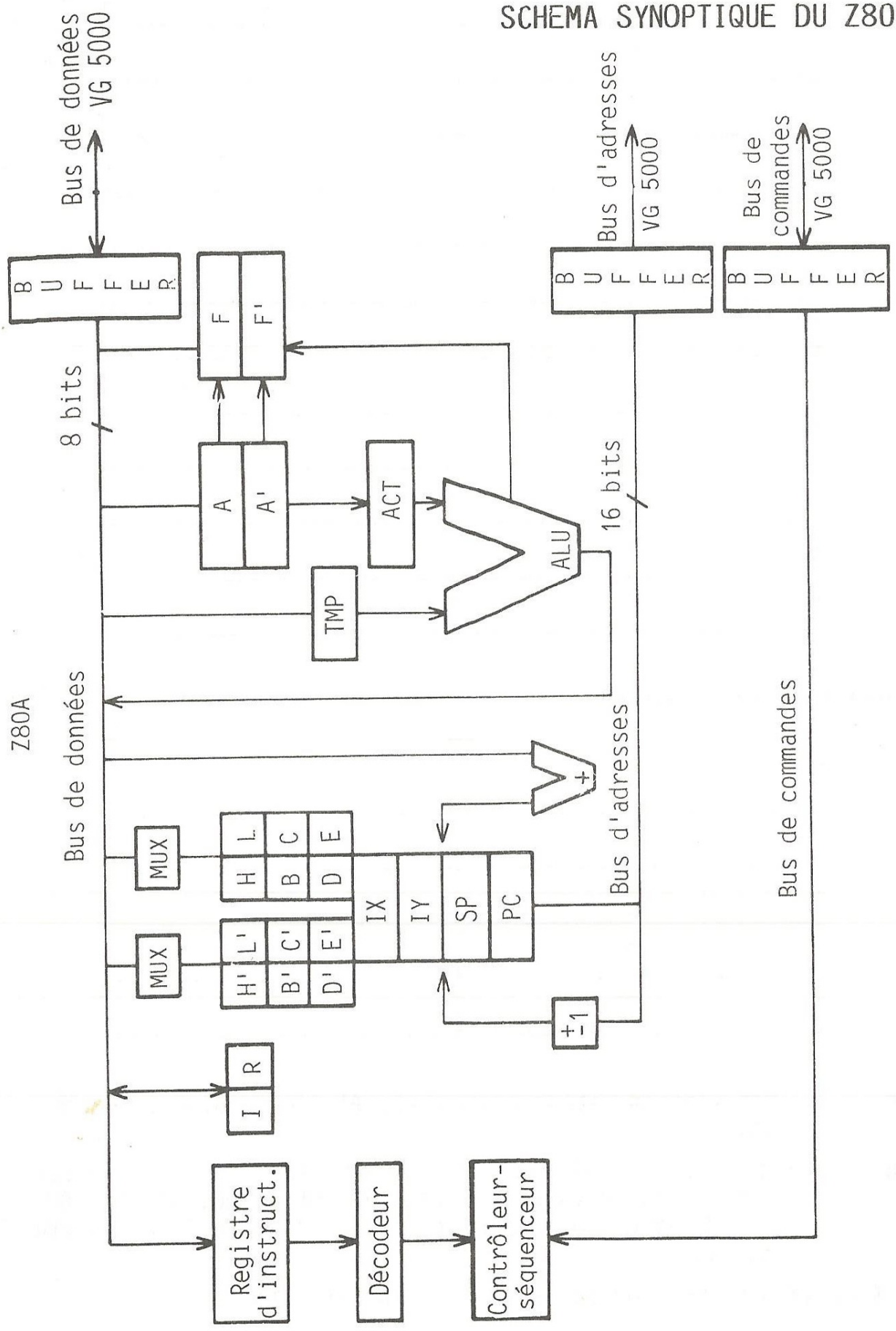



56 38 8
57 39 9
58 3A :
59 3B ;
60 3C <
61 3D =
62 3E >
63 3F ?
64 40 @
65 41 A
66 42 B
67 43 C
68 44 D
69 45 E
70 46 F
71 47 G
72 48 A
73 49 I
74 4A J
75 4B K
76 4C L
77 4D M
78 4E N
79 4F O
80 50 P
81 51 Q
82 52 R
83 53 S
84 54 T
85 55 U
86 56 V
87 57 W
88 58 X
89 59 Y
90 5A Z
91 5B [
92 5C
93 5D]
94 5E ^
95 5F _
96 60
97 61 a
98 62 b
99 63 c
100 64 d
101 65 e
102 66 f
103 67 g
104 68 h
105 69 i
106 6A j
107 6B k
108 6C l
109 6D m
110 6E n
111 6F o
112 70 p
113 71 q
114 72 r
115 73 s
116 74 t
117 75 u
118 76 v
119 77 w
120 78 x
121 79 y
122 7A z
123 7B
124 7C |
125 7D
126 7E
127 7F

GRILLE ECRAN



SCHEMA SYNOPTIQUE DU Z80



REGISTRES INTERNES

Registres principaux

7	07	0
A		F
I		R

15	87	0
B		C
D		E
H		L
IX		
IY		
SP		
PC		

Registres secondaires

7	07	0
A'		F'

15	07	0
B'		C'
D'		E'
H'		L'

A est l'accumulateur principal, **A'** l'accumulateur secondaire.

B, C, D, E, H et **L** sont des registres 8 bits qui s'apparient pour donner les registres 16 bits **BC**, **DE** et **HL**. **B', C', D', E', H' et L'** sont leurs équivalents en registres secondaires.

IX et IY sont des registres d'index sur 16 bits.

SP est le pointeur de pile sur 16 bits.

PC est le compteur ordinal sur 16 bits.

F est le registre d'état sur 8 bits, **F'** son équivalent en registre secondaire.

Détail du registre d'état

bit 7							bit 0
S	Z		H		P/V	N	C

Le bit 0 est le drapeau de retenue.

Le bit 1 est le drapeau d'opération précédente (1 si soustraction, 0 si addition).

Le bit 2 est le drapeau de parité/dépassement de capacité (=1 => nombre pair de bits à 1 ou bien débordement). Le sens de ce drapeau dépend de l'opération (logique ou arithmétique).

Le bit 3 est inutilisé.

Le bit 4 est le drapeau de demi-retenue (retenue du bit 3 sur le bit 4).

Le bit 5 est inutilisé.

Le bit 6 est le drapeau de zéro (1 si zéro, 0 sinon).

Le bit 7 est le drapeau de signe. Il est égal au bit 7 du nombre.

Détail de l'utilisation des registres Z80 par le VG 5000

AF Peut être altéré sans problème.

BC Idem.

DE Idem.

HL C'est le pointeur de texte Basic. Il faut le préserver (l'empiler en début de programme et le dépiler à la fin), si l'on ne veut pas avoir d'erreurs Basic.

AF' Peut être altéré sans problème.

BC' Idem.

DE' Idem.

HL' Idem.

IX Doit être préservé pour un appel aux routines situées en $0018, $001B, $008F, $009E, $00A1, $00AA, $00B9.

IY N'est jamais utilisé par le Basic.

I Idem.

R Idem.

Remarque préliminaire : dans la suite de ce mémento, toutes les valeurs hexadécimales seront précédées (sauf mention spéciale) d'un dollar "$".

Exemple : valeur décimale 255
valeur hexadécimale $FF

L'adressage implicite ou inhérent

Toutes les instructions sont sur un octet. Le code opération spécifie implicitement les informations permettant la réalisation de l'instruction.

Exemple : NOP SCF

L'adressage registre

Dans ce mode, la donnée se trouve dans un registre.

Exemple : LD A,B BIT 2,B EX DE,HL

L'adressage immédiat

La donnée est spécifiée dans l'instruction, elle suit immédiatement le code opération.

Exemple : LD A,$78 ADD A,$20

L'adressage direct

On prend la donnée directement dans l'adresse spécifiée derrière l'instruction.

Exemple : LD L,($3485) JP $7406

L'adressage indirect

Dans ce mode, l'adresse définitive est obtenue par chargement de l'adresse contenue dans un registre.

Exemple : LD A,(HL)

L'adressage relatif

Dans ce mode, l'opérande sera ajouté en complément à deux au compteur ordinal PC.

Exemple : JR $7684

L'adressage indirect indexé

Ce mode reprend le principe de l'adressage indirect, mais en lui ajoutant un déplacement signé. On utilise uniquement les registres IX et IY.

Exemple : LD (IY+8),C

Abréviations utilisées

sr représente un registre sur 8 bits (A, B, C, D, E, H, L, parfois I).
dr représente un registre sur 16 bits (BC, DE, HL, parfois IX, IY, SP).
op représente une donnée sur 8 bits.
ad représente une adresse sur 16 bits.
co représente une condition (C, M, NC, NZ, P, PE, PO, Z).
b représente le numéro d'un bit (entre 0 et 7).

ADC A,(HL)	Addition avec retenue de l'octet pointé par HL à A.
ADC A,(IX+op)	Addition avec retenue de l'octet pointé par IX+op.
ADC A,(IY+op)	Addition avec retenue de l'octet pointé par IY+op.
ADC A,op	Addition avec retenue de l'opérande op à A.
ADC HL,dr	Addition avec retenue du double registre dr à HL.
ADD A,(HL)	Addition sans retenue de l'octet pointé par HL à A.
ADD A,(IX+op)	Addition sans retenue de l'octet pointé par IX+op.
ADD A,(IY+op)	Addition sans retenue de l'octet pointé par IY+op.
ADD A,op	Addition sans retenue de l'opérande op à A.
ADD A,sr	Addition sans retenue du registre simple sr à A.
ADD HL,dr	Addition sans retenue du double registre dr à HL.
ADD IX,dr	Addition sans retenue du double registre dr à IX.
ADD IY,dr	Addition sans retenue du double registre dr à IY.
AND (HL)	Et logique entre l'octet pointé par HL à A.
AND (IX+op)	Et logique entre l'octet pointé par IX+op et A.
AND (IY+op)	Et logique entre l'octet pointé par IY+op et A.
AND op	Et logique entre l'opérande op et A.
AND sr	Et logique entre le registre simple sr et A.
BIT b,(HL)	Test du bit b de l'octet pointé par HL.
BIT b,(IX+op)	Test du bit b de l'octet pointé par IX plus op.
BIT b,(IY+op)	Test du bit b de l'octet pointé par IY plus op.
BIT b,sr	Test du bit b du registre simple dr.
CALL co,ad	Appel d'un sous-programme à l'adresse ad si la condition co est vraie.
CALL ad	Appel d'un sous-programme à l'adresse ad.
CCF	Complémente l'indicateur de retenue.
CP (HL)	Compare l'octet pointé par HL à A.
CP (IX+op)	Compare l'octet pointé par IX+op à A.
CP (IY+op)	Compare l'octet pointé par IY+op à A.
CP sr	Compare le registre simple sr à A.
CP op	Compare l'opérande op à A.

CPD	Compare le contenu de l'octet pointé par HL, décrémente HL et BC.
CPDR	Compare le contenu de l'octet pointé par HL, décrémente HL et BC et répète la séquence jusqu'à ce que BC=0.
CPI	Compare le contenu de l'octet pointé par HL, incrémente HL et décrémente BC.
CPIR	Compare le contenu de l'octet pointé par HL, incrémente HL, décrémente BC et répète la séquence jusqu'à ce que BC=0.
CPL.	Complémente A.
DAA	Ajustement décimal de A.
DEC (HL)	Décrémente l'octet pointé par HL.
DEC (IX+op)	Décrémente l'octet pointé par IX+op.
DEC (IY+op)	Décrémente l'octet pointé par IY+op.
DEC sr	Décrémente le registre simple sr.
DEC dr	Décrémente le double registre dr.
DI	Désactive les interruptions.
DJNZ op	Décrémente B et fait un saut relativement à op si B<>0.
EI	Active les interruptions.
EX AF,AF'	Echange les contenus des registres AF et AF'.
EX DE,HL	Echange les contenus des registres DE et HL.
EX (SP),HL	Echange le contenu des deux octets pointés par SP avec HL.
EX (SP),IX	Echange le contenu des deux octets pointés par SP avec IX.
EX (SP),IY	Echange le contenu des deux octets pointés par SP avec IY.
EXX	Echange le contenu des registres BC, DE et HL avec BC', DE' et HL'.
HALT	Arrêt et attente d'une interruption.
IM 0	Positionne le mode 0 d'interruptions.
IM 1	Positionne le mode 1 d'interruptions.
IM 2	Positionne le mode 2 d'interruptions.
IN sr,(C)	Charge le registre simple sr avec la donnée du port d'entrées-sorties adressé par C.
IN A,(op)	Charge A avec la donnée du port d'entrées-sorties n.
INC (HL)	Incrémente l'octet pointé par HL.
INC (IX+op)	Incrémente l'octet pointé par IX+op.
INC (IY+op)	Incrémente l'octet pointé par IY+op.
INC sr	Incrémente le registre simple sr.
INC dr	Incrémente le registre double dr.
IND	Charge l'octet pointé par HL avec la donnée du port d'entrées-sorties adressé par C. Décrémente HL et B.

INDR	Charge l'octet pointé par HL avec la donnée du port d'entrées-sorties adressée par C. Décrémente HL et B et répète la séquence jusqu'à ce que B=0.
INI	Charge l'octet pointé par HL avec la donnée du port d'entrées-sorties adressée par C. Incrémente HL et décrémente B.
INIR	Charge l'octet pointé par HL avec la donnée du port d'entrées-sorties adressée par C. Incrémente HL, décrémente BC et répète la séquence jusqu'à ce que B=0.
JP co,ad	Effectue un saut à l'adresse ad si la condition co est vraie.
JP ad	Effectue un saut à l'adresse ad.
JP (HL)	Effectue un saut à l'adresse donnée par le contenu de HL.
JP (IX)	Effectue un saut à l'adresse donnée par le contenu de IX.
JP (IY)	Effectue un saut à l'adresse donnée par le contenu de IY.
JP co,op	Effectue un saut relativement à op si la condition co est vraie.
JR op	Effectue un saut relativement à op.
LD dr,(ad)	Charge le double registre dr avec les deux octets pointés par ad.
LD dr,ad	Charge le double registre dr avec la valeur ad.
LD sr,op	Charge le registre simple sr avec la valeur op.
LD sr,sr'	Charge le registre sr avec le registre sr'.
LD sr,(dr)	Charge le registre simple sr avec l'octet pointé par le double registre dr. Si le registre simple n'est pas A, le registre double ne peut être que HL.
LD sr,(IX+op)	Charge le registre simple sr avec l'octet pointé par IX+op.
LD sr,(IY+op)	Charge le registre simple sr avec l'octet pointé par IY+op.
LD (dr),A	Charge à l'adresse pointée par dr la valeur de A.
LD (HL),sr	Charge à l'adresse pointée par HL la valeur du registre simple sr.
LD (HL),op	Charge à l'adresse pointée par HL la valeur op.
LD (IX+op),sr	Charge à l'adresse pointée par IX+op la valeur du registre simple ar.
LD (IX+op),op'	Charge à l'adresse pointée par IX+op la valeur op'.
LD (IY+op),sr	Charge à l'adresse pointée par IY+op la valeur du registre simple ar.
LD (IY+op),op'	Charge à l'adresse pointée par IY+op la valeur de op'.
LD (ad),A	Charge à l'adresse pointée par ad la valeur de A.
LD (ad),dr	Charge à l'adresse pointée par ad la valeur du registre double dr.

LDD	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, décrémente BC, HL et DE.
LDDR	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, décrémente BC, HL et DE, et répète la séquence jusqu'à ce que BC=0.
NEG	Complémente à deux A (inversion de signes).
NOP	Pas d'opération.
OR (HL)	Ou logique entre A et l'octet pointé par HL.
OR (IX+op)	Ou logique entre A et l'octet pointé par IX+op.
OR (IY+op)	Ou logique entre A et l'octet pointé par IY+op.
OR sr	Ou logique entre A et le registre simple sr.
OR op	Ou logique entre A et l'opérande op.
OTDR	Charge le port d'entrées-sorties adressé par C avec le contenu de l'adresse pointée par HL, décrémente HL et B et répète la séquence jusqu'à ce que B=0.
OTIR	Charge le port d'entrées-sorties adressé par C, incrémente HL, décrémente B et répète la séquence jusqu'à ce que B=0.
OUT (C),sr	Charge le port d'entrées-sorties adressé par C avec le contenu du registre simple sr.
OUT (op),A	Charge le port d'entrées-sorties op avec le contenu de A.
POP dr	Charge le double registre dr avec la valeur en sommet de pile (dépilement).
PUSH dr	Sauve le double registre dr en sommet de pile (empilement).
RES b,(HL)	Met à zéro le bit b de l'octet pointé par HL.
RES b,(IX+op)	Met à zéro le bit b de l'octet pointé par IX+op.
RES b,(IY+op)	Met à zéro le bit b de l'octet pointé par IY+op.
RES b,sr	Met à zéro le bit b du registre simple sr.
RET	Retour d'un sous-programme.
RET co	Retour d'un sous-programme si la condition co est vraie.
RETI	Retour d'un sous-programme de traitement d'interruptions.
RETN	Retour d'un sous-programme de traitement d'interruptions non masquables.
RL (HL)	Rotation à gauche via la retenue de l'octet pointé par HL.
RL (IX+op)	Rotation à gauche via la retenue de l'octet pointé par IX+op.
RL (IY+op)	Rotation à gauche via la retenue de l'octet pointé par IY+op.
RL sr	Rotation à gauche via la retenue du registre simple sr.
RLC (HL)	Rotation circulaire à gauche de l'octet pointé par HL.

RLC (IX+op)	Rotation circulaire à gauche de l'octet pointé par IX+op.
RLC (IY+op)	Rotation circulaire à gauche de l'octet pointé par IY+op.
RLC sr	Rotation circulaire à gauche du registre simple sr.
RLD	Rotation à gauche d'un quartet entre A et l'octet pointé par HL.
RR (HL)	Rotation à droite via la retenue de l'octet pointé par HL.
RR (IX+op)	Rotation à droite via la retenue de l'octet pointé par IX+op.
RR (IY+op)	Rotation à droite via la retenue de l'octet pointé par IY+op.
RR sr	Rotation à droite via la retenue du registre simple sr.
RRC (HL)	Rotation circulaire à droite de l'octet pointé par HL.
RRC (IX+op)	Rotation circulaire à droite de l'octet pointé par IX+op.
RRC (IY+op)	Rotation circulaire à droite de l'octet pointé par IY+op.
RRC sr	Rotation circulaire à droite du registre simple sr.
RRD	Rotation à droite d'un quartet entre A et l'octet pointé par HL.
RST op	Saut à l'adresse op.
SBC A,(HL)	Soustraction avec retenue de l'octet pointé par HL à A.
SBC A,(IX+op)	Soustraction avec retenue de l'octet pointé par IX+op à A.
SBC A,(IY+op)	Soustraction avec retenue de l'octet pointé par IY+op à A.
SBC HL,dr	Soustraction avec retenue du double registre dr à HL.
SCF	Positionne l'indicateur de retenue à 1.
SET b,HL	Met à 1 le bit b de l'octet pointé par HL.
SET b,(IX+op)	Met à 1 le bit b de l'octet pointé par IX+op.
SET b,(IY+op)	Met à 1 le bit b de l'octet pointé par IY+op.
SLA (HL)	Décalage arithmétique à gauche de l'octet pointé par HL.
SLA (IX+op)	Décalage arithmétique à gauche de l'octet pointé par IX+op.
SLA (IY+op)	Décalage arithmétique à gauche de l'octet pointé par IY+op.
SLA sr	Décalage arithmétique à gauche du registre simple sr.
SRA (HL)	Décalage arithmétique à droite de l'octet pointé par HL.
SRA (IX+op)	Décalage arithmétique à droite de l'octet pointé par IX+op.

SRA (IX+op)	Décalage arithmétique à droite de l'octet pointé par IX+op.
SRA (IY+op)	Décalage arithmétique à droite de l'octet pointé par IY+op.
SRA sr	Décalage arithmétique à droite du registre simple sr.
SRL (HL)	Décalage logique à droite de l'octet pointé par HL.
SRL (IX+op)	Décalage logique à droite de l'octet pointé par IX+op.
SRL (IY+op)	Décalage logique à droite de l'octet pointé par IY+op.
SRL sr	Décalage logique à droite du registre simple sr.
SUB (HL)	Soustraction sans retenue de l'octet pointé par HL à A.
SUB (IX+op)	Soustraction sans retenue de l'octet pointé par IX+op à A.
SUB (IY+op)	Soustraction sans retenue de l'octet pointé par IY+op à A.
SUB sr	Soustraction sans retenue du registre simple sr à A.
SUB op	Soustraction sans retenue de l'opérande op à A.
XOR (HL)	Ou exclusif entre A et l'octet pointé par HL.
XOR (IX+op)	Ou exclusif entre A et l'octet pointé par IX+op.
XOR (IY+op)	Ou exclusif entre A et l'octet pointé par IY+op.
XOR sr	Ou exclusif entre A et le registre simple sr.
XOR op	Ou exclusif entre A et l'opérande op.

Codes des indicateurs d'état

*	Drapeau affecté par l'opération.
0	Drapeau mis à 0 par l'opération.
1	Drapeau mis à 1 par l'opération.
?	Drapeau affecté de manière non significative par l'opération.
Rien	Drapeau non affecté par l'opération.

Abréviations utilisées pour les opérandes

op	Valeur sur 8 bits.
ad	Valeur sur 16 bits.

Mnémonique	*Code objet (hexa)*	*Cycles*	*Registre d'état*						*Adressage*
			S	*Z*	*H*	*P/V*	*N*	*C*	
ADC A,(HL)	8E	7	*	*	*	*	0	*	Indirect
ADC A,(IX+op)	DD8Eop	19	*	*	*	*	0	*	Indexé
ADC A,(IY+op)	FD8Eop	19	*	*	*	*	0	*	Indexé
ADC A,A	8F	4	*	*	*	*	0	*	Registre
ADC A,B	88	4	*	*	*	*	0	*	Registre
ADC A,C	89	4	*	*	*	*	0	*	Registre
ADC A,D	8A	4	*	*	*	*	0	*	Registre
ADC A,E	8B	4	*	*	*	*	0	*	Registre
ADC A,H	8C	4	*	*	*	*	0	*	Registre
ADC A,L	8D	4	*	*	*	*	0	*	Registre
ADC A,op	CEop	7	*	*	*	*	0	*	Immédiat
ADC HL,BC	ED4A	15	*	*	?	*	0	*	Registre
ADC HL,DE	ED5A	15	*	*	?	*	0	*	Registre
ADC HL,HL	ED6A	15	*	*	?	*	0	*	Registre
ADC HL,SP	ED7A	15	*	*	?	*	0	*	Registre
ADD A,(HL)	86	7	*	*	*	*	0	*	Indirect
ADD A,(IX+op)	DD86op	19	*	*	*	*	0	*	Indexé
ADD A,(IY+op)	FD86op	19	*	*	*	*	0	*	Indexé
ADD A,A	87	4	*	*	*	*	0	*	Registre
ADD A,B	80	4	*	*	*	*	0	*	Registre
ADD A,C	81	4	*	*	*	*	0	*	Registre
ADD A,D	82	4	*	*	*	*	0	*	Registre
ADD A,E	83	4	*	*	*	*	0	*	Registre
ADD A,H	84	4	*	*	*	*	0	*	Registre
ADD A,L	85	4	*	*	*	*	0	*	Registre
ADD A,op	C6op	7	*	*	*	*	0	*	Immédiat
ADD HL,BC	ED4A	11			?		0	*	Registre
ADD HL,DE	ED5A	11			?		0	*	Registre
ADD HL,HL	ED6A	11			?		0	*	Registre
ADD HL,SP	ED7A	11			?		0	*	Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
ADD IX,BC	DD09	15			?		0	*	Registre
ADD IX,DE	DD19	15			?		0	*	Registre
ADD IX,IX	DD29	15			?		0	*	Registre
ADD IX,SP	DD39	15			?		0	*	Registre
ADD IY,BC	FD09	15			?		0	*	Registre
ADD IY,DE	FD19	15			?		0	*	Registre
ADD IY,IY	FD29	15			?		0	*	Registre
ADD IY,SP	FD39	15			?		0	*	Registre
AND (HL)	A6	7	*	*	1	*	0	0	Indirect
AND (IX+op)	DDA6op	19	*	*	1	*	0	0	Indexé
AND (IY+op)	FDA6op	19	*	*	1	*	0	0	Indexé
AND A	A7	4	*	*	1	*	0	0	Registre
AND B	A0	4	*	*	1	*	0	0	Registre
AND C	A1	4	*	*	1	*	0	0	Registre
AND D	A2	4	*	*	1	*	0	0	Registre
AND E	A3	4	*	*	1	*	0	0	Registre
AND H	A4	4	*	*	1	*	0	0	Registre
AND L	A5	4	*	*	1	*	0	0	Registre
AND op	E6op	7	*	*	1	*	0	0	Immédiat
BIT 0,(HL)	CB46	12	?	*	1	?	0		Indirect
BIT 0,(IX+op)	DDCBop46	20	?	*	1	?	0		Indexé
BIT 0,(IY+op)	FDCBop46	20	?	*	1	?	0		Indexé
BIT 0,A	CB47	8	?	*	1	?	0		Registre
BIT 0,B	CB40	8	?	*	1	?	0		Registre
BIT 0,C	CB41	8	?	*	1	?	0		Registre
BIT 0,D	CB42	8	?	*	1	?	0		Registre
BIT 0,E	CB43	8	?	*	1	?	0		Registre
BIT 0,H	CB44	8	?	*	1	?	0		Registre
BIT 0,L	CB45	8	?	*	1	?	0		Registre
BIT 1,(HL)	CB4E	12	?	*	1	?	0		Indirect
BIT 1,(IX+op)	DDCBop4E	20	?	*	1	?	0		Indexé
BIT 1,(IY+op)	FDCBop4E	20	?	*	1	?	0		Indexé
BIT 1,A	CB4F	8	?	*	1	?	0		Registre
BIT 1,B	CB48	8	?	*	1	?	0		Registre
BIT 1,C	CB49	8	?	*	1	?	0		Registre
BIT 1,D	CB4A	8	?	*	1	?	0		Registre
BIT 1,E	CB4B	8	?	*	1	?	0		Registre
BIT 1,H	CB4C	8	?	*	1	?	0		Registre
BIT 1,L	CB4D	8	?	*	1	?	0		Registre
BIT 2,(HL)	CB56	12	?	*	1	?	0		Indirect
BIT 2,(IX+op)	DDCBop56	20	?	*	1	?	0		Indexé
BIT 2,(IY+op)	FDCBop56	20	?	*	1	?	0		Indexé
BIT 2,A	CB57	8	?	*	1	?	0		Registre
BIT 2,B	CB50	8	?	*	1	?	0		Registre
BIT 2,C	CB51	8	?	*	1	?	0		Registre
BIT 2,D	CB52	8	?	*	1	?	0		Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
BIT 2,E	CB53	8	?	*	1	?	0		Registre
BIT 2,H	CB54	8	?	*	1	?	0		Registre
BIT 2,L	CB55	8	?	*	1	?	0		Registre
BIT 3,(HL)	CB5E	12	?	*	1	?	0		Indirect
BIT 3,(IX+op)	DDCBop5E	20	?	*	1	?	0		Indexé
BIT 3,(IY+op)	FDCBop5E	20	?	*	1	?	0		Indexé
BIT 3,A	CB5F	8	?	*	1	?	0		Registre
BIT 3,B	CB58	8	?	*	1	?	0		Registre
BIT 3,C	CB59	8	?	*	1	?	0		Registre
BIT 3,D	CB5A	8	?	*	1	?	0		Registre
BIT 3,E	CB5B	8	?	*	1	?	0		Registre
BIT 3,H	CB5C	8	?	*	1	?	0		Registre
BIT 3,L	CB5D	8	?	*	1	?	0		Registre
BIT 4,(HL)	CB66	12	?	*	1	?	0		Indirect
BIT 4,(IX+op)	DDCBop66	20	?	*	1	?	0		Indexé
BIT 4,(IY+op)	FDCop66	20	?	*	1	?	0		Indexé
BIT 4,A	CB67	8	?	*	1	?	0		Registre
BIT 4,B	CB60	8	?	*	1	?	0		Registre
BIT 4,C	CB61	8	?	*	1	?	0		Registre
BIT 4,D	CB62	8	?	*	1	?	0		Registre
BIT 4,E	CB63	8	?	*	1	?	0		Registre
BIT 4,H	CB64	8	?	*	1	?	0		Registre
BIT 4,L	CB65	8	?	*	1	?	0		Registre
BIT 5,(HL)	CB6E	12	?	*	1	?	0		Indirect
BIT 5,(IX+op)	DDCBop6E	20	?	*	1	?	0		Indexé
BIT 5,(IY+op)	FDCBop6E	20	?	*	1	?	0		Indexé
BIT 5,A	CB6F	8	?	*	1	?	0		Registre
BIT 5,B	CB68	8	?	*	1	?	0		Registre
BIT 5,C	CB69	8	?	*	1	?	0		Registre
BIT 5,D	CB6A	8	?	*	1	?	0		Registre
BIT 5,E	CB6B	8	?	*	1	?	0		Registre
BIT 5,H	CB6C	8	?	*	1	?	0		Registre
BIT 5,L	CB6D	8	?	*	1	?	0		Registre
BIT 6,(HL)	CB76	12	?	*	1	?	0		Indirect
BIT 6,(IX+op)	DDCBop76	20	?	*	1	?	0		Indexé
BIT 6,(IY+op)	FDCBop76	20	?	*	1	?	0		Indexé
BIT 6,A	CB77	8	?	*	1	?	0		Registre
BIT 6,B	CB70	8	?	*	1	?	0		Registre
BIT 6,C	CB71	8	?	*	1	?	0		Registre
BIT 6,D	CB72	8	?	*	1	?	0		Registre
BIT 6,E	CB73	8	?	*	1	?	0		Registre
BIT 6,H	CB74	8	?	*	1	?	0		Registre
BIT 6,L	CB75	8	?	*	1	?	0		Registre
BIT 7,(HL)	CB7E	12	?	*	1	?	0		Indirect
BIT 7,(IX+op)	DDCBop7E	20	?	*	1	?	0		Indexé

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
BIT 7,(IY+op)	FDCBop7E	20	?	*	1	?	0		Indexé
BIT 7,A	CB7F	8	?	*	1	?	0		Registre
BIT 7,B	CB78	8	?	*	1	?	0		Registre
BIT 7,C	CB79	8	?	*	1	?	0		Registre
BIT 7,D	CB7A	8	?	*	1	?	0		Registre
BIT 7,E	CB7B	8	?	*	1	?	0		Registre
BIT 7,H	CB7C	8	?	*	1	?	0		Registre
BIT 7,L	CB7D	8	?	*	1	?	0		Registre
CALL C,ad	DCad	17/10							Immédiat
CALL M,ad	FCad	17/10							Immédiat
CALL NC,ad	D4ad	17/10							Immédiat
CALL NZ,ad	C4ad	17/10							Immédiat
CALL P,ad	F4ad	17/10							Immédiat
CALL PE,ad	ECad	17/10							Immédiat
CALL PO,ad	E4ad	17/10							Immédiat
CALL Z,ad	CCad	17/10							Immédiat
CALL ad	CDad	17							Immédiat
CCF	3F	4			?		0	*	Implicite
CP (HL)	BE	7	*	*	*	*	1	*	Indirect
CP (IX+op)	DDBEop	19	*	*	*	*	1	*	Indexé
CP (IY+op)	FDBEop	19	*	*	*	*	1	*	Indexé
CP A	BF	4	*	*	*	*	1	*	Registre
CP B	B8	4	*	*	*	*	1	*	Registre
CP C	B9	4	*	*	*	*	1	*	Registre
CP D	BA	4	*	*	*	*	1	*	Registre
CP E	BB	4	*	*	*	*	1	*	Registre
CP H	BC	4	*	*	*	*	1	*	Registre
CP L	BD	4	*	*	*	*	1	*	Registre
CP op	FEop	7	*	*	*	*	1	*	Immédiat
CPD	EDA9	16	*	*	*	*	1		Implicite
CPDR	EDB9	16/21	*	*	*	*	1		Implicite
CPI	EDA1	16	*	*	*	*	1		Implicite
CPIR	EDB1	16/21	*	*	*	*	1		Implicite
CPL	2F	4			1		1		Implicite
DAA	27	4	*	*	*	*		*	Implicite
DEC (HL)	35	11	*	*	*	*	1	*	Indirect
DEC (IX+op)	DD35op	23	*	*	*	*	1	*	Indexé
DEC (IY+op)	FD35op	23	*	*	*	*	1	*	Indexé
DEC A	3D	4	*	*	*	*	1	*	Registre
DEC B	05	4	*	*	*	*	1	*	Registre
DEC C	0D	4	*	*	*	*	1	*	Registre
DEC D	15	4	*	*	*	*	1	*	Registre
DEC E	1D	4	*	*	*	*	1	*	Registre
DEC H	25	4	*	*	*	*	1	*	Registre
DEC L	2D	4	*	*	*	*	1	*	Registre

TABLEAU D'ASSEMBLAGE

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
DEC BC	0B	6							Registre
DEC DE	1B	6							Registre
DEC HL	2B	6							Registre
DEC SP	3B	6							Registre
DEC IX	DD2B	10							Registre
DEC IY	FD2B	10							Registre
DI	F3	4							Implicite
DJNZ op	10op	8/13							Relatif
EI	FB	4							Implicite
EX (SP),HL	E3	19							Indirect
EX (SP),IX	DDE3	23							Indirect
EX (SP),IY	FDE3	23							Indirect
EX AF,AF'	08	4							Implicite
EX DE,HL	EB	4							Implicite
EXX	D9	4							Implicite
HALT	76	4							Implicite
IM0	ED46	4							Implicite
IM1	ED56	4							Implicite
IM2	ED5E	4							Implicite
IN A,(C)	ED78	12	*	*	*	*	0		Indirect
IN B,(C)	ED40	12	*	*	*	*	0		Indirect
IN C,(C)	ED48	12	*	*	*	*	0		Indirect
IN D,(C)	ED50	12	*	*	*	*	0		Indirect
IN E,(C)	ED58	12	*	*	*	*	0		Indirect
IN H,(C)	ED60	12	*	*	*	*	0		Indirect
IN L,(C)	ED68	12	*	*	*	*	0		Indirect
IN A,(op)	DBop	11							Indirect
INC (HL)	34	11	*	*	*	*	0		Indirect
INC (IX+op)	DD34op	23	*	*	*	*	0		Indexé
INC (IY+op)	FD34op	23	*	*	*	*	0		Indexé
INC A	3C	4	*	*	*	*	0		Registre
INC B	04	4	*	*	*	*	0		Registre
INC C	0C	4	*	*	*	*	0		Registre
INC D	14	4	*	*	*	*	0		Registre
INC E	1C	4	*	*	*	*	0		Registre
INC H	24	4	*	*	*	*	0		Registre
INC L	2C	4	*	*	*	*	0		Registre
INC BC	03	6							Registre
INC DE	13	6							Registre
INC HL	23	6							Registre
INC SP	33	6							Registre
INC IX	DD23	10							Registre
INC IY	FD23	10							Registre
IND	EDAA	16	?	*	?	?	1		Implicite
INDR	EDBA	16/21	?	1	?	?	1		Implicite
INI	EDA2	16	?	*	?	?	1		Implicite

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
INIR	EDB2	16/21	?	1	?	?	1		Implicite
JP ad	C3ad	10							Immédiat
JP (HL)	E9	4							Indirect
JP (IX)	DDE9	8							Indirect
JP (IY)	FDE9	8							Indirect
JP C,ad	DAad	10							Immédiat
JP M,ad	FAad	10							Immédiat
JP NC,ad	D2ad	10							Immédiat
JP NZ,ad	C2ad	10							Immédiat
JP P,ad	F2ad	10							Immédiat
JP PE,ad	EAad	10							Immédiat
JP PO,ad	E2ad	10							Immédiat
JP Z,ad	CAad	12/7							Immédiat
JR C,op	38op	12/7							Relatif
JR NC,op	30op	12/7							Relatif
JR NZ,op	20op	12/7							Relatif
JR Z,op	28op	12/7							Relatif
JR op	18op	12							Relatif
LD (BC),A	02	7							Indirect
LD (DE),A	12	7							Indirect
LD (HL),A	77	7							Indirect
LD (HL),B	70	7							Indirect
LD (HL),C	71	7							Indirect
LD (HL),D	72	7							Indirect
LD (HL),E	73	7							Indirect
LD (HL),H	74	7							Indirect
LD (HL),L	75	7							Indirect
LD (HL),op	36op	10							Imm./ind.
LD (IX+op),A	DD77op	19							Indexé
LD (IX+op),B	DD70op	19							Indexé
LD (IX+op),C	DD71op	19							Indexé
LD (IX+op),D	DD72op	19							Indexé
LD (IX+op),E	DD73op	19							Indexé
LD (IX+op),H	DD74op	19							Indexé
LD (IX+op),L	DD75op	19							Indexé
LD (IX+op),op'	DD36opop'	19							Ind./imm.
LD (IY+op),A	FD77op	19							Indexé
LD (IY+op),B	FD70op	19							Indexé
LD (IY+op),C	FD71op	19							Indexé
LD (IY+op),D	FD72op	19							Indexé
LD (IY+op),E	FD73op	19							Indexé
LD (IY+op),H	FD74op	19							Indexé
LD (IY+op),L	FD75op	19							Indexé
LD (IY+op),op'	FD36opop'	19							Ind./imm.
LD (ad),A	32ad	13							Direct
LD (ad),BC	ED43ad	20							Direct

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
LD (ad),DE	ED53ad	20							Direct
LD (ad),HL	22ad	16							Direct
LD (ad),IX	DD22ad	20							Direct
LD (ad),IY	FD22ad	20							Direct
LD (ad),SP	ED73ad	20							Direct
LD A,(BC)	0A	7							Indirect
LD A,(DE)	1A	7							Indirect
LD A,(HL)	7E	7							Indirect
LD A,(IX+op)	DD7Eop	19							Indexé
LD A,(IY+op)	FD7Eop	19							Indexé
LD A,(ad)	3Aad	13							Direct
LD A,A	7F	4							Registre
LD A,B	78	4							Registre
LD A,C	79	4							Registre
LD A,D	7A	4							Registre
LD A,E	7B	4							Registre
LD A,H	7C	4							Registre
LD A,L	7D	4							Registre
LD A,op	3Eop	7							Immédiat
LD A,I	ED57	9	*	*	0	*	0		Implicite
LD A,R	ED5F	9	*	*	0	*	0		Implicite
LD B,(HL)	46	7							Indirect
LD B,(IX+op)	DD46op	19							Indexé
LD B,(IY+op)	FD46op	19							Indexé
LD B,A	47	4							Registre
LD B,B	40	4							Registre
LD B,C	41	4							Registre
LD B,D	42	4							Registre
LD B,E	43	4							Registre
LD B,H	44	4							Registre
LD B,L	45	4							Registre
LD B,op	06op	7							Immédiat
LD C,(HL)	4E	7							Indirect
LD C,(IX+op)	DD4Eop	19							Indexé
LD C,(IY+op)	FD4Eop	19							Indexé
LD C,A	4F	4							Registre
LD C,B	48	4							Registre
LD C,C	49	4							Registre
LD C,D	4A	4							Registre
LD C,E	4B	4							Registre
LD C,H	4C	4							Registre
LD C,L	4D	4							Registre
LD C,op	0Eop	7							Immédiat
LD D,(HL)	56	7							Indirect
LD D,(IX+op)	DD56op	19							Indexé

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
LD D,(IY+op)	FD56op	19							Indexé
LD D,A	57	4							Registre
LD D,B	50	4							Registre
LD D,C	51	4							Registre
LD D,D	52	4							Registre
LD D,E	53	4							Registre
LD D,H	54	4							Registre
LD D,L	55	4							Registre
LD D,OP	16op	7							Immédiat
LD E,(HL)	5E	7							Indirect
LD E,(IX+op)	DD5Eop	19							Indexé
LD E,(IY+op)	FD5Eop	19							Indexé
LD E,A	5F	4							Registre
LD E,B	58	4							Registre
LD E,C	59	4							Registre
LD E,D	5A	4							Registre
LD E,E	5B	4							Registre
LD E,H	5C	4							Registre
LD E,L	5D	4							Registre
LD E,op	1Eop	7							Immédiat
LD H,(HL)	66	7							Indirect
LD H,(IX+op)	DD66op	19							Indexé
LD H,(IY+op)	FD66op	19							Indexé
LD H,A	67	4							Registre
LD H,B	60	4							Registre
LD H,C	61	4							Registre
LD H,D	62	4							Registre
LD H,E	63	4							Registre
LD H,H	64	4							Registre
LD H,L	65	4							Registre
LD H,op	26op	7							Immédiat
LD L,(HL)	6E	7							Indirect
LD L,(IX+op)	DD6Eop	19							Indexé
LD L,(IY+op)	FD6Eop	19							Indexé
LD L,A	6F	4							Registre
LD L,B	68	4							Registre
LD L,C	69	4							Registre
LD L,D	6A	4							Registre
LD L,E	6B	4							Registre
LD L,H	6C	4							Registre
LD L,L	6D	4							Registre
LD L,op	2Eop	7							Immédiat
LD I,A	ED47	9							Implicite
LD R,A	ED4F	9							Implicite
LD BC,(ad)	ED4Bad	20							Direct

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
LD BC,ad	01ad	10							Immédiat
LD DE,(ad)	ED5Bad	20							Direct
LD DE,ad	11ad	10							Immédiat
LD HL,(ad)	2Aad	16							Direct
LD HL,ad	21ad	10							Immédiat
LD IX,(ad)	DD2Aad	20							Direct
LD IX,ad	DD21ad	14							Immédiat
LD IY,(ad)	FD2Aad	20							Direct
LD IY,ad	FD21ad	14							Immédiat
LD SP,(ad)	ED7Bad	20							Direct
LD SP,HL	F9	6							Implicite
LD SP,IX	DDF9	10							Implicite
LD SP,IY	FDF9	10							Implicite
LD SP,ad	31ad	10							Immédiat
LDD	EDA8	16			0	*	0		Indirect
LDDR	EDB8	16/21			0	0	0		Indirect
LDI	EDA0	16			0	*	0		Indirect
LDIR	EDB0	16/21			0	0	0		Indirect
NEG	ED44	8	*	*	*	*	1	*	Implicite
NOP	00	4							Implicite
OR (HL)	B6	7	*	*	0	*	0	0	Indirect
OR (IX+op)	DDB6op	19	*	*	0	*	0	0	Indexé
OR (IY+op)	FDB6op	19	*	*	0	*	0	0	Indexé
OR A	B7	4	*	*	0	*	0	0	Registre
OR B	B0	4	*	*	0	*	0	0	Registre
OR C	B1	4	*	*	0	*	0	0	Registre
OR D	B2	4	*	*	0	*	0	0	Registre
OR E	B3	4	*	*	0	*	0	0	Registre
OR H	B4	4	*	*	0	*	0	0	Registre
OR L	B5	4	*	*	0	*	0	0	Registre
OR op	F6op	7	*	*	0	*	0	0	Immédiat
OTDR	EDBB	16/21	?	1	?	?	1		Indirect
OTIR	EDB3	16/21	?	1	?	?	1		Indirect
OUT (C),A	ED79	12							Indirect
OUT (C),B	ED41	12							Indirect
OUT (C),C	ED49	12							Indirect
OUT (C),D	ED51	12							Indirect
OUT (C),E	ED59	12							Indirect
OUT (C),H	ED61	12							Indirect
OUT (C),L	ED69	12							Indirect
OUT (op),A	D3op	11							Indirect
OUTD	EDAB	16	?	*	?	?	1		Indirect
OUTI	EDA3	16	?	*	?	?	1		Indirect
POP AF	F1	10							Indirect
POP BC	C1	10							Indirect

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
POP DE	D1	10							Indirect
POP HL	E1	10							Indirect
POP IX	DDE1	14							Indirect
POP IY	FDE1	14							Indirect
PUSH AF	F5	11							Indirect
PUSH BC	C5	11							Indirect
PUSH DE	D5	11							Indirect
PUSH HL	E5	11							Indirect
PUSH IX	DDE5	15							Indirect
PUSH IY	FDE5	15							Indirect
RES 0,(HL)	CB86	15	?	*	1	?	0		Indirect
RES 0,(IX+op)	DDCBop86	23	?	*	1	?	0		Indexé
RES 0,(IY+op)	FDCBop86	23	?	*	1	?	0		Indexé
RES 0,A	CB87	8	?	*	1	?	0		Registre
RES 0,B	CB80	8	?	*	1	?	0		Registre
RES 0,C	CB81	8	?	*	1	?	0		Registre
RES 0,D	CB82	8	?	*	1	?	0		Registre
RES 0,E	CB83	8	?	*	1	?	0		Registre
RES 0,H	CB84	8	?	*	1	?	0		Registre
RES 0,L	CB85	8	?	*	1	?	0		Registre
RES 1,(HL)	CB8E	15	?	*	1	?	0		Indirect
RES 1,(IX+op)	DDCBop8E	23	?	*	1	?	0		Indexé
RES 1,(IY+op)	FDCBop8E	23	?	*	1	?	0		Indexé
RES 1,A	CB8F	8	?	*	1	?	0		Registre
RES 1,B	CB88	8	?	*	1	?	0		Registre
RES 1,C	CB89	8	?	*	1	?	0		Registre
RES 1,D	CB8A	8	?	*	1	?	0		Registre
RES 1,E	CB8B	8	?	*	1	?	0		Registre
RES 1,H	CB8C	8	?	*	1	?	0		Registre
RES 1,L	CB8D	8	?	*	1	?	0		Registre
RES 2,(HL)	CB96	15	?	*	1	?	0		Indirect
RES 2,(IX+op)	DDCBop96	23	?	*	1	?	0		Indexé
RES 2,(IY+op)	FDCBop96	23	?	*	1	?	0		Indexé
RES 2,A	CB97	8	?	*	1	?	0		Registre
RES 2,B	CB90	8	?	*	1	?	0		Registre
RES 2,C	CB91	8	?	*	1	?	0		Registre
RES 2,D	CB92	8	?	*	1	?	0		Registre
RES 2,E	CB93	8	?	*	1	?	0		Registre
RES 2,H	CB94	8	?	*	1	?	0		Registre
RES 2,L	CB95	8	?	*	1	?	0		Registre
RES 3,(HL)	CB9E	15	?	*	1	?	0		Indirect
RES 3,(IX+op)	DDCBop9E	23	?	*	1	?	0		Indexé
RES 3,(IY+op)	FDCBop9E	23	?	*	1	?	0		Indexé
RES 3,A	CB9F	8	?	*	1	?	0		Registre
RES 3,B	CB98	8	?	*	1	?	0		Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
RES 3,C	CB99	8	?	*	1	?	0		Registre
RES 3,D	CB9A	8	?	*	1	?	0		Registre
RES 3,E	CB9B	8	?	*	1	?	0		Registre
RES 3,H	CB9C	8	?	*	1	?	0		Registre
RES 3,L	CB9D	8	?	*	1	?	0		Registre
RES 4,(HL)	CBA6	15	?	*	1	?	0		Indirect
RES 4,(IX+op)	DDCBopA6	23	?	*	1	?	0		Indexé
RES 4,(IY+op)	FDCBopA6	23	?	*	1	?	0		Indexé
RES 4,A	CBA7	8	?	*	1	?	0		Registre
RES 4,B	CBA0	8	?	*	1	?	0		Registre
RES 4,C	CBA1	8	?	*	1	?	0		Registre
RES 4,D	CBA2	8	?	*	1	?	0		Registre
RES 4,E	CBA3	8	?	*	1	?	0		Registre
RES 4,H	CBA4	8	?	*	1	?	0		Registre
RES 4,L	CBA5	8	?	*	1	?	0		Registre
RES 5,(HL)	CBAE	15	?	*	1	?	0		Indirect
RES 5,(IX+op)	DDCBopAE	23	?	*	1	?	0		Indexé
RES 5,(IY+op)	FDCBopAE	23	?	*	1	?	0		Indexé
RES 5,A	CBAF	8	?	*	1	?	0		Registre
RES 5,B	CBA8	8	?	*	1	?	0		Registre
RES 5,C	CBA9	8	?	*	1	?	0		Registre
RES 5,D	CBAA	8	?	*	1	?	0		Registre
RES 5,E	CBAB	8	?	*	1	?	0		Registre
RES 5,H	CBAC	8	?	*	1	?	0		Registre
RES 5,L	CBAD	8	?	*	1	?	0		Registre
RES 6,(HL)	CBB6	15	?	*	1	?	0		Indirect
RES 6,(IX+op)	DDCBopB6	23	?	*	1	?	0		Indexé
RES 6,(IY+op)	FDCBopB6	23	?	*	1	?	0		Indexé
RES 6,A	CBB7	8	?	*	1	?	0		Registre
RES 6,B	CBB0	8	?	*	1	?	0		Registre
RES 6,C	CBB1	8	?	*	1	?	0		Registre
RES 6,D	CBB2	8	?	*	1	?	0		Registre
RES 6,E	CBB3	8	?	*	1	?	0		Registre
RES 6,H	CBB4	8	?	*	1	?	0		Registre
RES 6,L	CBB5	8	?	*	1	?	0		Registre
RES 7,(HL)	CBBE	15	?	*	1	?	0		Indirect
RES 7,(IX+op)	DDCBopBE	23	?	*	1	?	0		Indexé
RES 7,(IY+op)	FDCBopBE	23	?	*	1	?	0		Indexé
RES 7,A	CBBF	8	?	*	1	?	0		Registre
RES 7,B	CBB8	8	?	*	1	?	0		Registre
RES 7,C	CBB9	8	?	*	1	?	0		Registre
RES 7,D	CBBA	8	?	*	1	?	0		Registre
RES 7,E	CBBB	8	?	*	1	?	0		Registre
RES 7,H	CBBC	8	?	*	1	?	0		Registre
RES 7,L	CBBD	8	?	*	1	?	0		Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
RET	C9	10							Indirect
RET C	D8	5/11							Indirect
RET M	F8	5/11							Indirect
RET NC	D0	5/11							Indirect
RET NZ	C0	5/11							Indirect
RET P	F0	5/11							Indirect
RET PE	E8	5/11							Indirect
RET PO	E0	5/11							Indirect
RET Z	C8	5/11							Indirect
RETI	ED4D	14							Indirect
RETN	ED45	14							Indirect
RL (HL)	CB16	15	*	*	0	*	0	*	Indirect
RL (IX+op)	DDCBop16	23	*	*	0	*	0	*	Indexé
RL (IY+op)	FDCBop16	23	*	*	0	*	0	*	Indexé
RL A	CB17	8	*	*	0	*	0	*	Registre
RL B	CB10	8	*	*	0	*	0	*	Registre
RL C	CB11	8	*	*	0	*	0	*	Registre
RL D	CB12	8	*	*	0	*	0	*	Registre
RL E	CB13	8	*	*	0	*	0	*	Registre
RL H	CB14	8	*	*	0	*	0	*	Registre
RL L	CB15	8	*	*	0	*	0	*	Registre
RLA	17	4			0		0	*	Implicite
RLC (HL)	CB06	15	*	*	0	*	0	*	Indirect
RLC (IX+op)	DDCBop06	23	*	*	0	*	0	*	Indexé
RLC (IY+op)	FDCBop06	23	*	*	0	*	0	*	Indexé
RLC A	CB07	8	*	*	0	*	0	*	Registre
RLC B	CB00	8	*	*	0	*	0	*	Registre
RLC C	CB01	8	*	*	0	*	0	*	Registre
RLC D	CB02	8	*	*	0	*	0	*	Registre
RLC E	CB03	8	*	*	0	*	0	*	Registre
RLC H	CB04	8	*	*	0	*	0	*	Registre
RLC L	CB05	8	*	*	0	*	0	*	Registre
RLCA	07	4			0		0	*	Implicite
RLD	ED6F	18	*	*	0	*	0		Implicite
RR (HL)	CB1E	15	*	*	0	*	0	*	Indirect
RR (IX+op)	DDCBop1E	23	*	*	0	*	0	*	Indexé
RR (IY+op)	FDCBop1E	23	*	*	0	*	0	*	Indexé
RR A	CB1F	8	*	*	0	*	0	*	Registre
RR B	CB18	8	*	*	0	*	0	*	Registre
RR C	CB19	8	*	*	0	*	0	*	Registre
RR D	CB1A	8	*	*	0	*	0	*	Registre
RR E	CB1B	8	*	*	0	*	0	*	Registre
RR H	CB1C	8	*	*	0	*	0	*	Registre
RR L	CB1D	8	*	*	0	*	0	*	Registre
RRA	1F	4			0		0	*	Implicite

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
RRC (HL)	CB0E	15	*	*	0	*	0	*	Indirect
RRC (IX+op)	DDCBop0E	23	*	*	0	*	0	*	Indexé
RRC (IY+op)	FDCBop0E	23	*	*	0	*	0	*	Indexé
RRC A	CB0F	8	*	*	0	*	0	*	Registre
RRC B	CB08	8	*	*	0	*	0	*	Registre
RRC C	CB09	8	*	*	0	*	0	*	Registre
RRC D	CB0A	8	*	*	0	*	0	*	Registre
RRC E	CB0B	8	*	*	0	*	0	*	Registre
RRC H	CB0C	8	*	*	0	*	0	*	Registre
RRC L	CB0D	8	*	*	0	*	0	*	Registre
RRCA	0F	4			0	*	0	*	Implicite
RRD	ED67	18	*	*	0	*	0		Indirect
RST 00h	C7	11							Indirect
RST 08h	CF	11							Indirect
RST 10h	D7	11							Indirect
RST 18h	DF	11							Indirect
RST 20h	E7	11							Indirect
RST 28h	EF	11							Indirect
RST 30h	F7	11							Indirect
RST 38h	FF	11							Indirect
SBC A,op	DEop	7	*	*	*	*	1	*	Immédiat
SBC A,(HL)	9E	7	*	*	*	*	1	*	Indirect
SBC A,(IX+op)	DD9E	19	*	*	*	*	1	*	Indexé
SBC A,(IY+op)	FD9E	19	*	*	*	*	1	*	Indexé
SBC A,A	9F	4	*	*	*	*	1	*	Registre
SBC A,B	98	4	*	*	*	*	1	*	Registre
SBC A,C	99	4	*	*	*	*	1	*	Registre
SBC A,D	9A	4	*	*	*	*	1	*	Registre
SBC A,E	9B	4	*	*	*	*	1	*	Registre
SBC A,H	9C	4	*	*	*	*	1	*	Registre
SBC A,L	9D	4	*	*	*	*	1	*	Registre
SBC HL,BC	ED42	15	*	*	?	*	1	*	Registre
SBC HL,DE	ED52	15	*	*	?	*	1	*	Registre
SBC HL,HL	ED62	15	*	*	?	*	1	*	Registre
SBC HL,SP	ED72	15	*	*	?	*	1	*	Registre
SCF	37	4			0		0	1	Implicite
SET 0,(HL)	CBC6	15	?	*	1	?	0		Indirect
SET 0,(IX+op)	DDCBopC6	23	?	*	1	?	0		Indexé
SET 0,(IY+op)	FDCBopC6	23	?	*	1	?	0		Indexé
SET 0,A	CBC7	8	?	*	1	?	0		Registre
SET 0,B	CBC0	8	?	*	1	?	0		Registre
SET 0,C	CBC1	8	?	*	1	?	0		Registre
SET 0,D	CBC2	8	?	*	1	?	0		Registre
SET 0,E	CBC3	8	?	*	1	?	0		Registre
SET 0,H	CBC4	8	?	*	1	?	0		Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
SET 0,L	CBC5	8	?	*	1	?	0		Registre
SET 1,(HL)	CBCE	15	?	*	1	?	0		Indirect
SET 1,(IX+op)	DDCBopCE	23	?	*	1	?	0		Indexé
SET 1,(IY+op)	FDCBopCE	23	?	*	1	?	0		Indexé
SET 1,A	CBCF	8	?	*	1	?	0		Registre
SET 1,B	CBC8	8	?	*	1	?	0		Registre
SET 1,C	CBC9	8	?	*	1	?	0		Registre
SET 1,D	CBCA	8	?	*	1	?	0		Registre
SET 1,E	CBCB	8	?	*	1	?	0		Registre
SET 1,H	CBCC	8	?	*	1	?	0		Registre
SET 1,L	CBCD	8	?	*	1	?	0		Registre
SET 2,(HL)	CBD6	15	?	*	1	?	0		Indirect
SET 2,(IX+op)	DDCBopD6	23	?	*	1	?	0		Indexé
SET 2,(IY+op)	FDCBopD6	23	?	*	1	?	0		Indexé
SET 2,A	CBD7	8	?	*	1	?	0		Registre
SET 2,B	CBD0	8	?	*	1	?	0		Registre
SET 2,C	CBD1	8	?	*	1	?	0		Registre
SET 2,D	CBD2	8	?	*	1	?	0		Registre
SET 2,E	CBD3	8	?	*	1	?	0		Registre
SET 2,H	CBD4	8	?	*	1	?	0		Registre
SET 2,L	CBD5	8	?	*	1	?	0		Registre
SET 3,(HL)	CBDE	15	?	*	1	?	0		Indirect
SET 3,(IX+op)	DDCBopDE	23	?	*	1	?	0		Indexé
SET 3,(IY+op)	FDCBopDE	23	?	*	1	?	0		Indexé
SET 3,A	CBDF	8	?	*	1	?	0		Registre
SET 3,B	CBD8	8	?	*	1	?	0		Registre
SET 3,C	CBD9	8	?	*	1	?	0		Registre
SET 3,D	CBDA	8	?	*	1	?	0		Registre
SET 3,E	CBDB	8	?	*	1	?	0		Registre
SET 3,H	CBDC	8	?	*	1	?	0		Registre
SET 3,L	CBDD	8	?	*	1	?	0		Registre
SET 4,(HL)	CBE6	15	?	*	1	?	0		Indirect
SET 4,(IX+op)	DDCBopE6	23	?	*	1	?	0		Indexé
SET 4,(IY+op)	FDCBopE6	23	?	*	1	?	0		Indexé
SET 4,A	CBE7	8	?	*	1	?	0		Registre
SET 4,B	CBE0	8	?	*	1	?	0		Registre
SET 4,C	CBE1	8	?	*	1	?	0		Registre
SET 4,D	CBE2	8	?	*	1	?	0		Registre
SET 4,E	CBE3	8	?	*	1	?	0		Registre
SET 4,H	CBE4	8	?	*	1	?	0		Registre
SET 4,L	CBE5	8	?	*	1	?	0		Registre
SET 5,(HL)	CBEE	15	?	*	1	?	0		Indirect
SET 5,(IX+op)	DDCBopEE	23	?	*	1	?	0		Indexé
SET 5,(IY+op)	FDCBopEE	23	?	*	1	?	0		Indexé
SET 5,A	CBEF	8	?	*	1	?	0		Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
SET 5,B	CBE8	8	?	*	1	?	0		Registre
SET 5,C	CBE9	8	?	*	1	?	0		Registre
SET 5,D	CBEA	8	?	*	1	?	0		Registre
SET 5,E	CBEB	8	?	*	1	?	0		Registre
SET 5,H	CBEC	8	?	*	1	?	0		Registre
SET 5,L	CBED	8	?	*	1	?	0		Registre
SET 6,(HL)	CBF6	15	?	*	1	?	0		Indirect
SET 6,(IX+op)	DDCBopF6	23	?	*	1	?	0		Indexé
SET 6,(IY+op)	FDCBopF6	23	?	*	1	?	0		Indexé
SET 6,A	CBF7	8	?	*	1	?	0		Registre
SET 6,B	CBF0	8	?	*	1	?	0		Registre
SET 6,C	CBF1	8	?	*	1	?	0		Registre
SET 6,D	CBF2	8	?	*	1	?	0		Registre
SET 6,E	CBF3	8	?	*	1	?	0		Registre
SET 6,H	CBF4	8	?	*	1	?	0		Registre
SET 6,L	CBF5	8	?	*	1	?	0		Registre
SET 7,(HL)	CBFE	15	?	*	1	?	0		Indirect
SET 7,(IX+op)	DDCBopFE	23	?	*	1	?	0		Indexé
SET 7,(IY+op)	FDCBopFE	23	?	*	1	?	0		Indexé
SET 7,A	CBFF	8	?	*	1	?	0		Registre
SET 7,B	CBF8	8	?	*	1	?	0		Registre
SET 7,C	CBF9	8	?	*	1	?	0		Registre
SET 7,D	CBFA	8	?	*	1	?	0		Registre
SET 7,E	CBFB	8	?	*	1	?	0		Registre
SET 7,H	CBFC	8	?	*	1	?	0		Registre
SET 7,L	CBFD	8	?	*	1	?	0		Registre
SLA (HL)	CB26	15	*	*	0	*	0	*	Indirect
SLA (IX+op)	DDCBop26	23	*	*	0	*	0	*	Indexé
SLA (IY+op)	FDCBop26	23	*	*	0	*	0	*	Indexé
SLA A	CB27	8	*	*	0	*	0	*	Registre
SLA B	CB20	8	*	*	0	*	0	*	Registre
SLA C	CB21	8	*	*	0	*	0	*	Registre
SLA D	CB22	8	*	*	0	*	0	*	Registre
SLA E	CB23	8	*	*	0	*	0	*	Registre
SLA H	CB24	8	*	*	0	*	0	*	Registre
SLA L	CB25	8	*	*	0	*	0	*	Registre
SRA (HL)	CB2E	15	*	*	0	*	0	*	Indirect
SRA (IX+op)	DDCBop2E	23	*	*	0	*	0	*	Indexé
SRA (IY+op)	FDCBop2E	23	*	*	0	*	0	*	Indexé
SRA A	CB2F	8	*	*	0	*	0	*	Registre
SRA B	CB28	8	*	*	0	*	0	*	Registre
SRA C	CB29	8	*	*	0	*	0	*	Registre
SRA D	CB2A	8	*	*	0	*	0	*	Registre
SRA E	CB2B	8	*	*	0	*	0	*	Registre
SRA H	CB2C	8	*	*	0	*	0	*	Registre

Mnémonique	Code objet (hexa)	Cycles	Registre d'état						Adressage
			S	Z	H	P/V	N	C	
SRA L	CB2D	8	*	*	0	*	0	*	Registre
SRL (HL)	CB3E	15	*	*	0	*	0	*	Indirect
SRL (IX+op)	DDCBop3E	23	*	*	0	*	0	*	Indexé
SRL (IY+op)	FDCBop3E	23	*	*	0	*	0	*	Indexé
SRL A	CB3F	8	*	*	0	*	0	*	Registre
SRL B	CB38	8	*	*	0	*	0	*	Registre
SRL C	CB39	8	*	*	0	*	0	*	Registre
SRL D	CB3A	8	*	*	0	*	0	*	Registre
SRL E	CB3B	8	*	*	0	*	0	*	Registre
SRL H	CB3C	8	*	*	0	*	0	*	Registre
SRL L	CB3D	8	*	*	0	*	0	*	Registre
SUB (HL)	96	7	*	*	*	*	1	*	Indirect
SUB (IX+op)	DD96op	19	*	*	*	*	1	*	Indexé
SUB (IY+op)	FD96op	19	*	*	*	*	1	*	Indexé
SUB A	97	4	*	*	*	*	1	*	Registre
SUB B	90	4	*	*	*	*	1	*	Registre
SUB C	91	4	*	*	*	*	1	*	Registre
SUB D	92	4	*	*	*	*	1	*	Registre
SUB E	93	4	*	*	*	*	1	*	Registre
SUB H	94	4	*	*	*	*	1	*	Registre
SUB L	95	4	*	*	*	*	1	*	Registre
SUB op	D6op	7	*	*	*	*	1	*	Immédiat
XOR (HL)	AE	7	*	*	0	*	0	0	Indirect
XOR (IX+op)	DDAEop	19	*	*	0	*	0	0	Indexé
XOR (IY+Op)	FDAEop	19	*	*	0	*	0	0	Indexé
XOR A	AF	4	*	*	0	*	0	0	Registre
XOR B	A8	4	*	*	0	*	0	0	Registre
XOR C	A9	4	*	*	0	*	0	0	Registre
XOR D	AA	4	*	*	0	*	0	0	Registre
XOR E	AB	4	*	*	0	*	0	0	Registre
XOR H	AC	4	*	*	0	*	0	0	Registre
XOR L	AD	4	*	*	0	*	0	0	Registre
XOR op	EEop	7	*	*	0	*	0	0	Registre

Abréviations utilisées

d représente un déplacement relatif sur huit bits.
n représente une valeur sur un octet.
nn représente une valeur sur deux octets.

Tous les codes sont en hexadécimal.

Instructions non préfixées

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Ø	NOP	LD BC,nn	LD (BC),A	INC BC	INC B	DEC B	LD B,n	RLCA	EX AF,AF'	ADD HL,BC	LD A,(BC)	DEC BC	INC C	DEC C	LD C,n	RRCA
1	DJNZ d	LD DE,nn	LD (DE),A	INC DE	INC D	DEC D	LD D,n	RLA	JR d	ADD HL,DE	LD A,(DE)	DEC DE	INC E	DEC E	LD E,n	RRA
2	JR NZ,d	LD HL,nn	LD (nn),HL	INC HL	INC H	DEC H	LD H,n	DAA	JR Z,d	ADD HL,HL	LD HL,(nn)	DEC HL	INC L	DEC L	LD L,n	CPL
3	JR NC,d	LD SP,nn	LD (nn),A	INC SP	INC (HL)	DEC (HL)	LD (HL),n	SCF C,d	JR C,d	ADD HL,SP	LD A,(nn)	DEC SP	INC A	DEC A	LD A,n	CCF
4	LD B,B	LD B,C	LD B,D	LD B,E	LD B,H	LD B,L	LD B,(HL)	LD B,A	LD C,B	LD C,C	LD C,D	LD C,E	LD C,H	LD C,L	LD C,(HL)	LD C,A
5	LD D,B	LD D,C	LD D,D	LD D,E	LD D,H	LD D,L	LD D,(HL)	LD D,A	LD E,B	LD E,C	LD E,D	LD E,E	LD E,H	LD E,L	LD E,(HL)	LD E,A
6	LD H,B	LD H,C	LD H,D	LD H,E	LD H,H	LD H,L	LD H,(HL)	LD H,A	LD L,B	LD L,C	LD L,D	LD L,E	LD L,H	LD L,L	LD L,(HL)	LD L,A
7	LD (HL),B	LD (HL),C	LD (HL),D	LD (HL),E	LD (HL),H	LD (HL),L	HALT	LD (HL),A	LD A,B	LD A,C	LD A,D	LD A,E	LD A,H	LD A,L	LD A,(HL)	LD A,A
8	ADD A,B	ADD A,C	ADD A,D	ADD A,E	ADD A,H	ADD A,L	ADD A,(HL)	ADD A,A	ADC A,B	ADC A,C	ADC A,D	ADC A,E	ADC A,H	ADC A,L	ADC A,(HL)	ADC A,A
9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB (HL)	SUB A	SBC A,B	SBC A,C	SBC A,D	SBC A,E	SBC A,H	SBC A,L	SBC A,(HL)	SBC A,A
A	AND B	AND C	AND D	AND E	AND H	AND L	AND (HL)	AND A	XOR B	XOR C	XOR D	XOR E	XOR H	XOR L	XOR (HL)	XOR A
B	OR B	OR C	OR D	OR E	OR H	OR L	OR (HL)	OR A	CP B	CP C	CP D	CP E	CP H	CP L	CP (HL)	CP A
C	RET NZ	POP BC	JP NZ,nn	JP nn	CALL NZ,nn	PUSH BC	ADD A,n	RST Ø	RET Z	RET	JP Z,nn		CALL Z,nn	CALL nn	ADC A,n	RST 8
D	RET NC	POP DE	JP NC,nn	OUT (n),A	CALL NC,nn	PUSH DE	SUB n	RST 16	RET C	EXX	JP C,nn	IN A,(n)	CALL C,nn		SBC A,n	RST 24
E	RET PO	POP HL	JP PO,nn	EX (SP),HL	CALL PO,nn	PUSH HL	AND n	RST 32	RET DE	JP (HL)	JP PE,nn	EX DE,HL	CALL PE,nn		XOR n	RST 40
F	RET P	POP AF	JP P,nn	DI	CALL P,nn	PUSH AF	OR n	RST 48	RET M	LD SP,HL	JP M,nn	EI	CALL M,nn		CP n	RST 56

Instructions préfixées par CB (hexa) (non indexées)

Chaque code doit être précédé de CB.

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Ø	RLC B	RLC C	RLC D	RLC E	RLC H	RLC L	RLC (HL)	RLC A	RRC B	RRC C	RRC D	RRC E	RRC H	RRC L	RRC (HL)	RRC A
1	RL B	RL C	RL D	RL E	RL H	RL L	RL (HL)	RL A	RR B	RR C	RR D	RR E	RR H	RR L	RR (HL)	RR A
2	SLA B	SLA C	SLA D	SLA E	SLA H	SLA L	SLA (HL)	SLA A	SRA B	SRA C	SRA D	SRA E	SRA H	SRA L	SRA (HL)	SRA A
3									SRL B	SRL C	SRL D	SRL E	SRL H	SRL L	SRL (HL)	SRL A
4	BIT Ø,B	BIT Ø,C	BIT Ø,D	BIT Ø,E	BIT Ø,H	BIT Ø,L	BIT Ø,(HL)	BIT Ø,A	BIT 1,B	BIT 1,C	BIT 1,D	BIT 1,E	BIT 1,H	BIT 1,L	BIT 1,(HL)	BIT 1,A
5	BIT 2,B	BIT 2,C	BIT 2,D	BIT 2,E	BIT 2,H	BIT 2,L	BIT 2,(HL)	BIT 2,A	BIT 3,B	BIT 3,C	BIT 3,D	BIT 3,E	BIT 3,H	BIT 3,L	BIT 3,(HL)	BIT 3,A
6	BIT 4,B	BIT 4,C	BIT 4,D	BIT 4,E	BIT 4,H	BIT 4,L	BIT 4,(HL)	BIT 4,A	BIT 5,B	BIT 5,C	BIT 5,D	BIT 5,E	BIT 5,H	BIT 5,L	BIT 5,(HL)	BIT 5,A
7	BIT 6,B	BIT 6,C	BIT 6,D	BIT 6,E	BIT 6,H	BIT 6,L	BIT 6,(HL)	BIT 6,A	BIT 7,B	BIT 7,C	BIT 7,D	BIT 7,E	BIT 7,H	BIT 7,L	BIT 7,(HL)	BIT 7,A
8	RES Ø,B	RES Ø,C	RES Ø,D	RES Ø,E	RES Ø,H	RES Ø,L	RES Ø,(HL)	RES Ø,A	RES 1,B	RES 1,C	RES 1,D	RES 1,E	RES 1,H	RES 1,L	RES 1,(HL)	RES 1,A
9	RES 2,B	RES 2,C	RES 2,D	RES 2,E	RES 2,H	RES 2,L	RES 2,(HL)	RES 2,A	RES 3,B	RES 3,C	RES 3,D	RES 3,E	RES 3,H	RES 3,L	RES 3,(HL)	RES 3,A
A	RES 4,B	RES 4,C	RES 4,D	RES 4,E	RES 4,H	RES 4,L	RES 4,(HL)	RES 4,A	RES 5,B	RES 5,C	RES 5,D	RES 5,E	RES 5,H	RES 5,L	RES 5,(HL)	RES 5,A
B	RES 6,B	RES 6,C	RES 6,D	RES 6,E	RES 6,H	RES 6,L	RES 6,(HL)	RES 6,A	RES 7,B	RES 7,C	RES 7,D	RES 7,E	RES 7,H	RES 7,L	RES 7,(HL)	RES 7,A
C	SET Ø,B	SET Ø,C	SET Ø,D	SET Ø,E	SET Ø,H	SET Ø,L	SET Ø,(HL)	SET Ø,A	SET 1,B	SET 1,C	SET 1,D	SET 1,E	SET 1,H	SET 1,L	SET 1,(HL)	SET 1,A
D	SET 2,B	SET 2,C	SET 2,D	SET 2,E	SET 2,H	SET 2,L	SET 2,(HL)	SET 2,A	SET 3,B	SET 3,C	SET 3,D	SET 3,E	SET 3,H	SET 3,L	SET 3,(HL)	SET 3,A
E	SET 4,B	SET 4,C	SET 4,D	SET 4,E	SET 4,H	SET 4,L	SET 4,(HL)	SET 4,A	SET 5,B	SET 5,C	SET 5,D	SET 5,E	SET 5,H	SET 5,L	SET 5,(HL)	SET 5,A
F	SET 6,B	SET 6,C	SET 6,D	SET 6,E	SET 6,H	SET 6,L	SET 6,(HL)	SET 6,A	SET 7,B	SET 7,C	SET 7,D	SET 7,E	SET 7,H	SET 7,L	SET 7,(HL)	SET 7,A

Instructions préfixées par ED (hexa)

Chaque code doit être précédé de ED.

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Ø																
1																
2																
3																
4	IN B,(C)	OUT (C),B	SBC HL,BC	ID (nn),BC	NEG	RETN	IM Ø	LD I,A	IN C,(C)	OUT (C),C	ADC HL,BC	LD BC,(nn)		RETI		LD R,A
5	IN D,(C)	OUT (C),D	SBC HL,DE	LD (nn),DE			IM 1	LD A,I	IN E,(C)	OUT (C),E	ADC HL,DE	LD DE,(nn)			IM 2	LD A,R
6	IN H,(C)	OUT (C),H	SBC HL,HL	LD (nn),HL				RRD	IN L,(C)	OUT (C),L	ADC HL,HL	LD HL,(nn)				RLD
7	IN F,(C)		SBC HL,SP	LD (nn),SP					IN A,(C)	OUT (C),A	ADC HL,SP	LD SP,(nn)				
8																
9																
A	LDI	CPI	INI	OUTI					LDD	CPD	IND	OUTD				
B	LDIR	CPIR	INIR	OTIR					LDDR	CPDR	INDR	OTDR				
C																
D																
E																
F																

Instructions indexées non préfixées

Chaque instruction doit être précédée de DD s'il s'agit du registre IX, et FD s'il s'agit du registre IY.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0										ADD IX,BC						
1										ADD IX,DE						
2		LD IX,nn	LD (nn),IX	INC IX						ADD IX,IX	LD IX,(nn)	DEC IX				
3					INC (IX+d)	DEC (IX+d)	LD (IX+d),nn			ADD IX,SP						
4							LD B,(IX+d)								LD C,(IX+d)	
5							LD D,(IX+d)								LD E,(IX+d)	
6							LD H,(IX+d)								LD L,(IX+d)	
7	LD (IX+d),B	LD (IX+d),C	LD (IX+d),D	LD (IX+d),E	LD (IX+d),H	LD (IX+d),L		LD (IX+d),A							LD A,(IX+d)	
8							ADD A,(IX+d)								ADC A,(IX+d)	
9							SUB (IX+d)								SBC A,(IX+d)	
A							AND (IX+d)								XOR (IX+d)	
B							OR (IX+d)								CP (IX+d)	
C																
D																
E		POP IX	EX (SP),IX			PUSH IX				JP (IX)						
F										LD SP,IX						

Instructions indexées préfixées par CB

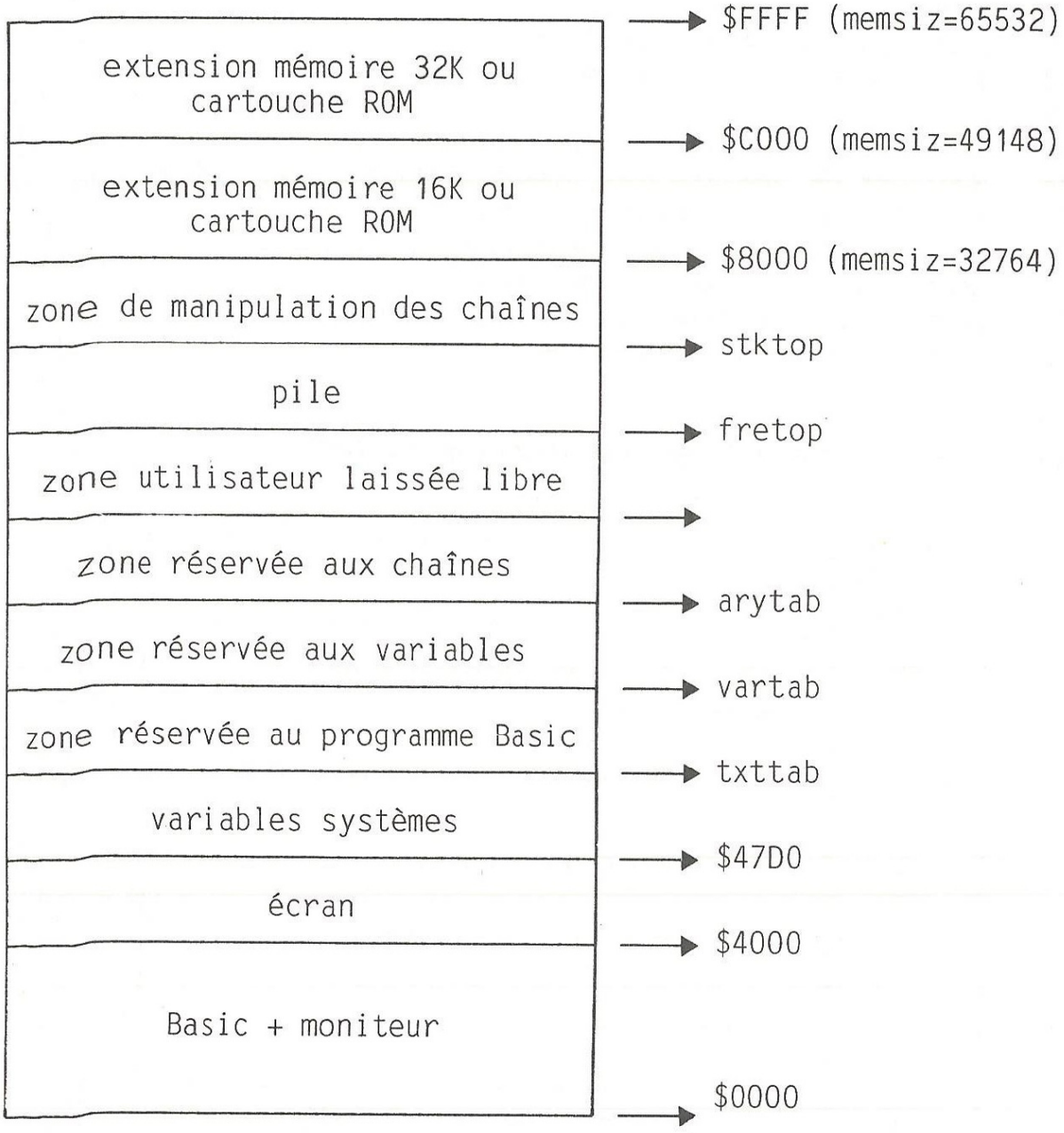
Préfixe d'indexation : IX → DD, IY → FD

Le format de codage est le suivant : (préfixe d'indexation) - CB - (déplacement) - (code obtenu dans le tableau).

Exemple : RES 2, (IX+4) → DD CB 04 96

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Ø							RLC (IX+d)								RRC (IX+d)	
1							RL (IX+d)								RR (IX+d)	
2							SLA (IX+d)								SRA (IX+d)	
3															SRL (IX+d)	
4							BITØ (IX+d)								BIT1 (IX+d)	
5							BIT2 (IX+d)								BIT3 (IX+d)	
6							BIT4 (IX+d)								BIT5 (IX+d)	
7							BIT6 (IX+d)								BIT7 (IX+d)	
8							RESØ (IX+d)								RES1 (IX+d)	
9							RES2 (IX+d)								RES3 (IX+d)	
A							RES4 (IX+d)								RES5 (IX+d)	
B							RES6 (IX+d)								RES7 (IX+d)	
C							SETØ (IX+d)								SET1 (IX+d)	
D							SET2 (IX+d)								SET3 (IX+d)	
E							SET4 (IX+d)								SET5 (IX+d)	
F							SET6 (IX+d)								SET7 (IX+d)	

CARTE MEMOIRE DU VG 5000



Organisation de la ROM

Elle fait 16 Ko et prend les adresses $0000-$3FFF. Elle est divisée en 14 modules distincts :

$0000 Routines d'initialisation du VG 5000 et de traitement des interruptions.
$02FF Routines de calcul en virgule flottante (Basic).
$0A75 Routines graphiques (Basic).
$1000 Routines d'initialisation du Basic.
$11F9 Zone des messages "texte".
$1945 Routines "cassette" du Basic.
$2000 Interpréteur Basic (première partie).
$2AD4 Editeur.
$2EAD Interpréteur Basic (seconde partie).
$3278 Routines de gestion des communications avec l'écran.
$3609 Routines de traitement des chaînes.
$38D0 Routines de saisie des pointeurs du Basic.
$3A59 Routines "cassette" du moniteur.
$3BD0 Contrôleur d'entrées-sorties du Basic.
$3CD2 Contrôleur de l'imprimante.

Organisation des ports d'entrées-sorties du Z80

Adresse (hexa)	*Fonction*
00 : : 7F	Réservé au périphérique en extension.
80 : : 88	Scrutation du clavier.
89 : : 8E	Inutilisés.
8F	Validation adresse : EF9345.
90 : : AE	Inutilisés.

→

Adresse (hexa)	*Fonction*
AF	Magnétophone à cassette et générateur musical.
B0 : CE	Inutilisés.
CF	Validation donnée : EF9345.
D0 : FF	Inutilisés.

Organisation des lignes Basic

Les lignes Basic sont "tokenisées", c'est-à-dire que chaque instruction n'apparaît pas sous forme ASCII (suite de lettres), mais sous forme d'un code qui lui est propre.

Le programme Basic commence à l'adresse pointée par txttab ($488E-488F), l'octet (txttab)-1 devant être à 0.

Le codage d'une ligne est ainsi conçu :

2 octets	2 octets	n octets	1 octet
adresse de la ligne suivante	numéro de la ligne	suite d'instructions	fin de ligne = 0

Chaque ligne possède l'adresse de début de la suivante, ce qui permet au Basic de se repérer.

La fin du programme est signalée par deux octets à 0 en plus de l'octet de fin de la dernière ligne.

L'adresse de fin du programme correspond à celle de début des variables, c'est-à-dire vartab.

Txttab pointe toujours le premier lien de chaîne du programme.

Les lignes contenant un appel à un numéro de ligne (GOTO, GOSUB,...) sont repérées par un code $0E, de manière à faciliter la renumérotation.

Organisation du processeur vidéo EF9345

Le Z80 et le EF9345 communiquent par l'intermédiaire d'un bus 8 bits multiplexé et par les ports d'entrées-sorties $8F et $CF.

Le processeur vidéo possède ses propres registres internes lui permettant de gérer de façon autonome tous les paramètres d'affichage.

L'utilisateur peut communiquer avec le EF9345 par l'intermédiaire des routines du Basic, par l'intermédiaire des routines d'accès direct (sans passer par le buffer vidéo ; voir "Routines utiles") ou bien même directement en utilisant tous les modes disponibles sur ce processeur (le VG 5000 utilise uniquement le mode 16 bits par caractère).

La description de ce circuit dépassant le cadre de ce mémento, pour plus de renseignements, veuillez consulter la documentation technique du EF9345 (Thomson-Efcis).

RST 00 Réinitialisation totale du VG 5000 (équivalant à la mise sous tension). La mémoire et les entrées-sorties sont mises à jour.

RST 08 Vérifie que le caractère pointé par HL (pointeur de texte) est le même que celui qui suit l'instruction RST 08. Dans l'affirmative, la routine rend la main à l'utilisateur, sinon, elle génère une erreur du type "syntaxe incorrecte". Les registres AF et HL sont altérés.

RST 10 Routine de lecture d'un caractère (chget). Un appel à cette routine renvoie dans A le caractère pointé par HL+1. S'il s'agit d'un caractère numérique, le drapeau C est mis à 1. S'il s'agit d'un caractère de fin de ligne, le drapeau Z est mis à 1.

RST 18 Routine d'envoi d'un caractère à l'organe d'entrées-sorties sélectionné (écran, imprimante ou lecteur de cassette, selon la valeur de l'octet $486F. Voir "Variables systèmes"). Le caractère doit être mis dans A, aucun registre n'est altéré.

RST 20 Routine de comparaison des registres HL et DE. S'ils sont égaux, le drapeau Z est mis à 1, si DE est supérieur à HL, alors le drapeau C est mis à 1.

RST 28 Routine du calculateur en virgule flottante. Un saut est fait en $059A pour continuer le traitement, si l'octet $49E9 est non nul.

RST 30 Routine laissée libre pour l'utilisateur. Un saut est effectué en $47DC, vecteur facilement modifiable (voir "Les comment ?").

RST 38 Routine de traitement des interruptions masquables.

LISTE DES COMMANDES BASIC

Instruction	*Token Dec*	*Adresse Hex*	*Instruction*	*Token Dec*	*Adresse Hex*
END	128	2F3A	**RENUM**	153	3058
FOR	129	247A	**AUTO**	154	316D
NEXT	130	2FD4	**LOAD**	155	197D
DATA	131	25F3	**SAVE**	156	1945
INPUT	132	2760	**CLOAD**	157	1A39
DIM	133	38E5	**CSAVE**	158	19DE
READ	134	27A3	**CALL**	159	0FAE
LET	135	2610	**INIT**	160	0D51
GOTO	136	25BA	**SOUND**	161	0C6A
RUN	137	259A	**PLAY**	162	0A75
IF	138	2679	**TX**	163	0EF8
RESTORE	139	2F1D	**GR**	164	0D37
GOSUB	140	25A9	**SCREEN**	165	0F8D
RETURN	141	25E6	**DISPLAY**	166	0F98
REM	142	25F5	**STORE**	167	0FA4
STOP	143	2F38	**SCROLL**	168	0DA8
ON	144	265D	**PAGE**	169	0DB1
LPRINT	145	2694	**DELIM**	170	0F52
DEF	146	2A13	**SETE**	171	0CDE
POKE	147	2ABB	**ET**	172	0D40
PRINT	148	269B	**EG**	173	0D49
CONT	149	2F6F	**CURSOR**	174	0D83
LIST	150	31A8	**DISK**	175	47F4
LLIST	151	31A3	**MODEM**	176	47F7
CLEAR	152	2FA0	**NEW**	177	3ECD

Instruction	*Token Dec*	*Adresse Hex*
SGN	195	05A4
INT	196	0660
ABS	197	05B8
USR	198	4833
FRE	199	38B1
LPOS	200	29FC
POS	201	2A01
SQR	202	081E
RND	203	090E
LOG	204	0434
EXP	205	0874
COS	206	097D
SIN	207	0984
TAN	208	0A17
ATN	209	0A2C
PEEK	210	2AB4
LEN	211	37FB
STR$	212	3636
VAL	213	388E
ASC	214	380A
STICKX	215	0DBA
STICKY	216	0DF0
ACTION	217	0F25
KEY	218	0F49
LPEN	219	47F1
CHR$	220	381B
LEFT$	221	3829
RIGHT$	222	3858
MID$	223	3861

$001B Permet de définir un caractère spécial aussi bien texte que graphique. Le code du caractère doit être mis dans les bits 0-6 de A. Le bit 7 de A sert de drapeau texte/graphique : bit 7 à 0 → mode texte, bit 7 à 1 → mode graphique.
Les dix octets de codage (identique à SETEG ou SETET) doivent être stockés dans un buffer, HL doit en contenir l'adresse. Tous les registres sont altérés.

£0033 Départ à chaud. L'appel à cette routine correspond à une pression sur les touches CTRL et Delta. Efface l'écran, réinitialise les couleurs et les entrées-sorties et joue la "mélodie" d'introduction.
Tous les registres sont altérés.

$0080 Sous-programme de retour qui doit terminer une routine USR. Ce sous-programme permet de renvoyer le contenu de l'accumulateur au Basic. HL doit être remis à jour pour l'appel à ce sous-programme (pointeur de texte).
Tous les registres sont altérés.

$0083 Permet d'acquérir un paramètre sur deux octets, le premier étant pointé par HL (pointeur de texte). Le paramètre revient dans DE.
Tous les registres sont altérés.

$0086 Permet d'acquérir un paramètre sur un octet pointé par HL (pointeur de texte). Le paramètre revient dans A.
Tous les registres sont altérés.

$0089 Permet d'acquérir un paramètre signé sur deux octets, le premier étant pointé par HL. Le résultat est stocké dans l'accumulateur virgule flottante. Si l'on veut le résultat dans DE, on peut faire appel en $0083 après avoir exécuté cette routine.
Tous les registres sont altérés.

$008C Permet d'émettre un son (équivalant au SOUND BASIC). DE doit contenir la durée du son, les fréquences extrêmes (fréquence de base + rapport cyclique, fréquence de base - rapport cyclique) doivent être stockées en $480E-480F (voir "Variables systèmes").
Tous les registres sont altérés.

$008F Permet de jouer une suite de notes de musique (équivalent du PLAY BASIC). La syntaxe est la même que celle utilisée en Basic, tous les codes doivent être mis sous forme ASCII et stockés dans un buffer. BC doit contenir l'adresse de ce buffer et E sa longueur.
Ex : le codage de l'équivalent de PLAY"T20CDE" est :
$54,$32,$30,$43,$44,$45
Tous les registres sont altérés.

$0092 Envoie un caractère de 16 bits à l'écran directement sans passer par le buffer vidéo géré par le Z80. La position du caractère est donnée par HL, H représentant l'ordonnée (ligne) 0 → ligne 0, 8-31 → lignes 1-24, et L représentant l'abscisse (colonne) 0-39 → colonnes 0-39. D doit contenir le code du caractère et E son attribut. Les registres BC et AF sont altérés.

$0095 Envoie un caractère de 16 bits à l'écran directement sans passer par le buffer vidéo géré par le Z80. L'affichage se fait à la position courante du curseur. D doit contenir le code du caractère, et E son attribut. Les registres BC et AF sont altérés.

$0098 Permet de lire un caractère de 16 bits directement depuis la mémoire écran à la position définie par HL (codage identique à $0092). Le code du caractère revient dans D, tandis que son attribut revient dans E. Les registres AF, BC et DE sont altérés.

$009B Permet de lire un caractère de 16 bits directement depuis la mémoire écran à la position courante du curseur. Le code du caractère revient dans D, tandis que son attribut revient dans E. Les registres AF, BC et DE sont altérés.

$009E Efface l'écran et réinitialise la couleur du fond définie par $4803, et de l'encre définie par $4802. Tous les registres sont altérés.

$00A1 Efface une ligne et réinitialise les couleurs de fond et d'encre (idem $009E). A doit contenir le numéro de la ligne à effacer. Tous les registres sont altérés.

$00A4 Interrompt le déroulement d'un programme jusqu'à ce que le circuit vidéo soit disponible pour un transfert de données. Le registre AF est altéré.

$00A7 Calcule l'adresse physique d'une position d'écran par rapport à son abscisse et son ordonnée. H doit contenir l'ordonnée et L l'abscisse. L'adresse calculée revient dans HL. Tous les registres sont altérés.

$00AA Scrute le clavier et renvoie dans A le code de la touche pressée. Tous les registres sont altérés.

$00AD Charge les registres du processeur vidéo EF9345 dans une table pointée par HL. Le premier octet de la table sera le nombre de données dans la table. Les registres seront codés sur deux octets, le premier octet étant l'adresse

de sélection du registre, et le second le contenu du registre.
Les registres AF, BC et HL sont altérés.

$00B0 Scrute le déplacement haut/bas des poignées de jeu (ou des touches du clavier). A doit contenir le code du périphérique :

A=0 : poignée de jeu gauche.
A=1 : poignée de jeu droite.
A=2 : flèches du clavier.

En retour, A contient :

0 : si pas de déplacement.
1 : si déplacement vers le bas.
255 : si déplacement vers le haut.

Les registres AF et B sont altérés.

$00B3 Scrute le déplacement droite/gauche des poignées de jeu (ou des touches du clavier). A doit contenir le code du périphérique :

A=0 : poignée de jeu gauche.
A=1 : poignée de jeu droite.
A=2 : flèches du clavier.

En retour, A contient :

0 : si pas de déplacement.
1 : si déplacement vers la droite.
255 : si déplacement vers la gauche.

$00B6 Scrute la pression sur les touches ACTION des poignées de jeu ou sur la barre d'espace du clavier. A doit contenir le code du périphérique :

A=0 : poignée de jeu gauche.
A=1 : poignée de jeu droite.
A=2 : flèches du clavier.

En retour, A contient :

0 : si pas de pression.
1 : si le bouton ACTION 1 est pressé (ou la barre d'espace).
2 : si le bouton ACTION 2 est pressé.
3 : si les deux boutons sont pressés.

Les registres AF et B sont altérés.

$00B9 Scrute les touches shift/stop du clavier.
En retour, A contient :

0 : si les touches SHIFT et STOP sont pressées. De plus, le drapeau C est mis à 1.
1 : si la touche STOP n'est pas pressée.
4 : si seule la touche STOP est pressée.

$030D Soustraction en virgule flottante.

$0310 Addition en virgule flottante.

$047A Multiplication en virgule flottante.

$04DE Division en virgule flottante.

$0A75 Joue une suite de notes de musique entre guillemets. Le format est le même que le format Basic. HL doit pointer sur le premier guillemet. S'il y a une erreur de syntaxe, le contrôle est rendu au Basic.
Tous les registres sont altérés.

$101C Réinitialise le VG 5000 totalement mais n'effectue pas de test de mémoire. HL doit contenir l'adresse du plus haut octet de la mémoire disponible.
Tous les registres sont altérés.

$10AB Effectue la séquence d'initialisation à partir de l'affichage du message "VG 5000 BASIC"...
Tous les registres sont altérés.

$116D Joue la "mélodie" d'initialisation.
Tous les registres sont altérés.

$1C23 Renvoie l'adresse du dernier lien de chaîne dans HL, soit l'adresse de début de la zone variable.
Les registres A, DE et HL sont altérés.

$1D97 Envoie d'une chaîne de caractères au lecteur de cassette.

$1E89 Acquisition d'une chaîne de caractères à partir du lecteur de cassette.

$1F9C Vérifie qu'il y a au moins 80 positions entre la valeur actuelle du pointeur de pile et HL.
Le drapeau C est mis à 0 si la condition est vérifiée, à 1 sinon.
Le registre A est altéré.

$1FAF Routine d'envoi d'un octet au lecteur de cassette.

$1FB9 Lecture d'un octet à partir du lecteur de cassette.

$2214 Affiche le message "Ok!".
Tous les registres sont altérés.

$228D Ferme le fichier imprimante, affiche "Ok!" et cherche l'éventuelle prochaine commande dans l'entrée-ligne.
Tous les registres sont altérés.

$2299 Gère l'entrée-ligne, envoie à la routine de "tokenisation" et fait exécuter les commandes en mode direct.
Tous les registres sont altérés.

$2328 Effectue un CLEAR, réinitialise la pile, rétablit éventuellement les liens de chaîne entre les lignes et branche sur l'interpréteur.

$24EA Scrute un programme, repère un "token", cherche l'adresse d'exécution de la routine correspondante et l'exécute. L'adresse $24EA est laissée sur la pile lorsqu'une instruction est exécutée, un RET rend donc la main à cette routine d'interprétation. HL doit pointer l'instruction précédente (ou être nul).
Tous les registres sont affectés.

$2526 Permet de lire un paramètre sur deux octets, le premier étant pointé par HL. Ce paramètre peut être une variable numérique. La valeur lue est retournée dans DE.
Tous les registres sont affectés.

$252D Permet le passage du paramètre à l'intérieur des parenthèses d'une fonction USR, soit I dans le cas de USR(I). Cette routine évalue ce paramètre qui peut être sous forme de valeur, de variable, ou même de formule numérique, et le stocke dans DE.
Ex : USR (12), USR (A) ou USR (A*2 + 3*B) ...
Tous les registres sont affectés.

$259A Commande RUN BASIC. L'appel à cette routine doit se faire par un JP. A doit être mis à 0, le drapeau Z à 1 et la valeur $24EA doit être mise au sommet de la pile.
Tous les registres sont altérés.

$2EEB Réinitialise un programme Basic en faisant un CLEAR, un RESTORE et en remettant la pile à jour.
Tous les registres sont altérés, sauf HL.

$32B7 Scrute le clavier sans attendre de pression sur une touche. Le code de la touche appuyée est dans A et en $47FF. Si A=0, aucune touche n'a été appuyée.
Si le drapeau C est à 1, il s'agit d'une touche répétée, sinon, il s'agit d'une nouvelle touche.
Le registre A est altéré.

$36AA Envoie une chaîne de caractères sur le périphérique courant (écran, imprimante ou magnétophone à cassette suivant la valeur de l'octet $486F (voir "Variables systèmes"). La chaîne doit se terminer par un zéro. HL doit pointer le premier octet de la chaîne.
Tous les registres sont altérés.

$3C40 Envoie un caractère d'autorisation de visualisation (15) et effectue un retour-chariot si le curseur n'est pas en première colonne.
Le registre A est altéré.

$3C43 Effectue un retour-chariot si le curseur n'est pas en première colonne.
Aucun registre n'est altéré.

$3CA2 Appel à l'éditeur Basic. Lorsque la touche RET est pressée, la ligne sur laquelle pointe le curseur est stockée dans un buffer situé à partir de $4898. La fin de ligne est repérée par un zéro. HL est pointé sur le début du buffer-1.
Le drapeau C est mis à 1 si la touche SHIFT/STOP a été pressée, à 0 sinon.
Les registres A, DE et HL sont altérés.

$3CD2 Envoie à l'imprimante le caractère dont le code est contenu dans A. L'impression ne se fait qu'après l'envoi d'un retour-chariot (code $0D).

Adresse (hexa)	*Longueur*	*Description*
4000	$7D0	Ecran (25 lignes de 80 octets).

Vecteurs

47D3	3	Vecteur des interruptions masquables.
47D6	3	Vecteur des interruptions non masquables.
47D9	3	Vecteur de la routine CALL.
47DC	3	Vecteur de la routine utilisateur RST.
47DF	3	Vecteur de la routine PLAY.
47E2	3	Vecteur de la commande PRINT.
47E5	3	Vecteur de la routine de sortie de caractères.
47E8	3	Vecteur de la routine retour-chariot et saut de ligne.
47EB	3	Vecteur de la routine d'acquisition de caractères.
47EE	3	Vecteur de la commande INPUT.
47F1	3	Vecteur du crayon optique.
47F4	3	Vecteur de l'interface disque.
47F7	3	Vecteur de l'interface modem.

Variables concernant l'écran et le clavier

47FA	1	Compteur d'interruptions pour le rafraîchissement de l'écran.
47FB	1	Drapeau indiquant si l'écran a besoin d'être rafraîchi.
47FC	1	Valeur de référence pour le compteur d'interruptions.
47FD	1	Couleur du pourtour et aspect du curseur.
47FE	1	Bascule majuscule/minuscule.
47FF	1	Valeur de la dernière scrutation clavier.
4800	1	Compteur pour la répétition automatique des touches du clavier.

Adresse (hexa)	*Longueur*	*Description*
4801	1	Drapeau de répétition automatique des touches du clavier.
4802	1	Attribut de couleur d'encre pour le prochain caractère à afficher : bits 0-2 : couleur d'encre ; bit 3 : clignotement ; bit 4 : double hauteur (texte) ; bit 5 : double largeur (texte) ; bits 4-6 : couleur de fond (graphique) ; bit 7 : 0 = texte, 1 = graphique.
4803	1	Attribut de couleur utilisé par init : bits 0-2 : encre de la colonne 0 ; bit 3 : = 0 ; bits 4-6 : couleur de fond ; bit 7 : = 1.
4804	1	Drapeau de redéfinition de caractères : bits 0-6 : = 0 ; bit 7 : 0 = normal, 1 = redéfini.
4805	1	Abscisse du curseur (colonne).
4806	1	Ordonnée du curseur (ligne).

Editeur de texte

4807	2	Précédent numéro de ligne.
4809	2	Numéro de la première ligne listée par la touche LIST.
480B	2	Adresse de la prochaine ligne à interpréter.
480D	1	Drapeau indiquant le mode programme (=82) ou le mode direct (=80).

Générateur de sons

480E	2	Fréquences extrêmes d'un son : pour SOUND A, B, C. 1er octet = 256-A+C ; 2ème octet = 256-A-C.

Adresse (hexa)	*Longueur*	*Description*
Cassette		
4810	30	Zone de stockage cassette.
482E	1	Paramètre de calibrage.
482F	1	Paramètre de calibrage.
Divers		
4830	3	Vecteur pour le départ à chaud.
4833	3	Vecteur pour la fonction USR.
4836	15	Zone de travail en virgule flottante.
4845	41	Table du générateur de nombres pseudo-aléatoires.
486E	1	Abscisse courante de l'imprimante.
486F	1	Sélection du périphérique pour la sortie de caractères : 00 : écran ; 01 : imprimante ; 255 : cassette.
4870	1	Sélection du périphérique pour l'acquisition de caractères : 00 : clavier ; 255 : cassette.
4871	1	Contrôle de l'affichage : bit 0 bit 1 : 1 = ligne entrée sous Basic ; bit 2 : 1 = CONT autorisé ; bit 3 bit 4 bit 5 : 1 = INPUT actif ; bit 6 : curseur en fin de page ; bit 7 : scrolling supprimé.
4872	1	Contrôle de l'interface cassette : bit 0 : suppression des messages ; bit 1 : suppression de L/H/H ; bit 2 : 0 = 1200 bd (bits/s) ; 1 = 2400 bd (bits/s) ; bit 3 : retour si erreur ; bit 4 : fichier trop long ;

Adresse (hexa)	*Longueur*	*Description*
		bit 5 : erreur de vérification ; bit 6 : erreur de cheksum ; bit 7 : opération interrompue. Seuls, les bits 0-3 sont utilisables par le programmeur.

Imprimante

4873	1	Type d'imprimante : 00 : imprimante MSX (jeu de caractères VG 5000) ; 01 : autre imprimante (jeu de caractères MSX) ; 255 : pas de contrôle.
4874	1	Contrôle de l'imprimante : bit 0 : occupée ; bit 6 : initialisée ; bit 7 : inutilisée.
4875	1	Drapeau de communication avec l'imprimante.
4876	2	Adresse du vecteur d'initialisation de l'imprimante.
4878	2	Adresse de la table de conversion de l'imprimante.

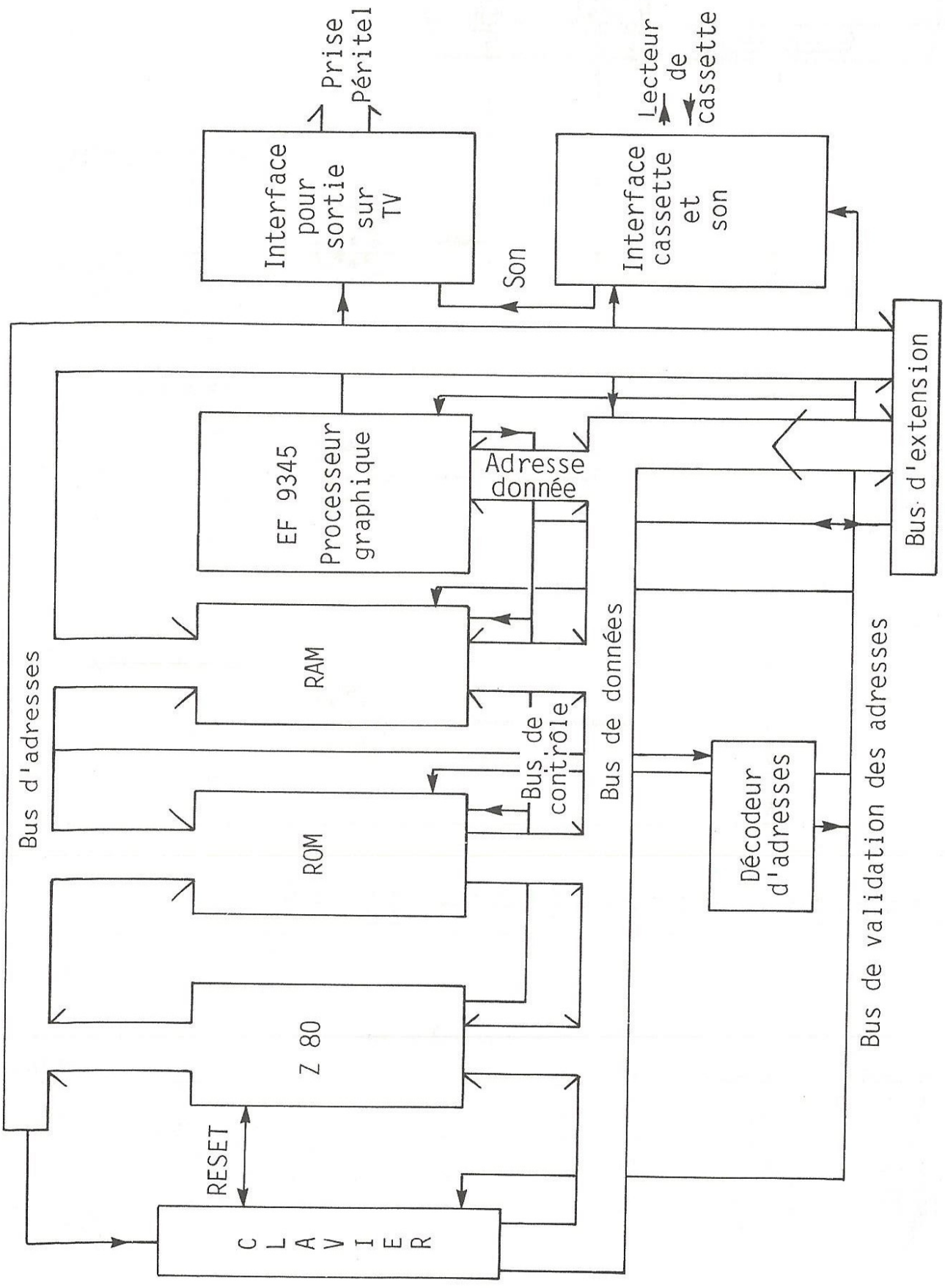
Editeur de texte

487A	10	Contrôle numérique des codes des touches.
4884	1	Drapeau indiquant la sélection du mode AUTO.
4885	2	Numéro de la ligne courante en mode AUTO.
4887	2	Incrément du mode AUTO.
4889	1	0 = exécution de la boucle de contrôle Basic.
488A	1	Longueur de la ligne.
488B	1	Dernière tabulation réalisée par une virgule.

Adresse (hexa)	*Longueur*	*Description*
Divers		
488C	2	Numéro de la ligne courante Basic.
488E	2	Adresse de début du programme Basic (txtab).
4890	1	Messages d'erreurs : 00 : en français ; 01 : en anglais.
4891	2	Adresse de la table de conversion du clavier.
4893	1	Tempo de la dernière instruction PLAY.
4894	1	Octave de la dernière instruction PLAY.
4895	2	Adresse du haut de la pile (sktop).
Interpréteur		
4897	1	Utilisé par l'instruction INPUT.
4898	129	Zone utilisée pour la "tokenisation" des lignes de programmes et des instructions en mode direct.
4919	1	Fin de la zone précédente.
491A	1	Indique l'activation de l'instruction DIM.
491B	1	Type de variable : 00 : numérique ; 01 : alphanumérique.
491C	1	Drapeau de suppression de la "tokenisation".
491D	2	Copie du pointeur de texte.
491F	2	Plus haute adresse RAM disponible (memsiz).
4921	2	Pointeur du descripteur de la chaîne effacée.
4923	120	Zone de stockage pour l'interprétation de variables.
499B	40	Zone de travail des fonctions de manipulation de chaînes.

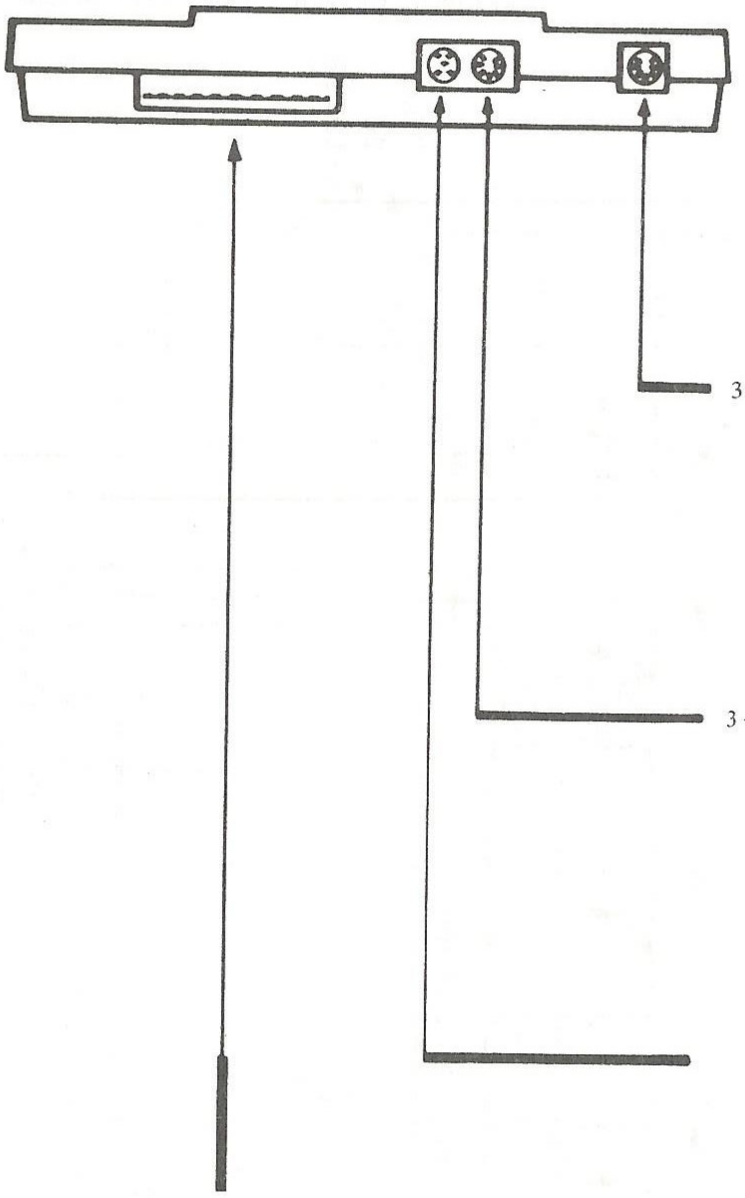
Adresse (hexa)	*Longueur*	*Description*
49C3	2	Adresse du haut de la zone "chaînes" (fretop).
49C5	2	Adresse de la fin de la chaîne effacée.
49C7	2	Utilisé pour la réorganisation des variables (garbage collection).
49C9	2	Copie du pointeur de texte pour une instruction FOR.
49CB	2	Numéro de la ligne de données.
49CD	1	Utilisé pour une instruction FOR.
49CE	1	Drapeau indiquant l'exécution d'un INPUT ou d'un READ.
49CF	2	Stockage temporaire du code d'une instruction.
49D1	1	Utilisé par l'instruction RENUM.
49D2	2	Evaluation d'une formule.
49D4	2	Ancien numéro de ligne.
49D6	2	Ancien pointeur de texte.
49D8	2	Adresse de début de la zone "variables" (vartab).
49DA	2	Adresse de début de la zone "chaînes" (arytab).
49DC	2	Adresse de fin du stockage en cours.
49DE	2	Pointeur de données dans les lignes DATA.
49E0	2	Nom du paramètre activé.
49E2	4	Valeur du paramètre activé.
49E6	3	Octets de poids faible de l'accumulateur virgule flottante.
49E9	2	Accumulateur virgule flottante.
49EB	13	Buffer virgule flottante.
49F8	1	Variable temporaire virgule flottante.
49F9	2	Variable temporaire virgule flottante.

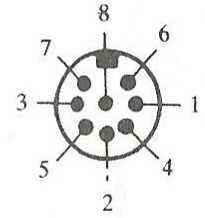
SYNOPTIQUE DU VG 5000



BROCHAGES DU CONNECTEUR ET DES PRISES

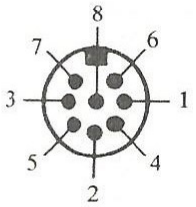
Schéma des prises
(vue AR du VG 5000)





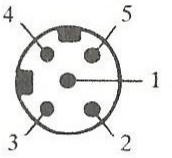
Prise magnétophone

1 masse
2 masse
3 masse
4 rouge (micro)
5 blanc (casque ou ligne)
6 noir (télécommande)
7 masse
8 masse



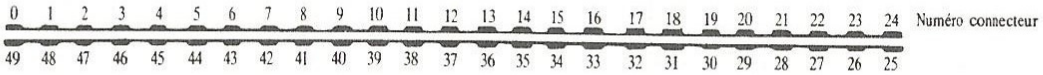
Prise péritélévision

1 Commande rapide
2 Masse
3 Bleu
4 Sync.
5 Rouge
6 Commande lente
7 BF
8 Vert



Prise alimentation

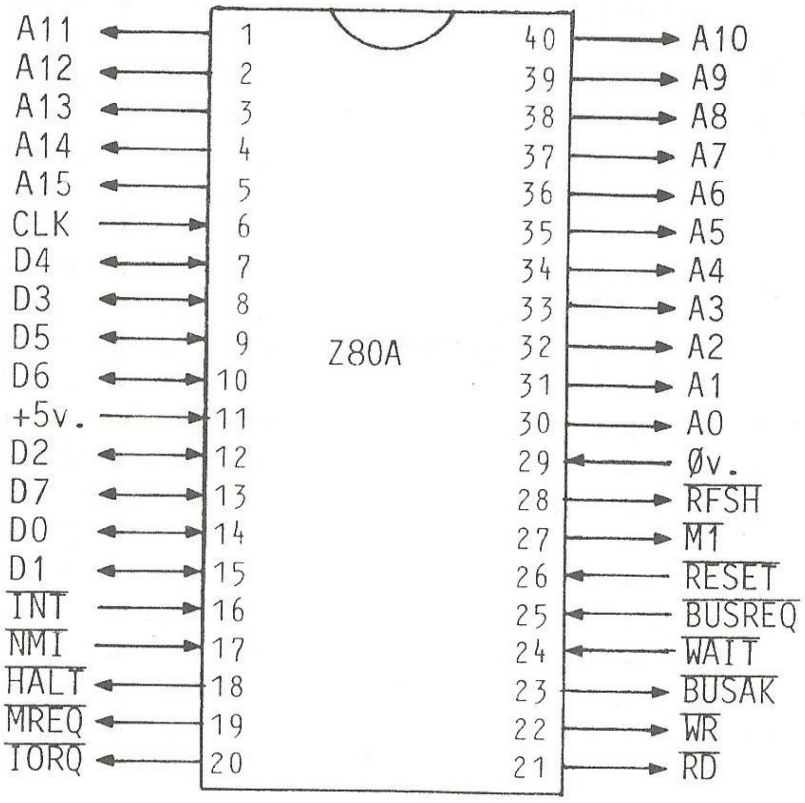
1 NC
2 −12 V
3 + 5 V
4 +12 V
5 Masse



Connecteur bord de carte

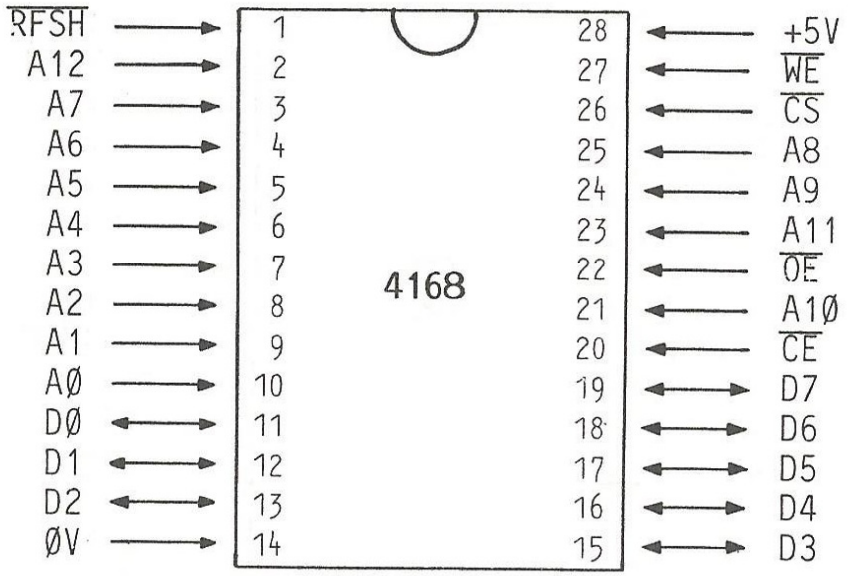
0	$\overline{\text{RFSH1}}$	7	Sound Ex	14	D_2	21	A4	28	A5	35	D_3	42	$\overline{\text{INTEX}}$	49	$\overline{\text{WAITE}}$
1	$\overline{\text{NC}}$	8	$\overline{\text{MI}}$	15	D_0	22	A_2	29	A_7	36	D_5	43	$\overline{\text{CE}_1}$		
2	+5V	9	$\overline{\text{WR}}$	16	A_{14}	23	A_0	30	A_9	37	D_7	44	$\overline{\text{CE}_3}$		
3	+5V	10	$\overline{\text{IORQ}}$	17	A_{12}	24	−12V	31	A_{11}	38	$\overline{\text{MREQ}}$	45	$\overline{\text{CSROM1}}$		
4	$\overline{\text{CE}_4}$	11	$\overline{\text{Reset}}$	18	A_{10}	25	+12V	32	A_{13}	39	$\overline{\text{RD}}$	46	Masse		
5	$\overline{\text{CE}_2}$	12	D_6	19	A_8	26	A_1	33	A_{15}	40	$\overline{\text{RFSH}}$	47	Masse		
6	$\overline{\text{CE}_0}$	13	D_4	20	A_6	27	A_3	34	D_1	41	Horl.	48	NC		

Unité centrale



1-Bit d'adresse 11
2-Bit d'adresse 12
3-Bit d'adresse 13
4-Bit d'adresse 14
5-Bit d'adresse 15
6-Signal d'horloge
7-Bit de donnée 4
8-Bit de donnée 3
9-Bit de donnée 5
10-Bit de donnée 6
11-Tension de 5 volts régulés
12-Bit de donnée 2
13-Bit de donnée 7
14-Bit de donnée Ø
15-Bit de donnée 1
16-Interruptions masquables
17-Interruptions non masquables
18-Arret pour attente sur interruption
19-Demande d'opération mémoire
20-Demande d'entrées/sorties
21-Commande de lecture mémoire
22-Commande d'écriture mémoire
23-Acceptation d'accès direct mémoire
24-Demande d'attente au microprocesseur
25-Demande d'accès direct mémoire
26-Initialisation du microprocesseur
27-Signal de premier cycle
28-Rafraichissement des mémoires dynamiques
29-Masse électrique
30-Bit d'adresse Ø
31-Bit d'adresse 1
32-Bit d'adresse 2
33-Bit d'adresse 3
34-Bit d'adresse 4
35-Bit d'adresse 5
36-Bit d'adresse 6
37-Bit d'adresse 7
38-Bit d'adresse 8
39-Bit d'adresse 9
40-Bit d'adresse 1Ø

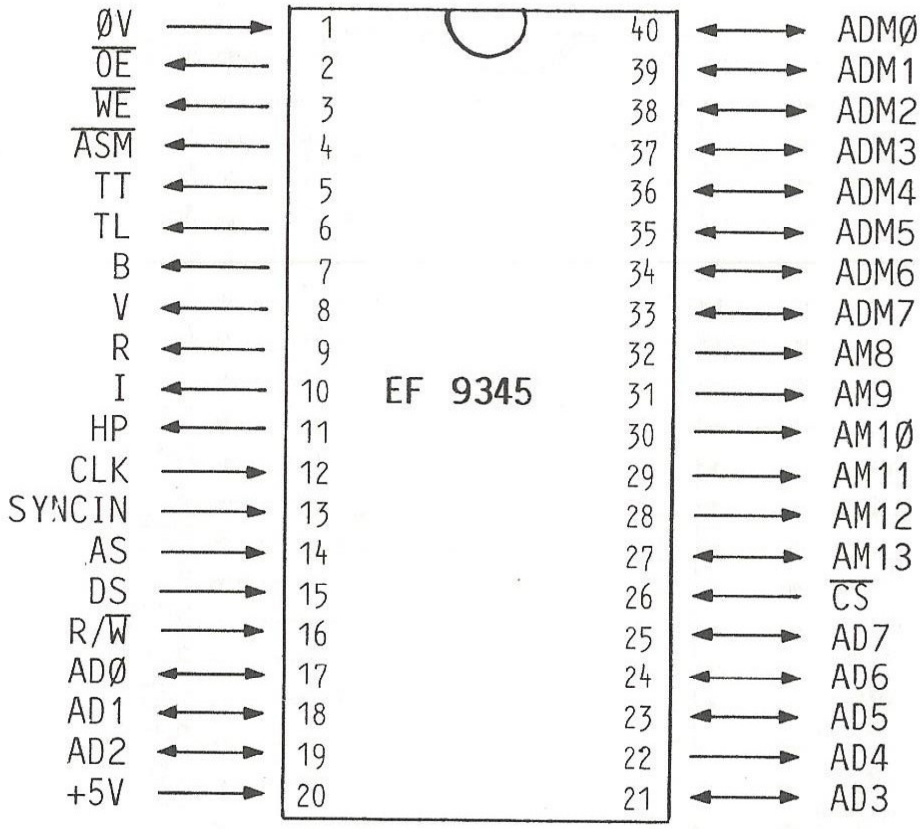
RAM 8K



1-Commande de rafraichissement
2-Bit d'adresse 12
3-Bit d'adresse 7
4-Bit d'adresse 6
5-Bit d'adresse 5
6-Bit d'adresse 4
7-Bit d'adresse 3
8-Bit d'adresse 2
9-Bit d'adresse 1
10-Bit d'adresse Ø
11-Bit de donnée Ø
12-Bit de donnée 1
13-Bit de donnée 2
14-Masse électrique
15-Bit de donnée 3
16-Bit de donnée 4
17-Bit de donnée 5
18-Bit de donnée 6
19-Bit de donnée 7
20-Validation du circuit
21-Bit d'adresse 1Ø
22-Validation en lecture
23-Bit d'adresse 11
24-Bit d'adresse 9
25-Bit d'adresse 8
26-Sélection du circuit
27-Validation en écriture
28-Alimentation +5V

BROCHAGE DES CIRCUITS INTEGRES

Microprocesseur graphique



1-Masse électrique
2-Lecture dans la RAM
3-Ecriture dans la RAM
4- Présence d'une donnée ou d'une adresse sur le bus ADMØ-ADM7
5-Sortie de synchro trame du TV
6-Sortie de symchro ligne du TV
7-Sortie bleu
8-Sortie vert
9-Sortie rouge
10-Sortie d'incrustation
11-Entrée d'horloge 12 mhz
12-Sortie d'horloge 4 mhz
13-Entrée de synchro
14-Indication d'adresse sur le bus de donnée
15-Indication de sortie d'une donnée
16-Indication d'entrée d'une donnée
17-Bit de donnée Ø
18-Bit de donnée 1
19-Bit de donnée 2
20-+5V alimentation
21-Bit de donnée 3
22-Bit de donnée 4
23-Bit de donnée 5
24-Bit de donnée 6
25-Bit de donnée 7
26-Validation du circuit
27-Bit 13
28-Bit 12
29-Bit 11
30-Bit 1Ø
31-Bit 9
32-Bit 8

Bus d'adresses poids fort

33-Bit 7
34-Bit 6
35-Bit 5
36-Bit 4
37-Bit 3
38-Bit 2
39-Bit 1
40-Bit Ø

Bus d'adresses poids faible multiplexé avec le bus de donnée du Z80

TRANSFORMER VOTRE VG 5000 version française en VG 5000 version internationale (ou inversement)

Version internationale

```
10 POKE &"4890",0      : REM messages d'erreurs (en anglais)
20 POKE &"4891",&"B9"  : REM tables du clavier (QWERTY)
30 POKE &"4892",&"12"
```

Version française

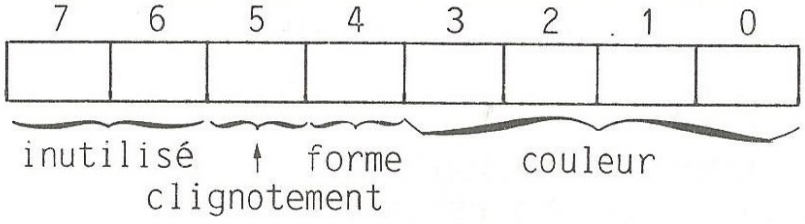
```
10 POKE &"4890",1      : REM messages d'erreurs (en français)
20 POKE &"4891",&"F9"  : REM table du clavier (AZERTY)
30 POKE &"4892",&"11"
```

PASSER EN MINUSCULES/MAJUSCULES PAR PROGRAMME

```
Majuscules  :  POKE &"47FE",1
Minuscules  :  POKE &"47FE",0
```

CHANGER LA COULEUR DU BORD ET L'ASPECT DU CURSEUR

C'est l'octet $47FD qui gère ces deux fonctions : le quartet de poids faible correspond à la couleur, le quartet de poids fort à l'aspect du curseur.

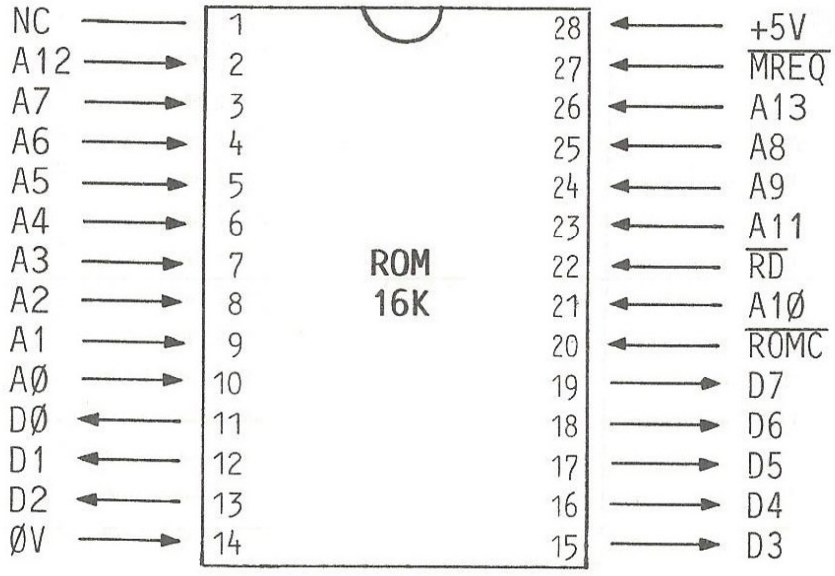


Si le bit 4 est à 0, le curseur est un pavé.
Si le bit 4 est à 1, le curseur est un trait.
Si le bit 5 est à 0, le curseur est fixe.
Si le bit 5 est à 1, le curseur est clignotant.

Ceci revient à choisir le code de la couleur et à ajouter 16 si l'on veut un trait comme curseur, et/ou ajouter 32 si l'on veut qu'il clignote.

BROCHAGE DES CIRCUITS INTEGRES

ROM 16K



1-Non connecté
2-Bit d'adresse 12
3-Bit d'adresse 7
4-Bit d'adresse 6
5-Bit d'adresse 5
6-Bit d'adresse 4
7-Bit d'adresse 3
8-Bit d'adresse 2
9-Bit d'adresse 1
10-Bit d'adresse Ø
11-Bit de donnée Ø
12-Bit de donnée 1
13-Bit de donnée 2
14-Masse électrique
15-Bit de donnée 3
16-Bit de donnée 4
17-Bit de donnée 5
18-Bit de donnée 6
19-Bit de donnée 7
20-Sélection de la ROM
21-Bit d'adresse 1Ø
22-Commande lecture mémoire
23-Bit d'adresse 11
24-Bit d'adresse 9
25-Bit d'adresse 8
26-Bit d'adresse 13
27-Demande d'accès mémoire
28-+5V alimentation

RESTAURER LA POSITION COURANTE DU CURSEUR

Si, pour une raison quelconque, vous désirez restaurer la position courante du curseur après une erreur, rien de plus simple :

```
POKE &"47FD,PEEK(&"47FD") AND 191 : POKE &"4871",
PEEK(&"4871") AND 64
```

LIMITER L'AFFICHAGE A n DECIMALE(S) (n compris entre 0 et 5)

Il suffit de définir la fonction :

```
10 DEFFN (X) = INT (X*10^n)/(10^n)
```

On obtient le résultat par PRINT FN (X).

STOCKER UN PROGRAMME EN LANGAGE MACHINE

Deux emplacements mémoire permettent de stocker un programme en langage machine.

◊ Dans une instruction REM en première ligne du programme.

Le seul intérêt est que l'on puisse sauver le langage machine en même temps que le programme Basic. Les inconvénients sont nombreux : on est limité à 255 caractères (longueur maximale d'une ligne), le listing est des plus bizarres, on n'est jamais sûr de l'adresse de départ (le premier caractère est normalement à l'adresse $4A01, mais cette adresse peut être différente si l'on a modifié txttab) et, enfin, il ne faut pas qu'il y ait de zéro dans la routine, sans quoi le Basic interprète le zéro comme une fin de ligne...

◊ Dans la mémoire réservée à l'utilisateur.

On peut se définir une zone mémoire dans laquelle le Basic n'aura pas accès. Il suffit d'utiliser la commande CLEAR A,B.

C'est l'opérande B qui devra prendre la valeur de début du programme -1 (l'opérande A n'a pas d'importance pour le langage machine).

Exemple

Pour un programme qui commence en $7000 :

```
CLEAR 50,&"6FFF"
```

COMMENT

Exemple

```
Couleur rouge, pavé clignotant → 1 + 32 = 33      donc POKE &"47FD",33
Couleur verte, trait fixe      → 2 + 16 = 18      donc POKE &"47FD",18
Couleur jaune, pavé fixe       → 3                donc POKE &"47FD",3
Couleur bleu, trait clignotant → 4 + 16 + 32 = 52 donc POKE &"47FD",52
```

OBTENIR LA LONGUEUR D'UN PROGRAMME

Il suffit d'exécuter la commande suivante :

```
PRINT (PEEK(&49D8)+256*PEEK(&"49D9)) - (PEEK(&"488E") +
256*PEEK(&"488F"))
```

Ceci revient à soustraire la valeur de (txttab) de (vartab).

OBTENIR LA LONGUEUR DE LA ZONE "VARIABLE"

Il suffit d'exécuter la commande suivante :

```
PRINT(PEEK(&"49DA")+256*PEEK(&"49DB")) - PEEK(&"49D8") +
256*PEEK(&"49D9"))
```

Ceci revient à soustraire la valeur de (vartab) de (arytab).

OBTENIR L'ADRESSE D'UNE LIGNE BASIC

Il suffit de mettre dans la ligne dont l'adresse est inconnue :

```
PRINT PEEK (&"49CF)+256*PEEK(&"49DO")
```

L'intérêt de cette méthode est la modification dynamique du contenu d'une ligne. Par exemple, on peut modifier aisément le contenu d'une instruction SETEG ou SETET.

Exemple de programme

```
100 SETEG 65,"AABBCCDDEEFFGGHHIIJJ"
110 I=PEEK(&"49CF")+256*PEEK(&"49DO")
120 I=I-22
```

On obtient dans I l'adresse du premier caractère de la chaîne de définition du SETEG. Notez la soustraction de 22, car l'on pointait avant sur l'adresse de la ligne 110. Bien entendu, il ne faut pas intercaler de lignes entre les lignes 100 et 110, sans modifier la correction de -22.

Exemple

Il existe notamment un vecteur très utile, celui du RST 30. Il est situé en $47DC. La routine RST 30 est laissée disponible à l'utilisateur ; on peut donc l'utiliser aisément et avantageusement. En effet, un JP ad prend trois octets, alors qu'un RST 30 ne prend qu'un octet.

Pour une routine située en $7000, il suffit de faire :

```
POKE &"47DC",&"C3" : POKE &"47DD",&"00" : POKE &"47DE",
&"70"
```

L'appel à votre routine se fera maintenant en langage d'assemblage par RST 30 (code $F7).

On peut aussi bien modifier le vecteur de sortie de caractère pour que la sortie se fasse, par exemple, à la fois sur l'imprimante et l'écran, ou alors modifier les vecteurs des interruptions pour effectuer des tâches répétitives...

Remarque importante : il est préférable de modifier un vecteur en langage machine, et ce, avec précautions ; en effet, le Basic fait appel constamment à ces vecteurs et une manipulation malencontreuse risque de tout faire "planter".

Notamment, pour modifier les vecteurs d'interruptions masquables, il est nécessaire d'inhiber ces interruptions avant la modification, et de les rétablir ensuite.

REVECTORISER LA TOUCHE CTRL DELTA (RESET)

Il peut être intéressant qu'une pression sur la touche CTRL Delta donne le contrôle à un programme en langage machine.

Il suffit de modifier le vecteur NMI (RESET sur VG 5000) situé en $47EF-47F0, en y mettant un saut à votre programme (voir "Comment ?" ci-dessus).

Puis, vous devez mettre, au début de votre programme qui reprendra le contrôle, les instructions :

EI (code $FB) (uniquement si l'on a besoin des interruptions).
CALL $0074 (code $CD7400) rétablit le niveau de pile.

Maintenant, toute pression sur la touche CTRL Delta fera exécuter votre programme.

UTILISER LA FONCTION USR (A)

On doit d'abord mettre à jour l'adresse à laquelle devra se brancher la fonction USR (adresse d'exécution de la routine). C'est la variable système située en $4834-4835.

Le poids faible de l'adresse d'exécution devra être stocké en $4834 ; quant au poids fort, il devra être implanté en $4835.

Exemple

Pour un programme commençant en $7000 :

```
POKE &"4834",&"00 : POKE &"4835",&"70"
```

A l'intérieur même de la routine, si l'on veut récupérer le paramètre à l'intérieur des parenthèses, il suffit de faire un CALL $252D, celui-ci reviendra dans DE.

Si l'on veut transmettre un paramètre au Basic en retour (A en l'occurrence), il suffit de finir le programme par un JP $29D5 ou JP $0080.

Ces deux routines sont expliquées dans les "routines utiles".

PASSER DES PARAMETRES AVEC L'INSTRUCTION CALL

Lors d'un appel à une routine en langage machine par un CALL, il faut savoir que HL pointe sur le caractère suivant l'opérande.

Exemple

```
CALL &"7000",12
```

Ici, HL pointe automatiquement sur la virgule, on peut donc lire les éventuels paramètres par les routines RST 10 (chget → acquisition d'un caractère), $83 et $86 (acquisition d'une valeur numérique) (voir "routines utiles").

UTILISER LES VECTEURS

Les vecteurs en RAM sont nombreux et très utiles pour des applications particulières pour lesquelles les routines du Basic ne sont pas suffisantes.

Un vecteur comprend trois octets, à la mise sous tension, chaque octet a la valeur $C9 (code de RET). Pour utiliser un vecteur, il suffit de mettre, dans le premier octet, le code $C3 (code de JP ad) et, dans les deux suivants, l'adresse de votre routine de traitement.

REMPLACER UNE PARTIE D'UNE CHAINE PAR UNE AUTRE CHAINE

Si l'on veut remplacer, par exemple, certains caractères de A$, à partir du nième, par B$, il suffit d'exécuter la commande suivante (la longueur de B$ doit être inférieure ou égale à la longueur de A$-n) :

```
A$=LEFT$(A$,n-1)+B$+RIGHT$(A$,LEN(A$)-n-LEN(B$)+1)
```

INSERER UNE CHAINE A INTERIEUR D'UNE AUTRE

Si l'on veut, par exemple, insérer B$ à partir du nième caractère de A$, il suffit d'exécuter la commande suivante :

```
A$=LEFT(A$,n-1)+B$+RIGHT$(A$,LEN(A$)-n+1)
```

COMMENT

REVECTORISER LA TOUCHE CTRL DELTA (RESET) (suite)

Pour exécuter un programme Basic à partir de la touche CTRL Delta, il suffit de vectoriser les interruptions NMI sur la routine suivante :

```
CALL $0074        CD 74 00
LD ($24EA),HL     21 EA 24
EX (SP),HL        E3
XOR A             AF
JP $259A          C3 9A 25
```

JUSTIFIER A DROITE L'AFFICHAGE D'UNE CHAINE DE CARACTERES

Si l'on veut obtenir l'affichage d'une chaîne A$, par exemple, en l'accolant à colonne n, il suffit d'exécuter la commande suivante (la longueur de A$ doit être nécessairement inférieure à n) :

```
PRINT SPC(LEN(A$)-n);A$;
```

JUSTIFIER A GAUCHE L'AFFICHAGE D'UNE CHAINE DE CARACTERES

Si l'on veut obtenir l'affichage d'une chaîne A$, par exemple, à partir de la colonne 0 jusqu'à la colonne n, il suffit d'exécuter la commande suivante (la longueur de A$ doit être nécessairement inférieure à n) :

```
PRINT A$;SPC(LEN(A$)-n);
```

CENTRER L'AFFICHAGE D'UNE CHAINE DE CARACTERES

Si l'on veut centrer une chaîne de caractères à l'écran, par exemple A$, par rapport à la colonne n, il suffit d'exécuter la commande suivante :

```
PRINT SPC(INT(N-LEN(A$)/2));A$;
```

ANNEXES

ANNEXE I

TABLEAU DE CONVERSION

Valeur de l'octet				Complément à un			Complément à deux		
Dec.	ASC		Binaire	Dec.	Hexa.	Binaire	Dec.	Hexa.	Binaire
0		0	00000000	255	FF	11111111	0	0	00000000
1		1	00000001	254	FE	11111110	255	FF	11111111
2		2	00000010	253	FD	11111101	254	FE	11111110
3		3	00000011	252	FC	11111100	253	FD	11111101
4		4	00000100	251	FB	11111011	252	FC	11111100
5		5	00000101	250	FA	11111010	251	FB	11111011
6		6	00000110	249	F9	11111001	250	FA	11111010
7		7	00000111	248	F8	11111000	249	F9	11111001
8		8	00001000	247	F7	11110111	248	F8	11111000
9		9	00001001	246	F6	11110110	247	F7	11110111
10		A	00001010	245	F5	11110101	246	F6	11110110
11		B	00001011	244	F4	11110100	245	F5	11110101
12		C	00001100	243	F3	11110011	244	F4	11110100
13		D	00001101	242	F2	11110010	243	F3	11110011
14		E	00001110	241	F1	11110001	242	F2	11110010
15		F	00001111	240	F0	11110000	241	F1	11110001
16		10	00010000	239	EF	11101111	240	F0	11110000
17		11	00010001	238	EE	11101110	239	EF	11101111
18		12	00010010	237	ED	11101101	238	EE	11101110
19		13	00010011	236	EC	11101100	237	ED	11101101
20		14	00010100	235	EB	11101011	236	EC	11101100
21		15	00010101	234	EA	11101010	235	EB	11101011
22		16	00010110	233	E9	11101001	234	EA	11101010
23		17	00010111	232	E8	11101000	233	E9	11101001
24		18	00011000	231	E7	11100111	232	E8	11101000
25		19	00011001	230	E6	11100110	231	E7	11100111
26		1A	00011010	229	E5	11100101	230	E6	11100110
27		1B	00011011	228	E4	11100100	229	E5	11100101
28		1C	00011100	227	E3	11100011	228	E4	11100100

TABLEAU DE CONVERSION

Valeur de l'octet				Complément à un			Complément à deux		
Dec.	ASC	Hexa.	Binaire	Dec.	Hexa.	Binaire	Dec.	Hexa.	Binaire
29		1D	00011101	226	E2	11100010	227	E3	11100011
30		1E	00011110	225	E1	11100001	226	E2	11100010
31		1F	00011111	224	E0	11100000	225	E1	11100001
32		20	00100000	223	DF	11011111	224	E0	11100000
33	!	21	00100001	222	DE	11011110	223	DF	11011111
34	¢	22	00100010	221	DD	11011101	222	DE	11011110
35	#	23	00100011	220	DC	11011100	221	DD	11011101
36	$	24	00100100	219	DB	11011011	220	DC	11011100
37	%	25	00100101	218	DA	11011010	219	DB	11011011
38	&	26	00100110	217	D9	11011001	218	DA	11011010
39	'	27	00100111	216	D8	11011000	217	D9	11011001
40	(	28	00101000	215	D7	11010111	216	D8	11011000
41	)	29	00101001	214	D6	11010110	215	D7	11010111
42	*	2A	00101010	213	D5	11010101	214	D6	11010110
43	+	2B	00101011	212	D4	11010100	213	D5	11010101
44	,	2C	00101100	211	D3	11010011	212	D4	11010100
45	-	2D	00101101	210	D2	11010010	211	D3	11010011
46	.	2E	00101110	209	D1	11010001	210	D2	11010010
47	/	2F	00101111	208	D0	11010000	209	D1	11010001
48	0	30	00110000	207	CF	11001111	208	D0	11010000
49	1	31	00110001	206	CE	11001110	207	CF	11001111
50	2	32	00110010	205	CD	11001101	206	CE	11001110
51	3	33	00110011	204	CC	11001100	205	CD	11001101
52	4	34	00110100	203	CB	11001011	204	CC	11001100
53	5	35	00110101	202	CA	11001010	203	CB	11001011
54	6	36	00110110	201	C9	11001001	202	CA	11001010
55	7	37	00110111	200	C8	11001000	201	C9	11001001
56	8	38	00111000	199	C7	11000111	200	C8	11001000
57	9	39	00111001	198	C6	11000110	199	C7	11000111
58	:	3A	00111010	197	C5	11000101	198	C6	11000110
59	;	3B	00111011	196	C4	11000100	197	C5	11000101
60	<	3C	00111100	195	C3	11000011	196	C4	11000100
61	=	3D	00111101	194	C2	11000010	195	C3	11000011
62	>	3E	00111110	193	C1	11000001	194	C2	11000010
63	?	3F	00111111	192	C0	11000000	193	C1	11000001
64	@	40	01000000	191	BF	10111111	192	C0	11000000
65	A	41	01000001	190	BE	10111110	191	BF	10111111
66	B	42	01000010	189	BD	10111101	190	BE	10111110
67	C	43	01000011	188	BC	10111100	189	BD	10111101
68	D	44	01000100	187	BB	10111011	188	BC	10111100
69	E	45	01000101	186	BA	10111010	187	BB	10111011
70	F	46	01000110	185	B9	10111001	186	BA	10111010
71	G	47	01000111	184	B8	10111000	185	B9	10111001
72	H	48	01001000	183	B7	10110111	184	B8	10111000
73	I	49	01001001	182	B6	10110110	183	B7	10110111
74	J	4A	01001010	181	B5	10110101	182	B6	10110110
75	K	4B	01001011	180	B4	10110100	181	B5	10110101
76	L	4C	01001100	179	B3	10110011	180	B4	10110100
77	M	4D	01001101	178	B2	10110010	179	B3	10110011
78	N	4E	01001110	177	B1	10110001	178	B2	10110010
79	O	4F	01001111	176	B0	10110000	177	B1	10110001
80	P	50	01010000	175	AF	10101111	176	B0	10110000
81	Q	51	01010001	174	AE	10101110	175	AF	10101111
82	R	52	01010010	173	AD	10101101	174	AE	10101110
83	S	53	01010011	172	AC	10101100	173	AD	10101101
84	T	54	01010100	171	AB	10101011	172	AC	10101100
85	U	55	01010101	170	AA	10101010	171	AB	10101011

TABLEAU DE CONVERSION

Valeur de l'octet				Complément à un			Complément à deux		
Dec.	ASC	Hexa.	Binaire	Dec.	Hexa.	Binaire	Dec.	Hexa.	Binaire
86	V	56	01010110	169	A9	10101001	170	AA	10101010
87	W	57	01010111	168	A8	10101000	169	A9	10101001
88	X	58	01011000	167	A7	10100111	168	A8	10101000
89	Y	59	01011001	166	A6	10100110	167	A7	10100111
90	Z	5A	01011010	165	A5	10100101	166	A6	10100110
91	[	5B	01011011	164	A4	10100100	165	A5	10100101
92	\	5C	01011100	163	A3	10100011	164	A4	10100100
93	]	5D	01011101	162	A2	10100010	163	A3	10100011
94	^	5E	01011110	161	A1	10100001	162	A2	10100010
95	_	5F	01011111	160	A0	10100000	161	A1	10100001
96	`	60	01100000	159	9F	10011111	160	A0	10100000
97	a	61	01100001	158	9E	10011110	159	9F	10011111
98	b	62	01100010	157	9D	10011101	158	9E	10011110
99	c	63	01100011	156	9C	10011100	157	9D	10011101
100	d	64	01100100	155	9B	10011011	156	9C	10011100
101	e	65	01100101	154	9A	10011010	155	9B	10011011
102	f	66	01100110	153	99	10011001	154	9A	10011010
103	g	67	01100111	152	98	10011000	153	99	10011001
104	h	68	01101000	151	97	10010111	152	98	10011000
105	i	69	01101001	150	96	10010110	151	97	10010111
106	j	6A	01101010	149	95	10010101	150	96	10010110
107	k	6B	01101011	148	94	10010100	149	95	10010101
108	l	6C	01101100	147	93	10010011	148	94	10010100
109	m	6D	01101101	146	92	10010010	147	93	10010011
110	n	6E	01101110	145	91	10010001	146	92	10010010
111	o	6F	01101111	144	90	10010000	145	91	10010001
112	p	70	01110000	143	8F	10001111	144	90	10010000
113	q	71	01110001	142	8E	10001110	143	8F	10001111
114	r	72	01110010	141	8D	10001101	142	8E	10001110
115	s	73	01110011	140	8C	10001100	141	8D	10001101
116	t	74	01110100	139	8B	10001011	140	8C	10001100
117	u	75	01110101	138	8A	10001010	139	8B	10001011
118	v	76	01110110	137	89	10001001	138	8A	10001010
119	w	77	01110111	136	88	10001000	137	89	10001001
120	x	78	01111000	135	87	10000111	136	88	10001000
121	y	79	01111001	134	86	10000110	135	87	10000111
122	z	7A	01111010	133	85	10000101	134	86	10000110
123	{	7B	01111011	132	84	10000100	133	85	10000101
124	|	7C	01111100	131	83	10000011	132	84	10000100
125	}	7D	01111101	130	82	10000010	131	83	10000011
126	~	7E	01111110	129	81	10000001	130	82	10000010
127	◆	7F	01111111	128	80	10000000	129	81	10000001
128		80	10000000	127	7F	01111111	128	80	10000000
129		81	10000001	126	7E	01111110	127	7F	01111111
130		82	10000010	125	7D	01111101	126	7E	01111110
131		83	10000011	124	7C	01111100	125	7D	01111101
132		84	10000100	123	7B	01111011	124	7C	01111100
133		85	10000101	122	7A	01111010	123	7B	01111011
134		86	10000110	121	79	01111001	122	7A	01111010
135		87	10000111	120	78	01111000	121	79	01111001
136		88	10001000	119	77	01110111	120	78	01111000
137		89	10001001	118	76	01110110	119	77	01110111
138		8A	10001010	117	75	01110101	118	76	01110110
139		8B	10001011	116	74	01110100	117	75	01110101
140		8C	10001100	115	73	01110011	116	74	01110100
141		8D	10001101	114	72	01110010	115	73	01110011
142		8E	10001110	113	71	01110001	114	72	01110010

TABLEAU DE CONVERSION

Valeur de l'octet				Complément à un			Complément à deux		
Dec.	ASC	Hexa.	Binaire	Dec.	Hexa.	Binaire	Dec.	Hexa.	Binaire
143		8F	10001111	112	70	01110000	113	71	01110001
144		90	10010000	111	6F	01101111	112	70	01110000
145		91	10010001	110	6E	01101110	111	6F	01101111
146		92	10010010	109	6D	01101101	110	6E	01101110
147		93	10010011	108	6C	01101100	109	6D	01101101
148		94	10010100	107	6B	01101011	108	6C	01101100
149		95	10010101	106	6A	01101010	107	6B	01101011
150		96	10010110	105	69	01101001	106	6A	01101010
151		97	10010111	104	68	01101000	105	69	01101001
152		98	10011000	103	67	01100111	104	68	01101000
153		99	10011001	102	66	01100110	103	67	01100111
154		9A	10011010	101	65	01100101	102	66	01100110
155		9B	10011011	100	64	01100100	101	65	01100101
156		9C	10011100	99	63	01100011	100	64	01100100
157		9D	10011101	98	62	01100010	99	63	01100011
158		9E	10011110	97	61	01100001	98	62	01100010
159		9F	10011111	96	60	01100000	97	61	01100001
160		A0	10100000	95	5F	01011111	96	60	01100000
161		A1	10100001	94	5E	01011110	95	5F	01011111
162		A2	10100010	93	5D	01011101	94	5E	01011110
163		A3	10100011	92	5C	01011100	93	5D	01011101
164		A4	10100100	91	5B	01011011	92	5C	01011100
165		A5	10100101	90	5A	01011010	91	5B	01011011
166		A6	10100110	89	59	01011001	90	5A	01011010
167		A7	10100111	88	58	01011000	89	59	01011001
168		A8	10101000	87	57	01010111	88	58	01011000
169		A9	10101001	86	56	01010110	87	57	01010111
170		AA	10101010	85	55	01010101	86	56	01010110
171		AB	10101011	84	54	01010100	85	55	01010101
172		AC	10101100	83	53	01010011	84	54	01010100
173		AD	10101101	82	52	01010010	83	53	01010011
174		AE	10101110	81	51	01010001	82	52	01010010
175		AF	10101111	80	50	01010000	81	51	01010001
176		B0	10110000	79	4F	01001111	80	50	01010000
177		B1	10110001	78	4E	01001110	79	4F	01001111
178		B2	10110010	77	4D	01001101	78	4E	01001110
179		B3	10110011	76	4C	01001100	77	4D	01001101
180		B4	10110100	75	4B	01001011	76	4C	01001100
181		B5	10110101	74	4A	01001010	75	4B	01001011
182		B6	10110110	73	49	01001001	74	4A	01001010
183		B7	10110111	72	48	01001000	73	49	01001001
184		B8	10111000	71	47	01000111	72	48	01001000
185		B9	10111001	70	46	01000110	71	47	01000111
186		BA	10111010	69	45	01000101	70	46	01000110
187		BB	10111011	68	44	01000100	69	45	01000101
188		BC	10111100	67	43	01000011	68	44	01000100
189		BD	10111101	66	42	01000010	67	43	01000011
190		BE	10111110	65	41	01000001	66	42	01000010
191		BF	10111111	64	40	01000000	65	41	01000001
192		C0	11000000	63	3F	00111111	64	40	01000000
193		C1	11000001	62	3E	00111110	63	3F	00111111
194		C2	11000010	61	3D	00111101	62	3E	00111110
195		C3	11000011	60	3C	00111100	61	3D	00111101
196		C4	11000100	59	3B	00111011	60	3C	00111100
197		C5	11000101	58	3A	00111010	59	3B	00111011
198		C6	11000110	57	39	00111001	58	3A	00111010
199		C7	11000111	56	38	00111000	57	39	00111001

TABLEAU DE CONVERSION

Valeur de l'octet				Complément à un			Complément à deux		
Dec.	ASC	Hexa.	Binaire	Dec.	Hexa.	Binaire	Dec.	Hexa.	Binaire
200		C8	11001000	55	37	00110111	56	38	00111000
201		C9	11001001	54	36	00110110	55	37	00110111
202		CA	11001010	53	35	00110101	54	36	00110110
203		CB	11001011	52	34	00110100	53	35	00110101
204		CC	11001100	51	33	00110011	52	34	00110100
205		CD	11001101	50	32	00110010	51	33	00110011
206		CE	11001110	49	31	00110001	50	32	00110010
207		CF	11001111	48	30	00110000	49	31	00110001
208		D0	11010000	47	2F	00101111	48	30	00110000
209		D1	11010001	46	2E	00101110	47	2F	00101111
210		D2	11010010	45	2D	00101101	46	2E	00101110
211		D3	11010011	44	2C	00101100	45	2D	00101101
212		D4	11010100	43	2B	00101011	44	2C	00101100
213		D5	11010101	42	2A	00101010	43	2B	00101011
214		D6	11010110	41	29	00101001	42	2A	00101010
215		D7	11010111	40	28	00101000	41	29	00101001
216		D8	11011000	39	27	00100111	40	28	00101000
217		D9	11011001	38	26	00100110	39	27	00100111
218		DA	11011010	37	25	00100101	38	26	00100110
219		DB	11011011	36	24	00100100	37	25	00100101
220		DC	11011100	35	23	00100011	36	24	00100100
221		DD	11011101	34	22	00100010	35	23	00100011
222		DE	11011110	33	21	00100001	34	22	00100010
223		DF	11011111	32	20	00100000	33	21	00100001
224		E0	11100000	31	1F	00011111	32	20	00100000
225		E1	11100001	30	1E	00011110	31	1F	00011111
226		E2	11100010	29	1D	00011101	30	1E	00011110
227		E3	11100011	28	1C	00011100	29	1D	00011101
228		E4	11100100	27	1B	00011011	28	1C	00011100
229		E5	11100101	26	1A	00011010	27	1B	00011011
230		E6	11100110	25	19	00011001	26	1A	00011010
231		E7	11100111	24	18	00011000	25	19	00011001
232		E8	11101000	23	17	00010111	24	18	00011000
233		E9	11101001	22	16	00010110	23	17	00010111
234		EA	11101010	21	15	00010101	22	16	00010110
235		EB	11101011	20	14	00010100	21	15	00010101
236		EC	11101100	19	13	00010011	20	14	00010100
237		ED	11101101	18	12	00010010	19	13	00010011
238		EE	11101110	17	11	00010001	18	12	00010010
239		EF	11101111	16	10	00010000	17	11	00010001
240		F0	11110000	15	F	00001111	16	10	00010000
241		F1	11110001	14	E	00001110	15	F	00001111
242		F2	11110010	13	D	00001101	14	E	00001110
243		F3	11110011	12	C	00001100	13	D	00001101
244		F4	11110100	11	B	00001011	12	C	00001100
245		F5	11110101	10	A	00001010	11	B	00001011
246		F6	11110110	9	9	00001001	10	A	00001010
247		F7	11110111	8	8	00001000	9	9	00001001
248		F8	11111000	7	7	00000111	8	8	00001000
249		F9	11111001	6	6	00000110	7	7	00000111
250		FA	11111010	5	5	00000101	6	6	00000110
251		FB	11111011	4	4	00000100	5	5	00000101
252		FC	11111100	3	3	00000011	4	4	00000100
253		FD	11111101	2	2	00000010	3	3	00000011
254		FE	11111110	1	1	00000001	2	2	00000010
255		FF	11111111	0	0	00000000	1	1	00000001

ANNEXE II

CARACTERISTIQUES PRINCIPALES

Présentation

Coffret	: moulé plastique noir.
Clavier	: 63 touches au standard AZERTY à déplacement. Blocage majuscules/minuscules. Touche CTRL permettant d'accéder aux 33 instructions Basic préprogrammées. Commandes à accès direct : PRINT, LIST, RUN, STOP. Effacement écran, effacement ligne, effacement caractère, insertion ligne, insertion caractère.
Dimensions	: 28 cm x 21 cm x 4 cm.

Alimentation extérieure

Boîtier	: moulé plastique noir.
Dimensions	: 17 cm x 10,3 cm x 6 cm.
Tension entrée	: 220 V $\pm$ 10/15 % 50 Hz.
Tensions sortie	: 12 V 150 mA. 5 V 1,1 A

CARACTERISTIQUES PRINCIPALES

Caractéristiques techniques

Processeur et mémoires

Microprocesseur	: CPU Z80, processeur graphique EF9345.
Horloge	: 4 Mhz.
Mémoire écran	: 8 Ko.
Mémoire utilisateur	: 16 Ko.
Mémoire morte	: 16 Ko.

Langage

Basic 80 de Microsoft.

Affichage

Résolution	: 25 lignes de 40 caractères.
Matrice caractère	: 8 x 10 points.
Couleurs	: 8.
Jeu de caractères	: 110 caractères ASCII texte, 128 caractères graphiques, 192 caractères utilisateur.

Musique

Synthétiseur	: tempo, octave, durée, pause programmables.
Octaves	: 4.
Tempo	: 256
Durée	: de 0 à 100.

Magnétophone

Type	: standard, avec prises jack micro et écouteur.
Vitesse	: 1200 ou 2400 bauds.

Cordons livrés avec l'appareil

Magnétophone	: cordon avec prises jack micro et écouteur.
Téléviseur	: cordon péritel.

Extensions

Manettes de jeu	: permet de connecter deux manettes de jeu équipées chacune de deux boutons "feu".
Imprimante	: imprimante à aiguilles, 40 colonnes, graphique.
Modulateur Secam	: permet aux possesseurs de téléviseurs non équipés d'une prise péritel d'utiliser le VG 5000.
Extension mémoire	: rajoute 16Ko de mémoire vive pour l'utilisateur.

INDEX

CONSEILS DE LECTURE

Pour approfondir vos connaissances en Basic VG5000, et mieux connaître le système de votre Philips, P.S.I. vous propose une palette d'ouvrages utiles.

Pour maîtriser le BASIC VG5000

VG5000 pour tous - Jean-Michel JEGO (Editions du P.S.I.)

Une initiation au BASIC VG5000 à l'aide d'exemples commentés et faciles à adapter.

Pour mieux connaître le système Philips

Programmer en Assembleur - Alain PINAUD
(Editions du P.S.I.)

Introduction complète au langage machine et à l'Assembleur Z80.

Le système CP/M pour Z80, adaptation du BIOS et compléments - Fabienne et Philippe GYSEL (Editests)

Pour initiés, des exemples très poussés d'utilisation de nombreuses fonctions du CP/M80 à partir des instructions Z80.

Achevé d'imprimer en juillet 1985
sur les presses de l'imprimerie Laballery et Cie
58500 Clamecy

Dépôt légal : juillet 1985
N° d'impression : 506084
N° d'édition : 86595-259-1
ISBN : 2-86595-259-2

Votre avis nous intéresse

- Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques sur le présent livre.
- Si vous souhaitez des éclaircissements techniques, écrivez-nous, nous adresserons votre demande à l'auteur qui ne manquera pas de vous répondre directement.

- Ce livre vous donne-t-il toute satisfaction?

- Y a-t-il un aspect du problème que vous auriez aimé voir abordé?

Comment avez-vous eu connaissance de ce livre?

- ☐ publicité
- ☐ catalogue
- ☐ boutique micro
- ☐ autres
- ☐ cadeau
- ☐ librairie
- ☐ exposition

Avez-vous déjà acquis des livres PSI?

lesquels?

qu'en pensez-vous?

Nom Prénom Age

Adresse

Profession

Centre d'intérêt

mémento

Ne tenez plus votre livre d'une main tout en pianotant de l'autre sur le clavier, "Clefs pour VG 5000" est un mémento qui s'ouvre à la bonne page et vous permet d'accéder efficacement à toutes les informations dont vous avez besoin : jeu d'instructions du Z80, répertoire des instructions et des opérateurs BASIC, carte mémoire du VG 5000, brochage des circuits électronique...

"Clefs pour VG 5000" est aussi un recueil d'astuces : changer la couleur du bord du curseur, utiliser la fonction USR (A), revectoriser la touche CTRL Delta, sont autant de conseils pour faire ressortir toute l'originalité de votre Philips. N'oubliez pas vos clefs !

CLEFS POUR VG 5000

9 782865 952595

Éditions du PSI
B.P. 86
77402 Lagny/Marne
France

ISBN : 2.86595.259.2

Couverture Marie-Line de Nay

120 FF