

# mémento

## **CLEFS POUR AMSTRAD CPC 464 ~ 664 et 6128**

**1. Système de base**

**Daniel Martin**

Editions du 

# **CLEFS POUR AMSTRAD**

## **1. Système de base**



# PRESENTATION

Le présent ouvrage constitue un véritable répertoire pour l'Amstrad. De l'architecture interne au Basic en passant par le langage machine, le brochage, la structure et la programmation des principaux circuits, le logiciel interne et les trucs et astuces, il recense tout et ce, en l'agrémentant d'explications succinctes et de quelques exemples.

Du programmeur-analyste au simple curieux de l'informatique, chacun y trouvera les informations nécessaires à la bonne utilisation de son Amstrad.

Amstrad Amdos et Amstrad CPC 464, 664 et 6128 sont des marques déposées de Amstrad Consumer Electronics.

# SOMMAIRE

	Pages
PRESENTATION	5
CHAPITRE I - SCHEMA GENERAL ET ARCHITECTURE INTERNE	11
CHAPITRE II - BASIC	13
Caractéristiques générales	13
Instructions Basic	14
Fonctions Basic	27
Mots-clés et codes associés	33
Codes ASCII et graphiques	35
Codes et messages d'erreurs	45
Format de stockage d'une ligne Basic en mémoire	48
CHAPITRE III - LANGAGE MACHINE	51
Organisation interne du Z80	51
Registres du Z80	52
Jeu d'instructions du Z80	53
Codes des instructions Z80 par ordre alphabétique	59
Tableaux de désassemblage	74



# SOMMAIRE

## CHAPITRE IV - LOGICIEL INTERNE 79

Généralités	79
Table des points d'entrée des routines système	81
- Le gestionnaire clavier	81
- Le gestionnaire du mode texte	84
- Le gestionnaire graphique	88
- Le gestionnaire d'écran	91
- Le gestionnaire cassette	95
- Le gestionnaire sonore	98
- Le noyau (Kernel)	100
- Interfaçage avec le matériel	103
- Le bloc de saut	104
Les vecteurs d'indirection	105
Les vecteurs noyau et les RESTART	107
Les vecteurs d'appel des routines mathématiques	111
Les principales variables système	115
Adresses principales de la ROM inférieure	120
Adresses principales de la ROM supérieure	126
Les adresses réelles ROM	131
Adresses d'exécution des mots-clés du Basic	133
Les blocs de contrôle	135
- ROM expansion	135
- Streams	135
- Queue sonore	135
- Bloc de contrôle d'amplitude ou de timbre	135
- Vecteur encre	136
- Format des deux octets qui suivent un RESTART	136
- Format des fichiers cassette	136
- Bloc d'événement	137
- Bloc de contrôle d'interruption normale	138
- Bloc d'interruption rapide et d'interruption CRT	138

## CHAPITRE V - STRUCTURE INTERNE ET PROGRAMMATION DES PRINCIPAUX CIRCUITS 139

Circuit AY3-8912	139
- Structure interne	139
- Les différents registres du PSG	139
- Programmation de l'AY3-8912	142
Circuit PPI 8255	143
- Généralités	143
- Découpage des PORTS	143
- Programmation	144
Circuit CRT 6845	146
- Généralités	146
- Les différents registres du 6845	146
- Programmation	147



Circuit VIDEO GATE ARRAY	148
- Généralités	148
- Programmation	148

## CHAPITRE VI - TRUCS ET ASTUCES 151

Dump hexa mémoire ROM inférieure et supérieure sur imprimante	151
Dump ASCII mémoire ROM inférieure et supérieure sur imprimante	152
Démarrage et arrêt du moteur de la cassette	153
Protection de programme	153
Bruits originaux	153
Programme permettant de tracer des cercles et des ellipses	154
Scanning du clavier	154
Modification originale de la couleur de fond	155
Installation d'une routine en langage machine dans une remarque	155

## CHAPITRE VII - CONNECTEURS ET BROCHAGE DES PRINCIPAUX CIRCUITS UTILISES 157

Brochage de l'AY3-8912	157
Brochage du CRT 6845	158
Brochage du PPI 8255	159
Brochage du Z80	160
Le connecteur pour la manette de jeux	162
Le connecteur sortie vidéo	163
Le connecteur sortie expansion	164
Le connecteur sortie imprimante	165

## ANNEXE 167

Table des valeurs pour la gamme chromatique	167
Table des codes de contrôle du terminal	168
Table des adresses des PORTS utilisés	169
Structure de la mémoire écran	170
Table des couleurs	172
Table des codes clavier (numéros des touches)	173

## INDEX 175



## SCHEMA GENERAL ET ARCHITECTURE INTERNE

Le diagramme de la page suivante nous montre les différents circuits composant le matériel.

Le système est articulé autour d'une unité centrale Z80 avec une horloge de 4mhz.

Le circuit le plus important de l'Amstrad, à l'exception du microprocesseur lui-même, est certainement le "**GATE ARRAY**" qui contient toute la logique de contrôle du système. En particulier, il contrôle la couleur, le mode écran et il gère les mémoires mortes (ROM).

En conjonction avec le CRTC 6845 (cathode ray tube controller), le "**GATE ARRAY**" gère tous les signaux vidéos pour le moniteur (écran).

Un autre circuit important est le PSG AY3-8912 (PSG = Programmable Sound Generator). Ce circuit possède trois canaux distincts avec un générateur de bruit et un contrôleur d'enveloppe pour chaque canal. La façon de le programmer sera décrite dans le *chapitre 5* du présent manuel.

Le système est également pourvu d'un port d'entrée-sortie qui est utilisé pour lire le clavier et la manette de jeux.

Le dernier circuit principal est le PPI 8255. Il joue un rôle important au niveau de la gestion de la manette de jeux, du port parallèle d'imprimante, de l'enregistreur à cassettes et aussi au niveau de la sélection des colonnes du clavier.

Le système possède 64 K de mémoire vive (RAM) et 32 K de mémoire morte (ROM) qui contiennent le système d'exploitation et le Basic.

La **mémoire morte** (ROM) de 32 K, située sur le circuit central, est découpée logiquement en deux blocs de 16 K. Les 16 K inférieurs vont de l'adresse 0000 à l'adresse 3FFF et les 16 K supérieurs occupent les adresses C000 à FFFF. Ces deux mémoires peuvent être séparément mises en circuit ou hors circuit par le contrôle de la **GATE ARRAY**.

Sur le **PORT** d'extension, nous trouvons un signal pouvant être utilisé pour déconnecter les mémoires mortes internes et permet-

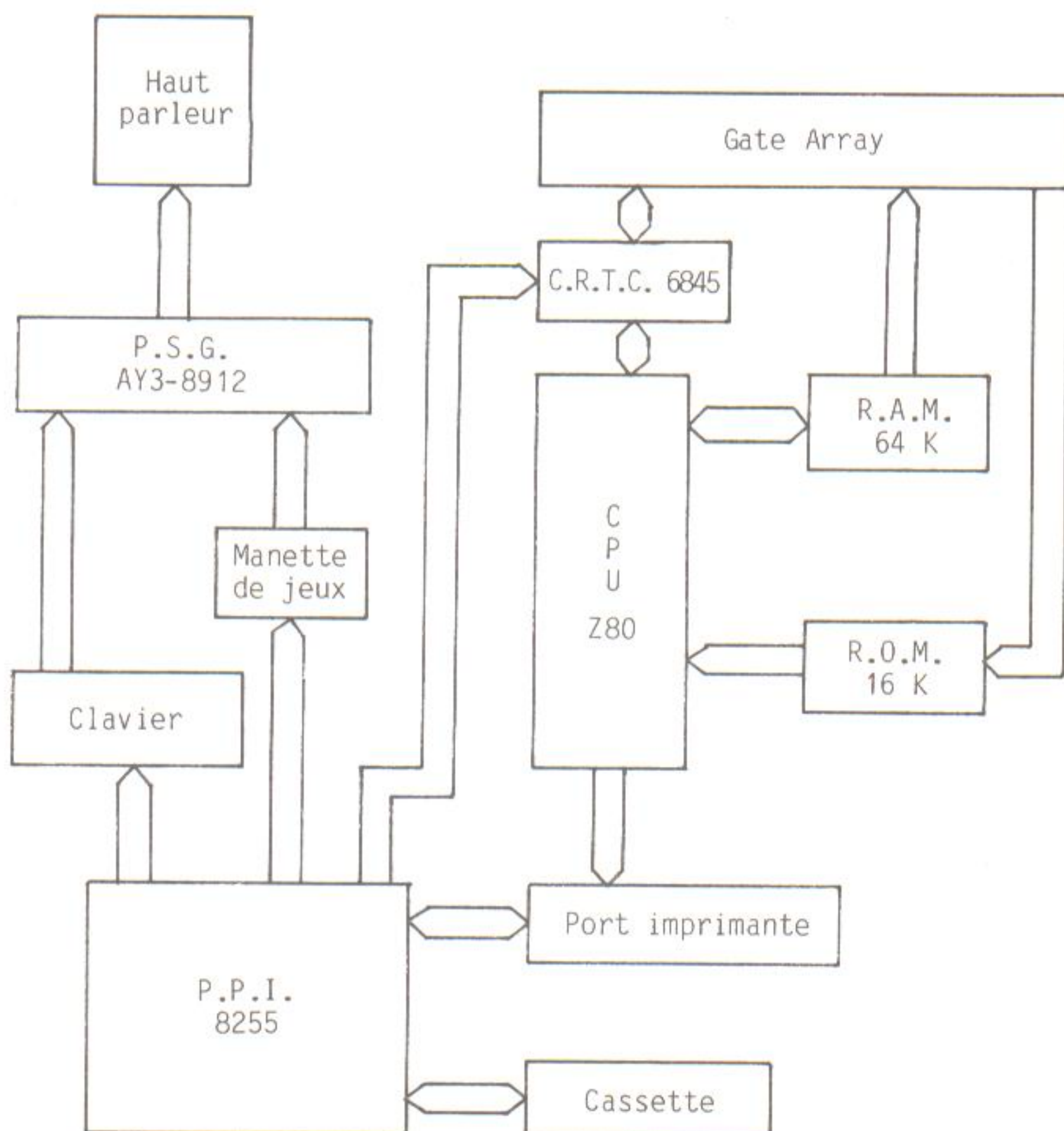


## SCHEMA GENERAL ET ARCHITECTURE INTERNE

tre à des mémoires externes d'accéder au processeur. Cette possibilité permet, par exemple, d'installer un disque souple.

La **mémoire vive** est constituée de 64 K octets de RAM dynamiques qui vont de l'adresse 0000 à FFFF. Les 16 K inférieurs et les 16 K supérieurs se trouvent donc en superposition avec la ROM. Normalement, cela ne pose aucun problème. Lorsque l'on écrit, on écrit automatiquement dans la RAM. Lorsque l'on lit, il faut sélectionner au préalable la ROM ou la RAM, suivant ce que l'on désire lire.

La **mémoire d'écran** se situe dans la mémoire centrale et occupe 16 K. Elle peut se trouver à l'adresse 0000, à l'adresse 4000, à l'adresse 8000 ou à l'adresse C000. Généralement, au départ, elle se trouve à l'adresse C000.





## CARACTERISTIQUES GENERALES

*Espace maximum disponible* : 43 533 octets.

*Nom des variables* : 1 à 40 caractères.

*Données*

Entier : de -32 768 à 32 767.

Simple précision : de 293 874 E-39 à 170 141 E30 sur neuf chiffres significatifs ou six en notation exponentielle.

Chaîne : 0 à 255 caractères.

*Longueur d'une ligne de programme* : 255 caractères maximum.

*Numéro de ligne* : de 1 à 65 535 inclus.

*Occupation en mémoire* : une ligne Basic occupe au minimum six octets. Deux pour le numéro de ligne, deux pour la longueur de la ligne, un pour la séparation et un pour une instruction minimum (REM, PRINT).

*Allocation des variables*

Entier positif de 1 à 9 : 1 octet.

Entier négatif de 1 à 9 : 2 octets.

Entier positif de 10 à 255 : 2 octets.

Entier négatif de 10 à 255 : 3 octets.

Simple précision positif (255-65535) : 3 octets.

Simple précision négatif (255-65535) : 4 octets.

Entier supérieur à 65535 positif ou non entier positif : 6 octets.

Entier supérieur à 65535 négatif ou non entier négatif : 7 octets.



## INSTRUCTIONS BASIC

AFTER	<p>AFTER X,[Y] GOSUB N</p> <p>Appelle un sous-programme après avoir attendu X cinquantièmes de seconde. Y, facultatif, indique quel chronomètre doit être utilisé. Il y en a quatre, numérotés de 0 à 3. Si Y n'est pas précisé, il est supposé être égal à 0.</p>
AUTO	<p>AUTO [N],[X]</p> <p>Numérote automatiquement les lignes d'un programme en commençant par la ligne N avec X comme incrément. Par défaut, N et X valent 10.</p>
BORDER	<p>BORDER X,[Y]</p> <p>X et Y représentent des numéros de couleurs (0 à 26). Permet de changer la couleur du bord de l'écran. Si Y est précisé, les deux couleurs alternent à la vitesse déterminée par la commande SPEED INK.</p>
CALL	<p>CALL ADR[,liste de paramètres]</p> <p>Commande utilisée en Basic pour faire appel à un sous-programme en langage machine situé à l'adresse ADR. On peut passer une liste de paramètres à ce sous-programme.</p>
CAT	<p>CAT</p> <p>Lit la cassette et donne les noms des fichiers trouvés sans effacer le programme qui se trouve en mémoire centrale.</p>
CHAIN	<p>CHAIN nom[,N]</p> <p>Charge un programme de la cassette vers la mémoire centrale en supprimant celui qui s'y trouve déjà, puis l'exécute en commençant par la ligne numéro N, ou si N n'est pas précisé, par la ligne portant le plus petit numéro.</p>
CLEAR	<p>CLEAR</p> <p>Efface les contenus de toutes les variables et de tous les fichiers.</p>
CLG	<p>CLG</p> <p>Efface l'écran graphique.</p>
CLOSEIN	<p>CLOSEIN</p> <p>Ferme un fichier sur cassette ouvert en entrée.</p>
CLOSEOUT	<p>CLOSEOUT</p> <p>Ferme un fichier sur cassette ouvert en sortie.</p>



CLS	CLS [#n] Nettoie l'écran ou la fenêtre d'écran et le laisse coloré dans la couleur définie par la dernière instruction PAPER. n est un numéro de canal compris entre 0 et 7 et correspond à l'écran défini par l'instruction WINDOW.
CONT	CONT Poursuit l'exécution d'un programme après STOP, END ou *break* pour autant que le programme n'ait pas été modifié.
DATA	DATA donnée1,donnée2,donnée3,... Permet de stocker dans un programme une liste de constantes lues par la fonction READ.
DEF FN	DEF FNf[(X,...)]=expr Permet de définir une fonction utilisateur ; f représente le nom de la fonction, (X,...) ses paramètres formels et expr son expression générale.
DEFINT	DEFINT X-Y      DEFINT X,Y,... Définition d'un ensemble de variables dans la plage X-Y ou suivant la liste X,Y,... comme étant de type ENTIER.
DEFREAL	DEFREAL X-Y      DEFREAL X,Y,... Définition d'un ensemble de variables dans la plage X-Y ou suivant la liste X,Y,... comme étant de type SIMPLE PRECISION.
DEFSTR	DEFSTR X-Y      DEFSTR X,Y,... Définition d'un ensemble de variables dans la plage X-Y ou suivant la liste X,Y,... comme étant de type ALPHANUMERIQUES (chaînes de caractères).
DEG	DEG Etablit le mode de calcul en degrés. En standard, les fonctions sinusoides utilisent les radians pour les données numériques. La commande DEG est désactivée par les commandes CLEAR et RAD ou par le chargement d'un autre programme.
DELETE	DELETE (N1,N2,...)      DELETE N1-N2 Efface du programme contenu en mémoire centrale soit la série de lignes (N1,N2,...), soit les lignes comprises entre la ligne portant le numéro N1 et la ligne portant le numéro N2.



## INSTRUCTIONS BASIC

DI	Arrêt des interruptions. Toutes les commandes générant des interruptions, à l'exception du BREAK, deviennent inopérantes.
DIM	DIM var(n)      DIM var(n1,n2,...),var(n1,n2,...) Dimensionnement d'un tableau (var) de 1 à N dimensions. Par défaut, une variable est automatiquement dimensionnée à 10 (var(10)).
DRAW	DRAW X,Y,A Dessine une ligne sur l'écran à partir du curseur graphique vers la position de coordonnées (X,Y) avec la couleur numéro A.
DRAWR	DRAWR X,Y,A Dessine une ligne sur l'écran à partir du curseur graphique vers une position relative de (+X,+Y) par rapport à ce curseur graphique avec la couleur numéro A.
EDIT	EDIT N Passage en mode édition pour la ligne numéro N.
EI	Autorisation d'interruptions. Annule l'effet de la commande DI.
END	END Instruction de fin d'exécution d'un programme.
ENT	ENT NE[,SE] Enveloppe de ton. Permet d'introduire des vibratos sur la note : NE représente le numéro d'enveloppe (de 0 à 15) ; SE comprend trois nombres par section (compte de pas, taille de pas en fréquence, temps de pause), cinq sections peuvent être décrites.
ENV	ENV NE[,SE] Enveloppe de volume. Permet de définir le type de son : NE représente le numéro d'enveloppe (de 0 à 15) ; SE comprend trois nombres par section (nombre de pas, taille en volume du pas, temps pour chaque pas), cinq sections peuvent être décrites.
ERASE	ERASE liste de noms de variables Permet de libérer la place mémoire retenue par la commande DIM.



ERROR	<p>ERROR N</p> <p>N représente un nombre entier. Permet de repérer un type d'erreur déterminé et de décider d'une action définie lorsqu'elle a lieu.</p>
EVERY	<p>EVERY N,M GOSUB NL</p> <p>Signifie : tous les N centièmes de seconde, comptés sur le chronomètre numéro M, effectuer le sous-programme situé à la ligne numéro NL. Quatre chronomètres sont disponibles et numérotés de 0 à 3. Cette commande permet de faire appel à un sous-programme à intervalles réguliers.</p>
FOR	<p>FOR var=D to F [STEP P]</p> <p>Introduit une boucle. Toutes les instructions, comprises entre FOR var=D to F [STEP P] et le NEXT var correspondant, seront répétées pour toutes les valeurs de var allant de D à F par pas de P (si P n'est pas spécifié, de 1 en 1):</p> <pre> 10 FOR I=1 TO 20 STEP 2 20 PRINT I," ",I*I NEXT I </pre>
GOSUB	<p>GOSUB NL</p> <p>Appelle le sous-programme situé à la ligne portant le numéro NL.</p>
GOTO	<p>GOTO NL</p> <p>Effectue un saut à la ligne portant le numéro NL.</p>
IF	<p>IF condition THEN instruction</p> <p>Effectue l'instruction qui suit le THEN si la condition suivant IF est remplie.</p> <pre> IF A=3 THEN GOSUB 1000 </pre>
INK	<p>INK encre,couleur[,couleur]</p> <p>Suivant le MODE choisi pour l'écran, un certain nombre d'encres (e) est possible. INK permet de changer la couleur d'encre (encre) et la couleur de fond (couleur). Si l'on spécifie deux couleurs de fond, elles alterneront tous les cinquantième de seconde.</p>



## INSTRUCTIONS BASIC

### INPUT

INPUT [#numéro de canal,][;][chaîne;]

liste de variables

INPUT [#numéro de canal,][;][chaîne,]

liste de variables

Lit les données venant du canal précisé et les affecte aux différentes variables nommées. Le premier [;] supprime le passage à la ligne à la fin de ce que l'on tape. Le ";", après la chaîne, fait apparaître un ? tandis qu'une "," fait apparaître le message "?redo from start" lorsque l'on a frappé quelque chose de faux.

Quand un canal cassette est précisé, il n'y a pas d'invite. Un élément du fichier sera affecté à chaque variable de la liste.

### KEY

KEY nombre entier, chaîne de caractères

Permet de définir une nouvelle touche de fonction. Le nombre entier définit la touche à laquelle sera affectée la chaîne de caractères. Il vaut de 128 à 140, la touche 0 du clavier est désignée par le nombre 128, la touche 1 par 129 ..., la touche 9 par 137, la touche "." par 138, la touche "ENTER" par 139 et les touches "CTRL" et "ENTER" ensemble par 140.

*Exemple*

KEY 132,"RUN"+CHR\$(13)

Associe à la touche 4 du clavier numérique la commande RUN suivie d'"ENTER".

### KEY DEF

KEY DEF numéro de touche, répétition, num caractère  
Change la valeur produite par une touche.

KEY DEF 45,1,65 fait afficher A à la touche J en permettant la répétition lorsque l'on continue à pousser dessus.

KEY DEF 46,0,63 fait afficher ? à la touche N sans permettre la répétition.

### LET

LET variable = expression

Affecte à une variable le résultat de l'expression à droite du signe égal.

LET A = 500 \* 3

En Basic AMSTRAD, on peut bien entendu se contenter d'écrire A = 500 \* 3. LET est utilisé seulement dans le cas où on désire qu'un programme reste compatible avec d'anciens programmes.



LINE INPUT	<p>LINE INPUT [#numéro de canal,][;]                                    [chaîne;]variable</p> <p>LINE INPUT "NOM";A\$</p> <p>Lit une ligne entière du canal indiqué (0 par défaut). Lorsqu'une virgule se trouve dans la variable, elle y reste incluse alors qu'un simple INPUT aurait scindé la variable à cet endroit.</p>
LIST	<p>LIST [numéros de lignes][, #numéro de canal]</p> <p>Liste le programme sur le canal désiré. 0 correspond à l'écran et 8 à l'imprimante. On peut arrêter le défilement sur écran en poussant ESC une fois et continuer en appuyant sur une autre touche. En appuyant deux fois sur ESC, on retourne au mode direct.</p>
LOAD	<p>LOAD [nom de fichier][, adresse]</p> <p>Charge un programme Basic de la cassette vers la mémoire centrale en remplaçant tout ce qui s'y trouvait déjà. Dans le cas d'un programme binaire, on peut spécifier l'adresse de chargement.</p>
LOCATE	<p>LOCATE [#n° de canal,]X,Y</p> <p>Place le curseur de texte à la position de coordonnées (X,Y) relative à l'origine de la fenêtre d'écran (WINDOW). Le point de coordonnées (1,1) est le point situé dans le coin supérieur gauche de la fenêtre.</p>
MEMORY	<p>MEMORY adresse</p> <p>Permet de redéfinir l'adresse de l'octet le plus élevé dans l'espace mémoire utilisé par le Basic. En standard, cet octet se trouve à l'adresse AB7F.</p>
MERGE	<p>MERGE ["nom de fichier"]</p> <p>Identique à LOAD, mais n'effectue pas de NEW avant le chargement du programme. Si des numéros de ligne sont identiques, leur contenu devient celui du nouveau programme. Si le nom du fichier n'est pas précisé, le premier programme rencontré sera lu. Un programme sur cassette, précédé du signe "!", est protégé et ne sera donc pas lu.</p>
MODE	<p>MODE N</p> <p>Permet de changer de mode écran (N=0, 1 ou 2). Efface l'écran et se met en INK0, quel que soit le PAPER INK utilisé à ce moment. Lors de cette commande, toutes les fenêtres de texte et de</p>



## INSTRUCTIONS BASIC

graphique sont retournées à l'écran entier, tandis que les curseurs retournent à leur point d'origine.

### MOVE

MOVE X,Y

Positionne le curseur graphique à la position absolue de coordonnées (X,Y).

### MOVER

MOVER X,Y

Positionne le curseur graphique à une position de coordonnées (X,Y) relative à la position actuelle.

### NEW

Nettoie la mémoire.

Les programmes et variables disparaissent, mais les définitions des touches (KEY) et l'affichage restent.

### NEXT

FOR I=1 TO 10 : NEXT [I]

Détermine la fin d'une boucle commencée par FOR.

### ON...GOTO

ON n GOTO liste de numéros de lignes

### ON...GOSUB

ON n GOSUB liste de numéros de lignes

Pour ON A GOTO 100,110,130,132,170,300,320,1000 :

si A=1, le branchement se fera à la ligne 100

si A=2, le branchement se fera à la ligne 110

.. ..

si A=7, le branchement se fera à la ligne 320,...

### ON BREAK GOSUB

ON BREAK GOSUB numéro de ligne

Appelle un sous-programme lorsqu'il y a un BREAK en cours de programme. Un BREAK s'effectue en appuyant deux fois consécutives sur la touche ESC.

### ON BREAK STOP

Annule l'effet de l'instruction ON BREAK GOSUB.

### ON ERROR GOTO

ON ERROR GOTO numéro de ligne

Branche l'exécution à la ligne précisée lorsqu'une erreur se produit.

### ON SQ GOSUB

ON SQ (n) GOSUB nl

Exécute la sous-routine située à la ligne nl lorsque la queue correspondant au canal sonore (n) n'est plus pleine. (n) ne peut valoir que 1, 2 ou 4 correspondant aux canaux respectifs A, B ou C.



OPENIN	<p>OPENIN "nom de fichier"</p> <p>Ouvre un fichier cassette pour permettre au programme présent en mémoire centrale d'utiliser les données qui s'y trouvent. Si le nom du fichier est précédé d'un "!", les messages habituels de manipulation de la cassette n'apparaissent pas et le programme lit directement le premier bloc du fichier.</p>
OPENOUT	<p>OPENOUT "nom de fichier"</p> <p>Ouvre un fichier sur cassette pour permettre au programme d'aller y déposer des données. Si le nom du fichier est précédé d'un "!", les messages habituels de manipulation de la cassette n'apparaissent pas. Le programme crée, de suite, son premier bloc de transfert (buffer) de 2 K et rien n'est écrit sur la cassette tant que le buffer n'est pas plein ou qu'une commande CLOSE-OUT n'a pas fermé le fichier.</p>
ORIGIN	<p>ORIGIN X,Y [,G,D,H,B]</p> <p>Détermine les coordonnées (X,Y) du point de départ du curseur graphique. Les éléments optionnels G,D,H,B permettent de définir une nouvelle fenêtre.</p>
OUT	<p>OUT numéro de port, nombre entier</p> <p>Envoie le nombre entier vers le port spécifié. Le nombre entier peut être compris entre 0 et 255 et le numéro du port entre 0 et 65535.</p>
PAPER	<p>PAPER [#n° du canal,] n° d'encre</p> <p>Définit la couleur de fond des prochains caractères qui seront écrits sur l'écran.</p>
PEN	<p>PEN [#n° de canal,] n° d'encre</p> <p>Définit la couleur des prochains caractères qui seront écrits sur l'écran.</p>
PLOT	<p>PLOT X,Y [,n° d'encre]</p> <p>Affiche, à l'écran, le point de coordonnées (X,Y) dans la couleur précisée par n° d'encre ou, à défaut, dans la couleur utilisée précédemment.</p>
PLOTR	<p>PLOTR X,Y [,n° d'encre]</p> <p>Affiche, à l'écran, le point de coordonnées (X,Y) relatives à la position du curseur. On peut choisir sa couleur.</p>



## INSTRUCTIONS BASIC

POKE	POKE adr,donnée Inscrit la donnée à l'adresse précisée.
PRINT	PRINT [#n° du canal,] données Imprime les données sur le canal précisé (0 = écran par défaut). PRINT USING permet de spécifier différents formats d'impression.
RAD	Etablit le mode en RADIANS.
RANDOMIZE	RANDOMIZE [N] Fixe la séquence des nombres pseudo-aléatoires à partir de N, N étant un nombre entier compris entre 0 et 65535. Par défaut, N est égal à 0.
READ	READ liste de variables Lit les données contenues dans les lignes de DATAs et les affecte aux différentes variables précisées.
RELEASE	RELEASE canaux sonores Permet de libérer un canal sonore en état d'attente.
REM	Introduit une ligne de commentaires.
RENUM	RENUM [nnl,][anl,][d] Renumérote les lignes du programme présent en mémoire centrale. nnl = nouveau numéro de ligne, vaut 10 par défaut ; anl = ancien numéro de ligne, vaut le numéro de la première ligne de programme par défaut ; d = différence entre deux lignes, vaut 10 par défaut.
RESTORE	RESTORE [numéro de ligne] Définit la ligne à laquelle doit commencer la lecture des DATAs. Si aucun numéro de ligne n'est spécifié, la lecture commence à la première ligne contenant des DATAs.
RESUME	RESUME [numéro de ligne de redépart] Permet à l'exécution du programme de continuer à partir de la ligne de redépart, après qu'une erreur ait été décelée et corrigée par une commande ON ERROR GOTO.
RETURN	Retour au programme principal, après l'appel d'un sous-programme, au moyen de l'instruction GOSUB...



## RUN

RUN [numéro de ligne]

Exécute le programme présent en mémoire centrale à partir de la ligne dont le numéro est spécifié ou, à défaut, à partir de la ligne portant le plus petit numéro.

RUN "nom de programme"

Charge le programme spécifié qui est sur cassette et commence son exécution. Si on ne met pas de nom de programme (RUN ""), le Basic charge et exécute le premier programme rencontré sur la cassette.

## SOUND

SOUND canal,période[,durée[,volume[,enveloppe de volume[,enveloppe de ton]]]]

Produit le son spécifié.

*Canal* : les trois canaux A, B et C peuvent être sélectionnés ensemble, deux à deux ou séparément. Pour plus de compréhension, voir les explications au sujet du CHIPS AY3-8912.

*Période* : prend les valeurs comprises entre 0 et 4095. La fréquence du son s'obtient en divisant 125.000 par la période.

*Durée* : peut prendre les valeurs comprises entre -32768 et +32767. Prend la valeur 20 par défaut. Si la durée a une valeur positive, elle représente un nombre de centièmes de seconde ; si elle a une valeur négative, elle représente le nombre de répétitions d'enveloppes de volume complètes.

*Volume* : prend une valeur comprise entre 0 et 15. Par défaut, il prend la valeur 12 si une commande ENV a été formulée et la valeur 4 sinon.

*Enveloppe de volume* : peut prendre une valeur comprise entre 0 et 15 (0 par défaut) et indique le type d'enveloppe définie par l'instruction ENV.

*Enveloppe de ton* : peut prendre une valeur comprise entre 0 et 15 (0 par défaut) et indique le type de l'enveloppe de ton définie par la commande ENT.

## SPEED INK

SPEEK INK nombre entier, nombre entier

Permet de modifier la vitesse d'alternance des deux couleurs de fond dans le cas où l'on a défini deux couleurs lors de la commande INK :

10 INK 0,1,9

20 SPEED INK 100,20



## INSTRUCTIONS BASIC

SPEED KEY	<p>SPEED KEY attente, période de répétition</p> <p>Permet de régler le temps pendant lequel on doit pousser sur une touche pour qu'elle se répète (attente) et la vitesse de répétition (période de répétition). Ces réglages se font au cinquantième de seconde avec une valeur par défaut de 10,10.</p>
SPEED WRITE	<p>SPEED WRITE n</p> <p>n = 1 ou 0</p> <p>Permet de modifier la vitesse d'enregistrement d'un programme sur cassette. Au moment de la lecture (LOAD), le CPC464 établit automatiquement la vitesse correcte de lecture.</p> <p>Pour n = 0, la vitesse d'écriture sera de 1000 baud ; pour n = 1, elle sera de 2000 baud. Par défaut, n = 0.</p>
STOP	<p>Permet de stopper l'exécution d'un programme tout en laissant à l'utilisateur la possibilité de le continuer au moyen de la commande CONT.</p>
SYMBOL	<p>SYMBOL numéro du caractère, liste de caractères</p> <p>Permet de redéfinir le caractère dont le numéro est indiqué. On peut redéfinir tous les caractères compris entre 240 et 255. Si on désire en redéfinir d'autres, voir la commande SYMBOL AFTER.</p>
SYMBOL AFTER	<p>SYMBOL AFTER nombre entier</p> <p>Fixe le nombre de caractères que l'on désire redéfinir. Vaut 240 en standard.</p>
TAG	<p>TAG [#numéro de canal]</p> <p>Permet d'inscrire des caractères à la position du curseur graphique. Des textes pourront ainsi être mélangés à des graphiques.</p> <pre>10 MOVE 200,300 20 PRINT "COUCOU" 30 TAG 40 PRINT "BONJOUR"</pre> <p>"COUCOU" s'inscrira à la position du curseur de texte tandis que "BONJOUR" s'inscrira à la position du curseur graphique (200,300).</p>
TAGOFF	<p>TAGOFF [numéro de canal]</p> <p>Annule les effets de la commande TAG sur le canal précisé (0 par défaut) et renvoie le texte là où était le curseur de texte avant la commande TAG.</p>



TRON  
TROFF

Validation du mode TRACE  
Dévalidation du mode TRACE  
En mode TRACE, lors de l'exécution d'un programme, tous les numéros de lignes par lesquelles passe l'exécution apparaissent à l'écran. Ce mode est très utile lors de la mise au point d'un programme.

WAIT

WAIT nPORT, octet de masque, octet de sélection  
Permet d'attendre qu'une combinaison déterminée de bits présente sur un port atteigne une valeur déterminée. Cette instruction lit le contenu du port nPORT, lui applique une fonction ET LOGIQUE avec l'octet de masque, puis une fonction OU EXCLUSIF avec l'octet de sélection et ne rend la main au programme que lorsque le résultat est différent de 0. La fonction de masque permet d'isoler le ou les bit(s) à tester. La fonction de sélection permet d'inverser l'état à tester.

WEND

Termine la boucle commencée par la commande WHILE.

WHILE

WHILE expression logique  
WHILE X>2 exécutera les lignes de programme entre WHILE et WEND tant que l'expression logique sera vraie (ici, tant que X sera plus grand que 2).

Le programme suivant :

```
10 X=4:Y=0
20 WHILE X<>Y
30 INPUT "combien font 2 et 2 ";Y
40 WEND
50 PRINT "bravo":END
```

fera exactement la même chose que le programme :

```
10 X=4
20 INPUT "combien font 2 et 2 ";Y
30 IF X<>Y THEN GOTO 20
40 PRINT "bravo":END
```

L'utilité des instructions WHILE et WEND n'est évidente que si l'on pratique la programmation structurée. En effet, un des principes de ce type de programmation est de proscrire les instructions de branchement (GOTO,...) afin de rendre les programmes plus lisibles.

WIDTH

WIDTH nombre entier  
Indique au programme Basic la largeur des lignes de l'imprimante en nombre de caractères.



## INSTRUCTIONS BASIC

WINDOW	WINDOW [#n° de canal,] gauche, droite, haut, bas Permet de définir une fenêtre de texte pour un canal donné de l'écran. Les canaux de 0 à 7 peuvent être utilisés pour définir des fenêtres de texte à l'écran.
WINDOW SWAP	WINDOW SWAP numéro de canal, numéro de canal Permet d'intervertir les contenus de deux fenêtres.
WRITE	WRITE [#n° de canal,][liste à écrire] Inscrit sur le canal précisé (0 par défaut) les expressions voulues sans rien changer à la ponctuation. WRITE "COUCOU", 23, 5 écrira à l'écran : "COUCOU", 23, 5
XPOS	Donne l'abscisse (position horizontale) du curseur graphique.
YPOS	Donne l'ordonnée (position verticale) du curseur graphique.
ZONE	ZONE nombre entier Permet de modifier l'espace compris entre deux tabulations automatiques obtenues par une virgule. Cette tabulation vaut 13 par défaut. ZONE 2:PRINT 1,2,3



ABS	ABS (expression numérique) Donne la valeur absolue de l'expression numérique entre parenthèses.
ASC	ASC (chaîne de caractères) Donne le code ASCII du premier caractère contenu dans la chaîne de caractères entre parenthèses.
ATN	ATN (expression numérique) Donne la valeur en radians ou en degrés de l'angle dont la tangente vaut l'expression numérique.
BIN\$	BIN\$ (nombre entier décimal [,N]) Convertit le nombre entier décimal en un nombre binaire exprimé sur N caractères (8 par défaut).
CHR\$	CHR\$ (N) Donne le caractère dont le code ASCII est N. N est un nombre entier compris entre 0 et 255.
CINT	CINT (expression numérique) Convertit une expression numérique en un nombre entier en arrondissant à l'unité supérieure lorsque la partie décimale de l'expression est supérieure ou égale à 0,5.
COS	COS (valeur d'angle) Donne la valeur du cosinus d'un angle en supposant que celui-ci est exprimé en radians. On peut choisir de l'exprimer en degrés en frappant, au préalable, la commande DEG.
CREAL	CREAL (expression numérique) Convertit un nombre entier en un nombre réel. C'est la fonction inverse de la fonction CINT.
EOF	PRINT EOF Indicateur de fin de fichier cassette. Prend la valeur -1 lorsque l'entrée cassette est à la fin du fichier et la valeur 0 dans tous les autres cas.
ERR	Variable contenant le numéro de la dernière valeur qui s'est produite.
ERL	Variable contenant le numéro de la ligne où la dernière erreur s'est produite.



## FONCTIONS BASIC

EXP	PRINT EXP (n) Calcule e exposant n (exponentielle).
FIX	FIX (n) Enlève la partie décimale du nombre n sans arrondir à l'entier le plus proche.
FRE	FRE (X)    FRE (" ") Fournit le nombre d'octets restés libres en mémoire.
HEX\$	HEX\$ (n) Convertit le nombre entier n en un nombre hexadécimal.
HIMEM	Donne l'adresse la plus haute utilisable par le Basic.
INKEY	INKEY (N) Scrute le clavier pour voir quelle touche a été frappée. Si la touche portant le numéro N a été frappée, INKEY (N) vaut 0 ; si la touche portant le numéro N a été frappée en même temps que la touche SHIFT, INKEY (N) vaut 32 ; et enfin, si aucune touche ou si une autre touche a été frappée, INKEY (N) vaut -1. 5 CLS 10 IF INKEY(54)=32 THEN 30 ELSE IF INKEY(54)=0 THEN 40 20 GOTO 10 30 PRINT"vous avez pressé SHIFT et B": GOTO 50 40 PRINT"vous avez frappé B" 50 GOTO 10
INKEY\$	A\$=INKEY\$ Affecte à la variable alphanumérique A\$ la valeur de la touche qui vient d'être pressée au clavier. Cette fonction est très utile pour attendre une réponse sans devoir enfoncer la touche RC. 10 CLS 20 PRINT"prenez-vous du sucre dans votre café ? "; 30 A\$=INKEY\$: IF A\$="" THEN 30 40 IF A\$ <> "0" AND A\$ <> "N" THEN 30 50 PRINT A\$
INP	PRINT INP (numéro de port d'entrée/sortie) Lit le contenu du port d'entrée/sortie spécifié.



## INSTR

INSTR ([N,]A\$,B\$)

Si la chaîne B\$ est un morceau de A\$, INSTR(A\$,B\$) prend une valeur numérique égale au numéro du caractère de A\$ où commence la chaîne B\$.

PRINT INSTR ("BANALE","AN") donnera : 2.

Si on précise N, la comparaison ne commencera qu'à partir du Nième caractère de la chaîne A\$.

## INT

INT (expression numérique)

Supprime la partie décimale et arrondit au plus petit nombre entier. Identique à FIX pour les nombres positifs, il donnera 1 de moins que FIX pour les nombres négatifs qui ne sont pas entiers.

## JOY

JOY (N)

Lit l'état de la manette de jeux. N indique le numéro de la manette (0 ou 1).

Le résultat de la fonction est exprimé sur 6 bits. Si la manette est au repos, les 6 bits valent 0. Les bits passent à 1 en fonction de la position de la manette ou de la pression sur les boutons de tir.

bit 0 = 1 : manette vers le haut : valeur = 1

bit 1 = 1 : manette vers le bas : valeur = 2

bit 2 = 1 : manette vers la gauche : valeur = 4

bit 3 = 1 : manette vers la droite : valeur = 8

bit 4 = 1 : bouton de tir 1 enfoncé : valeur = 16

bit 5 = 1 : bouton de tir 2 enfoncé : valeur = 32

Une combinaison de plusieurs actions est possible.

Exemple : si on déplace la manette en bas à droite et que l'on enfonce le bouton de tir 1, la fonction JOY fournira une valeur égale à la somme des valeurs qui seraient fournies pour chaque action séparée :

déplacement vers le bas : 2

déplacement vers la droite : 8

bouton de tir 1 enfoncé : 16

valeur fournie : 2+8+16 = 26

## LEFT\$

LEFT\$ (chaîne,N)

Extrait les N premiers caractères à gauche de la chaîne précisée, N étant un nombre entier.

PRINT LEFT\$ ("AMSTRAD",4) donnera : AMST.

## LEN

LEN (chaîne)

Détermine la longueur d'une chaîne de caractères, c'est-à-dire le nombre de caractères qui la constituent.



## FONCTIONS BASIC

LOG	LOG (X) Calcule le logarithme en base e de X.
LOG10	LOG10 (X) Calcule le logarithme en base 10 de X.
LOWER\$	LOWER\$ (chaîne) Transforme, dans la chaîne alphanumérique, toutes les majuscules en minuscules.
MAX	MAX (liste d'expressions numériques) Donne la plus grande valeur contenue dans la liste d'expressions numériques. PRINT MAX (2,67,34,987,12,9,876,0) donnera 987.
MID\$	MID\$ (chaîne,N[,M]) Extrait M caractères de la chaîne en commençant par le Nième caractère. M vaut 1 par défaut.
MIN	MIN (liste d'expressions numériques) Donne la plus petite valeur contenue dans la liste d'expressions numériques.
PEEK	PEEK (n) Lecture de la valeur contenue à l'adresse mémoire n.
PI	PRINT PI donne 3.14159265 Donne la valeur approchée du nombre PI.
POS	POS (#numéro de canal) Indique la position horizontale courante du curseur de texte pour un canal donné (coordonnée X). Dans le cas où le canal imprimante est précisé, POS donne la position du chariot, la position 1 étant la marge à gauche.
REMAIN	REMAIN (N) Supprime le chronomètre indiqué (N=0, 1, 2 ou 3) et lit le temps qui restait. Indique 0 si le chronomètre n'avait pas été mis en route.
RIGHT\$	RIGHT\$ (chaîne,N) Extrait N caractères à droite de la chaîne de caractères précisée.
RND	RND (N) Donne un nombre pseudo aléatoire, Nième d'une séquence déterminée par la commande RANDOMIZE.



ROUND	ROUND (expression numérique [,N]) Arrondit l'expression numérique à N chiffres après la virgule. N, nombre entier, vaut 0 par défaut.
SGN	SGN (expression numérique) Détermine le signe de l'expression numérique. Donne -1 si elle est négative, 0 si elle est nulle et 1 si elle est positive.
SIN	SIN (valeur d'angle) Donne la valeur du sinus d'un angle en supposant que celui-ci est exprimé en radians. On peut choisir de l'exprimer en degrés en frappant, au préalable, la commande DEG.
SPACE\$	SPACE\$ (N) Crée une chaîne de N espaces, N étant un nombre entier.
SQ	SQ (canal sonore) Détermine le nombre de places libres dans une queue pour un canal donné.
SQR	SQR (N) Calcule la racine carrée du nombre N.
STR\$	STR\$ ([&]N) Convertit l'expression numérique N en une chaîne de caractères. Si l'expression numérique est précédée du signe &, elle est considérée comme nombre hexadécimal et sera convertie en nombre décimal avant d'être convertie en chaîne de caractères. PRINT STR\$(123) donnera 123 sous forme de chaîne alphanumérique. PRINT STR\$(&10) donnera 16 sous forme de chaîne alphanumérique.
STRING\$	STRING\$ (N,caractère) PRINT STRING\$ (4,"*") donnera **** Crée une chaîne de caractères composée de N fois le caractère précisé. N peut être exprimé en hexadécimal à condition d'être précédé du signe &.
TAN	TAN (valeur d'angle) Donne la valeur de la tangente d'un angle en supposant que celui-ci est exprimé en radians. On peut choisir de l'exprimer en degrés en frappant au préalable la commande DEG..



## FONCTIONS BASIC

TEST	TEST (x,y) Donne la valeur de l'encre utilisée à l'endroit des coordonnées absolues (x,y) de l'écran.
TESTR	TESTR (x,y) Donne la valeur de l'encre utilisée à l'endroit de coordonnées relatives (x,y) à la position présente du curseur de l'écran.
TIME	Donne le temps écoulé en 1/300e de seconde depuis la mise en route sans compter les temps de lecture et d'écriture sur cassette.
UNT	UNT (nombre) Convertit un nombre entier sans signe en un entier compris entre -32767 et +32768. PRINT UNT(&7FFF) et PRINT UNT(32767) = 32767 PRINT UNT(&0010) et PRINT UNT(16) = 16 PRINT UNT(&0001) et PRINT UNT(1) = 1 PRINT UNT(&FFFF) et PRINT UNT(65535) = -1 PRINT UNT(&FFF6) et PRINT UNT(65526) = -10 PRINT UNT(&8000) et PRINT UNT(32768) = -32768
UPPER\$	UPPER\$ (chaîne) Transforme les minuscules d'une chaîne en majuscules.
VAL	VAL (chaîne) Transforme une chaîne en une expression numérique. Donnera 0 si la chaîne commence par une lettre. PRINT VAL("34E") donnera 34 PRINT VAL("123") donnera 123 PRINT VAL("A34") donnera 0
VPOS	VPOS (#numéro de canal) Donne la position verticale du curseur de texte pour le canal précisé (coordonnée Y).
XPOS	Donne la position horizontale du curseur graphique.
YPOS	Donne la position verticale du curseur graphique.



# MOTS-CLES ET CODES ASSOCIES

Tous les codes inférieurs à 127 sont précédés d'un octet à 255 (FF).

<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>	<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>
255+ 0	FF+ 0	<b>ABS</b>	255+ 27	FF+1B	<b>UNT</b>
255+ 1	FF+ 1	<b>ASC</b>	255+ 28	FF+1C	<b>UPPER\$</b>
255+ 2	FF+ 2	<b>ATN</b>	255+ 29	FF+1D	<b>VAL</b>
255+ 3	FF+ 3	<b>CHR\$</b>	255+ 64	FF+40	<b>EOF</b>
255+ 4	FF+ 4	<b>CINT</b>	255+ 65	FF+41	<b>ERR</b>
255+ 5	FF+ 5	<b>COS</b>	255+ 66	FF+42	<b>HIMEM</b>
255+ 6	FF+ 6	<b>CREAL</b>	255+ 67	FF+43	<b>INKEY\$</b>
255+ 7	FF+ 7	<b>EXP</b>	255+ 68	FF+44	<b>PI</b>
255+ 8	FF+ 8	<b>FIX</b>	255+ 69	FF+45	<b>RND</b>
255+ 9	FF+ 9	<b>FRE</b>	255+ 70	FF+46	<b>TIME</b>
255+ 10	FF+ A	<b>INKEY</b>	255+ 71	FF+47	<b>XPOS</b>
255+ 11	FF+ B	<b>INP</b>	255+ 72	FF+48	<b>YPOS</b>
255+ 12	FF+ C	<b>INT</b>	255+113	FF+71	<b>BIN\$</b>
255+ 13	FF+ D	<b>JOY</b>	255+114	FF+72	<b>DEC\$</b>
255+ 14	FF+ E	<b>LEN</b>	255+115	FF+73	<b>HEX\$</b>
255+ 15	FF+ F	<b>LOG</b>	255+116	FF+74	<b>INSTR</b>
255+ 16	FF+10	<b>LOG10</b>	255+117	FF+75	<b>LEFT\$</b>
255+ 17	FF+11	<b>LOWER\$</b>	255+118	FF+76	<b>MAX</b>
255+ 18	FF+12	<b>PEEK</b>	255+119	FF+77	<b>MIN</b>
255+ 19	FF+13	<b>REMAIN</b>	255+120	FF+78	<b>POS</b>
255+ 20	FF+14	<b>SGN</b>	255+121	FF+79	<b>RIGHT\$</b>
255+ 21	FF+15	<b>SIN</b>	255+122	FF+7A	<b>ROUND</b>
255+ 22	FF+16	<b>SPACE\$</b>	255+123	FF+7B	<b>STRING\$</b>
255+ 23	FF+17	<b>SQ</b>	255+124	FF+7C	<b>TEST</b>
255+ 24	FF+18	<b>SQR</b>	255+125	FF+7D	<b>TESTR</b>
255+ 25	FF+19	<b>STR\$</b>	255+127	FF+7F	<b>VPOS</b>
255+ 26	FF+1A	<b>TAN</b>			

Les codes suivants ne sont plus précédés de 255.

<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>	<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>
128	80	<b>AFTER</b>	138	8A	<b>CLS</b>
129	81	<b>AUTO</b>	139	8B	<b>CONT</b>
130	82	<b>BORDER</b>	140	8C	<b>DATA</b>
131	83	<b>CALL</b>	141	8D	<b>DEF</b>
132	84	<b>CAT</b>	142	8E	<b>DEFINT</b>
133	85	<b>CHAIN</b>	143	8F	<b>DEFREAL</b>
134	86	<b>CLEAR</b>	144	90	<b>DEFSTR</b>
135	87	<b>CLG</b>	145	91	<b>DEG</b>
136	88	<b>CLOSEIN</b>	146	92	<b>DELETE</b>
137	89	<b>CLOSEOUT</b>	147	93	<b>DIM</b>



# MOTS-CLES ET CODES ASSOCIES

<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>	<i>Code Dec</i>	<i>Code Hex</i>	<i>Mot-clé</i>
148	94	DRAW	195	C3	READ
149	95	DRAWR	196	C4	RELEASE
150	96	EDIT	197	C5	REM
151	97	ELSE	198	C6	RENUM
152	98	END	199	C7	RESTORE
153	99	ENT	200	C8	RESUME
154	9A	ENV	201	C9	RETURN
155	9B	ERASE	202	CA	RUN
156	9C	ERROR	203	CB	SAVE
157	9D	EVERY	204	CC	SOUND
158	9E	FOR	205	CD	SPEED
159	9F	GOSUB	206	CE	STOP
160	A0	GOTO	207	CF	SYMBOL
161	A1	IF	208	D0	TAG
162	A2	INK	209	D1	TAGOFF
163	A3	INPUT	210	D2	TROFF
164	A4	KEY	211	D3	TRON
165	A5	LET	212	D4	WAIT
166	A6	LINE	213	D5	WEND
167	A7	LIST	214	D6	WHILE
168	A8	LOAD	215	D7	WIDTH
169	A9	LOCATE	216	D8	WINDOW
170	AA	MEMORY	217	D9	WRITE
171	AB	MERGE	218	DA	ZONE
172	AC	MID\$	219	DB	DI
173	AD	MODE	220	DC	EI
174	AE	MOVE	234	EA	TAB
175	AF	MOVER	235	EB	THEN
176	B0	NEXT	236	EC	TO
177	B1	NEW	237	ED	USING
178	B2	ON	238	EE	>
179	B3	ON BREAK	239	EF	=
180	B4	ON ERROR GOTO	240	F0	>=
181	B5	ON SQ	241	F1	<
182	B6	OPENIN	242	F2	<>
183	B7	OPENOUT	243	F3	<=
184	B8	ORIGIN	244	F4	+
185	B9	OUT	245	F5	-
186	BA	PAPER	246	F6	*
187	BB	PEN	247	F7	/
188	BC	PLOT	248	F8	
189	BD	PLOTR	249	F9	\
190	BE	POKE	250	FA	AND
191	BF	PRINT	251	FB	MOD
192	C0	'	252	FC	OR
193	C1	RAD	253	FD	XOR
194	C2	RANDOMIZE	254	FE	NOT



# CODES ASCII ET GRAPHIQUES

Caractère	Code ASCII		
	Hexadécimal	Décimal	Octal
NUL (CTRL @)	00	0	000
SOH (CTRL A)	01	1	001
STX (CTRL B)	02	2	002
ETX (CTRL C)	03	3	003
EOT (CTRL D)	04	4	004
ENQ (CTRL E)	05	5	005
ACK (CTRL F)	06	6	006
BEL (CTRL G)	07	7	007
BS (CTRL H)	08	8	010
HT (CTRL I)	09	9	011
LF (CTRL J)	0A	10	012
VT (CTRL K)	0B	11	013
FF (CTRL L)	0C	12	014
CR (CTRL M)	0D	13	015
SO (CTRL N)	0E	14	016
SI (CTRL O)	0F	15	017
DLE (CTRL P)	10	16	020
DC1 (CTRL Q)	11	17	021
DC2 (CTRL R)	12	18	022
DC3 (CTRL S)	13	19	023
DC4 (CTRL T)	14	20	024
NAK (CTRL U)	15	21	025
SYN (CTRL V)	16	22	026
ETB (CTRL W)	17	23	027
CAN (CTRL X)	18	24	030
EM (CTRL Y)	19	25	031
SUB (CTRL Z)	1A	26	032
ESC	1B	27	033
FS	1C	28	034
GS	1D	29	035
RS	1E	30	036
US	1F	31	037
SP	20	32	040
!	21	33	041
"	22	34	042
#	23	35	043
\$	24	36	044
%	25	37	045
&	26	38	046
'	27	39	047
(	28	40	050
)	29	41	051
*	2A	42	052
+	2B	43	053
,	2C	44	054
-	2D	45	055



# CODES ASCII ET GRAPHIQUES

Caractère	Code ASCII		
	Hexadécimal	Décimal	Octal
.	2E	46	056
/	2F	47	057
0	30	48	060
1	31	49	061
2	32	50	062
3	33	51	063
4	34	52	064
5	35	53	065
6	36	54	066
7	37	55	067
8	38	56	070
9	39	57	071
:	3A	58	072
;	3B	59	073
<	3C	60	074
=	3D	61	075
>	3E	62	076
?	3F	63	077
@	40	64	100
A	41	65	101
B	42	66	102
C	43	67	103
D	44	68	104
E	45	69	105
F	46	70	106
G	47	71	107
H	48	72	110
I	49	73	111
J	4A	74	112
K	4B	75	113
L	4C	76	114
M	4D	77	115
N	4E	78	116
O	4F	79	117
P	50	80	120
Q	51	81	121
R	52	82	122
S	53	83	123
T	54	84	124
U	55	85	125
V	56	86	126
W	57	87	127
X	58	88	130
Y	59	89	131
Z	5A	90	132
[	5B	91	133



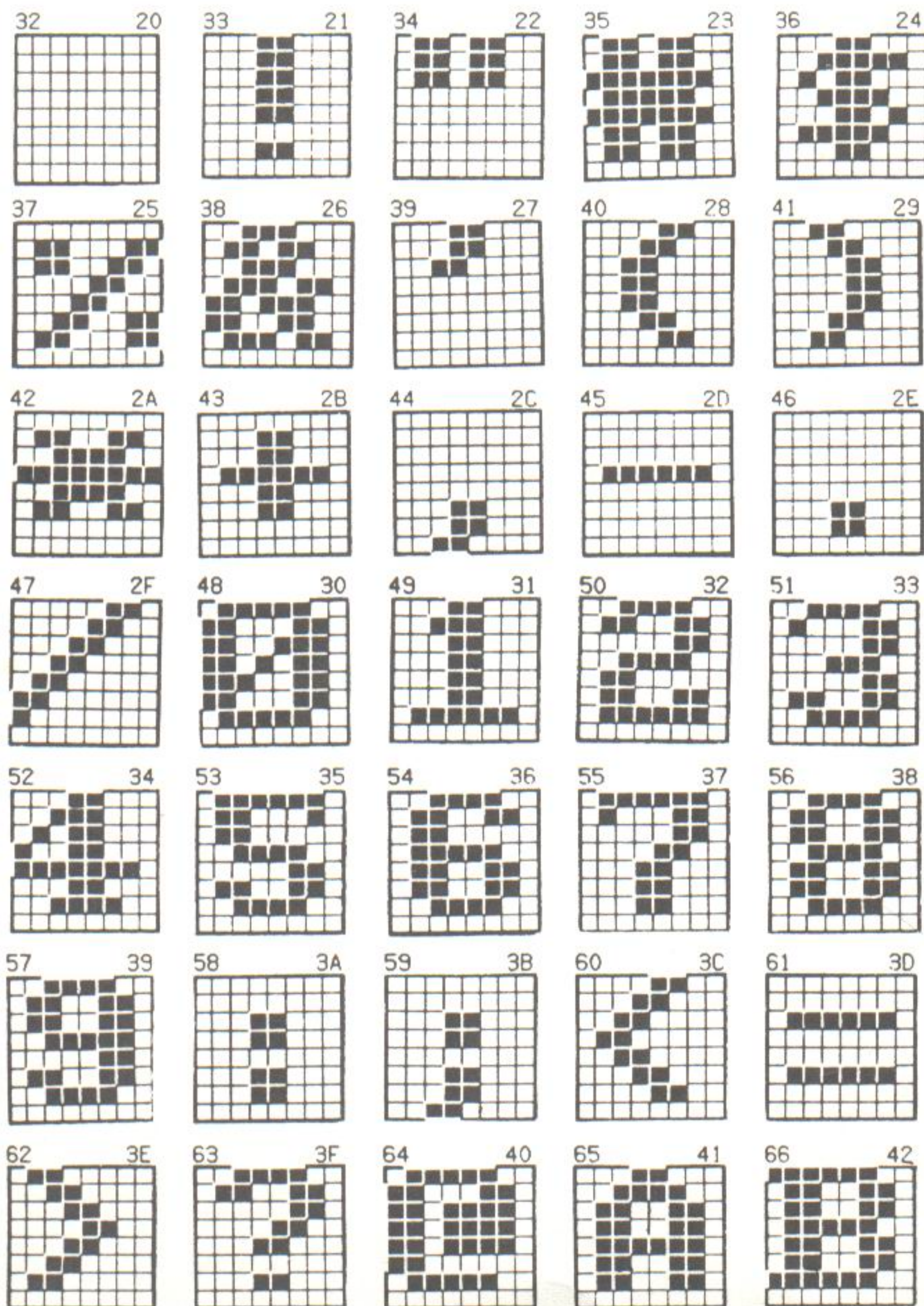
# CODES ASCII ET GRAPHIQUES

Caractère	Code ASCII		
	Hexadécimal	Décimal	Octal
\	5C	92	134
]	5D	93	135
†	5E	94	136
	5F	95	137
è	60	96	140
a	61	97	141
b	62	98	142
c	63	99	143
d	64	100	144
e	65	101	145
f	66	102	146
g	67	103	147
h	68	104	150
i	69	105	151
j	6A	106	152
k	6B	107	153
l	6C	108	154
m	6D	109	155
n	6E	110	156
o	6F	111	157
p	70	112	160
q	71	113	161
r	72	114	162
s	73	115	163
t	74	116	164
u	75	117	165
v	76	118	166
w	77	119	167
x	78	120	170
y	79	121	171
z	7A	122	172
{	7B	123	173
	7C	124	174
}	7D	125	175
~	7E	126	176



## CODES ASCII ET GRAPHIQUES

Le jeu de caractères du CPC464 représentés sur la matrice de 8x8 utilisée pour l'affichage sur l'écran. La commande **SYMBOL** permet de redéfinir de nouveaux caractères.





67	43	68	44	69	45	70	46	71	47
72	48	73	49	74	4A	75	4B	76	4C
77	4D	78	4E	79	4F	80	50	81	51
82	52	83	53	84	54	85	55	86	56
87	57	88	58	89	59	90	5A	91	5B
92	5C	93	5D	94	5E	95	5F	96	60
97	61	98	62	99	63	100	64	101	65



# CODES ASCII ET GRAPHIQUES



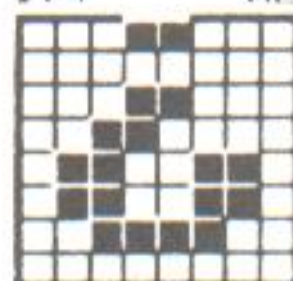

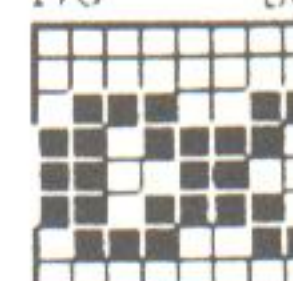
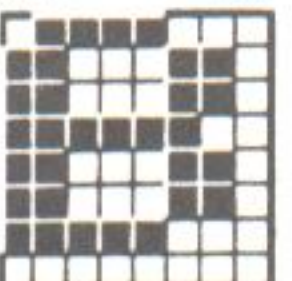
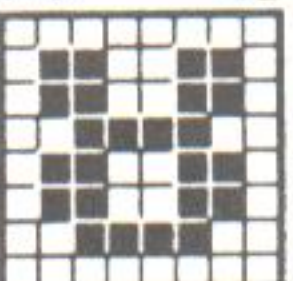
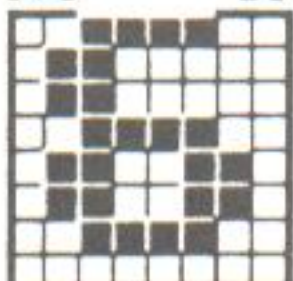
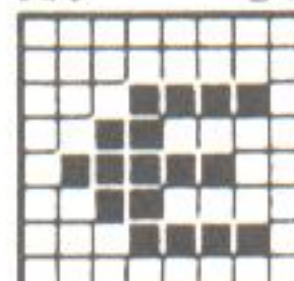
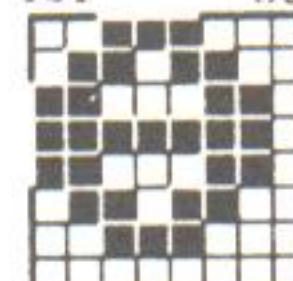
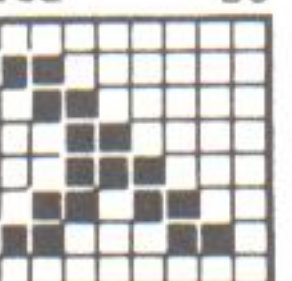
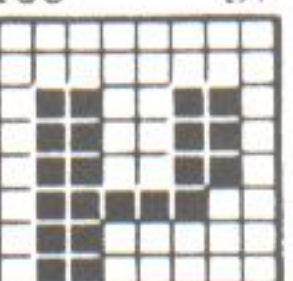
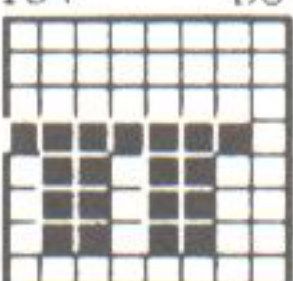
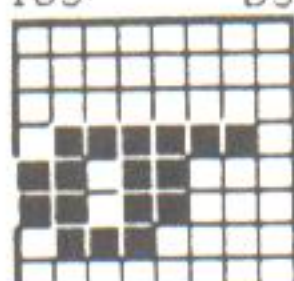

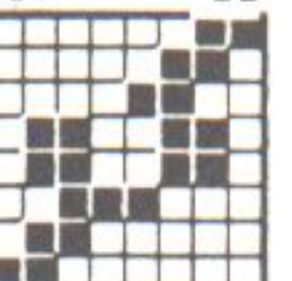

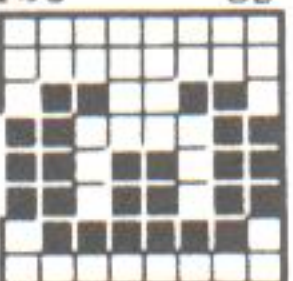

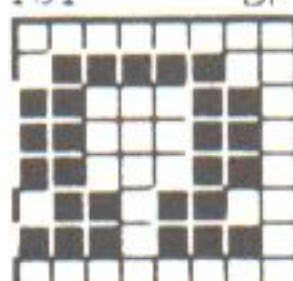

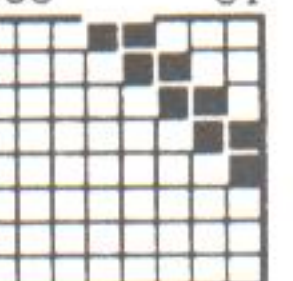

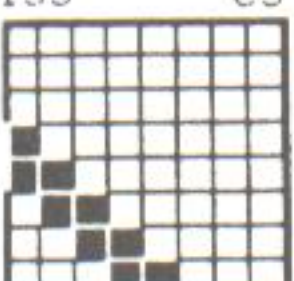


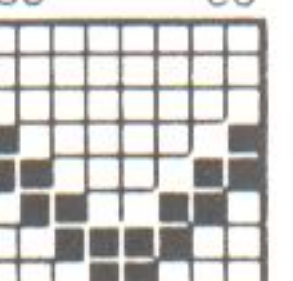
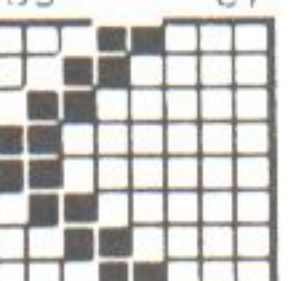
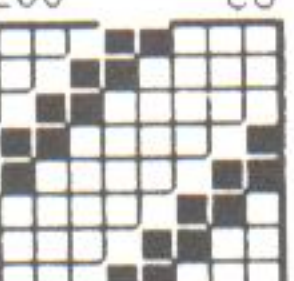
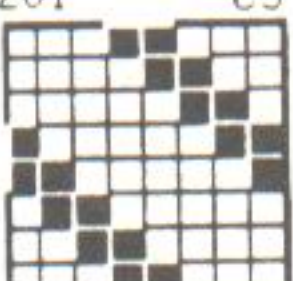
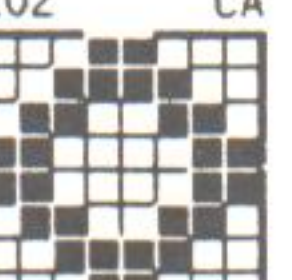
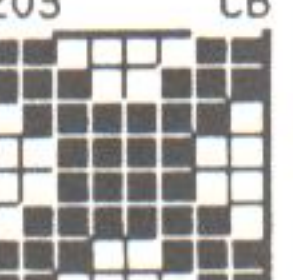
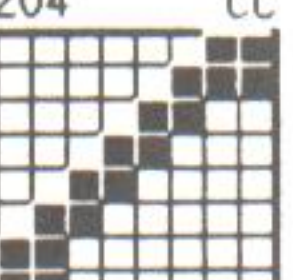

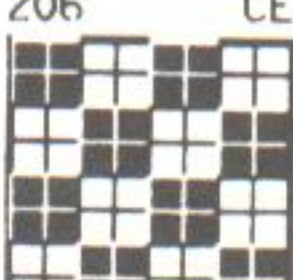
102 66	103 67	104 68	105 69	106 6A
107 6B	108 6C	109 6D	110 6E	111 6F
112 70	113 71	114 72	115 73	116 74
117 75	118 76	119 77	120 78	121 79
122 7A	123 7B	124 7C	125 7D	126 7E
127 7F	128 80	129 81	130 82	131 83
132 84	133 85	134 86	135 87	136 88



137 89 	138 8A 	139 8B 	140 8C 	141 8D 
142 8E 	143 8F 	144 90 	145 91 	146 92 
147 93 	148 94 	149 95 	150 96 	151 97 
152 98 	153 99 	154 9A 	155 9B 	156 9C 
157 9D 	158 9E 	159 9F 	160 A0 	161 A1 
162 A2 	163 A3 	164 A4 	165 A5 	166 A6 
167 A7 	168 A8 	169 A9 	170 AA 	171 AB 



# CODES ASCII ET GRAPHIQUES

172 AC 	173 AD 	174 AE 	175 AF 	176 B0 
177 B1 	178 B2 	179 B3 	180 B4 	181 B5 
182 B6 	183 B7 	184 B8 	185 B9 	186 BA 
187 BB 	188 BC 	189 BD 	190 BE 	191 BF 
192 C0 	193 C1 	194 C2 	195 C3 	196 C4 
197 C5 	198 C6 	199 C7 	200 C8 	201 C9 
202 CA 	203 CB 	204 CC 	205 CD 	206 CE 



207 CF	208 D0	209 D1	210 D2	211 D3
212 D4	213 D5	214 D6	215 D7	216 D8
217 D9	218 DA	219 DB	220 DC	221 DD
222 DE	223 DF	224 E0	225 E1	226 E2
227 E3	228 E4	229 E5	230 E6	231 E7
232 E8	233 E9	234 EA	235 EB	236 EC
237 ED	238 EE	239 EF	240 F0	241 F1



# CODES ASCII ET GRAPHIQUES

242 F2	243 F3	244 F4	245 F5	246 F6
247 F7	248 F8	249 F9	250 FA	251 FB
252 FC	253 FD	254 FE	255 FF	



- 1 - **Unexpected NEXT**  
NEXT inattendu. Une commande NEXT a été rencontrée sans qu'une commande FOR correspondante n'ait été exécutée.
- 2 - **Syntax error**  
Erreur de syntaxe. Le Basic ne peut comprendre la ligne.
- 3 - **Unexpected RETURN**  
Une commande RETURN est rencontrée alors qu'il n'y a pas eu de commande GOSUB.
- 4 - **DATA exhausted**  
Une commande READ a essayé de lire une ligne de DATA qui est déjà finie.
- 5 - **Improper argument**  
Le paramètre d'une commande, ou la valeur d'une fonction, n'est pas exprimé correctement.
- 6 - **Overflow**  
Débordement. Une valeur introduite ou calculée est trop grande ou trop petite pour être manipulée par l'ordinateur.
- 7 - **Memory full**  
Mémoire remplie. Toute la mémoire disponible a été utilisée ou protégée. Cela peut survenir dans le cas de tableaux très grands, de boucles FOR-NEXT ou d'appels GOSUB emboîtés.
- 8 - **Line does not exist**  
Le numéro de ligne demandé n'existe pas en mémoire.
- 9 - **Subscript out of range**  
Un indice d'un arrangement (matrice) est hors limites (trop grand ou trop petit).
- 10 - **Array already dimensionned**  
Un arrangement défini par DIM a déjà été défini.
- 11 - **Division by zero**  
On a essayé de diviser un nombre par zéro.
- 12 - **Invalid direct command**  
La commande tapée n'est pas acceptable en mode direct.
- 13 - **Type mismatch**  
Discordance de type. Tentative d'assignation d'une valeur alphanumérique à une variable numérique ou vice-versa.
- 14 - **String space full**  
L'espace réservé aux chaînes déborde.



## CODES ET MESSAGES D'ERREURS

- 15 - **String too long**  
Une chaîne de caractères comprend plus de 255 caractères.
- 16 - **String expression too complex**  
Expression alphanumérique trop complexe pour être manipulée par l'ordinateur.
- 17 - **Cannot CONTinue**  
L'exécution du programme ne peut reprendre suite à la commande CONT. Cela se produit si, après un arrêt (ESC ESC), on a modifié le programme.
- 18 - **Unknown user fonction**  
On a oublié de définir la fonction FN avec la commande DEF FN.
- 19 - **RESUME missing**  
La commande RESUME n'a pas été rencontrée dans un sous-programme appelé par la commande ON ERROR GOTO.
- 20 - **Unexpected RESUME**  
Une commande RESUME est rencontrée en dehors d'un sous-programme appelé par la commande ON ERROR GOTO.
- 21 - **Direct command found**  
En chargeant un programme d'une cassette, une ligne sans numéro de ligne a été rencontrée.
- 22 - **Operand missing**  
Le signe d'opération est absent dans l'expression rencontrée.
- 23 - **Line too long**  
Ligne trop longue. Le Basic n'accepte pas les lignes d'une longueur supérieure à 255 caractères.
- 24 - **EOF met**  
Le programme est arrivé en fin de fichier sur cassette.
- 25 - **File type error**  
Le fichier sur cassette n'est pas du type requis.
- 26 - **NEXT missing**  
Le NEXT correspondant à la commande FOR n'a pas été trouvé.
- 27 - **File already open**  
On a essayé d'ouvrir un fichier déjà ouvert.
- 28 - **Unknown command**  
Commande externe inconnue.



- 29 - **WEND missing**  
Le WEND correspondant à la commande WHILE est absent du programme.
- 30 - **Unexpected WEND**  
Un WEND sans WHILE préalable a été rencontré.
- 31 et suivantes - **Unknown error**  
Erreur de type inconnu.



## FORMAT DE STOCKAGE D'UNE LIGNE BASIC EN MEMOIRE

L'ordinateur ne comprend que des codes binaires. L'interpréteur Basic se charge de traduire vos programmes en langage binaire pour autant que ceux-ci soient écrits correctement.

### Stockage des instructions Basic

A chaque instruction Basic, l'interpréteur affecte un code encore appelé TOKEN. Ce système permet un gain de place considérable en mémoire centrale car il est évident qu'un octet prend moins de place qu'un mot entier. C'est pourquoi, il ne faut jamais utiliser les noms des instructions Basic pour définir des variables. En effet, l'interpréteur les remplacera par le TOKEN de l'instruction correspondante.

### Stockage d'une ligne de Basic

Le Basic commence à stocker ses lignes à partir de l'adresse 368. Voyons, à l'aide d'un exemple, comment il stocke une ligne.

Encodez le petit programme suivant :

```
1990 PRINT "COUCOU"
```

Ensuite, tapez en direct la suite d'instructions suivantes :

```
FOR I=368 TO 390:PRINT I;" ";PEEK(I):NEXT I
```

Voici la suite de nombres qui apparaît ainsi que leur signification.

Adresse	Valeur	Signification
368	15	Octet faible de la longueur de ligne.
369	0	Octet fort de la longueur de ligne.
370	198	Octet faible du numéro de ligne.
371	7	Octet fort du numéro de ligne.
372	191	Code de l'instruction PRINT.
373	32	Code ASCII de l'espace.
374	34	Code ASCII de ".
375	67	Code ASCII de C.
376	79	Code ASCII de O.
377	85	Code ASCII de U.
378	67	Code ASCII de C.
379	79	Code ASCII de O.
380	85	Code ASCII de U.
381	34	Code ASCII de ".
382	0	Code de séparation.



## FORMAT DE STOCKAGE D'UNE LIGNE BASIC EN MEMOIRE

La longueur de la ligne est exprimée sur deux octets. Elle s'obtient au moyen de la formule :

$$\text{OCTET FAIBLE} + (256 \times \text{OCTET FORT})$$

Donc, la longueur de la ligne vaut :  $15 + (0 \times 256) = 15$ . Nous constatons que la ligne occupe effectivement 15 places mémoire.

Le numéro de ligne est également exprimé sur deux octets. Il s'obtient au moyen de la même formule que pour la longueur. Il vaut donc :

$$198 + (256 \times 7) = 1990$$

Pour remplacer le PRINT par REM, il suffit d'encoder directement : POKE 372,197, puis de faire LIST, et on obtient :

```
1990 REM "COUCOU"
```

Dès lors, vous pouvez à votre guise modifier vos programmes, ou mieux, les faire se modifier aux-mêmes en introduisant les lignes de POKE à l'intérieur du programme lui-même... Bon amusement !

Voyons maintenant comment sont stockées les variables. Encodez le petit programme suivant :

```
10 ABC=20
```

Ensuite, tapez comme précédemment :

```
FOR I=368 TO 400:PRINT I;" ";PEEK(I):NEXT I
```

Vous voyez apparaître :

Adresse	Valeur	Signification
368	14	Octet faible de la longueur de ligne.
369	0	Octet fort de la longueur de ligne.
370	10	Octet faible du numéro de ligne.
371	0	Octet fort du numéro de ligne.
372	13	Indique une variable numérique.
373	7	Longueur du nom de la variable +4.
374	0	Séparation.
375	65	Code ASCII du premier caractère du nom de la variable.
376	66	Code ASCII du deuxième caractère du nom de la variable.
377	195	128 + code ASCII du dernier caractère du nom de la variable.
378	239	TOKEN du signe =.



## FORMAT DE STOCKAGE D'UNE LIGNE BASIC EN MEMOIRE

Adresse	Valeur	Signification
379	25	Grandeur de la variable.
380	20	Valeur de la variable.
381	0	Séparateur.

La valeur **13** en **372** est un code indiquant qu'il s'agit d'une variable numérique. Pour une variable alphanumérique, nous y trouverions la valeur 3.

De **375** à **377**, nous trouvons codé le nom de la variable. Tous les caractères sont codés en ASCII, sauf le dernier qui est représenté par son code ASCII augmenté de 128.

En **376**, la valeur **239** représente le TOKEN du signe =. Le TOKEN du signe = est différent de son code ASCII. Ainsi, l'ordinateur sait que le signe = ne fait pas partie du nom de la variable.

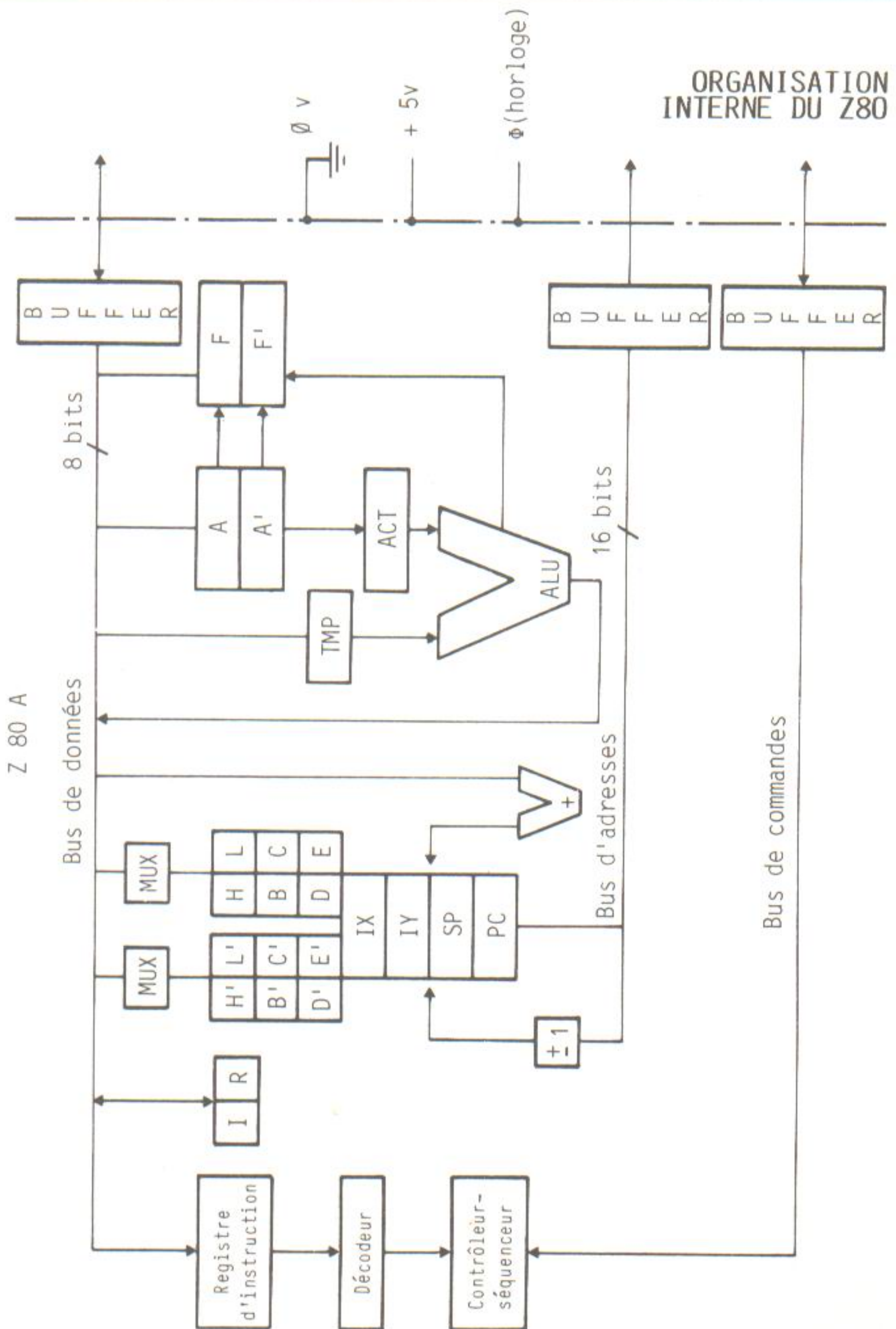
La valeur **25**, qui se trouve en **379**, indique la grandeur de la variable. Voici les différentes valeurs que l'on peut trouver à cet endroit :

Valeur	Longueur de la variable
15	Valeur de la variable = 1, n'est pas codée.
16	Valeur de la variable = 2, n'est pas codée.
..	.. ..
23	Valeur de la variable = 9, n'est pas codée.
25	Valeur de la variable comprise entre 10 et 255, codée sur un octet.
26	Valeur de la variable comprise entre 255 et 65535, codée sur deux octets.
31	Valeur de la variable supérieure à 65535 ou non entière, codée sur cinq octets suivant la formule suivante : $\text{VALEUR} = (2^{(o5 - 145)} * (65536 + (o2/128) + (o3 * 2) + (o4 * 512) + (o1/32800)))$ où o1, ..., o5 représentent les valeurs présentes aux cinq adresses servant à coder la variable

Dans le cas où la variable est un nombre négatif, le TOKEN du signe = (239) est suivi du TOKEN du signe - (245).



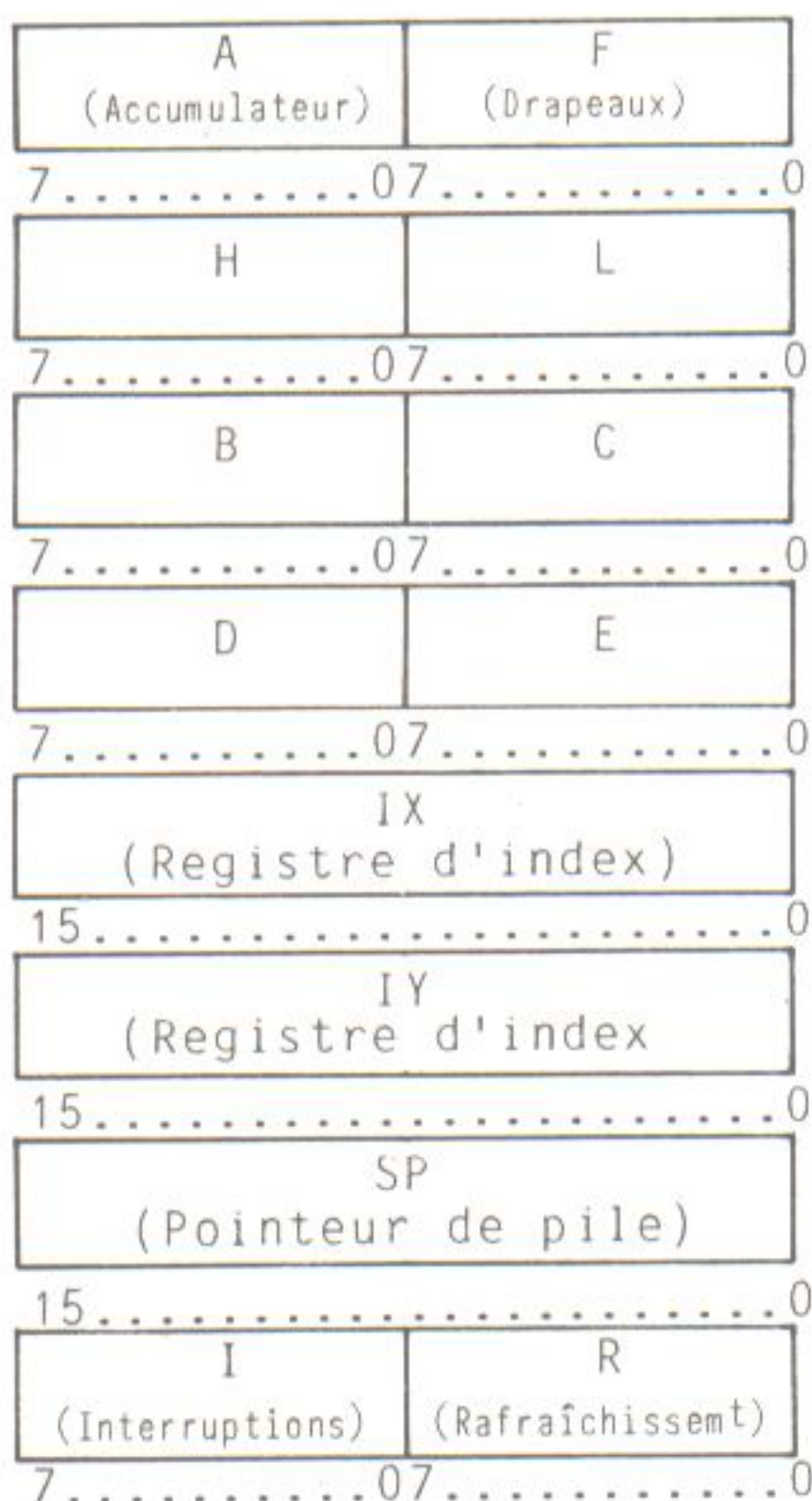
# LANGAGE MACHINE



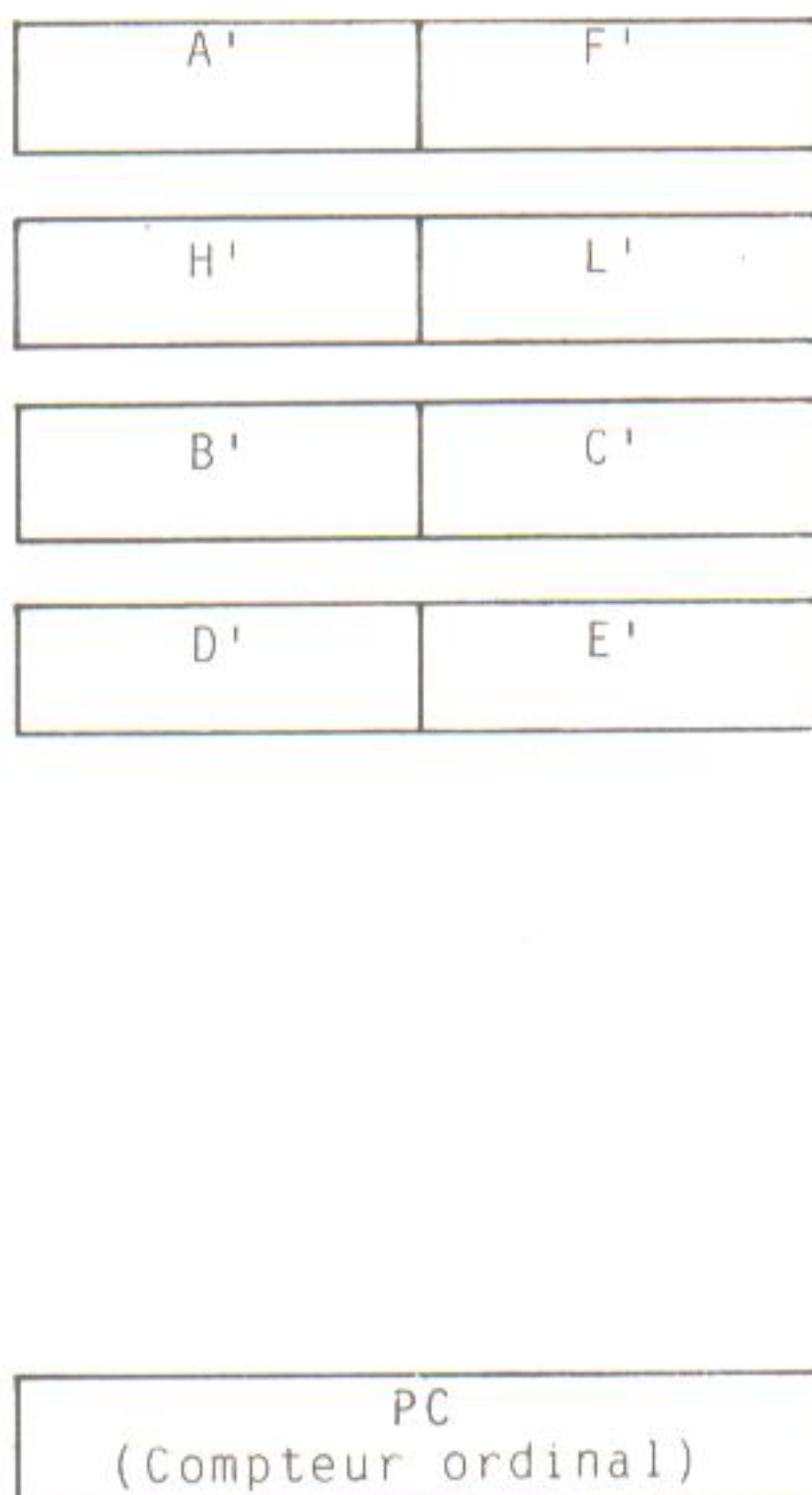


## REGISTRE DU Z80

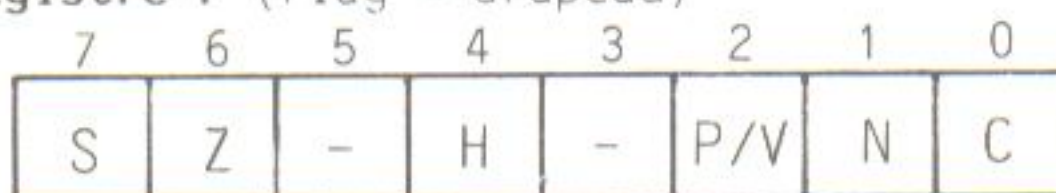
### Registres primaires



### Registres secondaires



### Détails du registre F (flag = drapeau)



- S = Signe** : passe à 1 si le bit le plus significatif du résultat d'une opération est 1.
- Z = Zéro** : passe à 1 si le résultat de l'opération est nul.
- H = Demi** : identique à C, mais pour les opérations sur des demi-octets.
- P/V = Parité/dépassement** : P=1 s'il y a un nombre pair de bits à 1, ou V=1 s'il y a dépassement de capacité après une opération avec des nombres signés.
- N = Opération précédente** : N=1 si l'opération précédente était une soustraction, et N=0 pour une addition.
- C = Retenue** : passe à 1 si le résultat nécessite une retenue (soustraction) ou un report (addition).

*Remarque* : les drapeaux H et N ne peuvent pas être testés.



<i>Mnémonique</i>	<i>Opération effectuée</i>
ADC	Addition avec retenue.
ADD	Addition sans retenue.
BIT	Teste un bit particulier d'un octet.
CALL cc,nn	Appel conditionnel d'un sous-programme.
CALL	Appel inconditionnel d'un sous-programme.
CCF	Complémente l'indicateur de retenue.
CP	Compare l'opérande et l'accumulateur.
CPD	Compare le contenu de l'adresse pointée par HL. Décrémente HL et BC.
CPDR	Compare le contenu de l'adresse pointée par HL. Décrémente HL et BC. Répète la séquence jusqu'à ce que BC=0.
CPI	Compare le contenu de l'adresse pointée par HL. Incrémente HL et décrémente BC.
CPIR	Compare le contenu de l'adresse pointée par HL. Incrémente HL et décrémente BC. Répète la séquence jusqu'à ce que BC=0.
CPL	Complémente l'accumulateur.
DAA	Ajustement décimal de l'accumulateur.
DEC	Décrémente un registre, une paire de registres ou une adresse pointée par HL.
DI	Désactive les interruptions.
DJNZ	Décrémente B et effectue un saut relatif si B≠0.
EI	Active les interruptions.
EX	Echange les contenus des registres.
EXX	Echange le contenu des registres BC, DE et HL avec les registres BC', DE' et HL'.
HALT	Place le microprocesseur en position d'attente d'une interruption ou d'un reset.
IM	Positionne un des trois modes d'interruption (de 0 à 2).
IN	Charge l'accumulateur ou un registre avec le contenu d'un port d'entrées/sorties.
INC	Incrémente un registre, une paire de registres ou le contenu de l'adresse pointée par HL.

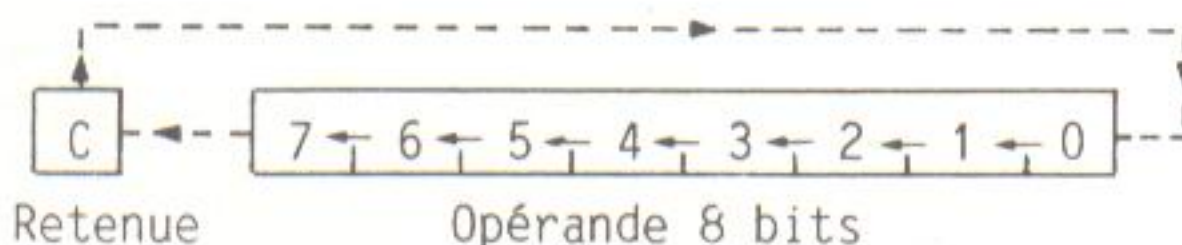


# JEU D'INSTRUCTIONS DU Z80

<i>Mnémonique</i>	<i>Opération effectuée</i>
<b>IND</b>	Charge l'adresse pointée par HL avec le contenu du port d'entrées/sorties pointé par C, et décrémente HL et B.
<b>INDR</b>	Charge l'adresse pointée par HL avec le contenu du port d'entrées/sorties pointé par C, et décrémente HL et B. Répète la séquence jusqu'à ce que B=0.
<b>INI</b>	Charge l'adresse pointée par HL avec le contenu du port d'entrées/sorties pointé par C, et incrémente HL et décrémente B.
<b>INIR</b>	Charge l'adresse pointée par HL avec le contenu du port d'entrées/sorties pointé par C, et incrémente HL et décrémente B. Répète la séquence jusqu'à ce que B=0.
<b>JP</b>	Saut inconditionnel à l'adresse donnée ou à celle pointée par HL, IX et IY.
<b>JP cc,aa</b>	Saut conditionnel (cc) à l'adresse donnée (aa).
<b>JR e</b>	Saut inconditionnel relatif à PC plus déplacement (e).
<b>JR cc,e</b>	Saut conditionnel (cc) relatif à PC plus déplacement (e).
<b>LD</b>	Charge l'accumulateur, un registre ou une adresse avec le contenu de l'accumulateur, d'un registre ou d'une adresse.
<b>LDD</b>	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, puis décrémente DE, HL et BC.
<b>LDDR</b>	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, puis décrémente DE, HL et BC. Répète la séquence jusqu'à ce que BC=0.
<b>LDI</b>	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, puis incrémente DE et HL et décrémente BC.
<b>LDIR</b>	Charge l'adresse pointée par HL avec le contenu de l'adresse pointée par DE, puis incrémente DE et HL et décrémente BC. Répète la séquence jusqu'à ce que BC=0.
<b>NEG</b>	Inverse le signe de l'accumulateur.
<b>NOP</b>	Le Z80 n'effectue pas d'instruction.

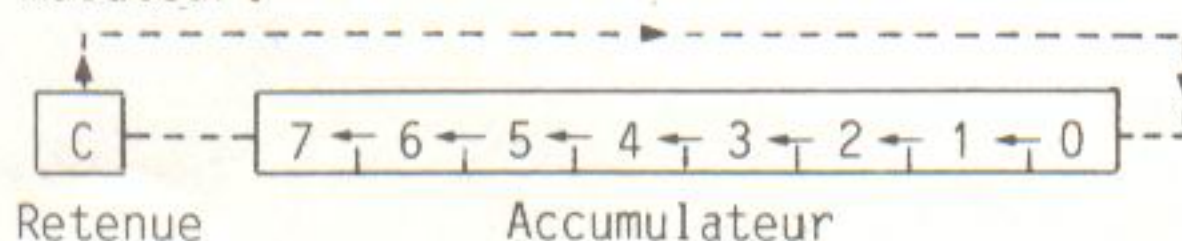


Mnémonique	Opération effectuée
OR	"Ou" logique entre l'opérande et l'accumulateur.
OTDR	Charge le port d'entrées/sorties pointé par C avec le contenu de l'adresse pointée par HL, puis décrémente HL et B. Répète la séquence jusqu'à ce que B=0.
OTIR	Charge le port d'entrées/sorties pointé par C avec le contenu de l'adresse pointée par HL, puis incrémente HL et décrémente B. Répète la séquence jusqu'à ce que B=0.
OUT	Charge le port d'entrées/sorties spécifié avec le contenu de l'accumulateur.
OUTD	Charge le port d'entrées/sorties pointé par C avec le contenu de l'adresse pointée par HL, puis décrémente HL et B.
OUTI	Charge le port d'entrées/sorties pointé par C avec le contenu de l'adresse pointée par HL, puis incrémente HL et décrémente B.
POP	Charge une paire de registres ou un index avec la dernière valeur de la pile (pointée par SP).
PUSH	Place le contenu d'une paire de registres ou d'un index dans la pile (pointée par SP).
RES	Place 0 dans le bit spécifié de l'opérande.
RET	Retour d'un sous-programme.
RETI	Retour d'un sous-programme d'interruption.
RETN	Retour d'un sous-programme d'interruption non masquable.
RL	Rotation à gauche à travers la retenue de l'opérande.




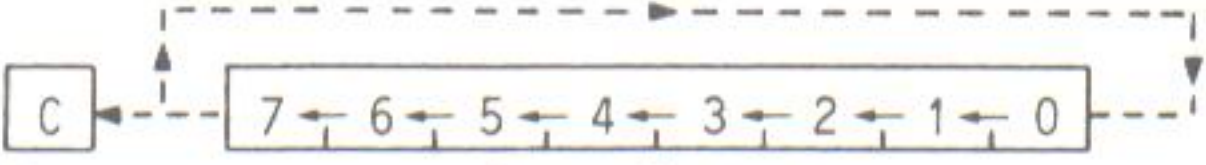
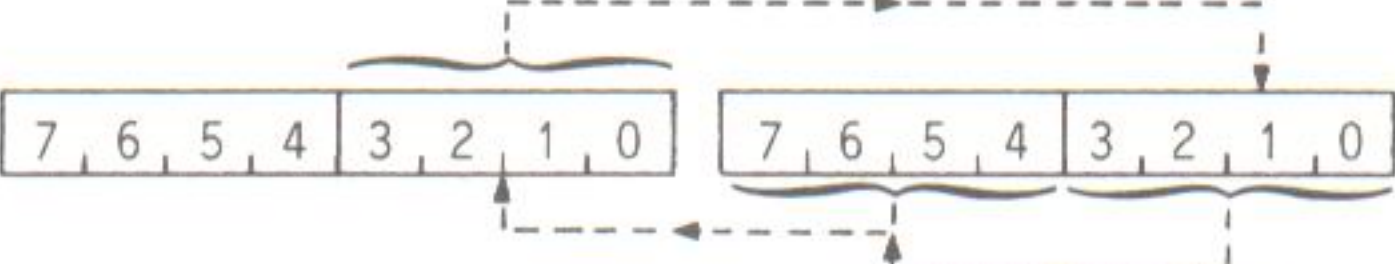
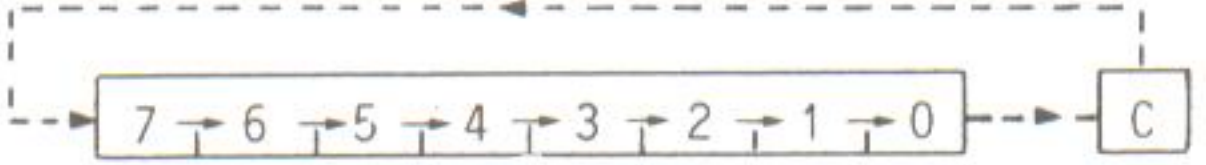
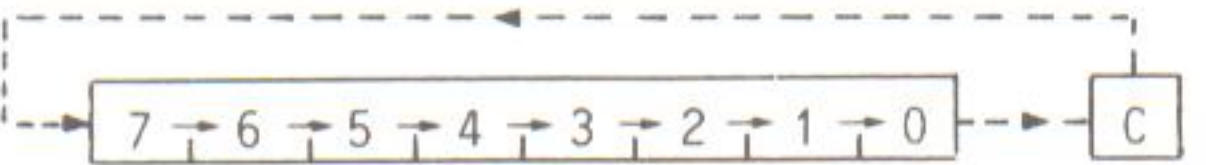
RLA

Rotation à gauche à travers la retenue de l'accumulateur.

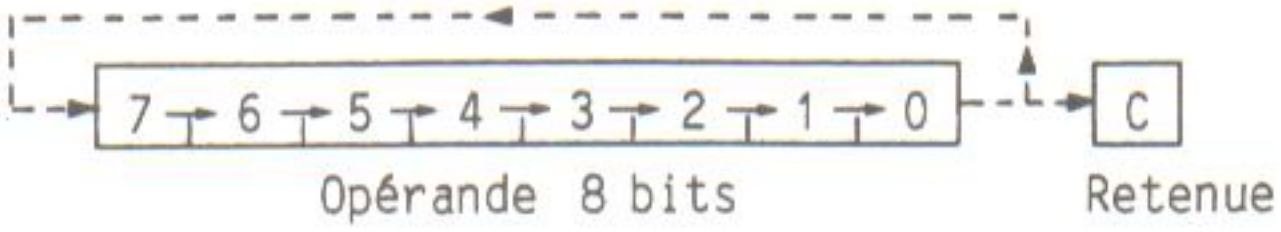
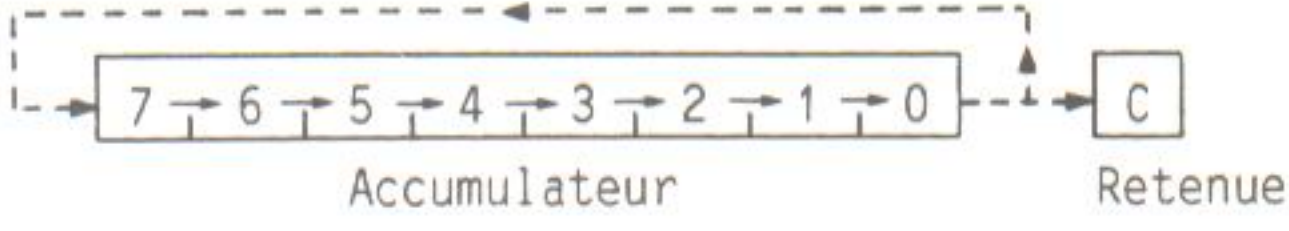
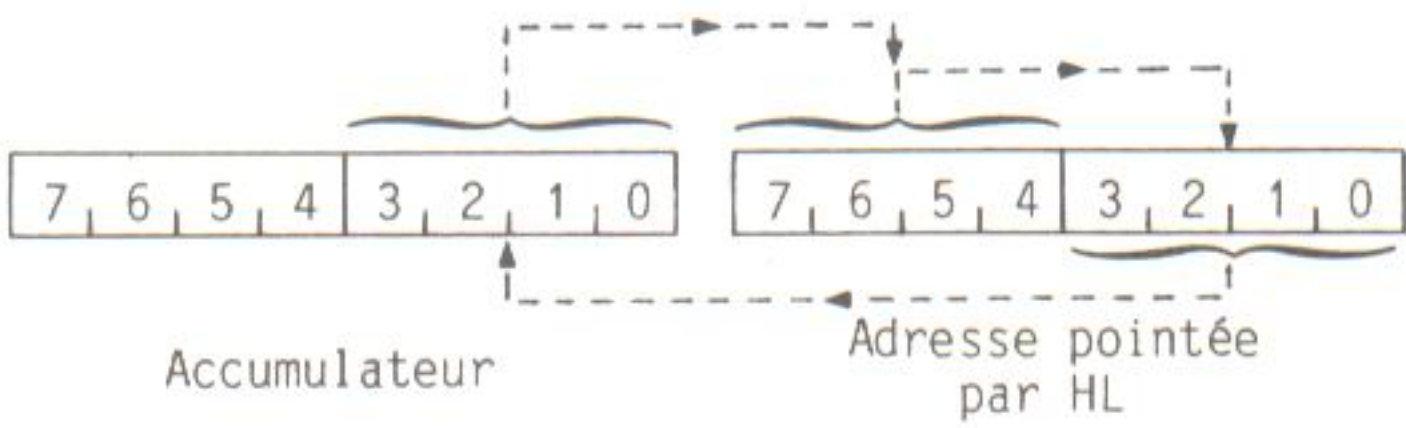
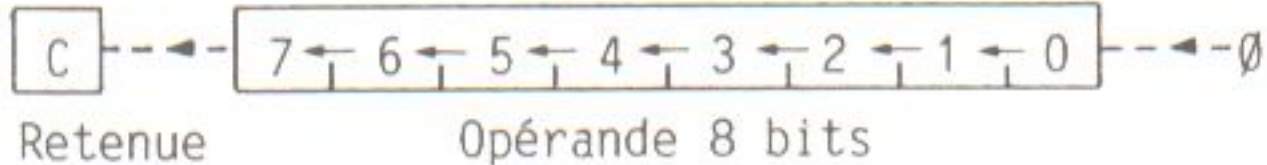
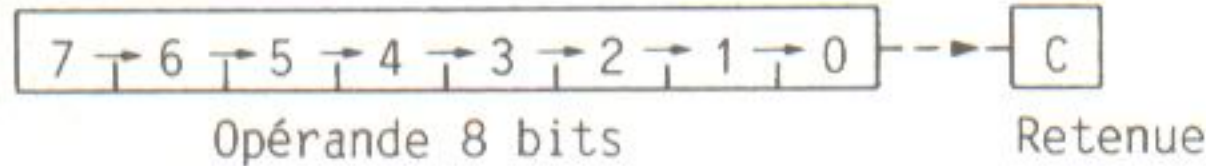




# JEU D'INSTRUCTIONS DU Z80

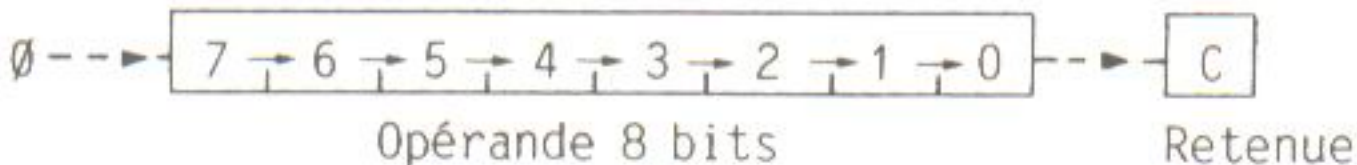
Mnémonique	Opération effectuée
RLC	<p>Rotation circulaire à gauche du contenu d'un registre ou d'une adresse pointée par HL ou par IX et IY plus déplacement.</p>  <p>Retenue                      Opérande 8 bits</p>
RLCA	<p>Rotation circulaire à gauche de l'accumulateur.</p>  <p>Retenue                      Accumulateur</p>
RLD	<p>Rotation BCD (binaire codé décimal) de 4 bits à gauche entre l'accumulateur et l'adresse pointée par HL.</p>  <p>Accumulateur                      Adresse pointée par HL</p>
RR	<p>Rotation à droite à travers la retenue de l'opérande.</p>  <p>Opérande 8 bits                      Retenue</p>
RRA	<p>Rotation à droite à travers la retenue de l'accumulateur.</p>  <p>Accumulateur                      Retenue</p>



Mnémonique	Opération effectuée
RRC	<p>Rotation circulaire à droite de l'opérande.</p>  <p>Opérande 8 bits      Retenue</p>
RRCA	<p>Rotation circulaire à droite de l'accumulateur.</p>  <p>Accumulateur      Retenue</p>
RRD	<p>Rotation BCD à droite de 4 bits entre l'accumulateur et le contenu de l'adresse pointée par HL.</p>  <p>Accumulateur      Adresse pointée par HL</p>
RST	Saut à une adresse sur un octet.
SBC	Soustraction avec retenue entre l'accumulateur et l'opérande ou HL et une paire de registres.
SCF	Met à 1 le drapeau de retenue.
SET	Met à 1 un bit particulier d'un registre ou d'une adresse pointée par HL ou IX et IY plus déplacement.
SLA	<p>Décalage arithmétique à gauche de l'opérande.</p>  <p>Retenue      Opérande 8 bits</p>
SRA	<p>Décalage arithmétique à droite de l'opérande.</p>  <p>Opérande 8 bits      Retenue</p> <p>NB : le bit 7 reste inchangé.</p>



# JEU D'INSTRUCTIONS DU Z80

<i>Mnémonique</i>	<i>Opération effectuée</i>
<b>SRL</b>	<p>Décalage logique à droite de l'opérande.</p>  <p style="text-align: center;">Opérande 8 bits <span style="float: right;">Retenue</span></p>
<b>SUB</b>	Soustraction à l'accumulateur de l'opérande.
<b>XOR</b>	"Ou" exclusif entre l'opérande et l'accumulateur.



# CODES DES INSTRUCTIONS Z80 PAR ORDRE ALPHABETIQUE

- d = donnée sur 8 bits
- dd = donnée sur 16 bits
- aa = adresse sur 16 bits

- ● = drapeau modifié
- 0 = drapeau à 0
- 1 = drapeau à 1

Code objet	Instruction	S	Z	P/V	C
8E	ADC A, (HL)	●	●	●	●
DD8Ed	ADC A, (IX+d)	●	●	●	●
FD8Ed	ADC A, (IY+d)	●	●	●	●
8F	ADC A, A	●	●	●	●
88	ADC A, B	●	●	●	●
89	ADC A, C	●	●	●	●
8A	ADC A, D	●	●	●	●
8B	ADC A, E	●	●	●	●
8C	ADC A, H	●	●	●	●
8D	ADC A, L	●	●	●	●
CEd	ADC A, d	●	●	●	●
ED4A	ADC HL, BC	●	●	●	●
ED5A	ADC HL, DE	●	●	●	●
ED6A	ADC HL, HL	●	●	●	●
ED7A	ADC HL, SP	●	●	●	●
86	ADD A, (HL)	●	●	●	●
DD86d	ADD A, (IX+d)	●	●	●	●
FD86d	ADD A, (IY+d)	●	●	●	●
87	ADD A, A	●	●	●	●
80	ADD A, B	●	●	●	●
81	ADD A, C	●	●	●	●
82	ADD A, D	●	●	●	●
83	ADD A, E	●	●	●	●
84	ADD A, H	●	●	●	●
85	ADD A, L	●	●	●	●
C6d	ADD A, d	●	●	●	●
09	ADD HL, BC				●
19	ADD HL, DE				●
29	ADD HL, HL				●
39	ADD HL, SP				●
DD09	ADD IX, BC				●
DD19	ADD IX, DE				●
DD29	ADD IX, IX				●
DD39	ADD IX, SP				●
FD09	ADD IY, BC				●
FD19	ADD IY, DE				●
FD29	ADD IY, IY				●
FD39	ADD IY, SP				●
A6	AND (HL)	●	●	●	0
DDA6d	AND (IX+d)	●	●	●	0
FDA6d	AND (IY+d)	●	●	●	0
A7	AND A	●	●	●	0



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
A0	AND B	●	●	●	0
A1	AND C	●	●	●	0
A2	AND D	●	●	●	0
A3	AND E	●	●	●	0
A4	AND H	●	●	●	0
A5	AND L	●	●	●	0
E6d	AND d	●	●	●	0
CB46	BIT 0, (HL)	●	●	●	
DDCBd46	BIT 0, (IX+d)	●	●	●	
FDCBd46	BIT 0, (IY+d)	●	●	●	
CB47	BIT 0, A	●	●	●	
CB40	BIT 0, B	●	●	●	
CB41	BIT 0, C	●	●	●	
CB42	BIT 0, D	●	●	●	
CB43	BIT 0, E	●	●	●	
CB44	BIT 0, H	●	●	●	
CB45	BIT 0, L	●	●	●	
CB4E	BIT 1, (HL)	●	●	●	
DDCBd4E	BIT 1, (IX+d)	●	●	●	
FDCBd4E	BIT 1, (IY+d)	●	●	●	
CB4F	BIT 1, A	●	●	●	
CB48	BIT 1, B	●	●	●	
CB49	BIT 1, C	●	●	●	
CB4A	BIT 1, D	●	●	●	
CB4B	BIT 1, E	●	●	●	
CB4C	BIT 1, H	●	●	●	
CB4D	BIT 1, L	●	●	●	
CB56	BIT 2, (HL)	●	●	●	
DDCBd56	BIT 2, (IX+d)	●	●	●	
FDCBd56	BIT 2, (IY+d)	●	●	●	
CB57	BIT 2, A	●	●	●	
CB50	BIT 2, B	●	●	●	
CB51	BIT 2, C	●	●	●	
CB52	BIT 2, D	●	●	●	
CB53	BIT 2, E	●	●	●	
CB54	BIT 2, H	●	●	●	
CB55	BIT 2, L	●	●	●	
CB5E	BIT 3, (HL)	●	●	●	
DDCBd5E	BIT 3, (IX+d)	●	●	●	
FDCBd5E	BIT 3, (IY+d)	●	●	●	
CB5F	BIT 3, A	●	●	●	
CB58	BIT 3, B	●	●	●	
CB59	BIT 3, C	●	●	●	
CB5A	BIT 3, D	●	●	●	
CB5B	BIT 3, E	●	●	●	
CB5C	BIT 3, H	●	●	●	
CB5D	BIT 3, L	●	●	●	



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

Code objet	Instruction	S	Z	P/V	C
CB66	BIT 4, (HL)	●	●	●	
DDCBd66	BIT 4, (IX+d)	●	●	●	
FDCBd66	BIT 4, (IY+d)	●	●	●	
CB67	BIT 4, A	●	●	●	
CB60	BIT 4, B	●	●	●	
CB61	BIT 4, C	●	●	●	
CB62	BIT 4, D	●	●	●	
CB63	BIT 4, E	●	●	●	
CB64	BIT 4, H	●	●	●	
CB65	BIT 4, L	●	●	●	
CB6E	BIT 5, (HL)	●	●	●	
DDCBd6E	BIT 5, (IX+d)	●	●	●	
FDCBd6E	BIT 5, (IY+d)	●	●	●	
CB6F	BIT 5, A	●	●	●	
CB68	BIT 5, B	●	●	●	
CB69	BIT 5, C	●	●	●	
CB6A	BIT 5, D	●	●	●	
CB6B	BIT 5, E	●	●	●	
CB6C	BIT 5, H	●	●	●	
CB6D	BIT 5, L	●	●	●	
DB76	BIT 6, (HL)	●	●	●	
DDCBd76	BIT 6, (IX+d)	●	●	●	
FDCBd76	BIT 6, (IY+d)	●	●	●	
CB77	BIT 6, A	●	●	●	
CB70	BIT 6, B	●	●	●	
CB71	BIT 6, C	●	●	●	
CB72	BIT 6, D	●	●	●	
CB73	BIT 6, E	●	●	●	
CB74	BIT 6, H	●	●	●	
CB75	BIT 6, L	●	●	●	
CB7E	BIT 7, (HL)	●	●	●	
DDCBd7E	BIT 7, (IX+d)	●	●	●	
FDCBd7E	BIT 7, (IY+d)	●	●	●	
CB7F	BIT 7, A	●	●	●	
CB78	BIT 7, B	●	●	●	
CB79	BIT 7, C	●	●	●	
CB7A	BIT 7, D	●	●	●	
CB7B	BIT 7, E	●	●	●	
CB7C	BIT 7, H	●	●	●	
CB7D	BIT 7, L	●	●	●	
DCaa	CALL C, aa				
FCaa	CALL M, aa				
D4aa	CALL NC, aa				
C4aa	CALL NZ, aa				
F4aa	CALL P, aa				
ECaa	CALL PE, aa				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
E4aa	CALL PO,aa				
CCaa	CALL Z;aa				
CDaa	CALL aa				
3F	CCF				•
BE	CP (HL)	•	•	•	•
DDBE d	CP (IX+d)	•	•	•	•
FDBE d	CP (IY+d)	•	•	•	•
BF	CP A	•	•	•	•
B8	CP B	•	•	•	•
B9	CP C	•	•	•	•
BA	CP D	•	•	•	•
BB	CP E	•	•	•	•
BC	CP H	•	•	•	•
BD	CP L	•	•	•	•
FE d	CP d	•	•	•	•
EDA9	CPD	•	•	•	
EDB9	CPDR	•	•	•	
EDB1	CPIR	•	•	•	
EDA1	CPI	•	•	•	
2F	CPL				
27	DAA	•	•	•	
35	DEC (HL)	•	•	•	
DD35d	DEC (IX+d)	•	•	•	
FD35d	DEC (IY+d)	•	•	•	
3D	DEC A	•	•	•	
05	DEC B	•	•	•	
0B	DEC BC				
0D	DEC C	•	•	•	
15	DEC D	•	•	•	
1B	DEC DE				
1D	DEC E	•	•	•	
25	DEC H	•	•	•	
2B	DEC HL				
DD2B	DEC IX				
FD2B	DEC IY				
2D	DEC L	•	•	•	
3B	DEC SP				
F3	DI				
10d	DJNZ d				
FB	EI				
E3	EX (SP),HL				
DDE3	EX (SP),IX				
FDE3	EX (SP),IY				
08	EX AF,AF'				
EB	EX DE,HL				
D9	EXX				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

Code objet	Instruction	S	Z	P/V	C
76	HALT				
ED46	IM 0				
ED56	IM 1				
ED5E	IM 2				
ED78	IN A, (C)	•	•	•	
ED40	IN B, (C)	•	•	•	
ED48	IN C, (C)	•	•	•	
ED50	IN D, (C)	•	•	•	
ED58	IN E, (C)	•	•	•	
ED60	IN H, (C)	•	•	•	
ED68	IN L, (C)	•	•	•	
DBd	IN A, (d)				
34	INC (HL)	•	•	•	
DD34d	INC (IX+d)	•	•	•	
FD34d	INC (IY+d)	•	•	•	
3C	INC A	•	•	•	
04	INC B	•	•	•	
03	INC BC				
0C	INC C	•	•	•	
14	INC D	•	•	•	
13	INC DE				
1C	INC E	•	•	•	
24	INC H	•	•	•	
23	INC HL				
DD23	INC IX				
FD23	INC IY				
2C	INC L	•	•	•	
33	INC SP				
EDAA	IND	•	•	•	
EDBA	INDR	•	•	•	
EDA2	INI	•	•	•	
EDB2	INIR	•	•	•	
C3aa	JP aa				
E9	JP (HL)				
DDE9	JP (IX)				
FDE9	JP (IY)				
DAdd	JP C, aa				
FAaa	JP M, aa				
D2aa	JP NC, aa				
C2aa	JP NZ, aa				
F2aa	JP P, aa				
EAaa	JP PE, aa				
E2aa	JP PO, aa				
CAaa	JP Z, aa				
38d	JR C, d				
30d	JR NC, d				
20d	JR NZ, d				



# CODES DES INSTRUCTIONS Z80 PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
28d	JR Z,d				
18d	JR d				
02	LD (BC),A				
12	LD (DE),A				
77	LD (HL),A				
70	LD (HL),B				
71	LD (HL),C				
72	LD (HL),D				
73	LD (HL),E				
74	LD (HL),H				
75	LD (HL),L				
36d	LD (HL),d				
DD77d	LD (IX+d),A				
DD70d	LD (IX+d),B				
DD71d	LD (IX+d),C				
DD72d	LD (IX+d),D				
DD73d	LD (IX+d),E				
DD74d	LD (IX+d),H				
DD75d	LD (IX+d),L				
DD36d20	LD (IX+d),d				
FD77d	LD (IY+d),A				
FD70d	LD (IY+d),B				
FD71d	LD (IY+d),C				
FD72d	LD (IY+d),D				
FD73d	LD (IY+d),E				
FD74d	LD (IY+d),H				
FD75d	LD (IY+d),L				
FD36d20	LD (IY+d),d				
32dd	LD (dd),A				
ED43dd	LD (dd),BC				
ED53dd	LD (dd),DE				
22dd	LD (dd),HL				
DD22dd	LD (dd),IX				
FD22dd	LD (dd),IY				
ED73dd	LD (dd),SP				
0A	LD A,(BC)				
1A	LD A,(DE)				
7E	LD A,(HL)				
DD7Ed	LD A,(IX+d)				
FD7Ed	LD A,(IY+d)				
3Add	LD A,(dd)				
7F	LD A,A				
78	LD A,B				
79	LD A,C				
7A	LD A,D				
7B	LD A,E				
7C	LD A,H				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

Code objet	Instruction	S	Z	P/V	C
ED57	LD A,I	●	●	●	
7D	LD A,L				
3E	LD A,d				
ED5F	LD A,R	●	●	●	
46	LD B,(HL)				
DD46d	LD B,(IX+d)				
FD46d	LD B,(IY+d)				
47	LD B,A				
40	LD B,B				
41	LD B,C				
42	LD B,D				
43	LD B,E				
44	LD B,H				
45	LD B,L				
06d	LD B,d				
ED4Bdd	LD BC,(dd)				
01dd	LD BC,dd				
4E	LD C,(HL)				
DD4Ed	LD C,(IX+d)				
FD4Ed	LD C,(IY+d)				
4F	LD C,A				
48	LD C,B				
49	LD C,C				
4A	LD C,D				
4B	LD C,E				
4C	LD C,H				
4D	LD C,L				
0Ed	LD C,d				
56	LD D,(HL)				
DD56d	LD D,(IX+d)				
FD56d	LD D,(IY+d)				
57	LD D,A				
50	LD D,B				
51	LD D,C				
52	LD D,D				
53	LD D,E				
54	LD D,H				
55	LD D,L				
16d	LD D,d				
ED5Bdd	LD DE,(dd)				
11dd	LD DE,dd				
5E	LD E,(HL)				
DD5Ed	LD E,(IX+d)				
FD5Ed	LD E,(IY+d)				
5F	LD E,A				
58	LD E,B				
59	LD E,C				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
5A	LD E,D				
5B	LD E,E				
5C	LD E,H				
5D	LD E,L				
1E20	LD E,n				
66	LD H,(HL)				
DD66d	LD H,(IX+d)				
FD66d	LD H,(IY+d)				
67	LD H,A				
60	LD H,B				
61	LD H,C				
62	LD H,D				
63	LD H,E				
64	LD H,H				
65	LD H,L				
26d	LD H,d				
2Add	LD HL,(dd)				
21dd	LD HL,dd				
ED47	LD I,A				
DD2Add	LD IX,(dd)				
DD21dd	LD IX,dd				
FD2Add	LD IY,(dd)				
FD21dd	LD IY,dd				
6E	LD L,(HL)				
DD6Ed	LD L,(IX+d)				
FD6Ed	LD L,(IY+d)				
6F	LD L,A				
68	LD L,B				
69	LD L,C				
6A	LD L,D				
6B	LD L,E				
6C	LD L,H				
6D	LD L,L				
2Ed	LD L,d				
ED4F	LD R,A				
ED7Bdd	LD SP,(dd)				
F9	LD SP,HL				
DDF9	LD SP,IX				
FDF9	LD SP,IY				
31dd	LD SP,dd				
EDA8	LDD			0	
EDB8	LDDR			0	
EDA0	LDI			0	
EDB0	LDIR			0	
ED44	NEG				
00	NOP				
B6	OR (HL)	0	0	0	0



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
DDB6d	OR (IX+d)	●	●	●	0
FDB6d	OR (IY+d)	●	●	●	0
B7	OR A	●	●	●	0
B0	OR B	●	●	●	0
B1	OR C	●	●	●	0
B2	OR D	●	●	●	0
B3	OR E	●	●	●	0
B4	OR H	●	●	●	0
B5	OR L	●	●	●	0
F6d	OR d	●	●	●	0
EDBB	OTDR	●	●	●	
EDB3	OTIR	●	●	●	
ED79	OUT (C),A				
ED41	OUT (C),B				
ED49	OUT (C),C				
ED51	OUT (C),D				
ED59	OUT (C),E				
ED61	OUT (C),H				
ED69	OUT (C),L				
D3d	OUT (d),A				
EDAB	OUTD	●	●	●	
EDA3	OUTI	●	●	●	
F1	POP AF				
C1	POP BC				
D1	POP DE				
E1	POP HL				
DDE1	POP IX				
FDE1	POP IY				
F5	PUSH AF				
C5	PUSH BC				
D5	PUSH DE				
E5	PUSH HL				
DDE5	PUSH IX				
FDE5	PUSH IY				
CB86	RES 0,(HL)				
DDCBd86	RES 0,(IX+d)				
FDCBd86	RES 0,(IY+d)				
CB87	RES 0,A				
CB80	RES 0,B				
CB81	RES 0,C				
CB82	RES 0,D				
CB83	RES 0,E				
CB84	RES 0,H				
CB85	RES 0,L				
CB8E	RES 1,(HL)				
DDCBd8E	RES 1,(IX+d)				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
FDCBd8E	RES 1, (IY+d)				
CB8F	RES 1, A				
CB88	RES 1, B				
CB89	RES 1, C				
CB8A	RES 1, D				
CB8B	RES 1, E				
CB8C	RES 1, H				
CB8D	RES 1, L				
CB96	RES 2, (HL)				
DDCBd96	RES 2, (IX+d)				
FDCBd96	RES 2, (IY+d)				
CB97	RES 2, A				
CB90	RES 2, B				
CB91	RES 2, C				
CB92	RES 2, D				
CB93	RES 2, E				
CB94	RES 2, H				
CB95	RES 2, L				
CB9E	RES 3, (HL)				
DDCBd9E	RES 3, (IX+d)				
FDCBd9E	RES 3, (IY+d)				
CB9F	RES 3, A				
CB98	RES 3, B				
CB99	RES 3, C				
CB9A	RES 3, D				
CB9B	RES 3, E				
CB9C	RES 3, H				
CB9D	RES 3, L				
CBA6	RES 4, (HL)				
DDCBdA6	RES 4, (IX+d)				
FDCBdA7	RES 4, (IY+d)				
CBA7	RES 4, A				
CBA0	RES 4, B				
CBA1	RES 4, C				
CBA2	RES 4, D				
DBA3	RES 4, E				
CBA4	RES 4, H				
CBA5	RES 4, L				
CBAE	RES 5, (HL)				
DDCBdAE	RES 5, (IX+d)				
FDCBdAE	RES 5, (IY+d)				
CBAF	RES 5, A				
CBA8	RES 5, B				
CBA9	RES 5, C				
CBAA	RES 5, D				
CBAB	RES 5, E				
CBAC	RES 5, H				



# CODES DES INSTRUCTIONS Z80 PAR ORDRE ALPHABETIQUE

LANGUAGE MACHINE

Code objet	Instruction	S	Z	P/V	C
CBAD	RES 5,L				
CBB6	RES 6,(HL)				
DDCBdB6	RES 6,(IX+d)				
FDCBdB6	RES 6,(IY+d)				
CBB7	RES 6,A				
CBB0	RES 6,B				
CBB1	RES 6,C				
CBB2	RES 6,D				
CBB3	RES 6,E				
CBB4	RES 6,H				
CBB5	RES 6,L				
CBBE	RES 7,(HL)				
DDCBdBE	RES 7,(IX+d)				
FDCBdBE	RES 7,(IY+d)				
CBBF	RES 7,A				
CBB8	RES 7,B				
CBB9	RES 7,C				
CBBA	RES 7,D				
CBBB	RES 7,E				
CBBC	RES 7,H				
CBBD	RES 7,L				
C9	RET				
D8	RET C				
F8	RET M				
D0	RET NC				
C0	RET NZ				
F0	RET P				
E8	RET PE				
E0	RET PO				
C8	RET Z				
ED4D	RETI				
ED45	RETN				
CB16	RL (HL)	•	•	•	•
DDCBd16	RL (IX+d)	•	•	•	•
FDCBd16	RL (IY+d)	•	•	•	•
CB17	RL A	•	•	•	•
CB10	RL B	•	•	•	•
CB11	RL C	•	•	•	•
CB12	RL D	•	•	•	•
CB13	RL E	•	•	•	•
CB14	RL H	•	•	•	•
CB15	RL L	•	•	•	•
17	RLA				•
CB06	RLC (HL)	•	•	•	•
DDCBd06	RLC (IX+d)	•	•	•	•
FDCBd06	RLC (IY+d)	•	•	•	•



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
CB07	RLC A	•	•	•	•
CB00	RLC B	•	•	•	•
CB01	RLC C	•	•	•	•
CB02	RLC D	•	•	•	•
CB03	RLC E	•	•	•	•
CB04	RLC H	•	•	•	•
CB05	RLC L	•	•	•	•
07	RLCA				•
ED6F	RLD	•	•	•	
CB1E	RR (HL)	•	•	•	•
DDCBd 1E	RR (IX+d)	•	•	•	•
FDCBd 1E	RR (IY+d)	•	•	•	•
CB1F	RR A	•	•	•	•
CB18	RR B	•	•	•	•
CB19	RR C	•	•	•	•
CB1A	RR D	•	•	•	•
CB1B	RR E	•	•	•	•
CB1C	RR H	•	•	•	•
CB1D	RR L	•	•	•	•
1F	RRA				•
CB0E	RRC (HL)	•	•	•	•
DDCBd0E	RRC (IX+d)	•	•	•	•
FDCBd0E	RRC (IY+d)	•	•	•	•
CB0F	RRC A	•	•	•	•
CB08	RRC B	•	•	•	•
CB09	RRC C	•	•	•	•
CB0A	RRC D	•	•	•	•
CB0B	RRC E	•	•	•	•
CB0C	RRC H	•	•	•	•
CB0D	RRC L	•	•	•	•
0F	RRCA				•
ED67	RRD	•	•	•	
C7	RST 00H				
CF	RST 08H				
D7	RST 10H				
DF	RST 18H				
E7	RST 20H				
EF	RST 28H				
F7	RST 30H				
FF	RST 38H				
DEd	SBC A,d	•	•	•	•
9E	SBC A, (HL)	•	•	•	•
DD9Ed	SBC A, (IX+d)	•	•	•	•
FD9Ed	SBC A, (IY+d)	•	•	•	•
9F	SBC A,A	•	•	•	•
98	SBC A,B	•	•	•	•



# CODES DES INSTRUCTIONS Z80 PAR ORDRE ALPHABETIQUE

Code objet	Instruction	S	Z	P/V	C
99	SBC A,C	●	●	●	●
9A	SBC A,D	●	●	●	●
9B	SBC A,E	●	●	●	●
9C	SBC A,H	●	●	●	●
9D	SBC A,L	●	●	●	●
ED42	SBC HL,BC	●	●	●	●
ED52	SBC HL,DE	●	●	●	●
ED62	SBC HL,HL	●	●	●	●
ED72	SBC HL,SP	●	●	●	●
37	SCF				1
CBC6	SET 0,(HL)				
DDCBdC6	SET 0,(IX+d)				
FDCBdC6	SET 0,(IY+d)				
CBC7	SET 0,A				
CBC0	SET 0,B				
CBC1	SET 0,C				
CBC2	SET 0,D				
CBC3	SET 0,E				
CBC4	SET 0,H				
CBC5	SET 0,L				
CBCE	SET 1,(HL)				
ddCBdCE	SET 1,(IX+d)				
FDCBdCE	SET 1,(IY+d)				
CBCF	SET 1,A				
CBC8	SET 1,B				
CBC9	SET 1,C				
CBCA	SET 1,D				
CBCB	SET 1,E				
CBCC	SET 1,H				
CBCD	SET 1,L				
CBD6	SET 2,(HL)				
DDCBdD6	SET 2,(IX+d)				
FDCBdD6	SET 2,(IY+d)				
CBD7	SET 2,A				
CBD0	SET 2,B				
CBD1	SET 2,C				
CBD2	SET 2,D				
CBD3	SET 2,E				
CBD4	SET 2,H				
CBD5	SET 2,L				
CBDE	SET 3,(HL)				
DDCBdDE	SET 3,(IX+d)				
FDCBdDE	SET 3,(IY+d)				
CBDF	SET 3,A				
CBD8	SET 3,B				
CBD9	SET 3,C				
CBDA	SET 3,D				



CODES DES INSTRUCTIONS Z80  
PAR ORDRE ALPHABETIQUE

<i>Code objet</i>	<i>Instruction</i>	<i>S</i>	<i>Z</i>	<i>P/V</i>	<i>C</i>
CBDB	SET 3,E				
CBDC	SET 3,H				
CBDD	SET 3,L				
CBE6	SET 4,(HL)				
DDCBdE6	SET 4,(IX+d)				
FDCBdE6	SET 4,(IY+d)				
CBE7	SET 4,A				
CBE0	SET 4,B				
CBE1	SET 4,C				
CBE2	SET 4,D				
CBE3	SET 4,E				
CBE4	SET 4,H				
CBE5	SET 4,L				
CBEE	SET 5,(HL)				
DDCBdEE	SET 5,(IX+d)				
FDCBdEE	SET 5,(IY+d)				
CBEF	SET 5,A				
CBE8	SET 5,B				
CBE9	SET 5,C				
CBEA	SET 5,D				
CBEB	SET 5,E				
CBEC	SET 5,H				
CBED	SET 5,L				
CBF6	SET 6,(HL)				
DDCBdF6	SET 6,(IX+d)				
FDCBdF6	SET 6,(IY+d)				
CBF7	SET 6,A				
CBF0	SET 6,B				
CBF1	SET 6,C				
CBF2	SET 6,D				
CBF3	SET 6,E				
CBF4	SET 6,H				
CBF5	SET 6,L				
CBFE	SET 7,(HL)				
DDCBdFE	SET 7,(IX+d)				
FDCBdFE	SET 7,(IY+d)				
CBFF	SET 7,A				
CBF8	SET 7,B				
CBF9	SET 7,C				
CBFA	SET 7,D				
CBFB	SET 7,E				
CBFC	SET 7,H				
CBFD	SET 7,L				
CB26	SLA (HL)	•	•	•	•
DDCBd26	SLA (IX+d)	•	•	•	•
FDCBd26	SLA (IY+d)	•	•	•	•
CB27	SLA A	•	•	•	•
CB20	SLA B	•	•	•	•



# CODES DES INSTRUCTIONS Z80 PAR ORDRE ALPHABETIQUE

Code objet	Instruction	S	Z	P/V	C
CB21	SLA C	●	●	●	●
CB22	SLA D	●	●	●	●
CB23	SLA E	●	●	●	●
CB24	SLA H	●	●	●	●
CB25	SLA L	●	●	●	●
CB2E	SRA (HL)	●	●	●	●
DDCBd2E	SRA (IX+d)	●	●	●	●
FDCBd2E	SRA (IY+d)	●	●	●	●
CB2F	SRA A	●	●	●	●
CB28	SRA B	●	●	●	●
CB29	SRA C	●	●	●	●
CB2A	SRA D	●	●	●	●
CB2B	SRA E	●	●	●	●
CB2C	SRA H	●	●	●	●
CB2D	SRA L	●	●	●	●
CB3E	SRL (HL)	●	●	●	●
DDCBd3E	SRL (IX+d)	●	●	●	●
FDCBd3E	SRL (IY+d)	●	●	●	●
CB3F	SRL A	●	●	●	●
CB38	SRL B	●	●	●	●
CB39	SRL C	●	●	●	●
CB3A	SRL D	●	●	●	●
CB3B	SRL E	●	●	●	●
CB3C	SRL H	●	●	●	●
CB3D	SRL L	●	●	●	●
96	SUB (HL)	●	●	●	●
DD96d	SUB (IX+d)	●	●	●	●
FD96d	SUB (IY+d)	●	●	●	●
97	SUB A	●	●	●	●
90	SUB B	●	●	●	●
91	SUB C	●	●	●	●
92	SUB D	●	●	●	●
93	SUB E	●	●	●	●
94	SUB H	●	●	●	●
95	SUB L	●	●	●	●
D6d	SUB d	●	●	●	●
AE	XOR (HL)	●	●	●	0
DDAEd	XOR (IX+d)	●	●	●	0
FDAEd	XOR (IY+d)	●	●	●	0
AF	XOR A	●	●	●	0
A8	XOR B	●	●	●	0
A9	XOR C	●	●	●	0
AA	XOR D	●	●	●	0
AB	XOR E	●	●	●	0
AC	XOR H	●	●	●	0
AD	XOR L	●	●	●	0
EEd	XOR d	●	●	●	0



# TABLEAUX DE DESASSEMBLAGE

## Instructions sans préfixes

- n : octet (8 bits, de 0 à 255) ;
- nn : double octet (16 bits, de 0 à 65535) ;
- d : déplacement pour l'adressage relatif (8 bits).

2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	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	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 0	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 avec le préfixe CB

Toutes les instructions de ce tableau doivent être précédées du préfixe CB.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	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 0,B	BIT 0,C	BIT 0,D	BIT 0,E	BIT 0,H	BIT 0,L	BIT 0,(HL)	BIT 0,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 0,B	RES 0,C	RES 0,D	RES 0,E	RES 0,H	RES 0,L	RES 0,(HL)	RES 0,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 0,B	SET 0,C	SET 0,D	SET 0,E	SET 0,H	SET 0,L	SET 0,(HL)	SET 0,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



# TABLEAUX DE DESASSEMBLAGE

## Instructions avec préfixe ED

Toutes les instructions de ce tableau doivent être précédées du préfixe ED.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2																
3																
4	IN B,(C)	OUT (C),B	SBC HL,BC	ID (nn),BC	NEG	RETN	IM 0	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

Toutes les instructions de ce tableau doivent être précédées par le préfixe DD, dans le cas du registre d'index IX, et de FD, pour le registre IY.

Code	Mnémonique	Code	Mnémonique
09	ADD IX,BC	CB d 0E	RRC (IX+d)
19	ADD IX,DE	CB d 16	RL (IX+d)
21	LD IX,nn	CB d 1E	RR (IX+d)
22	LD (nn),IX	CB d 26	SLA (IX+d)
23	INC IX	CB d 2E	SRA (IX+d)
29	ADD IX,IX	CB d 3E	SRL (IX+d)
2A	LD IX,(nn)	CB d 46	BIT 0,(IX+d)
2B	DEC IX	CB d 4E	BIT 1,(IX+d)
34	INC (IX+d)	CB d 56	BIT 2,(IX+d)
35	DEC (IX+d)	CB d 5E	BIT 3,(IX+d)
36	LD (IX+d),nn	CB d 66	BIT 4,(IX+d)
39	ADD IX,SP	CB d 6E	BIT 5,(IX+d)
46	LD B,(IX+d)	CB d 76	BIT 6,(IX+d)
4E	LD C,(IX+d)	CB d 7E	BIT 7,(IX+d)
56	LD D,(IX+d)	CB d 86	RES 0,(IX+d)
5E	LD E,(IX+d)	CB d 8E	RES 1,(IX+d)
66	LD H,(IX+d)	CB d 96	RES 2,(IX+d)
6E	LD L,(IX+d)	CB d 9E	RES 3,(IX+d)
70	LD (IX+d),B	CB d A6	RES 4,(IX+d)
71	LD (IX+d),C	CB d AE	RES 5,(IX+d)
72	LD (IX+d),D	CB d B6	RES 6,(IX+d)
73	LD (IX+d),E	CB d BE	RES 7,(IX+d)
74	LD (IX+d),H	CB d C6	SET 0,(IX+d)
75	LD (IX+d),L	CB d CE	SET 1,(IX+d)
77	LD (IX+d),A	CB d D6	SET 2,(IX+d)
7E	LD 1,(IX+d)	CB d DE	SET 3,(IX+d)
86	ADD A,(IX+d)	CB d E6	SET 4,(IX+d)
8E	ADC A,(IX+d)	CB d EE	SET 5,(IX+d)
96	SUB (IX+d)	CB d F6	SET 6,(IX+d)
9E	SBC A,(IX+d)	CB d FE	SET 7,(IX+d)
A6	AND (IX+d)	E1	POP IX
AE	XOR (IX+d)	E3	EX (SP),IX
B6	OR (IX+d)	E5	PUSH IX
BE	CP (IX+d)	E9	JP (IX)
CB d 06	RLC (IX+d)	F9	LD SP,IX



## GENERALITES

Le logiciel interne de l'Amstrad peut être divisé en trois parties :

- la ROM inférieure : elle contient les différents gestionnaires repris ci-dessous, les routines mathématiques et le générateur de caractères ;
- la ROM supérieure : elle contient le Basic proprement dit ;
- la zone de travail en mémoire vive : elle contient les variables système, les vecteurs d'appel des routines de la ROM inférieure et différents tampons utilisés par les gestionnaires et le Basic.

### Les gestionnaires peuvent être divisés en huit grandes classes

#### *Le gestionnaire clavier*

Il s'occupe de la gestion du clavier, de générer les caractères des touches de fonctions, de tester le BREAK et de gérer les manettes de jeux.

#### *Le gestionnaire du mode texte*

Il s'occupe de la gestion du curseur, des codes contrôle et de l'affichage des caractères sur l'écran.

#### *Le gestionnaire graphique*

Il s'occupe de tracer des points et de tirer des lignes sur l'écran.

#### *Le gestionnaire d'écran*

Il interface le texte et les graphiques avec les circuits spécialisés de gestion d'écran.

#### *Le gestionnaire cassette*

Il s'occupe de la lecture, de l'écriture et de la commande du moteur de la cassette.



## GENERALITES

### *Le gestionnaire sonore*

Il s'occupe de la gestion des queues sonores, des enveloppes, du synchronisme, etc.

### *Le noyau (kernel)*

C'est le coeur du système opératoire qui s'occupe des interruptions, du lancement des programmes et de la sélection des mémoires mortes.

### *L'unité de contrôle du matériel*

Il s'occupe de la gestion de l'imprimante et de l'interfaçage avec le matériel au niveau le plus bas.

### *Le bloc de saut*

Il s'occupe de réinitialiser tous les vecteurs.

**Pour plus de facilité, le logiciel système sera présenté de la façon suivante**

- La table des points d'entrée en mémoire vive des routines système utilisées par les gestionnaires (page 81).
- Les vecteurs d'indirection en mémoire vive (page 105).
- Les vecteurs du noyau et les restarts en mémoire vive (page 107).
- Les vecteurs d'appel des routines mathématiques en mémoire vive (page 111).
- Les principales variables système en mémoire vive (page 115).
- Les adresses principales de la ROM inférieure (page 120).
- Les adresses principales de la ROM supérieure (page 126).
- La table des correspondances entre les vecteurs et les adresses de la ROM inférieure (page 131).
- La table des points d'entrée des mots-clés du Basic (page 133).
- Les différents blocs de contrôle système (page 135).



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

Pour chaque sous-routine, vous trouverez, derrière le numéro de point d'entrée, son adresse en hexadécimal suivie d'une explication. Les abréviations CE et CS signifient Conditions d'Entrée et Conditions de Sortie.

**ABFF : Initialise le DOS.**

### Le gestionnaire clavier

- 00 BB00 Initialisation du gestionnaire clavier.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés. Tous les autres registres sont préservés.
- 01 BB03 RESET du gestionnaire clavier.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés. Tous les autres registres sont préservés.
- 02 BB06 Le gestionnaire clavier attend le caractère tapé au clavier.  
Pas de CE.  
CS : si le sémaphore de CARRY est vrai, l'Accumulateur contient le caractère qui a été tapé. Tous les registres sont préservés.
- 03 BB09 Lecture d'un caractère.  
Cette routine teste si un caractère est disponible depuis le clavier.  
Pas de CE.  
CS : si un caractère est disponible, le sémaphore de CARRY est vrai et A contient le caractère. S'il n'y a pas de caractère disponible, le sémaphore de CARRY est faux et A est modifié. Tous les autres registres sont conservés.
- 04 BB0C Réserve un caractère pour le prochain appel de la routine précédente.  
CE : A contient le caractère à sauver.  
CS : tous les registres sont préservés.
- 05 BB0F Positionne une chaîne de caractères associée à un code.  
CE : B contient le code qui doit être associé à une chaîne de caractères.  
C contient la longueur de la chaîne.  
HL contient son adresse.  
CS : Si l'expansion est faite, le sémaphore de CARRY est vrai. Si la chaîne est trop longue ou le code invalide, le sémaphore de CARRY est faux. A, BC, DE et HL sont modifiés.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 06 BB12 Lit un caractère depuis une chaîne de caractères expansée. Les caractères contenus dans la chaîne de caractères sont numérotés en partant de 0.  
CE : A contient le code d'expansion. L contient le numéro du caractère.  
CS : si le caractère est trouvé, A contient le caractère et le sémaphore de CARRY est vrai. Si le code est mauvais ou si la chaîne de caractères n'est pas assez longue, le sémaphore de CARRY est faux et A est modifié. DE est modifié.
- 07 BB15 Allocation d'un tampon pour une chaîne de caractères expansée.  
CE : DE contient l'adresse du tampon et HL sa longueur.  
CS : si tout est correct, le sémaphore de CARRY est vrai ; sinon, il est faux. Les registres A, BC, DE et HL sont modifiés.
- 08 BB18 Attente d'un caractère en provenance du clavier.  
Pas de CE.  
CS : le sémaphore de CARRY est vrai et A contient le caractère qui a été frappé. Tous les registres sont préservés.
- 09 BB1B Teste si une touche en provenance du clavier est disponible.  
Pas de CE.  
CS : si une touche est disponible, le sémaphore de CARRY est vrai et A contient le caractère ; sinon, le CARRY est faux.
- 10 BB1E Teste si une touche est pressée.  
Permet aussi de tester la manette de jeux.  
CE : A contient le numéro de la touche qui doit être testée.  
CS : si la touche n'est pas pressée, le sémaphore de CARRY est vrai ; si la touche est pressée, le sémaphore de CARRY est faux et A et HL sont modifiés et C contient l'état des touches SHIFT et CONTROLE.
- 11 BB21 Vérifie si la touche de verrouillage des majuscules est enfoncée (CAPS LOCK).  
Pas de CE.  
CS : L contient l'état de la touche SHIFT et A contient l'état de la touche de verrouillage des majuscules ; A contient 00 si elle n'est pas enfoncée et FF si elle l'est.  
Le registre AF est modifié.
- 12 BB24 Lecture de l'état de la manette de jeux.  
Pas de CE.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- CS : H contient l'état de la manette de jeux n° 0.  
L contient l'état de la manette de jeux n° 1.  
A contient l'état de la manette de jeux n° 0.  
La correspondance des bits est la même que celle de la fonction JOY déjà décrite dans les fonctions Basic.
- 13 BB27** Positionne le code qui sera fourni lors de la pression d'une touche sans CTRL ni SHIFT.  
CE : A contient le numéro de la touche.  
B contient le code ASCII que cette touche fournira.  
CS : AF et HL sont modifiés.
- 14 BB2A** Fournit le code correspondant au numéro de touche pressée.  
CE : A contient le numéro de la touche.  
CS : A contient le code ASCII correspondant à la touche.  
HL et F sont modifiés.
- 15 BB2D** Positionne le code qui sera fourni lors de la pression d'une touche conjointement à la pression de la touche SHIFT.  
CE : A contient le numéro de la touche.  
B contient le code ASCII que cette touche fournira.  
CS : AL et HL sont modifiés.
- 16 BB30** Fournit le code correspondant à une touche pressée conjointement à la touche SHIFT.  
CE : A contient le numéro de la touche.  
CS : A contient le code ASCII correspondant à la touche.  
HL est modifié.
- 17 BB33** Positionne le code que fournira une touche enfoncée conjointement à la touche CTRL.  
CE : A contient le numéro de la touche.  
B contient le code ASCII que cette touche fournira.  
CS : AF et HL sont modifiés.
- 18 BB36** Fournit le code ASCII correspondant à la touche pressée conjointement à la touche CTRL.  
CE : A contient le numéro de la touche.  
CS : A contient le code ASCII correspondant à la touche.  
HL est modifié.
- 19 BB39** Positionne l'entrée dans la table des touches qui doivent se répéter.  
CE : A contient le numéro de la touche.  
Si la touche peut être répétée, B contient FF ; sinon, B contient 00.  
CS : AF, BC et HL sont modifiés.
- 20 BB3C** Teste si une touche dont le numéro est fourni peut ou non être répétée.  
CE : A contient le numéro de la touche.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

CS : si la touche peut être répétée, le sémaphore 0 est faux ; si elle ne peut pas être répétée, le sémaphore 0 est vrai. De toute façon, le sémaphore de CARRY est faux et AF et HL sont modifiés.

- 21 BB3F Positionne le temps de délai avant la première répétition, ainsi que le temps prévu entre deux répétitions.  
CE : H contient le délai avant la première répétition.  
L contient la vitesse de répétition. Ces temps sont exprimés en cinquantième de seconde.  
CS : AF est modifié.

- 22 BB42 Lecture de la vitesse de répétition et du temps prévu avant la première répétition.  
Pas de CE.  
CS : H contient le délai avant la première répétition (en 1/50° de seconde) et L contient la vitesse de répétition. AF est modifié.

- 23 BB45 Positionne le mécanisme du BREAK.  
CE : DE contient l'adresse de la routine de traitement du BREAK.  
C contient l'adresse de ROM sélectionnée pour cette routine.  
CS : AF, BC, DE et HL sont modifiés.

*Remarque* : ce mécanisme peut être désarmé par l'appel à la routine suivante.

- 24 BB48 Désarmement du mécanisme du BREAK.  
Pas de CE.  
CS : AF et HL sont modifiés.
- 25 BB4B Génère une interruption de BREAK si le BREAK a été armé au moyen de la routine 23.  
Pas de CE.  
CS : AF et HL sont modifiés.

### Le gestionnaire du mode texte

- 26 BB4E Initialisation du mode TEXTE.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 27 BB51 RESET du mode TEXTE.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 28 BB54 Permet à un caractère d'être placé sur l'écran en mode texte.  
Pas de CE.  
CS : AF est modifié.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 29 BB57 Interdiction de placer des caractères sur l'écran.  
Pas de CE.  
CS : AF est modifié.
- + 30 BB5A Sortie sur l'écran en mode texte d'un caractère ou d'un code de contrôle interprété (les codes de contrôle sont tous des caractères compris entre 0 et 1F).  
CE : A contient le caractère à envoyer.  
CS : tous les registres sont préservés.
- 31 BB5D Sortie sur l'écran en mode texte d'un caractère ou du graphique correspondant à un code de contrôle.  
CE : A contient le caractère à imprimer.  
CS : AF, BC, DE et HL sont modifiés.
- 32 BB60 Lecture d'un caractère en provenance de l'écran à la position courante du curseur.  
Pas de CE.  
CS : si un caractère a été reconnu, le sémaphore de CARRY est vrai et A contient ce caractère. Dans le cas contraire, le CARRY est faux et A contient 0.
- 33 BB63 Positionnement ON ou OFF du dispositif de traitement des caractères graphiques.  
CE : A=0 si l'écriture de graphiques est interdite (OFF). Dans le cas contraire (ON), A est différent de 0.  
CS : AF est modifié.
- 34 BB66 Positionne la taille de la fenêtre de texte courante.  
CE : H contient la colonne des deux premiers coins.  
D contient la colonne des deux autres coins.  
L contient la ligne de deux coins.  
E contient la ligne des deux autres coins.  
CS : AF, BC, DE et HL sont modifiés.
- 35 BB69 Lecture de la taille de la fenêtre courante.  
Pas de CE.  
CS : si la fenêtre couvre l'écran complet, le CARRY est faux, sinon il est vrai. Dans les deux cas, H contient le numéro de la colonne de gauche, D le numéro de la colonne de droite, L le numéro de la ligne du haut et E le numéro de la ligne du bas. A est modifié.
- 36 BB6C Effacement de la fenêtre courante (CLS).  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 37 BB6F Détermine la position horizontale du curseur.  
CE : A contient le numéro de la colonne du curseur.  
CS : AF et HL sont modifiés.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 38 BB72 Détermine la position verticale du curseur.  
CE : A contient le numéro de la ligne du curseur.  
CS : AF et HL sont modifiés.
- 39 BB75 Détermine la position du curseur.  
CE : H contient le numéro de la colonne du curseur.  
L contient le numéro de la ligne du curseur.  
CS : AF et HL sont modifiés.
- 40 BB78 Lecture de la position du curseur.  
Pas de CE.  
CS : H contient le numéro de la colonne du curseur.  
L contient le numéro de la ligne du curseur.  
A contient le compteur de défilement (scrolling).
- 41 BB7B Utilisation de l'affichage du curseur en mode texte.  
Pas de CE.  
CS : AF est modifié.
- 42 BB7E Interdiction d'afficher le curseur en mode texte.  
Pas de CE.  
CS : AF est modifié.
- 43 BB81 Autorise l'affichage du curseur pour le système.  
Pas de CE.  
Pas de CS.
- 44 BB84 Interdit l'affichage du curseur pour le système.  
Pas de CE.  
Pas de CS.
- 45 BB87 Teste si une position curseur se trouve à l'intérieur d'une fenêtre.  
CE : H contient le numéro de la colonne de la position à tester.  
L contient le numéro de la ligne de la position à tester.  
CS : H contient le numéro de la colonne où le caractère sera imprimé.  
L contient le numéro de la ligne où le caractère sera imprimé.  
A et F sont modifiés. Si l'impression ne doit pas causer le défilement de la fenêtre (scrolling), le sémaphore de CARRY est vrai et B est modifié. Si elle doit causer le défilement de la fenêtre, le sémaphore de CARRY est faux et B contient FF. Si elle doit causer un défilement inverse, le CARRY est faux et B contient 00.
- 46 BB8A Positionne un curseur sur l'écran.  
Pas de CE.  
CS : AF est modifié.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 47 BB8D Enlève un curseur de l'écran.  
Pas de CE.  
CS : AF est modifié.
- > 48 BB90 Détermine la couleur des caractères.  
CE : A contient le numéro d'encre.  
CS : AF et HL sont modifiés.
- 49 BB93 Lecture de la couleur des caractères.  
Pas de CE.  
CS : A contient le numéro d'encre.  
F est modifié.
- 50 BB96 Détermine la couleur de fond pour le texte (papier).  
CE : A contient le numéro d'encre.  
CS : AF et HL sont modifiés.
- 51 BB99 Lecture de la couleur de fond pour le texte (papier).  
Pas de CE.  
CS : A contient le numéro de la couleur du papier.  
A et F sont modifiés.
- 52 BB9C Intervertit la couleur des caractères et la couleur du fond.  
Pas de CE.  
CS : AF et HL sont modifiés.
- 53 BB9F Permet ou interdit l'affichage du fond.  
CE : si le fond est affiché (mode opaque), A=0 ; si le fond n'est pas affiché (mode transparent), A est différent de 0.  
CS : AF et HL sont modifiés.
- 54 BBA2 Teste si le fond peut ou non être affiché.  
Pas de CE.  
CS : A=0 si le fond peut être affiché (mode opaque), sinon (mode transparent) A est différent de 0. DE, HL et F sont modifiés.
- 55 BBA5 Lecture de l'adresse d'une matrice de caractères.  
CE : A contient le caractère à rechercher dans la matrice.  
CS : A et F sont modifiés. Si la matrice est une matrice définie par l'utilisateur, le CARRY est vrai. Si la matrice est dans la ROM, le CARRY est faux et HL contient l'adresse de la matrice.
- > 56 BBA8 Positionnement d'une matrice pour un caractère défini par l'utilisateur.  
CE : A contient le caractère où la matrice doit être positionnée et HL contient l'adresse de la matrice.  
CS : si le caractère est définissable par l'utilisateur, le CARRY est vrai, sinon il est faux. AF, BC, DE et HL sont modifiés.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- > 57 BBAB Positionnement de l'adresse de la table d'une matrice définie par l'utilisateur.  
CE : DE contient le premier caractère de la table et HL contient l'adresse de départ de la nouvelle table.  
CS : s'il n'y avait pas encore de table, le CARRY est faux et A et HL sont modifiés. S'il y avait déjà une table définie par l'utilisateur, le CARRY est vrai, A contient le premier caractère de l'ancienne table, HL contient l'adresse de l'ancienne table et BC et DE sont modifiés.
- 58 BBAE Lecture de l'adresse de table pour une matrice définie par l'utilisateur.  
Pas de CE.  
CS : s'il n'y a pas de tables des matrices définies par l'utilisateur, le CARRY est faux et A et HL sont modifiés. S'il y en a une, le CARRY est vrai, A contient le premier caractère de la table et HL contient l'adresse de départ de la table.
- 59 BBB1 Lecture de l'adresse de la table des codes de contrôle.  
Pas de CE.  
CS : HL contient l'adresse des codes de contrôle. Tous les autres registres sont préservés.
- 60 BBB4 Positionne une nouvelle table d'attribut (VDU stream).  
CE : A contient le numéro du stream requis.  
CS : A contient le numéro de l'ancien stream.  
HL et F sont modifiés.
- 61 BBB7 Echange des états de deux tables d'attributs (STREAMS).  
CE : B contient le numéro du stream 1.  
C contient le numéro du stream 2.  
CS : AF, BC, DE et HL sont modifiés.
- Remarque* : le stream est composé :
- du numéro d'encre ;
  - du numéro de papier ;
  - du curseur position ;
  - des limites des fenêtres.

### Le gestionnaire graphique

- 62 BBBA Initialisation du mode graphique.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 63 BBBD RESET du gestionnaire graphique.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 64 BBC0 Déplacement de la position courante en coordonnées absolues.  
CE : DE contient la coordonnée absolue X.  
HL contient la coordonnée absolue Y.  
CS : AF, BC, DE et HL sont modifiés.
- 65 BBC3 Déplacement de la position courante en coordonnées relatives à la position du curseur.  
CE : DE contient la coordonnée relative X.  
HL contient la coordonnée relative Y.  
CS : AF, BC, DE et HL sont modifiés.
- 66 BBC6 Lecture de la position courante du graphique.  
Pas de CE.  
CS : DE contient la coordonnée X.  
HL contient la coordonnée Y.  
AF est modifié.
- 67 BBC9 Positionne l'origine du curseur par défaut.  
CE : DE contient la coordonnée X de l'origine.  
HL contient la coordonnée Y de l'origine.  
CS : AF, BC, DE et HL sont modifiés.
- 68 BBCC Lecture des coordonnées d'origine.  
Pas de CE.  
CS : DE contient la coordonnée X de l'origine.  
HL contient la coordonnée Y de l'origine.
- 69 BBCF Positionnement des bords droit et gauche d'une fenêtre graphique.  
CE : DE contient la coordonnée horizontale d'un bord.  
HL contient la coordonnée horizontale de l'autre bord.  
CS : AF, BC, DE et HL sont modifiés.
- 70 BBD2 Positionnement du sommet et du bas d'une fenêtre graphique.  
CE : DE contient la coordonnée Y d'un des bords.  
HL contient la coordonnée Y de l'autre bord.  
CS : AF, BC, DE et HL sont modifiés.
- 71 BBD5 Lecture des bords droit et gauche d'une fenêtre graphique.  
Pas de CE.  
CS : DE contient la coordonnée X du bord gauche.  
HL contient la coordonnée X du bord droit.  
AF est modifié.
- 72 BBD8 Lecture des bords haut et droit d'une fenêtre graphique.  
Pas de CE.  
CS : DE contient la coordonnée Y du haut de la fenêtre.  
HL contient la coordonnée Y du bas de la fenêtre.  
AF est modifié.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 73 BBDB Effacement d'une fenêtre graphique.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 74 BBDE Positionnement de la couleur d'écriture des graphiques.  
CE : A contient la couleur.  
CS : AF est modifié.
- 75 BBE1 Lecture de la couleur d'écriture graphique (encre).  
Pas de CE.  
CS : A contient la couleur d'écriture.
- 76 BBE4 Positionnement de la couleur du fond (papier).  
CE : A contient le numéro de la couleur.  
CS : AF est modifié.
- 77 BBE7 Lecture de la couleur de fond (papier).  
Pas de CE.  
CS : A contient le numéro de la couleur du papier.
- 78 BBEA Affiche un point à la coordonnée absolue spécifiée.  
CE : DE contient la coordonnée absolue X.  
HL contient la coordonnée absolue Y.  
CS : AF, BC, DE et HL sont effacés.
- 79 BBED Affiche un point à la coordonnée relative spécifiée.  
CE : DE contient la coordonnée relative X.  
HL contient la coordonnée relative Y.  
CS : AF, BC, DE et HL sont modifiés.
- 80 BBF0 Teste un point de coordonnées absolues.  
CE : DE contient la coordonnée absolue X.  
HL contient la coordonnée absolue Y.  
CS : A contient la couleur d'encre pour ce point.  
BC, DE et HL sont modifiés.
- 81 BBF3 Teste un point de coordonnées relatives.  
CE : DE contient la coordonnée relative X.  
HL contient la coordonnée relative Y.  
CS : A contient la couleur d'encre pour ce point.  
BC, DE et HL sont modifiés.
- 82 BBF6 Traçage d'une ligne en coordonnées absolues.  
CE : DE contient la coordonnée absolue X du point d'arrivée.  
HL contient la coordonnée absolue Y du point d'arrivée.  
La ligne sera tirée de la position courante à la position absolue (X,Y).  
CS : AF, BC, DE et HL sont modifiés.
- 83 BBF9 Traçage d'une ligne en coordonnées relatives.  
CE : DE contient la coordonnée relative X du point d'arrivée.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

: HL contient la coordonnée relative Y du point d'arrivée.

La ligne sera tirée de la position courante à la position relative (X,Y).

CS : AF, BC, DE et HL sont modifiés.

- 84 BBFC Ecrit un caractère sur l'écran à la position graphique courante.  
CE : A contient le caractère à écrire.  
CS : AF, BC, DE et HL sont modifiés.

### Le gestionnaire d'écran

- 85 BBFF Initialisation principale du gestionnaire écran, les modes, encres et papiers prennent leurs valeurs par défaut.  
CE : rien.  
CS : AF, BC, DE et HL sont modifiés.
- 86 BC02 Réinitialisation du gestionnaire écran.  
CE : rien.  
CS : AF, BC, DE et HL sont modifiés.
- 87 BC05 Positionne l'OFFSET de départ de l'écran. En modifiant cette valeur, l'écran peut rouler (SCROLLING).  
CE : HL contient l'OFFSET désiré.  
CS : AF et HL sont modifiés.
- 88 BC08 Positionne le point de départ en mémoire vive (RAM) du début de la mémoire écran.  
CE : A contient l'octet le plus significatif de l'adresse de départ.  
CS : AF et HL sont modifiés.
- 89 BC0B Lecture de l'adresse de départ de la mémoire écran et de l'offset.  
CE : rien.  
CS : A contient l'octet le plus significatif de l'adresse de la mémoire écran et HL contient l'OFFSET courant. F est modifié.
- 90 BC0E Positionne l'écran dans un mode précis.  
CE : A contient le numéro du mode.  
CS : AF, BC, DE et HL sont modifiés.
- 91 BC11 Lecture du mode courant.  
CE : rien.  
CS : A contient le numéro du mode, le CARRY et le ZERO sont positionnés en fonction du mode. Mode 0 : C=1 Z=0, mode 1 : C=0 Z=1, mode 2 : C=0 Z=0.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 92 BC14** Effacement de l'écran.  
CE : rien.  
CS : AF, BC, DE et HL sont modifiés.
- 93 BC17** Lecture de la taille de l'écran.  
CE : rien.  
CS : B contient la dernière colonne physique de l'écran, C contient la dernière ligne et AF est modifié.
- 94 BC1A** Calcule l'adresse réelle d'un caractère dont on fournit la position sur l'écran (colonne, ligne).  
CE : H contient la colonne et L contient la ligne.  
CS : HL contient l'adresse mémoire réelle, B contient la taille en octet d'un caractère dans la mémoire et AF est modifié.
- 95 BC1D** Calcul de l'adresse réelle d'un point dont on fournit la position sur l'écran.  
CE : DE contient l'abscisse d'un point (X) et HL contient l'ordonnée du point (Y).  
CS : HL contient l'adresse réelle du point en mémoire, B contient le nombre de points par octet diminué de 1, C contient le masque pour le point, et AF et DE sont modifiés.
- 96 BC20** Calcul de l'adresse réelle de l'octet à droite de l'adresse courante réelle.  
CE : HL contient l'adresse courante.  
CS : HL contient la nouvelle adresse et AF est modifié.
- 97 BC23** Comme 96 (BC20), mais pour l'octet à gauche.
- 98 BC26** Comme 96 (BC20), mais pour la ligne suivante (bas).
- 99 BC29** Comme 96 (BC20), mais pour la ligne précédente (haut).
- 100 BC2C** Conversion d'un numéro d'encre de façon à fournir un masque qui, appliqué à un octet représentant des points, affichera les points de cet octet dans la couleur de l'encre.  
CE : A contient le numéro de l'encre.  
CS : A contient le masque et F est modifié.
- 101 BC2F** Conversion inverse de la précédente.  
CE : A contient le masque.  
CS : A contient le numéro de l'encre et F est modifié.
- 102 BC32** Positionne les couleurs d'une encre.  
CE : A contient le numéro de l'encre.  
B contient la première couleur.  
C contient la seconde couleur.  
CS : AF, BC, DE et HL sont modifiés.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 103 BC35 Lecture des couleurs d'une encre.  
CE : A contient le numéro de l'encre.  
CS : B contient la première couleur.  
C contient la seconde couleur.  
AF, DE et HL sont modifiés.
- 104 BC38 Positionne les couleurs d'affichage du bord.  
CE : B contient la première couleur.  
C contient la seconde couleur.  
CS : AF, BC, DE et HL sont modifiés.
- 105 BC3B Lecture des couleurs de bord.  
CE : rien.  
CS : B contient la première couleur.  
C contient la seconde couleur.  
AF, DE et HL sont modifiés.
- 106 BC3E Positionne la durée de clignotement des couleurs de bord.  
CE : H contient la durée de la première couleur.  
L contient la durée de la seconde couleur.  
CS : AF et HL sont modifiés.
- 107 BC41 Lecture des durées de clignotement des couleurs de bord.  
CE : rien.  
CS : H contient la durée de la première couleur.  
L contient la durée de la seconde.  
AF est modifié.
- 108 BC44 Remplissage d'un rectangle avec une encre.  
CE : A contient le masque correspondant à l'encre.  
H contient le numéro de la colonne de gauche.  
D contient le numéro de la colonne de droite.  
L contient le numéro de la ligne supérieure.  
E contient le numéro de la ligne inférieure.  
CS : AF, BC, DE et HL sont modifiés.
- 109 BC47 Positionnement d'une suite d'octets en mémoire écran dans une encre.  
CE : A contient le masque correspondant à l'encre.  
HL contient l'adresse de la mémoire correspondant au coin supérieur gauche.  
D contient le nombre d'octets.  
E contient le nombre de lignes.  
CS : AF, BC, DE et HL sont modifiés.
- 110 BC4A Inversion de deux couleurs dans un caractère.  
CE : B contient le masque d'une couleur.  
C contient le masque de l'autre couleur.  
H contient le numéro de la colonne.  
L contient le numéro de la ligne.  
CS : AF, BC, DE et HL sont modifiés.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 111 BC4D** Déplace l'écran entier de huit points en haut ou en bas.  
 CE : B=0 pour un déplacement vers le bas.  
       B#0 pour un déplacement vers le haut.  
 CS : AF, BC, DE et HL sont modifiés.
- 112 BC50** Déplace une partie de l'écran de huit points en haut ou en bas.  
 CE : B comme ci-dessus (111).  
       A contient le masque de l'encre pour nettoyer la nouvelle ligne.  
       H contient le numéro de la colonne de gauche.  
       D contient le numéro de la colonne de droite.  
       L contient le numéro de la ligne du haut.  
       E contient le numéro de la ligne du bas.
- 113 BC53** Conversion d'une matrice de caractère de sa forme standard en une série de masques de points en fonction du mode courant.  
 CE : HL contient l'adresse de la matrice.  
       DE contient l'adresse de l'endroit où l'on trouvera le résultat de la conversion.  
 CS : AF, BC, DE et HL sont modifiés.
- 114 BC56** Conversion inverse de la précédente.  
 CE : A contient le masque de l'encre à convertir.  
       H contient la colonne du caractère.  
       L contient la ligne du caractère.  
       DE contient l'adresse où la matrice sera construite.  
 CS : AF, BC, DE et HL sont modifiés.
- 115 BC59** Positionne l'écran pour l'utilisation du mode graphique.  
 CE : A contient le mode (0 = Forçage, 1 = OU exclusif, 2 = ET, 3 = OU).  
 CS : AF, BC, DE et HL sont modifiés.
- 116 BC5C** Ecriture d'un point sur l'écran sans considération de mode défini par la routine précédente (115).  
 CE : B contient le masque de l'encre.  
       C contient le masque du point.  
       HL contient l'adresse mémoire du point.  
 CS : AF est modifié.
- 117 BC5F** Traçage d'une horizontale.  
 CE : A contient le masque de l'encre.  
       DE contient l'abscisse de départ.  
       BC contient l'abscisse d'arrivée.  
       HL contient l'ordonnée.  
 CS : AF, BC, DE et HL sont modifiés.



- 118 BC62 Traçage d'une verticale.  
CE : A contient le masque de l'encre.  
DE contient l'abscisse de la ligne.  
HL contient l'ordonnée de départ.  
BC contient l'ordonnée d'arrivée.  
CS : AF, BC, DE et HL sont modifiés.

### Le gestionnaire cassette

- 119 BC65 Initialisation du gestionnaire cassette.  
CE : rien.  
CS : AF, BC, DE et HL sont modifiés.
- 120 BC68 Positionnement de la vitesse d'écriture.  
CE : HL contient la longueur de la moitié d'un bit à zéro.  
A contient la précompensation à appliquer.  
CS : AF et HL sont modifiés.
- X 121 BC6B Autorise ou interdit l'affichage des messages.  
CE : A=0 autorisé. A#0 interdit.  
CS : AF est modifié.
- λ 122 BC6E Mise en route du moteur de la cassette.  
CE : rien.  
CS : si le moteur est OK, le CARRY est vrai ; si on a poussé sur ESC, le CARRY est faux, A contient l'état précédent du moteur.
- X 123 BC71 Arrêt du moteur.  
CE : rien.  
CS : comme ci-dessus (122).
- 124 BC74 Repositionne le moteur dans son état précédent.  
CE : A contient l'état précédent du moteur.  
CS : comme ci-dessus (122).
- 125 BC77 Positionne le tampon pour la lecture et lit le premier bloc.  
CE : B contient la longueur du nom du fichier.  
HL contient l'adresse du nom du fichier.  
ENTRE DE contient l'adresse du tampon (2K).  
CS : si OK, le CARRY est vrai et le ZERO est faux. HL contient l'adresse du tampon qui contient l'entête, DE contient l'adresse des données, BC contient la longueur du fichier et A contient le type de fichier. Si le STREAM (flux) est déjà utilisé, le CARRY est faux et A, BC, DE et HL sont modifiés. Si on pousse sur ESC, le CARRY est faux et le ZERO est vrai. AF, BC, DE et HL sont modifiés. Enfin, dans tous les cas, IX est modifié.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 126 BC7A** Ferme le fichier.  
CE : rien.  
CS : si OK, le CARRY est vrai, sinon le CARRY est faux.  
AF, BC, DE et HL sont modifiés dans les deux cas.
- 127 BC7D** Abandonne la lecture et ferme le fichier.  
CE : rien.  
CS : AF, BC, DE et HL sont modifiés.
- 128 BC80** Lecture d'un octet.  
CE : rien.  
CS : si OK, le CARRY est vrai, ZERO est faux et A contient le caractère lu. Si on rencontre la fin de fichier (EOF), le CARRY est faux, ZERO est faux et A est modifié. Si on pousse ESC, le CARRY est faux, ZERO est vrai et A est modifié. Dans tous les cas, IX est modifié.
- 129 BC83** Lecture d'un fichier et écriture en mémoire.  
CE : HL contient l'adresse d'écriture.  
CS : comme 128 pour les sémaphores CARRY et ZERO. En outre, HL contient le point d'entrée si la lecture est correcte. Dans tous les cas, AF, BC, DE, HL et IX sont modifiés.
- 130 BC86** Met le dernier caractère lu par la routine 128 dans le tampon de lecture.  
Pas de CE.  
Pas de CS.
- 131 BC89** Teste si on a atteint la fin de fichier.  
Pas de CE.  
CS : si on a atteint la fin de fichier, le CARRY est faux et le ZERO est faux. Si on n'a pas atteint la fin de fichier, le CARRY est vrai et le ZERO est faux. Si l'utilisateur a poussé sur ESC (BREAK), le CARRY est faux et le ZERO est vrai. De toute façon, AF et IX sont modifiés.
- 132 BC8C** Ouverture d'un fichier en sortie.  
CE : B contient la longueur du nom du fichier.  
HL contient l'adresse du nom du fichier.  
DE contient l'adresse d'un tampon de 2K disponible pour le fichier.  
CS : si le fichier a été ouvert correctement, le CARRY est vrai, le ZERO est faux et HL contient l'adresse d'un tampon qui contient l'en-tête qui sera écrite au début de chaque bloc de données. Si l'utilisateur a poussé sur ESC, le CARRY est faux et le ZERO est vrai. Si le tampon est déjà utilisé, le CARRY est faux et le ZERO est faux. De toute façon, AF, BC, DE, HL et IX sont modifiés.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 133 BC8F Fermeture propre d'un fichier en sortie.  
Pas de CE.  
CS : si la fermeture est OK, le CARRY est vrai et le ZERO est faux. Si le fichier n'était pas ouvert, le CARRY est faux et le ZERO est faux. Si on a poussé sur ESC, le CARRY est faux et le ZERO est vrai. De toute façon, AF, BC, DE, HL et IX sont modifiés.
- 134 BC92 Fermeture immédiate d'un fichier en sortie.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 135 BC95 Ecriture d'un caractère sur un fichier de sortie.  
CE : A contient le caractère à écrire.  
CS : si l'écriture est correcte, le CARRY est vrai et le ZERO est faux. Si le fichier n'était pas ouvert, le CARRY est faux et le ZERO est faux. Si on a poussé sur ESC, le CARRY est faux et le ZERO est vrai. De toute façon, AF et IX sont modifiés.
- 136 BC98 Ecriture directe du contenu d'une mémoire vers un fichier de sortie.  
CE : HL contient l'adresse mémoire.  
DE contient le nombre d'octets à écrire.  
BC contient le point d'entrée. *AUTO-RUN*  
A contient le type de fichier.  
CS : voir précédent, mais AF, BC, DE, HL et IX sont modifiés.
- 137 BC9B Génère le catalogue de la cassette.  
CE : DE contient l'adresse d'un tampon de 2K disponible.  
CS : si la lecture a eu lieu correctement, le CARRY est vrai et le ZERO est faux. Si le tampon est occupé, le CARRY est faux et le ZERO est faux. Si une erreur a eu lieu, le CARRY est faux et le ZERO est vrai. De toute façon, AF, BC, DE, HL et IX sont modifiés.
- 138 BC9E Ecrit un enregistrement sur cassette.  
CE : HL contient l'adresse des données à écrire.  
DE contient le nombre d'octets à écrire.  
A contient le caractère de synchronisation.  
CS : si l'enregistrement s'est déroulé correctement, le CARRY est vrai. Sinon, le CARRY est faux et A contient un code d'erreur. De toute façon, AF, BC, DE, HL et IX sont modifiés.
- 139 BCA1 Lit un enregistrement sur cassette.  
CE : HL contient l'adresse où seront écrites les données.  
DE contient le nombre d'octets à lire.  
A contient le caractère de synchronisation.



## TABLES DES POINTS D'ENTREE DES ROUTINES SYSTEME

CS : si la lecture s'est déroulée correctement, le CARRY est vrai. Sinon, le CARRY est faux et A contient un code d'erreur. De toute façon, AF, BC, DE, HL et IX sont modifiés.

- 140 BCA4 Compare un enregistrement sur cassette avec le contenu de la mémoire.  
CE : HL contient l'adresse des données à comparer.  
DE contient le nombre d'octets à comparer.  
A contient le caractère de synchronisation.  
CS : si la comparaison est correcte, le CARRY est vrai. Sinon, le CARRY est faux et A contient un code d'erreur. De toute façon, AF, BC, DE, HL et IX sont modifiés.

### Le gestionnaire sonore

- 141 BCA7 Initialise le gestionnaire sonore.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 142 BCAA Ajoute un son à une queue sonore.  
CE : HL contient l'adresse du programme sonore qui doit se trouver dans les 32K de mémoire vive centrale.  
CS : si le son a pu être ajouté à la queue sonore, le sémaphore de CARRY est vrai et HL est modifié. Si toutes les queues sonores sont remplies et que le son n'a pu être ajouté à l'une d'entre elles, le sémaphore de CARRY est faux et HL est préservé. De toute façon, AF, BC, DE et IX sont modifiés. Les autres registres sont préservés.
- 143 BCAD Vérifie s'il y a de la place dans une queue sonore.  
CE : A contient le numéro du canal à tester.  
Il vaut 0 si l'on veut tester le canal A.  
Il vaut 1 si l'on veut tester le canal B.  
Il vaut 2 si l'on veut tester le canal C.  
CS : A contient l'état du canal testé.  
F, BC, DE et HL sont modifiés.
- 144 BCB0 Prépare l'exécution d'une interruption lorsqu'une queue sonore est vide.  
CE : A contient le numéro du canal à tenir prêt (0=A, 1=B, 2=C).  
HL contient l'adresse du programme d'interruption.  
CS : AF, BC, DE et HL sont modifiés.
- 145 BCB3 Permet de relâcher les sons arrêtés sur chaque canal (voir routine suivante).  
CE : A contient le numéro du canal à relâcher (0=A, 1=B, 2=C).  
CS : AF, BC, DE et HL sont modifiés.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 146 BCB6** Arrêt de tous les sons.  
Pas de CE.  
CS : si un son était actif, le sémaphore de CARRY serait vrai. Si aucun son n'était actif, le CARRY serait faux. De toute façon, AF, BC et HL sont modifiés.
- 147 BCB9** Redémarre tous les sons qui ont été arrêtés par la routine précédente.  
Pas de CE.  
CS : AF, BC, DE et IX sont modifiés.
- 148 BCBC** Etablissement d'une des 15 enveloppes d'amplitude programmable.  
CE : A contient le numéro d'enveloppe.  
HL contient l'adresse des données concernant l'amplitude.  
CS : si une enveloppe a été établie, le CARRY est vrai, HL contient l'adresse du bloc de données augmentée de 16, A et BC sont modifiés. Si le numéro d'enveloppe est incorrect, le CARRY est faux, A, B et HL sont modifiés. De toute façon, F et DE sont modifiés.
- 149 BCBF** Etablissement d'une des 15 enveloppes de fréquence programmable.  
CE : A contient un numéro d'enveloppe.  
HL contient l'adresse des données concernant la fréquence.  
CS : si l'enveloppe de fréquence a bien été établie, le CARRY est vrai, HL contient l'adresse du bloc de données augmentée de 16, et A et BC sont modifiés. Si le numéro d'enveloppe est incorrect, le CARRY est faux et A, BC et HL sont préservés. De toute façon, F et DE sont modifiés.
- 150 BCC2** Fournit l'adresse d'une enveloppe d'amplitude.  
CE : A contient un numéro d'enveloppe.  
CS : si l'enveloppe a bien été trouvée, le CARRY est vrai, HL contient l'adresse de l'enveloppe d'amplitude et BC contient la longueur d'une enveloppe. Si le numéro d'enveloppe est incorrect, le CARRY est faux, HL est modifié et BC est préservé. De toute façon, AF est modifié.
- 151 BCC5** Fournit l'adresse d'une enveloppe de ton.  
CE : A contient un numéro d'enveloppe.  
CS : si l'enveloppe a bien été trouvée, le CARRY est vrai, HL contient l'adresse de l'enveloppe de ton et BC contient la longueur de l'enveloppe. Si le numéro d'enveloppe est incorrect, le CARRY est faux, HL est modifié et BC est préservé. De toute façon, AF est modifié.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

## Le noyau (kernel)

- 152 BCC8** Nettoie toutes les files d'interruption, les chronomètres...  
Pas de CE.  
CS : B contient l'adresse de sélection de la ROM s'il y en a une.  
DE contient le point d'entrée à l'intérieur de la ROM.  
C contient l'adresse de sélection de la ROM pour un programme en RAM.  
AF et HL sont modifiés.
- 153 BCCB** Trouve et initialise toutes les ROMs de second plan.  
CE : DE contient l'adresse du premier octet utilisable.  
HL contient l'adresse du dernier octet utilisable.  
CS : DE contient l'adresse du nouveau premier octet utilisable.  
HL contient l'adresse du nouveau dernier octet utilisable.  
AF et BC sont modifiés.
- 154 BCCE** Initialise une ROM de second plan.  
CE : C contient l'adresse de sélection de la ROM à initialiser.  
DE contient l'adresse du premier octet utilisable.  
HL contient l'adresse du dernier octet utilisable.  
CS : DE contient l'adresse du nouveau premier octet utilisable.  
HL contient l'adresse du nouveau dernier octet utilisable.  
AF et B sont modifiés.
- 155 BCD1** Introduit un RSX (extension résidente du système) dans le logiciel interne.  
CE : BC contient l'adresse de la table des commandes RSX.  
HL contient l'adresse de quatre octets disponibles en RAM pour le noyau.  
CS : DE est modifié.
- 156 BCD4** Recherche un RSX dans les ROMs pour effectuer une commande.  
CE : HL contient l'adresse où se trouve le nom de la commande à rechercher.  
CS : si un RSX a été trouvé, le CARRY est vrai, C contient l'adresse de sélection de la ROM et HL contient l'adresse de la routine. Si la commande n'a pas été trouvée, le CARRY est faux. De toute façon, AF, BC et DE sont modifiés.



# TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 157 BCD7** Initialise et dépose un bloc d'événement dans la liste des blocs à activer lors d'une interruption en provenance du CRT.  
CE : HL contient l'adresse du bloc d'événement.  
B contient la classe de l'événement.  
C contient l'adresse de sélection de la ROM.  
DE contient l'adresse de la routine de l'événement.  
CS : AF, DE et HL sont modifiés.
- 158 BCDA** Dépose un bloc d'événement dans la liste des blocs à activer lors d'une interruption en provenance du CRT.  
CE : HL contient l'adresse du bloc d'événement.  
CS : AF, DE et HL sont modifiés.
- 159 BCDD** Enlève un bloc d'événement dans la liste des blocs à activer lors d'une interruption en provenance du CRT.  
CE : HL contient l'adresse du bloc d'événement.  
CS : AF, DE et HL sont modifiés.
- 160 BCE0** Initialise et dépose un bloc d'événement dans la liste des blocs à activer lors d'une interruption rapide (1/300° de seconde).  
CE : HL contient l'adresse du bloc.  
B contient la classe de l'événement.  
C contient l'adresse de sélection de la ROM.  
DE contient l'adresse de la routine de l'événement.  
CS : AF, DE et HL sont modifiés.
- 161 BCE3** Pose un bloc d'événement dans la liste des blocs à activer lors d'une interruption rapide (1/300° de seconde).  
CE : HL contient l'adresse du bloc d'événement.  
CS : AF, DE et HL sont modifiés.
- 162 BCE6** Enlève un bloc d'événement de la liste des blocs à activer lors d'une interruption rapide (1/300° de seconde).  
CE : HL contient l'adresse du bloc d'événement.  
CS : AF, DE et HL sont modifiés.
- 163 BCE9** Dépose un bloc d'événement dans la liste des blocs à activer lors d'une interruption normale (1/50° de seconde).  
CE : HL contient l'adresse du bloc d'événement.  
DE contient la valeur initiale du compteur.  
BC contient la valeur de recharge du compteur lorsque celui-ci atteint 0.  
CS : AF, BC, DE et HL sont modifiés.
- 164 BCEC** Enlève un bloc d'événement de la liste des blocs à activer lors d'une interruption normale (1/50° de seconde).  
CE : HL contient l'adresse du bloc d'événement.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- CS : si le bloc a été trouvé dans la liste, le CARRY est vrai et DE contient le compteur, sinon le CARRY est faux. De toute façon, AF, DE et HL sont modifiés.
- 165 BCEF** Initialise un bloc d'événement.  
 CE : HL contient l'adresse du bloc d'événement.  
       B contient la classe d'événement.  
       C contient l'adresse de sélection de la ROM  
       DE contient l'adresse de la routine d'événement.  
 CS : HL contient l'adresse du bloc d'événement augmentée de 7.
- 166 BCF2** Actionne un bloc d'événement.  
 CE : HL contient l'adresse du bloc d'événement.  
 CS : AF, BC, DE et HL sont modifiés.
- 167 BCF5** Nettoie toutes les files d'attente des événements temporisés.  
 Pas de CE.  
 CS : AF et HL sont modifiés.
- 168 BCF8** Enlève un événement temporisé hors d'une file d'attente.  
 CE : HL contient l'adresse du bloc d'événement.  
 CS : AF, BC, DE et HL sont modifiés.
- 169 BCFB** Lecture de l'événement suivant dans la file d'attente.  
 Pas de CE.  
 CS : s'il y a un événement à traiter, le CARRY est vrai et HL contient l'adresse du bloc d'événement. A peut contenir le code de priorité de l'événement précédent. S'il n'y a pas d'événement à traiter, le CARRY est faux. De toute façon, AF, DE et HL sont modifiés.
- 170 BCFE** Traite une routine d'événement.  
 CE : HL contient l'adresse du bloc d'événement.  
 CS : AF, BC, DE et HL sont modifiés.
- 171 BD01** Termine le traitement d'un événement.  
 CE : HL contient l'adresse du bloc d'événement.  
       A contient le code de priorité de l'événement précédent.  
 CS : AF, BC, DE et HL sont modifiés.
- 172 BD04** Interdit les événements temporisés normaux.  
 Pas de CE.  
 CS : HL est modifié.
- 173 BD07** Autorise les événements temporisés normaux.  
 Pas de CE.  
 CS : HL est modifié.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 174 BDOA Interdit un événement particulier.  
CE : HL contient l'adresse du bloc d'événement.  
CS : AF est modifié.
- 175 BD0D Donne le temps écoulé en 1/300° de seconde.  
Pas de CE.  
CS : DEHL contient le temps écoulé sur quatre octets.

### Interfaçage avec le matériel

- 176 BD10 Positionne le compteur écoulé à une valeur précise.  
CE : DEHL contient la valeur sur quatre octets exprimée en 1/300° de seconde.  
CS : AF est modifié.
- 177 BD13 Charge un programme en RAM et le lance.  
CE : HL contient l'adresse de la routine à appeler pour charger le programme.  
CS : qui peut savoir ce que va faire le programme ?
- 178 BD16 Lance un programme dans une ROM de second plan.  
CE : HL contient le point d'entrée.  
C contient l'adresse de sélection de la ROM.  
CS : indéterminable.
- 179 BD19 Attend jusqu'à ce que le CRT génère un signal qui indique le début du balayage vertical.  
Pas de CE.  
Pas de CS.
- 180 BD1C Positionne le mode écran.  
CE : A contient le mode (0, 1 ou 2).  
CS : AF est modifié.
- 181 BD1F Positionne l'offset de mémoire écran.  
CE : A contient la base du nouvel écran.  
HL contient l'offset.  
CS : AF est modifié.
- 182 BD22 Met toutes les encres dans la même couleur pour donner un effet d'effacement d'écran.  
CE : DE contient l'adresse d'un vecteur encre.  
CS : AF est modifié.
- 183 BD25 Positionne les couleurs des encres et du bord.  
CE : DE contient l'adresse d'un vecteur encre.  
CS : AF est modifié.
- 184 BD28 Réinitialise le détournement vers l'imprimante.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.



## TABLE DES POINTS D'ENTREE DES ROUTINES SYSTEME

- 185 BD2B Envoie un caractère à l'imprimante (avec possibilité de retour si l'imprimante est occupée).  
CE : A contient le caractère à envoyer.  
CS : si le caractère a été envoyé, le CARRY est vrai. Si l'imprimante est restée occupée trop longtemps, le CARRY est faux. De toute façon, AF est modifié.
- 186 BD2E Teste si l'imprimante est occupée (BUSY).  
Pas de CE.  
CS : si l'imprimante est occupée, le CARRY est vrai, sinon il est faux.
- 187 BD31 Envoie un caractère à l'imprimante (elle ne doit pas être occupée).  
CE : A contient le caractère à envoyer.  
CS : CARRY vrai, AF modifié.
- 188 BD34 Envoie une donnée dans un registre du PSG.  
CE : A contient le numéro du registre.  
C contient la donnée.  
CS : AF et BC sont modifiés.

### Le bloc de saut

- 189 BD37 Repositionne les blocs de saut standard.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.



Les vecteurs d'indirection permettent à l'utilisateur d'intercepter et d'altérer un certain nombre d'actions du logiciel système sans devoir réécrire des routines complètes.

*Remarque* : les adresses données ici ne sont pas des points d'entrée mais bien des appels intérieurs que l'on peut dérouter.

- 1 **BDCD** Place le caractère du curseur à l'écran.  
Pas de CE.  
CS : AF est modifié.
- 2 **BDD0** Enlève le caractère du curseur de l'écran.  
Pas de CE.  
CS : AF est modifié.
- 3 **BDD3** Ecrit un caractère sur l'écran.  
CE : A contient le caractère à écrire.  
H contient le numéro de colonne.  
L contient le numéro de ligne.  
CS : AF, BC, DE et HL sont modifiés.
- 4 **BDD6** Lit un caractère sur l'écran.  
CE : H contient le numéro de colonne.  
L contient le numéro de ligne.  
CS : si le caractère est trouvé, le CARRY est vrai et A contient le caractère, sinon le CARRY est faux et A contient 0. De toute façon, AF, BC, DE et HL sont modifiés.
- 5 **BDD9** Ecriture d'un caractère ou traitement d'un code de contrôle.  
CE : A contient le caractère ou le code de contrôle.  
CS : AF, BC, DE et HL sont modifiés.
- 6 **BDDC** Dessine un point.  
CE : DE contient l'abscisse du point.  
HL contient l'ordonnée.  
CS : AF, BC, DE et HL sont modifiés.
- 7 **BDDF** Teste un point.  
CE : DE contient l'abscisse du point.  
HL contient l'ordonnée.  
CS : A contient l'encre du point spécifié.  
A, BC, DE et HL sont modifiés.
- 8 **BDE2** Trace une ligne à partir de la position courante.  
CE : DE contient l'abscisse du point final.  
HL contient l'ordonnée du point final.  
CS : AF, BC, DE et HL sont modifiés.
- 9 **BDE5** Lecture d'un point dans la mémoire écran et décodage de son encre.  
CE : HL contient l'adresse écran du point.  
C contient le masque pour le point.



## LES VECTEURS D'INDIRECTION

- CS : A contient l'encre décodée du point spécifié.  
AF est modifié.
- 10 BDE8 Ecrit un ou des points dans le mode graphique courant.  
CE : HL contient l'adresse écran du ou des points.  
C contient le masque pour le ou les points.  
B contient l'encre encodée.  
CS : AF est modifié.
- 11 BDEB Nettoie l'écran avec l'encre 0.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.
- 12 BDEE Teste la touche ESC (BREAK).  
CE : interruption interdite et C contient l'état des touches CTRL et SHIFT.  
CS : AF et HL sont modifiés.
- 13 BDF1 Ecrit un caractère sur l'imprimante.  
CE : A contient le caractère.  
CS : si le caractère a bien été écrit, le CARRY est vrai. Si l'imprimante est restée occupée trop longtemps, le CARRY est faux. De toute façon, AF et BC sont modifiés.



En dehors des points d'entrée principaux du logiciel système, il existe quelques routines qui gèrent la sélection et l'état des ROM. Ces routines ne peuvent en aucun cas être modifiées par l'utilisateur.

## Les vecteurs en haut de mémoire

- 1 B900 Sélectionne la ROM supérieure.  
Pas de CE.  
CS : A contient l'état précédent de la ROM.  
AF est modifié.
- 2 B903 Coupe la ROM supérieure pour resélectionner la RAM.  
Pas de CE.  
CS : A contient l'état précédent de la ROM.  
AF est modifié.
- 3 B906 Sélectionne la ROM inférieure.  
Pas de CE.  
CS : A contient l'état précédent de la ROM.  
AF est modifié.
- 4 B909 Coupe la ROM inférieure pour resélectionner la RAM.  
Pas de CE.  
CS : A contient l'état précédent de la ROM.  
AF est modifié.
- 5 B90C Restaure l'état antérieur d'une ROM.  
CE : A contient l'état antérieur de la ROM.  
CS : AF est modifié.
- 6 B90F Sélectionne une ROM supérieure particulière.  
CE : C contient l'adresse de sélection de la ROM requise.  
CS : C contient l'adresse de sélection de la ROM précédente.  
B contient l'état de la ROM précédente.  
AF est modifié.
- 7 B912 Demande quelle ROM est sélectionnée.  
Pas de CE.  
CS : A contient l'adresse de sélection de la ROM courante.
- 8 B915 Demande la classe et la version d'une ROM.  
CE : C contient l'adresse de sélection de la ROM à interroger.  
CS : A contient la classe de la ROM.  
H contient le numéro de version.  
L contient un numéro de marque.  
B et F sont modifiés.



## LES VECTEURS NOYAU ET LES RESTART

- 9 B918 Resélectionne la ROM supérieure précédemment sélectionnée.  
CE : C contient l'adresse de sélection de la ROM précédente.  
B contient l'état de la ROM précédente.  
CS : BC est modifié.
- 10 B91B Exécute un déplacement de bloc avec incrémentation (LDIR) avec les deux ROM déconnectées.  
CE : BC, DE et HL sont programmés comme pour un LDIR normal.  
CS : BC, DE, HL et F sont dans le même état qu'après un LDIR normal.
- 11 B91E Comme ci-dessus, mais avec décrémentation (LDDR).
- 12 B921 Teste si un événement avec une priorité supérieure à l'événement courant se produit.  
Pas de CE.  
CS : si un événement avec une priorité supérieure se produit, le CARRY est vrai, sinon il est faux. AF est modifié.

### Les vecteurs en bas de mémoire

- 1 0000 RST 0. Réinitialisation complète de la machine comme à l'allumage.  
Pas de CE.  
CS : on n'en sort pas !
- 2 0008 RST 8. Lancement d'une routine en ROM ou en RAM inférieure. Ce sont les deux octets qui suivent le RST qui contiennent l'adresse d'exécution et l'état de la ROM supérieure. Voir format au chapitre 4.11 (page 135).  
CE : tous les registres sont passés à la routine sans être affectés.  
CS : ne dépend que de la routine elle-même.
- 3 000B Lancement d'une routine en ROM ou en RAM inférieure.  
CE : HL contient l'adresse inférieure de la routine.  
CS : ne dépend que de la routine.
- 4 000E Saute à l'adresse contenue dans BC.  
CE : BC contient l'adresse.  
CS : ne dépend que de la routine.
- 5 0010 RST 10. Appel à une sous-routine d'une ROM secondaire. Ce sont les deux octets qui suivent le RST qui contiennent l'adresse d'exécution et l'adresse de sélection de la ROM. Voir format au chapitre 4.11 (page 135).



## LES VECTEURS NOYAU ET LES RESTART

CE : les registres sont passés à la routine sans être affectés, excepté IY.

CS : dépend de la routine.

- 6 0013 Appel à une sous-routine d'une ROM secondaire. L'adresse est contenue dans HL.  
CE : HL contient l'adresse et les registres sont passés à la routine sans être affectés, excepté IY.  
CS : dépend de la routine.
- 7 0016 Saut à l'adresse contenue dans DE.  
CE : DE contient l'adresse.  
CS : dépend de la routine.
- 8 0018 RST 18. Appel à une sous-routine en RAM ou en ROM. Ce sont les deux octets qui suivent qui contiennent l'adresse de la sous-routine.  
CE : tous les registres sont passés à la sous-routine, excepté IY.  
CS : dépend de la sous-routine.
- 9 001B Appel à une sous-routine en RAM ou en ROM avec l'adresse dans HL.  
CE : HL contient l'adresse.  
C contient l'octet de sélection de la ROM ou de la RAM.  
Tous les registres sont passés à la routine, sauf IY.  
CS : dépend de la routine.
- 10 001E Saut à l'adresse contenue dans HL.  
CE : HL contient l'adresse.  
CS : dépend de la routine.
- 11 0020 RST 20. Charge, dans l'accumulateur, le contenu de la RAM dont l'adresse se trouve dans HL, quel que soit l'état des ROM.  
CE : HL contient l'adresse.  
CS : A contient la valeur lue.
- 12 0023 Appelle une sous-routine en RAM ou en ROM, HL contient l'adresse où se trouve l'adresse de la sous-routine.  
CE : HL contient l'adresse où se trouve l'adresse de la sous-routine. Tous les registres sont passés à la sous-routine, sauf IY.  
CS : dépend de la sous-routine.
- 13 0028 RST 28. Saut à une adresse en ROM inférieure. Les deux octets qui suivent le RST contiennent l'adresse.  
CE : tous les registres sont préservés.  
CS : dépend de la sous-routine.



## LES VECTEURS NOYAU ET LES RESTART

- 14 0030 RST 30. Réservé à l'utilisateur.  
L'utilisateur peut faire ce qu'il veut avec les octets  
compris entre 30 et 37 inclus.
- 15 0038 RST 38. Point d'entrée des interruptions générées par  
le matériel.  
Pas de CE.  
CS : tous les registres sont préservés.
- 16 003B Routine de traitement des interruptions extérieures.  
Pas de CE.  
CS : AF, BC, DE et HL sont modifiés.



Les routines mathématiques contenues dans la ROM inférieure doivent être appelées souvent depuis la ROM Basic pour effectuer toutes les fonctions Basic de calcul (+, \*, /, sin, cos,...).

Une série de vecteurs a été créée pour faciliter cet appel.

Les fonctions mathématiques du Basic fonctionnent sur un accumulateur virtuel de six octets situé de B0C1 à B0C6. B0C1 contient le type de la variable (2=entier, 3=chaîne, 5=réel).

Une variable entière est codée sur deux octets en binaire signé.

Une variable réelle est plus complexe. Elle est représentée par cinq octets, suivant un codage binaire particulier :

- exprimer le nombre en binaire ;
- compter le nombre de chiffres significatifs situés avant la virgule et lui ajouter 128 (80H). On obtient l'octet 5 ;
- supprimer le premier bit de gauche et convertir les sept autres bits en décimal. Si le nombre est négatif, ajouter 128 (80H). On obtient l'octet 4 ;
- pour obtenir les octets 3, 2 et 1, prendre les bits suivants par tranche de 8 et les convertir en décimal.

*Exemple* : codage de la variable réelle : -2527.

2527 s'écrit en binaire 1 0011101 1111 (12 chiffres)

Octet 5 = 8C car :  $128 + 12 = 140 = 8C$

Octet 4 = 9D car : prendre les sept bits suivants : 0011101 = 29 = 1D. Le nombre étant négatif, ajouter 128 :  $29 + 128 = 157 = 9D$ .

Octet 3 = F0 car : les huit bits suivants sont 1111 0000 = 240 = F0.

Octet 2 et octet 1 = 00 car il n'y a plus de bits.

-2527 se code donc : 00 00 F0 9D 8C.

Adresse vecteur	Adresse réelle	Signification
BD3D	2E18	Copie les cinq octets pointés par DE vers la zone pointée par HL et passe dans A le contenu de l'octet qui se trouve à l'adresse HL-1 (type de variable).
BD40	2E29	Copie le contenu de A dans les cinq octets pointés par DE.
BD43	2E55	Conversion du nombre binaire pointé par HL en nombre au format de l'accumulateur (5 octets).



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES

Adresse vecteur	Adresse réelle	Signification
BD46	2E66	Transforme la valeur contenue dans les cinq octets pointés par HL en un entier contenu dans HL.
BD49	2E8E	Transforme la valeur contenue dans les cinq octets pointés par HL en un entier contenu dans les deux premiers octets pointés par HL.
BD4C	2EA1	Réalise la fonction FIX.
BD4F	2EAC	Réalise la fonction INT.
BD52	2EB6	Routine SGN utilisée par STR\$ et PRINT.
BD55	2F1D	Routine de multiplication par 10 exposant A.
BD58	333F	Addition de deux réels. HL pointe sur une zone de cinq octets représentant un nombre au format réel (appelée ACCUM1). DE pointe sur une autre zone de cinq octets (appelée ACCUM2). A l'issue de la routine, HL pointe toujours sur ACCUM1 et ACCUM1 contient la valeur de ACCUM1+ACCUM2.
BD5B	3337	Soustraction de deux réels. HL pointe sur une zone de cinq octets représentant un nombre au format réel (appelée ACCUM1). DE pointe sur une autre zone de cinq octets (appelée ACCUM2). A l'issue de la routine, HL pointe toujours sur ACCUM1 et ACCUM1 contient la valeur de ACCUM1-ACCUM2.
BD5E	333B	Soustraction de deux réels. Comme ci-dessus, mais ACCUM1 contient la valeur de ACCUM2-ACCUM1.
BD61	3415	Multiplication de deux réels. Comme ci-dessus, mais ACCUM1 contient la valeur de ACCUM1 * ACCUM2.
BD64	349E	Division de deux réels. Comme ci-dessus, mais ACCUM1 contient la valeur de ACCUM1 / ACCUM2.
BD67	3578	Ajoute A au dernier octet du nombre pointé par HL.
BD6A	359A	Comparaison de deux réels : si ACCUM1 > ACCUM2, alors A=1 si ACCUM1 < ACCUM2, alors A=255 si ACCUM1 = ACCUM2, alors A=0.
BD6D	35F8	Négation d'un réel. HL pointe sur ACCUM1 qui contient la valeur de -ACCUM1.



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES

Adresse vecteur	Adresse réelle	Signification
BD70	35E8	Teste le réel contenu dans ACCUM1. HL pointe sur ACCUM1. Si $ACCUM1 > 0$ , alors $A=1$ . Si $ACCUM1 < 0$ , alors $A=255$ . Si $ACCUM1 = 0$ , alors $A=0$ .
BD73	31AE	Positionnement du mode de calcul d'angles en degrés ou en radians. Si $A=0$ , on est en mode RADIANS. Si $A \neq 0$ , on est en mode DEGRES.
BD76	31A3	En sortie, la zone pointée par HL en entrée contient la constante PI.
BD79	310A	Extraction de la racine carrée d'un nombre réel. En entrée, HL pointe sur une zone de cinq octets qui contient un nombre. En sortie, cette zone contient la racine carrée du nombre.
BD7C	310D	Calcul de l'exposant d'un nombre réel. HL pointe sur ACCUM1 qui contient le nombre et DE pointe sur ACCUM2 qui contient l'exposant. En sortie, ACCUM1 contient la valeur de ACCUM1 exposant ACCUM2.
BD7F	3014	Calcul du logarithme népérien d'un nombre réel. HL pointe sur ACCUM1 qui contient le nombre en entrée. En sortie, ACCUM1 contient la valeur du logarithme népérien du nombre.
BD82	300F	Calcul du logarithme en base 10 d'un nombre réel. HL pointe sur ACCUM1 qui contient le nombre en entrée. En sortie, ACCUM1 contient la valeur du logarithme décimal du nombre.
BD85	3090	Calcul de l'exponentielle d'un nombre. HL pointe sur ACCUM1 qui, en sortie, contient la valeur de l'exponentielle du nombre.
BD88	31BC	Calcul du sinus d'un angle.
BD8B	31B2	Calcul du cosinus d'un angle.
BD8E	3231	Calcul de la tangente d'un angle.
BD91	3241	Calcul de l'arctangente d'un angle.
BD94	2E5E	Routine d'évaluation.
BD97	2F94	Routine de chargement de B8E4 et B8E6 à l'initialisation.



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES

Adresse vecteur	Adresse réelle	Signification
BD9A	2FA1	Routine utilisée pour la génération de nombres aléatoires.
BD9D	2FB7	Idem.
BDA0	2FE6	Idem.
BDA3	3708	Manipulation avec HL.
BDA6	370E	Pousse 0 dans B, 0 dans E et 2 dans C.
BDA9	3715	Manipulation avec HL.
BDAC	3728	Addition de deux nombres entiers. $HL=HL+DE$ . A=FF en cas de débordement.
BDAF	3731	Soustraction de deux nombres entiers. $HL=HL-DE$ . A=FF en cas de débordement.
BDB2	3730	Soustraction de deux nombres entiers. $HL=DE-HL$ . A=FF en cas de débordement.
BDB5	3739	Multiplication de deux nombres entiers. $HL=HL*DE$ . A=FF en cas de débordement.
BDB8	377A	Division de deux nombres entiers. $HL=HL/DE$ . DE contient le reste de la division en sortie.
BDBB	3781	Reste de la division de deux entiers. HL = reste de $HL/DE$ .
BDBE	3750	Manipulation particulièrement ténébreuse entre HL et DE.
BDC1	378C	Routine utilisée lors de l'instruction PRINT.
BDC4	37E9	Comparaison de deux nombres entiers. Si $HL > DE$ , alors A=1. Si $HL < DE$ , alors A=FF. Si $HL = DE$ , alors A=0.
BDC7	37D4	Négation d'un nombre entier. En sortie, $HL=-(HL)$ .
BDCA	37E0	Test de HL. Si $HL > 0$ , alors A=1. Si $HL < 0$ , alors A=255. Si $HL = 0$ , alors A=0.



## LES PRINCIPALES VARIABLES SYSTEME

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
AC00	26	26 fois le code C9 (RET).
AC1C	1	Sémaphore : 0=auto, 1=pas auto.
AC1D	2	Numéro de la ligne courante (utilisé par AUTO).
AC1F	2	Valeur de l'incrément entre deux lignes (AUTO).
AC24	1	Utilisé par l'instruction WIDTH.
AC26	2	Utilisé par l'instruction NEXT.
AC2C	2	Utilisé par l'instruction FOR.
AC2E	2	Utilisé par le couple d'instructions WHILE-WEND.
AC30	11	Utilisés par l'instruction ON...GOTO.
ACA4	1	Utilisé par l'instruction EVERY (valeur).
ACA5	256	Tampon d'entrée clavier.
AD81	2	Numéro de ligne pour l'instruction ON ERROR.
ADA6	2	Pointeur pour l'instruction RESUME.
ADA8	2	Utilisés pour le traitement d'erreur.
ADAA	1	Numéro d'erreur.
ADAB	2	Adresse du dernier octet exécuté.
ADAD	2	Adresse pour END, STOP et CONT.
ADB1	1	Numéro d'erreur pour la fonction ON ERROR GOTO.
ADB2	9	Paramètres utilisés par l'instruction SOUND.
AEOC	26	Table de déclaration des variables. Composée de 26 octets (1 par lettre de l'alphabet). Chaque octet contient un code qui détermine le type par défaut de chaque variable commençant par cette lettre.
AE2E	2	Adresse de la ligne courante pour le READ DATA.
AE30	2	Adresse de début de lecture de DATA pour RESTORE.
AE34	2	Utilisé par ON ERROR GOTO.
AE38	1	Sémaphore : 0=TROFF, 1=TRON.



## LES PRINCIPALES VARIABLES SYSTEME

Adresse	Longueur	Signification
~ AE72	2	Sauvegarde de DE pour l'utilisation de l'instruction CALL.
~ AE74	1	Sauvegarde de l'accumulateur pour l'utilisation de l'instruction CALL.
- AE75	2	Sauvegarde de HL pour l'utilisation de l'instruction CALL.
AE77	2	Sauvegarde de SP pour l'utilisation de l'instruction CALL.
AE79	2	Utilisé par l'instruction ZONE (adresse).
✗ AE7B	2	HIMEM (adresse supérieure pour le Basic).
AE7D	2	Utilisé par l'instruction SYMBOL (adresse).
✗ AE81	2	Adresse du début du programme Basic (par défaut 016F).
✗ AE83	2	Adresse de fin du programme Basic.
AE85	2	Adresse du début des tables de variables.
AE87	2	Adresse de la table des variables simples.
AE89	2	Adresse de la table des variables tableaux (DIM).
BOBA	1	Sémaphore permettant de voir si une touche a été pressée (utilisé par INKEY).
BOC1	1	Type de l'accumulateur virtuel.
BOC2	5	Cinq octets utilisés par l'accumulateur virtuel.
B1C7	1	Octet pour l'encodage (masque) de l'encre.
B1C8	1	Mode écran (0, 1 ou 2).
B1C9	2	Offset de l'écran compris entre 0 et 7FF.
> B1CB	1	Octet de poids fort du début de la mémoire réelle écran.
B1CC	1	Contient parfois un C3 (jump).
B1CD	2	Contient l'adresse pour le jump.
B1D7	1	Longueur de la première période de clignotement du bord.
B1D8	1	Longueur de la seconde période de clignotement du bord.
B1DA	32	Couleurs des encres (deux octets par couleur).



# LES PRINCIPALES VARIABLES SYSTEME

Adresse	Longueur	Signification
B1FC	1	Utilisé par border.
B20C	1	Numéro du STREAM.
B285	1	Position ligne curseur.
B286	1	Position colonne curseur.
B287	1	Indicateur de fenêtre.
B288	1	Ligne début fenêtre courante.
B289	1	Colonne début fenêtre courante.
B28A	1	Ligne fin fenêtre courante.
B28B	1	Colonne fin fenêtre courante.
B28D	1	0 = curseur autorisé, 255 = curseur interdit.
B28E	1	0 = affichage interdit, 255 = affichage autorisé.
271 B28F	1	Encre courante pour le crayon.
B290	1	Encre courante pour le papier.
B291	1	0 = affichage du fond permis, 255 = affichage du fond interdit.
B294	2	Premier caractère et état de la table des matrices définie par l'utilisateur.
B296	2	Adresse de la table des matrices définie par l'utilisateur.
B2C3	96	Table des codes de contrôle.
B328	2	Coordonnée de l'origine de l'axe X.
B32A	2	Coordonnée de l'axe Y.
B32C	2	Coordonnée graphique X.
B32E	2	Coordonnée graphique Y.
B330	2	Coordonnée X d'un bord de la fenêtre graphique.
B332	2	Coordonnée X de l'autre bord de la fenêtre graphique.
B334	2	Coordonnée Y d'un bord de la fenêtre graphique.
B336	2	Coordonnée Y de l'autre bord de la fenêtre graphique.
B338	1	Encre du crayon graphique.



## LES PRINCIPALES VARIABLES SYSTEME

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
B339	1	Encre du papier graphique.
B33A	8	Quatre zones de deux octets utilisées comme mémoire temporaire de calcul pendant le traçage d'une ligne.
B342	2	Coordonnée X du point final pour le traçage d'une ligne.
B344	2	Coordonnée Y du point final pour le traçage d'une ligne.
B34C	80	Table des valeurs des touches sans SHIFT ni CTRL.
B39C	80	Table des valeurs des touches avec SHIFT.
B3EC	80	Table des valeurs des touches avec CTRL.
B43C	80	Table des répétitions pour chaque touche.
B4DE	2	Utilisé pour le scanning (adresse).
B4E0	1	Sauvegarde temporaire du caractère scanné (BBOC).
B4E9	1	Valeur de la vitesse de répétition des touches.
B4EA	1	Valeur du délai avant la répétition d'une touche.
B4EB	10	Table de scanning des touches.
B4F1	1	Etat de la manette de jeux 1.
B4F4	1	Etat de la manette de jeux 2.
B50C	1	Utilisé pour le contrôle du BREAK.
B541	2	Adresse de la table des touches sans SHIFT ni CTRL.
B543	2	Adresse de la table des touches avec SHIFT.
B545	2	Adresse de la table des touches avec CTRL.
B547	2	Adresse de la table de répétitions des touches.
B551		Début de la zone des variables du gestionnaire sonore.
B60A	240	15x16 octets avec les valeurs des enveloppes d'amplitude.
B6FA	240	15x16 octets avec les valeurs des enveloppes tonales.



## LES PRINCIPALES VARIABLES SYSTEME

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
B800		Début de la zone des variables du gestionnaire cassette.
B800	1	Prompt message autorisé si 0, interdit si # 0.
B802	1	Indicateur d'ouverture de fichier.
B803	2	Adresse du tampon de 2K pour le catalogue.
B805	2	Adresse tampon lecture.
B819	1	Type de fichier cassette.
B81A	2	Adresse courante tampon de lecture.
B81C	2	Emplacement mémoire des données.
B81F	2	Longueur logique du fichier.
B847	1	Type du stream d'écriture.
B84A	2	Adresse du tampon d'écriture.
B85F	2	Adresse courante du tampon d'écriture.
B8CD	1	Caractère de synchronisation.
B8D1	2	Vitesse d'écriture ou de lecture.
B8F7	1	0 = mode <u>RADIANS</u> , 255 = mode DEGRES.



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

La ROM inférieure contient les routines système (communication avec le matériel), les routines mathématiques et le générateur de caractères.

*Remarque* : les adresses qui correspondent aux routines déjà décrites en détails, pages 81 à 111, sont indiquées avec uniquement le point d'entrée mémoire vive correspondant suivi d'une \*.

Dès lors, nous vous conseillons de vous reporter aux pages 81 à 111, pour de plus amples informations.

005C	BCC8 *
0099	BD0D *
00A3	BD10 *
0163	BCD7 *
016A	BCDA *
0170	BCDD *
0176	BCE0 *
017D	BCE3 *
0183	BCE6 *
01B3	BCE9 *
01C5	BCEC *
01D2	BCEF *
01E2	BCF2 *
021A	BCFE *
0228	BCF5 *
0256	BCFB *
0277	BD01 *
0285	BCF8 *
028E	BD0A *
0295	BD04 *
029B	BD07 *
02A1	BCD1 *
02B2	BCD4 *
0329	BCCB *
0332	BCCE *
05DC	BD13 *
060B	BD16 *
066D	Message 64K MICROCOMPUTER (V1).
068A	Message copyright 1984 Amstrad Consumer Electronics PLC and Locomotive Software Ltd.
06F4	Message *** program load failed ***
0727	Liste des compatibles Arnold, Amstrad, Orion, Schneider, Awa, Solavox, Saisho, Triumph, Isp.
0776	BD1C *
0786	BD22 *
0799	BD25 *
07BA	BD19 *
07C6	BD1F *
07E6	BD28 *



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

07F2	BD2B *	1174	BB75 *
07F8	BDF1 *	1180	BB78 *
0807	BD31 *	11CE	BB87 *
081B	BD2E *	120C	BB66 *
0826	BD34 *	1256	BB69 *
0888	BD37 *	1263	BDCD *
0AA0	BBFF *	1263	BDD0 *
0AB1	BC02 *	1268	BB8A *
0ACA	BC0E *	1268	BB8D *
0AEC	BC11 *	1279	BB81 *
0AF7	BC14 *	1281	BB84 *
0AF7	BDEB *	1289	BB7B *
0B3C	BC05 *	129A	BB7E *
0B45	BC08 *	12A9	BB90 *
0B50	BC0B *	12AE	BB96 *
0B57	BC17 *	12BD	BB93 *
0B64	BC1A *	12C3	BB99 *
0BA9	BC1D *	12C9	BB9C *
0BF9	BC20 *	12D3	BBA5 *
0C05	BC23 *	12F1	BBA8 *
0C13	BC26 *	12FD	BBAB *
0C2D	BC29 *	132A	BBAE *
0C49	BC59 *	1334	BB5D *
0C68	BDE8 *	134A	BDD3 *
0C6B	BC5C *	137A	BB9F *
0C82	BDE5 *	1387	BBA2 *
0C86	BC2C *	13A7	BB63 *
0CA0	BC2F *	13AB	BB60 *
0CE4	BC3E *	13C0	BDD6 *
0CE8	BC41 *	1400	BB5A *
0CEC	BC32 *	140C	BDD9 *
0CF1	BC38 *	144B	BB57 *
0D14	BC35 *	1451	BB54 *
0D19	BC3B *	146B	Table des codes de
0DB3	BC44 *		contrôle pour le ter-
0DB7	BC47 *		minal (voir page 170),
0DDF	BC4A *		96 octets.
0DFA	BC4D *	14CB	BBB1 *
0E3E	BC50 *	1540	BB6C *
0EF3	BC53 *	15B0	BBBA *
0F49	BC56 *	15DF	BBBD *
0FC4	BC5F *	15F1	BBC3 *
102F	BC62 *	15F4	BBC0 *
1078	BB4E *	15FC	BBC6 *
1088	BB51 *	1604	BBC9 *
10E8	BBB4 *	1612	BBCC *
1107	BBB7 *	1734	BBCF *
115E	BB6F *	1779	BBD2 *
1169	BB72 *	17A6	BBD5 *



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

17BC	BBD8 *
17C5	BBDB *
17F6	BBDE *
17FD	BBE4 *
1804	BBE1 *
180A	BBE7 *
1810	BBED *
1813	BBEA *
1816	BDDC *
1824	BBF3 *
1827	BBF0 *
182A	BDD0 *
182A	BDDF *
1836	BBF9 *
1839	BBF6 *
183C	BDE2 *
1945	BBFC *
19E0	BB00 *
1A1E	BB03 *
1A3C	BB06 *
1A42	BB09 *
1A77	BB0C *
1A7B	BB15 *
1AB3	Valeur par défaut des touches étendues (RUN pour CTRL CR)
1ABD	BB0F *
1B2E	BB12 *
1B56	BB18 *
1B5C	BB1B *
1BB3	BB21 *
1C2F	BDEE *
1C5C	BB24 *
1C6D	BB3F *
1C69	BB42 *
1C71	BB45 *
1C82	BB48 *
1C90	BB4B *
1CA6	BB3C *
1CAB	BB39 *
1CBD	BB1E *
1D52	BB27 *
1D3E	BB2A *
1D57	BB2D *
1D43	BB30 *
1D5C	BB33 *
1D48	BB36 *
1D69	Table des valeurs par défaut des touches clavier.
1E68	BCA7 *
1ECB	BCB6 *
1EE6	BCB9 *



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

1F9F	BCAA *
204A	BCB3 *
206C	BCAD *
2089	BCB0 *
2338	BCBC *
233D	BCBF *
2349	BCC2 *
234E	BCC5 *
2370	BC65 *
237F	BC68 *
238E	BC6B *
2392	BC77 *
23AB	BC8C *
23FC	BC7A *
2401	BC7D *
2415	BC8F *
242E	BC92 *
2435	BC80 *
245B	BC95 *
2496	BC89 *
249A	BC86 *
24AB	BC83 *
24EA	BC98 *
2528	BC9B *
27C5	Message press play then any key
27DB	Message error
27E5	Message REC
27E8	Message and
27ED	Message Read
27F3	Message write
27FA	Message Rewind
2800	Message tape
2805	Message found
280D	Message loading
2815	Message saving
281D	Message OK
2820	Message Block
2826	Message Unnamed
282D	Message File
2836	BCA1 *
283F	BC9E *
2851	BCA4 *
2A4B	BC6E *
2A4F	BC71 *
2A51	BC74 *
2E18	BD3D *
2E29	BD40 *
2E55	BD43 *
2E5E	BD94 *



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

2E66	BD46 *
2E8E	BD49 *
2EA1	BD4C *
2EAC	BD4F *
2EB6	BD52 *
2F10	BD55 *
2F53	Table des puissances de 10. 13 entrées de 5 octets pour les valeurs de 10 à 10E13.
2F94	BD97 *
2FA1	BD9A *
2FB7	BD9D *
2FE6	BDA0 *
300F	BD82 * LOG10
3014	BD7F * LOG
3086	Valeur codée de LOG(2) (0,693147181)
308C	Valeur codée de LOG10(2) (0,301029996)
3090	BD85 * EXP
30CC	Constante 0,5 codée
30FB	Constante 1,44269504
3100	Constante 88,0296919
3105	Constante -88,7228391
310A	BD79 * SQR
310D	BD7C *
31A3	BD76 * PI
31A9	Constante PI (3,14159265)
31AE	BD73 * DEG-RAD
31B2	BD8B * COS
31BC	BD88 * SIN
31EC	Table de 6 nombres codés sur 5 octets chacun pour le calcul des sinus et des cosinus.
321D	Table de 4 nombres codés sur 5 octets chacun pour le calcul des sinus et des cosinus.
3231	BD8E * TAN
3241	BD91 * ATN
3258	Table de 11 nombres codés sur 5 octets chacun pour le calcul de l'arc tangente.
3337	BD5B *
333B	BD5E *
333F	BD58 *
3415	BD61 *
349E	BD64 *
3578	BD67 *
359A	BD6A *
35E8	BD70 *
35F8	BD6D *
3708	BDA3 *
370E	BDA6 *
3715	BDA9 *
3728	BDAC *



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE

3730	BDB2 *
3731	BDAF *
3739	BDB5 *
3750	BDBE *
377A	BDB8 *
3781	BDBB *
378C	BDC1 *
37D4	BDC7 *
37E0	BDCA *
37E9	BDC4 *
3800	Début de la table du générateur de caractères (256 x 8 octets).
3FFF	Fin de la table



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE

La ROM supérieure contient toutes les routines de traitement de tous les mots-clés Basic.

C002	Initialisation + envoi du message Basic 1.0
C03F	Message Basic 1.0
C053	Fonction EDIT
C090	Entrée principale (affichage du READY)
C0CC	Message READY
C0DF	AUTO
C12B	NEW
C132	CLEAR
C20A	PAPER
C212	PEN
C221	BORDER
C22A	INK
C24F	MODE
C25A	CLS
C262	VPOS
C276	POS
C2D2	LOCATE
C2E1	WINDOW
C319	TAG
C320	TAGOFF
C337	Envoie le message pointé par HL
C3E3	WIDTH
C417	EOF
C48C	ORIGIN
C4B5	CLG
C4C6	DRAW
C4CB	DRAWR
C4D0	PLOT
C4D5	PLOTR
C4E9	TEST
C4EE	TESTR
C505	MOVE
C50A	MOVER
C529	FOR
C5FB	NEXT
C6C7	IF
C6E8	GOTO
C6ED	GOSUB
C70F	RETURN
C747	WHILE
C776	WEND
C7C3	ON
C8CB	ON BREAK
C8E1	DI
C8E7	EI
C940	ON SQ



## ADRESSES PRINCIPALES DE LA ROM SUPÉRIEURE

C971	AFTER
C979	EVERY
C99F	REMAIN
CA8F	ERROR
CB23	Message UNDEFINED LINE
CB33	Routine envoi message BREAK in
CB4F	Message BREAK
CB55	Message IN
CB5A	STOP
CB65	END
CBC0	CONT
CBF8	ON ERROR
CC03	RESUME
CC5B	Table des messages d'erreur
CE66	Fin de la table des messages d'erreur
CF81	Table des points d'entrée des opérations arithmétiques et logiques.
DOCA	Table des points d'entrée des fonctions EOF, ERR, HIMEM, INKEY\$, PI, RND, TIME, XPOS et YPOS.
D0DC	ERR
D0F4	HIMEM
D107	XPOS
D10E	YPOS
D190	Table des points d'entrée des fonctions
D219	ROUND
D1EA	MIN
D1EE	MAX
D256	OPENOUT
D25F	OPENIN
D298	CLOSEIN
D2A1	CLOSEOUT
D2C0	SOUND
D31E	RELEASE
D329	SQ
D34E	ENV
D385	ENT
D409	INKEY
D423	JOY
D439	KEY DEF
D494	SPEED
D4DB	PI
D4E7	DEG
D4EB	RAD
D4EF	SQR
D4F4	Routine d'élévation à une puissance.
D520	EXP
D525	LOG10
D52A	LOG
D52F	SIN



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE

D534	COS
D539	TAN
D53E	ATN
D543	Message RANDOM NUMBER SEED ?
D559	RANDOMIZE
D584	RND
D614	DEFSTR
D618	DEFINT
D61C	DEFREAL
D654	LET
D67D	DIM
D9C0	ERASE
DAF8	LINE
DB28	INPUT
DB77	Message ? redo from start
DCD9	RESTORE
DCEB	READ
DDEZ	TRON
DDEG	TROFF
DE01	Table des points d'entrée des mots-clés Basic
DEBA	Fin de la table
DFDC	Table des mots-clés qui peuvent être suivis d'un numéro de ligne (GOTO, RESTORE, AUTO, EDIT,...)
EOF7	LIST
E2DD	Routine de positionnement sur la table des lettres pour rechercher des mots-clés
E327	Routine de test qui vérifie si un mot-clé se trouve dans la table
E354	Table des adresses pour chacune des 26 lettres de l'alphabet
E388	Table des mots-clés avec leur code
E64A	Fin de la table
E728	DELETE
E7DF	RENUM
E8EF	DATA
E8F3	REM
E9BD	RUN
E9F6	LOAD
EA3C	CHAIN
EAA6	MERGE
EC09	SAVE
F158	PEEK
F15F	POKE
F16D	INP
F177	OUT
F17D	WAIT
F1BA	CALL
F1F6	ZONE
F1FD	PRINT



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE

F2C4	PRINT USING
F47B	WRITE
F4EF	MEMORY
F69D	SYMBOL
F834	LOWER\$
F839	Routine de conversion en minuscules
F842	UPPER\$
F8BA	BIN\$
F8C4	HEX\$
F8EA	DEC\$
F91E	STR\$
F93C	LEFT\$
F943	RIGHT\$
F993	MID\$
FA0A	LEN
FA10	ASC
FA16	CHR\$
FA24	INKEY\$
FA36	STRING\$
FA57	SPACE\$
FA77	VAL
FAA1	INSTR
FC2D	FRE
FCCC	Addition +
FCE1	Soustraction -
FCF5	Multiplication *
FD12	Division /
FD37	Division entière \
FD49	Modulo (reste de la division)
FD58	Fonction AND (ET LOGIQUE)
FD63	Fonction OR (OU LOGIQUE)
FD6D	Fonction XOR (OU EXCLUSIF)
FD85	ABS
FDE8	FIX
FDED	INT
FE8D	CINT
FEC2	UNT
FEEC	CREAL
FEF3	Nettoyage de l'accumulateur
FF02	SGN
FF0A	Positionnement d'un entier dans l'accumulateur
FF16	Conversion en réel
FF1D	Met le type de variable dans C
FF23	Met le type de variable dans A
FF27	Teste si l'accumulateur contient un pointeur de chaîne
FF62	Copie l'accumulateur dans la zone pointée par DE
FF71	Teste si majuscule
FF7B	Teste si numérique
FF8A	Conversion en majuscule



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE

FFAA	Compare A et le contenu de HL
FFB8	Compare HL et DE
FFBE	Compare HL et BC
FFC4	$DE = HL - DE$
FFCF	$HL = HL - DE$
FFDA	$BC = HL - DE$
FFE7	$HL = HL - BC$
FFF2	LDIR
FFF5	LDDR
FFF8	JP (HL)
FFF9	Retour au contenu de BC
FFFB	Retour au contenu de DE



Table des adresses de branchement réel  
des différents vecteurs

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>
BB00	19E0	BB81	1279	BC02	0AB1
BB03	1A1E	BB84	1281	BC05	0B3C
BB06	1A3C	BB87	11CE	BC08	0B45
BB09	1A42	BB8A	1268	BC0B	0B50
BB0C	1A77	BB8D	1268	BC0E	0ACA
BB0F	1ABD	BB90	12A9	BC11	0AEC
BB12	1B2E	BB93	12BD	BC14	0AF7
BB15	1A7B	BB96	12AE	BC17	0B57
BB18	1B56	BB99	12C3	BC1A	0B64
BB1B	1B5C	BB9C	12C9	BC1D	0BA9
BB1E	1CBD	BB9F	137A	BC20	0BF9
BB21	1BB3	BBA2	1387	BC23	0C05
BB24	1C5C	BBA5	12D3	BC26	0C13
BB27	1D52	BBA8	12F1	BC29	0C2D
BB2A	1D3E	BBAB	12FD	BC2C	0C86
BB2D	1D57	BBAE	132A	BC2F	0CA0
BB30	1D43	BBB1	14CB	BC32	0CEC
BB33	1D5C	BBB4	10E8	BC35	0D14
BB36	1D48	BBB7	1107	BC38	0CF1
BB39	1CAB	BBBA	15B0	BC3B	0D19
BB3C	1CA6	BBBD	15DF	BC3E	0CE4
BB3F	1C6D	BBC0	15F4	BC41	0CE8
BB42	1C69	BBC3	15F1	BC44	0DB3
BB45	1C71	BBC6	15FC	BC47	0DB7
BB48	1C82	BBC9	1604	BC4A	0DDF
BB4B	1C90	BBCC	1612	BC4D	0DFA
BB4E	1078	BBCF	1734	BC50	0E3E
BB51	1088	BBD2	1779	BC53	0EF3
BB54	1451	BBD5	17A6	BC56	0F49
BB57	144B	BBD8	17BC	BC59	0C49
BB5A	1400	BBDB	17C5	BC5C	0C6B
BB5D	1334	BBDE	17F6	BC5F	0FC4
BB60	13AB	BBE1	1804	BC62	102F
BB63	13A7	BBE4	17FD	BC65	2370
BB66	120C	BBE7	180A	BC68	237F
BB69	1256	BBEA	1813	BC6B	238E
BB6C	1540	BBED	1810	BC6E	2A4B
BB6F	115E	BBF0	1827	BC71	2A4F
BB72	1169	BBF3	1824	BC74	2A51
BB75	1174	BBF6	1839	BC77	2392
BB78	1180	BBF9	1836	BC7A	23FC
BB7B	1289	BBFC	1945	BC7D	2401
BB7E	129A	BBFF	0AA0	BC80	2435



# LES ADRESSES REELLES ROM

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>
BC83	24AB	BCE3	017D	B909	BA54
BC86	249A	BCE6	0183	B90C	BA72
BC89	2496	BCE9	01B3	B90F	BA7E
BC8C	23AB	BCEC	01C5	B912	BAA2
BC8F	2415	BCEF	01D2	B915	BA83
BC92	242E	BCF2	01E2	B918	BA8C
BC95	245B	BCF5	0228	B91B	BAA6
BC98	24EA	BCF8	0285	B91E	BAAC
BC9B	2528	BCFB	0256	BDCD	1263
BC9E	283F	BCFE	021A	BDD0	1263
BCA1	2836	BD01	0277	BDD3	134A
BCA4	2851	BD04	0295	BDD6	13C0
BCA7	1E68	BD07	029B	BDD9	140C
BCAA	1F9F	BD0A	028E	BDDC	1816
BCAD	206C	BD0D	0099	BDDF	182A
BCB0	2089	BD10	00A3	BDE2	183C
BCB3	204A	BD13	05DC	BDE5	0C82
BCB6	1ECB	BD16	060B	BDE8	0C68
BCB9	1EE6	BD19	07BA	BDEB	0AF7
BCBC	2338	BD1C	0776	BDEE	1C2F
BCBF	233D	BD1F	07C6	BDF1	07F8
BCC2	2349	BD22	0786	8	B982
BCC5	234E	BD25	0799	B	B97C
BCC8	005C	BD28	07E6	10	BA16
BCCB	0329	BD2B	07F2	13	BA10
BCCE	0332	BD2E	081B	18	B9BF
BCD1	02A1	BD31	0807	1B	B9B1
BCD4	02B2	BD34	0826	20	BACB
BCD7	0163	BD37	0888	23	B9B9
BCDA	016A	B900	BA5E	28	BA2E
BCDD	0170	B903	BA68	38	B939
BCE0	0176	B906	BA4A		



# ADRESSES D'EXECUTION DES MOTS-CLES DU BASIC

<i>Mot-clé</i>	<i>Adresse</i>	<i>Mot-clé</i>	<i>Adresse</i>
ABS	FD85	FOR	C529
AFTER	C971	FRE	FC2D
ASC	FA10	GOSUB	C6ED
ATN	D53E	GOTO	C6E8
AUTO	C0DF	HEX\$	F8C4
BIN\$	F8BA	HIMEM	D0F4
BORDER	C221	IF	C6C7
CALL	F1BA	INSTR	FAA1
CAT	D246	INK	C22A
CHAIN	EA3C	INKEY	D409
CHR\$	FA16	INKEY\$	FA24
CINT	FE8D	INP	F16D
CLEAR	C132	INPUT	DB2B
CLG	C4B5	INT	FDED
CLOSEIN	D298	JOY	D423
CLOSEOUT	D2A1	KEY	D439
CLS	C25A	LEFT\$	F93C
CONT	CBC0	LEN	FA0A
COS	D534	LET	D654
CREAL	FEEC	LINE	DAF8
DATA	E8EF	LIST	E0F7
DEC\$	F8EA	LOAD	E9F6
DEF	D417	LOCATE	C2D2
DEFINT	D618	LOG	D52A
DEFREAL	D61C	LOG10	D525
DEFSTR	D614	LOWER\$	F834
DEG	D4E7	MAX	D1EE
DELETE	E728	MEMORY	F4EF
DI	C8E1	MERGE	EAA6
DIM	D67D	MID\$	F993
DRAW	C4C6	MIN	D1EA
DRAWR	C4CB	MODE	C24F
EDIT	C052	MOVE	C505
EI	C8E7	MOVER	C50A
ELSE	E8F3	NEXT	C5FB
END	CB65	NEW	C12B
ENT	D385	ON	C7E3
ENV	D34E	ON BREAK	C8CB
EOF	C417	ON ERROR	CBF8
ERASE	D9C0	ON SQ	C940
ERR	D0DC	OPENIN	D25F
ERROR	CA8F	OPENOUT	D256
EVERY	C979	ORIGIN	C48C
EXP	D520	OUT	F177
FIX	FDE8	PAPER	C20A



# ADRESSES D'EXECUTION DES MOTS-CLES DU BASIC

<i>Mot-clé</i>	<i>Adresse</i>	<i>Mot-clé</i>	<i>Adresse</i>
PEEK	F158	SPEED	D494
PEN	C212	SQ	D329
PI	D4DB	SQR	D4EF
PLOT	C4D0	STOP	CB5A
PLOTR	C4D5	STR\$	F91E
POKE	F15F	STRING\$	FA36
POS	C276	SYMBOL	F69D
PRINT	F1FD	TAG	C319
' (REM)	E8F3	TAGOFF	C320
RAD	D4EB	TAN	D539
RANDOMIZE	D559	TEST	C4E9
READ	DCEB	TESTR	C4EE
RELEASE	D31E	TIME	D0E5
REM	E8F3	TROFF	DDE6
REMAIN	C99F	TRON	DDE2
RENUM	E7DF	UNT	FEC2
RESTORE	DCD9	UPPER\$	F842
RESUME	CC03	VAL	FA77
RETURN	C70F	VPOS	C262
RIGHT\$	F943	WAIT	F17D
RND	D584	WEND	C776
ROUND	D219	WHILE	C747
RUN	E9BD	WIDTH	C3E3
SAVE	EC09	WINDOW	C2E1
SGN	FF02	WRITE	F47B
SIN	D52F	XPOS	D107
SOUND	D2C0	YPOS	D10E
SPACE\$	FA57	ZONE	F1F6



## ROM expansion

octet 0	ROM TYPE
octet 1	MARQUE
octet 2	VERSION
octet 3	NIVEAU
octet 4	TABLE

## Streams

octet 0	VIDEO
octet 1	CURSEUR
octet 2	POSITION CURSEUR
octet 3	TAILLE FENETRE
octet 4	ENCRE
octet 5	CARACTERE
octet 6	GRAPHIQUE

## Queue sonore

octet 0	CANAUX QUI UTILISENT UN RENDEZ-VOUS
octet 1	ENVELOPPE D'AMPLITUDE
octet 2	ENVELOPPE DE TIMBRE
octets 3 et 4	PERIODE SON
octet 5	PERIODE BRUIT
octet 6	AMPLITUDE INITIALE
octets 7 et 8	DUREE DE L'ENVELOPPE

## Bloc de contrôle d'amplitude ou de timbre

octet 0	NOMBRE DE SECTIONS
octets 1, 2 et 3	PREMIERE SECTION
octets 4, 5 et 6	DEUXIEME SECTION
octets 7, 8 et 9	TROISIEME SECTION
octets 10, 11 et 12	QUATRIEME SECTION
octets 13, 14 et 15	CINQUIEME SECTION



## LES BLOCS DE CONTROLE

### Vecteur encre

octet 0	COULEUR DU BORD
octet 1	COULEUR ENCRE 0
octet 2	COULEUR ENCRE 1
...	...
octet 16	COULEUR ENCRE 15

### Format des deux octets qui suivent un RESTART

BIT 15	14	13...0
X	Y	ADRESSE

### ROM standard

X=0	ROM SUPERIEURE DECONNECTEE
X=1	ROM SUPERIEURE SELECTIONNEE
Y=1	ROM INFERIEURE DECONNECTEE
Y=0	ROM INFERIEURE SELECTIONNEE

### ROM supplémentaire

XY donne une valeur de 0 à 3 qui, ajoutée à l'adresse de sélection de la ROM principale, fournit l'adresse de la ROM secondaire.

### Format des fichiers cassette

#### Le bloc complet

	enregistrement	enregistrement
MOTEUR GAP*	EN EN-TETE DE BLOC	DONNEE

Le premier et le dernier bloc ont en plus un GAP qui permet la séparation de deux programmes ou fichiers.

Premier bloc	MOTEUR GAP	GAP DE DEBUT	EN-TETE	DONNEE
Dernier bloc	MOTEUR GAP	EN-TETE	DONNEE	GAP DE FIN

\* GAP = période de défilement sans écriture.



## Format d'un enregistrement

DEMARREUR	SEGMENT 1	SEGMENT 2	.....	SEGMENT n	FERMEUR
-----------	-----------	-----------	-------	-----------	---------

1 SEGMENT : 256 octets + 2 octets de contrôle (CRC).

Enregistrement d'en-tête : 1 SEGMENT

Enregistrement d'une donnée : 1 à 8 SEGMENTS (habituellement 8).

Démarreur : 2048 bits à 1 suivi d'un bit à 0 et d'un octet de synchronisation.

Fermeur : 32 bits à 1.

## Format de l'en-tête

octets 0 à 15	NOM DU FICHIER
octet 16	NUMERO DU BLOC
octet 17	#0 SI C'EST LE DERNIER BLOC
octet 18	TYPE DE FICHIER
octets 19 et 20	LONGUEUR DE L'ENREGISTREMENT DATA
octets 21 et 22	ADRESSE D'ECRITURE DATA
octet 23	#0 SI C'EST LE PREMIER BLOC
octets 24 et 25	LONGUEUR TOTALE DU FICHIER EN OCTETS
octets 26 et 27	POINT D'ENTREE
octets 28 à 63	NON UTILISES

Description de l'octet 18 (type de fichier) :

bit 0	1 si le fichier est protégé
bits 1 et 2	00 = BASIC 01 = BINAIRE 10 = IMAGE ECRAN (DUMP) 11 = ASCII
bit 3	Inutilisé
bits 4 à 7	Version toujours à 0, sauf dans le cas de fichiers ASCII pour lesquels le bit 4 est à 1.

## Bloc d'événement

octets 0 et 1	POINTEUR SYSTEME
octet 2	COMPTEUR
octet 3	CLASSE
octets 4 et 5	ADRESSE DE LA ROUTINE DE TRAITEMENT
octet 6	ADRESSE DE SELECTION ROM



## LES BLOCS DE CONTROLE

### Bloc de contrôle d'interruption normale

octets 0 et 1	POINTEUR SYSTEME
octets 2 et 3	COMPTEUR. Quand il atteint 0, l'interruption a lieu.
octets 4 et 5	RECHARGE. Valeur de réinitialisation après atteinte du 0.
octets 6,...	BLOC D'EVENEMENT (voir ci-dessus).

### Bloc d'interruption rapide et d'interruption CRT

octets 0 et 1	POINTEUR SYSTEME
octets 2,...	BLOC D'EVENEMENT (voir ci-dessus).



## CIRCUIT AY3-8912 (PSG)

### Structure interne

Le PSG est composé des éléments suivants :

- *Générateurs sonores* : au nombre de trois, ils produisent un signal carré dont la fréquence est programmable. On les appelle CANAUX A, B et C. Ils n'ont pas de priorité propre et sont indépendants.
- *Générateur de bruit blanc* : il produit un bruit à large spectre.
- *Mélangeur* : il permet de mélanger (combiner) les sorties des trois générateurs sonores et du générateur de bruit.
- *Contrôleur d'amplitude* : il permet de sélectionner l'amplitude de sortie du signal de deux façons différentes. La première est de contrôler l'amplitude par le microprocesseur lui-même : elle est dite amplitude fixe. La seconde est de contrôler l'amplitude par le générateur d'enveloppes : elle est dite amplitude variable.
- *Générateur d'enveloppe* : il produit une enveloppe de modulation de l'amplitude. Il possède huit formes d'enveloppes.
- *Convertisseurs digitaux-analogiques* : les trois convertisseurs D/A produisent les signaux à 16 niveaux tels que le contrôleur d'amplitude les détermine.
- *Port d'entrée/sortie* : il ne sert pas à la production sonore, il sera analysé à la fin de ce chapitre.

### Les différents registres du PSG

Les registres sont au nombre de 15, numérotés de R0 à R14. Le registre R14 sert à la gestion du port d'entrée/sortie et sera analysé par la suite.

Pour produire un son, une combinaison des registres R0 à R13 doit être chargée avec des données. Chaque paramètre doit être analysé de façon à dissocier la composante bruit, la composante



## CIRCUIT AY3-8912 (PSG)

son, la fréquence, la forme et la durée de l'enveloppe. Une fois cette analyse effectuée, les registres peuvent être chargés et le son produit.

### *Les registres R0 à R5*

Les trois premières paires de registres (R0-R1, R2-R3, R4-R5) sont les registres de contrôle de la fréquence des trois canaux A, B et C.

Les registres R0, R2 et R4 sont les registres de réglage fin et les huit bits sont utilisés. Les registres R1, R3 et R5 sont les registres de réglage grossier (seuls, les quatre bits de gauche LSB sont utilisés).

Ainsi, les valeurs chargées dans R0, R2 et R4 sont comprises entre 0 et 255 ; les valeurs chargées dans R1, R3 et R5 sont comprises entre 0 et 15.

La valeur se détermine par la formule suivante :

$$VL = 125000 / F$$

### *Le registre R6*

Le registre R6 détermine la fréquence du générateur de bruit ; seuls les cinq bits les moins significatifs sont utilisés. La valeur de R6 est donc comprise entre 1 et 31. La même formule que pour R0-R5 est utilisée.

### *Le registre R7*

Le registre R7 contrôle le mélange entre les trois générateurs sonores et le générateur de bruit. R7 sert aussi au contrôle du port dont nous parlerons par la suite.

Voici un tableau résumant les effets du registre R7.

BIT	= 0	= 1
7	NON UTILISE	NON UTILISE
6	PORT A ENTREE	PORT A SORTIE (inutile)
5	BRUIT SUR CANAL C ON	BRUIT SUR CANAL C OFF
4	BRUIT SUR CANAL B ON	BRUIT SUR CANAL B OFF
3	BRUIT SUR CANAL A ON	BRUIT SUR CANAL A OFF
2	SON SUR CANAL C ON	SON SUR CANAL C OFF
1	SON SUR CANAL B ON	SON SUR CANAL B OFF
0	SON SUR CANAL A ON	SON SUR CANAL A OFF

**Note :** mettre un canal sur OFF ne suffit pas pour arrêter l'émission de celui-ci ; il faut écrire un 0 dans le registre de contrôle d'amplitude (voir ci-après).



*Exemple*

Je désire, sur le canal A, du son et pas de bruit, sur le canal B, du bruit et du son et, sur le canal C, du bruit uniquement.

valeur : x x 0 0 1 1 0 0 = 12  
 bit : 7 6 5 4 3 2 1 0  
 (x x) = sans importance

Il suffit d'écrire 12 dans le registre 7.

*Les registres R8 à R10*

Les registres R8 à R10 contrôlent les amplitudes des canaux A, B et C ; seuls les quatre bits les moins significatifs sont utilisés, donc les valeurs possibles sont comprises entre 0 et 15. 0 signifie que l'amplitude est minimum (nulle) et 15 correspond à l'amplitude maximum. Le cinquième bit (bit 4) est le bit de sélection du mode de fonctionnement du contrôle de l'amplitude. Si BIT4 est 0, l'amplitude ne varie pas. Si BIT4 est 1, l'amplitude est contrôlée par le générateur d'enveloppe (*voir ci-dessous*).

*Les registres R11 et R12*

Ces deux registres contrôlent la période de l'enveloppe. Un calcul avec une formule similaire à celle utilisée pour R0-R5 est effectué pour déterminer la valeur de R11 et R12.

Formule :  $VL = 125000 * P / 16$  ou P est la période de l'enveloppe.

*Le registre R13*

Le registre R13 contrôle la forme de la modulation utilisée. Si le BIT4 décrit dans les registres R8 à R10 est 1, la modulation a lieu, sinon la programmation du registre 13 est ignorée.

Seuls, les quatre bits les moins significatifs sont utilisés.

Bit				Forme de l'enveloppe	Valeurs possibles
3	2	1	0		
0	0	x	x	A Un seul cycle commence avec une amplitude maximum qui diminue pour devenir nulle.	0, 1, 2, 3
0	1	x	x	B Un seul cycle commence avec une amplitude nulle qui augmente pour atteindre sa valeur maximale, ensuite retombe brusquement à 0.	4, 5, 6, 7
1	0	0	0	C Comme A, mais se répète sans cesse.	8



## CIRCUIT AY3-8912 (PSG)

Bit				Forme de l'enveloppe	Valeurs possibles
3	2	1	0		
1	0	1	0	D Comme C, mais remonte de façon plus marquée vers le maximum (ATTACK).	10
1	0	1	1	E Comme A, mais revient ensuite au maximum et y reste.	11
1	1	0	0	F Comme B, mais se répète sans cesse.	12
1	1	0	1	G Comme B, mais reste au maximum.	13
1	1	1	0	H Comme F, mais avec une attaque plus marquée.	14

### Le registre R14

Ce registre n'a rien à voir avec la production sonore. Il gère un port d'entrée-sortie qui s'occupe de la lecture du clavier et de la manette de jeux.

Le bit 6 du registre R7 règle le sens de la transmission, mais comme le port est utilisé uniquement en entrée, il suffit de toujours mettre le bit 6 de R7 à 0.

## Programmation de l'AY3-8912

Le PSG est accessible à travers les ports A et C du PPI 8255 (voir "Circuit PPI 8255", page 143).

Pour plus de facilité, l'écriture dans le PSG peut se faire au moyen de la routine 188 (BD34). La lecture de l'état du clavier et des manettes de jeux est donc plus difficile à mettre en oeuvre directement. Il est donc conseillé de passer par les points d'entrée standard du logiciel système.

Pour ceux qui voudraient quand même programmer directement le PSG, signalons que les deux signaux de commande BDIR et BC1 sont fournis par le port C du PPI 8255.

### Fonction de BDIR et BC1

BDIR	BC1	Fonction
0	0	Inactif : pas de fonction.
0	1	Lecture : le contenu du registre courant est mis sur le bus des données D0-D7.
1	0	Ecriture : le bus D0-D7 contient une donnée à écrire dans le registre courant.
1	1	Ecriture : le bus D0-D7 contient un numéro de registre qui deviendra le registre courant.



## Généralités

Le PPI est un circuit fabriqué par INTEL sous la dénomination 8255A. C'est un circuit d'interfaçage prévu pour les processeurs de la famille du 8080.

Il possède 24 bits d'entrée/sortie qui peuvent être programmés en deux groupes de 12 bits et utilisés dans trois modes principaux.

Dans le premier mode (mode 0), chaque groupe de 12 bits peut être programmé par tranches de 4 bits en entrée comme en sortie.

Dans le second mode (mode 1), chaque groupe de 12 bits peut être programmé de la façon suivante : 8 bits sont utilisés en entrée/sortie, les 4 autres sont utilisés pour le HANDSHAKING (contrôle de transmission).

Le troisième mode (mode 2) est un mode où 8 bits sont utilisés comme PORT bidirectionnel et 5 bits pour le HANDSHAKING.

Le PPI possède aussi la possibilité de positionner des bits à l'état 1 ou 0 directement.

Pour plus de facilité, le PPI est divisé en trois ports de 8 bits distincts appelés PORT A, PORT B et PORT C.

Le PORT C se divise en deux groupes de 4 bits pour former les groupes de 12 bits avec A et B.

## Découpage des PORTS

*Port A - utilisé en entrée et en sortie*

B0 à B7	Correspondent aux D0 à D7 de l'AY38912.
---------	---

*Port B - utilisé en entrée uniquement*

Bit 7	Lecture donnée cassette (INPUT).
Bit 6	Signal BUSY imprimante (INPUT).
Bit 5	Non utilisables.
Bit 4	
Bit 3	
Bit 2	
Bit 1	
Bit 0	Interruption en provenance du CRT.



## CIRCUIT PPI 8255

*Port C - utilisé en sortie uniquement*

Bit 7	Commande BDIR de l'AY3-8912 OUT.
Bit 6	Commande BC1 de l'AY3-8912 OUT.
Bit 5	Ecriture donnée sur cassette.
Bit 4	Démarrage moteur cassette.
Bits 3 à 0	Sélection de la ligne clavier.

### Programmation

Le PPI est interfacé aux adresses suivantes :

Adresse F4xx	Lecture et écriture PORT A.
Adresse F5xx	Lecture et écriture PORT B.
Adresse F6xx	Lecture et écriture PORT C.
Adresse F7xx	Ecriture dans le registre de contrôle.

#### *Remarques :*

- xx signifie n'importe quoi.
- A est utilisé en lecture et en écriture, B est utilisé en lecture seule et C en écriture seule.

Des trois modes décrits brièvement dans les généralités, seul le MODE 0 sera étudié car il suffit à toutes les manipulations envisagées.

Le PPI est programmable à travers un registre de contrôle dans lequel on ne peut qu'écrire. Aucune lecture de ce registre n'est permise.

#### *Ecriture dans le registre de contrôle*

On écrit dans le registre de contrôle par un simple OUT sur le PORT F7xx.

Le mot de contrôle est un mot de 8 bits dont voici le fonctionnement bit par bit.

Bit 7	Toujours 1 si c'est un mot de contrôle.
Bit 6	Détermination du mode de fonctionnement du groupe A. Pour sélectionner le MODE 0, ce bit doit être 0. S'il est à l'état 1, il sélectionne le MODE 2.
Bit 5	Détermination du mode de fonctionnement du groupe A. Pour sélectionner le MODE 0, ce bit doit être 0. S'il est à l'état 1, il sélectionne le MODE 1.



Bit 4	Détermination du sens de fonctionnement du PORT A ; 0 signifie EN SORTIE et 1 signifie EN ENTREE. Sera toujours 1.
Bit 3	Détermination du sens de fonctionnement de la partie haute du PORT C. 0 signifie EN SORTIE et 1 signifie EN ENTREE.
Bit 2	Détermination du mode de fonctionnement du groupe B. 0 signifie MODE 0 et 1 signifie MODE 1. Sera toujours 0.
Bit 1	Détermination du sens de fonctionnement du PORT B. 0 signifie EN SORTIE et 1 signifie EN ENTREE. Sera toujours 1.
Bit 0	Détermination du sens de fonctionnement de la partie basse du PORT C. 0 signifie EN SORTIE et 1 signifie EN ENTREE. Sera toujours 0.

Si le bit 7 est égal à 0, le registre n'est plus utilisé en tant que contrôleur des PORTS, mais il permet de positionner les bits du PORT C à 1 ou à 0.

Bit 7 = 0 : fonctionnement en positionnement de bit.  
 Bits 6, 5 et 4 : non utilisés.  
 Bits 3, 2 et 1 : donnent le numéro du bit à positionner.  
 Bit 0 : donne le sens du positionnement, 1 signifie positionnement du bit à 1 et 0 signifie positionnement du bit à 0.

La programmation se fait donc en envoyant le mot d'état convenable sur le registre de contrôle et en effectuant une lecture ou une écriture sur le PORT idoine.



## LE CIRCUIT CRT 6845

### Généralités

Le circuit **6845** contrôle la génération des signaux vidéo. Il possède un PORT bidirectionnel de 8 bits et peut être positionné au moyen de 19 registres internes. Un des registres sert d'ailleurs de tampon pour la programmation des 18 autres.

### Les différents registres du 6845

R0 à R3	Déterminent le format horizontal et le timing. Ils sont chargés en standard avec des valeurs précises en fonction du mode. Par exemple, en mode 1 : R0=63, R1=40, R2=46, R3=142.
R4 à R9	Déterminent le format vertical. Ils sont chargés avec des valeurs précises. R4=38, R5=0, R6=25, R7=30.
R10 à R15	Gèrent le curseur et sont continuellement modifiés par le software.
R16 à R17	S'occupent de la gestion du plotostyle (non implémenté).

R0	Nombre de caractères total en horizontal 0-255.
R1	Nombre de caractères affichés en horizontal 0-255.
R2	Synchronisation horizontale (position) 0-255.
R3	Longueur de synchronisation 0-15.
R4	Nombre de lignes total en vertical 0-127.
R5	Synchronisation verticale 0-31.
R6	Nombre de caractères affichés en vertical 0-127.
R7	Synchronisation verticale (position) 0-127.
R8	Mode entrelacé 0-3.
R9	Scanning 0-31.
R10	Ligne de départ du scanning du curseur 0-31.
R11	Ligne de fin de scanning du curseur 0-31.
R12	Octet le plus significatif de l'adresse de départ
R13	Octet le moins significatif de la vidéoram
R14	Position du curseur (OLPS). à 16383
R15	Position du curseur (OLMS). 0-16383



### Programmation

Deux adresses de PORT suffisent pour programmer le CRT.

Le PORT BCxx sert à donner des adresses de registre et le PORT BDxx sert à écrire des données sur le registre courant.

Les registres sont à écriture seule, à l'exception de R14 et R15 qui peuvent être lus, et donnent ainsi la position courante du curseur.



## LA VIDEO GATE ARRAY

### Généralités

L'**Amstrad** est équipé d'un circuit spécial qui s'occupe de la commutation des ROM et du contrôle du CRT 6845. Ce circuit ne porte pas de numéro standard et est appelé **GATE ARRAY**.

### Programmation

Le GATE ARRAY peut être considéré comme un port de sortie de 8 bits commandé par un OUT sur le PORT 7Fxx.

Les deux bits supérieurs contrôlent le type d'application.

B7	B6	
0	0	Chargement registre de palette.
0	1	Chargement mémoire de palette.
1	0	Commutation ROM et contrôle vidéo.
1	1	Réservé.

#### *Commutation ROM et contrôle vidéo*

BIT 7=1, BIT 6=0, BIT 5=0

BIT 4=1 : remettra à 0 le diviseur qui génère les interruptions.

BIT 3=0 : ROM supérieure connectée ; BIT 3=1 : ROM supérieure déconnectée.

BIT 2=0 : ROM inférieure connectée ; BIT 2=1 : ROM inférieure déconnectée.

BIT 1 : contrôle vidéo MC1.

BIT 0 : contrôle vidéo MC0.

MC1	MC0	
0	0	Mode 0 (20 x 24).
0	1	Mode 1 (40 x 24).
1	0	Mode 2 (80 x 24).
1	1	Non utilisable.

#### *Registre de palette*

BIT 7=0, BIT 6=0, BIT 5=0

BIT 4=0 : chargement de numéro de couleur d'encre donné par B0-B3.

BIT 4=1 : chargement de numéro de couleur de bord (B0-B3 ignorés).

BIT 3 à BIT 0 donnent le numéro d'encre (15 couleurs possibles).



*Mémoire de palette*

BIT 7=0, BIT 6=1, BIT 5=0

BIT 4 à BIT 0 : 31 valeurs pour le décodage de la couleur du registre de palette. Le nombre de couleurs possibles varie en fonction du mode choisi.



## DUMP HEXA MEMOIRE ROM INFERIEURE ET SUPERIEURE SUR IMPRIMANTE

Ces programmes permettent d'obtenir sur l'imprimante le contenu des mémoires ROM en hexadécimal.

### ♦ Dump hexa mémoire ROM inférieure

```
10 MEMORY &6000
15 CLS
20 FOR I=&A000 TO &A010
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 DATA F3,CD,06,B9,21,00,00,11,00,60,01,FF,3F, ED,B0,C9
70 DATA 00
80 CALL &A000
100 FOR I=&6000 TO 40960
120 IF INT(I/16)*16=I THEN PRINT #8,"":
    PRINT #8,HEX$(I-&6000);" ";
130 A=PEEK(I)
135 A$=RIGHT$("00"+HEX$(A),2)
140 PRINT #8,A$;" ";
150 NEXT I
```

### ♦ Dump hexa mémoire ROM supérieure

```
10 MEMORY &6000
15 CLS
20 FOR I=&A000 TO &A010
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 DATA F3,CD,00,B9,21,00,C0,11,00,60,01,FF,3F, ED,B0,C9
70 DATA 00
80 CALL &A000
100 FOR I=&6000 TO 40960
120 IF INT(I/16)*16=I THEN PRINT #8,"":
    PRINT #8,HEX$(I+&6000);" ";
130 A=PEEK(I)
135 A$=RIGHT$("00"+HEX$(A),2)
140 PRINT #8,A$;" ";
150 NEXT I
```



## DUMP ASCII MEMOIRE ROM INFÉRIEURE ET SUPÉRIEURE SUR IMPRIMANTE

### ♦ Dump ASCII mémoire ROM inférieure

```
10 MEMORY &6000
15 CLS
20 FOR I=&A000 TO &A010
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 DATA F3,CD,06,B9,21,00,00,11,00,
  60,01,FF,3F,ED,B0,C9
70 DATA 00
80 CALL &A000
100 FOR I=&6000 TO 40960
120 IF INT(I/64)*64=I THEN
  PRINT #8,"":
  PRINT #8,HEX$(I-&6000);" ";
130 A=PEEK(I)
140 IF (A>31 AND A<127) OR A>159
  THEN PRINT #8,CHR$(A); ELSE
  PRINT #8,".";
150 NEXT I
```

### ♦ Dump ASCII mémoire ROM supérieure

```
10 MEMORY &6000
15 CLS
20 FOR I=&A000 TO &A010
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 DATA F3,CD,00,B9,21,00,C0,11,00,
  60,01,FF,3F,ED,B0,C9
70 DATA 00
80 CALL &A000
100 FOR I=&6000 TO 40960
120 IF INT(I/64)*64=I THEN
  PRINT #8,"":
  PRINT #8,HEX$(I+&6000);" ";
130 A=PEEK(I)
140 IF (A>31 AND A<127) OR A>159
  THEN PRINT #8,CHR$(A); ELSE
  PRINT #8,".";
150 NEXT I
```



## DEMARRAGE ET ARRET DU MOTEUR DE LA CASSETTE

Démarrage : OUT &HF600,16  
Arrêt : OUT &HF600,0

## PROTECTION DE PROGRAMME

Taper comme première ligne : 10 REM  
comme deuxième ligne : 20 PRINT "DEBUT"

ensuite, encoder le programme à protéger.

Lorsque le programme est entièrement encodé, taper :

POKE 372,225

A partir de cet instant, il n'est plus possible de lister le programme et seul un RUN 20 permet de le lancer.

Le POKE 372,225 a pour effet de remplacer dans la mémoire le token de l'instruction REM par un token inexistant (225).

De cette façon, lorsque l'ordinateur essaie de lister le programme, il rencontre un token qu'il ne peut traduire et affiche SYNTAX ERROR.

De même, lorsqu'il essaie d'exécuter le programme (RUN), il rencontre en première ligne un token inexistant et se plante. Seul un RUN 20 permet de lancer l'exécution du programme car il évite la lecture de la ligne 10 contenant le token inexistant.

## BRUITS ORIGINAUX

```
5 REM SIRENE STARKY ET HUTCH
10 FOR I=80 TO 220 STEP 12
20 SOUND 1,I,2
30 NEXT I
40 FOR I=220 TO 80 STEP -12
50 SOUND 1,I,2
60 NEXT I
70 GOTO 10
```

```
5 REM TIR DE PHASER
10 FOR I=90 TO 125
20 SOUND 1,I,2,15
30 NEXT I
50 GOTO 10
```



## BRUITS ORIGINAUX (suite)

```
5 REM TOUCHE JE SUIS MORT
10 FOR I=15 TO 8 STEP -1
20 SOUND 1,500,20,I,,,1
30 NEXT I
```

## PROGRAMME PERMETTANT DE TRACER DES CERCLES ET DES ELLIPSES

Ce petit programme permet de simuler l'instruction CIRCLE du Basic Microsoft absente dans le Basic Amstrad.

**X** et **Y** représentent les coordonnées horizontale et verticale du centre du cercle.

**R** représente le rayon à donner au cercle.

**AD** représente l'angle de départ et **AF** l'angle d'arrivée. Ils sont tous deux exprimés en degrés et permettent de dessiner des arcs de cercle.

**FAP** représente le facteur d'aplatissement. Il permet de dessiner des ellipses.

```
10 CLS
20 X=320:Y=200:R=100
30 AD=0
40 AF=360
50 FAP=2
60 DEG
70 PLOT X+R*COS(AD),Y+R*SIN(AD)
80 FOR A=AD TO AF
90 X1=X+R*COS(A):Y1=Y+R*SIN(A)/FAP
100 DRAW X1,Y1
110 PLOT X1,Y1
120 NEXT A
```

## SCANNING DU CLAVIER

Encodez le petit programme suivant, lancez-le et poussez sur différentes touches. Notez les valeurs ainsi obtenues, vous pourrez les utiliser dans vos programmes en réalisant un PEEK de l'octet désiré et en testant sa valeur. Cette routine remplace avantageusement l'INKEY\$.

```
10 FOR I=&B4EB TO &B4F4
20 PRINT PEEK(I);
30 NEXT I
40 PRINT
50 GOTO 10
```



## MODIFICATION ORIGINALE DE LA COULEUR DE FOND

Le POKE suivant modifie la couleur du fond (PAPER) par petite bande. Je vous invite à essayer différentes valeurs pour N.

POKE &B290,N

Rappel : N doit être compris entre 0 et 255.

## INSTALLATION D'UNE ROUTINE EN LANGAGE MACHINE DANS UNE REMARQUE

Les routines très courtes et ne contenant pas 2 octets à 0 qui se suivent peuvent être installées dans une ligne de REM.

Ecrivez :

```
10 REM *****
```

Mettez autant d'astérisques que d'octets dans votre routine.

Le Basic commençant en 368, la première étoile se trouve en 374.

Le programme suivant installe la routine et s'efface ensuite.

```
20 FOR I=374 TO 379: REM SI LA ROUTINE A 6 OCTETS DE LONG
30 READ A$
40 POKE I,VAL("&H"+A$)
50 NEXT I
60 DATA 3E,19,21,88,CD,C9
70 DELETE 20-70
```

*Remarque* : la routine installée ci-dessus est un exemple, elle ne fait rien de particulier.



## BROCHAGE DE L'AY3-8912

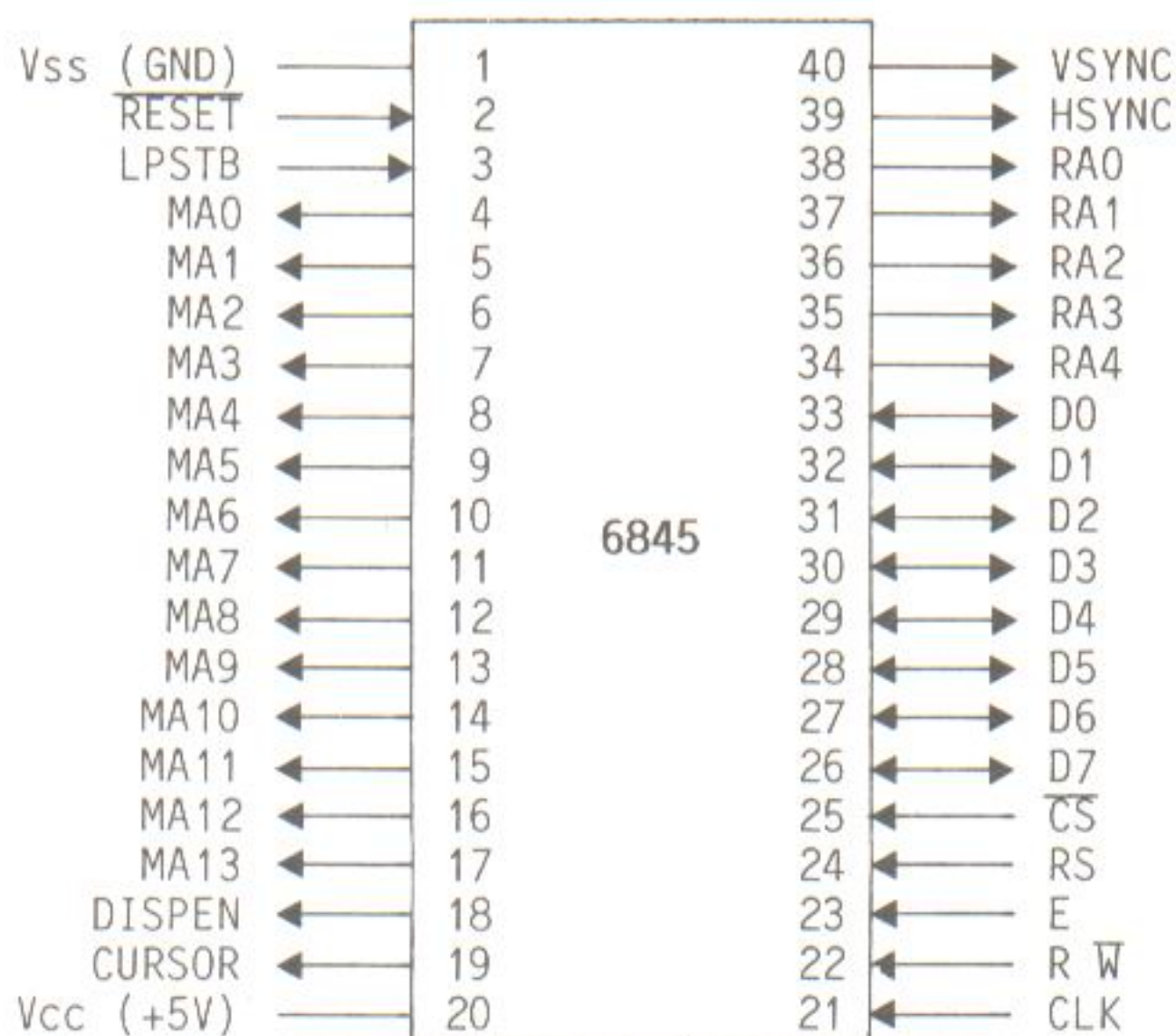
28 LEAD DUAL IN LINE  
AY3-8912

Top View

ANALOG CHANNEL C	1	28	DA0
TEST 1	2	27	DA1
Vcc (+5V)	3	26	DA2
ANALOG CHANNEL B	4	25	DA3
ANALOG CHANNEL A	5	24	DA4
Vss (GND)	6	23	DA5
IOA7	7	22	DA6
IOA6	8	21	DA7
IOA5	9	20	BC1
IOA4	10	19	BC2
IOA3	11	18	B DIR
IOA2	12	17	A8
IOA1	13	16	RESET
IOA0	14	15	CLOCK

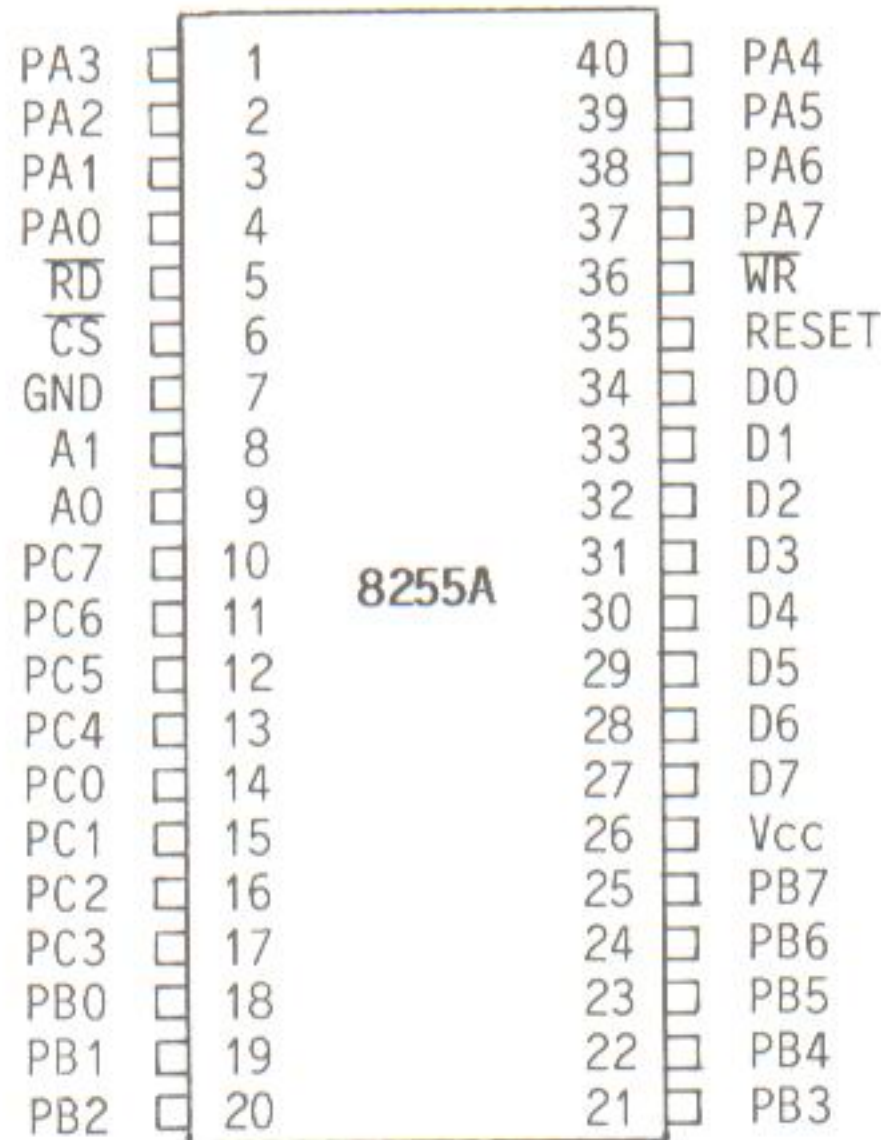


# BROCHAGE DU CRT 6845



Nom de broche	Description	Type
<u>D0-D7</u>	Bus de données.	Bidirectionnel
CS	Sélection du circuit.	Tristate
RS	Sélection du registre.	Input
R/W	Ecriture/lecture.	Input
E	Signal de synchronisation.	Input
CLK	Horloge.	Input
RESET	Initialisation.	Input
Vcc	Alimentation (+5V).	Input
MA0-MA13	Adresse mémoire (16K).	Output
RA0-RA4	Adresse ligne (scanning).	Output
HSYNC	Synchronisation horizontale.	Output
VSYNC	Synchronisation verticale.	Output
DISPEN	Validation affichage.	Output
CURSOR	Validation curseur.	Output
LPSTB	Présence photostyle.	Input

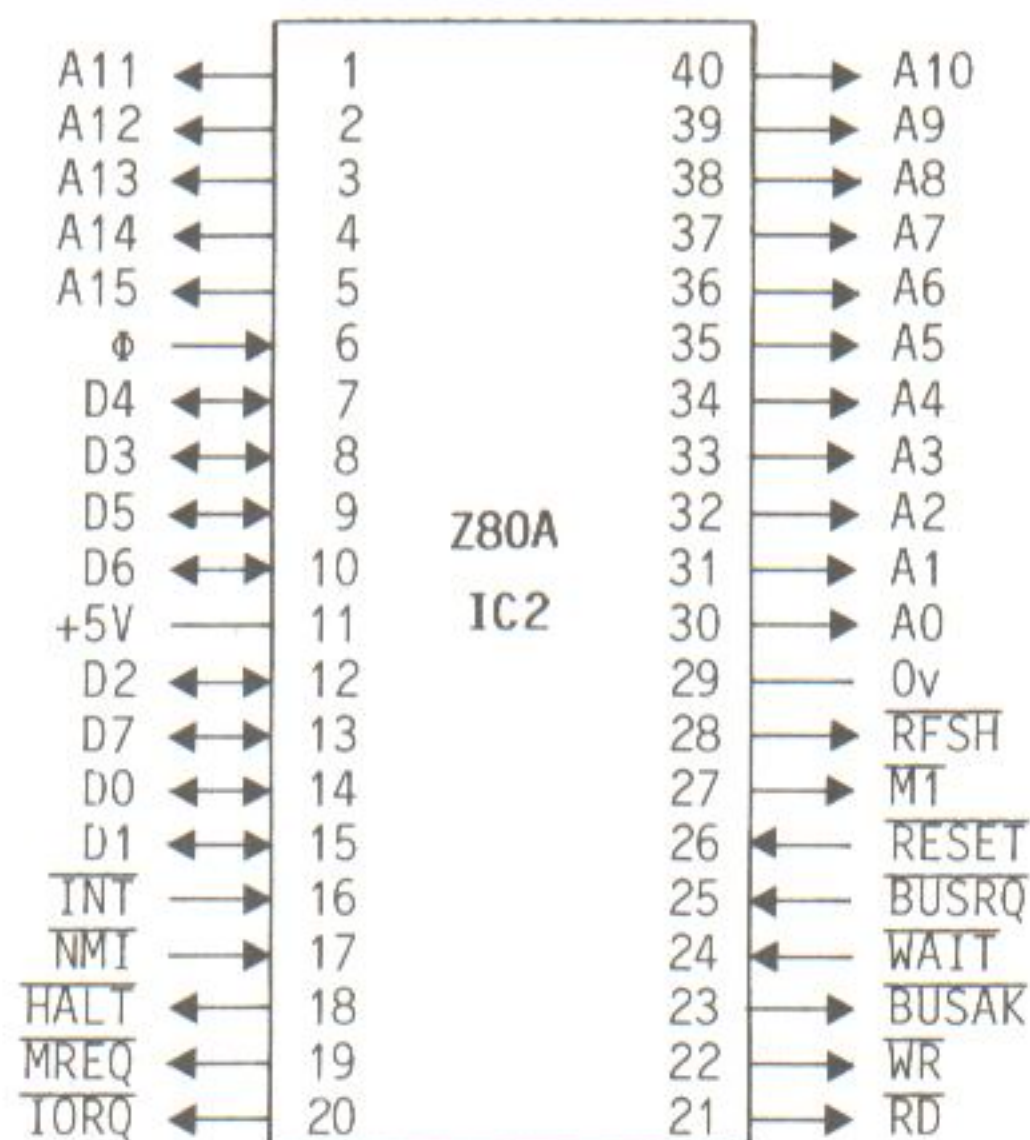




Nom de broche	Signification
D7-D0	Bus de données (bidirectionnel).
RESET	Initialisation.
CS	Sélection du circuit.
RD	Lecture input.
WR	Ecriture input.
A0, A1	Adresse de port.
PA7-PA0	Port A (bit).
PB7-PB0	Port B (bit).
PC7-PC0	Port C (bit).
Vcc	Alimentation (+5 volts).
GND	0 volt.



# BROCHAGE DU Z80



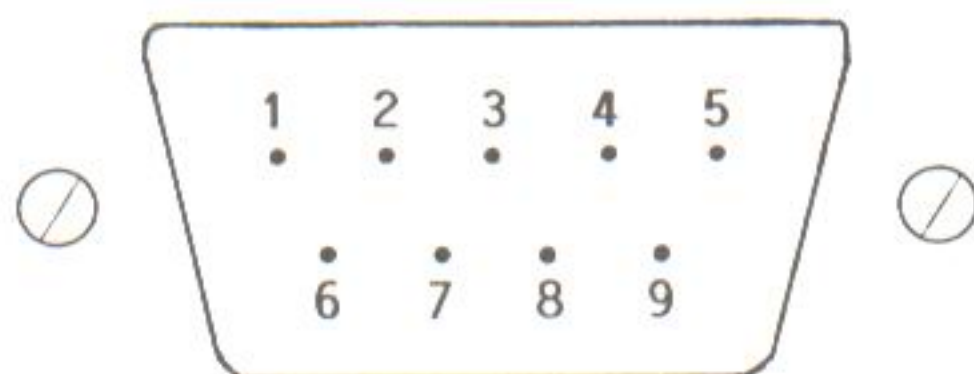
Broche	Signification
1	Bit d'adresse 11.
2	Bit d'adresse 12.
3	Bit d'adresse 13.
4	Bit d'adresse 14.
5	Bit d'adresse 15.
6	Horloge.
7	Bit de donnée 4.
8	Bit de donnée 3.
9	Bit de donnée 5.
10	Bit de donnée 6.
11	Tension de 5 volts régulés.
12	Bit de donnée 2.
13	Bit de donnée 7.
14	Bit de donnée 0.
15	Bit de donnée 1.
16	Interruptions masquables.
17	Interruptions non masquables.
18	Signal d'arrêt du microprocesseur.
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.



<i>Broche</i>	<i>Signification</i>
24	Demande d'attente au microprocesseur.
25	Demande d'accès direct mémoire.
26	Initialisation du microprocesseur.
27	Signal de début de cycle.
28	Rafraîchissement des mémoires dynamiques.
29	Tension 0 volt masse électriques.
30	Bit d'adresse 0.
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 10.



## LE CONNECTEUR POUR MANETTE DE JEUX



Broche 1	haut	Broche 6	bouton de tir 2
Broche 2	bas	Broche 7	bouton de tir 1
Broche 3	gauche	Broche 8	masse commune
Broche 4	droite	Broche 9	masse commune 2
Broche 5	disponible		



## LE CONNECTEUR SORTIE VIDEO



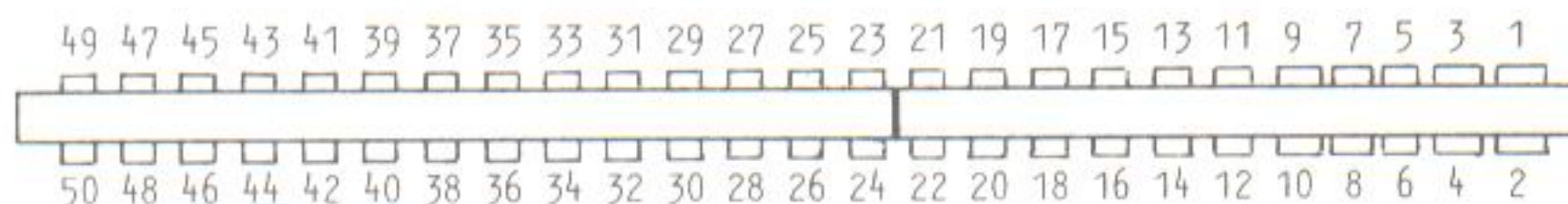
5      1  
4      6      2  
         3

Broche 1	rouge
Broche 2	vert
Broche 3	bleu

Broche 4	synchro
Broche 5	masse
Broche 6	luminance



## LE CONNECTEUR SORTIE EXPANSION

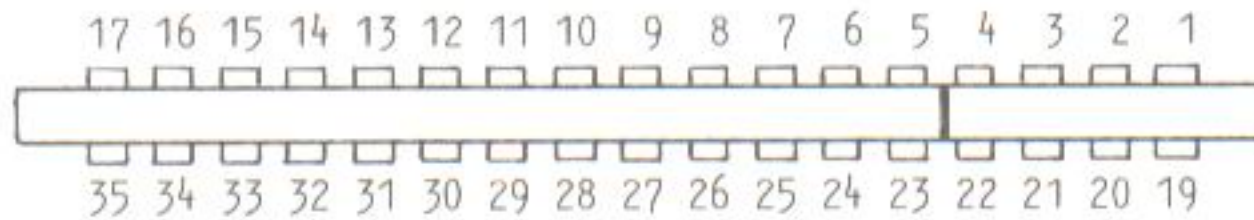


Broche	1	son
Broche	2	masse
Broche	3	A15
Broche	4	A14
Broche	5	A13
Broche	6	A12
Broche	7	A11
Broche	8	A10
Broche	9	A9
Broche	10	A8
Broche	11	A7
Broche	12	A6
Broche	13	A5
Broche	14	A4
Broche	15	A3
Broche	16	A2
Broche	17	A1
Broche	18	A0
Broche	19	D7
Broche	20	D6
Broche	21	D5
Broche	22	D4
Broche	23	D3
Broche	24	D2
Broche	25	D1

Broche	26	D0
Broche	27	+ 5 volts
Broche	28	<u>MREQ</u>
Broche	29	<u>M1</u>
Broche	30	<u>RFSH</u>
Broche	31	<u>TORQ</u>
Broche	32	<u>RD</u>
Broche	33	<u>WR</u>
Broche	34	<u>HALT</u>
Broche	35	<u>INT</u>
Broche	36	<u>NMI</u>
Broche	37	<u>BUSRD</u>
Broche	38	<u>BUSAK</u>
Broche	39	READY
Broche	40	<u>BUS RESET</u>
Broche	41	<u>RESET</u>
Broche	42	<u>ROMEN</u>
Broche	43	<u>ROMDIS</u>
Broche	44	<u>RAMRD</u>
Broche	45	<u>RAMDIS</u>
Broche	46	CURSOR
Broche	47	L. PEN
Broche	48	<u>EXP</u>
Broche	49	masse
Broche	50	0



## LE CONNECTEUR SORTIE IMPRIMANTE



Broche 1	STROBE	Broche 19	GND
Broche 2	D0	Broche 20	GND
Broche 3	D1	Broche 21	GND
Broche 4	D2	Broche 22	GND
Broche 5	D3	Broche 23	GND
Broche 6	D4	Broche 24	GND
Broche 7	D5	Broche 25	GND
Broche 8	D6	Broche 26	GND
Broche 9	D7	Broche 28	GND
Broche 11	BUSY	Broche 33	GND
Broche 14	GND (masse)	Les broches non décrites ne sont pas connectées.	
Broche 16	GND		



# **ANNEXES**



## INSTRUCTIONS ET FONCTIONS PROPRES AU CPC664

### Fonctions

COPYCHR\$	COPYCHR\$ (# numéro de canal) Copie dans une variable alphanumérique le caractère courant sous le curseur du canal spécifié.
DEC\$	DEC\$ (expression numérique, format) Exprime l'expression numérique avec le format spécifié (ce format est identique à celui de l'instruction PRINT USING). Cette fonction permet de mettre le résultat d'un USING dans une variable alphanumérique.
DERR	DERR Imprime le dernier numéro d'erreur produit.
SPC	SPC(n) Génère n espace : à utiliser avec PRINT.

### Instructions

CLEAR INPUT	CLEAR INPUT Cette instruction vide le tampon d'entrée. Elle enlève tous les caractères parasites qui pourraient s'y trouver.
CURSOR	CURSOR sémaphore système, sémaphore utilisateur. Cette instruction permet de forcer l'extinction ou l'allumage du curseur. Le sémaphore vaut 1 si le curseur doit être allumé, 0 sinon.
FILL	FILL encre Rempli la surface limitée par la couleur d'encre spécifiée.
FRAME	FRAME Synchronise l'écriture des graphiques avec le retour de balayage.



## INSTRUCTIONS ET FONCTIONS PROPRES AU CPC664

GRAPHICS	GRAPHICS PAPER encre / GRAPHICS PEN encre Positionne l'encre du papier graphique ou du crayon graphique sans toucher au crayon et au papier texte.
MASK	MASK nombre entier 0 à 255, nombre 0 à 7 Très belle instruction qui permet de spécifier la structure d'un trait pour dessiner en pointillé ou en trait mixte. Le premier octet spécifie la structure du trait sur 8 points (0 à 255), le second spécifie le point de départ à l'intérieur des 8 points. Exemple : pour dessiner en pointillé un point sur deux : MASK &X10101010,0 ou MASK 170,0
MID\$ à gauche	MID\$ (chaîne1, position, longueur) = chaîne2 Remplace à l'intérieur de chaîne1, à partir du caractère défini par position et sur une longueur définie par longueur, les caractères présents par les caractères de chaîne1.
ON BREAK CONT	ON BREAK CONT Empêche l'interruption du programme par l'appui sur la touche BREAK. Cette fonction est à utiliser avec circonspection sur des programmes terminés. Après lancement le seul moyen d'arrêter le programme est le RESET.



Les routines mathématiques contenues dans la ROM inférieure doivent être appelées souvent depuis la ROM BASIC pour effectuer toutes les fonctions BASIC de calcul (+,\*,/,sin,cos,...). Une série de vecteurs a été créée pour faciliter cet appel.

Les fonctions mathématiques du BASIC fonctionnent sur un accumulateur virtuel de six octets situé de B09F à B0A4. B09F contient le type de la variable (2=entier, 3=chaîne, 5=réel).

Une variable entière est codée sur deux octets en binaire signé.

Une variable réelle est plus complexe. Elle est représentée par cinq octets suivant un codage binaire particulier :

- exprimer le nombre en binaire ;
- compter le nombre de chiffres significatifs situés avant la virgule et lui ajouter 128 (80H). On a ainsi obtenu l'octet 5.
- supprimer le premier bit de gauche et convertir les sept autres bits en décimal. Si le nombre est négatif, ajouter 128 (80H). On obtient l'octet 4.
- pour obtenir les octets 3, 2 et 1, prendre les bits suivants par tranche de 8 et les convertir en décimal.

*Exemple* : codage de la variable réelle : -2527

2527 s'écrit en binaire 1 0011101 1111 (12 chiffres)

Octet 5 = 8C car :  $128 + 12 = 140 = 8C$

Octet 4 = 9D car : prendre les sept bits suivants :  
0011101 = 29 = 1D. Le nombre étant négatif, ajouter 128 :  $29 + 128 = 157 = 9D$ .

Octet 3 = F0 car : Les huit bits suivants sont 1111 (0000) = 240 = F0

Octet 2 et Octet 1 = 00 car il n'y a plus de bits.

-2527 se code donc : 00 00 F0 9D 8C



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES DU CPC664

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Signification</i>
BD5E	2F91	Copie les cinq octets pointés par DE vers la zone pointée par HL et passe dans A le contenu de l'octet qui se trouve à l'adresse HL-1 (type de variable).
BD61	2F9F	Conversion entier en flottant dans les cinq octets pointés par DE.
BD64	2FC8	Conversion du nombre binaire pointé par HL en nombre au format de l'accumulateur (5 octets).
BD67	2FD9	Transforme la valeur contenue dans les cinq octets pointés par HL en entier contenu dans HL.
BD6A	3001	Transforme la valeur contenue dans les cinq octets pointés par HL en un entier contenu dans les deux premiers octets pointés par HL.
BD6D	3014	Réalise la fonction FIX.
BD70	3055	Réalise la fonction INT.
BD73	305F	Fonction SGN (Routine utilisée par STR\$ et PRINT).
BD76	30C6	Routine de transformation (multiplie par 10 exp A).
BD79	34A2	Addition de deux réels. HL pointe sur une zone de cinq octets représentant un nombre au format réel (appelée ACCUM1). DE pointe sur une autre zone de cinq octets (appelée ACCUM2). A l'issue de la routine, HL pointe toujours sur ACCUM1 et ACCUM1 contient la valeur de ACCUM1 + ACCUM2.
BD7C	3159	Fonction RND.
BD7F	349E	Soustraction de deux réels. HL pointe sur une zone de cinq octets représentant un nombre au format réel (appelée ACCUM1). DE pointe sur une autre zone de cinq octets (appelée ACCUM2). A l'issue de la routine, HL pointe toujours sur ACCUM1 et ACCUM1 contient la valeur de ACCUM2-ACCUM1.



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES DU CPC664

Adresse vecteur	Adresse réelle	Signification
BD82	3577	Multiplication de deux réels. Comme ci-dessus mais ACCUM1 contient la valeur de $ACCUM1 * ACCUM2$ .
BD85	3604	Division de deux réels. Comme ci-dessus mais ACCUM1 contient la valeur de $ACCUM1 / ACCUM2$ .
BD88	3188	Fournir la dernière valeur RND.
BD8B	36DF	Comparaison de deux réels : Si $ACCUM1 > ACCUM2$ alors $A=1$ Si $ACCUM1 < ACCUM2$ alors $A=255$ Si $ACCUM1 = ACCUM2$ alors $A=0$
BD8E	3731	Négation d'un réel. HL pointe sur ACCUM1 qui contient la valeur de $-ACCUM1$ .
BD91	3727	Teste le réel contenu dans ACCUM1. HL pointe sur ACCUM1. Si $ACCUM1 > 0$ alors $A=1$ Si $ACCUM1 < 0$ alors $A=255$ Si $ACCUM1 = 0$ alors $A=0$
BD94	3345	Positionnement du mode de calcul d'angles en degrés ou en radians. Si $A=0$ on est en mode RADIANS. Si $A \neq 0$ on est en mode DEGRES.
BD97	2F73	En sortie, la zone pointée par HL en entrée contient la constante PI.
BD9A	32AC	Extraction de la racine carrée d'un nombre réel. En entrée, HL pointe sur une zone de cinq octets qui contient un nombre. En sortie, cette zone contient la racine carrée du nombre.
BD9D	32AF	Calcul de l'exposant d'un nombre réel. HL pointe sur ACCUM1 qui contient le nombre et DE pointe sur ACCUM2 qui contient l'exposant. En sortie, ACCUM1 contient la valeur de $ACCUM1^{exposant\ ACCUM2}$ .
BDA0	31B6	Calcul du logarithme népérien d'un nombre réel. HL pointe sur ACCUM1 qui contient



# LES VECTEURS D'APPEL DES ROUTINES MATHÉMATIQUES DU CPC664

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Signification</i>
		le nombre en entrée. En sortie, ACCUM1 contient la valeur du loga- rithme népérien du nombre.
BDA3	31B1	Calcul du logarithme en base 10 d'un nombre réel. HL pointe sur ACCUM1 qui contient le nombre en entrée. En sortie, ACCUM1 contient la valeur du loga- rithme décimal du nombre.
BDA6	322F	Calcul de l'exponentielle d'un nombre. HL pointe sur ACCUM1 qui en sortie contient la valeur de l'exponen- tielle du nombre.
BDA9	3353	Calcul du sinus d'un angle.
BDAC	3349	Calcul du cosinus d'un angle.
BDAF	33C8	Calcul de la tangente d'un angle.
BDB2	33D8	Calcul de l'arctangente d'un angle.
BDB5	2FD1	Routine d'évaluation.
BDB8	3136	Routine RND (B8E4 et B8E6) à l'ini- tialisation.
BDBB	3143	Routine utilisée pour la génération de nombres aléatoires.



# LES PRINCIPALES VARIABLES SYSTEME DU CPC664

CPC664

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
AC01	1	Sémaphore : 0 = auto, 1 = pas auto.
AC02	2	Numéro de la ligne courante (utilisé par AUTO).
AC04	2	Valeur de l'incrément entre deux lignes (AUTO).
AC09	1	Utilisé par l'instruction WIDTH.
AC0C	1	Utilisé par l'instruction NEXT.
AC12	2	Utilisé par l'instruction FOR.
AC14	2	Utilisé par le couple d'instructions WHILE-WEND.
AC16	11	Utilisé par l'instruction ON... GOTO.
AC8A	256	Tampon d'entrée clavier.
AD8C	2	Pointeur pour l'instruction RESUME
AD8E	2	Utilisé pour le traitement d'erreur.
AD90	1	Numéro d'erreur.
AD91	2	Adresse du dernier octet exécuté.
AD93	2	Adresse pour END, STOP et CONT.
AD98	1	Numéro d'erreur pour la fonction ON ERROR GOTO.
AD99	9	Paramètres utilisés par l'instruction SOUND.
ADF3	26	Table de déclaration des variables. Composée de 26 octets (un par lettre de l'alphabet). Chaque octet contient un code qui détermine le type par défaut de chaque variable commençant par cette lettre.
AE15	2	Adresse de la ligne courante pour le READ DATA.
AE17	2	Adresse de début de lecture de DATA pour RESTORE.
AE1B	2	Utilisé par ON ERROR GOTO.



## LES PRINCIPALES VARIABLES SYSTEME DU CPC664

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
AE1F	1	Sémaphore : 0 = TROFF, 1 = TRON.
AE55	2	Sauvegarde de DE pour l'utilisation de l'instruction CALL.
AE57	1	Sauvegarde de l'accumulateur pour l'utilisation de l'instruction CALL.
AE58	2	Sauvegarde de HL pour l'utilisation de l'instruction CALL.
AE5A	2	Sauvegarde de SP pour l'utilisation de l'instruction CALL.
AE5C	2	Utilisé par l'instruction ZONE (adresse).
AE5E	2	HIMEM (adresse supérieure pour le BASIC).
AE60	2	Utilisé par l'instruction SYMBOL (adresse).
AE64	2	Adresse du début du programme BASIC (par défaut 016F).
AE66	2	Adresse de fin du programme BASIC.
AE68	2	Adresse du début des tables de variables.
AE6A	2	Adresse de la table des variables simples.
AE6C	2	Adresse de la table des variables tableaux (DIM).
B06F	2	Adresse début de pile BASIC.
B09F	1	Type de l'accumulateur virtuel.
B0A0	5	Cinq octets utilisés par l'accumulateur virtuel.
B113	1	Mode radian/degré.
B118		Début de la zone des variables du gestionnaire cassette.
B118	1	Prompt message autorisé si 0, interdit si # 0.
B11A	1	Indicateur d'ouverture de fichier.
B11B	2	Adresse du tampon de 2K pour le catalogue.



# LES PRINCIPALES VARIABLES SYSTEME DU CPC664

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
B11D	2	Adresse tampon lecture.
B131	1	Type de fichier cassette.
B132	2	Adresse courante tampon de lecture.
B134	2	Emplacement mémoire des données.
B136	2	Longueur logique du fichier.
B15F	1	Type du stream d'écriture.
B162	2	Adresse du tampon d'écriture.
B176	2	Adresse courante du tampon d'écriture.
B1E5	1	Caractère de synchronisation.
B1E9	2	Vitesse d'écriture ou de lecture.
B1ED		Début de la zone des variables du gestionnaire sonore.
B2A6	240	15 x 16 octets avec les valeurs des enveloppes d'amplitude.
B396	240	15 x 16 octets avec les valeurs des enveloppes tonales.
B496	80	Table des valeurs des touches sans SHIFT ni CTRL.
B4E6	80	Table des valeurs des touches avec SHIFT.
B536	80	Table des valeurs des touches avec CTRL.
B586	80	Table des répétitions pour chaque touche.
B628	2	Utilisé pour le scanning (adresse).
B62A	1	Sauvegarde temporaire du caractère scanné (BBOC).
B633	1	Valeur de la vitesse de répétition des touches.
B634	1	Valeur du délai avant la répétition d'une touche.
B635	10	Table de scanning des touches.



## LES PRINCIPALES VARIABLES SYSTEME DU CPC664

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
B63B	1	Etat de la manette de jeux 1.
B63E	1	Etat de la manette de jeux 2.
B68B	2	Adresse de la table des touches sans SHIFT ni CTRL.
B68D	2	Adresse de la table des touches avec SHIFT.
B68F	2	Adresse de la table des touches avec CTRL.
B691	2	Adresse de la table de répétition des touches.
B693	2	Coordonnée de l'origine de l'axe X.
B695	2	Coordonnée de l'axe Y.
B697	2	Coordonnée graphique X.
B699	2	Coordonnée graphique Y.
B69B	2	Coordonnée X d'un bord de la fenêtre graphique.
B69D	2	Coordonnée X de l'autre bord de la fenêtre graphique.
B69F	2	Coordonnée Y d'un bord de la fenêtre graphique.
B6A1	2	Coordonnée Y de l'autre bord de la fenêtre graphique.
B6A3	1	Encre du crayon graphique.
B6A4	1	Encre du papier graphique.
B6A5	8	Quatre zones de deux octets utilisées comme mémoire temporaire de calcul pendant le traçage d'une ligne.
B6AD	2	Coordonnée X du point final pour le traçage d'une ligne.
B6AF	2	Coordonnée Y du point final pour le traçage d'une ligne.
B6B5	1	Numéro du STREAM
B726	1	Position ligne curseur.
B727	1	Position colonne curseur.



# LES PRINCIPALES VARIABLES SYSTEME DU CPC664

<i>Adresse</i>	<i>Longueur</i>	<i>Signification</i>
B728	1	Indicateur de fenêtre.
B729	1	Ligne début fenêtre courante.
B72A	1	Colonne début fenêtre courante.
B72B	1	Ligne fin fenêtre courante.
B72C	1	Colonne fin fenêtre courante.
B72E	1	0 = curseur autorisé, 255 = curseur interdit.
B72F	1	Encre courante pour le crayon.
B730	1	Encre courante pour le papier.
B731	1	0 = affichage du fond permis, 255 = affichage du fond interdit.
B734	2	Premier caractère et état de la table des matrices définie par l'utilisateur.
B736	2	Adresse de la table des matrices définie par l'utilisateur.
B763	96	Table des codes de contrôle.
B7C2	1	Octet pour l'encodage (masque) de l'encre.
B7C3	1	Mode écran (0, 1 ou 2).
B7C4	2	Offset de l'écran compris entre 0 et 7FF.
B7C6	1	Octet de poids fort du début de la mémoire réelle écran.
B7C7	1	Contient parfois un C3 (jump).
B7C8	2	Contient l'adresse pour le jump.
B7D2	1	Longueur de la première période de clignotement du bord.
B7D3	1	Longueur de la seconde période de clignotement du bord.
B7D4	32	Couleurs des encres (deux octets par couleur).
B7F7	1	Utilisé par BORDER.



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC664

La ROM inférieure contient les routines système (communication avec le matériel), les routines mathématiques et le générateur de caractères.

**Remarque :** les adresses qui correspondent aux routines déjà décrites en détail sont indiquées avec uniquement le point d'entrée mémoire vive correspondant suivi d'une \*.

Dès lors, nous vous conseillons de vous rapporter aux pages 81 à 111, pour de plus amples informations.

Les routines situées à une adresse identique dans le CPC464 sont indiquées par un signe = à la suite de l'adresse.

005C=	BCC8 *
0099=	BD0D *
00A3=	BD10 *
0163=	BCD7 *
016A=	BCDA *
0170=	BCDD *
0176=	BCE0 *
017D=	BCE3 *
0183=	BCE6 *
01B3=	BCE9 *
01C5=	BCEC *
01D2=	BCEF *
01E2=	BCF2 *
0219	BCFE *
0227	BCF5 *
0255	BCFB *
0276	BD01 *
0284	BCF8 *
028D	BD0A *
0294	BD04 *
029A	BD07 *
02A0	BCD1 *
02B1	BCD4 *
0326	BCCB *
0330	BCCE *
05D7	BD13 *
0606	BD16 *
066F	Message 64K MICROCOMPUTER (V2).
068B	Message copyright 1984 Amstrad Consumer Electronics PLC and Locomotive Software Ltd.
06F5	Message *** program load failed ***.
0728	Liste des compatibles Arnold, Amstrad, Orion, Schneider, Awa, Solavox, Saisho, Triumph, Isp.



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC664

CPC664

0766	BD1C	*	1070	BB4E	*
0776	BD22	*	1080	BB51	*
077C	BD25	*	10E0	BBB4	*
07A4	BD19	*	10FF	BBB7	*
07B0	BD1F	*	1156	BB6F	*
07D0	BD28	*	1161	BB72	*
080B	BD2B	*	116C	BB75	*
0825	BDF1	*	1178	BB78	*
0834	BD31	*	11C6	BB87	*
0848	BD2E	*	1204	BB66	*
0853	BD34	*	124E	BB69	*
08BB	BD37	*	125B	BDCD	*
0ABB	BBFF	*	125B	BDD0	*
0ACC	BC02	*	1261	BB8A	*
0AE5	BC0E	*	1261	BB8D	*
0B08	BC11	*	1272	BB81	*
0B13	BC14	*	127A	BB84	*
0B13	BDEB	*	1282	BB7B	*
0B33	BC05	*	1293	BB7E	*
0B38	BC08	*	12A2	BB90	*
0B52	BC0B	*	12A7	BB96	*
0B59	BC17	*	12B6	BB93	*
0B66	BC1A	*	12BC	BB99	*
0BAB	BC1D	*	12C2	BB9C	*
0C01	BC20	*	12D0	BBA5	*
0C0D	BC23	*	12EE	BBA8	*
0C1B	BC26	*	12FA	BBAB	*
0C35	BC29	*	1327	BBAE	*
0C51	BC59	*	1331	BB5D	*
0C6D	BDE8	*	1347	BDD3	*
0C70	BC5C	*	1377	BB9F	*
0C86	BDE5	*	1384	BBA2	*
0C8A	BC2C	*	13A4	BB63	*
0CA3	BC2F	*	13A8	BB60	*
0CE6	BC3E	*	13BA	BDD6	*
0CEA	BC41	*	13FA	BB5A	*
0CEE	BC32	*	1406	BDD9	*
0CF3	BC38	*	144E	BB57	*
0D16	BC35	*	1455	BB54	*
0D1B	BC3B	*	14D0	BBB1	*
0DB5	BC44	*	154B	BB6C	*
0DB9	BC47	*	15A4	BBBA	*
0DE1	BC4A	*	15D3	BBBD	*
0DFC	BC4D	*	15F7	BBC3	*
0E40	BC50	*	15FA	BBC0	*
0EF5	BC53	*	1602	BBC6	*
0F26	BC56	*	160A	BBC9	*
0F8F	BC5F	*	1618	BBCC	*
0F97	BC62	*	16A1	BBCF	*



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC664

16E6	BBD2	*
1713	BBD5	*
1729	BBD8	*
1732	BBDB	*
1763	BBDE	*
176A	BBE4	*
1771	BBE1	*
1776	BBE7	*
177C	BBED	*
177F	BBEA	*
1782	BDDC	*
1790	BBF3	*
1793	BBF0	*
1796	BDDF	*
17A2	BBF9	*
17A5	BBF6	*
17B0	BDE2	*
193C	BBFC	*
1B5C	BB00	*
1B98	BB03	*
1BBF	BB06	*
1BC5	BB09	*
1BFA	BB0C	*
1C04	BB15	*
1C3C	Valeur par défaut des touches étendues (RUN pour CTRL CR).	
1C46	BB0F	*
1CB3	BB12	*
1CDB	BB18	*
1CE1	BB1B	*
1D38	BB21	*
1DB8	BDEE	*
1DE5	BB24	*
1DF2	BB42	*
1DF6	BB3F	*
1DFA	BB45	*
1E0B	BB48	*
1E19	BB4B	*
1E2F	BB3C	*
1E34	BB39	*
1E45	BB1E	*
1EC4	BB2A	*
1EC9	BB30	*
1ECE	BB36	*
1ED8	BB27	*
1EDD	BB2D	*
1EE2	BB33	*
1EEF	Table des valeurs par défaut des touches clavier.	



1FE9	BCA7 *
2050	BCB6 *
206B	BCB9 *
2114	BCAA *
21AC	BCB3 *
21CE	BCAD *
21EB	BCB0 *
2495	BCBC *
249A	BCBF *
24A6	BCC2 *
24AB	BCC5 *
24BC	BC65 *
24CE	BC68 *
24E1	BC6B *
288B	BC77, BC7A, BC7D, BC80, BC83, BC86, BC89, BC8C, BC8F, BC92, BC95, BC98, BC9B *
	Routines cassette & disque.
2935	Message press play then any key.
294B	Message error.
2955	Message REC.
2958	Message and.
295D	Message Read.
2963	Message write.
296A	Message Rewind.
2970	Message tape.
2975	Message found.
297D	Message loading.
2985	Message saving.
298D	Message OK.
2990	Message Block.
2996	Message Unnamed.
299D	Message File.
29A6	BCA1 *
29AF	BC9E *
29C1	BCA4 *
2BBB	BC6E *
2BBF	BC71 *
2BC1	BC74 *
2F73	BD97 * PI
2F78	CONSTANTE PI
2F91	BD5E *
2F9F	BD61 *
2FC8	BD64 *
2FD1	BDB5 *
2FD9	BD67 *
3001	BD6A *
3014	BD6D *
3055	BD70 *
305F	BD73 *



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC664

30C6	BD76 *
30F5	Table des puissances de 10. Treize entrées de 5 octets pour les valeurs de 10 à 10E13.
3136	BDB8 * RND INT
3143	BDBB * RND SEED
3159	BD7C * RND
3188	BD88 * RND
31B1	BDA3 * LOG10
31B6	BDA0 * LOG
31EE	Constante pour calcul du LOG (4 x 5 octets).
3220	Valeur codée de 1/SQR(2).
3225	Valeur codée de LOG(2) (0,693147181).
322A	Valeur codée de LOG10(2) (0,301029996).
322F	BDA6 * EXP
329D	Constante 1,44269504.
32A2	Constante 88,0296919.
32A7	Constante -88,7228391.
32AC	BD9A * SQR
32AF	BD9C * PUISSANCE
3345	BD94 * DEG-RAD
3349	BDAC * COS
3353	BDA9 * SIN
3382	Table de 6 nombres codés sur 5 octets chacun pour le calcul des sinus et des cosinus.
33B4	Table de 4 nombres codés sur 5 octets chacun pour le calcul des sinus et des cosinus.
33C8	BDAF * TAN
33D8	BDB2 * ATN
33EE	Table de 11 nombres codés sur 5 octets chacun pour le calcul de l'arc-tangente.
349E	BD7F * SOUSTRACTION
34A2	BD79 * ADDITION
3577	BD82 * MULTIPLICATION
3604	BD85 * DIVISION
36DF	BD8B * COMPARAISON
3727	BD91 * SGN
3731	BD8E * CHANGEMENT SIGNE
3800	Début de la table du générateur de caractères (256 x 8 octets).
3FFF	Fin de la table.



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC664

La ROM supérieure contient toutes les routines de traitement de tous les mots clés BASIC.

C006	Initialisation + envoi du message BASIC 1.1
C033	Message BASIC 1.1
C046	Fonction EDIT
C058	Entrée principale (affichage du READY).
C0D7	Message READY
C0EA	AUTO
C128	NEW
C12F	CLEAR
C23C	PAPER
C227	PEN
C24B	BORDER
C254	INK
C278	MODE
C283	CLS
C29B	COPYCHR\$
C2A4	VPOS
C2A8	POS
C302	LOCATE
C311	WINDOW
C346	TAG
C34D	TAGOFF
C363	CURSOR
C42D	WIDTH
C452	EOF
C4E1	ORIGIN
C509	CLG
C515	FILL
C532	MOVE
C537	MOVER
C53C	DRAW
C541	DRAWR
C546	PLOT
C54B	PLOTR
C574	TEST
C579	TESTR
C59D	GRAPHICS
C5C3	MASK
C5D7	FOR
C6A5	NEXT
C76A	IF
C789	GOTO
C78F	GOSUB
C7B3	RETURN
C7EA	WHILE
C81D	WEND
C885	ON



# ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC664

C979	ON BREAK
C99A	DI
C9A0	EI
C9F8	ON SQ
CA25	AFTER
CA2D	EVERY
CA53	REMAIN
CB54	ERROR
CBF4	Message UNDEFINED LINE
CC04	Routine envoi message BREAK in.
CC1F	Message BREAK
CC25	Message IN
CC29	STOP
CC34	END
CC96	CONT
CCCD	ON ERROR
CCD8	RESUME
CD17	Table des messages d'erreur (MORCEAU DE MOT).
CFF0	Table des points d'entrée des opérations arithmétiques et logiques.
D11A	Table des points d'entrée des fonctions EOF, ERR, HIMEM, INKEY\$, PI, RND, TIME, XPOS et YPOS.
D12E	DERR
D133	ERR
D14B	HIMEM
D164	XPOS
D16B	YPOS
D1E8	Table des points d'entrée des fonctions.
D242	MIN
D246	MAX
D26D	ROUND
D2AB	OPENOUT
D2B7	OPENIN
D2F0	CLOSEIN
D2F8	CLOSEOUT
D316	SOUND
D373	RELEASE
D37E	SQ
D3A1	ENV
D3D7	ENT
D459	INKEY
D473	JOY
D489	KEY DEF
D4DE	SPEED
D520	PI
D52C	DEG
D530	RAD
D534	SQR



D539	Routine d'élévation à une puissance.
D563	EXP
D568	LOG10
D56D	LOG
D572	SIN
D577	COS
D57C	TAN
D581	ATN
D587	Message RANDOM NUMBER SEED ?
D59C	RANDOMIZE
D5C4	RND
D653	DEFSTR
D657	DEFINT
D65B	DEFREAL
D691	LET
D6B9	DIM
D9F4	ERASE
DB18	LINE
DB48	INPUT
DB7F	Message ?redo from start
DCCD	RESTORE
DCDF	READ
DEC6	TRON
DECA	TROFF
DEE5	Table des points d'entrée des mots clés BASIC.
DFA8	Fin de la table.
E0C8	Table des mots clés qui peuvent être suivis d'un numéro de ligne (GOTO, RESTORE, AUTO, EDIT,...).
E1D2	LIST
E3AD	Routine de positionnement sur la table des lettres pour rechercher des mots clés.
E3F0	Routine de test qui vérifie si un mot clé se trouve dans la table.
E41D	Table des adresses pour chacune des 26 lettres de l'alphabet.
E451	Table des mots clés avec leur code.
E73A	Fin de la table.
E7F3	DELETE
E8A3	RENUM
E9A8	DATA
E9AC	REM
EA7D	RUN
EABA	LOAD
EB02	CHAIN
EB59	MERGE
ECE1	SAVE
F20D	PEEK
F214	POKE



# ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC664

F21E	INP
F228	OUT
F232	WAIT
F261	CALL
F2A2	ZONE
F2A9	PRINT
F383	PRINT USING
F50D	WRITE
F570	MEMORY
F784	SYMBOL
F8EC	LOWER\$
F8F1	Routine de conversion en minuscules.
F8FA	UPPER\$
F964	BIN\$
F969	HEX\$
F98F	DEC\$
F9BC	STR\$
F9D3	LEFT\$
F9D8	RIGHT\$
FA07	MID\$
FA69	LEN
FA6E	ASC
FA74	CHR\$
FA7E	INKEY\$
FA8D	STRING\$
FAAD	SPACE\$
FABE	VAL
FAE5	INSTR
FC53	FRE
FD0C	Addition +
FD21	Soustraction -
FD35	Multiplication *
FD52	Division /
FD67	Division entière \
FD79	Modulo (reste de la division).
FD87	Fonction AND (ET LOGIQUE).
FD92	Fonction OR (OU LOGIQUE).
FD9C	Fonction XOR (OU EXCLUSIF).
FDB0	ABS
FE0E	FIX
FE13	INT
FEB6	CINT
FEEB	UNT
FF14	CREAL
FF1B	Nettoyage de l'accumulateur.
FF2A	SGN
FF32	Positionnement d'un entier dans l'accumulateur.
FF3E	Conversion en réel.
FF45	Met le type de variable dans C.



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC664

FF4B	Met le type de variable dans A.
FF83	Copie l'accumulateur dans la zone pointé par DE.
FF92	Teste si majuscule.
FF9C	Teste si numérique.
FFAB	Conversion en majuscule.
FFCA	Compare A et le contenu de HL.
FFD8	Compare HL et DE.
FFDE	Compare HL et BC.
FFE4	DE = HL - DE
FFF2	LDIR
FFF8	LDDR
FFFB	JP (HL)
FFFC	Retour au contenu de BC.
FFFE	Retour au contenu de DE.



# ADRESSES REELLES ROM DU CPC664

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>
BB00	1B5C	BB03	1B98	BB06	1BBF
BB09	1BC5	BB0C	1BFA	BB0F	1C46
BB12	1CB3	BB15	1C04	BB18	1CDB
BB1B	1CE1	BB1E	1E45	BB21	1D38
BB24	1DE5	BB27	1ED8	BB2A	1EC4
BB2D	1EDD	BB30	1EC9	BB33	1EE2
BB36	1ECE	BB39	1E34	BB3C	1E2F
BB3F	1DF6	BB42	1DF2	BB45	1DFA
BB48	1E0B	BB4B	1E19	BB4E	1070
BB51	1080	BB54	1455	BB57	144E
BB5A	13FA	BB5D	1331	BB60	13A8
BB63	13A4	BB66	1204	BB69	124E
BB6C	154B	BB6F	1156	BB72	1161
BB75	116C	BB78	1178	BB7B	1282
BB7E	1293	BB81	1272	BB84	127A
BB87	11C6	BB8A	1261	BB8D	1261
BB90	12A2	BB93	12B6	BB96	12A7
BB99	12BC	BB9C	12C2	BB9F	1377
BBA2	1384	BBA5	12D0	BBA8	12EE
BBAB	12FA	BBAE	1327	BBB1	14D0
BBB4	10E0	BBB7	10FF	BBBA	15A4
BBBD	15D3	BBC0	15FA	BBC3	15F7
BBC6	1602	BBC9	160A	BBC C	1618
BBCF	16A1	BBD2	16E6	BBD5	1713
BBD8	1729	BBDB	1732	BBDE	1763
BBE1	1771	BBE4	176A	BBE7	1776
BBEA	177F	BBED	177C	BBF0	1793
BBF3	1790	BBF6	17A5	BBF9	17A2
BBFC	193C	BBFF	0ABB	BC02	0ACC
BC05	0B33	BC08	0B38	BC0B	0B52
BC0E	0AE5	BC11	0B08	BC14	0B13
BC17	0B59	BC1A	0B66	BC1D	0BAB
BC20	0C01	BC23	0C0D	BC26	0C1B
BC29	0C35	BC2C	0C8A	BC2F	0CA3
BC32	0CEE	BC35	0D16	BC38	0CF3
BC3B	0D1B	BC3E	0CE6	BC41	0CEA
BC44	0DB5	BC47	0DB9	BC4A	0DE1
BC4D	0DFC	BC50	0E40	BC53	0EF5
BC56	0F26	BC59	0C51	BC5C	0C70
BC5F	0F8F	BC62	0F97	BC65	24BC
BC68	24CE	BC6B	24E1	BC6E	2BBB
BC71	2BBF	BC74	2BC1	BC77	288B



# ADRESSES REELLES ROM DU CPC664

<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>	<i>Adresse vecteur</i>	<i>Adresse réelle</i>
BC7A	288B	BC7D	288B	BC80	288B
BC83	288B	BC86	288B	BC89	288B
BC8C	288B	BC8F	288B	BC92	288B
BC95	288B	BC98	288B	BC9B	288B
BC9E	29AF	BCA1	29A6	BCA4	29C1
BCA7	1FE9	BCAA	2114	BCAD	21CE
BCB0	21EB	BCB3	21AC	BCB6	2050
BCB9	206B	BCBC	2495	BCBF	249A
BCC2	24A6	BCC5	24AB	BCC8	005C
BCCB	0326	BCCE	0330	BCD1	02A0
BCD4	02B1	BCD7	0163	BCDA	016A
BCDD	0170	BCE0	0176	BCE3	017D
BCE6	0183	BCE9	01B3	BCEC	01C5
BCEF	01D2	BCF2	01E2	BCF5	0227
BCF8	0284	BCFB	0255	BCFE	0219
BD01	0276	BD04	0294	BD07	029A
BD0A	028D	BD0D	0099	BD10	00A3
BD13	05D7	BD16	0606	BD19	07A4
BD1C	0766	BD1F	07B0	BD22	0776
BD25	077C	BD28	07D0	BD2B	080B
BD2E	0848	BD31	0834	BD34	0853
BD37	08BB	BD3A	1D3C	BD3D	1BFE
BD40	145C	BD43	15E8	BD46	19D1
BD49	17AC	BD4C	17A8	BD4F	1626
BD52	19D5	BD55	0B41	BD58	07FC
BD5B	2C02	BD5E	2F91	BD61	2F9F
BD64	2FC8	BD67	2FD9	BD6A	3001
BD6D	3014	BD70	3055	BD73	305F
BD76	30C6	BD79	34A2	BD7C	3159
BD7F	349E	BD82	3577	BD85	3604
BD88	3188	BD8B	36DF	BD8E	3731
BD91	3727	BD94	3345	BD97	2F73
BD9A	32AC	BD9D	32AF	BDA0	31B6
BDA3	31B1	BDA6	322F	BDA9	3353
BDAC	3349	BDAF	33C8	BDB2	33D8
BDB5	2FD1	BDB8	3136	BDBB	3143

CPC664



# ADRESSES D'EXECUTION DES MOTS CLES DU BASIC DU CPC664

<i>Mot-clé</i>	<i>Adresse</i>	<i>Mot-clé</i>	<i>Adresse</i>
ABS	FDB0	FRE	FC53
AFTER	CA25	GOSUB	C78F
ASC	FA6E	GOTO	C789
ATN	D581	HEX\$	F969
AUTO	C0EA	HIMEM	D14B
BIN\$	F964	IF	C76A
BORDER	C24B	INSTR	FAE5
CALL	F261	INK	C254
CAT	D299	INKEY	D459
CHAIN	EB02	INKEY\$	FA7E
CHR\$	FA74	INP	F21E
CINT	FEB6	INPUT	DB48
CLEAR	C12F	INT	FE13
CLG	C509	JOY	D473
CLOSEIN	D2F0	KEY	D489
CLOSEOUT	D2F8	LEFT\$	F9D3
CLS	C283	LEN	FA69
CONT	CC96	LET	D691
COS	D577	LINE	DB18
CREAL	FF14	LIST	E1D2
DATA	E9A8	LOAD	EABA
DEC\$	F9F8	LOCATE	C302
DEF	D174	LOG	D56D
DEFINT	D657	LOG10	D568
DEFREAL	D65B	LOWER\$	F8EC
DEFSTR	D653	MAX	D246
DEG	D52C	MEMORY	F570
DELETE	E7F3	MERGE	EB59
DI	C99A	MID\$	FA07
DIM	D6B9	MIN	D242
DRAW	C53C	MODE	C278
DRAWR	C541	MOVE	C532
EDIT	C046	MOVER	C537
EI	C9A0	NEXT	C6A5
ELSE	E9B2	NEW	C128
END	CC34	ON	C885
ENT	D3D7	ON BREAK	C979
ENV	D3A1	ON ERROR	CCCD
EOF	C452	ON SQ	C9F8
ERASE	D9F4	OPENIN	D2B7
ERR	D133	OPENOUT	D2AB
ERROR	CB54	ORIGIN	C4E1
EVERY	CA2D	OUT	F228
EXP	D563	PAPER	C23C
FIX	FE0E	PEEK	F2QD
FOR	C5D7	PEN	C227



# ADRESSES D'EXECUTION DES MOTS CLES DU BASIC DU CPC664

<i>Mot-clé</i>	<i>Adresse</i>	<i>Mot-clé</i>	<i>Adresse</i>
PI	D520	SQ	D37E
PLOT	C546	SQR	D534
PLOTR	C54B	STOP	CC29
POKE	F214	STR\$	F9CB
POS	C2AD	STRING\$	FA8D
PRINT	F2A9	SYMBOL	F784
' (REM)	E9AC	TAG	C346
RAD	D530	TAGOFF	C34D
RANDOMIZE	D59C	TAN	D57C
READ	DCDF	TEST	C574
RELEASE	D373	TESTR	C579
REM	E9AC	TIME	D13C
REMAIN	CA53	TROFF	DEC6
RENUM	E8A3	TRON	DECA
RESTORE	DCCD	UNT	FEEB
RESUME	CCD8	UPPER\$	F8FA
RETURN	C7B3	VAL	FABE
RIGHT\$	F9D8	VPOS	C2A4
RND	D5C4	WAIT	F2E2
ROUND	D26D	WEND	C81D
RUN	EA7D	WHILE	C7EA
SAVE	ECE1	WIDTH	C42D
SGN	FF2A	WINDOW	C311
SIN	D572	WRITE	F50D
SOUND	D316	XPOS	D164
SPACE\$	FAAD	YPOS	D16B
SPEED	D4DE	ZONE	F2A2

*Nouveaux Mots-clés :*

COPYCHR\$	C29B	FRAME	BD19
CURSOR	C363	GRAPHICS	C59D
DERR	D12E	MASK	C5C3
FILL	C515		

CPC664



La version CPC6128 présente quelques différences par rapport à la version CPC664.

La ROM inférieure est quelque peu modifiée. La ROM supérieure est presque identique mais les adresses sont toutes légèrement décalées.

Cependant, les vecteurs systèmes, les variables systèmes et les routines mathématiques sont exactement identiques en fonction et adresse à ceux du CPC664.

Les feuilles suivantes portent donc sur les modifications subies par les ROMs BIOS et BASIC.



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC6128

La ROM inférieure contient les routines système (communication avec le matériel), les routines mathématiques et le générateur de caractères.

**Remarque :** les adresses qui correspondent aux routines déjà décrites en détail sont indiquées avec uniquement le point d'entrée mémoire vive correspondant suivi d'une \*.

Dès lors, nous vous conseillons de vous reporter aux pages 81 à 111, pour de plus amples informations.

Les routines situées à une adresse identique dans le CPC664 sont indiquées par un signe = à la suite de l'adresse.

005C=	BCC8	*
0099=	BD0D	*
00A3=	BD10	*
0163=	BCD7	*
016A=	BCDA	*
0170=	BCDD	*
0176=	BCE0	*
017D=	BCE3	*
0183=	BCE6	*
01B3=	BCE9	*
01C5=	BCEC	*
01D2=	BCEF	*
01E2=	BCF2	*
0219=	BCFE	*
0227=	BCF5	*
0255=	BCFB	*
0276=	BD01	*
0284=	BCF8	*
028D=	BD0A	*
0294=	BD04	*
029A=	BD07	*
02A0=	BCD1	*
02B1=	BCD4	*
0326=	BCCB	*
0330=	BCCE	*
05ED	BD13	*
061C	BD16	*
0688	Message 128K MICROCOMPUTER (V3).	
068B	Message copyright 1985 Amstrad Consumer Electronics PLC and Locomotive Software Ltd.	
06F5	Message *** program load failed ***	
0728	Liste des compatibles Arnold, Amstrad, Orion, Schneider, Awa, Solavox, Saisho, Triumph, Isp.	



# ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC6128

0776	BD1C	*	1084	BB51	*
0786	BD22	*	10E4	BBB4	*
078C	BD25	*	1103	BBB7	*
07B4	BD19	*	115A	BB6F	*
07C0	BD1F	*	1165	BB72	*
07E0	BD28	*	1170	BB75	*
081B	BD2B	*	117C	BB78	*
0835	BDF1	*	11CA	BB87	*
0844	BD31	*	1208	BB66	*
0858	BD2E	*	1252	BB69	*
0863	BD34	*	125F	BDCD	*
08BD	BD37	*	125F	BDD0	*
0ABF	BBFF	*	1265	BB8A	*
0AD0	BC02	*	1265	BB8D	*
0AE9	BC0E	*	1276	BB81	*
0B0C	BC11	*	127E	BB84	*
0B17	BC14	*	1286	BB7B	*
0B17	BDEB	*	1297	BB7E	*
0B37	BC05	*	12A6	BB90	*
0B3C	BC08	*	12AB	BB96	*
0B56	BC0B	*	12BA	BB93	*
0B5D	BC17	*	12C0	BB99	*
0B6A	BC1A	*	12C6	BB9C	*
0BAF	BC1D	*	12D4	BBA5	*
0C05	BC20	*	12E2	BBA8	*
0C11	BC23	*	12FE	BBAB	*
0C1F	BC26	*	132B	BBAE	*
0C39	BC29	*	1335	BB5D	*
0C55	BC59	*	134B	BDD3	*
0C71	BDE8	*	137B	BB9F	*
0C74	BC5C	*	1388	BBA2	*
0C8A	BDE5	*	13A8	BB63	*
0C8E	BC2C	*	13AC	BB60	*
0CA7	BC2F	*	13BE	BDD6	*
0CEA	BC3E	*	13FE	BB5A	*
0CEE	BC41	*	140A	BDD9	*
0CF2	BC32	*	1452	BB57	*
0CF7	BC38	*	1459	BB54	*
0D1A	BC35	*	14D4	BBB1	*
0D1F	BC3B	*	154F	BB6C	*
0DB9	BC44	*	15A8	BBBA	*
0DBD	BC47	*	15D7	BBBD	*
0DE5	BC4A	*	15FB	BBC3	*
0E00	BC4D	*	15FE	BBC0	*
0E44	BC50	*	1606	BBC6	*
0EF9	BC53	*	160E	BBC9	*
0F2A	BC56	*	161C	BBCC	*
0F93	BC5F	*	16A5	BBCF	*
0F9B	BC62	*	16EA	BBD2	*
1074	BB4E	*	1717	BBD5	*

CPC6128



## ADRESSES PRINCIPALES DE LA ROM INFÉRIEURE DU CPC6128

172D	BBD8	*
1736	BBDB	*
1767	BBDE	*
176E	BBE4	*
1775	BBE1	*
177A	BBE7	*
1780	BBED	*
1783	BBEA	*
1786	BDDC	*
1794	BBF3	*
1797	BBF0	*
179A	BDDF	*
17A6	BBF9	*
17A9	BBF6	*
17B4	BDE2	*
1940	BBFC	*

1B5C= A partir de cette adresse, les routines de la ROM inférieure du CPC6128 commencent aux mêmes points d'entrée que celles de la ROM inférieure du CPC664.



## ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC6128

La ROM supérieure contient toutes les routines de traitement de tous les mots clés BASIC.

C006	Initialisation + envoi du message BASIC 1.1
C033	Message BASIC 1.1
C046	Fonction EDIT
C058	Entrée principale (affichage du READY)
C0D7	Message READY
C0EA	AUTO
C128	NEW
C12F	CLEAR
C224	PEN
C239	PAPER
C248	BORDER
C251	INK
C275	MODE
C280	CLS
C298	COPYCHR\$
C2A1	VPOS
C2A5	POS
C2FF	LOCATE
C30E	WINDOW
C343	TAG
C34A	TAGOFF
C360	CURSOR
C42A	WIDTH
C44F	EOF
C4DE	ORIGIN
C506	CLG
C512	FILL
C52F	MOVE
C534	MOVER
C539	DRAW
C53E	DRAWR
C543	PLOT
C548	PLOTR
C571	TEST
C576	TESTR
C59A	GRAPHICS
C5C0	MASK
C5D4	FOR
C6A2	NEXT
C767	IF
C786	GOTO
C78C	GOSUB
C7B0	RETURN
C7E7	WHILE
C81A	WEND
C882	ON



# ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC6128

C976	ON BREAK
C997	DI
C99D	EI
C9F5	ON SQ
CA22	AFTER
CA2A	EVERY
CA50	REMAIN
CB51	ERROR
CBF1	Message UNDEFINED LINE
CC01	Routine envoi message BREAK in
CC1C	Message BREAK
CC22	Message IN
CC26	STOP
CC31	END
CC93	CONT
CCCA	ON ERROR
CCD5	RESUME
CD14	Table des messages d'erreur (MORCEAU DE MOT).
CFED	Table des points d'entrée des opérations. arithmétiques et logiques.
D01D	-
D028	NOT
D036	+
D117	Table des points d'entrée des fonctions EOF, ERR, HIMEM, INKEY\$, PI, RND, TIME, XPOS et YPOS
D12B	DERR
D130	ERR
D139	TIME
D142	ERL
D148	HIMEM
D14E	@
D161	XPOS
D168	YPOS
D171	DEF
D1E5	Table des points d'entrée des fonctions
D23F	MIN
D243	MAX
DE6A	ROUND
D296	CAT
D2A8	OPENOUT
D2B4	OPENIN
D2ED	CLOSEIN
D2F5	CLOSEOUT
D313	SOUND
D370	RELEASE
D37B	SQ
D39E	ENV
D3D4	ENT



D456	INKEY
D470	JOY
D486	KEY DEF
D4DB	SPEED
D51D	PI
D529	DEG
D52D	RAD
D531	SQR
D536	Routine d'élévation à une puissance.
D560	EXP
D565	LOG10
D56A	LOG
D56F	SIN
D574	COS
D579	TAN
D57E	ATN
D584	Message RANDOM NUMBER SEED ?
D599	RANDOMIZE
D5C1	RND
D650	DEFSTR
D654	DEFINT
D658	DEFREAL
D68E	LET
D6B6	DIM
D9F0	ERASE
DB13	LINE
DB43	INPUT
DB7A	Message ?redo from start
DCC8	RESTORE
DCDA	READ
DEC1	TRON
DEC5	TROFF
DEE0	Table des points d'entrée des mots clés BASIC
DFA3	Fin de la table
E0C3	Table des mots clés qui peuvent être suivis d'un numéro de ligne (GOTO, RESTORE, AUTO, EDIT,...)
E1CD	LIST
E3A8	Routine de positionnement sur la table des lettres pour rechercher des mots clés
E3EB	Routine de test qui vérifie si un mot clé se trouve dans la table
E418	Table des adresses pour chacune des 26 let- tres de l'alphabet
E44C	Table des mots clés avec leur code
E735	Fin de la table
E7EE	DELETE
E89E	RENUM
E9A3	DATA



# ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC6128

E9A7	REM
E9AD	ELSE
EA78	RUN
EAB5	LOAD
EAFD	CHAIN
EB54	MERGE
ECDC	SAVE
F208	PEEK
F20F	POKE
F219	INP
F223	OUT
F229	WAIT
F25C	CALL
F29D	ZONE
F2A9	PRINT
F383	PRINT USING
F508	WRITE
F56B	MEMORY
F784	SYMBOL
F8EC	LOWER\$
F8F1	Routine de conversion en minuscules
F8FA	UPPER\$
F964	BIN\$
F969	HEX\$
F98F	DEC\$
F9BC	STR\$
F9D3	LEFT\$
F9D8	RIGHT\$
FA07	MID\$
FA69	LEN
FA6E	ASC
FA74	CHR\$
FA7E	INKEY\$
FA8D	STRING\$
FAAD	SPACE\$
FABE	VAL
FAE5	INSTR
FC53	FRE
FD0C	Addition +
FD21	Soustraction -
FD35	Multiplication *
FD52	Division /
FD67	Division entière \
FD79	Modulo (reste de la division).
FD87	Fonction AND (ET LOGIQUE).
FD92	Fonction OR (OU LOGIQUE).
FD9C	Fonction XOR (OU EXCLUSIF).
FDB0	ABS
FE0E	FIX



# ADRESSES PRINCIPALES DE LA ROM SUPERIEURE DU CPC6128

C  
P  
C  
6  
1  
2  
8

FE13	INT
FEB6	CINT
FEED	UNT
FF14	CREAL
FF1B	Nettoyage de l'accumulateur
FF2A	SGN
FF32	Positionnement d'un entier dans l'accumulateur
FF3E	Conversion en réel
FF45	Met le type de variable dans C
FF4B	Met le type de variable dans A
FF83	Copie l'accumulateur dans la zone pointée par DE
FF92	Teste si majuscule
FF9C	Teste si numérique
FFAB	Conversion en majuscule
FFCA	Compare A et le contenu de HL
FFD8	Compare HL et DE
FFDE	Compare HL et BC
FFE4	DE = HL - DE
FFF2	LDIR
FFF8	LDDR
FFFB	JP (HL)
FFFC	Retour au contenu de BC
FFFE	Retour au contenu de DE



## POUR LA TABLE DES VALEURS GAMME CHROMATIQUE

DO	3822	FA #	2703
DO #	3608	SOL	2551
RE	3405	SOL #	2408
RE #	3214	LA	2273
MI	3034	LA #	2145
FA	2863	SI	2025

Ces valeurs correspondent à l'octave -3 ; pour chaque octave supérieure, il suffit de diviser les valeurs par 2.



# TABLE DES CODES DE CONTROLE DU TERMINAL

Code	Action	Paramètre
0	--- --- --- ---	-
1	Imprime le caractère dont le code suit.	1
2	Enlève l'affichage curseur.	0
3	Autorise l'affichage curseur.	0
4	Positionne l'écran en mode 0, 1, 2.	1
5	Impression en mode graphique du caractère suivant.	1
6	Autorisation d'affichage vidéo.	0
7	Sonnette.	0
8	Déplace le curseur d'un caractère en arrière avec effacement (BS).	0
9	Déplace le curseur d'un caractère vers la droite.	0
10	Déplace le curseur d'une ligne vers le bas.	0
11	Déplace le curseur d'une ligne vers le haut.	0
12	Effacement de la fenêtre courante + home.	0
13	Retour chariot.	0
14	Positionne encre papier.	1
15	Positionne encre crayon.	1
16	Effacement du caractère courant.	0
17	Effacement du début gauche de la fenêtre jusqu'à la position courante du curseur.	0
18	Effacement de la position courante du curseur jusqu'au bord droit de la fenêtre.	0
19	Effacement du bord supérieur gauche de la fenêtre jusqu'à la position courante du curseur.	0
20	Effacement de la position courante du curseur jusqu'au bord inférieur droit de la fenêtre.	0
21	Inhibe la visualisation.	-
22	Positionne en mode opaque (0) ou transparent (1).	1
23	Positionne en mode écriture graphique.	1
24	Echange les encres papier et crayon.	0
25	Détermine la matrice d'un caractère.	9
26	Positionne les limites d'une fenêtre.	4
27	--- --- --- ---	-
28	Positionne les couleurs d'une encre.	3
29	Positionne les couleurs de bord.	2
30	Positionne le curseur en haut à gauche d'une fenêtre (home).	0
31	Positionnement du curseur en absolu dans une fenêtre.	2



# TABLE DES ADRESSES DES PORTS UTILISES

7FXX	VIDEO GATE ARRAY	S
BCXX	6845 (ADRESSE)	S
BDXX	6845 DONNEE	S
BEXX	6845 STATUS (ETAT)	E
BFXX	6845 DONNEE	E
DFXX	SELECTION NON EXTERNE	S
EFXX	PORT IMPRIMANTE	S
F4XX	8255 PORT A	E/S
F5XX	8255 PORT B	E/S
F6XX	8255 PORT C	E/S
	8255 PORT CONTROLE	--
FFXX	RESERVE A L'UTILISATEUR	

CPC464 ET 6128



## STRUCTURE DE LA MEMOIRE ECRAN

Taille : 16 K

Début standard : en C000, mais peut commencer en 0000, 4000 ou 8000

Quel que soit le mode, la mémoire écran peut être considérée comme 8000 mots de 16 bits qui définissent 4, 8 ou 16 points dans les modes respectifs 0, 1 et 2.

MODE 0	4 points sur 16 bits	4 bits par point	16 couleurs
MODE 1	8 points sur 16 bits	2 bits par point	4 couleurs
MODE 2	16 points sur 16 bits	1 bit par point	1 couleur

Les lignes 0, 8, 16, 24, ..., 192 sont codées dans les deux premiers K.

Les lignes 1, 9, 17, 25, ..., 193 sont codées dans les deux K suivants.

...

Les lignes 7, 15, 23, 31, ..., 199 sont codées dans les deux derniers K.

Le registre d'adresse du 6845 détermine l'adresse de départ dans un bloc de deux K (10 bits).

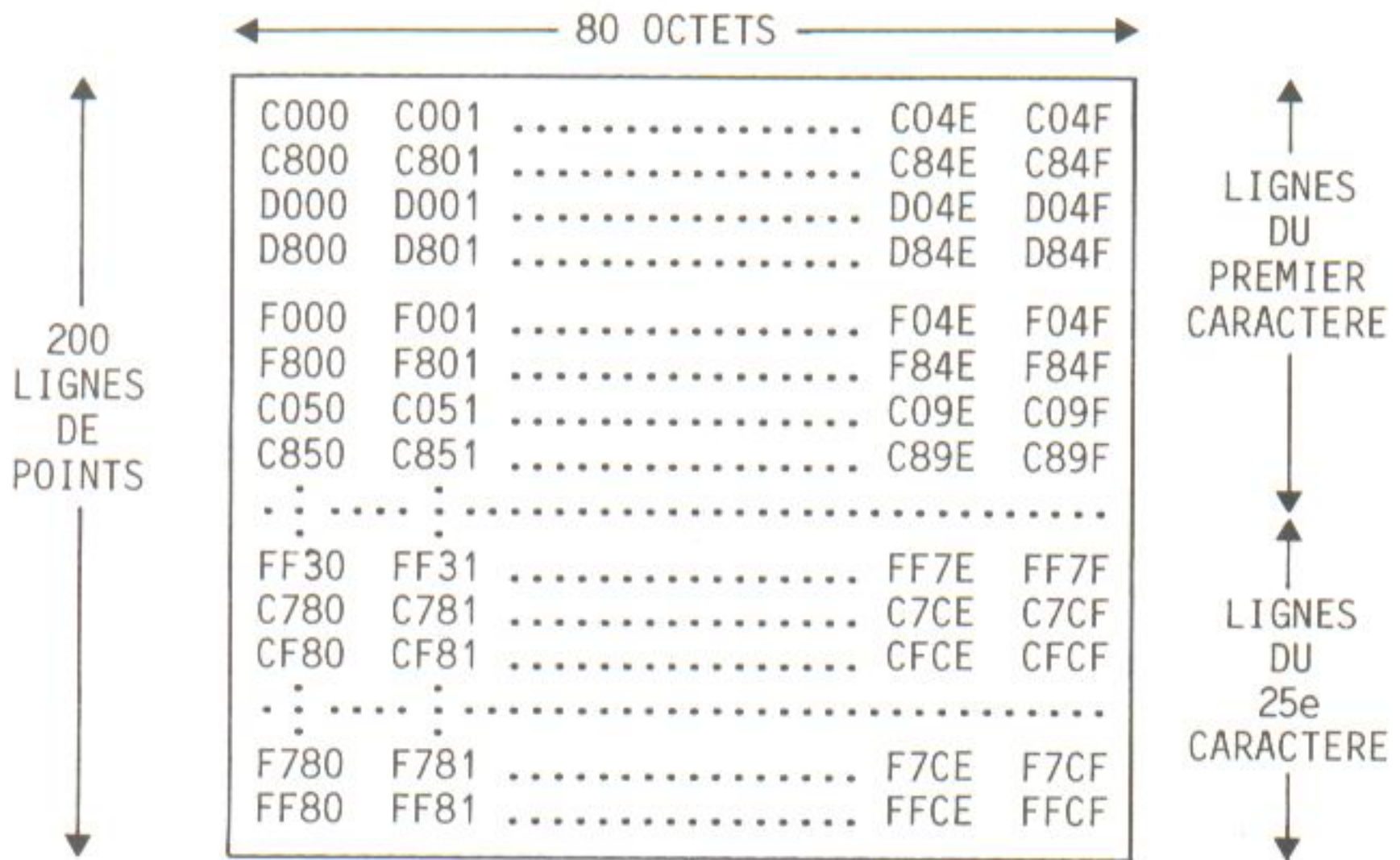
Chaque ligne utilise 80 octets consécutifs en mémoire.

Par exemple, si l'adresse de départ est C000, la ligne 0 occupe les 80 premiers octets, de C000 à C04F ; la ligne 1 occupe les 80 octets de C800 à C84F et la ligne 8 occupe les octets de C050 à C09F.

	Mode 0	Mode 1	Mode 2
Point le plus à gauche	bits 1, 5, 3, 7	bits 3, 7	bit 7
...			bit 6
...		bits 2, 6	bit 5
...			bit 4
...	bits 0, 4, 2, 6	bits 1, 5	bit 3
...			bit 2
...		bits 0, 4	bit 1
Point le plus à droite			bit 0



# STRUCTURE DE LA MEMOIRE ECRAN



C7D0...C7FF, CFD0...CFFF, ..., FFD0...FFFF ne sont pas utilisés.



# TABLE DES COULEURS

Numéro	Couleur	Valeur reg. 6845
0	Noir	20
1	Bleu	4
2	Bleu brillant	21
3	Rouge	28
4	Magenta	24
5	Mauve	29
6	Rouge brillant	12
7	Violet	5
8	Magenta brillant	13
9	Vert	22
10	Cyan (bleu)	6
11	Bleu ciel	23
12	Jaune	30
13	Blanc	0
14	Bleu pastel	31
15	Orange	14
16	Rose	7
17	Magenta pastel	15
18	Vert brillant	18
19	Vert marin	2
20	Cyan brillant	19
21	Jaune citron	26
22	Vert pastel	25
23	Cyan pastel	27
24	Jaune brillant	10
25	Jaune pastel	3
26	Blanc brillant	11

+64



# TABLE DES CODES CLAVIER (NUMEROS DES TOUCHES)

## Clavier

66	64	65	57	56	49	48	41	40	33	32	25	24	16	79
68	67	59	58	50	51	43	42	35	34	27	26	17	18	
70	69	60	61	53	52	44	45	37	36	29	28	19		
21	71	63	62	55	54	46	38	39	31	30	22	21		
47											23			

## Clavier numérique

10	11	3
20	12	4
13	14	5
15	7	6

## Flèches curseur

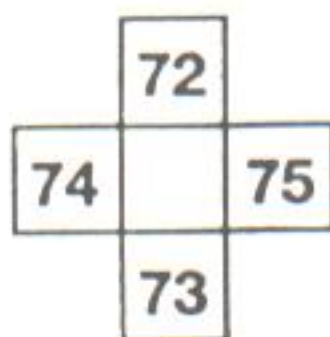
	0	
8	9	1
	2	

C  
P  
C  
4  
6  
4  
6  
6  
4  
E  
T  
6  
1  
2  
8



# TABLE DES CODES CLAVIER (NUMEROS DES TOUCHES)

Manette de jeux 0



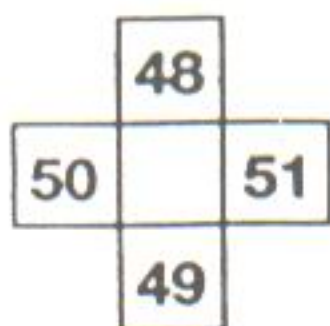
Bouton de tir 1

77

Bouton de tir 2

76

Manette de jeux 1



Bouton de tir 1

53

Bouton de tir 2

52



# INDEX

ACCUMULATEUR	Organisation interne du Z80	51-58
	Virtuel avec fonctions mathématiques	111
AFFICHAGE	Voir "ECRAN"	
ALPHANUMERIQUE	DEFSTR	15
	INKEY\$	28
	LOWER\$	30
	Codes erreurs	13-16 et 45-46
	COPYCHR\$	169
	MID\$ gauche	170
ASCII	ASC, CHR\$	27
	Codes	27-33
	Stockage d'une ligne BASIC	49
	Dump ASCII mémoire	152
BASIC STANDARD	Caractéristiques générales	13
	Format de stockage ligne BASIC	48
	Logiciel interne	79
	Adresses d'exécution	133-134
	Simulation instruction CIRCLE	154
	Gestionnaire cassette	79-95-98
	Format fichiers cassette	136-138
	Démarrage-arrêt moteur	153
CHRONOMETRE	AFTER	14
	EVERY	17
	REMAIN	30
CLAVIER	Configuration, matériel	12
	KEY	18
	KEY DEF	18



	INKEY\$	28
	Registre	14-142
	Programmation AY3-8912	142
	Table des codes clavier	211-212
CODE	Associé au mot-clé	33-34
	Instructions Z80	59-73
	Tableau désassemblage	77
	De contrôle (gestion du mode texte)	79-84
	Table des codes de contrôle du terminal	206
CONNECTEUR	Pour manettes de jeux	162
	Pour sortie vidéo	163
	Sortie expansion	164
	Sortie imprimante	165
CPU	Schéma général, matériel	11-12
	Organisation interne du Z80	51
CRTC 6845	Schéma général, matériel	11-12
	Généralités, registres	146
	Programmation	147
DESASSEMBLAGE	Tableaux de désassemblage	74-77
ECRAN	CLS, DRAW, ERASE, INK, MODE, MOVE...	15-20
	WINDOW, POS, TEST...	26-30-32
	SPC, CURSOR, FILL, FRAME...	169
	GRAPHICS, MASK...	170
	Jeux de caractères	38-44
	Vecteurs d'indirection	105
	CRTC 6845	146-147
	VIDEO GATE ARRAY	148-149
	Modification couleur de fond	154
ENVELOPPE	Schéma général, matériel	11
	SOUND	23
	Gestionnaire sonore	80-98-99
	PSG AY3-8912	139-142
ENVELOPPE DE VOLUME	ENV	16
	SOUND	23
	Contrôle, amplitude, timbre	135
	PSG AY3-8912	139
ENVELOPPE DE TON	ENT	16
	SOUND	23
	Contrôle, amplitude, timbre	135
	PSG AY3-8912	139
	Table des valeurs chromatiques	205



FILE	INPUT, MERGE	18-19
	Codes erreurs	21-24-25-27-46
	Format fichiers cassette	136-137
FONCTIONS BASIC MATHEMATIQUES	Définition	27-32
	Vecteurs appel routines math.	111-114
GATE ARRAY	Schéma général matériel	111-112
	VIDEO GATE ARRAY	148-149
GENERATEUR	Situation générateur de caractères	79
	Situation des caractères	120
	Sonore : voir ENVELOPPE	
GESTIONNAIRES	Introduction	79-80
	Développement	81-104
GRAPHIQUES	Codes ASCII et graphiques	36
	Gestionnaire graphique	79-88-91
	Table des couleurs	210
HORLOGE	Organisation interne du Z80	51
IMPRESSION	PRINT USING	22
	POS	30
	Dump hexa mémoire-imprimante	151
	Connecteur imprimante	165
	DEC\$	169
INTERRUPTION	ON BREAK CONT	170
JEU DE CARACTERES	Symboles	38-44
JOYSTICK	Configuration, matériel	12
	Registre	14-142
	Programmation AY3-8912	142
	Connecteurs	162-165
LANGAGE MACHINE	Instructions Z80	53-77
	Installation d'une routine en langage machine dans une REM	155
LIGNE	Caractéristiques du BASIC	13
	Code erreur	8-45
	Format de stockage d'une ligne BASIC	48
MANETTE DE JEUX	Voir "JOYSTICK"	
MATRICE (tableau)	DIM	16
	Pour caractères	38



	Code erreur	10-45
	Table des couleurs	210
<b>MEMOIRE ECRAN</b>	Situation	12
	Gestionnaire d'écran	79-91-95
	Structure	208-209
<b>MOTS-CLES DU BASIC</b>	Codes décim. et hexa.	33-34
	Adresses de la ROM supérieure	126-130
	Adresses d'exécution	133-134
<b>OVERFLOW</b>	Code erreur 6	45
<b>POINTS D'ENTREE</b>	Table	81-104
	Vecteurs d'indirection	105-106
	ROM	120-130
	Adresses réelles ROM	131-132
	Adresses exécution mots-clés	133-134
	Adresses exécution mots-clés (664)	192-193
<b>PORT</b>	Schéma général, matériel	11-12
	OUT	21
	WAIT	25
	PSG AY3-8912	139-142
	PPI 8255	143-145
	Table des adresses des ports	207
<b>PPI 8255</b>	Schéma général, matériel	11-12
	Ports, programmation	143-145
	Brochage	159
<b>PSG AY3-8912</b>	Schéma général, matériel	11-12
	SOUND	23
	Structure interne, programmation	139-142
	Brochage	157
<b>QUEUE SONORE</b>	SQ	31
	Gestionnaire sonore	80-98-99
	Tableau queue sonore	135
	Voir PSG AY3-8912, gestionnaire sonore...	
<b>RAM</b>	Schéma général, matériel	11-12
	Logiciel interne	79-80
	Gestionnaire sonore	98-99
<b>REGISTRES</b>	Du Z80	52
	Du PSG AY3-8912	139-142
	De contrôle	144-145
	Du CRT 6845	146-147



ROM : BASIC	Schéma général, matériel	11-12
	Vecteurs	107-110
	Adresses réelles ROM	131-132
	Adresses réelles ROM (664)	190-191
	ROM supplémentaire	136
	Commutation ROM INF/ROM SUP	111-148-149
	DUMP hexa ROM sur imprimante	151
ROM INFÉRIEURE	Logiciel interne	79-80
	Gestionnaire noyau	100-103
	Interfaçage avec le matériel	103-104
	Vecteurs noyau et restart	107
	Vecteurs appel routines mathématiques	111
	Adresses principales (464)	120-125
	Adresses principales (664)	180-184
	Adresses principales (6128)	196-198
ROM SUPÉRIEURE	Logiciel interne	79-80
	Vecteur noyau et restart	107
	Vecteur en bas de mémoire	108
	Adresses principales (464)	126-130
	Adresses principales (664)	185-189
	Adresses principales (6128)	199-203
ROUTINE	ON SQ GOSUB, ON BREAK GOSUB, RETURN	20-22
	Table des points d'entrée des routines système	81
	Utilisation des vecteurs d'indirection	105-106
	Vecteurs noyau et restart	107-110
	Écriture dans PSG par routine	142-188
	Remplacement d'INKEY\$	154
	En langage machine dans une REM	155
	MATHEMATIQUES dans ROM inférieure	79-80
	MATHEMATIQUES vecteurs d'appel	111-114
	SYSTEME	81-104
TAMPON D'ENTRÉE CLEAR INPUT		169
TOKEN	Définition	48
	Stockage des variables + exemple	49-50
VARIABLES	Caractéristiques BASIC	13
	CLEAR, DEFINT, DEFREAL, DEFSTR, DIM	14-15-16
	LINE INPUT	19
	ERR, ERL	27
	DERR	169
	Stockage des variables	49
	Variables système (situation)	79



	Routines math-var réelles et entières	111-114
	Principales variables système	
	+ adresses	115-119
<b>VECTEUR</b>	Représentation du logiciel système	80
	D'appel des routines mathématiques (464)	111
	D'appel des routines mathématiques (664)	171-179
	Adresses réelles de branchement	131-132
	D'indirection	105-106
	Noyau, en haut et bas de mémoire	107-110
	D'encre	136
<b>Z80</b>	Schéma général, matériel	11-12
	Organisation interne	51-52
	Jeu d'instructions	53-58
	Instructions Z80	59-73
	Brochage	160



# CONSEILS DE LECTURE

Pour approfondir vos connaissances en Basic et mieux connaître le système des CPC 464, 664 et 6128, P.S.I. vous propose une palette d'ouvrages utiles.

## Pour maîtriser le Basic Amstrad

### ■ BASIC AMSTRAD 1. METHODES PRATIQUES

par Jacques Boisgontier et Bruno Césard  
(Editions du P.S.I.)

Pour ceux qui ont déjà pratiqué un Basic, voici un ouvrage de perfectionnement au Basic Amstrad. Un chapitre sur le CP/M 2.2 et le CP/M Plus donne les principales commandes systèmes.

### ■ BASIC AMSTRAD 2. PROGRAMMES ET FICHIERS

par Jacques Boisgontier  
(Editions du P.S.I.)

Pour pratiquer le Basic Amstrad, cet ouvrage donne de nombreux programmes de gestion, d'éducation et de jeu où le rôle des fichiers est expliqué et largement commenté.

### ■ BASIC PLUS - 80 ROUTINES SUR AMSTRAD

par Michel Martin  
(Editions du P.S.I.)

Pour pousser votre Amstrad au maximum de ses capacités : 80 routines de simulation d'instructions qui n'existent pas en Basic Amstrad.



## Pour mieux connaître le système des CPC

### ■ CLEFS POUR AMSTRAD 2. SYSTEME DISQUE

par Daniel Martin et Philippe Jadoul  
(Editions du P.S.I.)

Ce deuxième tome consacré au système disque présente les points d'entrée des routines disque, les blocs de contrôle, la programmation et les brochages des circuits spécialisés... La deuxième partie du livre est aussi destinée aux possesseurs d'Amstrad 8256.

### ■ CP/M PLUS SUR AMSTRAD

par Yvon Dargery  
(Editions du P.S.I.)

Toutes les commandes CP/M et CP/M Plus pour maîtriser le système des 6128 et 8256 : un ouvrage de référence illustré par de nombreux programmes.

### ■ LE LIVRE DE L'AMSTRAD. TOME 1

par Daniel Martin et Philippe Jadoul  
(BCM - Diffusé par P.S.I.)

Ce livre, destiné aux programmeurs des CPC 464, 664, et 6128 donne une étude complète de tous les circuits internes, et analyse la structure interne du Basic. Vous y trouverez, en outre, une étude complète des RSX, et des programmes de scrolling, de traçage de rectangles, de coloriage de surface et de manipulation vectorielle.

## Pour programmer vos applications en assembleur Z80

### ■ GRAPHISME EN ASSEMBLEUR SUR AMSTRAD CPC

par Francis Piérot  
(Editions du P.S.I.)

Cet ouvrage met en pratique les notions d'assembleur acquises par le lecteur grâce à de nombreux programmes commentés. Du simple dessin d'un cercle à l'animation d'une soucoupe volante dans un décor en passant par la mise en mouvement d'une corne d'abondance, vous apprendrez à maîtriser l'assembleur et à créer de très belles pages-écran.

### ■ CREATION ET ANIMATIONS GRAPHIQUES SUR AMSTRAD CPC

par Gilles Fouchard et Jean-Yves Corre  
(Editions du P.S.I.)

Dessiner avec la souris ou le joystick et apprendre à faire des scrolling, à fabriquer une gomme, à inverser une image ou à l'éclater en une myriade de points, tel est l'objectif de ce livre écrit en Assembleur Amstrad.



# mémento

Ne tenez plus votre livre d'une main tout en pianotant de l'autre sur le clavier de votre ordinateur, "Clefs pour Amstrad" est un memento 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, points d'entrée des routines système, blocs de contrôle, structure interne, programmation, connecteurs et brochage des principaux circuits utilisés.

"Clefs pour Amstrad" est aussi un recueil d'astuces : protéger le programme, produire des bruits originaux, faire un scanning du clavier ou installer une routine en langage machine dans une remarque, sont autant de conseils pour découvrir toute l'originalité de votre CPC 464, 664 ou 6128. N'oubliez pas vos clefs !

## CLEFS POUR AMSTRAD

CPC 464 - 664 et 6128 1. Système de base



9 782865 952472

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

I.S.B.N. : 2.86595.247.9