9/8.12

Protection écran *(Screen saver)*

Votre frénétique activité de programmeur est parfois interrompue par une sonnerie de téléphone inopportune, ou la sonnette de la porte d'entrée.

Ce dérangement peut parfois se prolonger plus longtemps que vous ne l'auriez désiré, et vous faire oublier votre ordinateur.

Hors, laisser une page écran fixe sur la visu n'est pas sans conséquence dommageable à long terme sur la durée de vie de votre écran vidéo.

Ce problème peut apparaître aussi chez nos lecteurs passionnés de calculs mathématiques ou astronomiques, longs, qui laissent l'écran du CPC allumé sur une page fixe durant l'attente du résultat.

La solution idéale serait de couper automatiquement l'alimentation du moniteur vidéo au bout de quelques minutes sans frappe sur le clavier. Cette solution s'avère compliquée et probablement coûteuse, sans compter les problèmes de fiabilité, de matériel électronique, d'approvisionnement des composants, et le risque d'électrocution dû à la manipulation près des organes électroniques de la THT (Très Haute Tension) appliquée sur le tube vidéo. La perte de la garantie est aussi à craindre si votre matériel est récent.

La solution la moins dangereuse et la plus économique est une solution logicielle telle que celle que nous vous proposons.

A défaut d'éteindre complètement l'écran, nous allons inhiber l'affichage des caractères sur celui-ci. Cette solution est acceptable, dans la mesure où ce sont les pixels allumés qui risquent de brûler le tube cathodique.

Nous aurions pu provoquer un CLS de l'écran, mais il faut que celui-ci restitue intégralement son contenu dès la frappe d'une touche, lors du retour de l'utilisateur.

Nos problèmes sont donc les suivants :

— reconnaître la frappe ou non d'une touche sur le clavier ;

— inhiber l'affichage sur la visu ;

— effectuer ces différentes opérations sans perturber un programme en cours ou les manipulations éventuelles de l'utilisateur (frappe d'un listing, par exemple),

— restituer l'affichage lors de la frappe d'une touche.

La résolution de ces problèmes vous fera, tout au long de ce chapitre, découvrir de nouvelles possibilités, qui vous pemettront de sophistiquer vos propres programmes.

I. Reconnaître une touche frappée

Afin de reconnaître si une touche a été frappée par l'utilisateur, nous pouvons, dans un programme Basic, utiliser l'instruction :

A$ = INKEY$

mais, lors de l'utilisation en mode commande (ou direct) de l'AMSTRAD, cette instruction n'a aucun effet.

Penchons-nous donc sur le fonctionnement du CPC en mode commande.

ADRESSES DE GESTIONS DES TOUCHES FRAPPÉES

Il faut savoir que toutes les 3,3 millisecondes (tous les 1/300e de seconde exactement), le CPC est interrompu afin d'effectuer des routines essentielles à son fonctionnement (ré-actualisation de la variable TIME, gestion de la table d'événements, actualisation du compteur de gestion clavier, ...) et, entre autres, la scrutation du clavier (grâce au compteur précédemment nommé) tous les 1/50e de seconde (toutes les 20 ms).

La scrutation du clavier remet à jour une zone de mémoire située, pour les CPC 664 et CPC 6128, entre les adresses &B63F et &B648 ainsi qu'entre les adresses &B649 et &B652.

Ces deux zones d'adresses sont en fait complémentaires au sens binaire du terme : c'est-à-dire que le contenu de l'adresse &B63F est le complément binaire du contenu de l'adresse &B649, le contenu de l'adresse &B640 le complément binaire du contenu de &B64A, ... etc.

Initialement, quand aucune touche n'est enfoncée, le contenu des adresses &B63F à &B648 est fixé à &FF (255 décimal), donc celui des adresses &B649 à &B652 fixé à 00.

Nous vous proposons ci-après un programme permettant d'explorer plus en détail le contenu de ces adresses.

PROGRAMME EXPÉRIMENTAL

```
10 MODE 2
20 PRINT "ADRESSE", "HEXA", "DECIMAL", "BINAIRE"
30 FOR ADRESSE = &B63F TO &B648
40 PRINT HEX$(ADRESSE);SPACE$(3);"";"";HEX$(PEEK(ADRESS-
E));SPACE$(3),PEEK(ADRESSE);SPACE$(4),USING"# # # # # # #
# #";VAL(BIN $(PEEK(ADRESSE)))
50 NEXT ADRESSE
```

```
60 PRINT STRING$(40,"-"):REM separation
70 PRINT "ADRESSE", "HEXA", "DECIMAL", "BINAIRE"
80 FOR ADRESSE = &B649 TO &B652
90 PRINT HEX$(ADRESSE);SPACE$(3);";";HEX$(PEEK(ADRESS-
E));SPACE$(3),PEEK(ADRESSE);SPACE$(4),USING"# # # # # #
# #";VAL(BIN $(PEEK(ADRESSE)))
100 NEXT ADRESSE
110 PRINT CHR$(30):REM retour en haut de l'écran
120 GOTO 20
130 END
```

Ce petit programme permet de lire et d'afficher en permanence le contenu des adresses précédemment citées.

Vous pourrez ainsi appuyer sur une touche de votre choix et visualiser la modification du contenu d'une adresse spécifique. L'appui sur une touche devra être suffisamment long pour permettre au Basic d'afficher les contenus.

Vous trouverez, par exemple, les valeurs suivantes pour les touches :

— **DEL** : &60 à l'adresse &B652, &7F à l'adresse &B648
— **f6** : &10 à l'adresse &B649, &EF à l'adresse &B63F
— **ESPACE** : &80 à l'adresse &B64E, &7F à l'adresse &B644.

II. Inhibition de l'affichage écran

LE CONTROLEUR VIDÉO

Pour modifier l'affichage sur l'écran, il nous faut connaître comment sont organisées les entrailles du CPC, et notamment nous renseigner sur la gestion de l'écran.

Le contrôle de l'affichage sur l'Amstrad a été dédié au VGA (*Vidéo Gate Array*, composant spécifique conçu par AMSTRAD), et surtout au composant 6845 conçu par MOTOROLA initialement pour la famille 68XX, mais adaptable facilement à d'autres microprocesseurs tel le Z80. Nous vous conseillons de vous reporter avec attention à la description qui en est faite à la Partie 2, chapitre 3.2.

Ce composant contient 19 registres (le registre AR et les 18 registres R0 à R17), auxquels on accède en écriture par deux adresses de ports (au lieu de 19). Cette astuce est possible grâce au registre AR, placé à une adresse de port, qui sert de pointeur de registre, dans lequel on place d'abord le numéro du registre de gestion vidéo à atteindre, l'autre adresse du port étant celle du registre pointé.

Les constructeurs de l'AMSTRAD ont placé le composant 6845 aux adresses &BCXX (pour le registre AR) et &BDXX (pour écrire dans le registre dont le numéro est pointé par le contenu de AR). Chacun des X des adresses peut être n'importe quelle valeur hexadécimale comprise entre &0 et &F, ceci est dû à un décodage partiel des composants sur les CPCs.

Pour simplifier notre étude, nous prendrons l'adresse &BC00 pour le registre AR et l'adresse &BD00 pour le registre pointé par le contenu de AR.

L'adressage d'un port s'effectue différemment de l'adressage d'une case mémoire. Par exemple en Basic, on écrit dans la case mémoire en RAM &adresse grâce à l'instruction :

POKE &adresse,&valeur

et on lit par :

A = PEEK(&adresse)

Remarque :

Le caractère **&** indique que la valeur est en hexadécimal, mais elle peut aussi être décimale, on enlève cette fois-ci le caractère **&**.

Par contre pour lire ou écrire sur un port, on utilisera les instructions **INP** (pour lire) et **OUT** (pour écrire) :

OUT &adresse,&valeur et A = INP (&adresse)

Nous vous proposons d'expérimenter le programme ci-dessous qui accède au registre de largeur de synchronisation horizontale :

```
10  MODE 1
20  PRINT "SI VOUS APPUYEZ SUR UNE TOUCHE"
30  PRINT "JE DETRUIS VOTRE AMSTRAD"
60  CLEAR INPUT
70  A$ = INKEYS
80  IF A$ = "" THEN 70
90  OUT &BC00,3:REM accès au registre
100 OUT &BD00,1:REM modification de la synchronisation
110 END
```

Suivez le conseil, et, pour tout remettre dans l'ordre, frappez en aveugle la commande :

OUT &BD00,&8E

Vous pouvez ainsi essayer, en vous aidant des renseignements fournis Partie 2, chapitre 3.2, différentes combinaisons sur les registres du contrôleur Vidéo. Rassurez-vous sur les effets produits, ils ne peuvent en aucun cas endommager votre AMSTRAD. Si vous vous retrouvez bloqué suite à certaines commandes, un RESET par l'appui simultané sur <SHIFT> <CONTROL> <ESC> vous redonnera la configuration initiale.

L'INITIALISATION DU CONTROLEUR VIDÉO

Lors de l'initialisation, l'AMSTRAD effectue un saut à l'adresse 0000. A cette adresse est effectuée une commutation sur la ROM du système d'exploitation et un saut à l'adresse &0591H. A partir de là, sont effectuées les différentes initialisations des composants de l'unité centrale, et notamment à partir de l'adresse &05B1H l'initialisation du contrôleur Vidéo.

Nous vous donnons ci-dessous un désassemblage commenté de cette initialisation :

```
05B1    21E505   LD HL,05E5; pointeur fin table CRTC
05B4    010FBC   LD BC,BC0F; adresse registre (BCXX)
05B7    ED49     OUT(C),C  ; adresse registre dans AR
05B9    2B       DEC HL    ; pointe valeur registre suivant
05BA    7E       LD A,(HL) ; charge valeur
05BB    04       INC B     ; position BDXX
05BC    ED79     OUT(C),A  ; charge registre
05BE    05       DEC B     ; position BCXX (reg. AR)
05BF    0D       DEC C     ; position registre suivant
05C0    F2B705   JP P,05B7 ; saut si pas dernier registre traité
05C3    1820     JR 05E5   ; saut pour suite init.
...     ...      ...         ...
;**** TABLE CRTC ****
05D5    3F       ; fréquence de synchro horizontale → RO
05D6    28       ; nombre de caractères par ligne → R1
05D7    2E       ; délai de synchro horizontale →R2
05D8    8E       ; largeur de synchronisation horizontale → R3
05D9    1F       ; nombre de lignes par écran → R4
05DA    06       ; ajustage du nombre de lignes → R5
05DB    19       ; nombre de lignes affichées → R6
05DC    1B       ; synchronisation verticale →R7
05DD    00       ; mode → R8
05DE    07       ; nombre de lignes par caractère → R9
05DF    00       ; début de curseur → R10
05E0    00       ; fin de curseur → R11
05E1    30       ; poids fort adresse RAM écran
05E2    30       ; poids faible adresse RAM écran
05E3    C0       ; poids fort adresse position curseur
05E4    00       ; poids faible adresse position curseur
```

INHIBITION DE LA VISUALISATION DES CARACTÈRES

Il nous faut maintenant déterminer comment inhiber la visualisation de l'affichage sur la visu sans modifier le contenu de la mémoire écran.

Le registre du CRTC 6845 qui nous intéresse est le registre R1 contrôlant le nombre de caractères affichés par ligne. Celui-ci est, à la mise sous tension, chargé avec la valeur 40 (&28 hexadécimal), et ce quel que soit le mode de travail choisi pour l'écran (20, 40 ou 80 colonnes).

Si nous nous amusons sous Basic à modifier le nombre contenu par ce registre, nous modifierons ainsi le nombre de caractères visualisables, la commande sera :

OUT &BC00,1
OUT &BD00, **nb. caractères** (inférieur à 40)

Le petit programme suivant vous permettra d'effectuer une petite animation agréable, qui peut être utilisée lors de modification d'une page d'un menu, par exemple :

```
10 REM AFFICHAGE PAGE ECRAN 1
20 REM ATTENTE CHOIX SUR MENU
30 GOSUB 60000:REM ANIMATION NON VISUALISATION PAGE
40 REM TRAITEMENT DU CHOIX ET AFFICHAGE NOUVELLE PAGE
50 GOSUB 60060:REM ANIMATION VISUALISATION PAGE
60 REM SUITE...
. . . . . .
60000 REM ANIMATION NON VISUALISATION
60010 OUT &BC00,1
60020 FOR J = 40 TO 0 STEP -1
60030 OUT &BD00,J
60040 NEXT J
60050 RETURN
60060 REM ANIMATION VISUALISATION
60070 OUT &BC00,1
60080 FOR J = 0 TO 40
60090 OUT %BD00,J
60100 NEXT J
60110 RETURN
```

Vous pouvez visualiser cette animation en mode direct par les commandes :

```
GOSUB 60000 : GOSUB 60060 <RETURN>
```

On remarquera, lors de cette animation, que les caractères supplémentaires de la ligne originale sont reportés sur la ligne suivante ; de plus en mode 0, c'est un demi-caractère qui est reporté, tandis qu'en mode 1, deux caractères sont reportés ; la ligne suivante commençant avec le nombre de caractères reportés.

Pour résoudre notre problème, il nous suffira d'inhiber l'écran à l'aide de l'instruction :

```
OUT &BC00,1
OUT &BD00,0
```

et de le valider à nouveau, lors de la frappe sur une touche par :

```
OUT &BC00,1
OUT &BD00,40
```

III. Interruption du fonctionnement pour traitement

Dans ce paragraphe, nous allons étudier la façon de détecter l'appui sur une touche afin de protéger notre visu.

Cette gestion ne doit tout d'abord perturber aucun programme Basic en cours, nous pourrions utiliser pour cela la gestion sous interruption logicielle telle que décrite assez précisément dans le chapitre concernant l'affichage d'un CAPS LOCK interactif (voir Partie 9, chapitre 8. 11).

INTERRUPTION LOGICIELLE

Pour ceux qui désireraient essayer ce traitement, nous vous donnons ci-après l'algorithme permettant d'écrire le programme à insérer. La lecture du chapitre précédemment cité vous y aidera aussi.

- DEBUT
  - Initialiser une variable de comptage
  - Installer le chronomètre d'interruption logicielle
- FIN
- DEBUT
    - CO
      routine de traitement à saisir en fin de programme
    - FINCO
  - Initialiser une variable témoin à zéro
  - Initialiser la variable début de tableau « touche frappée » à &B63F
  - REPETER
    - Lire le contenu de la variable « touche frappée »
      - CO
        par l'instruction **PEEK**
      - FINCO
    - SI le contenu est différent de &FF (=255)
      - ALORS
      - Placer 1 dans la variable témoin
    - FINSI
    - Incrémenter la variable touche frappée
  - JUSQU'A ce que le contenu de la variable touche frappée soit supérieur à la dernière adresse du tableau touche frappée
  - SI le contenu de la variable témoin est différent de 1
    - ALORS
    - Incrémenter le contenu de la variable de comptage
    - SI le contenu de la variable de comptage est supérieur à une valeur déterminée **Y**
      - CO
        Y sera calculée par la formule :
        Temps d'attente = Y × Délai × 50 ms
      - FINCO
    - ALORS
    - Inhiber l'affichage
    - FINSI
  - SINON
    - Réinitialiser la variable de comptage
    - Autoriser l'affichage
  - FINSI
- FIN

Ce type d'interruption ne fonctionne cependant que lors de l'exécution d'un programme Basic, ce qui n'est pas des plus intéressants, surtout lorsque c'est durant la frappe de votre listing que vous êtes interrompu !

Nous devons donc penser à un autre type d'interruption, qui est exécutée quelle que soit la façon dont vous faites travailler votre AMSTRAD (en mode direct ou en mode programme Basic).

INTERRUPTION MATÉRIELLE

Nous allons donc utiliser une interruption qui est générée toutes les 3.3 millisecondes par le composant GATE ARRAY (on se reportera au paragraphe concernant les interruptions logicielles dans le chapitre traitant de la touche CAPS LOCK, Partie 9, chapitre 8.11).

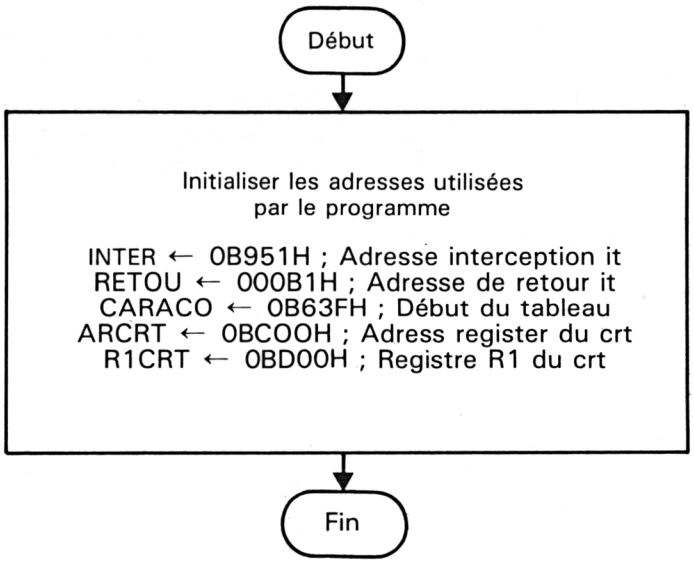
Rappelons que cette interruption est placée sur la broche $\overline{IRQ}$ du microprocesseur Z80, et qu'elle l'oblige à effectuer une instruction dénommée **RST 8,** qui effectue un branchement à l'adresse &B941 pour les CPC 664 et CPC 6128 (à l'adresse &B939 pour le CPC 464).

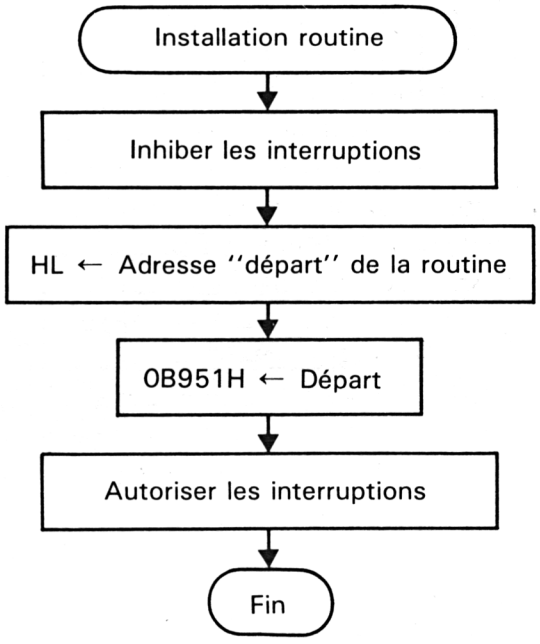
Nous pouvons détourner cette routine en plaçant une adresse différente de &00B1 aux adresses &B951 et &B952 (&B949 et &B94A pour le CPC 464).

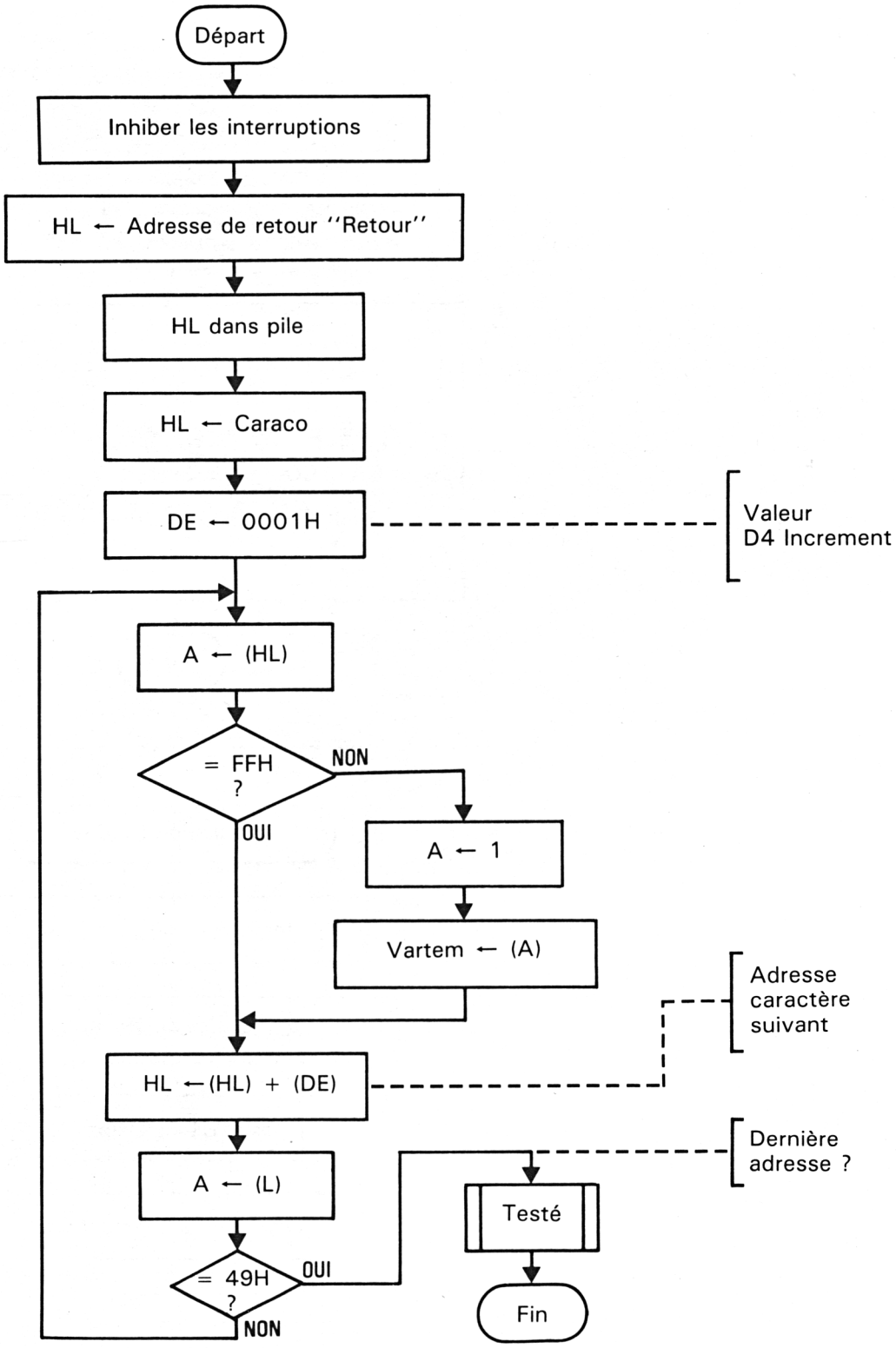
Suite à ce détournement, il nous faudra bien sûr retourner à l'adresse &00B1, faute de quoi l'AMSTRAD n'y retrouverait plus ses petits, au mieux effectuerait un splendide RESET.

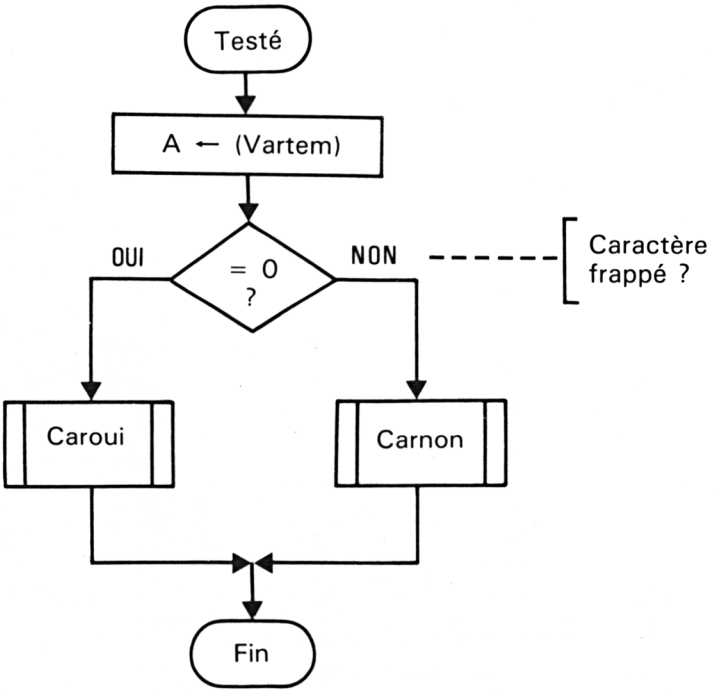
L'ORDINOGRAMME

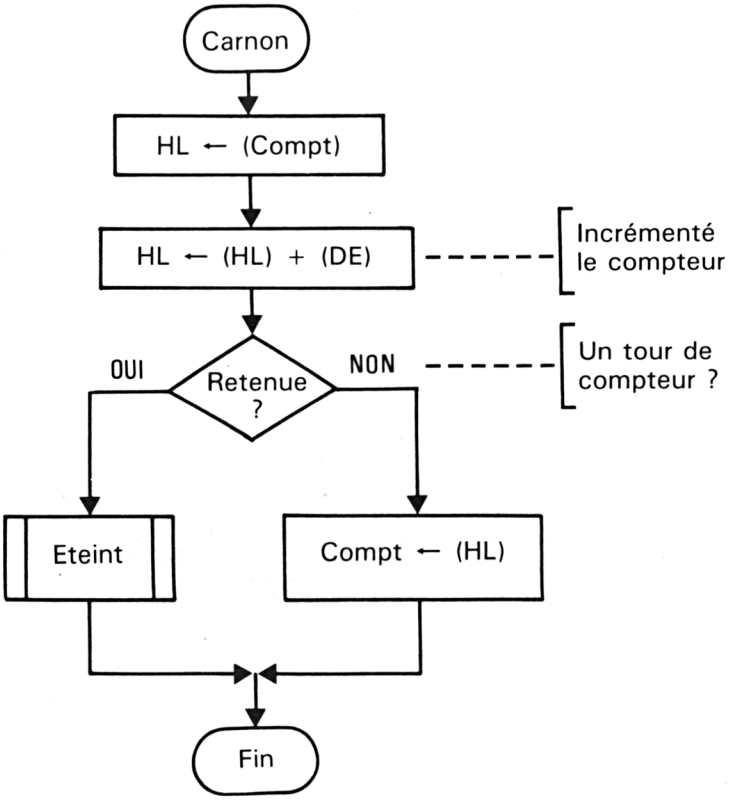
En tenant compte de ces précisions, nous pouvons adapter l'algorithme précédent au langage machine pour proposer l'ordinogramme suivant :

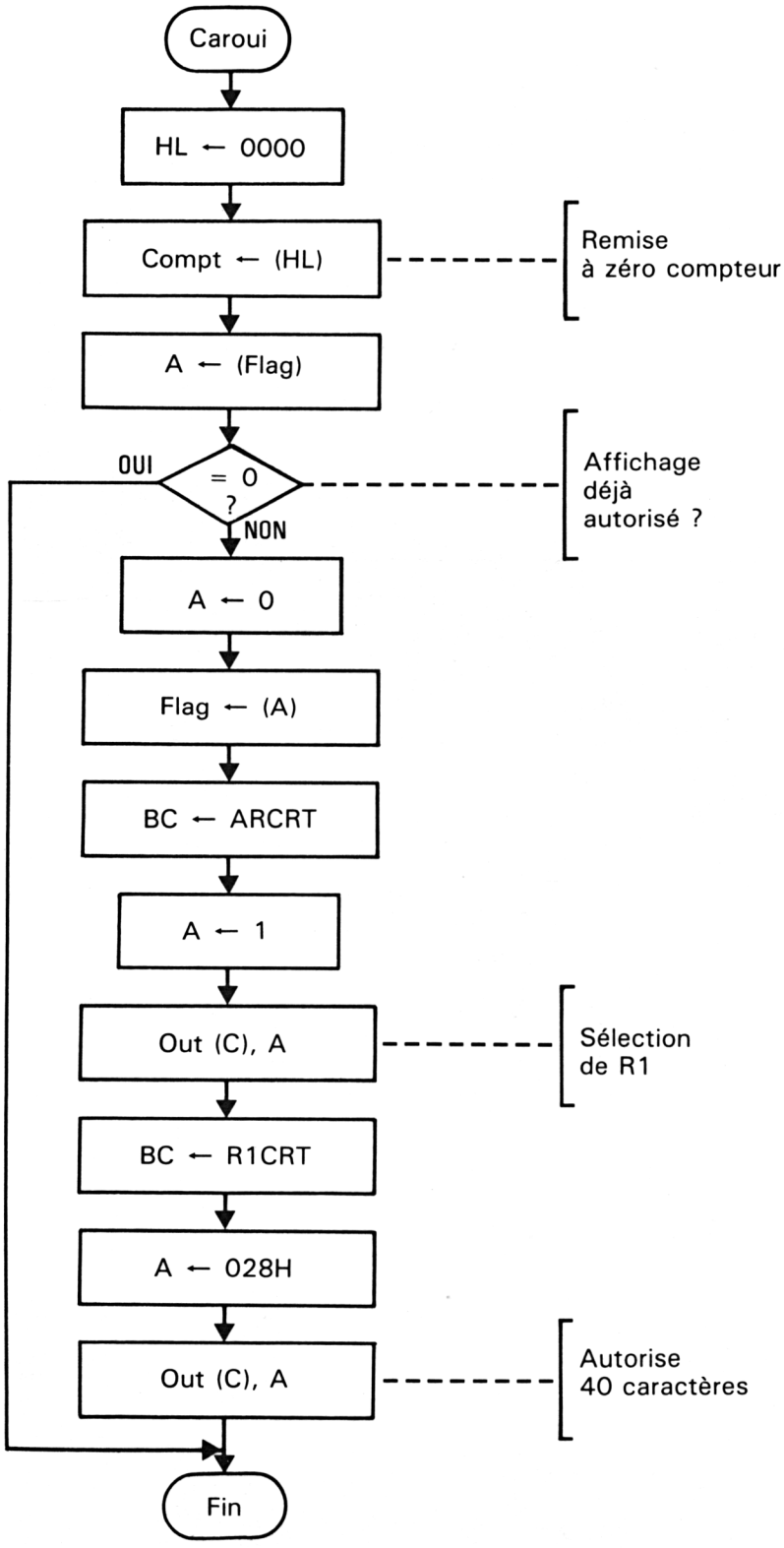


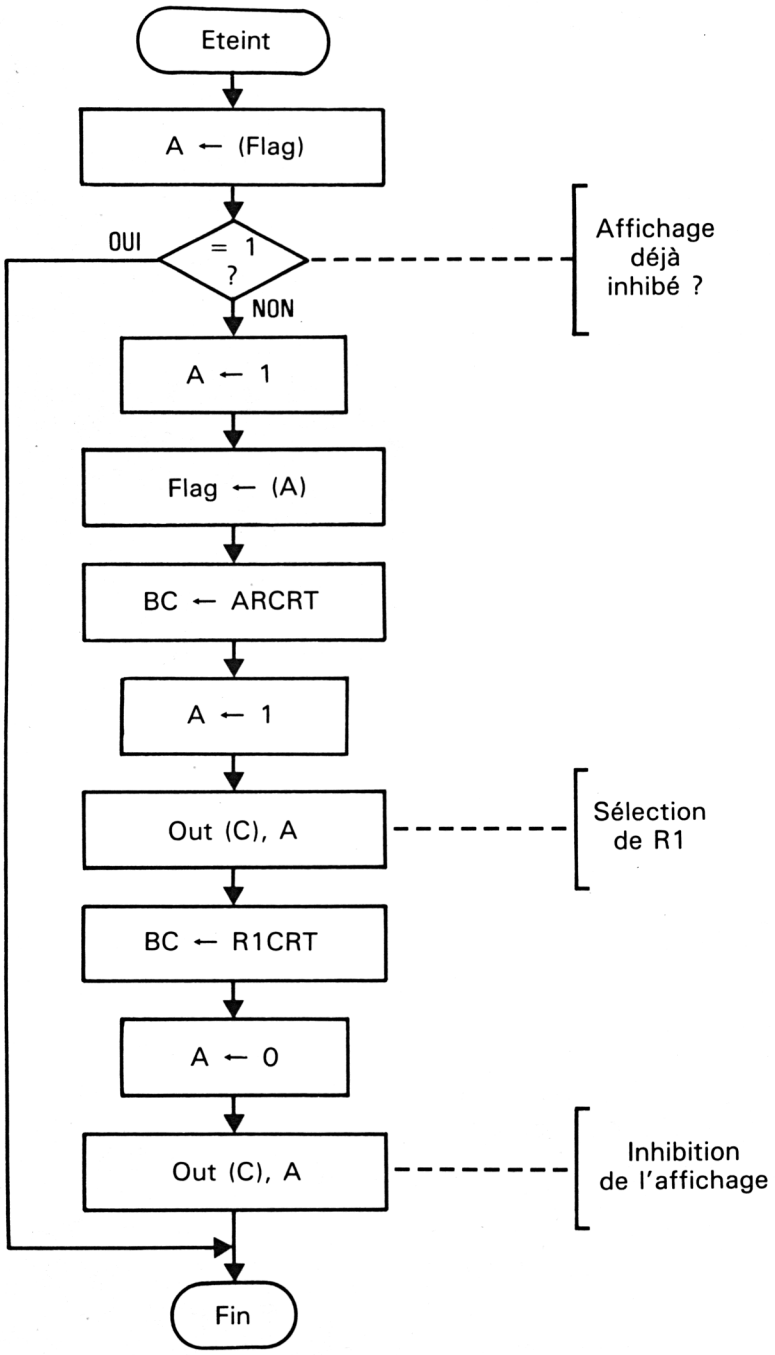



Départ
Inhiber les interruptions
HL ← Adresse de retour ''Retour''
HL dans pile
HL ← Caraco
DE ← 0001H
Valeur
D4 Increment
A ← (HL)
= FFH
?
NON
OUI
A ← 1
Vartem ← (A)
Adresse
caractère
suivant
HL ←(HL) + (DE)
Dernière
adresse ?
A ← (L)
Testé
= 49H
?
OUI
NON
Fin


Testé
A ← (Vartem)
OUI
= 0
?
NON
Caractère frappé ?
Caroui
Carnon
Fin


Carnon
HL ← (Compt)
HL ← (HL) + (DE)
Incrémenté le compteur
OUI
Retenue
?
NON
Un tour de compteur ?
Eteint
Compt ← (HL)
Fin


Caroui
HL ← 0000
Compt ← (HL)
Remise à zéro compteur
A ← (Flag)
OUI
= 0 ?
NON
Affichage déjà autorisé ?
A ← 0
Flag ← (A)
BC ← ARCRT
A ← 1
Out (C), A
Sélection de R1
BC ← R1CRT
A ← 028H
Out (C), A
Autorise 40 caractères
Fin


Eteint
A ← (Flag)
OUI
= 1 ?
NON
Affichage déjà inhibé ?
A ← 1
Flag ← (A)
BC ← ARCRT
A ← 1
Out (C), A
Sélection de R1
BC ← R1CRT
A ← 0
Out (C), A
Inhibition de l'affichage
Fin

LE PROGRAMME ASSEMBLEUR

Grace à l'ordinogramme précédent, nous allons déduire le programme Assembleur Z80 suivant.

Ce programme devra peut-être subit quelques modifications dans la notation des valeurs hexadécimales (certains Assembleurs utilisent la notation # en tant que préfixe, len ôtre utilise le **O** (zéro) en préfixe et le **H** en suffixe) — dans tous les cas, reportez-vous à la notice fournie avec celui-ci.

```
 1                ;*** PROGRAMME ASSEMBLEUR ***
 2                ;*** DE PROTECTION ECRAN  ***
 3                ;***   PAR INHIBITION DE  ***
 4                ;***    DE  L'AFFICHAGE   ***
 5                ;*****************************
 6                ;
 7                ;
 8                ;DEFINITION DES ADRESSES UTILES
 9                ;*****************************
10                ;
11                ;ADRESSE DETOURNEMENT INTERRUPTION
12                INTER:      EQU  0B951H
13                ;
14                ;ADRESSE DE RETOUR D'INTERRUPTION
15                RETOU:      EQU  000B1H
16                ;
17                ;ADRESSE DEBUT TABLEAU TOUC.FRAP.
18                CARACO:     EQU  0B63FH
19                ;
20                ;REGISTER ADRESS DU CRT
21                ARCRT:      EQU  0BC00H
22                ;
23                ;REGISTRE R1 DU CRT
24                R1CRT:      EQU  0BD00H
25                ;
26                ;*********************************
27                ;
28                ;DIRECTIVE DE L'ORIGINE
29                ;D'ASSEMBLAGE
30                ;
31                            ORG  0A000H
32                            LOAD 0A000H
33                ;
34                ;**********************
35                ;
36                ;
37                ;ROUTINE D'INSTALLATION POUR
38                ;DETOURNEMENT DE LA ROUTINE
39                ;DE TRAITEMENT D'INTERRUPTION
40                ;*****************************
41                ;
42 A000 F3                 DI                      ;INHIBITION INTERRUPTION
43 A001 2109A0             LD   HL,DEPART          ;ADRESSE DEPART ROUTIN
44 A004 2251B9             LD   (INTER),HL         ;A LA PLACE DE INTER
45 A007 FB                 EI
46 A008 C9                 RET                     ;FIN D'INSTALLATION
```

```
  47                ;****************************
  48                ;
  49                ;
;R00 E              DEPART:;PREPARATION AU RETOUR EN
  51                ;ROM EN FIN DE TRAITEMENT
  52 A009 F3                    DI                      ;INHIBITION INTERRUPTION
  53 A00A 21B100                LD    HL,RETOU          ;ADRESSE DE RETOUR
  54 A00D E5                    PUSH  HL                ;DANS LA PILE
  55                ;
X0B6A               DEBUT:
  57 A00E AF                    XOR   A                 ;PLACE 00H
  58 A00F 3278A0                LD    (VARTEM),A        ;DANS VARIABLE TEST
  59 A012 213FB6                LD    HL,CARACO         ;POINTE PREMIERE
  60                ;ADRESSE DU TABLEAU
  61 A015 110100                LD    DE,000001H        ;VALEUR D'INCREMENT
  62                ;
  63 A018 7E        LOOP1:      LD    A,(HL)            ;CHARGE POUR TEST
  64 A019 FEFF                  CP    0FFH              ;DU CONTENU ADRESSE
  65 A01B 2805                  JR    Z,LOOP2           ;TOUCHE FRAPPEE ?
  66 A01D 3E01                  LD    A,01H
  67 A01F 3278A0                LD    (VARTEM),A
  68 A022 19        LOOP2:      ADD   HL,DE             ;INCREMENTE ADRESSE
  69 A023 7D                    LD    A,L
  70 A024 FE49                  CP    049H              ;DERNIERE ADRESSE?
  71 A026 20F0                  JR    NZ,LOOP1          ;SAUT SI NON
  72                ;
  73 A028 3A78A0    TESTE:      LD    A,(VARTEM)        ;CHARGE A
  74 A02B FE00                  CP    00H               ;AVEC VARIABLE INDIQUANT
  75 A02D 2822                  JR    Z,CARNON          ;TOUCHE FRAPPEE
  76                ;
  77 A02F 210000    CAROUI:     LD    HL,0000           ;RE-INITIALISE
  78 A032 227AA0                LD    (COMPT),HL        ;LE COMPTEUR
  79 A035 3A79A0                LD    A,(FLAG)          ;CHARGE A AVEC LE
  80 A038 FE00                  CP    00H               ;FLAG POUR TESTER SI
  81 A03A 2840                  JR    Z,FIN             ;L'ECRAN EST DEJA ETEINT
  82 A03C 3E00                  LD    A,00H             ;SINON
  83 A03E 3279A0                LD    (FLAG),A          ;POSITIONNER LE FLAG
  84 A041 0100BC                LD    BC,ARCRT          ;PREPARER PORT ARCRT
  85 A044 3E01                  LD    A,01H             ;POUR
  86 A046 ED79                  OUT   (C),A             ;SELECTIONNER R1CRT
  87 A048 0100BD                LD    BC,R1CRT          ;PREPARER PORT R1CRT
  88 A04B 3E28                  LD    A,028H            ;POUR
  89 A04D ED79                  OUT   (C),A             ;AUTORISER AFFICHAGE
  90 A04F 182B                  JR    FIN               ;SAUT A FIN
  91                ;
  92 A051 2A7AA0    CARNON:     LD    HL,(COMPT)        ;CHARGER
  93 A054 19                    ADD   HL,DE             ;COMPTEUR POUR INCREMENTE
  94 A055 3805                  JR    C,ETEINT          ;SAUT SI UN TOUR
  95 A057 227AA0                LD    (COMPT),HL        ;SINON SAUVER COMPTEU
  96 A05A 1820                  JR    FIN               ;SAUT A FIN
  97                ;
  98 A05C 3A79A0    ETEINT:     LD    A,(FLAG)          ;CHARGE FLAG
  99 A05F FE01                  CP    01H               ;POUR
 100 A061 2819                  JR    Z,FIN             ;POUR TEST SI DEJA ETEINT
 101 A063 3E01                  LD    A,01H             ;SINON SIGNALER EXTINCTIO
 102 A065 3279A0                LD    (FLAG),A          ;DANS LE FLAG
 103 A068 0100BC                LD    BC,ARCRT          ;PREPARER PORT ARCRT
 104 A06B 3E01                  LD    A,01H             ;POUR
```

```
105 A06D ED79                OUT   (C),A              ;SELECTIONNER R1CRT
106 A06F 0100BD              LD    BC,R1CRT           ;PREPARER PORT R1CRT
107 A072 3E00                LD    A,00H              ;POUR
108 A074 ED79                OUT   (C),A              ;INHIBER L'AFFICHAGE
109 A076 1804                JR    FIN                ;SAUT A FIN
110                     ;
111                     ;
112                     ;ADRESSES VARIABLES UTILISEE
113                     ;***************************
114                     ;
115 A078 00             VARTEM:     DEFB 00H
116 A079 00             FLAG:       DEFB 00H
117 A07A 0000           COMPT:      DEFB 00H,00H
118                     ;***************************
119                     ;
120                     ;
121 A07C C9             FIN:        RET                ;FIN DE TRAITEMENT
122                     ;
123                                 END
```

Après compilation et sauvegarde, vous pourrez utiliser ce programme à partir du Basic en frappant :

MEMORY &9FFF : LOAD « EFFECR.BIN » : CALL &A000

si vous avez dénommé le programme sauvegarde : EFFECR.BIN

LE CHARGEUR BASIC DES CODES MACHINES

Pour ceux d'entre vous qui sont intéressés par l'utilisation de cette routine et qui ne possédent pas d'Assembleur, nous vous donnons ci-dessous le chargeur Basic.

```
10 REM ***  INSTALLATION EN BASIC  ***
20 REM ***  DE LA ROUTINE MACHINE  ***
30 REM *** DE TRAITEMENT CAPS LOCK ***
40 REM ********************************
50 FOR ADRESSE = &A000 TO &A07C:REM adre
sses de chargement
60 READ DONNEE$:REM lecture des codes ma
chines
70 DONNEE=VAL("&"+DONNEE$):REM transform
ation hexa
80 SOMME = SOMME + DONNEE:REM somme des
codes machines
90 POKE ADRESSE,DONNEE:REM chargement ef
fectif d'un code
```

```
100 NEXT:REM code suivant
110 READ CONTROLE:REM lecture de la somm
e de controle
120 IF CONTROLE = SOMME THEN 140:REM ver
ification
130 MODE 2:PRINT "ERREUR DANS LES LIGNES
 DE DATAs":LIST:REM erreur
140 MODE 2:PRINT "SAUVEGARDE PAR :":REM
chargement ok
150 PRINT "SAVE "+CHR$(34)+"EFFECR.BIN"+
CHR$(34)+",B,&A000,&A07C-&A000":REM reco
mmendations d'utilisation
160 PRINT:PRINT
170 PRINT"UTILISATION PAR :"
180 PRINT"MEMORY &9FFF : LOAD "+CHR$(34)
+"EFFECR.BIN"+CHR$(34)+",&A000 : CALL &A
000"
190 REM ********************************
200 REM *** codes operations a charger *
**
210 DATA F3,21,09,A0,22,51,B9,FB
220 DATA C9,F3,21,B1,00,E5,AF,32
230 DATA 78,A0,21,3F,B6,11,01,00
240 DATA 7E,FE,FF,28,05,3E,01,32
250 DATA 78,A0,19,7D,FE,49,20,F0
260 DATA 3A,78,A0,FE,00,28,22,21
270 DATA 00,00,22,7A,A0,3A,79,A0
280 DATA FE,00,28,40,3E,00,32,79
290 DATA A0,01,00,BC,3E,01,ED,79
300 DATA 01,00,BD,3E,28,ED,79,18
310 DATA 2B,2A,7A,A0,19,38,05,22
320 DATA 7A,A0,18,20,3A,79,A0,FE
330 DATA 01,28,19,3E,01,32,79,A0
340 DATA 01,00,BC,3E,01,ED,79,01
350 DATA 00,BD,3E,00,ED,79,18,04
354 DATA 00,00,00,00,C9
355 DATA 11218
360 REM ********************************
```

Les contenus des lignes de DATAs correspondent aux codes hexadécimaux résultant de la compilation.

Nous vous conseillons, avant toute utilisation de ce programme, de le sauvegarder sur disquette.

Lors du lancement, si une erreur de frappe des DATAs a été commise, le programme vous redonne le listing complet pour corriger.

Suite au succès de chargement des codes machines, le programme vous indique la méthode de sauvegarde de la routine ainsi créée (l'instruction **MEMORY &9FFF** sert à protéger le sous-programme d'un éventuel débordement d'un programme ou des variables Basic sur cette zone d'adresses).

Surtout, n'oubliez pas le **CALL &A000** après chargement du code machine, sinon la routine ne sera pas initialisée.

ESSAIS PRÉALABLES

La routine permet d'inhiber l'affichage sur l'écran de votre CPC au bout d'un certain temps déterminé par le compteur « COMPT ».

Ce compteur effectue un comptage complet sur 2 octets. Le temps au bout duquel l'affichage sera inhibé est calculé par la formule :

Temps = (COMPT) × 3.3 ms
= (&FFFF) × 3.3 ms
= (15×16×16×16 + 15×16×16 + 15×16 + 15) × 3.3 ms
= 216,3 secondes
= 3 minutes et 16 secondes environ

Si nous voulons, lors des essais de cette routine, ne pas attendre le temps complet, nous vous proposons le petit programme Basic suivant qui initialise le compteur à une valeur relativement élevée (&D000), et visualise sa progression.

Attention, dès que vous appuyez sur une touche, le compteur est réinitialisé à zéro.

```
10 FOR I = I TO 500 : REM attente fin appui sur
20 NEXT I : REM la touche <RETURN>
30 POKE &A07B,&E0 : REM octet de poids fort du compteur
40 POKE &A07A,&00 : REM octet de poids faible du compteur
50 PRINT HEX$(PEEK(&a07B)); : affiche octet fort
60 PRINT HEX$(PEEK(&A07A)); : REM affiche octet faible
70 PRINT SPACE$(3);CHR$(13) : REM retour position initial
80 GOTO 50
90 END
```