

Un moniteur/désassembleur pour le Canon X07

La programmation en langage machine à l'aide des seules instructions PEEK, POKE, EXEC... est souvent très fastidieuse. Le logiciel proposé ici est une aide à la programmation et remplit les fonctions essentielles d'un désassembleur-moniteur. Il est utilisable directement sur un Canon X07 en version de base ou, après adaptation, sur toute machine disposant d'un Basic standard et architecturée autour d'un microprocesseur Z 80 ou compatible.

Un désassembleur a pour fonction de traduire un programme, présent sous forme de codes opératoires, en mnémoniques appartenant au langage d'assemblage du microprocesseur utilisé. Cette représentation symbolique est très pratique pour l'utilisateur car beaucoup plus naturelle qu'une présentation sous forme binaire ou hexadécimale. L'intérêt essentiel d'un tel programme est de permettre une étude rigoureuse du logiciel de base contenu dans la ROM du micro-ordinateur. En outre, il peut être utilisé pour vérifier que l'implantation d'un programme en mémoire s'est faite correctement ou pour examiner un logiciel avant de procéder à d'éventuelles modifications.

Méthodologie

Le problème que le programme est amené à résoudre est de transformer le code opératoire d'une instruction en sa représentation symbolique. La première idée qui vient en général à l'esprit est de stocker toutes les mnémoniques dans des DATA\$, en indiquant devant chacun d'elles le nombre d'octets utilisés ainsi que le type de donnée. Étant donné le grand nombre d'instructions que possède le Z 80 (plus de 700), cette méthode se révèle beaucoup trop coûteuse en place mémoire. Il a été choisi ici d'utiliser plutôt les relations liant codes et mnémoniques.

Prenons un exemple caractéristique : celui des instructions dont le code est compris entre 64 et 127 soit, en hexa, de 40 à 7F (fig. 1).

UTILITAIRE
Un moniteur désassembleur Z 80
d'Emmanuel SANDER
Avec cet utilitaire, inexistant dans
la version de base du X07, analysez
la ROM et découvrez les secrets de
votre interpréteur.
Langage : Basic
Ordinateur : Canon X07

En les observant, nous remarquons que ces instructions sont toutes du type :

LD R, R'

où R et R' représentent des registres simples ou (HL).

Nous nous apercevons également que de 40h(exa) à 47h, les instructions sont de la forme LD B,R' puis, de 48h à 4Fh, de la forme LD C,R' et ce jusqu'à l'intervalle 78h - 7Fh du type LD A,R'. Soustrayons 40 h au code de l'instruction, divisons ce qui est produit par 8 et ne gardons que la partie entière du résultat. Si le code était compris entre 40h et 47h, nous trouvons finalement, entre 40h et 47h : « 0 », entre 48h et 4Fh : 1... entre 78h et 7Fh : 7.

Selon ce résultat, nous pouvons donc déduire si l'instruction était du type LD B,R' (résultat = 0) ; LD C,R' (résultat = D...) LD A,R' (résultat = 7).

Nous venons de déterminer un moyen pour trouver R dans une instruction de la forme LD R,R' en connaissant le code de l'instruction et l'ordre dans lequel les registres se succèdent.

Cherchons maintenant comment déterminer R' : l'instruction correspondant au code 40h est LD B,B ; 41h = LD B,C ; 42h = LD B,D... ; 48h = LD C,B ; 49h = LD C,C ; 4Ah = LD C,D... Nous remarquons que, chaque fois que R varie, R' redevient égal à B puis C, D, E, H, L, (HL), A. Cet ordre de variation est d'ailleurs le même que pour R. Considérons comme un cycle la suite d'instructions durant laquelle R reste constant. Ce qui nous intéresse n'est pas de savoir dans quel cycle nous sommes (si R = B ou R = C...) mais à quel

40	LD b,b	41	LD b,c
42	LD b,d	43	LD b,e
44	LD b,h	45	LD b,l
46	LD b,(hl)	47	LD b,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
4E	LD c,(hl)	4F	LD c,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
56	LD d,(hl)	57	LD d,a
58	LD e,b	59	LD e,c
5A	LD e,d	5B	LD e,e
5C	LD e,h	5D	LD e,l
5E	LD e,(hl)	5F	LD e,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
66	LD h,(hl)	67	LD h,a
68	LD i,b	69	LD i,c
6A	LD i,d	6B	LD i,e
6C	LD i,h	6D	LD i,l
6E	LD i,(hl)	6F	LD i,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
76	HALT	77	LD (hl),a
78	LD a,b	79	LD a,c
7A	LD a,d	7B	LD a,e
7C	LD a,h	7D	LD a,l
7E	LD a,(hl)	7F	LD a,a

Fig. 1. – Codes hexadécimaux et mnémoniques correspondant aux valeurs 40h et 7Fh.

niveau de ce cycle (la valeur de R'). Or la durée du cycle est de 8 octets. Pour connaître le niveau du cycle, il nous suffit donc de connaître le reste de la division entière du nombre par 8 (cette opération est appelée modulo). Nous en déduisons ainsi R' selon le résultat obtenu (B = 0, C = 1, ..., (HL) = 6, A = 7). Ce sont exactement ces deux opérations qu'effectue le X07 lors du décodage. Ayant stocké le nom des registres et leur ordre en Data et connaissant le code de l'instruction, il déduit R et R' selon les méthodes décrites précédemment.

Mais si vous programmez en Assembleur Z80, vous savez sans doute que l'instruction ayant pour code 76h n'est pas LD (HL),(HL) comme nous l'aurait fourni notre algorithme, mais HALT. Il s'agit de la seule exception à la règle énoncée dans cet intervalle, mais le test nécessaire à son décodage prend presque autant de place mémoire que la ligne qui décode toutes les instructions de l'intervalle. Comme vous vous en apercevez en regardant le programme, cette configuration est à peu près générale. Car si presque toutes les instructions sont codées suivant des règles précises, il existe souvent des exceptions qui alourdissent considérablement les tests.

Utilisation du désassemblleur

Celui-ci est tout à fait classique. Présentons-le rapidement. Les mnémoniques utilisées sont standards (**fig. 2 et 3**).

A gauche de l'écran est indiquée l'adresse courante, suivie de la mnémonique correspondante.

Tous les nombres sont représentés en hexadécimal. Un nombre codé sur 16 bits est systématiquement affiché sous forme de 4 chiffres hexadécimaux. Un nombre codé sur 8 bits est affiché sous forme de 2 chiffres hexadécimaux. Pour ne pas prêter à confusion, les registres sont représentés par des lettres minuscules. Dans un même esprit de clarification, l'adresse de branchement des sauts relatifs est indiquée en valeur d'adresse absolue.

Si vous disposez d'une imprimante, le désassemblage se fait simultanément sur papier. (Pour ce faire, une instruction

017A	LD (035E),a
017D	LD (035F),bc
0181	LD (0361),de
0185	LD (0363),hl
0188	LD (0365),sp
018C	LD (0367),IX
0190	LD (0369),IY
0194	PUSH hl
0195	PUSH af
0196	LD a,i
0198	LD (036B),a
019B	LD a,r
019D	LD (036C),a
01A0	POP hl
01A1	LD (036D),hl
01A4	PUSH hl
01A5	POP af
01A6	POP hl
01A7	EX (sp),hl
01A8	LD (036F),hl
01AB	EX (sp),hl
01AC	RET

Fig. 2. - Désassemblage du programme langage machine permettant l'affichage du contenu des registres.

C3C3	DI
C3C4	LD sp,01D4
C3C7	XOR a :
C3C8	OUT (F1),a
C3CA	OUT (F0),a
C3CC	DEC a
C3CD	OUT (BB),a
C3CF	LD a,97
C3D1	OUT (F5),a
C3D3	IN a,(F2)
C3D5	RRCA
C3D6	JR C,C3CF
C3D8	LD hl,C6DB
C3DB	LD de,0000
C3DE	LD bc,00AE
C3E1	LDIR
C3E3	CALL C39D
C3E6	CALL C635
C3E9	CALL C0BD
C3EC	CALL C557
C3EF	CALL C62E
C3F2	LD a,00
C3F4	JR NZ,C3FE

Fig. 3. - Exemple de désassemblage (option D) : début du programme d'initialisation.

LPRINT suit chaque instruction PRINT. La non-connexion d'une imprimante ne provoque pas l'arrêt du programme lors de l'exécution du LPRINT et n'entraîne qu'un très faible ralentissement d'exécution.)

Après avoir entré « RUN », choisissez l'option D (taper « D » puis Return) et entrez l'adresse de début de désassemblage (la faire précédé de &H si elle est en hexadécimal).

Vous pouvez vous servir utilement de deux commandes qui sont l'appui sur la touche R pour revenir au menu et sur CTRL-S pour arrêter momentanément le déroulement du programme ; appuyez alors sur n'importe quelle touche pour reprendre.

0000	C9 00 00 1E
0004	50 02 77 02
0008	C3 2F F5 64
000C	00 00 00 00
0010	C3 37 F5 00
0014	00 00 03 12
0018	C3 C7 C9 3A
001C	00 00 00 00
0020	C3 45 EE 00
0024	00 00 00 00
0028	C3 8F E8 00
002C	C9 00 00 00
0030	C3 2F FC 00

Fig. 4. - Exemple d'utilisation du chargeur hexadécimal (option P) : entrée d'un court programme.

Le moniteur

Un moniteur est une aide à la programmation en langage machine. Il contient des fonctions importantes permettant l'implantation, la mise au point et l'exécution de programmes écrits en langage machine.

La méthode de programmation utilisée ne présente ni originalité ni difficulté particulière. Les instructions spécifiques au Canon dont il est fait usage sont décrites dans la partie « adaptation ».

Outre l'option D déjà largement décrite, le moniteur comprend onze fonctions. Nous allons les examiner séparément :

■ **Option P** : Elle donne accès à un chargeur hexadécimal. Il est possible d'introduire le programme ou les données octet par octet en validant chaque entrée par un appui sur la touche Return. Mais vous pouvez aussi introduire vos codes à la suite, et ce à condition que chaque nombre soit exprimé sous forme de deux chiffres hexadécimaux. Ce n'est que lorsque l'entrée est terminée que vous devez appuyer sur la touche Return. L'adresse courante devient alors la première adresse non modifiée, et vous pouvez continuer votre implantation. En cas d'erreur, un appui sur la touche « - » décrémentera l'adresse courante d'un octet et vous permettra ainsi de procéder à une correction, tandis qu'un appui sur la touche R provoquera un retour au menu (**fig. 4**).

■ **Option T** : Elle a pour fonction d'effectuer le transfert d'un bloc d'octets d'une zone mémoire vers une autre.

On peut, en particulier, recopier un bloc de ROM en RAM et procéder ainsi à certaines modifications.

■ **Option M** : Elle permet d'implanter sous forme ASCII un message ou toute autre chaîne de caractères en mémoire à partir d'une adresse donnée.

■ **Option Z** : Cette fonction est à appeler lorsque vous désirez effacer un programme ou des données.

L'utilisation de cette option provoque la mise à zéro des octets de la zone mémoire concernée.

■ **Option V** : Il s'agit d'un viddage mémoire hexadécimal du contenu de la mémoire.

Le déroulement est continu, mais un appui sur CTRL-S en permet un arrêt temporaire, tandis qu'une pression sur la touche « - » décrémente de 8 octets l'adresse courante et qu'un appui sur la touche « R » provoque un retour au menu.

Un tel type d'examen mémoire est moins lisible, dans le cas d'un programme, qu'une liste de mnémoniques, mais permet une meilleure appréciation de la structure de la mémoire (zone réservée aux données, au programme Basic, non utilisée par le système...).

L'adresse courante est affichée tous les quatre octets ; l'écran permet donc un affichage simultané de seize octets (**fig. 5**).

```

C677 l o u e
C67B # F S
C67F E r r
C683 o r
C687 # M
C68B C E r
C68F r o r
C693 C r
C697 e a t e
C69B s y s
C69F t e m
C6A3 B y t
C6A7 e s f
C6AB r e e
C6AF C
C6B3 o p y r
C6B7 i g h t
C6BB ( c )
C6BF 1 9 8 3
C6C3 b y

```

Fig. 5. – Exemple de vidage mémoire hexadécimal par l'option V des premiers octets de la RAM.

■ Option C : Il s'agit cette fois d'un dump ASCII du contenu de la mémoire, ce qui correspond à l'affichage sous forme de caractères des codes rencontrés. Cette option a essentiellement pour but de déchiffrer les messages contenus dans la ROM du X07 : mots clés Basic, messages d'erreur... et d'éviter ainsi certaines confusions. On pourra, de même que pour l'option V, utiliser le CTRL-S, les touches « - » et « R ». (fig. 6).

■ Option R : Elle affiche le contenu de différents registres du microprocesseur. Tous les registres sont représentés en hexadécimal sauf le registre d'état qui est inscrit en binaire. Un appui sur une touche quelconque provoque un retour au menu (fig. 7).

■ Option G : Cette fonction permet l'exécution d'une routine écrite en langage machine avec toutefois une certaine spécificité. En effet, son exécution achevée, votre programme ne revient pas tout de suite au Basic, mais sauvegarde auparavant la valeur du contenu de ses différents registres dans des adresses mémoires. Lors du re-

```

1B58 00 16 05 3E FF
1B5C 00 03 F4 2E 00 0E F3 ED 69 AF
1B65 22 03 F2
1B67 22 06 00 10 FD
1B6B 22 3C 20 F7
1B6E 22 2C 7D FE 03 20 EE
1B74 22 15 20 E7 AF 03 F4 C9
1B7B 00

```

Fig. 6. – Exemple de vidage mémoire ASCII par l'option C de quelques messages contenus dans la ROM.

```

a=FF bc=000A de=CECD
hl=017A sp=1F89 i=00
IX=0FF9 IY=2FF7 r=28
t=10001100(sp)=CECD

```

Fig. 7. – Affichage du contenu des registres du microprocesseur par l'option R.

tour au Basic, l'impression du contenu des différents registres est faite. Il est ainsi possible de connaître le contenu des registres pour fournir des indications lors de la correction de certaines erreurs et de savoir l'utilisation qui est faite des différents registres lors de l'appel de routines situées en ROM.

■ Option K : Il s'agit d'un convertisseur décimal-hexadécimal et réciproquement. Dans le cas d'une conversion hexadécimal-décimal, il est nécessaire de faire précédé le nombre écrit en hexadécimal de &H.

Une fois la conversion effectuée, un appui sur une touche quelconque provoque un retour au menu.

■ Option S : Sauvegarde sur cassette, identifiée par un nom, d'une zone donnée de la mémoire. Il s'agit d'une fonction puissante du moniteur, car le Basic du X07 ne dispose pas d'instruction spécifique à cet effet.

■ Option L : Chargement d'un programme sauvegardé sur cassette à l'aide de l'option S.

Utilisation du moniteur

Dans chaque cas, il faut introduire la lettre caractérisant l'option choisie puis valider son entrée par un appui sur la tou-

che Return. Il est ensuite nécessaire de respecter la syntaxe spécifique à l'option choisie, indiquée dans l'**encadré 1**.

Comme pour le désassemblleur, si une imprimante est connectée, l'impression se fait simultanément sur papier.

Le moniteur seul occupe à peu près 1 600 octets, ce qui représente avec le désassemblleur un total de 4,6 Ko. D'autre part, les variables occupent approximativement 400 octets. Il vous reste donc, dans le cas d'un Canon en version de base, à peu près 1,7 Ko pour écrire vos propres programmes, ce qui se révèle amplement suffisant pour la plupart des applications en langage machine.

Remarques importantes

L'imprimante table traçante qui équipe le X07 est disponible pour plusieurs ordinateurs. Sa police de caractères ne lui est donc pas spécifique et ne dispose pas du signe $\frac{X}{Y}$ (symbole de la division entière) qui est remplacé par le signe « \ » dans le listing du programme.

Il est nécessaire, pour utiliser les options R et G, d'implanter un programme écrit en langage machine. Cette implantation doit être faite lors de la première utilisation du programme ou après chaque utilisation du bouton Reset. Pour cela, il suffit de lancer le programme par « GOTO 10000 ».

Les adaptations

Si vous ne possédez pas de Canon X07 mais que votre ordinateur dispose d'un Basic standard et est architecturé autour d'un Z 80, vous pouvez, moyennant quelques modifica-

tions, adapter ce programme sur votre machine.

Pour cela, nous allons examiner rapidement les fonctions spécifiques au X07.

- $\frac{X}{Y}$ (lire yen) est le symbole de la division entière ainsi $X \frac{Y}{Y} = \text{int}(X/Y)$.

- MOD calcule le module de deux nombres. En Basic standard, $X \text{ MOD } Y = X - \text{int}(X/Y) * Y$. Il est conseillé dans ce dernier cas, de définir une fonction qui serait équivalente à l'opération MOD.

- STRING\$ (N,A\$) : produit N fois le premier caractère de la chaîne A\$. Cette fonction peut être remplacée par le plus classique LEFT\$ en produisant une chaîne LEFT\$ (B\$, N) où B\$ est une chaîne qui contient la valeur maximale de N fois le caractère que l'on veut reproduire. Signalons également les différentes instructions de fichier utilisées dans le programme.

Toutefois, celles-ci ne sont utilisées que pour pallier une déficience du Basic du X07 qui est l'absence d'instruction permettant la sauvegarde et le chargement cassette d'une zone mémoire. Il y a donc tout lieu de penser que les possesseurs d'autres matériels n'auront pas à utiliser les deux options concernées.

- INIT # : ouvre une unité avant l'utilisation d'une opération d'entrée-sortie.

- OUT # : délivre une valeur comprise entre 0 et 255 vers un fichier.

- INP (#N) : produit une donnée extraite du fichier de numéro N.

D'autre part, les options R et G utilisent un petit programme écrit en langage machine qui se contente d'implanter en mémoire la valeur des registres. L'adresse de stockage du programme et des données varie selon le matériel utilisé ; il est donc nécessaire de procéder à quelques modifications. Le listing (en langage d'assemblage) de ce court programme (fig. 2) vous y aidera. Enfin, on remarquera l'utilisation du DEFSTR qui permet de préciser les variables alphanumériques.

Nous espérons que ce programme vous sera utile et qu'il contribuera à faire de vous, si vous ne l'êtes pas déjà, un adepte de la programmation en Assembleur. ■

OPTION	ROLE	SYNTAXE	COMMANDES
C	Dump ASCII	Adresse de départ	CTRL-S, -, R
D	Désassemblage	Adresse de départ	CTRL-S, R
G	Exécution	Adresse d'exécution	Touche quelconque
K	Conversion	Nombre à convertir (le précéder de &H pour une conversion hexadécimale)	Touche quelconque
L	Chargement	Donnée numérique fictive puis nom du programme	
M	Implantation d'un message	Adresse de début d'implantation puis message	
P	Chargeur hexa	Adresse de début de chargement	RETURN, -, R
R	Affichage du contenu des registres	Donnée numérique fictive	Touche quelconque
S	Sauvegarde	Adresse de début de sauvegarde Adresse de fin de sauvegarde Nom du programme	
T	Transfert	Adresse initiale Adresse de destination Nombre d'octets à transférer	
U	Dump hexa	Adresse de départ	CTRL-S, -, R
Z	Remise à zéro	Adresse de départ, adresse finale	

Encadré 1 : Utilisation du moniteur.
Pour plus de précisions sur le rôle de chaque option et des commandes, se reporter à la description générale de chaque option.

En général :
CTRL-S provoque un arrêt temporaire.
R ou une touche quelconque provoque un retour au menu.
- Provoque une décrémentation de l'adresse courante.

RETURN valide une entrée.
Une donnée numérique peut être introduite en hexa si elle est précédée de &H.

Structure interne du programme

1-40 : Initialisation : définition des fonctions, choix de l'option.
70-9080 : Désassembleur décomposé en :
70-75 : Chargement du tableau.
80-190 : Branchement selon le code de l'instruction et traitement de certaines exceptions.
200 : Traitement des instructions dont le code est compris entre 40 h et 7Fh.
250 : Traitement des instructions dont le code est compris entre 80 h et BFh.
500-910 : Traitement des instructions dont le code est inférieur à 40 h.
990-1020 : Traitement des instructions dont le code est préfixé par CBh.
2000-2500 : Traitement des instructions dont le code est supérieur à COh.
3500-4300 : Traitement des instructions dont le code est préfixé par EDh.
4400-4500 : Traitement des instructions dont le code est préfixé par DDh ou FDh.
5000-5110 : Routine d'impression appelée après le décodage de chaque mnémonique.
9010-9080 : Données du désassembleur.
9090-9660 : Moniteur décomposé en :
9090-9110 : Traitement de l'option G.
9120 : Traitement de l'option K.
9170 : Traitement de l'option M.
9180-9230 : Traitement de l'option S.
9240-9300 : Traitement de l'option L.
9310 : Traitement de l'option Z.
9370-9380 : Test des options R et P.
9390 : Traitement de l'option T.
9410 : Test et traitement dans le cas d'une option inexisteante.
9420-9490 : Traitement des options V et C.
9500-9540 : Traitement de l'option R.
9600-9660 : Traitement de l'option P.
10000-10050 : Programme d'implantation des codes machines pour les options R et G.
10100-10300 : Programme machine contenu sous forme de code hexa dans des lignes de Data.

Liste du rôle des variables utilisées

Le \$ rendu facultatif pour certaines variables alphanumériques par l'instruction DEF STR est indiqué entre () lorsque le cas se présente.
A : Contient le « peek » de l'adresse courante dans le désassembleur.
B(\$) : Chaîne contenant la mnémonique de l'instruction courante.
CO : Adresse du premier octet de la mnémonique courante.
E(\$) : Option choisie et plusieurs rôles annexes.
F : Contient, dans le moniteur, les adresses de fin pour les options L, S et Z.
F(\$) : Concerne les registres IX ou IY : contient la partie de la mnémonique de la forme (IX + d) ou (IY + d).
I : Adresse courante dans le désassembleur et adresse de départ dans le moniteur.
J : Contient la valeur de la division entière de A par 8.
I(\$) : Chaîne contenant des données destinées à être affichées et imprimées.
K : Contient le reste de la division entière de A par 16.
L(\$) : Chaîne contenant IX ou IY selon le préfixe utilisé.
M : Nombre d'octets à transférer (option T).
N : Adresse de destination (option T).
N(\$) : Chaîne contenant les codes des octets à planter (option P).
O(\$) : Tableau contenant les données nécessaires au fonctionnement du désassembleur.
W(\$) : Contient le nom du registre double utilisé.
X(\$) : Contient, en hexa, une opérande codée sur 16 bits.
Z : Contient le reste de la division entière de A par 8 fonctions utilisées.
FNG(\$) : Convertit un nombre décimal sous forme de 4 chiffres hexa.
FNU(\$) : Convertit une opérande occupant un octet sous forme de 2 chiffres hexa.
FNY(\$) : Convertit un nombre codé sur 2 octets sous forme de 4 chiffres hexa.

```

1 REM DESASSEMBLEUR- MONITEUR POUR X07
2 REM (c) EMMANUEL SANDER 1984
4 GOTO9
5 GOTO5000
9 CLS
15 CLEAR99
20 DEFSTRO,B,G,Y,W,X,L,E,U
25 DEFFNG(K)=STRING$(4-LEN(HEX$(K)), "0")+
HEX$(K)
30 DEFFNU(I)=STRING$(-(PEEK(I)<16), "0")+
HEX$(PEEK(I))
35 INPUTE,I:LPRINT
40 DEFFNY(I)=FNG(256*PEEK(I)+PEEK(I-1)):
IFE <>"D"THEN9090
70 DIMO(75)
75 FORT=0TO75:READO(T):NEXT
80 A=PEEK(I):C0=I
85 Z=AMOD8
95 J=A\8
100 IFA=203THEN990
105 IFA=232THEN3500
110 IFA=0THENB="NOP":GOTOS
120 IFA=8THENB="EX af,af)":GOTOS
150 IFA<64THEN500
155 IFA>191THEN2000
190 IFA=118THENB="HALT":GOTOS
200 IFA<128THENB="LD "+ "+0(12+J)+", "+0(
20+Z):GOTOS
250 B=0(J-15)+0(20+Z):GOTOS
500 W=0(A\16+28)
505 IFZ=0THENGOSUB550
510 ONZGOSUB600,680,760,810,830,860,900:
GOTOS
550 B=0(30+J)
560 I=I+1
570 B=B+FNG(I+PEEK(I)+1-(PEEK(I)\128)*25
6)
580 RETURN
600 IFAMOD16=9THENB="ADD "+0(30)+", "+W:RET
URN
610 I=I+2
620 B="LD "+W+": "+FNY(I):RETURN
680 IFA<32THEN700
682 I=I+2:X=FNY(I):IFA=34THENB="LD ("+X+
"), "+0(30)
687 IFA=42THENB="LD "+0(30)+", ("+X+")
690 IFA=50THENB="LD ("+X+"), a"
692 IFA=58THENB="LD a, ("+X+")
695 RETURN
700 IFAMOD16=10THENL="a," :E="" ELSEL="" :E
="," ,a"
705 B="LD "+L+"(" +W+ ")" +E:RETURN
760 IFAMOD16=11THENB="DEC "ELSEB="INC "
880 RETURN
900 B=0(J+38)
910 RETURN

```

```

990 I=I+1:A=PEEK(I):IFA<64THENB=0(9+A\8)
+" "+0(20+AMOD8):GOTOS
1020 B=0(16+A\64)+STR$((AMOD64)\8)+", "+0(
20+AMOD8):GOTOS
2000 E=""
2010 IFZ=0THENB="RET "+0(J+22):GOTOS
2020 IFZ=2THENE="JP"
2030 IFZ=4THENE="CALL"
2050 IFE <>"" THENI=I+2:B=E+0(J+22)+", "+FN
Y(I):GOTOS
2060 K=AMOD16
2070 IFK=1THENE="POP"
2080 IFK=5THENE="PUSH"
2090 IFE <>"" THENB=E+0(A\16+42):GOTOS
2100 IFZ=6THENI=I+1:B=0(J-23)+FNU(I)
2110 IFZ=7THENB="RST "+HEX$(A-199):GOTOS
2130 IFK=9THENB=0(A\16+46):GOTOS
2200 IFA=221THENL="IX":GOTO4400
2220 IFA=253THENL="IY":GOTO4400
2250 IFA=243THENB="DI"
2255 IFA=251THENB="EI"
2260 IFA=235THENB="EX de,hl"
2270 IFA=227THENB="EX (sp), "+0(30)
2300 IFA=211THENI=I+1:B="OUT (" +FNU(I)+"
), a"
2310 IFA=219THENI=I+1:B="IN a, (" +FNU(I)+"
)"
2350 IFA=195THENI=I+2:B="JP "+FNY(I)
2380 IFA=205THENI=I+2:B="CALL "+FNY(I)
2500 GOTOS
3500 I=I+1
3510 A=PEEK(I)
3520 J=A\8
3530 IFA>159THENB=0(AMOD4+62)+0(J+46):GO
TOS
3600 Z=AMOD8
3610 IFZ=0THENB="IN "+0(J+12)+", (c)":GOT
OS
3620 IFZ=1THENB="OUT(c), "+0(J+12):GOTOS
3680 X=0(A\16+24):IFZ<>2THEN3800
3710 IFAMOD16=2THENB="SBC hl, ELSEB="ADC
hl,"
3720 B=B+X:GOTOS
3800 IFZ<>3THEN4000
3830 I=I+2:E=FNY(I):IFAMOD16=3THENB="LD
(" +E+"), "+XELSEB="LD "+X+", (" +E+")
"
3850 GOTOS
4000 IFZ=7THENB=0(J+62):GOTOS
4200 B"":IFA=68THENB="NEG"
4220 IFA=69THENB="RETN"
4230 IFA=70THENB="IM 0"
4240 IFA=77THENB="RETI"
4260 IFA=86THENB="IM 1"
4280 IFA=94THENB="IM 2"
4300 GOTOS
4400 I=I+1:F=PEEK(I+1):IFF>127THENF=F-25

```

Fig. 8. - Listing du programme complet.

```

6 :E="--ELSEE=+""
4415 F$=""+L+E+HEX$(ABS(F))+")"
4430 IFPEEK(I)=54THENI=I+2:B="LD "+F$+","
"+FNU(I):GOT05050
4450 O(30)=L:O(26)=F$:O(56)=L:O(60)="JP
(" +L+)":IFPEEK(I)=203THENI=I+1:GOT0990
4500 GOT080
5000 O(26)="(hl)":O(30)="hl":O(56)="hl":
O(60)="JP (hl)"
5010 IFL<>"ANDINSTR(B,"(+L)<>0ANDPEEK(
I-2)<>203THENI=I+1
5050 I=I+1:PRINTFNG(CO);";B
5060 LPRINTFNG(CO);";B
5100 IFINKEY$="R"THENCLS:RUN
5110 GOT080
9010 DATA LD,"ADD a,","ADC a,","SUB ,","SBC
a,","AND ,XOR ,OR ,","CP "
9015 DATARLC,RRC,RL,RR,SLA,SRA,SLI,SRL,B
IT,RES,SET
9020 DATA b,c,d,e,h,l,(hl),a
9025 DATA b,c,d,e,h,l,sp
9030 DATA DJNZ ,JR ,JR NZ,"JR Z,","JR
NC,","JR C,"
9035 DATARLCA,RRCA,RLA,RRA,DAA,CPL,SCF,C
CF
9040 DATA NZ,Z,NC,C,PO,PE,P,M
9050 DATA b,c,d,e,h,l,af
9060 DATA RET,EXX,JP (hl),"LD sp,hl"
9070 DATA LD,CP,IN,OUT,I,D,IR,DR
9080 DATA "LD i,a","LD r,a","LD a,i","LD
a,r",RRD,RLD
9090 CLS:IFE<>"G"THEN9120
9095 IF I<0THENI=I+65536
9100 POKE375,205
9110 POKE376,I-INT(I/256)*256:POKE377,IN
T(I/256):EXEC375:GOT09505
9120 IFE="K"THENI$=STR$(I-65536*(I<0))+"
=&H"+HEX$(I):PRINTI$:$LPRINTI$:$GOT09540
9130 IFE="M"THENINPUTE:FORT=ITOI+LEN(E)-
1:POKET,ASC(MID$(E,T+1-I)):NEXT:RUN
9140 IFE<>"S"THEN9240
9150 INPUTF,E:INIT#1,"CASO:"
9200 PRINT#1,E,I,F
9205 FORT=0TO200:NEXT
9210 FORT=ITOF
9220 OUT#1,PEEK(T)
9230 NEXT:RUN
9240 IFE<>"L"THEN9310
9250 INPUTE
9260 INIT#1,"CASI :"
9270 INPUT#1,Z$,I,F:IFZ$<>ETHEN9270
9280 FORT=I-1TOF
9290 POKET,INP(#1)
9300 NEXT:RUN
9310 IFE="Z"THENINPUTF:FORT=ITOF:POKET,0
:NEXT:RUN

```

```

9320 IFE="R"THEN9500
9330 K=I:IFE="P"THEN9600
9340 IFE="T"THENINPUTN,M:FORT=0TOM:POKEN
+T,PEEK(I+T):NEXT:RUN
9410 IFE<>"C"ANDE<>"U"THENPRINT"OPTION I
NUALIDE":RUN
9420 IF(I-K)MOD4=0THENPRINT:PRINTFNG(I);
" ";:LPRINT:LPRINTFNG(I);" ";
9430 IFE="U"THENPRINTFNU(I);" ";
:LPRINTF
NU(I);" ";
9450 IFE="C"ANDPEEK(I)>31THENPRINTCHR$(P
EEK(I));" ";
ELSEIFE="C"THENPRINT" ";
9460 IFE="C"ANDPEEK(I)>31THENLPRINTCHR$(P
EEK(I));" ";
ELSEIFE="C"THENLPRINT" ";
9470 A$=INKEY$:IFA$="R"THEN9
9480 IFA$="-"THENI=I-9:K=I+1:CLS
9490 I=I+1:GOT09410
9500 EXEC378
9505 CLS:I$="a=" +FNU(862)+bc=" +FNY(864
)+de=" +FNY(866):PRINTI$:$LPRINTI$:
9520 I$="hl=" +FNY(868)+sp=" +FNY(870)+"
r=" +FNU(875):PRINTI$:$LPRINTI$:
9525 I$="IX=" +FNY(872)+Y=" +FNY(874)+"
r=" +FNU(876):PRINTI$:$LPRINTI$:$I$=" "
9530 A=PEEK(877):FORT=7TO0STEP-1:IFA\2^T
=1THENI$=I$+"1":A=A-2^TELSEI$=I$+"0"
9535 NEXT:I$="f=" +I$+(sp)=" +FNY(880):PR
INTI$:$LPRINTI$:
9540 IFINKEY$=""THEN9540ELSERUN
9600 PRINTFNG(I);";FNU(I);";";
9605 LPRINTFNG(I);";FNU(I);";";
9610 A$=INKEY$:IFA$="R"THEN9
9620 IFA$="-"THENI=I-1:PRINT:LPRINT:GOTO
9600
9630 IFA$=""THEN9610ELSEPRINTA$:$PRINTA
$;
9635 N$=N$+A$:$IFLEN(N$)MOD2=0THENPRINT"
";:LPRINT" ";
9640 IFA$<>CHR$(13)THEN9610
9645 IFN$=CHR$(13)THENI=I+1:N$=""$:PRINT:
LPRINT:GOT09600
9650 IFLEN(N$)>1THENPOKEI,VAL("&H"+LEFT$(
N$,2)):I=I+1:N$=MID$(N$,3):GOT09650
9660 N$=""$:PRINT:LPRINT:GOT09600
10000 RESTORE10100
10010 DIMA$(50)
10020 FORT=0T050
10030 READA$(T)
10035 POKE378+T,VAL("&H"+A$(T))
10040 NEXT
10050 RUN
10100 DATA32,5E,3,ED,43,5F,3,ED,53,61,3,
22,63,3,ED,73,65,3
10200 DATA22,22,67,3,FD,22,69,3,E5,F5,ED
,57,32,6B,3,ED,5F,32,6C,3,E1
10300 DATA22,6D,3,E5,F1,E1,E3,22,6F,3,E3
,C9

```

Fig. 8. Listing (suite).

FORTH : un exercice de style

Le véritable but de cet article est d'étudier l'éditeur de Sprite/Lutin paru dans le numéro 38 : très beau travail, simple et agréable à utiliser... Un seul défaut majeur : il édite les sprites « à l'envers », ce dont on peut s'apercevoir en formant des figures asymétriques (Brest se retrouve à Strasbourg, et inversement!). L'outil est très beau mais déborde du Jupiter de base. Voici comment on peut optimiser le programme, optimisation en emplacement puisque, dans ce cas, l'optimisation en temps n'a guère d'intérêt (le programme passera en effet le plus clair de son temps à attendre votre réaction).

Le premier travail de cet éditeur consiste à afficher la grille de saisie. Pour ce faire, nous avons deux boucles successives, l'une pour les lignes de points, l'autre pour mettre en tête le numéro de ligne, suivi d'un espace.

Lorsque l'on sait qu'un nombre affiché est systématiquement suivi d'un espace, on peut faire l'économie d'un repositionnement par AT et afficher les points « dans la foulée » du numéro de ligne, ce qui nous donne le mot GRILLE, figure 1.

Ensuite, chaque déplacement du curseur souhaité par l'opérateur doit pouvoir être assuré. Les coordonnées H et V n'occupent chacune qu'un octet.

Voyons si nous pouvons regrouper les quatre mots de déplacements du curseur proposés en un seul. Ceci semble en effet facilité par la nature des touches de commande, les effets étant symétriques par rapport à la position médiane. Un DUP + 109 – transformera les codes ASCII des touches en -3 -1 1 3 et 2/MOD ABS donnera le résultat suivant :

Touche	5	6	7	8
Effet	←	↑	↓	→
ASCII	53	54	55	56
(MOD)	-1	-1	1	1
(Quotient)	1	0	0	1

Le modulo donnera la valeur du déplacement selon l'axe défini par le quotient. Le quotient va donc servir à cueillir l'octet de POS à modifier, POS étant la variable remplaçant V et H.

INITIATION Un éditeur de « Sprites » amélioré de M. THIBERGE

Il y a de quoi rester confondu devant les résultats obtenus aux jeux d'initiation au Forth. Tout ne vous ayant pas été dit, certaines réponses fournies peuvent, bien sûr, être améliorées en fonction des mots non dévoilés.

Langage : Forth
Ordinateur : Jupiter Ace

```
: GRILLE
16 0
DO
  I 10 AT I 8
  MOD 1+I' 0
  DO
    ." "
  LOOP
LOOP
17 12 AT ." 1234567812345678"
;
```

Fig. 1. – Le mot de définition de la grille de saisie des lutins. On remarquera le 8 MOD 1+ qui provoque l'affichage de deux séries de 1 à 8 (au lieu de 1 à 16).

```
: DECUR
DUP + 109 - 2
/MOD ABS 2 - 0<
IF
DUP POS + ROT OVER
C@ + ROT 12 *
SWAP OVER 15 + MIN
MAX RESTOR OVER C! CURSEUR
ELSE
DROP
THEN
;
: COOR
POS C@ POS 1+ C@
AT
;
: EFFACE, RESTOR et IMPRESSION sont inchangés
ou encore
: DECUR
DUP + 109 - 2
/MOD ABS SWAP OVER 2 -
0 < * OVER POS +
SWAP OVER C@ + ROT
12 * SWAP OVER 15
+ MIN MAX RESTOR OVER C! CURSEUR
;
```

Fig. 2. – La gestion du curseur peut être ramenée à cinq mots seulement. Deux définitions de DECUR sont fournies ici : nous laissons à votre sagacité de déterminer en quoi elles diffèrent et quels sont leurs avantages respectifs.

Un test limitera la réaction du curseur à ces quatre touches (quotient < 2). Il reste à vérifier que le curseur ne sort pas de la grille ; on utilise deux mots de Forth, MAX et MIN, qui prennent deux nombres sur la pile pour y restituer, après comparaison, le plus fort (MAX) ou le plus faible (MIN)*. La figure 2 nous fournit les mots exploitant la variable POS.

La saisie des sprites

Sept mots permettent cette saisie : DEFINITION, ↑, GR, suivis de CAR1 CAR2 CAR3 CAR4. A deux paramètres près, ces quatre derniers ont un solide air de famille ; en effet, après avoir adressé le bord supérieur gauche de l'un des quatre carrés, le mot va lire huit lignes de huit caractères. Il suffit donc à CAR1, 2, 3 ou 4 de donner l'adresse de la case départ, et un mot unique fera le reste. Le mode de calcul de la « description » d'une ligne permet de se passer de ↑. Enfin, l'expérience montre que vos morceaux de lutins viennent progressivement remplacer les caractères usuels lus par ENTRÉE et rendent sous peu vos textes illisibles : un 32 MOD inséré après ENTRÉE limite les lutins aux caractères graphiques.

(*) Ces mêmes mots peuvent être appliqués à PISTE de la course auto du N° 37.
: PISTE
BORD @ 3 RND 1-
+ 24 MIN 1 MAX
BORD !

ques, ce qui permet quand même de mettre en place sept lutins, le caractère 13 qui provoque un retour à la ligne devant être évité (il correspond aux touches GRAPHIC + 3, M et m), de même que le caractère nul.

Il n'y a, par ailleurs, aucun scrupule à avoir concernant le fait de mettre en CAR1 à 4 l'adresse de la première case de Jupiter, la portabilité sur une autre machine n'étant de toute façon pas assurée : AT n'est pas du Forth standard et INKEY opère différemment du KEY de Forth.

La figure 3 nous indique les nouvelles versions des mots de saisie.

Enfin, la pratique montre que l'on a de la peine à ne pas avoir une fraction de lutin venant remplacer la précédente si l'on n'est pas suffisamment prompt à relâcher la touche de « code ». La généralisation de ATTEND avant ENTREE permet de l'inclure dans ce mot, et donc de le faire disparaître du

lexique (nous obtenons pratiquement le KEY du Forth standard) (fig. 4).

La figure 5 fournit le listing du nouvel éditeur de lutins.

Un point d'usage général : il est souvent nécessaire d'introduire 0 (début de boucle, par exemple). De ce fait, il est intéressant de créer une constante 0 par : 0 CONSTANT 0 qui occupe 10 octets dans le dictionnaire mais vous en épargne deux à chaque fois que vous voulez mettre 0 sur la pile.

Signalons enfin que, si le besoin s'en fait sentir, on peut encore « glaner » quelques octets en remplaçant les valeurs inférieures à 256 par ASCII « caractère » puisque cette méthode n'occupe que 3 octets au lieu de 4 ; ainsi « ASCII m » peut remplacer « 109 », etc.

Pour les valeurs inférieures à 32, il est nécessaire de passer en mode graphique : les codes ASCII des touches sont alors pris « modulo 32 » (A ou a donnent 1, H ou h donnent 8...) ; il

n'est toutefois pas possible d'obtenir 0 par ce moyen. Il peut aussi être utile de penser aux économies réalisables sur les noms des mots.

FORTH... (les mots indiqués en gras sont inchangés par rapport à ce qui a été fourni dans Micro-Systèmes n° 38).

J'espère que ces quelques réflexions faites sous vos yeux vous donneront des idées pour optimiser vos propres programmes. Le bénéfice direct en est un éditeur de lutins qui n'occupe plus que 816 octets (avec la constante 0 et premier DECUR, 808 avec le deuxième DECUR) – tenant donc dans la mémoire de base de JUPITER – au lieu des 1 288 initiaux ! ■

Conclusion

Pour contrôler votre travail, VLIST doit donner :

**EDITEUR DEFINITION
ENTREE CAR4 CAR3 CAR2
CAR1 CAR GR DECUR
RESTOR IMPRESSION
CURSEUR EFFACE COOR
GRILLE POS STOCK 0**

```
: ENTREE
BEGIN
INKEY 0=
UNTIL
BEGIN
INKEY ? DUP
UNTIL
;
```

Fig. 4. – Un mot ENTREE, correspondant quasiment au mot KEY du Forth standard.

```
: EDITEUR
CLS GRILLE
BEGIN
ENTREE DUP ASCII q = (1)
IF
ABORD
ELSE
DUP ASCII e = (1)
IF
EFFACE
ELSE
DUP ASCII d = (1)
IF
DEFINITION
ELSE
DUP 13 =
IF
IMPRESSION
ELSE
DECUR
THEN
THEN
THEN
0=
UNTIL
;
```

Fig. 5. – Le nouvel éditeur, utilisant les nouveaux mots, occupe seulement 816 octets de mémoire vive.

```
: CAR
DUP 255 + SWAP
DO
0 I 8 + I
DO
DUP + I C@ ASCII ■
= +
LOOP
-32
+LOOP
ENTREE 32 MOD GR
;

: DEFINITION
0 0 AT ." Code 1" CAR1
0 0 AT ." Code 2" CAR2
0 0 AT ." Code 3" CAR3
0 0 AT ." Code 4" CAR4
;

: CAR1
9228 CAR
;
;
: CAR2
9236 CAR
;
;
: CAR3
9492 CAR
;
;
: CAR4
9484 CAR
;

GR est inchangé (en remplaçant toutefois le premier + par *)
```

Fig. 3. – Les mots de saisie d'un sprite peuvent être considérablement compactés par une mise en facteur des actions de quatre d'entre eux.

J.A.O.

Jeu Assisté par Ordinateur

Capturer les hors-la-loi ou au contraire échapper à la poursuite acharnée d'un représentant de l'ordre a toujours été votre rêve ? Il est maintenant réalisable si vous possédez un ZX 81 muni de l'extension 16 Ko. Mais patience ! Il vous faut d'abord entrer dans votre ordinateur favori le programme qui suit (environ 8 Ko).

Le principe de ce jeu est assez simple. Sur un échiquier infini (**fig. 1**), deux personnages sont représentés : un gendarme et un voleur. Pour gagner, le gendarme doit se placer sur une des huit cases contiguës à celle du voleur ou sur cette dernière. Le voleur doit, quant à lui, résister un nombre de coups fixé au début du jeu.

Mais avant de gagner, il faut jouer ! A chaque tour, le voleur, représenté par un V en vidéo inversée, peut avancer d'une case dans toutes les directions. Pour le gendarme, c'est un peu plus compliqué ! En effet, ce dernier est « orienté ». Il est alors représenté par l'initiale en vidéo inversée de son orientation géographique (N pour nord, E pour est, S pour sud et O pour ouest). Quand c'est à son tour de jouer, il a le choix entre avancer de deux cases dans sa direction ou tourner à sa droite puis avancer de deux cases (son orientation variera alors d'un quart de tour vers l'Est).

Maintenant que vous connaissez les règles, vous voulez sans doute savoir comment utiliser le programme. Après l'avoir chargé, lancez son exécution par « RUN ». Le ZX 81 vous demandera si vous souhaitez consulter les règles (**fig. 2**). Après les avoir lues (ou non), vous devrez choisir le personnage que vous désirez « incarner ». Il vous faudra ensuite décliner du niveau de jeu (de « 1 » le plus simple à « 10 » le plus complexe). C'est seulement maintenant que le jeu proprement dit démarre ! Le ZX 81, en bon gentleman, vous laissera toujours commencer. Pour indi-

quer vos coups, il suffit de mentionner les coordonnées de la case d'arrivée (abscisse puis ordonnée). Votre compagnon favori, en plus de devenir votre adversaire, gère aussi totalement l'écran ! Mais nous vous laissons la surprise.

Dernière remarque : au début notamment, il est souvent nécessaire de presser une touche quelconque pour poursuivre l'exécution du programme.

Le fonctionnement du programme

Si vous souhaitez laisser tout son mystère à ce jeu, ne lisez pas ces lignes, car elles révèlent son secret. Tout réside dans le tableau A\$! La case A\$ (10,5) représente toujours la position du gendarme orienté vers le nord. (G.H.) sont les coordonnées du voleur dans ce même tableau. A chaque mouvement du voleur, ces dernières sont modifiées en conséquence. Quant aux mouvements du gendarme, ils sont « traduits » en coups voleur. Ainsi, pour le programme, seul le voleur bouge ! Les valeurs du tableau (jusqu'à B inclus) indiquent en hexadécimal le nombre de coups qu'il faudra au gendarme, si c'est à lui de jouer, pour rattraper le voleur.

Si le voleur est sur une case C ou hors du tableau, il est hors d'atteinte. Les techniques sont alors simples : pour le gendarme, il faut ramener le voleur sur les cases aux numéros les plus faibles, et pour le voleur, il faut aller sur les cases aux numéros les plus élevés ou hors du tableau. ■

JEU

Un gendarme et un voleur
de J.-C. RIAT

Fuyez ou poursuivez, telles sont les deux règles de ce jeu ancien, dont nous avons tous été, tôt ou tard, des adeptes.

L'informatique le remet ici à la mode.

LANGAGE : Basic

ORDINATEUR : ZX 81
+ extension 16 Ko.

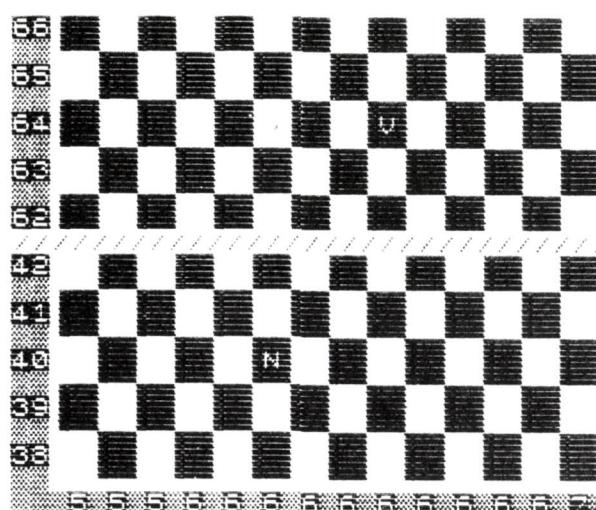


Fig. 1. – Les deux protagonistes se déplacent sur un échiquier infini.

GENDARME ET VOLEUR

REGLES

SUR UN DAMIER INFINI UN GENDARME POURSUIT UN MECHANT BANDIT.

LE BUT DU JEU EST :

POUR LE GENDARME D'ATTRAPER LE VOLEUR EN SE PLACANT SUR UNE DES 8 CASES VOISINES.

POUR LE BANDIT D'ÉCHAPPER À SON POURSUVEUR EN RÉSISTANT UN NOMBRE FIXE DE COUPS.

MAIS VOILA : A CHAQUE TOUR LE BANDIT N'AVANCE QUE D'UNE CASE DANS TOUTES LES DIRECTIONS ET LE GENDARME DE DEUX MAIS SEULEMENT DANS SA DIRECTION, INDICÉE PAR L'INITIALE DES POINTS CARDINAUX, OU EN TOURNANT À SA DROITE.

Fig. 2. – Au début de la partie le joueur peut afficher les règles du jeu.

Liste des variables utilisées

A : variable de boucle et utilisation temporaire.
B : variable de boucle et utilisation temporaire.
C : abscisse du voleur sur l'échiquier.
D : ordonnée du voleur sur l'échiquier.
E : abscisse du gendarme sur l'échiquier.
F : ordonnée du gendarme sur l'échiquier.
G : ordonnée du voleur dans le tableau A\$ ayant comme centre de repère la position du gendarme.
H : abscisse du voleur dans le tableau A\$ ayant comme centre de repère la position du gendarme.
I : orientation du gendarme : 1 pour nord, 2 pour ouest, 3 pour sud, 4 pour est.
J : position de « pokage » du gendarme.
K : position de « pokage » du voleur.
L : variable d'adressage indirect.
M : utilisation temporaire pour trouver le coup du voleur joué par le ZX 81.
N : utilisation temporaire pour trouver le coup du gendarme joué par le ZX 81.
O : nombre de coups restant à jouer pour le voleur.
S : utilisation temporaire.
T : utilisation temporaire.
U : utilisation temporaire.
V : utilisée pour l'affichage des chiffres verticaux.
W : utilisée pour l'affichage des chiffres verticaux.
X : utilisée pour l'affichage des chiffres horizontaux.
Y : utilisée pour l'affichage des chiffres horizontaux.
Z : adresse du début de la zone d'affichage.
A\$: tableau principal.
B\$: contient les réponses aux questions ainsi que les coups joués.

Décomposition du programme

100 à 280 : sous-programme regardant si le gendarme se trouve sur la case du voleur ou sur une des huit contiguës et le cas échéant arrête le jeu.

300 à 400 : sous-programme d'introduction du coup.

Remarque : les deux POKEs permettent d'utiliser les deux lignes libres du bas de l'écran.

1000 à 1110 : dessin de l'échiquier.

1000 à 1810 : sous-programme de gestion de l'affichage.

1120 à 1280 : affichage des chiffres verticaux avec éventuellement la bande blanche hachurée horizontale.

1300 à 1570 : affichage des chiffres horizontaux avec éventuellement la bande blanche hachurée verticale.

1600 à 1740 : effacement et affichage des pièces.

1800 à 1810 : sous-programme calculant la nouvelle position de « pokage » pour les pièces.

2000 à 2220 : introduction du coup du gendarme avec vérification de sa validité et modification des coordonnées du voleur pour le tableau.

2230 à 2270 : modification de l'affichage des pièces.

3000 à 3290 : le ZX 81 joue le voleur.

3020 : orientation selon que le voleur est dans le tableau A\$ ou pas.

3030 à 3060 : le ZX 81 joue au hasard car tous les coups sont gagnants.

3100 à 3210 : le ZX 81 regarde tous les coups possibles et choisit le meilleur.

3250 à 3290 : modification des variables.

3300 à 3340 : modification de l'affichage des pièces.

4000 à 4090 : introduction du coup du voleur avec vérification de sa validité et modification des coordonnées du voleur pour le tableau.

5000 à 5510 : le ZX 81 joue le gendarme.

5000 à 5120 : regarde les deux coups possibles et choisit de jouer le meilleur ou d'aller en 5500 s'il est perdant.

5130 à 5510 : le ZX 81 joue le coup qu'il a choisi.

7000 à 7360 : initialisation des variables et du tableau.

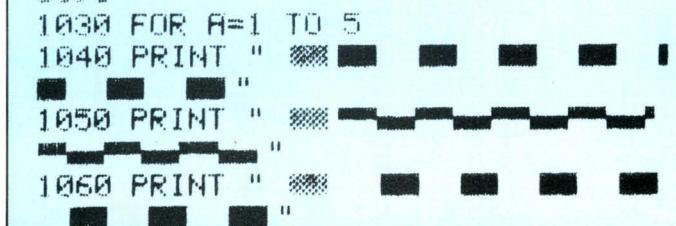
8000 à 8120 : règles du jeu.

8200 à 8320 : le joueur choisit son « personnage » et le niveau de difficulté du jeu.

```

10 REM GENDARME ET VOLEUR
20 REM
30 REM RIAT JEAN-CHRISTOPHE
40 REM
50 REM 29/07/83
60 REM
65 RAND
70 GOTO 8000
100 REM REGARDE SI C'EST FINI
110 FOR A=-1 TO 1
120 FOR B=-1 TO 1
130 IF C=E+A AND D=F+B THEN GOT
O 180
140 NEXT B
150 NEXT A
160 IF D=0 THEN GOTO 220
170 RETURN
180 PRINT AT 21,0
190 POKE 16418,3
200 PRINT "LE GENDARME A GAGNE."
"
210 GOTO 250
220 PRINT AT 21,0
230 POKE 16418,4
240 PRINT "LE VOLEUR A GAGNE."
250 PRINT "UNE REVANCHE ?"
254 POKE 16418,2
255 FOR A=1 TO 75
256 NEXT A
260 INPUT B$
270 IF B$="OUI" THEN RUN
280 STOP
300 REM DEMANDE DU COUP
310 PRINT AT 21,0
320 POKE 16418,4
330 PRINT
340 PRINT "VOTRE COUP , SVP ?"
345 POKE 16418,2
350 FOR A=1 TO 75
355 NEXT A
360 INPUT B$
370 IF LEN B$<4 THEN GOTO 310
380 LET A=VAL B$(0 TO 2)
390 LET B=VAL B$(3 TO 5)
400 RETURN
1000 REM AFFICHAGE
1010 CLS
1020 LET Z=PEEK 16397*256+PEEK 1
6396
1030 FOR A=1 TO 5
1040 PRINT " ■■■■■ "
1050 PRINT " ■■■■■ "
1060 PRINT " ■■■■■ "

```



Listing du programme. Édité sur une imprimante « Seikosha 250 ». Ne comportant pas de caractère inversé, il faut reconnaître ceux-ci : ils sont imprimés sous la forme de minuscules.

```

1070 IF A=5 THEN GOTO 1100
1080 PRINT " "
1090 NEXT A
1100 PRINT " - - - "
1110 PRINT " "
1120 IF ABS (D-F)>6 THEN GOTO 11
1130 LET W=INT ((D+F)/2)
1140 LET V=W+5
1150 GOTO 1210
1160 LET R=D
1170 IF F>R THEN LET R=F
1180 LET V=R+2
1190 LET W=V-ABS (D-F)
1200 PRINT AT 9,0;" //////////////"
1210 FOR R=0 TO 264 STEP 66
1220 POKE Z+2+R, INT (V/10)+156
1230 POKE Z+332+R, INT (W/10)+156
1240 POKE Z+3+R, V-INT (V/10)*10+
156
1250 POKE Z+333+R, W-INT (W/10)*1
0+156
1260 LET V=V-1
1270 LET W=W-1
1280 NEXT A
1300 IF ABS (C-E)>10 THEN GOTO 1
400
1310 LET X=INT ((C+E)/2)+1
1320 LET Y=X-7
1330 GOTO 1500
1400 LET R=C
1410 IF E>R THEN LET R=E
1420 LET X=R-3
1430 LET Y=X-ABS (C-E)
1440 FOR R=0 TO 21
1450 POKE Z+18+R*33,24
1460 NEXT A
1500 FOR R=0 TO 12 STEP 2
1510 POKE Z+665+R, INT (Y/10)+156
1520 POKE Z+698+R, Y-INT (Y/10)*1
0+156
1530 POKE Z+679+R, INT (X/10)+156
1540 POKE Z+712+R, X-INT (X/10)*1
0+156
1550 LET X=X+1
1560 LET Y=Y+1
1570 NEXT A
1600 REM DESSIN DES PIECES
1603 IF L=2000 OR L=5000 THEN PO
KE J, PEEK (J-1)-5
1606 IF L=3000 OR L=4000 THEN PO
KE K, PEEK (K-1)-5
1610 LET R=D

```

```

1620 LET B=C
1630 GOSUB 1800
1640 LET J=R
1650 LET R=F
1660 LET B=E
1670 GOSUB 1800
1680 LET K=R
1690 LET R=170+(I=1)*9+(I=2)*10+
(I=3)*14
1710 POKE J,187
1730 POKE K,R
1740 RETURN
1800 LET R=Z+1+((V>R)*(W+10)+(V<
=R)*(V+5)-R)*66+(B-(Y>B)*(Y-9)-(Y<
=B)*(X-16))*2
1810 RETURN
2000 REM INTRODUCTION DU COUP G
2010 GOSUB 100
2020 GOSUB 300
2050 IF R=E+2*(I-3)*(I<>1) AND B
=F+2*(2-I)*(I<>4) THEN GOTO 2100
2060 IF R=E+2*(2-I)*(I<>4) AND B
=F+2*(3-I)*(I<>1) THEN GOTO 2150
2070 GOTO 2020
2100 LET G=G-2
2110 GOTO 2200
2150 LET I=I-1+(I=1)*4
2160 LET T=G
2170 LET G=H-2
2180 LET H=-T
2200 LET E=A
2210 LET F=B
2220 LET L=3000
2230 IF F>V+5 OR F<=W OR (F>W+5
AND F<=V) OR E>Y-7 OR E>=X OR (E
>=Y AND E<X-7) THEN GOTO 2260
2240 GOSUB 1600
2250 GOTO L
2260 GOSUB 1000
2270 GOTO L
3000 REM ZX81 JOUE VOLEUR
3010 GOSUB 100
3015 LET D=0-1
3020 IF G<9 AND G>-7 AND H>-4 AN
D H<9 THEN GOTO 3100
3030 LET T=1-INT (RND*3)
3040 LET U=1-INT (RND*3)
3050 IF ABS (T+U)<>1 THEN GOTO 3
030
3060 GOTO 3250
3100 LET M=0
3110 FOR R=-1 TO 1
3120 FOR B=-1 TO 1
3130 IF ABS (R+B)<>1 THEN GOTO 3
200
3140 LET S=CODE R$(10-G-R,H+5+B)
3150 IF S=M THEN GOTO 3200

```

3160 IF S=M AND RND*4<3 THEN GOTO
 0 3200
 3170 LET M=S
 3180 LET T=A
 3190 LET U=B
 3200 NEXT B
 3210 NEXT A
 3250 LET G=G+T
 3260 LET H=H+U
 3270 LET D=D+(2-I)*(I<>4)*T+(3-I)
 *(I<>1)*U
 3280 LET C=C+(2-I)*(I<>4)*U+(I-3)
 *(I<>1)*T
 3290 LET L=2000
 3300 IF D>V+5 OR D<=W OR (D>W+5
 AND D<=V) OR (D>Y-7 OR C>X OR (C
 >=Y AND C<X-7)) THEN GOTO 3330
 3310 GOSUB 1600
 3320 GOTO L
 3330 GOSUB 1000
 3340 GOTO L
 4000 REM INTRODUCTION DU COUP V
 4010 GOSUB 100
 4020 GOSUB 300
 4025 LET O=0-1
 4030 IF (RBS*(B-D)<>1 OR A<>C) A
 ND (RBS*(A-C)<>1 OR B<>D) THEN G
 OTO 4020
 4040 LET G=G+(2-I)*(I<>4)*(B-D)+
 (I-3)*(I<>1)*(A-C)
 4050 LET H=H+(2-I)*(I<>4)*(B-C)+
 (3-I)*(I<>1)*(B-D)
 4060 LET C=A
 4070 LET D=B
 4080 LET L=5000
 4090 GOTO 3300
 5000 REM ZX81 JOUE LE GENDARME
 5010 GOSUB 100
 5015 LET N=0
 5020 LET T=41
 5025 LET U=41
 5030 LET S=G-2
 5040 IF S>7 OR S<-5 OR H<-2 OR H
 >7 THEN GOTO 5060
 5050 LET T=CODE R\$(10-S,H+5)
 5060 LET R=H-2
 5070 LET B=-G
 5080 IF R>7 OR R<-5 OR B<-2 OR B
 >7 THEN GOTO 5100
 5090 LET U=CODE R\$(10-R,B+5)
 5100 IF T*U>1599 THEN GOTO 5500
 5110 IF T>U THEN GOTO 5200
 5120 IF T=U AND RND*.5 THEN GOTO
 5200
 5130 LET N=N+1
 5135 IF N>3 AND UK>28 AND T<>28
 AND (R\$(11-S,H+5)="C" OR R\$(9-S,

H+5)="C" OR R\$(10-S,H+6)="C" OR
 R\$(10-S,H+4)="C") THEN GOTO 5200
 5140 LET G=S
 5150 LET F=F+2*(2-I)*(I<>4)
 5160 LET E=E+2*(I-3)*(I<>1)
 5170 GOTO 5400
 5200 LET N=N+1
 5205 IF N>3 AND T<40 AND UK>28
 AND (R\$(11-R,B+5)="C" OR R\$(9-R,
 B+5)="C" OR R\$(10-R,B+6)="C" OR
 R\$(10-R,B+4)="C") THEN GOTO 5130
 5210 LET G=R
 5220 LET H=B
 5230 LET F=F+2*(3-I)*(I<>1)
 5240 LET E=E+2*(2-I)*(I<>4)
 5250 LET I=I-1+(I=1)*4
 5400 LET L=4000
 5410 GOTO 2230
 5500 IF G>1 THEN GOTO 5140
 5510 GOTO 5210
 7000 REM VARIABLES
 7010 DIM R\$(17,14)
 7020 LET R\$(1)="CCCCCCCCCCCCCCCC"
 7030 LET R\$(2)="CCCCCCCCCCCCCCCC"
 7040 LET R\$(3)="CCCC008B00000000"
 7050 LET R\$(4)="CCCC005800000000"
 7060 LET R\$(5)="CCCC347800000000"
 7070 LET R\$(6)="CCCC234700000000"
 7080 LET R\$(7)="CCC111369000000"
 7090 LET R\$(8)="CCC111236000000"
 7100 LET R\$(9)="CCC000115800000"
 7110 LET R\$(10)="CCC000112300000"
 7120 LET R\$(11)="CC900011345800"
 7130 LET R\$(12)="CCC74323478B00"
 7140 LET R\$(13)="CCC074367800000"
 7150 LET R\$(14)="CCCC856900000000"
 7160 LET R\$(15)="CCCCB80000000000"
 7170 LET R\$(16)="CCCCCCCCCCCCCCCC"
 7180 LET R\$(17)="CCCCCCCCCCCCCCCC"
 7200 LET C=40+INT (RND*2*RBS (17
 *(T-1)-N-3))
 7210 LET D=40+INT (RND*2*RBS (17
 *(T-1)-N-3))
 7220 LET E=40+INT (RND*2*RBS (17
 *(T-1)-N-3))
 7230 LET F=40+INT (RND*2*RBS (17
 *(T-1)-N-3))
 7240 LET I=INT (RND*4)+1
 7250 LET G=D-F
 7260 LET H=C-E
 7270 FOR R=1 TO I-1
 7280 LET B=G
 7290 LET G=-H
 7300 LET H=B
 7310 NEXT R
 7340 LET L=0
 7350 GOSUB 1000

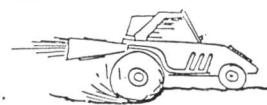
```

7360 GOTO T*2000
8000 REM REGLES
8010 CLS
8020 PRINT TAB 7;"GENDARME ET VO
LEUR"
8030 PRINT TAB 7;"*****"
8040 PRINT AT 5,0;"VOULEZ-VOUS L
ES REGLES ?"
8050 INPUT B$
8060 IF B$="NON" THEN GOTO 8200
8065 PRINT AT 2,0;"REGLES:";,"***"
*****"
8070 PRINT "SUR UN DAMIER INFINI
UN GENDARME POURSUIT UN MECHANT
BANDIT."
8080 PRINT ",,"LE BUT DU JEU EST:
",," POUR LE GENDARME D""ATTRAPE
R LE VOLEUR EN SE PLACANT SUR UNE
DES 8 CASES VOISINES."
8090 PRINT " POUR LE BANDIT D""E
CHAPPER A SON POURSUITEUR EN
RESISTANT UN NOMBRE FIXE DE COUP
S."
8100 PRINT ",,"MAIS VOILA: A CHAQUE
TOUR LE BANDIT N""AVANCE A
UE D""UNE CASE"
8110 PRINT "DANS TOUTES LES DIRE
CTIONS ET LEGENDARME DE DEUX MAI
S SEULEMENT DANS SA DIRECTION. I
NDIQUEE PAR L""INITIALE DES POIN
TS CARDINAUX, OU EN TOURNANT A SA
DROITE."
8120 IF INKEY$="" THEN GOTO 8120
8200 CLS
8210 PRINT "PREFEREZ-VOUS JOUER
L""ADORABLE GENDARME (1) OU L""I
GNOBLE BANDIT(2) ?"
8220 PRINT "LE ZX81 SERA DANS LE
S DEUX CAS VOTRE ADVERSAIRE."
8230 INPUT T
8240 IF T<>1 AND T>2 THEN GOTO
8230
8250 PRINT ",,"QUEL NIVEAU ?(1 A
10)"
8260 INPUT N
8270 IF N<1 OR N>10 THEN GOTO 82
60
8280 LET D=6+ABS (5*(11*ABS (T-2
)-N))
8290 PRINT ",,"LE VOLEUR DOIT RES
ISTER ";O;" COUPS"
8300 PRINT ",,"DONNER VOS COUPS E
N N""INDIQUANT QUE LES COORDONNE
ES DE LA CASE D""ARRIVEE."
8310 IF INKEY$="" THEN GOTO 8310
8320 GOTO 7000

```

**Si votre APPLE fait BIP quand vous l'allumez
il a forcément l'une des vocations ci-dessous.**

L'Apple Turbo



Rend vos applications plus rapides.

- * Carte Accélérateur
- * Carte Microbuffer Imprimante
- * Carte Mémoire 128/256 K Legend

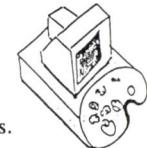
L'Apple Vision



Vous fait voir vos pages de texte en 48 lignes.
Vous fait voir vos calculs en 132 colonnes.

- * Carte Ultraterm de Videx.

L'Apple Palette

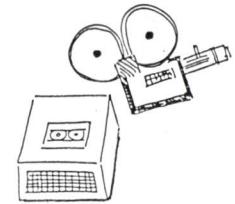


Dessinez et redessinez sur l'ardoise magique.
Coloriez et changez des détails.

- Sauvez et imprimez.
- Facile, rapide et agréable.

* Tablette Graphique Koalapad et ses programmes.
Sur Apple, Commodore, IBM PC.

L'Apple Ciné



il enregistre vos images ou les anime,
au gré de votre fantaisie et de vos besoins.

- * Carte à Digitaliser pour capter les images ;
- * Programme TGS pour les animier.

L'Apple Musical



Composez puis écoutez vos mélodies sur votre chaîne.

- * Programme Music Construction Set pour créer ;
- * Mockingboard (6 canaux synthétiseurs) pour l'audition ou jeux sur votre APPLE.

L'Apple Pro



Il met à jour vos prix de revient,
tient vos statistiques, réalise en temps réel tous vos calculs:

- * MAGICALC, tableur en FRANÇAIS.
- Nouveau !

* THE BRIDGE, fait le Pont entre PFS et vos Cales ou Traitements de texte.

Ces produits sont en vente chez les meilleurs revendeurs.

venez les voir au MICRO-EXPO 84 stand P15

Demande de documentation gratuite :

Nom Prénom

Adresse

Code postal Ville

TURBO VISION CINE
 PALETTE MUSICAL PRO



l'informatique personnalisée

Direction Commerciale pour la France
13, rue Duc - 75018 PARIS
Tél. (1) 255.44.63

SERVICE-LECTEURS N° 165