VIRUSDOG

TEAM : WWW.CPC-POWER.COM

3 versions :

- Virusdog (UK) (2016) (contest version) [Original] [TAPE].cdt
  Tape version for the competition

- Virusdog (UK) (K7) (2016) (contest version) [Original].dsk
  Tape version transferred on floppy disc.
  Contain a special display for the catalog (CATART)

- Virusdog (UK) (D7) (2016) (GIFT - extended version) [Original].dsk
  This version is not for the competition, it's a gift to permit to save and load progression in the game using CTRL+S (Save score) and CTRL+L (Load score) on the main menu of the game. Just because it is very long to finish all 50 levels with in "perfect" to discover the little end picture.

Files for all versions :

DISC.BAS	7 Ko
INTRO.BIN	16 Ko
INGAME.BAS	12 Ko
INGAME.PIC	10 Ko
INGAME.PAK	16 Ko

SUMMARY

Disc version only, includes a CATART using MODE 2 :

A) First step, creating the idea in text file :

```
column   : 16   21   26   31   36   41   46   51   56   61
           12345123451234512345123451234512345123451234512345
line 1   : ___     __                                  _
line 2   : \  \  _|__|_____  __ __ _____ __| | ___   ____
line 3   :  \  \/ /  \_  __ \|  |  Y  ___// __ |/ _ \ / __ \
line 4   :   \   /|  ||  | \/|  |  |___ \/ /_/ ( (_) ) /_/ /
line 5   :    \_/ |__||__|   |____/_____/\_____|\___/\__  /
line 6   :                                              /___/
line RUN :                    RUN"DISC.BAS"
```

B) Second step, try in BASIC

```
1 'FAT - 52 entries
10 MODE 2
20 BORDER 14
30 INK 0,14
40 INK 1,0
50 PEN 1
99 'line 1
100 LOCATE 16,9:PRINT "___   "
110 LOCATE 21,9:PRINT "    __"
120 LOCATE 51,9:PRINT "_     "
199 'line 2
200 LOCATE 16,10:PRINT "\  \ "
210 LOCATE 21,10:PRINT " _|__"
220 LOCATE 26,10:PRINT "|_____"
230 LOCATE 31,10:PRINT "__  _"
240 LOCATE 36,10:PRINT "_   "
250 LOCATE 41,10:PRINT "_____"
260 LOCATE 46,10:PRINT "_ __|"
270 LOCATE 51,10:PRINT " | __"
280 LOCATE 56,10:PRINT "_  _"
290 LOCATE 61,10:PRINT "___  "
299 'line 3
300 LOCATE 16,11:PRINT " \  \"
310 LOCATE 21,11:PRINT "/ /  "
320 LOCATE 26,11:PRINT "\_  _"
330 LOCATE 31,11:PRINT "_ \| "
340 LOCATE 36,11:PRINT " |  Y"
350 LOCATE 41,11:PRINT "  ___"
360 LOCATE 46,11:PRINT "//  _"
370 LOCATE 51,11:PRINT " |/ _"
380 LOCATE 56,11:PRINT " \ / "
390 LOCATE 61,11:PRINT "__ \ "
399 'line 4
400 LOCATE 16,12:PRINT "  \  "
410 LOCATE 21,12:PRINT " /|  "
420 LOCATE 26,12:PRINT "||  |"
430 LOCATE 31,12:PRINT " \/| "
440 LOCATE 36,12:PRINT " |  |"
450 LOCATE 41,12:PRINT "___ \"
460 LOCATE 46,12:PRINT "/ /_/"
470 LOCATE 51,12:PRINT " ( (_"
480 LOCATE 56,12:PRINT ") ) /"
490 LOCATE 61,12:PRINT "_/ / "
499 'line 5
500 LOCATE 16,13:PRINT "   \_"
```

```
510 LOCATE 21,13:PRINT "/ |__"
520 LOCATE 26,13:PRINT "||__|"
530 LOCATE 31,13:PRINT "   |_"
540 LOCATE 36,13:PRINT "___/_"
550 LOCATE 41,13:PRINT "____/"
560 LOCATE 46,13:PRINT "\____"
570 LOCATE 51,13:PRINT "_|\__"
580 LOCATE 56,13:PRINT "_/\__"
590 LOCATE 61,13:PRINT "  /  "
599 'line 6
600 LOCATE 58,14:PRINT "/___/"
699 'RUN
700 LOCATE 34,17:PRINT "RUN";CHR$(34);"D"
710 LOCATE 39,17:PRINT "ISC.B"
720 LOCATE 44,17:PRINT "AS";CHR$(34)
```

C) Third step, encoding on the FAT (Sectors &C1, &C2, &C3 and &C4)

```
CATART using 52 / 64 entries
Rest only 12 entries free
```

```
//TRACK 00 ; SECTOR &C1 (16 entries used)
000000: 00 04 02 15 00 00 00 00 00 00 00 00 00 00 00 00  ................
000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000020: 00 06 1D 0E 0E 15 00 00 00 00 00 00 00 00 00 00  ................
000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000040: 00 06 1C 00 0E 0E 15 00 00 00 00 00 00 00 00 00  ................
000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000060: 00 06 1C 01 00 00 15 00 00 00 00 00 00 00 00 00  ................
000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000080: 00 06 0F 01 15 00 00 00 00 00 00 00 00 00 00 00  ................
000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000A0: 00 06 1F 10 09 5F 5F 5F 20 08 20 15 00 00 00 00  ....._____ . .....
0000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000C0: 00 06 1F 15 09 20 20 20 5F 08 5F 15 00 00 00 00  .....   _._.....
0000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000E0: 00 06 1F 33 09 5F 20 20 20 08 20 15 00 00 00 00  ...3._   . .....
0000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000100: 00 06 1F 10 0A 5C 20 20 5C 08 20 15 00 00 00 00  .....\  \. .....
000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000120: 00 06 1F 15 0A 20 5F 7C 5F 08 5F 15 00 00 00 00  ..... _|_._._....
000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000140: 00 06 1F 1A 0A 7C 5F 5F 5F 08 5F 15 00 00 00 00  .....|___._._....
000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000160: 00 06 1F 1F 0A 5F 5F 20 20 08 5F 15 00 00 00 00  .....__  ._.....
000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000180: 00 06 1F 24 0A 5F 20 5F 5F 08 20 15 00 00 00 00  ...$._ __. .....
000190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001A0: 00 06 1F 29 0A 5F 5F 5F 5F 08 5F 15 00 00 00 00  ...)._____._.....
0001B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001C0: 00 06 1F 2E 0A 5F 20 5F 5F 08 7C 15 00 00 00 00  ....._ __.|.....
0001D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001E0: 00 06 1F 33 0A 20 7C 20 5F 08 5F 15 00 00 00 00  ...3. | _._.....
0001F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................

//TRACK 00 ; SECTOR &C2 (16 entries used)
000000: 00 06 1F 38 0A 5F 20 20 20 08 5F 15 00 00 00 00  ...8._   ._.....
000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000020: 00 06 1F 3D 0A 5F 5F 5F 20 08 20 15 00 00 00 00  ...=.___ . .....
000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000040: 00 06 1F 10 0B 20 5C 20 20 08 5C 15 00 00 00 00  ..... \  .\.....
000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000060: 00 06 1F 15 0B 2F 20 2F 20 08 20 15 00 00 00 00  ...../ / . .....
000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000080: 00 06 1F 1A 0B 5C 5F 20 20 08 5F 15 00 00 00 00  .....\_  ._.....
000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
0000A0: 00 06 1F 1F 0B 5F 20 5C 7C 08 20 15 00 00 00 00  ....._ \|. .....
0000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000C0: 00 06 1F 24 0B 20 7C 20 20 08 59 15 00 00 00 00  ...$. |  .Y.....
0000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000E0: 00 06 1F 29 0B 20 20 5F 5F 08 5F 15 00 00 00 00  ...).  __._.....
0000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000100: 00 06 1F 2E 0B 2F 2F 20 5F 08 5F 15 00 00 00 00  ...../// _._.....
000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000120: 00 06 1F 33 0B 20 7C 2F 20 08 5F 15 00 00 00 00  ...3. |/ ._.....
000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000140: 00 06 1F 38 0B 20 5C 20 2F 08 20 15 00 00 00 00  ...8. \ /. .....
000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000160: 00 06 1F 3D 0B 5F 5F 20 5C 08 20 15 00 00 00 00  ...=.__ \. .....
000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000180: 00 06 1F 10 0C 20 20 5C 20 08 20 15 00 00 00 00  .....  \ . .....
000190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001A0: 00 06 1F 15 0C 20 2F 7C 20 08 20 15 00 00 00 00  ..... /| . .....
0001B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001C0: 00 06 1F 1A 0C 7C 7C 20 20 08 7C 15 00 00 00 00  .....||  .|.....
0001D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001E0: 00 06 1F 1F 0C 20 5C 2F 7C 08 20 15 00 00 00 00  ..... \/|. .....
0001F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

//TRACK 00 ; SECTOR &C3 (16 entries used)

```
000000: 00 06 1F 24 0C 20 7C 20 20 08 7C 15 00 00 00 00  ...$. |  .|.....
000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000020: 00 06 1F 29 0C 5F 5F 5F 20 08 5C 15 00 00 00 00  ...)___ .\.....
000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000040: 00 06 1F 2E 0C 2F 20 2F 5F 08 2F 15 00 00 00 00  ...../ /_./.....
000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000060: 00 06 1F 33 0C 20 28 20 3C 08 5F 15 00 00 00 00  ...3. ( <._.....
000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000080: 00 06 1F 38 0C 3E 20 29 20 08 2F 15 00 00 00 00  ...8.> ) ./.....
000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000A0: 00 06 1F 3D 0C 5F 2F 20 2F 08 20 15 00 00 00 00  ...=._/ /. .....
0000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000C0: 00 06 1F 10 0D 20 20 20 5C 08 5F 15 00 00 00 00  .....   \._.....
0000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0000E0: 00 06 1F 15 0D 2F 20 7C 5F 08 5F 15 00 00 00 00  ...../ |_._.....
0000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000100: 00 06 1F 1A 0D 7C 7C 5F 5F 08 7C 15 00 00 00 00  .....||__.|.....
000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000120: 00 06 1F 1F 0D 20 20 20 7C 08 5F 15 00 00 00 00  .....   |._.....
000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000140: 00 06 1F 24 0D 5F 5F 5F 2F 08 5F 15 00 00 00 00  ...$.___/._.....
000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000160: 00 06 1F 29 0D 5F 5F 5F 5F 08 2F 15 00 00 00 00  ...)____./.....
000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000180: 00 06 1F 2E 0D 5C 5F 5F 5F 08 5F 15 00 00 00 00  .....\___._.....
000190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001A0: 00 06 1F 33 0D 5F 7C 5C 5F 08 5F 15 00 00 00 00  ...3._|\_._.....
0001B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001C0: 00 06 1F 38 0D 5F 2F 5C 5F 08 5F 15 00 00 00 00  ...8._/\_._.....
0001D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001E0: 00 06 1F 3D 0D 20 20 2F 20 08 20 15 00 00 00 00  ...=.  / . .....
0001F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

//TRACK 00 ; SECTOR &C4 (4 entries used - 12 entries free)

```
000000: 00 06 1F 3A 0E 2F 5F 5F 5F 08 2F 15 00 00 00 00  ...:./___./.....
000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000020: 00 06 1F 22 11 52 55 4E 22 08 44 15 00 00 00 00  ...".RUN".D.....
000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000040: 00 06 1F 27 11 49 53 43 2E 08 42 15 00 00 00 00  ...'.ISC..B.....
000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000060: 00 06 1F 2C 11 41 53 22 20 08 20 15 00 00 00 00  ...,.AS" . .....
000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
000080: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000090: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000A0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000B0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000C0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000D0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000E0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0000F0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000100: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000110: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000120: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000130: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000140: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000150: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000160: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000170: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000180: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
000190: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001A0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001B0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001C0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001D0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001E0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
0001F0: E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5  ................
```

D) The result if you tap CAT on basic.

All the project uses the MODE 1 screen (4 colors) with Black, White, Blue and Dark Red only.

MENU TO LAUNCH OR SKIP INTRODUCTION

DISC.BAS : One file with BASIC + ASM (7 Ko)

```
1 REM ********************
2 REM *  V I R U S D O G  *
3 REM * By Kukulcan  2016 *
4 REM * www.cpc-power.com *
5 REM ********************
10 CALL &BB4E:POKE &BDEE,&C9:KEY DEF 66,0,0,0,0:CALL &260
20 IF PEEK(&BE80)=2 THEN RUN"!INTRO.BIN
30 RUN"!INGAME.BAS
```

Normal screen

Normal screen + Rasters to custom the number of colors

Code in ASM for the RASTERS (comments in french)

```
tailleraster1     EQU 26
tailleraster2     EQU 20

ORG &77BC

      DI                ; Desactiver les interruptions
      LD HL,(&38)       ; Lire l'ancienne interruption
      LD (Sauver),HL    ; Sauvegarder la valeur 16 bits
      LD HL,&C9FB       ; Registre 16 bits EI (&FB) + RET (&C9)
      LD (&38),HL       ; Ecrire en &0038 et &0039 le contenu du registre HL
      EI                ; Interruption autorisees

.Programme
      ;raster1b rotation vers le bas
      LD A,(Raster1b+tailleraster1-1)
      LD HL,Raster1b+tailleraster1-2
      LD DE,Raster1b+tailleraster1-1
      LD BC,tailleraster1-1
      LDDR
      LD (Raster1b),A
      ;raster 2 rotation vers le haut
      LD A,(Raster2)
      LD HL,Raster2+1
```

```
      LD DE,Raster2
      LD BC,tailleraster2-1
      LDIR
      LD (Raster2+tailleraster2-1),A


      LD B,&F5          ; Se connecter au PPI/8255 port B accessible via &F5xx
.Balayage
      IN A,(C)          ; Lire le contenu
      RRA               ; On teste si le bit 0 = 1
      JP NC,Balayage    ; Si le bit 0 = 0 on boucle jusqu'a la fin du balayage
                        ; On est en haut de l'ecran, on peut continuer le programme

      HALT              ; Attendre interruption 1
      HALT              ; Attendre interruption 2

      DS 20             ; 20 nop - pour une position en debut de ligne

      LD B,&7F          ; On selection le Gate Array
      LD HL,Raster      ; HL pointe sur la table de couleurs
.BoucleRaster
      LD a,(hl)         ; On charge la couleur dans A
      CP &00            ; Si A=0 alors
      JP Z,Touche       ; Saut au test clavier

      LD C,16
      OUT (C),C         ; Selection du border
      OUT (C),A         ; Ecrire la couleur

      LD C,2
      OUT (C),C         ; Selection encre 2
      OUT (C),A         ; Ecrire la couleur

      INC HL            ; Couleur suivante dans le tableau
      DS 32             ; On attend la fin de la ligne... (32 nops)
      JP BoucleRaster


; Test touche
.Touche
      LD BC,&F40E
      OUT (C),C         ;[PIO A] PSG -> Select Reg14
      LD BC,&F6C0       ;PPI I/O
      OUT (C),C
      XOR A
      OUT (C),A
      LD BC,&F792       ;PPI I/O C Control
      OUT (C),C
      LD BC,&F645       ;PPI I/O C - &45 = ligne 5
      OUT (C),C
      LD B,&F4          ;PPI I/O A - lire ligne clavier
      IN A,(C)
      LD BC,&F782       ;PPI I/O C Control
      OUT (C),C
      LD BC,&F600       ;PPI I/O C
      OUT (C),C
      RLA               ;Rotation a gauche - bit 7 (touche ESPACE)
      RLA               ;Rotation a gauche - bit 6 (touche N)
      JR NC,ToucheN     ;Si la touche N est enfoncee on quit
      RLA               ;Rotation a gauche - bit 5 (touche J)
      RLA               ;Rotation a gauche - bit 4 (touche H)
      RLA               ;Rotation a gauche - bit 3 (touche Y)
      JP C,Programme    ;Si la touche Y n'est pas enfoncee on boucle

.ToucheY
      LD A,1
```

```
      LD (&BE80),A      ;Ecrire 1 en &BE80
      CALL Restaurer
      RET

.ToucheN
      LD A,2
      LD (&BE80),A      ;Ecrire 2 en &BE80
      CALL Restaurer
      RET

.Restaurer
      DI                ; Desactiver les interruptions
      LD HL,(Sauver)    ; Restauration des anciennes interruptions
      LD (&38),hl       ; Ecrire
      EI                ; Interruption autorisees
      RET               ; Retour

.Sauver
      DW #0000    ;Sauvegarde valeur 16 bits

.Raster
      DB &5F            ;border bleu pastel

.Raster1
      ;zone fixe
      DB &58            ;magenta
      DB &5F            ;bleu pastel
      DB &58,&58        ;magenta
      DB &5F            ;bleu pastel
      DB &58,&58,&58    ;magenta
      DB &4D,&4D        ;magenta vif
      DB &58            ;magenta
      DB &4D,&4D,&4D    ;magenta vif
      DB &4F            ;magenta pastel
      DB &4D,&4D        ;magenta vif
      DB &4F,&4F        ;magenta pastel

.Raster1b
      DB &56            ;vert
      DB &5A,&5A,&5A    ;vert citron
      DB &56            ;vert
      DB &5A,&5A        ;vert citron
      DB &4A            ;jaune vif
      DB &43,&43        ;jaune pastel
      DB &42,&42        ;vert marin
      DB &4B            ;blanc

      ;zone fixe
      DB &4B            ;blanc
      DB &42,&42        ;vert marin
      DB &43,&43        ;jaune pastel
      DB &4A            ;jaune vif
      DB &5A,&5A        ;vert citron
      DB &56            ;vert
      DB &5A,&5A,&5A    ;vert citron
      DB &56            ;vert
      DB &4F,&4F        ;magenta pastel
      DB &4D,&4D        ;magenta vif
      DB &4F,&4F        ;magenta pastel
      DB &4D,&4D        ;magenta vif
      DB &4F            ;magenta pastel
      DB &4D,&4D,&4D    ;magenta vif
      DB &58            ;magenta
      DB &4D,&4D        ;magenta vif
      DB &58,&58,&58    ;magenta
      DB &5F            ;bleu pastel
```

```
        DB &58,&58          ;magenta
        DB &5F              ;bleu pastel
        DB &58              ;magenta


        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F
        DB &5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F,&5F

.Raster2
        DB &5C       ;rouge fonce
        DB &4C,&4C   ;rouge vif
        DB &4E,&4E   ;orange
        DB &4A,&4A   ;jaune vif
        DB &43,&43   ;jaune pastel
        DB &4B,&4B   ;blanc
        DB &43,&43   ;jaune pastel
        DB &4A,&4A   ;jaune vif
        DB &4E,&4E   ;orange
        DB &4C,&4C   ;rouge vif
        DB &5C       ;rouge fonce

        DB &5F       ;border bleu pastel
        DB 0         ;fin
```

INTRODUCTION - OVERSCAN TITLESCREEN (FULLSCREEN) WITH MUSIC

INTRO.BIN : One file with BASIC + ASM (16 Ko)

```
1 'Kukulcan 2016
10 MODE 1:CALL &3F19:CALL &A000
20 RUN"!INGAME.BAS
```

MODE 1 fullscreen (only 4 colors, no more) with Black, White, Blue and Dark Red.

Code in ASM for the DISPLAY (comments in french)

```
ORG #A000

DI

;BORDER 0 (Noir)
LD BC,&0000
CALL &BC38

;INK 0,0 (Noir)
XOR A
LD BC,&0000
CALL &BC32

;INK 1,3 (Rouge fonce)
LD A,1
LD BC,&0303
CALL &BC32

;INK 2,14 (Bleu)
LD A,2
LD BC,&0E0E
CALL &BC32

;INK 3,26 (Blanc)
LD A,3
LD BC,&1A1A
CALL &BC32

CALL &2E46 ;decompactage

;correction du probleme avec Zenith II (1er octet mange)
LD A,&5A
LD (&1FE),a

;correction du probleme avec convimgcpc (8x8 absent)
LD A,&33
LD (&07FE),a:LD (&07FF),a
LD A,&CC
LD (&0FFE),a:LD (&0FFF),a
LD A,&33
LD (&17FE),a:LD (&17FF),a
LD A,&CC
LD (&1FFE),a:LD (&1FFF),a
LD A,&33
LD (&27FE),a:LD (&27FF),a
LD A,&CC
LD (&2FFE),a
XOR A
LD (&2FFF),a
LD A,&33
LD (&37FE),a
XOR A
LD (&37FF),a
LD A,&CC
LD (&3FFE),a
LD A,&88
LD (&3FFF),a

;OUT overscan
;largeur
LD BC,&BC01
OUT (C),C
LD BC,&BD30
OUT (C),C
```

```
;hauteur (&22)
LD BC,&BC06
OUT (C),C
LD BC,&BD0B ;11 lignes texte uniquement
OUT (C),C

;decalage horizontal
LD BC,&BC02
OUT (C),C
LD BC,&BD32
OUT (C),C

;decalage vertical
LD BC,&BC07
OUT (C),C
LD BC,&BD23
OUT (C),C

;ecran 32 ko (&000 a &7FFF)
LD BC,&BC0C
OUT (C),C
LD BC,&BD0C
OUT (C),C

;debut offset (en &01FE)
LD BC,&BC0D
OUT (C),C
LD BC,&BDFF
OUT (C),C


CALL tempo

;petite animation pour centrer le titre
;animation sur R7 = &23 a &18
LD BC,&BC07
OUT (C),C
LD B,&BD

LD A,&23-&18 ;Dec 11 fois
LD C,&23
OUT (C),C

.anim1
DEC C
OUT (C),C

;ralentissement
CALL &BD19:CALL &BD19:CALL &BD19

DEC A
CP 0
JP NZ,anim1

CALL tempo

;petite animation pour positionner le titre tout en bas
;animation sur R7 = &18 a &C
LD BC,&BC07
OUT (C),C
LD B,&BD

LD A,&18-&C ;DEC nb fois
LD C,&18
OUT (C),C
```

```
.anim2
DEC C
OUT (C),C

;ralentissement
CALL &BD19:CALL &BD19:CALL &BD19

DEC A
CP 0
JP NZ,anim2

;CALL tempo

;petite animation pour remonter le titre
;animation sur R7 = &C a &23
LD BC,&BC07
OUT (C),C
LD B,&BD

LD A,&17 ;INC nb fois
LD C,&C
OUT (C),C

.anim3
INC C
OUT (C),C

;ralentissement
CALL &BD19:CALL &BD19:CALL &BD19

DEC A
CP 0
JP NZ,anim3

;CALL tempo

;animation sur R6 = &B a &22
LD BC,&BC06
OUT (C),C
LD B,&BD

LD A,&17
LD C,&B
OUT (C),C

.anim4
INC C
OUT (C),C

ralentissement
PUSH AF
CALL tempo2
POP AF

DEC A
CP 0
JP NZ,anim4

;Init musique
CALL &C000

loop:
CALL &BD19 ;attente debut balayage
CALL &C006 ;joueur musique

LD A,&2F ;barre d'espace
```

```
CALL &BB1E
CP &80
JP Z,exit
JP loop

exit:
CALL &BCA7 ;nettoyage queue sonore

;animation sur la hauteur R6 = &22 a 0 (masquer)
LD BC,&BC06
OUT (C),C
LD B,&BD

LD A,&22
LD C,&22
OUT (C),C

.anim5
DEC C
OUT (C),C

;ralentissement
CALL &BD19:CALL &BD19:CALL &BD19:CALL &BD19
CALL &BD19:CALL &BD19:CALL &BD19:CALL &BD19

DEC A
CP 0
JP NZ,anim5


;OUT R1=32 R6=0 (On masque reste masque)
;largeur
LD BC,&BC01
OUT (C),C
LD BC,&BD20
OUT (C),C

;hauteur
LD BC,&BC06
OUT (C),C
LD BC,&BD0
OUT (C),C

;decalage horizontal
LD BC,&BC02
OUT (C),C
LD BC,&BD2A
OUT (C),C

;decalage vertical
LD BC,&BC07
OUT (C),C
LD BC,&BD22
OUT (C),C

;ecran 17 ko (&C00 a &FFFF)
LD BC,&BC0C
OUT (C),C
LD BC,&BD3C
OUT (C),C

;debut offset (en &C000)
LD BC,&BC0D
OUT (C),C
LD BC,&BD00
OUT (C),C
```

```
;effacer l'ecran
CALL &BB6C

EI

RET

tempo2:
LD A,8
LOOPtempo2:
CALL &BD19
DEC A
CP 0
JP NZ,LOOPtempo2
RET

;ralentissement
tempo:
LD A,&A0
LOOPtempo:
CALL &BD19
DEC A
CP 0
JP NZ,LOOPtempo
RET
```

INGAME DESIGN

INGAME.PIC : 10 Ko

The screen don't have a normal format, but always use 16 Ko (&C000 to &FFFF).
CRTC R1=32 (width; standard=40)
CRTC R2=42 (horizontal align; standard=46)
CRTC R6=32 (height; standard=25)
CRTC R7=34 (vertical align; standard=30)

50 LEVELS

All levels is checked and it's always possible to finish with a perfect.

```
10 MODE 1:ADR=&C000
20 FOR level=1 TO 50
30 FOR i=0 TO 81
40 READ a
50 POKE ADR+CPT,a
60 CPT=CPT+1
70 NEXT i
80 NEXT level
90 SAVE "INGAME.LVL",B,&C000,CPT
100 END
1010 REM 01
1020 DATA
10,1,0,1,0,1,0,1,1,0,0,0,0,0,0,9,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,8,10,1,0,0,8,0,0
,0,1,0,1,0,0,0,9,0,0,1,0,0,0,0,1,0,1,0,0,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,1,0,0,0,6
1030 REM 02
1040 DATA
1,1,0,1,0,1,0,1,1,1,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0
,0,0,0,3,1,0,0,0,0,0,0,0,0,0,0,0,0,4,0,2,3,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,4,1,7
1050 REM 03
1060 DATA
1,1,0,1,0,1,0,1,0,0,1,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,3
,0,1,0,4,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,2,1,1,1,1,2,1,0,0,4,0,0,8
1070 REM 04
1080 DATA
1,8,1,0,1,0,1,8,1,0,0,0,0,0,0,0,0,0,1,0,0,0,11,0,0,0,1,0,0,0,9,0,0,0,0,10,1,0,0,0,0,0
,0,0,1,0,0,0,0,10,0,0,0,0,0,0,0,0,0,11,0,1,0,1,0,0,0,0,0,0,0,9,0,1,0,1,0,1,0,0,0,0,8
1090 REM 05
1100 DATA
0,0,0,1,9,8,0,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,10,0,0,0,11,0,1,0,0,0,0,0
,1,0,0,11,0,0,0,0,0,0,1,1,1,0,0,10,0,0,0,9,8,0,0,0,0,0,0,0,1,1,0,0,1,0,1,0,0,0,0,8
1110 REM 06
1120 DATA
0,0,0,0,0,0,0,0,0,0,1,2,0,0,0,2,1,0,0,1,2,0,0,0,2,1,0,0,1,1,0,0,0,1,1,0,0,1,2,0,2,0,2
,1,0,0,1,1,2,2,2,1,1,0,0,1,1,2,0,2,1,1,0,0,1,2,0,0,0,2,1,0,0,0,0,0,0,0,0,0,0,0,8
1130 REM 07
1140 DATA
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,0,4,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,3,0,0,0,0,0,4,0,0,2,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9
1150 REM 08
1160 DATA
0,1,0,1,4,0,1,1,1,1,0,0,0,0,0,0,0,0,0,3,0,0,0,2,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,3,0,0
,0,0,0,0,0,0,4,0,0,0,0,1,0,0,0,1,0,0,1,1,0,0,2,0,1,0,0,1,1,1,1,0,0,0,1,1,1,1,9
1170 REM 09
1180 DATA
1,1,1,0,0,1,1,0,1,0,0,1,0,2,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,8,0,0,1,1,9,0,0,0,3,0,0
,0,0,0,0,0,0,1,0,5,0,0,1,11,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,10,0,0,4,1,8
1190 REM 10
1200 DATA
1,3,1,0,1,1,1,0,1,0,0,0,0,0,0,0,5,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,0,0,1,0,1,8,0,0,0,0,0
,0,0,0,0,0,0,0,0,11,0,2,4,0,0,0,0,9,0,0,0,0,1,0,0,0,0,0,0,0,0,10,1,1,0,0,1,0,1,0,0,10
1210 REM 11
1220 DATA
1,8,0,0,1,1,9,0,1,0,0,1,9,0,10,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,1,0,0,0,1,
1,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,10,8,1,1,1,0,0,1,1,0,1,10
1230 REM 12
1240 DATA
1,4,0,8,1,0,1,1,1,0,0,0,0,1,0,1,0,1,5,0,3,0,0,0,0,0,0,0,0,0,0,0,0,1,11,0,1,1,0,1,0,0,0,
0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,10,1,9,1,2,0,1,0,1,0,9
```

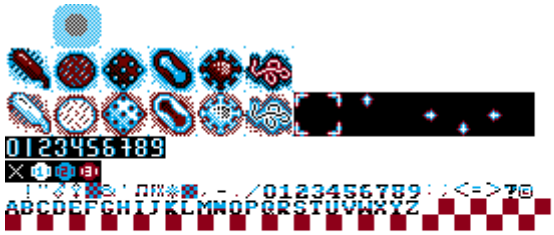
```
1250 REM 13
1260 DATA
1,1,0,1,8,1,1,0,1,1,0,0,0,0,0,0,10,1,1,0,0,0,0,0,10,0,1,0,11,0,9,0,0,0,0,0,1,0,0,0,0,
0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,9,0,0,11,0,1,0,1,0,0,0,1,0,1,1,0,1,1,8,1,1,0,1,9
1270 REM 14
1280 DATA
1,0,1,1,8,1,0,1,0,0,0,0,0,0,0,0,1,0,1,0,0,1,0,0,0,0,0,0,10,0,0,9,0,9,0,0,1,0,0,0,0,0,
0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,11,0,10,0,1,11,1,8,9
1290 REM 15
1300 DATA
1,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,3,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,4,0,0,1,11,0,0,0,0,
0,5,0,0,10,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9,0,8,1,0,1,0,1,0,0,0,0,0,0,0,0,2,0,0,0,0,0,9
1330 REM 16
1340 DATA
1,0,0,0,1,0,1,0,0,0,11,0,0,1,9,1,0,1,0,0,1,0,0,0,0,0,0,8,0,0,0,0,0,10,0,0,0,0,0,0,0,0
,0,0,1,0,0,0,0,0,0,0,0,0,9,1,0,0,0,0,1,0,0,1,10,0,0,11,0,0,0,0,0,0,8,0,0,0,0,0,0,1,9
1350 REM 17
1360 DATA
1,0,1,0,1,1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,9,0,0,0,0,1,10,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,
1,0,1,9,0,0,10,0,0,0,8,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,8,1,0,1,0,1,9
1370 REM 18
1380 DATA
1,0,8,1,8,1,0,1,0,10,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,9,0,0,0,0,
0,0,0,0,10,0,9,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,9
1390 REM 19
1400 DATA
1,5,2,0,0,0,0,0,4,0,0,0,0,0,0,1,0,1,0,1,0,0,0,0,0,0,3,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0
,0,1,3,0,0,0,0,0,0,0,4,0,0,0,2,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,1,0,5,1,0,1,0,9
1410 REM 20
1420 DATA
1,0,0,1,0,1,10,1,1,0,0,0,0,10,0,0,0,0,0,0,8,0,0,0,0,1,0,0,0,0,0,0,11,0,0,0,1,0,8,0,0,
0,0,0,1,9,0,0,0,0,0,9,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,0,11,1,0,1,0,1,0,9
1430 REM 21
1440 DATA
0,0,0,0,0,0,0,0,0,0,2,1,1,1,1,1,0,0,0,0,11,4,0,0,4,9,0,0,0,1,1,0,0,11,1,0,0,0,1,1,1,1
,1,0,0,0,0,1,1,0,1,1,0,0,0,0,1,1,0,0,1,1,0,0,8,1,1,0,0,0,1,3,0,0,0,0,0,0,0,0,0,9
1450 REM 22
1460 DATA
1,0,1,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,3,0,0,10,0,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,5,0,2,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,8,0,9,0,11,1,1,0,0,0,0,0,0,0,9
1470 REM 23
1480 DATA
1,1,1,1,0,3,0,1,1,1,0,1,1,1,0,1,1,1,0,0,0,0,0,0,2,4,1,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0
,0,4,0,1,0,0,0,1,1,0,1,1,0,0,0,0,0,0,0,1,1,0,0,2,0,0,0,0,3,0,1,0,0,0,0,0,0,1,1,10
1490 REM 24
1500 DATA
1,0,0,1,0,0,1,1,1,0,0,1,0,0,2,0,0,1,0,3,10,0,4,0,1,0,1,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,
0,0,1,1,0,0,1,0,11,0,0,0,0,0,5,1,0,0,0,0,1,1,0,0,0,0,0,0,0,9,1,1,0,0,0,8,0,1,1,10
1510 REM 25
1520 DATA
1,0,0,1,0,0,0,1,1,0,0,5,0,0,2,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,5,0,0,0,0,1,0,0,0,0,0,0
,0,1,4,0,0,1,1,0,0,3,0,1,0,0,0,1,0,0,1,0,4,0,0,1,0,0,3,1,1,1,0,0,0,1,2,0,0,1,10
1530 REM 26
1540 DATA
1,8,1,0,1,0,1,0,0,1,9,1,11,1,0,1,0,0,0,0,0,10,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,9,0
,1,0,10,1,0,0,0,0,8,1,0,1,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,11,1,1,1,0,1,0,1,0,0,1,1,10
1550 REM 27
1560 DATA
1,0,0,1,0,1,0,1,0,0,0,0,0,9,0,0,0,0,1,3,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0
,0,1,1,11,0,1,4,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,5,0,0,0,0,0,0,8,2,0,1,0,0,0,0,10,1,10
1570 REM 28
1580 DATA
0,0,1,0,1,4,0,1,0,1,0,0,0,0,0,0,1,0,0,0,0,4,0,0,0,5,0,1,0,0,5,1,0,0,0,1,0,0,0,0,3,0,0
,1,0,1,0,0,0,0,0,0,0,0,2,0,0,0,3,2,0,0,0,1,0,0,0,0,0,0,0,1,0,0,1,0,1,0,1,0,0,10
1590 REM 29
1600 DATA
```

```
0,1,9,1,10,1,9,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11,0,1,0,1,0,0,0,0,0,0,0,0,8,0,0,0,0,0
,0,0,0,0,0,0,11,0,10,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,1,0,1,8,0,1,0,0,11
1610 REM 30
1620 DATA
1,1,0,1,0,0,0,1,1,1,8,0,0,0,0,1,1,0,0,10,0,0,0,0,0,0,0,1,1,0,0,0,0,0,9,1,0,10,0,0,0,0
,0,0,0,1,0,0,0,0,1,0,0,8,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,1,1,0,0,1,0,0,0,9,0,1,1,11
1630 REM 31
1640 DATA
1,1,9,0,0,0,3,1,1,1,0,0,1,1,1,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,10,0,
8,0,0,1,0,2,0,0,0,1,0,0,1,0,4,0,0,1,0,0,11,1,1,0,1,0,0,0,1,0,1,0,0,0,5,0,0,0,0,11
1650 REM 32
1660 DATA
1,0,1,0,1,0,1,0,1,2,0,0,0,0,0,5,0,0,1,0,0,0,0,0,0,0,1,0,0,3,0,0,0,0,0,8,1,0,0,0,0,11,
0,1,1,1,1,0,9,0,1,0,0,1,1,0,0,4,0,1,0,0,1,0,0,0,0,0,0,0,0,0,1,0,10,1,0,0,0,1,1,11
1690 REM 33
1700 DATA
1,0,1,11,1,1,0,0,0,0,0,0,0,0,1,0,0,1,9,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,
0,10,0,1,0,0,0,0,0,0,11,0,0,0,0,0,0,10,0,0,8,1,0,8,0,9,1,0,0,0,1,0,0,0,0,1,0,0,1,11
1710 REM 34
1720 DATA
0,0,0,0,0,1,1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,5,0,0,0,0,0,3,0,0,0,0,0,1,0,0,1,9,0,0,0,0
,0,0,0,10,0,0,0,1,0,0,2,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,8,11,0,1,1,1,1,4,1,0,0,1,11
1730 REM 35
1740 DATA
1,1,0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,11,9,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,1,0,0,0,10,0,0
,0,8,0,1,0,0,8,0,0,0,0,0,0,0,0,0,0,0,11,1,0,1,0,0,1,0,10,9,1,0,0,0,1,0,0,0,0,0,0,9
1790 REM 36
1800 DATA
0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,1,1,1,0,0,10,0,0,0,0,8,0,0,1,9,0,0,0,1,0,0,0,0,0,1,0,
10,0,0,1,0,0,0,0,0,0,0,1,9,0,0,8,0,0,0,0,0,1,0,0,11,0,11,0,0,1,0,0,1,0,1,0,0,1,1,11
1810 REM 37
1820 DATA
0,0,1,5,0,5,1,0,0,0,1,8,7,0,6,10,1,0,0,1,1,0,0,0,1,1,0,0,1,9,0,0,0,7,1,0,1,0,0,0,0,0,
0,0,1,0,1,0,0,0,0,0,1,0,7,0,3,0,0,0,7,0,12,0,0,0,10,0,8,0,0,0,0,1,0,1,0,1,0,1,0,12
1830 REM 38
1840 DATA
1,1,2,1,1,1,1,0,1,1,0,0,0,0,0,0,4,1,0,0,0,1,0,0,1,0,0,0,1,4,3,1,1,3,0,0,1,0,0,1,0,2,0
,0,0,1,1,0,0,0,1,0,0,0,1,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,1,1,1,1,12
1850 REM 39
1860 DATA
1,0,0,1,0,1,0,0,1,1,0,5,0,0,0,0,3,1,0,0,1,11,0,0,0,0,0,1,0,0,0,0,0,0,2,0,1,0,0,0,1,0,
0,0,1,4,0,0,0,0,0,0,0,0,0,8,0,0,0,0,0,9,0,1,0,0,0,0,0,0,0,0,0,10,0,1,1,0,1,0,0,1,12
1870 REM 40
1880 DATA
1,0,1,1,1,0,1,0,1,1,0,0,1,0,0,0,0,1,1,1,0,10,0,0,0,1,1,1,0,0,9,0,0,0,0,1,0,0,8,1,1,0,
0,0,0,1,0,1,1,0,0,0,0,0,0,1,0,0,9,0,0,0,0,1,1,0,0,10,0,0,0,0,0,1,1,1,8,1,1,1,0,1,12
1890 REM 41
1900 DATA
1,1,0,1,0,1,1,0,1,1,2,0,3,0,0,0,0,1,11,0,0,4,0,0,0,0,5,0,0,0,0,1,0,0,0,0,8,0,1,0,0,0,
0,0,1,1,0,0,10,0,0,1,0,0,0,0,0,0,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,9,1,10
1910 REM 42
1920 DATA
3,9,0,0,0,0,0,0,0,10,0,1,1,1,1,1,0,0,0,1,1,0,0,2,1,1,0,0,1,1,0,1,1,1,1,0,0,1,1,0,1,1,
1,1,0,0,1,1,0,1,1,1,1,0,0,1,1,0,0,0,0,0,0,0,0,1,1,1,1,10,0,0,0,0,0,0,0,0,0,0,8,11
1930 REM 43
1940 DATA
4,1,0,1,0,0,0,0,10,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,9,0,0,0,0,8,0,1,0,11,0,0,0
,0,2,0,1,0,0,1,0,3,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,5,1,1,0,1,0,1,0,0,0,13
1950 REM 44
1960 DATA
1,0,0,5,0,0,0,0,3,1,0,0,0,0,0,1,1,0,0,1,1,1,2,0,4,0,0,0,0,0,0,0,0,2,0,0,0,0,0,1,0,0,0,0,0,0
,0,0,0,0,0,0,1,5,1,1,0,1,0,0,1,0,1,0,1,3,1,0,0,0,0,0,0,0,1,1,1,4,0,0,0,0,0,0,1,1,1,14
1970 REM 45
1980 DATA
0,0,0,0,0,0,0,0,0,0,0,0,0,1,5,1,0,0,0,0,0,0,1,2,0,3,1,0,0,0,0,1,1,0,0,0,0,1,1,0,0,0,1,4,0,0,0,5
,1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,1,3,0,2,1,0,0,0,0,0,0,1,4,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14
```

```
1990 REM 46
2000 DATA
0,0,0,0,3,7,0,0,0,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,1,1,1,0,1,1,1,0,2,0,4,0,6,3,0
,0,4,0,1,1,1,0,1,1,1,0,0,0,0,1,0,1,0,6,0,0,1,2,1,0,1,0,1,0,11,0,0,0,11,0,0,0,7,14
2010 REM 47
2020 DATA
0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,1,9,0,1,1,0,0,1,1,9,0,0,0,0,0,0,10,1,10,0,0
,0,0,0,0,1,1,8,0,0,0,0,0,0,0,1,1,8,0,1,1,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,13
2030 REM 48
2040 DATA
0,0,0,0,0,0,0,0,0,0,0,0,0,11,0,0,0,0,0,0,0,3,1,9,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,1,
0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,4,1,10,1,0,0,0,8,1,0,5,0,1,2,0,0,0,0,0,0,0,0,0,0,16
2050 REM 49
2060 DATA
0,0,0,0,0,0,0,0,0,0,0,10,9,1,12,9,12,8,0,0,8,0,0,0,0,10,11,0,0,0,0,0,0,0,1,9,0,0,0,0,0,
0,1,8,0,0,0,0,0,0,0,1,10,0,0,0,0,0,0,0,0,9,11,0,0,0,0,0,0,0,0,10,8,0,0,0,0,0,0,0,0,0,0,0,
21
2070 REM 50
2080 DATA
0,0,0,0,0,0,0,0,0,0,0,0,8,11,0,10,9,0,0,0,0,10,1,1,1,1,1,10,0,0,9,1,1,1,1,1,9,0,0,8,1,1,1
,1,1,8,0,0,0,11,1,1,1,11,0,0,0,0,0,9,11,10,0,0,0,0,0,0,0,0,8,0,0,0,0,0,0,0,0,0,0,0,0,0,
24
```

TILES

For encoding the TILES in memory i using an ASCII file with adress,width,height for each tiles.
Two type of tiles, opaque and transparent. For the target and arrows, the black color is transparent.




```
1 'Kukulcan
10 OPENOUT "D":MEMORY &3FFF:CLOSEOUT
20 LOAD"INGAME.RSX",&9000:CALL &9000
30 MODE 1:INK 0,0:INK 1,3:INK 2,14:INK 3,26:PEN 1:LOAD"tile.scr",&C000
40 base=&4000
50 MEM=base
60 longueur=0
70 OPENIN "adresses.txt"
80 WHILE NOT EOF
90 INPUT #9,adresse
100 IF adresse=0 THEN GOTO 150
110 INPUT #9,largeur,hauteur
120 |CONV,adresse,MEM,largeur,hauteur
130 longueur=longueur+(largeur*hauteur)
140 MEM=base+longueur
150 WEND
160 SAVE "ingame.spr",b,base,longueur
170 PRINT "TILES SAVED"
```

For the viruses explosion, a new séries of tiles are created with black color is transparent.

```
1 'Kukulcan
10 OPENOUT "D":MEMORY &3FFF:CLOSEOUT
20 LOAD"INGAME.RSX",&9000:CALL &9000
30 MODE 1:INK 0,0:INK 1,3:INK 2,14:INK 3,26:PEN 1:LOAD"tilecls.scr",&C000
40 base=&4000
50 MEM=base
60 longueur=0
70 adresse=&C000
80 FOR i=0 TO 5
90 largeur=6:hauteur=22
100 |CONV,adresse,MEM,largeur,hauteur
110 longueur=longueur+(largeur*hauteur)
120 MEM=base+longueur
130 adresse=adresse+largeur
140 NEXT
150 SAVE "ingame.cls",b,base,longueur
160 PRINT "TILESCLS SAVED"
```

For finding the good adress in screen, i using a little program in BASIC :

```
1 'Kukulcan
10 OPENOUT "D":MEMORY &3FFF:CLOSEOUT
20 LOAD"INGAME.RSX",&9000:CALL &9000
30 MODE 1:INK 0,0:INK 1,3:INK 2,14:INK 3,26:LOAD"tile.scr",&4000
40 GOSUB 230:CALL &BE80
50 REM ADRESSE A L'ECRAN
60 ROW=0:COL=0:ADR=&C000:GOSUB 190
70 REM DEPLACEMENT
80 IF INKEY(0)=0 AND ROW>0 THEN GOSUB 210:ROW=ROW-1:GOSUB 140:GOSUB 190:'Haut
90 IF INKEY(2)=0 AND ROW<199 THEN GOSUB 210:ROW=ROW+1:GOSUB 140:GOSUB 190:'Bas
100 IF INKEY(8)=0 AND COL>0 THEN GOSUB 210:COL=COL-1:GOSUB 140:GOSUB 190:'Gauche
110 IF INKEY(1)=0 AND COL<79 THEN GOSUB 210:COL=COL+1:GOSUB 140:GOSUB 190:'Droite
120 IF INKEY(47)=0 THEN GOSUB 260
130 GOTO 70
140 REM calcul adresse ecran
150 NBROW = ROW\8
160 NBSUBROW = ROW - (NBROW * 8)
170 ADR=&C000+(NBSUBROW* &800)+(NBROW* &50)+COL
180 RETURN
190 REM stocker l'octet
200 A=PEEK(ADR):POKE ADR,0:RETURN
210 REM restaurer octet
220 POKE ADR,A:RETURN
230 REM transferer image image de &4000 vers &C000
240 DATA 21,00,40,11,00,C0,01,00,40,ED,B0,C9
250 RESTORE 240:FOR i=&BE80 TO &BE8B:READ a$:POKE i,VAL("&"+a$):NEXT:RETURN
260 REM affichage adresse
270 LOCATE 1,1:PRINT "&";HEX$(ADR,4):CLEAR INPUT:CALL &BB06:CALL &BE80:RETURN
```

RSX

The game is coded in BASIC and associated with personnal RSX for good speed for display and treatment of data.

A) RSX in ASM (with comments in french)

```
ADRSCORE    EQU   &A000
TILESFONTE  EQU   &7078
;adr = &67E0 + recompense (4*3*11 = 132) + chiffres (10*2*11 = 220) + vide/mur/virus
(14*6*22 = 1848)


ORG &9000
; Kukulcan 2016
; ***************************
; *** RSX - INITIALISATION ***
; ***************************
.RSX_Init :
      LD    HL,RSX_Init
      LD    (HL),&C9            ; Ecrire un RET pour empecher une nouvelle
initialisation
      LD    BC,RSX_Commandes    ; BC pointe sur la table des Commandes RSX
      LD    HL,RSX_Tampon       ; HL pointe sur 4 octets libres.
      JP    &BCD1

.RSX_Tampon :
      DEFS  4                   ; Tampon de quatre octets.

.RSX_Commandes :
      DEFW  RSX_Mots_Clefs      ; Adresse des mots clefs
      JP    CONV_LINEAIRE
      JP    AFF_CIBLE
      JP    AFF_SPRITE
      JP    OUT40
      JP    OUT32
      JP    MENU
      JP    ROUTINEHIST
      JP    ROUTINECONTROLS
      JP    ROUTINECREDIT
      JP    MSG
      JP    SCORECLEAN
      JP    STATS
      JP    FX
      JP    AFFRESULT
      JP    TILE8
      JP    CLS

.RSX_Mots_Clefs
      DEFB    "CON","V"+&80
      DEFB  "CIBL","E"+&80
      DEFB  "SP","R"+&80
      DEFB  "OUT4","0"+&80
      DEFB  "OUT3","2"+&80
      DEFB  "MEN","U"+&80
      DEFB  "AFFHIS","T"+&80
      DEFB  "AFFCTR","L"+&80
      DEFB  "AFFCREDIT","S"+&80
      DEFB  "MS","G"+&80
      DEFB  "SCORECLEA","N"+&80
      DEFB  "STAT","S"+&80
      DEFB  "F","X"+&80
```

```
      DEFB  "AFFRESUL","T"+&80
      DEFB  "TILE","8"+&80
      DEFB  "CL","S"+&80
      DEFB    0               ; Fin de la table.

;syntax |CONV,adresse_ecran,adresse_destination,largeur,hauteur
CONV_LINEAIRE
            ;lecture valeur hauteur
            LD A,(IX+0)
            LD (conv_hauteur+1),A

            ;lecture valeur largeur
            LD A,(IX+2)
            LD (conv_largeur+1),A

            ;lecture valeur adresse_destination
            LD D,(IX+5)
            LD E,(IX+4)

            ;lecture valeur adresse_ecran
            LD H,(IX+7)
            LD L,(IX+6)

conv_hauteur:
            LD B,0            ;//nb de ligne (Hauteur)
conv_infos:
            PUSH BC           ;//ecrire BC dans la pile
            PUSH HL           ;//ecrire HL dans la pile
conv_largeur
            LD B,0            ;//largeur du sprite (en octet)
conv_ligne:
            LD A,(HL)         ;//a = valeur de l'octet a l'ecran
            LD (DE),A         ;//ecrire dans destination la valeur de a
            INC DE            ;//adresse de destination suivante
            INC HL            ;//adresse ecran suivante
            DJNZ conv_ligne   ;//B=B-1 si B est <> 0 alors on boucle sur ligne
                              ;B=0, calculer l'adresse de la prochaine ligne a
l'ecran
            POP HL            ;//recuperer HL de la pile
            ;CALL &BC26       ;//Calcul ligne ecran suivante (vers le BAS)
            CALL conv_ligne_suivante
            POP BC            ;//recuperer BC de la pile (C'est surtout pour le
registre B
                              ;//contenant le nombre de ligne que c'est interessant,
vu qu'on
                              ;//l'utilise 2 fois ce registre dans la routine)
            DJNZ conv_infos   ;//B=B-1 si B <>0 on boucle dans infos
            RET               ;//retour au basic, toutes les lignes ont ete traitees

conv_ligne_suivante:
            LD A,H
            ADD A,#08
            LD H,A
            RET NC
            LD BC,#C050
            ADD HL,BC
            RET


;syntax |CIBLE,adresse_ecran,adresse_sprite,largeur,hauteur
;Routine de Christphe PETIT - Affichage d'un sprite entier avec transparence sur
l'encre 0 en mode 1
.AFF_CIBLE
            ;lecture valeur hauteur
            LD A,(IX+0)
            LD (cible_hauteur+1),A
```

```
            ;lecture valeur largeur
            LD A,(IX+2)
            LD (cible_largeur+1),A

            ;lecture valeur adresse_sprite
            LD H,(IX+5)
            LD L,(IX+4)

            ;lecture valeur adresse_ecran
            LD D,(IX+7)
            LD E,(IX+6)

cible_hauteur     LD C,0
AFF_CIBLE1  PUSH DE
cible_largeur     LD B,0     ; B = nombre d'octets a copier par ligne
AFF_CIBLE2  PUSH BC
            LD A,(HL)        ; A = octet courant du sprite
            LD C,A           ; puis C
              RLCA
            RLCA
            RLCA
            RLCA             ; Inversion poids faible <-> poids fort
            OR C             ; A = masque des couleurs de sprite a conserver
            CPL              ; inversion pour masque des couleurs de fond a supprimer
            LD B,A           ; B = masque des couleurs a supprimer
            LD A,(DE)        ; A = octet de fond d'ecran courant
            AND B            ; suppression des couleurs genantes
            OR C             ; ajout des couleurs du sprite
            LD (DE),A        ; affichage du resultat
            INC HL
            INC DE
            POP BC
            DJNZ AFF_CIBLE2   ; Rebouclage jusqu'a traitement complet de la ligne

            POP DE
            PUSH BC
            CALL ligne_suivante_suivant_R1
            POP BC
            DEC C
            JR NZ,AFF_CIBLE1
            RET


;syntax |FX,valeur (0 ou 1)
.FX
            LD A,(IX+0)

            LD C,8            ;la parenthese ouverte commence a la position 8
            ADD C

            LD L,A            ;L = A
            LD H,0            ;H = 0
            ;HL contient A (A * 2*8=16)
            ADD HL,HL         ;*2
            ADD HL,HL         ;*4
            ADD HL,HL         ;*8
            ADD HL,HL         ;*16
            LD DE,TILESFONTE
            ADD HL,DE         ;HL contient l'adresse du tiles

            LD A,8
            LD (spr_hauteur+1),A
            LD A,2
            LD (spr_largeur+1),A
```

```
            LD DE,&DF95
            CALL spr_hauteur
            RET

;syntax |SPR,adresse_ecran,adresse_sprite,largeur,hauteur
.AFF_SPRITE
            ;lecture valeur hauteur
            LD A,(IX+0)
            LD (spr_hauteur+1),A

            ;lecture valeur largeur
            LD A,(IX+2)
            LD (spr_largeur+1),A

            ;lecture valeur adresse_sprite
            LD H,(IX+5)
            LD L,(IX+4)

            ;lecture valeur adresse_ecran
            LD D,(IX+7)
            LD E,(IX+6)
spr_hauteur
            LD A,&20    ;hauteur
spr_loop
            PUSH AF
            PUSH DE
spr_largeur
            LD BC,&0000 ;largeur
            LDIR
            POP DE
            CALL ligne_suivante_suivant_R1
            POP AF
            DEC A
            JR NZ,spr_loop

.ligne_suivante_suivant_R1
            LD A,D
            ADD A,8
            LD D,A
            AND &38
            RET NZ
            LD A,D
            SUB &40
            LD D,A
            LD A,E
            ADD A,&40   ;si R1=40 alors &50 si R1=41 alors &52 si R1=32 alors &40
            LD E,A
            RET NC
            INC D
            LD A,D
            AND 7
            RET NZ
            LD A,D
            SUB 8
            LD D,A
            RET

.OUT40
      LD BC,&BC01
      OUT(C),C
      LD BC,&BD00+40
      OUT(C),C

      LD BC,&BC02
      OUT(C),C
      LD BC,&BD00+46
```

```
      OUT(C),C

      LD BC,&BC06
      OUT(C),C
      LD BC,&BD00+25
      OUT(C),C

      LD BC,&BC07
      OUT(C),C
      LD BC,&BD00+30
      OUT(C),C
      RET
.OUT32
      ;largeur
      LD BC,&BC01
      OUT (C),C
      LD BC,&BD20
      OUT (C),C

      ;hauteur
      LD BC,&BC06
      OUT (C),C
      LD BC,&BD20
      OUT (C),C

      ;decalage horizontal
      LD BC,&BC02
      OUT (C),C
      LD BC,&BD2A
      OUT (C),C

      ;decalage vertical
      LD BC,&BC07
      OUT (C),C
      LD BC,&BD22
      OUT (C),C
      RET

.MENU
AFFICHAGE_MENU_PRINCIPAL
      LD HL,MENU0
      LD DE,&D957
      CALL directmini
      LD HL,MENU1
      LD DE,&FA94
      CALL directmini
      LD HL,MENU2
      LD DE,&EB54
      CALL directmini
      LD HL,MENU3
      LD DE,&DC14
      CALL directmini
      LD HL,MENU4
      LD DE,&CCD4
      CALL directmini
      LD HL,MENU5
      LD DE,&FD54
      CALL directmini
      LD HL,MENU6
      LD DE,&EE08
      CALL directmini
      RET
MENU0 DEFB "*M*E*N*U*"
      DB 0 ;fin d'affichage
MENU1 DEFB "1 - PLAY"
      DB 0 ;fin d'affichage
```

```
MENU2 DEFB "2 - THE STORY"
      DB 0 ;fin d'affichage
MENU3 DEFB "3 - CONTROLS"
      DB 0 ;fin d'affichage
MENU4 DEFB "4 - CREDITS"
      DB 0 ;fin d'affichage
MENU5 DEFB "5 - RESULTS"
      DB 0 ;fin d'affichage
MENU6 DEFB "2016 @ WWW.CPC-POWER.COM"
      DB 0 ;fin d'affichage

ROUTINEHIST
      LD HL,HIST0
      LD DE,&D951
      CALL directmini
      LD HL,HIST1
      LD DE,&FA85+1
      CALL directmini
      LD HL,HIST2
      LD DE,&EB45+1
      CALL directmini
      LD HL,HIST3
      LD DE,&DC05+1
      CALL directmini
      LD HL,HIST4
      LD DE,&CCC5+1
      CALL directmini
      LD HL,HIST5
      LD DE,&FD45+1
      CALL directmini
      LD HL,HIST6
      LD DE,&EE05+1
      CALL directmini
      RET
           ;012345678901234567890123456
HIST0 DEFB "**THE***STORY**"
      DB 0 ;fin d'affichage
HIST1 DEFB "YOUR DOG W.CAMELOT IS ILL."
      DB 0 ;fin d'affichage
HIST2 DEFB "VIRUSES ARE ATTACKING HIM."
      DB 0 ;fin d'affichage
HIST3 DEFB "YOUR MISSION IS TO GET RID"
      DB 0 ;fin d'affichage
HIST4 DEFB "OF  THEM  THROUGHOUT  THE "
      DB 0 ;fin d'affichage
HIST5 DEFB "50 LEVELS.       BEWARE!!!"
      DB 0 ;fin d'affichage
HIST6 DEFB " VIRUSES   ARE   CLEVER!!!"
      DB 0 ;fin d'affichage

ROUTINECONTROLS
      LD HL,CTRL0
      LD DE,&D94F
      CALL directmini
      LD HL,CTRL1
      LD DE,&FA85+1
      CALL directmini
      LD HL,CTRL2
      LD DE,&EB45+1
      CALL directmini
      LD HL,CTRL3
      LD DE,&DC05+1
      CALL directmini
      LD HL,CTRL4
      LD DE,&CCC5+1
      CALL directmini
```

```
      LD HL,CTRL5
      LD DE,&FD45+1
      CALL directmini
      LD HL,CTRL6
      LD DE,&EE05+1
      CALL directmini
      RET
           ;01234567890123456789012345678
CTRL0 DEFB "*C*O*N*T*R*O*L*S*"
      DB 0 ;fin d'affichage
CTRL1 DEFB "MOVE CURSOR AND VIRUS WITH"
      DB 0 ;fin d'affichage
CTRL2 DEFB "JOYSTICK  OR  ARROW  KEYS."
      DB 0 ;fin d'affichage
CTRL3 DEFB "     R=RESTART LEVEL      "
      DB 0 ;fin d'affichage
CTRL4 DEFB "A=ABORT GAME      M=( OR )"
      DB 0 ;fin d'affichage
CTRL5 DEFB "KILL EVERY VIRUS TO ACCESS"
      DB 0 ;fin d'affichage
CTRL6 DEFB "   TO  THE  NEXT  LEVEL.  "
      DB 0 ;fin d'affichage

ROUTINECREDIT
      LD HL,CREDIT0
      LD DE,&D951
      CALL directmini
      LD HL,CREDIT1
      LD DE,&FA85
      CALL directmini
      LD HL,CREDIT2
      LD DE,&EB45
      CALL directmini
      LD HL,CREDIT3
      LD DE,&DC05
      CALL directmini
      LD HL,CREDIT4
      LD DE,&CCC5
      CALL directmini
      LD HL,CREDIT5
      LD DE,&FD45
      CALL directmini
      LD HL,CREDIT6
      LD DE,&EE05
      CALL directmini
      RET

            ;01234567890123456789012345678
CREDIT0     DEFB "*C*R*E*D*I*T*S*"
      DB 0 ;fin d'affichage
CREDIT1     DEFB "   IDEA & CODE: KUKULCAN   "
      DB 0 ;fin d'affichage
CREDIT2     DEFB "    GFX: KUKULCAN & CED    "
      DB 0 ;fin d'affichage
CREDIT3     DEFB "      MUSIC: T&J/GPA       "
      DB 0 ;fin d'affichage
CREDIT4     DEFB "   COVER & MANUAL: IXIEN   "
      DB 0 ;fin d'affichage
CREDIT5     DEFB "TESTERS: AST*GALAMOTH*LONE*"
      DB 0 ;fin d'affichage
CREDIT6 DEFB " *MAXIT*C.PETIT*FREDOUILLE*"
      DB 0 ;fin d'affichage

;
.AFFRESULT
      LD HL,AFFRES0
```

```
      LD DE,&D951
      CALL directmini

      ;cadre
      LD HL,resultligne
      LD DE,&D4D3
      CALL directmini

      LD HL,&70C8
      LD DE,&D513
      CALL stats_aff_TILES_1
      LD HL,&70C8
      LD DE,&D513+22
      CALL stats_aff_TILES_1

      LD HL,&70C8
      LD DE,&D553
      CALL stats_aff_TILES_1
      LD HL,&70C8
      LD DE,&D553+22
      CALL stats_aff_TILES_1

      LD HL,&70C8
      LD DE,&D593
      CALL stats_aff_TILES_1
      LD HL,&70C8
      LD DE,&D593+22
      CALL stats_aff_TILES_1

      LD HL,&70C8
      LD DE,&D5D3
      CALL stats_aff_TILES_1
      LD HL,&70C8
      LD DE,&D5D3+22
      CALL stats_aff_TILES_1

      LD HL,&70C8
      LD DE,&D613
      CALL stats_aff_TILES_1
      LD HL,&70C8
      LD DE,&D613+22
      CALL stats_aff_TILES_1

      LD HL,resultligne
      LD DE,&D653
      CALL directmini

      ;sprite
      LD A,40
      LD (spr_hauteur+1),A
      LD A,20
      LD (spr_largeur+1),A
      LD HL,&A1D0
      LD DE,&D515
      CALL spr_hauteur

      RET
resultligne       DEFB "%%%%%%%%%%%%"
      DB 0;fin d'affichage

AFFRES0     DEFB "*R*E*S*U*L*T*S*"
      DB 0 ;fin d'affichage


AFFICHAGE_MINI_TEXTE
;     LD HL,TEXTE1
```

```
;     LD DE,&C000       ;adresse affichage a l'ecran

;le texte devra obligatoirement avoir un zero a la fin
      ;lecture valeur adresse_texte
      LD H,(IX+3)
      LD L,(IX+2)

      ;lecture valeur adresse_ecran
      LD D,(IX+1)
      LD E,(IX+0)
directmini
      ;lecture valeur hauteur
      LD A,8
      LD (spr_hauteur+1),A

      ;lecture valeur largeur
      LD A,2
      LD (spr_largeur+1),A
loop_affichage_mini_texte
      LD A,(HL)
      CP 0
      RET Z

      LD C,32
      SUB C ;A = A-32

      PUSH HL           ;sauver HL
      PUSH DE           ;sauver DE
      ;CALL &BB5A
      LD L,A            ;L = A
      LD H,0            ;H = 0
      ;HL contient A (A * 2*8=16)
      ADD HL,HL         ;*2
      ADD HL,HL         ;*4
      ADD HL,HL         ;*8
      ADD HL,HL         ;*16
      LD DE,TILESFONTE
      ADD HL,DE         ;HL contient l'adresse du tiles

      POP DE            ;recuperer l'adresse d'affichage a l'ecran
      PUSH DE           ;on resauve DE
      CALL spr_hauteur  ;affichage du sprite
      POP DE            ;on recupere
      INC DE
      INC DE            ;position suivante a l'ecran +2

      POP HL            ;recuperer HL
      INC HL            ;position caractere suivant

      JR loop_affichage_mini_texte

;syntax |MSG,numero_texte
.MSG
      LD A,(IX+0)

      LD HL,adresse_messages-26

.msgloop
      LD DE,26
      ADD HL,DE

      DEC A
      CP -1
      JP NZ,msgloop

      LD DE,&D6C7
```

```
      CALL directmini

      RET

.adresse_messages
          ;"01234567890123456789012345" 25 caracteres + 1 a zero
      DEFB "         EMPTY           ":DB 0 ;0
      DEFB "        OBSTACLE         ":DB 0 ;1
      DEFB " VIRUS *BDCIRON*  TYPE # ":DB 0 ;2
      DEFB " VIRUS *PLISSKEN* TYPE # ":DB 0 ;3
      DEFB " VIRUS *DECKARD*  TYPE # ":DB 0 ;4
      DEFB " VIRUS *XTRABET*  TYPE # ":DB 0 ;5
      DEFB " VIRUS *KALIDOR*  TYPE # ":DB 0 ;6
      DEFB " VIRUS *ZISQUIER* TYPE # ":DB 0 ;7
      DEFB " VIRUS *BDCIRON*  TYPE $ ":DB 0 ;8
      DEFB " VIRUS *PLISSKEN* TYPE $ ":DB 0 ;9
      DEFB " VIRUS *DECKARD*  TYPE $ ":DB 0 ;10
      DEFB " VIRUS *XTRABET*  TYPE $ ":DB 0 ;11
      DEFB " VIRUS *KALIDOR*  TYPE $ ":DB 0 ;12
      DEFB " VIRUS *ZISQUIER* TYPE $ ":DB 0 ;13
      DEFB "WARNING, IT'S NOT A VIRUS":DB 0 ;14
      DEFB "VIRUS SELECTED, MOVE NOW?":DB 0 ;15
      DEFB "      ILLEGAL MOVE       ":DB 0 ;16
      DEFB "WELL DONE VIRUS DESTROYED":DB 0 ;17
      DEFB "    VIRUS UNSELECTED     ":DB 0 ;18
      DEFB "                         ":DB 0 ;19
      DEFB "  WAIT FOR LEVEL RELOAD  ":DB 0 ;20
      DEFB "       GAME ABORDED      ":DB 0 ;21
      DEFB "   WELCOME TO VIRUSDOG   ":DB 0 ;22
      DEFB "***** LEVEL FINISHED ****":DB 0 ;23
      DEFB "      SCORES LOADED      ":DB 0 ;24
      DEFB "      SCORES SAVED       ":DB 0 ;25
      DEFB "      SCORES RESET       ":DB 0 ;26

;syntax |#
.SCORECLEAN
      LD HL,ADRSCORE
      LD DE,ADRSCORE+1
      LD BC,150   ;(2 scores + 1 rank)*50 niveaux
      LD (HL),0
      LDIR
      ;lvlunlock
      LD A,4
      LD (ADRSCORE+151),A
      RET


;|STATS
.STATS
      LD IX,stats_texte
      LD DE,&C20F ;affichage ecran
      LD A,11     ;nombre de ligne a afficher

.stats_aff_3
      PUSH AF
      PUSH DE     ; on sauvegarde l'adresse video de la premiere ligne de sprites a
afficher
      LD A,17     ;nombre de colonne a afficher

.stats_aff_2
      PUSH AF

      LD A,(IX+&00)
      INC IX

      LD L,A
```

```
      LD H,0
      ADD HL,HL ; x2
      ADD HL,HL ; x4
      ADD HL,HL ; x8
      ADD HL,HL ; x16 (largeur 2 octets x 8 lignes pour un tile en 8x8)

      LD BC,TILESFONTE-&200   ;(evite une soustraction sur A avec
32caracteres*2octets*8lignes)
      ADD HL,BC

      PUSH DE
      CALL stats_aff_TILES_1
      POP DE

.stats_aff_1
      INC DE      ;Case suivante (decalage de 2 octets sur la droite)
      INC DE

      POP AF
      DEC A
      JR NZ,stats_aff_2

      POP HL
      LD BC,&0040 ;&50 = normal - R1=32 alors &40
      ADD HL,BC
      EX DE,HL    ;Ligne suivante
      POP AF
      DEC A
      JR NZ,stats_aff_3

      RET

;syntax |tile8,adresse_memoire,adresse_ecran
.TILE8
      ;lecture valeur adresse_memoire
      LD H,(IX+3)
      LD L,(IX+2)

      ;lecture valeur adresse_ecran
      LD D,(IX+1)
      LD E,(IX+0)

.stats_aff_TILES_1
        LD A,8          ;Hauteur_du_tile
.stats_aff_TILES_2
        LDI
        LDI             ;8 pixels = 2 octets en MODE 1
        EX DE,HL
        LD BC,&07FE     ;&800 - 2 octets (largeur ecran - largeur tiles)
        ADD HL,BC
        JR NC,stats_aff_TILES_3
        LD BC,&C040     ;R1=40 alors &C050 (normal) - R1=32 alors &C040 (reduit)
        ADD HL,BC
.stats_aff_TILES_3
        EX DE,HL
        DEC A
        JR NZ,stats_aff_TILES_2
        RET

;syntax |CLS,adresse_ecran,numero
.CLS
;&5D10 effacement,1
;&5D94 effacement,2
;&5E18 effacement,3
;&5E9C effacement,4
;&5F20 effacement,5
```

```
;&5FA4 effacement,6
      LD A,(IX+0)
      LD HL,&5D10-132
.clsloop
      LD DE,132
      ADD HL,DE

      DEC A
      CP -1
      JP NZ,clsloop

      ;lecture valeur adresse_ecran
      LD D,(IX+3)
      LD E,(IX+2)

      ;ecriture valeur hauteur
      LD A,22
      LD (cible_hauteur+1),A
      ;ecriture valeur largeur
      LD A,6
      LD (cible_largeur+1),A

      ;lecture valeur adresse_sprite
      CALL cible_hauteur
      RET

org #9F00
.stats_texte
DEFB  "%%%%%%%%%%%%%%%%"
DEFB  "% *L*E*V*E*L*S* %"
DEFB  "%+++++++++++++++%"
DEFB  "%1-PERFECT % 00 %"
DEFB  "%++++++++++%++++%"
DEFB  "%2-GOOD    % 00 %"
DEFB  "%++++++++++%++++%"
DEFB  "%3-BAD     % 00 %"
DEFB  "%++++++++++%++++%"
DEFB  "%X-NOTHING % 00 %"
DEFB  "%++++++++++%++++%"
DEFB    0
```

B) main program in BASIC

INGAME.BAS (12 Ko)

In line 20, the variable **k7** is egal at **one** for desactivate the option to load and save scores (No extra load for respect the rules of the competition)

For gain place in memory, the BASIC is reducing (Tip) :
1 - saved file in ASCII (SAVE"INGAME.BAS",A).
2 - reload file : POKE &AC00,&FF:LOAD"INGAME.BAS":SAVE"INGAME.BAS"

```
1 'Kukulcan 2016
10 CALL &BB4E:POKE &BDEE,&C9:KEY DEF 66,0,0,0,0:OPENOUT"D":MEMORY &469F:CLOSEOUT:MODE
1:BORDER 14:INK 0,0:INK 1,3:INK 2,14:INK 3,26:PEN 1:OUT &BC00,6:OUT &BD00,0
20 DEFINT A-L,N-Z:DEFREAL M:k7=1
30
ML=&77F0:MH=&A000:MR=&67E0:MC=MR+(4*3*11):MV=MC+(10*2*11):MF=MV+(14*6*22):MP=MF+(59*2
*8):MS=&9F00:DIM D(9,9):DIM R(4)
40 DIMMD(81):FORI=0TO8:MD(I)=&F845+(I*6):MD(I+9)=&E905+(I*6):MD(I+18)=&D9C5+
(I*6):MD(I+27)=&CA85+(I*6):MD(I+36)=&FB05+(I*6):MD(I+45)=&EBC5+(I*6):MD(I+54)=&DC85+
(I*6):MD(I+63)=&CD45+(I*6):MD(I+72)=&FDC5+(I*6):NEXT
50 LOAD"!ingame.pic":CALL &846A:LOAD"!ingame.pak":CALL &82A9:CALL &9000:zic=0:fx=0:|
FX,fx:GOSUB 1790:|OUT32:|SCORECLEAN
```

```
60 DIM MM(5):MM(1)=&FA94-6:MM(2)=&EB54-6:MM(3)=&DC14-6:MM(4)=&CCD4-6:MM(5)=&FD54-
6:i=200:GOSUB 1610:lvl=0:lvlunlock=4:lvlmax=49:GOSUB 1620:GOSUB 1630:GOSUB 220
70 'MENU
80 S=1:GOSUB1500:GOSUB1640:|MENU:|MSG,22:cc=1:cold=cc:GOSUB180
90 IF(INKEY(0)=0ORJOY(0)=1)THENcc=cc-1:GOSUB180
100 IF(INKEY(2)=0ORJOY(0)=2)THENcc=cc+1:GOSUB180
110 IF(INKEY(8)=0ORJOY(0)=4)THENlvl=lvl-1:GOSUB220
120 IF(INKEY(1)=0ORJOY(0)=8)THENlvl=lvl+1:GOSUB220
130 IF(INKEY(47)=0 OR JOY(0)=16)THEN ON cc GOTO 330,260,270,280,290
140 IF(k7=0 AND INKEY(36)=128)THEN LOAD"!virusdog.sav":lvlunlock=PEEK(MH+151):GOSUB
1630:|MSG,24:GOSUB 1770:GOSUB 220:CALL &BB03:CALL &BB06:GOTO 70:' L
150 IF(k7=0 AND INKEY(60)=128)THEN SAVE"!virusdog.sav",b,MH,152:|MSG,25:GOSUB
1770:GOSUB 220:CALL &BB03:CALL &BB06:GOTO 70:' S
160 IF INKEY(50)=128 THEN |SCORECLEAN:lvlunlock=PEEK(MH+151):GOSUB 1630:|MSG,26:GOSUB
1770:GOSUB 220:CALL &BB03:CALL &BB06:GOTO 70:' R
170 GOSUB1820:GOTO90
180 'move choice
190 z=0:GOSUB 1780:IF cc<1 THEN cc=5
200 IF cc>5 THEN cc=1
210 |SPR,MM(cold),MF,2,8:|SPR,MM(cold)+34,MF,2,8:cold=cc:|
SPR,MM(cc),MF+(30*2*8),2,8:|SPR,MM(cc)+34,MF+(28*2*8),2,8:GOSUB1750:RETURN
220 'move level
230 z=0:GOSUB1780:IFlvl<0THENlvl=lvlunlock
240 IFlvl>lvlunlockTHENlvl=0
250 i=7:GOSUB1610:GOSUB1620:GOSUB1630:i=PEEK(ML+(lvl*82)+81):GOSUB1710:i=(PEEK(MH+
(lvl*3))*100)+PEEK(MH+(lvl*3)+1):GOSUB1720:grank=PEEK(MH+(lvl*3)+2):GOSUB1730:RETURN
260 s=2:GOSUB1500:|AFFHIST:GOSUB1640:CALL&BB03:CALL&BB06:GOTO70:'The Story
270 s=2:GOSUB1500:|AFFCTRL:GOSUB1640:CALL&BB03:CALL&BB06:GOTO70:'Controls
280 s=2:GOSUB1500:|AFFCREDITS:GOSUB1640:CALL&BB03:CALL&BB06:GOTO70:'Credits
290 s=2:GOSUB 1500:OUT &BC00,1:FOR i=32 TO 0 STEP-1:CALL &BD19:OUT &BD00,i:NEXT:|
AFFRESULT:|STATS:GOSUB 300:GOSUB 1640:GOSUB 1770:OUT &BC00,1:FOR i=0 TO 32:CALL
&BD19:OUT &BD00,i:NEXT:CALL &BB03:CALL &BB06:GOTO 70:'Results
300 MA=MH+2:ME=&D515:b=0:FOR i=0 TO 49:a=PEEK(MA+(i*3)):IF a<>1 THEN |TILE8,&7268,ME
310 b=b+1:IF b=10 THEN b=0:ME=ME+46 ELSE ME=ME+2
320 NEXT:RETURN
330 'PLAY
340 s=2:GOSUB1500:gmoves=0:i=gmoves:GOSUB1690:gtarget=PEEK(ML+
(lvl*82)+81):i=gtarget:GOSUB1710:grecord=(PEEK(MH+(lvl*3))*100)+PEEK(MH+
(lvl*3)+1):i=grecord:GOSUB1720:grank=PEEK(MH+(lvl*3)+2):GOSUB1730
350 tdv=0:Mbase=ML+(lvl*82):RESTORE 1580:FOR i=0 TO 80:READ b:a=PEEK(Mbase+b):IF a>1
THEN tdv=tdv+1
360 y=b\ 9:x=b-(y*9):d(x,y)=a:|SPR,MD(b),MV+(a*6*22),6,22:NEXT:tcv=tdv:GOSUB
1530:posx=4:posy=4:GOSUB 500
370 'move cible
380 IF(INKEY(0)=0ORJOY(0)=1)THENGOSUB460
390 IF(INKEY(2)=0ORJOY(0)=2)THENGOSUB470
400 IF(INKEY(8)=0ORJOY(0)=4)THENGOSUB480
410 IF(INKEY(1)=0ORJOY(0)=8)THENGOSUB490
420 IF(INKEY(47)=0ORJOY(0)=16)THENGOSUB1750:GOTO550
430 IFINKEY(50)=0THENz=8:GOSUB1780:|
MSG,20:OUT&BC00,2:FORa=42TO62STEP2:OUT&BD00,a:CALL&BD19:NEXT:FORa=0TO42STEP2:OUT&BD00
,a:CALL&BD19:NEXT:GOTO330:'R
440 IF(INKEY(67)=0ORINKEY(69)=0)THENz=7:GOSUB1780:|MSG,21:GOTO70:'A or Q
450 GOSUB1820:GOTO370
460 IFposy-1>=0THENGOSUB520:posy=posy-1:GOSUB500:RETURNELSERETURN:'ð
470 IFposy+1<=8THENGOSUB520:posy=posy+1:GOSUB500:RETURNELSERETURN:'ñ
480 IFposx-1>=0THENGOSUB520:posx=posx-1:GOSUB500:RETURNELSERETURN:'ò
490 IFposx+1<=8THENGOSUB520:posx=posx+1:GOSUB500:RETURNELSERETURN:'ó
500 'aff case+aff cible
510 a=d(posx1,posy1):|SPR,MD(posx1+(posy1*9)),MV+(a*6*22),6,22:a=d(posx,posy):|
MSG,a:|CIBLE,MD(posx+(posy*9)),MP,6,22:i=3:GOSUB1610:RETURN
520 posx1=posx:posy1=posy:RETURN:'Save Pos
530 a=d(posx1,posy1):|SPR,MD(posx+(posy*9)),MV+(a*6*22),6,22:RETURN:'Restaure case
540 |CIBLE,MD(posx+(posy*9)),MP+(b*6*22),6,22:RETURN:'cible move choix
550 'virus select ?
560 a=d(posx,posy):IFa<2THEN|MSG,14:GOTO370
```

```
570 z=2:GOSUB1780
580 'aff direction move
590 GOSUB520:GOSUB530:i=0
600 'possible ð ?
610 IFposy=0THEN630
620 IFd(posx,posy-1)=0THENi=i+1:b=1:GOSUB540
630 'possible ó ?
640 IFposx=8THEN660
650 IFd(posx+1,posy)=0THENi=i+1:b=2:GOSUB540
660 'possible ñ ?
670 IFposy=8THEN690
680 IFd(posx,posy+1)=0THENi=i+1:b=3:GOSUB540
690 'possible ò ?
700 IFposx=0THEN720
710 IFd(posx-1,posy)=0THENi=i+1:b=4:GOSUB540
720 'no move possible ?
730 IFi=0THENGOTO370
740 'Virus wait for move
750 |MSG,15
760 'move2
770 IF(INKEY(47)=0ORJOY(0)=16)THENz=3:GOSUB1780:|
MSG,18:GOSUB530:b=0:GOSUB540:GOSUB1750:GOTO370
780 IF(INKEY(0)=0ORJOY(0)=1)THENdx=0:dy=-1:GOSUB1750:GOTO830
790 IF(INKEY(2)=0ORJOY(0)=2)THENdx=0:dy=1:GOSUB1750:GOTO830
800 IF(INKEY(8)=0ORJOY(0)=4)THENdx=-1:dy=0:GOSUB1750:GOTO870
810 IF(INKEY(1)=0ORJOY(0)=8)THENdx=1:dy=0:GOSUB1750:GOTO870
820 GOSUB1820:GOTO760
830 'Move Vertical
840 a=0:WHILEdy<>0:GOSUB1010:WEND:IFa=1THENgmoves=gmoves+1:i=gmoves:GOSUB1690
850 GOSUB520:GOSUB500:IFtcv=0THEN910
860 GOTO370
870 'Move Horizontal
880 a=0:WHILEdx<>0:GOSUB1250:WEND:IFa=1THENgmoves=gmoves+1:i=gmoves:GOSUB1690
890 GOSUB520:GOSUB500:IFtcv=0THEN910
900 GOTO370
910 'end lvl
920 |SPR,MD(posx+(posy*9)),MV,6,22:'del cible
930 IF(grecord=0ORgmoves<grecord)THENa=gmoves\ 100:b=gmoves-(a*100):POKE(MH+
(lvl*3)),a:POKE(MH+(lvl*3)+1),b:i=gmoves:GOSUB1720
940 IF gmoves<=gtarget THEN grank=1:POKE(MH+(lvl*3)+2),grank:GOSUB 1730
950 IF(gmoves<=gtarget+7 AND(grank=0 OR grank=3))THEN grank=2:POKE(MH+
(lvl*3)+2),grank:GOSUB 1730
960 IF(gmoves>gtarget+7 AND grank=0)THEN grank=3:POKE(MH+(lvl*3)+2),grank:GOSUB 1730
970 GOSUB 1770:|MSG,23:FOR a=0 TO 20:FOR b=60 TO 90 STEP 1.5:OUT
&7F00,b:NEXT:NEXT:CALL &BB03:CALL &BB06:lvl=lvl+1:IF lvl>lvlmax THEN
lvl=0:lvlunlock=49
980 IF lvl>lvlunlock THEN lvlunlock=lvl
990 IFlvlunlock>lvlmaxTHENlvlunlock=lvlmax
1000 POKE MH+151,lvlunlock:GOSUB 1620:GOSUB 1630:GOTO 330
1010 'gestion move vertical
1020 IFposy+dy<0THENdy=0:z=1:GOSUB1780:RETURN:'bord
1030 IFposy+dy>8THENdy=0:z=1:GOSUB1780:RETURN:'bord
1040 actuel=d(posx,posy):suivant=d(posx,posy+dy)
1050 IF suivant>0 THEN dy=0:RETURN:'impossible
1060 'no problem=move
1070 a=1:d(posx,posy)=0:|SPR,MD(posx+(posy*9)),MV,6,22:d(posx,posy+dy)=actuel:|
SPR,MD(posx+((posy+dy)*9)),MV+(actuel*6*22),6,22:posy=posy+dy
1080 'choc ?
1090 IFposy+dy<0THENdy=0:z=1:GOSUB1780:RETURN:'bord
1100 IFposy+dy>8THENdy=0:z=1:GOSUB1780:RETURN:'bord
1110 suivant=d(posx,posy+dy)
1120 IFsuivant=0THENRETURN:'rien touche et move possible
1130 'choc
1140 IFsuivant=1THENdy=0:z=1:GOSUB1780:RETURN:'paf un obstacle
1150 IFsuivant=actuelTHENz=5:GOSUB1780:GOSUB1480:dy=0:RETURN:'2 virus
identique=destruction
```

```
1160 'definir couleur
1170 IFsuivant>7THENcs=2ELSEcs=1
1180 IFactuel>7THENca=2ELSEca=1
1190 IF cs=ca THEN dy=0:RETURN:'sameColor=noEffect
1200 'colorDifferent=swap
1210 z=4:GOSUB1780
1220 IFca=1THENb=6ELSEb=-6
1230 IFcs=1THENc=6ELSEc=-6
1240 d(posx,posy)=actuel+b:d(posx,posy+dy)=suivant+c:|SPR,MD(posx+(posy*9)),MV+
((actuel+b)*6*22),6,22:|SPR,MD(posx+((posy+dy)*9)),MV+
((suivant+c)*6*22),6,22:dy=0:RETURN
1250 'gestion move horizontal
1260 IFposx+dx<0THENz=1:GOSUB1780:dx=0:RETURN:'bord
1270 IFposx+dx>8THENz=1:GOSUB1780:dx=0:RETURN:'bord
1280 actuel=d(posx,posy):suivant=d(posx+dx,posy)
1290 IF suivant>0 THEN dx=0:RETURN:'impossible
1300 'no problem=move
1310 a=1:d(posx,posy)=0:|SPR,MD(posx+(posy*9)),MV,6,22:d(posx+dx,posy)=actuel:|
SPR,MD(posx+dx+(posy*9)),MV+(actuel*6*22),6,22:posx=posx+dx
1320 'choc ?
1330 IF posx+dx<0 THEN dx=0:z=1:GOSUB 1780:RETURN:'bord
1340 IFposx+dx>8THENdx=0:z=1:GOSUB1780:RETURN:'bord
1350 suivant=d(posx+dx,posy)
1360 IFsuivant=0THENRETURN:'rien touche et move possible
1370 'choc
1380 IFsuivant=1THENz=1:GOSUB1780:dx=0:RETURN:'paf un obstacle
1390 IFsuivant=actuelTHENz=5:GOSUB1780:GOSUB1480:dx=0:RETURN:'2 virus
identique=destruction
1400 'definir couleur
1410 IFsuivant>7THENcs=2ELSEcs=1
1420 IFactuel>7THENca=2ELSEca=1
1430 IF cs=ca THEN dx=0:RETURN:'sameColor=noEffect
1440 'colorDifferent=swap
1450 z=4:GOSUB 1780:IF ca=1 THEN b=6 ELSE b=-6
1460 IF cs=1 THEN c=6 ELSE c=-6
1470 d(posx,posy)=actuel+b:d(posx+dx,posy)=suivant+c:|SPR,MD(posx+(posy*9)),MV+
((actuel+b)*6*22),6,22:|SPR,MD(posx+dx+(posy*9)),MV+
((suivant+c)*6*22),6,22:dx=0:RETURN
1480 '-2Virus
1490 d(posx,posy)=0:d(posx+dx,posy+dy)=0:tcv=tcv-2:GOSUB 1530:FOR i=0 TO 5:|
CLS,MD(posx+(posy*9)),i:|CLS,MD(posx+dx+((posy+dy)*9)),i:NEXT:RETURN
1500 'cleanscreen
1510 IFS=1THENRESTORE1580ELSERESTORE1600
1520 FORi=0TO80:READa:|SPR,MD(a),MV,6,22:NEXT:RETURN
1530 'barometre
1540 IF tdv=tcv THEN PLOT 580,29,1:DRAW 580-398,29,1:RETURN:'full
1550 IFtcv=0THENPLOT580,29,2:DRAW580-398,29,2:RETURN:'goal
1560 PLOT 580,29,2:DRAW 580-398,29,2:PLOT 182,29,1:DRAW 182+
(tcv/tdv*398),29,1:RETURN:'virus-
1580 DATA
0,1,2,3,4,5,6,7,8,17,26,35,44,53,62,71,80,79,78,77,76,75,74,73,72,63,54,45,36,27,18,9
,10,11,12,13,14,15,16,25,34,43,52,61,70,69,68,67,66,65,64,55,46,37,28,19,20,21,22,23,
24,33,42,51,60,59,58,57,56,47,38,29,30,31,32,41,50,49,48,39,40
1600 DATA
40,39,48,49,50,41,32,31,30,29,38,47,56,57,58,59,60,51,42,33,24,23,22,21,20,19,28,37,4
6,55,64,65,66,67,68,69,70,61,52,43,34,25,16,15,14,13,12,11,10,9,18,27,36,45,54,63,72,
73,74,75,76,77,78,79,80,71,62,53,44,35,26,17,8,7,6,5,4,3,2,1,0
1610 FORj=0TOi:CALL&BD19:NEXT:RETURN:'tempo
1620 i=lvl+1:GOSUB1650:M=&E785:GOSUB1660:RETURN:'aff lvl
1630 i=lvlunlock+1:GOSUB1650:M=&E78D:GOSUB1660:RETURN:'aff lvlunlock
1640 RANDOMIZE TIME:I=2+INT(RND*12):|SPR,MD(9),MV+(I*6*22),6,22:|SPR,MD(17),MV+
(I*6*22),6,22:RETURN:'VirusMENU
1650 a=i\ 10:b=i-(a*10):RETURN:'2digits
1660 |SPR,M,MC+(a*2*11),2,11:|SPR,M+2,MC+(b*2*11),2,11:RETURN:'affLEVEL,UNLOCK
1670 RETURN
1680 a=i\ 100:b=(i-(a*100))\ 10:c=i-(a*100)-(b*10):RETURN:'3digits
```

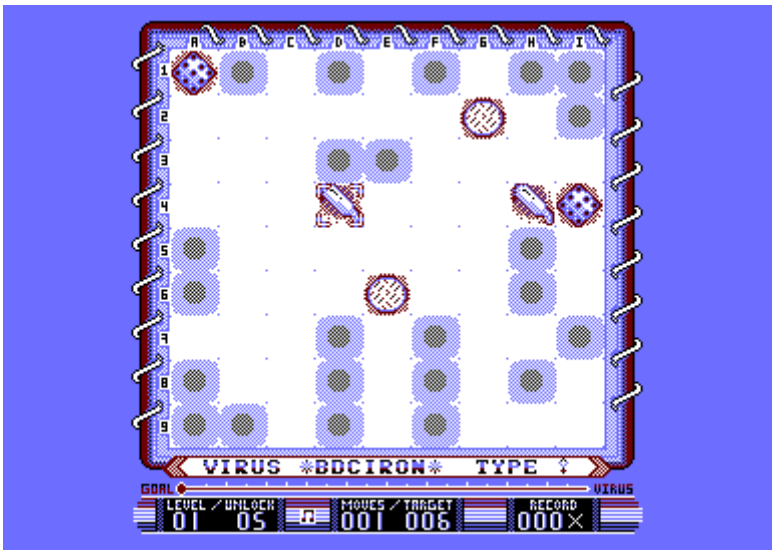
```
1690 IFgmoves>999THENgmoves=999
1700 GOSUB 1680:M=&E79A:GOSUB 1740:RETURN:'affMOVES
1710 GOSUB 1680:M=&E7A2:GOSUB 1740:RETURN:'affTARGET
1720 GOSUB 1680:M=&E7B0:GOSUB 1740:RETURN:'affRECORD
1730 |SPR,&E7B6,MR+(grank*3*11),3,11:RETURN:'affRANK
1740 |SPR,M,MC+(a*2*11),2,11:|SPR,M+2,MC+(b*2*11),2,11:|SPR,M+4,MC+
(c*2*11),2,11:RETURN:'affMOVES,TARGET,RECORD
1750 'relacher
1760 IF(INKEY(47)=-1ANDINKEY(0)=-1ANDINKEY(2)=-1ANDINKEY(8)=-1ANDINKEY(1)=-
1ANDJOY(0)=0)THENRETURNELSE1760
1770 FORi=0TO3:R(i)=0:NEXT:FORi=0TO49:a=PEEK(MH+(i*3)+2):R(a)=R(a)+1:NEXT:a=R(0)\
10:b=R(0)-(a*10):POKEMS+166,48+a:POKEMS+167,48+b:FORi=1TO3:a=R(i)\ 10:b=R(i)-
(a*10):POKEMS+30+(i*34),48+a:POKEMS+31+(i*34),48+b:NEXT:|STATS:RETURN
1780 IF fx=1 THEN CALL &4700,z:RETURN ELSE RETURN:'sound(0a8)
1790 'music(9a11)
1800 IFfx=0THENzic=zic+1:IFzic>2THENzic=0
1810 CALL&4700,9+zic:RETURN
1820 'changeFX/ZIC
1830 IF (INKEY(29)=0 OR INKEY(38)=0) THEN GOSUB 1850
1840 RETURN
1850 IFfx=0THENfx=1:z=2:GOSUB1780ELSEfx=0:GOSUB1790
1860 |FX,fx
1870 IF (INKEY(29)=-1 AND INKEY(38)=-1) THEN RETURN ELSE 1870
```

INGAME - MENU

Up or down (arrow keys or joystick) to select an option. Space bar or Fire button to validate.
Left or right (arrow keys or joystick) to choose a level among previously unlocked ones.
M key (Qwerty or Azerty) to swap between music or SFXFx.
The game contains 3 different musics.

CTRL+R = Reset high score table
CTRL + S = Save scores (only on the gift version)
CTRL + L = Load scores (only on the gift version)

INGAME - PLAY



On first level, the target is to finish it in 6 moves.

VIRUS

6 viruses exist with two genre.

If a virus crashes other ones without the same genre, the result is an inversion of genre.

Move on a virus to see the name and the genre (male of female)

To select a virus, move target above it and press Space Bar or Fire button on joystick. The possible moves appear, choose a direction to move or press Space bar or Fire button again to cancel.

VIRUS EXPLOSION

Viruses explode only if the virus moved crash other identical virus.

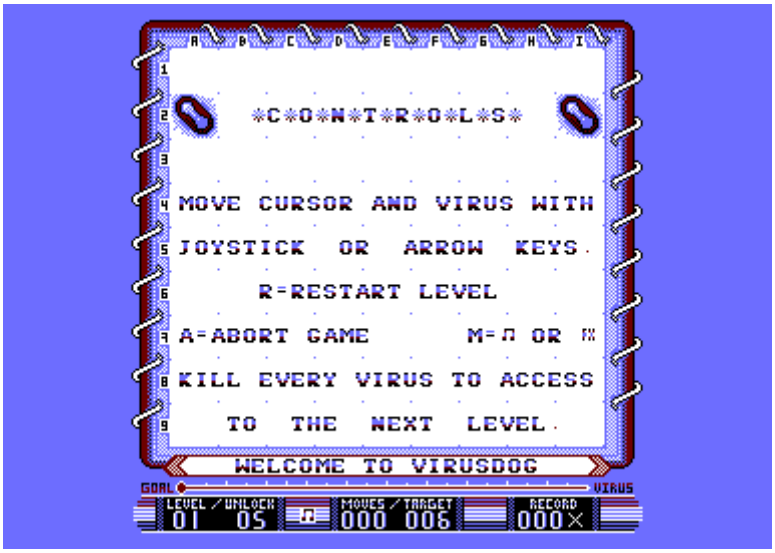
RECORD

Rang 1 : PERFECT, if you finish the level with less or exactly the number of moves indicated.
Rang 2 : Finish by exceeding the number of moves indicated in the limit of 7 more moves.
Rang 3: Finish the level exceeding 7 more moves.

INGAME - THE STORY

The dog is named "W. CAMELOT" in hommage of the spanish game Camelot Warriors :-)

INGAME - CONTROLS



INGAME - CREDITS

Musics under Creative Common licence : http://tj.gpa.free.fr/html/product/music2.htm

Others testers not credited in the game : Megachur, Snake PLISSKEN.

INGAME - RESULTS

Each "?" disappear after you win a level with a perfect. Finish the 50 levels for see the little hidden picture.

TOOLS

EMULATORS :

WinAPE 2.0 Alpha 18 (Richard WILSON)
Sugarbox (Lone)
Caprice Forever (Fredouille)

TOOLS ON PC :

ManageDsk v0.20 (Ludovic DEPLANQUE)
ConvImgCpc v0.17 (Ludovic DEPLANQUE)
2CDT (Kevin THACKER)

TOOLS ON AMSTRAD CPC :

Compression : Zenith II (Xavier NICOLAY)
Painting : The Advanced OCP Art Studio (Rainbird Software)