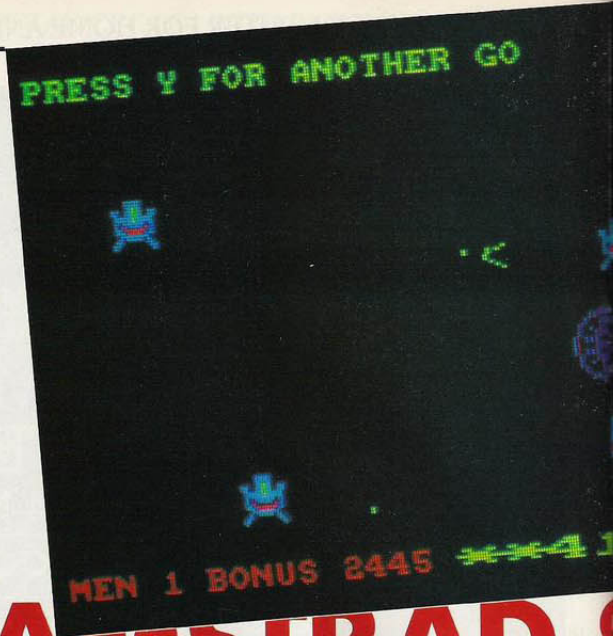
HERE'S A colourful asteroids-style game for the Amstrad CPC-464. While using Mode 1 it manages to give a whole range of colours and provides Mode 0 size text printing on the screen using a Rom indirection. Get yourself a high score by shooting the eggs and aliens but watch the bonus. Quick shots will score well but be slow and the bonus will turn against you!

The game features 44 sprites of different sizes with up to 25 on the screen at any one time. These are controlled by a machine-code routine that can easily be adapted for your own purposes. In fact, most of the action is controlled by machine-code making the main Basic game loop very short and speedy.

The main program appears in listing 1 and may be typed in directly. It should be saved — with Goto 20000 — before running as any mistakes in the data may cause the system to crash. The machine-code is in five routines, three of which are listed so that you can used them in your own programs. The first one — listing 2 — is for wide printing mimicking mode 0 printing.

In fact, it will work in any mode with suitable changes to the final few bytes to reset the cursor position. It works by altering the high byte of the Rom indirection at &BDD5 using Poke so that instead of pointing to the Rom routine at &134A, it points to our routine at &A34A.

We can then use the usual Print command including the Print Using format. The colour of the printing is given by the pen mask — changing this can give striped writing or mixed colours. Note that the indirection supplies the screen position as a physical



AMSTRAD

Listing 1.

```
10 REM ***********************************
20 REM *            SPACE EGGS           *
30 REM *                by               *
40 REM *        Christopher Leigh        *
50 REM ***********************************
100 GOTO 3000
200 LOCATE 12,25:PRINT b%:LOCATE 10,25:POKE &BDD5,&A3:
PRINT USING "******";score:POKE &BDD5,&13:RETURN
1000 REM ** main loop **
1010 IF INKEY(27)=0 THEN head=(head+1)AND 7:|ERASE,&A5
00:POKE &A506,17*head+&65:POKE &A500,1
1020 IF INKEY(35)=0 THEN head=(head-1)AND 7:|ERASE,&A5
00:POKE &A506,17*head+&65:POKE &A500,1
1030 g%=(head=5 OR head=6 OR head=7)-(head=1 OR head=2
 OR head=3):h%=(head=3 OR head=4 OR head=5)-(head=7 OR
 head=0 OR head=1)
1040 IF INKEY(60)=0 THEN POKE &A503,(PEEK(&A503)+h%)AN
D 255:POKE &A504,(PEEK(&A504)+g%)AND 255
1050 IF INKEY(47)=0 THEN SOUND 130,10,10,15,1,1:CALL &
A803,g%,h%
1060 IF INKEY(63)=0 THEN FOR N=50 TO 10 STEP-2:SOUND 1
30,n,5,15,1,1:NEXT:|ERASE,&A500:POKE &A500,1:POKE &A50
1,RND*79:POKE &A502,RND*47
1070 FOR n=0 TO p%:NEXT:|MOVE:b%=b%-5:CALL &A700,@a%:I
F a%=0 THEN 1010 ELSE IF a%=4 THEN 2000
1080 IF a%=3 THEN SOUND 133,800,150,15,2,1:score=score
+20:GOTO 1010
1090 IF a%=2 THEN SOUND 133,10,100,15,2,1:score=score+
50:GOTO 1010
1100 IF a%=1 THEN SOUND 133,10,150,15,2,2,30:score=sco
re+100:e%=e%-1:IF e%<>0 THEN 1010 ELSE score=score+b%:
POKE &A503,RND*3:POKE &A504,RND*3:GOTO 4000
2000 REM ** lives **
2010 BORDER 26:SOUND 135,100,150,15,2,2,15:FOR n=0 TO
10:NEXT:BORDER 0
2020 men=men-1:IF men>0 THEN DI:LOCATE 1,25:PRINT "MEN
";men:EI:|ERASE,&A500:POKE &A500,1:POKE &A501,39:POKE
&A502,24:GOTO 1000
2030 IF score>hi THEN hi=score
2040 POKE &A501,39:POKE &A502,24:POKE &A503,0:POKE &A5
04,0:POKE &A506,&65
2050 DI:PEN 3:LOCATE 1,5:PRINT "PRESS Y FOR ANOTHER GO
":PEN 2:EI
2060 IF INKEY(43)<>0 THEN 2060 ELSE head=0:men=3:score
=0:GOTO 4000
3000 REM ** start game **
3010 MODE 1:BORDER 1:INK 3,24:INK 2,21:INK 1,16:INK 0,
1:MEMORY &A2FF
3020 FOR n=&A34A TO &A37F:READ a$:POKE n,VAL("&"+a$):N
EXT
3030 DATA e5,f5,cd,1a,bc,f1,eb,cd,a5,bb,06,07,4e,c5,d5
,06,04,af,cb,21,30,02,f6,cc,cb,21,30,02,f6,33,12,13,10
,ef,d1,e5,21,00,00,19,eb,e1,23,c1,10,de,e1,24,24,24,2c
,c3,75,bb
3040 PEN 3:LOCATE 2,2:POKE &BDD5,&A3:PRINT "SPACE EGGS
 ";:POKE &BDD5,&13:PRINT "by C.J.Leigh"
3050 PEN 2:PRINT:PRINT " Due to a fault in your hyper
space     generator you find yourself in an        unch
arted region of space.  Suddenly allyour emergency sys
tems go wild and your sights are filled with strange e
gg like"
3060 PRINT "creatures.  Madly you fire your laser   ca
non but hitting an egg merely releasestwo aliens and h
itting them produces    even more seed eggs.  And all
the time  the void is filled with alien screams."
3070 PRINT "If you are not quick enough they will   ge
t you and then you will scream as the madness takes ov
er."
3080 GOSUB 10000
3090 PEN 1:LOCATE 1,20:PRINT "  Use I & P for rotate l
eft and right;         S for subspace thrusters;
         X for hyperspace jump
Space for fire"
3100 IF INKEY$<>"" THEN 3100
3110 IF INKEY$="" THEN 3110
3120 a%=0:head=0:hi=1000:men=3:score=0
3130 ENV 1,15,-1,1:ENT -1,15,10,1,15,-10,1:ENV 2,15,-1
,10:ENT 2,15,10,10
3140 EVERY 50,2 GOSUB 200
4000 REM ** new screen **
4010 BORDER 0:INK 0,0:INK 1,11:INK 2,6:INK 3,18:MODE 1
:b%=3000:e%=16:POKE &A802,0:IF score>4000 THEN p%=0 EL
SE p%=40
4020 DI:LOCATE 1,25:PEN 2:PRINT "MEN";men;"BONUS";b%:L
OCATE 18,25:POKE &BDD5,&A3:PRINT USING "******";score;
:POKE &BDD5,&13:PEN 1:PRINT " HI";hi:PEN 2:EI
4030 FOR n=100 TO 800 STEP 20:SOUND 7,n,10,15:NEXT
4040 FOR n=1 TO 8:POKE &A500+n*9,0:NEXT
4050 FOR n=9 TO 12:POKE &A500+n*9,1:NEXT
4060 FOR n=13 TO 36:POKE &A500+n*9,0:NEXT
4070 GOTO 1000
10000 REM ** machine code **
10010 FOR n=&A400 TO &A8BF:READ a$:POKE n,VAL("&"+a$)
10020 NEXT:CALL &A400:RETURN
10030 DATA 01,0e,a4,21,0a,a4,cd,d1,bc,c9,00,00,00,00,1
6,a4,c3,ec,a4,c3,20,a4,45,52,41,53,c5,4d,4f,56,c5,00
10040 DATA dd,21,00,a5,10,49,21,ca,a4,36,00,dd,7e,01,0
0,00,00,cd,79,a4,21,ca,a4,dd,7e,05,77,dd,7e,02,dd,86,0
4,f2,46,a4,c6,30,fe,30,38,02,d6,30,dd,77,02
10050 DATA dd,7e,01,dd,86,03,f2,5a,a4,c6,50,fe,50,38,0
2,d6,50,dd,77,01,dd,36,00,00,cd,79,a4,11,09,00,dd,19,d
d,7e,00,1f,38,b1,1f,30,f2,c9
10060 DATA f5,d6,50,ed,44,4f,dd,66,02,dd,5e,06,dd,56,0
7,1a,47,6f,13,f1
10070 DATA e5,f5,d5,11,00,00,f5,7c,fe,30,38,02,d6,30,c
b,3f,30,02,16,20,6f,26,00,f1,d5,29,29,29,29,e5,d1,29,2
9,19,d1,19,16,c0,5f,19,d1
10080 DATA 3e,04,f5,c5,e5,1a,fe,00,28,0c,7e,fe,00,20,0
3,dd,77,00,1a,e6,ff,77,13,23,0d,20,08,d5,af,11,50,00,e
d,52,d1,10,e8
10090 DATA e1,01,00,08,09,c1,f1,3d,20,d3,f1,e1,24,2d,2
0,a2,c9
10100 DATA d5,21,ca,a4,36,00,dd,e1,dd,36,00,00,dd,7e,0
1,c3,79,a4,00,00
11000 REM ** move data **
11010 ' spaceship & bullets
11020 DATA 01,27,18,00,00,ff,65,a6,00
11030 DATA 00,00,00,00,00,ff,60,a6,00,00,00,00,00,00,f
f,60,a6,00
11040 DATA 00,00,00,00,00,ff,60,a6,00,00,00,00,00,00,f
f,60,a6,00
11050 DATA 00,00,00,00,00,ff,60,a6,00,00,00,00,00,00,f
f,60,a6,00
11060 DATA 00,00,00,00,00,ff,60,a6,00,00,00,00,00,00,f
f,60,a6,00
11070 ' asteroids
11080 DATA 01,1f,0f,01,01,a5,ed,a6,00,01,31,0f,ff,01,f
a,ed,a6,00
11090 DATA 01,20,20,01,00,af,ed,a6,00,01,31,21,00,ff,d
7,ed,a6,00
11100 DATA 00,00,00,01,ff,ff,7e,a7,00,00,00,00,ff,ff,f
f,7e,a7,00
11110 DATA 00,00,00,00,ff,ff,7e,a7,00,00,00,00,ff,00,f
f,7e,a7,00
11120 DATA 00,00,00,ff,01,ff,7e,a7,00,00,00,00,01,ff,f
f,7e,a7,00
11130 DATA 00,00,00,00,01,ff,7e,a7,00,00,00,00,01,00,f
f,7e,a7,00
11140 DATA 00,00,00,fe,fe,0f,bf,a7,00,00,00,00,02,fe,f
0,bf,a7,00
11150 DATA 00,00,00,fe,00,a5,bf,a7,00,00,00,00,00,02,a
5,bf,a7,00
11160 DATA 00,00,00,02,02,ff,bf,a7,00,00,00,00,fe,02,f
f,bf,a7,00
11170 DATA 00,00,00,02,00,f5,bf,a7,00,00,00,00,00,fe,5
f,bf,a7,00
11180 DATA 00,00,00,fe,fe,5f,bf,a7,00,00,00,00,02,fe,f
5,bf,a7,00
11190 DATA 00,00,00,fe,00,a0,bf,a7,00,00,00,00,00,02,a
a,bf,a7,00
11200 DATA 00,00,00,02,02,a5,bf,a7,00,00,00,00,fe,02,a
a,bf,a7,00
11210 DATA 00,00,00,02,00,f0,bf,a7,00,00,00,00,00,fe,0
f,bf,a7,00
11220 ' extra
11230 DATA 02,00,09,01,00,ff,11,a8,00,00,00,0a,01,01,f
f,16,a8,00
11240 DATA 02
12000 REM ** shape data **
12010 ' bullet & spaceship
12020 DATA 01,00,66,66,00
12030 DATA 02,cc,00,33,00,00,cc,00,33,00,33,00,cc,33,0
0,cc,00
12040 DATA 02,00,88,00,88,00,44,00,44,cc,22,33,22,00,f
f,00,33
12050 DATA 02,88,11,00,11,44,22,44,22,22,44,22,44,11,8
8,11,88
12060 DATA 02,11,00,11,00,22,00,22,00,44,33,44,cc,ff,0
0,cc,00
12070 DATA 02,00,33,00,cc,33,00,cc,00,cc,00,33,00,00,c
c,00,33
12080 DATA 02,cc,00,ff,00,44,cc,44,33,22,00,22,00,11,0
0,11,00
12090 DATA 02,11,88,11,88,22,44,22,44,44,22,44,22,88,1
1,88,11
12100 DATA 02,00,33,00,ff,33,22,cc,22,00,44,00,44,00,8
8,00,88
12110 ' asteroids
12120 DATA 06,00,00,77,ee,00,00,00,33,ff,ff,cc,00,00,7
7,88,11,ee,00
12130 DATA 00,cc,33,cc,33,00,11,88,77,ee,11,88,33,00,6
6,66,00,cc
12140 DATA 66,22,66,66,44,66,66,11,ff,ff,88,66,66,11,f
f,ff,88,66
12150 DATA cc,ff,00,11,ff,33,cc,ff,00,11,ff,33,cc,11,f
f,ff,88,33
12160 DATA cc,11,ff,ff,88,33,cc,ff,88,11,ff,33,cc,ff,0
0,11,ff,33
12170 DATA 66,11,ff,ff,88,66,66,11,ff,ff,88,66,66,22,6
6,66,44,66
12180 DATA 33,00,66,66,00,cc,11,88,77,ee,11,88,00,cc,3
3,cc,33,00
12190 DATA 00,77,00,11,ee,00,00,33,ff,ff,cc,00,00,00,7
7,ee,00,00
12200 DATA 04,00,70,e0,00,00,70,e0,00,00,71,e8,00,e0,7
1,e0,70
12210 DATA e0,71,e0,70,70,f1,f8,e0,70,f1,f0,e0,61,f0,f
0,68
12220 DATA 61,70,e1,68,70,0f,0f,e0,70,87,1e,e0,10,f0,f
0,80
12230 DATA 30,70,e0,c0,70,00,00,e0,e0,00,00,70,c0,00,0
0,30
12240 DATA 02,33,cc,77,ee,ff,ff,ff,ff,ff,ff,ff,ff,77,e
e,33,cc
13000 REM ** check collision **
13010 DATA af,21,ca,a4,77,eb,77,dd,21,00,a5,11,09,00,0
e,04,dd,be,00,20,02,71,c9,dd,19,06,24,af,dd,be,00,20,1
e,3a,01,a5,dd,be,01,28,06,3c,dd,be,01,20,10,3a,02,a5,d
d,be,02,28,06,3c,dd,be,02,20,02,71,c9,dd,19,10,d8
13020 DATA dd,21,09,a5,06,08,af,dd,be,00,c4,78,a8,dd,1
9,10,f5
13030 DATA fd,21,75,a5,0d,06,04,3e,66,cd,49,a8,06,08,0
d,cd,49,a8,06,10,0d,3e,66,dd,be,00,20,04,71,c3,78,a8,d
d,19,10,f1,c9
13040 DATA dd,be,00,20,21,71,00,cd,78,a8,fd,36,00,01,f
d,36,09,01,dd,7e,01,fd,77,01,fd,77,0a,dd,7e,02,fd,77,0
2,fd,77,0b,f1,c9,fd,19,fd,19,dd,19,10,d2,c9
13050 DATA c5,d5,e5,cd,f4,a4,e1,d1,c1,c9
13060 REM ** fire bullets **
13070 DATA 00,dd,4e,00,dd,46,02,c5,3a,02,a8,c6,09,fe,4
8,38,01,af,32,02,a8,c6,09,5f,16,a5,d5,cd,ec,a4,e1,36,0
1,eb,13,c1,cb,20,cb,21,21,01,a5,7e,01,12,13,23,7e,00,1
2,13,23,7e,01,12,13,23,7e,00,12,c9
20000 SPEED WRITE 1:SAVE "space eggs"
20010 SPEED WRITE 0:SAVE "!space eggs"
20020 PRINT CHR$(7)"Rewind the tape":CAT
```

Christopher Leigh presents a new version of a favourite shoot-'em-up using sprites in glorious colour.

SPACE EGGS

position — top left = 0, 0 — whilst the cursor must be reset to the logical position — top left = 1, 1 — hence the extra increment instructions.

The second and third routines control the sprites and are called using Resident System Extension (RSX) commands. Move and Erase must be preceded by the elongated colon — shift @ — and 1Erase must be followed by a comma and its parameter which is the address of the first byte of the move data for the sprite.

The screen is 80 bytes wide and the sprite routines divide it into 50 half lines high, so that each sprite unit is a quarter of a Mode 1 character. These routines can cope with sprites of any size and — with slight alterations — of any shape. All our sprites will be set in a square sprite shape definition, but since zero bytes are not written to the screen — making the sprite transparent — the sprite can be any shape within that framework.

As written the procedure allows full wrap-around, adjusting for sprites being partly off a screen edge. Again fairly simple alterations will allow sprites to bounce.

1Erase simply erases a sprite and turns it off. 1Move works by calculating the old sprite position and then writing it with an ink mask of zero to rub it out, then calculating the new position and writing with the ink mask given in the move data.

This ink mask can be set to produce pure colours or colour mixtures for a whole sprite. The new position is calculated by adding the speed components to the old position and then ensuring it is on the screen. The move data also includes the address of the shape data for a particular sprite and a collision byte. This collision byte is the last non zero byte read off the screen when writing the sprite. This allows us to know if it is on top of anything and also what it is n top of.

The move data consists of nine bytes formatted thus: on/off flag, right position, down position, right speed, down speed, ink mask, shape address low, shape address high, collision byte. 1Move will, in fact, move every sprite, whose on/off flag is one, in the block of move data and the routine is stopped by a value of two. The shape data address can be altered to change the shape of a sprite during the game as is done to rotate your space ship.

The first byte of the shape data is the size of the sprite in quarters. The rest of the shape data comprises bytes made up in the same way as characters are plotted on the screen in Mode 1.

As already suggested, 1Move only needs to be called once a game cycle to move everything. Printing of score and bonus is done once a second by calling the routine at line 200. All that remains is to read the keys, produce sound effects, check for collisions and keep the bullets firing.

For the sake of speed the last two requirements are covered by two routines tailored for this game. Collision checking is done by reading the collision flags of each sprite and by checking for identical positioning. The latter is only needed for a stationary sprite.

Eight bullets are allowed on screen at any time so as each is fired the one eight back must be erased. Key checking is left in Basic so that you can easily change the program to suit your fingers, and the speed can be changed using p% in 1070 and 4010.

Note that your subspace thrusters always work in the direction you are pointing so that once moving you need to turn round in order to slow down. Remember your hyper space dive is kaput so using it could well land you in the middle of one of those eggs or in the firing line of your own bullets. The faster you shoot the aliens the larger the bonus — if you take too long your bonus will become negative having a disastrous effect on your score!

Should you wish to start firing immediately without typing in the lengthy data, you should send £3 for a tape to C.J. Leigh, 12 The Bassetts, Cashes Green, Stroud, Glos GL5 4SJ. Ask for Space Eggs and don't forget your name and address.

Listing 2.

```
                ORIGIN  A34A
E5              PUSH HL         ;save cursor position
F5              PUSH AF         ;save character code
CD1ABC          CALL BC1A       ;SCR_CHAR_POSITION
F1              POP AF
EB              EX DE,HL        ;screen address in DE
CDA5BB          CALL BBA5       ;TXT_GET_MATRIX
0607            LD B,07         ;only top 7 rows
4E      CHRROW  LD C,(HL)       ;character byte
C5              PUSH BC
D5              PUSH DE
0604            LD B,04         ;stretch to 4 bytes
AF      BYTE    XOR A
CB21            SLA C           ;first pixel
3002            JRNC 02         ;pixel to screen byte
F6CC            OR A,CC         ;mask pen 3
CB21            SLA C           ;next pixel
```

(continued on page 67)

(continued from page 65)

```
3002            JRNC 02
F633            OR A,33         ;other half of byte
12              LD (DE),A       ;byte to screen
13              INC DE          ;next screen address
10EF            DJNZ BYTE       ;back for next pixel
D1              POP DE          ;screen address
E5              PUSH HL         ;save matrix address
210008          LD HL,800
19              ADD HL,DE       ;next screen row
EB              EX DE,HL        ;back in DE
E1              POP HL
23              INC HL          ;next character byte
C1              POP BC          ;recover row count
10DE            DJNZ CHRROW     ;next character row
E1              POP HL          ;recover cursor position
24              INC H
24              INC H
24              INC H
2C              INC L           ;reposition cursor
C375BB          JP BB75         ;TXT_SET_CURSOR
```

Listing 3.

```
                ORIGIN A400
010EA4  LOGON   LD BC,JMPTAB    ;set up new commands
210AA4          LD HL,BUFFER    ;system workspace
CDD1BC          CALL BCD1       ;KL_LOG_EXT
C9              RET
00000000 BUFFER DEFS 4
16A4    JMPTAB  DEFW NMETAB     ;command names address
C3ECA4          JP ERASE
C320A4          JP MOVE
4D524153        DEFM "ERAS"
C5              DEFB "E"+80
4D4F56          DEFM "MOV"
C5              DEFB "E"+80
00              NOP             ;end marker
DD2100A5 MOVE   LD IX,A500
1849            JR CHKEND
21CAA4  NEXSPR  LD HL,MASK+1
3600            LD (HL),0
DD7E01          LD A,(IX+1)     ;right position
000000          DEFS 3          ;for MC_WAIT_FLYBACK
CD79A4          CALL WRISPR     ;erase old sprite
21CAA4          LD HL,MASK+1
DD7E05          LD A,(IX+5)     ;sprite mask
77              LD (HL),A
DD7E02          LD A,(IX+2)     ;down position
DD8604          ADD A,(IX+4)    ;down speed
F246A4          JP P,02
C630            ADD 30          ;ensure positive
FE30            CP 30
3802            JR C,02
D630            SUB 30          ;not too large
DD7702          LD (IX+2),A     ;new down position
DD7E01          LD A,(IX+1)     ;right position
DD8603          ADD A,(IX+3)    ;right speed
F25AA4          JP P,02
C650            ADD 50          ;ensure positive
FE50            CP 50
3802            JR C,02
D650            SUB 50          ;not too large
DD7701          LD (IX+1),A     ;new right position
DD360800        LD (IX+8),00    ;clear collision flag
CD79A4          CALL WRISPR     ;write new sprite
110900  MOVEON  LD DE,09
DD19            ADD IX,DE       ;check next sprite
DD7E00  CHKEND  LD A,(IX+0)
1F              RRA             ;sprite on?
38B1            JR C,NEXSPR
1F              RRA             ;no more sprites?
30F2            JR NC,MOVEON
C9              RET
F5      WRISPR  PUSH AF         ;save right position
D650            SUB 50
ED44            NEG
4F              LD C,A          ;room for sprite
DD6602          LD H,(IX+2)     ;down position
DD5E06          LD E,(IX+6)     ;shape data address
DD5607          LD D,(IX+7)     ;shape high byte
1A              LD A,(DE)       ;size of sprite
47              LD B,A          ;width
6F              LD L,A          ;height
13              INC DE
F1              POP AF          ;right position
E5      VERT    PUSH HL
F5              PUSH AF
D5              PUSH DE
110000          LD DE,0000
F5              PUSH AF
7C              LD A,H          ;down position
FE30            CP 30           ;off screen bottom?
3802            JR C,02
D630            SUB 30
CB3F            SRL A           ;divide by two
3002            JR NC,02
1620            LD,20           ;middle of line
6F              LD L,A
2600            LD H,00         ;prepare to multiply
F1              POP AF
D5              PUSH DE
29              ADD HL,HL       ;times two
29              ADD HL,HL
29              ADD HL,HL
29              ADD HL,HL
E5              PUSH HL         ;sixteen times
D1              POP DE
29              ADD HL,HL
29              ADD HL,HL       ;times sixty four
19              ADD HL,DE       ;times eighty
D1              POP DE          ;half line offset
19              ADD HL,DE
16C0            LD D,C0         ;start of screen
5F              LD E,A          ;right position
19              ADD HL,DE       ;screen address
D1              POP DE          ;data address
3E04            LD A,04         ;four lines a block
F5      LINE    PUSH AF
C5              PUSH BC         ;width parameters
E5              PUSH HL
1A      BYTE    LD A,(DE)
FE00            CP 00
280C            JR Z,ZERO       ;ignore zero bytes
7E              LD A,(HL)       ;screen byte
FE00            CP 00           ;check collision
2803            JR Z,03
DD7708          LD (IX+8),A     ;collision flag
1A              LD A,(DE)       ;sprite byte
E6FF    MASK    AND FF          ;pen mask
77              LD (HL),A       ;write screen
13      ZERO    INC DE          ;next data
23              INC HL          ;next screen byte
0D              DEC C
2008            JR NZ,ROOM      ;room for sprite
D5              PUSH DE
AF              XOR A
115000          LD DE,0050
ED52            SBC HL,DE       ;start of line
D1              POP DE
10E0    ROOM    DJNZ BYTE       ;next width byte
E1              POP HL
010008          LD BC,0800
09              ADD HL,BC       ;next screen line
C1              POP BC
F1              POP AF
3D              DEC A           ;four lines
20D3            JRNZ,LINE
F1              POP AF
E1              POP HL
24              INC H
2D              DEC L
20A2            JRNZ,VERT       ;next vertical block
C9              RET
D5      ERASE   PUSH DE         ;data base address
21CAA4          LD HL,MASK+1
3600            LD (HL),00      ;clear pen mask
DDE1            POP IX
DD360000        LD (IX+0),00    ;turn sprite off
DD7E01          LD A,(IX+1)
C379A4          JP WRISPR
```