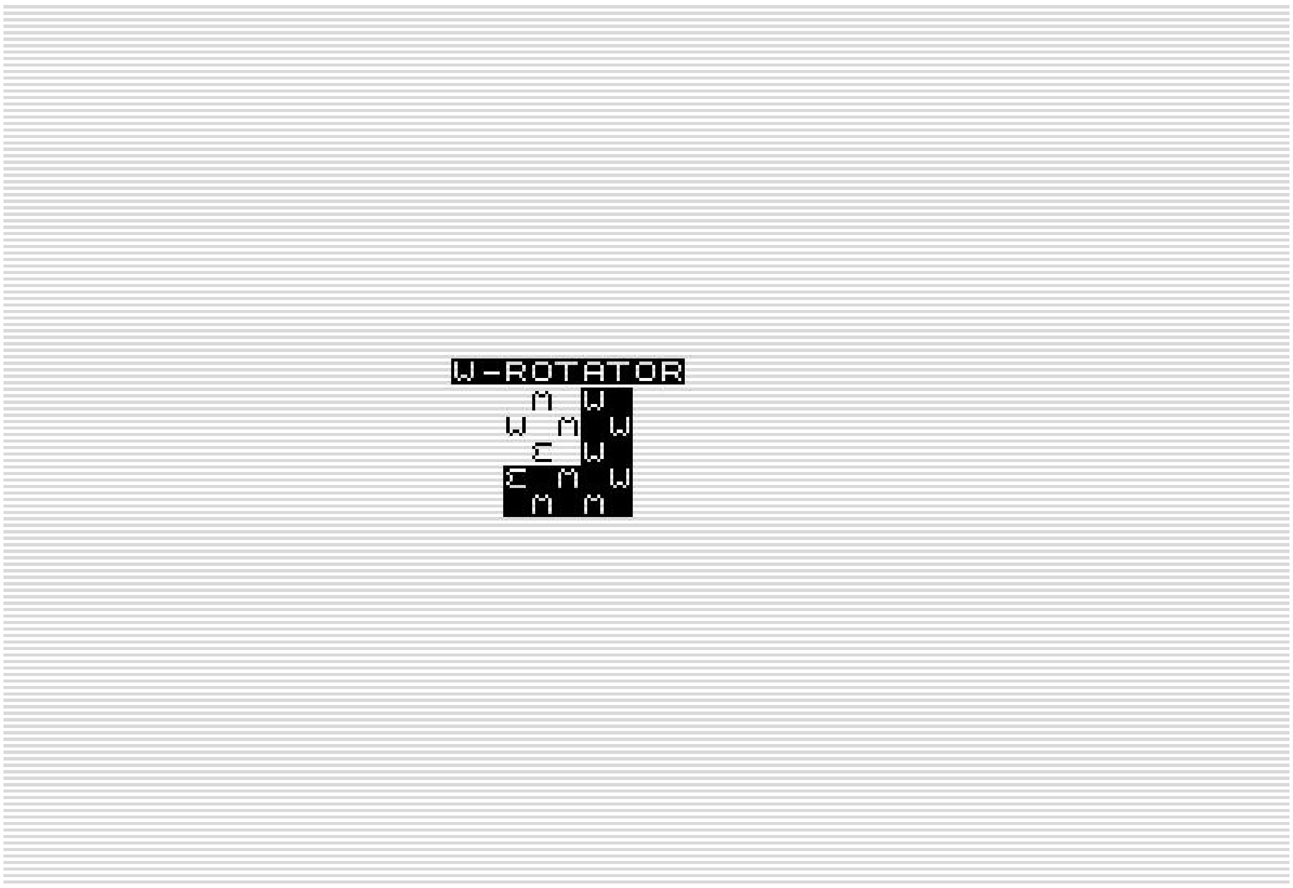
W-rotator



A routine for the ZX Spectrum gave me the idea for this game. I coded it in a day on the ZX Spectrum and I knew it would fit 1K on a ZX81. Due to the use of 1 letter only it looks lowres, but the character must be rotated which makes it a hires screen.

```
; ; W-Rotator, a lowres looking hires game in 1K
; Game 61 in 1K hires for the ZX81

; controls:
; Newline = shuffle
; Q = Up
; A = Down
; O = Left
; P = Right
; M = Rotate clockwise
; Z = Rotate anticlockwise
; R = restart



? * TORNADO *


           ORG  #4009
           DUMP 49161

basic      LD   D,#C0              ; preset for 48K bug
           JR   init0              ; this game has no 48K bug

           DEFB 236,212,28         ; The BASIC
           DEFB 126                ; fully placed over sysvar
           DEFB 143,0,18           ; start to BASIC=#4009
```

```
eline       DEFW last               ; needed to load
chadd       DEFW last-1
xptr        DEFW 0
stkbot      DEFW last               ; needed to load
stkend      DEFW last               ; needed to load
berg        DEFB 0
mem         DEFW 0
            DEFB 0

init1       JP   init               ; init can be anywhere

; all above reusable AFTER loading

lastk       DEFB 255,255,255        ; used by ZX81
margin      DEFB 55                 ; used by ZX81
nxtlin      DEFW basic              ; reusable after load

init0       LD   E,L                ; delay intrupts by
            DEFB #26                ; LD H,64
flagx       DEFB 64                 ; clever setting of flags

            XOR  A                  ; intruptcounter reset
            EX   AF,AF'

taddr       DEFW 0                  ; used by ZX81,no hurting code
            LD   B,4                ; frames is set ok

frames      DEFW #DD*256+1          ; used by ZX81, clever IX set
coprcc      LD   HL,hr              ; set IX
sposn       JR   init1
cdflag      DEFB 64                 ; used by zx81


; 5 linebuffers are needed to show a cursor

lbuf1       LD   R,A
            DEFB #80,#80,#80,#80,#80
            RET
lbuf2       LD   R,A
            DEFB #80,#80,#80,#80,#80
            RET
lbuf3       LD   R,A
            DEFB #80,#80,#80,#80,#80
            RET
lbuf4       LD   R,A
            DEFB #80,#80,#80,#80,#80
            RET
lbuf5       LD   R,A
            DEFB #80,#80,#80,#80,#80
            RET

; When solved the celibration routine is called

celibrate   LD   E,20                 ; An even number is needed
cel0        LD   B,5                  ; 5 rows
cel1        LD   C,5                  ; 5 columns
cel2        LD   HL,lbuf1-7           ; start of LBUF-pointer
            PUSH BC                   ; save coordinates
            LD   A,L
cel3        ADD  A,8
            DJNZ cel3                 ; each line is 8 bytes
            LD   L,A
            ADD  HL,BC                ; add column too
            LD   A,(HL)               ; get displayfield
```

```
           XOR  128                ; invert display to simulate FLASH
           LD   (HL),A
           POP  BC
           DEC  C
           JR   NZ,cel2
           DJNZ cel1
           CALL delay              ; show screen some time
           DEC  E
           JR   NZ,cel0            ; swap inversion

restart    LD   DE,text            ; The text of the game
           LD   B,5
nline      LD   C,5
ncol       PUSH BC
           PUSH DE
           CALL readrom            ; From ASCII to ROM-pointer
deset      LD   B,9                ; 8x, 1x C below zero
letcp      LDI                     ; copy ROM-character to hires screen
           INC  DE
           INC  DE
           INC  DE
           INC  DE
           DJNZ letcp              ; do full byte
           POP  DE
           INC  DE
           POP  BC
           DEC  C
           JR   NZ,ncol
           DJNZ nline              ; full screen printed

start      LD   A,191              ; Start new game with
           IN   A,(254)
           RRA                     ; newline
           JR   C,start

           LD   B,20
shuffle    PUSH BC                 ; save counter
           CALL rnd
           LD   B,A                ; set random Y
           CALL rnd
           LD   C,A                ; set random X
           CALL rnd                ; set random moves
           CALL rotate             ; do the rotation
           POP  BC
           DJNZ shuffle            ; shuffle the board

           LD   BC,#101            ; set XY
playloop   CALL cursor             ; show cursor

           PUSH BC                 ; save XY
w4down     LD   BC,(lastk)
           LD   A,C
           INC  A
           JR   Z,w4down
           CALL NZ,#7BD            ; read pressed key
           POP  BC
           PUSH AF
           CALL cursor             ; erase cursor
           POP  AF

           CP   13                 ; "R"
           JR   Z,restart

           CP   10                 ; "Q"
           JR   NZ,dir2
up         DEC  B
           JR   Z,down1           ; not out of board
           DEC  B
dir2       CP   5                 ; "A"
           JR   NZ,dir3
           INC  B
down1      INC  B
dir3       CP   26                ; "O"
           JR   NZ,dir4
left       DEC  C
           JR   Z,right1
           DEC  C
dir4       CP   25                ; "P"
           JR   NZ,other
           INC  C
right1     INC  C
other      LD   E,1
           CP   37                ; "M"
           CALL Z,rotate
           DEC  A                 ; "Z"
           CALL Z,rotatel

valid      LD   A,4
           CP   B
           JR   C,up             ; test out of board
           CP   C
           JR   C,left

; test solved
           PUSH BC
           LD   DE,text
           LD   B,5
chb0       LD   C,5
chb1       PUSH BC
           PUSH DE
           CALL readrom
; HL rom DE field
           LD   B,8
chbyte     LD   A,(DE)            ; get Screen-value
           CP   (HL)              ; test against ROM
           JR   NZ,nosolve        ; So when rotated not solved
           INC  HL
           LD   A,5
           ADD  A,E
           LD   E,A
           DJNZ chbyte
           POP  DE
           INC  DE
           POP  BC
           DEC  C
           JR   NZ,chb1
           DJNZ chb0
           DEFB 1
nosolve    POP  BC
           POP  BC
           POP  BC
           JR   NZ,playloop       ; here not solved
           JP   celibrate         ; here solved

readrom    LD   A,(DE)            ; get ascii
           CALL field             ; calculate screenaddress
           EX   DE,HL
           LD   H,4
           LD   L,A
           ADD  HL,HL
           ADD  HL,HL
           DEC  H
           ADD  HL,HL             ; HL now ROM-pointer
           RET

cursor     PUSH BC
           LD   D,3
lineloop   LD   E,3
colloop    LD   HL,lbuf1-16
           PUSH BC
           LD   A,B
           ADD  A,D
           LD   B,A
           LD   A,C

           ADD  A,E
           LD   C,A               ; Now BC holds relative position
           LD   A,L
calcl      ADD  A,8
           DJNZ calcl
           LD   L,A
           ADD  HL,BC
           LD   A,(HL)            ; get linebufposition
           XOR  128
           LD   (HL),A            ; invert display
           POP  BC
           DEC  E
           JR   NZ,colloop
           DEC  D
           JR   NZ,lineloop
           POP  BC

delay      LD   A,251
           LD   HL,frames
           ADD  A,(HL)
wfr        CP   (HL)
           JR   NZ,wfr
           RET

rotate     PUSH DE
           LD   HL,rtab           ; right rotation
block      PUSH BC
           PUSH HL
           CALL dispbc            ; calculate relative position
           PUSH HL
           CALL cpscrbf           ; copy screen to buffer
           POP  DE
           LD   B,8
fullbyte   LD   HL,#4000          ; buffer
           LD   A,128
nrow       RLC  (HL)              ; rotate buffer
           RRA
           INC  HL
           JR   NC,nrow
           LD   (DE),A            ; and write to screen
           LD   A,E
           ADD  A,5
           LD   E,A
           DJNZ fullbyte
           POP  HL
           POP  BC
           LD   A,(HL)
           INC  HL
           OR   A
           JR   NZ,block           ; rotate all 9 bytes of the block

           CALL shift1             ; now shift 9 positions 1 position
           CALL shift1             ; twice needed for a 90 degrees turn
           POP  DE
           DEC  E
           JR   NZ,rotate          ; 3x left = 1x right turn
           RET

; For speed during gameplay a left rotation is extra coded
rotatel    LD   HL,ltab
blockl     PUSH BC
           PUSH HL
           CALL dispbc
           PUSH HL
           CALL cpscrbf
           POP  DE
           LD   B,8
fullbytl   LD   HL,#4000           ; buffer
           LD   A,1
nrowl      RRC  (HL)
           RLA
           INC  HL
           JR   NC,nrowl
           LD   (DE),A
           LD   A,E
           ADD  A,5
           LD   E,A
           DJNZ fullbytl
           POP  HL
           POP  BC
           LD   A,(HL)
           INC  HL
           OR   A
           JR   NZ,blockl
           LD   HL,ltab
           PUSH HL
           CALL shift1+3
           POP  HL
           JR   shift1+3

shift1     LD   HL,rtab
           PUSH HL
           PUSH BC
           CALL dispbc
           PUSH HL
           CALL cpscrbf
           POP  DE
           POP  BC
           POP  HL
moveall    INC  HL
           PUSH HL
           PUSH BC
           CALL dispbc
           PUSH HL
           LD   B,8
movebyte   LD   A,(HL)
           LD   (DE),A
           LD   A,5
           ADD  A,L
           LD   L,A              ; L=L+5
           LD   A,5
```

```
            ADD  A,E
            LD   E,A                ; E=E+5
            DJNZ movebyte
            POP  DE
            POP  BC
            POP  HL
            LD   A,(HL)
            CP   9
            JR   NZ,moveall

; now the first saved position must go to the screen also
            LD   HL,#4000
buf2scr     LD   A,(HL)
            LD   (DE),A
            INC  HL
            LD   A,E
            ADD  A,5
            LD   E,A
            LD   A,L
            AND  7
            JR   NZ,buf2scr
            RET

cpscrbf     LD   B,8
            LD   DE,#4000
scr2buf     LD   A,(HL)
            LD   (DE),A
            INC  DE
            LD   A,L
            ADD  A,5
            LD   L,A
            DJNZ scr2buf
            RET

dispbc      LD   A,(HL)
            AND  7
            ADD  A,C
            LD   C,A
            LD   A,(HL)
            AND  #F8
            RRCA
            RRCA
            RRCA
            ADD  A,B
            LD   B,A

field       LD   HL,hrscreen-41
            PUSH DE
            LD   DE,40
ycol        ADD  HL,DE
            DJNZ ycol
            ADD  HL,BC
            POP  DE
            RET

rnd         LD   HL,(frames)
rseed       LD   DE,0
            ADD  HL,DE
            INC  HL
            LD   A,H
            AND  #1F
            LD   H,A
            LD   (rseed+1),HL
            LD   A,(HL)
            AND  3
            LD   E,A
            AND  1
            JR   Z,rnd
            LD   A,E              ; 1 or 3 only
            RET

hr          LD   HL,lowres+#8000  ; the lowres display
            LD   BC,#251          ; minimum lines in this game
            LD   A,#1E            ; needed to prevent scrolling
            LD   I,A
            LD   A,#FB
            CALL #2B5

            LD   B,5              ; outline delay for hires
hr00        DJNZ hr00
            DEC  HL

            LD   HL,lbuf1+#8000-8
            LD   A,hrscreen/256
            LD   I,A
            LD   E,hrscreen*256/256-5
            LD   B,6
bloop       DEC  B
            JR   Z,exit
            LD   C,8
            LD   A,C
            ADD  A,L
            LD   L,A

cloop       EX   (SP),HL
            EX   (SP),HL
            EX   (SP),HL
            EX   (SP),HL

            LD   A,E
            INC  DE
            DEC  E
            ADD  A,5
            LD   E,A

            CALL #7D              ; call (hl)
            DEC  C
            JR   Z,bloop

            LD   A,(HL)
            LD   A,(HL)
            LD   A,R

            JR   cloop

exit        LD   B,252
filler      EX   (SP),HL
            EX   (SP),HL
            DJNZ filler

            CALL #292             ; back from intrupt
            CALL #220
            LD   IX,hr
            JP   #2A4

x           EQU  101
n           EQU  27
```

```
lowres      DEFB 118
            DEFW 0,0,0,0,0
            DEFB "W"+x,150,"R"+x,"O"+x,"T"+x,"A"+x
            DEFB "T"+x,"O"+x,"R"+x
            DEFB 118

; when needed tables can be copied over sysvar

; 5478963215
rtab        DEFB 9,8,16
            DEFB 17,18,10
            DEFB 2,1,0,9

; 5236987415
ltab        DEFB 9,1,2
            DEFB 10,18,17
            DEFB 16,8,0,9

w           EQU  60

text        DEFB 0,w,0,w,0
            DEFB w,0,w,0,w
            DEFB 0,w,0,w,0
            DEFB w,0,w,0,w
            DEFB 0,w,0,w,0

; codeable block-stack
space       EQU  #4335-$
            DEFS space

; screen as far possible to end of memory
hrscreen    DEFB 0
init        LDIR                    ; repair 48K bug
            LD   SP,hrscreen
            JP   restart

vars        DEFB 128
?
last        EQU  $
```