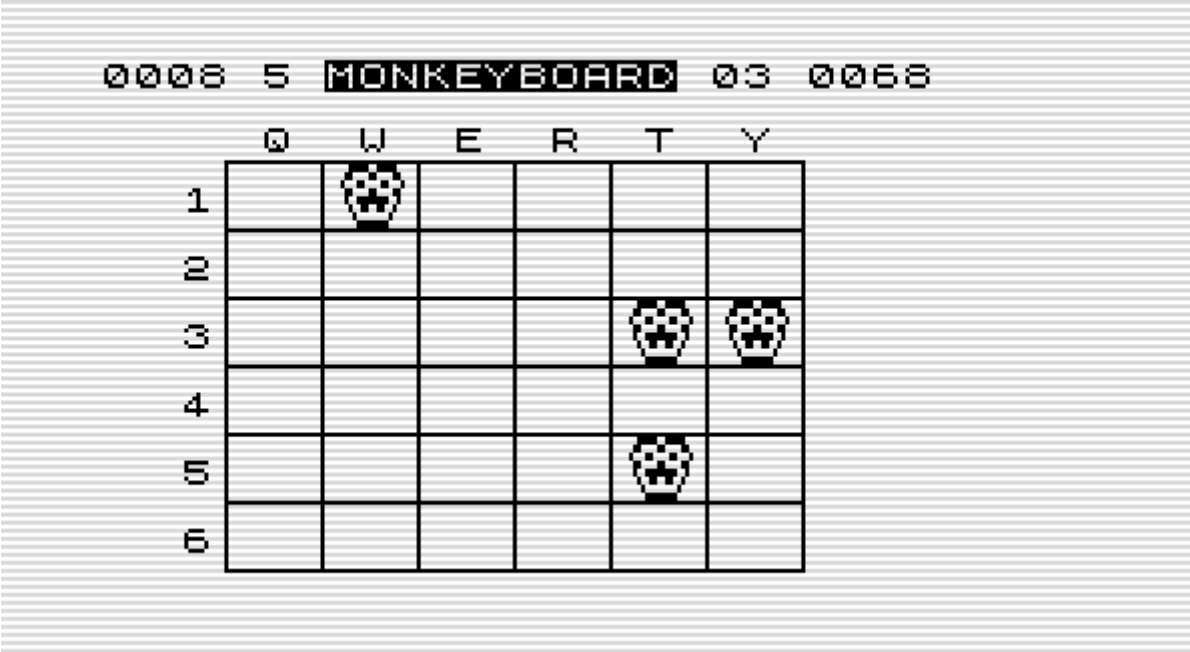
Monkeyboard



'68 is the chinese year of the monkey. In this game you press the coordinates where the monkey comes on the board. The coordinates you type on your keyboard. Hence the double meaning of the name of the game. I wanted to code this game for a long time but the display of lowres numbers on a hires screen was difficult to do in a small amount of memory.
Game 67 bought me the solution.

```
; Monkeyboard
; Game 68 in 1K hires for the ZX81

horline    EQU  #4010-2

? * TORNADO *

           ORG  #4009             ; #4009
           DUMP 49161

basic      LD   D,#C0             ; preset for 48K bug
           JR   init0

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

eline      DEFW last              ; needed by loading
chadd      DEFW last-1
xptr       DEFW 0
stkbot     DEFW last
stkend     DEFW last
berg       DEFB 0
mem        DEFW 0
           DEFB 128

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
```

```
; 11 bytes useable with LDIR, 10+5 when frames is skipped
nxtlin     DEFW basic             ; reusable after load

init0      LD   E,L               ; DE now #C0.L
           DEFB #26               ; HL now #40.L
flagx      DEFB 64

           XOR  A                 ; intruptcounter reset
           EX   AF,AF'

taddr      DEFW 0                 ; used by ZX81 on loading
           LD   B,4               ; copy >1K of code

frames     DEFW #DD01             ; used by ZX81, opcode IX
coprcc     LD   HL,hr             ; set IX to HR with opcode DD
sposn      JR   init1             ; continue to mainprog

cdflag     DEFB 64                ; used by ZX81

keytab     DEFB 10,11,12,13,14,29 ; QWERTY key values
           DEFW 0                 ; filler to set mokeys ok

monkeys    DEFB 0,16,62,124,16,62,124,16
           DEFB 62,124,16,62,124,16,62,124,16
           DEFB 62,124,16

m2         DEFB 0,16,97,134,16,97,134,16
           DEFB 97,134,16,97,134,16,97,134,16
           DEFB 97,134,16

m3         DEFB 0,16,140,49,16,140,49,16
           DEFB 140,49,16,140,49,16,140,49,16
           DEFB 140,49,16

m4         DEFB 0,16,97,134,16,97,134,16
           DEFB 97,134,16,97,134,16,97,134,16
           DEFB 97,134,16

m5         DEFB 0,16,79,242,16,79,242,16
           DEFB 79,242,16,79,242,16,79,242,16
           DEFB 79,242,16

m6         DEFB 0,16,38,100,16,38,100,16
           DEFB 38,100,16,38,100,16,38,100,16
           DEFB 38,100,16

m7         DEFB 0,16,16,8,16,16,8,16
           DEFB 16,8,16,16,8,16,16,8,16
           DEFB 16,8,16

m8         DEFB 0,16,15,240,16,15,240,16
           DEFB 15,240,16,15,240,16,15,240,16
           DEFB 15,240,16

;the display routine lowres and hires
hr         LD   HL,lowres+#8000   ; the lowres display
           LD   BC,#431           ; minimum needed #11
           LD   A,#1E
           LD   I,A
           LD   A,#FB
           CALL #2B5

           LD   B,8
```

```
hr00       DJNZ hr00

           LD   A,monkeys/256
           LD   I,A               ; set highbyte

           LD   IX,retlbuf        ; return from highmem
           EXX
           LD   HL,#1EE8          ; start of number "1" in ROM

           LD   (savesp+1),SP
           LD   SP,#4000          ; displaystack is on sysvar
           LD   A,(HL)            ; filler

bloop      EXX                    ; delay
           LD   HL,addtab         ; next line pointer table
           EXX
           LD   DE,monkeys        ; start of monkeys data
           LD   A,horline*256/256 ; but first show a line
           JP   lbuf2+#8000

retlbuf    EXX                    ; alternate registers
           ADD  A,(HL)            ; calculate next pointer
           INC  L
           EXX
           LD   E,A               ; save pointer
           DEC  C
           JP   Z,bdelay          ; check all lines displayed
           DEC  SP                ; undo RET
           DEC  SP

cloop      LD   A,B
           CP   8                 ; check lowrescounter
           JR   C,wrnr            ; show a number
           XOR  A                 ; or show a space
           NOP
           DEFB #DA               ; "JP C", skip get nr from ROM
wrnr       LD   A,(HL)
           INC  HL
           LD   (DE),A            ; write nr or space
           DEC  B
           LD   A,E               ; set start of data to show
           RET

bdelay     POP  DE                ; POP before PUSH to keep
           PUSH DE                ; stack unchanged
           JR   bloop             ; do next line

lbuf2      LD   R,A
addtab     DEFB 0,20,0,20,0,20    ; table to double display
           DEFB 0,20,0,20,0,20    ; same line 2x
           DEFB 0,20,0,0,0,0,0    ; can be stored in LBUF2
           JP   back              ; 48K bug

back       OR   (HL)              ; delay
           RET  C                 ; delay
           RET  C                 ; delay
           LD   A,monkeys*256/256
           LD   BC,#C10           ; 4 sp , 8 chr, 4 sp, 16 lines
           JR   cloop

; fixed end of HR-routine
savesp     LD   SP,0              ; repair stack
exit       CALL #292              ; back from intrupt
           CALL #220
```

```
           LD   IX,hr             ; set for next display
           JP   #2A4

x          EQU  101
lowres     DEFB 118
score      DEFB 28,28,28,28,0
lives      DEFB 28,0
           DEFB "M"+x,"O"+x,"N"+x,"K"+x,"E"+x
           DEFB "Y"+x,"B"+x,"O"+x,"A"+x,"R"+x,"D"+x,0
time       DEFB 28,28,0
hiscore    DEFB 28,28,34,36
           DEFB 118,118
           DEFB 0,0,0,0,0
qwerty     DEFB "Q"-27,0,0,"W"-27,0,0,"E"-27,0,0
           DEFB "R"-27,0,0,"T"-27,0,0,"Y"-27,118


jumptab    DEFB 3,2,1,2,3,2,1,2,3,2,1,2,3,255

char       DEFB %10001000         ; HI text
           DEFB %10000000
           DEFB %11101000
           DEFB %10101000
           DEFB 255

eog        LD   DE,hiscore-1      ; pointer to hiscore
           LD   HL,score-1        ; pointer to current score
           LD   BC,5              ; lenght to test
fihi       INC  HL                ; next digit in score
           INC  DE                ; next digit in hiscore
           DEC  C                 ; digit less to copy
           LD   A,(DE)            ; get hiscore digit
           CP   (HL)              ; test against score
           JR   Z,fihi            ; still the same
           JR   NC,start          ; added to skip celebration
           CALL #19F9             ; new hiscore through ROM

;hiscore celibration
           LD   DE,jumptab

celibrate  PUSH DE
           CALL cls
           LD   A,(DE)            ; start of char
           LD   B,A               ; set start of character
           LD   DE,char
showchar   PUSH DE                ; save position
           LD   C,1
           LD   A,(DE)
show       ADD  A,A               ; shift character in carry
           PUSH AF
           CALL field
           POP  AF
           JR   NC,noshow
           LD   (HL),B            ; show monkey
           INC  HL
           LD   (HL),B
noshow     INC  C                 ; next textbitposition
           OR   A
           JR   NZ,show
           INC  B                 ; next part of char
           POP  DE
           INC  DE
           LD   A,(DE)
           INC  A                 ; test end of char
```

```
            JR   NZ,showchar

            LD   A,250              ; show char some time
            CALL delay

            POP  DE

            INC  DE
            LD   A,(DE)
            INC  A                  ; test end reached
            JR   NZ,celibrate

start       LD   A,(lastk)          ; game over, wait for
            SUB  %10111111          ; newline
            JR   NZ,start

            LD   HL,score           ; erase old score
clsc        LD   (HL),28
            INC  HL
            CP   (HL)
            JR   NZ,clsc

            CALL cls

            LD   A,37               ; 9 misses is game over
            LD   (lives),A

newmonkey   LD   HL,qwerty
clbit       RES  7,(HL)             ; erase inverted key
            INC  HL
            LD   A,(HL)
            CP   #76
            JR   NZ,clbit

            LD   A,230
            CALL delay



rndpos      CALL rnd
            LD   B,A                ; set random Y
            CALL rnd
            LD   C,A                ; set random X
newfld      DEC  C                  ; goto X - 1
            JR   NZ,fieldtest       ; test out of range
            LD   C,6                ; set at end
            DEC  B                  ; goto Y - 1
            JR   NZ,fieldtest       ; test valid field
            LD   B,C                ; go to end of screen

fieldtest   CALL field              ; get field
            BIT  6,(HL)             ; test used
            JR   Z,newfld           ; used, then previous field

            XOR  A
            LD   (HL),A             ; show monkey on field
            INC  HL
            LD   (HL),A             ; double sized

            LD   L,10
            CALL rnd+2              ; get random timer

            ADD  A,L                ; add minimum time
            AND  254                ; make A even
```

```
            LD   (timer+1),A       ; set timer

            LD   A,keytab*256/256-1
            ADD  A,C
            LD   E,A
            LD   D,keytab/256
            LD   A,(DE)            ; which letter is to press
            LD   (keytest+1),A     ; save letter to check

loop        LD   A,250             ; delay, but game is still
            CALL delay

            PUSH BC
            LD   BC,(lastk)
            LD   A,C
            INC  A
            CALL NZ,#7BD           ; translate a pressed key
            POP  BC

keytest     LD   E,0
            CP   E

            JR   NZ,notsame        ; not correct key pressed
            LD   HL,qwerty-3
            LD   A,C
            ADD  A,A
            ADD  A,C
            ADD  A,L
            LD   L,A
            SET  7,(HL)            ; signal QWERTY ok

            LD   A,B               ; now we need a number to pres
            CP   6                 ; is it 1-5
            JR   NZ,seta
            LD   A,10              ; number 6 is not in order

seta        ADD  A,14
            LD   (keytest+1),A     ; set next key to check
            LD   A,E               ; get key to test
            SUB  15
            CP   10                ; does it match the number
            JR   NC,notsame        ; if not, no score

            CALL field
            LD   (HL),64           ; erase captured monkey
            INC  HL
            LD   (HL),64

timer       LD   B,0               ; remaining time as score
points      LD   HL,score+4
            DEFB 17
ten         LD   (HL),28
            DEC  HL
            INC  (HL)
            LD   A,(HL)
            CP   38
            JR   Z,ten
            DJNZ points
            JR   nextmnk           ; set new monkey

notsame     CALL field
            LD   A,(HL)            ; swap display
            XOR  64
```

```
            LD   (HL),A             ; when time runs out
            INC  HL                 ; always DISPLAY ON
            LD   (HL),A

            LD   HL,timer+1
            DEC  (HL)
            LD   A,(HL)
            PUSH AF                 ; save timer result

            LD   HL,time            ; show time on screen
            LD   (HL),27
setten      INC  (HL)
            SUB  10
            JR   NC,setten
            INC  HL
            ADD  A,38
            LD   (HL),A

            POP  AF                 ; get timer result
            JR   NZ,loop            ; not out of time

            LD   HL,lives           ; out of time, not captured
            DEC  (HL)
            LD   A,(HL)
            CP   28
            JP   Z,eog              ; game over, test hiscore

nextmnk     JP   newmonkey          ; new monkey after dead

field       PUSH BC                 ; field is in display buffers
            PUSH DE
            LD   HL,lbuf1-23
            LD   DE,24              ; size of buffer
frow        ADD  HL,DE              ; find right buffer
            DJNZ frow
            ADD  HL,BC              ; find position in buffer
            ADD  HL,BC
            ADD  HL,BC
            POP  DE
            POP  BC
            RET


cls         LD   HL,lbuf1           ; erase board
cls0        LD   BC,#640
            INC  HL                 ; skip LD R,A
            INC  HL
            INC  HL                 ; skip number
            INC  HL                 ; skip border
cl1         LD   (HL),C             ; erase monkey
            INC  HL
            LD   (HL),C             ; erase monkey
            INC  HL
            INC  HL
            DJNZ cl1                ; erase  do 6 monkeys
            INC  HL                 ; skip JP IX
            INC  L                  ; test next boundary
            JR   NZ,cls0            ; clear 6 lines
            RET


delay       LD   HL,frames          ; fast
            ADD  A,(HL)
wfr         CP   (HL)
```

```
            JR   NZ,wfr
            RET

rnd         LD   L,6
            LD   A,(frames)
rseed       ADD  A,1                ; get seed
            LD   H,A
            RRCA
            RRCA                    ; a=a/2
            RRCA                    ; a=a/2
            XOR  31                 ; swap low bits
            ADD  A,H                ; add seed
            SBC  A,255
            LD   (rseed+1),A        ; save new seed
subl        SUB  L
            JR   NC,subl
            ADC  A,L
            RET                     ; exit rnd

size        EQU  6*24

st          EQU  26                 ; stack size

space       EQU  #4400-size-st-$

            DEFS space

; executable code on the stack, 1 time only

stackcode   LD   DE,screen          ; set is behind first line
            LD   BC,size-24         ; copy it 5 times
            LDIR                    ; copy all linedisplay buffers
            JP   eog                ; start through end of game

            DEFS st-11              ; SP-filler: size SP 26 bytes

lbuf1       LD   R,A
            DEFW 0,0                ; on load all fields visible
            DEFW 0,0                ; program will determine
            DEFW 0,0                ; which fields need to show
            DEFW 0,0                ; the UDG on that field
            DEFW 0,0
            JP   (IX)               ; return lowmemory

screen      EQU  $
; in fact 2nd line of screen will start here, 1st is lbuf1

; initialization code on screen is done before
; first screen is called to be drawn

init        LDIR                    ; repair 48K bug from LBUF2
            LD   HL,lbufstack       ; get displaystack from screen
            LD   DE,#4000           ; destination: sysvar
            LD   C,36               ; copy stack to now free mem
            LDIR                    ; reuse sysvar memory this way

            LD   HL,lbuf1           ; get displayline
            LD   SP,HL              ; move SP from end of RAM
            JP   stackcode          ; copy must be done elsewhere

lbufstack   DEFW lbuf1+#8000        ; line 1
            DEFW lbuf1+#8000+24     ; line 2
            DEFW lbuf1+#8000+48     ; line 3
            DEFW lbuf1+#8000+72     ; line 4
```

```
            DEFW lbuf1+#8000+96    ; line 5
            DEFW lbuf1+#8000+120   ; line 6

            DEFW savesp            ; exit screen by RET

            DEFB 127,255,255,255,255,255
            DEFB 255,255,255,255,255,255
            DEFB 255,255,255,255,255,255
            DEFB 192


vars        DEFB 128
?
last        EQU  $
```