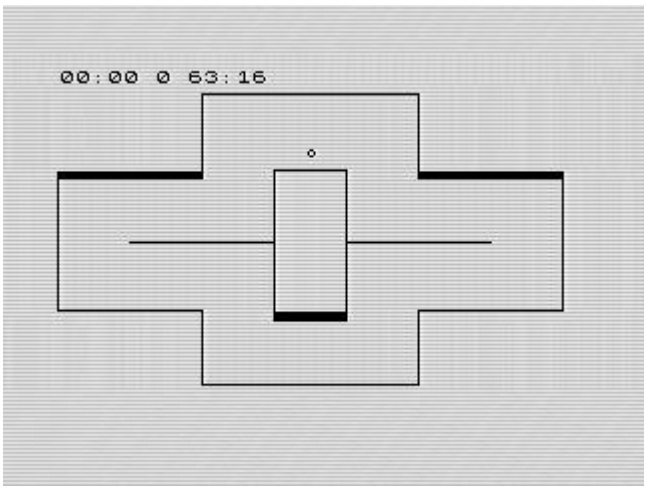
Marble Racer



The gameidea is based on Jim Bagley's racing game in hires for the ZX81, combined with my game to guide a gyroscopic ball to a hole. Take the ball and a circuit and you have marble racer. The use of a marble is done for practical use only. A marble is always round, no matter what direction it rolls. You don't need graphics for other directions. To solve a collisionbug some walls had to be thickened (cheapest solution in memory). In theory other tracks are possible, but the initialisation must be rewritten for it.

```
; Marble racer, another 1K hires game for the ZX81
; controls:
;  0 = increase x speed
;  9 = decrease x speed
;  z = increase y speed
;  a = decrease y speed
; NL = start game

maxleft    EQU  #90
delaytime  EQU  4
nrround    EQU  33              ; "0" + 5

? * TORNADO *

           ORG  #4009           ;#4009
           DUMP 49161

           LD   HL,#4400        ; preload for SP
           JR   init0           ; next init step

;d_file     DEFW dfile
dfcc       DEFW dfile+1
var        DEFW vars
dest       DEFW 0
eline      DEFW last
chadd      DEFW last-1
xptr       DEFW 0
stkbot     DEFW last
stkend     DEFW last            ; memory above reused for data

berg       DEFB 0
mem        DEFW 0
init0      LD   SP,HL           ; set SP to save memory
           JP   init            ; do initialization
```

```
;           DEFB 0
;           DEFB 2
;           DEFW 1

lastk       DEFB 255,255,255        ; used by ZX81

margin      DEFB 55
nxtlin      DEFW basic
count       DEFB 0
            DEFB 0
flagx       DEFB 0                  ; x
strlen      DEFW 0

taddr       DEFW 3213

seed        DEFW 0
frames      DEFW 65535              ; used by ZX81
coords      DEFB 0,0
prcc        DEFB 188
sposn       DEFB 33,24
cdflag      DEFB 64


lowres      DEFB 118                ; timer and roundcounter
score       DEFB 0,28,28
            DEFB 14,28
time        DEFB 28,0,28
hiscore     DEFB 0,37,37,14,37,37,0
attop       DEFB 0
            DEFB 118

hr          LD   B,14               ; the screenroutine
h1          DJNZ h1                 ; start on screen

            LD   HL,lowres+#8000    ; start with lowres
            LD   BC,#208
            LD   A,#1E
            LD   I,A
            LD   A,#F5
            CALL #2B5

            LD   B,5                ; bring hires in sync
h2          DJNZ h2

            LD   HL,screen
            LD   A,line1/256
            LD   I,A
            DEFB #DD
            LD   H,hitop/256
            CALL hitop              ; display top
            CALL himid              ; display mid and bottom

            CALL #292               ; and back to program
            CALL #220
            LD   IX,hr              ; set hr start
            JP   #2A4



placescr    LD   D,#40              ; main table shifted marbles
            LD   A,(DE)             ; get marble pixels
            INC  DE
            LD   D,A                ; save for test
```

```
            OR   (HL)              ; bring in screen
            XOR  (HL)              ; filter of screen
            CP   D
            JR   NZ,oldxy          ; collision, get old xy
            OR   (HL)              ; set marble
            LD   (HL),A
            RET

oldxy       LD   BC,0              ; fetch old xy on collision
            POP  AF                ; drop return from above
            LD   A,50              ; 1 sec wait penalty
            LD   (wait+1),A        ; set penalty

startnew    LD   HL,0              ; entry for start new game
            LD   (dydx+1),HL       ; stop movement

dydx        LD   HL,0              ; fetch dy and dx
            LD   A,B               ; move the marble
            ADD  A,H
            LD   B,A
            LD   A,C
            ADD  A,L
            LD   C,A


            LD   HL,curstab        ; get original index
            LD   A,4               ; undo display current marble
oldpos      LD   E,(HL)
            INC  HL
            LD   D,(HL)
            INC  HL                ; DE line of marblepointer
            INC  BC                ; undo DEC BC from LDI
            LDI                    ; undo marble, set index back
            INC  HL
            DEC  A
            JR   NZ,oldpos         ; erase all marblelines

dispmarb    LD   HL,screen-1       ; make new display

            PUSH BC                ; save original xy

            LD   DE,2              ; alter direct to 1 for top


alterde     INC  HL                ; skip blockmarker
            LD   A,3
            SUB  E
            LD   E,A
            DEFB 62                ; hl already ok


ftop        ADD  HL,DE             ; goto next line
            LD   A,(HL)
            OR   A
            JR   Z,alterde         ; start not in this part
            DJNZ ftop

exfnd       LD   A,maxleft         ; preset test for mid
            CP   C
            SBC  A,A
            LD   (lrtest+1),A
yfound      LD   BC,curstab        ; pointer to repairindex
            EXX
            LD   D,linec1*256/256  ; marble data
```

```
            LD   BC,line1/256*256+4 ; counter and highbyte
setcurs     EXX
            PUSH HL                 ; save linepointer
            LD   A,E
            DEC  A
            JR   Z,tb               ; not in mid
lrtest      ADD  A,0                ; 1 + 255 or 1 + 0  ; r or l
            JR   NZ,tb              ; in mid, but left
            INC  HL                 ; in mid right
tb          LD   A,L                ; save marbleindex address
            LD   (BC),A
            INC  BC
            LD   A,H
            LD   (BC),A
            INC  BC
            LD   A,(HL)
            LD   (BC),A             ; save original index
            EXX
            LD   E,A
            PUSH DE                 ; save for return
            PUSH BC                 ; save C reg
            LD   C,14
copyloop    PUSH BC                 ; copy background to marble
            LD   C,E
            LD   A,(BC)
            LD   C,D
            LD   (BC),A
            INC  E
            INC  D
            POP  BC                 ; get copycounter
            DEC  C
            JR   NZ,copyloop
            POP  BC                 ; get loopcounter
            EXX
            POP  AF                 ; D to A
            LD   (HL),A             ; set marbleindex in screen
            POP  HL
            INC  BC
            LD   A,E
            LD   (BC),A             ; part index in table
            INC  BC
            ADD  HL,DE
cont1       LD   A,(HL)
            OR   A
            JR   NZ,cont2           ; marble partly on 2 parts
            LD   A,3
            SUB  E
            LD   E,A                ; adjust counter top/mid/bot
            INC  HL
            JR   cont1
cont2       EXX
            DEC  C
            JR   NZ,setcurs
            LD   HL,linec1-1        ; first line to set saved in
            EXX                     ; alternate registers

            POP  BC                 ; fetch xy

            LD   HL,curstab+3       ; index for top/mid/bottom
            PUSH HL                 ; later again needed

            LD   D,4                ; do 4 loops

marbledisp  LD   A,(HL)
```

```
            DEC  A
            LD   E,#48              ; for top and bottom
            JR   Z,topbot           ; top or bottom
            LD   A,C
            LD   E,0                ; zero for left mid
            CP   maxleft
            JR   C,topbot           ; left mid screen
midright    LD   E,#90              ; right mid screen
topbot      LD   (HL),E             ; save adjustment for display
            INC  HL                 ; from counter to index in
            INC  HL                 ; cursor table
            INC  HL
            INC  HL
            DEC  D
            JR   NZ,marbledisp
            LD   A,C
            EXX
            LD   E,nxtlin*256/256   ; marble0
            AND  7
            JR   Z,mfound
            DEC  A                  ; table shifted 1 pos
            ADD  A,A
            ADD  A,A
            LD   E,A                ; de index calculated
mfound      EXX
            POP  HL                 ; retrieve table

            LD   D,4                ; again 4 loops
setmarble   LD   A,C
            SUB  (HL)               ; adjustment on x for display
            INC  HL
            INC  HL
            INC  HL
            INC  HL
            EXX                     ; now to lines of marble
            LD   B,15
fstart      DEC  B
            INC  HL
            SUB  8
            JR   NC,fstart
            CALL placescr           ; set marble left part
            INC  HL
            CALL placescr           ; set marble right part
            DEC  HL
            LD   A,E
            ADD  A,C
            LD   E,A
            DEC  C                  ; from 1 to -1 to -3
            DEC  C                  ; so all marbles fit 4 bytes

            DEFB 62                 ; skip increase
fnextline   INC  HL
            DJNZ fnextline          ; B deliberately 1 too much
            EXX
            DEC  D
            JR   NZ,setmarble

; the actual gameloop
; we have a valid position, save on old for collision
            LD   (oldxy+1),BC

; delay for gameplay

wait        LD   D,delaytime        ; also collision penalty
```

```
            LD   HL,frames          ; nice steady screendisplay
            LD   A,(HL)
            SUB  D
wfr         CP   (HL)
            JR   NZ,wfr

timejp      LD   DE,#1C00           ; D preload "0"
            LD   (timejp+1),A       ; E old delay, A is new E
doadd       INC  A                  ; increases, needed when
            INC  E                  ; frames goes around
            CP   E
            JR   NC,doadd           ; repeat until not around
            SUB  E                  ; delay remains constant
            LD   E,A                ; so add passed time
settime     LD   HL,time
            INC  (HL)               ; 1/50
            DEFB #CA                ; jp z never true, always odd
inctime     LD   (HL),D
time2       DEC  HL
            LD   A,(HL)
            ADD  A,A                ; 2 x ":" = "0"
            JR   Z,begin
            CP   D                  ; ":", test on "0"
            JR   Z,time2            ; skip part sec counter
            INC  (HL)               ; 2/50 or higher counter
            LD   A,38
            CP   (HL)               ; test next 10 boundary
            JR   Z,inctime
            INC  E                  ; do all passed time
            JR   NZ,settime

; now test 1 round and eog (end of game)
            LD   HL,attop           ; now used as roundchecker
            LD   A,B
            CP   #75                ; at bottom of track
            JR   C,test2
            SET  0,(HL)             ; signal bottom reached
test2       CP   #27                ; at top of track
            LD   A,C                ; preload x-coordinate
            JR   NC,test3           ; not at top

testround   CP   #80
            JR   C,test3            ; not passed finish
            LD   A,(HL)
            SUB  31
            JR   NZ,keys            ; not 1 round made yet
            LD   (HL),A             ; set value back, always 0
            LD   HL,time+2          ; lap made
            INC  (HL)               ; lap counted
            LD   A,(HL)
            CP   nrround            ; test on full race
            JR   NZ,keys            ; not finished

; test on hiscore
            LD   HL,score-1
            LD   DE,hiscore-1
            LD   BC,7               ;  1 too much saves a byte
findhi      INC  DE
            INC  HL
            DEC  C
            JR   Z,begin
            LD   A,(DE)
            CP   (HL)
            JR   C,begin            ; slower no high
```

```
            JR   Z,findhi          ; time same until now
sethi       LDIR                   ; here faster, set hi

begin       LD   A,(lastk)         ; wait for NEWLINE
            SUB  %10111111         ; to start game
            JR   NZ,begin
            LD   (timejp+1),A      ; set timedelay and
            LD   (frames),A        ; frames in sync
            LD   (attop),A         ; repair out of time bug
            LD   HL,lowres+1       ; reset time
            LD   B,8               ; and rounds on screen
clear       LD   A,28
            CP   (HL)
            JR   NC,nreset
            LD   (HL),A
nreset      INC  HL
            DJNZ clear

            LD   BC,#2080          ; x=128 y=32, top block start
            JP   startnew

test3       CP   #DC               ; on right side of track
            JR   C,keys
            LD   (HL),30           ; set checkpoint 1 + direction

keys        LD   A,delaytime       ; undo possible penalty
            LD   (wait+1),A

            LD   HL,(dydx+1)       ; fetch x/y movement

            LD   A,#EF             ; read 6-0
            IN   A,(254)

right       RRA
            JR   C,left            ; test 0 pressed
            INC  L
left        RRA
            JR   C,up              ; test 9 pressed
            DEC  L
up          LD   A,%11111100       ; ports a-g and sp-v
            IN   A,(254)           ; done together
            RRA
            JR   C,down            ; test A pressed
            DEC  H
down        RRA
            JR   C,fkey            ; test Z pressed
            INC  H
fkey        LD   DE,#409           ; preload to save byte
            LD   A,H               ; fetch dy
            ADD  A,D               ; add 4 to reduce test
            CP   E                 ; test within range
            JR   NC,illmove
            LD   A,L               ; same for dx
            ADD  A,D
            CP   E
            JR   NC,illmove
            LD   (dydx+1),HL       ; set new dydx
illmove     JP   dydx              ; do next gameloop


; now all hires routines. Set in shortest order
```

```
space       EQU  #4235-$           ; remaining room for code

            DEFS space             ; nothing left (0)


hitop       DEFB #DD
            LD   L,hiret*256/256
            NOP
            LD   A,(HL)            ; fetch index
            INC  HL                ; point to next
            OR   A
            RET  Z                 ; part done when 0
            DEFB #DD
            LD   L,hitop*256/256
            EX   (SP),HL
            EX   (SP),HL
            EX   (SP),HL
            EX   (SP),HL
            JP   hibot+#8000       ; do display

high        LD   R,A               ; first part
line1       DEFB 16,0,0,0,0,0,0,0,0,0,0,0,0,1 ; hidden in Buffer
            LD   A,(HL)            ; load second index
hibot       LD   R,A               ; second part
line3       DEFB 0,0,0,0,0,0,0,0,0,0,0,0,0,0
            JP   (IX)              ; back to low memory


hiret       INC  HL                ; again timingdelay
            LD   A,(HL)
            LD   A,(HL)
            NOP
himid       NOP
            LD   A,(HL)            ; fetch index
            INC  HL                ; point to next


            OR   A
            LD   E,(HL)            ; timing only
            JP   NZ,high+#8000     ; do mid disp
            LD   IX,hitop          ; now do bottom part
            LD   A,(HL)
            RET  NZ                ; timing, not true
            JP   hibot+#8000       ; do bottom display

; graphical data and calculation data

curstab     DEFW 1,1,1,1,1,1,1,1

line4       DEFB 31,255,255,255,255,255,255,255,255,255,255,255,255
            DEFB 255,255


line5       DEFB 16,0,0,0,1,255,255,255,255,240,0,0,0,1

line6       DEFB 31,255,255,255,255,255,255,255,255
            DEFB 240,0,0,0,1
line7       DEFB 16,0,0,0,001,255,255,255,255
            DEFB 255,255,255,255,255

linec1      DEFB 32,0,0,0,0,1,0,0,0,0,0,0,0,1
linec2      DEFB 32,0,0,0,0,2,0,0,0,0,9,0,0,1
```

```
linec3     DEFB 32,0,0,0,0,4,0,0,0,0,6,0,0,1
linec4     DEFB 32,0,0,0,0,8,0,0,0,0,0,0,0,1

; the screendata, first with data to copy over sysvar

screen     DEFB line3*256/256
           DEFB line3*256/256
           DEFB line3*256/256
           DEFB line4*256/256
topblock   DEFB line1*256/256


init       LD   DE,#4000          ; shifted marble to sysvar
           LD   HL,marbletab
           LD   BC,30
           LDIR
           LD   HL,marb0          ; first marble too
           LD   DE,nxtlin         ; due to free size elsewhere
           LD   C,4
           LDIR

           LD   HL,init2
           PUSH HL
           LD   HL,topblock
           LD   DE,init

size       EQU  here -init
           LD   C,size
           JP   #19F9             ; repair init code


here       DEFB line5*256/256     ; small part uncompressed
           DEFB 0
           DEFB line6*256/256
           DEFB line7*256/256
           DEFB line6*256/256
           DEFB line7*256/256
           DEFB line6*256/256
           DEFB line7*256/256
           DEFB line6*256/256
           DEFB line7*256/256

           DEFB line1*256/256

init2      LD   HL,shrtscr        ; now decompress screen
           LD   DE,eshrt
dec1       LD   B,(HL)
           INC  HL
           LD   A,(HL)
dec2       LD   (DE),A
           INC  DE
           DJNZ dec2
           INC  HL
           LD   A,(HL)
           INC  A
           JR   NZ,dec1


           LD   HL,begin          ; now only repair init2
           PUSH HL                ; same way as init
           LD   HL,init2-1        ; only now start game
           LD   DE,init2

size2      EQU  eshrt-init2
```

```
           LD   BC,size2
           LD   IX,hr
           JP   #19F9


shrtscr    DEFB 1,line7*256/256
           DEFB 1,line6*256/256
           DEFB 66,line1*256/256
           DEFB 1,line6*256/256
           DEFB 1,line7*256/256
           DEFB 1,0
           DEFB 4,line5*256/256
           DEFB 31,line1*256/256
           DEFB 1,line4*256/256
           DEFB 3,line3*256/256
           DEFB 1,0
           DEFB 255

eshrt      EQU  $

basic      DEFB 0,1               ; only used to start program
           DEFW lenbas-$
           DEFB 249,212,28
           DEFB 126
           DEFB 143,0,18,0,0
dfile      EQU  $
lenbas     DEFB 118,0,0

marb0      DEFB %01100000,0
           DEFB %10010000,0


marbletab  DEFB %00110000,0       ; 1
           DEFB %01001000,0       ; 1
           DEFB %00011000,0       ; 2
           DEFB %00100100,0       ; 2
           DEFB %00001100,0       ; 3
           DEFB %00010010,0       ; 3
           DEFB %00000110,0       ; 4
           DEFB %00001001,0       ; 4
           DEFB 3,0               ; 5
           DEFB 4,128             ; 5
           DEFB 1,128             ; 6
           DEFB 2,64              ; 6
           DEFB 0,192             ; 7
           DEFB 1,32              ; 7
           DEFB 0,0


vars       DEFB 128
?
last       EQU  $
```