```
; SHOGUN 1K ZX81
; problems to expect
; size, original prog needs 3x64 bytes checkmemory, screen also        v
; zx81 must fit 80 bytes screen and 2x64 bytes memoryscreen      v
; stack needs room for recursion, 70 bytes minimal               v

; display, in 1 character difference normal and shogunstone and value
;          solved by 1,2,3,4 and inversion, shogun as I and Z for 1 and 2


; 12 bytes from #4000 to #400B free reuseble for own "variables"

      org #4009

fields equ #4400-64
bckfld equ fields-64

; in LOWRES more sysvar are used, but in this way the shortest code
; over sysvar to start machinecode. This saves 11 bytes of BASIC

d_file      equ basic+3

; DO NOT CHANGE AFTER BASIC+3 (=DFILE)
basic   ld h,dfile/256              ; highbyte of dfile
        jr init1

      db 236                        ; BASIC over DFILE data

keysys      db 212,28
      db 126,143,0,18

eline dw last
clhl  dw last-1
        db 0,0,0,0,0,0              ; x
berg  db 0                    ; x

mem   db 0,0                        ; x OVERWRITTEN ON LOAD

init1 ld l, dfile mod 256           ; low byte of dfile
      jr init2

lastk db 255,255,255
margin  db 55

nxtlin  dw basic              ; BASIC-line over sysvar

flagx   equ init2+2
init2 ld (basic+3),hl               ; repair correct DFILE flagx will be set
with 64, the correct value

      ld h,vars/256
      db 0,0                        ; x used by ZX81, not effective code after
loading

      ld l,vars mod 256
frames  db #37                      ; 1 more than actual command
      db #e9                        ; this must have bit 7 set

      jp gameini              ; YOUR ACTUAL GAMECODE, can be everywhere

      dw 0

cdflag  db 64
; DO NOT CHANGE SYSVAR ABOVE!
```

```
pieces     DEFB %00011011
           DEFB %11100100
           DEFB %11001001
           DEFB %01010000     ; shogun
           DEFB %10110100
           DEFB %10110001
           DEFB %00100111

rseed      DEFB 0

           DEFB %10000111
         DEFB %10001101
         DEFB %01001011
           DEFB %00010100     ; shogun

         DEFB %10011100
         DEFB %01001110
           DEFB %11100001

; field
; adcvvvii
; a = attacked
; d = defended
; c = colour opponents
; vvv = indexvalue of piece 1..7 = 7 pieces, 1 doubled
; ii = indicator of value from piece 0..3 = 4 values

; Shogun is always 00010000
; Field in use when AND 00011100 returns a value

; free codeable memory

; erase calc fields
ertop LD   HL,fields
erfld   ld a,(hl)
      and 63                  ; take of attacked and defended
      LD   (HL),a
        INC  L                      ; end of field is at end of 256 bytes
        JR   NZ,erfld
;        RET                              exit through dwn saves a byte, B not
needed

dwn        CP   b                   ; opcode 88 used as used value when move is
out of board
           RET  Z
           INC  B
           RET

rgt        CP   c             ; opcode 89 used as used value when move is out of
board
           RET  Z
           INC  C
           RET
; false move is always Z-flag set



showwin     call showb-3

start       in a,(254)         ;
            and 4              ; 3 d x 8 i k m to start the game
            jr nz,start
```

```
            ld hl,fields
clfld       ld (hl),255       ; all fields cleared
            inc l
            jr nz,clfld

            LD   B,64
; 64 fields with 0-3
; rnd number here
nfld        call nxtlin       ; get a random nr
frnd        sub b
            jr nc,frnd        ; from 64 down to 1
            adc a,b                ; a = 1 to B

newl        LD   HL,fields-1
fempty      INC  HL
            LD   E,(HL)
            INC  E                 ; find empty field
            JR   NZ,fempty

            DEC  A                 ; until random steps are done
            JR   NZ,fempty

            LD   A,B
            AND  3                 ; value pointer only
            LD   (HL),A            ; set pointer on field in memory
            DJNZ nfld         ; do all 64 fields, 16x0 16x1 16x2 16x3

; pieces on the board
            ld d,32                ; colour player
              LD   E,B               ; start pointer player
            LD   B,7               ; bottom row
sstone      LD   C,8          ; 8 pieces
srow        CALL #4000             ; calculate field
nxte             inc e
            ld a,e
              and 7
            jr z,nxte         ; keep E in range 1-7

            ADD  A,A
            ADD  A,A               ; move to stoneposition
            OR   (HL)              ; add fieldpointer

            or d              ; add player or computer

hla              LD   (HL),A       ; stone info in field
            LD   A,C               ; test end of row reached
            OR   A
            JR   NZ,srow
            ld d,c                 ; d now 0
            ld e,6                 ; start pointer computer
            OR   B                 ; A=0 B=?
            LD   B,C          ; Make B=0 for top row
            JR   NZ,sstone         ; second turn is exit

; BC always on top of board on move of player

; show board
loop        call showb        ; show board and count nr of stones
            ld a,256+22-29         ; "-" signal player lost
sw1         jr nc,showwin          ; player lost
scrfld           ld l,shogscr mod 256    ; H is already right value
            call #4004        ; use fieldroutine to calculate screenposition
            add a,b                ; screen has 1 position per line more
```

```
(Newline)
            ld l,a                  ; visible screen = 9 positions
            ld a,(hl)           ; get field

            xor 128                 ; invert display
            ld (hl),a           ; show field

; A never 255
            exx                 ; keep BC
w4up        inc a               ; test key down
w4dwn       ld bc,(lastk)           ; get keypress
            ld a,c                  ; inport to test
            jr nz,w4up          ; result from above, wait for no key down
            inc a               ; now key is up, test key down
            jr z,w4dwn          ; wait for key down
            call nz,#7bd            ; translate pressed key
            exx                 ; get BX

fir         dec a               ; test Z-key
            jp z,fire           ; test SELECT-key out of table

            ld d,4                  ; test 4 directions
            ld hl,keysys            ; start of table
fkey        cp (hl)                 ; test direction
            inc hl                  ; point to index of routine
            call z,clhl         ; if pressed goto direction
nodir       inc hl                  ; point to next direction
            dec d               ; test all directions
            jr nz,fkey          ; test 4 directions

nokey       jr loop                 ; the playloop


; Player destination reached and selected jumps to F2
f2          BIT  5,(HL)             ; on own stone
            JR   NZ,nokey           ; if so no move

          XOR  A
            LD   (fsel+1),A         ; always undo selected

          bit 7,(hl)                ; not in reach?
       ld a,256+15-29               ; preload "?"
          jr z,noshat               ; not a valid move but stone is deselected,
show error


          ex de,hl              ; DE now destination
frompos         ld hl,0             ; get from position

          call domove               ; move the stone
          call showb                ; show the new board, count stones
          ld a,256+21-29        ; "+" signal player won
          jr nc,sw1                 ; only computer loses stones, so player
should win

; player move now ready, so computer move starts here

       push bc                      ; save XY cursor
       call ertop               ; erase pointers attacked/defended
       call chckbrd-3               ; check pointers for all stones

       ld (bestsc+1),a              ; reset best score index, reset best move

; save current pointers for all possible moves
```

```
      ld hl,fields
      ld de,bckfld                  ; backup position of the board
      call bc64                ; copy 64 bytes and make checkbrd react for
computer

      call chckbrd-3                ; now find best move with current
stonepositions

; all checked, do the best move

bestto      ld de,0                       ; set by routine above
bestfrom ld hl,0              ; set by routine above
      call domove             ; make the computermove
      call chckbrd-3                ; check pointers for all, but we need
shogunp layer only
      ld a,254                ; 27-29
shogpl      ld hl,fields                  ; position of shogun player, set
during testing
      bit 6,(hl)              ; player shogun attacked by computer?
      pop bc                        ; retrieve cursor
      jr z,noshat             ; if not, NO SHogun ATtack
      res 7,a                       ; signal attack in the display of the board
noshat      ld (empval+1),a               ; set attack or not

      jr nokey                ; continue for player

      ld a,csxy mod 256       ; entry computer move
chckbrd ld (doubuse+1),a            ; set which boardroutine to do
      LD   B,8                        ; 8 rows to check
cline   LD   C,8                      ; 8 columns to check
crow    dec b                 ; for field, B and C 1 less, here B-1
      CALL #4000                      ; Here C-1 and get boardfield

      ld a,(hl)               ; get value of the field
      and %00111100                 ; test used
      jr z,nfb                ; if not no test needed
      bit 5,a                       ; test player or computer stone
      ld a,64                       ; preset bit for computer
      jr z,cstone             ; if so, done
      add a,a                       ; make it bit player

cstone  ld (bitset+1),a             ; set bit on endposition for player or
computer
doubuse     call compin             ; set all fields attacked or protected,
computer find best
nfb     inc b                       ; undo DEC B from above
      ld a,c                        ; test end of row
      or a
      jr nz,crow              ; do next position on row
      djnz cline              ; do next line
      ret                     ; board now checked

domove  push de                     ; save destination
      ld a,(hl)               ; get fieldvalue from
      ld d,a                        ; save in D
      and 3                   ; only keep stonepointer
      ld (hl),a               ; erase stone
      ld a,d                        ; retrieve original value
      and %00111100                 ; stone value only
      ld d,a                        ; save again in D
      pop hl                        ; get destination
      ld a,(hl)               ; get fieldvalue to
      and 3                   ; only keep stonepointer
      or d                    ; add stone value
```

```
      ld (hl),a               ; move stone
      ld a,201                ; after a move depthsearch computer is turned off

      ld (depth),a
        ret                         ; move done


fire    CALL #4001                  ; FIELD-routine over sysvar
fsel  LD   A,0                        ; get fire status
        OR   A                      ; test value
        JP   NZ,f2                  ; fire pressed on possible end field
        BIT  5,(HL)                 ; my stone selected?
        jr z,loopjp                       ; false selected, empty or opponent
      ld (frompos+1),hl       ; save from position
        LD   A,128                      ; preset for bitset but also not zero
      LD   (fsel+1),A             ; signal stone selected, not zero written
        LD   (bitset+1),A           ; preset check on player fields in reach
      exx                     ; keep HL without use of stack
      call ertop              ; erase previous pointers set
      exx                     ; get field
      call csxy               ; set all destinations in reach
loopjp      jp loop                         ; continue the player loop

ndir  ld a,24                       ; 4 directions ; 4 3 2 1
dird    ld d,a                      ; save remaining directions
      push de                       ; save remaining directions and info
      push bc                       ; save position

      ld a,e                        ; get direction info and step counter
      and %00111000                 ; current active direction only
      cp d                    ; test current direction
      jr z,nodel              ; same direction, move allowed

      xor 8                   ; make it opposite direction
      cp d
      jr z,nomove             ; backwards is not allowed

      bit 7,e                       ; we have a different direction, change of
direction allowed?
      jr z,nomove             ; if not, skip this direction

      ld a,e                        ; get check register
      sub 64                        ; take off changecounter
      and %11000111                 ; take out old dir
      or d                    ; put in new dir
      ld e,a                        ; save all

nodel LD   A,D                ; 24 16  8  0
      rra                     ; 12  8  3  0
      rra                     ;  6  4  2  0

      dec e                   ; we do a move
        LD   HL,keysys+1            ; preset movetable
        ADD  A,L
        LD   L,A
      call clhl                     ; do move up, down, left or right
      jr z,nomove                   ; out of board
        call #4001                  ; get field when on screen
      ld a,e
      and 7
      jr z,movend             ; we move to an end field, always allowed even
with stone
      ld a,(hl)                     ; fieldvalue or u/d/l/r first byte
      and %00011100                 ; field in use or out of board?
```

```
movend      CALL Z,moves                ; only do valid moves, check
recursively

nomove  POP  BC                     ; undo move
       POP  DE                      ; undo step or change in dir

       ld a,d
       sub 8
         JR   NC,dird               ; test each field 4 directions done
         RET                            ; end of this field test

csxy  ld (fromfld+1),hl       ; save where your stone started
compin  ld a,(hl)             ; get fieldvalue
       and %00011100                ; stone only
       cp  %00010000                ; test for shougun
       jr nz,noshog                 ; if not, no save needed
       ld de,shogpc+1               ; preset for computer, needed to test
attacked
       bit 5,(hl)             ; test player vs comnputer
       jr z,savshog
       ld de,shogpl+1               ; player shogun, needed to signal attacked
savshog     ld a,l                      ; get position
       ld (de),a              ; save on right place
noshog      call stval             ; get stone value

       add a,%11111001              ; add changecounter, impossible direction
and 1 step extra
         LD   E,A                       ; change (192), current dir (56), nr
of moves (1-4)
; ccdddvvv
; cc change allowed, bit 7 on, bit 6 extra step allowed
; ddd current direction
; vvv nr moves

moves       LD   A,E
          and 7
            JR   NZ,ndir            ; still moves left, repeat recursively

; endposition reached
          ld a,(hl)                 ; get field
bitset      OR   0             ; add reachable for player or computer
            LD   (HL),A                 ; save field

depth       RET                       ; player now ready, but computer can go on.
changed in code

; computer reached a destination, now we check the score of the move
; 255 = hit shogun                        (before move)     v
;  67 = shogun not attacked               (after move)    v.
;  32 = shogun attacked                   (before move)     v
;  16 = hit other piece                   (before move)     v
;   8 = attacked before move              (before move)     v
;   4 = current stone not attacked        (after move)    v
;   2 = protected after move by own stone       (after move)    v

       xor a                    ; reset score of current move
       ld (seta+1),a

       ld a,(hl)
       and %00011100                ; test used
       jr z,empok               ; an empty field is ok to go to

       bit 5,(hl)
       ret z                    ; however you can't go on own stone
```

```
; test from position
empok exx                   ; save HL as destination
fromfld     ld hl,0
      bit 5,(hl)            ; due to double use we could have started a human
stone
      ret nz                      ; which we DON'T test

      ld (fromcp+1),hl      ; Now we know where computer started

; 32 = shogun attacked
;  8 = normal stone
      ld de,#2008           ; preset shogun or normal
      bit 7,(hl)            ; test bit 7, field attacked before move
      call nz,setnas              ; if so, set attackscore
      exx                   ; back to destination

; now we test destination before move
; 255 = hit shogun
;  16 = hit other piece
      ld de,#ff10           ; hit shogun or other stone
      call setnas           ; test a piece here

; do move
      ld (topos+1),hl             ; not best move, that is after test
      ex de,hl              ; DE-stination
      ld hl,(fromcp+1)      ; get from again
      call domove           ; do the move

; depth=RET

      call ertop            ; erase current steps
      ld a,compin mod 256         ; do the second check for computer
      call chckbrd                ; set new steps after move (double
recursive, large stack)

;  67 = shogun not attacked
shogpc      ld hl,fields                ; get position of shogun
      bit 7,(hl)            ; test attacked after move
      ld a,h                      ; preset score
      call z,seta           ; not attacked, add score

;  4 = dest field is not attacked
topos ld hl,0
      bit  7,(hl)                 ; test attacked after move
      ld a,4
      call z,seta           ; if not, set score not atttacked

; extra AI not in original
;  2 = protected after move by own stone
      bit 6,(hl)            ; test destination is protected by other stone
      ld a,2
      call nz,seta                ; if so add 2 points of score

; score analys
       ld a,(seta+1)              ; get score current move
bestsc   cp 0                     ; test against current best
       jr c,worse                 ; not better means exit
       ld (bestsc+1),a      ; set new best score
       ld (bestto+1),hl           ; save current TO   as best
fromcp   ld hl,0            ; get from position current move
       ld (bestfrom+1),hl         ; save current FROM as best

; undo move
```

```
worse ld de,fields
      ld hl,bckfld                ; get copy to undo changes

bc64  ld bc,64
      ldir

; check next moves, set pointers back needed
; depth= test AI move, also save start
      xor a                   ; computer goes on where player stops
      ld (depth),a
      ld a,csxy mod 256       ; set test without overwriting data
      ld (doubuse+1),a
      ret

setnas      ld a,(hl)               ; get current stone
      and %00011100                 ; stone value and colour only
      ret z                   ; no stone exit
      cp  %00010000                 ; is it player shogun
      ld a,e
      jr nz,seta              ; other stone is attacked
      ld a,d                        ; shogun attacked
seta  or 0                    ; mix with current score
      ld (seta+1),a                 ; save new score
      ret

            ld (empval+1),a
showb       push bc                 ; save pointer
            ld hl,fields            ; the memoryboard
            ld de,shogscr           ; the screenboard
            ld b,d                  ; set counter player stones
            ld c,d                  ; set counter computer stones

setline     ld a,(hl)           ; get field
            and %00011100           ; test used
            jr z,notused            ; no stone is not counting
            cp %10000           ; is it shogun?
            jr z,shst           ; if so add 16
            ld a,1                  ; otherwise add 1
shst        bit 5,(hl)          ; player or computer
            jr z,addc           ; set computer value
            add a,b                 ; add player
            ld b,a                  ; save player
            db #ca                  ; JP Z, never true, saves a byte from JR
NOTUSED
addc        add a,c                 ; add computer
            ld c,a                  ; save computer

notused           call stval        ; get value of the stone
stret       jr nz,stfnd         ; NZ = not shogun
            or a                ; test 0
            ld a,"I"-56         ; if so show I
            jr z,stfnd          ; shogun value 1=I
            ld a,"Z"-56         ; otherwise shogun value 2=Z
stfnd       add a,29            ; make is ZX81 ascii number
            bit 5,(hl)          ; inverted display player1 or player2
            jr z,fldval
            set 7,a                 ; invert display
fldval            ld (de),a         ; show field, empty or stone

            inc l               ; next memory field, not HL, H=#43 elsewhere
needed
            inc de                  ; next screen field
            ld a,l
            and 7
```

```
            jr nz,setline     ; still on current line

            or l              ; test end of board
            inc de                 ; step over Newline
            jr nz,setline          ; do next line

            ld a,254          ; 27-29
            ld (empval+1),a        ; possible signal reset

            ld a,17+#43       ; minimal score to play
            cp b
            jr nc,dead1       ; too few stones, 2 or shogun lost
            cp c              ; same test other player
dead1       pop bc                 ; retrieve cursor position
            ret               ; display ready

stval       ld a,(hl)         ; get field
            exx               ; save HL
            ld b,a                 ; double copy of value
            and %00011100          ; test used
            jr z,empval       ; not used
            xor %00010001          ; add 1, swap shogun, only shogun will make
it 1
            ld c,a                 ; save result
            ld a,b                 ; get screen
            rra               ; divide by 4
            rra
            and 15                 ; stone position only
            ld hl,pieces-1         ; index to stones
            add a,l                ; add stonenumber
            ld l,a                 ; HL now stoneposition

            ld a,b                 ; get screen
            and 3             ; position within stone only
            inc a             ; position in stone >0

            ld b,a                 ;
            ld a,(hl)         ; get stonetable
ffld        rrca
            rrca              ; rotate to right value
            djnz ffld         ; rotate to position

            and 3             ; value of position
            dec c             ; test shogun
            db 1              ; hide empty value
empval          or 254             ; 27-29
              exx
            ret

      block #4380-74+gameini-dfile-$,0 ; room for stack

; 29 bytes once used in SP-area or size program is too large
gameini ex af,af'
      ld hl,rnd1              ; random routine
      ld de,nxtlin                 ; to be set over sysvar
      ld bc,18                ; copr random routine
      ldir

      ld hl,field             ; field routine over sysvar
      ld c,12
      ld e,b                       ; DE=#4000
      ldir

      ld hl,keytab                 ; key table and 2 directions
```

```
      ld c,23
      ld e,keysys mod 256

      jr ini4                       ; continue after screen

; the display file, Code the lines needed.
dfile       db 118
shogscr     db "S"-27,"H"-27,"O"-27,"G"-27,"U"-27,"N"-27,29,"K"-27,118
      db 21,"Y"-27,"O"-27,"U"-27,0,"W"-27,"I"-27,"N"-27,118
      db 22,"A"-27,"I"-27,0,"W"-27,"I"-27,"N"-27,"S"-27,118
      db 27+128,"S"-27,"H"-27,"O"-27,"G"-27,"U"-27,"N"-27,0,118
      db "A"-27,"T"-27,"T"-27,"A"-27,"C"-27,"K"-27,"E"-27,"D"-27,118
      db "C"-27,"O"-27,"N"-27,"T"-27,"R"-27,"O"-27,"L"-27,"S"-27,118
      db "Q"-27,"A"-27,"O"-27,"P"-27,0,"Z"-27,0,"X"-27,118
      db 16,"C"-27,17,0,30,28,30,29,118

; this byte fills the unused part of the screen
      db #e9                        ; JP (HL) is screenfiller

ini4  ldir                    ; copy keytable

      ld sp,dfile             ; set SP where needed
; 42 bytes saved in normal coding memory
      jp start                ; start the game

; routines placed over sysvar
rnd1  ld hl,rseed       ; seed pointer
      ld a,(hl)   ; get seed RRCA ; a=a/2
      rrca        ; a=a/2
      rrca
      rrca
      xor 31            ; swap low bits
      add a,(hl)  ; add seed
      db 17             ; hide frames in DE
      dw 65535    ; frames used by zx81
      add a,e     ; add framecounter
      ld (hl),a   ; save new seed
      ret

field      DEC  C
         ld hl,fields
           LD   A,B
           ADD  A,A
           ADD  A,A
           ADD  A,A
           ADD  A,C
         add  a,l
           LD   L,A
           RET

upt   equ keysys+up-keytab
lftt  equ keysys+left-keytab

keytab      db 5-1
      db dwn mod 256
      db 10-1
      db upt mod 256
      db 26-1
      db lftt mod 256
      db 25-1
      db rgt mod 256

      ld a,7
      ld l,(hl)
```

```
      jp (hl)

up    xor a       ; opcode AA
      cp b
      ret z
      dec b
      inc a       ; undo possible z-flag
      ret

left       xor a  ; opcode AA
         cp c
           RET  Z
           DEC  BC               ; c>0, never dec c, no flag
           RET

vars    db 128
last  equ $

end
```