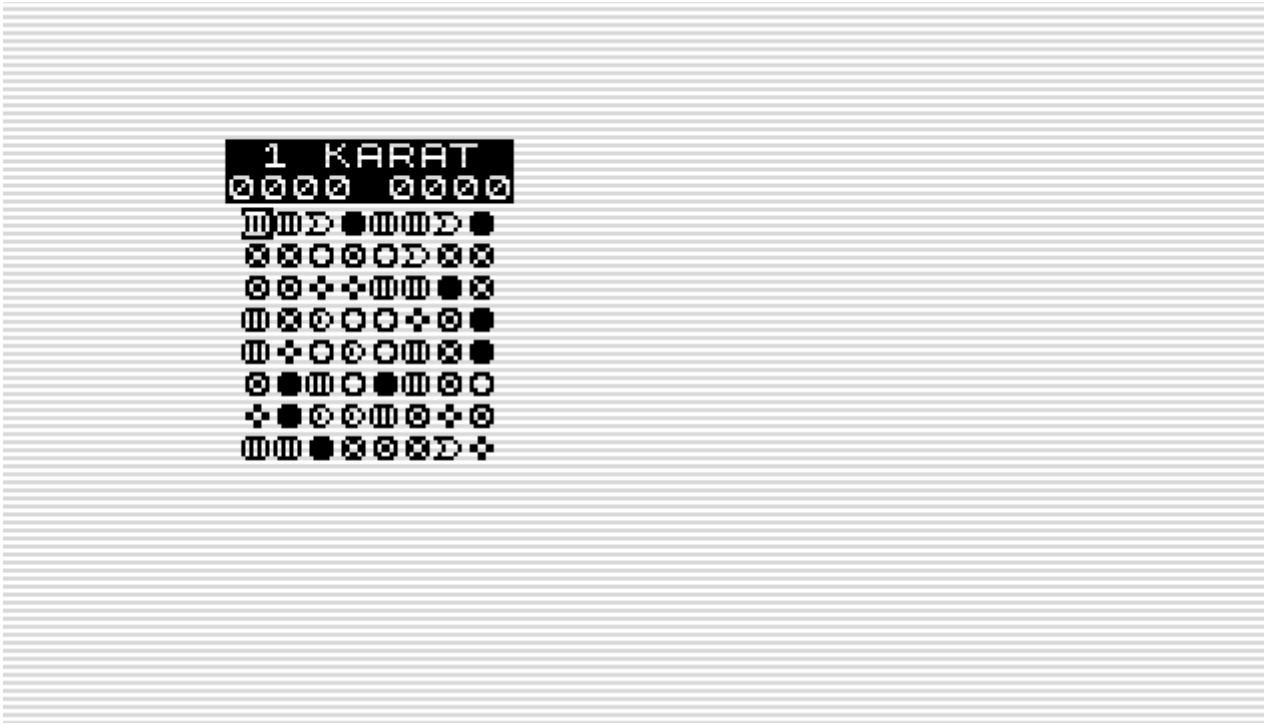
1 Karat



Simon Dwyer set the base with his “chain reaction”. Altering and optimizing his code gave room to add a hires screen as a shell over the game.

```
; 1 Karat
; Game 62 in 1K hires for the ZX81

; This game is based on Simon Dwyer's game 1K CHAIN REACTION.
; His game is optimizied in size and a hires screen is is coded as a shell over
; the lowres screen.
; Furthermore the setup to the start is now done offscreen.
; Controls are set to QAOPZ


      org #4009

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

basic   ld h,dfile/256              ; highbyte of dfile
        jr init1

      db 236                        ; BASIC over DFILE data
rtab  db 212,28               ; rotation table goes over sysvar
      db 126,143,0,18

eline dw last
chadd 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
CURSOR:
      jr init2

lastk db 255,255,255
```

```
margin  db 55

nxtlin  dw basic              ; BASIC-line over sysvar

      db 0,0
flagx db 0

      db 0,0,0                ; x
      db 0,0,0                ; x

frames  dw 65535

init2 jp init4                ; overwrite sysvar
      db 0,0

cdflag  db 64

lbuf       LD   R,A
           DEFB 0,0,0,0
         defb 0,0,0,0
           RET


; start is used by restart with key 1 only
SWAP:
; on entry A not equal 255 so INC A will not set zero flag
      inc a             ; test a key pressed

NOSWAPKEY:

      ld bc,(lastk)
      ld a,c                    ; get last key in A
      jr nz,SWAP        ; wait until no key pressed
      inc a             ; test key down
      jr z,NOSWAPKEY          ; wait until key pressed

      push hl
      CALL $07BD
      pop hl

      call udlr
      ret nz

SWAP_ITEMS:
;HL CONTAINS WHERE ITEM IS GOING TO
      LD A,(HL)



      AND $C0 ;INVALID SWAP IS OUTSIDE PLAY AREA
      RET NZ

      ld (flag1+1),a
      LD b,h
      ld c,l

      LD HL,(CURSOR)

      LD A,(BC)
      LD D,A
      LD A,(HL)
      LD (BC),A
      ld (hl),d
```

```
      push hl
      CALL DROP_AND_BOOM
      pop hl

flag1 LD A,0
      or a

      ret nz            ; swap did work

      ld a,(lives)
      dec a
      ld (lives),a
      cp $9c
      ret nz
      pop hl            ; drop call return

; from above it is gameover.

      ld hl,dispsc-1
      ld de,hiscore-1
      ld bc,5
same  inc hl
      inc de
      dec c
      ld a,(de)
      cp (hl)
      jr z,same
      call c,#19f9
start:
      ld a,(lastk)
      cp 191
      jr nz,start

      ld hl,hr2
      ld (hr+1),hl

      LD HL,ONEKDISPLAY+1

      LD (CURSOR),HL


      LD B,$48    ;CLEAR SCREEN
CLEARLOOP:
      LD A,(HL)
      CP $76
      JR Z,NOCLEAR
      LD (HL),$00
NOCLEAR:
      INC HL
      DJNZ CLEARLOOP

      CALL DROP_AND_BOOM

      ld hl,#9c9c ; inverted "00"
      ld (dispsc),hl    ; reset score
      ld (dispsc+2),hl

      ld a,#9d
      ld (lives),a

      ld hl,hr1
      ld (hr+1),hl
```

```
GAMELOOP:

      LD HL,(CURSOR)
      SET 7,(HL)
      push hl
      CALL delay

      ld bc,(lastk)
      ld a,c
      inc a

      CALL nz,$07BD

;A REGISTER NOW CONTAINS KEY CODES or value 0 when no key pressed
;$0F = 1,$13=5,$18=6,$17-7,$16=8,$14=0
      pop hl
      RES 7,(HL)

      call udlr

;     CP $14 ;SELECT
      dec a       ; Z
      CALL Z,SWAP

ASSESS_CURSOR:
      LD A,(HL)
      AND $C0
      JR NZ,NOKEY
      LD (CURSOR),HL

NOKEY:
      JR GAMELOOP

udlr  CP 26 ; LEFT=O
      JR NZ,KDOWN
      DEC HL
      ret

KDOWN:
      LD DE,$09   ; preset DE for UP and down
      CP 5 ; DOWN=A
      JR NZ,KUP

      ADD HL,DE
      xor a
      ret

KUP:
      CP 10 ; UP=Q
      JR NZ,KRIGHT
      sbc hl,de   ; saves 2 bytes Carry is off
      xor a
      ret

KRIGHT:
      CP 25 ; RIGHT=P
      ret nz
      INC HL
      ret



DROP_AND_BOOM:
;POPULATE AND DROP ITEMS
```

```
;REPEAT UNTIL NO MORE ITEMS TO DROP

CHAIN_REACTION:
      xor a             ; reset loop points
      ld (MULTIPLIER+1),a

POPULATE_TOP:
      call delay
      LD HL,ONEKDISPLAY

TOPROW:
      INC HL

      LD A,(HL)
      CP $76
      JR Z,DROP_ITEMS
      or a
      JR NZ,NOTBLANK


RANDOM:
;GENERATE RANDOMISH NUMBER INTO A REGISTER

      ld a,(frames)           ; make it also time random
rndseed     add A,$44
      LD D,A
      RRCA ; multiply by 32
      RRCA
      RRCA

      XOR $1F

      ADD A,D
      SBC A,$FF ; carry
      ld (rndseed+1),a

      AND $07

NOTZERO:
      inc a
;     ADD A,16
      LD (HL),A

NOTBLANK:
      JR TOPROW


DROP_ITEMS:
;DROP ITEMS, ROUTINE IS FINSHED WHEN NO DROPS ARE POSSIBLE

      LD HL,ONEKDISPLAY+$3E
      LD DE,ONEKDISPLAY+$3E+$09
;DE CONTAINS LOWER ROW, HL THE ROW ABOVE

      LD BC,$3F01 ;NUMBER OF SCROLLS TO DO
NEXTSCROLL:
      LD A,(DE)
      or a
      JR NZ,NOSCROLL ;DE CONTENTS NOT EMPTY
      LD A,(HL)
      LD (DE),A
      LD (HL),$00

      inc c
```

```
NOSCROLL:
      DEC HL
      DEC DE
      DJNZ NEXTSCROLL

      dec c
      jr nz,POPULATE_TOP

;END OF DROP

;GO THROUGH THE MATRIX AND FIND ROWS OF THREE, MARK THEM AND LOOK FOR LINKED OF
SAME TYPE
BOOM:

      ld c,2
ch2   LD HL,ONEKDISPLAY

      LD B,$08

SEARCHLOOP_VERTICAL:
      inc hl

SEARCHLOOP_HORIZONTAL:

;SEARCH 8 DIRECTIONS, SET BIT 7 IF LINE FOUND

;HL NOW POINTS TO SQUARE INSIDE THE MATRIX. SEARCH AROUND FOR IDENTICAL NUMBERS
      ld a,c
      dec a
      jr nz,norem
      BIT 7,(HL)
      JR Z,norem
      LD (HL),a
      PUSH HL
      LD HL,MULTIPLIER+1
      INC (HL)
      POP HL
      jr check

norem ld de,1
      call find_line
      ld e,9
      call find_line

check inc hl

      ld a,(hl)
      cp $76
      jr nz, SEARCHLOOP_HORIZONTAL


      DJNZ SEARCHLOOP_VERTICAL
      dec c
      jr nz,ch2

MULTIPLIER:
      ld a,0
      or a
      ret z       ;

      ld (flag1+1),a    ; signal swap gave score
      ld b,a            ; number of increases to B
      cp 4
```

```
      jr c,addsc
      ld hl,lives
      ld a,(hl)
      cp $9f
      jr z,addsc
      inc (hl)

addsc ld hl,dispsc+4
      defb 17           ; hide ten in LD DE,NN
ten   ld (hl),28+128    ; inverted "0"
      dec hl
      inc (hl)
      ld a,(hl)
      cp 38+128   ; from "9" to "10"?
      jr z,ten
      djnz addsc

      JP CHAIN_REACTION

find_line:
      push hl
      ld a,(hl)
      add hl,de
      xor (hl)
      and #7f
      ld a,(hl)
      jr nz,nomatch
      add hl,de
      xor (hl)
      and #7f
nomatch     pop hl
      ret nz
      push hl
      set 7,(hl)
      add hl,de
      set 7,(hl)
      add hl,de
      set 7,(hl)
      call delay
      pop hl
      ret


; the hires screen is like a shell around the lowres screen.
; The lowres screen now holds character 0 to 8.
; This is translated to a UDG on the hires screen which will be displayed by
; the intruptroutine.
; Therefore just before the intrupts (the DELAY with FRAMES)
; the hires screen is built

delay exx
      ld hl,screen
      ld de,dfile+1
copyline:
      ld a,(de)   ; get udgpointer
      ld bc,#4040 ; preset on space
      or a
      jr z,udg
      dec a
      add a,a           ; double and test bit 7
      ld c,a            ; save double

      sbc a,a           ; bit 7 becomes 255 when set
      ld (invert+1),a   ; set invert value or not
```

```
      rlc c       ; c =4*(a-1)

udg   push hl           ; save current screen
      xor a       ; first time add 0
copyudg:
      add a,l
      ld l,a
      ld a,(bc)   ; get UDG
invert      xor 0       ; invert when needed
      ld (hl),a   ; write UDG
      inc c
      ld a,c
      and 3       ; do all 4 UDG values
      ld a,8            ; preset to next line
      jr nz,copyudg
      ld c,l            ; save current endvalue
      pop hl            ; get original screen pointer
      inc de            ; point to next udgpointer
      ld a,(de)
      cp #76            ; test end of line
      inc hl            ;`point to next screenpos
      jr nz,copyline
      inc de            ; skip newline
      ld l,c            ; get end of screenline
      inc l       ; point to next line on screen
      jr nz,copyline    ; do all 8 lines
      exx


      ld a,251
      ld hl,frames
      add a,(hl)
wfr   cp (hl)
      jr nz,wfr

      ret


; HR must be disabled on set up, this must be done here....

displow LD   A,#1E
        LD   I,A
        LD   A,#Fb
        jp #2B5


hr    jp hr1

hr1   LD   HL,hrlow+#8000
        LD   BC,#311           ; the lowres print
      call displow

           LD   B,8
hr01       DJNZ hr01

           LD   A,screen/256
           LD   I,A               ; highbyte screen
;           LD   A,screen mod 256  ; lowbyte screen
      ld a,(hl)
      xor a

           LD   B,8               ; 7 rows
loop7      LD   C,2               ; of 2 repeating
           LD   D,8               ; forward and backward
```

```
loop2      LD   E,4              ; groups of 4 lines
loop4      LD   L,A              ; save current screenpos

           PUSH HL               ; timing delay
           POP  HL
           DEC  HL
           INC  HL

           LD   A,L              ; display current line
           CALL lbuf+#8000

           ADD  A,D              ; calculate next line
           DEC  E

           CALL NZ,del1          ; timing delay

           JR   NZ,loop4         ; go forward
           LD   A,L              ; back to startpos lastline
           LD   D,255-7          ; -8 = backwards display
           DEC  C

           CALL NZ,del2          ; timing delay

           JR   NZ,loop2         ; do backwards
           ADD  A,32             ; goto next line of data
           DJNZ loop7            ; do all 7 lines

hr03  CALL del1              ; screen fill delay
           DJNZ hr03

exit       CALL #292             ; end of hires handling
           CALL #220
           LD   IX,hr
           JP   #2A4


hr2   ld hl,wait+#8000
      ld bc,#249        ;a9
      call displow
      jr hr03



del1       EX   (SP),HL
           EX   (SP),HL
           DEFB 38               ; ld h,n
del2       DEC  HL
           DEC  HL
           RET



;POSITION CURSOR VALUE IS SWAPPING WITH

hrlow db $76
      db $80
lives DB $9d,$80,48+$80,38+$80,55+$80,38+$80,57+$80,$80,$76
hiscore db $9c,$9c,$9c,$9c,$80
dispsc      DB $9C,$9C,$9C,$9C,$76


dfile:
ONEKDISPLAY:
      DB $76
```

```
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00,$76
      DB $00,$00,$00,$00,$00,$00,$00,$00


wait  DB $76
      db #b8,#aa,#b9,#b9,#ae,#b3,#ac,#80,#b8,#a8,#b7,#Aa,#aa,#b3
      db $76


space equ #4300-$
      defs space

screen  defb 0,60,102,90
      defb 0,56,100,82
      defb 0,60,102,66
      defb 0,248,68,34
      defb 0,24,24,102
      defb 0,60,90,102
      defb 0,124,170,170
      defb 0,60,126,126

init4 ld hl,screen
      ld sp,hl
      push hl
      ld de,#4000
      ld bc,32
      ldir              ; UDG over sysvar

      pop hl

      ld (hl),b
      ld de,screen+1
      ld c,255

      ld ix,hr
      scf

      JP start-3



vars    db 128
last  equ $

end
```