```
 1: ;
 2: ;       title   DRIVER19.ASM    H19 Driver in Assembly Language
 3: ;
 4: ;*************************************************
 5: ;*      last update: January 29, 1982             *
 6: ;*      programmer : Les Bird                     *
 7: ;*      company    : CompTec Software Dept.       *
 8: ;*      language   : 8080 Assembly                *
 9: ;*      O S        : CP/M ver 2.0                 *
10: ;*************************************************
11: ;
12: ; H19 escape codes
13: ;
14: esc     equ     1bh     ; escape
15: hcuh    equ     'H'     ; home cursor
16: hcuf    equ     'C'     ; cursor right
17: hcub    equ     'D'     ; cursor left
18: hcud    equ     'B'     ; cursor down
19: hcuu    equ     'A'     ; cursor up
20: hri     equ     'I'     ; reverse index
21: hcpr    equ     'n'     ; cursor position report
22: hscp    equ     'j'     ; save cursor position
23: hrcp    equ     'k'     ; return to saved
24: hdca    equ     'Y'     ; direct addressing
25: hcd     equ     'E'     ; clear screen
26: hbd     equ     'b'     ; erase beginning
27: heop    equ     'J'     ; end of page
28: hel     equ     'l'     ; entire line
29: hebl    equ     'o'     ; beginning line
30: heol    equ     'K'     ; end of line
31: hil     equ     'L'     ; insert line
32: hdl     equ     'M'     ; delete line
33: hdch    equ     'N'     ; delete character
34: heim    equ     '@'     ; enter insert mode
35: herm    equ     'O'     ; exit insert mode
36: hram    equ     'z'     ; reset terminal
37: hsm     equ     'x'     ; set mode
38: hrm     equ     'y'     ; reset mode
39: herv    equ     'p'     ; reverse video
40: hxrv    equ     'q'     ; exit reverse video
41: hegm    equ     'F'     ; enter graphics mode
42: hxgm    equ     'G'     ; exit graphics mode
43: hf1     equ     'S'     ; f1
44: hf2     equ     'T'     ; f2
45: hf3     equ     'U'     ; f3
46: hf4     equ     'V'     ; f4
47: hf5     equ     'W'     ; f5
48: hf7     equ     'P'     ; blue
49: hf8     equ     'Q'     ; red
50: hf9     equ     'R'     ; white
51: offset  equ     1fh
52: ;
53: ; bdos entry points
54: ;
55: reboot: equ     0000    ; reboot system
56: bdos    equ     5
57: orgin   equ     100h
58: direct  equ     6
59: dinput  equ     0ffh
60: pstring equ     9
```

```
 61: clock:  equ     0bh
 62: cr      equ     0dh
 63: lf      equ     0ah
 64: bs      equ     8
 65: ap      equ     27h
 66: bel     equ     7
 67: cstat   equ     0edh    ; port 355Q
 68: cdata   equ     0e8h    ; port 350Q
 69: credy   equ     20h     ; crt ready bit
 70: dredy   equ     01h     ; data ready at keyboard
 71: ;
 72:         org     orgin
 73: start:  lxi     h,0
 74:         dad     sp
 75:         shld    oldstack
 76:         lxi     sp,stack
 77:         jmp     begin
 78: ; CLEAR DISPLAY
 79: cls:    push    h
 80:         lxi     h,101h
 81:         shld    curco
 82:         call    adj     ; incase on 25th line
 83:         mvi     a,hcd
 84:         call    process
 85:         lxi     h,memmap
 86:         shld    scrpnt
 87:         pop     h
 88: zsmem:  push    h
 89:         lxi     h,memmap
 90:         lxi     d,1920
 91: zsmem1: mvi     m,32
 92:         inx     h
 93:         dcx     d
 94:         mov     a,e
 95:         ora     d
 96:         jnz     zsmem1
 97:         pop     h
 98:         ret
 99: ;
100: ceol:   push    h
101:         mvi     a,heol
102:         call    process
103:         lhld    curco
104:         xchg
105:         lhld    scrpnt
106: eolloop:mvi     m,0
107:         inx     h
108:         dcr     d
109:         jnz     eolloop
110:         pop     h
111:         ret
112: crlf:   mvi     a,cr
113:         call    output
114:         mvi     a,lf
115:         call    output
116:         push    h
117:         lhld    curco
118:         mvi     h,1
119:         inr     l
120:         shld    curco
```

```
121:            pop     h
122:            ret
123: ;
124: insertl:
125:            mvi     a,hil
126:            jmp     process
127: ;
128: deletel:
129:            mvi     a,hdl
130:            jmp     process
131: ;
132: insertc:
133:            mvi     a,heim
134:            jmp     process
135: ;
136: deletec:
137:            mvi     a,hdch
138:            jmp     process
139: ;
140: xinsertc:
141:            mvi     a,herm
142:            jmp     process
143: ;
144: cup:      mvi     a,hcuu
145:            call    process
146: decrl:    call    savall
147:            lhld    curco
148:            dcr     l
149:            shld    curco
150:            lhld    scrpnt
151:            lxi     d,-80
152:            dad     d
153:            shld    scrpnt
154:            call    retall
155:            ret
156: ;
157: cleft:    mvi     a,hcub
158:            call    process
159: decrh:    call    savall
160:            lhld    curco
161:            dcr     h
162:            shld    curco
163:            lhld    scrpnt
164:            dcx     h
165:            shld    scrpnt
166:            call    retall
167:            ret
168: ;
169: cright:   mvi     a,hcuf
170:            call    process
171: incrh:    call    savall
172:            lhld    curco
173:            inr     h
174:            shld    curco
175:            lhld    scrpnt
176:            inx     h
177:            shld    scrpnt
178:            call    retall
179:            ret
180: ;
```

```
cdown:  mvi     a,hcud
        call    process
incrl:  call    savall
        lhld    curco
        inr     l
        shld    curco
        lhld    scrpnt
        lxi     d,80
        dad     d
        shld    scrpnt
        call    retall
        ret
;
home:   push    h
        mvi     a,hcuh
        call    process
        lxi     h,101h
        shld    curco
        lxi     h,memmap
        shld    scrpnt
        pop     h
        ret
;
process:
        push    psw
        mvi     a,esc
        call    output
        pop     psw
        jmp     output
hset:   push    psw
        mvi     a,hsm
hsetm:  call    process
        pop     psw
        call    output
        ret
;
clr25:  mvi     a,'1'
        call    hrset
        mvi     a,'1'
        jmp     hset
;
hrset:  push    psw
        mvi     a,hrm
        jmp     hsetm
;
output: mov     e,a
        mvi     c,direct
        jmp     bentry
output1:lhld    retvec
        pchl
;
input:  mvi     e,dinput
        mvi     c,direct
        jmp     bentry
;
pollin: in      cdata   ; get data
        ani     dredy   ; data ready?
        jz      pollin  ; loop until ready
        ret             ; A=data
;
```

```
241: pollout:push    psw      ; save character
242: pollout1:
243:         in      cstat    ; CRT ready?
244:         ani     credy    ; test bit
245:         jz      pollout1
246:         pop     psw      ; get character
247:         out     cdata    ; output to CRT
248:         ret
249: ;
250: wait:   call    input
251:         ora     a
252:         jz      wait
253:         ret
254: ;
255: bentry: push    b
256:         push    d
257:         push    h
258:         call    bdos
259:         pop     h
260:         pop     d
261:         pop     b
262:         ret
263: ;
264: savall: xthl
265:         push    d
266:         push    b
267:         push    psw
268:         push    h
269:         ret
270: ;
271: retall: pop     h
272:         pop     psw
273:         pop     b
274:         pop     d
275:         xthl
276:         ret
277: ;
278: adj:    push    h
279:         mvi     a,hdca
280:         call    process
281:         mov     a,l
282:         adi     offset
283:         call    output
284:         mov     a,h
285:         adi     offset
286:         call    output
287:         pop     h
288: adjmem: push    h
289:         mov     b,h
290:         mov     c,l
291:         lxi     h,memmap
292:         lxi     d,50h
293:         dcr     c
294:         jz      adjloop1
295: adjloop:dad     d
296:         dcr     c
297:         jnz     adjloop
298: adjloop1:
299:         mov     e,b
300:         mvi     d,0
```

```
301:            dad     d
302:            shld    scrpnt
303:            pop     h
304:            ret
305: ;
306: update:    push    h
307:            lhld    scrpnt
308:            mov     m,a
309:            inx     h
310:            shld    scrpnt
311:            pop     h
312:            ret
313: ;
314: rsmem:     push    h
315:            call    adjmem  ; 01/19/83
316:            lhld    scrpnt
317:            mov     a,m
318:            pop     h
319:            ret
320: ;
321: ssmem:     push    h
322:            call    adjmem  ; 01/19/83
323:            lhld    scrpnt
324:            mov     m,a
325:            pop     h
326:            ret
327: ;
328: graphix:
329:            mvi     a,hegm
330:            call    process
331:            mvi     a,1
332:            sta     gbit
333:            ret
334: ;
335: xgraphix:
336:            mvi     a,hxgm
337:            call    process
338:            mvi     a,0
339:            sta     gbit
340:            ret
341: ;
342: reverse:
343:            mvi     a,herv
344:            call    process
345:            mvi     a,1
346:            sta     rbit
347:            ret
348: ;
349: xreverse:
350:            mvi     a,hxrv
351:            call    process
352:            mvi     a,0
353:            sta     rbit
354:            ret
355: ;
356: cursoff:
357:            mvi     a,'x'
358:            call    process
359:            mvi     a,'5'
360:            jmp     output
```

```
;
cursor:
        mvi     a,'y'
        call    process
        mvi     a,'5'
        jmp     output
;
curs1:  mvi     a,hrm
        call    process
        mvi     a,'4'
        jmp     output
;
curs0:  mvi     a,hsm
        call    process
        mvi     a,'4'
        jmp     output
;
savecurs:
        mvi     a,hscp
        jmp     process
;
retcurs:
        mvi     a,hrcp
        jmp     process
;
boot:
restore:
        mvi     a,hram
        call    process
        lhld    oldstack
        sphl
        ret
;
show:   push    h
        lxi     h,show1
        shld    spec1+1
        pop     h
show1:  mov     a,m
        cpi     '@'
        jz      printat
        cpi     '['
        jz      special
        cpi     0
        rz
        inx     h
        call    update
        call    output
        jmp     show1
printat:
        inx     h
        mov     c,m
        inx     h
        mov     b,m
        inx     h
        call    saval1
        mov     l,c
        mov     h,b
        call    adj
        call    retall
        jmp     show1
```

```
special:
        inx     h
        mov     a,m
        cpi     'R'
        cz      reverse
        cpi     'r'
        cz      xreverse
        cpi     'G'
        cz      graphix
        cpi     'g'
        cz      xgraphix
        cpi     'S'
        cz      setmodes
        cpi     's'
        cz      rsetmodes
        cpi     'C'
        cz      cls
        cpi     'c'
        cz      ceol
        cpi     ']'
        jnz     special
        inx     h
spec1:  jmp     show1
setmodes:
        mvi     a,hsm
setentry:
        call    process
        inx     h
        mov     a,m
        jmp     output
rsetmodes:
        mvi     a,hrm
        jmp     setentry
;
; showout prints HL pointer on screen
; but does not update memmap
;
showout:push    h
        lxi     h,showout1
        shld    spec1+1
        pop     h
showout1:
        mov     a,m     ; get character
        ora     a       ; test for 0
        rz              ; return if so.
        cpi     '['     ; special routine?
        jz      special ;
        inx     h       ; next character
        push    h       ; save pointer
        call    output  ; output to CRT
        pop     h       ;
        jmp     showout1;
;
; delay delays for certain time
;
delay:  call    savall
        lhld    time
delay1: dcx     h
        mov     a,h
        ora     l
```

```
481:          jnz     delay1
482:          call    retall
483:          ret
484: ;
485: ; USER PROGRAM STARTS HERE
486: ;
487: ; contains programmable delay - HL=delay, SHLD TIME
488: ;
489: ;         CRT status bit
490: ;         0000.0000         = normal mode
491: ;         0000.0001         = graphics mode
492: ;         0000.0010         = reverse video mode
493: ;         0000.0100         = block cursor
494: ;         0000.1000         = cursor off
495: ;         0001.0000         = 25th line enabled
496: ;         0010.0000         = keypad shifted mode
497: ;         0100.0000         = alternate keypad mode
498: ;         1000.0000         = wrap around at end of line
499: ;
500: begin:   call    cursoff ; turn cursor off
501:          call    zsmem   ; set memmap to zeroes
502:          lxi     h,3030h
503:          shld    pscore
504:          shld    pscore+2
505:          shld    pscore1
506:          mvi     a,3
507:          sta     menleft
508:          mvi     a,1
509:          sta     dots    ; clear dots eaten
510:          xra     a
511:          sta     scrns   ; clear number of screens
512: begin5:  lxi     h,maze  ; start of maze
513:          call    show    ; print maze
514:          lxi     h,maze2 ;
515:          call    show    ;
516:          mvi     a,0fh   ;
517:          sta     enbit   ; energizers
518: begin2:  call    pacmen  ;
519:          lxi     h,2614h ; line 10, column 40
520:          shld    placo1  ; top of pacman
521:          call    adj     ; put cursor there
522:          lxi     h,paclft1
523:          call    show    ; top half left
524:          lxi     h,2615h ; line 11, column 40
525:          shld    placo2  ; bottom half of pacman
526:          call    adj     ; ready to print
527:          lxi     h,paclft2
528:          call    show    ; bottom half
529:          lxi     h,2606h ;
530:          shld    comco1  ;
531:          shld    comco3
532:          shld    comco5
533:          shld    comco7
534:          inr     l
535:          shld    comco2  ;
536:          shld    comco4
537:          shld    comco6
538:          shld    comco8
539:          lxi     h,0020h
540:          shld    ready1
```

```
541:           shld    ready2
542:           shld    ready3
543:           shld    ready4
544:           lxi     h,0
545:           shld    energy
546:           call    wait     ; wait for response
547:           call    begin3   ;
548: begin1:   call    ghost    ; get move from player 1
549:           call    delay
550: player:   lda     scrns    ; screens cleared
551:           cpi     0        ; 1st screen
552:           jz      player1
553:           cpi     1        ; 2nd screen
554:           jz      player1
555:           cpi     2        ; 3rd screen
556:           jz      player2
557: player3:  call    begin4   ; move pacman
558:           call    scrup    ; update screen
559:           call    delay
560:           call    ghost    ; move ghost
561:           jmp     begin1   ; move ghost again
562: player2:  call    begin4   ; move pacman
563:           call    scrup    ; update CRT
564:           call    delay
565:           jmp     begin1   ; move ghost
566: player1:  call    begin4   ; move pacman
567:           call    scrup    ;
568:           call    delay    ; delay
569:           call    begin4   ; move pacman again
570:           call    scrup
571:           call    delay    ; delay
572:           jmp     begin1   ; move ghost
573: begin4:   call    pldead6  ;
574:           call    input    ;
575:           ora     a        ; see if key pressed
576:           jz      nomove0  ; no key pressed, then skip
577: begin3:   sta     oldin1   ; else, store key pressed
578:           cpi     '8'      ; up?
579:           jz      pacup    ; move up
580:           cpi     '6'      ; right?
581:           jz      pacright ; move right
582:           cpi     '4'      ; left?
583:           jz      pacleft  ; move left
584:           cpi     '2'      ; down?
585:           jz      pacdown  ; move down
586:           cpi     'p'      ; pause
587:           jz      pause    ;
588:           cpi     'q'      ; quit?
589:           jz      boot     ; reboot system
590:           cpi     esc      ;
591:           jz      gameover ; start over
592:           jmp     nomove0  ; illegal response, move ghost
593: pause:    call    wait     ;
594:           jmp     begin3   ;
595: tune11:   mvi     a,2      ;
596:           sta     tune11+1 ;
597:           mvi     a,0d8h   ;
598:           sta     tune12   ; RC op-code
599:           mvi     a,4ah    ; column 74
600:           sta     tune13+1 ;
```

```
1141:           shld    ready2
1142:           shld    ready3
1143:           shld    ready4
1144:           lxi     h,0
1145:           shld    energy
 146:           call    wait    ; wait for response
1147:           call    begin3  ;
1148: begin1:   call    ghost   ; get move from player 1
1149:           call    delay
1150: player:   lda     scrns   ; screens cleared
1151:           cpi     0       ; 1st screen
1152:           jz      player1
1153:           cpi     1       ; 2nd screen
1154:           jz      player1
1155:           cpi     2       ; 3rd screen
1156:           jz      player2
1157: player3:  call    tunel4+1;
1158:           lxi     h,pacleft
1159:           shld    tunel5+1;
1160:           lhld    placo1  ;
1161: tunel0:   mvi     a,0ah   ; line 10
1162:           cmp     l       ; test y
1163:           rnz             ; not on right line
1164: tunel1:   mvi     a,2     ; column 2
1165:           cmp     h       ; test x
1166: tunel2:   rc              ; not right column
1167:           call    adj     ;
1168:           lxi     h,pacera;
1169:           call    show    ;
1170:           lhld    placo2  ;
1171:           call    adj     ;
1172:           lxi     h,pacera;
1173:           call    show    ; erase pacman
1174:           lhld    placo1  ; get top half
1175: tunel3:   mvi     h,74    ; set x to 76
1176:           shld    placo1  ;
1177:           lhld    placo2  ;
1178: tunel4:   mvi     h,74    ;
1179:           shld    placo2  ; set bottom x to 76
1180:           pop     h       ;
1181: tunel5:   jmp     pacleft ;
1182: tunelr:   mvi     a,73    ; x test
1183:           sta     tunel1+1;
1184:           mvi     a,0d0h  ; RNC op code
1185:           sta     tunel2  ;
1186:           mvi     a,2     ;
1187:           sta     tunel3+1;
1188:           sta     tunel4+1;
1189:           lxi     h,pacright
1190:           shld    tunel5+1;
1191:           lhld    placo1  ;
1192:           jmp     tunel0  ;
1193: pacleft:lhld    placo1  ; get upper half
1194:           dcr     h       ; decrement x-position
1195:           call    rsmem   ; see if anything there
1196:           cpi     ' '     ; empty space?
 197:           jz      test1   ; check bottom half
1198:           cpi     '1'     ; dots?
1199:           jnz     pacexit1; not legal character, skip
1200:           shld    oldco2  ; save x,y
```

STA TUNEL4+1

```
1201:           lxi     h,paclef t1
1202:           shld    retvec  ; returning vector
1203:           jmp     score   ; increment score
1204: test1:    inr     l       ; test bottom half
1205:           call    rsmem   ; see what's there
 206:           cpi     ' '     ; empty space?
1207:           jnz     pacexit1; skip if not space.
1208:           dcr     l       ; set for upper half
1209:           shld    oldco2  ; save x,y
1210: pacleft1:
1211:           mvi     a,'4'   ;
1212:           sta     oldin   ;
1213:           call    tunel1  ; check tunnel
1214:           lda     open1   ;
1215:           ora     a       ;
1216:           jz      pacleft2;
1217:           mvi     a,Ø     ;
1218:           sta     open1   ;
1219:           lxi     h,paclft1
1220:           shld    entry1+1;
1221:           lxi     h,paclft2
1222:           shld    entry2+1;
1223:           call    pacmove
1224:           ret
1225: pacleft2:
1226:           mvi     a,1     ; open mouth
1227:           sta     open1   ;
1228:           lxi     h,paclft3
1229:           shld    entry1+1;
1230:           lxi     h,paclft4
1231:           shld    entry2+1;
1232:           call    pacmove
1233:           ret
1234: pacmove:lhld    oldco2  ; get x,y
1235:           shld    placo1  ;
1236:           lhld    placo1  ;
1237:           inr     l       ;
1238:           shld    placo2  ;
1239:           ret             ;
1240: pacexit1:
1241:           lda     oldin   ; pacman can not move
1242:           cpi     '4'
1243:           rz
1244:           jmp     nomove
1245: pacright:
1246:           lhld    placo1  ; get x,y
1247:           mov     a,h     ; move x to accumalator
1248:           adi     4       ; right side of pacman
1249:           mov     h,a     ; put accumalator back
1250:           call    rsmem   ; read memmap
1251:           cpi     ' '     ; space?
1252:           jz      test3   ; test lower half
1253:           cpi     '1'     ; dots?
1254:           jnz     pacexit2; skip if not
1255:           shld    oldco2  ; save x,y
1256:           lxi     h,pacright1
 257:           shld    retvec  ; return to pacright1
1258:           jmp     score   ; update score
1259: test3:    inr     l       ; increment y
1260:           call    rsmem   ; read memmap
```

```
1261:           cpi     ' '     ; space?
1262:           jnz     pacexit2; skip if not space
1263:           dcr     l       ; decrement y
1264:           shld    oldco2  ; save x,y
1265: pacright1:
 266:           mvi     a,'6'   ;
1267:           sta     oldin   ;
1268:           call    tunelr  ; check tunnel
1269:           lda     open2   ;
1270:           ora     a       ;
1271:           jz      pacright2
1272:           mvi     a,0     ;
1273:           sta     open2   ;
1274:           lxi     h,pacrgt1
1275:           shld    entry1+1;
1276:           lxi     h,pacrgt2
1277:           shld    entry2+1;
1278:           call    pacmove1
1279:           ret
1280: pacright2:
1281:           mvi     a,1     ;
1282:           sta     open2   ;
1283:           lxi     h,pacrgt3
1284:           shld    entry1+1;
1285:           lxi     h,pacrgt4
1286:           shld    entry2+1;
1287:           call    pacmove1
1288:           ret
1289: pacmove1:
1290:           lhld    oldco2  ;
'291:           mov     a,h
.292:           sui     4
1293:           mov     h,a
1294:           shld    placo1  ;
1295:           call    adj     ; position cursor
1296:           mvi     a,' '   ; empty space
1297:           call    update  ; update memory map
1298:           call    output  ; erase character
1299:           lhld    placo1  ; get bottom coordinates
1300:           inr     l       ;
1301:           call    adj     ; put cursor there
1302:           mvi     a,' '   ; empty space for bottom
1303:           call    update  ; update memory map
1304:           call    output  ; erase character
1305:           lhld    placo1  ; get original position
1306:           inr     h       ; and adjust new position
1307:           shld    placo1
1308:           lhld    placo1
1309:           inr     l
1310:           shld    placo2
1311:           ret
1312: pacexit2:
1313:           lda     oldin
1314:           cpi     '6'
1315:           rz
1316:           jmp     nomove
 317: pacdown:lhld     placo1
1318:           mov     a,l
1319:           adi     2
1320:           mov     l,a
```

```
1321:           call    rsmem
1322:           cpi     32
1323:           jnz     pacexit3
1324:           mov     a,h
1325:           adi     3
 326:           mov     h,a
1327:           call    rsmem
1328:           cpi     ' '
1329:           jnz     pacexit3
1330:           dcr     h
1331:           shld    oldco2
1332:           call    rsmem
1333:           cpi     'm'       ; energizer?
1334:           cz      testen1   ;
1335:           lhld    oldco2
1336:           dcr     h         ; see if dot
1337:           shld    oldco2
1338:           call    rsmem     ;read memory map
1339:           cpi     ' '       ; empty space
1340:           jz      test6
1341:           cpi     'l'       ; dot?
1342:           jz      test5     ; add 10 points
1343:           jmp     pacexit3; anything else
1344: test5:    shld    oldco2    ;
1345:           lxi     h,test6 ; return to test6
1346:           shld    retvec    ; store it
1347:           jmp     score     ; add 10
1348: test6:    lhld    oldco2    ; get x,y
1349:           dcr     h         ; decrement count
1350:           shld    placo1    ;
1351: pacdown1:
1352:           mvi     a,'2'
1353:           sta     oldin
1354:           lda     open3     ;
1355:           ora     a         ;
1356:           jz      pacdown2;
1357:           mvi     a,0       ;
1358:           sta     open3     ;
1359:           lxi     h,pacdwn1
1360:           shld    entry1+1;
1361:           lxi     h,pacdwn2
1362:           shld    entry2+1;
1363:           call    pacmove2
1364:           ret
1365: pacdown2:
1366:           mvi     a,1       ;
1367:           sta     open3     ;
1368:           lxi     h,pacdwn3
1369:           shld    entry1+1;
1370:           lxi     h,pacdwn4
1371:           shld    entry2+1;
1372:           call    pacmove2
1373:           ret
1374: pacmove2:
1375:           lhld    placo1
1376:           dcr     l
 377:           shld    placo1
1378:           inr     l
1379:           shld    placo2
1380:           ret
```

```
pacexit3:
          lda     oldin
          cpi     '2'
          rz
          jmp     nomove
;
pacup:    lhld    placo1
          dcr     l
          call    rsmem
          cpi     32
          jnz     pacexit4
          mov     a,h
          adi     3
          mov     h,a
          call    rsmem
          cpi     ' '
          jnz     pacexit4
          dcr     h
          shld    oldco2
          call    rsmem
          cpi     'm'       ; energizer?
          cz      testen1
          lhld    oldco2
          dcr     h
          shld    oldco2
          call    rsmem
          cpi     ' '
          jz      test10
          cpi     '1'
          jz      test9
          jmp     pacexit4
test9:    lxi     h,test10
          shld    retvec
          jmp     score
test10:   lhld    oldco2
          dcr     h
          shld    placo1
pacup10:
          mvi     a,'8'
          sta     oldin
          lda     open4     ;
          ora     a         ;
          jz      pacup11   ;
          mvi     a,0       ;
          sta     open4     ;
          lxi     h,pacup1
          shld    entry1+1  ;
          lxi     h,pacup2
          shld    entry2+1  ;
          call    pacmove3
          ret
pacup11:
          mvi     a,1
          sta     open4
          lxi     h,pacup3
          shld    entry1+1
          lxi     h,pacup4
          shld    entry2+1
          call    pacmove3
          ret
```

```
1441: pacmove3:
1442:          lhld    placo1
1443:          inr     l
1444:          shld    placo2
1445:          ret
 446: pacexit4:
1447:          call    pacup11
1448:          lda     oldin
1449:          cpi     '8'
1450:          rz
1451:          jmp     nomove
1452: ;
1453: testen1:push    b
1454:          push    h
1455:          lhld    energy
1456:          lxi     b,035h
1457:          dad     b
1458:          shld    energy
1459:          pop     h
1460:          mov     a,h
1461:          cpi     6         ; test for left side
1462:          jnc     testen2   ; right side
1463:          mov     a,l       ; test top or bottom
1464:          cpi     6         ;
1465:          jnc     testen3   ; bottom
1466:          mvi     b,1       ; top/left
1467: testen5:lda     enbit     ;
1468:          xra     b         ; turn bit off
1469:          sta     enbit     ;
1470:          call    score50   ; 50 points
1471:          pop     b
1472:          ret               ;
1473: testen3:mvi     b,4       ; bottom/left
1474:          jmp     testen5   ;
1475: testen2:mov     a,l       ; test top or bottom
1476:          cpi     6         ;
1477:          jnc     testen4   ; bottom
1478:          mvi     b,2       ; top/right
1479:          jmp     testen5   ;
1480: testen4:mvi     b,8       ; bottom/right
1481:          jmp     testen5   ;
1482: ;
1483: ; screen update routine
1484: ;
1485: scrup:  lda     oldin     ; get last move
1486:          cpi     '2'       ;
1487:          jz      entry3    ;
1488:          cpi     '8'
1489:          jz      entry4
1490: scrup1: lhld    placo1    ;
1491:          call    adj       ;
1492: entry1: lxi     h,paclft1 ; changes according to direction
1493:          call    show
1494:          lhld    placo2
1495:          call    adj
1496: entry2: lxi     h,paclft2
 497:          call    show
1498:          jmp     scrup2    ; ghost
1499: entry3: lhld    placo1
1500:          dcr     l
```

```
1501: entry5: call    adj
1502:         lxi     h,pacera
1503:         call    show
1504:         jmp     scrup1
1505: entry4: lhld    placo2
 506:         inr     l
1507:         jmp     entry5
1508: scrup2: ret
1509: ghost:  call    blink
1510:         call    ghostm1 ; move ghost #1
1511:         call    fruit   ; display fruit
1512:         ret
1513: ; GHOST #1
1514: ghostm1:lhld    ready1  ; test ghost #1
1515:         mov     a,h     ;
1516:         ora     l       ; test for zero
1517:         jz      ghone   ; move ghost #1
1518:         dcx     h       ; decrement timer
1519:         shld    ready1  ; store
1520:         ret
1521: ghone:  lxi     h,comco1; top x,y of ghost
1522:         shld    load1+1
1523:         lxi     h,comco2; bottom x,y of ghost
1524:         shld    load2+1
1525:         lhld    comco1  ; ghost x,y
1526:         shld    oldco1
1527:         lda     dir1    ; direction of 1st ghost
1528:         sta     bit2
1529:         xra     a       ; clear accumalator
1530:         sta     bit1
1531:         call    ghost6  ; check up or down first
1532:         lda     bit2    ; get new direction
1533:         sta     dir1    ; save it
1534:         xra     a
1535:         sta     bit1
1536: ; GHOST #2
1537: ghostm2:lhld    ready2  ; see if timer is zero
1538:         mov     a,h     ;
1539:         ora     l
1540:         jz      ghtwo   ; move second ghost
1541:         dcx     h
1542:         shld    ready2
1543:         ret             ; update timer, return
1544: ghtwo:  lxi     h,comco3
1545:         shld    load1+1
1546:         lxi     h,comco4
1547:         shld    load2+1
1548:         lhld    comco3
1549:         shld    oldco1
1550:         lda     dir2
1551:         sta     bit2
1552:         call    ghost2  ; move left/right
1553:         lda     bit2
1554:         sta     dir2
1555:         xra     a
1556:         sta     bit1
 557: ; GHOST #3
1558: ghostm3:lhld    ready3  ; get timer for 3rd ghost
1559:         mov     a,h
1560:         ora     l
```

```
1561:           jz      ghthree ; move 3rd ghost
1562:           dcx     h
1563:           shld    ready3
1564:           ret
1565: ghthree:lxi       h,comco5
1566:           shld    load1+1
1567:           lxi     h,comco6
1568:           shld    load2+1
1569:           lhld    comco5
1570:           shld    oldco1
1571:           lda     dir3
1572:           sta     bit2;
1573:           lhld    comco5  ; t3
1574:           call    adj     ;
1575:           call    ghoerase;
1576:           lhld    comco6  ; b3
1577:           call    adj     ;
1578:           call    ghoerase;
1579:           lhld    comco7  ; t4
1580:           call    adj     ;
1581:           call    ghoerase;
1582:           lhld    comco8  ; b4
1583:           call    adj     ;
1584:           call    ghoerase;
1585:           lhld    placo1  ;
1586:           call    adj     ;
1587:           lxi     h,pdead1;
1588:           call    show    ;
1589:           lhld    placo2  ;
1590:           call    adj     ;
1591:           lxi     h,pdead2;
1592:           call    show    ;
1593:           lxi     h,3f00h ;
1594: pldead1:dcx     h       ;
1595:           mov     a,h     ;
1596:           ora     l       ;
1597:           jnz     pldead1 ;
1598:           lhld    placo1  ;
1599:           call    adj     ;
1600:           lxi     h,pdead3;
1601:           call    show    ;
1602:           lxi     h,3f00h ;
1603: pldead2:dcx     h       ;
1604:           mov     a,h     ;
1605:           ora     l       ;
1606:           jnz     pldead2 ;
1607:           lhld    placo1  ;
1608:           call    adj     ;
1609:           lxi     h,pacera;
1610:           call    show    ;
1611:           lxi     h,3f00h ;
1612: pldead3:dcx     h       ;
1613:           mov     a,h     ;
1614:           ora     l
1615:           jnz     pldead3
1616:           lhld    placo2  ;
 617:           call    adj
1618:           lxi     h,pdead4;
1619:           call    show
1620:           lxi     h,3f00h ;
```

```
1621: pldead4:dcx     h
1622:         mov     a,h
1623:         ora     l
1624:         jnz     pldead4
1625:         lhld    placo1
1626:         call    adj     ;
1627:         lxi     h,pdead5
1628:         call    show    ;
1629:         lhld    placo2
1630:         call    adj
1631:         lxi     h,pdead6
1632:         call    show    ;
1633:         lxi     h,3f00h
1634: pldead5:dcx     h
1635:         mov     a,h
1636:         ora     l
1637:         jnz     pldead5
1638:         lhld    placo1
1639:         call    adj
1640:         lxi     h,pacera
1641:         call    show
1642:         lhld    placo2
1643:         call    adj
1644:         lxi     h,pacera
1645:         call    show
1646:         pop     h       ; return address
1647:         lda     menleft
1648:         dcr     a
1649:         sta     menleft
1650:         jz      gameover;
1651:         pop     h       ; get return address
1652:         jmp     begin2  ;
1653: ;
1654: pldead6:lhld    oldco1
1655:         xchg
1656:         lhld    placo1
1657:         mov     a,d
1658:         cmp     h
1659:         jnz     pldead8
1660:         mov     a,e
1661:         cmp     l
1662:         jz      pldead7
1663: pldead8:lhld    placo1
1664:         dcr     h
1665:         mov     a,d
1666:         cmp     h
1667:         jnz     pldead9
1668:         mov     a,e
1669:         cmp     l
1670:         jz      pldead7
1671: pldead9:lhld    placo1
1672:         inr     h
1673:         mov     a,d
1674:         cmp     h
1675:         jnz     pldead10
1676:         mov     a,e
 677:         cmp     l
1678:         jz      pldead7
1679: pldead10:
1680:         lhld    placo1
```

```
1681:           dcr     l
1682:           mov     a,e
1683:           cmp     l
1684:           jnz     pldead11
1685:           mov     a,d
1686:           cmp     h
1687:           jz      pldead7
1688: pldead11:
1689:           lhld    placo1
1690:           inr     l
1691:           mov     a,e
1692:           cmp     l
1693:           rnz
1694:           mov     a,d
1695:           cmp     h
1696:           rnz
1697: pldead7:lhld    energy
1698:           mov     a,h
1699:           ora     l
1700:           jz      pldead
1701:           lhld    oldco1
1702:           call    adj
1703:           call    ghoerase
1704:           lhld    oldco2
1705:           call    adj
1706:           call    ghoerase
1707:           lxi     h,2606h
1708:           shld    oldco1
1709: ;
1710: score200:
1711:           lxi     h,pscore
1712:           inx     h
1713:           inx     h
1714:           inx     h
1715:           mvi     a,2
1716:           add     m
1717:           mov     m,a
1718:           mvi     a,'9'
1719:           cmp     m
1720:           jnc     score4
1721:           mov     a,m
1722:           sui     0ah
1723:           mov     m,a
1724:           dcx     h
1725:           jmp     score3
1726: incnext1:
1727:           mvi     m,'0'
1728:           dcx     h
1729: score3:  mvi     a,'9'
1730:           cmp     m
1731:           jz      incnext1
1732:           inr     m
1733: score4:  lxi     h,819h
1734:           call    adj
1735:           lxi     h,pscore
1736:           call    show
 737:           ret
1738: ;
1739: nmaze:   call    scrup   ; update last move
1740:           call    nmaze1
```

```
1741:           call    nmaze2
1742:           mvi     a,1
1743:           sta     dots
1744:           lda     scrns   ; get # of screens cleared
1745:           inr     a       ; increment
1746:           sta     scrns   ;
1747:           pop     h       ; return address
1748:           jmp     begin5  ;
1749: nmaze1:   call    reverse
1750:           lxi     h,maze1
1751:           call    show
1752:           ret
1753: nmaze2:   call    xreverse
1754:           lxi     h,maze1
1755:           call    show
1756:           ret
1757: ; game over
1758: gameover:
1759:           pop     h       ; get return address of stack
1760:           lxi     h,gameov1
1761:           call    show
1762:           call    xgraphix
1763:           lxi     h,4f19h
1764:           call    adj
1765:           lxi     h,gameov
1766: gameover1:
1767:           push    h       ; save pointer
1768:           mov     a,m
1769:           cpi     0
1770:           jz      gameover
1771:           call    output
1772:           lxi     h,2500h
1773: gameover4:
1774:           dcx     h
1775:           mov     a,h
1776:           ora     l
1777:           jnz     gameover4
1778:           lxi     h,0119h
1779:           call    adj
1780:           call    deletec
1781:           lxi     h,4f19h
1782:           call    adj
1783:           call    input
1784:           ora     a
1785:           pop     h       ; get pointer back
1786:           jnz     gameover2
1787: gameover3:
1788:           inx     h
1789:           jmp     gameover1
1790: gameover2:
1791:           cpi     'y'
1792:           jz      begin
1793:           cpi     'Y'
1794:           jz      begin
1795:           cpi     'n'
1796:           jz      restore
 797:           cpi     'N'
1798:           jz      restore
1799:           jmp     gameover3
1800: ;
```

```
1801: ; GAME STORAGE
1802: ;
1803: scrns:   db      0       ; number of screens cleared
1804: ;
1805: maze:    db      '[CG]'
1806: maze1:   db      '@',1,1
1807:          db      'faaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaas'
1808:          db      'saaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaac',cr,lf
1809:          db      '` 1  1  1  1  1  1  1  1  1  1  1  1  `'
1810:          db      '` 1  1  1  1  1  1  1  1  1  1  1  1  `',cr,lf
1811:          db      '`                                     `'
1812:          db      '`                                     `',cr,lf
1813:          db      '` 1m faaaaaac 1  faaaaaaaaaaaaaac 1  `'
1814:          db      '` 1  faaaaaaaaaaaaaac 1  faaaaaac 1m `',cr,lf
1815:          db      '`    eaaaaaad    eaaaaaaaaaaaaaad    e'
1816:          db      'd    eaaaaaaaaaaaaaad    eaaaaaad    `',cr,lf
1817:          db      '` 1  1  1  1  1  1  1  1  1  1  1  1  1'
1818:          db      '  1  1  1  1  1  1  1  1  1  1  1  1  `',cr,lf
1819:          db      '`                                     '
1820:          db      '                                      `',cr,lf
1821:          db      'vaaaaaaaaaaac 1  faaaaaaaaasaaaaaaaac{{'
1822:          db      '{{faaaaaaaasaaaaaaaac 1  faaaaaaaaaat',cr,lf
1823:          db      'eaaaaaaaaaad    eaaaaaaac`            '
1824:          db      '            `faaaaaaad    eaaaaaaaaaad',cr,lf
1825:          db      '     1  1  1  1  1  1  1  ``           '
1826:          db      '           `` 1  1  1  1  1  1  1      ',cr,lf
1827:          db      '                                `vaaaaaaaaaa'
1828:          db      'aaaaaaaaaat`                          ',cr,lf
1829:          db      'faaaaaaaaaac 1  faaac 1  `` 1  1  1  1'
1830:          db      '  1  1  1  `` 1  faaac 1  faaaaaaaaaac',cr,lf
1831:          db      'vaaaaaaaaad    eaaad    ed            '
1832:          db      '           ed    eaaad    eaaaaaaaaaat',cr,lf
1833:          db      '` 1  1  1  1  1  1  1  1  1  1  faaaaa'
1834:          db      'aaaaac 1  1  1  1  1  1  1  1  1  1  `',cr,lf
1835:          db      '`                              eaaaac'
1836:          db      'faaaad                              `',cr,lf
1837:          db      '` 1  faaaaac 1  faaac 1  fc 1  1  1  `'
1838:          db      '` 1  1  1  fc 1  faaac 1  faaaaac 1  `',cr,lf
1839:          db      '`    `     `    `   `    ``          `'
1840:          db      '`          ``   `   `    `     `    `',cr,lf
1841:          db      '` 1m `      ` 1  `   ` 1  `eaaaaac 1  `'
1842:          db      '` 1  faaaad` 1  `   ` 1  `       ` 1m `',cr,lf
1843:          db      '`    `faaaad    eaac`    `faaaad    e'
1844:          db      'd    eaaaaac`    `faad    eaaaaac`    `',cr,lf
1845:          db      '` 1  `` 1  1  1  1  `` 1  `` 1  1  1  1'
1846:          db      '  1  1  1  `` 1  `` 1  1  1  1  `` 1  `',cr,lf
1847:          db      '`    ed             ed    ed           '
1848:          db      '           ed    ed             ed    `',cr,lf
1849:          db      '` 1  1  1  faaac 1  1  1  1  1  faaaaa'
1850:          db      'aaaaaac 1  1  1  1  1  faaac 1  1  1  `',cr,lf
1851:          db      '`          `   `                `     '
1852:          db      '     `                   `   `       `',cr,lf
1853:          db      'eaaaaaaaaaauaaauaaaaaaaaaaaaaaaauaaaaaa'
1854:          db      'aaaaaauaaaaaaaaaaaaaaauaaauaaaaaaaaaad',0
1855: maze2:   db      '[s1S1g]@',25,1
1856:          db      'Score: 000000@',25,30,'High Score: 000000[G]',0
 857: energymes:
1858:          db      '@',10,31,'[R]E N E R G I Z E D[r]',0
1859: energymes1:
1860:          db      '@',10,30,'[R]iEiNiEiRiGiIiZiEiDi[r]',0
```

```
1861: energyera:
1862:          db      '@',10,30,'                          ',0
1863: gameov:  db      '(C)opyright 1983 CompTec Software. . .Press (Y)'
1864:          db      ' to play or (N) to reboot. . .',0
1865: gameov1:db       '@',13,32,'[R]G A M E  O V E R[r]',0
1866: ;
1867: ; Pacman character definitions
1868: ;
1869: erase:  db       '       ',0
1870: enbit:  db       0fh               ; all energizers on
1871: bit1:   db       0
1872: bit2:   db       0
1873: paclft1:db       '_[R]l_[r]  ',0
1874: paclft2:db       '[R]r [r]r  ',0
1875: paclft3:db       '[R]rl_[r]  ',0
1876: paclft4:db       '_[R] [r]r  ',0
1877: pacrgt1:db       ' [R]rm[r]r',0
1878: pacrgt2:db       ' _[R] _[r]',0
1879: pacrgt3:db       ' [R]rm_[r]',0
1880: pacrgt4:db       ' _[R] [r]r',0
1881: pacup1: db       '[R]r_r_[r]',0
1882: pacup2: db       '_[R]n [r]r',0
1883: pacup3: db       '[R]r  _[r]',0
1884: pacup4: db       '_[R]n [r]r',0
1885: pacdwn1:db       '[R]rm _[r]',0
1886: pacdwn2:db       '_r_r',0
1887: pacdwn3:db       '[R]rm _[r]',0
1888: pacdwn4:db       '_[R]  [r]r',0
1889: pacera: db       '     ',0
1890: ghostl1:db       '[R]rmm_[r]',0
1891: ghostl2:db       'r_r_',0
1892: ghostl3:db       '[R]rmm_[r]',0
1893: ghostl4:db       '_r_r',0
1894: ghostr1:db       '[R]rll_[r]',0
1895: ghostr2:db       'r_r_',0
1896: ghostr3:db       '[R]rll_[r]',0
1897: ghostr4:db       '_r_r',0
1898: ghostu1:db       '[R]rno_[r]',0
1899: ghostu2:db       'r_r_',0
1900: ghostu3:db       '[R]rno_[r]',0
1901: ghostu4:db       '_r_r',0
1902: ghostd1:db       '[R]rml_[r]',0
1903: ghostd2:db       'r_r_',0
1904: ghostd3:db       '[R]rml_[r]',0
1905: ghostd4:db       '_r_r',0
1906: ghoera: db       '     ',0
1907: pdead1: db       '[R]rm _[r]',0
1908: pdead2: db       '_[R]  [r]r',0
1909: pdead3: db       '[R]r_r_[r]',0
1910: pdead4: db       ' [R]r_[r] ',0
1911: pdead5: db       '{yx{',0
1912: pdead6: db       'zxyz',0
1913: dot:    db       ' 1  ',0
1914: dot1:   db       '1',0
1915: energy: ds       2
1916: ;
 917: retvec: ds       2
1918: curco:  ds       2
1919: gbit:   ds       1
1920: rbit:   ds       1
```

```
1921: dir1:    db      0
1922: dir2:    db      0
1923: dir3:    db      0
1924: dir4:    db      0
1925: ready1: ds       2
 926: ready2: ds       2
1927: ready3: ds       2
1928: ready4: ds       2
1929: menleft:db       3
1930: oldin:   db       '4'
1931: oldin1: db       0
1932: placo1: ds       2
1933: placo2: ds       2
1934: comco1: ds       2
1935: comco2: ds       2
1936: comco3: ds       2
1937: comco4: ds       2
1938: comco5: ds       2
1939: comco6: ds       2
1940: comco7: ds       2
1941: comco8: ds       2
1942: oldco1: ds       2
1943: oldco2: ds       2
1944: pscore: db       '0000'
1945: pscore1:db       '00',0
1946: dots:    db       1
1947: open1:   db       0
1948: open2:   db       0
1949: open3:   db       0
1950: open4:   db       0
*951: time:    db       01,00
.952: crtbit: db       0
1953: oldstack:
1954:          ds       64h
1955: stack:
1956: scrpnt: ds       2
1957: memmap: ds       1
```