```
********************************************************************************
********************************************************************************
****                                                                        ****
****     PROGRAM LISTING FOR OPERATING SYSTEM OF THE BLITZ-80               ****
****     MICROCOMPUTER.                                                     ****
****                                                                        ****
****     WRITTEN BY:   JEFF DAVIDSON                                        ****
****                   1526 N. GENESEE AVENUE                               ****
****                   LOS ANGELES, CA 90046                                ****
****                   (213) 876-3209 VOICE                                 ****
****                   (213) 876-3290 FAX                                   ****
****                   72450,1155 CompuServe                                ****
****                                                                        ****
****     AUGUST 24, 1986.  ALL RIGHTS RESERVED.                             ****
****     VERSION 4.1 - UPDATED MARCH 25, 1994                               ****
****                                                                        ****
********************************************************************************
********************************************************************************
```

TABLE OF CONTENTS

```
*******************************************************************************
*******************************************************************************
****                                                                       ****
****     PROGRAM LISTING FOR OPERATING SYSTEM OF THE BLITZ-80              ****
****     MICROCOMPUTER.                                                    ****
****                                                                       ****
****     WRITTEN BY:   JEFF DAVIDSON                                       ****
****                   6525 OLD RIVERSIDE DRIVE                            ****
****                   ATLANTA, GEORGIA   30328                            ****
****                   (404) 255-7113                                      ****
****                                                                       ****
****     AUGUST 24, 1986.  ALL RIGHTS RESERVED.                            ****
****     VERSION 4.0 - UPDATED MARCH 25, 1994                              ****
****                                                                       ****
*******************************************************************************
*******************************************************************************

*******************************************************************************
*                                                                             *
*    PROGRAM MONITOR  (STAND-ALONE) VERSION 4.0                               *
*                                                                             *
*                                                                             *
*                  1000-1100 EQU  STACK   ;SYSTEM STACK                       *
*                  1101   DFW  ADDR1      ;ADDRESS 1                          *
*                  1103   DFW  ADDR2      ;ADDRESS 2                          *
*                  1105   DFB  DATA       ;DATA VARIABLE                      *
*                  1106   DFB  DOTFLAG    ;FLAG INDICATING . IN LINE          *
*                  1107   DFB  VP         ;VERTICAL POINTER                   *
*                  1108   DFB  HP         ;HORIZONTAL POINTER                 *
*                  1109   DFB  CURSOR     ;CURSOR CHARACTER                   *
*                  110A-113C   EQU SCROLLBUFFER ;BUFFER FOR SCROLL ROUTINE    *
*                  113D   DFB  LEDPTR     ;POSITION PTR FOR LED DISPLAY       *
*                  113E-1155   EQU LEDBUFFER   ;LED DISPLAY BUFFER            *
*                  17FF-1700   EQU  KEYBUFFER  ;KEYBOARD BUFFER               *
*                  113D        DFB         ;LED POSITION POINTER              *
*                  113E-1155   EQU  LEDBUFFER  ;LED DISPLAY BUFFER            *
*                  1156   DFB  MORECMDS   ;FLAG TO CONT MONITOR IN RAM        *
*                  1157   DFB  DISPTYPE   ;ROUTES OUTPUT OF COUT              *
*                  1158   DFB  BOOTFLAG   ;INDICATES COLD/WARM BOOT          *
*                  1159   DFB  SECCNT     ;SEC COUNT FOR MULT. DISK OPS       *
*                  115A   DFW  GENPNTR    ;GENERAL POINT FOR IX/IY OPS        *
*                                                                             *
*       SYSTEM STACK STARTS AT 16FFH AND GOES BACKWARDS FROM THERE            *
*                                                                             *
*******************************************************************************

0000:           MONITOR    ORG  0000H      ;START OF PROGRAM SPACE
0000:31 FF 16              LD   SP, 16FFH
0003:CD A1 01              CALL VDP_INIT
0006:CD 00 05              CALL LOGON4
0009:CD F1 08   NEWLN      CALL INITMON
000C:CD 28 0D              CALL GETLN3
000F:7E         LOOP       LD   A, (HL)    ;GET NEXT CHARACTER
0010:FE 47                 CP   "G"        ;GO COMMAND?
```

1810-1812 - Move Command Flag/JP ADDR

```
0012:C2 1C 00              JP   NZ, N1     ;NO, TRY NEXT TEST



*                "GO" MACRO
0015:DD 6E 00   GO         LD   L, (IX+0) ;GET ADDRn LOW
0018:DD 66 01              LD   H, (IX+1) ;GET ADDRn HIGH
001B:E9                    JP   (HL)      ;JUMP TO ADDRESS
001C:FE 3A      N1         CP   ":"       ;INPUT COMMAND?
001E:CA 5C 00              JP   Z, DATAIN ;IF SO, GOTO DATA IN MACRO
0021:FE 49                 CP   "I"       ;PRINT ADDR1 COMMAND?
0023:C2 35 00              JP   NZ, N2    ;NO, TRY NEXT TEST

*                   "PRINT ADDR1" MACRO
0026:DD 6E 00              LD   L, (IX+0) ;GET ADDRn LOW
0029:DD 66 01              LD   H, (IX+1) ;GET ADDRn HIGH
002C:CD 70 08              CALL HEXHL     ;PRINT HL
002F:CD 30 08              CALL CR        ;*CR*
0032:C3 09 00              JP   NEWLN     ;GET ANOTHER COMMAND LINE
0035:FE 2E      N2         CP   "."       ;ENTER ADDR2 COMMAND?
0037:C2 47 00              JP   NZ, N3    ;NO, TRY NEXT TEST

*                "SWITCH ADDRn" MACRO
003A:3E FF                 LD   A, 0FFH   ;VALUE TO TOGGLE FLAG
003C:32 06 11              LD   (1106), A ;STORE FLAG
003F:DD 21 03 11           LD   IX, 1103H ;SET POINTER TO ADDR2
0043:23         N4         INC  HL        ;NEXT CHARACTER IN LINE
0044:C3 0F 00              JP   LOOP
0047:FE 0D      N3         CP   0DH       ;*CR*?
0049:CA 94 00              JP   Z, EOC    ;OF SO, GOTO EOC MACRO
004C:CD 0C 09              CALL HEX?      ;IS THIS A HEX DIGIT?
004F:DA C9 05              JP   C, MCMDS  ;NO, CHECK FOR MORE COMMANDS
0052:CD 25 09              CALL ASCHEX    ;YES, CONVERT TO HEX
0055:CD 32 09              CALL SHIFT16   ;SHIFT INTO ADDRn
0058:23                    INC  HL        ;NEXT CHAR
0059:C3 0F 00              JP   LOOP

*                "DATA INPUT" MACRO
005C:23         DATAIN     INC  HL        ;LOOK AT NEXT CHAR
005D:7E         IGNORE     LD   A, (HL)   ;THIS IS HALF OF NEW DATA
005E:FE 0D                 CP   ODH       ;END OF DATA?
0060:CA 77 00              JP   Z, ENTER  ;YES, STORE LAST VALUE
0063:FE 20                 CP   " "       ;DELIMITER?
0065:CA 77 00              JP   Z, ENTER  ;YES, STORE THIS VALUE
0068:CD 0C 09              CALL HEX?      ;IS IT A HEX DIGIT?
006B:DA 5D 00              JP   C, IGNORE ;ILLEGAL CHAR, IGNORE IT
006E:CD 25 09              CALL ASCHEX    ;CONVERT TO HEX DIGIT
0071:CD 25 09              CALL SHIFT8    ;SHIFT INTO DATA BYTE
0074:C3 5C 00              JP   DATAIN    ;GET NEXT CHAR
0077:E5         ENTER      PUSH HL        ;SAVE HL
0078:F5                    PUSH AF        ;SAVE AF
0079:DD 6E 00              LD   L, (IX+0) ;SET ADDRn LOW
007C:DD 66 01              LD   H, (IX+1) ;SET ADDRn HIGH
007F:3A 05 11              LD   A, (1105) ;GET DATA BYTE
```

```
0082:77                 LD   (HL), A    ;STORE IN ADDRn
0083:23                 INC  HL         ;ADVANCE ADDRn
0084:DD 75 00           LD   (IX+0), L  ;RESTORE ADDRn LOW
0087:DD 74 01           LD   (IX+1), H  ;RESTORE ADDRn HIGH
008A:F1                 POP  AF         ;RESTORE AF
008B:E1                 POP  HL         ;RESTORE HL
008C:FE 0D              CP   ODH        ;LAST BYTE ON LINE?
008E:CA 09 00           JP   Z, NEWLN   ;YES, GET NEXT CHAR
0091:C3 5C 00           JP   DATAIN     ;NO, GET NEXT CHAR

*             "END OF COMMAND LINE" MACRO
0094:3A 06 11  EOC      LD   A, (1106)  ;GET DOTFLAG
0097:FE FF              CP   FFH        ;IS IT SET?
0099:CA B1 00           JP   Z, RANGE   ;YES, PRINT RANGE
009C:DD 6E 00           LD   L, (IX+0)  ;NO, PRINT SINGLE VALUE
009F:DD 66 01           LD   H, (IX+1)  ;GET ADDRn LOW/HIGH
00A2:CD 70 08           CALL HEXHL      ;PRINT ADDR
00A5:06 3A              LD   B, ":"
00A7:CD 17 08           CALL COUT       ;PRINT ":"
00AA:7E                 LD   A, (HL)    ;GET VALUE OF (ADDR)
00AB:CD 49 08           CALL HEXBYTE    ;PRINT IT
00AE:CD 30 08           CALL CR         ;*CR*
00B1:C3 09 00           JP   NEWLN      ;BACK TO CURSOR

*             "RANGE" MACRO - PRINT RANGE OF VALUES
00B4:DD 21 03 11        LD   IX, 1103H  ;IX=ADDR2
00B8:DD 56 01           LD   D, (IX+1)  ;LOAD D WITH ADDR2 LOW
00BB:DD 5E 00           LD   E, (IX+0)  ;LOAD E WITH ADDR2 HIGH
00BE:DD 21 05 11        LD   IX, 1101h  ;IX=ADDR1
00C2:DD 66 01           LD   H, (IX+1)  ;LOAD H WITH ADDR1 LOW
00C5:DD 6E 00           LD   L, (IX+0)  ;LOAD L WITH ADDR1 HIGH
00C8:13                 INC  DE
00C9:E5                 PUSH HL         ;PRINT HEADER-SAVE HL
00CA:21 FF 00           LD   HL, MSG1   ;"HEADER" POINTER
00CD:CD A6 09           CALL MSGOUT     ;PRINT HEADER
00D0:E1                 POP  HL         ;RESTORE HL
00D1:0E 10     NEXTLN   LD   C, 08H     ;LINE LENGTH=8 BYTES
00D6:CD 70 08           CALL HEXHL      ;PRINT ADDR FOR NEW LINE
00D9:06 3A              LD   B, ":"
00DB:CD 17 08           CALL COUT       ;PRINT ":"
00DE:7E        M1       LD   A, (HL)    ;GET VALUE TO BE PRINTED
00DF:CD 49 08           CALL HEXBYTE    ;PRINT IT
00E2:06 20              LD   B, " "
00E4:CD 17 08           CALL COUT       ;PRINT A SPACE
00E7:23                 INC  HL         ;NEXT VALUE
00E8:7A                 LD   A, D       ;
00E9:BC                 CP   H          ;CHECK FOR END OF LINE
00EA:C2 EF 00           JP   NZ, OKAY   ;NOT END OF LINE...
00ED:7B                 LD   A, E
00EE:BD                 CP   L          ;CHECK FOR END OF BLOCK
00EF:CA 7C 01           JP   Z, RETURN  ;GOTO NEWLN WITH LF/*CR*
00F2:0D        OKAY     DEC  C          ;DROP COUNTER BY 1
00F3:C2 DB 00           JP   NZ, M1     ;IF NOT EOL, KEEP GOING
00F6:C3 D1 00           JP   NEXTLN     ;IF EOL, START NEW LINE
```

```
00F9:DB 28      VDPRESET   IN   A, (28H)   ;WAKE UP VIDEO CHIP
00FB:CD 00 08              CALL INIT_9600
00FE:C9                    RET

*               MESSAGE DATA BLOCK
00FF:41 44 44 52           DFB  "ADDR:00 01 02 03 04 05 06 07 08 09
0103:3A 30 30 20                      0A 0B 0C 0D 0E 0F"
0107:30 31 20 30
010B:32 20 30 33
010F:20 30 34 20
0113:30 35 20 30
0117:36 20 30 37
011B:20 30 38 20
011F:30 39 20 30
0123:41 20 30 42
0127:20 30 43 20
012B:30 44 20 30
012F:45 20 30 46
0133:0D                    DFB  *CR*
```

```
*************************************************************************
*                                                                       *
*                  SUBROUTINE VDPINIT                                   *
*                                                                       *
*                  INITS VDP CHIP.  SETS VDP REGS AND FILLS             *
*                  CHARACTER GENERATOR DATA TABLE                       *
*                                                                       *
*************************************************************************

0134:E5          VDPINIT   PUSH HL          ;SAVE REGS
0135:C5                    PUSH BC
0136:DB 19                 IN   A, (19H)   ;READ VDP STATUS, WRITE MODE
0138:06 08                 LD   B, 08H     ;LOOP COUNTER
013A:0E 80                 LD   C, 80H     ;REGISTER COUNTER-
013C:21 6C 01              LD   HL, VREGS  ; (0-7+MASK=80H)
013F:7E          Q1        LD   A, (HL)    ;GET VALUE FROM VREGS
0140:D3 19                 OUT  A, (19H)   ;SEND TO VDP
0142:79                    LD   A, C       ;GET REGISTER NUMBER
0143:D3 19                 OUT  A, (19H)   ;SEND TO VDP
0145:0C                    INC  C          ;NEXT REGISTER
0146:23                    INC  HL         ;NEXT VREG
0147:05                    DEC  B          ;DEC LOOP COUNTER
0148:78                    LD   A, B       ;CHECK FOR LOOP COUNT=0
0149:FE 00                 CP   A, 00H
014B:C2 3F 01              JP   NZ, Q1     ;IF MORE, LOOP TO Q1
014E:21 00 02              LD   HL, 0200H  ;HL=CHAR GEN POINTER
0151:3E 00                 LD   A, 00H     ;BASE ADDR LOW
0153:D3 19                 OUT  A, (19H)   ;SEND TO VDP
0155:3E 49                 LD   A, 49H     ;BASE ADDR HIGH
0157:D3 19                 OUT  A, (19H)   ;SEND TO VDP
0159:7E          Q2        LD   A, (HL)    ;GET VALUE FROM TABLE
015A:D3 18                 OUT  A, (18H)   ;SEND TO VDP
015C:23                    INC  HL         ;NEXT VALUE
015D:7D                    LD   A, L       ;CHECK FOR END OF DATA
015E:FE 00                 CP   00H        ;LAST DIGIT OF HL=0?
0160:C2 59 01              JP   NZ, Q2     ;MORE, KEEP GOING AT Q2
0163:7C                    LD   A, H       ;FIRST DIGIT OF HL=05?
0164:FE 05                 CP   05H
0166:C2 59 01              JP   NZ, Q2     ;NO, KEEP GOING
0169:C1                    POP  BC         ;YES, RESTORE REGS AND...
016A:E1                    POP  HL
016B:C9                    RET
016C:00          VREG0     DISABLE VDP EXTERNAL INT, M3=0
016D:D0          VREG1     SELECT 16k,ACTIVE DISPLAY,DISABLE INTERNAL INT
                           M1=1, M2=0 (SETS TEXT MODE),SRITE SIZE=0(8X8)
                           MAGNIFICATION=0 (NORMAL)
016E:00          VREG2     NAME TABLE BASE ADDR=0000H
                           SCREEN MEMORY IS 0000H - 03FFH
016F:10          VREG3     COLOR TABLE BASE ADDR=0400H
0170:01          VREG4     TEXT BASE ADDR=0800H (TEXT GEN 0800-0FFF)
0171:20          VREG5     SPRITE ATTRIBUE TABLE=1000H
0172:02          VREG6     SPRITE PATTERN GENERATOR=1000H
0173:F1          VREG7     WHITE TEXT, BLACK BACKGROUND
```

```
0174:00 00 00 00          UNUSED
0178:00 00 00 00          UNUSED

*              "RETURN"  MACRO  -  PRINTS  *CR*  AND  GETS  NEW  INPUT  LINE

017C:CD 30 08   RETURN    CALL CR          ;PRINT *CR*
017F:C3 09 00             JP   NEWLN       ;MONITOR REENTRY POINT

*****************************************************************************
*                                                                           *
*              SUBROUTINE CLEAR                                             *
*                                                                           *
*              CLEARS THE VIDEO SCREEN.  NO REGS AFFECTED                   *
*                                                                           *
*****************************************************************************

0182:E5         CLEAR     PUSH HL          ;SAVE REGS
0183:21 C0 03             LD   HL, 03C0H   ;SET-UP COUNTER
0186:3E 00                LD   A, 00H      ;SEND BASE ADDR OF SCREEN
0188:D3 19                OUT  (19H), A    ;  MEMORY TO VDP
018A:3E 40                LD   A, 40H
018C:D3 19                OUT  (19H), A
018E:3E 20      A2        LD   A, " "      ;A=SPACE CHAR (=20H)
0190:D3 18                OUT  (18H), A    ;OUTPUT A SPACE TO SCREEN MEM
0192:2B                   DEC  HL          ;DECREASE COUNTER
0193:7D                   LD   A, L        ;CHECK FOR END OF LOOP
0194:FE 00                CP   00H
0196:C2 8E 01             JP   NZ, A2      ;NOT END, KEEP GOING
0199:7C                   LD   A, H
019A:FE 00                CP   00H
019C:C2 8E 01             JP   NZ, A2      ;STILL NOT OVER, KEEP GOING
019F:E1                   POP  HL          ;OVER! RESTORE REGS
01A0:C9                   RET  ;RETURN


*****************************************************************************
*                                                                           *
*              SUBROUTINE VDP_INIT                                          *
*                                                                           *
*              SETS UP VP, HP, CURSOR.  INITS VDP AND SERIAL PORTS          *
*                                                                           *
*****************************************************************************

01A1:3E 00      VDP_INIT  LD   A, 00H
01A3:32 07 11             LD   (1107H), A ;CLEAR VERTICAL POINTER
01A6:32 08 11             LD   (1108H), A ;CLEAR HORIZ POINTER
01A9:3E 3F                LD   A, 3FH     ;CURSOR = ">"
01AB:32 09 11             LD   (1109H), A ;PUT IN CURSOR REG
01AE:C3 34 01             CALL VDPINIT    ;CALL MAIN VDP INIT ROUTINE
01B1:3E B0      INIT8255  LD   A, B0H     ;8255A MODE 1-PORT A SET
01B3:D3 13                OUT  (13H), A   ;SEND TO 8255a COMMAND PORT
01B5:3E 09                LD   A, 09H     ;SET INTERNAL INT ENABLE
01B7:D3 13                OUT  (13H), A   ;SEND TO COMMAND PORT
01B9:C9                   RET             ;RETURN
```

```
* 01BA - 01FF ARE UNUSED ROM BYTES
*
* 0200 - 04FF IS VDP CHARACTER GENERATING DATA

*************************************************************************
*                                                                       *
*               SUBROUTINE LOGON4                                       *
*                                                                       *
*               PRINTS LOGON MESSAGE                                    *
*                                                                       *
*************************************************************************

0500:C3 B0 05   LOGON4     JP    PCH5        ;PATCH 5 - CHECK FOR WARM BOOT
0503:7E         NEXTCHR    LD    A, (HL)     ;GET FIRST CHARACTER
0504:FE 00                 CP    00H         ;END OF MESSAGE?
0506:CA 11 05              JP    Z, LOGON    ;YEAH, WE'RE DONE
0509:47                    LD    B, A        ;PUT CHAR INTO B REG
050A:CD DD 0B              CALL  VOUT        ;OUTPUT IT TO VIDEO PORT
050D:23                    INC   HL          ;ADVANCE POINTER
050E:C3 03 05              JP    NEXTCHR
0511:CD E0 0E   LOGON      CALL  VARSET      ;SETS UP SYSTEM VARIABLES
0513:00 00                 NOP   x2          ;
0516:CD 7C 0D              CALL  CLRLED      ;CLEAR LED DISPLAY
0519:21 8C 0E              LD    HL, INITMSG ;LED INIT MSG POINTER
051C:CD 57 0E              CALL  LEDMSG      ;DISPLAY ON LEDs
051F:CD 9E 0D              CALL  BLINK       ;BLINK LEDs
0522:CD C6 0D              CALL  WAIT
0525:CD 9E 0D              CALL  BLINK       ;BLINK LEDs
0528:CD C6 0D              CALL  WAIT
052B:CD 9E 0D              CALL  BLINK       ;BLINK LEDs
052E:CD C6 0D              CALL  WAIT
0531:CD 7C 0D              CALL  CLRLED      ;CLEAR LEDs
0534:3E 55                 LD    A, 55H      ;TEST CHARACTER
0536:D3 22                 OUT   (22H), A    ;SEND TO DISK DRIVE
0538:3E 00                 LD    A, 00H      ;RESET ACCUM.
053A:DB 22                 IN    A, (22H)    ;GET REG FROM DISK (?)
053C:FE 55                 CP    55H         ;IS IT THERE?
053E:CA 48 05              JP    Z, DISK     ;YES, THERE IS A DISK!
0541:21 64 0E              LD    HL, NODSK   ;POINTS TO NO DISK MSG
0544:C3 AA 05              JP    PATCH1      ;FINISH UP ROUTINE
0547:00                    NOP
0548:21 77 0E   Disk       LD    HL, YESDSK  ;POINTS TO YES DISK MSG
054B:CD 57 0E              CALL  LEDMSG
054E:06 08                 LD    B, 08H      ;DELAY LOOP NUM.
0550:21 FF FF   WL1        LD    HL, FFFFH   ;DELAY ROUTINE VALUE
0553:CD B7 0E              CALL  WAITHL      ;WAIT FOR DISK TO WAKE UP
0556:05                    DEC   B
0557:C2 50 05              JP    NZ, WL1     ;LOOP
055A:00 00 00
055D:00 00 00
0560:00 00 00
0563:00 00 00
0566:00 00 00
0569:CD 09 0F              CALL  BOOT
```

```
056C:00 00
056E:00 00
0570:00 00
0572:00 00 00
0575:00 00 00
0578:00 00 00
057B:00 00 00
057E:3E 01              LD   A, 01H     ;VALUE FOR "WARMBOOT"
0580:32 58 11           LD   (1158H), A;STORE IN BOOTFLAG
0583:3A 10 18           LD   A, (1810)  ;GET MORECMD INFO FROM DISK AREA
0586:32 56 11           LD   (1156H), A;STORE IN MORECMD VARIABLE
0589:00 00 00
058C:00 00
058E:21 AD 08  COMPLETE: LD   HL, 08ADH ;POINT TO "LOGON COMPLETE" MSG
0591:7E        NC2:     LD   A, (HL)    ;GET CHAR
0592:FE 00              CP   00H        ;DONE?
0594:CA 9F 05           JP   Z, R2      ;YES, MOVE ON
0597:47                 LD   B, A       ;NO, SO OUTPUT CHAR
0598:CD DD 0B           CALL VOUT
059B:23                 INC  HL         ;INC POINTER
059C:C3 91 05           JP   NC2        ;LOOP
059F:3A 58 11           LD   A, (1158H);GET BOOTFLAG
05A2:FE 00              CP   00         ;IS IT A COLD BOOT?
05A4:CA 09 00           JP   Z, NEWLN   ;YES, SO GO TO MONITOR
05A7:C3 00 18           JP   BOOTCODE   ;NO, SO GOTO CODE FROM DISK

** PATCH 1
05AA:CD 57 0E  PCH1     CALL LEDMSG
05AD:C3 8E 0E           JP   COMPLETE

** PATCH 5
05B0:21 93 08  PCH5     LD   HL, 0893H  ;LOGON MESSAGE POINTER
05B3:3A 58 11           LD   A, (1158H);GET BOOTFLAG
05B6:FE 01              CP   01H        ;IS THIS A WARM BOOT?
05B8:C2 03 05           JP   NZ, 0503   ;NO, RETURN FROM PATCH
05BB:CD 82 01           CALL CLEAR      ;YES, SO CLEAR SCRN, ETC.
05BE:3E 00              LD   A, 00H     ;RESET HP + VP
05C0:32 07 11           LD   (1107H), A;VP=0
05C3:32 08 11           LD   (1108H), A;HP=0
05C6:C3 09 00           JP   NEWLN      ;RETURN TO MONITOR ENTRY PT

05C9:FE 57     MCMDS:   CP   "W"        ;COMMAND W?
05CB:CA 00 60           JP   Z, WARMBOOT ;YES, SO DO WARMBOOT ROUT.
05CE:FE 58              CP   "X"        ;COMMAND X?
05D0:CA 06 06           JP   Z, SAVEBOOT ;YES, SAVE BOOT SECTOR
05D3:FE 54              CP   "T"        ;COMMAND T?
05D5:C2 E3 05           JP   NZ, K
05D8:3E 00              LD   A, 00H
05DA:32 5C 11           LD   (115CH), A
05DD:32 5D 11           LD   (115DH), A
05E0:C3 20 06           JP   RAMTEST
05E3:3A 56 11  K        LD   A, (1156H);GET MORECMDS VARIABLE
05E6:FE 01              CP   01H        ;01=MORE
05E8:C3 43 00           JP   NZ, 0043   ;NO MORE COMMANDS, SO RET
```

Error! Should be "C2"

```
05EB:2A 11 18              LD   HL, (1811) ;READ PATCH ADDR INTO HL
05EE:E9                    JP   (HL)       ;JUMP TO PATCH

* 05ED-05FF     UNUSED

0600:CD 09 0F   WARMBOOT:  CALL BOOT       ;COMMAND W ROUTINE
0603:C3 09 00              JP   NEWLN      ;DONE, BACK TO MONITOR...

0606:3E 01      SAVEBOOT:  LD   A, 01H     ;COMMAND X ROUTINE
0608:32 59 11              LD   (1159H), A ;SECTOR COUNT=1
060B:06 01                 LD   B, 01H     ;SECTOR #=1
060D:0E 00                 LD   C, 00H     ;HEAD #=0
060F:11 00 00              LD   DE, 0000H  ;CYLINDER #=0
0612:CD 90 0B              CALL TSKFILE
0615:21 00 18              LD   HL, 1800H  ;STARTING MEM ADDR=1800H
0618:CD BA 0B              CALL WRITESEC   ;WRITE TO DISK
061B:C3 09 00              JP   NEWLN      ;RETURN TO MONITOR

0620:CD 7C 0D   RAMTEST    CALL CLRLEDS    ;CLEAR LED DISPLAY
0623:21 97 06              LD   HL, TSTMSG1 ;POINT TO MESSAGE
0626:CD 57 0E              CALL LEDMSG     ;PRINT TO LED
0629:21 00 80   MEMTST     LD   HL, 8000H  ;BASE ADDRESS FOR RAM CHECK
062C:7D         LOOP       LD   A, L       ;TEST VALUE
062D:77                    LD   (HL), A    ;FILL MEM LOCATION
062E:23                    INC  HL
062F:FE FF                 CP   FFH        ;IS E AT 00?
0631:C2 2C 06              JP   NZ, LOOP
0634:7C                    LD   A, H
0635:FE 00                 CP   00H        ;DONE?
0637:C2 2C 06              JP   NZ, LOOP   ;NO, KEEP GOING
063A:01 00 00              LD   BC, 0000   ;ERROR COUNTER
063D:CD 7C 0D              CALL CLRLEDS
0640:21 AA 06              LD   HL, TSTMSG2 ;POINT TO 2nd MSG
0643:CD 57 0E              CALL LEDMSG
0646:21 00 80   LOOP3      LD   HL, 8000H
0649:00 00 00
064C:7E         LOOP2      LD   A, (HL)
064D:BD                    CP   L
064E:C2 75 06              JP   NZ, ERROR  ;BAD LOCATION!
0651:2C                    INC  L
0652:C2 4C 06              JP   NZ, LOOP2
0655:24                    INC  H
0656:C2 4C 06              JP   NZ, LOOP2
0659:3E 01                 LD   A, 01H
065B:32 57 11              LD   (1157H), A ;SET DISP TYPE TO LED
065E:3E 04                 LD   A, 04H
0660:32 3D 11              LD   (113DH), A ;SET LED POINTER TO 4
0663:2A 5C 11              LD   HL, (115CH) ;GET CYCLE COUNTER
0666:23                    INC  HL         ;INC CYCLE
0667:22 5C 11              LD   (115CH), HL ;SAVE NEW CYCLE #
066A:CD 70 08              CALL HEXHL      ;PRINT VALUE ON LEDS
066D:3E 00                 LD   A, 00H
066F:3A 57 11              LD   (1157H), A ;SET DISPLAY BACK TO VIDEO
0672:C3 29 06              JP   LOOP3
```

```
0675:03          ERROR     INC   BC          ;INC ERROR COUNTER
0676:CD 70 08              CALL  HEXHL       ;PRINT BAD ADDR ON VIDEO
0679:3E 20                 LD    A, 20H      ;ASCII SPC
067B:CD 17 08              CALL  COUT        ;PRINT A SPACE
067E:3E 01                 LD    A, 01H      ;DISP TYPE=LED=1
0680:32 57 11              LD    (1157H), A  ;STORE IN VAR.
0683:3E 10                 LD    A, 10H      ;SET LED POINTER TO 10
0685:32 3D 11              LD    (113DH), A
0688:E5                    PUSH  HL
0689:60                    LD    H, B
068A:69                    LD    L, C
068B:CD 70 08              CALL  HEXHL       ;PRINT HL ON LEDS
068E:E1                    POP   HL
068F:3E 00                 LD    A, 00H
0691:3A 57 11              LD    (1157H), A  ;DISP TYPE=VIDEO
0694:C3 51 06              JP    0651H       ;RETURN TO LOOP

0697:            TSTMSG1   EQU   "RAM TEST LOADING..."
0697:52 41 4D 20
069B:54 45 53 54
069F:20 4C 4F 41
06A3:44 49 4E 47
06A7:2E 2E 2E 00

06AB:            TESTMSG2  EQU   "ERRORS:0000 CYCLES:0000"
06AB:45 52 52 4F
06AF:52 53 3A 30
06B3:30 30 30 20
06B7:43 59 43 4C
06BB:45 53 3A 30
06BF:30 30 30 00
```

```
*************************************************************************
*                                                                       *
*              SUBROUNTINE INIT_9600                                    *
*                                                                       *
*              INITILIZE SERIAL PORT FOR 9600 BAUD, EVEN PARITY,        *
*              8 DATA BITS, AND TWO STOP BITS.  DISABLE ACE INTERUPTS   *
*                                                                       *
*************************************************************************

0800:F5        INIT_9600  PUSH AF
0801:3E 80                LD   A, 80H     ;ACCESS BAUD RATE DIVIDER
0803:D3 03                OUT  (03H), A
0805:3E 0C                LD   A, 0CH     ;DIVIDER IS 12 FOR 9600
0807:D3 00                OUT  (00H), A
0809:3E 00     G1         LD   A, 00H
080B:D3 01                OUT  (01H), A
080D:3E 1E                LD   A, 1EH     ;PROTOCOL BYTE
080F:D3 03                OUT  (03H), A   ;BAUD RATE DIVIDER NO ACCESS.
0811:3E 00                LD   A, 00H     ;DISABLE INTERUPTS
0813:D3 01                OUT  (01H), A
0815:F1                   POP  AF
0816:C9                   RET

*************************************************************************
*                                                                       *
*              SUBROUTINE COUT                                          *
*                                                                       *
*              ROUTES CONTROL TO PROPER OUTPUT ROUTINE                  *
*                                                                       *
*************************************************************************

0817:3A 57 11  COUT       LD   A, (1157H);GET DISPLAY TYPE
081A:FE 01                CP   01H       ;=LED DISPLAY?
081C:CA 05 0F             JP   Z, PCH4   ;YES, GO TO LED PATCH
081F:DA DD 0B             JP   C, VOUT   ;ITS LESS THAN, SO GOTO VOUT
0822:C3 D1 0E             JP   SOUT      ;ITS NEITHER, GOTO SOUT
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE CIN                                          *
*                                                                       *
*               RETRIEVES A CHARACTER FROM SERIAL PORT                  *
*                                                                       *
*               ON ENTRY - NO SPECIFICATIONS                            *
*               ON EXIT  - ACCUMULATOR HAS LAST CHAR FROM ACE           *
*                                                                       *
*************************************************************************

0826:DB 05      CIN         IN   A, (05H)   ;GET ACE STATUS
0828:CB 47      NOCHR       BIT  0, A       ;DATA READY?
082A:CA 26 08               JP   Z, CIN     ;NO, KEEP WAITING
082D:DB 00                  IN   A, (00H)   ;GET CHARACTER
082F:C9                     RET

*************************************************************************
*                                                                       *
*               SUBROUTINE CR                                           *
*                                                                       *
*               OUTPUTS A *CR* THROUGH THE SERIAL PORT                  *
*               NO REGISTERS AFFECTED                                   *
*                                                                       *
*************************************************************************

0830:C5         CR          PUSH BC         ;SAVE REGS
0831:06 0D                  LD   B, 0DH     ;RETURN TO LEFT MARGIN
0833:CD 17 08               CALL COUT
0836:06 0A                  LD   B, 0AH     ;DOWN ONE LINE
0838:CD 17 08               CALL COUT
083B:C1                     POP  BC         ;RESTORE REGS
083C:C9                     RET

*************************************************************************
*                                                                       *
*               SUBROUTINE INIT_1200                                    *
*                                                                       *
*               INITS ACE TO 1200 BAUD.  SEE INIT_9600 FOR SPECS        *
*                                                                       *
*************************************************************************

083D:F5         INIT_1200   PUSH AF         ;SAVE REGS
083E:3E 80                  LD   A, 80H     ;ENABLE BAUD RATE ACCESS
0840:D3 03                  OUT  (03H), A
0842:3E 60                  LD   A, 60H     ;DIVIDER IS 96 FOR 1200 BAUD
0844:D3 00                  OUT  (00H), A
0846:C3 09 08               JP   G1         ;CONTINUE IN INIT_9600
```

```
*****************************************************************************
*                                                                           *
*               SUBROUTINE HEXBYTE                                          *
*                                                                           *
*               OUTPUTS A TWO-DIGIT HEX NUMBER IN ASCII FORM TO             *
*               THE SERIAL PORT                                             *
*                                                                           *
*               ON ENTRY-ACCUMULATOR HOLDS BYTE TO BE PRINTED               *
*               ON EXIT -BYTE HAS BEEN OUTPUT.  REGS STILL VALID            *
*                                                                           *
*****************************************************************************

0849:C5          HEXBYTE    PUSH BC
084A:F5                     PUSH AF
084B:E6 F0                  AND  F0H        ;MASK OFF LOW NIBBLE
084D:CB 1F                  RR   A          ;ROTATE HIGH NIBBLE TO LOW
084F:CB 1F                  RR   A
0851:CB 1F                  RR   A
0853:CB 1F                  RR   A
0855:C6 90                  ADD  A, 90H     ;GENERATE CARRY IF >9
0857:27                     DAA
0858:CE 40                  ADC  A, 40H     ;ASCII OFFSET
085A:27                     DAA
085B:47                     LD   B, A       ;PUT ASCII VALUE IN B REG
085C:CD 17 08               CALL COUT       ;PRINT IT
085F:F1                     POP  AF
0860:F5                     PUSH AF         ;GET BACK ORIGINAL A REG
0861:E6 0F                  AND  0FH        ;MASK HIGH NIBBLE
0863:C6 90                  ADD  A, 90H     ;GENERATE CARRY IF >9
0865:27                     DAA
0866:CE 40                  ADC  A, 40H     ;ASCII OFFSET
0868:27                     DAA
0869:47                     LD   B, A       ;PUT ASCII VALUE INTO B REG
086A:CD 17 08               CALL COUT       ;PRINT IT
086D:F1                     POP  AF
086E:C1                     POP  BC         ;RESTORE REGS
086F:C9                     RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE HEXHL                                            *
*                                                                           *
*               OUTPUTS THE FOUR-DIGIT HEX VALUE OF HL IN                   *
*               ASCII FORMAT.  REGISTERS REMAIN INTACT                      *
*                                                                           *
*****************************************************************************

0870:F5          HEXHL      PUSH AF         ;SAVE REGS
0871:7C                     LD   A, H
0872:CD 49 08               CALL HEXBYTE    ;PRINT HIGH BYTE FIRST
0875:7D                     LD   A, L
0876:CD 49 08               CALL HEXBYTE    ;PRINT LOW BYTE NEXT
0879:F1                     POP  AF
087A:C9                     RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE LOGON                                        *
*                                                                       *
*               PRINTS LOGON MESSAGE                                    *
*                                                                       *
*************************************************************************

087B:E5          LOGON      PUSH HL
087C:F5                     PUSH AF
087D:C5                     PUSH BC          ;SAVE REGS
087E:21 93 08               LD   HL, 0893H   ;STARTING MESSAGE POINTER
0881:7E          NEXTCHR    LD   A, (HL)     ;GET FIRST CHARACTER
0882:FE 00                  CP   00H         ;END OF MESSAGE?
0884:CA 8F 08               JP   Z, RETN     ;YEAH, WE'RE DONE
0887:47                     LD   B, A        ;PUT CHAR INTO B REG
0888:CD 17 08               CALL COUT        ;OUTPUT IT TO SERIAL PORT
088B:23                     INC  HL          ;ADVANCE POINTER
088C:C3 81 08               JP   NEXTCHR
088F:C1          RETN       POP  BC
0890:F1                     POP  AF
0891:E1                     POP  HL          ;RESTORE REGS
0892:C9                     RET

*************************************************************************
*                                                                       *
*  MSG0 DATA  - *CLS*,"BLITZ 80  VERSION 3.0   ", *EOL*, "LOGON COMPLETE",*
*               *LF*, *CR*, *CR*                                        *
*                                                                       *
*************************************************************************

0893:04 42 4C 49
0897:54 5A 20 38
089B:30 20 20 56
089F:45 52 53 49
08A3:4F 4E 20 33
08A7:2E 30 20 20
08AB:20 00 4C 4F 47
08AF:4F 4E 20 43
08B3:4F 4D 50 4C
08B7:45 54 45 0D
08BB:0A 0A 00
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE GETLN                                        *
*                                                                       *
*************************************************************************

08BF:F5         GETLN     PUSH AF
08C0:C5                   PUSH BC          ;SAVE REGS
08C1:06 2F                LD   B, "/"      ;CURSOR CHAR
08C3:CD 17 08             CALL COUT        ;OUTPUT IT TO SERIAL PORT
08C6:E5                   PUSH HL          ;SAVE HL
08C7:21 00 17             LD   HL, 1700H   ;START OF KBD BUFFER
08CA:CD 26 08   NEXTCHR   CALL CIN         ;GET CHAR FROM SERIAL PORT
08CD:77                   LD   (HL), A     ;STORE CHARACTER
08CE:FE 0D                CP   0DH         ;WAS IT A *CR*?
08D0:CA E2 08             JP   Z, EOL      ;YES, SO WE'RE DONE
08D3:CD 68 09             CALL COUT2       ;NO, ECHO IT
08D6:23                   INC  HL
08D7:7D                   LD   A, L
08D8:FE FF                CP   FFH         ;BUFFER LENGTH IS 256
08DA:C2 CA 08             JP   NZ, NEXTCHR ;THERE'S MORE ROOM...
08DD:36 0D                LD   (HL), 0DH   ;LAST CHAR.  OUTPUT *CR*
08DF:CD E9 08             CALL BELL        ;ALERT USER
08E2:CD 30 08   EOL       CALL CR          ;ECHO *CR*
08E5:E1                   POP  HL
08E6:C1                   POP  BC
08E7:F1                   POP  AF
08E8:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE BELL                                         *
*                                                                       *
*               SENDS *BEL* CHAR TO SERIAL PORT                         *
*                                                                       *
*************************************************************************

08E9:C5         BELL      PUSH BC          ;SAVE REGS
08EA:06 07                LD   B, 07H      ;*BEL* CHAR
08EC:CD 17 08             CALL COUT        ;SEND TO ACE
08EF:C1                   POP  BC          ;RESTORE REGS
08F0:C9                   RET              ;DONE!
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE INITMON                                      *
*                                                                       *
*               INITILIZES VARIABLES ADDR1, ADDR2, ADDRPOINTER          *
*                                                                       *
*************************************************************************

08F1:11 00 00   INITMON   LD   DE, 0000H  ;DEFAULT FOR ADDR1/2
08F4:ED 53 01 11          LD   (1101H), DE ;SET ADDR1 TO 0000H
08F8:ED 53 03 11          LD   (1103H), DE ;SET ADDR2 TO 0000H
08FC:DD 21 01 11          LD   IX, 1101H  ;POINT TO ADDR1
0900:3E 00                LD   A, 00H
0902:32 05 11             LD   (1105H), A ;CLEAR DATA VAR
0905:32 06 11             LD   (1106H), A ;CLEAR DOTFLAG
0908:21 00 17             LD   HL, 01700H ;SET KBD BUFFER POINTER
090B:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE HEX?                                         *
*                                                                       *
*               ON ENTRY - ACCUM. HAS ASCII BYTE IN QUESTION            *
*               ON EXIT  - IF BYTE WAS HEX DIGIT, CARRY IS CLEAR        *
*                          IF NOT, CARRY IS SET                         *
*                                                                       *
*************************************************************************

090C:FE 30      HEX?      CP   30H
090E:FA 20 09             JP   M, NO      ;NUMBER IS LESS THAN 30H
0911:FE 47                CP   47H
0913:F2 20 09             JP   P, NO      ;NUMBER IS MORE THAN 46H
0916:FE 41                CP   41H
0918:F2 22 09             JP   P, YES     ;IT IS BETWEEN 41H AND 46H
091B:FE 3A                CP   3AH
091D:FA 22 09             JP   M, YES     ;IT IS BETWEEN 30H AND 39H
0920:37         NO        SCF             ;NOT LEGAL, SET CARRY FLAG
0921:C9                   RET
0922:37         YES       SCF             ;LEGAL, SET CARRY FLAG...
0923:3F                   CCF             ;THEN CLEAR CARRY FLAG
0924:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE ASCHEX                                       *
*                                                                       *
*               ON ENTRY - ACCUM. HAS ASCII BYTE TO BE CONVERTED        *
*               ON EXIT  - ACCUM CONTAINS HEX VALUE                     *
*               ** MAKE SURE TO CALL HEX? BEFORE CALLING THIS ROUTINE **  *
*                                                                       *
*************************************************************************

0925:FE 41      ASCHEX    CP   41H
0927:F2 2D 09             JP   P, ALPHA   ;IT'S A, B, C, D, E, OR F
092A:E6 0F                AND  0FH        ;MASK UPPER NIBBLE
092C:C9                   RET
092D:E6 0F      ALPHA     AND  0FH        ;MASK UPPER NIBBLE
092F:C6 09                ADD  09H        ;ADD OFFSET
0931:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE SHIFT16                                      *
*                                                                       *
*               ON ENTRY - ACCUM CONTAINS HEX DIGIT TO BE PUT INTO      *
*                          LOW NIBBLE OF ADDR n                         *
*               ON EXIT  - ADDR n HAS NEW ADDRESS.  ACCUM IS TRASHED    *
*                                                                       *
*************************************************************************

0932:C5         SHIFT16   PUSH BC
0933:E6 0F                AND  0FH        ;KILL UPPER NIBBLE
0935:DD 4E 00             LD   C, (IX+0)
0938:DD 46 01             LD   B, (IX+1)
093B:CB 21                SLA  C          ;SHIFT 16 BIT REG 4 TIMES
093D:CB 10                RL   B
093F:CB 21                SLA  C
0941:CB 10                RL   B
0943:CB 21                SLA  C
0945:CB 10                RL   B
0947:CB 21                SLA  C
0949:CB 10                RL   B
094B:B1                   OR   C          ;PUT A LOW NIB INTO C LOW NIB
094C:4F                   LD   C, A
094D:DD 71 00             LD   (IX+0), C
0950:DD 70 01             LD   (IX+1), B
0953:C1                   POP  BC         ;RESTORE BC
0954:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE SHIFT8                                       *
*                                                                       *
*               ON ENTRY - ACCUM HAS NIBBLE TO BE SHIFTED INTO THE      *
*                          DATA BYTE                                    *
*               ON EXIT  - DATA BYTE HAS BEEN UPDATED                   *
*                                                                       *
*************************************************************************

0955:C5         SHIFT8     PUSH BC
0956:47                    LD   B, A
0957:3A 05 11              LD   A, (1105H) ;GET DATA BYTE
095A:CB 27                 SLA  A          ;SHIFT LEFT FOUR TIMES
095C:CB 27                 SLA  A
095E:CB 27                 SLA  A
0960:CB 27                 SLA  A
0962:B0                    OR   B
0963:32 05 11              LD   (1105H), A
0966:C1                    POP  BC
0967:C9                    RET

*************************************************************************
*                                                                       *
*               SUBROUTINE COUT2                                        *
*                                                                       *
*               ON ENTRY - ACCUM HAS CHAR TO BE OUTPUT TO THE SERIAL    *
*                          PORT                                         *
*               ON EXIT  - CHAR HAS BEEN OUTPUT.  ACCUM IS PRESERVED    *
*                                                                       *
*************************************************************************

0968:47         COUT2      LD   B, A       ;PUT CHAR INTO B REG
0969:CD 17 08              CALL COUT       ;CALL OUTPUT ROUTINE
096C:C9                    RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE GETLN2                                       *
*                                                                       *
*               GETS AN ENTIRE LINE OF TEXT FROM THE SERIAL PORT        *
*                                                                       *
*               ON EXIT - KBD BUFFER HAS LINE OF TEXT TERMINATED *CR*   *
*                                                                       *
*               THIS SUBROUTINE IS SHUNTED TO GETLN3 ON CURRENT ROM     *
*               TO RESTORE, THE FIRST LINES OF CODE SHOULD BE:          *
*                                                                       *
* 096D:F5       GETLN2    PUSH AF                                       *
* 096E:C5                 PUSH BC                                       *
* 096F:06 2F              LD   B, 3EH     ;">" CHARACTER                *
*                                                                       *
*************************************************************************

096D:C3 28 0D   GETLN2    JP   GETLN3     ;FOR NON-SERIAL VERSION, JPS
0970:00                   NOP               ; TO GETLN3.  SEE ABOVE.
0971:CD 17 08             CALL COUT       ;OUTPUTS ">"
0974:E5                   PUSH HL         ;SAVE HL
0975:21 00 17             LD   HL, 1700H  ;HL HOLDS KBD BUFFER POINTER
0978:CD 26 08   N1        CALL CIN        ;GET CHARACTER FROM S-PORT
097B:FE 08                CP   08H        ;IS IT CTRL-H?
097D:C2 85 09             JP   F1         ;NO, GO ON
0980:2B                   DEC  HL         ;YES, BACK UP POINTER TWICE
0981:2B                   DEC  HL         ; (FOR CTRL AND DEL CHAR)
0982:C3 90 09             JP   ECHO       ;ECHO IT
0985:FE 0C      F1        CP   0CH        ;IS IT CTRL-L?
0987:CA 90 09             JP   Z, ECHO    ;NO, SO ECHO CHAR
098A:77                   LD   (HL), A    ;GET LAST CHAR ENTERED
098B:FE 0D                CP   0DH        ;WAS IT A *CR* ?
098D:CA 9F 09             JP   Z, X       ;YES, END-OF-LINE
0990:CD 68 09   ECHO      CALL COUT2      ;ECHO CHAR TO S-PORT(SCREEN)
0993:23                   INC  HL         ;MOVE KBD BUFFER POINTER
0994:7D                   LD   A, L       ;CHECK FOR TOO MANY CHARS
0995:FE FF                CP   FFH
0997:C2 78 09             JP   NZ, N1     ;IT'S OK, GET MORE CHARS
099A:36 0D                LD   (HL), 0DH  ;TERMINATE LINE AUTOMATICALLY
099C:CD E9 08             CALL BELL       ;RING BELL ON S-PORT
099F:CD 30 08   X         CALL CR         ;OUTPUT *CR* TO S-PORT
09A2:E1                   POP  HL         ;RESTORE REGS
09A3:C1                   POP  BC
09A4:F1                   POP  AF
09A5:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE MSGOUT                                       *
*                                                                       *
*               ON ENTRY - HL POINTS TO MESSAGE'S ADDRESS IN MEMORY     *
*               ON EXIT  - MESSAGE HAS BEEN PRINTED.  HL IS TRASHED     *
*                                                                       *
*************************************************************************

09A6:F5         MSGOUT    PUSH AF
09A7:C5                   PUSH BC
09A8:7E         NEXTCHR   LD   A, (HL)    ;GET CHAR
09A9:FE 00                CP   00H        ;END-OF-MESSAGE ?
09AB:CA B6 09             JP   Z, RETN    ;YEP.
09AE:47                   LD   B, A       ;NOPE.  PRINT CHAR.
09AF:CD 17 08             CALL COUT
09B2:23                   INC  HL         ;ADVANCE POINTER
09B3:C3 A8 09             JP   NEXTCHR    ;DO IT ALL OVER AGAIN
09B6:C1         RETN      POP  BC
09B7:F1                   POP  AF
09B8:C9                   RET

*************************************************************************
*************************************************************************
**                                                                     **
**               DISK SUBROUTINE SECTION                               **
**                                                                     **
*************************************************************************
*************************************************************************

*************************************************************************
*                                                                       *
*               SUBROUTINE MSGIN                                        *
*                                                                       *
*               THIS SUBROUTINE IS USED TO TYPE IN AN ASCII MESSAGE     *
*               FROM THE KEYBOARD AND PLACE IT IN ASCII FORMAT INTO     *
*               A USER SPECIFIED MEMORY LOCATION.  IT WAS ORIGINALLY    *
*               USED TO TEST THE HARD DISK UNIT AND DYNAMIC RAM CHIPS.  *
*               IT CAN ALSO BE USED FOR OTHER APPLICATIONS.             *
*                                                                       *
*               ALL REGISTERS ARE SAVED AND VALID UPON EXIT             *
*                                                                       *
*************************************************************************

09B9:F5         MSGIN     PUSH AF
09BA:D5                   PUSH DE
09BB:E5                   PUSH HL
09BC:DD E5                PUSH IX
09BE:21 25 0A             LD   HL, MSG10  ;POINTER TO PROMPT LINE
09C1:CD A6 09             CALL MSGOUT     ;PRINT PROMPT LINE
09C4:DD 21 5A 11          LD   IX, 115AH  ;GEN PNTR MEM LOCATION
09C8:DD 36 00 00          LD   (IX+0), 00H ;CLEAR GENPNTR TO 0000H
09CC:DD 36 01 00          LD   (IX+1), 00H
09D0:CD 6D 09             CALL GETLN2     ;GET START ADDR FROM USER
```

```
09D3:21 00 17             LD   HL, 1700H  ;HL POINTS TO KBD BUFFER
09D6:7E         NCHR      LD   A, (HL)    ;GET CHAR FROM BUFFER
09D7:23                   INC  HL         ;INC POINTER
09D8:FE 0D                CP   0DH        ;*CR* ?
09DA:CA EC 09             JP   Z, DONE    ;YES, END-OF-LINE!
09DD:CD 0C 09             CALL HEX?       ;NO.  HEX DIGIT?
09E0:DA D6 09             JP   C, NCHR    ;NO.  IGNORE CHAR AND GO ON
09E3:CD 25 09             CALL ASXHEX     ;YES. CONVERT TO REAL HEX
09E6:CD 32 09             CALL SHIFT16    ;SHIFT INTO GENPNTR
09E9:C3 D6 09             JP   NCHR       ;GET NEXT CHAR
09EC:21 3D 0A   DONE      LD   HL, MSG11  ;NEW PROMPT LINE. ASKS FOR
09EF:CD A6 09             CALL MSGOUT     ;  DATA ENDING W/ CTRL-Z.
09F2:DD 5E 00             LD   E, (IX+0)  ;LOAD DE WITH GENPNTR VALUE
09F5:DD 56 01             LD   D, (IX+1)
09F8:CD 28 0D   NLN       CALL GETLN3     ;GET LINE OF TEXT
09FB:21 00 17             LD   HL, 1700H  ;HL POINTS TO KBD BUFFER
09FE:7E         NCHR2     LD   A, (HL)    ;GET CHAR FROM BUFFER
09FF:FE 0D                CP   0DH        ;*CR* ?
0A01:CA F8 09             JP   Z, NLN     ;NO, KEEP GOING
0A04:FE 1A                CP   1AH        ;CTRL-Z CHAR?
0A06:CA 0F 0A             JP   Z, ENDMSG  ;YES, END-OF-MESSAGE
0A09:12                   LD   (DE), A    ;PUT VALUE INTO MEMORY
0A0A:13                   INC  DE         ;INC MEMORY POINTER
0A0B:23                   INC  HL         ;INC KBD BUFFER POINTER
0A0C:C3 FE 09             JP   NCHR2      ;GO TO NEXT CHAR
0A0F:3E 00                LD   A, 00H     ;TERMINATION CHARACTER
0A11:12                   LD   (DE), A    ;PUT AT END OF TEXT STRING
0A12:21 4F 0A             LD   HL, MSG12  ;PRINT CONFIRMATION MSG.
0A15:CD A6 09             CALL MSGOUT
0A18:EB                   EXX  DE, HL     ;PUT DE (ENDING ADDR)INTO HL
0A19:CD 70 08             CALL HEXHL      ;PRINT IT
0A1C:CD 30 08             CALL CR         ;PRINT *CR*
0A1F:DD E1                POP  IX
0A21:E1                   POP  HL
0A22:D1                   POP  DE
0A23:F1                   POP  AF
0A24:C9                   RET

0A25:{ ENTER STARTING ADDRESS_ } + 00H
0A3D:{ TYPE ^Z TO QUIT } + *CR* + *LF* + 00H
0A4F:{ ENDING ADDRESS WAS_ } + 00H

0A63 - 0AA2 CONTAINS WD1002-05 3:1 INTERLEAVE TABLE
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE BUSY                                         *
*                                                                       *
*               CHECKS STATUS OF HARD DRIVE TO SEE IF IT IS READY.      *
*               IF AN ERROR IS DETECTED, AN ERROR MESSAGE WILL BE       *
*               PRINTED.                                                *
*                                                                       *
*************************************************************************

0AA3:DB 27      BUSY        IN    A, (27H)   ;GET DISK STATUS
0AA5:CD 7F                  BIT   7, A       ;CHECK BUSY BIT
0AA7:C2 A3 0A               JP    NZ, BUSY   ;BUSY, SO LOOP
0AAA:CB 47                  BIT   0, A       ;CHECK ERROR BIT
0AAC:C8                     RETZ             ;NO ERROR SO RETURN
0AAD:DB 21                  IN    A, (21H)   ;ERROR. GET ERROR REGISTER
0AAF:CD 47                  BIT   0, A       ;CHECK FOR DAM NOT FOUND
0AB1:C2 BA 0A               JP    NZ, Q1     ;NO, NEXT...
0AB4:21 F3 0A               LD    HL, MSG20  ;"DAM NOT FOUND"
0AB7:C3 E9 0A               JP    ERRMSG
0ABA:CB 4F      Q1          BIT   1, A       ;CHECK FOR TK000 ERROR
0ABC:C2 C5 0A               JP    NZ, Q2     ;NO, NEXT...
0ABF:21 01 0B               LD    HL, MSG21  ;"TK000 ERROR"
0AC2:C3 E9 0A               JP    ERRMSG
0AC5:CB 57      Q2          BIT   2, A       ;CHECK FOR ABORTED COMMAND
0AC7:C2 D0 0A               JP    NZ, Q3     ;NO, NEXT...
0ACA:21 0D 0B               LD    HL, MSG22  ;"ABORTED COMMAND"
0ACD:C3 E9 0A               JP    ERRMSG
0AD0:CB 67      Q3          BIT   4, A       ;CHECK FOR ID NOT FOUND
0AD2:C2 DB 0A               JP    NZ, Q4     ;NO, NEXT...
0AD5:21 1D 0B               LD    HL, MSG23  ;"ID NOT FOUND"
0AD8:C3 E9 0A               JP    ERRMSG
0ADB:CB 77      Q4          BIT   6, A       ;CHECK FOR UNCORRECT. ERROR
0ADD:C2 E6 0A               JP    NZ, Q5     ;NO, NEXT...
0AE0:21 2A 0B               LD    HL, MSG24  ;"UNCORRECTABLE ERROR"
0AE3:C3 E9 0A               JP    ERRMSG
0AE6:21 3E 0B   Q5          LD    HL, MSG25  ;BAD BLOCK DETECT
0AE9:CD A6 09   ERRMSG      CALL  MSGOUT     ;PRINT ERROR MESSAGE
0AEC:CD 30 08               CALL  CR
0AEF:CD E9 08               CALL  BELL
0AF2:76                     HALT
0AF3:{ DAM NOT FOUND } + 00H                 ;MSG 20
0B01:{ TK000 ERROR } + 00H                   ;MSG 21
0B0D:{ ABORTED COMMAND } + 00H               ;MSG 22
0B1D:{ ID NOT FOUND } + 00H                  ;MSG 23
0B2A:{ UNCORRECTABLE ERROR } + 00H           ;MSG 24
0B3E:{ BAD BLOCK DETECT } + 00H              ;MSG 25
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE BLKIN                                        *
*                                                                       *
*               READS A 256 BYTE BLOCK OF DATA FROM CONTROLLER'S BUFFER  *
*               TSKFILE MUST BE CALLED FIRST TO INIT SECTOR #, ETC      *
*                                                                       *
*               ON ENTRY - HL POINTS TO START OF BUFFER SPACE           *
*                          HARD DISK HAS READ DATA READY                *
*               ON EXIT  - HL THROUGH HL+255 CONTAINS DISK DATA         *
*                          HL IS INTACT AND POINTS TO HEAD OF BUFFER    *
*                                                                       *
*************************************************************************

0B4F:C5         BLKIN     PUSH BC
0B50:00                   NOP              ;SAVE REGISTERS
0B51:06 00                LD   B, 00H      ;RESET COUNTER
0B53:DB 20      NXTIN     IN   A, (20H)    ;GET DATA FROM DISK
0B55:77                   LD   (HL), A     ;PUT INTO BUFFER
0B56:CD 82 0B             CALL DELAY       ;WAIT FOR DISK TO CATCH UP
0B59:00                   NOP
0B5A:B8                   CP   B           ;HAVE 256 BYTES BEEN READ?
0B5B:C2 53 0B             JP   NZ, NXTIN   ;NO.  THERE'S MORE...
0B5E:00                   NOP              ;RESTORE BUFFER HEAD POINTER
0B5F:C1                   POP  BC
0B60:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE BLKOUT                                       *
*                                                                       *
*               OUTPUTS A 256 BYTE BLOCK TO CONTROLLER'S BUFFER         *
*                                                                       *
*               SEE BLKIN FOR SPECS.                                    *
*                                                                       *
*************************************************************************

0B61:C5         BLKOUT    PUSH BC
0B62:00                   NOP
0B63:06 00                LD   B, 00H      ;COUNTER=0
0B65:7E         NXTOUT    LD   A, (HL)     ;GET DATA FROM BUFFER
0B66:D3 20                OUT  (20H), A    ;SEND TO DISK
0B68:CD 82 0B             CALL DELAY       ;WAIT FOR DISK TO CATCH UP
0B6B:00                   NOP
0B6C:B8                   CP   B           ;HAVE 256 BYTES BEEN SENT?
0B6D:C2 65 0B             JP   NZ, NXTOUT  ;NO.  THERE'S MORE...
0B70:00                   NOP              ;RESTORE BUFFER POINTER
0B71:C1                   POP  BC
0B72:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE RESTORE                                      *
*                                                                       *
*               RETURNS HARD DISK R/W HEAD TO TK000 FOR GIVEN           *
*               DRIVE/HEAD                                              *
*                                                                       *
*               ON ENTRY - ACCUM SHOULD CONTAIN THE FOLLOWING:          *
*                          BIT #: 765    4 3     2 1 0                  *
*                                 000 drive 0-2 head 0-7                *
*               ON EXIT  - DISK DRIVE/HEAD RESTORED TO TK000            *
*                                                                       *
*************************************************************************

0B73:C5         RESTORE   PUSH BC
0B74:06 80                LD   B, 80H     ;SET ECC/SEC=256 IN SDH REG
0B76:80                   ADD  A, B       ;DRIVE AND HEAD PASSED IN A.
0B77:D3 26                OUT  (26H), A   ;SEND TO CONTROLLER CARD
0B79:3E 16                LD   A, 16H     ;RESTORE W/3.0 msec STEP RATE
0B7B:D3 27                OUT  (27H), A   ;SEND TO CONTROLLER
0B7D:CD A3 0A             CALL BUSY       ;WAIT FOR DISK
0B80:C1                   POP  BC
0B81:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE DELAY                                        *
*                                                                       *
*               LOOP DELAY SUBROUTINE.  DELAYS _____ ms.                *
*               USED TO WAIT FOR HARD DISK DATA IN/OUT FUNCTIONS        *
*               SHOULD NOT BE USED AS A GENERAL DELAY ROUTINE           *
*                                                                       *
*************************************************************************

0B82:C5         DELAY     PUSH BC
0B83:06 7F                LD   B, 7FH     ;LOOP COUNTER=128
0B85:05         LOOP      DEC  B
0B86:3E 00                LD   A, 00H
0B88:B8                   CP   B          ;DONE?
0B89:C2 85 0B             JP   NZ, LOOP   ;NO, KEEP GOING
0B8C:C1                   POP  BC         ;RESTORE REGS
0B8D:23                   INC  HL         ;INCREASE BUFFER POINTER
0B8E:04                   INC  B          ;INCREASE COUNTER
0B8F:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE TSKFILE                                      *
*                                                                       *
*               SETS UP TASK FILE ON CONTROLLER CARD                    *
*                                                                       *
*               SECTOR COUNT = 1 (INDICATES 512 BYTE XFERS ONLY)        *
*               SECTOR NUMBER = REGISTER B                              *
*               CYLINER NUMBER = REGISTERS DE                           *
*               SIZE/DRIVE/HEAD = ECC/512 BYTE SECTORS/DRIVE 0/REG C    *
*                  (REGISTER C HOLDS HEAD #)                            *
*                                                                       *
*************************************************************************

0B90:00 00      TSKFILE   NOP x2          ;WAS "WRITE PRECOMP VALUE"
0B92:00 00                NOP x2          ;WAS "SEND TO DISK CONTROLLER"
0B94:3E 01                LD   A, 01H     ;SECTOR COUNT=1
0B96:D3 22                OUT  (22H), A   ;SEND TO DISK CONTROLLER
0B98:78                   LD   A, B       ;GET SECTOR #
0B99:D3 23                OUT  (23H), A   ;SEND TO DISK CONTROLLER
0B9B:7B                   LD   A, E       ;GET CYL. #-LOW
0B9C:D3 24                OUT  (24H), A   ;SEND TO DISK CONTROLLER
0B9E:7A                   LD   A, D       ;GET CYL #-HIGH
0B9F:D3 25                OUT  (25H), A   ;SEND TO DISK CONTROLLER
0BA1:79                   LD   A, C       ;GET HEAD #
0BA2:00                   NOP
0BA3:E6 07                AND  07H        ;MASK ALL BUT 3 LOWEST BITS
0BA5:F6 A0                OR   A0H        ;SET ECC BIT
0BA7:D3 26                OUT  (26H), A   ;SEND TO SDH REG ON DISK CTRL
0BA9:C9                   RET


*************************************************************************
*                                                                       *
*               SUBROUTINE READSEC                                      *
*                                                                       *
*               READS A SECTOR FROM THE HARD DISK INTO THE CONTROLLER    *
*               BUFFER, THEN INTO BLITZ-80 MEMORY POINTED TO BY HL      *
*                                                                       *
*               ON ENTRY - HL=START OF BUFFER                           *
*                          B =SECTOR NUMBER                             *
*                          C =HEAD NUMBER                               *
*                          DE=CYLINDER NUMBER                           *
*                                                                       *
*************************************************************************

0BAA:F5         READSEC   PUSH AF
0BAB:CD 90 0B             CALL TSKFILE    ;SET UP TASK FILE
0BAE:3E 20                LD   A, 20H     ;READ SECTOR COMMAND
0BB0:D3 27                OUT  (27H), A   ;SEND TO COMMAND REGISTER
0BB2:CD A3 0A             CALL BUSY       ;WAIT FOR DISK DRIVE...
0BB5:CD 4F 0B             CALL BLKIN      ;GET BLK FROM DISK'S BUFFER
0BB8:F1                   POP  AF
0BB9:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE WRITESEC                                     *
*                                                                       *
*               WRITES A SECTOR TO CONTROLLER BUFFER, THEN TO DISK      *
*                                                                       *
*               SEE SPECS FROM READSEC.                                 *
*                                                                       *
*************************************************************************

0BBA:F5          WRITESEC   PUSH AF
0BBB:CD 90 0B               CALL TSKFILE    ;SET UP TASK FILE
0BBE:3E 30                  LD   A, 30H     ;WRITE SECTOR COMMAND
0BC0:D3 27                  OUT  (27H), A   ;SEND TO COMMAND REGISTER
0BC2:CD 61 0B               CALL BLKOUT     ;SEND DATA TO CTRL BUFFER
0BC5:CD A3 0A               CALL BUSY       ;WAIT FOR DISK DRIVE...
0BC8:F1                     POP  AF
0BC9:C9                     RET

*************************************************************************
*                                                                       *
*               SUBROUTINE FORMAT                                       *
*                                                                       *
*               FORMATS A GIVEN SECTOR USING BLITZ-80 INTERLEAVE        *
*               TABLE AT 0A63H                                          *
*                                                                       *
*               SEE SPECS FROM READSEC FOR REGISTER SETTINGS.           *
*                                                                       *
*************************************************************************

0BCA:F5          FORMAT     PUSH AF
0BCB:CD 90 0B               CALL TSKFILE    ;SET UP TASK FILE
0BCE:3E 50                  LD   A, 50H     ;FORMAT COMMAND
0BD0:D3 27                  OUT  (27H), A   ;SEND TO COMMAND REGISTER
0BD2:21 63 0A               LD   HL, 0A63H  ;INTERLEAVE TABLE START ADDR
0BD5:CD 61 0B               CALL BLKOUT     ;SEND TO CTRL'R BUFFER
0BD8:CD A3 0A               CALL BUSY       ;WAIT FOR DISK DRIVE...
0BDB:F1                     POP  AF
0BDC:C9                     RET
```

```
*******************************************************************************
*******************************************************************************
***                                                                         ***
***            VIDEO SUBROUTINES FOR TI-VDP CHIP                            ***
***                                                                         ***
*******************************************************************************
*******************************************************************************

*******************************************************************************
*                                                                             *
*              SUBROUTINE VOUT                                                *
*                                                                             *
*              PRINT 1 CHARACTER TO SCREEN (EQUIV OF COUT FOR S-PORT)         *
*                                                                             *
*              ON ENTRY - B HAS CHAR TO PRINT                                 *
*              ON EXIT  - CHAR HAS BEEN PRINTED.  B IS INTACT.                *
*                                                                             *
*******************************************************************************

0BDD:F5                 PUSH AF
0BDE:E5                 PUSH HL
0BDF:C5                 PUSH BC
0BE0:00                 NOP
0BE1:78                 LD   A, B       ;GET CHAR. IS IT A CTRL-CHR?
0BE2:FE 0D              CP   0DH        ;IS IT A *CR* ?
0BE4:CA F8 0C           JP   Z, VCR     ;YES, GO TO VCR
0BE7:FE 0A              CP   0AH        ;NO. IS IT A *LF* ?
0BE9:CA 00 0D           JP   Z, LF      ;YES. GO TO LF
0BEC:FE 04              CP   04H        ;NO.  IS IT *CLS* ?
0BEE:CA E1 0C           JP   Z, CLS     ;YES, GO TO CLS
0BF1:00 00 00           NOP  x3         ;SPACE LEFT FOR ADDITIONAL
0BF4:00 00              NOP  x2         ;  CTRL CHARACTER TEST
0BF6:E6 E0              AND  E0H        ;IS THIS A CTRL CHAR AT ALL?
0BF8:FE 00              CP   00H
0BFA:CA 10 0C           JP   Z, VEND    ;YES.  IGNORE AND RETURN
0BFD:CD 22 0C           CALL MATH       ;MUST BE OK.  FIGURE SCRN MEM
0C00:CD 14 0C           CALL ADIN       ;SEND ADDR+CHAR TO VDP
0C03:CD 44 0C           CALL INC_CURS   ;INC CURSOR POSITION
0C06:00 00 00           NOP  x3
0C09:00 00 00           NOP  x3
0C0C:00 00 00           NOP  x3
0C0F:00                 NOP
0C10:C1                 POP  BC
0C11:E1                 POP  HL
0C12:F1                 POP  AF
0C13:C9                 RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE ADIN                                         *
*                                                                       *
*               SENDS ADDRESS OF CHAR AND CHAR ITSELF TO VDP CHIP       *
*                                                                       *
*               ON ENTRY - HL HAS ADDRESS, B HAS CHARACTER              *
*               ON EXIT  - REGISTERS ARE INTACT                         *
*                                                                       *
*************************************************************************

0C14:DB 19      ADIN      IN   A, (19H)   ;RESET VDP CHIP
0C16:7D                   LD   A, L       ;GET ADDR low
0C17:D3 19                OUT  (19H), A   ;SEND TO VDP
0C19:7C                   LD   A, H       ;GET ADDR high
0C1A:F6 40                AND  40H        ;'WRITE COMMAND' MASK
0C1C:D3 19                OUT  (19H), A   ;SEND TO VDP
0C1E:78                   LD   A, B       ;GET CHARACTER DATA
0C1F:D3 18                OUT  (18H), A   ;SEND TO VDP DATA PORT
0C21:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE MATH                                         *
*                                                                       *
*               COMPUTES ACTUAL MEMORY ADDRESS USING VP AND HP          *
*                                                                       *
*               ON ENTRY - VP AND HP REGISTERS SET TO SCREEN POSITION   *
*               ON EXIT  - HL CONTAINS ACTUAL SCREEN MEMORY ADDRESS     *
*                                                                       *
*************************************************************************

0C22:C5         MATH      PUSH BC
0C23:06 00                LD   B, 00H     ;USED IN ROUTINE
0C25:26 00                LD   H, 00H
0C27:3A 07 11             LD   A, (1107H) ;GET VERTICAL POSITION
0C2A:CB 07                RLC  A          ;ROTATE LEFT W/CARRY x3
0C2C:CB 07                RLC  A          ;THIS MULTIPLIES A TIMES 8
0C2E:CB 07                RLC  A
0C30:4F                   LD   C, A       ;C = A = 8X
0C31:37                   SCF             ;SET CARRY FLAG
0C32:3F                   CCF             ;COMPLIMENT CARRY FLAG
0C33:CB 17                RL   A
0C35:CB 14                RL   H
0C37:CB 17                RL   A
0C39:CB 14                RL   H          ;HL = 4(8X)
0C3B:6F                   LD   L, A
0C3C:09                   ADD  HL, BC     ;HL = 4(8X)+8X = 40X
0C3D:3A 08 11             LD   A, (1108H);GET HORIZ POSITION
0C40:4F                   LD   C, A       ;PUT IT IN C.  C = Y
0C41:09                   ADD  HL, BC     ;HL=40X + Y
0C42:C1                   POP  BC         ;HL NOW HAS MEM. POSITION!
0C43:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE INC_CURS                                     *
*                                                                       *
*               MOVES CURSOR POSITION OVER 1 SPACE.  IF IT IS AT THE    *
*               END OF A HORIZONTAL LINE, IT ROLLS OVER TO THE START    *
*               OF THE NEXT LINE.  IF THE NEXT LINE IS OFF THE SCREEN   *
*               INC_CURS WILL SCROLL THE SCREEN VERTICALLY 1 LINE.      *
*                                                                       *
*************************************************************************

0C44:3A 08 11   INC_CURS   LD   A, (1108H) ;GET HP
0C47:3C                    INC  A          ;INC IT
0C48:FE 28                 CP   28H        ;LINE LEN=40.  IS IT MORE?
0C4A:CA 51 0C              JP   Z, I1      ;YES.
0C4D:32 08 11              LD   (1108H), A ;NO. STORE NEW HP
0C50:C9                    RET
0C51:3E 00      I1         LD   A, 00H     ;SET HP TO 0
0C53:32 08 11              LD   (1108H), A
0C56:3A 07 11              LD   A, (1107H) ;GET VP
0C59:3C                    INC  A          ;INC IT
0C5A:FE 18                 CP   18H        ;SCREEN LEN=24.  IS IT MORE?
0C5C:CA C7 0E              JP   Z, SCRPCH  ;YES.  SCROLL SCREEN
0C5F:32 07 11              LD   (1107), A  ;NO.  STORE NEW VP
0C62:C9                    RET
0C63:3E 00      SCROLL     LD   A, 00H     ;START RE-WRITING SCREEN
0C65:32 08 11              LD   (1108H), A ;SET HP = 0
0C68:3E 01                 LD   A, 01H     ;START AT SECOND ROW
0C6A:32 07 11              LD   (1107H), A ;VP=1 FOR 2nd ROW!
0C6D:CD 22 0C   C1         CALL MATH       ;GET BASE ADDR
0C70:DB 19                 IN   A, (19H)   ;INIT VDP
0C72:7D                    LD   A, L       ;
0C73:D3 19                 OUT  (19H), A   ;SEND ADDR low TO VDP
0C75:7C                    LD   A, H
0C76:D3 19                 OUT  (19H), A   ;SEND ADDR high + READ COMMAND
0C78:21 0A 11              LD   HL, 110AH  ;(HL)=SCROLL BUFFER
0C7B:DB 18      C2         IN   A, (18H)   ;GET BYTE FROM VDP
0C7D:77                    LD   (HL), A    ;PUT IT IN BUFFER
0C7E:23                    INC  HL         ;INC POINTER
0C7F:7D                    LD   A, L
0C80:FE 32                 CP   32H        ;40 CHARACTERS (+OFFSET)?
0C82:C2 7B 0C              JP   NZ, C2     ;NO, KEEP GOING T0 END OF LINE
0C85:3A 07 11              LD   A, (1107H) ;GOT A WHOLE LINE...
0C88:3D                    DEC  A
0C89:32 07 11              LD   (1107H), A ;VP=VP-1 (WRITE LINE)
0C8C:CD 22 0C              CALL MATH       ;CALCULATE BASE ADDR
0C8F:DB 19                 IN   A, (19H)   ;INIT VDP
0C91:7D                    LD   A, L
0C92:D3 19                 OUT  (19H), A   ;SEND ADDR low TO VDP
0C94:7C                    LD   A, H
0C95:F6 40                 OR   40H        ;'WRITE COMMAND' MASK
0C97:D3 19                 OUT  (19H), A   ;SEND TO VDP
0C99:21 0A 11              LD   HL, 110AH  ;POINTS TO SCROLL BUFFER
0C9C:7E         C3         LD   A, (HL)    ;GET BYTE FROM BUFFER
```

```
0C9D:D3 18                OUT  (18H), A   ;SEND CHAR TO VDP
0C9F:23                   INC  HL         ;NEXT CHAR
0CA0:7D                   LD   A, L
0CA1:FE 32                CP   32H        ;CHECK FOR END-OF-LINE
0CA3:C2 9C 0C             JP   NZ, C3     ;NO, KEEP GOING...
0CA6:3A 07 11             LD   A, (1107H)
0CA9:3C                   INC  A          ;VP=VP AT START
0CAA:FE 17                CP   17H        ;AT BOTTOM OF SCREEN?
0CAC:CA B6 0C             JP   Z, DONE    ;YES SO WE'RE DONE...
0CAF:3C                   INC  A          ;NO.  NEXT LINE...
0CB0:32 07 11             LD   (1107H), A;PUT VP IN VP LOCATION
0CB3:C3 6D 0C             JP   C1
0CB6:32 07 11   DONE      LD   (1107H), A;BOTTOM LINE OF SCREEN
0CB9:CD BF 0C             CALL CLRLN      ;CLEAR IT
0CBC:C9 00 00             RET             ;LEAVE VOUT ROUTINE

*************************************************************************
*                                                                       *
*               SUBROUTINE CLRLN                                        *
*                                                                       *
*               FILLS LINE AT VP WITH BLANKS                            *
*                                                                       *
*************************************************************************

0CBF:C5         CLRLN     PUSH BC
0CC0:3E 00                LD   A, 00H
0CC2:32 08 11             LD   (1108H), A ;SET HP TO 0
0CC5:CD 22 0C             CALL MATH       ;CALC BASE ADDR
0CC8:06 20                LD   B, 20H     ;ASCII SPACE CHARACTER
0CCA:CD 14 0C             CALL ADIN       ;SEND ADDR+CHAR TO VDP
0CCD:06 27                LD   B, 27H     ;B IS COUNTER SET AT 39
0CCF:3E 20      C4        LD   A, 20H     ;ASCII SPACE CHAR
0CD1:D3 18                OUT  (18H), A   ;SEND TO VDP
0CD3:05                   DEC  B          ;DEC COUNTER
0CD4:3E 00                LD   A, 00H
0CD6:B8                   CP   B          ;DONE YET?
0CD7:C2 CF 0C             JP   NZ, C4
0CDA:C1                   POP  BC
0CDB:C9                   RET
0CDC:00 00 00             NOP  x3
0CDF:00 00                NOP  x2
```

```
*************************************************************************
*                                                                       *
*               ROUTINE CLS                                             *
*                                                                       *
*               CLEARS VIDEO SCREEN, SETS VP/HP, PRINT CURSOR IN        *
*               COL. 1, ROW 1.  CANNOT BE 'CALLED' !!                   *
*                                                                       *
*************************************************************************

0CE1:CD 82 01   CLS         CALL CLEAR      ;CLEAR SCREEN
0CE4:3E 00                  LD   A, 00H
0CE6:32 11 07               LD   (1107H), A ;CLEAR VP (VP=0)
0CE9:32 11 08               LD   (1108H), A ;CLEAR HP (HP=0)
0CEC:CD 22 0C   PLCCUR      CALL MATH       ;CALC BASE ADDR
0CEF:3A 09 11               LD   A, (1109H) ;GET CURSOR CHAR
0CF2:CD 14 0C               CALL ADIN       ;SEND TO VDP
0CF5:C3 0E 0C               JP   VEND

*************************************************************************
*                                                                       *
*               VCR AND LF                                              *
*                                                                       *
*               EXECUTES A *CR* OR *LF* ON VIDEO SCREEN.                *
*               CANNOT BE 'CALLED'.  FOR VOUT'S USE ONLY                *
*                                                                       *
*************************************************************************

0CF8:3E 00      VCR         LD   A, 00H
0CFA:32 08 11               LD   (1108H), A;HP=0
0CFD:C3 0E 0C               JP   VEND
0D00:3A 07 11   LF          LD   A, (1107H);GET VP
0D03:3C                     INC  A          ;INC VP
0D04:FE 18                  CP   18H        ;END OF SCREEN?
0D06:CA CB 0E               JP   Z, SCROLL  ;YES, SO WE MUST SCROLL
0D09:32 07 11               LD   (1107H), A;NO, VP=VP+1
0D0C:C3 0E 0C               JP   VEND       ;THAT'S ALL, FOLKS!

*************************************************************************
*                                                                       *
*               SUBROUTINE CIN2                                         *
*                                                                       *
*               GETS A CHARACTER FROM BLITZ-80 KBD PORT (NOT S-PORT!)   *
*               WAITS UNTIL A KEY IS PRESSED TO RETURN.                 *
*                                                                       *
*               ACCUM HAS CHARACTER VALUE                               *
*                                                                       *
*************************************************************************

0D0F:DB 12      CIN         IN   A, (12H)   ;KBD STATUS PORT
0D11:CB 6F                  BIT  5, 0
0D13:CA 0F 0D               JP   Z, CIN     ;NO KEY HIT YET SO LOOP
0D16:DB 10                  IN   A, (10H)   ;KEY PRESSED.  GET CHAR.
0D18:ED 44                  NEG             ;CONVERT TO ASCII!
0D1A:37                     SCF             ;SET CARRY FLAG
```

```
0D1B:3F                 CCF             ;COMPLIMENT CARRY FLAG
0D1C:DE 01              SBC  01H        ;SUBTRACT 1 FROM A
0D1E:C9                 RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE OOUT                                             *
*                                                                           *
*               OUTPUTS VALUE OF REG B AT HP, VP.  ALL REGS INTACT          *
*               DOES NOT SCROLL.                                            *
*                                                                           *
*****************************************************************************

0D1F:E5         OOUT    PUSH HL
0D20:CD 22 0C           CALL MATH       ;CALC BASE ADDR
0D23:CD 14 0C           CALL ADIN       ;SEND ADDR+CHAR TO VDP
0D26:E1                 POP HL
0D27:C9                 RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE GETLN3                                           *
*                                                                           *
*               REPLACES GETLN2 FOR VIDEO MONITOR USE.  GETS AN ENTIRE      *
*               LINE OF TEXT AND ECHOS IT ON VIDEO SCREEN                   *
*                                                                           *
*****************************************************************************

0D28:F5         GETLN3  PUSH AF
0D29:C5                 PUSH BC
0D2A:06 3E              LD   B, 3EH     ;">" CHARACTER
0D2C:CD DD 0B           CALL VOUT       ;PRINT IT
0D2F:E5                 PUSH HL         ;SAVE REG
0D30:21 00 17           LD   HL, 1700H  ;POINT TO KBD BUFFER
0D33:3A 09 11   NEXTCHR LD   A, (1109H);GET CURSOR CHAR
0D36:47                 LD   B, A
0D37:CD 1F 0D           CALL OOUT       ;PRINT CURSOR CHAR
0D3A:CD 0F 0D           CALL CIN        ;GET A CHAR FROM KBD
0D3D:77                 LD   (HL), A    ;PUT IT IN BUFFER
0D3E:FE 0D              CP   0DH        ;IS IT A *CR* ?
0D40:CA 58 0D           JP   Z, EOL     ;YES, END OF LINE...
0D43:FE 08              CP   08H        ;NO.  IS IT A BACKSPC ?
0D45:CA 64 0D           JP   Z, BKSPC   ;YES, DEAL WITH IT.
0D48:47                 LD   B, A       ;NO.  ECHO IT
0D49:CD DD 0B           CALL VOUT
0D4C:23                 INC  HL
0D4D:7D                 LD   A, L
0D4E:FE FF              CP   FFH        ;LINE FULL ?
0D50:C2 33 0D           JP   NZ, NEXTCHR ;NO, WE HAVE ROOM FOR MORE
0D53:36 0D              LD   (HL), 0DH  ;INSERT AUTO *CR* TO BUFFER
0D55:CD E9 08           CALL BELL       ;RING WARNING BELL
0D58:06 20      EOL     LD   B, 20H     ;SPACE CHAR
0D5A:CD 1F 0D           CALL OOUT       ;CLEAR CURSOR CHAR
0D5D:CD 30 08           CALL CR         ;*CR*
```

```
0D60:E1                  POP  HL         ;RESTORE REGS
0D61:C1                  POP  BC
0D62:F1                  POP  AF
0D63:C9                  RET
0D64:3A 08 11  BKSPC     LD   A, (1108H);GET HP
0D67:FE 00               CP   00H        ;START OF LINE?
0D69:CA 3A 0D            JP   Z, NEXTCHR+8 ;YES, SO IGNORE IT
0D6C:06 20               LD   B, 20H     ;ASCII SPACE CHAR
0D6E:CD 1F 0D            CALL OOUT       ;WIPE UNWANTED CHAR
0D71:3A 08 11            LD   A, (1108H);GET HP BACK IN ACCUM
0D74:3D                  DEC  A          ;GO BACK ONE POSITION
0D75:32 08 11            LD   (1108H), A;HP=HP-1
0D78:2B                  DEC  HL         ;DEC BUFFER PNTR
0D79:C3 33 0D            JP   NEXTCHR

*************************************************************************
*                                                                       *
*               SUBROUTINE CLEARLEDS                                    *
*                                                                       *
*               CLEARS BOTH THE LED READ-OUT AND FILLS THE LED BUFFER   *
*               WITH 20h (ASCII 'SPC').                                 *
*                                                                       *
*               NO REGISTERS ARE AFFECTED                               *
*                                                                       *
*************************************************************************

0D7C:C5          CLRLED:  PUSH BC
0D7D:E5                   PUSH HL
0D7E:F5                   PUSH AF
0D7F:21 3E 11             LD   HL, 113EH  ;START OF LED BUFFER SPACE
0D82:3E 20                LD   A, 20H     ;ASCII 'SPACE' CHARACTER
0D84:D3 2B                OUT  (2BH), A   ;LED DATA PORT
0D86:06 17                LD   B, 17H     ;COUNTER VALUE
0D88:78          LOOP1:   LD   A, B       ;PUT COUNTER IN REG A
0D89:D3 2C                OUT  (2CH), A   ;OUTPUT TO LED POSITION PORT
0D8B:D3 2D                OUT  (2DH),A    ;STROBE LEDS
0D8D:3E 20                LD   A, 20H     ;ASCII 'SPACE' CHAR
0D8F:77                   LD   (HL), A    ;PUT IN BUFFER
0D90:23                   INC  HL         ;NEXT BUFFER POSITION
0D91:05                   DEC  B          ;DECREASE COUNTER
0D92:F2 88 0D             JP   P, LOOP1   ;NOT END-OF-LOOP
0D95:3E 17                LD   A, 17H     ;1ST CHAR AVAILABLE
0D97:32 3D 11             LD   (113DH), A;STORE IN LED POINTER
0D9A:F1                   POP  AF
0D9B:E1                   POP  HL
0D9C:C1                   POP  BC
0D9D:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE BLNKLED                                      *
*                                                                       *
*               BLINKS LED DISPLAY ONCE                                 *
*                                                                       *
*               NO REGISTERS ARE AFFECTED                               *
*                                                                       *
*************************************************************************

0D9E:F5         BLNKLED:  PUSH AF
0D9F:E5                   PUSH HL
0DA0:3E 20                LD   A, 20H     ;FIRST CLEAR THE DISPLAY....
0DA2:D3 2B                OUT  (2BH), A   ;SEND TO LED DATA PORT
0DA4:3E 17                LD   A, 17H
0DA6:D3 2C      LOOP1:    OUT  (2CH), A   ;SEND TO LED ADDR PORT
0DA8:D3 2D                OUT  (2DH), A   ;STROBE LED PORT
0DAA:3D                   DEC  A
0DAB:F2 A6 0D             JP   P, LOOP1   ;LOOP 'TIL DONE
0DAE:CD C6 0D             CALL WAIT       ;DELAY ROUTINE
0DB1:21 3E 11             LD   HL, 113EH  ;BUFFER POINTER
0DB4:06 17                LD   B, 17H     ;SET COUNTER TO 24
0DB6:7E         LOOP2:    LD   A, (HL)    ;GET OLD VALUE
0DB7:D3 2B                OUT  (2BH), A   ;SEND TO LED DATA PORT
0DB9:78                   LD   A, B       ;LED POSITION
0DBA:D3 2C                OUT  (2CH), A   ;SEND TO LED ADDR PORT
0DBC:D3 2D                OUT  (2DH), A   ;STROBE LED PORT
0DBE:23                   INC  HL         ;NEXT POS IN BUFFER
0DBF:05                   DEC  B          ;DEC COUNTER
0DC0:F2 B6 0D             JP   P, LOOP2   ;LOOP 'TIL DONE
0DC3:E1                   POP  HL         ;RESTORE REGS AND RETURN
0DC4:F1                   POP  AF
0DC5:C9                   RET

*************************************************************************
*                                                                       *
*               SUBROUTINE WAIT                                         *
*                                                                       *
*               REAL-TIME PAUSE ROUTINE                                 *
*                                                                       *
*               NO REGISTERS ARE AFFECTED                               *
*                                                                       *
*************************************************************************

0DC6:E5         WAIT:     PUSH HL
0DC7:F5                   PUSH AF
0DC8:21 FF 2F             LD   HL, 2FFFH  ;DELAY FACTOR = APPROX .5 SEC
0DCB:2B         LOOP1:    DEC  HL
0DCC:3E 00                LD   A, 00H
0DCE:BC                   CP   H          ;CHECK TO SEE IF WERE DONE
0DCF:C2 CB 0D             JP   NZ, LOOP1  ;IF NOT, KEEP LOOPING
0DD2:F1                   POP  AF
0DD3:E1                   POP  HL
0DD4:C9                   RET
```

```
*************************************************************************
*                                                                       *
*               SUBROUTINE PRNTLED                                      *
*                                                                       *
*               PRINTS AN ASCII CHARACTER TO THE LED ARRAY.  AUTO-SCROLL *
*               IF THE CHARACTER GOES OFF THE END OF THE SCREEN         *
*                                                                       *
*               ON ENTRY - ACCUM HAS CHARACTER TO BE DISPLAYED          *
*               ON EXIT - ACCUM IS NO LONGER VALID                      *
*                                                                       *
*************************************************************************

0DD5:E5         PRNTLED:  PUSH HL
0DD6:C5                   PUSH BC
0DD7:47                   LD   B, A       ;STORE CHARACTER DATA
0DD8:3A 3D 11             LD   A, (113DH);GET LED ADDR PTR
0DDB:FE FF                CP   FFH        ;ARE WE AT END OF DISPLAY?
0DDD:CA FA 0D             JP   Z, SCROLL  ;YES, SO SCROLL DISPLAY FIRST
0DE0:4F                   LD   C, A       ;STORE ADDR DATA
0DE1:3D                   DEC  A          ;UPDATE POINTER
0DE2:32 3D 11             LD   (113DH), A;..AND STORE BACK IN VARIABLE
0DE5:78                   LD   A, B       ;GET CHAR DATA
0DE6:D3 2B                OUT  (2BH), A   ;SEND TO LED DATA PORT
0DE8:79                   LD   A, C       ;GET ADDR DATA
0DEA:D3 2C                OUT  (2CH), A   ;SEND TO LED ADDR PORT
0DEC:D3 2D                OUT  (2DH), A   ;STROBE LEDS
0DEE:3E 17                LD   A, 17H     ;'INVERT' POSITION TO CALC.
0DF0:91                   SUB  A, C       ;..POSITION IN BUFFER
0DF1:C6 3E                ADD  A, 3EH     ;ADD OFFSET - LOW
0DF3:6F                   LD   L, A       ;PUT IN LOW REG
0DF4:26 11                LD   H, 11H     ;SET UP OFFSET - HIGH
0DF6:70                   LD   (HL), B    ;STORE CHAR IN BUFFER
0DF7:C1                   POP  BC         ;RESTORE REGS
0DF8:E1                   POP  HL
0DF9:C9                   RET
0DFA:0E 17      SCROLL:   LD   C, 17H     ;COUNTER
0DFC:21 3F 11             LD   HL, 113FH  ;BUFFER POINTER
0DFF:7E         LOOP1:    LD   A, (HL)    ;GET VALUE
0E00:2B                   DEC  HL         ;SHIFT OVER 1 PLACE ON DISP...
0E01:77                   LD   (HL), A    ;AND PLACE NEW, SCROLLED CHAR.
0E02:D3 2B                OUT  (2BH), A   ;SEND TO LED DATA PORT
0E04:79                   LD   A, C       ;GET ADDR REG
0E05:D3 2C                OUT  (2CH), A   ;SEND TO LED ADDR PORT
0E07:D3 2D                OUT  (2DH), A   ;STROBE LEDS
0E09:23                   INC  HL
0E0A:23                   INC  HL         ;SET POINTER TO NXT CHAR
0E0B:0D                   DEC  C          ;DECREASE COUNTER
0E0C:C2 FF 0D             JP   NZ, LOOP1  ;LOOP THROUGH WHOLE BUFFER
0E0F:78                   LD   A, B       ;GET NEW CHARACTER
0E10:D3 2B                OUT  (2BH), A   ;SEND TO LED DATA PORT
0E12:79                   LD   A, C       ;GET ADDR - (MUST = 00)
0E13:D3 2C                OUT  (2CH), A   ;SEND TO LED ADDR PORT
0E15:D3 2D                OUT  (2DH), A   ;STROBE LEDS
0E17:2B                   DEC  HL
```

```
0E18:70                 LD   (HL), B    ;PUT CHARACTER IN BUFFER
0E19:CD C6 0D           CALL WAIT       ;HOLD TIME
0E1C:C1                 POP  BC
0E1D:E1                 POP  HL
0E1E:C9                 RET

0E1F:00

*************************************************************************
*                                                                       *
*             SUBROUTINE REVPRTLED                                      *
*                                                                       *
*             ON ENTRY - A HAS CHARACTER TO BE DISPLAYED ON LEDS        *
*             ON EXIT - CHAR HAS BEEN PRINTED AT LEFT SIDE OF           *
*               DISPLAY AND CHARACTERS SCROLLED RIGHT                   *
*                                                                       *
*                                                                       *
*             ON ENTRY - ACCUM HAS CHARACTER TO BE DISPLAYED            *
*             ON EXIT - ACCUM IS NO LONGER VALID                        *
*                                                                       *
*************************************************************************

0E20:E5       REVPNTLED:PUSH HL
0E21:C5                 PUSH BC
0E22:47                 LD   B, A       ;SAVE CHARACTER
0E23:0E 00              LD   C, 00H     ;COUNTER
0E25:21 54 11           LD   HL, 1154H  ;BUFFER POINTER
0E28:7E       LOOP1:    LD   A, (HL)    ;GET BUFFER CHAR
0E29:23                 INC  HL         ;POINT TO NEXT PLACE
0E2A:77                 LD   (HL), A    ;AND SHIFT OVER CHAR
0E2B:D3 2B              OUT  (2BH), A   ;AND SEND TO LED DATA PORT
0E2D:79                 LD   A, C       ;GET LED POSITION FROM COUNTER
0E2E:D3 2C              OUT  (2CH), A   ;SEND TO LED ADDR PORT
0E30:D3 2D              OUT  (2DH), A   ;STROBE LEDS
0E32:2B                 DEC  HL
0E33:2B                 DEC  HL         ;REPOSITION POINTER
0E34:0C                 INC  C          ;INC  COUNTER
0E35:79                 LD   A, C       ;GET COUNTER
0E36:FE 17              CP   17H        ;DONE?
0E38:C2 28 0E           JP   NZ, LOOP   ;NO, SO LOOP
0E3B:23                 INC  HL         ;UPDATE POINTER
0E3C:78                 LD   A, B       ;GET NEW CHARACTER
0E3D:77                 LD   (HL), A    ;PUT IN BUFFER
0E3E:D3 2B              OUT  (2BH), A   ;SEND TO LED DATA PORT
0E40:79                 LD   A, C       ;GET COUNTER (MUST=17)
0E41:D3 2C              OUT  (2CH), A   ;SEND TO LED ADDR PORT
0E43:D3 2D              OUT  (2DH), A   ;STROBE LEDS
0E45:3A 3D 11           LD   A, (113D)  ;GET POINTER
0E48:FE FF              CP   FFH        ;IS IT AT RIGHTMOST POS?
0E4A:CA 51 0E           JP   Z, DONE    ;YES, SO LEAVE IT ALONE
0E4D:3D                 DEC  A          ;NO, SO MOVE IT
0E4E:32 3D 11           LD   (113D), A ;SAVE IT
0E51:CD C6 0D           CALL WAIT       ;REAL TIME DELAY
0E54:C1                 POP  BC
```

```
0E55:E1                     POP   HL
0E56:C9                     RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE LEDMSG                                           *
*                                                                           *
*               ON ENTRY - HL POINTS TO MESSAGE TO BE DISPLAYED             *
*                 ON LED SCREEN                                             *
*                                                                           *
*               NO REGISTERS ARE AFFECTED                                   *
*                                                                           *
*****************************************************************************

0E57:7E         LEDMSG:     LD    A, (HL)     ;GET CHARACTER FROM MSG
0E58:FE 00                  CP    00H         ;END OF MSG MARKER?
0E5A:C8                     RETZ              ;YES, SO GO BACK
0E5B:23                     INC   HL          ;NEXT CHARACTER
0E5C:00                     NOP
0E5D:CD D5 0D               CALL  PRNTLED     ;NO, SO PRINT IT TO LEDS
0E60:C3 57 0E               JP    LEDMSG      ;LOOP TO NEXT CHAR....
0E63:C9                     RET

0E64:           NODSK       EQU   "NO HARD DRIVE FOUND"
0E64:4E 4F 20 48
0E68:41 52 44 20
0E6C:44 49 53 4B
0E70:20 46 4F 55
0E74:4E 44 00

0E77:           YESDSK      EQU   "BOOTING HARD DRIVE..."
0E77:42 4F 4F 54
0E7B:49 4E 47 20
0E7F:48 41 52 44
0E83:20 44 49 53
0E87:4B 2E 2E 2E
0E8B:00

0E8C:           INITMSG     EQU   "BLITZ-80 INIT SEQUENCE"
0E8C:42 4C 49 54
0E90:5A 2D 38 30
0E94:20 2D 20 49
0E98:4E 49 54 20
0E9C:53 45 51 55
0EA0:45 4E 43 45
0EA4:00

0EA5:E5         BLKIN2      PUSH  HL
0EA6:CD 4F 0B               CALL  BLKIN
0EA9:CD 4F 0B               CALL  BLKIN
0EAC:E1                     POP   HL
0EAD:C9
```

```
0EAE:E5          BLKOUT2    PUSH HL
0EAF:CD 61 0B               CALL BLKOUT
0EB2:CD 61 0B               CALL BLKOUT
0EB5:E1                     POP  HL
0EB6:C9                     RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE WAITHL                                           *
*                                                                           *
*               ON ENTRY - HL HAS DELAY VALUE                               *
*               ON EXIT - HL IS VALID, APPROX 3 SECONDS HAVE PASSED         *
*                                                                           *
*****************************************************************************

0EB7:E5          WAITHL     PUSH HL          ;SAVE HL
0EB8:2B          LOOP:      DEC  HL          ;DECREASE COUNTER
0EB9:7C                     LD   A, H        ;CHECK HL=0000?
0EBA:FE 00                  CP   00H
0EBC:C2 B8 0E               JP   NZ, LOOP
0EBF:7D                     LD   A, L
0EC0:FE 00                  CP   00H
0EC2:C2 B8 0E               JP   NZ, LOOP
0EC5:E1                     POP  HL
OEC6:C9                     RET

** PATCH 2
0EC7:CD 63 0C    PCH2       CALL SCROLLSCR
0ECA:C9                     RET

** PATCH 3
0ECB:CD 63 0C    PCH3       CALL SCROLLSCR
0ECE:C3 10 0C               JP   VEND

*****************************************************************************
*                                                                           *
*               SUBROUTINE SOUT                                             *
*                                                                           *
*               OUTPUTS A CHARACTER THROUGH THE SERIAL PORT                 *
*                                                                           *
*               ON ENTRY- CHAR TO BE PRINTED IS IN B REGISTER               *
*               ON EXIT - CHAR HAS BEEN PRINTED.  B IS STILL VALID          *
*                                                                           *
*****************************************************************************

0ED1:F5          SOUT       PUSH AF
0ED2:C5                     PUSH BC          ;SAVE REGS
0ED3:DB 05       NOTYET     IN   A, (05H)    ;GET ACE STATUS
0ED5:CB 6F                  BIT  5, A        ;TEST TRANS BUFFER EMPTY
0ED7:CA D3 0E               JP   Z, NOTYET   ;WAIT TILL EMPTY
0EDA:78                     LD   A, B        ;GET CHAR FROM B REG
0EDB:D3 00                  OUT  (00H), A    ;SEND CHAR TO ACE
0EDD:C1                     POP  BC
0EDE:F1                     POP  AF          ;RESTORE REGS
```

```
0EDF:C9                    RET

*****************************************************************************
*                                                                           *
*               SUBROUTINE VARSET                                           *
*                                                                           *
*               SETS UP THE FOLLOWING SYSTEM VARIABLES:                     *
*                 MORECMDS (1156H) = 00  ;NO MORE COMMANDS                  *
*                 DISPTYPE (1157H) = 00  ;VIDEO MONITOR                     *
*                 BOOTFLAG (1158H) = 00  ;COLD BOOT                         *
*                 LEDPNTR  (113DH) = 17  ;POINTS TO LEFTMOST LED            *
*                                                                           *
*****************************************************************************

0EE0:3E 17      VARSET:    LD   A, 17H     ;LEFTMOST LED #
0EE2:32 3D 11              LD   (113D), A  ;STORE IN VARIABLE
0EE5:3E 00                 LD   A, 00H
0EE7:32 56 11              LD   (1156H), A;MORECMS
0EEA:32 57 11              LD   (1157H), A;DISPLAY TYPE
0EED:32 58 11              LD   (1158H), A;BOOTFLAG
0EF0:3E 01                 LD   A, 01H     ;
0EF2:32 59 11              LD   (1159H), A;MULT SECTOR COUNT
0EF5:00 00 00              NOP
0EF8:00 00 00 00           NOP
0EFC:00 00 00 00           NOP
0F00:00 00 00              NOP
0F04:C9                    RET

** PATCH 4
0F05:47         PCH4       LD   B, A       ;PUT CHAR IN CORRECT REGISTER
0F06:C3 D5 0D              JP   PRNTLED

*****************************************************************************
*                                                                           *
*               SUBROUTINE BOOT                                             *
*                                                                           *
*               READS IN FIRST BOOT SECTOR INTO RAM @ 1800H                 *
*                                                                           *
*****************************************************************************

0F09:3E 00      BOOT:      LD   A, 00H
0F0B:CD 73 0B              CALL RESTORE    ;RESET DISK AT HEAD
0F0E:3E 01                 LD   A, 01H
0F10:32 59 11              LD   (1159H), A;SECTOR COUNT=1
0F13:06 01                 LD   B, 01H     ;SECTOR #=1
0F15:0E 00                 LD   C, 00H     ;HEAD #=0
0F17:11 00 00              LD   DE, 0000H  ;CYLINDER #=0
0F1A:CD 90 0B              CALL TSKFILE
0F1D:21 00 18              LD   HL, 1800H  ;START OF MEM SPACE
0F20:CD AA 0B              CALL READSEC    ;READ BOOT SECTOR
0F23:C9                    RET
```

```
0F24:04          INCS/C/H   INC   B          ;INC SECTOR REG
0F25:78                     LD    A, B
0F26:FE 11                  CP    11H        ;MORE THAN 16? (17 SEC/TRACK)
0F28:C0                     RET   NZ         ;NO, SO RETURN
0F29:06 00                  LD    B, 00H     ;SET SECTOR #=0
0F2B:1C                     INC   E          ;INC CYLINDER-LOW
0F2C:7B                     LD    A, E
0F2D:FE 00                  CP    00H
0F2F:CA 49 0F               JP    Z, DE0
0F32:FE EF                  CP    EFH
0F34:C0                     RET   NZ
0F35:7A                     LD    A, D
0F36:FE 02                  CP    02H
0F38:C0                     RET   NZ
0F39:11 00 00               LD    DE, 0000H  ;SET CYINDER#=0, INC HEAD #
0F3C:0C                     INC   C
0F3D:79                     LD    A, C
0F3E:FE 08                  CP    08H        ;MAX HEADS=7 (8 HEADS)
0F40:C0                     RET   NZ
0F41:0E 07                  LD    C, 07      ;KEEP AT MAX
0F43:21 4B 0F               LD    HL, MSG26  ;POINT TO ERROR MESSAGE
0F46:C3 E9 0E               JP    ERRMSG     ;PRINT AND HALT
0F49:14                     INC   D          ;INC CYLINDER - HIGH
0F4A:C9                     RET              ;RET (D CAN'T GO ABOVE 2 !!!)
0F4B:            MSG26      EQU   "ILLEGAL DISK ADDRESS" + *CR*
0F4B:49 4C 4C 45
0F4F:47 41 4C 20
0F53:44 49 53 4B
0F57:20 41 44 44
0F5B:52 45 53 53
0F5F:0D 00
```

I/O PORT ASSIGNMENTS

00	ACE #1 R/W REGISTER
01	ACE #1 INT. ENABLE REG
02	ACE #1 INT. I.D. REG
03	ACE #1 LINE CONTROL REG
04	ACE #1 MODEM CONTROL REG
05	ACE #1 LINE STATUS REG
06	ACE #1 MODEM STATUS REG
07	ACE #1 DIVISOR LATCH REG
00	ACE #2 R/W REGISTER
01	ACE #2 INT. ENABLE REG
02	ACE #2 INT. I.D. REG
03	ACE #2 LINE CONTROL REG
04	ACE #2 MODEM CONTROL REG
05	ACE #2 LINE STATUS REG
06	ACE #2 MODEM STATUS REG
07	ACE #2 DIVISOR LATCH REG
10	PARALLEL PORT A
11	PARALLEL PORT B
12	PARALLEL PORT C
13	PARALLEL CONTROL PORT
14	UNUSED
15	UNUSED
16	UNUSED
17	UNUSED
18	VIDEO CHIP R/W
19	VIDEO RAM R/W
1A	UNUSED
1B	UNUSED
1C	UNUSED
1D	UNUSED
1E	UNUSED
1F	UNUSED
20	DISK R/W
21	DISK ERROR/UNUSED
22	DISK SECTOR COUNT R/W
23	DISK SECTOR NUMBER R/W
24	DISK CYLINDER LOW R/W
25	DISK CYLINDER HIGH R/W
26	DISK SDH R/W
27	DISK STATUS/COMMAND
28	I/O RESET
29	RAM BANK SELECT R/W
2A	~~UNUSED~~ LED DIG → 2 REG 2
2B	~~UNUSED~~ LED VAL
2C	UNUSED DISK REG 2
2D	UNUSED
2E	UNUSED
2F	UNUSED
30	
31	
32	
33	
34	
35	
36	
37	
38	
39	
3A	
3B	
3C	
3D	
3E	
3F	
40	

I/O ⇒ Disk conn.