Micropolis Extension Module Users Manual

19 December 2013

Martin Eberhard

CONTENTS

Micmod

Micropolis Disk Extension Module for Polex

Introduction

Polex is an extension for the Poly-88 monitor that resides in the Poly-88 CPU's ROM socket 1 at address 0400h. It is automatically invoked upon reset or power-on of the Poly-88.

At initialization, Polex checks ROM socket 2 (address 0800) to see if any Polex extensions are installed. Micmod is such an extension. It is automatically invoked during Polex initialization, and relocates itself into high memory at address F000h.

For further information, see the Polex User's Manual.

Memory Requirements

The Micropolis disk controller's default address conflicts with the standard address for the Poly-88 video memory at address F800h. For this reason, Micmod the disk controller to be at a nonstandard address range, addresses FC00h-FFFFh.

When installed, the system memory map looks like this:

Address Range	Occupant
FC00h-FFFFh	Micropolis Disk Controller
F800h-FBFFh	Poly-88 Video Board
F400h-F7FFh	Polex RAM image
F000h-F3FFh	Micmod RAM image
0040h-EFFFh	Available RAM
0000h-003Fh	Interrupt Vectors

MICROPOLIS DISK EXTENSION MODULE COMMANDS

These Micmod commands may be typed at the Polex prompt '>'.

BO (BOOT FROM DISK)

Boot from Micropolis Disk. This jumps to the boot code on the Micropolis disk controller, at address FC00h.

DM <MODEL> (SET DRIVE MODEL)

Sets the Micropolis floppy disk drive model for the other Micropolis Extension Module commands:

<MODEL>	Drive Type	Tracks	Sides
1	Mod I	35	1
2	Mod II	77	1
3	(Mod IV)	77	2
4	Mod IV	77	2
5	Mod V	80	1
6	Mod VI	80	2

FC <START> <END> (FORMAT FOR CP/M)

Formats tracks for CP/M starting at <START> and continuing through <END>. The first 10 data bytes of each sector are written as 00, the remaining 256 sector data bytes are written as E5. If no <END> is specified (or if <END> <= <START>), then just one track is formatted. If no <START> is specified, then the entire disk is formatted.

FM <START> <END> (FORMAT FOR MICROPOLIS)

Format tracks for Micropolis OS starting at <START> and continuing through <END>. All 266 sector data bytes are written as 00. If no <END> is specified (or if <END> <= <START>), then just one track is formatted. If no <START> is specified, then the entire disk is formatted.

LH <HEAD> (LOAD HEAD)

Loads and selects the specified head, and keeps it loaded. This is useful for alignment, because the disk controller board will normally unload the head after about 4 seconds' idle time.

RE <TRACK> (RESTORE)

Restores to track 0, then seeks specified track, if a track was specified. If no track is specified, then this command just restores to track 0. An error will be reported if track 0 could not be found.

RC <SECTOR> <ADDRESS> (READ CP/M SECTOR)

Reads the specified CP/M sector into memory starting at the specified address. The first 10 data bytes of each sector are ignored, except that they contribute to the sector checksum. The subsequent 256 data bytes from each sector are loaded into memory. Errors will be reported for sector not found, incorrect header information, and for checksum errors.

RM <SECTOR> <ADDRESS> (READ MICROPOLIS SECTOR)

Reads the specified Micropolis sector into memory at the specified address. All 266 data bytes from each sector are loaded into memory. Errors will be reported for sector not found, incorrect header information, and for checksum errors.

SK <TRACK> (SEEK TRACK)

Seeks the specified track. If the track is unknown (because track 0 has not yet been detected since this Unit was selected), then a restore to track 0 will occur first.

WC <SECTOR> <ADDRESS> (WRITE CP/M SECTOR)

Writes the specified CP/M sector from memory, starting with the specified address. The first 10 data bytes of each sector are written as 00. The subsequent 256 data bytes of each sector are loaded from memory. Errors will be reported for sector not found.

WM <SECTOR> <ADDRESS> (WRITE MICROPOLIS SECTOR)

Writes the specified Micropolis sector from memory at the specified address. All 266 data bytes of each sector are loaded from memory. Errors will be reported for sector not found.

INTERFACES

MODULE INTERFACE

Polex provides an interface for an extension module in ROM socket 2, at address 0800h.

When Polex detects a ROM at 0800h, it calls this address, allowing the Micropolis Extension Module to initialize itself, and to copy its code into high memory, so that it is available once Polex disables onboard memory.

Upon return from its initialization, Micmod returns the address of its command processor (F000h) in HL. Polex will call this address after each command line from the user, before searching its own command list.

On Micmod Command Processor entry at address F000h, DE = the address of an input line, with the first byte being a potential first command character (i.e. leading zeros have already been skipped). The input line is terminated with a Poly-88 graphics cursor, 0FFh.

Micmod returns with the Z flag set if it does not recognize the command from the user, allowing Polex to search its own command list.

If Micmod recognizes the command, it will execute the command, and then return with the Z flag cleared.

For some errors, Micmod returns by jumping to PXWARM, where the stack will get repaired.

POLEX ENTRY POINTS

Micmod calls the following Polex entry points:

PXCSTA (F403) Get Console Status

Call this address to get the console keyboard status.
On Return:
A=0 and Z flag set if no keyboard character waiting
A=FFh and Z flag cleared if a keyboard character is waiting.
All other registers are preserved

PXCIN (F406) Get Console Input

Call this address to get one keyboard character.
Waits for a keyboard character, and returns it in A. The Z flag is always cleared. All other registers are preserved.

PXCOUT (F409) Console Output

Call this address to send one character to the console video screen.

On Entry:
C=character to print on the video screen
On Return:
A=C
All other registers are preserved.

PXIHEX (F41B) Get hex input from console

Call this address to get one 16-bit hexadecimal value from the user input buffer.

Leading blanks will be skipped.

On Entry:
DE=address of text string to parse, which must be terminated with space or FFh
On Return:
PSW trashed
BC preserved
DE=first address after the last hex character
HL=16-bit hex value
Z flag set and carry cleared only if no value found
Abort to PXWARM if bad hex found
All other registers are preserved.

PXOHEX (F41E) Print A in Hexadecimal on the Console

Call this address to print the value in A in hexadecimal on the console.

All registers are preserved.

PXWARM (F421) Warm-start re-entry

This entry point restores the stack and sets the baud rate to its default (9600 baud), before returning to the command processor.

PXCIN2 (F424) Get Console Character, if Available

Call this address to get one console keyboard character, only if one is available.
On Return:
A=0 and Z flag is set only if no keyboard character waiting.
A=keyboard character, if one was waiting
The ‘ALT MODE’ key (which produces an ASCII ‘}‘ character) aborts to Polex’s command processor.
All other registers are preserved.

MICMOD SOURCE CODE LISTING

```
;============================================================
; MICMOD MICROPOLIS MODULE FOR THE POLEX MONITOR,
; FOR THE POLYMORPHIC SYSTEMS POLY-88
;
; INSTALLS IN THE 3RD CPU BOARD ROM SOCKET, TO ADD SUPPORT FOR
; THE MICROPOLIS FLOPPY SUBSYSTEM
;
; THE POLY-88 MONITOR IS STILL ACTIVE, PERFORMING THE
; INTERRUPT-DRIVEN INPUT ROUTINES, THE VIDEO OUTPUT, AS WELL
; AS THE CTRL-Z-ACTIVATED FRONT PANEL MODE.
;
; MICROPOLIS MODULE COMMANDS (ALL VALUES ARE IN HEX):
;
;
;   BO                    BOOT FROM MICROPOLIS DISK
;
;   DM <MODEL>            SET MICROPOLIS DISK MODEL:
;                           1= MOD  I: 35 TRACKS, 1-SIDED
;                           2= MOD II: 77 TRACKS, 1-SIDED
;                           3= (MOD II: 77 TRACKS, 1-SIDED)
;                           4= MOD IV: 77 TRACKS, 2-SIDED
;                           5= MOD  V: 80 TRACKS, 1-SIDED
;                           6= MOD VI: 80 TRACKS, 2 SIDED
;
;   FC <TRK1> <TRK2>      CP/M-FORMAT TRACKS FROM TRK1 THROUGH
;                           TRK2. DEFAULT IS ENTIRE DISK
;
;   FM <TRK1> <TRK2>      MICROPOLIS-FORMAT TRACKS FROM TRK1
;                           THROUGH TRK2. DEFAULT IS ENTIRE DISK
;
;   LH <HEAD>             LOAD HEAD AND KEEP IT LOADED (FOR
;                           ALIGNMENT.) TYPE ANY KEY TO ABORT
;
;   RE <TRACK>            RESTORE, THEN OPTIONALLY SEEK TRACK
;
;   RC <SECTOR> <ADDR>    READ CP/M (256-BYTE) SECTOR INTO MEMORY
;                           AT ADDR
;
;   RM <SECTOR> <ADDR>    READ FULL (266-BYTE) SECTOR INTO
;                           MEMORY AT ADDR
;
;   SK <TRACK>            SEEK TRACK. WILL RESTORE FIRST IF THE
;                           TRACK IS UNKNOWN.
;
;   SU <UNIT>             SELECT UNIT FOR ALL OPERATIONS
;
;   WC <SECTOR> <ADDR>    WRITE CP/M (256-BYTE) SECTOR FROM
;                           MEMORY AT ADDR
;
;   WM <SECTOR> <ADDR>    WRITE FULL (266-BYTE) SECTOR FROM
;                           MEMORY AT ADDR
;
;       FOR RM AND WM ON 2-SIDED DRIVES, THE LSB OF SS (0-F)
;       SPECIFIES THE SECTOR, AND THE THE MSB (0 OR 1)
;       SPECIFIES THE SIDE.
;
;   FORMATTED TO ASSEMBLE WITH DIGITAL RESEARCH'S ASM.
;============================================================
; NOTES
;
; SOFTWARE MUST READ OR WRITE 270 BYTES FOR EACH SECTOR:
```

```
                ;  BYTE   FUNCTION
                ;   0   SYNC BYTE = 0FFH
                ;   1   HEADER: TRACK NUMBER
                ;   2   HEADER: SECTOR NUMBER
                ; 3-268 DATA (266 BYTES)
                ;  269  CHECKSUM
                ;
                ; MICROPOLIS DISK SECTOR HEADERS ARE 2 BYTES IN LENGTH. THE 1ST
                ; BYTE IS THE TRACK NUMBER. THE SECOND BYTE IS THE SECTOR
                ; NUMBER, BETWEEN 00 AND 0FH.  MICROPOLIS APPARENTLY DOES NOT
                ; ENCODE THE SIDE INTO THE SECTOR HEADERS OF 2-SIDED DISKS.
                ;
                ; CHECKSUMS ARE COMPUTED WITH THE CARRY BIT - I.E. THE ADC
                ; INSTRUCTION IS USED, RATHER THAN THE ADD INSTRUCTION. THE
                ; CHECKSUM IS THE SUM (WITH CARRY) OF BYTES 1 THROUGH 268.
                ;
                ; NOTE THAT CP/M ON MICROPOLIS DISKS READS AND WRITES ONLY 256
                ; DATA BYTES PER SECTOR, PADDING THE BEGINNING OF SECTORS WITH
                ; 10 BYTES OF 00.
                ;===============================================================
                ; REVISION HISTORY
                ; VERS. 1.00 BY M. EBERHARD 23 NOV 2013
                ;   CREATED
                ;
                ; VERS. 1.01 BY M. EBERHARD 3 DEC 2013
                ;   ADD RC & WC COMMANDS, TO READ/WRITE CP/M (256-BYTE) SECTORS
                ;
                ; VERS. 2.00 BY M. EBERHARD 6 DEC 2013
                ;   RETURN RAM ADDRESS IN HL INSTEAD OF DE, GET COMMAND ADDRESS
                ;   IN HL INSTEAD OF DE.
                ;   SUPPORT FORMATTING SELECTED TRACKS
                ;   ADD FC COMMAND TO FORMAT FOR CP/M
                ;
                ; POSSIBLE FUTURE ENHANCEMENTS:
                ;   SELECTED SIDE(S) FOR FORMAT COMMAND
                ;   GET TRACK ID CMD
                ;   MEASURE SPINDLE SPEED & TEST FOR 16-SECTOR DISK
                ;   DECODE STATUS AND/OR STAT COMMAND
                ;===============================================================

                ;*******
                ; ASCII
                ;*******
000D =          CR      EQU     0DH

                ;****************
                ; PROGRAM EQUATES
                ;****************
073A =          CPUKHZ  EQU     1850    ;CPU CLOCK IN KHZ
0001 =          DEFMOD  EQU     1       ;DEFAULT DISK DRIVE MODEL
0000 =          MFDATA  EQU     00H     ;DATA FOR MICROPOLIS-FORMAT SECTORS
                                        ;AND ALSO 1ST 10 BYTES OF CP/M SECS
00E5 =          CFDATA  EQU     0E5H    ;DATA FOR CP/M-FORMAT SECTORS

                ;*******************************************
                ; MEMORY LOCATION BEFORE & AFTER CODE IS MOVED
                ;*******************************************
0800 =          ROMLOC  EQU     0800H   ;MICMOD ROM ADDRESS
F000 =          RAMLOC  EQU     0F000H  ;LOCATION AFTER MOVED
F400 =          PBASE   EQU     0F400H  ;POLEX RAM BASE

                ;***********************************
                ; POLEX ENTRY POINTS
```

```
                ;**********************************
F400 =          PXINIT  EQU     PBASE           ;RESTART POLEX
F403 =          PXCSTA  EQU     PBASE+03H       ;CONSOLE STATUS
F406 =          PXCIN   EQU     PBASE+06H       ;CONSOLE INTO A
F409 =          PXCOUT  EQU     PBASE+09H       ;CONSOLE OUT C. A=C ON RET
F40C =          PXSSTA  EQU     PBASE+0CH       ;SERIAL PORT STATUS
F40F =          PXSIN   EQU     PBASE+0FH       ;SERIAL PORT IN
F412 =          PXSOUT  EQU     PBASE+12H       ;SERIAL PORT OUT
F415 =          PXSOST  EQU     PBASE+15H       ;SERIAL PORT OUTPUT STATUS
F418 =          PXBAUD  EQU     PBASE+18H       ;SET SERIAL PORT BAUD RATE
F41B =          PXIHEX  EQU     PBASE+1BH       ;SCAN PAST BLANKS, GET HEX VAL
                                                ;HL=VALUE
                                                ;DE ADVANCED PAST CHR
                                                ;Z SET, CARRY CLEAR IF VALUE
                                                ;CARRY SET IF NO VALUE FOUND
F41E =          PXOHEX  EQU     PBASE+1EH       ;PRINT A AS HEX ON THE CONSOLE
F421 =          PXWARM  EQU     PBASE+21H       ;WARM-START RE-ENTRY
F424 =          PXCIN2  EQU     PBASE+24H       ;TEST KBD STATUS, RZ IF NO CHR
                                                ;IF CHR, ABORT IF CABKEY
                                                ;ELSE RET WITH CHR IN A, NZ
F427 =          PXGLIN  EQU     PBASE+27H       ;GET A LINE OF USER INPUT

                ;***********************************************
                ; MICROPOLIS DISK CONTROLLER MEMORY-MAPPED PORTS
                ; NOTE: THE "STANDARD" ADDRESS FOR THE MICROPOLIS
                ; CONTROLLER IS F800, WHICH CONFLICTS WITH THE
                ; STANDARD POLY-88 VIDEO MEMORY ADDRESS. STDMEM
                ; SETS THE VIDEO MEMORY TO ITS POLY-88 STANDARD.
                ;***********************************************
FC00 =          MBASE   EQU     0FC00H  ;NON-STANDARD CONTROLLER ADDR
FC00 =          MBOOT   EQU     MBASE           ;BOOT CODE ENTRY
FE00 =          MCMD    EQU     MBASE+200H      ;COMMAND WRITE-REGISTER
                                                ;..ALSO AT MBASE+201H
0020 =          MDHSEL    EQU     20H   ;SELECT HEAD/UNIT
                                        ;BITS 1:0 SELECT THE UNIT
                                        ;BIT 4 SELECTS THE HEAD
0040 =          MINTC     EQU     40H   ;INTERRUPT CONTROL
                                        ;BIT 0=1 TO ENABLE, =0 DISABLE
0060 =          MSTEP     EQU     60H   ;STEP TRACK (30 mS PER STEP)
                                        ;BIT 0=0 STEPS OUT, 1 STEPS IN
0080 =          MWREN     EQU     80H   ;WRITE ENABLE
00A0 =          MRESET    EQU     0A0H  ;RESET CONTROLLER

FE00 =          MSECTR  EQU     MBASE+200H      ;SECTOR READ-REGISTER
000F =          MSMASK    EQU     0FH   ;SECTOR BITS MASK
0010 =          MJW10     EQU     10H   ;JUMPER W10 (RESERVED)
0020 =          MJW9      EQU     20H   ;JUMPER W9 (CPU SPEED)
                                        ;0=4 MHZ, 1=2 MHZ
0040 =          MSINT     EQU     40H   ;SECTOR INTERRUPT
0080 =          MSFLAG    EQU     80H   ;SECTOR VALID FLAG

FE01 =          MSTAT   EQU     MBASE+201H      ;STATUS READ-REGISTER
0007 =          MUMASK    EQU     07H   ;UNIT & SELECTED MASK
0004 =          MSLTD     EQU     04H   ;UNIT IS SELECTED
0008 =          MTK0      EQU     08H   ;TRACK 0 DETECTED
0010 =          MWPT      EQU     10H   ;WRITE PROTECT DETECTED
0020 =          MRDY      EQU     20H   ;DRIVE READY
0040 =          MPINTE    EQU     40H   ;S-100 PINTE STATUS
0080 =          MTF       EQU     80H   ;TRANSFER FLAG

FE02 =          MDATA   EQU     MBASE+202H      ;READ/WRITE DATA REGISTER

010A =          MBPS    EQU     266     ;DATA BYTES PER SECTOR - DOES NOT
```

```
                                ;..INCLUDE HEADER OR CHECKSUM

0003 =          MAXUNT  EQU     3       ;MAX UNIT NUMBER
000F =          MAXSEC  EQU     15      ;MAX SECTOR NUMBER
00FF =          SYNCHR  EQU     0FFH    ;SECTOR SYNCH CHARACTER

0006 =          MAXMOD  EQU     6       ;MAX SUPPORTED DRIVE MODEL NUMBER

                ;=============================================================
                ; START OF ROM CODE
                ; GETS CALLED BY POLEX DURING INITIALIZATION
                ; INSTALLS MODULE CODE IN MEMORY AT F000
                ; ON EXIT:
                ;   HL=ADDRESS OF COMMAND PROCESSOR IN RAM
                ;=============================================================
0800                    ORG     ROMLOC

                ;=========================
                ; MICMOD ROM ENTRY POINT =
                ;=========================

                ;MOVE CODE INTO RAM, BACKWARDS, SO HL IS RIGHT AT THE END

0800 01D203     MDINIT: LXI     B,RAMEND-RAMSRC         ;BYTE COUNT
0803 11E80B             LXI     D,RAMEND                ;END OF RAM SOURCE
0806 21D2F3             LXI     H,RAMEND-RAMSRC+RAMLOC  ;END OF DEST

0809 1B         MOVER:  DCX     D
080A 2B                 DCX     H

080B 1A                 LDAX    D
080C 77                 MOV     M,A

080D 0B                 DCX     B               ;BUMP COUNT
080E 78                 MOV     A,B
080F B1                 ORA     C               ;DONE?
0810 C20908             JNZ     MOVER

                ;PRINT BANNER AND RETURN TO POLEX

0813 C326F0             JMP     BANNER+ROF      ;MUST BE IN RAM CODE

                ;=============================================================
                ; RAM CODE, GETS MOVED INTO RAM AT RAMLOC (F000)
                ;=============================================================
                RAMSRC:
E7EA =          ROF     EQU     RAMLOC-RAMSRC   ;RAM OFFSET

                ;=========================
                ; MICMOD RAM ENTRY POINT =
                ;=========================

                ;*******************************************
                ; MICMOD COMMAND PROCESSOR
                ;
                ; SCAN FOR MODULE COMMANDS, AND EXECUTE THEM
                ;
                ; ON ENTRY:
                ;   DE POINTS TO POTENTIAL COMMAND
                ; ON EXIT:
                ;   Z CLEAR MEANS SUCCESFUL COMMAND EXECUTION
                ;   Z SET MEANS NO COMMAND FOUND
                ;   SOME ERRORS EXIT DIRECTLY TO PXWARM
```

```
                ;***********************************************
0816 1A         MDCMD:  LDAX    D               ;1ST CHR
0817 47                 MOV     B,A             ;1ST CHR INTO B
0818 13                 INX     D
0819 1A                 LDAX    D               ;2ND CHR
081A 4F                 MOV     C,A             ;2ND CHR INTO C
081B 1B                 DCX     D               ;FIX DE IN CASE NO MATCH

081C 219CF3             LXI     H,COMTAB+ROF-1  ;POINT TO COMMAND TABLE

                ;SEARCH THROUGH TABLE AT HL FOR A 2-CHR MATCH OF BC

081F 23         NCOM:   INX     H               ;GO TO NEXT ENTRY
0820 7E                 MOV     A,M             ;1ST TABLE CHR
0821 B7                 ORA     A               ;END OF TABLE?
0822 C8                 RZ                      ;Y: NOT A MODULE CMD

0823 23                 INX     H
0824 B8                 CMP     B               ;MATCH?
0825 7E                 MOV     A,M             ;2ND TABLE CHR
0826 23                 INX     H
0827 C209F0             JNZ     NCOM+ROF        ;N: KEEP LOOKING
082A 91                 SUB     C               ;MATCH?
082B C209F0             JNZ     NCOM+ROF        ;N: KEEP LOOKING

                ;GOT A MATCH. COMPUTE ROUTINE ADDRESS AND PUT IT ON STACK
                ;A=0
                ;(HL)= COMMAND ADDRESS OFFSET
                ;DE=ADDRESS OF START OF 2-LETTER COMMAND

082E 6E                 MOV     L,M             ;GET ADDRESS OFFSET
082F 67                 MOV     H,A             ;H=A=0

0830 0140F0             LXI     B,CMDBAS+ROF    ;BASE OF COMMAND ROUTINES
0833 09                 DAD     B               ;HL=ADDRESS OF ROUTINE

0834 E5                 PUSH    H               ;GO ADDRESS ON STACK

0835 13                 INX     D               ;SKIP PAST 2-LETTER COMMAND
0836 13                 INX     D

                ;GET FOLLOWING PARAMETER (IF ANY) AND PUT IT IN HL AND A.
                ;SET Z FLAG IF NO PARAMETER PRESENT

0837 CD1BF4             CALL    PXIHEX          ;GET 1ST PARAM, CY SET IF NONE
083A 7D                 MOV     A,L             ;POTENTIAL LOW BYTE IN A
083B C9                 RET                     ;GO TO COMMAND ROUTINE:
                                                ;CY SET IF NO PARAMETERS
                                                ;HL=1ST PARAMETER
                                                ;A=L
                                                ;DE POINTS TO 1ST CHR
                                                ;..AFTER 1ST PARAM

                ;***************************************************
                ;INIT BANNER ROUTINE - MUST BE IN RAM TO USE CLPRNT
                ;***************************************************
083C CD88F3     BANNER: CALL    CILPRT+ROF
083F 4D6963726F         DB      'Micropolis Module 2.0','0'+80H
0855 C9                 RET

                ;=========================
                ; BASE ADDRESS FOR COMMANDS
                ;=========================
```

```
0856 =          CMDBAS  EQU     $

                ;***COMMAND ROUTINE*******************
                ; BO (BOOT FROM MICROPOLIS DRIVE)
                ;*************************************
0856 C300FC     GOMICR: JMP     MBOOT

                ;***COMMAND ROUTINE*******************
                ; DM (SET MICROPOLIS MODEL)
                ; ON ENTRY:
                ;   A=VALUE (1-6)
                ;*************************************
0859 3D         SETMOD: DCR     A
085A FE06               CPI     MAXMOD          ;BOUNDS CHECK
085C D21CF3             JNC     ERROR+ROF
                                                ;Z IS CLEARED

085F 5F                 MOV     E,A             ;COMPUTE TABLE ENTRY
0860 1600               MVI     D,0
0862 21C2F3             LXI     H,MODTAB+ROF

0865 19                 DAD     D
0866 19                 DAD     D               ;2 BYTES/ENTRY

0867 7E                 MOV     A,M             ;GET MAXTRK
0868 23                 INX     H
0869 66                 MOV     H,M             ;GET SIDED-NESS
086A 6F                 MOV     L,A
086B 22D0F3             SHLD    MAXTRK+ROF      ;SAVE BOTH

086E C9                 RET                     ;RET WITH Z CLEAR

                ;***COMMAND ROUTINE*******************
                ; SU (SET UNIT)
                ; ALSO SET TRACK TO 0FFH, MEANING WE
                ; DON'T KNOW WHERE THE HEAD IS.
                ;*************************************
086F FE04       STUNIT: CPI     MAXUNT+1        ;BOUNDS CHECK
0871 D21CF3             JNC     ERROR+ROF       ;Z IS CLEARED
0874 32CEF3             STA     UNIT+ROF

0877 3EFF               MVI     A,0FFH          ;WE DON'T KNOW WHERE
0879 32CFF3             STA     TRACK+ROF       ;..WE ARE ANY MORE

087C C9                 RET                     ;RET WITH Z CLEAR

                ;***COMMAND ROUTINE***********************************
                ; LH (LOAD HEAD AND KEEP IT LOADED)
                ; THE MICROPOLIS CONTROLLER HAS A TIMER THAT UNLOADS
                ; THE HEAD(S) AFTER ABOUT 4 SECONDS OF INACTIVITY.
                ; THIS TIMER IS RESET WITH EACH READ OF THE STATUS
                ; REGISTER. THIS ROUTINE REDS THE STATUS REGISTER
                ; OVER AND OVER UNTIL TOLD TO STOP BY THE USER.
                ; ON ENTRY:
                ;   A=HEAD
                ;*************************************************
087D FE02       DHLOAD: CPI     2               ;HEAD MUST BE 0 OR 1
087F D21CF3             JNC     ERROR+ROF

0882 07                 RLC
0883 07                 RLC
0884 07                 RLC
0885 07                 RLC                     ;0 OR 10H
```

```
0886 47                 MOV     B,A             ;HEAD IN B FOR SELECT
0887 CD74F2             CALL    SELECT+ROF

088A CD88F3             CALL    CILPRT+ROF
088D 5120746F20         DB      'Q to qui','t'+80H

0896 7E         DHLUP:  MOV     A,M             ;..TO KEEP HEAD LOADED

0897 CD03F4             CALL    PXCSTA          ;ANYTHING FROM KBD?
089A CA80F0             JZ      DHLUP+ROF       ;N: KEEP HEAD LOADED

089D C306F4             JMP     PXCIN           ;GET CHR, RET W/ Z CLEAR

                ;***COMMAND ROUTINE*********************
                ; SK (SEEK TRACK)
                ;   WILL RESTORE IF LOST
                ; ON ENTRY:
                ;   L=TARGET TRACK
                ;************************************
08A0 3ACFF3     DSEEK:  LDA     TRACK+ROF       ;DO WE KNOW WHERE WE ARE?
08A3 57                 MOV     D,A             ;D=CURRENT TRACK
08A4 3C                 INR     A               ;FF MEANS NO
08A5 CAAAF0             JZ      DRESTR+ROF      ;N: RESTORE-SEEK INSTEAD

08A8 CD69F2             CALL    SKSTUP+ROF      ;CHECK TRACK #, SELECT
                                                ;C=MSTEP+1 (STEP INWARD)
                                                ;E=TARGET TRACK
                                                ;HL=MCMD+1

08AB 7B                 MOV     A,E             ;HL=TRACK IN OR OUT?

08AC 92                 SUB     D               ;DESIRED TK - CURRENT TK
08AD CAA8F0             JZ      SKDONE+ROF      ;ALREADY THERE: RET W/ Z

08B0 D2A0F0             JNC     DSK1+ROF        ;C MEANS DESIRED<CURRENT
                                                ;NC MEANS CURRENT<DESIRED
08B3 0D                 DCR     C               ;TOWARD SMALLER TRACK #
08B4 2F                 CMA                     ;MAKE IT A POSITIVE NUMBER
08B5 3C                 INR     A

08B6 47         DSK1:   MOV     B,A             ;NUMBER OF STEPS

08B7 CD59F2     GOSTEP: CALL    STEP+ROF        ;B=0 ON EXIT

08BA 7B                 MOV     A,E             ;REMEMBER TRACK
08BB 32CFF3             STA     TRACK+ROF

08BE 3C         SKDONE: INR     A               ;CLEAR Z FOR RET
08BF C9                 RET

                ;***COMMAND ROUTINE*********************
                ; RE (RESTORE TO TRACK)
                ; IGNORES ALL BUT THE LAST 2 HEX DIGITS
                ; ON ENTRY:
                ;   L=TARGET TRACK
                ; ON EXIT:
                ;   HL=MCMD+1
                ;   D PRESERVED
                ;************************************
08C0 CD69F2     DRESTR: CALL    SKSTUP+ROF      ;CHECK TRACK #, SELECT
                                                ;C=MSTEP+1
                                                ;E=TARGET TRACK
                                                ;HL=MCMD+1
```

```
                ; STEP AWAY FROM THE TRACK ZERO SENSOR

08C3 0602               MVI     B,2             ;2 STEPS SHOULD DO IT
08C5 CD59F2             CALL    STEP+ROF        ;IN CASE BEYOND TK 0

                ; STEP OUTWARD TO TRACK 0

08C8 0D                 DCR     C               ;C=MSTEP: STEP OUT
08C9 3AD0F3             LDA     MAXTRK+ROF
08CC C606               ADI     6               ;A FEW EXTRA
08CE 47                 MOV     B,A
08CF CD59F2             CALL    STEP+ROF
08D2 CADAF2             JZ      RSTERR+ROF      ;Z:CAN'T FIND TRACK 0

                ;STEP INWARD TO THE TARGET TRACK

08D5 43                 MOV     B,E             ;B=TARGET TRACK
08D6 0C                 INR     C               ;C=MSTEP+1: STEP
08D7 C3A1F0             JMP     GOSTEP+ROF      ;.. INWARD B STEPS

                ;***COMMAND ROUTINE******************
                ; RC (READ CP/M-STYLE 256-BYTE SECTOR)
                ;  THE FIRST 10 BYTES OF SECTOR DATA
                ;  ARE IGNORED, EXCEPT THAT THEY ARE
                ;  INCLUDED IN THE CHECKSUM.
                ; ON ENTRY:
                ;   A<3:0>=SECTOR
                ;   A<4>  = HEAD
                ;************************************
08DA CDD3F1     DRD256: CALL    RDSTUP+ROF      ;READ SETUP
                                                ;  HL=DATA REG
                                                ;  DE=DEST
                                                ;  A=CHECKSUM SO FAR
                                                ;  CARRY VALID FROM CKSUM
                                                ;  B=MBPS-256=10
                                                ;  HEADER ALREADY READ

                ;READ AND CHUCK THE FIRST 10 DATA BYTES
                ;  HL=DATA REG
                ;  DE=DEST
                ;  A=CHECKSUM SO FAR
                ;  B=MBPS-256
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

08DD 8E         DRLP0:  ADC     M               ;(4)JUST DO CHECKSUM W/CARRY
08DE 05                 DCR     B               ;(4)NEXT
08DF C2C7F0             JNZ     DRLP0+ROF       ;(10)42 CYCLES READ-READ

                ;GO READ AND STORE THE REMAINING 256 BYTES

08E2 C3DBF0             JMP     RD256+ROF

                ;***COMMAND ROUTINE*************
                ; RM (READ FULL 266-BYTE SECTOR)
                ; ON ENTRY:
                ;   A<3:0>=SECTOR
                ;   A<4>  = HEAD
                ;*******************************
08E5 CDD3F1     DRD266: CALL    RDSTUP+ROF      ;READ SETUP
                                                ;  HL=DATA REG
                                                ;  DE=DEST
                                                ;  A=CHECKSUM SO FAR
```

```
                                                ;  CARRY VALID FROM CKSUM
                                                ;  B=MBPS-256=10
                                                ;  HEADER ALREADY READ

                ;READ THE FIRST 10 DATA BYTES
                ;  HL=DATA REG
                ;  DE=DEST
                ;  A=CHECKSUM SO FAR
                ;  B=MBPS-256
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

08E8 4F         DRLP1:  MOV     C,A             ;(5)SAVE CHECKSUM SO FAR
08E9 7E                 MOV     A,M             ;(7)GET DATA BYTE
08EA 12                 STAX    D               ;(7)SAVE IT
08EB 89                 ADC     C               ;(4)DO CHECKSUM W/CARRY
08EC 13                 INX     D               ;(5)NEXT
08ED 05                 DCR     B               ;(4)NEXT
08EE C2D2F0             JNZ     DRLP1+ROF       ;(10)42 CYCLES READ-READ


                ;READ THE LAST 256 DATA BYTES
                ;  HL=DATA REG
                ;  DE=DEST
                ;  A=CHECKSUM SO FAR
                ;  B=0
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

08F1 4F         RD256:  MOV     C,A             ;(5)SAVE CHECKSUM SO FAR
08F2 7E                 MOV     A,M             ;(7)GET DATA BYTE
08F3 12                 STAX    D               ;(7)SAVE IT
08F4 13                 INX     D               ;(5)NEXT
08F5 89                 ADC     C               ;(4)DO CHECKSUM W/CARRY
08F6 05                 DCR     B               ;(4)NEXT
08F7 C2DBF0             JNZ     RD256+ROF       ;(10)42 CYCLES READ-READ

                ;READ AND COMPARE THE CHECKSUM TO A

08FA BE                 CMP     M               ;GET & COMPARE CHECKSUM
08FB FB                 EI                      ;INTS OKAY FINALLY
08FC C423F3             CNZ     CSMERR+ROF      ;BAD CHECKSUM:ERROR

                ;GO SEE IF THE SECTOR AND TRACK MATCH WHAT WE EXPECTED

08FF C348F3             JMP     CHKHDR+ROF      ;CHECK & PRINT HEADER INFO
                                                ;AND RETURN TO POLEX

                ;***COMMAND ROUTINE*********************
                ; WC (WRITE CP/M-STYLE 256-BYTE SECTOR)
                ;   WRITES 0'S TO THE 1ST 10 SECTOR BYTES
                ; ON ENTRY:
                ;   DRIVE IS SELECTED (OR WE WILL ERROR)
                ;   A<3:0>=SECTOR
                ;   A<4>  = HEAD
                ;***************************************
0902 CDE9F1     DWR256: CALL    WRSTUP+ROF      ;A=CHECKSUM SO FAR
                                                ;CARRY VALID FROM CKSUM
                                                ;B=MBPS-256=10
                                                ;DE=SOURCE ADDRESS
                                                ;HL=DATA REGISTER ADDRESS
                                                ;HEADER ALREADY WRITTEN

                ;WRITE ZEROS FOR THE 1ST 10 BYTES OF SECTOR DATA
                ;  HL=DATA REG
```

```
                ;  DE=SOURCE
                ;  A=CHECKSUM SO FAR
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

0905 3600       DWLP0:  MVI     M,0             ;(11)
0907 05                 DCR     B               ;(5)
0908 C2EFF0             JNZ     DWLP0+ROF       ;(10)

090B C304F1             JMP     WR256+ROF       ;(10)

                ;***COMMAND ROUTINE*********************
                ; WM (WRITE FULL 266-BYTE SECTOR)
                ; ON ENTRY:
                ;   DRIVE IS SELECTED (OR WE WILL ERROR)
                ;   A<3:0>=SECTOR
                ;   A<4>  = HEAD
                ;***************************************
090E CDE9F1     DWR266: CALL    WRSTUP+ROF      ;A=CHECKSUM SO FAR
                                                ;CARRY VALID FROM CKSUM
                                                ;B=MBPS-256=10
                                                ;DE=SOURCE ADDRESS
                                                ;HL=DATA REGISTER ADDRESS
                                                ;HEADER ALREADY WRITTEN

                ;WRITE THE 1ST 10 BYTES OF SECTOR DATA
                ;  HL=DATA REG
                ;  DE=SOURCE
                ;  A=CHECKSUM SO FAR
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

0911 4F         DWLP1:  MOV     C,A             ;(4)
0912 1A                 LDAX    D               ;(7)
0913 13                 INX     D               ;(5)
0914 77                 MOV     M,A             ;(7)
0915 89                 ADC     C               ;(4)CHECKSUM

0916 05                 DCR     B               ;(5)
0917 C2FBF0             JNZ     DWLP1+ROF       ;(10)

                ;WRITE THE LAST 256 BYTES OF SECTOR DATA
                ;  HL=DATA REG
                ;  DE=SOURCE
                ;  A=CHECKSUM SO FAR
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

091A 4F         WR256:  MOV     C,A             ;(4)
091B 1A                 LDAX    D               ;(7)
091C 77                 MOV     M,A             ;(7)
091D 13                 INX     D               ;(5)
091E 89                 ADC     C               ;(4)CHECKSUM

091F 05                 DCR     B               ;(5)
0920 C204F1             JNZ     WR256+ROF       ;(10)

                ;WRITE THE CHECKSUM

0923 77                 MOV     M,A

                ;WAIT FOR TRANSFER FLAG TO CLEAR, SO THAT THE CONTROLLER
                ;MAY FINISH WRITING THE POSTAMBLE (NOT NECESSARY HERE,
                ;BECAUSE WE WRITE ONE SECTOR AT A TIME, MANUALLY.)

                ;       XRA     A
```

```
                ;WNTF:  ORA     M               ;MSB IS TR FLAG
                ;       JM      WNTF+ROF

                ;       EI                      ;INTS FIXED AT MAIN

0924 B4                 ORA     H               ;CLEAR Z
0925 C9                 RET

                ;***COMMAND ROUTINE*********************************
                ; FC (FORMAT CP/M DISK)
                ;  WRITES CORRECT SECTOR HEADERS AND CHECKSUMS,
                ;  FILLS FIRST 10 BYTES OF DATA FIELDS WITH MFDATA (00),
                ;  AND THE REMAINING 256 BYTES WITH CFDATA (E5H).
                ; ON ENTRY:
                ;   CARRY SET IF NO PARAMETERS
                ;   L=1ST TRACK TO FORMAT
                ;  (HL)=LAST TRACK TO FORMAT
                ;*****************************************************
0926 3EE5       CFORMT: MVI     A,CFDATA        ;SECTOR DATA
0928 01                 DB      01H             ;'LXI B' OPCODE SKIPS 2
                                                ;..BYTES TO GO TO FORMAT

                ;***COMMAND ROUTINE*****************************
                ; FM (FORMAT MICROPOLIS DISK)
                ;  WRITES CORRECT SECTOR HEADERS AND CHECKSUMS,
                ;  FILLS ALL DATA FIELDS WITH MFDATA (00)
                ; ON ENTRY:
                ;   CARRY SET IF NO PARAMETERS
                ;   L=1ST TRACK TO FORMAT
                ;  (HL)=LAST TRACK TO FORMAT
                ;*****************************************
0929 3E00       MFORMT: MVI     A,MFDATA        ;SECTOR DATA,
                                                ;LEAVE CARRY ALONE

                ; FALL INTO FORMAT

                ;*****************************************
                ; GENERAL FORMAT ROUTINE
                ; ON ENTRY:
                ;   CARRY SET IF NO PARAMETERS
                ;   L=1ST TRACK TO FORMAT
                ;  (HL)=LAST TRACK TO FORMAT
                ;   A=DATA FOR LAST 256 BYTES OF SECTORS
                ;   THE FIRST 10 BYTES ALWAYS GET MFDATA (00).
                ;*****************************************
092B 32D3F3     FORMAT: STA     FORDAT+ROF      ;GET SECTOR DATA

092E 0600               MVI     B,0             ;DEFAULT INITIAL TRACK
0930 3AD0F3             LDA     MAXTRK+ROF      ;DEFAULT FINAL TRACK
0933 3C                 INR     A               ;..+1
0934 4F                 MOV     C,A
0935 DA2DF1             JC      FORM1+ROF       ;DEFAULTS IF NO PARAMS

0938 45                 MOV     B,L             ;GOT START TRACK
                                                ;DRESTR WILL VALIDATE B

0939 CD1BF4             CALL    PXIHEX          ;GET END TRACK
093C 7D                 MOV     A,L
093D B9                 CMP     C               ;VALID FINAL TRACK?
093E D21CF3             JNC     ERROR+ROF

0941 2C                 INR     L               ;LAST TRACK+1
```

```
0942 4D                 MOV     C,L

0943 51         FORM1:  MOV     D,C             ;D GETS FINAL TRACK

                ;B=START TRACK, D=FINAL TRACK+1

0944 CD88F3             CALL    CILPRT+ROF      ;ARE YOU SURE?
0947 466F726D61         DB      'Format (Y/N)','?'+80H
0954 CD06F4             CALL    PXCIN
0957 FE59               CPI     'Y'
0959 C0                 RNZ                     ;NOT Y: DONE, Z CLEARED

095A CD8CF3             CALL    ILPRNT+ROF
095D 590D               DB      'Y',CR          ;FAKE ECHO
095F 466F726D61         DB      'Formattin','g'+80h

                ; RESTORE TO TRACK B, SELECT SIDE 0

0969 68                 MOV     L,B             ;RESTORE TO TRACK B
096A CDAAF0             CALL    DRESTR+ROF      ;SELECT & RESTORE
                                                ;HL=MCMD+1

                ;ABORT IF DISK IS WRITE-PROTECTED

096D 7E                 MOV     A,M             ;READ STATUS REG
096E E610               ANI     MWPT
0970 C2C7F2             JNZ     WPTERR+ROF

                ;LOOP TO FORMAT ALL REQUESTED TRACKS. IF THE REQUESTED
                ;LAST TRACK IS LESS THAN THE REQUESTED FIRST TRACK, OR IF
                ;NO LAST TRACK WAS SPECIFIED, THEN JUST FORMAT THE FIRST
                ;REQUESTED TRACK.
                ; A=0
                ; D=FINAL TRACK
                ; HEAD IS ALREADY ON REQUESTED FIRST TRACK

0973 47                 MOV     B,A             ;0 STEPS 1ST TIME

0974 0E61       FTLOOP: MVI     C,MSTEP+1       ;STEP INWARD B STEPS
0976 CD59F2             CALL    STEP+ROF

0979 CD8CF3             CALL    ILPRNT+ROF      ;TRACK PACIFIER
097C AE                 DB      '.'+80H

097D CD24F4             CALL    PXCIN2          ;USER ABORT?

0980 AF                 XRA     A               ;SIDE 0 1ST

                ;LOOP TO FORMAT SIDES, IF THE DRIVE IS 2-SIDED

0981 5F         FHLOOP: MOV     E,A             ;E REMEMBERS SIDE
0982 47                 MOV     B,A             ;SELECT SIDE
0983 CD74F2             CALL    SELECT+ROF      ;SETS HL=MCMD+1 TOO

                ;LOOP TO FORMAT ALL 16 SECTORS. START WITH WHATEVER SECTOR WE
                ;FINDFIRST, AND PROCEED TO DO ALL SECTORS SEQUENTIALLY FROM
                ;THERE.GET THE SECTOR NUMBER FOR THE SECTOR HEADER FROM THE
                ;CONTROLLER'S SECTOR REGISTER.

0986 D5                 PUSH    D               ;SAVE D=LAST TRACK
                                                ; & E=SIDE

0987 3AD3F3             LDA     FORDAT+ROF      ;DATA FOR 256 SECTOR BYTES
```

```
098A 5F                 MOV     E,A

098B 010000             LXI     B,0             ;1ST SECTOR TIMEOUT
098E 1610               MVI     D,MAXSEC+1      ;D COUNTS SECTORS
0990 F3                 DI                      ;INTS OFF WHILE WE DO TRACK

0991 2D         FSLOOP: DCR     L               ;HL=MCMD=MSECTR

                ;HUNT FOR THE START OF ANY SECTOR

0992 0B         FHUNT:  DCX     B               ;(7)TIMEOUT?
0993 78                 MOV     A,B             ;(5)
0994 B1                 ORA     C               ;(4)
0995 CAA3F2             JZ      STOERR+ROF      ;(10)Y:ERROR

0998 AF                 XRA     A               ;(4)
0999 B6                 ORA     M               ;(7)TEST MSB=MSFLAG
099A F27CF1             JP      FHUNT+ROF       ;(10)WAIT FOR MSFLAG
                                                ;(47 CYCLES PER PASS)

099D 4F                 MOV     C,A             ;(4)SAVE SECTOR FOR HEADER
099E 2C                 INR     L               ;(4)HL=MCMD+1

099F 06FF               MVI     B,SYNCHR        ;(7)

09A1 3680               MVI     M,MWREN         ;(11)INITIATE WRITE
                                                ;(87 CYCLES MAX TO HERE)

                ;WAIT FOR THE TF FLAG TO BECOME TRUE.
                ;MUST BEGIN WRITING WITHIN 20 uS OF TF BECOMING TRUE
                ;THIS PRECLUDES THE POSSIBILITY OF A TIMEOUT HERE

09A3 AF                 XRA     A               ;(4)

09A4 B6         FWTF:   ORA     M               ;(7)MSB IS TR FLAG
09A5 F28EF1             JP      FWTF+ROF        ;(10) 17 CYCLES/PASS
                                                ;25 CYCLES MAX FROM TF

                ;WRITE THE SYNCH CHR AND 2-BYTE HEADER
                ;HEADER 1ST BYTE IS TRACK NUMBER FROM (TRACK)
                ;HEADER 2ND BYTE IS THE SECTOR NUMBER FROM C
                ;BEGIN CHECKSUM COMPUTATION TOO

09A8 2C                 INR     L               ;(4)HL=MCMD+2=DATA REG

09A9 70                 MOV     M,B             ;(10)WRITE SYNCH CHR

09AA 3ACFF3             LDA     TRACK+ROF       ;WRITE HEADR 1ST BYTE
09AD 77                 MOV     M,A             ;..WHICH IS THE TRACK #
09AE 47                 MOV     B,A             ;START OF CHECKSUM

09AF 79                 MOV     A,C             ;GET SECTOR REG DATA
09B0 E60F               ANI     MSMASK          ;STRIP OFF THE SECTOR

09B2 77                 MOV     M,A             ;2ND HEADER IS SECTOR #
09B3 80                 ADD     B               ;FIRST CKSUM ADD W/O CARRY

                ;WRITE THE FIRST 10 BYTES OF SECTOR DATA AS MFDATA
                ;(USUALLY 0). NOTE THAT CP/M IGNORES THE 1ST 10 BYTES
                ;OF EACH SECTOR.

                ;  HL=DATA REG
                ;  A=CHECKSUM SO FAR
```

```
09B4 060A               MVI     B,MBPS-256

09B6 3600       FBLP1:  MVI     M,MFDATA        ;WRITE A BYTE

                 IF MFDATA                      ;MUST CKSUM OF MFDATA<>0
                        ACI     MFDATA          ;CHECKSUM
                 ENDIF
09B8 05                 DCR     B
09B9 C2A0F1             JNZ     FBLP1+ROF

                ;WRITE THE REMAINING 256 BYTES OF SECTOR DATA PER E
                ;  HL=DATA REG
                ;  A=CHECKSUM SO FAR
                ;  E=SECTOR FORMAT DATA
                ;  B=0
                ; ONE BYTE TYME = 32 uS = 58 CYCLES

09BC 73         FBLP2:  MOV     M,E             ;WRITE A BYTE
09BD 8B                 ADC     E               ;COMPUTE CHECKSUM
09BE 05                 DCR     B
09BF C2A6F1             JNZ     FBLP2+ROF

                ; WRITE THE SECTOR CHECKSUM

09C2 0EFF               MVI     C,255           ;B=0. SHORT TIMEOUT FOR
                                                ;...THE NEXT SECTOR
09C4 77                 MOV     M,A

                ;NEXT SECTOR, QUICKLY, SO AS NOT TO MISS IT

09C5 2D                 DCR     L               ;HL=MCMD+1
09C6 15                 DCR     D               ;D COUNTS SECTORS
09C7 C27BF1             JNZ     FSLOOP+ROF

                ;WAIT FOR TRANSFER FLAG TO CLEAR BEFORE SWITCHING HEADS
                ;OR MOVING ACTUATOR, SO THAT THE CONTROLLER MAY FINISH
                ;WRITING THE POSTAMBLE

09CA AF         FWNTF:  XRA     A
09CB B6                 ORA     M               ;MSB IS TR FLAG
09CC FAB4F1             JM      FWNTF+ROF

                ;NEXT SIDE

09CF FB                 EI                      ;INTS ON FOR A MOMENT
09D0 D1                 POP     D               ;RECOVER D=LAST TRACK
                                                ;& E = SIDE

09D1 3AD1F3             LDA     SIDED2+ROF
09D4 BB                 CMP     E               ;AT MAX SIDE?
09D5 C26BF1             JNZ     FHLOOP+ROF

                ;NEXT TRACK

09D8 04                 INR     B               ;B=1: STEP ONCE

09D9 3ACFF3             LDA     TRACK+ROF       ;BUMP TRACK
09DC 3C                 INR     A
09DD 32CFF3             STA     TRACK+ROF

09E0 BA                 CMP     D               ;D=LAST TRACK + 1
09E1 DA5EF1             JC      FTLOOP+ROF      ;TOLERATES END < START
```

```
                ; DONE: RESTORE TO TRACK 0

09E4 2E00               MVI     L,0             ;SPECIFY TRACK 0
09E6 C3AAF0             JMP     DRESTR+ROF

                ;***SUBROUTINE*********************
                ; SET UP TO READ A SECTOR
                ; ON EXIT:
                ;   A = CHECKSUM SO FAR
                ;  CARRY VALID FROM CKSUM
                ;   B = MBPS-256
                ;  DE = DESTINATION ADDRESS
                ;  HL = DATA REGISTER ADDRESS
                ;  THE HEADER HAS ALREADY BEEN READ
                ;**********************************
09E9 CD12F2     RDSTUP: CALL    RWSTUP+ROF      ;GET PARAMS, HUNT FOR SECT
                                                ;A=0
                                                ;B=SYNCHR
                                                ;C=SECTOR #
                                                ;DE=DEST ADDRESS
                                                ;HL=MSTAT
                                                ;HEAD ON SECTOR FOR 46 uS
                                                ;INTERRUPTS DISABLED

                ; WAIT FOR THE TF FLAG TO BECOME TRUE
                ; MUST BEGIN READING WITHIN 20 uS OF TF BECOMING TRUE
                ; THIS PRECLUDES THE POSSIBILITY OF A TIMEOUT HERE

09EC B6         DRWTF:  ORA     M               ;(7)MSB IS TR FLAG
09ED F2D6F1             JP      DRWTF+ROF       ;(10) 17 CYCLES/PASS
                                                ;27 CYCLES MAX TO HERE

                ;READ SYNC BYTE, READ & SAVE 2-BYTE HEADER

09F0 2C                 INR     L               ;(4)POINT HL TO DATA REG

09F1 7E                 MOV     A,M             ;(7)GET/CHUCK SYNC BYTE

09F2 060A               MVI     B,MBPS-256      ;BYTE  FOR DATA LOOP
                                                ;(CAN'T DO BACK-TO-BACK READS)

09F4 7E                 MOV     A,M             ;(7)GET/SAVE 1ST HEADER
09F5 32D4F3             STA     HDRTRK+ROF      ;(13)..TRACK NUMBER
09F8 4F                 MOV     C,A             ;(4)INITIATE CHECKSUM

09F9 7E                 MOV     A,M             ;(7)GET/SAVE 2ND HEADER
09FA 32D5F3             STA     HDRSEC+ROF      ;(13)..SECTOR NUMBER
09FD 81                 ADD     C               ;(4)COMPUTE CHECKSUM
                                                ;CARRY IS PART OF IT
09FE C9                 RET

                ;***SUBROUTINE*********************
                ; SET UP TO WRITE A SECTOR
                ; ON EXIT:
                ;   A = CHECKSUM SO FAR
                ;  CARRY VALID FROM CKSUM
                ;   B = MBPS-256
                ;  DE = SOURCEN ADDRESS
                ;  HL = DATA REGISTER ADDRESS
                ;  THE HEADER HAS ALREADY BEEN WRITTEN
                ;**********************************
09FF F5         WRSTUP: PUSH    PSW             ;SAVE SECTOR/HEAD
```

```
0A00 3ACFF3            LDA     TRACK+ROF       ;DO WE KNOW WHERE WE ARE?
0A03 3C                INR     A               ;FFH MEANS NO
0A04 CA09F3            JZ      LSTERR+ROF      ;NO: WE ARE LOST

                ;ABORT IF DRIVE IS WRITE-PROTECTED

0A07 3A01FE            LDA     MSTAT           ;STATUS REG
0A0A E610              ANI     MWPT            ;WRITE PROTECT FLAG
0A0C C2C7F2            JNZ     WPTERR+ROF

0A0F F1                POP     PSW             ;RECOVER A=SECTOR/HEAD

0A10 CD12F2            CALL    RWSTUP+ROF      ;GET PARAMS, HUNT FOR SECT
                                               ;A=0
                                               ;B=SYNCHR
                                               ;SECTOR=SECTOR #
                                               ;DE=DEST ADDRESS
                                               ;HL=MSTAT
                                               ;HEAD ON SECTOR FOR 46 uS
                                               ;CURTRK = CURRENT TRACK
                                               ;INTERRUPTS DISABLED

                ; INITIATE WRITE, AND WAIT FOR THE TF FLAG TO BECOME TRUE
                ; MUST BEGIN WRITING WITHIN 20 uS OF TF BECOMING TRUE
                ; THIS PRECLUDES THE POSSIBILITY OF A TIMEOUT HERE

0A13 3680              MVI     M,MWREN         ;(11)INITIATE WRITE

0A15 B6        DWWTF:  ORA     M               ;(7)MSB IS TR FLAG
0A16 F2FFF1            JP      DWWTF+ROF       ;(10) 17 CYCLES/PASS
                                               ;27 CYCLES MAX TO HERE

                ;WRITE THE SYNCH CHR AND 2-BYTE HEADER
                ;HEADER 1ST BYTE IS TRACK NUMBER FROM (TRACK)
                ;HEADER 2ND BYTE IS THE SECTOR NUMBER
                ;BEGIN CHECKSUM COMPUTATION TOO

0A19 2C                INR     L               ;(4)HL=DATA REG

0A1A 70                MOV     M,B             ;(7)WRITE SYNCH CHR

0A1B 3ACFF3            LDA     TRACK+ROF       ;(13)WRITE HEADR 1ST BYTE
0A1E 77                MOV     M,A             ;(7)..IS THE TRACK #
0A1F 4F                MOV     C,A             ;(4)BEGIN CHECKSUM

0A20 060A              MVI     B,MBPS-256      ;(7)BYTE CNT FOR DATA LOOP

0A22 3AD2F3            LDA     SECTOR+ROF      ;(13)REQUESTED SECTOR
0A25 77                MOV     M,A             ;(7)2ND HEADER IS THE SECTOR
0A26 81                ADD     C               ;(4)FIRST CKSUM ADD W/O CARRY
                                               ;CARRY IS PART OF IT
0A27 C9                RET

                ;***SUBROUTINE*********************************
                ; SET UP UNIT FOR READ/WRITE
                ;   IGNORES JUMPER W9 & W10 BITS IN MSECTR
                ; ON ENTRY:
                ;   A<3:0> = DESIRED SECTOR
                ;   A<4>   = HEAD
                ;   DE POINTS TO NEXT PARAM
                ;   UNIT HAS SELECTED UNIT
                ; ON EXIT:
```

```
                ;   DRIVE IS READY FOR READ/WRITE FOR 46 uS MIN
                ;   A=0
                ;   B=SYNCHR
                ;   DE=SOURCE/DEST ADDRESS
                ;   HL=MSTAT
                ;   SECTOR=SECTOR NUMBER
                ;   INTERRUPTS ARE DISABLED
                ;***********************************************
0A28 4F         RWSTUP: MOV     C,A             ;SAVE SECTOR #

                ;GET SOURCE/DEST ADDRESS INTO HL

0A29 CD1BF4             CALL    PXIHEX          ;GET ADDRESS
0A2C DA1CF3             JC      ERROR+ROF       ;MUST SPECIFY ADDRESS
0A2F E5                 PUSH    H               ;SOURCE/DEST ADDRESS

                ; VALIDATE SIDE AND SECTOR, SET B=SIDE BIT, C=SECTOR

0A30 3AD1F3             LDA     SIDED2+ROF      ;GET SIDED MASK
0A33 47                 MOV     B,A
0A34 C60F               ADI     MAXSEC
0A36 B9                 CMP     C               ;SECTOR TOO BIG?
0A37 DA1CF3             JC      ERROR+ROF

0A3A 79                 MOV     A,C             ;SECTOR & SIDE
0A3B A0                 ANA     B               ;STRIP JUST THE SIDE
0A3C 47                 MOV     B,A             ;..INTO B FOR SELECT


0A3D 79                 MOV     A,C
0A3E E60F               ANI     0FH             ;STRIP JUST THE SECTOR
0A40 32D2F3             STA     SECTOR+ROF      ;SAVE FOR VALIDATION

0A43 F680               ORI     MSFLAG          ;SECTOR MUST BE VALID
0A45 4F                 MOV     C,A             ;INTO C FOR HUNT LOOP

                ;SELECT THE SPECIFIED UNIT AND SIDE

0A46 CD74F2             CALL    SELECT+ROF      ;SELECT DRIVE & HEAD
                                                ;HL=MCMD+1=MSTAT

                ;STALL FOR >250 mS TO LET SECTOR LOGIC SETTLE
                ;NOTE: INTERRUPTS ARE BLOCKED DURING STALL

0A49 11324B             LXI     D,(CPUKHZ/24)*250
0A4C CD50F2             CALL    STALLD+ROF      ;..SECTOR CNTR SETTLE
                                                ;EXITS WITH DE=0

0A4F 06FF               MVI     B,SYNCHR        ;B=SYNCHR FOR EXIT

                ;HUNT FOR SPECIFIED SECTOR. 200 mS/REV
                ;TIMEOUT IF NOT FOUND. DE=0 FOR ~1.5 uS TIMEOUT

0A51 2D                 DCR     L               ;POINT HL TO MSECTR

0A52 F3                 DI                      ;INTS OFF WHILE WE HUNT

0A53 1B         HUNT:   DCX     D               ;(7)TIMEOUT?
0A54 7A                 MOV     A,D             ;(5)
0A55 B3                 ORA     E               ;(4)
0A56 CAA3F2             JZ      STOERR+ROF      ;(10)Y:ERROR

0A59 3E8F               MVI     A,MSMASK+MSFLAG ;(7)CORRECT SECTOR, READY
```

```
0A5B A6                 ANA     M               ;(7)READ SECTOR REG
0A5C 91                 SUB     C               ;(4)MATCH DESIRED SECTOR?
                                                ;   A=0 IF SO
0A5D C23DF2             JNZ     HUNT+ROF        ;(10)N: KEEP LOOKING
                                                ;55 CYCLES PER PASS
                                                ;WORST CASE 75 CYCLES TO HERE

0A60 2C                 INR     L               ;(4)POINT HL TO STATUS REG

0A61 D1                 POP     D               ;(10)SOURCE/DEST ADDR

0A62 C9                 RET                     ;(10)100 CYCLES MAX FROM
                                                ;.. ON-SECTOR = 54 uS

                ;***SUBROUTINE*************************
                ;STALL FOR 30 uS
                ;NOTE: INTERRUPTS ARE OFF DURING STALL
                ;
                ; ON RETURN:
                ;  A,DE=0
                ; INTS ARE ENABLED
                ;***************************************
0A63 110809     STAL30: LXI     D,(30*CPUKHZ)/24 ;MAKES IT 30 mS

                ; FALL INTO STALLD

                ;***SUBROUTINE*************************
                ;STALL FOR DE*24 CYCLES
                ;NOTE: INTERRUPTS ARE OFF DURING STALL
                ;
                ; ON RETURN:
                ;  A,DE=0
                ; INTS ARE ENABLED
                ;***************************************
0A66 F3         STALLD: DI
0A67 1B         STALUP: DCX     D               ;(5)
0A68 7A                 MOV     A,D             ;(5)
0A69 B3                 ORA     E               ;(4)
0A6A C251F2             JNZ     STALUP+ROF      ;(10)
                                                ;24 CYCLES/PASS
0A6D FB                 EI
0A6E C9                 RET

                ;***SUBROUTINE*********************************
                ;STEP POSITIONER
                ; ON ENTRY:
                ;   B = NUMBER OF STEPS
                ;   C = MSTEP TO STEP OUTWARD
                ;     = MSTEP+1 TO STEP INWARD
                ;   HL= MCMD+1
                ; ON EXIT:
                ;   Z SET AND A=0 IF B COUNTED ALL THE WAY DOWN
                ;   Z CLEARED IF EXIT BECAUSE OF TRACK 0
                ; TRASHES A,B
                ;*******************************************
0A6F 04         STEP:   INR     B

0A70 05         STLUP:  DCR     B               ;ON DESIRED TRACK?
0A71 C8                 RZ                      ;Y: DONE
0A72 71                 MOV     M,C             ;STEP INWARD/OUTWARD
0A73 D5                 PUSH    D
0A74 CD4DF2             CALL    STAL30+ROF      ;STALL FOR STEP
0A77 D1                 POP     D
```

```
0A78 7E              MOV     A,M             ;DISK STAT REG
0A79 E608            ANI     MTK0            ;TRACK 0?
0A7B C0              RNZ                     ;DONE IF ON TRACK 0
0A7C C35AF2          JMP     STLUP+ROF

                ;***SUBROUTINE*********************************
                ; SET UP FOR SEEK
                ; ON ENTRY:
                ;   L=TARGET TRACK
                ; ON EXIT:
                ;   C=MSTEP+1
                ;   E=TARGET TRACK
                ;   HL=MCMD+1
                ;   (HL)=DRIVE STATUS
                ; TRASHES A,B
                ; ABORT IF DRIVE NOT READY OR BAD TRACK NUMBER
                ;*****************************************
0A7F 3AD0F3     SKSTUP: LDA     MAXTRK+ROF      ;GET MAX TRACK #
0A82 BD                 CMP     L               ;REQUESTED TRACK
0A83 DA1CF3             JC      ERROR+ROF       ;BOUNDS CHECK

0A86 5D                 MOV     E,L             ;E=TARGET TRACK

0A87 016100             LXI     B,MSTEP+1       ;HEAD 0, C FOR RET

                ; FALL INTO SELECT

                ;***SUBROUTINE*******************
                ; SELECT UNIT AND HEAD
                ; ON ENTRY:
                ;   B=0 FOR SIDE 0, 10 FOR SIDE 1
                ; ON EXIT:
                ;   Z SET
                ;   HL=MCMD+1
                ;   (HL)=DRIVE STATUS
                ; ABORT IF DRIVE NOT READY
                ; TRASHES A,B
                ;*********************************
0A8A 2101FE     SELECT: LXI     H,MCMD+1
0A8D 7E                 MOV     A,M             ;WAKE UP CONTROLLER

0A8E 3640               MVI     M,MINTC         ;DISABLE DISK INTS

0A90 3ACEF3             LDA     UNIT+ROF        ;SELECT UNIT

0A93 B0                 ORA     B               ;COMBINE SIDE

0A94 47                 MOV     B,A             ;B HAS HEAD & UNIT

0A95 F620               ORI     MDHSEL          ;SELECT COMMAND
0A97 77                 MOV     M,A             ;TO CONTROLLER

0A98 D5                 PUSH    D
0A99 CD4DF2             CALL    STAL30+ROF
0A9C D1                 POP     D

0A9D 3E20               MVI     A,MRDY          ;ALSO TEST DRIVE READY
0A9F AE                 XRA     M               ;TEST SELECTED UNIT

0AA0 E627               ANI     MUMASK+MRDY     ;DID UNIT GET SELECTED?
0AA2 B8                 CMP     B               ;THE RIGHT ONE?

0AA3 C8                 RZ
```

```
                ;FALL INTO DRIVE NOT READY ERROR

                ;***ERROR EXIT**********
                ; DRIVE NOT READY
                ; ON ENTRY:
                ;   HL=MSTAT
                ;***********************
0AA4 CD88F3     DNRERR: CALL    CILPRT+ROF      ;CR FIRST
0AA7 4472697665         DB      'Drive not read','y'+80H
0AB6 C3F3F2             JMP     DSKERR+ROF      ;WILL PRINT STATUS REG

                ;***ERROR EXIT******************
                ; SECTOR NOT FOUND (HUNT TIMEOUT)
                ; ON ENTRY:
                ;   INTERRUPTS ARE DISABLED
                ;   HL=MSECTR
                ;*******************************
0AB9 FB         STOERR: EI
0ABA CD88F3             CALL    CILPRT+ROF      ;CR FIRST
0ABD 536563746F         DB      'Sector not found'
0ACD 0D53656374         DB      CR,'Sector reg=',' '+80H
0ADA C3F3F2             JMP     DSKERR+ROF      ;WILL PRINT SECTOR REG

                ;***ERROR EXIT**********
                ; WRITE-PROTECTED
                ; ON ENTRY:
                ;   HL=MSTAT
                ;***********************
0ADD CD88F3     WPTERR: CALL    CILPRT+ROF      ;CR FIRST
0AE0 5772697465         DB      'Write protec','t'+80H
0AED C3F3F2             JMP     DSKERR+ROF      ;WILL PRINT STATUS REG

                ;***ERROR EXIT**********
                ; RESTORE ERROR -
                ; TRACK 0 NOT FOUND
                ; ON ENTRY:
                ;   HL=MSTAT
                ;************************
0AF0 3EFF       RSTERR: MVI     A,0FFH          ;DON'T KNOW WHAT TRACK
0AF2 32CFF3             STA     TRACK+ROF
0AF5 CD88F3             CALL    CILPRT+ROF      ;CR FIRST
0AF8 547261636B         DB      'Track 0 not foun','d'+80H

                ;FALL INTO DSKERR

                ;***ERROR EXIT**********************
                ; DISK ERROR
                ; PRINT DISK STATUS & QUIT
                ; ON ENTRY:
                ;   HL=ADDRESS OF STATUS REG TO READ
                ;***********************************
0B09 CD88F3     DSKERR: CALL    CILPRT+ROF
0B0C 5374617475         DB      'Status reg=',' '+80H

0B18 7E         DSKER2: MOV     A,M
0B19 CD1EF4             CALL    PXOHEX          ;PRINT VALUE
0B1C C321F4             JMP     PXWARM          ;BACK TO MAIN
                                                ;WHERE STACK GETS FIXED

                ;***ERROR EXIT**********
                ; DON'T KNOW WHAT TRACK
                ;***********************
```

```
0B1F CD88F3    LSTERR: CALL    CILPRT+ROF
0B22 4D75737420        DB      'Must RE firs','t'+80h
0B2F C321F4            JMP     PXWARM          ;BACK TO MAIN

               ;***ERROR EXIT**********
               ; GENERIC ERROR HANDLER
               ; PRINT ERROR MESSAGE,
               ; RETURN WITH Z CLEARED
               ;***********************
0B32 CD88F3    ERROR:  CALL    CILPRT+ROF      ;CR FIRST
0B35 BF                DB      '?'+80H         ;CLEARS Z FOR RET

0B36 C321F4            JMP     PXWARM          ;BACK TO MAIN
                                               ;WHERE STACK GETS FIXED

               ;***ERROR SUBROUTINE****
               ; READ CHECKSUM
               ;***********************
0B39 CD88F3    CSMERR: CALL    CILPRT+ROF      ;CR FIRST
0B3C 436865636B        DB      'Checksum erro','r'+80H
0B4A C9                RET

               ;***ERROR SUBROUTINE***********
               ; SECTOR HEADER MISMATCH ERROR
               ;******************************
0B4B CD88F3    HMMERR: CALL    CILPRT+ROF
0B4E 4865616465        DB      'Header mismatc','h'+80h
0B5D C9                RET

               ;*****EXIT ROUTINE*********************************
               ;COMPARE TRACK AND SECTOR FROM THE SECTOR HEADER
               ;TO THE EXPECTED TRACK AND SECTOR, AND REPORT ERROR
               ;IF DIFFERENT. HOWEVER, IF WE DON'T KNOW WHAT TRACK
               ;WE ARE ON (TRACK=FF), THEN UPDATE TRACK WITH THE
               ;HEADER'S TRACK NUMBER. REPORT THE HEADER'S TRACK
               ;AND SECTOR.
               ; ON ENTRY:
               ;   HDRTRK = FOUND TRACK NUMBER
               ;   HDRSEC = FOUND SECTOR NUMBER
               ;   TRACK = EXPECTED SECTOR, 0FFH IF LOST
               ;   SECTOR = EXPECTED SECTOR
               ;   B=0
               ;************************************************
0B5E 2AD4F3    CHKHDR: LHLD    HDRTRK+ROF      ;FOUND TRACK AND SECTOR

0B61 3ACFF3            LDA     TRACK+ROF       ;EXPECTED TRACK
0B64 FEFF              CPI     0FFH            ;DO WE KNOW WHAT TRACK?
0B66 C257F3            JNZ     CHKH1+ROF       ;Y: JUST COMPARE

0B69 7D                MOV     A,L             ;LOST: NOW WE ARE FOUND
0B6A 32CFF3            STA     TRACK+ROF       ;REMEMBER CURRENT TRACK

0B6D AD        CHKH1:  XRA     L               ;CORRECT TRACK?
0B6E 4F                MOV     C,A

0B6F 3AD2F3            LDA     SECTOR+ROF      ;EXPECTED SECTOR
0B72 AC                XRA     H               ;CORRECT SECTOR?
0B73 B1                ORA     C
0B74 C435F3            CNZ     HMMERR+ROF      ;EITHER ERROR

0B77 CD88F3            CALL    CILPRT+ROF      ;CR FIRST
0B7A 4865616465        DB      'Header: track',' '+80H
```

```
0B88 7D                 MOV     A,L             ;FOUND TRACK
0B89 CD1EF4             CALL    PXOHEX

0B8C CD8CF3             CALL    ILPRNT+ROF
0B8F 2C20736563         DB      ', sector',' '+80H

0B98 7C                 MOV     A,H             ;FOUND SECTOR
0B99 CD1EF4             CALL    PXOHEX

0B9C 04                 INR     B               ;CLEAR Z
0B9D C9                 RET

                ;*****SUBROUTINE*****************************
                ; PRINT CR THEN INLINE MESSAGE
                ; CALLS TO CILPRT ARE FOLLOWED BY THE STRING
                ; THE LAST STRING BYTE HAS ITS MSB SET
                ; A TRASHED, ALL OTHER REGISTERS PRESERVED
                ;*****************************************
0B9E CD8CF3     CILPRT: CALL    ILPRNT+ROF
0BA1 8D                 DB      CR+80H

                ; FALL INTO ILPRNT

                ;*****SUBROUTINE*****************************
                ; PRINT INLINE MESSAGE
                ; CALLS TO ILPRNT ARE FOLLOWED BY THE STRING
                ; THE LAST STRING BYTE HAS ITS MSB SET
                ; Z CLEARED ON EXIT
                ; A TRASHED, ALL OTHER REGISTERS PRESERVED
                ;*****************************************
0BA2 E3         ILPRNT: XTHL                    ;SAVE HL, GET MSG ADDR
0BA3 C5                 PUSH    B

0BA4 7E         IPLOOP: MOV     A,M             ;LOOP THROUGH MESSAGE
0BA5 E67F               ANI     7FH             ;STRIP END-MARKER
0BA7 4F                 MOV     C,A             ;PXCOUT WANTS CHR IN C
0BA8 CD09F4             CALL    PXCOUT
0BAB BE                 CMP     M               ;END?
0BAC 23                 INX     H
0BAD CA8EF3             JZ      IPLOOP+ROF

0BB0 C1                 POP     B
0BB1 E3                 XTHL                    ;RESTORE HL
                                                ;..GET RET ADDRESS
0BB2 C9                 RET

                ;*********************************
                ; COMMAND TABLE
                ; NOTE THAT ASCII CHRS STORED IN
                ; VIDEO MEMORY HAVE THEIR MSB SET
                ;*********************************
0BB3 C2CF       COMTAB: DB      'B'+80H,'O'+80H ;BOOT FROM MICROPOLIS
0BB5 00                 DB      GOMICR-CMDBAS
0BB6 D2C5               DB      'R'+80H,'E'+80H ;RESTORE TO TRACK
0BB8 6A                 DB      DRESTR-CMDBAS
0BB9 D3CB               DB      'S'+80H,'K'+80H ;SEEK
0BBB 4A                 DB      DSEEK-CMDBAS
0BBC D2CD               DB      'R'+80H,'M'+80H ;READ 266-BYTE SECTOR
0BBE 8F                 DB      DRD266-CMDBAS
0BBF D2C3               DB      'R'+80H,'C'+80H ;READ 256-BYTE SECTOR
0BC1 84                 DB      DRD256-CMDBAS
0BC2 D7CD               DB      'W'+80H,'M'+80H ;WRITE 266-BYTE SECTOR
0BC4 B8                 DB      DWR266-CMDBAS
```

```
0BC5 D7C3               DB      'W'+80H,'C'+80H ;WRITE 256-BYTE SECTOR
0BC7 AC                 DB      DWR256-CMDBAS
0BC8 C4CD               DB      'D'+80H,'M'+80H ;DISK MODE LEVEL
0BCA 03                 DB      SETMOD-CMDBAS
0BCB CCC8               DB      'L'+80H,'H'+80H ;LOAD HEAD
0BCD 27                 DB      DHLOAD-CMDBAS
0BCE D3D5               DB      'S'+80H,'U'+80H ;SET UNIT
0BD0 19                 DB      STUNIT-CMDBAS
0BD1 C6CD               DB      'F'+80H,'M'+80H ;FORMAT DISK
0BD3 D3                 DB      MFORMT-CMDBAS
0BD4 C6C3               DB      'F'+80H,'C'+80H ;CP/M FORMAT DISK
0BD6 D0                 DB      CFORMT-CMDBAS

0BD7 00                 DB      0               ;END OF TABLE MARK

                ;*****************************
                ; MICROPOLIS MODEL TABLE
                ; 2 BYTES/ENTRY
                ; 1ST BYTE IS MAX TRACK NUMBER
                ; 2ND BYTE IS SIDED
                ;*****************************
0BD8 2200       MODTAB: DB      34,0    ;MOD 1
0BDA 4C00               DB      76,0    ;MOD 2
0BDC 4C10               DB      76,10H  ;MOD 3 (DOES NOT EXIST)
0BDE 4C10               DB      76,10H  ;MOD 4
0BE0 4F00               DB      79,0    ;MOD 5
0BE2 4F10               DB      79,10H  ;MOD 6

                ;*** END OF PROGRAM ***

                ;***********************
                ; INITIALIZED VARIABLES
                ;***********************
0BE4 00         UNIT:   DB      0       ;CURRENTLY SELECTED DRIVE (0-3)
0BE5 FF         TRACK:  DB      0FFH    ;CURRENT TRACK (FF MEANS LOST)

                ;SIDED2 MUST FOLLOW MAXTRK

0BE6 22         MAXTRK: DB      34      ;MAX TRACK NUMBER (INIT TO MOD 1)
0BE7 00         SIDED2: DB      0       ;0 MEANS 1-SIDED (INIT TO MOD 1)

                RAMEND:                 ;END OF COPIED CODE

                ;****************************
                ; UNITIALIZED RAM VARIABLES
                ;****************************
0BE8            SECTOR: DS      1       ;REQUESTED SECTOR

0BE9            FORDAT: DS      1       ;FORMAT DATA FOR 256 BYTES

                ; HDRSEC MUST FOLLOW HDRTRK

0BEA            HDRTRK: DS      1       ;1ST SECTOR HEADER BYTE IS TRACK #
0BEB            HDRSEC: DS      1       ;2ND SECTOR HEADER BYTE IS SECTOR #
0BEC                    END
```