

# Appendix A

## CP/M Error Messages

---

This appendix lists in alphabetical order the error messages for the SoftCard implementation of CP/M. Each error message includes the possible causes and what actions you may take in response to them. In each of the error message descriptions, *d:* represents the disk drive identifier (A:-D:).

### Aborted

*Cause.* PIP stopped; a key was pressed by the user.

*Action.* Retry the command.

### Bad Delimiter

*Cause.* The wrong delimiting character was used in the STAT command line.

*Action.* Check for typing errors and try the command again.

### BDOS ERR ON *d:* Bad Sector

*Cause.* An attempt was made to execute a command (built-in or transient) when:

There was no disk in the specified drive

The drive door was not closed

The disk was inserted into the specified drive improperly

The drive was not connected to a disk controller board (see SELECT error)

The disk was damaged or worn

*Action.* Correct the error condition. Then press the CONTROL-C keys to perform a warm start. Retry the command.

## **BDOS ERR ON d: File R/O**

*Cause.* A write operation was attempted to a file that has a Read Only (R/O) attribute.

*Action.* Type any character to perform a warm start and return to CP/M command level. (

## **BDOS ERR ON d: R/O**

*Cause.* The disk in the accessed drive was changed without pressing CONTROL-C; or there is a write-protect tab on the disk.

*Action.* Press any key to perform a warm start and return to CP/M command level.

## **BDOS ERR ON d: Select**

*Cause.* An attempt was made to access a disk drive when either the drive was not connected to a controller, or the controller has been installed in the wrong accessory slot.

Note that if you have only one drive attached to a disk controller board, an attempt to access a drive that is not installed results in a BAD SECTOR error instead of a SELECT error.

*Action.* Press any key to perform a warm start and return to CP/M command level. (

## **Cannot Close Destination File-*filespec***

*Cause.* The output file specified in the PIP command line cannot be closed. This is usually caused by a write-protect tab on the disk.

*Action.* Remove the write-protect tab from the disk and try the command again.

**Cannot Close File**

*Cause.* A write operation has been attempted to a disk that has a write-protect tab on it.

*Action.* Make sure the disk is not write-protected and try the command again.

**Cannot Read**

*Cause.* The PIP program cannot read the source device specified in the command line.

*Action.* Check the RDR: device assignment and the physical connections to the current reader device.

**Cannot Write**

*Cause.* An invalid destination device was specified in the PIP command line.

*Action.* Check the device assignments and retry the command.

**Checksum Error**

*Cause.* PIP encountered a hex checksum record error while copying a hex file.

*Action.* Recreate the hex file with an assembler program and retry the command.

**Checksum Error Load Address ...**

*Cause.* The file specified in the LOAD command line contains errors.

*Action.* Recreate the hex file with an assembler program and retry the command.

**Command Buffer Overflow**

*Cause.* There are too many characters in the SUBMIT command buffer.

*Action.* Make sure that the submit input file doesn't exceed 2048 characters.

## Command Error

*Cause.* Either there is a syntax error in the command line or the command is not understood (i.e., the arguments in the command line were not recognized by the program). Command Error is generated by utility programs written by Microsoft.

*Action.* Retype the command line in the correct format and retry the command.

## Command Too Long

*Cause.* A command in the submit input file is longer than 125 characters.

*Action.* Check the commands in the submit input file and re-submit the input file.

## Correct Error, Type Return or CTRL-Z

*Cause.* A hex record checksum error was encountered during the transfer of a hex file by PIP.

*Action.* Correct the hex file and retry the command.

## Destination is R/O, Delete (y/n)

*Cause.* The destination file specified in the PIP command line is designated R/O.

*Action.* Enter *Y* to delete the existing file and PIP will complete the copy process. Enter *N* to abort the copy process.

## Directory Full

*Cause.* An attempt was made to copy files to a destination disk which has no more storage space.

*Action.* Insert another disk in the destination drive and retry the command.

## Disk Full

*Cause.* An attempt was made to copy files to a destination disk which has no more storage space. This error message is generated by the APDOS, MFT, or UPLOAD programs.

*Action.* Insert another disk in the destination drive and retry the command.

### **Disk I/O Error**

*Cause.* The COPY program cannot format the disk. This is caused by either a bad or a worn-out disk, or the disk drive door is not closed.

*Action.* Ensure that the disk drive door is closed. If the same error message appears, replace the disk.

### **Disk Read Error**

*Cause.* The source file specified in the PIP command contains an end-of-file character in the wrong place.

*Action.* Make sure that the end-of-file character is in the right place.

### **Disk Write Error**

*Cause.* A disk write operation was attempted to a full disk.

*Action.* Either erase some files or try the write operation with another disk.

### **Disk Write-Protected**

*Cause.* An attempt was made to write to a disk that has a write-protect tab on it. This error message is generated by the APDOS and COPY programs.

*Action.* Remove the write-protect tab from the disk and retry the command.

### **Error:Bad Parameter**

*Cause.* There is an illegal parameter in the PIP command line.

*Action.* Check the command line and retry the command.

### **Error:Cannot Close File, Load Address xxxx**

*Cause.* An error exists in the program being loaded with the LOAD program. The disk may be write-protected.

*Action.* Check the source program for errors. Check the disk for a write-protect tab.

**Error:Cannot Open Source, Load Address xxxx**

*Cause.* The LOAD program cannot find the file specified in the LOAD command line; or no filename was specified.

*Action.* Check the filename of the source file to be loaded. Make sure the filename is included in the LOAD command line. Retry the command.

**Error:Disk Inverted, Load Address xxxx**

*Cause.* The address of a record was too far from the address of the previously processed record.

*Action.* Use DDT to read the hex file into memory, then use the SAVE command to store the file back to the disk. Retry the LOAD command.

**Error:Disk No More Directory Space, Load Address xxxx**

*Cause.* The destination disk in the active drive is full.

*Action.* Change disks and retry the command.

**Error:Disk Read, Load Address xxxx**

*Cause.* The file specified in the LOAD command line cannot be found on the disk.

*Action.* Check to see that the file exists on the disk in the active drive.

**Error:Disk Write, Load Address xxxx**

*Cause.* The destination disk in the active drive is full.

*Action.* Change disks and retry the command.

**Error On Line nnn**

*Cause.* There is an error in the submit input file at the specified line number (nnn).

*Action.* Correct the error and retry the command.

**File Error**

*Cause.* The disk is full and the ED program cannot write any more data to the accessed file.

*Action.* Copy the file to another disk or delete other files from the same disk.

**File Exists**

*Cause.* An attempt was made to change the name of a file to an existing filename.

*Action.* Make sure the "new" filespec argument in the REN command line does not match any existing filenames on the same disk. Retry the command.

**File Exists, Erase It**

*Cause.* The destination file named in the ED command line already exists.

*Action.* Place the destination file on another disk or in a different user area.

**File Is Read Only**

*Cause.* The file specified in the ED command line has an R/O attribute.

*Action.* Change the file status with the STAT program.

**File Not Found**

*Cause.* The source file specified in the APDOS, AUTORUN, CAT, COPY, MFT, PATCH, STAT, or UPLOAD command line does not exist.

*Action.* Check the spelling of the filename and reenter the command line.

## Filename?

*Cause.* An incorrect use of wild card characters in the REN command line.

*Action.* Retry the command with no wild card characters in the command line.

## Filename Required

*Cause.* ED was invoked without a filename argument in the command line.

*Action.* Include a filename in the ED command line.

## hhhh?? = dd

*Cause.* The mnemonic (*dd*) at address (*hhhh??*) is not an 8080 or Z80 assembly language instruction.

*Action.* Correct the mnemonic.

## Insufficient Memory

*Cause.* There is not enough memory available to load the specified file with the DDT R or E command.

*Action.* Reduce the size of the file and load in segments of the file.



### Invalid Assignment

*Cause.* One of the device names specified in the STAT command line is either misspelled or cannot be assigned to the other specified device.

*Action.* Check the spelling of the device name and retry the command. If the same error message appears again, check for the valid device assignments by typing *STAT VAL:*.

### Invalid Control Character

*Cause.* An invalid CONTROL character was included in a submit input file.

*Action.* Use only ^A through ^Z CONTROL characters in a submit input file.

### Invalid Digit

*Cause.* The hex file specified in the PIP command line contains an invalid hex digit.

*Action.* Correct the hex file and retry the PIP command.

### Invalid Disk Assignment

*Cause.* An attempt was made to assign an attribute other than R/O to a disk drive with the STAT program.

*Action.* Assign only the R/O attribute to disk drives. Remove the R/O attribute with the STAT program.

### Invalid Disk Select

*Cause.* A command line specified a nonexistent disk drive.

*Action.* Specify only disk drives A: through D: in the command line. Check for any loose connections to the disk drives and for unformatted disks.

### Invalid File Indicator

*Cause.* STAT did not recognize the attribute in the STAT command line.

*Action.* Specify only R/O, R/W, DIR, or SYS in the STAT command line.

### Invalid Format

*Cause.* The PIP command line was in the wrong format.

*Action.* Check the command line format and retry the command.

### Invalid Hex Digit ...

*Cause.* The file specified in the LOAD command line contains an incorrect hex digit.

*Action.* Correct the file and retry the command.

### Invalid Separator

*Cause.* An invalid separator character was used in the PIP command line.

*Action.* Check the command line format and retry the command.

### Invalid User Number

*Cause.* An invalid user number was specified in the PIP command line.

*Action.* Use only user numbers 1 through 15.

### *n?*

*Cause.* A number greater than 15 was specified in the USER command line.

*Action.* Use only user numbers 1 through 15.

### No Directory Space

*Cause.* There is no room on the disk for the .PRN and .HEX files generated by ASM.

*Action.* Either delete files from the active drive or specify another drive.

### No File *filespec*

*Cause.* The file specified in the command line cannot be found.

*Action.* Recheck the spelling of the filespec and try again.

### **No File(s) Found, xxxk Bytes Available**

*Cause.* The file specified in the CAT command line does not exist.

*Action.* Check the spelling of the filename and reenter the command line.

### **No Input File Present On Disk**

*Cause.* The file specified in the DUMP command line does not exist.

*Action.* Recheck the spelling of the filespec and try again.

### **No Source File Present**

*Cause.* The ASM assembler could not find the file specified in the command line.

*Action.* Check spelling of the file and ensure that the disk is listed in the disk directory. Retry the command.

### **No Space**

*Cause.* An attempt was made to save the contents of memory with the SAVE command, but there is no space left on the disk.

*Action.* Use a disk with sufficient storage space and retry the command.

### **No Sub File Present**

*Cause.* The SUBMIT program was run but no submit input file was specified.

*Action.* Create a submit input file.

### **Not A Character Source**

*Cause.* An invalid source device was specified in the PIP command line.

*Action.* Use either RDR: or CON: as source devices.

## Not Deleted

*Cause.* The file specified in the PIP command line has an R/O attribute and cannot be deleted.

*Action.* Change the status of the file with the STAT program.

## Not Found

*Cause.* PIP cannot find the file specified in the command line.

*Action.* Check the spelling of the file and try again.

## Output File Write Error

*Cause.* Either a file with write-protect status has been specified as the ASM destination file, or there is no free space left on the disk.

*Action.* Check the attributes of the destination file and the amount of free disk space with the STAT program.

## Parameter Error

*Cause.* The submit input file contains an invalid parameter.

*Action.* Use only valid parameters (\$0 through \$9) in the submit input file.

## Quit Not Found

*Cause.* The Q parameter was specified in the PIP command line but there is no string argument in the input file.

*Action.* Insert the appropriate string argument in the input file specified by PIP.

## Read Error

*Cause.* The file specified in the TYPE command line contains an error.

*Action.* Use the STAT program to check the disk and the file. Retry the command.

**Reader Stopping**

*Cause.* The read operation from the RDR: device has been interrupted. (A key was pressed during the read operation.)

*Action.* Retry the command.

**Record Too Long**

*Cause.* The file or device specified in the PIP command line contains a record longer than 128 bytes.

*Action.* Use the STAT program to check for any records longer than 128 bytes.

**Source File Name Error**

*Cause.* Wild card characters \* and ? were specified in the source filename argument of the ASM command line.

*Action.* Specify only one source filename in the ASM command line.

**Source File Read Error**

*Cause.* The file read by the ASM assembler is in the wrong format or has instructions the ASM assembler cannot understand.

*Action.* Check the file for the proper format and check that the assembly language instructions are 8080 mnemonics.

**Start Not Found**

*Cause.* PIP cannot find the string argument in the input file specified by the S parameter.

*Action.* Check the input file for the appropriate string argument.

**Too Many Files**

*Cause.* STAT cannot process the files specified. Either there are too many files (more than 512), or there is not enough free RAM available to process the files.

*Action.* Delete or transfer files to another disk. Retry the command and specify fewer files.

### Unexpected End Of Hex File *filespec*

*Cause.* The hex file specified in the PIP command line contains an end-of-file character before a termination hex record.

*Action.* Correct the hex file and retry the command.

### Unrecognized Destination

*Cause.* PIP did not recognize the destination file or device specified in the command line.

*Action.* Make sure that the destination device is a currently assigned device, or that the destination file exists.

### Use: STAT *d*:=R/O

*Cause.* The drive argument (*d*:) in the STAT command line was used in the wrong format.

*Action.* Use the proper format (STAT *d*:=R/O) for the drive argument.

### Verify Error

*Cause.* The data copied onto a destination disk does not match the data on the source disk. This is caused by a worn or damaged disk, or by hardware problems. The VERIFY ERROR message is generated by the COPY and PATCH transient programs.

*Action.* Try using a different disk and repeat the command. If the same error message appears again, check the connections to the disk drives and the disk controllers. If there is a hardware problem, contact your dealer.

### ? (Syntax error)

*Cause.* The command was not understood. Either the command was mistyped, or invalid arguments were included in the command line.

*Action.* Retype the command line in the correct format.

## **Appendix B**

# **Downloading to the SoftCard**

---

Program Requirements 249

Downloading Procedure 250

    UPLOAD Source Listing 254

    DOWNLOAD Source Listing 256

C

C

C



The Premium SoftCard IIe System allows you to copy CP/M files from CP/M-based computer systems to Apple systems through an RS-232 Serial I/O port. UPLOAD and DOWNLOAD programs permit you to copy programs that are in a different disk format (for example, on 8-inch disk drives) to the SoftCard CP/M format (5-1/4-inch disk drives).

The UPLOAD program is a source listing that must be entered into the source computer (a system that does not support SoftCard) as part of the communication link. UPLOAD must be configured for the source computer's specific I/O environment with the DDT transient program.

DOWNLOAD is the counterpart to the UPLOAD program that must be entered into your Apple IIe computer system.

---

### **Warning**

Use of UPLOAD and DOWNLOAD assumes familiarity with 8080 assembly language programming. These programs are intended for experienced programmers only!

---

## **Program Requirements**

To use the UPLOAD and DOWNLOAD programs, you will need the following:

1. A working knowledge of the DDT transient program and 8080 assembly language programming.
2. A CP/M-based computer system (in addition to your Apple IIe) with an RS-232 Serial I/O port other than the port used for console I/O.
3. An Apple Communications Interface board or a CCS 7710A Serial Interface board installed in slot 2 of the Apple IIe computer.
4. An RS-232 Serial I/O port cable.

## Downloading Procedure

The following procedure shows how to copy files from another CP/M system to your Apple IIe system.

1. Insert a copy of the SoftCard Master disk into drive A:.
2. Load CP/M with a cold start and then load the DDT transient program.
3. Using DDT, enter the following machine-language program (UPLOAD) into the source computer starting at address 0163H:

```

0163 3A 80 00 B7 11 D7 01 CA CC 01 CD 03 01
0170 0E 0F 11 5C 00 CD 05 00 3C 11 E5 01 CA CC 01 3E
0180 52 CD 43 01 CD 23 01 FE 53 C2 7F 01 3E 47 CD 43
0190 01 11 06 02 CD D2 01 0E 14 11 5C 00 CD 05 00 B7
01A0 C2 C9 01 21 80 00 0E 00 16 80 7E CD 43 01 A9 4F
01B0 23 15 C2 AA 01 79 CD 43 01 CD 23 01 FE 42 CA A3
01C0 01 FE 47 CA 97 01 C3 B9 01 11 F4 01 CD D2 01 C3
01D0 00 00 0E 09 C3 05 00 43 6F 6D 6D 61 6E 64 20 45
01E0 72 72 6F 72 24 46 69 6C 65 20 6E 6F 74 20 66 6F
01F0 75 6E 64 24 0D 0A 55 50 4C 4F 41 44 20 43 6F 6D
0200 70 6C 65 74 65 24 55 70 6C 6F 61 64 69 6E 67 2E
0210 2E 2E 24 FE
  
```

4. Enter the following three bytes (a JMP 0163H instruction) at address 0100H:

```
C3 63 01
```

When you have finished, check the data entered. Use the DDT "L" command to list the program and compare it with the displayed UPLOAD listing in this section. Before making any of the following program modifications, exit DDT and save the program by typing in

```
SAVE 2 UPLOAD.COM.
```

and pressing the RETURN key.

5. Modify the UPLOAD program to recognize the RS-232 Serial I/O port on the source computer. Use DDT to write subroutines at the three memory addresses outlined in the following list. Each subroutine must begin at the address specified next to the subroutine. 32 bytes are allocated for each subroutine.

<i>Subroutine</i>	<i>Address</i>
-------------------	----------------

INITSER	0103H
---------	-------

This subroutine initializes the serial port on the source machine (baud rate, data format, etc.). The data format should be set up for eight data bits, one stop bit, no parity, and for compatibility with the Apple Communications Interface board or CCS 7710A Serial Interface board.

<i>Subroutine</i>	<i>Address</i>
-------------------	----------------

RETSER	0123H
--------	-------

If no character is available at the serial port, this subroutine must return A=00. If a byte is available, the routine should read the byte and return it in the A register.

<i>Subroutine</i>	<i>Address</i>
-------------------	----------------

WRSER	0143H
-------	-------

This subroutine outputs a byte to the serial port. You must save all registers including register A.

Once the routines are written into the UPLOAD program, save the program in a disk file with the SAVE command.

6. Connect an RS-232 Serial I/O port cable from the Apple IIe to the source computer. One port must be configured as a send (DTE) device, and the other as a receiver (DCE). The XMIT and RCVE lines (pins 2 and 3) may have to be reversed, or certain "handshaking" lines connected. If you are using a CCS 7710A Serial Interface board, connect pins 4, 6, and 20 together on the Apple port. Ensure that the data formats used by the two serial ports are the same.
7. Once UPLOAD has been loaded into the source computer and the connecting cable is configured properly, you are ready to start transferring data.

On the Apple IIe, type

DOWNLOAD *filespec*

*filespec* is the name of the file to be transferred.

On the source computer, type

UPLOAD *filespec*

As soon as communication is established, the Apple displays

DOWNLOADING

and the source computer displays

UPLOADING...

As each 128-byte record is transferred successfully, a period (.) is displayed on the Apple screen. If an error is detected during the transfer of a 128-byte record, a "B" is printed, and the record is retransmitted.

When the transfer process has been completed, the source computer displays

UPLOAD COMPLETE

and returns to CP/M command level. When this message appears on the source computer, press CONTROL-C. The Apple then displays

DOWNLOAD COMPLETE

and returns to CP/M command level.

8. This step completes the data transfer procedure. Step 7 must be repeated for each file to be transferred. To transfer more than one file at a time, use the SUBMIT transient program to create a batch file. The DOWNLOAD and UPLOAD programs can also be modified for use with serial communication accessory boards other than those listed in "Program Requirements" at the beginning of this appendix. The source listing for UPLOAD and DOWNLOAD is provided for this purpose.

## UPLOAD Source Listing

254

```

017F 3E52      RDYLP      MVI    A,'R'      ;Send 'R' for 'READY'
0181 CD4301    CALL    OUTPUT  ;Send via serial line
0184 CD2301    CALL    INPSTS   ;Response received?
0187 FE53      CPI      'S'      ;'S' for 'SET'
0189 C27F01    JNZ      RDYLP    ;Then try again
018C 3E47      MVI      A,'G'
018E CD4301    CALL    OUTPUT  ;Send to DOWNLOAD
0191 110602    LXI      D,WRKMSG ;Indicate downloading is in
                                ;progress
                                ;
0194 CDD201    CALL    PRMSG    ;
0197 0E14      READ      MVI      C,20    ;Read sequential function
0199 115C00    LXI      D,FCB
019C CD0500    CALL    BDOS      ;Go read 128 bytes
                                ;
019F B7        ORA      A          ;Error?
01A0 C2C901    JNZ      EOF        ;End of file
01A3 218000    TRYAGN:  LXI      H,BUFFER ;Point to the 128 bytes
                                ;
01A6 0300      MVI      C,0        ;Checksum = 0
01A8 1680      MVI      D,80H     ;Byte count = 128
                                ;
01AA 7E        LOOP1    MOV      A,M    ;Get character
01AB CD4301    CALL    OUTPUT  ;Send it (must save A)
01AE A9        XRA      C          ;Calculate checksum
01AF 4F        MOV      C,A        ;Update it
01B0 23        INX      H          ;Point to the next byte
01B1 15        DCR      D          ;Decrement byte count
01B2 C2AA01    JNZ      LOOP1    ;Keep going until all 128 are sent
01B5 79        MOV      A,C        ;Now send checksum byte
01B6 CD4301    CALL    OUTPUT  ;Send it
                                ;
                                ;Wait for verification: 'G'=GOOD, 'B'=BAD
01B9 CD2301    VFYLP:    CALL    INPSTS   ;Get a character
01BC FE42      CPI      'B'        ;A bad read?
01BE CAA301    JZ       TRYAGN    ;Start again
01C1 FE47      CPI      'G'        ;A good read?
01C3 CA9701    JZ       READ      ;Go get next record then
01C6 C3B901    JMP      VFYLP    ;Character must not have been
                                ;ready
                                ;
01C9 11F401    EOF:      LXI      D,DONMSG ;
01CC CDD201    EXIT:    CALL    PRMSG    ;Output message
01CF C30000    JMP      BOOT      ;All finished
                                ;
01D2 0E09      PRMSG:    MVI      C,9    ;Print message function
01D4 C30500    JMP      BDOS
                                ;
01D7 436F6D6D61CMDMSG:  DB      'COMMAND ERRORS$'
01E5 46696C6520FNFMMSG: DB      'FILE NOT FOUND$'
01F4 0D0A55504CDONMSG:  DB      '13,10,UNLOAD COMPLETES$'
0206 55706C6F61WRKMSG:  DB      'UNLOADING....$'
                                ;
0213          END

```

## DOWNLOAD Source Listing

### DOWNLOAD

This program works with an Apple Communications Interface or a CCS 7720A Serial Interface in slot two.

(C) 1980 MICROSOFT

```

0000 =      BOOT      EQU      0000H      ;Boot system
0005 =      BDOS      EQU      0005H      ;BDOS entry point
005C =      FCB        EQU      005CH      ;Default FCB
0080 =      BUFFER     EQU      0080H      ;Default buffer address

E0AE =      COMSTS     EQU      0E0AEH      ;Communication card status
                                           ;location
E0AF =      COMDAT     EQU      0E0AFH      ;Communication card data—slot 2
E000 =      APPKBD     EQU      0E000H      ;Apple keyboard

0100                                ORG      0100H      ;Start at TPA

0100 3A8000  DWNLOD: LDA      BUFFER      ;Make sure there's a filename
0103 B7      ORA      A                  ;Any characters in command line?
0104 11C001  LXI      D,CMDMSG           ;Point to CMD error message
0107 CA8D01  JZ       EXIT               ;Quit
010A 0E13    MVI      C,19               ;Delete file
010C 115C00  LXI      D,FCB
010F D5      PUSH    D                  ;Save PTR: to FCB
0110 CD0500  CALL    BDOS
0113 D1      POP     D                  ;Reget PTR: to FCB
0114 0E16    MVI      C,22               ;Make file
0116 CD0500  CALL    BDOS

0119 3C      INR      A                  ;Check for error
011A 11CE01  LXI      D,NDSMSG           ;Get ready to print
                                           ;'NO DIR. SPACE'
011D CA8D01  JZ       EXIT

                                           ;Wait till UPLOAD sends an 'R'

0120 CDA001  RDYLP:  CALL    RDCOM        ;Get a character from
                                           ;communication card
0123 FE52    CPI      'R'                ;'R' for 'READY'?
0125 C22001  JNZ      RDYLP

0128 1E53    MVI      E,'S'              ;Get 'S' for 'SET'
012A CD9301  CALL    WRCOM

```



012D	CDA001	GETGEE:	CALL	RDCOM	;Wait for 'G' for 'GO'
0130	FE47		CPI	'G'	
0132	C22D01		JNZ	GETGEE	
0135	21F501		LXI	H,WRKMSG	;Point to 'DOWNLDNG' message
0138	7E	PRLP:	MOV	A,M	;Get character
0139	B7		ORA	A	;Set CC's
013A	CA4701		JZ	TRYAGN	;Go do DOWNLOAD
013D	E5		PUSH	H	
013E	5F		MOV	E,A	;Character to register E for
					;CONOUT
013F	CDBB01		CALL	CONOUT	
0142	E1		POP	H	
0143	23		INX	H	
0144	C33801		JMP	PRLP	
0147	218000	TRYAGN:	LXI	H,BUFFER	;Point to 128-byte buffer
014A	0E00		MVI	C,0	;Clear checksum
014C	0E81		MVI	C,81H	;Read 128 bytes + 1 checksum
014E	CDA001	LOOP1	CALL	RDCOM	;Read a byte
0151	77		MOV	M,A	;Store it
0152	A9		XRA	C	;Calculate checksum
0153	4F		MOV	C,A	;Update it
0154	23		INX	H	;Next byte
0155	15		DCR	D	;Decrement byte count
0156	C24E01		JNZ	LOOP1	;Not done—continue
0159	B7		ORA	A	;Was checksum zero?
015A	CA6A01		JZ	GOODRD	;Things are OK
015D	1E42	BADRD:	MVI	E,'B'	; 'B' for 'BAD'
015F	CDBB01		CALL	CONOUT	
0162	1E42		MVI	E,'B'	;Send 'B' to upload
0164	CD9301		CALL	WRCOM	
0167	C34701		JMP	TRYAGN	
016A	1E2E	GOODRD:	MVI	E,.'	;Print a period
016C	CDBB01		CALL	CONOUT	
016F	115C00		LXI	D,FCB	;Point to FCB
0172	0E15		MVI	C,21	;Write sequentially
0174	CD0500		CALL	BDOS	
0177	1E47		MVI	E,'G'	;Send UPLOAD 'G' for 'GOOD'
0179	CD9301		CALL	WRCOM	
017C	C34701		JMP	TRYAGN	
017F	3210E0	DONE:	STA	APPKBD+10H	;Clear keyboard strobe
0182	115C00		LXI	D,FCB	
0185	0E10		MVI	C,16	;Close the file
0187	CD0500		CALL	BDOS	
018A	11E101		LXI	D,DONMSG	;All done message

# Premium SoftCard IIe Programmer's Manual

018D	CDB601	EXIT:	CALL	PRMSG	;Print the message
0190	C30000		JMP	BOOT	
0193	3AA330	WRCOM:	LDA	COMSTS	;Communication card status
0196	E602		ANI	2	;Check bit 2
0198	CA9301		JZ	WRCOM	
0198	7B		MOV	A,E	;Get character to send
019C	32AFE0		STA	COMDAT	;Store here
019F	C9		RET		
01A0	3AAEE0	RDCOM:	LDA	COMSTS	;Communication card status
01A3	1F		RAR		;STS bit to carry
01A4	DAB201		JC	READIT	
01A7	3A00E0		LDA	APPKBD	;See if CONTROL-C was typed
01AA	FE83		CPI	083H	
01AC	CA7F01		JZ	DONE	
01AF	C3A001		JMP	RDCOM	;No, wait for character
01B2	3AAFE0	READIT:	LDA	COMDAT	;Get incoming character
01B5	C9		RET		
01B6	0E09	PRMSG:	MVI	C,9	;Print message
01B8	C30500		JMP	BDOS	
01BB	0E02	CONOUT:	MVI	C,2	;Console output
01BD	C30500		JMP	BDOS	
01C0	436F6D6D61	CMDMSG:	DB	'COMMAND ERRORS'	
01CE	4E6F206469	NDSMSG:	DB	'NO DIRECTORY SPACES'	
01E1	0D0A446F77	DONMSG:	DB	'13,10,DOWNLOADED COMPLETES'	
01F5	446F776E6C	WRKMSG:	DB	'DOWNLOADING',0	
0201			END		

# **Appendix C**

## **SoftCard Version Differences**

SoftCard Enhancements	261
CP/M Implementation Differences	261
SoftCard Differences	262
Differences in Hardware	262
Differences in Software	263
Differences in I/O Operation	264

C

C

C

## SoftCard Enhancements

Because of the Apple I/O interface and dual microprocessor environment, the SoftCard implementation of CP/M has the following enhancements:

- Patch areas in the BIOS for adding user-written I/O driver software

- A screen function interface for modifying the screen attributes for a specific terminal or program

- A character redefinition table for redefining the ASCII characters produced by the keys

- A type-ahead buffer for keyboard entry while CP/M is performing other operations

- A print buffer that allows the printing of a file while performing other operations

- 80-column display operation is standard

- Up to 59.5K bytes of memory for application programs

## CP/M Implementation Differences

The SoftCard version does not include the MOVCPM or the SYSGEN utilities. Because the SoftCard implementation of CP/M is a "fixed" size, there is no need for either utility.

## SoftCard Differences

The Premium SoftCard IIe System has several features that the previous SoftCards do not have. There are also differences in the way the Premium SoftCard IIe System performs I/O functions.

The following features are unique to the Premium SoftCard IIe System.

64K bytes of RAM which can be accessed by either the Z80 or the 6502 microprocessors

Circuitry for a full 80 columns of display

A type-ahead buffer for keyboard entry while CP/M performs other operations

A print buffer that allows printing of a file while CP/M performs other operations

## Differences in Hardware

The SoftCard IIe circuit board contains a Z80B microprocessor which operates three times as fast as those for previous SoftCard circuit boards. The Z80B is not synchronized or phase-locked to the Apple IIe internal clocks.

The Premium SoftCard IIe circuit board is physically larger than the previous SoftCards. It is also designed to install into the AUXILIARY connector slot only.

There are no switches on the Premium SoftCard IIe circuit board.

Z80 interrupts to the 6502 are not allowed; there is no circuit path provided by the Apple IIe.

## Differences in Software

The Premium SoftCard IIe has a larger TPA (59.5K bytes) for running programs.

Because all memory is contained on the Premium SoftCard IIe circuit board, there is no need to change the size of CP/M. Therefore, the SoftCard IIe package does not include the CPM60 utility program.

There is no I/O Configuration Block (IOCB). Some of the SoftCard IIe I/O configuration tables and routines are located in different areas of the BIOS and not in a contiguous block.

The screen menus in the CONFIGIO utility program have been changed.

The DOWNLOAD program is provided on the SoftCard IIe Supplement disk instead of Premium SoftCard IIe Master disk. The Microsoft BASIC Interpreter has been condensed into one file (GBASIC.COM). High-resolution graphic commands are available whenever BASIC is running.

Because of the Z80B microprocessor, programs running under the SoftCard IIe version of CP/M execute three times faster than programs running under previous SoftCards.

## Differences in I/O Operation

The SoftCard IIe uses a method of accessing the I/O system that differs from the previous SoftCards. The SoftCard Z80 microprocessor uses the 6502 as an I/O processor and the Apple memory for I/O communications. Therefore, it is not possible with the SoftCard IIe version of CP/M to directly access Apple I/O memory locations.

---

### *Note*

The previous SoftCards access I/O functions directly through memory-mapped locations in the Apple's memory and do not use the 6502 except to call 6502 subroutines.

---

The SoftCard IIe calls 6502 routines differently than previous SoftCards. The Z80 performs I/O operations through the 6502 microprocessor by accessing a program called the "6502 Basic Input/Output System," or 6502 BIOS. There are 15 separate functions. All are accessed by storing information in a seven-byte area located at 45—4B, and then performing a Z80 CALL instruction to memory location 40. Information from the I/O system is returned to the same seven-byte area.



# Index

---

## 6502 BIOS

- BEEP (call 12), 112
- call example, 24
- calling subroutines, 31
- CALLSUB (call 0), 100
- CLEAR (call 13), 113
- CMDJMP, 95
- CMDONE, 95
- entry points, 99
- FORMAT (call 10), 110
  - general description, 23
  - guidelines for use, 23
- INITSLOT (call 8), 108
- INVERT (call 14), 114
- memory map, 97
- operation, 94
- parameters, 93
- READMEM (call 2), 102
- READSEC (call 3), 103
- READSLOT (call 5), 105
- SETPT1 (call 15), 115
- SETPT2 (call 16), 116
- STATSLOT (call 7), 107
- technical description, 95-98
- UPDATE (call 11), 111
- WRITEMEM (call 1), 101
- WRITESEC (call 4), 104
- WRITESLOT (call 6), 106
- WSTART (call 9), 109

## 8080A

- assembly language, 121
- microprocessor, 22

## 80-column display

- circuitry, 197
- deactivating, 200
- operation, 202

- video display memory, 207, 218

- 80STORE soft switch, 218, 221, 222

- Accessory slots, 192

- Address byte pairs, 229, 230

## Allocation

- blocks, 16
- vector, 75

- ALTZP soft switch, 221, 227

## APDOS

- command line format, 120
- error messages, 236, 237, 239

## Apple

- 48K bank switching, 222

- 80-column operation, 202

- 80STORE soft switch, 218, 221

- ALTZP soft switch, 221

- Applesoft BASIC commands
  - for display modes, 202

## auxiliary memory

- addressing, 208

- description, 207

- subroutines, 229

- switching, 277

## AUXMOVE soft

- switch, 229, 230

- CONTROL key sequences, 205

- copying data between main
  - memory and SoftCard, 229

- cursor symbols, 201

- deactivating 80-column
  - display mode, 200

## display

### mode

- commands, 198

- default, 199

- switching, 216

- pages, 215

- DOS, 120, 197

- double high-resolution

- graphics, 211

- ESCAPE key sequences, 204

- European version
  - differences, ix

**Apple (continued)**

- extended display, 214
- high memory
  - switching, 210, 221
- HIRES soft switch, 221, 225
- I/O device protocols, 192
- memory selection switches, 277
- mixed mode text, 211
- PAGE2 soft switch, 218, 221, 225
- Pascal, 192
- RAMRD soft switch, 221, 225
- RAMWRT soft switch, 221, 225
- SoftCard features, 197
- stack switching, 210, 221
- switching display modes, 198
- text pages, 213
- transferring control to
  - SoftCard memory, 231
- XFER soft switch, 229, 231
- zero page switching, 210, 221

**ASM**

- assembler directives, 122
- command line format, 121
- error messages, 242-245

**Assembly language**

See also ASM

- calling 6502 subroutines, 31
- example, 26-30
- instruction and register
  - differences, 22
- instruction execution times, 22
- programming tools provided, 21
- source program, 121
- using system calls, 23
- Z80/8080 compatibility, 22

**AUTORUN**

- command line format, 123
- error messages, 239

**Auxiliary memory**

- description, 207
- subroutines, 229

**AUXMOVE, 229, 230****BDOS**

- general description, 4
- primitive functions, 7

**BEEP (6502 BIOS call 12), 112****BIOS**

- disk drive byte, 194
- filter routines, 182, 191
- general description, 4
- Hardware Screen Function
  - Table, 165
- I/O configuration, 159
- I/O Vector Table, 184, 185
- keyboard characters
  - definition, 178
- nonstandard devices
  - or software, 182
- screen function
  - interface, 164, 167, 176
  - tables, 165
- Software Screen Function
  - Table, 165
- substitution routines, 182, 190
- user patch areas, 184, 186, 187
- vector patches, 183

**BOOT command line format, 124****Buffered I/O, 34****Calling 6502 subroutines, 31****CALLSUB (6502 BIOS call 0), 100****CAT**

- command line format, 125
- error messages, 239, 243

**CCP (Console Command**

Processor), 5

**Character I/O functions, 7****CLEAR (6502 BIOS call 13), 113****Close File (system call 16), 64****Closing files, 36****Cold start, 123****Command directory, 117**

Compute File Size (system call 35), 85

CON: device, 9, 33, 34, 54

## CONFIGIO

adding I/O software to patch areas, 186

configuring for application programs, 174

configuring for external terminal, 168

configuring screen function interface, 167

initial loading, 162

keyboard character

definition, 178

main selection menu, 162

menu selections, 163

purpose, 161

saving changes, 175

Console buffer, 56, 57

CONSOLE device, See CON: device

Console Input (system call 1), 46

Console Output (system call 2), 47

CONTROL key sequences, 205

## COPY

command line formats, 126

error messages, 237, 239, 246

switch options, 127

## CP/M

allocation vector, 75

APDOS, 120

ASM, 121

BDOS, 4

BIOS, 4, 161

calling from assembly

language program, 25

calling from high-level

language, 31

## CP/M (continued)

CAT, 125

CCP, 5, 31

cold start, 123

CON: device, 9, 33, 34, 54

CONSOLE device, See CON: device

COPY, 126

CRT: device, 10

d:, 129

data disks, 127

DDT, 130, 188

DIR, 134

disk

drive byte, 194

error messages, 233

downloading from other systems, 247

DUMP, 135

ED, 136

ERA, 140

error messages, 233

extents, 16, 17

File Control Block (FCB), 14, 15

file structure, 16

implementation differences, 261

IOBYTE, 9, 12, 13, 35, 54

I/O Vector Table, 184, 185

LIST device, See LST: device

logical device assignments, 9, 35

LPT: device, 11

LST: device, 9, 54

memory organization, 3

nonstandard devices or software, 182

physical device

assignments, 10, 35, 53

description, 10, 11

PIP, 145

## Index

- CP/M (*continued*)
  - primitive functions, 7, 14
  - PTP: device, 11
  - PTR: device, 10
  - PUNCH device, See PUN:  
device
  - PUN: device, 9, 54
  - RDR: device, 9, 54
  - READER device, See RDR:  
device
  - records, 16
  - REN, 149
  - SAVE, 150
  - Slots Type Table, 193
  - SoftCard implementation  
differences, 9
  - STAT, 151
  - SUBMIT, 154
  - system
    - calls, 25, 41
    - disks, 126, 127
    - operation, 7
    - parameters, 5
  - text editing, 136
  - TPA, 5
  - TTY: device, 10
  - TYPE, 156
  - UC1: device, 10
  - UL1: device, 11
  - UP1: device, 11
  - UP2: device, 11
  - UR1: device, 11
  - UR2: device, 11
  - USER, 157
  - XSUB, 158
- Creating files, 35
- CRT: device, 10
- Cursor symbols, 201
- d: command line format, 129
- Datamedia terminals, 164, 169
- DDT
  - command line format, 130
  - commands, 131, 132
  - error messages, 240
  - I/O configuration usage, 188
- Debugging, See DDT
- Delete File (system call 19), 67
- Deleting files, 35
- DIR command line format, 134
- Direct console access system  
calls, 32-34
- Direct Console I/O (system call  
6), 51
- Disk
  - allocation map, 15
  - attributes, 152
  - communication, 14
  - controllers, 194
  - data buffer, 14
  - drive byte, 194
  - error messages, 233
  - I/O functions, 7
  - I/O system calls, 14
  - system error messages, 233
- Display modes
  - 80-column Applesoft BASIC  
commands, 202
  - 80-column operation, 202
  - changing default display  
modes, 199
  - commands, 198
  - CONTROL key sequences, 205
  - cursor symbols, 201
  - ESCAPE key sequences, 204
  - switching, 198
- Display pages, 215
- DMA, 74

**DOWNLOAD**

program description, 250  
requirements, 249

Drive code, 15

**DUMP**

command line format, 135  
error messages, 243

**ED**

command line format, 136  
editing commands, 137, 138  
error messages, 239, 240

Editing, See ED

ERA command line format, 140

Erasing files, 35, 140

Error messages, 233

ESCAPE key sequences, 204

European Apple, ix

Extent number, 15, 16

External terminal, 168

FCB, See File Control Block

**File**

attributes, 78, 152  
closing, 36  
creating, 35  
deleting, 35  
directories, 134  
opening, 36  
read and write operations, 37  
searching for, 37  
size display, 153  
type, 15

File Control Block, 14, 15

Filename, 15

**Filter**

I/O routine, 191  
routines, 182

FORMAT (6502 BIOS call 10), 110

Format for user written patch  
routines, 187

Get Console Status (system call  
11), 58

Get IOBYTE (system call 7), 52

Get Read/Only Vector (system  
call 29), 77

Graphic displays, 211

Hardware conventions, 192-194

Hardware Screen Function

Table, 165, 167

Hazeltine terminals, 164, 168

High-level languages, 31

HIRES switch, 221, 225

INIT SLOT (6502 BIOS call 8), 108

Interrupts, 31

INVERT (6502 BIOS call 14), 114

**I/O**

communication, 33

configuration, 159

device

assignment calls, 36

protocols, 192

software, 186

IOBYTE, 9-13, 35, 54

I/O Vector Table, 184, 185

Keyboard character definition, 178

Lead-in character, 169, 172

LIST device, See LST: device

List Output (system call 5), 50

**LOAD**

command line format, 141

error messages, 235, 237, 238, 242

Logical device

definition of, 8

device assignment, 32, 33

LPT: device, 11

LST: device, 9, 54

Get Addr Alloc (system call 27), 75

Get Addr Disk Params (system  
call 31), 79

## Index

Make File (system call 22), 70

MFT

- command line format, 142

- error messages, 206, 239

Mixed mode text, 211

Multiple drive systems, 129

Nonstandard

- peripherals, 182, 192

- software, 182

Notation, viii

Open File (system call 15), 62

Opening and closing files, 36

Overflow byte, 15

PAGE2 switch, 218, 221, 225

Parameter block, 25

PATCH

- command line format, 143

- error messages, 239, 246

Peripheral boards, 193

Physical device

- definition, 10

- descriptions, 10, 11

- implementation, 12

PIP

- command line formats, 145

- error messages, 233-237, 241-246

- parameter summary, 147

Portability, 8

Primitive functions, 7

Printer echo, 46

Print String (system call 9), 55

Programming tools, 21

PTP: device, 11, 33

PTR: device, 10, 33

PUNCH device, See PUN: device

Punch Output (system call 4), 49

PUN: device, 9, 54

RAMRD switch, 221, 225

RAMWRT switch, 221, 225

Random

- access, 38

- record number, 15

RDR: device, 9, 54

Read Console Buffer (system call 10), 56

READER device, See RDR: device

Reader Input (system call 10), 48

READMEM (6502 BIOS call

- 2), 102

Read Random (system call 33), 81

READSEC (6502 BIOS call 3), 103

Read Sequential (system call 20), 68

READSLOT (6502 BIOS call

- 5), 105

Record

- count, 15

- definition, 16

REN

- command line format, 149

- error messages, 239, 240

Rename File (system call 23), 71

Reset Disk System (system call 13), 60

Reset Drive (system call 37), 89

Return Current Disk (system call 25), 73

Return Login Vector (system call 24), 72

Return Version Number (system call 12), 59

SAVE

- command line format, 150

- error messages, 243

Screen function

- definition for undefined

- terminals, 164

- descriptions, 165, 166

- interface,

- filter routines, 182

- installing nonstandard

- software, 182

- I/O Vector Table, 184, 185

- keyboard characters

- definition, 178

- Screen function (*continued*)
  - interface (*continued*)
    - nonstandard devices, 182, 192
    - saving changes, 175
    - screen function memory
      - addresses, 176
    - substitution routines, 182
    - user patch areas, 184, 186, 187
    - vector patches, 183
- Search for First (system call 17), 65
- Search for Next (system call 18), 66
- Searching for a file, 37
- Select Disk (system call 14), 61
- Sequential access, 37
- Set DMA Address (system call 26), 39, 74
- Set File Attributes (system call 30), 78
- Set/Get User Code (system call 32), 80
- Set IOBYTE (system call 8), 53
- SETPT1 (6502 BIOS call 15), 115
- SETPT2 (6502 BIOS call 16), 116
- Set Random Record (system call 36), 87
- Single-drive systems, 127, 142
- Single file copy program, 142
- Slots Type Table, 193
- SoftCard
  - 48K bank switching, 222
  - 80-column operation, 202
  - Apple
    - CONTROL key sequences, 205
    - ESCAPE key sequences, 204
  - assembly language
    - programming, 21
  - AUTORUN utility program, 123
  - auxiliary memory, 207, 208
  - BOOT utility program, 124
  - CONFIGIO utility program, 161, 179
  - CP/M
    - enhancements, 261
    - implementation differences, 261
  - SoftCard (*continued*)
    - cursor symbols, 201
    - deactivating 80-column
      - display mode, 200
    - default display mode, 199
    - differences
      - between SoftCards, 262
      - in hardware, 262
      - in I/O operation, 264
      - in software, 263
    - display
      - features, 197
      - mode commands, 198
      - modes, 198
    - DOWNLOAD utility
      - program, 247
    - features under Apple DOS, 197
    - graphic displays, doubling
      - resolution, 211
    - high-resolution graphic
      - jumpers, 212
    - memory
      - addressing, 208
      - available, 207
      - switching, 277
    - PATCH utility program, 143
    - programming tools provided, 21
    - unique features, 262
    - UPLOAD utility program, 247
    - video display memory, 207
- Software Screen Function Table, 165, 167
- Soroc terminals, 164, 168
- Source listings
  - DOWNLOAD, 256
  - UPLOAD, 254
- Startup disks, 123
- STAT
  - attribute settings, 152
  - command line formats, 151
  - error messages, 233, 239, 241, 245, 246
- STATSLOT (6502 BIOS call 7), 107
- SUBMIT
  - command line format, 154
  - error messages, 235, 236, 238, 243

Substitution routines, 182, 190

## System

calls, See System calls

disk, 126

parameters, 5

## System calls

buffered I/O, 34

calling from a high-level

language, 31

calling from an assembly

language program, 25

call numbers, 43

Close File (16), 64

Compute File Size (35), 85

Console calls, 33, 34

Console Input (1), 46

Console Output (2), 47

creating files, 35

definition, 8

Delete File (19), 67

deleting files, 35

direct console device calls, 32-34

Direct Console I/O (6), 51

disk I/O calls, 14

file read and write operations, 37

general description, 8

Get Addr Alloc (27), 75

Get Addr Disk Parms (31), 79

Get Console Status (11), 58

Get IOBYTE (7), 52

Get Read/Only Vector (29), 77

guidelines on use, 23

I/O device

assignment calls, 35

calls, 32

List Output (5), 50

Make File (22), 70

Open File (15), 62

opening and closing files, 36

parameter descriptions, 44

Print String (9), 55

program example, 26

## System calls (*continued*)

Punch Output (4), 49

random access, 38

Read Console Buffer (10), 56

Reader Input (3), 48

Read Random (33), 81

Read Sequential (20), 68

Rename File (23), 71

Reset Disk System (13), 60

Reset Drive (37), 89

Return Current Disk (25), 73

returning control to the CCP, 31

Return Login Vector (24), 72

Return Version Number (12), 59

Search for First (17), 65

Search for Next (18), 66

searching for a file, 37

Select Disk (14), 61

sequential access, 37

Set DMA Address (26), 39, 74

Set File Attributes (30), 78

Set/Get User Code (32), 80

Set IOBYTE (8), 53

Set Random Record (36), 87

System Reset (0), 45

Write Protect Disk (28), 76

Write Random (34), 83

Write Random With Zero Fill

(40), 90

Write Sequential (21), 69

System Reset (system call 0), 45

## Text

editor, 136

pages, 213

TPA (Transient Program Area), 5

Transferring control to SoftCard

memory, 231

TTY: device, 10, 33

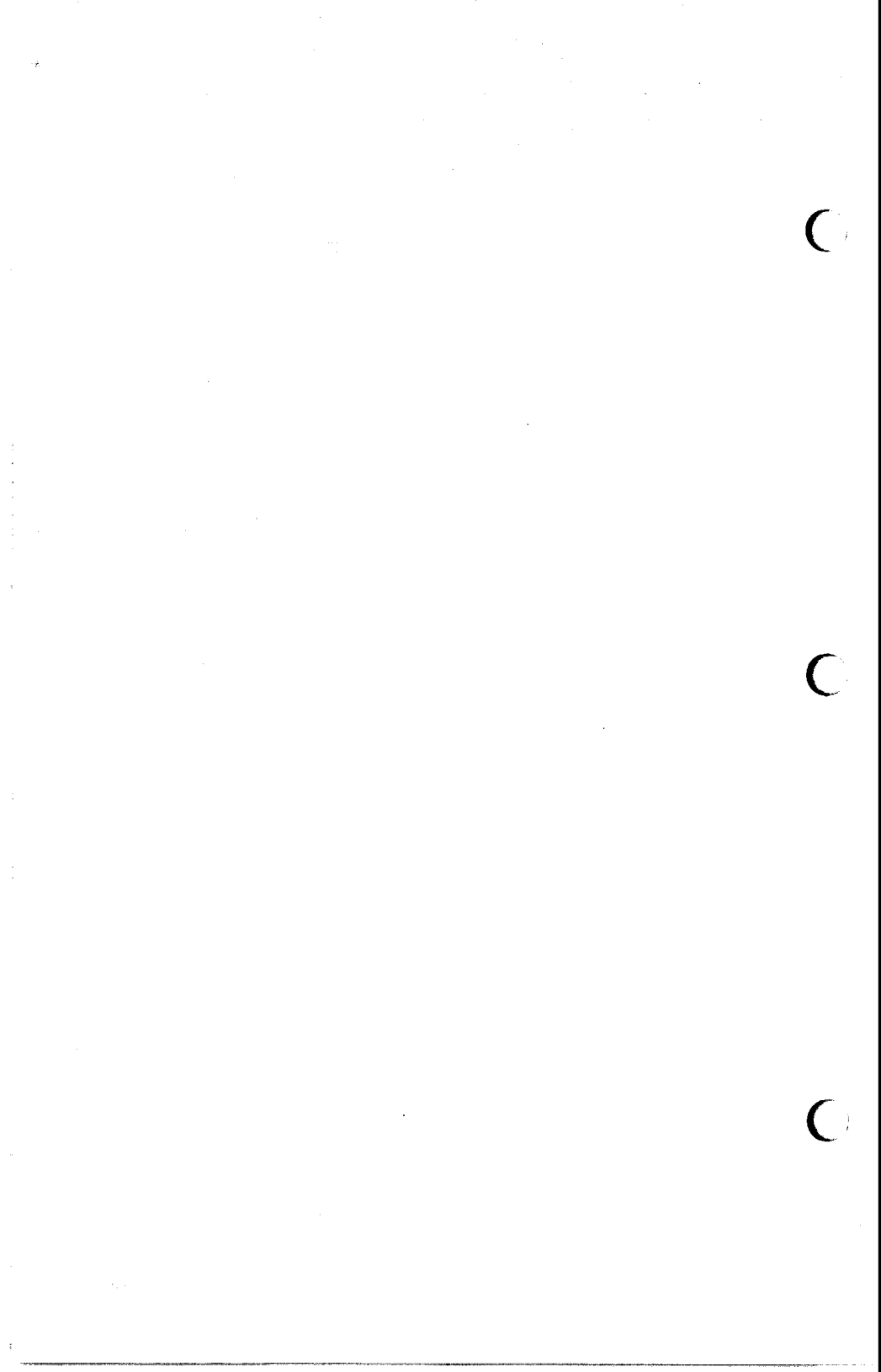
## TYPE

command line format, 156

error messages, 244



- UC1: device, 10
- UL1: device, 11
- UP1: device, 11
- UP2: device, 11
- UPDATE (6502 BIOS call 11), 111
- UPLOAD
  - error messages, 236, 239
  - program description, 254
  - requirements, 249
- UR1: device, 11
- UR2: device, 11
- USER
  - command line format, 157
  - error messages, 242
- User
  - I/O software, 163, 182
  - patch areas, 184
- Utility programs, 119
  
- Vector patches, 183, 184
- Video display
  - boards (80-column), 182
  - display memory, 207
  
- Warm start, 45
- Word processor, 136
- WRITEMEM (6502 call 1), 101
- Write Protect Disk (system call 28), 76
- Write Random (system call 34), 83
- Write Random With Zero Fill (system call 40), 90
- WRITESEC (6502 BIOS call 4), 104
- Write Sequential (system call 21), 69
- WRITESLOT (6502 BIOS call 6), 106
- WSTART (6502 BIOS call 9), 109
  
- XFER, 229, 231
- XSUB command line format, 158
  
- Z80 microprocessor, 22, 262



**MICROSOFT**<sup>TM</sup>

10700 Northup Way, Bellevue, WA 98004

**Software  
Problem Report**

Name \_\_\_\_\_

Street \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_ Date \_\_\_\_\_

**Instructions**

Use this form to report software bugs, documentation errors, or suggested enhancements. Mail the form to Microsoft.

**Category**

\_\_\_\_\_ Software Problem

\_\_\_\_\_ Documentation Problem  
(Document # \_\_\_\_\_)

\_\_\_\_\_ Software Enhancement

\_\_\_\_\_ Other

**Software Description**

Microsoft Product \_\_\_\_\_

Rev. \_\_\_\_\_ Registration # \_\_\_\_\_

Operating System \_\_\_\_\_

Rev. \_\_\_\_\_ Supplier \_\_\_\_\_

Other Software Used \_\_\_\_\_

Rev. \_\_\_\_\_ Supplier \_\_\_\_\_

**Hardware Description**

Manufacturer \_\_\_\_\_ CPU \_\_\_\_\_ Memory \_\_\_\_\_ KB

Disk Size \_\_\_\_\_ " Density: \_\_\_\_\_ Sides: \_\_\_\_\_

Single \_\_\_\_\_ Single \_\_\_\_\_

Double \_\_\_\_\_ Double \_\_\_\_\_

Peripherals \_\_\_\_\_

## Problem Description

---

Describe the problem. (Also describe how to reproduce it, and your diagnosis and suggested correction.) Attach a listing if available.

