.TITLE SHEP,’APPLE DOS’

\* 6.3 10-6-78

\* 8 BIT ASSEMBLER

 .M6502

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* (C) COPYRIGHT 1978 APPLE COMPUTER, INC

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ORG1 EQU $1B00

ORG2 EQU $3600

DISKIO EQU $3D00

ASC1 EQU $3800

AEC1 EQU $3ABF

ASC2 EQU $3D00

AEC2 EQU $3FFF

EDOS EQU $4000

 PAGE

 ORG ORG1

BEQIN JMP DBINIT

;

DOSREL

;

; GET RELOCATION PARMS

;

DR0

LOC1 EQU $26

 LDA #$BF ; START AT BF00

 STA ZPGWRK+1 ; TO LOOK FOR

 LDX #0 ; HIGH RAM

 STX ZPGWRK

 LDY #0 ;APPLE TEST

DR1B

 LDA (ZPGWRK,X)

 STA LOC1

DR1 TYA

 EOR LOC1

 STA LOC1

 TYA

 EOR (ZPGWRK,X)

 STA (ZPGWRK,X)

 CMP LOC1

 BNE DR1A

 INY

 BNE DR1B

 BEQ DR2 ; BR IF TOOK

DR1A

 DEC ZPGWRK+1 ; NOT RAM

 BNE DR1 ; TRY NEXT PAGE

;

DR2

; LDY ZPGWRK+1

 INY ; NEW END OF DOS

 STY NEPAGE

 SEC

 TYA

 SBC DOSLNG ; MINUS DOS LENGTH

 STA NSPAGE ; IS NEW START OF DOS

 SEC

 SBC RSPAGE ; MINUS OLD DOS START

 BEQ BEQIN ; (BREIF NO DELTA)

 STA DELTA ; IS DELTA

 PAGE

 LDA RSPAGE ; RESET START PAGE TO NORMAL

 STA ASTART+1

;

 LDA #DBINIT/256 ;RESET PI RTN TO NORMAL

 STA DI3+2

 LDA #DBINIT&255

 STA DI3+1

;

;

 PAGE

;

;

; RELOCATE ADR TABLES

;

 LDX #0

 STX ZPGWRK

DR3

 LDA ADRTAB+1,X

 TAY

 LDA ADRTAB+2,X

 STA ZPGWRK+1

 JMP DR5

;

DR4

 CLC

 LDA (ZPGWRK),Y

 ADC DELTA

 STA (ZPGWRK),Y

 INY

 BNE DR5

 INC ZPGWRK+1

DR5 INY

 BNE DR6

 INC ZPGWRK+1

;

DR6

 LDA ZPGWRK+1

 CMP ADRTAB+4,X

 BCC DR4

 TYA

 CMP ADRTAB+3,X

 BCC DR4

;

 TXA

 CLC

 ADC #4

 TAX

 CPX ADRTAB

 BCC DR3

 PAGE

;

; RELOCATE CODE

;

 LDX #0

DR7 STX TEMP1

;

 LDA CDETAB+1,X ; GET A START OF CODE ADR

 STA ZPGWRK ; PUT IN ZPG

 LDA CDETAB+2,X

 STA ZPGWRK+1

;

DR8 LDX #0

 LDA (ZPGWRK,X) ; GET OP CODE

 JSR INSDS2 ; GO FIND OUT HOW LONG

;

 LDY LENGTH ; GET HOW LONG

 CPY #2 ; IF IT AIN’T

 BNE DR9 ; 3 THEN DON’T RELOC

 LDA (ZPGWRK),Y ; GET PAGE FROM INST

 CMP RSPAGE ; IF PAGE < REL START

 BCC DR9 ; THE IGNOR

 CMP REPAGE ; IF PAGE >= REL END

 BCS DR9 ; THEN IGNORE

 ADC DELTA ; ELSE ADD DELTA

 STA (ZPGWRK),Y ; TO RELOCATE

;

DR9 SEC

 LDA LENGTH ; ADD LENGTH

 ADC ZPGWRK ; TO PC

 STA ZPGWRK

 LDA #0

 ADC ZPGWRK+1

 STA ZPGWRK+1

;

 LDX TEMP1 ; CHECK FOR END

 CMP CDETAB+4,X ; OF CODE SEGMENT

 BCC DR8 ; BR NOT END

 LDA ZPGWRK

 CMP CDETAB+3,X

 BCC DR8 ; BR NOT END

;

 TXA

 CLC

 ADC #4 ; INCREMENT TABLE INDEX

 TAX

 CPX CDETAB ; DONE

 BCC DR7 ; BR IF NOT

;

 PAGE

;

; MOVE TO RELOCATED CODE

;

 LDA #DEPAGE-1

 STA ZPGWRK+1 ; ZPGWRK=FROM

 LDY NEPAGE

 DEY

 STY ZPGFCB+1 ; ZPGFCB = TOO

 LDA #0

 STA ZPGWRK

 STA ZPGFCB

 TAY

;

DR10 LDA (ZPGWRK),Y ; BYTE FROM

 STA (ZPGFCB),Y ; BYTE TO

 INY ; INCREMENT

 BNE DR10 ; BR NOT FULL PAGE

 DEC DPGCNT ; DECREMENT PAGE CNT

 BEQ DR11 ; BR IF DONE

 DEC ZPGWRK+1 ; INC FROM PAGE

 DEC ZPGFCB+1 ; INC TOO PAGE

 BNE DR10 ; MOVE PAGE

;

DR11 JMP DBVECT+3 ; DONE

 PAGE

DEPAGE EQU EDOS/256

DSPAGE EQU START/256

INSDS2 EQU $F88E

LENGTH EQU $2F

ADRTAB DB 9\*4

 DB @@SAT1,@@EAT1

 DB @@RUN,@@RUN+2

 DB @@IBVT+2,@@IBVT+4

 DB @@AS1VT,@@AS1VT+4

 DB @@AS2VT,@@AS2VT+4

 DB @@AS2VT+6,@@AS2VT+8

 DB @@SAT2,@@EAT2

 DB @@BAIOB,@@ADOSLD+2

 DB @@IBDCTP,@@IBDCTP+2

 DB @0,@0

 DB @0,@0

 DB @0,@0

CDETAB

 DB 6\*4

 DB @@SC1,@@EC1

 DB @@SC2,@@EC2

 DB @@SC3,@@EC3

 DB SDP1,@@EDP1

 DB ASC1,@@AEC1

 DB @@ASC2,@@AEC2

 DB @0,@0

;

RSPAGE DB DSPAGE

REPAGE DB DEPAGE

;

NSPAGE DB 0

NEPAGE DB 0

;

DOSLING DB DEPAGE-DSPAGE

;

DELTA DB 0

DPGCNT DB DEPAGE-DSPAGE

 PAGE

 BOUND 256

;

; RELOCATION TABLES

;

START

SAT1

FTAB DB @@\*-45 ;START OF FTABS

CINA DB @@CHRIN ;CHAR IN ADR

COUTA DB @@CHROUT ;CHAR OUT ADR

FN1ADR DB @@FNAME1

FN2ADR DB @@FNAME2

SVBLA DB @@SVBL

ASTART DB @@BEQIN ;CHANGED TO START BY RELOCATE

CCBADR DB @@CCB

;

OUTSVT ;CHAR OUTPUT STATE VECTOR TABLE

 DB @@COS0-1

 DB @@COS1-1

 DB @@COS2-1

 DB @@COS3-1

 DB @@COS4-1

 DB @@COS5-1

 DB @@COS6-1

; COMMAND EXECUTION TABLE

CMDETB

 DB @@EINIT-1

 DB @@ELOAD-1

 DB @@ESAVE-1

 DB @@ERUN-1

 DB @@ECHAIN-1

 DB @@EDEL-1

 DB @@ELOCK-1

 DB @@EUNLK-1

 DB @@ECLOSE-1

 DB @@EREAD-1

 DB @@EEXEC-1

 DB @@EWRITE-1

 DB @@EPOS-1

 DB @@EOPEN-1

 DB @@EAPND-1

 DB @@EREN-1

 DB @@ECAT-1

 DB @@EMON-1

 DB @@ENOMON-1

 DB @@EPR-1

 DB @@EIN-1

 DB @@EMAXF-1

 DB @@EAS-1

 DB @@EINT-1

 DB @@EBSV-1

 DB @@EBLD-1

 DB @@EBRUN-1

 DB @@EVAR-1

EAT1

 PAGE

;

; NON-RELOCATING ADRS

;

IBASVT

CHAIN DB @@IBCHN

RUN DB @@IBRUN

BREAK DB @@IBBRK

GO DB @@IBGO

CONT DB @@IBCONT ;BASIC CONT ENTRY POINT

IBVT DB @@IBCHN,@@IBRUN,@@IBRK

 DB @@IBGO,@@0

IBVTL EQU \*-IBVT

;

AS1VTL DB @@ASRUN1,@@ASRUN1,@@ASBRK1

 DB @@IBGO,@@0

AS1VTL EQU \*-AS1VT

;

AS2VT DB @@ASRUN2,@@ASRUN2,@@ASBRK2

 DB @@DBINIT,@@0

AS2VTL EQU \*-AS2VT

 PAGE

;

; EQUATES REQD TO FIND THINGS IN APPLE II

;

SETVID EQU $FE93

SETKBD EQU $FE89

PROMPT EQU $33 ;PROMPT CHAR

OUTSW EQU $36 ;OUTPUT VECTOR SWITCH

INSW EQU $38 ;INPUT VECTOR SWITCH

ZPGWRK EQU $40 ;ZERO PAGE WORK CELL

CNUM EQU $44 ;CONVERTED NUMERIC

LBUFF EQU $200 ;LINE BUFFER

MULT EQU $FB63 ;MULT ROUTINE

INPRT EQU $FE8B ;SET OUT PORT

OUTPRT EQU $FE95 ;SET OUT PORT

IBCHN EQU $E836 ;BASIC RUN

IBLMEM EQU $4A ;BASIC LOW MEMORY

IBHMEM EQU $4C ;INTEGER BASIC HIMEM

IBSOP EQU $CA ;INTEGER BASIC START OF CGM

IBBRK EQU $3E3 ;BASIC BREAK

IBGO EQU $E000 ;BASIC ENTRY POINT

IBCONT EQU $E003 ;BASIC CONTINUE ENTRY POINT

IBSOV EQU $CC ;BASIC START OF VARIABLES

ASSOP EQU $67 ;AS START OF PROGRAM

ASEOP EQU $AF ;AS END OF PROGRAM

ASEOP2 EQU $69 ;AS END-OF PGM 2

ASHM1 EQU $73 ;AS HIGH MEM 1

ASHM2 EQU $6F ;AS HIGH MEM 2

ASLMEM EQU $ASSOP ;AS LOW MEM

ASBRK1 EQU $D865 ;AS ROM BREAK

ASBRK2 EQU $1067 ;AS RAM BREAK

AITSTL EQU $E000 ;AS 1 IB TEST LOC

ATSTV EQU $4C ;AS TEST VALUE

ITSTV EQU $20 ;IB TEST VALUE

BOOTSL EQU $2E ;BOOT FROM SLOT

ZPQFCB EQU $42 ;ZERO PAGE WORK CELL

HOME EQU $FC58

PRINT EQU $FDED

GETKEY EQU $FD0C

 PAGE

;

; DOS BASIC INTERPRETER – INITIAL ENTRY

;

SC1

DBINIT

 LDA IBSLOT ;GET BOOT SLOT

 LSRA

 LSRA

 LSRA

 LSRA

 STA CS ;SET AS CURRENT SLOT

 LDA IBDRVN ;GET BOOT DRIVE NUMBER

 STA CD ;SET AS CURRENT DRIVE

 LDA AITSTL ;GET APPLESOFT/IB TEST

 EOR #ITSTV ;IF AS THEN

 BNE IAS1 ;GO TO AS INIT

; ;ELSE INIT FOR IB

 STA ASIBSW ;SET SW FOR IB

 LDX #IBVTL ;GET IB VT LENGTH

IIB1 LDA IBVT-1,X ;MOVE IB ADDR

 STA IBASVT-1,X

 DEX

 BNE

 JMP INITAA

;

IAS1

 LDA #$40 ;INDICATE ROM APPLESOFT

 STA ASIBSW

 LDX #AS1VTL

IAS1A LDA AS1VT-1,X ;MOVE ROM AS ADRS

 STA IBASVT-1,X

 DEX

 BNE IAS1A

;

INITAA

 SEC ;INDICATE INIT

 BCS INITA

DBRST

 LDA ASIBSW ;GET AS/IB FLAG

 BNE INITA1 ;BR IF NOT IB

 LDA #ITSTV ;GET IB TEST VALUE

 BNE INITA2 ;GO SET IB

INITA1 ASLA ;TST ROM AS

 BPL INITA3 ;BR IF NOT ROM TEST VALUE

 LDA #ATSTV ;GET AS ROM TEST VALUE

INITA2 JSR SWTST ;GO SET

INITA3

 CLC ;INDICATE RESET

;

INITA

 PHP ;SAVE INIT/RESET

 JSR MVCSW ;GO MOVE CHAR SWITCH

 LDA #MC+MI+MO ;SET MONITOR MODES

 STA MONMOD

;

 LDA #0

 STA OSTATE ;CLEAR OUTSTATE AND EXECUTE STATE

 PLP ;GET INSTATE

 RORA ;SHIFT CARRY TO MSB

 STA ISTATE ;SAVE INSTATE

 BMI INITB ;BR IF INIT

 JMP (CONT) ;GO TO CONTINUE ENTRY

INITB JMP (GO) ;GO TO GO ENTRY

 PAGE

INITC

 ASLA ;OF ISTATE NOT ON

 BPL INITD ;THEN NOT RAM AS

 STA ASIBSW ;SET RAM AS

 LDX #AS2VTL

IAS2A LDA AS2VT-1,X ;MOVE RAM AS ADRS

 STA IBASVT-1.X

 DEX

 BNE IAS2A

 LDX #29

IAS2B LDA FNAME2,X

 STA FNAME1,X

 DEX

 BPL IAS2B

;

INITD

 LDA DFNFTS ;GO BUILD FILE TABS

 STA CNFTBS ;AND SET MEM BOUNDS

 JSR BLDFTB

 LDA ESTATE ;GET EXEC STATE

 BEQ INITZ ;BR IF NOT EXECTUTE

 PHA ;SVE CHAR

 JSR MVEFTA ;GO MOVE EX FILE TAB ADR TO ZP

 PLA ;GET SAVED CHAR

 LDY #0

 STA (ZPGWRK),Y ;

INITZ

 JSR CLRSTS ;SET IN AND OUT STATES TO ZERO

 LDX #IFBL

INITE LDA DBVECT,X ;MOVE RESTART VECTORS

 STA $3D0,X

 DEX

 BPL INITE

 LDA CMDNO ;IF NOT BOOT

 BNE INITF ;THEN DONE

 LDA FNAME1 ;IF FN1

 EOR #$A0 ;NOT DONE

 BEQ INITF ;THEN DONE

 JMP ERUN ;ELSE

;

IFB

INITF

 LDA SVCMD

 BEQ INITG

 STA CMDNO

 JMP CMDGO

INITG

 JMP ORTN

;

DBVECT JMP DBRST

 JMP DBINIT

 JMP DOSENT

 JMP DISKIO

CCBLDR

 LDA CCBADR+1

 LDY CCBADR

 RTS

IOBLDR

 LDA AIOB+1

 LDY AIOB

 RTS

IFBL EQU \*-IFB-1

 PAGE

;

; CHRIN – CHAR RCVD VIA IN SWITCH

;

CHRIN

 JSR SVREGS

 LDA ISTATE ;IF NOT DISKIN

 BEQ CHIN1 ;THEN BRANCH, ELSE

 BPL CHINO

 JMP INITC

CHINO

 LDA SVA

 STA ($28),Y

 JMP ICFD ;AND GET CHAR FROM DISK

CHIN1

 LDA ESTATE

 BEQ CHIN2

 JMP NXTEXC

CHIN2

 LDA #3 ;SET OUT CHAR

 STA OSTATE ;STATE TO INPUT ECHO

 JSR LDREGS

 JSR GETIN

 STA SVA

 JMP ORTN

;

GETIN JMP (INSW)

;

; CHROUT – CHAR RCVD VIA OUTPUT SWITCH

;

CHROUT

 JSR SVREGS ;SAVE REGS

 LDA OSTATE ;GET OUT SPARE

 ASLA

TAX
LDA OUTSVT+1,X ;GET ROUTINE ADR

PHA

LDA OUTSVT,X

PHA

LDA SVA

RTS ;GO TO ROUTINE

;

; SVREGS – SAVE REGS WHILE PROCESSING CHARS

;

SVREGS

 STA SVA ;SAVE ACU

SVRGSA

 STX SVX ;SAVE X

 STY SVY ;SAVE Y

 LDX #3 ;SET FOR FOUR BYTE MOVE

SVRB LDA SVOUTS,X ;MOVE SAVED OUT AND IN SW

 STA OUTSW,X ;TO APPLE OUT/IN SW

 DEX

 BPL SVRB

 RTS ;DONE

 PAGE

;

; COS0 – 1ST CHAR OF PRINTED OUTPUT LINE

; CHECK FOR CNTL-D

;

COS0

 LDX ISTATE ;IS IN STATE NOT ZERO

 BEQ COS01

 CMP #’?+$80 ;THEN IS THIS?

 BEQ COS6 ;THEN PRINT ONLY IF MONITOR

 CMP PROMPT

 BEQ COS6

COS01

 LDX #2

 STX OSTATE

 CMP CCHAR ;IF NOT CNTL-D

 BNE COS2 ;THEN GO TO STATE 2

 DEX

 STX OSTATE ;ELSE STATE = 1

 DEX

 STX LBUFD ;AND LBUFD=0

;

; COS1 – ACCUMULATE CMD FROM PRINTED OUTPUT

;

COS1

 LDX LBUFD ;GET LINE BUFF DISPL

COS1A STA LBUFF,X ;PUT CHAR IN BUFF

 INX ;INCR PTR

 STX LBUFD ;SAVE PTR

 CMP #$8D ;WAS THIS A CR

 BNE CMDRTN ;IF NOT THEN PR CHAR

;

 JMP SCNCMD ;GO SCAN COMMAND

;

; COS2 – PRINTED OUTPUT, NOT FIRST CHAR

;

COS2

 CMP #$8D ;IS IT A CR

 BNE PRRTN ;BR IF NOT

 LDX #0 ;SET FOR POSSIBLE C-D NEXT

 STX OSTATE ;NEXT STATE

 JMP PRRTN ;GO PRINT CHAR

 PAGE

;

; COS3 – KEY IN ECHO PRINT

;

COS3

 LDX #0

 STX OSTATE ;RESET OUT STATE

 CMP #$8D ;IS IT CR

 BEQ COS3A ;IS CR THEN CMD CHECK

COS3B

 LDA ESTATE ;ELSE IF NOT EXECUTE

 BEQ PRRTN ;THEN PRINT CHAR

 BNE DRTNI ;ELSE PRINT IF MON INPUT

COS3A

;

 JSR TSTRUN

 BCS COS3B

 LDX SVX ;GET LINE INDEX

 JMP COS1A

;

; COS4 – DISK OUTPUT MODE

;

COS4

 CMP #$8D ;IS IT CR

 BNE COS4A ;BR IF NOT CR
 LDA #5 ;SET STATE FOR CNTL-D

 STA OSTATE ;EXAMINE

COS4A JSR OCTD ;GO OUTPUT CHAR TO DISK

 JMP DRTNO ;GO TO DATA RETURN (OUT)

;

; COS5 – DISK OUTPUT MODE – 1ST CHAR OF A LINE

;

COS5

 CMP CCHAR ;IS IT CNTL D

 BEQ COS0 ;BR IF CNTL– D

 CMP #$8A ;LINE FEED?

 BEQ COS4A

 LDX #4

 STX OSTATE ;SET NEW OUT STATE

 BNE COS4 ;BR IF NOT CNTL D

;

; COS6 – DISK INPUT ECHO

;

COS6 LDA #0

 STA OSTATE ;RESET OUT STATE = 0

 BEQ DRTNI ;GO TO DATA IN RETURN

 PAGE

;

; PRRTN – PRINT CHAR RETURN

;

;

; CMDRTN – PRINT CHAR IF MONITOR CMBS MODE

; DRTNO – PRINT CHAR IF MONITOR DATA OUT

; DRTNI – PRINT CHAR IF MONITOR DATA IN

;

CERTN

 LDA LBUFF ;CHECK FOR PRINTED COMMAND

 CMP CCHAR

 BEQ CMDRTN ;IF PC THEN NO RESET X REG

 LDA #$A0 ;BLANK

 STA LBUFF

 LDA #$8D ;PLUS CR

 STA LBUFF+1 ;TO OUT BUFFER

 LDX #0 ;RESET TO SOL

 STX SVX

CMDRTN LDA #MC

 BNE MODECK

DRTNO LDA #MO

 BNE MODECK

DRTNI LDA #MI

;

MODECK

 AND MONMOD ;AND WITH MODE

 BEQ ORTN ;BR IF NOT PRINT

PRRTN JSR LDREGS

 JSR ORTN1

 STA SVA

 STY SVY

 STX SVX

;

ORTN

 JSR MVCSW ;GO MOVE CHAR I/O SWITCH

LDREGS

 LDA SVA ;ACU

 LDY SVY ;Y

 LDX SVX ;X

 SEC

 RTS ;BY PASS PRINT

;

ORTN1 JMP (OUTSW)

;

; PRCRIF – PRINT CR IF MON CMDS

;

PRCRIF

 BIT MONMOD ;IF NOT MON CMDS

 BVC PRCIFR ;THEN RETURN

 LDA #$8D ;ELSE PRINT CR

 JSR ORTN1

PRCIFR RTS

 PAGE

;

; SCNCMD – SCAN COMMAND

;

SCNCMD

 LDY #$FF

 STY CMDNO ;RESET COMMAND NUMBER

 INY

 STY SVCMD

SC0

 INC CMDNO ;INCR CMD NO

 LDX #0 ;RESET LINE INDEX TO 0

 PHP ;SAVE EQ STATUS

 LDA LBUFF,X ;GET 1ST LINE CHAR

 CMP CCHAR ;IS IT CONTROL D

 BNE SC0A ;BR /IF NOT

 INX ;INCR OVER CNTLD
SC0A STX LBUFD

;

SC1X

 JSR GNBC ;GET NON BLANK INPUT CHAR

 AND #$7F ;MSB OF CHAR OFF

 EOR CMDNTB.Y ;EOR WITH INPUT

 INY ;INCREMENT TABLE INDEX

 ASLA ;IF MSB OF EOR RESULT ON

 BEQ SC1A ;IF RESULT NOT NOW ZERO

 PLA ;THEN INPUT DOES NOT

 PHP ;EQUAL ENTRY

SC1A BCC SC1X ;LOOP FOR END ENTRY

;

 PLP ;IF INPUT EQUALS END

 BEQ SYNTAX ;THEN GO SYNTAX

;

 LDA CMDNTB,Y ;IF NEXT TABLE CHAR NOT ZERO

 BNE SC0 ;THEN SCAN THE NEXT TABLE ENTRY

CNF ;COMMAND NOT FOUND

 LDA LBUFF ;LINE IS A CONTROL-D

 CMP CCHAR ;THEN THIS IS A

 BEQ CNF1 ;POSSIBLE SYNTAX ERROR, ELSE

 JMP PRRTN ;IT’S A BASIC INPUT LINE

CNF1

 LDA LBUFF+1 ;GET NEXT CHAR

 CMP #$8D ;IS IT A CR

 BNE CSERR ;BR IF CR

 JSR CLRSTS ;CLEAR THE STATES

 JMP CMDTRN ;CNTL-D ONLY

;

CSERR JMP ESYNTX

 PAGE

;

; SYNTAX – FIGURE OUT WHAT WE GOT HERE

;

SYNTAX

 LDA CMDNO ;CMDNO=CMDNO\*2

 ASLA

 STA CMDNO

;

 TAY

 LDA #FN1

 AND CMDSTB,Y ;IS FN1 REGD

 BEQ SN10 ;BR IF NOT

 JSR CLRFNS

 PHP ;SAVE EG STATUS

;

SN2

 JSR GNBC ;GET NON BLANK CHAR

 BEQ SN6 ;BR IF CR OR COMMA

 ASLA ;TEST FOR ALPHA

 BCC SN2A ;BR IF ALPHA

 BMI SN2A ;BR IS ALPHA

 JMP CNF ;LURCH IF NOT ALPHA

SN2A RORA ;RESTORE BITS

 JMP SN4 ;AWAY WE GO

SN3 JSB GNXTC ;GO GET NEXT CHAR

 BEQ SN6 ;BR IF COMMA OR CHAR

SN4 STA FNAME1,Y ;PUT INTO ILENAME

 INY ;INC FN INDEX

 CPY #60 ;ATFN FN INDEX

 BCC SN3 ;BR IF NOT

SN5 JSR GNXTC ;LOOP UNTIL CR OR COMMA

 BEQ SN5

;

SN6 PLP ;WAS THIS FN2 L OO

 BNE SN7 ;BR IF IT WAS

;

 LDY CMDNO

 LDA #FN2

 AND CMDSTB,Y ;IF FN2 NOT REGD THEN

 BEQ SN8 ;BRANCH

;

 LDY #30 ;SET FN2 INDEX

 PHP ;INDICATE FN2 SEEK

 BNE SN2 ;GO LOOK FOR FN2

;

SN7 LDA FNAME2 ;IF 1ST CHAR OF

 CMP #$A0 ;FN2 IS BLANK THEN

 BEQ SERR1 ;SYNTAX ERROR

;

SN8 LDA FNAME1 ;IF 1ST CHAR OF

 CMP #$A0 ;FN1 IS NOT BLANK

 BNE SOPTS ;THEN GO LOOK FOR OPTIONS

;

 LDY CMDNO

 LDA #NPB+NPE ;IF CMD MUST HAVE FILENAME

 AND CMDSTB,Y ;THEN

 BEQ SERR1 ;THIS IS ERROR ELSE

;

 BPL SOPTS ;ITS EXECUTABLE WITHOUT

;

SERR1 JMP CNF

;

CLRFNS

 LDA #0

 LDY #60

CLRFNA

 LDA #$A0

SN1 STA FNAME1-1,Y ;CLEAR FN1, FN2

 DEY

 BNE SN1

 RTS

 PAGE

SN10 ;FILE NAMES NOT REGD

 STA FNAME1

 LDA #NUM1+NUM2 ;IF NEITHER NUM1

 AND CMDSTB,Y ;OR NUM2 IS REGD

 BEQ SOPTS ;THEN GO LOOK AT OPTIONS

;

 JSR GETNUM ;GO GET NUMERICS

 BCS SERR1

;

 TAY ;IF HIGH DIGIT NOT

 BNE SERR1 ;ZERO THEN BAD

;

 CPX #17 ;IF LOW DIGIT GT 16

 BCS SERR1 ;THEN BAD

;

 LDY CMDNO

 LDA #NUM1

 AND CMDSTB,Y ;IF WE WANT NUM2

 BEQ SN11

;

 CPX #8 ;IF NUM2>1

 BCS SERR1 ;THEN ERROR, ELSE

 BCC SOPTS ;GO SCAN OPTIONS

;

SN11
 TXA ;IF NUM1=0

 BEQ SERR1 ;THEN ERROR, ELSE

;

 PAGE

;

; SOPTS – LOOK FOR SYNTAX OPTIONS

;

SOPTS

 LDA #0

 STA INOPTS ;CLEAR INPUT OPTIONS

 STA IMBITS

 STA CL

 STA CL+1

 LDA LBUFD ;SET PASS 1

 STA TEMP1A

;

SP1 JSR GNBC ;GO GET NON-BLANK CHAR

 BNE SP2 ;BR IF NOT COMMA OR CR

 CMP #$8D ;IF CHAR IS COMMA

 BNE SP1 ;THEN GO GET CHAR

;

 LDX CMDNO ;OPTIONS INPUT = I

 LDA INOPTS ;ALLOW OPTS = A

 ORA CMDSTB+1,X ;IF (A OR I)
 EOR CMDSTB+1,X ;XOR A NOT = 0 THEN

 BNE SERR1 ;WE HAVE UNALLOWED OPTIONS

;

 LDX TEMP1A ;IF THIS IS PASS 2

 BEQ CMDGO ;THEN DONE,

 STA TEMP1A ;ELSE SET PASS

 STX LBUFD ;RESTORE LBUFD AND

 BNE SP1 ;GO DO PASS 2

;

SP2 LDX #OPT1L ;COMPARE CHAR HAVE WITH

SP3 CMP OPTAB1-1,X ;CHARS IN OPT TABLE

 BEQ SP4 ;IF NOT FOUND CONTINUE

 DEX

 BNE SP3 ;IF NOT FOUND

 BEQ SERR2 ;THEN SYNTAX ERROR

;

SP4 LDA OPTAB2-1,X ;IF CORRESPONDING OP TAB 2 IS

 BMI SP8 ;MINUS THEN IT MONITOR BITS

 ORA INOPTS

 STA INOPTS

 DEX

;

 STX TEMP2A ;ELSE A NUMERIC MUST FOLLOW

 JSR GETNUM ;FOLLOW

 BCS SERR2

;

 LDA TEMP2A ;GET IOTION NUMBER

 ASLA ;MULT BY 4

 ASLA

 TAY

;

 LDA CNUM+1 ;IF RESULT NUM HI IS

 BNE SP5 ;GT 0, THEN GT LOW RANGE

 LDA CNUM ;TEST RESULT LOW

 CMP OPTAB3,Y ;WITH LOW RANGE (LOW)

 BCC SERR2 ;BR IF RESULT < LR

 LDA CNUM+1

SP5 CMP OPTAB3+3,Y

 BCC SP6 ;BR IF LESS

 BNE SERR2 ;BR IF GRREATER

 LDA CNUM

 CMP OPTAB3+2,Y

 BCC SP6 ;BR IF LESS

 BNE SERR2 ;BR IF GREATER

;

SP6 LDA TEMP1A ;IF PASS 1,THEN

 BNE SP1 ;DON’T STORE RESULT

 TYA

 LSRA

 TAY

;

 LDA CNUM+1 ;STORE THE RESULT

 STA CUROPT+1,Y

 LDA CNUM

 STA CUROPT,Y

SP7 JMP SP1 ;GO FOR NEXT OPT

;

SP8 ;MONITOR REG

 PHA ;SAVE TYPE REG

 LDA #CIO ;SET OPTION OF CIO

 ORA INOPTS

 STA INOPTS

 PLA ;RESTORE REG

 AND #$7F ;CLEAR CIO

 ORA IMBITS ;OR WITH PREV IMBITS

 STA IMBITS

 BNE SP7 ;GO FOR NEXT

;

SERR2 JMP CNF

 PAGE

;

; CMDGO – EXECUTE COMMAND

;

CMDGO

 JSR CLRSTS

 JSR CLRCCB ;GO CLEAR CCB

 JSR ECMD ;GO EXECUTE

 JMP CERTN

ECMD

 LDA CMDNO ;COMMAND NO

 TAX ;IS CMD EXEC TAB INDEX

 LDA CMDETB+1,X ;GET CMD ADR

 PHA ;ONTO STACK

 LDA CMDETB,X

 PHA

 RTS ;AND GOTO COMMAND

;

; GNXTC – GET NEXT CHAR

;

GNXTC

 LDX LBUFD

 LDA LBUFF,X ;GET NEXT CHAR AND IF

 CMP #$8D ;IT IS A CR

 BEQ GNXTCR ;THEN RETURN WITHOUT

 INX ;INCR TO NEXT CHAR

 STX LBUFD

 CMP #’,+$80 ;TEST FOR COMMA

GNTXR RTS

;

; GNBC – GET NON BLANK CHAR

;

GNBC

 JSR GNXTC ;GO GET NEXT CHAR

 BEQ GNXTCR ;BR IF COMMA OR CR

 CMP #$A0 ;IS IT BLANK

 BEQ GNBC ;BR IF BLANK

 RTS ;DONE

;

; CLRCCB – CLEAR CCB

;

CLRCCB

 LDA #0

 LDY #CCBLEN ;CCBLENGTH

CLC1 STA CCB-1,Y ;CLEAR BYTE

 DEY

 BNE CLC1

 RTS

 PAGE

;

; GETNUM – COVERT ASCII INPUT TO NUMERIC

;

GETNUM

 LDA #0 ;CLEAR WORK AREA

 STA CNUM

 STA CNUM+1

 JSR GNBC

 PHP

 CMP #$A4

 BEQ HEXNUM

 PLP

 JMP GN2A

;

GN2 JSR GNBC ;GET NEXT NON BLANK

GN2A

 BNE GN3 ;BR NOT COMMA OR CR

 LDX CNUM ;X=RESULT LOW

 LDA CNUM+1 ;Y=RESULT HI

 CLC

 RTS ;DONE

;

GN3 SEC

 SBC #$B0 ;SUBTRACT ASCII 0

 BMI GN4 ;BR IF NOT NUM

 CMP #10

 BCS GN4 ;BR IF NOT NUM

 JSR GN5 ;OLD\*2

 ADC CNUM ;PLUS NEW

 TAX

 LDA #0

 ADC CNUM+1

 TAY

 JSR GN5 ;OLD\*4

 JSR GN5 ;OLD\*8

 TXA ;OLD\*8 + OLD\*2 + NEW

 ADC CNUM

 STA CNUM ;=OLD\*10 + NEW

 TYA

 ADC CNUM+1

 STA CNUM+1

 BCC GN2

;

GN4

 SEC

 RTS ;DONE

GN5

 ASL CNUM ;CNUM \* 2

 ROL CNUM+1

 BCS GN4

 RTS

 PAGE

;

HEXNUM

 PLP

HNO

 JSR GNBC ;GO GET CHAR

 BEQ GN2A ;BR IF CR OR COMMA

;

 SEC

 SBC #$B0 ;CHAR – ASCII0

 BMI GN4 ;BR IF LTO

 CMP #10 ;IS IT LT10

 BCC HN1 ;BR IF LT

 SBC #$7 ;SUB 7 FOR ASCII A

 BMI GN4 ;BR IF LT A

 CMP #16 ;TEST GT 15

 BCS GN4 ;BR GT 15

HN1 JSR GN5 ;OLD\*2

 JSR GN5 ;OLD\*4

 JSR GN5 ;OLD\*8

 JSR GN5 ;OLD\*16

 ORA CNUM ;OR IN NEW

 STA CNUM ;SAVE NEW

 JMP HN0 ;GO FOR NEXT CHAR

 PAGE

;

; EPR – EXECUTE PR#

;

EPR

 LDA CNUM ;GET PORT

 JMP OUTPRT ;GO DO IT

;

; EIN – EXECUTE IN#

;

EIN

 LDA CNUM ;GET PORT

 JMP INPRT ;GO DO IT

;

; EMON – EXECUTE MONITOR CMD

;

EMON

 LDA MONMOD ;GET CURRETN BITS

 ORA IMBITS ;OR IN NEW BITS

 STA MONMOD ;SET NEW MODE

 RTS

;

; ENOMON – EXECUTE NO MONITOR CMD

;

ENOMON

 BIT IMBITS

 BVC ENM1

 JSR PRCRIF

ENM1

 LDA #$70

 EOR IMBITS ;INVERT INPUT BITS

 AND MONMOD ;AND WITH CURRENT

 STA MONMOD ;SET NEW MODE

 RTS

 PAGE

;

; EMAXF – EXECUTE MAX FILES

;

EMAXF

 LDA #0 ;RESET EXECUTE

 STA ESTATE

 LDA CNUM ;SAVE NEW NO FILES

 PHA

 JSR CLALL ;GO CLOSE ALL TBLS

 PLA

 STA CNFTBS ;SET NEW NO FILE TBLS

 JMP BLDFTB ;GO BUILD NEW ONES

;

; EDEL – DELETE A FILE

;

EDEL

 LDA #CRGDEL ;DELETE REQUEST

 JSR OPEN ;GO OPEN

 JSR FILSRC ;FIND FILE

 LDY #0

 TYA

 STA (ZPGWRK),Y ;RESET FN

 RTS

;

; ELOCK – LOCK A FILE

;

ELOCK

 LDA #CRGLCK ;SET LOCK

 BNE ELGO

;

; EUNLK – UNLOCK A FILE

;

EUNLK

 LDA #CRGUNL ;SET UNLOCK

ELGO

 JSR OPEN ;OPEN FILE & UNLOCK

 JSR TSTFNF

 JMP ECLOSE

;

; EVAR – VERIFY A FILE

;

EVAR

 LDA #CRGVAR ;SET VARIFY

 BNE ELGO

 PAGE

;

; EREN – RENAME A FILE

;

EREN

 LDA FN2ADR ;MOVE FILE NAME2

 STA CCBFN2

 LDA FN2ADR+1

 STA CCBFN2+1

 LDA #CRGRNM

 STA TEMP1A ;SET RENAME

 JSR EO3 ;GO OPEN AND RENAME

 JMP ECLOSE ;GO CLOSE

;

; EAPND – OPEN FILE FOR APPEND

;

EAPND

 JSR EOPEN ;GO OPEN

 LDA #CREFNF

 CMP CCBSTA ;IF FILE CREATED

 BNE AP1

 RTS

AP1

 JSR RBYTE ;READ A BYTE

 BNE AP1 ;BR IF NOT ZERO

;

 JMP RWP3 ;GO RE-POSITION

 PAGE

;

; EOPEN – OPEN A FILE

;

EOPEN

 LDA #CRGOPN

OPEN

 STA TEMP1A

 LDA CL ;IF NO LENGTH ENTERED

 BNE EO1 ;THEN SET DEFAULT OF 1

 LDA CL+1

 BNE EO1

 LDA #1

 STA CL

EO1

 LDA CL ;MOVE REC LENGTH

 STA CCBRLN

 LDA CL+1

 STA CCBRLN+1

EO3

 JSR ECLOSE ;GO CLOSE IF OPEN

EO4

 LDA CNUM+1 ;GET AVALL ENTRY

 BNE EO5 ;BR IF ONE AVAIL

 JMP ENFA ;DONE – NO FILES AVAIL

EO5

 STA ZPGWRK+1 ;MOVE AVAIL SLOT TO ZPG

 LDA CNUM

 STA ZPGWRK
EO6

 JSR MVFN1 ;GO MOVE FILE NAME

 JSR MVBUFP ;GO MOVE BUF PTRS

 JSR OPNSUP ;GO SET UP OPEN

 LDA TEMP1A ;SET OPEN REG

 STA CCBREG

 JMP DOSGO ;GO OPEN

 PAGE

;

; ECLOSE – EXECUTE CLOSE FILE COMMAND

;

ECLOSE

 LDA FNAME1

 CMP #$A0

 BEQ CLALL

 JSR FILSRC ;GO FIND FILE

 BCS ECL1 ;BR IF NOT FOUND

 JSR CLOSE ;GO CLOSE

 JMP ECLOSE ;GO SEE IF ANY MORE OPEN

ECL1 RTS

;

; CLOSE – CLOSE A FILE

;

CLOSE

 JSR TSTEXC

 BNE CLX

 LDA #0

 STA ESTATE

CLX

 LDY #0 ;CLEAR 1ST FN

 TYA ;CHAR TO ZERO

 STA (ZPGWRK),Y

 JSR MVBUFP ;MOVE BUFFER PTRS

 LDA #CRGCLS ;SET CLOSE

 STA CCBREG

 JMP DOSGO ;GO CLOSE

;

; CLALL – CLOSE ALL FILES

;

CLALL

 JSR TSINIT ;GO INIT FILE SEARCH

 BNE CL1

CL0

 JSR TSNXT ;NEXT ENTRY

 BEQ CL2 ;BR IF NO MORE

CL1

 JSR TSTEXC

 BEQ CL0

 JSR TSTOPN ;GOL TEST OPEN

 BEQ CL0 ;BR NOT OPEN

 JSR CLOSE ;GO CLOSE

 JMP CLALL ;START OVER

CL2 RTS ;DONE

 PAGE

;

; EBSV – EXECUTE BINARY SAVE

;

EBSV

 LDA #A+L ;IF ALL

 AND INOPTS

 CMP #A+L

 BEQ EBSV1

 JMP CNF ;THEN ERROR

EBSV1

 LDA #4 ;SET BINARY FILE

 JSR SV1 ;GO OPEN & TEST

 LDA CA+1 ;OUTPUTADR OF BLOCK

 LDY CA

 JSR SV2

 LDA CL+1 ;GO OPEN AND TEST

 LDY CL

 JSR SV2 ;OUTPUT LENGTH

 LDA CA+1 ;GET ADR GIVEN

 LDY CA

 JMP SV3 ;OUTPUT BLOCK

;

; EBLD – EXECUTE BINARY LOAD

;

EBLD

 JSR EOPEN

 JSR TSTFNF

EBLD2

 LDA #$7F

 AND CCBFUC

 CMP #4

 BEQ EBLD3

 JMP ENBF

EBLD3

 LDA #4 ;SET BINARY FILE

 JSR SV1 ;GO OPEN $ TEST

 JSR LD2 ;GO GET ADR

 TAX

 LDA INOPTS

 AND #A ;IF ADR NOT GIVEN

 BNE EBLD1

 STX CA ;THEN USE ADR FROM FILE

 STY CA+1

EBLD1

 JSR LD2 ;GET LENGTH

 LDX CA ;GET GIVEN ADR

 LDY CA+1

 JMP LD3 ;GO GET BLOCK

;

; EBRUN – EXECUTE BINARY RUN

;

EBRUN

 JSR EBLD ;GO LOAD FILE

 JSR MVCSW ;GO RESTORE CHAR I/O SW

 JMP (CA) ;GO EXEC THE STUFF

TSTFNF LDA #CREFNF ;FILE NOT FOUND ERROR CODE

 CMP CCBSTA ;TEST FILE NOT FOUND

 BEQ FNF ;BR IF FILE NOT FOUND

 RTS ;FILE FOUND, RETURN

FNF JMP KLUTZ ;GO FIX THINGS

 PAGE

;

; ESAVE – EXECUTE SAVE REQUEST

;

ESAVE

 LDA ASIBSW ;IF IB THEN

 BEQ EIBSV ;GO TO IB SAVE

 LDA #2 ;GET APPLESOFT PGM

 JSR SV1 ;GO OPEN AND TEST

;

 SEC ;BLOCK LENGTH

 LDA ASEOP ;=EOP-SOP

 SBC ASSOP

 TAY

 LDA ASEOP+1

 SBC ASSOP+1

 JSR SV2 ;GO OUTPUT BLOCK

;

 LDA ASSOP+1 ;BLOCK ADR

 LDY ASSOP ;=SOP

 JMP SV3 ;GO OUTPUT BLOCK

;

EIBSV

 LDA #1 ;SET IB PGM

 JSR SV1 ;GO OPEN AND TEST

;

 SEC ;BLOCK LENGTH

 LDA IBHMEM ;=HIMEM-SOP

 SBC IBSOP

 TAY

 LDA IBHMEM+1

 SBC IBSOP+1

 JSR SV2 ;GO OUTPUT LENGTH

;

 LDA IBSOP+1 ;BLOCK ADR

 LDY IBSOP ;=SOP

 JMP SV3 ;GO OUTPUT

;

SV1

SV1A

 STA CCBFUC ;SET BLOCK PGM TYPE

 PHA ;SAVE PGM TYPE

 JSR EOPEN ;GO OPEN FILE

 PLA ;GET SAVE TYPE

 JMP TSTFUC ;GO CHECK

;

SV2

 STY CCBBLN ;SET BLOCK LENGTH

 STY CCBDAT ;AND DATA BYTE

 STA CCBBLN+1

 LDA #CRGWR ;INDICATE WRITE

 STA CCBREG

 LDA #CRMNBT ;NEXT BYTE

 STA CCBRQM

 JSR DOSGO ;GO WRITE

 LDA CCBBLN+1 ;OTHER BYTE TOO

 STA CCBDAT

 JMP DOSGO

;

SV3 STY CCBBBA ;SET BLOCK ADR

 STA CCBBBA+1

 LDA #CRMNBL ;INDICATE BLOCK I/L

 STA CCBRQM

 JSR DOSGO ;GO DO IT

 JMP ECLOSE ;CLOSE FILE

 PAGE

NBPER JMP ERNU1

;

; ELOAD – EXECUTE LOAD REQUEST

;

ELOAD

 JSR CLALL ;GO CLOSE ALL

 JSR EOPEN ;OPEN FILE

 LDA #CREFNF

 CMP CCBSTA ;WAS FILE FOUND

 BNE ELD1 ;BR IF FOUND

;

KLUTZ JSR EDEL ;DELETE NEW FILE

 LDA #CREFNF ;FILE NOT FOUND MSG

 JMP ERROR ;GO

;

ELD1

 LDA #$7F ;MASK PROTECT BIT

 AND CCBFUC ;OUT OF FUC

 BEQ NBPER ;BR IF ERROR

 AND #$03 ;ISOLOLATE IB & AS

 BEQ NBPER ;BR IF ERROR

 STA CCBFUC ;SAVE IB/AS ONLY

 LDA ASIBSW ;IF IB THEN

 BEQ EIBL ;GO TO IB LOAD

 LDA #2

 JSR LD1 ;GO OPEN AND TEST

;

 JSR LD2 ;GO GET BLOCK LENGTH

;

 CLC

 ADC ASSOP ;ADD BLOCK LENGTH TO SOP

 TAX

 TYA

 ADC ASSOP+1

;

 CMP ASHM1+1 ;IF BL+SOP >=HMEM

 BCS MFULL ;THEN WON’T FIT

;

EASL1

 STA ASEOP+1 ;SET NEW EOP ADR

 STA ASEOP2+1

 STX ASEOP

 STX ASEOP2

 LDX ASSOP ;GET ADR WHERE TO LOAD

 LDY ASSOP+1

 JMP LD3 ;GO LOAD

;

EIBL

 LDA #1 ;SET IB PGM

 JSR LD1 ;GO OPEN AND TEST

;

 JSR LD2 ;GO GET BLOCK LENGTH

;

 SEC ;HMEM – BLOCK LENGTH

 LDA IBHMEM ;IS NEW SOP

 SBC SVBL

 TAX

 LDA IBHMEM+1

 SBC SVBL+1

 BCC MFULL

 TAY

;

 CPY IBLMEM+1 ;IF NEW SOP <= LMEM

 BCC MFULL

 BEQ MFULL

 STY IBSOP+1 ;SET NEW SOP

 STX IBSOP

 JMP LD3

;

LD2

 LDA SVBLA ;MOVE ADR OF WHERE

 STA CCBBBA ;TO PUT DATA TO

 LDA SVBLA+1 ;CCBN

 STA CCBBBA+1

 LDA #0

 STA CCBBLN+1 ;READ INTO

 LDA #2

 STA CCBBLN

 LDA #CRGRD ;READ

 STA CCBREG

 LDA #CRMNBL ;BLOCK

 STA CCBRQM

 JSR DOSGO

 LDA SVBL+1

 STA CCBBLN+1

 TAY

 LDA SVBL

 STA CCBBLN

 RTS

;

LD3

 STX CCBBBA ;SET BLOCK ADR

 STY CCBBBA+1

 JSR DOSGO ;GEET BLOCK

 JMP ECLOSE ;GO CLOSE FILE

;

MFULL

 JSR ECLOSE ;GO CLOSE FILE

 JMP MFERR ;AND GIVE ERR MSG

LD1

 CMP CCBFUC ;TEST TYPE

 BEQ LD1C ;BR IF MATCH

 LDX CMDNO

 STX SVCMD

 LSRA

 BEQ LD1A ;BR IF PGM IS AS

 JMP EINT ;GO FOR INTO BASIC

;

LD1A

 LDX #29 ;SAVE FILE NAME

LD1B LDA FNAME1,X ;INCASE IS RAM APPLESOFT

 STA FNAME2,X

 DEX

 BPL LD1B

 JMP EAS ;GO FOR AS

;

LD1C RTS

 PAGE

;

; ERUN – EXECUTE RUN REQUEST

;

ERUN

 JSR ELOAD ;LOAD PGM

 JSR PRCRIF

 JSR MVCSW ;GO RESTORE CHAR I/O SW

 JMP (RUN)

;

; IBRUN – INT BASIC RUN

;

IBRUN

 LDA IBLMEM ;RESET START OF VARS

 STA IBSOV

 LDA IBLMEM+1

 STA IBSOV+1

 JMP (CHAIN)

;

; EHCAIN – EXECUTE CHAIN REQUEST

;

ECHAIN

 JSR ELOAD ;LOAD PGM

 JSR PRCRIF

 JSR MVCSW ;GO RESTORE CHAR I/O SW

 JMP (CHAIN)

ASRUN1 JSR $D665 ;ROM

 JMP $D7D2

ASRUN2 JSR $E65 ;RAM

 JMP $FD4

 PAGE

;

; EWRITE – WRITE CMD EXECUTE

;

EWRITE

 JSR RWPOSN ;GO POSITION FILE IF REGD

 JSR TSTFNF

 LDA #5

 STA OSTATE ;SET OSTATE=5

 JMP CERTN ;DONE

;

; EREAD – READ COMD EXECUTE

;

EREAD

 JSR RWPOSN ;GO POSITION FILE IF REGD

 LDA #1

 STA ISTATE ;SET ISTATE = DISK INPUT

 JMP CERTN ;DONE

;

; RWPOSN – POSITION FOR READ/WRITE

;

RWPOSN

 JSR FILSRC ;FIND THE FILE

 BCC RWP1 ;BR IF FILE FOUND

 JSR EOPEN ;GO OPEN FOR KLUTZ

 JMP RWP2 ;THEN SKIP NEXT LINE

RWP1

 JSR MVBUFP ;MOVE BUFF POINTERS

RWP2

 LDA INOPTS ;GET IN OPTIONS

 AND #R+B ;WAS IT B OR R

 BEQ RWPR ;BR IF NOT

 LDX #3

RWP2A LDA CR,X ;MOVE REL REC

 STA CCBRRN,X ;AND REL BYTE

 DEX

 BPL RWP2A

RWP3

 LDA #CRGPOS ;INDICATE POSITION REQUIRED

 STA CCBREG

 JSR DOSGO

RWPR RTS ;DONE

 PAGE

;

;

; EINIT – EXECUTE INIT COMMAND

;

EINIT

 LDA #V ;MUST HAVE

 AND INOPTS ;VOL OPTION

 BEQ INER

 LDA CV

 BEQ INER ;BE GT 0

 LDA ASTART+1

 STA CCBBSA

 LDA #CRQFMT

 JSR OPEN

 JMP ESAVE

;

INER JMP CNF

;

; ECAT – PRINT CATALOG

;

ECAT

 LDA #CRGDIR

 JSR OPEN ;GO PRETEND OPEN

 LDA CCBVOL

 STA CV

 RTS

 PAGE

;

; EAS – EXECUTE APPLESOFT REQUEST

;

EAS

 LDA #ATSTV ;GET APPLESOFT TEST VALUE

 JSR SWTST ;GO SWITCH AND TEST

 BEQ GOINIT ;GO SWITCH AND TEST

 LDA #0

 STA ASIBSW

;

EAS0

 LDY #30

 JSR CLRFNA

 LDX #FASBL

EAS1 LDA FASB-1,X ;MOVE SYSTEM FILE NAME

 STA FNAME1-1,X

 DEX

 BNE EAS1

;

EAS2

 LDA #$C0

 STA ISTATE ;FOR RAM APPLESOFT

 JMP ERUN ;GO LOAD AND RUN

;

; EINT – EXECUTE INTEGER REQUEST

;

EINT

 LDA #ITSTV ;GET IB TEST VALUE

 JSR SWTST ;GO SWITCH AND RUN

 BNE EAS ;BR IF NOT IB

GOINIT

 JMP DBINIT ;GO INIT DOS

SWTST

 CMP AITSTL ;TEST CURRENT VALUE

 BEQ SWTR

 STA $C080 ;TRY SWITCH 1

 CMP AITSTL ;TEST AGAIN

 BEQ SWTR ;BR IF NOW SAME

 STA $C081 ;TRY SWITCH 2

 CMP AITSTL ;TEST AND

SWTR RTS

;

 PAGE

;

; EEXEC – EXEC CMD

;

EEXEC

 JSR EOPEN ;OPEN FILE

 JSR TSTFNF

 LDA CFTABA ;MOVE TABLE POINTERS

 STA EFTABA

 LDA CFTABA+1

 STA EFTABA+1

 LDA FNAME1 ;USE FILENAME

 STA ESTATE ;SET EX STATE NON ZERO

 BNE EXP2

;

;

; EPOS – EXECUTE POSITION

;

EPOS

 JSR FILSRC

 BCC EXP1

 JSR EOPEN

 JSR TSTFNF

 JMP EXP2

EXP1 JSR MVBUFP

EXP2

 LDA INOPTS ;GET OPTIONS

 AND #R ;TEST R

 BEQ EX2 ;BR NOT R

;

EXO LDA CR ;IF CR NOT ZERO

 BNE EX1A ;THEN DECREMENT

 LDX CR+1

 BEQ EX2

 DEC CR+1

EX1A DEC CR

EX1 JSR RBYTE ;AND READ A RCORD

 BEQ ICFD4

 CMP #$8D ;UNTIL CR

 BNE EX1

 BEQ EX0 ;THEN TEST CR AGAIN

;

EX2 RTS ;DONE

 PAGE

;

; OCTD – OUTPUT A CHAR TO DISK

;

OCTD

 JSR TSTRUN ;GO TEST RUN

 BCS ICFDB

 LDA SVA ;CHAR IN SAVED ACU

 STA CCBDAT ;PUT INTO CCBDATA AREA

 LDA #CRQWR ;SET WRITE

 STA CCBREQ

 LDA #CRMNBT ;SET NEXT BYTE

 STA CCBRQM

 JMP DOSGO ;GO WRITE BYTE

;

; INCFD – INPUT A CHAR FROM DISK

;

ICFD

 JSR TSTRUN ;GO TEST RUN

 BCS ICFDB

 LDA #6 ;SET OUT STE = 6

ICFD3

 STA OSTATE ;TO CATCH ECHO

 JSR RBYTE

 BNE ICFD1 ;BR IF NOT ZERO CHAR

ICCFD2

 JSR CLOSE

 LDA #3

 CMP OSTATE

 BEQ ICFD0

ICFD4

 LDA #CREEOF

 JMP ERROR ;GO TO ERROR

ICFD1

 STA SVA ;PUT INTO SAVED ACU

ICFD0

 JMP ORTN ;GO RESTORE REGS AND RTS

;

TSTRUN

 LDA ASIBSW ;GET AS/INT BASIC SWITCH

 BEQ TR1 ;BR IF INT

 LDX $76 ;TEST AS RUN

 BNE NOTRUN ;BR IF NOT RUN

TR0 CLC

 RTS

TR1

 LDA $D9 ;GET INT RUN FLAG

 BMI TR0 ;BR IF RUN

NOTRUN SEC

 RTS

ICFDB ;NOT RUN MODE

 JSR CLOSE ;GO CLOSE FILE

 JSR CLRSTS ;GO CLEAR STATES

 JMP ORTN

 PAGE

;

; NXTEXC – NEXT EXECUTE CHAR

;

NXTEXC

 JSR MVEFTA

 JSR MVBUFP ;GO MOVE PTRS

 LDA #3

 BNE ICFD3

;

; RBYTE – READ NEXT BYTE

;

RBYTE

 LDA #CRQRD ;SET READ

 STA CCBREQ

 LDA #CRMNBT ;SET NEXT BYTE

 STA CCBRQM

 JSR DOSGO ;GO TO DOS

 LDA CCBDAT ;GET THE DATA BYTE

 RTS

MVEFTA

 LDA EFTABA+1 ;MOVE TABLE ADR

 STA ZPGWRK+1 ;NO ZPG

 LDA EFTABA

 STA ZPGWRK

 RTS

 PAGE

;

; DOSGO – GOTO DOS

;

DOSGO

 JSR DOSENT ;GO TO DOS

 BCS DG1 ;BR IF ERROR

 RTS ;DONE

;

DG1 ;\*\*\* ERROR \*\*\*

 JSR FILSRC ;GET FILE TABLE

 BCS DG2 ;BR IF NOT FOUND

 LDA #0

 TAY

 STA (ZPGWRK), Y ;CLOSE FILE HERE

DG2

 LDA CCBSTA ;GET STATUS OF I/O

 CMP #CREEOF ;EOF ?

 BNE DG3 ;BR IF NOT

 LDX #0 ;SET OTHER EIF

 STX CCBDAT ;DONE

 RTS

DG3

 JMP ERROR ;GO DO ERROR

;

 PAGE

;

; ERROR ROUTINE

;

ESYNTX LDA #CREFLK+1

 BNE ERROR

ENFA LDA #CREFLK+2

 BNE ERROR

MFERR LDA #CREFLK+4

 BNE ERROR

ERNU1 LDA #CREFLK+3

 BNE ERROR

ENBF LDA #CREFLK+5

;

ERROR

 STA SVA ;SAVE MSG NUMBER

 JSR CLRSTS

 LDA ASIBSW ;GET AS/IN BASIC SWITCH

 BEQ ERNAS ;BR IF NOT APPLESOFT

 LDA $D8 ;GET ON ERR FLAG

 BMI ERRTN ;BRT IF ON ERR IS GO

ERNAS

 LDX #0

 JSR EMPR ;GO OUTPUT

 LDX SVA ;GET SAVE MSG

 JSR EMPR ;GO OUTPUT MSG

 LDX #CREFLK+6

 JSR EMPR

ERRTN JSR MVCSW ;GO MOVE CHAR I/ SW

 LDX SVA

 LDA #03

 JMP (BREAK)

;

EMPR

 LDA EMDTB,X ;GET ITS DISPL

 TAX ;INTO X

EMPR1

 STX TEMP1A ;SAVE DISPL

 LDA EMSG,X ;GET MSG CHAR

 PHA ;SAVE CHAR

 ORA #$80 ;SET MSB ON

 LDX TEMP1A ;GET INDEX

 INX ;INCREMENT IT

 PLA ;RE-LOAD CHAR

 BPL EMPR1 ;BR IF MORE CHARS

 RTS ;DONE

 PAGE

;

; OPNSUP – OPEN SET UP

;

OPNSUP

 LDA CV ;VOLUME

 STA CCBVOL

 LDA CD ;DRIVE

 STA CCBDRV

 LDA CS ;SLOT

 STA CCBSLT

 LDA FN1ADR ;FILENAME 1 ADR

 STA CCBFN1

 LDA FN1ADR+1

 STA CCBFN1+1

 LDA ZPGWRK

 STA CFTABA

 LDA ZPGWRK+1

 STA CFTABA+1

 RTS

;

; MVFN1 – MOVE FILE NAME 1 TO FILE PTR

;

MVFN1

 LDY #29

MVFN1A LDA FNAME1,Y

 STA (ZPGWRK),Y

 DEY

 BPL MVFN1A

 RTS

;

; MVBUFP – MOVE BUFFER PTRS TO CCB

;

MVBUFP

 LDY #30

MVBP1 LDA (ZPGWRK),Y

 STA CCBFCB-30,Y

 INY

 CPY #38

 BNE MVBP1

 RTS

;

; CLRSTS – CLEAR STATES

;

CLRSTS

 LDY #0

 STY ISTATE

 STY OSTATE

 RTS

 PAGE

;

; FILSRC – SEARCH FOR FILE NAME1

;

FILSRC

 LDA #0 ;CLEAR SV AVAIL

 STA CNUM+1

;

 JSR TSINIT ;GO INIT SEARCH

 JMP FLS1A

FLS1 JSR TSNXT ;LOOK AT NEXT

 BEQ FLS4 ;BR IF NO NEXT

;

FLS1A JSR TSTOPN ;GO TEST OPEN

 BNE FLS2 ;BR IF OPEN

;

 LDA ZPGWRK ;SAVE AVAIL ENTRY ADR

 STA CNUM

 LDA ZPGWRK+1

 STA CNUM+1

 BNE FLS1 ;GO LOOK SOME MORE

;

FLS2 LDY #29 ;FILE HAD 30 CHARS

FLS3 LDA (ZPGWRK),Y ;GET CHAR

 CMP FNAME1,Y ;TEST CHAR

 BNE FLS1 ;BR NOT

 DEY

 BPL FLS3 ;LOOK AT 30 CHARS

 CLC ;FOUND

 RTS ;DONE

;

FLS4 SEC ;NOT FOUND

 RTS ;DONE

 PAGE

;

; TSINIT – INITIALIZE FOR FTAB SEARCH

; TSNXT – GET NEXT FTAB ENTRY

;

TSINIT

 LDA FTAB ;GET 1ST PTR ADR

 LDX FTAB+1

 BNE TSST

TSNXT

 LDY #37 ;GET LINK

 LDA (ZPGWRK),Y

 BEQ TSR ;BR IF NO LINK

;

 TAX

 DEY

 LDA (ZPGWRK),Y

TSST

 STX ZPGWRK+1

 STA ZPGWRK

 TXA ;SET NE CC

TSR RTS ;RTN

;

; TSTOPN – TST FOR OPEN FILE

;

TSTOPN

 LDY #0 ;GET 1ST CHAR OF FN

 LDA (ZPGWRK),Y

 RTS

;

; TSTEXC – TEST CURRENT FILE FOR EXECUTE

;

TSTEXC

 LDA ESTATE ;IF ESTATE = 0

 BEQ TXTC1 ;THEN NO EXECUTE FILE

 LDA EFTABA ;TEST CURRENT

 CMP ZPGWRK

 BNE TXC2 ;IS NOT

 LDA EFTABA+1

 CMP ZPGWRK+1

 BEQ TXC2 ;IS

TXC1 DEX ;IS NOT

TXC2 RTS ;DONE

 PAGE

;

; TSTFUC – TEST FILE USE CODE FOR PGM

;

TSTFUC

 EOR CCBFUC

 BEQ TFUCR

 AND #$7F

 BEQ TFUCR

 JSR ECLOSE ;GO CLOSE THE SOB

 JMP ERNU1

TFUCR RTS

 PAGE

;

; BLDFTB – BUILD FILE TABLES

; TABLE MAP:

; HIMEM,SOP

; SBUFF N (256)

; DBUFF N (256)

; FTB N (FCBLEN)

; HEADER N (38)

;

;

; SBUFF 1

; DBUFF 1

; FTB 1

; HEADER 1

; THIS PROGRAM

;

; HEADER MAP:

; FILENAME (30)

; FTB PTR (2)

; DBUF PTR (2)

; SBUF PTR (2)

; LINK (2)

;

BLDFTB

 SEC

 LDA FTAB ;START OF FTAB AREA

 STA ZPGWRK ;IS 1ST FTB PTR

 LDA FTAB+1 ;HEADER

 STA ZPGWRK+1

 LDA CNFTBS ;MOVE NO FTABS

 STA TEMP1A ;TO TEMP

;

BFT1 LDY #0

 TYA

 STA (ZPGWRK),Y ;1ST CHAR FN=0

 LDY #30 ;INC Y TO FCB PTR

 SEC

 LDA ZPGWRK ;END OF PTR HEADER

 SBC #FCBLEN ;MINUS FTAB LENGTH

 STA (ZPGWRK),Y ;IS START OF FTB

 PHA ;SAVE LOW ADR BYTE

 LDA ZPGWRK+1

 SBC #0

 INY

 STA (ZPGWRK),Y

 TAX

 DEX ;FTB ADR – 256

 PLA ;IS ADR DIR BUFF

 PHA

 INY

 STA (ZPGWRK),Y ;SET DIR BUF PTR

 TXA

 INY

 STA (ZPGWRK),Y

 TAX

 DEX ;DIR BUFF – 256

 PLA ;IS SBUFF ADR

 PHA

 INY

 STA (ZPGWRK),Y

 INY

 TXA

 STA (ZPGWRK),Y

;

 DEC TEMP1A ;DECREMENT TABLE INDEX

 BEQ BFT2 ;COUNT AND BR IF DONE

 TAX

 PLA

 SEC

 SBC #38 ;SBUFF ADR – 38

 INY

 STA (ZPGWRK),Y ;IF ADR OF NEXT TAB

 PHA ;WHICH GOES INTO

 TXA ;LINK

 SBC #0

 INY

 STA (ZPGWRK),Y

 STA ZPGWRK+1 ;AND INTO ZPGWRK

 PLA ;FOR NEXT ENTRY

 STA ZPGWRK ;BUILD

 JMP BFT1 ;GO BUILD NEXT

;

BFT2

 PHA

 LDA #0 ;SET LAST LINK

 INY ;TO ZERO

 STA (ZPGWRK),Y

 INY

 STA (ZPGWRK),Y

;

 LDA ASIBSW ;IF IB THEN GO

 BEQ BFTIB

;

 PLA ;SET APPLESOFT

 STA ASHM1+1 ;UPPER MEM LIMITS

 STA ASHM2+1

 PLA

 STA ASHM1

 STA ASHM2

 RTS

;

BFTIB

 PLA ;SET IB

 STA IBHMEM+1 ;UPPER MEM LIMITS

 STA IBSOP+1

 PLA

 STA IBHMEM

 STA IBSOP

 RTS

 PAGE

;

; MVISW – MOVE INPUT SWITCH

;

MVCSW

 LDA INSW+1

 CMP CINA+1

 BEQ MVOSW

 STA SVINS+1

 LDA INSW ;SAVE CHAR INSWITCH

 STA SVINS

;

 LDA CINA ;SET DB CHAR IN ADR

 STA INSW

 LDA CINA+1

 STA INSW+1

;

;

; MVOSW – MOVE OUTPUT SWITCH

;

MVOSW

 LDA OUTSW+1

 CMP COUTA+1

 BEQ MVSRTN

 STA SVOUTS+1

 LDA OUTSW ;SAVE CHAR OUT SWITCH

 STA SVOUTS

;

 LDA COUTA ;SET DB CHAR OUT ADR

 STA OUTSW

 LDA COUTA+1

 STA OUTSW+1

MVSRTN

 RTS

 PAGE

;

; COMMAND NAME TABLE

;

EC1

CMDNTB

 DB01 "INIT"

 DB01 "LOAD"

 DB01 "SAVE"

 DB01 "RUN"

 DB01 "CHAIN"

 DB01 "DELETE"

 DB01 "LOCK"

 DB01 "UNLOCK"

 DB01 "CLOSE"

 DB01 "READ"

 DB01 "EXEC"

 DB01 "WRITE"

 DB01 "POSITION"

 DB01 "OPEN"

 DB01 "APPEND"

 DB01 "RENAME"

 DB01 "CATALOG"

 DB01 "MON"

 DB01 "NOMON"

 DB01 "PR#"

 DB01 "IN#"

 DB01 "MAXFILES"

 DB01 "FP"

 DB01 "INT"

 DB01 "BSAVE"

 DB01 "BLOAD"

 DB01 "BRUN"

 DB01 "VERIFY"

 DB 0

 PAGE

;

; COMMAND SYNTAX OP EQUATES FOR SYNTAX BYTE ONE

;

NPB EQU $80 ;NO PARMS OK, COMMAND GOES TO BASIC

NPE EQU $40 ;NO PARMS OK, COMMAND TO EXECUTION RTN

FN1 EQU $20 ;FILE NAME1 REGD

FN2 EQU $10 ;FILE NAME2 REGD

NUM1 EQU $08 ;NUMERIC 0-7 REGD

NUM2 EQU $04 ;NUMERIC 1-10 REGD

;

; COMMAND SYNTAX OP EQUATES FOR SYNTAX BYTE TWO

;

V EQU $40 ;VOLUME ALLOWED

D EQU $20 ;DRIVE ALLOWED

S EQU $10 ;SLOT ALLOWED

L EQU $08 ;LENGTH ALLOWED

R EQU $04 ;RECORD NUMBER ALLOWED

B EQU $02 ;BYTE NUMBER ALLOWED

A EQU $01 ;ADDRESS

CIO EQU $80 ;C,I, OR O ALLOWED

;

; COMMAND SYNTAX TABLE

; EACH COMMAND HAS TWO BYTE ENTRY

;

;

CMDSTB

 DB FN1,V+D+S ;INIT

 DB NPB+FN1,V+D+S ;LOAD

 DB NPB+FN1,V+D+S ;SAVE

 DB NPB+FN1,V+D+S ;RUN

 DB FN1,V+D+S ;CHAIN

 DB FN1,V+D+S ;DELETE

 DB FN1,V+D+S ;LOCK

 DB FN1,V+D+S ;UNLOCK

 DB NPE+FN1,0 ;CLOSE

 DB FN1,B+R ;READ

 DB FN1,R+V+D+S ;EXEC

 DB FN1,B+R ;WRITE

 DB FN1,R ;POSITION

 DB FN1,L+V+D+S ;OPEN

 DB FN1,L+V+D+S ;APPEND

 DB FN1+FN2,V+D+S ;RENAME

 DB NPE,V+D+S ;CATALOG

 DB NPE,CIO ;MONITOR

 DB NPE,CIO ;NO MONITOR

 DB NUM1,0 ;PR#

 DB NUM1,0 ;IN#

 DB NUM2,0 ;MAXFILES

 DB NPE,V+D+S ;APPLESOFT

 DB NPE,0 ;INT

 DB FN1,V+D+S+A+L ;BSAVE

 DB FN1,V+D+S+A ;BLOAD

 DB FN1,V+D+S+A ;BRUN

 DB FN1,V+D+S ;VERIFY

 PAGE

;

; OPTAB – OPTIONAL PARMS SYNTAX TABLES

;

OPTAB1

 DB11 "VDSLRBACIO"

OPT1L EQU \*-OPTAB1

OPTAB2

 DB V,D,S,L,R,B,A,CIO+MC,CIO+MI,CIO+MO

OPTAB3

 DB @@0,@@254 ;VOL RANGE

 DB @@1,@@2 ;DRIVE RANGE

 DB @@1,@@7 ;SLOT RANGE

 DB @@1,@@32767 ;LENGTH RANGE

 DB @@0,@@32767 ;REC NO RANGE

 DB @@0,@@32767 ;REC BYTE NO RANGE

 DB @@0,@@$C000 ;ADDRESS RANGE

 PAGE

;

; ERROR MESSAGE TABLES

;

EMSG

 DB $0D,$07

 DB01 "\*\*\*DISK: "

EM1 EQU \*-EMSG

EM2 EQU \*-EMSG

EM3 EQU \*-EMSG

 DB01 "SYS"

EM4 EQU \*-EMSG

 DB01 "WRITE PROTECT"

EM5 EQU \*-EMSG

 DB01 "END OF DATA"

EM6 EQU \*-EMSG

 DB01 "FILE NOT FOUND"

EM7 EQU \*-EMSG

 DB01 "VOLUME MISMATCH"

EM8 EQU \*-EMSG

 DB01 "DISK I/O"

EM9 EQU \*-EMSG

 DB01 "DISK FULL"

EM10 EQU \*EMSG

 DB01 "FILE LOCKED"

EM11 EQU \*EMSG

 DB01 "CMD SYNTAX"

EM12 EQU \*-EMSG

 DB01 "NO FILE BUFFS AVAIL"

EM13 EQU \*-EMSG

 DB01 "NOT BASIC PROGRAM"

EM14 EQU \*-EMSG

 DB01 "PROGRAM TOO LARGE"

EM15 EQU \*-EMSG

 DB01 "NOT BINARY FILE"

;

EML EQU \*-EMSG

 DB " ERROR"

 DB $8D

EMDTB

 DB 0,EM1,EM2,EM3,EM4

 DB EM5,EM6,EM7,EM8,EM9

 DB EM10,EM11,EM12,EM13,EM14

 DB EM15

 PAGE

;

; MISC BUT REQD CELLS

;

CFTABA DB @0 ;CURRENT FILE TABLE POINTER

ISTATE DB 0 ;INPUT STATE

OSTATE DB 0 ;OUTPUT STATE

SVOUTS DB @0 ;SAVED OUT SWITCH

SVINS DB @0 ;SAVED IN SWITCH

CNFTBS DB 0 ;CURRENT NO FILE TABLES

DFNFTB DB 3 ;DEFAULT NO FILE TABLES

SVSTK DB 0 ;SAVED STACK PTR

SVX DB 0 ;DSAVED X REG

SVY DB 0 ;SAVED XREG

LBUFD DB 0 ;SAVED ACU

MONMOD DB 0 ;LINE BUFF DISPL

MC EQU $40 ;MONITOR MODE BITS

MI EQU $20 ;MONITOR CMDS

MO EQU $10 ;MONITOR OUTPUT

CMDNO DB $FF ;COMMAND NO

SVBL DB 0,0

SVCMD DB 0

TEMP1A DB 0

TEMP2A DB 0

INOPTS DB 0 ;INPUT OPTIONS

CUROPT ;CURRENT OPTIONS

CV DB @@0 ;VOLUME

CD DB @@0 ;DRIVE

CS DB @@0 ;SLOT

CL DB @@1 ;RECORD LENGTH

CR DB @@0 ;RECORD NUMBER

CB DB @@0 ;RECORD BYTE

CA DB @@0 ;ADDRESS

IMBITS DB 0

FNAME1 RMB 30 ;FILENAME 1

FNAME2 RMB 30 ;FILENAME 2

DFNFTS DB 3 ;DEFAULT FILE TABLES = 3

CCHAR DB $84 ;CONTROL CHAR

ESTATE DB 0 ;EXECUTE STATE

EFTABA DB 0,0 ;EXECUTE FILE TABLE POINTER

ASIBSW DB 0 ;APPLESOFT, IB SWITCH

FASB DB11 "APPLESOFT"

FASBL EQU \*-FASB

 PAGE

;

; DOS ADR TABLES (RELOCATED)

;

SAT2

AIOB DB @@IOB ;5-ADR IOB

AVTOC DB @@VTOC ;6-ADR VTOC

AVOLDR DB @@VOLDIR ;7-ADR VOLDIR

AEND DB @@EDOS ;FEND OF DOS

;

CMDVT DB @@GOODIO-1 ;0-NULL

 DB @@FOPEN-1 ;1-OPEN FILE

 DB @@FCLOSE-1 ;2-CLOSE FILE

 DB @@FREAD-1 ;3-READ FILE

 DB @@FWRITE-1 ;4-WRITE DATA

 DB @@FDEL-1 ;5-DELETE FILE

 DB @@FDIR-1 ;6-READ DIRECTORY

 DB @@FLOCK-1 ;7-LOCK A FILE

 DB @@FUNLCK-1 ;8-UNLOCK A FILE

 DB @@FRNME-1 ;9-RENAME

 DB @@FPOSTN-1 ;10-POSITION A FILE

 DB @@FFMT-1 ;FORMAT

 DB @@FVAR-1 ;VARIFY

 DB @@GOODIO-1 ;11-SPARE

;

RVT

 DB @@GOODIO-1

 DB @@RNXBYT-1 ;1-RD NEXT BYTE

 DB @@RNXBLK-1 ;1-RD NEXT BLOCK

 DB @@RSPBYT-1 ;2-RD SPECIFIC BYTE

 DB @@RSPBLK-1 ;3-RD SPECIFIC BLOCK

 DB @@GOODIO-1 ;4-SPARE

;

WVT

 DB @@GOODIO-1

 DB @@WNXBYT-1 ;1-WR NEXT BYTE

 DB @@WNXBLK-1 ;WR NEXT BLOCK

 DB @@WSPBYT-1 ;2-WR SPECIFIC BYTE

 DB @@WSPBLK-1 ;3-WR SPECIFIC BLOCK

 DB @@GOODIO-1 ;4- SPARE

EAT2

 PAGE

;

; DOSENT – DOS EXTERNAL ENTRY POINT

; EXIT PARM

; CARRY CLEAR = OPERATION

; CARRY SET = ERROR

;

SC2

DOSENT

 TSX

 STX ENTSTK

 JSR CLCFCB ;GO CALCULATE FCB

 LDA CCBREG ;GET REQUEST

 CMP #CRGMAX ;TTEST REQ RANGE

 BCS ERR2 ;BR OUT OF RANGE

 ASLA ;REQ CODE \*2

 TAX

 LDA CMDVT+1,X ;PUSH ADR ONTO STACK

 PHA

 LDA CMDVT,X

 PHA

DENRTS RTS

ERR2 JMP ERROR2

 PAGE

;

; FOPEN – OPEN A FILE

;

FOPEN

 JSR DOPEN

 JMP GOODIO

;

DOPEN

;

 JSR DCBSUP

;

;

 LDA #1

 STA DCBSDL+1

 LDX CCBRLN+1 ;MOVE RECORD LENGTH

 LDA CCBRLN

 BNE F02

 CPX #0

 BNE F02

 INX ;SET RL=256

F02 STA DCBRCL

 STX DCBRCL+1

;

 JSR FNDFIL ;GO FIND FILE

 BCC F03 ;BR IF FOUND

; ;CREATE FILE

 LDA #0

 STA VDFILE+34,X

 LDA #1

 STA VDFILE+33,X

 STX TEMP1 ;SAVE VDIR INDEX

 JSR GETSEC ;GO ALLOCATE SECTOR

 LDX TEMP1

 STA VDFILE+1,X ;PUT SECTOR INTO VDIR

 STA DCBFDS ;PUT SECTOR AS 1ST FILE DIR

 STA DCBCDS ;PUT SECTOR AS CURRENT FILE DIR

;

 LDA DCBATK ;GET ALLOCATED TRACK

 STA VDFILE,X ;PUT INTO VDIR

 STA DCBFDT ;AND AS 1ST FILE DIR

 STA DCBCDT ;AND AS CURRENT FILE DIR

;

 LDA CCBFUC ;SET USE CODE

 STA VDFILE+2,X ;INTO DIRECTORY

;

 JSR WRVDIR ;GO WRITE VOL DIRECTORY

;

 JSR MVFCBD ;MOVE FILE DIR ADR TO ZP

 JSR CLRSEC ;GO CLEAR IT

 JSR WRFGO ;GO WRITE FILE DIRECTORY DONE CREATION

;

 LDX TEMP1 ;RE-GET INDEX

 LDA #CREFNF

 STA CCBSTA

;

F03

 LDA VDFILE,X ;MOVE FILE DIR TRACK

 STA DCBFDT

 LDA VDFILE+1,X ;MOVE FILE DIR SECTOR

 STA DCBFDS

 LDA VDFILE+2,X ;70VE FILE USE CODE

 STA CCBFUC

 STA DCBFUC

 LDA VDFILE+33,X

 STA DCBNSA

 LDA VDFILE+34,X

 STA DCBNSA+1

 STX DCBVDI ;SAVE DIR INDEX

;

 LDA #255 ;INDICATE NO SECTOR

 STA DCBCMS ;IN MEMORY

 STA DCBCMS+1

 LDA VTDMS ;MOVE MAX FD SECTS

 STA DCBDMS ;TO DCB

 CLC

 JMP RDFDIR ;READ 1ST DIRECTORY RECORD

;

;

;

;

DCBSUP

 LDA #0

 TAX

F01 STA FCBDCB,X ;CLEAR DCB

 INX

 CPX #DCBLEN

 BNE F01

;

 LDA CCBVOL ;MOVE VOL

 EOR #$FF ;INVERT VOL BITS

 STA DCBVOL

 LDA CCBDRV ;MOVE DRIVE

 STA DCBDRV

 LDA CCBSLT ;GET USER SPEC SLOT

 ASLA ;SLOT\*16

 ASLA

 ASLA

 ASLA

 TAX

F01A

 STX DCBSLT

 LDA #17

 STA DCBVTN

 RTS

 PAGE

;

; FCLOSE – CLOSE A FILE

;

FCLOSE

 JSR WRSECT ;WRITE OPEN SECTOR

 JSR WRFDIR ;GO WRITE FILE DIRECTORY

 JSR FRETRK ;FREE UNUSED SECTORS

 LDA #IBCWTS

 AND DCBWRF

 BEQ FC2

;

 JSR RDVTOC ;READ VTOC

 LDA #0

 CLC

FC1

 JSR RDVDIR ;READ VDIR

 SEC

 DEC DCBVDR

 BNE FC1 ;BR IF NOT

 LDX DCBVDI ;GET FILES INDEX

 LDA DCBNSA ;MOVE NO SECTIONS ALLOCATED

 STA VDFILE+33,X

 LDA DCBNSA+1

 STA VDFILE+34,X

 JSR WRVDIR ;WRITE VOL DIR REC

;

;

FC2

 JMP GOODIO ;DONE

 PAGE

;

; FRNME – RENAME A FILE

;

FRNME

 JSR DOPEN ;GO OPEN FILE

 LDA DCBFUC ;GET USE CODE

 BMI ER10 ;BR IF LOCKED

 LDA CCBFN2 ;MOVE NEW FN

 STA ZPGFCB ;PTR TO ZPG

 LDA CCBFN2+1

 STA ZPGFCB+1

 LDX TEMP1 ;GET VDIR INDEX

 JSR MVFN ;GO MOVE FILE NAME

 JSR WRVDIR ;GO WRITE FILE VDIR

 JMP GOODIO ;DONE RENAME

 PAGE

;

; FREAD – READ A FILE

;

FREAD

;

 LDA CCBRQM ;GET REQ MOD

 CMP #CRMMAX ;TEST LIMIT

 BCS ERR3A

;

 ASLA ;CODE\*2

 TAX

 LDA RVT+1,X ;GET READ ROUTINE

 PHA ;VECTOR ADR

 LDA RVT,X

 PHA ;AND

 RTS ;GO TO IT

;

ERR3A JMP ERROR3

ER10 JMP ERRR10

;

; FWRITE – WRITE A FILE

;

FWRITE

 LDA DCBFUC ;IS FILE LOCKED

 BMI ER10 ;BR IF LOCKED

 LDA CCBRQM ;GET REQ MOD

 CMP #CRMMAX ;IN RANGE

 BCS ERR3A ;BR IF NOT IN RANGE

;

 ASLA

 TAX

 LDA WVT+1,X ;GET ROUTINE ADR

 PHA

 LDA WVT,X

 PHA

 RTS ;AND GO TO IT

 PAGE

;

; RSPBYT – READ A SPECIFIC BYTE

;

RSPBYT

 JSR LOCSEC ;GO GET REQD REL SECTOR

;

; RNXBYT – READ NEXT BYTE

;

RNXBYT JSR GETBYT ;GO GET BYTE

 STA CCBDAT ;PUT IN CCB

 JMP GOODIO ;DONE

;

; RSPBLK – READ SPECIFIC BLOCK

;

RSPBLK JSR LOCSEC ;GO LOCATE REL SECTOR

;

; RNXBLK – READ NEXT BLOCK

;

RNXBLK

 JSR DTBLN ;GO DECR LEN (NOT RTN IF=0)

 JSR GETBYT ;GO GET BYTE

 PHA

 JSR MIBDA ;GO MOVE BLOCK ADR AND INCR

 LDY #0

 PLA

 STA (ZPGFCB),Y ;SET DATA BYTE

 JMP RNXBLK ;GO FOR NEXT BYTE

;

; GETBYT – GET A DATA BYTE

;

GETBYT

 JSR LOCNXB ;LOCATE NEXT BYTE

 BCS EOFIN ;BR IF EOF

 LDA (ZPGFCB),Y ;BR IF EOF

 PHA ;SAVE IT

 JSR INCRRB ;INCR REC BYTE

 JSR INCSCB ;INCR SAVED BYTE

 PLA ;GET SAVED BYTE

 RTS ;RETURN

;

EOFIN JMP ERROR5 ;GO TO EOF RTN
 PAGE

;

; WSPBYT – WRITE SPECIFIC BYTE

;

WSPBYT

 JSR LOCSEC ;GO LOCATE SECTOR

;

; WNXBYT – WRITE NEXT BYTE

;

WNXBYT

 LDA CCBDAT ;GET THE BYTE

 JSR PUTBYT ;GO WRITE BYTE

 JMP GOODIO ;DONE

;

; WSPBLK – WRITE NEXT BLOCK

;

WSPBLK

 JSR LOCSEC ;GO LOCATE SECTOR

;

; WNXBLK – WRITE BLOCK

;

WNXBLK

 JSR MIBDA ;GO MOVE ADR TO ZPG AND DEC

 LDY #0

 LDA (ZPGFCB),Y ;GET DATA BYTE

 JSR PUTBYT ;GO PUT IT

 JSR DTBLN ;GO DEC BLK LEN (NOT RTN IF = 0)

 JMP WNXBLK

;

; PUTBYT – PUT OUT ONE BYTE

;

PUTBYT

 PHA ;SAVE DATA BYTE

 JSR LOCNXB ;GO LOCATE NEXT BYTE

;

PB0 PLA ;GO SAVED BYTE

 STA (ZPGFCB),Y ;PUT THE BYTE

 LDA #$40 ;SET WRITE SECTOR REQD

 ORA DCBWRF

 STA DCBWRF

;

 JSR INCRRB ;INCR REL REC BYTE

 JMP INCSCB ;INCR SECTOR BYTE

 PAGE

;

; FLOCK – LOCK A FILE

;

FUNLOCK LDA #$80 ;REMEMBER LOCK

 STA TEMP3

 BNE LCKGO

;

; FUNLCK – UNLOCK A FILE

;

FUNLCK LDA #00 ;REMEMBER UNLOCK

 STA TEMP3

;

LCKGO

;

 JSR DOPEN ;GO OPEN FILE

 LDX TEMP1

 LDA VDFILE+2,X ;GET FILE USE CODE

 AND #$7F ;TURN OFF LOCK

 ORA TEMP3

 STA VDFILE+2,X

 JSR WRVDIR

 JMP GOODIO

;

; FPOSTN – POSITION A FILE

FPOSTN JSR LOCSEC ;GO POSITION

 JMP GOODIO ;DONE

;

;

; FVAR – VARIFY A FILE

;

FVAR

 JSR DOPEN ;OPEN FILE

VAR1 JSR LOCNXB ;READ A SECTOR

 BCS VAR2 ;BR IF EOF

 INC DCBCRS ;INCREMENT SECTOR

 BNE VAR1

 INC DCBCRS+1

 JMP VAR1 ;READ THIS ONE

VAR2 JMP GOODIO ;DONE

 PAGE

;

; FDEL – DELETE A FILE

;

FDEL

 JSR DOPEN ;GO OPEN FILE

;

FD2 LDX TEMP1 ;SAVED INDEX

 LDA VDFILE+2,X ;IS FILE LOCKED

 BPL FD3 ;BR NOT LOCKED

 JMP ERRR10

;

FD3

 LDX TEMP1 ;GET SAVED INDEX

 LDA VDFILE,X ;GET DIR TRACK

 STA DCBFDT ;SET AS 1ST FD TRACK

 STA VDFILE+32,X ;SAVE IN LC OF FN

 LDA #$FF ;DELETED FILE MARKER

 STA VDFILE,X ;CLEAR ENTRY

 LDY VDFILE+1,X ;GET DIR SECTOR

 STY DCBFDS ;SET AS 1ST FD SEC

 JSR WRVDIR ;GO WRITE VOLUME DIR

 CLC

FD4 JSR RDFDIR ;GET 1ST FILE DIR SECTOR

 BCS FD7 ;BR IF NO MORE

 JSR MVFCBD ;MOVE DIR TO ZPG

 LDY #FDENT ;POINT Y TO 1ST SEC ENT

FD5 STY TEMP1 ;SAVE Y

 LDA (ZPGFCB),Y ;GET REACK

 BMI FD6 ;BR IF DONE

 BEQ FD6 ;BR IF END OF FILE

 PHA ;SAVE TRK

 INY

 LDA (ZPGFCB),Y ;GET SECTOR

 TAY ;TO Y

 PLA ;GET TRK

 JSR FDSUB ;GO FREE SECTOR

FD6 LDY TEMP1 ;GET DIR INDEX

 INY ;INCR TO NEXT ENTRY

 INY

 BNE FD5 ;BR NOT DONE THIS DIR

 LDA DCBCDT ;GET THIS DIR TRK

 LDY DCBCDS ;AND SECTOR

 JSR FDSUB ;AND GO FREE IT

 SEC ;GO

 BCS FD4 ;READ NEXT DIR

FD7

 JSR WRVTOC

 JMP GOODIO

;

FDSUB

 SEC ;SET FOR RE USE OF SEC

 JSR FRESEC ;GO FREE SECTOR

 LDA #0 ;CLEAR DCB BIT MAP

 LDX #3

FDS1 STA DCBALS,X

 DEX

 BPL FDS1

 RTS

 PAGE

;

; RDIR – PRINT DIRECTORY

;

RDIR

 JSR DCBSUP

 LDA #$FF

 STA DCBVOL

 JSR RDVTOC

 LDA #22 ;SET 21 LINES

 STA TEMP2

 JSR PRCR ;GO PRINT

 JSR PRCR ;PRINT ANOTHER CHAR

 LDX #VML ;VOLUME MSG LENGTH

RDO LDA VOLMES,X ;GET MSG CHAR

 JSR PRINT ;PRINT IT

 DEX ;DECREMENT COUNT

 BPL RDO ;BR IF MORE

;

 STX CNUM+1

 LDA IBSMOD ;MOVE VOL NO FOR

 STA CNUM ;CONVERSION

 JSR PRNUM ;GO PRINT VOL NO

;

 JSR PRCR ;PRINT CR

 JSR PRCR ;AND AGAIN

;

 CLC ;FIRST RECORD

;

RD1 JSR RDVDIR ;GO READ REC

 BCS RD5

 LDX #0 ;SET INDEX=0

RD2 STX TEMP1 ;SAVE INDEX

 LDA VDFILE,X ;GET TRACK

 BEQ RD5 ;BR IF END OF DIR

 BMI RD4 ;BR IF DELETED

;

 LDY #$A0 ;BLANK

 LDA VDFILE+2,X ;GET TYPE

 BPL RD2A ;BR IF NOT LOCKED

 LDY #’\*+$80 ;AST

RD2A TYA ;ACU = AST OR BLANK

 JSR PRINT ;PRINT ACU

;

 LDA VDFILE+2,X ;GET TYPE

 AND #$07 ;MASK OUT MISC

 LDY #3 ;SET INDEX = 3

RD2B LSRA ;SHIFT OUT LSB

 BCS RD2C ;BR IF TYPE BIT OUT

 DEY ;DEC INDEX

 BNE RD2B ;BR IF NOT ACC BITS

RD2C

 LDA FTTAB,Y ;GET TYPE CODE

 JSR PRINT ;PRINT IT

 LDA #$A0 ;BLANK

 JSR PRINT ;PRINT

;

 LDA VDFILE+33,X ;MOVE FILE LENGTH

 STA CNUM ;TO CNUM

 LDA VDFILE+34,X

 STA CNUM+1

 JSR PRNUM ;GO PRINT NUMBER

 LDA #$A0 ;BLANK

 JSR PRINT ;PRINT

;

 INX

 INX

 INX

 LDY #29

RD3 LDA VDFILE,X ;GET CHAR

 JSR PRINT ;PRINT CHAR

 INX

 DEY

 BPL RD3

RD3A

 JSR PRCR ;GO PRINT CR

RD4 JSR VDINC ;INCR INDEX

 BCC RD2 ;BR IF MORE IN DIR

 BCS RD1 ;GO READ NEXT DIR SECT

;

RD5 JMP GOODIO ;DONE

;

PRCR

 LDA #$8D ;CR

 JSR PRINT ;PRINTED

 DEC TEMP2 ;DEC LINE COUNTER

 BNE PRCR1 ;BR IF NOT ZERO

 JSR GETKEY ;WAIT FOR INPUT

 LDA #21 ;RESET LINE COUNTER

 STA TEMP2

PRCR1 RTS ;DONE

 PAGE

PRNUM

 LDY #2 ;3 DIGITS

PRN1 LDA #0 ;INIT DIGIT TO ZERO

 PHA ;SAVE IT

;

PRN2 LDA CNUM ;GET NUMBER

 CMP CVTAB,Y ;IF NUM < CVTAB ENTRY

 BCC PRN3 ;THEN DONE THIS DIGIT

;

 SBC CVTAB,Y ;SUBTRACT TABLE ENTRY

 STA CNUM ;FROM NUM

 LDA CNUM+1

 SBC #0

 STA CNUM+1

 PLA ;INCREMENT DIGIT

 ADC #0

 PHA

 JMP PRN2 ;TRY AGAIN

;

PRN3

 PLA ;GET DIGIT

 ORA #$B0 ;ADD ASCII

 JSR PRINT ;PRINT IT

 DEY ;DECREMENT DIGIT COUNT

 BPL PRN1 ;BR IF MORE DIGIT

;

 RTS ;DONE

 PAGE

;

; CLCFCB – GET FCB VIA INDEX AND MOVE IT

;

CLCFCB

;

 JSR MVFCBP ;MOVE FCB PTR TO ZPG

 LDY #0

 STY CCBSTA

CF3 LDA (ZPGFCB),Y ;MOVE FCB TO

 STA FCB,Y ;FCB WORK AREA

 INY

 CPY #FCBLEN

 BNE CF3

;

 CLC ;DONE

 RTS

;

; RTNFCB – MOVE FCB FROM WORK AREA TO FCB

;

RTNFCB

 JSR MVFCBP ;MOVE FCB ADR TO ZPG

;

 LDY #0

RF1 LDA FCB,Y

 STA (ZPGFCB),Y

 INY

 CPY #FCBLEN

 BNE RF1

 RTS

 PAGE

;

; FFMT – EXECUTE FORMAT REQUEST

;

FFMT

 JSR DCBSUP ;SET UP DCB

 LDA #IBFMT

 JSR DCBIO2

 LDA DCBVOL ;SET VOL NO

 EOR #$FF

 STA VVOLNO

 LDA #17

 STA VALCA1 ;ALOCATE BYTE 1

 LDA #1

 STA VALCA2 ;ADD BYTE 2

;

 LDX #VSECAL-VTOC

 LDA #0

NT1 STA VTOC,X ;CLEAR SECTOR AREA

 INX

 BNE NT1

;

 LDX #3\*4 ;START AT TRACK 3

NT2 CPX #35\*4 ;END AT TRACK 35

 BEQ NT4

 LDY #3 ;4 BYTES OF INFO

NT3 LDA ALC10S,Y ;10 SECTORS ALLOCATE

 STA VSECAL,X

 INX

 DEY

 BPL NT3

 CPX #17\*4 ;AT TRACK 17

 BNE NT2 ;BR IF NOT

 LDX #18\*4 ;SKIP TO 18

 BNE NT2

;

NT4 JSR WRVTOC ;WRITE NEW VTOC

;

 LDX #0

 TXA

NT5 STA VOLDIR,X ;CLEAR VOLDIR

 INX

 BNE NT5

;

 JSR MVVDBA ;MOVE BUF PTRS

;

 LDA #17 ;TRACK 17

 LDY VNOSEC

 DEY

 DEY

 STA IBTRK ;INTO IOB

NT6 STA VDLTRK ;INTO LINK

NT7 STY VDLSEC

 INY

 STY IBSECT

 LDA #IBCWTS

 JSR DCBIO2

 LDY VDLSEC

 DEY ;DECREMENT SECTOR

 BMI NT8 ;BR LAST WRITTEN

 BNE NT7 ;BR NOT LAST

 TYA ;LAST, SET LINK TRK=0

 BEQ NT6

;

NT8

 JSR DLDSUP ;GO SET UP FOR DOSLDR

 JSR WBOOT ;GO WRITE THE BOOT

 JMP GOODIO ;DONE

 PAGE

;

; DLDSUP – SET UP FOR DOSLDR

;

DLDSUP

 LDA CCBBSA

 STA IBBUFP+1 ;START ADR

 LDA #0

 STA IBBUFP

 LDA DCBVOL ;VOL

 EOR #$FF

 STA IBVOL

 RTS

 PAGE

;

; MVFCBX – MOVE FCB ADRS TO ZPGFCB

;

MVFCBP LDX #0 ;MOVE FCB ADR

 BEQ MVF1

MVFCBD LDX #2 ;MOVE FCB DIR BUFF

 BNE MVF1

MVFCBS LDX #4 ;MOVE FCB SECTOR BUFF

;

MVF1

 LDA CFCBAD,X ;DO THE MOVE

 STA ZPGFCB

 LDA CFCBAD+1,X

 STA ZPGFCB+1

 RTS

;

; CLRSEC – CLEAR SECTOR

;

CLSEC

 LDA #0

 TAY

CS1 STA (ZPGFCB),Y

 INY

 BNE CS1

 RTS

 PAGE

;

; WRSECT – WRITE CURRENT SECTOR IF REQD

;

WRSECT

 BIT DCBWRF ;GET WRITE REQD FLAG

 BVS WRSGO ;BR IF WRITE SECTOR REQD

 RTS ;RTS

;

WRSGO

 JSR MVSBA ;GO MOVE SECT BUFF ADR

;

 LDA #IBCWTS ;GET COMMAND

 JSR DCBIO ;GO FILL IN IOB AND DO IO

;

 LDA #$BF ;SET WRITE SECTOR REQD BIT OFF

 AND DCBWRF

 STA DCBWRF

 RTS ;DONE

 PAGE

;

; WRFDIR – WRITE FILE DIRECTORY IF REQD

;

WRFDIR

 LDA DCBWRF ;GET WRITE REQD FLAG

 BMI WRFDGO ;BR IF WRITE DIR REQD

 RTS ;DONE IF NOT

;

WRFDGO

 JSR MVFDBA

;

 LDA #IBCWTS ;GET WRITE CMD

 JSR DCBIO ;GO FILL IN IOB AND DO I/O

;

 LDA #$7F ;TURN WRITE DIR REQD BIT OFF

 AND DCBWRF

 STA DCBWRF

 RTS ;DONE

;

; MVFDBA – MOVE FILE DIRECTORY BUFF ASDR TO IOD

;

MVFDBA

 LDA CFCBDR ;MOVE ADR

 STA IBBUFP

 LDA CFCBDR+1

 STA IBBUFP+1

 LDX DCBCDT ;GET TRACK

 LDY DCBCDS ;GET SECTOR

 RTS

 PAGE

;

; RDFDIR – READ FILE DIRECTORY

;

RDFDIR

 PHP ;SAVE STATUS

 JSR WRFDIR ;GO WRITE CURRENT DIR IF REQD

 JSR MVFDBA ;GO MOVE DBUFF ADR TO ZPG

 JSR MVFCBD ;MOVE DBUFF ADR TO ZPG

 PLP ;GET SAVED STATUS

 BCS RFDNXT ;BR IF RD NEXT

;

 LDX DCBFDT ;TRACK

 LDY DCBFDS ;SECTOR

 JMP RFDIO1 ;GO READ

;

RFDNXT

 LDY #FDLTRK ;GET LINK TRACK

 LDA (ZPGFCB),Y

 BEQ RFDNL ;NR NO LINK

 TAX ;PUT TRACK INTO X

 INY

 LDA (ZPGFCB),Y ;SET LINK SECTOR

 TAY ;PUT SECTOR INTO Y

 JMP RFDIO1 ;GO DO I/O

;

RFDNL

 LDA CCBREG ;THIS A WRITE

 CMP #CRQWR

 BEQ RFDNL1 ;BR IF WRITE

 SEC ;SET EOF

 RTS ;RETURN

;

RFDNL1

 JSR GETSEC ;GET A SECTOR

 LDY #FDLSEC

 STA (ZPGFCB),Y ;PUT IN LINK

 PHA ;SAVE SECTOR

 DEY

 LDA DCBATK ;GET TRACK

 STA (ZPGFCB),Y ;PUT IN LINK

 PHA ;SAVE TRACK

 JSR WRFDGO ;GO WRITE OLD DIR DEC

;

 JSR CLRSEC ;CLEAN OUT DIR

 LDY #FDFRS ;SET NEW DIR SEC 1ST REL

 LDA DCBDNF ;FILE SECTOR

 STA (ZPGFCB),Y

 INY

 LDA DCBDNF+1

 STA (ZPGFCB),Y

;

 PLA ;GET SAVED TRACK

 TAX ;INTO X

 PLA ;GET SAVED SECTOR

 TAY ;INTO Y

 LDA #IBCWTS ;SET WRITE CMD
 BNE RFDIO2 ;GO DO I/O

;

RFDIO1 LDA #IBCRTS ;SET READ CMD

RFDIO2 STX DCBCDT ;SET CURR TRACK

 STY DCBCDS ;SET CURR SECTOR

 JSR DCBIO ;GO I/O

;

RDFDC LDY #FDFRS ;GET POINTER TO FIRST RE SECTOR

 LDA (ZPGFCB),Y ;GET FRS

 STA DCBDFS ;SET INTO DCB

 CLC

 ADC DCBDMS ;ADD MAX SECTORS

 STA DCBDNF ;PUT INTO DCB

;

 INY ;DO SAME FOR HI BYTE

 LDA (ZPGFCB),Y

 STA DCBDFS+1

 ADC DCBDMS+1

 STA DCBDNF+1

;

 CLC

 RTS ;DONE

 PAGE

;

; RDSECT – READ A SECTOR

;

RDSECT

 JSR MVSBA ;GO MOVE SECTOR BUFFER ADR

;

 LDA #IBCRTS

 JMP DCBIO ;GO DO I/O

;

; MVSBA – MOVE SECTOR BUFFER ADR FOR I/O

;

MVSBA

 LDY CFCBSB ;GET SECTOR BUFF ADR

 LDA CFCBSB+1

MSB1 STY IBBUFP ;SET IOB SECTOR

 STA IBBUFP+1 ;BUFF PTR

 LDX DCBTRK ;GET TRACK

 LDY DCBSEC ;GET SECTOR

 RTS ;RTN

 PAGE

;

; RDVTOC – READ VTOC

; WRVTOC – WRITE VTOC

;

RDVTOC

 LDA #IBCRTS ;READ

 BNE VTIO

WRVTOC

 LDA #IBCWTS ;WRITE

;

VTIO LDY AVTOC ;MOVE BUFF ADR

 STY IBBUFP

 LDY AVTOC+1

 STY IBBUFP+1

;

 LDX DCBVTN ;GET TRACK

 LDY #0

 JMP DCBIO ;GO DO I/O

 PAGE

;

; RDVIR – READ VOLUME DIRECTOR

;

RCVIR

 PHP ;SAVES STATUS

 JSR MVVDBA

;

 PLP ;GET STATUS

 BCS RVDA ;BR IF R0 NEXT

;

RVDC LDY VDIRSC ;GET LINK SECTOR

 LDX VDIRTK ;GET FIRST TRK

 BNE RVDGO ;GO READ

;

RVDA

 LDX VDLTRK ;GET SECTOR

 BNE RDVC ;BR IF A LINK

 SEC ;SET END OF DIR

 RTS

;

RDVC LDY VDLSEC ;GET SECTOR

RVDGO

 STX CVDTRK ;SET CUR TRACK

 STY CVDSEC ;SET CUR SECTOR

 LDA #IBCRTS ;GET CMD

 JSR DCBIO ;GO DO I/O

 CLC

 RTS

 PAGE

;

; WRVDIR – WRITE VOLUME DIRECTORY SECTOR

;

WRVDIR

 JSR MVVDBA

;

 LDX CVDTRK ;CURRENT TRACK

 LDY CVDSEC ;CURRENT SECTOR

 LDA #IBCWTS ;WRITE COMMAND

 JMP DCBIO ;GO DO I/O

;

; MVVDBA – MOVE VOL DIR BUF ADR TO IOB

;

MVVDBA

 LDA AVOLDR ;MOVE ADR

 STA IBBUFP

 LDA AVOLDR+1

 STA IBBUFP+1

 RTS

 PAGE

;

; DCBIO – DO I/O FOR A DCB

;

DCBIO

 STX IBTRK ;TRACK

 STY IBSECT ;SECTOR

DCBIO2

 STA IBCMD ;COMMAND

 CMP #IBCWTS

 BNE DCBIO1

 ORA DCBWRF

 STA DCBWRF

DCBIO1

 LDA DCBVOL ;VOL

 EOR #$FF ;UNINVERT VOL BITS

 STA IBVOL

 LDA DCBSLT ;SLOT

 STA IBSLOT

 LDA DCBDRV ;DRIVE

 STA IBDRVN

 LDA DCBSDL ;LENGTH

 STA IBDLEN

 LDA DCBSDL+1

 STA IBDLEN+1

 LDA #1 ;IOB TYPE

 STA IBTYPE

;

 LDY AIOB ;IOB ADR

 LDA AIOB+1

 JSR DISKIO ;GO DO I/O

;

 LDA IBSMOD

 STA CCBVOL

 LDA #$FF ;RESET VOL

 STA IBVOL

 BCS BADIO ;BR IF BAD

 RTS ;RTN IF GOOD

;

BADIO LDA IBSTAT ;GET STATUS

 LDY #CREVMM

 CMP #IBVMME ;WAS IT VOLUME MISMATCH

 BEQ BD2 ;BR IF YES

 LDY #CREPRO

 CMP #IBWPER

 BEQ BD2

 LDY #CREIOE

BD2 TYA

 JMP ERRORB ;GO RTN

 PAGE

;

; LOCNXB – LOCATE NEXT BYTE

;

LOCNXB

 LDA DCBCRS ;IS THE CURRENT RELATIVE SECTOR

 CMP DCBCMS ;EQUAL TO THE CURRENT MEM SECTOR

 BNE LNB1 ;BR IF NOT EQ

 LDA DCBCRS+1

 CMP DCBCMS+1

 BEQ LNB8 ;BR IF REQD SECTOR IN MEM

;

LNB1 ;NEED A DIFFERENT SECTOR IN MEM

 JSR WRSECT ;GO WRITE SECTOR(IF REQD)

;

LNB2 LDA DCBCRS+1 ;IS CURRENT REL SECTORY

 CMP DCBDFS+1 ;IS CURRENT DIRECTORY (LOW LIMIT)

 BCC LNB4 ;BR IF IN A PREVIOUS DIR

 BNE LNB3 ;BR IF MAYBE IN THIS ONE

 LDA DCBCRS ;TEST LOW BYTES

 CMP DCBDFS

 BCC LNB4 ;BR IF IN PREVIOUS DIR

;

LNB3 LDA DCBCRS+1 ;IS CURRENT REL SECTOR

 CMP DCBDNF+1 ;IN CURRENT DIRECTOR (HI LIMIT)

 BCC LNB6 ;BR IF IN THIS ONE

 BNE LNB4 ;BR IF IN A NEXT DIR

 LDA DCBCRS

 CMP DCBDNF

 BCC LNB6 ;BR IF IN THIS ONE

; ;REQD SECTOR IN A NEXT DIRECTORY

LNB4 JSR RDFDIR ;GO READ NEXT FILE DIR

 BCC LNB2 ;BR NXT AVAIL

 RTS ;RETURN IF EOF DIR

;

;

LNB6

 SEC ;CALCULATE DISPL INTO DIR

 LDA DCBCRS ;REQD REL SECTOR MINUS

 SBC DCBDFS

 ASLA ;TIMES 2

 ADC #FDENT ;PLUS DISPL TO 1ST

 TAY

 JSR MVFCBD ;MOVE ADR TO ZPG

 LDA (ZPGFCB),Y ;GET TRACK

 BNE LNB7 ;BR IF NOT ZERO

 LDA CCBREQ

 CMP #CRQWR ;WRITE

 BEQ LNB7A

 SEC

 RTS

LNB7A JSR GNWSEC ;GO GET A NEW SECTOR

 JMP LNBCON

LNB7 STA DCBTRK ;SET TRK INTO DCB

 INY

 LDA (ZPGFCB),Y ;GET SECTOR

 STA DCBSEC ;PUT INTO DCB

 JSR RDSECT ;GO READ SECTOR

LNBCON LDA DCBCRS ;MOVE CUR REL SECTOR

 STA DCBCMS

 LDA DCBCRS+1 ;TO CUR MEM SECTOR

 STA DCBCMS+1

;

LNB8

 JSR MVFCBS ;MOVE SECTOR BUFF ADR TO ZP

 LDY DCBCSB ;GET SECT BYTE

 CLC ;CARRY CLEAR = ALL OK

 RTS ;DONE

 PAGE

;

;

GNWSEC ;NEED NEW SECTOR

 STY TEMP2 ;SAVE DIR INDEX

 JSR GETSEC ;GET A SECTOR

 LDY TEMP2

 INY

 STA (ZPGFCB),Y ;SET NEW TRACK

 STA DCBSEC

 DEY

 LDA DCBATK

 STA (ZPGFCB),Y ;SET NEW TRACK

 STA DCBTRK

;

 JSR MVFCBS

 JSR CLRSEC ;GO CLEAR SECTOR

;

;

 LDA #$C0 ;INDICATE BOTH

 ORA DCBWRF ;DIR AND SECTOR

 STA DCBWRF ;MUST BE WRITTEN

 RTS ;DONE

 PAGE

;

; INCRRB – INCREMENT RELATIVE RECORD BYTE

;

INCRRB

 LDX DCBCRR ;MOVE BYTE JUST READ OR WRITTEN

 STX CCBRRN

 LDX DCBCRR+1

 STX CCBRRN+1

 LDX DCBCRB ;X=REL BYTE (LOW)

 LDY DCBCRB+1 ;Y=REL BYTE HI

 STX CCBBYT

 STY CCBBYT+1

 INX ;INC REL BYTE (LOW)

 BNE INCR1 ;BR IF NO CARRY

 INY ;INC REL BYTE (HI)

;

INCR1 CPY DCBRCL+1 ;REL BYTE=REC LENGTH

 BNE INCR2 ;BR IF NOT

 CPX DCBRCL ;TEST LOW BYTES

 BNE INCR2

 LDX #0

 LDY #0 ;RESET REL BYTE TO ZERO

 INC DCBCRR ;AND INCR

 BNE INCR2 ;RELATIVE RECORD

 INC DCBCRR+1

;

INCR2 STX DCBCRB ;SAVE NEW RELATIVE BYTE

 STY DCBCRB+1

;

 RTS

 PAGE

;

; INCSCB – INCREMENT SECTOR BYTE

;

INCSCB

 INC DCBCSB ;INC SECTOR BYTE

 BNE INCS2 ;BR IF NOT FULL

 INC DCBCRS ;AND INCR

 BNE INCS2 ;RELATIVE SECTOR

 INC DCBCRS+1

;

;

INCS2

 RTS ;DONE

 PAGE

;

; MIBDA – MOVE AND INCREMENT CCBDAT

;

MIBDA

 LDY CCBBBA ;Y=ADR LOW

 LDX CCBBBA+1 ;X=ADR HI

 STY ZPGFCB ;PUT ADR INTO ZPG

 STX ZPGFCB+1

;

 INC CCBBBA ;INC ADR LOW

 BNE MIB1 ;BR IF NOT ZERO

 INC CCBBBA+1 ;INC ADR HI

MIB1 RTS ;DONE

;

; DTBLN – DCREMENT BLOCK LENGTH AND TEST ZERO

;

DTBLN

 LDY CCBBLN ;GET LEN LOW

 BNE DTB1 ;BR IF NOT ZERO

 LDX CCBBLN+1 ;GET LEN HI

 BEQ DTB2 ;BR IF LEN=0

 DEC CCBBLN+1 ;DEC LEN (HIGH)

DTB1 DEC CCBBLN ;DEC LEN (LOW)

 RTS ;DONE

;

DTB2 JMP GOODIO ;FINISHED BLOCK

 PAGE

;

; FNDFIL – FIND FILE NAME IN VOLUME DIR

;

FNDFIL

 JSR RDVTOC ;GO GET VTOC

 LDA CCBFN1 ;MOVE FN PTR

 STA ZPGFCB ;TO ZERO PAGE

 LDA CCBFN1+1

 STA ZPGFCB+1

 LDA #1

FF1 STA TEMP2

 LDA #0

 STA DCBVDR

 CLC

FF2

 INC DCBVDR

 JSR RDVDIR ;GO GET VDIR SECTOR

 BCS FF4A

 LDX #0 ;SET FOR 1ST FILE

;

FF3 STX TEMP1 ;SAVE INDEX

 LDA VDFILE,X ;GET FILE TRK

 BEQ FF6 ;BR IF LAST ENTRY

 BMI FF7 ;BR DELETED ENTRY

 LDY #0 ;X=X+3

 INX

 INX

FF4 INX

 LDA (ZPGFCB),Y ;GET FN CHAR

 CMP VDFILE,X ;COMPARE TO ENTRY CHAR

 BNE FF5 ;BR IF NOT SAME

 INY

 CPY #30 ;ALL 30 CHARS

 BNE FF4 ;BR IF NOT

 LDX TEMP1 ;GET INDEX

 CLC ;FILE FOUND

 RTS ;RETURN

;

FF5

 JSR VDINC

 BCC FF3

 BCS FF2

;

FF6 LDY TEMP2 ;LOOKING FOR DELETED

 BNE FF1 ;BR IF NOT (DO)

;

FF7 LDT TEMP2 ;LOOKING FOR EMPTY

 BNE FF5 ;BR IF NOT

;

MVFN

 LDY #0 ;HAVE NEW ENTRY

 INX

 INX

FF8 INX

 LDA (ZPGFCB),Y ;MOVE FILE NAME

 STA VDFILE,X

 INY

 CPY #30

 BNE FF8

;

 LDX TEMP1 ;GET INDEX

 SEC ;SET NOT OLD

 RTS ;DONE

VDINC

 CLC

 LDA TEMP1

 ADC #35

 TAX

 CPX #VDFLEN

 RTS

FF4A

 LDA #0

 LDY TEMP2

 BNE FF1

 JMP ERROR9

 PAGE

;

; GETSEC – GET A SECTOR

;

GETSEC

 LDA DCBATK ;GET ALLOCATED TRK

 BEQ GSS1 ;BR IF NONE

;

GSO

 DEC DCBALS ;DECREMENT SECTOR NO

 BMI CS2 ;BR IF NO SECTORS REM

;

 CLC

 LDX #4 ;4 BYTE SHIFT

GS1 ROL DCBABM-1,X ;SHIFT BYTE LEFT

 DEX

 BNE GS1

 BCC GS0 ;BR IF NO SECTOR

;

 INC DCBNSA

 BNE GS1A

 INC DCBNSA+1

GS1A

 LDA DCBALS ;GET ALLOCATED SECTOR

 RTS ;RETURN

;

CS2 LDA #0 ;CLEAR ALLOCATED

 STA DCBATK ;TRK

;

GSS1 LDA #0 ;SET SEARCH STATE=0

 STA TEMP3

 JSR RDVTOC ;GET VTOC

;

GS2

 CLC

 LDA VALCA1 ;GET LAST ALLOCATED TRK

 ADC VALCA2 ;AD (+1) OR (-1)

 BEQ GS3 ;BR IF DECK TO ZERO

 CMP VNOTRK

 BCC GS5 ;BR IF NOT AT OUTER LIMIT

 LDA #$FF ;SET (-1)

 BNE GS4

GS3 LDA TEMP3 ;GET SEARCH STATE

 BNE ERR9 ;BR IF NOT ZERO

 LDA #1 ;SET (+1)

 STA TEMP3 ;SET SEARCH STATE = 1

GS4 STA VALCA2 ;SET NEW (+1) OR (-1)

 CLC

 ADC #17 ;ADD VTOC TRK NO

GS5 STA VALCA1 ;SET NEW LAST ALLOCATED

 STA DCBATK ;PUT IN DCB

;

 TAY ;ALLOCATED TRACK

 ASLA ;TIME 4

 ASLA

 TAY

 LDX #4

 CLC

GS6 LDA VSECAL+3,Y ;MOVE BIT MAP BYTE

 STA DCBABM-1,X

 BEQ GS7 ;BR IF NO BITS ON

 SEC ;SET HAVE A SECTOR

 LDA #0 ;CLEAR VTOC BYTE

 STA VSECAL+3,Y

GS7 DEY

 DEX

 BNE GS6 ;BR IF MORE TO MOVE

 BCC GS2

 LSR WRVTOC ;GO WRITE VTOC

 LDA VNOSEC ;GET NO SECTORS

 STA DCBALS ;SET IN DCB SECTOR BYTE

 BNE GS0 ;GO ALLOCATED SECTOR

ERR9 JMP ERROR9

 PAGE

;

; FRETRK – FREE TRACK OF SECTORS

;

FRETRK

 LDA DCBATK ;GET ALLOCATED TRACK

 BNE FT1 ;BR IF NONE

 RTS ;DONE

FT1 PHA

 JSR RDVTOC ;GET VTOC

 LDY DCBALS ;GET SECTORS

 PLA ;GET TRACK

 CLC ;SET FREE

 JSR FRESEC ;GO FREE

 LDA #0 ;CLEAR ALLOCATED TRK

 STA DCBATK

 JMP WRVTOC ;WRITE VTOC

;

; FRESEC – FREE A SECTOR

; A=TRK, Y=SECTOR, C=ON/OFF

;

FRESEC

FS1 LDX #252 ;4 BYTE SHIFT

FS2 ROR DCBABM-252,X ;SHIFT IN CARRY

 INX ;NEXT BYTE

 BNE FS2 ;BR IF NOT DONE

 INY ;INC SECTOR NO

 CPY VNOSEC ;NORMAL

 BNE FS1 ;BR IF NOT

;

 ASLA ;TRACK\*4

 ASLA

 TAY

 BEQ FS4

 LDX #4

FS3 LDA DCBABM-1,X ;GET BIT MAP BYTE

 ORA VSECAL+3,Y ;GET BIT MAP BYTE

 STA VSECAL+3,Y ;OR WITH VTOC BM

 DEY

 DEX

 BNE FS3

FS4 RTS ;DONE

 PAGE

;

; LOCSEC – LOCATE SECTOR FOR RECORD I/O

;

; RELSEC = (REL REC \* RECLEN + RELBYTE)/256

; SECBYT = REMAINDER

;

LOCSEC

 LDA CCBRRN ;RELATIVE RECORD NUMBER

 STA DCBCSB ;TO CSB FOR MULT

 STA DCBCRR ;AND CRR FOR SAVE

 LDA CCBRRN+1

 STA DCBCRS

 STA DCBCRR+1

 LDA #0

 STA DCBCRS+1 ;HIGH CRS=0

 LDY #16 ;16 BIT MULT

;

LS1 TAX ;SAVE MS BYTE

 LDA DCBCSB

 LSRA ;IF NO CARRY THEN NO PART PROD

 BCS LS1A

 TXA

 BCC LS2

LS1A CLC

 LDA DCBCRS+1 ;FPORM PARTIAL PROD

 ADC DCBRCL

 STA DCBCRS+1

 TXA

 ADC DCBCRL+1

;

LS2 RORA ;MULT BY 2

 ROR DCBCRS+1

 ROR DCBCRS

 ROR DCBCSB

 DEY ;DEC BIT COUNT

 BNE LS1 ;BR IF MORE BITS

;

 LDA CCBBYT ;LADD REL BYTE RESULT

 STA DCBCRB ;(SAVE REL BYTE)

 ADC DCBCSB

 STA DCBCSB

 LDA CCBBYT+1

 STA DCBCRB+1 ;(SAVE REL BYTE)

 ADC DCBCRS

 STA DCBCRS

 LDA #0

 ADC DCBCRS+1

 STA DCBCRS+1

 RTS

 PAGE

ERROR1 LDA #CREFUN

 BNE ERRORA

ERROR2 LDA #CRERR

 BNE ERRORA

ERROR3 LDA #CREMRE

 BNE ERRORA

ERROR4 LDA #CREPRO

 BNE ERRORA

ERROR5 LDA #CREEOF

 BNE ERRORA

ERROR6 LDA #CREFNF

 BNE ERRORA

ERROR9 LDA #CRENSA

 BNE ERRORA

ERROR10 LDA #CREFLK

 BNE ERRORA

GOODIO LDA CCBSTA

 CLC ;CARRY=CLR

 BCC RETURN ;GO RETURN

ERRORA

ERRORB SEC ;CARRY=SET

RETURN

 PHP

 STA CCBSTA ;SET STA

 JSR RTNFCB ;GO RTN FCB

 PLP ;GET STATUS

 LDX ENTSTK ;GET ENT STACK

 TXS ;RESTORE STACK

 RTS ;DONE

EC2

 PAGE

;

; MISC DOS WORK CELLS

;

CVDTRK DB 0 ;CUR VOL DIR TRK

CVDSEC DB 0 ;CUR VOL DIR SECTOR

CURCCB DB 0,0 ;CURRENT CCB ADR

ENTSTK DB 0 ;ENTRY STACK POINTER

TEMP1 DB 0 ;TEMP BYTE 1

TEMP2 DB 0 ;TEMP BYTE 2

TEMP3 DB 0 ;TEMP BYTE 3

ENTSLT DB 0 ;BOOT SLOT SAVED

ALC10S DB 0,0,$F8,$FF ;ALLOCATATION TRACK BIT MAP

CVTAB DB 1,10,100 ;CONVERSION TABLE

FTTAB DB11 "TBAI" ;FILE TYPE CONVERSION TABLE

VOLUMES DB11 " EMULOV KSID"

VML EQU \*-VOLMES-1

 PAGE

;

; VTOC RECORD AREA

;

VTOC

VDOST DB 2 ;DOS TYPE

VDIRTK DB 17 ;COLUME DIRECTORY SECTOR

VDIRSC DB 12 ;VOLUME DIRECTORY SECTOR

VDOSRN DB 2 ;DOS RELEASE NUMBER

 DB 0 ;SPARE

 DB 0 ;SPARE

VVOLNO DB 0 ;VOLUME NUMBER

 RMB 32 ;SPARE

VTDMS DB 122 ;MAX SECTORS IN A FILE DIR

VSPARE RMB 8 ;SPARES

;

VALCA1 DB 17 ;ALOCATION ALGORITHM BYTE 1

VALCA2 DB 1 ;AA BYTE 2

VALCA3 DB 0 ;AA BYTE 3

VALCA4 DB 0 ;AA BYTE 4

VNOTRK DB 35 ;NO TRACKS ON VOL

VNOSEC DB 13 ;NO SECTORS PER TRACK

VSECLN DB @@256 ;NO BYTES PER SECTOR

;

VSECAL EQU \* ;SECTOR ALLOCATION AREA

; SECTORS ALLOCATED BY BIT MAP

; 4 BYTES OF BITS PER TRACK

; LEFT MOST BIT REPRESENTS SECTOR N

; WHERE N=NO SECTORS PER TRACK

;

;

 PAGE

 ORG VTOC+256

;

; VOLUME DIRECTORY AREA

;

VOLDIR

VDTCDE DB 2 ;VOLUME DIRECTORY TYPECODE

VDLTRK RMB 1 ;VD LINK TRACK

VDLSEC RMB 1 ;VD LINK SECTOR

VDNF RMB 1 ;VD NUMBER FILES THIS SECTOR

VDSPAR RMB 7 ;SPARES

;

VDFILE EQU \* ;FILE ALLOCATION AREA (7 FILES)

; EACH FILE

; FILE DIR TRK

; FILE DIR SECTOR

; FILE USE CODE

; FILE NAME (30)

; FILE SECTOR COUNT (2)

;

 ORG VOLDIR+256

VDEND EQU \*

VDLEN EQU \*-VOLDIR

VDFLEN EQU \*-VDFILE

 PAGE

;

; COMMAND CONTROL BLOCK (CCB)

;

CCB

CCBREQ RMB 1 ;USER REQUEST BYTE

CRQNUL EQU 0 ;0-NO REQUEST

CRQOPN EQU 1 ;1-OPEN FILE

CRQCLS EQU 2 ;2-CLOSE FILE

CRQRD EQU 3 ;3-READ DATA

CRQWR EQU 4 ;WRITE DATA

CRQDEL EQU 5 ;5-DELETE FILE

CRQDIR EQU 6 ;6-READ DIRECTORY

CRQLCK EQU 7 ;7-LOCK FILE

CRQUNL EQU 8 ;8-UNLOCK FILE

CRQRNM EQU 9 ;9-RENAME

CRQPOS EQU 10 ;10-POSITION FILE

CRQFMT EQU 11 ;11-FORMAT

CRQVAR EQU 12 ;12-VERIFY

CRQMAX EQU 13

;

CCBBSA ;FORMAT – BOOT START ADR PAGE

CCBRQM RMB 1 ;REQUEST MODIFIER BYTE

CRMNUL EQU 0 ;NO MODIFIER

CRMNBT EQU 1 ;R/W – 1 – NEXT BYTE

CRMNBL EQU 2 ;R/W – 2 – NEXT BLOCK

CRMSBT EQU 3 ;R/W – 3 – SPECIFC BYTE
CRMSBL EQU 4 ;R/W – 4 – SPECIFIC BLOCK

CRMMAX EQU 5

;

CCBRRN ;I/O – RELATIVE RECORD NUMBER

CCBFN2 ;RENAME – FILE NAME 2 PTR

CCBRLN RMB 2 ;OPEN – RECORD LENGTH

;

CCBBYT ;I/O – RELATIVE RECORD NO(2 BYTES)

CCBVOL RMB 1 ;OPEN – VOL NO.

CCBDRV RMB 1 ;OPEN – DRIVE

;

CCBBLN ;I/O – BLOCK LENGTH (2 BYTES)

CCBSLT RMB 1 ;OPEN – SLOT NO.

CCBFUC RMB 1 ;OPEN - FILE USE CODE

;

CCBFN1 ;OPEN, DELETE, LOCK, UNLOCK, RENAME – FILENAME P

CCBBBA ;BLCOKK I/O – BLOCK BUFFER PTR

CCBDAT RMB 2 ;BYTE I/O – DTA BYTE

;

CCBSTA RMB 1 ;RESULT STATUS

CREFUN EQU 1 ;FCB UNALLOCATED

CRERR EQU 2 ;CCB REQ RANGE ERR

CREMRE EQU 3 ;REQ MOD RANGE ERR

CREPRO EQU 4 ;WRITE PROTECT

CREEOF EQU 5 ;END OF FILE ON READ

CREFNF EQU 6 ;FILE NOT FOUND

CREVMM EQU 7 ;VOL MIS MATCH

CREIOE EQU 8 ;I/O ERR

CRENSA EQU 9 ;NO SECTORS AVAILABLE

CREFLK EQU 10 ;FILE LOCKED

;

CCBSM RMB 1 ;STATUS MODIFIER

CCBFCB RMB 2 ;FCB PTR

CCBDBP RMB 2 ;DIR BUF PTR

CCBSBP RMB 2 ;SECTOR BUF PTR

CCBSPR RMB 4 ;SPARE

CCBLEN EQU \*-CCB ;CCB LENGTH

CFCBAD EQU CCBFCB

CFCBDR EQU CCBDBP

CFCBSB EQU CCBSBP

 PAGE

;

; FILE CONTROL BLOCK (FCB) DEFINITION

; DCB – FILE DATA CONTROL BLOCK

;

FCB

;

; DATA CONTROL BLOCK

;

FCBDCB

DCBFDT RMB 1 ;1ST FILE DIRECTORY TRACK

DCBFDS RMB 1 ;1ST FILE DIRECTORY SECTOR

DCBCDT RMB 1 ;CURRENT FILE DIRECTORY

DCBCDS RMB 1 ;CURRENT FILE DIRECTORY

DCBWRF RMB 1 ;WRITE REQD FLAG

; ;$80=WRITE FILE DIR

; ;$40=WRITE SECTOR DIR

DCBTRK RMB 1 ;SECTOR TRACK ADR

DCBSEC RMB 1 ;SECTOR ADR

DCBVDR RMB 1 ;VOL DIR REC

DCBVDI RMB 1 ;VOL DIR INDEX

DCBDMS RMB 2 ;MAX NO DIRECTORY SECTORS

DCBDFS RMB 2 ;CURRENT DIR 1ST REL SECTRO

DCBDNF RMB 2 ;REL SECTOR OF NXT DIR

DCBCMS RMB 2 ;SECTOR CURRENTLY IN MEMORY

DCBSDL RMB 2 ;SECTOR DATA LENGTH

DCBCRS RMB 2 ;CURRENT RELATIVE SECTOR

DCBCSB RMB 2 ;REL SECTOR OF NXT DIR

DCBRCL RMB 2 ;SECTOR CURRENTLY IN MEMORY

DCBCRR RMB 2 ;SECTOR DATA LENGTH

DCBCRB RMB 2 ;CURRENT RELATIVE SECTOR

DCBNSA RMB 2 ;NO SECTORS ALLOCATED

;

DCBALS RMB 1 ;ALLOCATION SECTOR BYTE

DCBATK RMB 1 ;ALLOCATION TRACK

DCBABM RMB 4 ;ALLOCATION TRACK SECTOR BIT MAP

;

DCBFUC RMB 1 ;FILE USE CODE

DCBSLT RMB 1 ;SLOT NUMBER

DCBDRV RMB 1 ;DRIVE NUMBER

DCBVOL RMB 1 ;VOLUME DRIVER

DCBVTN RMB 1 ;VTOC TRACK NUMBER

;

DCBSPR RMB 3 ;SPARES

;

DCBLEN EQU \*-FCBDCB ;DCB LENGTH

FCBLEN EQU \*-FCB ;FCB LENGTH

 PAGE

;

; DOS PATCH AREA 1

SDP1

EDP1 EQU ORG2-2

;

; DOSLDR – DOS LOADER AND WRITTER

;

 ORG ORG2

DOSLDR

; GARBAGED BOOT REC 0 HERE

 RMB 254

GRSPG DB 0

GRPGC DB 0

 PAGE

SC3

;

; READ DOS AFTER BOOT

;;

 STX IBSLOT ;SET BOOT SLOT

 STX IBPSLT ;SET PREVIOUS SLOT

 LDA #1 ;SET PREV DRIVE

 STA IBPDRV

 STA IBDRVN

;

 LDA NDPGS ;COPY NO PAGES TO GET

 STA BRWCNT

 LDA #0

 STA IBTRK ;SET TRACK 0

;

 LDA BSDSEC ;COPY START DOS SECTOR

 STA IBSECT

;

 LDA BGNDOS ;COPY STARTR DOS ADR

 STA IBBUFP+1

;

 LDA #IBCRTS ;SET READ

 STA IBCMD

;

 TXA ;SET PREV TRACK = 0

 LSRA

 LSRA

 LSRA

 LSRA

 TAX

 LDA #0

 STA $4F8,X

 STA $478,X

 JSR BOOTIO ;GO READ DOS

;

; DOSINT – INITIALIZE DOS

;

DOSINT

 LDX #$FF

 TXS

 STX IBVOL

 JSR SETVID

 JSR SETKBD

;

DI3 JMP DOSREL ;GO TO POST INIT ROUTINE

 PAGE

WBOOT
 LDA IBBUFP+1 ;GET START OF DOS

 STA BGNDOS ;SAVE IR

 SEC

 LDA ADOSLD+1 ;CALCULATE

 SBC BGNDOS

 STA NDPGS ;NO DOS PAGES

;

 LDA #0

 STA IBTRK ;TRACK=0

 STA IBSECT ;SECTOR=0

 STA IBBUFP

;

 LDA ADOSLD+1 ;GET BOOT START ADR

 STA IBBUFP+1 ;TO BUFP

 STA GRSPG ;TO GARBAGE RECORD

;

 LDA #10 ;NO OF BOOT PAGES

 STA BRWCNT ;TO BOOT I/O COUNTER

 STA BSDSEC

 LDA #$48

 STA GRPGC

;

 LDA #IBCWTS ;SET WRITE

 STA IBCMD

;

 JSR BOOTIO ;GO WRITE BOOT SECTORS

;

 LDA BGNDOS ;SET START OF DOS

 STA IBBUFP+1

;

 LDA NDPGS

 STA BRWCNT

 JSR BOOTIO ;GO WRITE DOS

;

 RTS ;DONE

 PAGE

BOOTIO

 LDA BAIOB+1

 LDY BAIOB

 JSR DISKIO

 LDY IBSECT ;GET SECTOR

 INY ;INCREMENT TO NEXT

 CPY #13 ;AT END OF TRACK

 BNE BI01 ;BR IF NOT SECTOR ZERO

 LDY #0 ;SET TO SECTOR ZERO

 INC IBTRK

BI01 STY IBSECT ;SET NEXT SECTOR

;

 INC IBBUFP+1 ;INCREMENT BUFFER POINTER

 DEC BRWCNT ;DECREMENT PAGE COUNTER

 BNE BOOTIO ;BR IF NOT DONE

 RTS

;

 PAGE

;

; DOS PATCH AREA 1

;

DP1 EQU \*

 BOUND 256

 ORG \*-$20

EC3

NDPGS DB 0

BRWCNT DB 0

BSDSEC DB 0

BGNDOS DB 0

BAIOB DB @@IOB

ADOSLD DB @@DOSLDR

 PAGE

;

; IOB – INPUT / OUTPUT CONTROL BLOCK

; THE IOB IS USED FOR THE INTERFACE

; BETWEEN DOS AND THE DISK I/O ROUTINES

;

IOB

IBTYPE DB 1 ;IOB TYPE CODE

IBSLOT DB 7 ;CONTROLLER SLOT NO.

IBDRVN DB 0 ;DRIVE NUMBER

IBVOL DB $FF ;VOLUME NUMBER

IBTRK DB 0 ;TRACK NUMBER

IBSECT DB 0 ;SECTOR NUMBER

IBDCTP DB @@DCT

IBBUFP DB @@0 ;POINTER TO BUFFER

IBDLEN DB @@256 ;DATA LENGTH

IBCMD DB 0 ;COMMAND

IBCNUL EQU 0 ;0-NULL COMMAND

IBCRTS EQU 1 ;1-READ TRACK, SECTOR

IBCWTS EQU 2 ;2-WRITE TRACK, SECTOR

IBFMT EQU 4 ;4-FORMAT DISK

IBBOOT EQU 8 ;8-WRITE BOOT

IBSTAT DB 0 ;STATUS

IBRERR EQU $80 ;READ ERR

IBDERR EQU $40 ;DRIVE ERR

IBVMME EQU $20 ;VOLUME MISMATCH

IBWPER EQU $10 ;WRITE PROTECT ERROR

IBSMOD DB 0 ;STATUS MODIFIER BYTE

IBPSLT DB 0 ;PREVIOUS SLOT

IBPDRV DB 0 ;PREVIOUS SLOT

IBSPAR RMB 2 ;IOB SPARES

DCT DB 0,1,$EF,$D8

 PAGE

;

; FILE DIRECTORY DEFINITION

;

 ORG 0

FILDIR

FDUCDE RMB 1 ;FILE USE CODE

FDLTRK RMB 1 ;LINK TO NXT DIR TRACK

FDLSEC RMB 1 ;LINK TO NEXT DIR SECTOR

FDNSA RMB 1 ;NO SECTOR ALLOCATED

FDLSDL RMB 1 ;LAST SECTOR DATA LENGTH

FDFRS RMB 2 ;1ST RELATIVE SECTOR IN THIS DIR

FDSPAR RMB 5 ;SPARES

;

FDENT RMB 1 ;START OF FILE ENTRIES (122)

FDTRK EQU 0 ;TRACK

FDSEC EQU 1 ;SECTOR

;

FDLAST EQU FILDIR+256

 PAGE

 END