```
                        **********************************************
                        *         "ST-MON"
                        *
                        * VERSION 2.04 ROM MONITOR FOR 64K ST-2900
                        *  IMPLEMENTING POWERUP INITIALIZATION
                        *  AND DIAGNOSTICS AND "M", "T", "L",
                        *  "D", "F", "S", "A", "B", "E"
                        *  AND "J" COMMANDS
                        *
                        * DIFFERENCES BETWEEN 2732 AND 2764 VERSIONS
                        *  ARE ONLY IN THE VALUE OF THE "EROM" EQUATE
                        *  AND THE "FILLER" SIZE
                        *
                        * REFER TO COPYRIGHT NOTICE AND LICENCE AT END
                        *
                        * Last modified June 19, 1985 3:45 pm
                        **********************************************

                        *
                        * EQUATES
                        *
                000D    CR      EQU     $0D       ASCII CARRIAGE RETURN
                000A    LF      EQU     $0A       ASCII LINE FEED
                001B    ESC     EQU     $1B       ASCII ESCAPE
                0008    BKSPC   EQU     $08       ASCII BACKSPACE
                0007    BELL    EQU     $07       ASCII BELL
                0004    EOT     EQU     $04       END-OF-STRING CODE
                FF20    CPORT   EQU     $FF20     2681 DUART PORT A (CONSOLE)
                FF28    NPORT   EQU     $FF28     2681 DUART PORT B (PRINTER/MODEM)
                FFD4    PAGE0   EQU     $FFD4     6883 SAM PAGE 1 SELECT
                FFD5    PAGE1   EQU     $FFD5     6883 SAM PAGE 0 SELECT
                FFDE    MAP0    EQU     $FFDE     6883 SAM ROM/RAM MAP SELECT
                FFDF    MAP1    EQU     $FFDF     6883 SAM ALL-RAM MAP SELECT
                0000    SRAM    EQU     $0000     START OF RAM
                7FFF    ERAM    EQU     $7FFF     END OF RAM ($3FFF FOR 16K, $7FFF FOR 32/64K)
                A000    SROM    EQU     $A000     START OF ROM
                AFFF    EROM    EQU     $AFFF     END OF ROM ($A7FF-2716, $AFFF-2732, $BFFF-64)
                FEFF    MEND    EQU     $FEFF     END OF MONITOR IN RAM
                CD03    WARMS   EQU     $CD03     FLEX WARMSTART
                D3E1    VRETURN EQU     $D3E1     VECTOR TO "RETURN-TO-FLEX" ROUTINE
                00BB    B9600   EQU     $BB       9600 BAUD RATE CODE
                00CC    B1920O  EQU     $CC       1920O BAUD
                0066    B1200   EQU     $66       1200 BAUD
                0044    B300    EQU     $44       300 BAUD
                00FF    BIPX    EQU     $FF       BAUD RATE CODE SET BY IP3/IP4
                00FF    WIP5    EQU     $FF       WORD LENGTH CODE SET BY IP5
                FFB8    RAMBD1  EQU     $FFB8     512K RAM BOARD #1 CONTROL PORT
                FFBC    RAMBD2  EQU     $FFBC     512K RAM BOARD #2 CONTROL PORT
                0000    BEGFLX  SET     0

A000                            ORG     SROM
                        *
                        * SERIAL PORT INITIALIZATION CONSTANTS
                        *
A000 FF                 BAUDRA  FCB     BIPX      BAUD RATE FOR PORT A RX & TX
A001 44                 BAUDRB  FCB     B300      BAUD RATE FOR PORT B RX & TX
A002 FF                 WRDLEN  FCB     WIP5      7 OR 8 DATA BITS ($FF = CHECK IP5)
```

```
                * LIB STMONF.TXT.1
                *
                * ADJUST THE NUMBER OF FILLER BYTES SO THE END OF SUBROUTINE
                *  "CIN" BUTTS RIGHT UP TO THE BEGINNING OF "ROMMAP"
                *
A003            FILLER  RMB    1095      (unused space) <<<<<<<<<<<<<<<<<<<<<<<<<<<
                *   END STMONF.TXT

           A44A MONBEG  EQU    *

                *      LIB STMONO.TXT.1
                ********************************************
                *    (c) 1984,1985 by David C. Wiens
                *          All rights reserved
                * FLEX AND OS-9 DISK BOOT ROUTINES FOR THE
                *  VERSION 2.04 ROM MONITOR FOR 64K ST-2900
                * Last modified June 19, 1985  3:40 pm.
                ********************************************
                *
                * OS-9 EQUATES
                *
           0008 DD_DIR  EQU    $08
           0015 DD_BT   EQU    $15
           0018 DD_BSZ  EQU    $18
           0010 DD_FMT  EQU    $10
           0011 DD_SPT  EQU    $11
           0003 DD_TKS  EQU    $03
           0009 FD_SIZ  EQU    $09
           0010 FD_SEG  EQU    $10
                *
                * I/O EQUATES
                *
           FF00 COMREG  EQU    $FF00     1793 COMMAND REGISTER
           FF00 STATRG  EQU    $FF00     1793 STATUS REGISTER
           FF01 TRKREG  EQU    $FF01     1793 TRACK REGISTER
           FF02 SECREG  EQU    $FF02     1793 SECTOR REGISTER
           FF03 DATREG  EQU    $FF03     1793 DATA REGISTER
           FF2E DRVREG  EQU    $FF2E     DISK DRIVE SELECT REGISTER
                *
                * OTHER EQUATES
                *
           0020 BRA     EQU    $20       "BRA" OPCODE
           FEDC FARTOC  EQU    $FEDC
           FEDD FARTOX  EQU    $FEDD
           FED4 FBDRIVE EQU    $FED4
           FED0 FOFFSET EQU    $FED0
           FED9 FARTLEN EQU    $FED9
           FED5 FDBLSTP EQU    $FED5
           FED8 FARTBAU EQU    $FED8
           FEDA FARTACR EQU    $FEDA
           FEDB FARTINT EQU    $FEDB
           FEE2 FBTADDR EQU    $FEE2
           FEE4 FBTSIZE EQU    $FEE4
           FEE6 FBOOTFL EQU    $FEE6
           B000 SECBUF  EQU    $B000     256 BYTE BUFFER ($B000-$B0FF)
           8000 KERBUF  EQU    $8000     4K BYTE BUFFER  ($8000-$8FFF)
           2000 BTBUFR  EQU    $2000     24K BYTE BUFFER ($2000-$7FFF)
```

```
            A44A  BEGDSK  EQU    *

            00FF          SETDP  $FF       <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
            F4FD  M       SET    *+MEND-MONEND
                  ****************************************
                  * MOVE ST-MON DOWN TO $A000 AREA
                  * IN - DP=$FF
                  * OUT - A,B,X,Y,CC UNDEFINED
                  *       U UNCHANGED
                  ****************************************
A44A 8E   AE34    MOVMON  LDX    #SWI3
A44D 108E FEE7            LDY    #SWI3+MEND-MONEND
A451 A6   A2      MV20    LDA    ,-Y
A453 A7   82              STA    ,-X
A455 8C   A44A            CMPX   #BEGDSK
A458 26   F7              BNE    MV20
                  *
                  * CORRECT ADDRESS VECTORS OF ROUTINES
                  *
A45A 8E   AE0A            LDX    #ADRTAB
A45D EC   84      MV40    LDD    0,X
A45F 83   50B3            SUBD   #MEND-MONEND
A462 ED   81              STD    ,X++
A464 8C   AE34            CMPX   #ENDADR
A467 26   F4              BNE    MV40
                  *
                  * OVERLAY ST-MON WITH NEW RESTART CODE
                  *
A469 EC   8C 05           LDD    <NEWINIT,PCR
A46C ED   9F FFFE         STD    [$FFFE]

A470 39                   RTS

                  *
                  * OVERLAY CODE
                  *
A471 20   FE      NEWINIT BRA    NEWINIT

            F526  M       SET    *+MEND-MONEND
                  ************************************************
                  * MOVE OS-9 KERNEL FROM BUFFER TO $FXXX,
                  *  THEN JUMP TO IT
                  * (THIS ROUTINE IS ONLY CALLED FROM THE ST-MON COPY AT $AXXX)
                  * IN - DP=$FF
                  * OUT - NEVER RETURNS
                  ************************************************
            A473  JMPKER  EQU    *
                  *
                  * RE-LOCATE KERNEL
                  *
A473 8E   8000            LDX    #KERBUF   MOVE KERNEL TO $FXXX
A476 108E F000            LDY    #$F000
A47A A6   84      GK40    LDA    0,X
A47C 63   80              COM    ,X+       "DISFIGURE" OLD COPY
A47E A7   A0              STA    ,Y+
A480 108C FEFF            CMPY   #$FEFF
A484 23   F4              BLS    GK40
                  *
                  * OVERLAY $FEXX DATA AREA WITH CURRENT VALUES
                  *
A486 EC   8D 0979         LDD    ARTACR,PCR move ARTACR/ARTINT to OS-9 location
```

```
A48A FD   FEDA             STD    FARTACR
A48D A6   8D 0976          LDA    ARTBAU,PCR move ARTBAU
A491 B7   FED8             STA    FARTBAU
A494 A6   8D 0957          LDA    DBLSTP,PCR move DBLSTP
A498 B7   FED5             STA    FDBLSTP
A49B A6   8D 094F          LDA    BDRIVE,PCR move BDRIVE
A49F B7   FED4             STA    FBDRIVE
A4A2 EC   8D 0946          LDD    ARTOPC,PCR move ARTOPC/ARTOPX
A4A6 FD   FEDC             STD    FARTOC
A4A9 CC   50B3             LDD    #MEND-MONEND calc offset re $FXXX and $AXXX ST-MON locn
A4AC FD   FED0             STD    FOFFSET
A4AF 86   10               LDA    #$10      reset pointer to MR1A
A4B1 97   22               STA    CPORT+2
A4B3 96   20               LDA    CPORT+0   get current MR1A contents re parity/data bits
A4B5 84   1F               ANDA   #$1F
A4B7 8A   A0               ORA    #$A0      set to 2 stop bits
A4B9 B7   FED9             STA    FARTLEN
A4BC CC   2000             LDD    #BTBUFR   move boot buffer size/location
A4BF FD   FEE2             STD    FBTADDR
A4C2 EC   8D 02D9          LDD    BTSIZE,PCR
A4C6 FD   FEE4             STD    FBTSIZE
A4C9 7F   FEE6             CLR    FBOOTFL
                  *
                  * JUMP TO KERNEL
                  *
A4CC 4F                    CLRA
A4CD 1F   8B               TFR    A,DP

A4CF 108E F000             LDY    #$F000    FIND OS9 OR OS9P1 MODULE
A4D3 30   8D 02C0          LEAX   NOS9,PCR
A4D7 8D   78               BSR    FIND
A4D9 24   0A               BCC    GK80
A4DB 108E F000             LDY    #$F000
A4DF 30   8D 02B7          LEAX   NOS9P1,PCR
A4E3 8D   6C               BSR    FIND
A4E5 EC   29       GK80    LDD    9,Y       jump to OS-9 kernel
A4E7 6E   AB               JMP    D,Y

             F59C  M       SET    *+MEND-MONEND
                  ****************************************
                  * PRE-FILL DUART RECEIVER FIFOS TO PREVENT STRANGE RESULTS
                  *  CAUSED BY "OS9P1" READING THE RECEIVE BUFFERS WHEN
                  *  SEARCHING FOR ROM MODULES
                  * IN - NONE
                  * OUT - CC ALWAYS CC
                  *       B,U UNCHANGED
                  *       A,X,Y UNDEFINED
                  ****************************************
A4E9 8E   7A12     FILLF  LDX    #31250    DELAY 1/4 SEC.
A4EC 30   1F       FR01   LEAX   -1,X      (5)
A4EE 26   FC               BNE    FR01      (3)

A4F0 31   8D 0901          LEAY   ICLSW,PCR
A4F4 6F   A8 01            CLR    <OCLSW-ICLSW,Y process port A
A4F7 6F   A8 00            CLR    <ICLSW-ICLSW,Y
A4FA 8E   FF20             LDX    #CPORT
A4FD 8D   0B               BSR    FILLR
A4FF 63   A8 01            COM    <OCLSW-ICLSW,Y process port B
A502 63   A8 00            COM    <ICLSW-ICLSW,Y
A505 30   08               LEAX   8,X
A507 8D   01               BSR    FILLR
```

```
A509 39                  RTS

                *
                * FILL DUART'S RECEIVER FIFO
                *
A50A AF   A8 08    FILLR   STX    <OPORTA-ICLSW,Y SET UP PORT ADDRESS FOR OUTCH8
A50D AF   A8 04            STX    <IPORTL-ICLSW,Y
A510 17   0892    FR05    LBSR   INCHEK    FLUSH RECEIVER FIFO
A513 27   05               BEQ    FR07
A515 17   082F             LBSR   INCHN
A518 20   F6               BRA    FR05
A51A 86   40       FR07    LDA    #$40      RESET OVERRUN ERROR
A51C A7   02               STA    2,X
A51E A6   84               LDA    0,X       MAKE SURE WE POINT TO MR2
A520 A6   84               LDA    0,X       DISABLE CTS CONTROL
A522 84   EF               ANDA   #$EF
A524 A7   84               STA    0,X
A526 A6   01       FR10    LDA    1,X       WAIT UNTIL TXEMT = 1
A528 85   08               BITA   #$08
A52A 27   FA               BEQ    FR10
A52C A6   84               LDA    0,X       SET TO LOCAL LOOPBACK MODE
A52E 84   3F               ANDA   #$3F
A530 8A   80               ORA    #$80
A532 A7   84               STA    0,X
A534 86   FF               LDA    #$FF      SEND 4 CHARACS TO RECEIVER
A536 17   079E             LBSR   OUTCH
A539 17   079B             LBSR   OUTCH
A53C 17   0798             LBSR   OUTCH
A53F 17   0795             LBSR   OUTCH
A542 A6   01       FR60    LDA    1,X       WAIT UNTIL TXEMT = 1
A544 85   08               BITA   #$08
A546 27   FA               BEQ    FR60
A548 A6   84               LDA    0,X       DISABLE LOCAL LOOPBACK, RE-ENABLE CTS CONTROL
A54A 84   3F               ANDA   #$3F
A54C 8A   10               ORA    #$10
A54E A7   84               STA    0,X
A550 39                    RTS

             F604  M       SET    *+MEND-MONEND
                ****************************************
                * FIND AN OS-9 MODULE
                * IN - X address of module name string (high order
                *         bit set in last character)
                *      Y address where search is to begin (search
                *         ends at $FEF0)
                * OUT - Y address of module
                *       A,B undefined
                *       X,U unchanged
                *       CC CS=not found
                ****************************************
A551 108C FEF0     FIND    CMPY   #$FEF0    STILL WITHIN VALID RANGE?
A555 24   3C               BHS    FI70      .N
A557 EC   A4               LDD    0,Y       HEADER SYNC BYTES?
A559 81   87               CMPA   #$87
A55B 26   32               BNE    FI50      .N
A55D C1   CD               CMPB   #$CD
A55F 26   2E               BNE    FI50      .N

A561 C6   09               LDB    #9        VALID HEADER PARITY?
A563 34   20               PSHS   Y
A565 4F                    CLRA
```

```
A566 A8   A0       FI2O    EORA   ,Y+
A568 5A            FI2O    DECB
A569 26   FB                BNE    FI2O
A56B 35   20                PULS   Y
A56D 81   FF                CMPA   #$FF
A56F 26   1E                BNE    FI5O      .N

A571 34   30                PSHS   X,Y
A573 EC   24                LDD    4,Y       COMPARE NAME STRINGS
A575 31   AB                LEAY   D,Y
A577 A6   8O       FI3O    LDA    ,X+
A579 84   5F                ANDA   #$5F
A57B 34   O2                PSHS   A
A57D A6   A0                LDA    ,Y+
A57F 84   5F                ANDA   #$5F
A581 A1   E0                CMPA   ,S+
A583 26   O8                BNE    FI4O
A585 6D   1F                TST    -1,X
A587 2A   EE                BPL    FI3O
A589 35   30                PULS   X,Y
A58B 2O   O9                BRA    FI8O
A58D 35   30       FI4O    PULS   X,Y

A58F 31   21       FI5O    LEAY   1,Y       NOT FOUND YET, TRY NEXT LOCATION
A591 2O   BE                BRA    FIND

A593 53            FI7O    COMB              ERROR
A594 2O   O1                BRA    FI9O

A596 5F            FI8O    CLRB              OK
A597 39            FI9O    RTS


               F64B  M       SET    *+MEND-MONEND
                     ****************************************
                     * LOAD OS-9 KERNEL AND BOOT FILES FROM
                     *  CONFIGURED DISK AND JUMP TO OS-9.
                     *  (EACH MUST CONSIST OF ONLY ONE EXTENT,
                     *   AND THE KERNEL FILE MUST APPEAR IN THE
                     *   1ST SECTOR OF THE ROOT DIRECTORY)
                     *
                     * IN - U POINTS TO DSKVAR
                     *      DBLSTP/BDRIVE
                     *      DP=$FF
                     * OUT - NO RETURN IF OK
                     ****************************************
A598 32   7C       LOADO9  LEAS   -4,S
A59A 6F   C8 FF           CLR    <COCO-DSKVAR,U
A59D 17   O7C2            LBSR   INCHE     COCO/STD?
A5AO 81   53              CMPA   #'S
A5A2 27   O9              BEQ    LB1O      .STD
A5A4 81   43              CMPA   #'C
A5A6 1O26 OOAD            LBNE   LB9O
A5AA 6C   C8 FF           INC    <COCO-DSKVAR,U .COCO
A5AD 17   O3AC     LB1O   LBSR   MOTOR
A5BO 17   O344            LBSR   RECALB
                     *
                     * READ LSN O AND GET POINTERS AND FORMAT CODES
                     *
A5B3 CC   OOOA            LDD    #1O
A5B6 ED   C8 F7           STD    <BT_SPT-DSKVAR,U
```

```
A5B9 6F   C8 F6           CLR    <BT_FMT-DSKVAR,U
A5BC 8E   0000            LDX    #0000     READ LSN 0
A5BF 108E B000            LDY    #SECBUF
A5C3 17   010E            LBSR   READLS
A5C6 1025 008D            LBCS   LB90
A5CA A6   A8 10           LDA    DD_FMT,Y
A5CD A7   C8 F6           STA    <BT_FMT-DSKVAR,U
A5D0 EC   A8 11           LDD    DD_SPT,Y
A5D3 ED   C8 F7           STD    <BT_SPT-DSKVAR,U

A5D6 AE   A8 16           LDX    DD_BT+1,Y SAVE OS9 BOOT FILE LOCN/SIZE
A5D9 AF   E4              STX    0,S
A5DB 26   06              BNE    LB30
A5DD 30   8D 01A0         LEAX   MSG4,PCR
A5E1 20   71              BRA    LB80
A5E3 AE   A8 18    LB30   LDX    DD_BSZ,Y
A5E6 AF   62              STX    2,S
A5E8 AF   C8 F4           STX    <BTSIZE-DSKVAR,U
                  *
                  * FIND AND LOAD KERNEL FILE
                  *
A5EB 8E   8000            LDX    #KERBUF   ZERO OUT KERNEL BUFFER
A5EE 6F   80       LB40   CLR    ,X+
A5F0 8C   9000            CMPX   #KERBUF+$1000
A5F3 25   F9              BLO    LB40

A5F5 AE   29              LDX    DD_DIR+1,Y DETERMINE 1ST DATA SECTOR OF ROOT DIRECTORY
A5F7 8D   61              BSR    FINDDT
A5F9 25   5C              BCS    LB90
A5FB 108E B000            LDY    #SECBUF   READ 1ST DATA SECTOR OF ROOT DIRECTORY
A5FF 17   00D2            LBSR   READLS
A602 25   53              BCS    LB90
A604 30   8D 0186         LEAX   KERNAM,PCR FIND KERNEL FILE NAME IN DIRECTORY
A608 8D   61              BSR    FINDFL
A60A 24   06              BCC    LB60
A60C 30   8D 015B         LEAX   MSG3,PCR
A610 20   42              BRA    LB80
A612 8D   46       LB60   BSR    FINDDT    DETERMINE 1ST DATA SECTOR AND SIZE
A614 25   41              BCS    LB90      OF KERNEL FILE
A616 108E 8000            LDY    #KERBUF
A61A 17   0080            LBSR   READCF    LOAD KERNEL FILE INTO BUFFER
A61D 25   38              BCS    LB90
A61F 30   8D 0174         LEAX   NOS9,PCR  VERIFY THAT MODULE OS9 OR OS9P1 IS IN KERNEL
A623 108E 8000            LDY    #KERBUF
A627 17   FF27            LBSR   FIND
A62A 24   0D              BCC    LB70
A62C 30   8D 016A         LEAX   NOS9P1,PCR
A630 108E 8000            LDY    #KERBUF
A634 17   FF1A            LBSR   FIND
A637 25   1E              BCS    LB90
                  *
                  * LOAD OS-9 BOOT FILE
                  *
A639 AE   E4       LB70   LDX    0,S
A63B EE   62              LDU    2,S
A63D 108E 2000            LDY    #BTBUFR
A641 8D   5A              BSR    READCF
A643 25   12              BCS    LB90
```

```
                        *
                        * MOVE ST-MON TO $AXXX, RELOCATE KERNEL TO $FXXX,
                        *  THEN JUMP TO OS-9
                        *
A645 17   0683                  LBSR   PCRLF
A648 17   FE9E                  LBSR   FILLF     FILL DUART'S FIFOS

A64B 17   FDFC                  LBSR   MOVMON    MOVE ST-MON TO $AXXX

A64E 16   AF4D                  LBRA   LB75-MEND+MONEND jump to relocated ST-MON
A651 16   FE1F     LB75         LBRA   JMPKER    relocate kernel and jump to it
                        *
                        * ERROR RETURN
                        *
A654 17   0667     LB80         LBSR   PSTRNG
A657 32   64       LB90         LEAS   4,S
A659 39                         RTS

             F70D  M            SET    *+MEND-MONEND
                        ****************************************
                        * FIND BEGINNING OF DATA OF FILE
                        * IN - X LSN OF FD SECTOR
                        *      DP=$FF
                        *      COCO,BT_SPT,BT_FMT
                        * OUT - X STARTING LSN OF DATA
                        *       U FILE SIZE IN BYTES
                        *       A,B,Y UNCHANGED
                        * ERROR - CC CS
                        ****************************************
A65A 34   20       FINDDT       PSHS   Y
A65C 108E B000                  LDY    #SECBUF
A660 8D   72                    BSR    READLS
A662 25   05                    BCS    FD90
A664 EE   2B                    LDU    FD_SIZ+2,Y
A666 AE   A8 11                 LDX    FD_SEG+1,Y
A669 35   A0       FD90         PULS   Y,PC

             F71E  M            SET    *+MEND-MONEND
                        ****************************************
                        * FIND FILE NAME IN DIRECTORY (MUST BE IN FIRST
                        *  SECTOR OF FIRST EXTENT OF THE DIRECTORY)
                        * IN - X ADDRESS OF NAME STRING (MUST BE 28 CHARAC
                        *          OR LESS, WITH HIGH ORDER BIT SET ON LAST)
                        *      Y ADDRESS OF DIRECTORY SECTOR BUFFER
                        * OUT - X LSN OF FD SECTOR OF DESIRED FILE
                        *       A,B,Y,U UNCHANGED
                        * ERROR - CC CS IF NOT FOUND, ETC.
                        ****************************************
A66B 34   22       FINDFL       PSHS   A,Y
A66D 86   08                    LDA    #8        INIT COUNTER (8 ENTRIES / SECTOR)
A66F A7   E2                    STA    ,-S

A671 34   30       FF20         PSHS   X,Y
A673 A6   80       FF30         LDA    ,X+
A675 84   5F                    ANDA   #$5F      CONVERT TO UPPER CASE, STRIP BIT 7
A677 34   02                    PSHS   A
A679 A6   A0                    LDA    ,Y+
A67B 84   5F                    ANDA   #$5F
A67D A1   E0                    CMPA   ,S+       CHARACTER MATCHES?
A67F 26   0B                    BNE    FF40      .N
A681 6D   1F                    TST    -1,X      .Y, END OF STRING?
```

```
A683 2A   EE              BPL    FF3O      .      N, TRY NEXT CHARAC
A685 35   3O              PULS   X,Y       .      Y, STRING MATCHES
A687 AE   A8 1E           LDX    3O,Y      GET LSN OF FILE FD
A68A 2O   OC              BRA    FF8O
A68C 35   3O       FF4O   PULS   X,Y
A68E 31   A8 2O           LEAY   32,Y      point to next entry
A691 6A   E4              DEC    O,S       all 8 entries checked?
A693 26   DC              BNE    FF2O      .n
A695 43                   COMA             ERROR .y, error
A696 2O   O1              BRA    FF9O

A698 4F           FF8O    CLRA             OK
A699 32   61      FF9O    LEAS   1,S
A69B 35   A2              PULS   A,Y,PC

             F75O  M       SET    *+MEND-MONEND
                   ****************************************
                   * READ CONTIGUOUS FILE FROM DISK TO BUFFER
                   * IN - X STARTING LSN OF FILE DATA
                   *      U FILE SIZE IN BYTES
                   *      Y FILE BUFFER ADDRESS
                   *      DP=$FF
                   *      COCO,BT_SPT,BT_FMT
                   * OUT - A,B,X,Y,U UNCHANGED
                   * ERROR - CC CS
                   ****************************************
A69D 34   76       READCF PSHS   A,B,X,Y,U
A69F 34   3O              PSHS   X,Y
A6A1 1O8E BOOO     RC1O   LDY    #SECBUF   READ SECTOR INTO SECTOR BUFFER
A6A5 8D   2D              BSR    READLS
A6A7 25   27              BCS    RC9O

A6A9 5F                   CLRB             MOVE SECTOR BUFFER TO FILE BUFFER
A6AA AE   62              LDX    2,S
A6AC A6   AO       RC2O   LDA    ,Y+
A6AE A7   8O              STA    ,X+
A6BO 33   5F              LEAU   -1,U
A6B2 1183 OOOO            CMPU   #OOOO     FILE DONE?
A6B6 27   OD              BEQ    RC6O      .Y
A6B8 5A                   DECB             .N, SECTOR DONE?
A6B9 26   F1              BNE    RC2O      .     N
A6BB AF   62              STX    2,S

A6BD AE   E4              LDX    O,S       INCR LSN
A6BF 3O   O1              LEAX   1,X
A6C1 AF   E4              STX    O,S
A6C3 2O   DC              BRA    RC1O

A6C5 4F           RC6O    CLRA             ERASE SECTOR BUFFER
A6C6 5F                   CLRB
A6C7 8E   BOOO            LDX    #SECBUF
A6CA A7   8O       RC65   STA    ,X+
A6CC 5A                   DECB
A6CD 26   FB              BNE    RC65

A6CF 5F                   CLRB
A6DO 32   64       RC9O   LEAS   4,S
A6D2 35   F6              PULS   A,B,X,Y,U,PC
```

```
                F787  M       SET    *+MEND-MONEND
                      ****************************************
                      * READ LOGICAL SECTOR FROM DISK
                      * IN - X LSN
                      *      Y SECTOR BUFFER ADDRESS
                      *      DP=$FF
                      *      COCO,BT_SPT,BT_FMT,DBLSTP
                      * OUT - A,B,X,Y,U UNCHANGED
                      * ERROR - CC CS
                      ****************************************
A6D4 34   16          READLS  PSHS   A,B,X
A6D6 8D   07                  BSR    LTOP      CONVERT LSN TO TRACK/SECTOR/SIDE
A6D8 1F   21                  TFR    Y,X
A6DA 17   01AB                LBSR   GETSEC    READ SECTOR FROM DISK
A6DD 35   96                  PULS   A,B,X,PC

                F792  M       SET    *+MEND-MONEND
                      ****************************************
                      * CONVERT LSN TO TRACK/SECTOR/SIDE/DENSITY
                      * IN - X LSN
                      *      COCO,BT_SPT,BT_FMT
                      * OUT - A PHYSICAL TRACK #
                      *       B PHYSICAL SECTOR #
                      *       SIDE,DENS
                      *       X,Y,U UNCHANGED
                      *       CC UNDEFINED
                      ****************************************
A6DF 34   50          LTOP    PSHS   X,U
A6E1 6F   E2                  CLR    ,-S       SET UP CYLINDER # COUNTER
A6E3 33   8D 00C4             LEAU   DSKVAR,PCR
A6E7 6F   C8 02               CLR    <SIDE-DSKVAR,U
A6EA 6F   C8 03               CLR    <DENS-DSKVAR,U
                      *
                      * CALC SECTORS PER CYLINDER
                      *
A6ED 6D   C8 FF               TST    <COCO-DSKVAR,U
A6F0 26   04                  BNE    LP10
A6F2 C6   0A                  LDB    #10
A6F4 20   03                  BRA    LP15
A6F6 E6   C8 F8    LP10       LDB    <BT_SPT+1-DSKVAR,U
A6F9 4F            LP15       CLRA
A6FA ED   C8 F9               STD    <BT_TOS-DSKVAR,U
A6FD A6   C8 F6               LDA    <BT_FMT-DSKVAR,U SS/DS ?
A700 85   01                  BITA   #$01
A702 27   03                  BEQ    LP20      SS
A704 EB   C8 F8               ADDB   <BT_SPT+1-DSKVAR,U DS
A707 4F            LP20       CLRA
A708 ED   C8 FD               STD    <BT_COS-DSKVAR,U

A70B E6   C8 F8               LDB    <BT_SPT+1-DSKVAR,U
A70E A6   C8 F6               LDA    <BT_FMT-DSKVAR,U
A711 85   01                  BITA   #$01      SS/DS ?
A713 27   01                  BEQ    LP30      SS
A715 58                       LSLB             DS
A716 4F            LP30       CLRA
A717 ED   C8 FB               STD    <BT_SPC-DSKVAR,U
                      *
                      * CALC CYLINDER #
                      *
A71A 1F   10                  TFR    X,D
A71C 10A3 C8 FD               CMPD   <BT_COS-DSKVAR,U
```

```
A720 25   10              BLO    LP50
A722 A3   C8 FD           SUBD   <BT_COS-DSKVAR,U
A725 6C   E4       LP40   INC    0,S
A727 10A3 C8 FB           CMPD   <BT_SPC-DSKVAR,U
A72B 25   05              BLO    LP50
A72D A3   C8 FB           SUBD   <BT_SPC-DSKVAR,U
A730 20   F3              BRA    LP40
                  *
                  * DETERMINE SIDE AND SECTOR NUMBER
                  *
A732 6D   E4       LP50   TST    0,S
A734 26   0B              BNE    LP56
A736 10A3 C8 F9           CMPD   <BT_TOS-DSKVAR,U
A73A 25   11              BLO    LP60
A73C A3   C8 F9           SUBD   <BT_TOS-DSKVAR,U
A73F 20   09              BRA    LP58
A741 10A3 C8 F7   LP56    CMPD   <BT_SPT-DSKVAR,U
A745 25   06              BLO    LP60
A747 A3   C8 F7           SUBD   <BT_SPT-DSKVAR,U
A74A 6C   C8 02   LP58    INC    <SIDE-DSKVAR,U
                  *
                  * DETERMINE DENSITY, AND ADJUST SECTOR #
                  *
A74D A6   C8 F6   LP60    LDA    <BT_FMT-DSKVAR,U DD?
A750 85   02              BITA   #$02
A752 27   0C              BEQ    LP68      .N
A754 6D   E4              TST    0,S       .Y, TRK 0?
A756 26   05              BNE    LP66      .     N
A758 6D   C8 02           TST    <SIDE-DSKVAR,U .     Y, SIDE 0?
A75B 27   03              BEQ    LP68
A75D 6C   C8 03   LP66    INC    <DENS-DSKVAR,U
A760 6D   C8 FF   LP68    TST    <COCO-DSKVAR,U
A763 27   04              BEQ    LP80
A765 5C                   INCB             ADJ SECTOR # FOR COCO
A766 6C   C8 03           INC    <DENS-DSKVAR,U

A769 35   D2       LP80   PULS   A,X,U,PC  GET TRACK # AND RETURN

             F81E  M        SET    *+MEND-MONEND
                   ****************************************
                   *
                   * VARIABLES AND STRINGS
                   *
                   ****************************************
                   *
                   * OS-9
                   *
A76B 4B 45 52 4E   MSG3   FCC    'KERNEL FILE NOT FOUND',EOT
A76F 45 4C 20 46
A773 49 4C 45 20
A777 4E 4F 54 20
A77B 46 4F 55 4E
A77F 44 04
A781 4E 4F 20 42   MSG4   FCC    'NO BOOT FILE',EOT
A785 4F 4F 54 20
A789 46 49 4C 45
A78D 04
A78E 4F 53 39 4B   KERNAM  FCC    'OS9Kerne'
A792 65 72 6E 65
A796 EC                   FCB    'l+$80
A797 4F 53         NOS9   FCC    'OS'
```

```
A799 B9                    FCB    '9+$80
A79A 4F 53 39 70   NOS9P1  FCC    'OS9p'
A79E B1                    FCB    '1+$80
A79F 0000          BTSIZE  FDB    0         SIZE OF BOOT FILE
A7A1 00            BT_FMT  FCB    0         DISK FORMAT CODE
A7A2 0000          BT_SPT  FDB    0         SECTORS PER TRACK
A7A4 0000          BT_TOS  FDB    0         SECTORS PER TRACK (TRK 0)
A7A6 0000          BT_SPC  FDB    0         SECTORS PER CYLINDER
A7A8 0000          BT_COS  FDB    0         SECTORS PER CYLINDER (TRK 0)
A7AA 00            COCO    FCB    0         0=STD, 1=COCO

             F85E  M       SET    *+MEND-MONEND
                   *
                   * FLEX AND COMMON
                   *
             A7AB  DSKVAR  EQU    *
             A7AB  BEGFLX  SET    DSKVAR
A7AB 0A            NUMSEC  FCB    10        SECTORS PER ONE SIDE OF ONE TRACK (TRKS 1-N)
A7AC 00            DENSFL  FCB    0         $00=SINGLE-DENSITY, NON-ZERO=DOUBLE (TRKS 1-N)
A7AD 00            SIDE    FCB    0         0=SIDE 0, 1=SIDE 1 (OF CURRENT SECTOR)
A7AE 00            DENS    FCB    0         0=SD, NON-ZERO=DD (OF CURRENT SECTOR)

A7AF 4E 54 20 4C   MSG1    FCC    'NT LNKD',EOT
A7B3 4E 4B 44 04
A7B7 42 54 20 45   MSG2    FCC    'BT ERR',EOT
A7BB 52 52 04


             F871  M       SET    *+MEND-MONEND
                   *********************************************************
                   * BOOT UP FLEX OR OS-9 OPERATING SYSTEM FROM DISK
                   *  (RETURNS TO MONITOR IF UNSUCCESSFUL;
                   *   OTHERWISE CONTROL IS PASSED TO THE
                   *   LOADED SYSTEM)
                   * IN - DBLSTP/BDRIVE
                   *********************************************************
A7BE 17   05A1     DISKBT  LBSR   INCHE
A7C1 C6   FF               LDB    #$FF
A7C3 1F   9B               TFR    B,DP
A7C5 33   8C E3            LEAU   DSKVAR,PCR
A7C8 81   46               CMPA   #'F       FLEX?
A7CA 27   09               BEQ    DB50
A7CC 81   4F               CMPA   #'O       OS9?
A7CE 26   0C       DB10    BNE    DB80
                   *
                   * BOOT OS-9
                   *
A7D0 17   FDC5             LBSR   LOAD09    load OS9Kernel and OS9Boot and relocate ST-MON
A7D3 20   07               BRA    DB80      . error
                   *
                   * BOOT FLEX
                   *
A7D5 86   20       DB50    LDA    #BRA      DISABLE "D 0" COMMAND
A7D7 A7   8C F4            STA    <DB10,PCR
A7DA 8D   09               BSR    LOADFX    BOOT FLEX
                   *
                   * ERROR
                   *
A7DC 4F            DB80    CLRA
A7DD 1F   8B               TFR    A,DP
A7DF 30   8C D5            LEAX   MSG2,PCR  ERROR
```

```
A7E2 16   04D9             LBRA   PSTRNG

             F898  M       SET    *+MEND-MONEND
                   ****************************************
                   * LOAD THE FLEX OPERATING SYSTEM, OR WHATEVER
                   *  PROGRAM WAS LINKED TO, AND JUMP TO IT.
                   *  THE LAST TRANSFER ADDRESS FOUND IS USED.
                   * IN - DP=$FF
                   *      U POINTS TO DSKVAR
                   *      DBLSTP/BDRIVE
                   * OUT - NO RETURN IF OK
                   ****************************************
A7E5 17   0174     LOADFX  LBSR   MOTOR
A7E8 17   010C             LBSR   RECALB
A7EB 86   0A               LDA    #10
A7ED A7   C4               STA    NUMSEC-DSKVAR,U
A7EF 30   8D 03D7          LEAX   SIGNON,PCR set jump address to return to ST-MON if
A7F3 AF   8D 05FC          STX    JMPADDR,PCR . no transfer address found
                   *
                   * READ TRACK 0, SECTOR 1 AND EXTRACT PARAMETERS
                   *
A7F7 CC   0001             LDD    #$0001
A7FA 8E   B000             LDX    #SECBUF
A7FD 8D   76               BSR    READPS
A7FF 25   4D               BCS    LF90
A801 A6   02               LDA    2,X       NUMSEC
A803 A7   C4               STA    NUMSEC-DSKVAR,U
A805 A6   04               LDA    4,X       DENSFL
A807 A7   41               STA    DENSFL-DSKVAR,U
A809 EC   05               LDD    5,X       LNKADR
A80B 27   3A               BEQ    LF80
                   *
                   * READ DATA SECTOR AND PROCESS IT
                   *
A80D 8D   66               BSR    READPS
A80F 25   3D               BCS    LF90
A811 30   04               LEAX   4,X
A813 8D   47       LF10    BSR    GETB      GET BYTE
A815 25   37               BCS    LF90
A817 C1   02               CMPB   #$02      START OF OBJ CODE DATA RECORD?
A819 27   0E               BEQ    BINR      .Y
A81B C1   16               CMPB   #$16      START OF TRANSFER ADDRESS?
A81D 26   F4               BNE    LF10      .N
A81F 8D   2E               BSR    GETA      .Y, GET ADDRESS
A821 25   2B               BCS    LF90
A823 ED   8D 05CC          STD    JMPADDR,PCR SAVE TSF ADDRESS
A827 20   EA               BRA    LF10
                   *
                   * PROCESS A BINARY (OBJECT CODE) RECORD
                   *
A829 8D   24       BINR    BSR    GETA      STARTING MEMORY LOCATION
A82B 25   21               BCS    LF90
A82D 1F   02               TFR    D,Y
A82F 8D   2B               BSR    GETB      BYTE COUNT
A831 25   1B               BCS    LF90
A833 5D                    TSTB
A834 27   DD               BEQ    LF10      OMIT NULL RECORD
A836 1F   98               TFR    B,A
A838 34   02       LF40    PSHS   A
A83A 8D   20               BSR    GETB
A83C 35   02               PULS   A
```

```
A83E 25   0E              BCS    LF90
A840 E7   A0              STB    ,Y+       WRITE TO MEMORY
A842 4A                   DECA
A843 26   F3              BNE    LF40
A845 20   CC              BRA    LF10

A847 30   8D FF64  LF80    LEAX   MSG1,PCR  "NOT LINKED"
A84B 17   0470            LBSR   PSTRNG
A84E 39           LF90    RTS

             F902 M       SET    *+MEND-MONEND
                  ****************************************
                  * GET NEXT TWO BYTES FROM SECTOR BUFFER
                  *  If last byte of file has already been read,
                  *  control is passed to the transfer address.
                  * IN - X POINTS TO NEXT BYTE IN BUFFR
                  *      U POINTS TO DSKVAR
                  *      DP=$FF
                  * OUT - X INCREMENTED BY 2
                  *       D DATA BYTES
                  *       Y,U UNCHANGED
                  *       [No return if end-of-file]
                  * ERROR - CC CS
                  ****************************************
A84F 8D   0B      GETA    BSR    GETB      MSB
A851 25   06              BCS    GA90
A853 34   04              PSHS   B
A855 8D   05              BSR    GETB      LSB
A857 35   02              PULS   A
A859 39           GA90    RTS

             F90D M       SET    *+MEND-MONEND
                  ****************************************
                  * GETB - GET NEXT BYTE FROM SECTOR BUFFER
                  *  If last byte of file has already been read,
                  *  control is passed to the transfer address.
                  * IN - X POINTS TO NEXT BYTE IN BUFFER
                  *      U POINTS TO DSKVAR
                  *      DP=$FF
                  * OUT - X INCREMENTED BY 1
                  *       B DATA BYTE
                  *       A DESTROYED
                  *       Y,U UNCHANGED
                  *       [No return if end-of-file]
                  * ERROR - CC CS
                  ****************************************
A85A 30   04      GB10    LEAX   4,X

A85C 8C   B100    GETB    CMPX   #SECBUF+256 END OF SECTOR?
A85F 27   04              BEQ    GB20      .Y
A861 E6   80              LDB    ,X+       .N, GET BYTE
A863 4F                   CLRA
A864 39                   RTS

A865 8E   B000    GB20    LDX    #SECBUF
A868 EC   84              LDD    0,X       GET POINTER TO NEXT SECTOR
A86A 27   05              BEQ    GB90
A86C 8D   07              BSR    READPS
A86E 24   EA              BCC    GB10
A870 39                   RTS
```

```
A871 6E   9D 057E  GB90    JMP    [JMPADDR,PCR] JUMP TO TRANSFERR ADDR

            F928  M       SET    *+MEND-MONEND
                  ****************************************
                  * READ PHYSICAL SECTOR FROM DISK
                  *  (ASSUMES THAT THE FILE TO BE LOADED IS NOT
                  *   ON TRACK 0, AND THAT TRACK 0 IS SINGLE
                  *   DENSITY, WITH OTHER TRACKS EITHER SD OR DD,
                  *   SS OR DS)
                  * IN - A/B TRACK/SECTOR (PHYSICAL)
                  *      X ADDRESS OF SECTOR BUFFER
                  *      U POINTS TO DSKVAR
                  *      DP=$FF
                  *      DENSFL,NUMSEC,DBLSTP
                  * OUT - A,B,X,Y,U UNCHANGED
                  *       SIDE,DENS
                  * ERROR - CC CS
                  ****************************************
            A875  READPS  EQU    *
                  *
                  * CALCULATE SIDE AND DENSITY
                  *
A875 6F   42              CLR    SIDE-DSKVAR,U
A877 6F   43              CLR    DENS-DSKVAR,U
A879 4D                   TSTA             TRACK 0?
A87A 27   06              BEQ    RP20      .Y, SD
A87C 6D   41              TST    DENSFL-DSKVAR,U .N, DD DISK?
A87E 27   02              BEQ    RP20      .        N
A880 6C   43              INC    DENS-DSKVAR,U .     Y, SET TO DD
A882 E1   C4        RP20  CMPB   NUMSEC-DSKVAR,U ON SIDE 0?
A884 23   02              BLS    RP60      .Y
A886 6C   42              INC    SIDE-DSKVAR,U .Y, SET TO SIDE 1
                  *
                  * READ SECTOR
                  *
            A888  RP60    EQU    *         (fall thru to GETSEC)


            F93B  M       SET    *+MEND-MONEND
                  ****************************************
                  * GET SPECIFIED SECTOR FROM DISK - SEEK/READ/RETRY
                  * IN - A TRACK # DESIRED
                  *      B SECTOR # DESIRED
                  *      X ADDRESS OF SECTOR BUFFER
                  *      DP=$FF
                  *      SIDE,DENS,DBLSTP
                  * OUT - A,B,X,Y,U UNCHANGED
                  * ERROR - CC CS
                  ****************************************
A888 34   16        GETSEC  PSHS   A,B,X
A88A 32   7D              LEAS   -3,S

A88C 86   03              LDA    #3        OVERALL RETRY COUNT = 3
A88E A7   E4              STA    0,S

A890 86   03        GS10  LDA    #3        SEEK RETRY COUNT = 3   - - - - -
A892 A7   61              STA    1,S       .                              |

A894 EC   63        GS20  LDD    3,S       SEEK      - - - - - - -
A896 8D   28              BSR    SEEK      .                       |
A898 27   08              BEQ    GS40      .OK                     |
```

```
A89A 6A   61              DEC    1,S       .ERROR, RETRY?        |
A89C 27   1A              BEQ    GS7O      .    N, HARD ERROR    |
A89E 8D   57              BSR    RECALB    .    Y                |
A8A0 2O   F2              BRA    GS2O      .         - - - - - - -

A8A2 86   O5       GS4O   LDA    #5        READ RETRY COUNT = 5
A8A4 A7   62              STA    2,S

A8A6 AE   65       GS5O   LDX    5,S       READ SECTOR     - - - -
A8A8 8D   72              BSR    READS     .                       |
A8AA 27   OF              BEQ    GS8O      .OK                     |
A8AC 6A   62              DEC    2,S       .ERROR, RETRY?        |
A8AE 26   F6              BNE    GS5O      .       Y       - - - -
A8B0 6A   E4              DEC    O,S       .       N, RETRY ENTIRE PROCESS?
A8B2 27   O4              BEQ    GS7O      .               N, HARD ERROR
A8B4 8D   41              BSR    RECALB    .               Y               |
A8B6 2O   D8              BRA    GS1O      .               - - - - - - - - -

A8B8 53           GS7O    COMB             HARD ERROR
A8B9 2O   O1              BRA    GS9O
A8BB 5F           GS8O    CLRB
A8BC 32   63       GS9O   LEAS   3,S
A8BE 35   96              PULS   A,B,X,PC

           F973  M        SET    *+MEND-MONEND
                 ****************************************
                 * SEEK TO DESIRED TRACK AND SELECT SIDE AND
                 *  DENSITY AND SECTOR.
                 *  (When switching sides, some drives require a delay of
                 *   1OO usec. before issuing a read/write command)
                 * IN - A DESIRED LOGICAL TRACK
                 *      B DESIRED SECTOR
                 *      DP=$FF
                 *      SIDE,DENS,DBLSTP
                 * OUT - CC NE=ERROR, EQ=OK
                 *       B STATUS WORD, IF ERROR
                 *       A PHYSICAL TRACK
                 *       X,Y,U UNCHANGED
                 ****************************************
           A8CO  SEEK     EQU    *
                 *
                 * ADJUST FOR DOUBLE STEPPING
                 *
A8CO 6D   8D O52B        TST    DBLSTP,PCR
A8C4 27   O1             BEQ    SK1O
A8C6 48                  LSLA
A8C7 34   42       SK1O  PSHS   A,U
A8C9 33   8D FEDE        LEAU   DSKVAR,PCR
                 *
                 * SELECT SECTOR
                 *
A8CD D7   O2             STB    SECREG
A8CF 8D   43             BSR    DELAY
                 *
                 * SELECT SIDE AND DENSITY
                 *
A8D1 4F                  CLRA
A8D2 6D   42             TST    SIDE-DSKVAR,U
A8D4 26   O2             BNE    SK2O
A8D6 8A   1O             ORA    #$1O
A8D8 6D   43       SK2O  TST    DENS-DSKVAR,U
```

```
A8DA 26   02              BNE    SK30
A8DC 8A   20              ORA    #$20
A8DE C6   30       SK30   LDB    #$30
A8E0 17   0090            LBSR   SETOPR
A8E3 8D   2F              BSR    DELAY
                  *
                  * SEEK TO TRACK
                  *
A8E5 A6   E4              LDA    0,S
A8E7 91   01              CMPA   TRKREG    ALREADY ON TRACK?
A8E9 27   0A              BEQ    SK90      .Y
A8EB 97   03              STA    DATREG
A8ED 8D   25              BSR    DELAY
A8EF 86   1B              LDA    #$1B      SEEK/LOAD HEAD/NO VERIFY/SLOW STEP
A8F1 8D   16              BSR    EXCMDW
A8F3 C5   10              BITB   #$10      SEEK ERROR?

A8F5 35   C2       SK90   PULS   A,U,PC

            F9AA  M       SET    *+MEND-MONEND
                  ****************************************
                  * RECALIBRATE THE DRIVE'S HEAD POSITION
                  *  (IE. RESTORE TO TRACK 0)
                  * IN - NONE, EXCEPT DRIVE MUST BE SELECTED
                  *      DP=$FF
                  * OUT - A UNDEFINED
                  *       B 1793 STATUS WORD
                  *       CC ALWAYS EQ
                  *       X,Y,U UNCHANGED
                  ****************************************
A8F7 8D   0E       RECALB BSR    RESTOR    RESTORE/LOAD HEAD/NO VERIFY/SLOW STEP
A8F9 8D   59              BSR    DLY100
A8FB 86   4B              LDA    #$4B      STEP IN/LOAD HEAD/NO VERIFY/SLOW STEP
A8FD 8D   0A              BSR    EXCMDW
A8FF 8D   08              BSR    EXCMDW
A901 8D   06              BSR    EXCMDW
A903 8D   04              BSR    EXCMDW
A905 8D   4D              BSR    DLY100
A907 86   0B       RESTOR LDA    #$0B      RESTORE/LOAD HEAD/NO VERIFY/SLOW STEP

            F9BC  M       SET    *+MEND-MONEND
                  ****************************************
                  * EXCMDW - INITIATE DISK COMMAND, THEN WAIT
                  *          UNTIL FINISHED
                  *  IN - A COMMAND
                  *       DP=$FF
                  *  OUT - CC ALWAYS EQ
                  *        B STATUS WORD
                  *        A,X,Y,U UNCHANGED
                  *
                  * BUSYW - WAIT UNTIL COMMAND FINISHED
                  *  IN - NONE
                  *       DP=$FF
                  *  OUT - CC ALWAYS EQ
                  *        B STATUS WORD
                  *        A,X,Y,U UNCHANGED
                  *
                  * EXCMD - INITIATE DISK COMMAND
                  *  IN - A COMMAND
                  *       DP=$FF
                  *  OUT - CC UNDEFINED
```

```
                    *        ALL OTHERS UNCHANGED
                    ****************************************
 A909 8D   07       EXCMDW  BSR    EXCMD

 A90B D6   00       BUSYW   LDB    STATRG
 A90D C5   01               BITB   #$01
 A90F 26   FA               BNE    BUSYW
 A911 39                    RTS

 A912 97   00       EXCMD   STA    COMREG

              F9C7  M       SET    *+MEND-MONEND
                    ****************************************
                    * DELAY APPROX 56 MICROSECONDS
                    * IN - NONE
                    * OUT - ALL REGISTERS UNCHANGED
                    ****************************************
>A914 17   0000     DELAY   LBSR   DY10
>A917 17   0000     DY10    LBSR   DY20
 A91A 12            DY20    NOP
 A91B 39                    RTS

              F9CF  M       SET    *+MEND-MONEND
                    ****************************************
                    * READ SECTOR
                    * IN - DRIVE/TRACK/SECTOR/SIDE/DENSITY ALREADY SELECTED
                    *      X ADDRESS OF SECTOR BUFFER
                    *      DBLSTP
                    *      DP=$FF
                    * OUT - CC NE=ERROR, EQ=OK
                    *       A,B,X,Y,U UNCHANGED
                    ****************************************
 A91C 34   36       READS   PSHS   A,B,X,Y
 A91E 31   89 0100          LEAY   256,X
 A922 6D   8D 04C9          TST    DBLSTP,PCR
 A926 27   04               BEQ    RS10
 A928 04   01               LSR    TRKREG    temporarily adjust track # re dbl stp
 A92A 8D   E8               BSR    DELAY
 A92C 86   80       RS10    LDA    #$80      READ SECTOR COMMAND
 A92E 8D   E2               BSR    EXCMD
 A930 20   04               BRA    RS40

 A932 96   03       RS30    LDA    DATREG    (4) .Y, GET DATA BYTE AND STORE
 A934 A7   80               STA    ,X+       (6)
 A936 D6   00       RS40    LDB    STATRG    (4) 19/33/47
 A938 C5   02               BITB   #$02      (2) DRQ?
 A93A 26   F6               BNE    RS30      (3) .Y
 A93C C5   01               BITB   #$01      (2) BUSY?
 A93E 26   F6               BNE    RS40      (3)

 A940 6D   8D 04AB          TST    DBLSTP,PCR
 A944 27   04               BEQ    RS70
 A946 08   01               LSL    TRKREG    restore physical track#
 A948 8D   CA               BSR    DELAY
 A94A 34   20       RS70    PSHS   Y
 A94C AC   E1               CMPX   ,S++      WERE EXACTLY 256 BYTES PROCESSED?
 A94E 26   02               BNE    RS90      .N
 A950 C5   1C               BITB   #$1C      ERRORS? (CRC,LOST DATA,RNF)
 A952 35   B6       RS90    PULS   A,B,X,Y,PC
```

```
             FA07  M       SET    *+MEND-MONEND
                   ****************************************
                   * DELAY APPROX. 100 MSEC.
                   * IN -
                   * OUT - X,CC UNDEFINED
                   *       A,B,Y,U UNCHANGED
                   ****************************************
A954 8E   30D4     DLY100  LDX    #12500
A957 30   1F       DLY101  LEAX   -1,X      (5)
A959 26   FC               BNE    DLY101    (3)
A95B 39                    RTS

             FA0F  M       SET    *+MEND-MONEND
                   ****************************************
                   * SELECT DRIVE, THEN START DRIVE MOTOR AND
                   *  WAIT 1 SEC. FOR IT TO COME UP TO SPEED
                   * IN - BDRIVE
                   *      DP=$FF
                   * OUT - A,B,X,CC UNDEFINED
                   *       Y,U UNCHANGED
                   ****************************************
A95C A6   8D 048E  MOTOR   LDA    BDRIVE,PCR SELECT DRIVE
A960 84   03               ANDA   #$03
A962 44                    LSRA
A963 46                    RORA
A964 46                    RORA
A965 C6   C0               LDB    #$C0
A967 8D   0A               BSR    SETOPR

A969 8E   0000             LDX    #0000     START MOTOR AND WAIT 1 SEC.
A96C 0D   00       MR10    TST    STATRG    (6)
A96E 30   1F               LEAX   -1,X      (5)
A970 26   FA               BNE    MR10      (3)
A972 39                    RTS

             FA26  M       SET    *+MEND-MONEND
                   ****************************************
                   * SET DUART OUTPUT LINES OP0-OP7
                   * IN - A VALUE TO SET OUTPUT LINES TO (BITS NOT
                   *         TO BE CHANGED SHOULD BE ZEROS)
                   *      B MASK RE WHICH BITS ARE TO BE AFFECTED
                   *      ARTOPX
                   *      DP=$FF
                   * OUT - CC UNDEFINED
                   *       A,B,X,Y,U UNCHANGED
                   *       ARTOPX UPDATED
                   ****************************************
A973 34   06       SETOPR  PSHS   A,B
A975 53                    COMB
A976 E4   8D 0473          ANDB   ARTOPX,PCR
A97A EA   E4               ORB    0,S
A97C E7   8D 046D          STB    ARTOPX,PCR
A980 A8   61               EORA   1,S
A982 E6   E4               LDB    0,S
A984 DD   2E               STD    DRVREG
A986 35   86               PULS   A,B,PC

             A988  ENDDSK  EQU    *
             FA3B  M       SET    *+MEND-MONEND
             0000          SETDP  $00       <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

```
                *      END STMONO.TXT

             FA3B  M       SET    *+MEND-MONEND
                   ***************************************
                   *
                   * LOAD MIKBUG S1/S9 FORMAT DATA INTO MEMORY
                   *  (DON'T ECHO)
                   * CALLABLE FROM USER PROGRAM AS WELL AS
                   *  FROM MONITOR MAIN LOGIC
                   ***************************************
A988 6C   8D 0469  LOAD    INC    ICLSW,PCR USE ALTERNATE PORT
A98C 6F   8D 0470          CLR    ECHOFL,PCR
A990 17   03C9     LD05    LBSR   INCHTE
A993 81   53       LD10    CMPA   #'S       START OF RECORD?
A995 26   F9               BNE    LD05
A997 17   03C2             LBSR   INCHTE
A99A 81   39               CMPA   #'9       END-OF-FILE RECORD?
A99C 27   40               BEQ    LD90      .Y
A99E 81   31               CMPA   #'1       DATA RECORD?
A9A0 26   F1               BNE    LD10
A9A2 17   03D0             LBSR   IN2HEX    INPUT BYTE COUNT
A9A5 34   02               PSHS   A
A9A7 25   2C               BCS    LDERR
A9A9 17   03BB             LBSR   IN4HEX    INPUT LOAD ADDRESS
A9AC 25   27               BCS    LDERR
A9AE 34   10               PSHS   X
A9B0 EC   8D 044D          LDD    OFFSET,PCR MODIFY ADDRESS WITH OFFSET
A9B4 30   8B               LEAX   D,X
A9B6 E6   E0               LDB    ,S+       INCLUDE ADDRESS IN CHECKSUM
A9B8 EB   E0               ADDB   ,S+
A9BA EB   E4               ADDB   ,S        INCLUDE BYTE COUNT IN CHECKSUM
A9BC 6A   E4               DEC    ,S        DECR BYTE COUNT RE ADDRESS BYTES
A9BE 6A   E4               DEC    ,S

A9C0 17   03B2     LD20    LBSR   IN2HEX    GET NEXT DATA BYTE
A9C3 25   10               BCS    LDERR
A9C5 34   02               PSHS   A
A9C7 EB   E0               ADDB   ,S+       ADD TO CHECKSUM
A9C9 6A   E4               DEC    ,S        BYTE COUNT ZERO?
A9CB 27   09               BEQ    LD30      .Y
A9CD A7   84               STA    ,X        STORE DATA IN MEMORY
A9CF A1   80               CMPA   ,X+
A9D1 26   02               BNE    LDERR
A9D3 20   EB               BRA    LD20

A9D5 5F           LDERR   CLRB             FORCE DUMMY ERROR CHECKSUM

A9D6 35   02       LD30    PULS   A         RESTORE STACK RE BYTE COUNT
A9D8 5C                    INCB             CHECKSUM O.K.?
A9D9 27   B5               BEQ    LD05      .Y, GET NEXT LINE
A9DB 17   02F7             LBSR   OUTQM     .N, PRINT ERROR MSG AND ABORT
A9DE 63   8D 041E  LD90    COM    ECHOFL,PCR
A9E2 6F   8D 040F          CLR    ICLSW,PCR
A9E6 39                    RTS

             FA9A  M       SET    *+MEND-MONEND
                   ***********************************
                   *
                   * FILL A BLOCK OF MEMORY WITH A SPECIFIED PATTERN
                   *
                   ***********************************
```

```
A9E7 17   037D     FILL    LBSR   IN4HEX    GET STARTING ADDRESS
A9EA 25   22               BCS    FA9O
A9EC 1F   12               TFR    X,Y
A9EE 17   0322             LBSR   OUT1SP
A9F1 17   0373             LBSR   IN4HEX    GET ENDING ADDRESS
A9F4 25   18               BCS    FA9O
A9F6 34   10               PSHS   X
A9F8 1F   21               TFR    Y,X
A9FA 17   0316             LBSR   OUT1SP
A9FD 17   0375             LBSR   IN2HEX    GET PATTERN
AAOO 25   OA               BCS    FA8O

AAO2 AC   E4       FA2O    CMPX   O,S
AAO4 27   O4               BEQ    FA3O
AAO6 A7   8O               STA    ,X+
AAO8 2O   F8               BRA    FA2O
AAOA A7   84       FA3O    STA    O,X

AAOC 32   62       FA8O    LEAS   2,S
AAOE 39            FA9O    RTS

            FAC2   M       SET    *+MEND-MONEND
                   ****************************************
                   *
                   * BLOCK MOVE
                   *
                   ****************************************
AAOF 17   0355     BLKMOV  LBSR   IN4HEX    GET STARTING ADDR OF 'FROM' BLOCK
AA12 25   26               BCS    BM9O
AA14 1F   12               TFR    X,Y
AA16 17   O2FA             LBSR   OUT1SP
AA19 17   034B             LBSR   IN4HEX    GET END ADDR OF 'FROM' BLOCK
AA1C 25   1C               BCS    BM9O
AA1E 34   10               PSHS   X
AA2O 17   O2FO             LBSR   OUT1SP
AA23 17   0341             LBSR   IN4HEX    GET STARTING ADDR OF 'TO' BLOCK
AA26 25   10               BCS    BM8O
AA28 1E   12               EXG    X,Y

AA2A AC   E4       BM2O    CMPX   O,S
AA2C 27   O6               BEQ    BM3O
AA2E A6   8O               LDA    ,X+
AA3O A7   AO               STA    ,Y+
AA32 2O   F6               BRA    BM2O
AA34 A6   84       BM3O    LDA    O,X
AA36 A7   A4               STA    O,Y

AA38 32   62       BM8O    LEAS   2,S
AA3A 39            BM9O    RTS

            FAEE   M       SET    *+MEND-MONEND
                   ****************************************
                   * SEARCH FOR A SPECIFIED BIT PATTERN WITHIN
                   * SPECIFIED BOUNDS OF MEMORY.  HIT "ESC" TO
                   * HALT DISPLAY, HIT "ESC" AGAIN TO RESUME,
                   * "RETURN" TO ABORT.
                   ****************************************
AA3B 17   0329     SEARCH  LBSR   IN4HEX    INPUT STARTING ADDRESS
AA3E 25   47               BCS    SH99
AA4O 1F   12               TFR    X,Y
AA42 17   O2CE             LBSR   OUT1SP
```

```
AA45 17   031F            LBSR   IN4HEX    INPUT ENDING ADDRESS
AA48 25   3D              BCS    SH99
AA4A 17   02C6            LBSR   OUT1SP
AA4D 17   0325            LBSR   IN2HEX    INPUT BIT PATTERN
AA50 25   35              BCS    SH99
AA52 34   12              PSHS   A,X

AA54 A1   A0       SH50   CMPA   ,Y+       DOES PATTERN MATCH?
AA56 26   24              BNE    SH70      .N
AA58 8D   2E              BSR    SUSPEND   .Y, SUSPEND DISPLAY?
AA5A 25   29              BCS    SH90      .        ABORT

AA5C 17   026C            LBSR   PCRLF     DISPLAY ADDRESS OF MATCHING BYTE
AA5F 30   3F              LEAX   -1,Y
AA61 17   0291            LBSR   OUT4HX
AA64 17   02AC            LBSR   OUT1SP    DISPLAY 5 BYTES DATA (XXXX XX XXXX)
AA67 AE   3D              LDX    -3,Y
AA69 17   0289            LBSR   OUT4HX
AA6C 17   02A4            LBSR   OUT1SP
AA6F A6   3F              LDA    -1,Y
AA71 17   0289            LBSR   OUT2HX
AA74 17   029C            LBSR   OUT1SP
AA77 AE   A4              LDX    0,Y
AA79 17   0279            LBSR   OUT4HX

AA7C 10AC 61       SH70   CMPY   1,S       REACHED END LIMIT?
AA7F 22   04              BHI    SH90      .Y
AA81 A6   E4              LDA    0,S
AA83 20   CF              BRA    SH50

AA85 35   12       SH90   PULS   A,X
AA87 39            SH99   RTS

            FB3B  M       SET    *+MEND-MONEND
                  ******************************************
                  * CHECK TO SEE IF SUSPEND CHARACTER ("ESC") KEYED.
                  *  IF YES, WAIT FOR RESUME ("ESC") OR ABORT ("RETURN")
                  *  CHARACTER TO BE KEYED.
                  * IN - NONE
                  * OUT - CC CC=CONTINUE, CS=ABORT
                  *       A UNDEFINED
                  *       B,X,Y,U UNCHANGED
                  ******************************************
AA88 17   031A     SUSPEND LBSR   INCHEK    CHARACTER KEYED?
AA8B 27   11              BEQ    SU80      .N
AA8D 17   02B7            LBSR   INCHN     .Y, GET IT
AA90 81   1B              CMPA   #ESC      SUSPEND CHARAC?
AA92 26   0A              BNE    SU80      .N, IGNORE
AA94 17   02B0            LBSR   INCHN     .Y, WAIT FOR RESTART CHARAC
AA97 81   0D              CMPA   #CR       "RETURN"?
AA99 26   03              BNE    SU80      .N, RESUME
AA9B 43                   COMA             .Y, ABORT
AA9C 20   01              BRA    SU90

AA9E 4F            SU80   CLRA             CONTINUE
AA9F 39            SU90   RTS

            FB53  M       SET    *+MEND-MONEND
                  ******************************************
                  *
                  * EXAMINE A BLOCK OF MEMORY, AND DISPLAY
```

```
                *  IN HEX AND ASCII.  HIT "ESC" TO HALT DISPLAY,
                *  HIT "ESC" AGAIN TO RESUME, "RETURN" TO ABORT.
                *****************************************
AAA0 17   02C4     EXAMB   LBSR   IN4HEX    INPUT STARTING ADDRESS
AAA3 25   32               BCS    EB95
AAA5 1F   12               TFR    X,Y
AAA7 17   0269             LBSR   OUT1SP
AAAA 17   02BA             LBSR   IN4HEX    INPUT ENDING ADDRESS
AAAD 25   28               BCS    EB95

AAAF 1F   10               TFR    X,D       ADJUST ENDING ADDR TO NEXT 16 BYTE BOUNDARY
AAB1 C3   0010             ADDD   #16
AAB4 C4   F0               ANDB   #$F0
AAB6 34   06               PSHS   A,B

AAB8 1F   20               TFR    Y,D       ADJUST STARTING ADDR TO BEGINNING OF THIS
AABA C4   F0               ANDB   #$F0      . 16 BYTE BOUNDARY
AABC 1F   02               TFR    D,Y

AABE 86   10               LDA    #16       16 BYTES PER LINE
AAC0 34   22               PSHS   A,Y
AAC2 10AC 63       EB50    CMPY   3,S       END OF LIMIT?
AAC5 24   0E               BHS    EB90      .Y
AAC7 8D   BF               BSR    SUSPEND   .N, SUSPEND?
AAC9 25   0A               BCS    EB90      .       ABORT
AACB 17   01FD             LBSR   PCRLF
AACE 17   0242             LBSR   OUT1SP
AAD1 8D   05               BSR    EXAML     DISPLAY LINE
AAD3 20   ED               BRA    EB50

AAD5 32   65       EB90    LEAS   5,S
AAD7 39            EB95    RTS

            FB8B   M       SET    *+MEND-MONEND
                *****************************************
                * DISPLAY 16 BYTES OF MEMORY IN
                *  HEX AND ASCII FORM ON ONE LINE
                *****************************************
AAD8 AE   63       EXAML   LDX    3,S       OUTPUT ADDR OF BEGIN OF 16 BYTE LINE
AADA 17   0218             LBSR   OUT4HX
AADD 17   0233             LBSR   OUT1SP
AAE0 17   0230             LBSR   OUT1SP

AAE3 E6   62               LDB    2,S       DISPLAY LINE IN HEX IN GROUPS OF 4 BYTES EACH
AAE5 57                    ASRB
AAE6 57                    ASRB
AAE7 10AE 63               LDY    3,S
AAEA AE   A1       EL20    LDX    ,Y++
AAEC 17   0206             LBSR   OUT4HX
AAEF AE   A1               LDX    ,Y++
AAF1 17   0201             LBSR   OUT4HX
AAF4 17   021C             LBSR   OUT1SP
AAF7 5A                    DECB
AAF8 26   F0               BNE    EL20
AAFA 17   0216             LBSR   OUT1SP

AAFD E6   62               LDB    2,S       DISPLAY LINE IN ASCII
AAFF 10AE 63               LDY    3,S
AB02 A6   A0       EL40    LDA    ,Y+
AB04 81   20               CMPA   #$20      PRINTABLE CHARAC?
AB06 25   04               BLO    EL60      .N
```

```
AB08 81   7E              CMPA   #$7E
AB0A 23   02              BLS    EL65      .Y
AB0C 86   2E       EL60   LDA    #'.       .N
AB0E 17   01C6     EL65   LBSR   OUTCH
AB11 5A                   DECB
AB12 26   EE              BNE    EL40

AB14 10AF 63              STY    3,S       UPDATE POINTER
AB17 39                   RTS

             FBCB  M      SET    *+MEND-MONEND
                   ****************************************
                   * MEMORY TEST - 65536 PASSES -
                   *  PRINT "*" AFTER EACH SUCCESSFUL PASS;
                   *  PRINT ADDRESS AND BIT ERRORS FOR EACH ERROR.
                   *  HITTING ANY KEY ABORTS THE TEST AT THE END OF
                   *  THE CURRENT PASS.
                   ****************************************
AB18 32   7A       MEMTST LEAS   -6,S
AB1A 6F   E4              CLR    0,S       INIT PASS # TO 0
AB1C 6F   61              CLR    1,S
AB1E 17   0246            LBSR   IN4HEX    INPUT STARTING ADDRESS
AB21 25   78              BCS    MT90
AB23 1F   12              TFR    X,Y
AB25 17   01EB            LBSR   OUT1SP
AB28 17   023C            LBSR   IN4HEX    INPUT ENDING ADDRESS
AB2B 25   6E              BCS    MT90
AB2D AF   64              STX    4,S
AB2F 10AF 62              STY    2,S
AB32 10AC 64              CMPY   4,S       BEGIN <= END ?
AB35 22   64              BHI    MT90      .N, ERROR
                   *
                   * STORE PATTERN INTO MEMORY LOCATIONS
                   *
AB37 1F   20       MT10   TFR    Y,D       USE ADDRESS
AB39 A3   E4              SUBD   0,S       SUBTRACT PASS #
AB3B 34   02              PSHS   A         ADD MSB & LSB
AB3D EB   E0              ADDB   ,S+
AB3F E7   A0              STB    ,Y+       STORE
AB41 10AC 64              CMPY   4,S       END OF PASS?
AB44 23   F1              BLS    MT10
                   *
                   * DELAY FOR 200 MSEC TO TEST FOR DYNAMIC MEMORY DROPOUTS
                   *  (DO NOT COMBINE THE TWO DELAYS INTO 2 CALLS TO ONE SUBRTN)
                   *
AB46 86   9C              LDA    #156      (2)
AB48 5F           MT20    CLRB             (2) -
AB49 5A           MT25    DECB             (2)  >  1282 CYCLES
AB4A 26   FD              BNE    MT25      (3) -
AB4C 4A                   DECA             (2)
AB4D 26   F9              BNE    MT20      (3)

AB4F 86   9C              LDA    #156
AB51 5F           MT30    CLRB
AB52 5A           MT35    DECB
AB53 26   FD              BNE    MT35
AB55 4A                   DECA
AB56 26   F9              BNE    MT30
                   *
                   * READ PATTERN BACK TO SEE IF OK
                   *
```

```
AB58 10AE 62              LDY    2,S
AB5B 1F   20       MT40   TFR    Y,D
AB5D A3   E4              SUBD   0,S
AB5F 34   02              PSHS   A
AB61 EB   E0              ADDB   ,S+
AB63 E8   A0              EORB   ,Y+       DATA MATCHES?
AB65 27   15              BEQ    MT50      .Y
AB67 30   3F              LEAX   -1,Y      .N, OUTPUT ADDRESS AND BITS IN ERROR
AB69 17   0189            LBSR   OUT4HX
AB6C 17   01A4            LBSR   OUT1SP
AB6F 1F   98              TFR    B,A
AB71 17   0189            LBSR   OUT2HX
AB74 17   019C            LBSR   OUT1SP
AB77 17   022B            LBSR   INCHEK
AB7A 26   1C              BNE    MT89

AB7C 10AC 64       MT50   CMPY   4,S
AB7F 23   DA              BLS    MT40
AB81 86   2A              LDA    #'*
AB83 17   0151            LBSR   OUTCH
AB86 17   021C            LBSR   INCHEK
AB89 26   0D              BNE    MT89
AB8B 10AE 62              LDY    2,S
AB8E AE   E4              LDX    0,S       INCREMENT PASS COUNT
AB90 30   01              LEAX   1,X
AB92 AF   E4              STX    0,S
AB94 26   A1              BNE    MT10      65,536 PASSES DONE?
AB96 20   03              BRA    MT90

AB98 17   0180     MT89   LBSR   INCH8     DISCARD ABORT CHARAC

AB9B 32   66       MT90   LEAS   6,S
AB9D 39                   RTS

            FC51   M      SET    *+MEND-MONEND
                   ***********************************************
                   *
                   * EXECUTE ROUTINE AT SPECIFIED ADDRESS.
                   *  RETURN VIA 'RTS', OR JUMP TO 'SIGNON'
                   * Upon return, the stack pointer should be the same
                   *  as it was upon entry.
                   * Do NOT use to return to FLEX or to a program that
                   *  will exit to FLEX -- use the "F W" or "F U"
                   *  commands instead.
                   *
                   ***********************************************
AB9E 17   01C6     JUMP   LBSR   IN4HEX
ABA1 25   02              BCS    JM90
ABA3 AD   84              JSR    0,X
ABA5 39            JM90   RTS

            FC59   M      SET    *+MEND-MONEND
                   ***********************************************
                   * COMMAND TABLE
                   *  Each entry consists of 3 bytes:
                   *   - byte 0 = command code
                   *   - bytes 1,2 = 16 bit signed offset of beginning
                   *                 of routine to beginning of CMDTAB
                   ***********************************************
ABA6 44            CMDTAB FCB    'D
ABA7 FC18                 FDB    DISKBT-CMDTAB
```

```
ABA9 4A                   FCB    'J
ABAA FFF8                 FDB    JUMP-CMDTAB
ABAC 54                   FCB    'T
ABAD FF72                 FDB    MEMTST-CMDTAB
ABAF 45                   FCB    'E
ABBO FEFA                 FDB    EXAMB-CMDTAB
ABB2 53                   FCB    'S
ABB3 FE95                 FDB    SEARCH-CMDTAB
ABB5 42                   FCB    'B
ABB6 FE69                 FDB    BLKMOV-CMDTAB
ABB8 41                   FCB    'A
ABB9 FE41                 FDB    FILL-CMDTAB
ABBB 4C                   FCB    'L
ABBC FDE2                 FDB    LOAD-CMDTAB
ABBE FF FF FF             FCB    $FF,$FF,$FF (spare)
ABC1 FF FF FF             FCB    $FF,$FF,$FF (spare)
ABC4 FF FF FF             FCB    $FF,$FF,$FF (spare)
ABC7 FF FF FF             FCB    $FF,$FF,$FF (spare)

             FC7D  M      SET    *+MEND-MONEND
                   ****************************************
                   *
                   *    M A I N     L O G I C
                   *
                   ****************************************
ABCA 1OEE 8D O221  SIGNON LDS    STACK,PCR
ABCF 17   OOF9            LBSR   PCRLF
ABD2 30   8D O1E4         LEAX   TITLE,PCR OUTPUT SIGN ON MESSAGE
ABD6 17   OOE5            LBSR   PSTRNG

ABD9 4F           NXTCMD  CLRA
ABDA A7   8D O217         STA    ICLSW,PCR
ABDE A7   8D O214         STA    OCLSW,PCR
ABE2 43                   COMA
ABE3 A7   8D O219         STA    ECHOFL,PCR
ABE7 A7   8D O214         STA    UCFLAG,PCR
ABEB 30   8D O1F2         LEAX   PROMPT,PCR OUTPUT PROMPT
ABEF 17   OOCC            LBSR   PSTRNG
ABF2 17   O16D            LBSR   INCHE     INPUT COMMAND FROM CONSOLE AND ECHO
ABF5 17   O11B            LBSR   OUT1SP
ABF8 81   4D              CMPA   #'M       MEMORY CHANGE?
ABFA 27   2A              BEQ    MEMCHG
ABFC 81   46              CMPA   #'F       RETURN TO FLEX?
ABFE 1027 0098            LBEQ   FLXRTN
AC02 81   0D              CMPA   #CR       CR?
AC04 27   D3              BEQ    NXTCMD

AC06 E6   8D O1E8         LDB    TABSIZ,PCR EXTENDED COMMAND TABLE?
AC0A 27   0C              BEQ    ERR
AC0C 30   8C 97           LEAX   CMDTAB,PCR
AC0F A1   80      MN50    CMPA   ,X+       FOUND IN COMMAND TABLE?
AC11 27   0A              BEQ    MN80      .Y
AC13 30   02              LEAX   2,X       .N, TRY NEXT ENTRY
AC15 5A                   DECB             END OF TABLE?
AC16 26   F7              BNE    MN50      .N

AC18 17   00BA    ERR     LBSR   OUTQM     DISPLAY "?"
AC1B 20   BC              BRA    NXTCMD

AC1D EC   84      MN80    LDD    0,X       GET DISPLACEMENT
AC1F 30   8C 84           LEAX   CMDTAB,PCR
```

```
AC22 AD   8B              JSR    D,X       EXECUTE COMMAND
AC24 20   B3              BRA    NXTCMD

            FCD9  M       SET    *+MEND-MONEND
                  ************************************
                  * MEMORY CHANGE
                  *  DISPLAY MEMORY DATA (HEX AND ASCII)
                  *  AND ALLOW TO BE MODIFIED, ONE BYTE
                  *  AT A TIME
                  ************************************
AC26 17   013E    MEMCHG  LBSR   IN4HEX    INPUT ADDRESS
AC29 25   6C              BCS    MC90

AC2B 17   009D    MC10    LBSR   PCRLF     NEW LINE
AC2E 17   00C4            LBSR   OUT4HX    DISPLAY ADDRESS
AC31 17   00DF            LBSR   OUT1SP
AC34 A6   84              LDA    ,X        DISPLAY CURRENT VALUE IN HEX
AC36 34   02              PSHS   A
AC38 17   00C2            LBSR   OUT2HX
AC3B 17   00D5            LBSR   OUT1SP
AC3E 35   02              PULS   A         DISPLAY CURRENT VALUE IN ASCII
AC40 81   20              CMPA   #$20      PRINTABLE?
AC42 25   04              BLO    MC20
AC44 81   7E              CMPA   #$7E
AC46 23   02              BLS    MC25
AC48 86   20      MC20    LDA    #$20
AC4A 17   008A    MC25    LBSR   OUTCH
AC4D 17   00C3            LBSR   OUT1SP
AC50 17   0122            LBSR   IN2HEX    INPUT NEW VALUE
AC53 24   2F              BCC    MC50      VALID HEX DATA, GO TO STORE ROUTINE
AC55 81   2C              CMPA   #',       "," OR " " = NO CHANGE, DISPLAY NEXT
AC57 27   33              BEQ    MC80
AC59 81   20              CMPA   #'        .
AC5B 27   2F              BEQ    MC80
AC5D 81   08              CMPA   #BKSPC    "BACKSPACE" OR "-" OR "^" = DISPLAY PREV
AC5F 27   2F              BEQ    MC85
AC61 81   2D              CMPA   #'-
AC63 27   2B              BEQ    MC85
AC65 81   5E              CMPA   #'^
AC67 27   27              BEQ    MC85
AC69 81   2E              CMPA   #'.       "." OR "RETURN" = STOP
AC6B 27   2A              BEQ    MC90
AC6D 81   0D              CMPA   #CR
AC6F 27   26              BEQ    MC90
AC71 81   27              CMPA   #''       "'" = ENTER ASCII DATA
AC73 27   04              BEQ    MC40
AC75 8D   5E              BSR    OUTQM     INVALID DATA OR COMMAND, DISPLAY "?"
AC77 20   B2              BRA    MC10

AC79 17   009F    MC40    LBSR   INCH8     INPUT ASCII DATA
AC7C 84   7F              ANDA   #$7F      STRIP PARITY
AC7E 81   1F              CMPA   #$1F      DONT'T ECHO CONTROL CHARACTERS
AC80 23   02              BLS    MC50
AC82 8D   53              BSR    OUTCH     ECHO
AC84 A7   84      MC50    STA    0,X       STORE NEW VALUE INTO MEMORY
AC86 A1   84              CMPA   0,X       CHECK TO SEE IF STORED OK
AC88 27   02              BEQ    MC80      .Y
AC8A 8D   49              BSR    OUTQM     .N, OUTPUT "?"
AC8C 30   01      MC80    LEAX   1,X       INCR. ADDRESS COUNTER
AC8E 20   02              BRA    MC87
AC90 30   1F      MC85    LEAX   -1,X      DECR. ADDRESS COUNTER
```

```
AC92 8C   FF00     MC87    CMPX   #$FF00
AC95 25   94               BLO    MC10

AC97 16   FF3F     MC90    LBRA   NXTCMD    RETURN

             FD4D  M       SET    *+MEND-MONEND
                   ********************************************
                   * RETURN TO FLEX ENVIRONMENT, EITHER TO FLEX WARMSTART
                   *  OR TO ADDRESS SPECIFIED.  FLEX MUST HAVE BEEN
                   *  ACTIVE PREVIOUS TO ENTERING "MONITOR" AND MUST
                   *  NOT BE DAMAGED, OTHERWISE RESULTS WILL BE
                   *  UNPREDICTABLE.  IF "MONITOR" WAS ENTERED VIA THE
                   *  "MON" COMMAND OF FLEX, EVERYTHING IS OK AS LONG
                   *  AS NO FLEX MEMORY LOCATIONS HAVE BEEN MODIFIED.
                   ********************************************
AC9A 17   00C5     FLXRTN  LBSR   INCHE     INPUT TYPE OF RETURN
AC9D 81   55               CMPA   #'U       RETURN TO "USER" ADDRESS?
AC9F 27   09               BEQ    FX40
ACA1 81   57               CMPA   #'W       RETURN TO "WARMS" OF FLEX?
ACA3 26   14               BNE    FX80

ACA5 8E   CD03             LDX    #WARMS    STORE "WARMS" IN "JMPADDR"
ACA8 20   07               BRA    FX70

ACAA 8D   67       FX40    BSR    OUT1SP
ACAC 17   00B8             LBSR   IN4HEX    GET ADDRESS
ACAF 25   08               BCS    FX80

ACB1 AF   8D 013E  FX70    STX    JMPADDR,PCR
ACB5 6E   9F D3E1          JMP    [VRETURN] RETURN TO FLEX ENVIRONMENT

ACB9 8D   1A       FX80    BSR    OUTQM     ERROR
ACBB 16   FF1B             LBRA   NXTCMD

             FD71  M       SET    *+MEND-MONEND
                   ********************************************
                   * PSTRNG - OUTPUT CHARACTER STRING TO SERIAL
                   *  PORT A OR B;  PRECEDE WITH CR/LF
                   * IN - X POINTS TO START OF STRING
                   * OUT - A UNDEFINED
                   *       X POINTS TO END OF STRING + 1
                   *       B,Y,U UNCHANGED
                   *       CC ALWAYS EQ
                   *
                   * PDATA - OUTPUT CHARACTER STRING TO SERIAL
                   *  PORT A OR B
                   * IN - X POINTS TO START OF STRING
                   * OUT - A UNDEFINED
                   *       X POINTS TO END OF STRING + 1
                   *       B,Y,U UNCHANGED
                   *       CC ALWAYS EQ
                   *
                   * PCRLF - OUTPUT CARRIAGE RETURN, LINE FEED, NULLS
                   *  TO SERIAL PORT A OR B
                   * IN - NONE
                   * OUT - CC ALWAYS EQ
                   *       A,B,X,Y,U  UNCHANGED
                   ********************************************
ACBE 8D   0B       PSTRNG  BSR    PCRLF

ACC0 A6   80       PDATA   LDA    ,X+
```

```
ACC2 81   04              CMPA   #EOT
ACC4 27   04              BEQ    PS20
ACC6 8D   0F              BSR    OUTCH
ACC8 20   F6              BRA    PDATA
ACCA 39           PS20    RTS

ACCB 34   12      PCRLF   PSHS   A,X
ACCD 30   8D 0112         LEAX   CRLF,PCR
ACD1 8D   ED              BSR    PDATA
ACD3 35   92              PULS   A,X,PC

            FD88  M       SET    *+MEND-MONEND
                  ****************************
                  * OUTPUT A QUESTION MARK TO SERIAL PORT A OR B
                  * IN - OCLSW/OPORT/OPORTA
                  * OUT - A,CC UNDEFINED
                  *       B,X,Y,U UNCHANGED
                  ****************************
ACD5 86   3F      OUTQM   LDA    #'?

            FD8A  M       SET    *+MEND-MONEND
                  ************************************************
                  * OUTPUT ONE 8-BIT CHARACTER TO SERIAL PORT A OR B
                  *  (WAIT UNTIL READY TO ACCEPT),
                  *  HIGH-ORDER BIT IS NOT CHANGED, ALTHOUGH IT IS
                  *   TRUNCATED IF PORT INITIALIZED FOR ONLY 7 DATA BITS
                  * IN - A CHARAC TO SEND
                  *      OCLSW/OPORT/OPORTA
                  * OUT - CC UNDEFINED
                  *       ALL OTHER REGISTERS UNCHANGED
                  ************************************************
ACD7 34   12      OUTCH   PSHS   A,X
ACD9 6D   8D 0119         TST    OCLSW,PCR
ACDD 27   06              BEQ    OT05
ACDF AE   8D 011A         LDX    OPORTA,PCR
ACE3 20   04              BRA    OT50
ACE5 AE   8D 0112 OT05    LDX    OPORT,PCR

ACE9 A6   01      OT50    LDA    1,X       TEST STATUS
ACEB 85   04              BITA   #$04
ACED 27   FA              BEQ    OT50
ACEF 35   02              PULS   A
ACF1 A7   03              STA    3,X       OUTPUT CHARACTER

ACF3 35   90              PULS   X,PC      RESTORE X, RETURN

            FDA8  M       SET    *+MEND-MONEND
                  ********************
                  * OUTPUT 4 ASCII HEX DIGITS TO SERIAL PORT A OR B
                  * IN - X CONTAINS 4 DIGIT HEX VALUE
                  *      OCLSW/OPORT/OPORTA
                  * OUT - A UNDEFINED
                  *       ALL OTHER REGISTERS PRESERVED
                  ********************
ACF5 34   10      OUT4HX  PSHS   X
ACF7 35   02              PULS   A
ACF9 8D   02              BSR    OUT2HX
ACFB 35   02              PULS   A
```

```
                FDBO  M       SET    *+MEND-MONEND
                      ********************
                      * OUTPUT 2 ASCII HEX DIGITS TO SERIAL PORT A OR B
                      * IN - A CONTAINS 2 DIGIT HEX VALUE
                      *      OCLSW/OPORT/OPORTA
                      * OUT - A UNDEFINED
                      *       ALL OTHER REGISTERS PRESERVED
                      ********************
ACFD 34   02          OUT2HX  PSHS   A
ACFF 44               LSRA
AD00 44               LSRA
AD01 44               LSRA
AD02 44               LSRA
AD03 8D   04          BSR    OUT1HX
AD05 35   02          PULS   A
AD07 84   0F          ANDA   #$0F

                FDBC  M       SET    *+MEND-MONEND
                      ********************
                      * OUTPUT 1 ASCII HEX DIGIT TO SERIAL PORT A OR B
                      * IN - A LOWER 4 BITS CONTAINS 1 DIGIT HEX VALUE
                      *      OCLSW/OPORT/OPORTA
                      * OUT - A UNDEFINED
                      *       ALL OTHER REGISTERS PRESERVED
                      ********************
AD09 8B   30          OUT1HX  ADDA   #$30
AD0B 81   39          CMPA   #$39
AD0D 2F   02          BLE    OT190
AD0F 8B   07          ADDA   #$07
AD11 20   C4          OT190   BRA    OUTCH

                FDC6  M       SET    *+MEND-MONEND
                      **********************
                      * OUTPUT 1 SPACE CHARACTER TO SERIAL PORT A OR B
                      * IN - OCLSW/OPORT/OPORTA
                      * OUT - CC UNDEFINED
                      *       ALL OTHER REGISTERS PRESERVED
                      **********************
AD13 34   02          OUT1SP  PSHS   A
AD15 86   20          LDA    #$20
AD17 8D   BE          BSR    OUTCH
AD19 35   82          PULS   A,PC

                FDCE  M       SET    *+MEND-MONEND
                      ************************************************
                      * GET ONE 8-BIT CHARACTER FROM SERIAL PORT A OR B, NO ECHO
                      *  (WAIT UNTIL ONE HAS BEEN RECEIVED)
                      * IF OVERRUN ERROR DETECTED, AN ASCII "BELL" CHARACTER
                      *  IS SENT VIA ROUTINE "OUTCH"
                      * IN - ICLSW/IPORT/IPORTL
                      * OUT - A CHARC RECEIVED (PARITY LEFT AS IS)
                      *       CC UNDEFINED
                      *       B,X,Y,U UNCHANGED
                      ************************************************
AD1B 34   10          INCH8   PSHS   X         SAVE X
AD1D 6D   8D 00D4             TST    ICLSW,PCR
AD21 27   06                  BEQ    IN05
AD23 AE   8D 00D2             LDX    IPORTL,PCR
AD27 20   04                  BRA    IN50
AD29 AE   8D 00CA   IN05      LDX    IPORT,PCR
```

```
AD2D A6   01       IN5O    LDA    1,X       TEST STATUS
AD2F 85   01               BITA   #$01      CHARACTER WAITING?
AD31 27   FA               BEQ    IN5O      .N
AD33 85   10               BITA   #$10      OVERRUN ERROR?
AD35 27   0C               BEQ    IN6O      .N
AD37 86   07               LDA    #BELL     .Y, OUTPUT "BELL"
AD39 8D   9C               BSR    OUTCH
AD3B 12                    NOP               // TC // re FLEX Conversion PATCHES.OVR old file
AD3C 12                    NOP               // TC //
AD3D 20   00               BRA    IN55      // TC //
AD3F 86   40       IN55    LDA    #$40      RESET OVERRUN ERROR
AD41 A7   02               STA    2,X

AD43 A6   03       IN6O    LDA    3,X       .Y,GET CHARACTER

AD45 35   90       IN9O    PULS   X,PC

            FDFA  M       SET    *+MEND-MONEND
                  *********************
                  * GET ONE CHARACTER FROM SERIAL PORT A OR B, NO ECHO,
                  *  (WAIT UNTIL ONE HAS BEEN RECEIVED),
                  *  STRIP PARITY BIT, CONVERT a - z TO A - Z IF FLAG SET
                  * IN - ICLSW/IPORT/IPORTL/UCFLAG
                  * OUT - A CHARC RECEIVED
                  *       CC UNDEFINED
                  *       B,X,Y,U UNCHANGED
                  *********************
AD47 8D   D2       INCHN   BSR    INCH8
AD49 84   7F               ANDA   #$7F      STRIP PARITY
AD4B 6D   8D 00B0          TST    UCFLAG,PCR CONVERT TO UPPER CASE?
AD4F 27   0A               BEQ    IN19O
AD51 81   61               CMPA   #'a
AD53 25   06               BLO    IN19O
AD55 81   7A               CMPA   #'z
AD57 22   02               BHI    IN19O
AD59 84   5F               ANDA   #$5F      .Y
AD5B 39           IN19O    RTS

            FEOF  M       SET    *+MEND-MONEND
                  **********************
                  * GET ONE CHARAC FROM SERIAL PORT A OR B,
                  *  (WAIT UNTIL ONE IS RECEIVED), STRIP PARITY BIT,
                  *  CONVERT TO UPPER CASE, DEPENDING ON 'UCFLAG',
                  *  ECHO TO SERIAL PORT A OR B DEPENDING ON 'ECHOFL'
                  * IN - ICLSW/IPORT/IPORTL/UCFLAG/ECHOFL
                  * OUT - A CHARAC RECEIVED
                  *       CC UNDEFINED
                  *       ALL OTHERS UNCHANGED
                  **********************
AD5C 6D   8D 00A0  INCHTE  TST    ECHOFL,PCR
AD60 27   E5               BEQ    INCHN

            FE15  M       SET    *+MEND-MONEND
                  *********************
                  * GET ONE CHARAC FROM SERIAL PORT A OR B,
                  *  (WAIT UNTIL ONE IS RECEIVED), STRIP PARITY BIT,
                  *  CONVERT TO UPPER CASE, DEPENDING ON 'UCFLAG',
                  *  ECHO TO SERIAL PORT A OR B
                  * IN - ICLSW/IPORT/IPORTL/UCFLAG
                  * OUT - A CHARAC RECEIVED
                  *       CC UNDEFINED
```

```
                    *       ALL OTHERS UNCHANGED
                    ********************
AD62 8D   E3        INCHE   BSR    INCHN
AD64 16   FF70              LBRA   OUTCH

              FE1A  M       SET    *+MEND-MONEND
                    **********************
                    * INPUT 4 ASCII HEX DIGITS (ECHO IF FLAG)
                    *  RETURN WITH 16 BITS IN REG X, CARRY SET IF ERROR
                    *  REG A UNDEFINED, ALL OTHERS PRESERVED
                    **********************
AD67 8D   0C        IN4HEX  BSR    IN2HEX    GET 1ST 2 HEX DIGITS
AD69 25   37                BCS    HEXERR
AD6B 34   06                PSHS   A,B       SAVE 1ST BYTE, SAVE SPACE FOR 2ND
AD6D 8D   06                BSR    IN2HEX    GET 2ND 2 DIGITS
AD6F 25   2D                BCS    HEXERR-4
AD71 A7   61                STA    1,S
AD73 35   90                PULS   X,PC      LOAD VALUE INTO X, RETURN

              FE28  M       SET    *+MEND-MONEND
                    **********************
                    * INPUT 2 ASCII HEX DIGITS (ECHO IF FLAG)
                    * OK RETURN - A 8 BIT BINARY VALUE
                    *             CC CARRY CLEAR
                    *             ALL OTHERS UNCHANGED
                    * ERROR RETURN - A LAST ASCII HEX DIGIT KEYED
                    *                CC CARRY SET
                    *                ALL OTHERS UNCHANGED
                    **********************
AD75 8D   0F        IN2HEX  BSR    IN1HEX    GET 1ST HEX DIGIT
AD77 25   29                BCS    HEXERR
AD79 48                     ASLA             SHIFT INTO LEFT SIDE OF BYTE
AD7A 48                     ASLA
AD7B 48                     ASLA
AD7C 48                     ASLA
AD7D 34   02                PSHS   A
AD7F 8D   05                BSR    IN1HEX    GET 2ND DIGIT
AD81 25   1D                BCS    HEXERR-2
AD83 AB   E0                ADDA   ,S+       MERGE 2 DIGITS
AD85 39                     RTS

              FE39  M       SET    *+MEND-MONEND
                    **********************
                    * INPUT 1 ASCII HEX DIGIT (ECHO IF FLAG)
                    * OK RETURN - A BINARY VALUE IN LOWER 4 BITS
                    *             CC CARRY CLEAR
                    *             ALL OTHERS UNCHANGED
                    * ERROR RETURN - A ASCII HEX DIGIT
                    *                CC CARRY SET
                    *                ALL OTHERS UNCHANGED
                    **********************
AD86 8D   D4        IN1HEX  BSR    INCHTE
AD88 81   30                CMPA   #'0
AD8A 25   16                BLO    HEXERR
AD8C 81   39                CMPA   #'9
AD8E 22   03                BHI    IHAF
AD90 80   30                SUBA   #$30
AD92 39                     RTS

AD93 81   41        IHAF    CMPA   #'A
AD95 25   0B                BLO    HEXERR
```

```
AD97 81   46              CMPA   #'F
AD99 22   07              BHI    HEXERR
AD9B 80   37              SUBA   #$37
AD9D 39                   RTS

AD9E 32   61              LEAS   1,S
ADA0 32   61              LEAS   1,S
ADA2 1A   01       HEXERR SEC             SET ERROR FLAG
ADA4 39                   RTS

             FE58 M       SET    *+MEND-MONEND
                  ******************************************
                  * CHECK SERIAL PORT A OR B TO SEE IF CHARACTER
                  *  WAITING TO BE RECEIVED.
                  * IN - ICLSW/IPORT/IPORTL
                  * OUT - CC EQ IF NO CHARAC WAITING
                  *          NE IF CHARAC WAITING
                  *       ALL OTHER REGISTERS UNCHANGED
                  ******************************************
ADA5 34   12       INCHEK PSHS   A,X
ADA7 6D   8C 4B           TST    <ICLSW,PCR
ADAA 27   05              BEQ    IC05
ADAC AE   8C 4A           LDX    <IPORTL,PCR
ADAF 20   03              BRA    IC50
ADB1 AE   8C 43    IC05   LDX    <IPORT,PCR

ADB4 A6   01       IC50   LDA    1,X
ADB6 85   01              BITA   #$01
ADB8 35   92              PULS   A,X,PC


                  ****************************
                  *
                  *   D A T A     S E C T I O N
                  *
                  ****************************

             FE6D M       SET    *+MEND-MONEND
                  *
                  * MESSAGE STRINGS
                  *
ADBA 53 54 2D 4D  TITLE  FCC    'ST-MON 2.04 (c) 1984 BY DAVID C. WIENS',EOT
ADBE 4F 4E 20 32
ADC2 2E 30 34 20
ADC6 28 63 29 20
ADCA 31 39 38 34
ADCE 20 42 59 20
ADD2 44 41 56 49
ADD6 44 20 43 2E
ADDA 20 57 49 45
ADDE 4E 53 04
ADE1 3D 04        PROMPT FCB    '=,EOT
ADE3 0D 0A 00 00  CRLF   FCB    CR,LF,0,0,0,0,EOT
ADE7 00 00 04

             FE9D M       SET    *+MEND-MONEND
                  *
                  * VARIABLES IN RAM -- WITH INITIALIZED DEFAULT VALUES
                  *
ADEA FF                   FCB    $FF       spare
ADEB FF                   FCB    $FF       spare
```

```
ADEC 00           ARTOPC  FCB    $00       COPY OF CURRENT DUART OPCR REGISTER
ADED 00           ARTOPX  FCB    $00       COPY OF CURRENT DUART OPX SETTINGS
ADEE 00           BDRIVE  FCB    0         DRIVE NUMBER TO BOOT FROM
ADEF 00           DBLSTP  FCB    0         DOUBLE-STEPPING CODE (0=NO, NON-ZERO=YES)
ADF0 C07F         STACK   FDB    $C07F     TOP-OF-STACK ADDRESS
ADF2 0C           TABSIZ  FCB    12        # OF ENTRIES IN CMDTAB
ADF3 0000         JMPADDR FDB    0         TRANSFER ADDRESS OF USER PROGRAM
ADF5 00           ICLSW   FCB    0         SWITCH: $00-USE IPORT, NOT $00- USE IPORTL
ADF6 00           OCLSW   FCB    0         SWITCH: $00- USE OPORT, NOT $00- USE OPORTA
ADF7 FF20         IPORT   FDB    CPORT     PORT FOR NORMAL INPUT
ADF9 FF20         IPORTL  FDB    CPORT     PORT FOR INPUT FOR "L" COMMAND
ADFB FF20         OPORT   FDB    CPORT     PORT FOR NORMAL OUTPUT
ADFD FF20         OPORTA  FDB    CPORT     PORT FOR ALTERNATE OUTPUT
ADFF FF           UCFLAG  FCB    $FF       FLAG RE CONVERT TO UPPERCASE (0=NO, NON-ZERO=Y
AE00 FF           ECHOFL  FCB    $FF       NON-ZERO = ECHO RECD CHARAC, ZERO = DON'T
AE01 0000         OFFSET  FDB    0         ADDRESS OFFSET FOR "L" COMMAND
AE03 30           ARTACR  FCB    $30       COPY OF CURRENT DUART ACR
AE04 00           ARTINT  FCB    $00       COPY OF CURRENT DUART IMR
AE05 00           VIAINT  FCB    $00       COPY OF CURRENT VIA INTERRUPT ENABLE FLAGS
AE06 FF           RESETF  FCB    $FF       RESET FLAG (0=STMON ENTERED FROM FLEX,
                  *                         NON-ZERO=STMON ENTERED VIA RESET)
AE07 FF           ARTBAU  FCB    $FF       PORT A BAUD RATE CODE (SET BY RESET INIT)
AE08 AE08         HERE    FDB    *         "HERE"
            FEBD  M       SET    *+MEND-MONEND
                  *
                  * ADDRESSES OF USER CALLABLE ROUTINES
                  *
AE0A F85E         ADRTAB  FDB    BEGFLX+MEND-MONEND
AE0C FC59                 FDB    CMDTAB+MEND-MONEND

AE0E FA3B                 FDB    LOAD+MEND-MONEND
AE10 FC7D                 FDB    SIGNON+MEND-MONEND
AE12 FE58                 FDB    INCHEK+MEND-MONEND
AE14 FDCE                 FDB    INCH8+MEND-MONEND
AE16 FDFA                 FDB    INCHN+MEND-MONEND
AE18 FE15                 FDB    INCHE+MEND-MONEND
AE1A FE0F                 FDB    INCHTE+MEND-MONEND
AE1C FE39                 FDB    IN1HEX+MEND-MONEND
AE1E FE28                 FDB    IN2HEX+MEND-MONEND
AE20 FE1A                 FDB    IN4HEX+MEND-MONEND
AE22 FD8A                 FDB    OUTCH+MEND-MONEND (avoid using)
AE24 FD8A                 FDB    OUTCH+MEND-MONEND
AE26 FDC6                 FDB    OUT1SP+MEND-MONEND
AE28 FDBC                 FDB    OUT1HX+MEND-MONEND
AE2A FDB0                 FDB    OUT2HX+MEND-MONEND
AE2C FDA8                 FDB    OUT4HX+MEND-MONEND
AE2E FD7E                 FDB    PCRLF+MEND-MONEND
AE30 FD73                 FDB    PDATA+MEND-MONEND
AE32 FD71                 FDB    PSTRNG+MEND-MONEND
            AE34  ENDADR  EQU    *

            FEE7  M       SET    *+MEND-MONEND
                  *
                  * JUMP TABLE FOR INTERRUPTS
                  *
AE34 7E   FC7D    SWI3    JMP    SIGNON+MEND-MONEND
AE37 12                   NOP
AE38 7E   FC7D    SWI2    JMP    SIGNON+MEND-MONEND
AE3B 12                   NOP
AE3C 7E   FC7D    FIRQ    JMP    SIGNON+MEND-MONEND
AE3F 12                   NOP
```

```
AE40 7E   FC7D     IRQ     JMP    SIGNON+MEND-MONEND
AE43 12                    NOP
AE44 7E   FC7D     SWI     JMP    SIGNON+MEND-MONEND
AE47 12                    NOP
AE48 7E   FC7D     NMI     JMP    SIGNON+MEND-MONEND
AE4B 12                    NOP
AE4C FF                    FCB    $FF       RESERVED TO PREVENT UNINTENTIONAL ACCESS TO $FF00 I/O
            FEFF  M        SET    *+MEND-MONEND-1

            AE4C  MONEND   EQU    *-1       END OF MONITOR THAT IS COPIED TO RAM



                  ************************************************
                  *
                  * INITIALIZE SYSTEM AT POWERUP/RESET
                  *
                  ************************************************
            AE4D  INIT     EQU    *
                  *
                  * DELAY TO MAKE SURE SAM AND PERIPHERAL CHIPS ARE READY
                  *
AE4D 8E   FFFF             LDX    #$FFFF    3/4 SEC.
AE50 30   82       IT05    LEAX   ,-X       (6)
AE52 12                    NOP               (2)
AE53 26   FB               BNE    IT05      (3)
                  *
                  * INITIALIZE SAM CHIP
                  *
AE55 B7   FFDD             STA    $FFDD     SELECT 64K RAM FOR SAM.
                  *
                  * DELAY AGAIN TO MAKE SURE SAM AND RAM ARE READY
                  *
AE58 8E   2B00             LDX    #$2B00    1/10 SEC.
AE5B 30   82       IT10    LEAX   ,-X       (6)
AE5D 26   FC               BNE    IT10      (3)
                  *
                  * DISABLE RAM BOARD #2 (IF PRESENT), AND SWITCH IN
                  *  BANK 0 OF RAM BOARD #1 (IF PRESENT)
                  *
AE5F 7F   FFBC             CLR    RAMBD2
AE62 86   80               LDA    #$80
AE64 B7   FFB8             STA    RAMBD1
                  *
                  * INITIALIZE COMMON AREAS OF 2681 PORTS A AND B
                  *  (CF ARTINT, ARTACR INITIALIZED BELOW)
                  *
AE67 8E   FF20             LDX    #CPORT

AE6A 4F                    CLRA              NO INTERRUPTS
AE6B A7   05               STA    5,X       (IMR)
AE6D 86   0A               LDA    #$0A      DEACTIVATE BOTH RECEIVERS AND TRANSMITTERS
AE6F A7   02               STA    2,X       (CRA)
AE71 A7   0A               STA    10,X      (CRB)
AE73 A6   0F               LDA    15,X      STOP COUNTER/TIMER
AE75 4F                    CLRA              RESET COUNT TO ZERO
AE76 A7   06               STA    6,X       (CTUR)
AE78 A7   07               STA    7,X       (CTLR)
AE7A 86   B0               LDA    #$B0      SET TO COUNTER MODE WITH CRYSTAL/16 CLOCK
AE7C A7   04               STA    4,X       (ACR)   BAUD RATE SET #2, NO CHANGE OF STATE INTERRUPTS
AE7E A6   0F               LDA    15,X      STOP COUNTER
```

```
AE80 86   FF              LDA    #$FF      SET RTSA AND ALL OTHER OUTPUT BITS LOW
AE82 A7   0E              STA    14,X      (SET BITS IN OUTPUT REGISTER HIGH - OUTPUT LINES LOW)
AE84 4F                   CLRA             CONFIG ALL OUTPUT LINES TO GENERAL PURPOSE
AE85 A7   0D              STA    13,X      (OPCR)
                *
                * INITIALIZE PORT A
                *
AE87 B6   A000            LDA    BAUDRA    BAUD RATE FOR RX & TX
AE8A A7   01              STA    1,X       (CSRA) (leave in to make sure IP3/IP4 work)
AE8C 81   FF              CMPA   #BIPX     TAKE VALUE FROM IP3/IP4 ?
AE8E 26   1A              BNE    IT21      .N, USE VALUE IN BAUDRA
AE90 E6   0D              LDB    13,X      READ IP4/IP3 VALUE
AE92 C4   18              ANDB   #$18
AE94 86   BB              LDA    #B9600    9600
AE96 C1   00              CMPB   #$00
AE98 27   0E              BEQ    IT20
AE9A 86   CC              LDA    #B19200   19200
AE9C C1   08              CMPB   #$08
AE9E 27   08              BEQ    IT20
AEA0 86   44              LDA    #B300     300
AEA2 C1   10              CMPB   #$10
AEA4 27   02              BEQ    IT20
AEA6 86   66              LDA    #B1200    1200
AEA8 A7   01       IT20   STA    1,X
AEAA 1F   8B       IT21   TFR    A,DP      save baud rate code in DP register

AEAC 86   10              LDA    #$10      RESET MR POINTER
AEAE A7   02              STA    2,X       (CRA)
AEB0 B6   A002            LDA    WRDLEN    word length code
AEB3 81   FF              CMPA   #WIP5     take value from IP5?
AEB5 26   09              BNE    IT24      .n, use value in WRDLEN
AEB7 4F                   CLRA             7 data bits
AEB8 E6   0D              LDB    13,X      read IP5 value
AEBA C4   20              ANDB   #$20
AEBC 27   02              BEQ    IT24
AEBE 86   01              LDA    #$01      8 data bits
AEC0 84   01       IT24   ANDA   #$01      use only bit 0
AEC2 8A   12              ORA    #$12      NO PARITY, 7 OR 8 DATA BITS, CHARACTER ERROR MODE,
AEC4 A7   84              STA    0,X       (MR1A)   NO RX RTS CONTROL
AEC6 86   1F              LDA    #$1F      2 STOP BITS, NORMAL CHANNEL MODE, NO TX RTS CONTROL
AEC8 A7   84              STA    0,X       (MR2A)   CTS ENABLES TX
                *
                * INITIALIZE PORT B
                *
AECA B6   A001            LDA    BAUDRB    BAUD RATE FOR RX & TX
AECD A7   09              STA    9,X       (CSRB)
AECF 86   10              LDA    #$10      RESET MR POINTER
AED1 A7   0A              STA    10,X      (CRB)
AED3 86   13              LDA    #$13      NO PARITY, 8 DATA BITS, CHARACTER ERROR MODE,
AED5 A7   08              STA    8,X       (MR1B)   NO RX RTS CONTROL
AED7 86   1F              LDA    #$1F      2 STOP BITS, NORMAL CHANNEL MODE, NO TX RTS CONTROL
AED9 A7   08              STA    8,X       (MR2B)   CTS ENABLES TX
                *
                * RESET AND ACTIVATE BOTH 2681 PORTS
                *
AEDB 86   70              LDA    #$70      STOP BREAK
AEDD A7   02              STA    2,X       (CRA)
AEDF A7   0A              STA    10,X      (CRB)
AEE1 86   50              LDA    #$50      RESET BREAK INTERRUPT
AEE3 A7   02              STA    2,X       (CRA)
AEE5 A7   0A              STA    10,X      (CRB)
```

```
AEE7 86   30              LDA    #$30      RESET TRANSMITTER
AEE9 A7   02              STA    2,X       (CRA)
AEEB A7   0A              STA    10,X      (CRB)
AEED 86   20              LDA    #$20      RESET RECEIVER
AEEF A7   02              STA    2,X       (CRA)
AEF1 A7   0A              STA    10,X      (CRB)
AEF3 86   40              LDA    #$40      RESET ERROR STATUS
AEF5 A7   02              STA    2,X       (CRA)
AEF7 A7   0A              STA    10,X      (CRB)

AEF9 86   05              LDA    #$05      ACTIVATE BOTH RECEIVERS AND TRANSMITTERS
AEFB A7   02              STA    2,X       (CRA)
AEFD A7   0A              STA    10,X      (CRB)
                  *
                  * EXECUTE MEMORY TEST SUBROUTINES
                  *  (WITHOUT USING ANY RAM TO EXECUTE THEM)
                  *
AEFF 10CE AF5E            LDS    #ROMTEST  TEST ROM
AF03 1E   45              EXG    S,PC

AF05 10CE AF83            LDS    #RAMTEST  TEST RAM AND SET TO PAGE 1
AF09 1E   45              EXG    S,PC

                  *
                  * PROMPT FOR COLD VS WARM START RE MONITOR
                  *
AF0B 86   43              LDA    #'C
AF0D CE   AFC4            LDU    #COUT
AF10 1E   35              EXG    U,PC
AF12 86   57              LDA    #'W
AF14 CE   AFC4            LDU    #COUT
AF17 1E   35              EXG    U,PC
AF19 86   3F       IT30   LDA    #'?
AF1B CE   AFC4            LDU    #COUT
AF1E 1E   35              EXG    U,PC
AF20 CE   AFD2            LDU    #CIN      GET RESPONSE
AF23 1E   35              EXG    U,PC
AF25 81   57              CMPA   #'W
AF27 27   14              BEQ    IT50
AF29 81   43              CMPA   #'C
AF2B 26   EC              BNE    IT30
                  *
                  * COPY MONITOR FROM ROM TO HIGH END OF RAM (PAGE 1)
                  *  AND INIT VARIABLES
                  *
AF2D 8E   AE4D            LDX    #MONEND+1
AF30 108E 7F00            LDY    #MEND+1-$8000
AF34 A6   82       IT40   LDA    ,-X
AF36 A7   A2              STA    ,-Y
AF38 8C   A44A            CMPX   #MONBEG
AF3B 22   F7              BHI    IT40
                  *
                  * (CF. 2681 INITIALIZED ABOVE)
                  *  (DON'T RELY ON DEFAULTS IN VARIABLE SECTION)
                  *
AF3D 86   B0       IT50   LDA    #$B0
AF3F B7   7EB6            STA    ARTACR+MEND-MONEND-$8000
AF42 B7   7EB9            STA    RESETF+MEND-MONEND-$8000 SET RESET FLAG
AF45 4F                   CLRA
AF46 B7   7EB7            STA    ARTINT+MEND-MONEND-$8000
AF49 B7   7EB8            STA    VIAINT+MEND-MONEND-$8000
```

```
AF4C B7   7E9F            STA    ARTOPC+MEND-MONEND-$8000
AF4F B7   7EA0            STA    ARTOPX+MEND-MONEND-$8000
AF52 1E   8B              EXG    A,DP      (retrieve saved baud rate code)
AF54 B7   7EBA            STA    ARTBAU+MEND-MONEND-$8000
                  *
                  * JUMP TO RAM COPY OF ST-MON
                  *
AF57 33   8D 4D22         LEAU   (SIGNON+MEND-MONEND),PCR
AF5B 7E   FFE5            JMP    RAMMAP+$FFFF-EROM


                  ************************
                  * READ-ONLY MEMORY TEST
                  *  (CALLED VIA LDS #ROMTEST;  EXG S,PC)
                  * OUT - DP UNCHANGED
                  ************************
AF5E 108E A003    ROMTEST LDY    #SROM+3   START OF ROM (BYPASS 1ST 3 BYTES)
AF62 8E   5555            LDX    #$5555    INIT CHECKSUM

AF65 E6   A0      RM10    LDB    ,Y+
AF67 3A                   ABX              ACCUM CHECKSUM
AF68 108C AFEE            CMPY   #(EROM-17) DONE?
AF6C 26   F7              BNE    RM10

AF6E BC   AFEE            CMPX   CHKSUM    CHECKSUMS IDENTICAL?
AF71 27   0E              BEQ    RM190     .Y

AF73 86   4D      RMERR1  LDA    #'M       .N, OUTPUT ERROR MESSAGE "M1"
AF75 CE   AFC4            LDU    #COUT
AF78 1E   35              EXG    U,PC
AF7A 86   31              LDA    #'1
AF7C CE   AFC4            LDU    #COUT
AF7F 1E   35              EXG    U,PC

AF81 1E   45      RM190   EXG    S,PC      RETURN

                  ***************************
                  * READ/WRITE-MEMORY TEST
                  *  (NON-DESTRUCTIVE)
                  * TEST BOTH 32K BANKS, AND END IN PAGE 1
                  * (CALLED VIA LDS #RAMTEST;  EXG S,PC)
                  * OUT - DP UNCHANGED
                  * ERROR - PAGE # IN $C000
                  *         ADDRESS (WITHIN 32K PAGE) IN $C001/$C002
                  ***************************
             AF83 RAMTEST EQU    *
                  *
                  * COMPLEMENT CONTENTS OF RAM LOCATION
                  *  AND READ BACK TO SEE IF STORED O.K.
                  *
AF83 B7   FFD4            STA    PAGE0     START WITH PAGE 0
AF86 5F                   CLRB
AF87 8E   0000    RA05    LDX    #SRAM     START OF RAM

AF8A A6   84      RA10    LDA    0,X       COMPLEMENT CONTENTS
AF8C 43                   COMA
AF8D A7   84              STA    0,X
AF8F A1   84              CMPA   0,X       MATCHES?
AF91 26   18              BNE    RMERR2    .ERROR
AF93 43                   COMA             .Y, RESTORE
AF94 A7   84              STA    0,X
```

```
AF96 A1   84              CMPA   0,X
AF98 26   11              BNE    RMERR2
AF9A 30   01              LEAX   1,X
AF9C 8C   8000            CMPX   #ERAM+1   END OF RAM
AF9F 26   E9              BNE    RA10

AFA1 C1   01              CMPB   #1        SECOND BANK DONE?
AFA3 27   1D              BEQ    RA90      .Y, ALL O.K.
AFA5 B7   FFD5            STA    PAGE1
AFA8 5C                   INCB
AFA9 20   DC              BRA    RA05
                  *
                  * ERROR MESSAGE
                  *
AFAB B7   FFD5    RMERR2  STA    PAGE1
AFAE F7   4000            STB    $C000-$8000 SAVE PAGE #
AFB1 BF   4001            STX    $C001-$8000 SAVE ADDRESS
AFB4 86   4D              LDA    #'M       OUTPUT ERROR MESSAGE "M2"
AFB6 CE   AFC4            LDU    #COUT
AFB9 1E   35              EXG    U,PC
AFBB 86   32              LDA    #'2
AFBD CE   AFC4            LDU    #COUT
AFC0 1E   35              EXG    U,PC

AFC2 1E   45      RA90    EXG    S,PC      RETURN

                  ******************
                  * OUTPUT CHARACTER FOR MEMORY TEST ERROR ROUTINES
                  *  (CALLED VIA LDU #COUT;  EXG U,PC)
                  * OUT - DP UNCHANGED
                  ******************
AFC4 F6   FF21    COUT    LDB    CPORT+1   TEST STATUS
AFC7 C5   04              BITB   #$04
AFC9 27   F9              BEQ    COUT
AFCB 8A   80              ORA    #$80
AFCD B7   FF23            STA    CPORT+3   OUTPUT CHARACTER
AFD0 1E   35              EXG    U,PC      RETURN

                  ******************
                  * INPUT CHARACTER FOR INIT ROUTINE
                  *  (CALLED VIA LDU #CIN;  EXG U,PC)
                  * OUT - DP UNCHANGED
                  ******************
AFD2 F6   FF21    CIN     LDB    CPORT+1   TEST STATUS
AFD5 C5   01              BITB   #$01
AFD7 27   F9              BEQ    CIN
AFD9 B6   FF23            LDA    CPORT+3
AFDC 84   5F              ANDA   #$5F
AFDE 1E   35              EXG    U,PC

                  ****************************
                  * STEPPING STONE CODE TO SWITCH MAP
                  *  (ALSO DECODED TO $FFEX)
                  *
                  ****************************
AFE0                      ORG    EROM-$1F
AFE0 B7   FFDE    ROMMAP  STA    MAP0
AFE3 6E   C4              JMP    0,U

AFE5 B7   FFDF    RAMMAP  STA    MAP1
AFE8 6E   C4              JMP    0,U
```

```
  AFEA FF FF FF FF           FCB    $FF,$FF,$FF,$FF (spare)

                     * LIB STMONC.TXT.1
                     ***********************************
                     * CHECKSUM
                     ***********************************
  AFEE 07BD          CHKSUM  FDB    $07BD     ROM CHECKSUM
                     *   END STMONC.TXT

                     *****************************
                     *    INTERRUPT VECTORS
                     * (THIS TABLE ALSO DECODED TO $FFF0)
                     *
                     *****************************
  AFF0 FFFF                  FDB    $FFFF     (RESERVED BY MOTOROLA)
  AFF2 FEE7                  FDB    SWI3+MEND-MONEND
  AFF4 FEEB                  FDB    SWI2+MEND-MONEND
  AFF6 FEEF                  FDB    FIRQ+MEND-MONEND
  AFF8 FEF3                  FDB    IRQ+MEND-MONEND
  AFFA FEF7                  FDB    SWI+MEND-MONEND
  AFFC FEFB                  FDB    NMI+MEND-MONEND
  AFFE AE4D                  FDB    INIT

                             END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ADRTAB AE0A   ARTACR AE03   ARTBAU AE07   ARTINT AE04   ARTOPC ADEC
ARTOPX ADED   B1200  0066   B19200 00CC   B300   0044   B9600  00BB
BAUDRA A000   BAUDRB A001   BDRIVE ADEE   BEGDSK A44A   BEGFLX A7AB
BELL   0007   BINR   A829   BIPX   00FF   BKSPC  0008   BLKMOV AA0F
BM20   AA2A   BM30   AA34   BM80   AA38   BM90   AA3A   BRA    0020
BTBUFR 2000   BTSIZE A79F   BT_COS A7A8   BT_FMT A7A1   BT_SPC A7A6
BT_SPT A7A2   BT_TOS A7A4   BUSYW  A90B   CHKSUM AFEE   CIN    AFD2
CMDTAB ABA6   COCO   A7AA   COMREG FF00   COUT   AFC4   CPORT  FF20
CR     000D   CRLF   ADE3   DATREG FF03   DB10   A7CE   DB50   A7D5
DB80   A7DC   DBLSTP ADEF   DD_BSZ 0018   DD_BT  0015   DD_DIR 0008
DD_FMT 0010   DD_SPT 0011   DD_TKS 0003   DELAY  A914   DENS   A7AE
DENSFL A7AC   DISKBT A7BE   DLY100 A954   DLY101 A957   DRVREG FF2E
DSKVAR A7AB   DY10   A917   DY20   A91A   EB50   AAC2   EB90   AAD5
EB95   AAD7   ECHOFL AE00   EL20   AAEA   EL40   AB02   EL60   AB0C
EL65   AB0E   ENDADR AE34   ENDDSK A988   EOT    0004   ERAM   7FFF
EROM   AFFF   ERR    AC18   ESC    001B   EXAMB  AAA0   EXAML  AAD8
EXCMD  A912   EXCMDW A909   FA20   AA02   FA30   AA0A   FA80   AA0C
FA90   AA0E   FARTAC FEDA   FARTBA FED8   FARTIN FEDB   FARTLE FED9
FARTOC FEDC   FARTOX FEDD   FBDRIV FED4   FBOOTF FEE6   FBTADD FEE2
FBTSIZ FEE4   FD90   A669   FDBLST FED5   FD_SEG 0010   FD_SIZ 0009
FF20   A671   FF30   A673   FF40   A68C   FF80   A698   FF90   A699
FI20   A566   FI30   A577   FI40   A58D   FI50   A58F   FI70   A593
FI80   A596   FI90   A597   FILL   A9E7   FILLER A003   FILLF  A4E9
FILLR  A50A   FIND   A551   FINDDT A65A   FINDFL A66B   FIRQ   AE3C
FLXRTN AC9A   FOFFSE FED0   FR01   A4EC   FR05   A510   FR07   A51A
FR10   A526   FR60   A542   FX40   ACAA   FX70   ACB1   FX80   ACB9
GA90   A859   GB10   A85A   GB20   A865   GB90   A871   GETA   A84F
GETB   A85C   GETSEC A888   GK40   A47A   GK80   A4E5   GS10   A890
GS20   A894   GS40   A8A2   GS50   A8A6   GS70   A8B8   GS80   A8BB
GS90   A8BC   HERE   AE08   HEXERR ADA2   IC05   ADB1   IC50   ADB4
ICLSW  ADF5   IHAF   AD93   IN05   AD29   IN190  AD5B   IN1HEX AD86
```

```
IN2HEX AD75   IN4HEX AD67   IN5O   AD2D   IN55   AD3F   IN6O   AD43
IN9O   AD45   INCH8  AD1B   INCHE  AD62   INCHEK ADA5   INCHN  AD47
INCHTE AD5C   INIT   AE4D   IPORT  ADF7   IPORTL ADF9   IRQ    AE4O
IT05   AE5O   IT1O   AE5B   IT2O   AEA8   IT21   AEAA   IT24   AECO
IT3O   AF19   IT4O   AF34   IT5O   AF3D   JM9O   ABA5   JMPADD ADF3
JMPKER A473   JUMP   AB9E   KERBUF 8OOO   KERNAM A78E   LB1O   A5AD
LB3O   A5E3   LB4O   A5EE   LB6O   A612   LB7O   A639   LB75   A651
LB8O   A654   LB9O   A657   LDO5   A99O   LD1O   A993   LD2O   A9CO
LD3O   A9D6   LD9O   A9DE   LDERR  A9D5   LF     OOOA   LF1O   A813
LF4O   A838   LF8O   A847   LF9O   A84E   LOAD   A988   LOADFX A7E5
LOADO9 A598   LP1O   A6F6   LP15   A6F9   LP2O   A7O7   LP3O   A716
LP4O   A725   LP5O   A732   LP56   A741   LP58   A74A   LP6O   A74D
LP66   A75D   LP68   A76O   LP8O   A769   LTOP   A6DF   M      FEFF
MAPO   FFDE   MAP1   FFDF   MC1O   AC2B   MC2O   AC48   MC25   AC4A
MC4O   AC79   MC5O   AC84   MC8O   AC8C   MC85   AC9O   MC87   AC92
MC9O   AC97   MEMCHG AC26   MEMTST AB18   MEND   FEFF   MN5O   ACOF
MN8O   AC1D   MONBEG A44A   MONEND AE4C   MOTOR  A95C   MOVMON A44A
MR1O   A96C   MSG1   A7AF   MSG2   A7B7   MSG3   A76B   MSG4   A781
MT1O   AB37   MT2O   AB48   MT25   AB49   MT3O   AB51   MT35   AB52
MT4O   AB5B   MT5O   AB7C   MT89   AB98   MT9O   AB9B   MV2O   A451
MV4O   A45D   NEWINI A471   NMI    AE48   NOS9   A797   NOS9P1 A79A
NPORT  FF28   NUMSEC A7AB   NXTCMD ABD9   OCLSW  ADF6   OFFSET AEO1
OPORT  ADFB   OPORTA ADFD   OTO5   ACE5   OT19O  AD11   OT5O   ACE9
OUT1HX ADO9   OUT1SP AD13   OUT2HX ACFD   OUT4HX ACF5   OUTCH  ACD7
OUTQM  ACD5   PAGEO  FFD4   PAGE1  FFD5   PCRLF  ACCB   PDATA  ACCO
PROMPT ADE1   PS2O   ACCA   PSTRNG ACBE   RAO5   AF87   RA1O   AF8A
RA9O   AFC2   RAMBD1 FFB8   RAMBD2 FFBC   RAMMAP AFE5   RAMTES AF83
RC1O   A6A1   RC2O   A6AC   RC6O   A6C5   RC65   A6CA   RC9O   A6DO
READCF A69D   READLS A6D4   READPS A875   READS  A91C   RECALB A8F7
RESETF AEO6   RESTOR A9O7   RM1O   AF65   RM19O  AF81   RMERR1 AF73
RMERR2 AFAB   ROMMAP AFEO   ROMTES AF5E   RP2O   A882   RP6O   A888
RS1O   A92C   RS3O   A932   RS4O   A936   RS7O   A94A   RS9O   A952
SEARCH AA3B   SECBUF BOOO   SECREG FFO2   SEEK   A8CO   SETOPR A973
SH5O   AA54   SH7O   AA7C   SH9O   AA85   SH99   AA87   SIDE   A7AD
SIGNON ABCA   SK1O   A8C7   SK2O   A8D8   SK3O   A8DE   SK9O   A8F5
SRAM   OOOO   SROM   AOOO   STACK  ADFO   STATRG FFOO   SU8O   AA9E
SU9O   AA9F   SUSPEN AA88   SWI    AE44   SWI2   AE38   SWI3   AE34
TABSIZ ADF2   TITLE  ADBA   TRKREG FFO1   UCFLAG ADFF   VIAINT AEO5
VRETUR D3E1   WARMS  CDO3   WIP5   OOFF   WRDLEN AOO2
```