START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

```
K    K   SSSS  PPPP    RRRR     OOO    U    U  L
K   K   S      P   P   R   R   O   O   U    U  L
K  K    S      P   P   R   R   O   O   U    U  L
KKK      SSS   PPPP    RRRR    O   O   U    U  L
K  K        S  P       R R     O   O   U    U  L
K   K       S  P       R  R    O   O   U    U  L
K    K  SSSS   P       R   R    OOO    UUUUU   LLLLL
```

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

```
N   N                               SSSS   999    1    OOO           H   H   OOO   L      EEEEE  SSSS
N   N              t               S      9   9  11   O   O          H   H  O   O  L      E     S
NN  N   ooo   ttt      eee    ##   S      9   9   1   O  OO          H   H  O   O  L      E     S
N N N  o   o   t      e   e   ##    SSS    9999   1   O O O   -----  HHHHH  O   O  L      EEEE   SSS
N  NN  o   o   t      eeee              S     9   1   OO  O          H   H  O   O  L      E         S
N   N  o   o   t      e       ##        S     9   1   O   O          H   H  O   O  L      E         S
N   N   ooo     tt     eeee   ##   SSSS    999   111   OOO           H   H   OOO   LLLLL  EEEEE  SSSS
```

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

START Job KSMON Req #911 for KSPROUL   Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*

****

*

* * * * * * * N O T I C E * * * * * *

*

* ATTENTION!! NEW BIN NUMBERS AT HILL CENTER!

* -STARTS AUGUST 29- CHECK CAMPUS MAIL

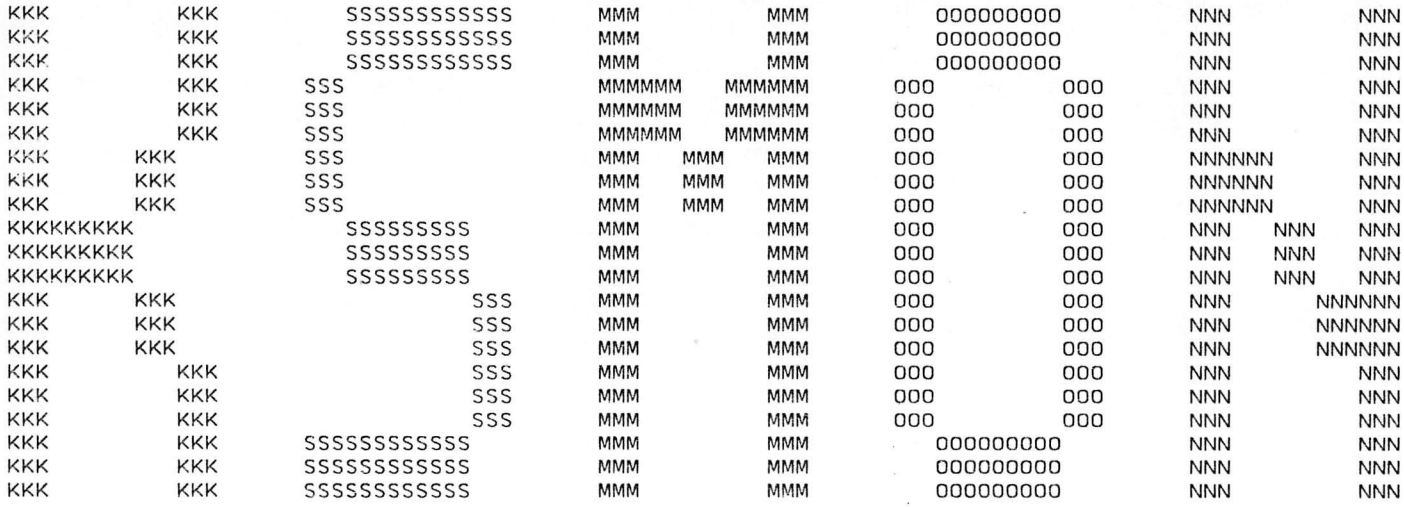
* FOR DESCRIPTIVE MEMO OR PICK ONE UP AT HILL

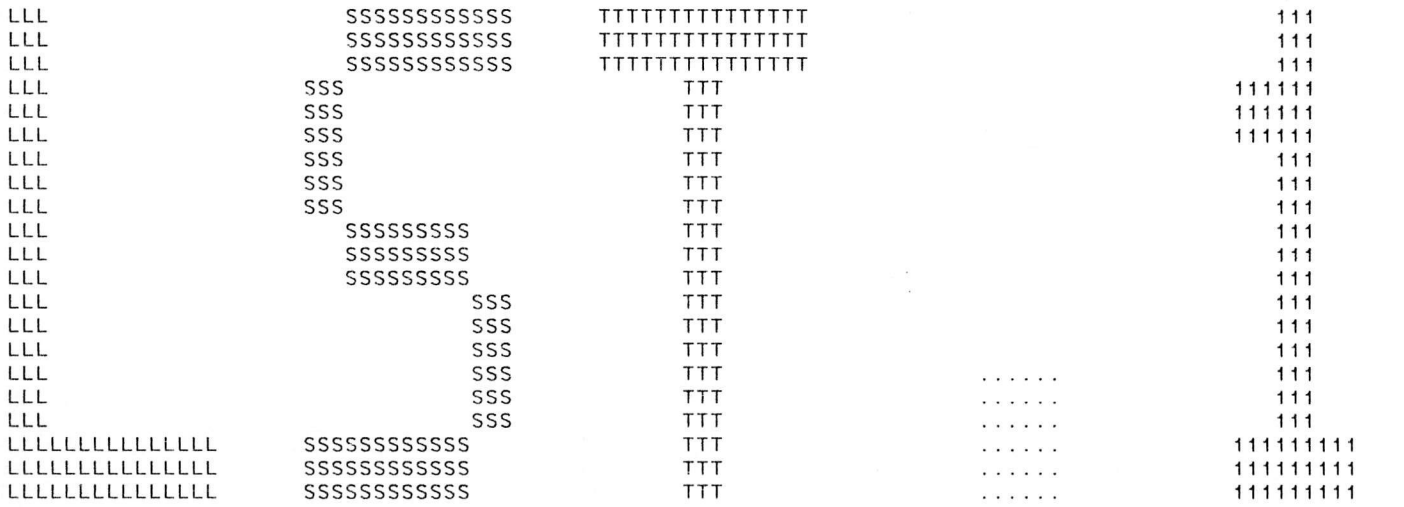
* I/O COUNTER OR AID STATION-

* RESET BIN IN WYLBUR TO CONFORM!

*

****





START Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:46:59 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit *START*
File PS:<KSPROUL>KSMON.LST.1, created: 11-Sep-82 20:30:06, printed: 11-Sep-82 20:47:14
Job parameters: Request created:11-Sep-82 20:31:18 Page limit:180 Forms:TN Account:OPS Note:S910-HOLES
File parameters: Copy: 1 of 1 Spacing:SINGLE File format:ASCII Print mode:ASCII /DELETE

Table of contents

```
                        .SBTTL  Zero Page Storage & Equates
                        .TITLE K-Sproul Monitor for the 6502

                        ;   by  Keith A. Sproul                          Copyright:
                        ;       1368 Noah Road                          November 1978, 1980
                        ;       North Brunswick, N.J. 08902             Major Revision October 1980
                        ;       (201) 246-3749

                        ;       28-MAY-81       REMOVED KIM FROM SYSTEM
                        ;       05-JUN-81       REMOVED CASSETTE LOAD/SAVE ROUTINES FROM MONITOR
                        ;       04-SEP-81       ADDED DISK STATUS COMMAND
                        ;       23-DEC-81       CHANGED DISASM CONTINUE

                        ; a
                        ; B     BLKCMP                  ; BLOCK COMPARE
                        ; c
                        ; D     DBOOT                   ; DISK BOOT
                        ; E     ENABLE                  ; ENABLE WRITE PROTECTED MEMORY
                        ; F     FILLD                   ; BLOCK FILL with a CONSTANT
                        ; g
                        ; H     HEXDMP                  ; HEX & ASCII DUMP (STANDARD HEX DUMP)
                        ; I     INSERT                  ; INSERT ASCII TEXT
                        ; J     JUMP                    ; JUMP (ADDR1)
                        ; k
                        ; l
                        ; M     MEMORY                  ; EXAMINE & CHANGE MEMORY
                        ; n
                        ; o
                        ; P     PROTCT                  ; PROTECT SYSTEM RAM
                        ; Q     QUERRY                  ; QUERRY DISK-STATUS
                        ; R     RSTINI                  ; INITIALIZE THE NMI/IRQ/BRK VECTORS
                        ; S     SEARCH                  ; VARIABLE BYTE SEARCH
                        ; t
                        ; u
                        ; V     MVDATA                  ; BLOCK MOVE
                        ; W     CODOS                   ; CODOS WARM START
                        ; x
                        ; y
                        ; Z     DISASM                  ; DISASSEMBLER
                        ;
                        ; ?     MEMTST                  ; MEMORY TEST (NON DESTRUCTIVE)
                        ; !     MTEST                   ; MEMORY TEST (DESTRUCTIVE)
```

```
                                ;
                                ; SDTXT Z-PAGE STORAGE
                                ; CODOS SCRATCH RAM AREA
                                ;
00C0 0001  LMNEM:   .BLKB   1   ; LEFT HALF of MNEMONIC
00C1 0001  RMNEM:   .BLKB   1   ; RIGHT HALF of MNEMONIC
00C2 0001  FORMAT:  .BLKB   1   ; PRINT FORMAT
00C3 0001  LENGTH:  .BLKB   1   ; BYTE LENGTH
                                ;
                                ;
                                ;
00F0 0001  ADDR1:   .BLKW   1   ; ADDR1 MAIN TEMPORARY ADDRESS
00F2 0001  ADDR2:   .BLKW   1   ; USED TO BE POINTL,POINTH
00F4 0001  ADDRE:   .BLKW   1   ; ADDRESS END ADDRESS
00F6 0001  COUNT:   .BLKB   1   ; COUNT
00F6       BUFFER   =       COUNT ; SEARCH BUFFER
                                ;
                                ; REG SAVE AREA IN PAGE ZERO
                                ;                           ; OLD KIM ADDRESSES
00F7 0001  PCL:     .BLKB   1   ; $00EF ;\
00F8 0001  PCH:     .BLKB   1   ; $00F0 ; \
00F9 0001  PREG:    .BLKB   1   ; $00F1 ;  \
00FA 0001  SPUSER:  .BLKB   1   ; $00F2 ;   > SAVED & PRINTED on NMI or IRQ/BRK
00FB 0001  ACC:     .BLKB   1   ; $00F3 ;  /
00FC 0001  YREG:    .BLKB   1   ; $00F4 ; /
00FD 0001  XREG:    .BLKB   1   ; $00F5 ;/
                                ;
00FE 0001  TEMP:    .BLKB   1   ; TEMP USED IN GET-BYTE ONLY!
00FF 0001  TMPX:    .BLKB   1   ; USED IN A FEW MONITOR COMMANDS
```

```
                        .SBTTL  M.T.U. System Equates

      8000              USRRAM  =       $8000
      C000              SYSRAM  =       $C000

                        .MCALL  DSKEQU, BNKEQU, K1012, IRQNMI
                        .LIST   ME
 0100                   DSKEQU
(1)                     ;*************** FLOPPY DISK CONTROLLER REGISTER ADDRESSES ********************
(1)
(1)     C603            CODOS   =       SYSRAM+$0603    ; CODOS WARM START ADDRESS
(1)     DF00            DSKIPL  =       SYSRAM+$1F00    ; DISK BOOT ENTRY POINT
(1)     DFE8            FDCIRQ  =       SYSRAM+$1FE8    ; K-1013 INTERRUPT REG.         (READ-ONLY)
(1)     DFE8            FDCHWC  =       SYSRAM+$1FE8    ; DMA MODE (B0) WRITE PROT (B1) (WRITE-ONLY)
(1)     DFEA            FDCDMA  =       SYSRAM+$1FEA    ; FDC DMA ADDRESS REGISTER
(1)     DFEE            FDCMSR  =       SYSRAM+$1FEE    ; FDC NEC-765 MAIN STATUS REGISTER
(1)     DFEF            FDCDR   =       SYSRAM+$1FEF    ; FDC NEC-765 DATA REGISTER
(1)
(1)                     ;*****************************************************************************
 0100                   BNKEQU
(1)                     ;&&&&&&&&&&&&&&& BANKER BOARD EQUATES & REGISTER ADDRESSES &&&&&&&&&&&&&&&&&&&
(1)
(1)     EF80            BASE    =       $EF80
(1)     EF80            VIA1    =       BASE+$0000      ; 6522 #1
(1)     EF90            VIA2    =       BASE+$0010      ; 6522 #1
(1)     EFA0            BNKREG  =       BASE+$0020      ; ENABLE REGISTER
(1)
(1)                     ;&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
 0100                   K1012
(1)     EF00            CTY     =       $EF00           ; ACIA #1       (CONSOLE)
(1)     EF04            PIA1A   =       $EF04           ; PIA   1 PORT  A $EF04 $EF05   INPUT
(1)     EF06            PIA1B   =       $EF06           ;       1       B $EF06 $EF07   OUTPUT
(1)     EF08            PIA2A   =       $EF08           ;       2       A $EF08 $EF09   INPUT   EPROM
(1)     EF0A            PIA2B   =       $EF0A           ;       2       B $EF0A $EF0B   OUTPUT  EPROM
(1)
(1)     EF10            TTY     =       $EF10           ; ACIA #2       (TERMINAL)
(1)     EF14            PIA3A   =       $EF14           ; PIA   1 PORT  A $EF14 $EF15   INPUT
(1)     EF16            PIA3B   =       $EF16           ;       1       B $EF16 $EF17   OUTPUT  PRINTER
(1)     EF18            PIA4A   =       $EF18           ;       2       A $EF18 $EF19   INPUT
(1)     EF1A            PIA4B   =       $EF1A           ;       2       B $EF1A $EF1B   OUTPUT
(1)
 0100                   IRQNMI
(1)     DDFA            NMIV    =       $DDFA           ; NON-MASKABLE INTERRUPT
(1)     DDFC            RSTV    =       $DDFC           ; RESET (UN-USED)
(1)     DDFE            IRQV    =       $DDFE           ; INTERRUPT REQUEST
                        .NLIST  ME
                        .NLIST  BEX
                        .NLIST  CND
```

```
                        ;       BTCMDS - BOOTSTRAP FDC COMMAND STRINGS

        0000                    .=      $0000

                        ; DATA TABLE FROM ROM GETS MOVED HERE DURING DISK BOOT

0000                    BTCMDS:                         ; DUMMY LABEL
0000 0004               BTSPC:  .BLKB   3+1             ; 3 BYTES IN "SPECIFY" COMMAND STRING
0004 0003               BTRCL:  .BLKB   2+1             ; 2 BYTES IN "RECALIBRATE"
0007 0002               BTSIS:  .BLKB   1+1             ; 1 BYTE IN "SENSE INT STAT" CMD
0009 000A               BTRC:   .BLKB   9+1             ; 9 BYTES IN "READ" COMMAND
0013 000A               BTCRS:  .BLKB   10              ; STO, ETC. (FDC RESULTT PHASE READOUT)
                        ;               ---
                        ;               29
```

```
111                             .SBTTL  Initialization & Command Parser
112
113         F000                        .=      $F000           ; RESET VECTOR
114                                                             ; Jump Table for Convienence
115  F000   4C   30   F0       RESET:   JMP     RESET1
116  F003   4C   9D   F3                JMP     CTYHLF          ; CTY INPUT HALF DUPLEX
117  F006   4C   97   F3                JMP     CTYIN           ; CONSOLE (CTY:) INPUT
118  F009   A9   20             CTYSPA: LDA     #$20            ; PRINT a <space> to the CTY:
119  F00B   EA                          NOP                     ; FALL THROUGH
120  F00C   4C   12   F4                JMP     CTYOUT          ; CONSOLE (CTY:) OUTPUT
121  F00F   4C   C3   F3                JMP     GETBYT
122  F012   4C   F8   F3                JMP     PRTBYT
123  F015   4C   B7   F0                JMP     CRLF            ; NEW LINE
124  F018   4C   4B   F4                JMP     PCNTRL          ; (^S)/(^Q) & (^C)
125  F01B   4C   00   00                JMP     0               ; (UN-USED)
126  F01E   4C   32   F4                JMP     TTYIN           ; SYSTEM (TTY:) INPUT
127  F021   4C   3E   F4                JMP     TTYOUT          ; SYSTEM (TTY:) OUTPUT
128  F024   4C   B4   F3                JMP     KEYDWN          ; CHECK IF KEY DOWN ON CONSOLE (CTY:)
129  F027   4C   00   00                JMP     0               ; PARALLEL KEYBOARD INPUT (UN-USED)
130  F02A   4C   26   F4                JMP     PRINTR          ; PARALLEL PRINTER OUTPUT
131  F02D   4C   00   00                JMP     0               ; (UN-USED)
132
133  F030   A9   03             RESET1: LDA     #$03            ; MASTER RESET for ACIA
134  F032   8D   00   EF                STA     CTY             ; CONSOLE
135  F035   8D   10   EF                STA     TTY             ; TERMINAL
136  F038   A9   11                     LDA     #$11            ; SET ACIAs TO:
137  F03A   8D   00   EF                STA     CTY             ; [8 DATA BITS, 2 STOP BITs, NO PARITY, /16]
138  F03D   8D   10   EF                STA     TTY             ; (DO ONLY ON RESET)
139  F040   A9   00                     LDA     #$00            ; ACCESS DATA DIRECTION REGISTER
140  F042   8D   17   EF                STA     PIA3B+1
141  F045   A9   FF                     LDA     #$FF            ; SET UP PIA #4 PORT-B for PRINTER OUTPUT
142  F047   8D   16   EF                STA     PIA3B+0         ; $EF1A
143  F04A   A9   2E                     LDA     #$2E            ; ACCESS DATA REG & SET UP FOR STROBE
144  F04C   8D   17   EF                STA     PIA3B+1         ; CENTRONICS PRINTER PORT       (OUTPUT)
145
146  F04F   A2   FF             RESET2: LDX     #$FF            ; INIT STACK POINTER
147  F051   9A                          TXS
148  F052   D8                          CLD                     ; CLEAR DECIMAL MODE (to be Safe)
149  F053   20   59   F0                JSR     CMD.LP          ; Command Loop
150  F056   4C   4F   F0                JMP     RESET2          ; Start Over
151
```

```
153
154  F059  20  AF  F0  CMD.LP: JSR   PROMPT
155  F05C  20  A9  F3           JSR   UPRCHR          ; GET COMMAND (UPPER CASE CHARACTER)
156  F05F  A2  00               LDX   #$00
157  F061  BC  7B  F0  LOOP:    LDY   TABLE,X         ; CHECK if END of TABLE
158  F064  F0  F3               BEQ   CMD.LP          ; IF END START OVER
159  F066  DD  7B  F0           CMP   TABLE,X         ; CMP ACC with CMD TABLE
160  F069  F0  05               BEQ   FOUND
161  F06B  E8                   INX
162  F06C  E8                   INX
163  F06D  E8                   INX
164  F06E  D0  F1               BNE   LOOP            ; WILL ALWAYS BRANCH
165
166  F070  BD  7D  F0  FOUND:   LDA   TABLE+2,X       ; GET ADDR HIGH
167  F073  48                   PHA
168  F074  BD  7C  F0           LDA   TABLE+1,X       ; GET ADDR LOW
169  F077  48                   PHA
170  F078  4C  09  F0           JMP   CTYSPA          ; CTYSPA's RTS RETURNS to COMMAND
171
```

```
                        .SBTTL  Command Table


F07B  4A          TABLE:  .BYTE   'J
F07C  20  F1              .ADDR   JUMP-1          ; JUMP (ADDR1)
F07E  56                  .BYTE   'V
F07F  26  F1              .ADDR   MVDATA-1        ; BLOCK MOVE
F081  46                  .BYTE   'F
F082  57  F1              .ADDR   FILLD-1         ; BLOCK FILL with a CONSTANT
F084  48                  .BYTE   'H
F085  73  F1              .ADDR   HEXDMP-1        ; HEX & ASCII DUMP (STANDARD HEX DUMP)
F087  4D                  .BYTE   'M
F088  E3  F1              .ADDR   MEMORY-1        ; EXAMINE & CHANGE MEMORY
F08A  49                  .BYTE   'I
F08B  35  F2              .ADDR   INSERT-1        ; INSERT ASCII TEXT
F08D  53                  .BYTE   'S
F08E  5F  F2              .ADDR   SEARCH-1        ; VARIABLE BYTE SEARCH
F090  3F                  .BYTE   '?
F091  AA  F2              .ADDR   MEMTST-1        ; MEMORY TEST (NON DESTRUCTIVE)
F093  21                  .BYTE   '!
F094  FB  F2              .ADDR   MTEST-1         ; MEMORY TEST (DESTRUCTIVE)
F096  42                  .BYTE   'B
F097  49  F3              .ADDR   BLKCMP-1        ; BLOCK COMPARE
F099  5A                  .BYTE   'Z
F09A  78  F4              .ADDR   DISASM-1        ; DISASSEMBLER
F09C  44                  .BYTE   'D
F09D  70  F6              .ADDR   DBOOT-1         ; DISK BOOT
F09F  51                  .BYTE   'Q
F0A0  29  F7              .ADDR   DSTAT-1         ; DISK STATUS
F0A2  45                  .BYTE   'E
F0A3  A1  F7              .ADDR   ENABLE-1        ; ENABLE WRITE PROTECTED MEMORY
F0A5  50                  .BYTE   'P
F0A6  AB  F7              .ADDR   PROTCT-1        ; PROTECT WRITE PROTECABLE MEMORY
F0A8  57                  .BYTE   'W
F0A9  B5  F7              .ADDR   JCODOS-1        ; CODOS WARM START
F0AB  52                  .BYTE   'R
F0AC  D4  FF              .ADDR   RSTINI-1        ; INITIALIZE THE NMI/IRQ/BRK VECTORS
F0AE  00                  .BYTE   0               ; END-GO BACK to START  (INVALID COMMAND)

```

```
                                .SBTTL  Subroutines

FOAF  20  B7  FO  PROMPT: JSR   CRLF          ; PRINT PROMPT
FOB2  A9  26              LDA   #'&
FOB4  4C  12  F4          JMP   CTYOUT        ; USE RTS FROM CTYOUT

                                              ; NEW LINE
FOB7  A9  OD      CRLF:   LDA   #$OD          ; <cr>
FOB9  20  12  F4          JSR   CTYOUT
FOBC  A9  OO              LDA   #$OO          ; <nul>
FOBE  20  12  F4          JSR   CTYOUT
FOC1  A9  OA              LDA   #$OA          ; <lf>
FOC3  20  12  F4          JSR   CTYOUT
FOC6  A9  OO              LDA   #$OO          ; <nul>
FOC8  4C  12  F4          JMP   CTYOUT        ; RTS X = X  Y = Y  A = O

FOCB  A9  24      G.ADDR: LDA   #'$           ; SUBROUTINE TO
FOCD  20  E7  FO          JSR   CHRHEX        ; PUT ADDRESS IN
FODO  85  F1              STA   ADDR1+1       ; ADDR1+1 (HIGH BYTE)
FOD2  20  C3  F3          JSR   GETBYT
FOD5  85  FO              STA   ADDR1+O       ; ADDR1 (LOW BYTE)
FOD7  60                  RTS                 ; RTS  X = X  Y = Y  A = ?

FOD8  A9  2D      ENDADR: LDA   #'-           ; SUBROUTINE TO
FODA  20  E7  FO          JSR   CHRHEX        ; PUT END ADDRESS
FODD  85  F5              STA   ADDRE+1       ; IN ADDRE & ADDRE+1
FODF  20  C3  F3          JSR   GETBYT
FOE2  85  F4              STA   ADDRE
FOE4  4C  O9  FO          JMP   CTYSPA        ; USE CTYOUT RTS

FOE7  20  12  F4  CHRHEX: JSR   CTYOUT        ; SUBROUTINE to
FOEA  4C  C3  F3          JMP   GETBYT        ; PRINT ACC & GET HEX BYTE

FOED  20  B7  FO  PRTADR: JSR   CRLF          ; NEW LINE
FOFO  A9  24              LDA   #'$
FOF2  20  12  F4          JSR   CTYOUT
FOF5  A5  F1              LDA   ADDR1+1       ; SUBROUTINE to
FOF7  20  F8  F3          JSR   PRTBYT        ; PRINT ADDRESS
FOFA  A5  FO              LDA   ADDR1+O       ; IN ADDR1
FOFC  4C  1F  F4          JMP   BYTSPA        ; USE BYTSPA RTS
```

```
255
256
257  F0FF  E6  F2        INCPT:  INC     ADDR2+0
258  F101  D0  02                BNE     .+4
259  F103  E6  F3                INC     ADDR2+1
260  F105  60                    RTS                     ; RTS X = X  Y = Y  A = A
261
262
263                              .MCALL  INCDEC          ; INCREMENT & DECREMENT ADDRESS
264  F106                        INCDEC  ADDR1
265
266  F118  A5  F0        FINISH: LDA     ADDR1+0         ; SUBR to TELL IF
267  F11A  C5  F4                CMP     ADDRE+0         ; AT END OF MEMORY BLOCK
268  F11C  A5  F1                LDA     ADDR1+1         ; RTS WITH CARRY SET
269  F11E  E5  F5                SBC     ADDRE+1         ; WHEN FINISHED
270  F120  60                    RTS                     ; RTS X = X  Y = Y  A = ?
271
272
```

```
                        .SBTTL  Commands

                        ; J                             ; J $XXXX
F121  20  CB  F0    JUMP:   JSR     G.ADDR
F124  6C  F0  00            JMP     (ADDR1)

                        ; V                             ; V $XXXX-XXXX to $XXXX
F127  20  CB  F0    MVDATA: JSR     G.ADDR              ; SUBROUTINE to
F12A  20  D8  F0            JSR     ENDADR              ; MOVE DATA FROM AREA - AREA
F12D  A9  74                LDA     #'t                 ; ADDR1 ADDRESS of START of SOURCE
F12F  20  12  F4            JSR     CTYOUT              ; ADDRE ADDRESS of  END  of SOURCE
F132  A9  6F                LDA     #'o
F134  20  12  F4            JSR     CTYOUT
F137  20  09  F0            JSR     CTYSPA
F13A  A9  24                LDA     #'$
F13C  20  E7  F0            JSR     CHRHEX
F13F  85  F3                STA     ADDR2+1             ; ADDR2  ADDRESS of DESTINATION
F141  20  C3  F3            JSR     GETBYT
F144  85  F2                STA     ADDR2+0
F146  A0  00                LDY     #$00                ; STRAIGHT INDIRECT ADDRESSING

F148  B1  F0    MVLOOP: LDA     (ADDR1),Y
F14A  91  F2            STA     (ADDR2),Y
F14C  20  FF  F0            JSR     INCPT
F14F  20  06  F1            JSR     INCADR
F152  20  18  F1            JSR     FINISH              ; CHECK if FINISHED
F155  90  F1                BCC     MVLOOP
F157  60                    RTS                         ; RTS X = X  Y = 0  A = ?

                        ; F                             ; F $XXXX-XXXX $XX
F158  20  CB  F0    FILLD:  JSR     G.ADDR              ; SUBROUTINE to
F15B  20  D8  F0            JSR     ENDADR              ; FILL AREA with CONSTANT
F15E  A9  24                LDA     #'$
F160  20  E7  F0            JSR     CHRHEX              ; GET BYTE
F163  48                    PHA                         ; SAVE BYTE
F164  A0  00                LDY     #$00                ; STRAIGHT INDIRECT ADDRESSING
F166  68            FILL:   PLA
F167  91  F0                STA     (ADDR1),Y
F169  48                    PHA                         ; SAVE BYTE
F16A  20  06  F1            JSR     INCADR
F16D  20  18  F1            JSR     FINISH              ; CHECK if FINISHED
F170  90  F4                BCC     FILL
F172  68                    PLA                         ; RESTORE STACK to ORIGINAL STATE
F173  60                    RTS                         ; RTS X = X  Y = 0 A = CONSTANT
```

```
                        .SBTTL  Commands - Hex Dump

                        ; H                             ; H #XX $XXXX
F174  A9  23            HEXDMP: LDA     #'#
F176  20  E7  F0                JSR     CHRHEX          ; GET # of BYTES/LINE
F179  85  FF                    STA     TMPX
F17B  20  09  F0                JSR     CTYSPA
F17E  20  CB  F0                JSR     G.ADDR
F181  A9  10            PAGE:   LDA     #16
F183  85  F6                    STA     COUNT           ; SET = 16 LINES/PAGE
F185  20  ED  F0        LINE:   JSR     PRTADR          ; PRINT CRLF, '$', ADDR & SPA
F188  A0  00                    LDY     #$00
F18A  B1  F0            HEXDAT: LDA     (ADDR1),Y
F18C  20  1F  F4                JSR     BYTSPA          ; PRINT BYTE & SPACE
F18F  C8                        INY
F190  C4  FF                    CPY     TMPX            ; CPY TMPX (# OF BYTES/LINES)
F192  D0  F6                    BNE     HEXDAT          ; END of HEX
F194  20  09  F0                JSR     CTYSPA
F197  A9  28                    LDA     #'(
F199  20  12  F4                JSR     CTYOUT
F19C  A0  00                    LDY     #$00
F19E  B1  F0            ASCDAT: LDA     (ADDR1),Y
F1A0  20  D6  F1                JSR     PVCHAR          ; PRINT CHAR/(.)
F1A3  C8                        INY
F1A4  C4  FF                    CPY     TMPX            ; END of LINE?
F1A6  D0  F6                    BNE     ASCDAT
F1A8  A9  29                    LDA     #')
F1AA  20  12  F4                JSR     CTYOUT
F1AD  A5  F0                    LDA     ADDR1
F1AF  18                        CLC
F1B0  65  FF                    ADC     TMPX            ; ADD # of BYTES/LINE
F1B2  85  F0                    STA     ADDR1+0         ; LOW
F1B4  A5  F1                    LDA     ADDR1+1
F1B6  69  00                    ADC     #$00
F1B8  85  F1                    STA     ADDR1+1         ; HIGH
F1BA  C6  F6                    DEC     COUNT           ; KEEP TRACK of # of LINES
F1BC  D0  C7                    BNE     LINE
F1BE  A9  3F                    LDA     #'?
F1C0  20  12  F4                JSR     CTYOUT
F1C3  20  9D  F3                JSR     CTYHLF          ; HALF DUP so CHAR WON'T PRINT
F1C6  C9  03                    CMP     #'C&$3F         ; IF (^C) QUIT
F1C8  F0  19                    BEQ     HM.END
F1CA  C9  0D                    CMP     #$0D            ; IF <cr> QUIT
F1CC  F0  15                    BEQ     HM.END
F1CE  A9  0A                    LDA     #$0A            ; PRINT <lf>
F1D0  20  12  F4                JSR     CTYOUT
F1D3  4C  81  F1                JMP     PAGE
```

```
370
371                             ; ROUTINES USED by both HEXDMP & MEMORY
372
373                                                             ; PRINT VALID CHAR
374  F1D6   C9   7F             PVCHAR: CMP     #$7F            ; RUBOUT
375  F1D8   B0   04                     BGE     PERID1
376  F1DA   C9   20                     CMP     #$20            ; <space>
377  F1DC   B0   02                     BGE     PRINT
378  F1DE   A9   2E             PERID1: LDA     #'.
379  F1E0   4C   12   F4        PRINT:  JMP     CTYOUT
380
381
382
383
384  F1E3   60                  HM.END: RTS                     ; RETURN to RESET2
385
```

```
                        .SBTTL  Commands - Memory

                        ; M                                     ; M $XXXX
F1E4  20  CB  F0    MEMORY: JSR     G.ADDR                      ; GET ADDR
F1E7  20  ED  F0    MEM1:   JSR     PRTADR                      ; PRT PROMPT, ADDR & SPA
F1EA  A0  00                LDY     #$00                        ; STRAIGHT INDIRECT ADDRESSING
F1EC  B1  F0                LDA     (ADDR1),Y
F1EE  48                    PHA                                 ; SAVE ACC
F1EF  20  1F  F4            JSR     BYTSPA                      ; MUST PRESERVE %Y
F1F2  68                    PLA                                 ; RESTORE ACC
F1F3  20  D6  F1            JSR     PVCHAR                      ; PRINT CHAR/(.)
F1F6  20  09  F0            JSR     CTYSPA
F1F9  20  A9  F3            JSR     UPRCHR                      ; GET UPPER CASE CHARACTER
F1FC  C9  0D                CMP     #$0D                        ; EXIT on <cr>
F1FE  F0  E3                BEQ     HM.END
F200  C9  08                CMP     #'H&$3F                     ; (^H) \
F202  F0  2C                BEQ     MEMDEC                      ;       >DECREMENT MEMORY POINTER
F204  C9  5E                CMP     #'^                         ; (^)  /
F206  F0  28                BEQ     MEMDEC
F208  C9  47                CMP     #<'F +1>                    ; ASCII F +1
F20A  B0  1E                BGE     MEMINC
F20C  C9  2E                CMP     #'.                         ; SWTBUG COMPATIBLE
F20E  F0  1A                BEQ     MEMINC
F210  C9  22                CMP     #'"
F212  90  16                BLT     MEMINC
F214  20  C6  F3            JSR     GETBYT+3                    ; IF CHAR = (") or (') RETURNS NEXT CHAR
F217  A0  00                LDY     #$00
F219  91  F0                STA     (ADDR1),Y
F21B  D1  F0                CMP     (ADDR1),Y
F21D  F0  0B                BEQ     MEM.OK                      ; SKIP ERROR INDICATION IF OK
F21F  20  09  F0            JSR     CTYSPA
F222  A9  3F                LDA     #'?
F224  20  12  F4            JSR     CTYOUT
F227  4C  E7  F1            JMP     MEM1                        ; RE-DO THE SAME LOCATION ON ERROR
F22A                    MEM.OK:
F22A  20  06  F1    MEMINC: JSR     INCADR
F22D  4C  E7  F1            JMP     MEM1
F230  20  0D  F1    MEMDEC: JSR     DECADR
F233  4C  E7  F1            JMP     MEM1

```

```
428
429                                ; I                                  ; I $XXXX ............<^Z>
430  F236  20  CB  F0  INSERT: JSR  G.ADDR                              ; SUBROUTINE
431  F239  20  97  F3  INSRT1: JSR  CTYIN                               ; to Insert ASCII
432  F23C  C9  1A               CMP  #'Z&$3F                            ; EXIT if <^Z>
433  F23E  F0  1D               BEQ  INSEND
434  F240  C9  08               CMP  #'H&$3F                            ; (^H) BACKSPACE
435  F242  F0  13               BEQ  INSDEC
436  F244  A0  00               LDY  #$00                               ; STRAIGHT INDIRECT ADDRESSING
437  F246  91  F0               STA  (ADDR1),Y
438  F248  D1  F0               CMP  (ADDR1),Y
439  F24A  F0  05               BEQ  INS.OK                             ; SKIP ERROR EXIT IF OK
440  F24C  A9  3F               LDA  #'?
441  F24E  4C  12  F4           JMP  CTYOUT                             ; RETURN to RESET2 (USE CTYOUTs RTS)
442
443  F251  20  06  F1  INS.OK: JSR  INCADR
444  F254  4C  39  F2           JMP  INSRT1
445  F257  20  0D  F1  INSDEC: JSR  DECADR
446  F25A  4C  39  F2           JMP  INSRT1
447  F25D  4C  ED  F0  INSEND: JMP  PRTADR                              ; USE PRTADR RTS
448
```

```
                                .SBTTL  Commands - Search

                                ; S                                     ; S $XXXX-XXXX $XX XX XX .. .. .. <cr>
F260  20  CB  F0        SEARCH: JSR     G.ADDR                          ; VARIABLE BYTE SEARCH (MAX = 17)
F263  20  D8  F0                JSR     ENDADR
F266  A9  24                    LDA     #'$
F268  20  12  F4                JSR     CTYOUT
F26B  A2  00                    LDX     #$00
F26D  20  A9  F3        SRCH:   JSR     UPRCHR                          ; GET UPPER CASE CHARACTER
F270  C9  0D                    CMP     #$0D                            ; QUIT ON <cr>
F272  F0  10                    BEQ     SRCHO
F274  20  C6  F3                JSR     GETBYT+3                        ; ENTER AFTER 1st JSR CTYIN
F277  95  F6                    STA     BUFFER,X
F279  20  09  F0                JSR     CTYSPA                          ; PRINT SPACE
F27C  E8                        INX
F27D  E0  09                    CPX     #TMPX-BUFFER                    ; ? BYTES LONG (CAN go over SAVE AREA)
F27F  90  EC                    BLT     SRCH

F281  20  B7  F0                JSR     CRLF                            ; NEW LINE
F284  CA                SRCHO:  DEX
F285  86  FF                    STX     TMPX
F287  A4  FF            SRCH1:  LDY     TMPX
F289  B1  F0            SRCH2:  LDA     (ADDR1),Y
F28B  D9  F6  00                CMP     BUFFER,Y
F28E  F0  09                    BEQ     YES
F290  20  06  F1        SRCINC: JSR     INCADR
F293  20  18  F1                JSR     FINISH                          ; CHECK if FINISHED
F296  90  EF                    BCC     SRCH1
F298  60                        RTS                                     ; RETURN to RESET2

F299  C0  00            YES:    CPY     #0
F29B  F0  03                    BEQ     YESS                            ; IF ZERO then SUCCESSFUL
F29D  88                        DEY
F29E  10  E9                    BPL     SRCH2                           ; WILL ALWAYS BRANCH

F2A0  20  ED  F0        YESS:   JSR     PRTADR
F2A3  A9  20                    LDA     #$20                            ; <space>
F2A5  20  4B  F4                JSR     PCNTRL                          ; ENABLE (^S)/(^Q) & (^C)
F2A8  4C  90  F2                JMP     SRCINC

```

```
                        .SBTTL  Commands - Memory Tests

                        ; ?                                 ; ? $XXXX-XXXX
F2AB  20  CB  F0    MEMTST: JSR     G.ADDR                  ; BLOCK MEMORY TEST (NON-DESTRUCTIVE)
F2AE  20  D8  F0            JSR     ENDADR
F2B1  A0  00                LDY     #$00                    ; STRAIGHT INDIRECT ADDRESSING

F2B3  B1  F0        TEST1:  LDA     (ADDR1),Y
F2B5  48                    PHA                             ; SAVE ORIGINAL BYTE

F2B6  49  FF                EOR     #$FF                    ; FLIP THE BITS
F2B8  91  F0                STA     (ADDR1),Y               ; STORE THE INVERTED VALUE
F2BA  D1  F0                CMP     (ADDR1),Y               ; CHECK
F2BC  D0  20                BNE     TSTBAD                  ; PRINT IF BAD

F2BE  A9  00                LDA     #$00
F2C0  91  F0                STA     (ADDR1),Y               ; STORE $00
F2C2  D1  F0                CMP     (ADDR1),Y               ; CHECK
F2C4  D0  18                BNE     TSTBAD                  ; PRINT IF BAD

F2C6  A9  FF                LDA     #$FF
F2C8  91  F0                STA     (ADDR1),Y               ; STORE $FF
F2CA  D1  F0                CMP     (ADDR1),Y               ; CHECK
F2CC  D0  10                BNE     TSTBAD                  ; PRINT IF BAD

F2CE  A9  55                LDA     #$55
F2D0  91  F0                STA     (ADDR1),Y               ; STORE $55
F2D2  D1  F0                CMP     (ADDR1),Y               ; CHECK
F2D4  D0  08                BNE     TSTBAD                  ; PRINT IF BAD

F2D6  A9  AA                LDA     #$AA
F2D8  91  F0                STA     (ADDR1),Y               ; STORE $AA
F2DA  D1  F0                CMP     (ADDR1),Y               ; CHECK
F2DC  F0  12                BEQ     TEST2                   ; SKIP OVER IF GOOD

F2DE  48            TSTBAD: PHA                             ; SAVE what SHOULD be in MEMORY
F2DF  20  ED  F0            JSR     PRTADR
F2E2  B1  F0                LDA     (ADDR1),Y
F2E4  20  F8  F3            JSR     PRTBYT                  ; PRINT what IS in MEMORY
F2E7  A9  20                LDA     #$20                    ; <space>
F2E9  20  4B  F4            JSR     PCNTRL                  ; ENABLE (^S)/(^Q) & (^C)
F2EC  68                    PLA
F2ED  20  F8  F3            JSR     PRTBYT                  ; PRINT what SHOULD be in MEMORY

F2F0  68            TEST2:  PLA
F2F1  91  F0                STA     (ADDR1),Y               ; RESTORE ORIGINAL BYTE
F2F3  20  06  F1            JSR     INCADR
F2F6  20  18  F1            JSR     FINISH                  ; CHECK if FINISHED
F2F9  90  B8                BCC     TEST1
F2FB  60                    RTS                             ; RETURN to RESET2
```

```
                                ; !                             ; ! $XX-XX
F2FC  A9  24        MTEST:  LDA     #'$                     ; PAGE MEMORY TEST (DESTRUCTIVE)
F2FE  20  E7  FO            JSR     CHRHEX                  ; GET PAGE # of START PAGE
F301  85  F1                STA     ADDR1+1
F303  85  F2                STA     ADDR2                   ; SAVE in LOW POINTER
F305  A9  00                LDA     #$00
F307  85  F0                STA     ADDR1+0                 ; ZERO OUT LOW BYTE
F309  A9  2D                LDA     #'-
F30B  20  E7  FO            JSR     CHRHEX                  ; GET PAGE # of END PAGE (+1)
F30E  85  F3                STA     ADDR2+1                 ; SAVE in HIGH POINTER

F310  A0  00        MTSTO:  LDY     #0                      ; INIT Y REG to ZERO
F312  98                    TYA                             ; START OFF WITH ACC = ZERO
F313  51  F0        MTST1:  EOR     (ADDR1),Y               ; COMPLIMENT with what ever was in ACC
F315  91  F0                STA     (ADDR1),Y               ; STORE
F317  D1  F0                CMP     (ADDR1),Y               ; CHECK
F319  F0  18                BEQ     MTSTOK                  ; SKIP OVER PRINT if GOOD.

F31B  48                    PHA                             ; SAVE ACC (WHAT SHOULD be in MEMORY)
F31C  20  B7  FO            JSR     CRLF
F31F  A5  F1                LDA     ADDR1+1
F321  20  F8  F3            JSR     PRTBYT
F324  98                    TYA                             ; GET LOW BYTE of ADDR
F325  20  F8  F3            JSR     PRTBYT                  ; PRINT LOW BYTE
F328  A9  20                LDA     #$20                    ; <space>
F32A  20  4B  F4            JSR     PCNTRL                  ; ENABLE (^S)/(^Q) & (^C)
F32D  68                    PLA                             ; RESTORE WHAT SHOULD BE THERE
F32E  51  F0                EOR     (ADDR1),Y               ; EOR WITH BAD BYTE TO GIVE BAD BITS
F330  20  F8  F3            JSR     PRTBYT                  ; DISPLAY BAD BITS ONLY

F333  C8            MTSTOK: INY                             ; INCREMENT LOW POINTER
F334  D0  DD                BNE     MTST1
F336  E6  F1                INC     ADDR1+1
F338  A5  F1                LDA     ADDR1+1
F33A  C5  F3                CMP     ADDR2+1                 ; CHECK IF DONE
F33C  D0  D5                BNE     MTST1
F33E  A5  F2                LDA     ADDR2+0                 ; RE-INIT ADDR1 TO ADDR LOW
F340  85  F1                STA     ADDR1+1
F342  A9  23                LDA     #'#                     ; PRINT
F344  20  4B  F4            JSR     PCNTRL                  ; ENABLE (^S)/(^Q) & (^C)
F347  4C  10  F3            JMP     MTSTO                   ; CONTINUE
                                                            ; USE (^C) TO EXIT
```

```
                        .SBTTL  Commands - Block Compare

                        ; B                             ; B $XXXX-XXXX / $XXXX
F34A  20  CB  F0    BLKCMP: JSR     G.ADDR              ; BLOCK MEMORY COMPARE
F34D  20  D8  F0            JSR     ENDADR
F350  A9  2F                LDA     #'/
F352  20  12  F4            JSR     CTYOUT
F355  20  09  F0            JSR     CTYSPA
F358  A9  24                LDA     #'$
F35A  20  E7  F0            JSR     CHRHEX
F35D  85  F3                STA     ADDR2+1
F35F  20  C3  F3            JSR     GETBYT
F362  85  F2                STA     ADDR2
F364  A0  00                LDY     #0                  ; STRAIGHT INDIRECT ADDRESSING
F366  B1  F0        BLKCP1: LDA     (ADDR1),Y
F368  D1  F2                CMP     (ADDR2),Y
F36A  F0  1F                BEQ     BLKCP2

F36C  20  ED  F0            JSR     PRTADR              ; PRINT DIFFERENT ADDR's & DATA
F36F  B1  F0                LDA     (ADDR1),Y
F371  20  1F  F4            JSR     BYTSPA              ; PRINT BYTE & SPACE
F374  20  09  F0            JSR     CTYSPA
F377  A9  24                LDA     #'$
F379  20  4B  F4            JSR     PCNTRL              ; ENABLE (^S)/(^Q) & (^C)
F37C  A5  F3                LDA     ADDR2+1
F37E  20  F8  F3            JSR     PRTBYT
F381  A5  F2                LDA     ADDR2
F383  20  1F  F4            JSR     BYTSPA              ; PRINT BYTE & SPACE
F386  B1  F2                LDA     (ADDR2),Y
F388  20  F8  F3            JSR     PRTBYT

F38B  20  06  F1    BLKCP2: JSR     INCADR              ; INCREMENT the POINTERS
F38E  20  FF  F0            JSR     INCPT
F391  20  18  F1            JSR     FINISH              ; CHECK if FINISHED
F394  90  D0                BCC     BLKCP1
F396  60                    RTS                         ; RETURN to RESET2
```

```
                        .SBTTL  CTY  In-Put Routines


                        ;  I/O ROUTINES for using a 6850 ACIA at $EF00
                        ;       (M.T.U. K-1012 EPROM-I/O BOARD)


F397  20  9D  F3  CTYIN:  JSR   CTYHLF          ; GET CHAR IN HALF DUPLEX
F39A  4C  12  F4          JMP   CTYOUT          ; PRINT CHAR & RETURN
                                                ; RTS  X = X  Y = Y  A = CHAR  C = 1

                                                ; GET CHAR from CTY: ($EF00) (HALF DUPLEX)
F39D  AD  00  EF  CTYHLF: LDA   CTY             ; ACIA CONTROL REG
F3A0  4A                  LSR   A               ; TEST RDRF (BIT 0)
F3A1  90  FA              BCC   CTYHLF          ; BRANCH if no CHARACTER PRESENT
F3A3  AD  01  EF          LDA   CTY+1           ; GET CHAR
F3A6  29  7F              AND   #$7F            ; MASK off PARITY
F3A8  60                  RTS                   ; RTS  X = X  Y = Y  A = CHAR  C = 1

                                                ; GET UPPER CASE CHARACTER
F3A9  20  9D  F3  UPRCHR: JSR   CTYHLF          ; GET CHAR HALF DUPLEX
F3AC  C9  60              CMP   #$60            ; CHECK if LOWER CASE
F3AE  90  62              BLT   CTYOUT          ; IF NOT, GO PRINT IT
F3B0  29  5F              AND   #$5F            ; MAKE UPPER CASE if LOWER CASE
F3B2  10  5E              BPL   CTYOUT          ; PRINT IT IN UPPER CASE
                                                ; WILL ALWAYS BRANCH


F3B4  AD  00  EF  KEYDWN: LDA   CTY             ; ACIA CONTROL REG
F3B7  4A                  LSR   A               ; TEST RDRF (BIT 0)
F3B8  90  06              BCC   NO.KEY          ; SKIP IF NO KEY
F3BA  AD  01  EF          LDA   CTY+1           ; GET THE CHAR
F3BD  29  7F              AND   #$7F            ; TURN OFF BIT 7 (INDICATING CHAR PRESENT)
F3BF  60                  RTS                   ; INDICATING KEY PRESENT

F3C0  A9  80  NO.KEY:     LDA   #$80            ; TURN ON BIT 7
F3C2  60                  RTS                   ; INDICATING NO CHAR PRESENT


                        ; RDRF == Receiver Data Register Full
                        ; TDRE == Transmit Data Register Empty
```

```
677
678  F3C3  20  A9  F3   GETBYT: JSR   UPRCHR          ; GET HIGH NIBBLE  (UPPER CASE)
679  F3C6  C9  22               CMP   #'"             ; GETBYT + 3
680  F3C8  F0  CD               BEQ   CTYIN           ; ASCII <">
681  F3CA  C9  27               CMP   #''
682  F3CC  F0  C9               BEQ   CTYIN           ; ASCII <'>
683  F3CE  20  E0  F3           JSR   HEX
684  F3D1  0A                   ASL   A
685  F3D2  0A                   ASL   A
686  F3D3  0A                   ASL   A
687  F3D4  0A                   ASL   A
688  F3D5  85  FE               STA   TEMP            ; SAVE HIGH NIBBLE
689  F3D7  20  A9  F3           JSR   UPRCHR          ; GET LOW  NIBBLE  (UPPER CASE)
690  F3DA  20  E0  F3           JSR   HEX
691  F3DD  05  FE               ORA   TEMP
692  F3DF  60                   RTS                   ; RTS X = X  Y = Y  A = BYTE  C = 1
693
694
695  F3E0  C9  30       HEX:    CMP   #'0             ; LESS THAN 0 ?
696  F3E2  90  11               BLT   NONHEX
697  F3E4  C9  3A               CMP   #<'9 +1>        ; ASCII 9 +1
698  F3E6  90  0A               BLT   HEXOK
699  F3E8  C9  47               CMP   #<'F +1>        ; ASCII F +1
700  F3EA  B0  09               BGE   NONHEX
701  F3EC  E9  06               SBC   #$06
702  F3EE  C9  3A               CMP   #$3A            ; 9 < CHR < A ?
703  F3F0  90  03               BLT   NONHEX
704  F3F2  29  0F       HEXOK:  AND   #$0F
705  F3F4  60                   RTS
706
707  F3F5  4C  4F  F0   NONHEX: JMP   RESET2          ; GO TO RESET2 if WRONG
708
709
```

```
                                .SBTTL  CTY  Out-Put Routines

F3F8  48          PRTBYT: PHA                     ; SAVE BYTE TWICE
F3F9  48                  PHA
F3FA  4A                  LSR     A             ; GET HIGH NIBBLE
F3FB  4A                  LSR     A
F3FC  4A                  LSR     A
F3FD  4A                  LSR     A
F3FE  20  07  F4          JSR     HEXASC          ; PRINT 1st NIBBLE
F401  68                  PLA                     ; RESTORE BYTE
F402  20  07  F4          JSR     HEXASC          ; PRINT 2nd NIBBLE
F405  68                  PLA                     ; RESTORE BYTE
F406  60                  RTS                     ; RTS  X = X  Y = Y  A = A

F407  29  0F      HEXASC: AND     #$0F            ; CONVERT NIBBLE TO ASCII CHAR
F409  C9  0A              CMP     #$0A
F40B  18                  CLC
F40C  30  02              BMI     HEXASI
F40E  69  07              ADC     #$07
F410  69  30      HEXASI: ADC     #$30            ; FALL THROUGH to CTYOUT
                                                  ; & PRINT the NIBBLE


F412  48          CTYOUT: PHA                     ; PRINT CHAR to CTY: ($EF00)
F413  AD  00  EF  CTYOU1: LDA     CTY             ; GET CONTROL REGISTER
F416  4A                  LSR     A
F417  4A                  LSR     A               ; TEST TDRE (BIT 1)
F418  90  F9              BCC     CTYOU1          ; BRANCH IF NOT READY
F41A  68                  PLA
F41B  8D  01  EF          STA     CTY+1           ; PRINT CHAR
F41E  60                  RTS                     ; RTS  X = X  Y = Y  A = A  C = 1

F41F  20  F8  F3  BYTSPA: JSR     PRTBYT          ; PRINT BYTE & <space>
F422  4C  09  F0          JMP     CTYSPA          ; RTS  X = X  Y = Y  A = $20


```

```
                                .SBTTL  System I/O

F425                            KYBDIN:                         ; DOESN'T EXIST YET
F425  60                                RTS

                                ; 07-NOV-80                     ; PARALLEL PRINTER OUTPUT ROUTINE
                                ; 17-JAN-81                     ; CHANGED A LITTLE BIT
                                ; 28-AUG-81                     ; CHANGED PORT TO NOT CONFLICT WITH EPROM PGMR
                                                                ; USES CENTRONICS STANDARDS
F426  8D  16  EF                PRINTR: STA     PIA3B+0         ; PUT DATA IN PIA DATA REG
F429  AD  17  EF                PRTWLP: LDA     PIA3B+1         ; PIA WAIT LOOP
F42C  10  FB                            BPL     PRTWLP          ; (WAIT TILL PRINTER READY)
F42E  AD  16  EF                        LDA     PIA3B+0         ; RESET PORT
F431  60                                RTS                     ; RTS  X = X  Y = Y  A = A


                                                                ; GET CHAR from TTY: ($EF10) (HALF DUPLEX)
F432  AD  10  EF                TTYIN:  LDA     TTY             ; ACIA CONTROL REG
F435  4A                                LSR     A               ; TEST RDRF (BIT 0)
F436  90  FA                            BCC     TTYIN           ; BRANCH if no CHARACTER PRESENT
F438  AD  11  EF                        LDA     TTY+1           ; GET CHAR
F43B  29  7F                            AND     #$7F            ; MASK off PARITY
F43D  60                                RTS                     ; RTS  X = X  Y = Y  A = CHAR  C = 1

                                                                ; PRINT CHAR to TTY: ($EF10)
F43E  48                        TTYOUT: PHA                     ; SAVE CHAR TO BE PRINTED
F43F  AD  10  EF                TTYOU1: LDA     TTY             ; GET CONTROL REGISTER
F442  4A                                LSR     A
F443  4A                                LSR     A               ; TEST TDRE (BIT 1)
F444  90  F9                            BCC     TTYOU1          ; BRANCH IF NOT READY
F446  68                                PLA                     ; RESTORE CHAR TO BE PRINTED
F447  8D  11  EF                        STA     TTY+1           ; PRINT CHAR
F44A  60                                RTS                     ; RTS  X = X  Y = Y  A = A  C = 1


```

```
                        .SBTTL  Special I/O


F44B  48          PCNTRL: PHA                     ; PRINT WITH (^S)/(^Q) & (^C) IMPLEMENTED
F44C  20  B4  F3          JSR     KEYDWN          ; SEE IF A KEY IS DEPRESSED
F44F  30  13              BMI     PCTRL2          ; BRANCH OUT IF NOT
F451  C9  03              CMP     #'C&$3F         ; (^C)  ABORT
F453  F0  14              BEQ     ABORT
F455  C9  13              CMP     #'S&$3F         ; (^S)  STOP PRINTING
F457  D0  0B              BNE     PCTRL2

F459  20  9D  F3  CNTRLS: JSR     CTYHLF
F45C  C9  03              CMP     #'C&$3F
F45E  F0  09              BEQ     ABORT           ; ABORT ON (^C)
F460  C9  11              CMP     #'Q&$3F         ; (^Q)  CONTINUE PRINTING
F462  D0  F5              BNE     CNTRLS


F464  68          PCTRL2: PLA                     ; RESTORE CHARACTER
F465  20  12  F4          JSR     CTYOUT          ; AND PRINT IT
F468  60                  RTS


F469  20  B7  F0  ABORT:  JSR     CRLF            ; NEW LINE
F46C  A9  5E              LDA     #'^
F46E  20  12  F4          JSR     CTYOUT          ; PRINT '^C'
F471  A9  43              LDA     #'C
F473  20  12  F4          JSR     CTYOUT
F476  4C  4F  F0          JMP     RESET2          ; JUMP to RESET2


```

```
                        .SBTTL  Disassembler

                        ;  DISASSEMBLER for the 6502
                        ; Written by Steve Wozniak & Allen Baum
                        ; Published in 'DR. DOBB'S JOURNAL' Sept. 1976
                        ; Altered by Keith Sproul July 1979

                        ;  IF A SPACE IS TYPED AFTER THE 'Z', THE DISASSEMBLER WILL DISASSMBLE
                        ; STARTING AT THE ADDR IN PCL,PCH.  THE INTERRUPT PROCCESOR, PAPER TAPE,
                        ; & CASSETTE LOAD ROUTINES PUT THE APPROPRIATE ADDRESSES AT PCL,PCH.


                        ;       OPERAND         ADDRESS MODE

                        ;       (empty) INVALID,IMPLIED.
                        ;       (empty) ACCUMULATOR.
                        ;       $xx             ZERO PAGE.
                        ;       $xxxx           ABSOLUTE BRANCH (TARGET PRINTED).
                        ;       #$xx            IMMEDIATE.
                        ;       $xx,X           ZERO PAGE,INDEXED BY X.
                        ;       $xx,Y           ZERO PAGE,INDEXED BY Y.
                        ;       $xxxx,X         ABSOLUTE,INDEXED BY X.
                        ;       $xxxx,Y         ABSOLUTE.INDEXED BY Y.
                        ;       ($xxxx)         INDIRECT
                        ;       ($xx,X)         INDEXED INDIRECT.
                        ;       ($xx),Y         INDIRECT INDEXED.

```

```
849
850                                ; Z                                  ; Z XXXX
851  F479  20  A9  F3  DISASM: JSR  UPRCHR
852  F47C  C9  20               CMP  #$20                               ; IS IT a <space>?
853  F47E  F0  0A               BEQ  DSMBL1                             ; CONTINUE AT PRESENT P.C. IF SO.
854  F480  20  C6  F3           JSR  GETBYT+3                           ; GET REST OF BYTE
855  F483  85  F8               STA  PCH
856  F485  20  C3  F3           JSR  GETBYT
857  F488  85  F7               STA  PCL
858
859  F48A  A9  10       DSMBL1: LDA  #16                                ; COUNT FOR 16 INSTR DISASSEMBLY
860  F48C  85  F6               STA  COUNT
861  F48E  20  AC  F4   DSMBL2: JSR  INSTDS                             ; DISASSEMBLE & DISPLAY INSTR.
862  F491  20  84  F5           JSR  PCADJ                              ; UPDATE PCL,PCH TO NEXT INSTR.
863  F494  85  F7               STA  PCL                                ; SAVE THEM
864  F496  84  F8               STY  PCH
865  F498  C6  F6               DEC  COUNT
866  F49A  D0  F2               BNE  DSMBL2                             ; DONE? THEN LOOP.
867
868  F49C  20  9D  F3   DSWAIT: JSR  CTYHLF
869  F49F  C9  03               CMP  #'C&$3F                            ; QUIT ON (^C)
870  F4A1  F0  08               BEQ  DSMRTS
871  F4A3  C9  13               CMP  #'S&$3F                            ; DONT ADVANCE ON ^S
872  F4A5  F0  F5               BEQ  DSWAIT
873  F4A7  C9  0D               CMP  #$0D                               ; EXIT IF <cr>
874  F4A9  D0  DF               BNE  DSMBL1                             ; ELSE CONTINUE
875  F4AB  60           DSMRTS: RTS
876
877  F4AC  20  6C  F5   INSTDS: JSR  PRNTPC                             ; PRINT PCL,H.
878  F4AF  A1  F7               LDA  (PCL,X)                            ; GET OP CODE.
879  F4B1  A8                   TAY                                     ; SAVE OP CODE in Y
880  F4B2  4A                   LSR  A                                  ; * EVEN/ODD TEST.
881  F4B3  90  0B               BCC  IEVEN
882  F4B5  4A                   LSR  A                                  ; *  TEST B1.
883  F4B6  B0  17               BCS  ERR                                ; *  XXXXXX11 INSTR INVALID.
884  F4B8  C9  22               CMP  #$22
885  F4BA  F0  13               BEQ  ERR                                ; *  10001001 INSTR INVALID.
886  F4BC  29  07               AND  #%00000111                         ; MASK 3 BITS FOR ADDRESS MODE
887  F4BE  09  80               ORA  #$80                               ; *  ADD INDEXING OFFSET.
888  F4C0  4A           IEVEN:  LSR  A                                  ; * LSB INTO CARRY FOR
889  F4C1  AA                   TAX                                     ; * LEFT/RIGHT TEST BELOW.
890  F4C2  BD  93  F5           LDA  AMODE,X                            ; INDEX INTO ADDRESS MODE TABLE.
891  F4C5  B0  04               BCS  RTMODE                             ; IF CARRY SET USE LSD FOR
892  F4C7  4A                   LSR  A                                  ; *  PRINT FORMAT INDEX.
893  F4C8  4A                   LSR  A
894  F4C9  4A                   LSR  A                                  ; *  IF CARRY CLEAR USE MSD.
895  F4CA  4A                   LSR  A
896  F4CB  29  0F       RTMODE: AND  #%00001111                         ; MASK FOR 4-BIT INDEX.
897  F4CD  D0  04               BNE  GETFMT                             ; $0 FOR INVALID OPCODES.
898  F4CF  A0  80       ERR:    LDY  #$80                               ; SUBSTITUTE $80 FOR INVALID OP.
899  F4D1  A9  00               LDA  #$00                               ; SET PRINT FORMAT INDEX TO 0.
900  F4D3  AA           GETFMT: TAX
901  F4D4  BD  D7  F5           LDA  MODE2,X                            ; INDEX INTO PRINT FORMAT TABLE.
902  F4D7  85  C2               STA  FORMAT                             ; SAVE FOR ADDRESS FIELD FORMAT.
903  F4D9  29  03               AND  #%00000011                         ; MASK 2-BIT LENGTH. 0=1-BYTE.
```

```
 904  F4DB  85  C3          STA   LENGTH          ; *  1=2-BYTE, 2=3-BYTE.
 905  F4DD  98              TYA                   ; *  OP CODE.
 906  F4DE  29  8F          AND   #%10001111      ; $8F MASK IT FOR 1XXX1010 TEST.
 907  F4E0  AA              TAX                   ; *  SAVE IT.
 908  F4E1  98              TYA                   ; *  OP CODE TO A AGAIN.
 909  F4E2  A0  03          LDY   #$3
 910  F4E4  E0  8A          CPX   #$8A
 911  F4E6  F0  0B          BEQ   MNNDX3
 912
 913  F4E8  4A        MNNDX1: LSR   A
 914  F4E9  90  08          BCC   MNNDX3          ; FORM INDEX INTO MNEMONIC TABL.
 915  F4EB  4A              LSR   A
 916  F4EC  4A        MNNDX2: LSR   A             ; *  1XXX1010 -> 001010XX
 917  F4ED  09  20          ORA   #$20            ; *  XXXYYY01 -> 001110XX
 918  F4EF  88              DEY                   ; *  XXXYYY10 -> 001100XX
 919  F4F0  D0  FA          BNE   MNNDX2          ; *  XXXYY100 -> 001000XX
 920  F4F2  C8              INY                   ; *  XXXXX000 -> 000XXXXX
 921  F4F3  88        MNNDX3: DEY
 922  F4F4  D0  F2          BNE   MNNDX1
 923  F4F6  48              PHA                   ; * SAVE MNEMONIC TABLE INDEX.
 924
 925  F4F7  B1  F7    PRNTOP: LDA   (PCL),Y       ; PRINT OP-CODE
 926  F4F9  20  1F  F4        JSR   BYTSPA        ; PRINT BYTE & SPACE
 927  F4FC  C4  C3            CPY   LENGTH        ; PRINT INSTR (1 TO 3 BYTES)
 928  F4FE  C8                INY                 ; *  IN A 12-CHARACTER FIELD.
 929  F4FF  90  F6            BCC   PRNTOP
 930
 931  F501  20  7B  F5 FILBLK: JSR   PRBLNK       ; PRINT 3 SPACES.
 932  F504  C8                INY
 933  F505  C0  04            CPY   #4            ; FILL OUT REST OF FIELD.
 934  F507  90  F8            BLT   FILBLK
 935
 936  F509  68                PLA                 ; RECOVER MNEMONIC INDEX.
 937  F50A  A8                TAY                 ; PUT INTO Y REG
 938  F50B  B9  F1  F5        LDA   MNEML,Y       ; USE AS INDEX
 939  F50E  85  C0            STA   LMNEM         ; FETCH 3-CHAR MNEMONIC.
 940  F510  B9  31  F6        LDA   MNEMR,Y       ; *  (PACKED IN 2 BYTES)
 941  F513  85  C1            STA   RMNEM
 942  F515  A2  03            LDX   #3            ; SET CHAR COUNT FOR MNEMONIC
 943  F517  A9  00    PRTMN1: LDA   #$00          ; PRINT MNEMONIC
 944  F519  A0  05            LDY   #5
 945  F51B  06  C1    PRTMN2: ASL   RMNEM
 946  F51D  26  C0            ROL   LMNEM         ; SHIFT 5 BITS OF CHAR INTO A.
 947  F51F  2A                ROL   A             ; *  (CLEARS CARRY)
 948  F520  88                DEY
 949  F521  D0  F8            BNE   PRTMN2
 950  F523  69  3F            ADC   #'?           ; ADD $3F OFFSET.
 951  F525  20  4B  F4        JSR   PCNTRL        ; OUTPUT A CHARACTER OF MNEMONIC
 952  F528  CA                DEX
 953  F529  D0  EC            BNE   PRTMN1
 954
 955  F52B  20  7B  F5        JSR   PRBLNK        ; OUTPUT 3 BLANKS.
 956  F52E  A2  06            LDX   #6            ; COUNT FOR 6 PRINT FORMAT BITS.
 957
 958  F530  E0  03    PRADR1: CPX   #$3
```

```
 959  F532  D0  12          BNE   PRADR3          ; IF X=3 THEN PRINT ADDRESS VAL.
 960  F534  A4  C3          LDY   LENGTH
 961  F536  F0  0E          BEQ   PRADR3          ; NO PRINT IF LENGTH=0.
 962
 963  F538  A5  C2    PRADR2: LDA   FORMAT
 964  F53A  C9  E8          CMP   #$E8            ; HANDLE REL ADDRESSING MODE
 965  F53C  B1  F7          LDA   (PCL),Y         ; SPECIAL (PRINT TARGET ADR)
 966  F53E  B0  1C          BGE   RELADR          ; *  (NOT DISPLACEMENT)
 967  F540  20  F8  F3      JSR   PRTBYT          ; OUTPUT 1- OR 2-BYTE ADDRESS.
 968  F543  88              DEY                   ; *  MORE SIGNIFICANT BYTE FIRST
 969  F544  D0  F2          BNE   PRADR2
 970
 971  F546  06  C2    PRADR3: ASL   FORMAT          ; TEST NEXT PRINT FORMAT BIT.
 972  F548  90  0E          BCC   PRADR4          ; IF 0,DON'T PRINT
 973  F54A  BD  E4  F5      LDA   CHAR1-1,X       ; *  CORRESPONDING CHARS.
 974  F54D  20  4B  F4      JSR   PCNTRL          ; OUTPUT 1 OR 2 CHARS.
 975  F550  BD  EA  F5      LDA   CHAR2-1,X       ; * (IF CHAR FROM CHAR2 IS 0,
 976  F553  F0  03          BEQ   PRADR4          ;       DON'T  OUTPUT IT)
 977  F555  20  4B  F4      JSR   PCNTRL
 978  F558  CA        PRADR4: DEX
 979  F559  D0  D5          BNE   PRADR1
 980  F55B  60              RTS                   ; * RETURN IF DONE 6 FORMAT BITS.
 981
```

```
                        .SBTTL  Disassembler - Subroutines

F55C  20 87 F5  RELADR: JSR     PCADJ3          ; PCL,H + DISPL + 1 TO A,Y.
F55F  AA                TAX
F560  E8                INX
F561  D0 01             BNE     PRNTYX          ; *  +1 TO X,Y.
F563  C8                INY
F564  98        PRNTYX: TYA
F565  20 F8 F3  PRNTAX: JSR     PRTBYT          ; PRINT TARGET ADDR OF BRANCH
F568  8A                TXA                     ;       & RETURN
F569  4C F8 F3          JMP     PRTBYT          ; USE PRTBYT RTS

F56C  20 B7 F0  PRNTPC: JSR     CRLF            ; OUTPUT <cr> & <lf>.
F56F  A9 24             LDA     #'$
F571  20 4B F4          JSR     PCNTRL          ; PRINT '$'
F574  A5 F8             LDA     PCH
F576  A6 F7             LDX     PCL
F578  20 65 F5          JSR     PRNTAX          ; PRINT PCH & PCL.

                                                ; FALL THROUGH & PRINT 3 SPACES

F57B  A2 03     PRBLNK: LDX     #3              ; BLANK COUNT.
F57D  20 09 F0  PRBL2:  JSR     CTYSPA          ; OUTPUT A BLANK.
F580  CA                DEX
F581  D0 FA             BNE     PRBL2           ; LOOP UNTIL COUNT = 0.
F583  60                RTS

F584  A5 C3     PCADJ:  LDA     LENGTH          ; 0=1-BYTE, 1=2-BYTE, 2=3-BYTE.
F586  38                SEC
F587  A4 F8     PCADJ3: LDY     PCH
F589  AA                TAX                     ; *  TEST DISPL SIGN (FOR REL
F58A  10 01             BPL     PCADJ4          ; *     BRANCH).  EXTEND NEG
F58C  88                DEY                     ; *     BY DECREMENTING PCH.
F58D  65 F7     PCADJ4: ADC     PCL
F58F  90 01             BCC     RTS1            ; PCL+LENGTH (OR DISPL) +1 TO A.
F591  C8                INY                     ; *  CARRY INTO Y (PCH)
F592  60        RTS1:   RTS

                        ; DIS-ASSEMBLER DATA TABLES

                        ;       .NLIST
```

```
1027                            .SBTTL  Disassembler - Format Tables
1028
1029
1030  F593  40          AMODE:  .BYTE   $40              ; ADDRESS MODE TABLE
1031  F594  02                  .BYTE   $2
1032  F595  45                  .BYTE   $45
1033  F596  03                  .BYTE   $3
1034  F597  D0                  .BYTE   $D0
1035  F598  08                  .BYTE   $8
1036  F599  40                  .BYTE   $40
1037  F59A  09                  .BYTE   $9
1038  F59B  30                  .BYTE   $30
1039  F59C  22                  .BYTE   $22              ; XXXXXXZ0 INSTRS.
1040  F59D  45                  .BYTE   $45
1041  F59E  33                  .BYTE   $33
1042  F59F  D0                  .BYTE   $D0
1043  F5A0  08                  .BYTE   $8
1044  F5A1  40                  .BYTE   $40
1045  F5A2  09                  .BYTE   $9
1046  F5A3  40                  .BYTE   $40
1047  F5A4  02                  .BYTE   $2
1048  F5A5  45                  .BYTE   $45
1049  F5A6  33                  .BYTE   $33              ; *  Z=0, LEFT HALF-BYTE
1050  F5A7  D0                  .BYTE   $D0              ; *  Z=1, RIGHT HALF-BYTE
1051  F5A8  08                  .BYTE   $8
1052  F5A9  40                  .BYTE   $40
1053  F5AA  09                  .BYTE   $9
1054  F5AB  40                  .BYTE   $40
1055  F5AC  02                  .BYTE   $2       ; $0
1056  F5AD  45                  .BYTE   $45      ; $40
1057  F5AE  B3                  .BYTE   $B3      ; $B0
1058  F5AF  D0                  .BYTE   $D0
1059  F5B0  08                  .BYTE   $8       ; $0
1060  F5B1  40                  .BYTE   $40
1061  F5B2  09                  .BYTE   $9       ; $0
1062  F5B3  00                  .BYTE   $0
1063  F5B4  22                  .BYTE   $22
1064  F5B5  44                  .BYTE   $44
1065  F5B6  33                  .BYTE   $33
1066  F5B7  D0                  .BYTE   $D0
1067  F5B8  8C                  .BYTE   $8C
1068  F5B9  44                  .BYTE   $44
1069  F5BA  00                  .BYTE   $0
1070  F5BB  11                  .BYTE   $11
1071  F5BC  22                  .BYTE   $22
1072  F5BD  44                  .BYTE   $44
1073  F5BE  33                  .BYTE   $33
1074  F5BF  D0                  .BYTE   $D0
1075  F5C0  8C                  .BYTE   $8C
1076  F5C1  44                  .BYTE   $44
1077  F5C2  9A                  .BYTE   $9A
1078  F5C3  10                  .BYTE   $10
1079  F5C4  22                  .BYTE   $22
1080  F5C5  44                  .BYTE   $44
1081  F5C6  33                  .BYTE   $33
```

```
1082  F5C7  DO                        .BYTE   $DO
1083  F5C8  08                        .BYTE   $8
1084  F5C9  40                        .BYTE   $40
1085  F5CA  09                        .BYTE   $9
1086  F5CB  10                        .BYTE   $10
1087  F5CC  22                        .BYTE   $22
1088  F5CD  44                        .BYTE   $44
1089  F5CE  33                        .BYTE   $33
1090  F5CF  DO                        .BYTE   $DO
1091  F5DO  08                        .BYTE   $8
1092  F5D1  40                        .BYTE   $40
1093  F5D2  09                        .BYTE   $9
1094  F5D3  62                        .BYTE   $62
1095  F5D4  13                        .BYTE   $13             ; YYXXXZO1 INSTRS.
1096  F5D5  78                        .BYTE   $78
1097  F5D6  A9                        .BYTE   $A9
1098
1099
1100                                                            ; PRINT FORMAT TABLES
1101
1102                            ;                       $ADDR   ACC=A
1103
1104  F5D7  00                  MODE2:  .BYTE   $O                      ; ERR
1105  F5D8  21                          .BYTE   $21                     ; IMM
1106  F5D9  81                          .BYTE   $81     ; $81   ; $01   ; Z-PAGE
1107  F5DA  82                          .BYTE   $82     ; $82   ; $02   ; ABS           BIT O   1-BYTE
1108  F5DB  00                          .BYTE   $O                      ; IMPL          BIT 1   2-BYTE
1109  F5DC  00                          .BYTE   $00     ; $00   ; $80   ; ACC           BIT 2   ,Y
1110  F5DD  59                          .BYTE   $59                     ; (Z-PAG,X)     BIT 3   ,X
1111  F5DE  4D                          .BYTE   $4D                     ; (Z-PAG),Y     BIT 4   (
1112  F5DF  91                          .BYTE   $91     ; $91   ; $11   ; Z-PAG,X       BIT 5   #
1113  F5EO  92                          .BYTE   $92     ; $92   ; $12   ; ABS,X         BIT 6   )
1114  F5E1  86                          .BYTE   $86     ; $86   ; $06   ; ABS,Y         BIT 7   '$ or 'A
1115  F5E2  4A                          .BYTE   $4A                     ; (ABS)
1116  F5E3  85                          .BYTE   $85     ; $85   ; $05   ; Z-PAG,Y
1117  F5E4  9D                          .BYTE   $9D     ; $9D   ; $1D   ; REL
1118
1119  F5E5  2C                  CHAR1:  .BYTE   ',
1120  F5E6  29                          .BYTE   ')
1121  F5E7  2C                          .BYTE   ',
1122  F5E8  23                          .BYTE   '#
1123  F5E9  28                          .BYTE   '(
1124  F5EA  24                          .BYTE   '$      ; '$    ; 'A
1125
1126  F5EB  59                  CHAR2:  .BYTE   'Y
1127  F5EC  00                          .BYTE   0
1128  F5ED  58                          .BYTE   'X
1129  F5EE  24                          .BYTE   '$
1130  F5EF  24                          .BYTE   '$
1131  F5FO  00                          .BYTE   0
1132
```

```
1134                            .SBTTL  Disassembler - Mnemonic Tables
1135
1136
1137  F5F1  1C          MNEML:  .BYTE   $1C      ; BRK    ; XXXXX000 INSTRS.
1138  F5F2  8A                  .BYTE   $8A      ; PHP
1139  F5F3  1C                  .BYTE   $1C      ; BPL
1140  F5F4  23                  .BYTE   $23      ; CLC
1141  F5F5  5D                  .BYTE   $5D      ; JSR
1142  F5F6  8B                  .BYTE   $8B      ; PLP
1143  F5F7  1B                  .BYTE   $1B      ; BMI
1144  F5F8  A1                  .BYTE   $A1      ; SBC
1145  F5F9  9D                  .BYTE   $9D      ; RTI
1146  F5FA  8A                  .BYTE   $8A      ; PHA
1147  F5FB  1D                  .BYTE   $1D      ; BVC
1148  F5FC  23                  .BYTE   $23      ; CLI
1149  F5FD  9D                  .BYTE   $9D      ; RTS
1150  F5FE  8B                  .BYTE   $8B      ; PLA
1151  F5FF  1D                  .BYTE   $1D      ; BVS
1152  F600  A1                  .BYTE   $A1      ; SEI
1153  F601  00                  .BYTE   $0       ; ???
1154  F602  29                  .BYTE   $29      ; DEX
1155  F603  19                  .BYTE   $19      ; BCC
1156  F604  AE                  .BYTE   $AE      ; TYA
1157  F605  69                  .BYTE   $69      ; LDY
1158  F606  A8                  .BYTE   $A8      ; TAY
1159  F607  19                  .BYTE   $19      ; BCS
1160  F608  23                  .BYTE   $23      ; CLV
1161  F609  24                  .BYTE   $24      ; CPY
1162  F60A  53                  .BYTE   $53      ; INY
1163  F60B  1B                  .BYTE   $1B      ; BNE
1164  F60C  23                  .BYTE   $23      ; CLD
1165  F60D  24                  .BYTE   $24      ; CPX
1166  F60E  53                  .BYTE   $53      ; INY
1167  F60F  19                  .BYTE   $19      ; BEQ
1168  F610  A1                  .BYTE   $A1      ; SED
1169  F611  00                  .BYTE   $0       ; ???
1170  F612  1A                  .BYTE   $1A      ; BIT
1171  F613  5B                  .BYTE   $5B      ; JMP
1172  F614  5B                  .BYTE   $5B      ; JMP
1173  F615  A5                  .BYTE   $A5      ; STY
1174  F616  69                  .BYTE   $69      ; LDY
1175  F617  24                  .BYTE   $24      ; CPY
1176  F618  24                  .BYTE   $24      ; CPX
1177  F619  AE                  .BYTE   $AE      ; TXA    ; XXXYYY10 INSTRS.
1178  F61A  AE                  .BYTE   $AE      ; TXS
1179  F61B  A8                  .BYTE   $A8      ; TAX
1180  F61C  AD                  .BYTE   $AD      ; TSX
1181  F61D  29                  .BYTE   $29      ; DEX
1182  F61E  00                  .BYTE   $0       ; ???
1183  F61F  7C                  .BYTE   $7C      ; NOP
1184  F620  00                  .BYTE   $0       ; ???
1185  F621  15                  .BYTE   $15      ; ASL    ; XXXYYY10 INSTRS.
1186  F622  9C                  .BYTE   $9C      ; ROL
1187  F623  6D                  .BYTE   $6D      ; LSR
1188  F624  9C                  .BYTE   $9C      ; ROR
```

```
1189  F625  A5                        .BYTE   $A5     ; STX
1190  F626  69                        .BYTE   $69     ; LDX
1191  F627  29                        .BYTE   $29     ; DEC
1192  F628  53                        .BYTE   $53     ; INC
1193  F629  84                        .BYTE   $84     ; ORA   ; XXXYYY01 INSTRS.
1194  F62A  13                        .BYTE   $13     ; AND
1195  F62B  34                        .BYTE   $34     ; EOR
1196  F62C  11                        .BYTE   $11     ; ADC
1197  F62D  A5                        .BYTE   $A5     ; STA
1198  F62E  69                        .BYTE   $69     ; LDA
1199  F62F  23                        .BYTE   $23     ; CMP
1200  F630  A0                        .BYTE   $A0     ; SBC
1201
1202  F631  D8                MNEMR:  .BYTE   $D8     ; BRK   ; XXXXX000 INSTRS.
1203  F632  62                        .BYTE   $62     ; PHP
1204  F633  5A                        .BYTE   $5A     ; BPL
1205  F634  48                        .BYTE   $48     ; CLC
1206  F635  26                        .BYTE   $26     ; JSR
1207  F636  62                        .BYTE   $62     ; PLP
1208  F637  94                        .BYTE   $94     ; BMI
1209  F638  88                        .BYTE   $88     ; SEC
1210  F639  54                        .BYTE   $54     ; RTI
1211  F63A  44                        .BYTE   $44     ; PHA
1212  F63B  C8                        .BYTE   $C8     ; BVC
1213  F63C  54                        .BYTE   $54     ; CLI
1214  F63D  68                        .BYTE   $68     ; RTS
1215  F63E  44                        .BYTE   $44     ; PLA
1216  F63F  E8                        .BYTE   $E8     ; BVS
1217  F640  94                        .BYTE   $94     ; SEI
1218  F641  00                        .BYTE   $0      ; ???
1219  F642  B4                        .BYTE   $B4     ; DEX
1220  F643  08                        .BYTE   $8      ; BCC
1221  F644  84                        .BYTE   $84     ; TYA
1222  F645  74                        .BYTE   $74     ; LDY
1223  F646  B4                        .BYTE   $B4     ; TAY
1224  F647  28                        .BYTE   $28     ; BCS
1225  F648  6E                        .BYTE   $6E     ; CLV
1226  F649  74                        .BYTE   $74     ; CPY
1227  F64A  F4                        .BYTE   $F4     ; INY
1228  F64B  CC                        .BYTE   $CC     ; BNE
1229  F64C  4A                        .BYTE   $4A     ; CLD
1230  F64D  72                        .BYTE   $72     ; CPX
1231  F64E  F2                        .BYTE   $F2     ; INX
1232  F64F  A4                        .BYTE   $A4     ; BEQ
1233  F650  8A                        .BYTE   $8A     ; SED
1234  F651  00                        .BYTE   $0      ; ???   ; XXXYY100 INSTRS.
1235  F652  AA                        .BYTE   $AA     ; BIT
1236  F653  A2                        .BYTE   $A2     ; JMP
1237  F654  A2                        .BYTE   $A2     ; JMP
1238  F655  74                        .BYTE   $74     ; STY
1239  F656  74                        .BYTE   $74     ; LDY
1240  F657  74                        .BYTE   $74     ; CPY
1241  F658  72                        .BYTE   $72     ; CPX
1242  F659  44                        .BYTE   $44     ; TXA   ; 1XXX1010 INSTRS.
1243  F65A  68                        .BYTE   $68     ; TXS
```

```
1244  F65B  B2                    .BYTE   $B2     ; TAX
1245  F65C  32                    .BYTE   $32     ; TSX
1246  F65D  B2                    .BYTE   $B2     ; DEX
1247  F65E  00                    .BYTE   $0      ; ???
1248  F65F  22                    .BYTE   $22     ; NOP
1249  F660  00                    .BYTE   $0      ; ???
1250  F661  1A                    .BYTE   $1A     ; ASL   ; XXXYYY10 INSTRS.
1251  F662  1A                    .BYTE   $1A     ; ROL
1252  F663  26                    .BYTE   $26     ; LSR
1253  F664  26                    .BYTE   $26     ; ROR
1254  F665  72                    .BYTE   $72     ; STX
1255  F666  72                    .BYTE   $72     ; LDX
1256  F667  88                    .BYTE   $88     ; DEC
1257  F668  C8                    .BYTE   $C8     ; INC
1258  F669  C4                    .BYTE   $C4     ; ORA   ; XXXYYY01 INSTRS.
1259  F66A  CA                    .BYTE   $CA     ; AND
1260  F66B  26                    .BYTE   $26     ; EOR
1261  F66C  48                    .BYTE   $48     ; ADC
1262  F66D  44                    .BYTE   $44     ; STA
1263  F66E  44                    .BYTE   $44     ; LDA
1264  F66F  A2                    .BYTE   $A2     ; CMP
1265  F670  C8                    .BYTE   $C8     ; SBC
1266
1267                      ;       .LIST
1268
```

```
1270                            .SBTTL  Disk Boot
1271
1272                            ; Disk Boot for MTU K-1013 Disk Board
1273                            ; NEC-765 Disk controler chip
1274                            ; (May 30, 1980)
1275
1276                            ;       KSPROUL VERSION         Copyright by M.T.U. 1980
1277
1278                            ;       THIS PROGRAM READS TRACK 0, SECTOR 0 INTO THE HIGHEST
1279                            ;       256 BYTES OF SYSTEM RAM, EXTRACTS THE SYSTEM ENTRY POINT,
1280                            ;       FINAL SECTOR, AND DMA DESTINATION CODE FOR THE SYSTEM
1281                            ;       MEMORY IMAGE, AND INSTALLS THESE PARAMETERS INTO THE
1282                            ;       COMMAND STRING OR SAVES THEM, AS APPROPRIATE. A
1283                            ;       MULTI-SECTOR READ IS THEN PERFORMED TO LOAD THE OPERATING
1284                            ;       SYSTEM IMAGE INTO RAM. FINALLY CONTROL IS PASSED TO THE
1285                            ;       ENTRY POINT READ FROM THE FIRST SECTOR, ANY ERROR STARTS
1286                            ;       THE PROCESS ALL OVER.
1287
1288                                                            ; $F8 CODE FOR SYSRAM+$1E00
1289       00F8                 DMACL   =       $F8             ; DMA CODE FOR LAST PAGE OF RAM
1290       DE00                 BB      =       SYSRAM+$1E00    ; XX00 = NOMINAL ADDR OF DMA BUFFER
1291
1292
1293
1294  F671                      DBOOT:
1295  F671  A2  FF                      LDX     #$FF            ; INIT THE STACK POINTER
1296  F673  9A                          TXS
1297  F674  A2  13                      LDX     #BTCRS-BTCMDS   ; MOVE THE COMMAND STRINGS
1298  F676  BD  17  F7          BOOT1:  LDA     DATA,X
1299  F679  95  00                      STA     BTCMDS,X
1300  F67B  CA                          DEX
1301  F67C  10  F8                      BPL     BOOT1
1302                                                            ; START OF BOOT PGM
1303  F67E  20  9D  F6                  JSR     BTRST           ; RESET DRIVE 0
1304  F681  A9  00                      LDA     #0
1305  F683  85  0E                      STA     BTRC+5          ; SETUP FOR SECTOR 0
1306  F685  A9  F8                      LDA     #DMACL          ; DMA CODE FOR INITIAL SECTOR READ
1307  F687  20  C7  F6                  JSR     BTRD            ; READ SECTOR 0 TRACK 0 TO LAST RAM PAGE
1308  F68A  E6  0E                      INC     BTRC+5          ; BUMP POINTER
1309  F68C  A2  03                      LDX     #3
1310  F68E  BD  3C  DE          BOOT2:  LDA     BB+60,X         ; *** GET ENTRY-1, DMA CODE, FINAL SECTOR
1311  F691  48                          PHA                     ; ONTO STACK
1312  F692  CA                          DEX
1313  F693  10  F9                      BPL     BOOT2
1314  F695  68                          PLA
1315  F696  85  10                      STA     BTRC+7          ; INSTALL FINAL SECTOR NO. INTO STRING
1316  F698  68                          PLA
1317  F699  20  C7  F6                  JSR     BTRD            ; READ SECTORS INTO MEMORY
1318  F69C  60                          RTS
1319
1320
```

```
                                .SBTTL  Disk Boot - Subroutines

F69D  AD  E8  DF    BTRST:  LDA     FDCIRQ          ; *** INTERRUPT STATUS REG
F6A0  30  03                BMI     BTRST2
F6A2  20  BE  F6            JSR     BTIS3           ; SENSE INTERRUPT STATUS
F6A5  A2  00        BTRST2: LDX     #BTSPC-BTCMDS   ; INDEX FOR "SPECIFY" FDC COMMAND
F6A7  20  EF  F6            JSR     BTCPH           ; ISSUE COMMAND
F6AA  A2  04                LDX     #BTRCL-BTCMDS   ; INDEX FOR "RECALIBRATE" FDC COMMAND
F6AC  20  EF  F6            JSR     BTCPH           ; ISSUE COMMAND
F6AF  20  B9  F6            JSR     BTIS            ; WAIT  ISSUE SENSE IRQ STAUS
F6B2  A5  13                LDA     BTCRS           ; ST-0
F6B4  29  D8                AND     #$D8
F6B6  D0  B9                BNE     DBOOT           ; IF ERROR, START OVER
F6B8  60                    RTS

                            ; BTIS - Issue Interrupt Status Command when Ready

F6B9  AD  E8  DF    BTIS:   LDA     FDCIRQ          ; *** INTERRUPT STATUS REG
F6BC  30  FB                BMI     BTIS            ; WAIT TILL READY

                            ;       *** NOTE...THIS IS ALTERNATE ENTRY POINT...***

F6BE  A2  07        BTIS3:  LDX     #BTSIS-BTCMDS   ; INDEX FOR "SENSE INTERRUPT STATUS"
F6C0  20  EF  F6            JSR     BTCPH           ; ISSUE COMMAND
F6C3  20  04  F7            JSR     BTRPH           ; RESULT PHASE
F6C6  60                    RTS

                            ; BTRD - Read Sector(s)

                            ;       ON CALL... A = DMA ADDRESS CODE FOR SECTOR TO READ,
                            ;       FORMAT STRING IS FULLY DEFINED.
F6C7  A2  01        BTRD:   LDX     #1
F6C9  8E  E8  DF            STX     FDCHWC          ; *** DMA MODE SELECT
F6CC  8D  EA  DF            STA     FDCDMA          ; *** DMA ADDRESS CODE SELECT
F6CF  A2  09                LDX     #BTRC-BTCMDS    ; INDEX FOR "READ" COMMAND
F6D1  20  EF  F6            JSR     BTCPH           ; ISSUE COMMAND
F6D4  AD  E8  DF    BTRD3:  LDA     FDCIRQ          ; *** INTERRUPT STATUS REG
F6D7  30  FB                BMI     BTRD3
F6D9  20  04  F7            JSR     BTRPH           ; RESTORE PHASE
F6DC  A5  13                LDA     BTCRS           ; ST-0
F6DE  29  D8                AND     #$D8
F6E0  F0  0C                BEQ     BTRD7
F6E2  C9  40                CMP     #$40            ; "ABNORMAL TERM" BIT
F6E4  D0  8B                BNE     DBOOT           ; SHOULD BE SET FOR NORMAL READ
                                                    ; IF ERROR START OVER
F6E6  A5  14                LDA     BTCRS+1         ; ST-1
F6E8  29  B7                AND     #$B7
F6EA  C9  80                CMP     #$80            ; SHOULD HAVE "END-CYL" SET+ABNORM TERM
F6EC  D0  83                BNE     DBOOT           ; ELSE START OVER
F6EE  60            BTRD7:  RTS
```

```
1377
1378
1379                            ; BTCPH - Command Phase Execution
1380
1381
1382                            ;       ON CALL ... X INDEXES START OF FDC COMMAND STRING RELATIVE
1383                            ;       TO BTCMDS.
1384
1385  F6EF  B4  00          BTCPH:  LDY     BTCMDS,X        ; FETCH NO. OF BYTES IN STRING
1386  F6F1  E8              BTCPH3: INX
1387  F6F2  B5  00                  LDA     BTCMDS,X        ; FETCH NEXT BYTE OF COMMAND STRING
1388  F6F4  20  FE  F6              JSR     BTWMS           ; WAIT FOR MAIN STAT REG
1389  F6F7  8D  EF  DF              STA     FDCDR           ; *** OUTPUT TO FDC
1390  F6FA  88                      DEY
1391  F6FB  D0  F4                  BNE     BTCPH3          ; REPEAT TILL DONE
1392  F6FD  60                      RTS
1393
1394                            ; BTWMS - Wait for MAIN Stat Reg REQ Master:
1395
1396
1397  F6FE  2C  EE  DF      BTWMS:  BIT     FDCMSR          ; *** MAIN STATUS REGISTER
1398  F701  10  FB                  BPL     BTWMS           ; WAIT FOR 'REQ FOR MASTER' = TRUE
1399  F703  60                      RTS
1400
1401
1402  F704  A2  00          BTRPH:  LDX     #0
1403  F706  20  FE  F6              JSR     BTWMS           ; WAIT FOR MAIN
1404  F709  AD  EF  DF      BTRPH2: LDA     FDCDR           ; READ BYTE OUT OF FDC
1405  F70C  95  13                  STA     BTCRS,X         ; SAVE
1406  F70E  E8                      INX
1407  F70F  AD  EE  DF              LDA     FDCMSR          ; MAIN STAUS REGISTER
1408  F712  29  10                  AND     #$10
1409  F714  D0  F3                  BNE     BTRPH2          ; BRANCH IF NOT DONE
1410  F716  60                      RTS
1411
1412
```

```
1414
1415  F717                    DATA:                           ; FDC CMD DATA
1416  F717  03                       .BYTE   3       ; #     ; 3 BYTES IN "SPECIFY" COMMAND STRING
1417  F718  03                       .BYTE   $03     ; CMD   ; SPECIFY CMD
1418  F719  AF                       .BYTE   $AF     ; SRT   ; SEEK = 15 MS, HD UNL =  240 MS
1419  F71A  20                       .BYTE   $20     ; HLT   ; HD LD = 32 MS, DMA MODE
1420
1421  F71B  02                       .BYTE   2       ; #     ; 2 BYTES IN "RECALIBRATE"
1422  F71C  07                       .BYTE   $07     ; CMD   ; RECALIBRATE CMD
1423  F71D  00                       .BYTE   00      ; DRV   ; DRIVE
1424
1425  F71E  01                       .BYTE   1       ; #     ; 1 BYTE IN "SENSE INT STAT" CMD
1426  F71F  08                       .BYTE   $08     ; CMD   ; SENSE INT STATUS CMD
1427
1428  F720  09                       .BYTE   9       ; #     ; 9 BYTES IN "READ" COMMAND
1429  F721  46                       .BYTE   $46     ; CMD   ; READ CODE, 1 SIDED , MFM
1430  F722  00                       .BYTE   00      ; DRV   ; DRIVE
1431  F723  00                       .BYTE   00      ; C     ; TRACK
1432  F724  00                       .BYTE   00      ; H     ; HEAD
1433  F725  00                       .BYTE   00      ; R     ; ** STARTING SECTOR
1434  F726  01                       .BYTE   $01     ; N     ; CODE, 256 BYTES/SEC
1435  F727  00                       .BYTE   00      ; EOT   ; ** FINAL SECTOR
1436  F728  0E                       .BYTE   $0E     ; GPL   ; GAP, 26 SECS, MFM
1437  F729  FF                       .BYTE   $FF     ; DTL   ; DATA LENGTH: = $FF SINCE N <> 0
1438
1439
```

```
1441
1442        0013                DSKSTS  =       BTCRS           ; BOT COMMAND RESULTS
1443
1444                            ; Q                             ; DISK STATUS COMMAND
1445  F72A  A9  00              DSTAT:  LDA     #0              ; DRIVE #0
1446  F72C  20  3F  F7                  JSR     CHKDRV
1447  F72F  A9  01                      LDA     #1              ; DRIVE #1
1448  F731  20  3F  F7                  JSR     CHKDRV
1449  F734  A9  02                      LDA     #2              ; DRIVE #2
1450  F736  20  3F  F7                  JSR     CHKDRV
1451  F739  A9  03                      LDA     #3              ; DRIVE #3
1452  F73B  20  3F  F7                  JSR     CHKDRV
1453  F73E  60                          RTS
1454
1455  F73F  48                  CHKDRV: PHA                     ; SAVE DRIVE #
1456  F740  20  B7  F0                  JSR     CRLF
1457  F743  68                          PLA                     ; RESTORE BUT LEAVE ON STACK
1458  F744  48                          PHA
1459  F745  20  8E  F7                  JSR     SDRVST          ; SENCE DRIVE STATUS
1460  F748  A9  23                      LDA     #'#
1461  F74A  20  12  F4                  JSR     CTYOUT
1462  F74D  68                          PLA                     ; RESTORE DRIVE #
1463  F74E  09  30                      ORA     #$30            ; CONVERT TO ASCII #
1464  F750  20  12  F4                  JSR     CTYOUT
1465  F753  20  09  F0                  JSR     CTYSPA
1466  F756  20  09  F0                  JSR     CTYSPA
1467  F759  A5  13                      LDA     DSKSTS+0
1468  F75B  85  F6                      STA     COUNT           ; USE COUNT AS TEMP FOR SHIFTING
1469  F75D  20  F8  F3                  JSR     PRTBYT          ; PRINT IT
1470  F760  20  09  F0                  JSR     CTYSPA
1471  F763  20  09  F0                  JSR     CTYSPA
1472  F766  A2  46                      LDX     #'F             ; INDICATE DRIVE FALUT
1473  F768  20  80  F7                  JSR     CHKSUB
1474  F76B  A2  57                      LDX     #'W             ; INDICATE DRIVE WRITE PROTECTED
1475  F76D  20  80  F7                  JSR     CHKSUB
1476  F770  A2  52                      LDX     #'R             ; INDICATE DRIVE READY
1477  F772  20  80  F7                  JSR     CHKSUB
1478  F775  A2  30                      LDX     #'0             ; INDICATE DRIVE AT TRACK ZERO
1479  F777  20  80  F7                  JSR     CHKSUB
1480  F77A  A2  32                      LDX     #'2             ; INDICATE DRIVE DOUBLE SIDED
1481  F77C  20  80  F7                  JSR     CHKSUB
1482  F77F  60                          RTS
1483
1484  F780                      CHKSUB:                         ; CHECK DRIVE SUBROUTINE
1485  F780  A9  20                      LDA     #$20            ; SPACE
1486  F782  06  F6                      ASL     COUNT           ; SHIFT TO TEST BIT IN QUESTION
1487  F784  90  01                      BCC     CHKSB1          ; IF CONDITION NOT MEET, PRINT A SPACE
1488  F786  8A                          TXA                     ; OTHERWISE, PUT THE CHAR IN ACC
1489  F787  20  12  F4          CHKSB1: JSR     CTYOUT          ; AND PRINT IT
1490  F78A  20  09  F0                  JSR     CTYSPA
1491  F78D  60                          RTS
1492
1493
1494  F78E                      SDRVST:                         ; SENSE DRIVE STATUS
1495  F78E  48                          PHA                     ; SAVE THE DRIVE #
```

```
1496  F78F  20  FE  F6          JSR   BTWMS          ; BOOT-WAIT FOR MAIN STATUS REGISTER
1497  F792  A9  04              LDA   #$04           ; SENSE DRIVE STATUS COMMAND
1498  F794  8D  EF  DF          STA   FDCDR
1499  F797  20  FE  F6          JSR   BTWMS          ; BOOT-WAIT FOR MAIN STATUS REGISTER
1500  F79A  68                   PLA                  ; RESTORE DRIVE #
1501  F79B  8D  EF  DF          STA   FDCDR
1502  F79E  20  04  F7          JSR   BTRPH
1503  F7A1  60                   RTS
1504
1505                       ; E                        ; ENABLE WRITE TO SYSTEM RAM
1506  F7A2  A9  00         ENABLE: LDA   #$00         ; UN-WRITE-PROTECT SYSRAM
1507  F7A4  8D  E8  DF          STA   FDCHWC         ; Put in Hardware Control Register
1508  F7A7  A9  2D               LDA   #'-
1509  F7A9  4C  12  F4          JMP   CTYOUT         ; USE CTYOUT's RTS
1510
1511
1512                                                  ; 22-MAY-81
1513                       ; P                        ; PROTECT WRITE PROTECTABLE RAM
1514  F7AC  A9  02         PROTCT: LDA   #$02         ; WRITE PROTECT SYSRAM (BIT-1)
1515  F7AE  8D  E8  DF          STA   FDCHWC         ; Put in Hardware Control Register
1516  F7B1  A9  2B               LDA   #'+
1517  F7B3  4C  12  F4          JMP   CTYOUT         ; USE CTYOUT's RTS
1518
1519                                                  ; 27-JAN-81
1520                       ; W                        ; CODOS WARM START
1521  F7B6                 JCODOS:                    ; JUMP TO CODOS
1522  F7B6  A9  02               LDA   #$02           ; WRITE PROTECT SYSRAM (BIT-1)
1523  F7B8  8D  E8  DF          STA   FDCHWC         ; Put in Hardware Control Register
1524  F7BB  4C  03  C6          JMP   CODOS          ; CODOS WAR-START
1525
1526
```

```
1528                            .SBTTL  Interrupt Processor
1529                            ; NMI/IRQ/BRK                   ($17FA - $17FF FOR INDIRECT JMP RAM)
1530
1531        FFAO                        .=      $FFAO           ; $FFAO (PUT AT TOP of 4th K by VECTORS)
1532                                                            ; $PCH,PCL A X Y P S
1533  FFAO  85  FB              INTRPT: STA     ACC             ; SAVE ACC
1534  FFA2  86  FD                      STX     XREG            ; SAVE X
1535  FFA4  84  FC                      STY     YREG            ; SAVE Y
1536  FFA6  68                          PLA
1537  FFA7  85  F9                      STA     PREG            ; PROCESSOR STATUS (CONDITION CODES)
1538  FFA9  68                          PLA
1539  FFAA  85  F7                      STA     PCL             ; Program Counter Low
1540  FFAC  85  FO                      STA     ADDR1
1541  FFAE  68                          PLA
1542  FFAF  85  F8                      STA     PCH             ; Program Counter High
1543  FFB1  85  F1                      STA     ADDR1+1
1544  FFB3  BA                          TSX
1545  FFB4  86  FA                      STX     SPUSER          ; STACK POINTER (before Interrupt)
1546
1547  FFB6  20  ED  FO                  JSR     PRTADR          ; $PCH,PCL
1548  FFB9  A5  FB                      LDA     ACC             ; A
1549  FFBB  20  1F  F4                  JSR     BYTSPA
1550  FFBE  A5  FD                      LDA     XREG            ; X
1551  FFCO  20  1F  F4                  JSR     BYTSPA
1552  FFC3  A5  FC                      LDA     YREG            ; Y
1553  FFC5  20  1F  F4                  JSR     BYTSPA
1554  FFC8  A5  F9                      LDA     PREG            ; P  (N V - B D I Z C)
1555  FFCA  20  1F  F4                  JSR     BYTSPA
1556  FFCD  A5  FA                      LDA     SPUSER          ; S
1557  FFCF  20  F8  F3                  JSR     PRTBYT
1558  FFD2  4C  4F  FO                  JMP     RESET2
1559
1560
1561                            ; R
1562  FFD5  A9  AO              RSTINI: LDA     #INTRPT&$FF     ; LOW  BYTE  of ADDR of REGISTER DUMP
1563  FFD7  8D  FA  DD                  STA     NMIV
1564  FFDA  8D  FE  DD                  STA     IRQV
1565  FFDD  A9  FF                      LDA     #INTRPT^        ; HIGH BYTE of ADDR of REGISTER DUMP
1566  FFDF  8D  FB  DD                  STA     NMIV+1
1567  FFE2  8D  FF  DD                  STA     IRQV+1
1568  FFE5  OO                          BRK                     ; FORCE REGISTER DUMP
1569
1570  FFE6  OO  OO  OO                  .WORD   O,O,O,O         ; TAKE UP SPACE
1571  FFEE  OO  OO  OO                  .WORD   O,O,O
1572
1573  FFF4  6C  FA  DD  NMI:    JMP     (NMIV)          ; JMP ($17FA)
1574  FFF7  6C  FE  DD  IRQ:    JMP     (IRQV)          ; JMP ($17FE)
1575
1576                                                    ; VECTORS
1577  FFFA  F4  FF                      .ADDR   NMI             ; NMI
1578  FFFC  OO  FO                      .ADDR   RESET           ; RESET
1579  FFFE  F7  FF                      .ADDR   IRQ             ; IRQ/BRK
1584
1585        FOOO                        .END    RESET
```

KSMON.M65 Cross reference table

```
ABORT   F469    794     800     811#
ACC     00FB     69#    1533*   1548
ADDRE   00F4     59#     238*    240*    267     269
ADDR1   00F0     57#     231*    233*    249     251     264*    266     268     278     295     311*    332
                341     348     351*    352     354*    393     414*    415     437*    438     472     498
                502*    503     507*    508     512*    513     517*    518     522*    523     528     536*
                548*    551*    558     559*    560     565     572     577*    578     582*    604     609
               1540*   1543*
ADDR2   00F2     58#     257*    259*    290*    292*    296*    549*    554*    579     581     600*    602*
                605     614     616     618
AMODE   F593    890     1030#
ASCDAT  F19E    341#    345
BASE  = EF80     86#
BB    = DE00   1290#   1310
BLKCMP  F34A    195     593#
BLKCP1  F366    604#    624
BLKCP2  F38B    606     621#
BNKREG= EFA0     86#
BOOT1   F676   1298#   1301
BOOT2   F68E   1310#   1313
BTCMDS  0000    101#   1297    1299*   1328    1330    1347    1361    1385    1387
BTCPH   F6EF   1329    1331    1348    1362    1385#
BTCPH3  F6F1   1386#   1391
BTCRS   0013    106#   1297    1333    1366    1372    1405*   1442
BTIS    F6B9   1332    1342#   1343
BTIS3   F6BE   1327    1347#
BTRC    0009    105#   1305*   1308*   1315*   1361
BTRCL   0004    103#   1330
BTRD    F6C7   1307    1317    1358#
BTRD3   F6D4   1363#   1364
BTRD7   F6EE   1368    1376#
BTRPH   F704   1349    1365    1402#   1502
BTRPH2  F709   1404#   1409
BTRST   F69D   1303    1325#
BTRST2  F6A5   1326    1328#
BTSIS   0007    104#   1347
BTSPC   0000    102#   1328
BTWMS   F6FE   1388    1397#   1398    1403    1496    1499
BUFFER= 00F6     61#    462*    465     473
BYTSPA  F41F    252     333     395     610     617     744#    926     1549    1551    1553    1555
CHAR1   F5E5    973    1119#
CHAR2   F5EB    975    1126#
CHKDRV  F73F   1446    1448    1450    1452    1455#
CHKSB1  F787   1487    1489#
CHKSUB  F780   1473    1475    1477    1479    1481    1484#
CHRHEX  F0E7    230     237     243#    289     307     324     547     553     599
CMD.LP  F059    149     154#    158
CNTRLS  F459    798#    802
CODOS = C603     85#   1524
COUNT   00F6     60#     61     329*    355*    860*    865*   1468*   1486*
CRLF    F0B7    123     215     220#    246     468     564     811     996    1456
CTY   = EF00     87#    134*    137*    643     646     661     664     735     740*
CTYHLF  F39D    116     359     638     643#    645     652     798     868
CTYIN   F397    117     431     638#    680     682
CTYOUT  F412    120     217     221     223     225     227     243     248     284     286     339     347
```

```
KEYDWN  F3B4              128    661#   791
KYBDIN  F425              753#
LENGTH  00C3               52#   904*   927    960    1011
LINE    F185              330#   356
LMNEM   00C0               49#   939*   946*
LOOP    F061              157#   164
MEMDEC  F230              403    405    424#
MEMINC  F22A              407    409    411    422#
MEMORY  F1E4              185    390#
MEMTST  F2AB              191    494#
MEM.OK  F22A              416    421#
MEM1    F1E7              391#   420    423    425
MNEML   F5F1              938   1137#
MNEMR   F631              940   1202#
MNNDX1  F4E8              913#   922
MNNDX2  F4EC              916#   919
MNNDX3  F4F3              911    914    921#
MODE2   F5D7              901   1104#
MTEST   F2FC              193    546#
MTSTOK  F333              561    575#
MTSTO   F310              556#   585
MTST1   F313              558#   576    580
MVDATA  F127              179    281#
MVLOOP  F148              295#   300
NMI     FFF4             1573#  1577
NMIV  = DDFA               88#  1563*  1566*  1573
NONHEX  F3F5              696    700    703    707#
NO.KEY  F3C0              663    668#
PAGE    F181              328#   366
PCADJ   F584              862   1011#
PCADJ3  F587              985   1013#
PCADJ4  F58D             1015   1017#
PCH     00F8               66#   855*   864*   999    1013   1542*
PCL     00F7               65#   857*   863*   878    925    965    1000   1017   1539*
PCNTRL  F44B              124    487    531    570    584    613    790#   951    974    977    998
PCTRL2  F464              792    796    805#
PERID1  F1DE              375    378#
PIA1A = EF04               87#
PIA1B = EF06               87#
PIA2A = EF08               87#
PIA2B = EF0A               87#
PIA3A = EF14               87#
PIA3B = EF16               87#   140*   142*   144*   760*   761    763
PIA4A = EF18               87#
PIA4B = EF1A               87#
PRADR1  F530              958#   979
PRADR2  F538              963#   969
PRADR3  F546              959    961    971#
PRADR4  F558              972    976    978#
PRBLNK  F57B              931    955   1005#
PRBL2   F57D             1006#  1008
PREG    00F9               67#  1537*  1554
PRINT   F1E0              377    379#
PRINTR  F426              130    760#
PRNTAX  F565              992#  1001
```

```
PRNTOP  F4F7              925#    929
PRNTPC  F56C              877     996#
PRNTYX  F564              988     991#
PROMPT  FOAF              154     215#
PROTCT  F7AC              205    1514#
PRTADR  FOED              246#    330     391     447     485     527     608    1547
PRTBYT  F3F8              122     250     529     533     566     568     573     615     619     713#    744     967
                          992     994    1469    1557
PRTMN1  F517              943#    953
PRTMN2  F51B              945#    949
PRTWLP  F429              761#    762
PVCHAR  F1D6              342     374#    397
RELADR  F55C              966     985#
RESET   FOOO              115#   1578    1585
RESET1  FO3O              115     133#
RESET2  FO4F              146#    150     707     816    1558
RMNEM   OOC1               50#    941*    945*
RSTINI  FFD5              209    1562#
RSTV  = DDFC               88#
RTMODE  F4CB              891     896#
RTS1    F592             1018    1020#
SDRVST  F78E             1459    1494#
SEARCH  F26O              189     453#
SPUSER  OOFA               68#   1545*   1556
SRCH    F26D              458#    466
SRCHO   F284              460     469#
SRCH1   F287              471#    477
SRCH2   F289              472#    483
SRCINC  F29O              475#    488
SYSRAM= COOO               81#     85    1290
TABLE   FO7B              157     159     166     168     176#
TEMP    OOFE               73#    688*    691
TEST1   F2B3              498#    539
TEST2   F2FO              524     535#
TMPX    OOFF               74#    325*    335     344     350     465     470*    471
TSTBAD  F2DE              504     509     514     519     526#
TTY   = EF1O               87#    135*    138*    768     771     777     782*
TTYIN   F432              126     768#    770
TTYOUT  F43E              127     776#
TTYOU1  F43F              777#    780
UPRCHR  F3A9              155     399     458     652#    678     689     851
USRRAM= 8000               80#
VIA1  = EF8O               86#
VIA2  = EF9O               86#
XREG    OOFD               71#   1534*   1550
YES     F299              474     480#
YESS    F2AO              481     485#
YREG    OOFC               70#   1535*   1552
      = 0000               46#     47#     49#     50#     51#     52#     55#     57#     58#     59#     60#     65#
                           66#     67#     68#     69#     70#     71#     73#     74#     97#    102#    103#    104#
                          105#    106#    113#    258     264    1531#   1580
```

```
BNKEQU      83#      86
DSKEQU      83#      85
INCDEC     263#     264
IRQNMI      83#      88
K1012       83#      87
```

KSMON.M65 Cross reference table

```
ADC     350   353   729   730   950  1017
AND     647   655   665   704   725   772   886   896   903   906  1334  1367  1373  1408
ASL     684   685   686   687   945   971  1486
BCC     300   315   477   539   624   645   663   738   770   780   881   914   929   972
       1018  1487
BCS     883   891
BEQ     158   160   361   363   401   403   405   409   416   433   435   439   460   474
        481   524   561   606   680   682   794   800   853   870   872   885   911   961
        976  1368
BGE     375   377   407   700   966
BIT    1397
BLT     411   466   654   696   698   703   934
BMI     728   792  1326  1343  1364
BNE     164   258   264   336   345   356   504   509   514   519   576   580   796   802
        866   874   897   919   922   949   953   959   969   979   988  1008  1335  1370
       1375  1391  1409
BPL     483   656   762  1015  1301  1313  1398
BRK    1568
CLC     349   727
CLD     148
CMP     159   267   360   362   374   376   400   402   404   406   408   410   415   432
        434   438   459   473   503   508   513   518   523   560   579   605   653   679
        681   695   697   699   702   726   793   795   799   801   852   869   871   873
        884   964  1369  1374
CPX     465   910   958
CPY     335   344   480   927   933
DEC     264   355   865
DEX     469   952   978  1007  1300  1312
DEY     482   918   921   948   968  1016  1390
EOR     501   558   572
INC     257   259   264   577  1308
INX     161   162   163   464   987  1386  1406
INY     334   343   575   920   928   932   989  1019
JMP     115   116   117   120   121   122   123   124   125   126   127   128   129   130
        131   150   170   217   227   241   244   252   278   366   379   420   423   425
        441   444   446   447   488   585   639   707   745   816   994  1509  1517  1524
       1558  1573  1574
JSR     149   154   155   215   221   223   225   230   232   237   239   243   246   248
        250   277   281   282   284   286   287   289   291   297   298   299   304   305
        307   313   314   324   326   327   330   333   337   339   342   347   358   359
        365   390   391   395   397   398   399   412   417   419   422   424   430   431
        443   445   453   454   456   458   461   463   468   475   476   485   487   494
        495   527   529   531   533   537   538   547   553   564   566   568   570   573
        584   593   594   596   597   599   601   608   610   611   613   615   617   619
        621   622   623   638   652   678   683   689   690   719   721   744   791   798
        806   811   813   815   851   854   856   861   862   868   877   926   931   951
        955   967   974   977   985   992   996   998  1001  1006  1303  1307  1317  1327
       1329  1331  1332  1348  1349  1362  1365  1388  1403  1446  1448  1450  1452  1456
       1459  1461  1464  1465  1466  1469  1470  1471  1473  1475  1477  1479  1481  1489
       1490  1496  1499  1502  1547  1549  1551  1553  1555  1557
LDA     118   133   136   139   141   143   166   168   216   220   222   224   226   229
        236   247   249   251   264   266   268   283   285   288   295   306   323   328
        332   338   341   346   348   352   357   364   378   393   418   440   455   472
        486   498   506   511   516   521   528   530   546   550   552   565   569   578
        581   583   595   598   604   609   612   614   616   618   643   646   661   664
```

KSMON.M65 Cross reference table

```
        1222    1223    1224    1225    1226    1227    1228    1229    1230    1231    1232    1233    1234    1235
        1236    1237    1238    1239    1240    1241    1242    1243    1244    1245    1246    1247    1248    1249
        1250    1251    1252    1253    1254    1255    1256    1257    1258    1259    1260    1261    1262    1263
        1264    1265    1416    1417    1418    1419    1421    1422    1423    1425    1426    1428    1429    1430
        1431    1432    1433    1434    1435    1436    1437
.END    1585
.ENDC   1583
.IFNE   1580
.IFT    1581
.LIST     84
.MCALL    83     263
.NLIST    89      90      91
.PAGE     44      93     152     254     369     427     543     676     848    1413    1440
.SBTTL     1      78     111     173     213     274     320     387     450     491     590     630     711     750
         787     821     983    1027    1134    1270    1322    1528
.TITLE     2
.WORD   1570    1571
```

Errors detected: 0

*,KSMON=KSMON
Run-time: 1 3 0 Seconds
Core used: 10K

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

* * * L P T S P L R u n L o g * * *

```
20:46:59 LPDAT  LPTSPL version 104(16650)       Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit
20:46:59 LPDAT  Job KSMON sequence #1426 on Printer 0 [LOCAL] at 11-Sep-82 20:46:59
20:47:05 LPMSG  Starting File PS:<KSPROUL>KSMON.LST.1
20:50:12 LPMSG  Finished File PS:<KSPROUL>KSMON.LST.1
20:50:12 LPEND  Summary:   49 Pages of Output
20:50:12 LPEND             20 Disk Pages Read
20:50:12 LPEND         25.497 Seconds CPU Time Used
```

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**

END Job KSMON Req #911 for KSPROUL Date 11-Sep-82 20:50:12 Monitor: Rutgers/LCSR DEC-20 (Red), TOPS-20 Monit **END**