Listing 1: *Assembly listing of MON88. The flowchart in figure 2 outlines the general operation of the program.*

```
MCS-86 MACRO ASSEMBLER    VID88


ISIS-II MCS-86 MACRO ASSEMBLER V2.0 ASSEMBLY OF MODULE VID88
OBJECT MODULE PLACED IN :F0:VID88.OBJ
ASSEMBLER INVOKED BY: ASM86 VID88.A86

LOC  OBJ                  LINE     SOURCE

                             1     ;       ********************************************************
                             2     ;       *                                                      *
                             3     ;       *                       M O N 8 8                      *
                             4     ;       *                                                      *
                             5     ;       * A video oriented system monitor for the INTEL 8088  *
                             6     ;       *            written Q1 1980 - revision 0              *
                             7     ;       *            by.. Thomas Woodward Cantrell             *
                             8     ;       *                                                      *
                             9     ;       ********************************************************
                            10     ;
                            11             ASSUME  DS:ABS_0,CS:ABS_0,ES:ABS_0
----                        12     ABS_0   SEGMENT BYTE AT 0
0000                        13     M       LABEL   BYTE
F800                        14             ORG     0F800H
                            15     ;
                            16     ;       ********************************************************
                            17     ;       *                                                      *
                            18     ;       *                    EQUATES FOLLOW                    *
                            19     ;       *                                                      *
                            20     ;       ********************************************************
                            21     ;
F400                        22     VIDBUF  EQU     0F400H          ;video buffer
F450                        23     XY      EQU     VIDBUF+80       ;holder for cursor position
F452                        24     UCFLAG  EQU     XY+2            ;upper/lower case flag
000C                        25     FF      EQU     0CH             ;form feed (clear screen)
000A                        26     LF      EQU     0AH             ;line feed
000D                        27     CR      EQU     0DH             ;carriage return
0008                        28     BS      EQU     08H             ;backspace
```

Listing 1 continued on page 348

where TOPMEM is the address of the top of memory.

Initialization: I/O initialization is done in the INIT section of the monitor (see listing 1, starting at line 76). Starting at hexadecimal F81D, I initialize the Tarbell cassette interface and TDL Video Interface. Replace the section of code from hexadecimal F81D to F828 to suit your I/O needs.

I/O Drivers

MON88 currently uses the following environment-dependent I/O routines (their hexadecimal addresses are given in parentheses):

- KEYIN (F90F)—Reads a byte from the console keyboard, strips off the parity bit, and returns the character in the AL accumulator.
- KEYSTAT (F922)—Reads the console keyboard's status and returns AL=0 if a key has not been pressed and AL = hexadecimal FF if a key has been pressed.
- CIN (F955)—Reads a byte from a mass-storage device (Tarbell cassette, in my case) and returns the byte in the AL accumulator.
- COUT (F964)—Writes the byte contained in the AL accumulator to the mass-storage device.
- CSTART (FB60)—Sets up the mass-storage device for a write operation. For the Tarbell interface, a start byte and a sync byte are required. Replace this code as necessary for your device.
- READINIT (FB9D)—Sets up the mass-storage device for a read operation. Replace the relevant code as necessary.
- PUTSYNC (FBBF)—Outputs a stream of sync bytes to my cassette. This allows calibrating the interface. If your device has a similar feature, modify the PUTSYNC routine accordingly. If not needed, the whole P (PUTSYNC) command can be removed.
- VIDOUT (FCDA)—This routine outputs the character in the AL accumulator to the console display device. In my case, I converted an 8080 version of the video driver to 8088 code using Intel's CONV86 program. Using the code converter, it took only an hour or so to get the driver up and running. I will rewrite it as necessary to reduce the amount of memory used by MON88.

Adding or Removing Commands

All commands are referenced through CTABLE (Command Jump Table) located at hexadecimal F8B8. Note that the commands are arranged in alphabetical order, A thru Z. To remove a command, simply replace its reference in CTABLE with ERR. For example, to remove the K command (uppercase/lowercase toggle), change:

```
F8CC    DW    KTOGGLE
```

to

```
F8CC    DW    ERR
```

then remove the KTOGGLE code (hexadecimal FCD1 to FCD9).

Similarly, to add a special memory test (for example) and call it using the letter T, first write the code (for example, starting label TESTMEM) for the command,

Text continued on page 360

Listing 1 continued:

```
E2E2                    29      KSTAT   EQU     0E2E2H          ;keyboard status port
E3E3                    30      KDATA   EQU     0E3E3H          ;keyboard data port
6E6E                    31      CSTAT   EQU     6E6EH           ;Tarbell status port
6F6F                    32      CDATA   EQU     6F6FH           ;Tarbell data port
F7FF                    33      STACKP  EQU     0F7FFH          ;Stack address
0003                    34      CTLC    EQU     03H             ;ascii ctl-c
0004                    35      CTLD    EQU     04H             ;ascii ctl-d
0013                    36      CTLS    EQU     13H             ;ascii ctl-s
0011                    37      CTLQ    EQU     11H             ;ascii ctl-q
0000                    38      FALSE   EQU     0
00FF                    39      TRUE    EQU     0FFH
                        40 +1   $EJECT


MCS-86 MACRO ASSEMBLER  VID88


LOC  OBJ                LINE    SOURCE

                          41    ;
                          42    ;       ******************************************************
                          43    ;       *                                                    *
                          44    ;       *                  JUMP  TABLE                       *
                          45    ;       *                                                    *
                          46    ;       ******************************************************
                          47    ;
                          48    ;
F800 E80D90               49            JMP     INIT    ;RESETS STACK, SEGMENT REGISTERS, CASSETTE INTERFACE
                          50    ;                       ;ALSO PRINTS SIGN-ON MESSAGE
                          51    ;
F803 E83090               52            JMP     START   ;'WARM START'- REGISTERS NOT INITIALIZED
                          53    ;
F806 E91901               54            JMP     KEYSTAT ;RETURNS [AL]=0 IF NO KEYPRESS PENDING.  ELSE [AL]=0FFH
                          55    ;
F809 E9E000               56            JMP     CONIN   ;WAITS FOR KEYPRESS.  RETURNS [AL]=CHAR AND PRINTS IT.
                          57    ;
F80C E9CB04               58            JMP     VIDOUT  ;PRINTS CHAR IN AL ON CONSOLE
                          59    ;
                          60    ;
                          61    ;       ******************************************************
                          62    ;       *                                                    *
                          63    ;       *          I N I T I A L I Z A T I O N               *
                          64    ;       *                                                    *
                          65    ;       ******************************************************
                          66    ;
                          67    ;
F80F FC                   68    INIT:   CLD                             ;direction flag points 'up'
F810 FA                   69            CLI                             ;disable interrupts
F811 8CC8                 70            MOV     AX,CS                   ;initialize
F813 8ED8                 71            MOV     DS,AX                   ;  segment
F815 8EC0                 72            MOV     ES,AX                   ;  registers
F817 8ED0                 73            MOV     SS,AX                   ;  and set
F819 BCFFF7               74            MOV     SP,STACKP               ;  stack pointer
F81C FB                   75            STI                             ;enable interrupts
F81D B010                 76            MOV     AL,10H                  ;Reset Cassette
F81F BA6E6E               77            MOV     DX,CSTAT
F822 EE                   78            OUT     DX,AL                   ;  Interface
F823 BAE0E0               79            MOV     DX,0E0E0H               ;Reset Video
F826 B088                 80            MOV     AL,88H                  ;  Interface
F828 EE                   81            OUT     DX,AL                   ;Inverse video w/cursor
F829 C60652F400           82            MOV  BYTE PTR [UCFLAG],0        ;0=lower case, FFH=U/C only
                          83
F82E BE5AF890             84            MOV     SI,OFFSET SIGNON        ;get sign on message
F832 E86301               85            CALL    PRINTMESS               ;and print it
                          86 +1   $EJECT


MCS-86 MACRO ASSEMBLER  VID88


LOC  OBJ                LINE    SOURCE

                          87    ;
                          88    ;       ******************************************************
                          89    ;       *                                                    *
                          90    ;       *          W O R K I N G   L O O P                   *
                          91    ;       *                                                    *
                          92    ;       ******************************************************
                          93    ;
F835 E83D01               94    START:  CALL    CRLF                    ;print CRLF
F838 B03E                 95            MOV     AL,'>'                  ;and prompt
F83A E89D04               96            CALL    VIDOUT
                          97    ;
F83D                      98    MAINLOOP:
F83D B400                 99            MOV     AH,0                    ;clear AH
F83F E8AA00              100            CALL    CONIN                   ;get a command
F842 3C41                101            CMP     AL,'A'                  ;check range for
F844 72EF                102            JB      START                   ;     A
F846 3C5A                103            CMP     AL,'Z'                  ;   thru
F848 7FEB                104            JG      START                   ;     Z
F84A 2C41                105            SUB     AL,'A'                  ;calculate offset
```

```
F84C D0E0                106             SHL     AL,1
F84E 05B8F890            107             ADD     AX,OFFSET CTABLE        ;and multiply by 2
F852 8BD8                108             MOV     BX,AX
F854 8B07                109             MOV     AX,WORD PTR [BX]
F856 FFD0                110             CALL    AX
F858 EBDB                111             JMP     START                   ;go do it
                         112 +1  $EJECT                                   ;start over


MCS-86 MACRO ASSEMBLER    VID88


LOC  OBJ                 LINE    SOURCE

                          113    ;
                          114    ;       ***********************************************
                          115    ;       *                                             *
                          116    ;       *              M E S S A G E S                *
                          117    ;       *                                             *
                          118    ;       ***********************************************
                          119    ;
F85A 0C                   120    SIGNON  DB      0CH
F85B 38303838204D6F       121            DB      '8088 Monitor <rev. 0>'
     6E69746F72203C
     7265762E20303E
F870 00                   122    DBYTE   DB      0                       ;dummy byte
                          123    ;
F871 41444452204D20       124    COMHEAD DB      'ADDR M  T    DIFF
     20542020202044
     494646202020
F885 00                   125            DB      0
                          126    ;
F886 53554D20204449       127    MHEAD   DB      'SUM  DIFF'
     4646
F88F 00                   128            DB      0
                          129    ;
F890 53524320204D20       130    VHEAD   DB      'SRC  M   DEST M    DIFF'
     20204445535420
     4D202020204449
     4646
F8A7 00                   131            DB      0
                          132    ;
F8A8 4C454E47544820       133    CHEAD   DB      'LENGTH (HEX) = '
     2848455829203D
     20
F8B7 00                   134            DB      0
                          135    ;
                          136 +1  $EJECT


MCS-86 MACRO ASSEMBLER    VID88


LOC  OBJ                 LINE    SOURCE

                          137    ;
                          138    ;       ***************************************************
                          139    ;       *                                                 *
                          140    ;       *       C O M M A N D   J U M P   T A B L E       *
                          141    ;       *                                                 *
                          142    ;       ***************************************************
                          143    ;
F8B8 B0FC                 144    CTABLE  DW      AENTER  ;ENTER ASCII TEXT INTO MEMORY
F8BA A7F9                 145            DW      ERR     ;B
F8BC D2FB                 146            DW      COMPARE ;COMPARE CASSETTE INPUT WITH MEMORY
F8BE 00FB                 147            DW      DUMP    ;DISPLAY MEMORY
F8C0 78FC                 148            DW      ESUBST  ;ENTER HEX DATA INTO MEMORY
F8C2 7EFA                 149            DW      FILL    ;FILL MEMORY WITH A CONSTANT
F8C4 4AFB                 150            DW      GOTO    ;GO TO & EXECUTE A USER PROGRAM
F8C6 1AFC                 151            DW      HEXMATH ;COMPUTE SUM AND DIFFERENCE OF HEX #'S
F8C8 2CFB                 152            DW      INPUT   ;INPUT FROM A PORT
F8CA A7F9                 153            DW      ERR     ;J
F8CC D1FC                 154            DW      KTOGGLE ;TOGGLE KEYBOARD UPPER/LOWER CASE FLAG
F8CE A7F9                 155            DW      ERR     ;L
F8D0 E7FA                 156            DW      MOVE    ;MOVE MEMORY
F8D2 3BFC                 157            DW      NTEST   ;NON DESTRUCTIVE MEMORY TEST
F8D4 3FFB                 158            DW      OUTPUT  ;OUTPUT TO A PORT
F8D6 BFFB                 159            DW      PUTSYNC ;OUTPUT CONTINUOUS SYNC STREAM TO CASSETTE
F8D8 A7F9                 160            DW      ERR     ;Q
F8DA 82FB                 161            DW      READ    ;READ FROM CASSETTE
F8DC A7F9                 162            DW      ERR     ;S
F8DE A7F9                 163            DW      ERR     ;T
F8E0 A7F9                 164            DW      ERR     ;U
F8E2 8DFA                 165            DW      VERIFY  ;VERIFY EQUALITY OF TWO MEMORY BLOCKS
F8E4 4FFB                 166            DW      CWRITE  ;WRITE TO CASSETTE
F8E6 A7F9                 167            DW      ERR     ;X
F8E8 A7F9                 168            DW      ERR     ;Y
F8EA A7F9                 169            DW      ERR     ;Z
                          170 +1  $EJECT
```

Listing 1 continued on page 350

Listing 1 continued:

```
MCS-86 MACRO ASSEMBLER     VID88

LOC  OBJ                   LINE     SOURCE

                            171     ;
                            172     ;
                            173     ;       ************************************************
                            174     ;       *                                              *
                            175     ;       *          U T I L I T Y   R O U T I N E S     *
                            176     ;       *                      and                     *
                            177     ;       *      I / O   D E V I C E   H A N D L E R S   *
                            178     ;       *             (except  video  driver)          *
                            179     ;       *                                              *
                            180     ;       ************************************************
                            181     ;
                            182     ;
F8EC E82000                 183     CONIN:  CALL    KEYIN           ;get a keyboard character
F8EF 50                     184             PUSH    AX
F8F0 A052F4                 185             MOV     AL,BYTE PTR [UCFLAG]    ;check for case conversion
F8F3 0AC0                   186             OR      AL,AL           ;0?
F8F5 7405                   187             JZ      CONNEXT         ;YES..no conversion
F8F7 58                     188             POP     AX              ;restore character
F8F8 E80900                 189             CALL    UCCHEK          ;convert to UC
F8FB 50                     190             PUSH    AX
F8FC 58                     191     CONNEXT:POP     AX
F8FD E8DA03                 192             CALL    VIDOUT          ;and echo it on console
F900 E80100                 193             CALL    UCCHEK          ;always return UC
F903 C3                     194     KQUIT:  RET
                            195     ;
F904 3C61                   196     UCCHEK: CMP     AL,'a'
F906 7206                   197             JC      UQUIT
F908 3C7B                   198             CMP     AL,'z'+1
F90A 7302                   199             JNC     UQUIT
F90C 245F                   200             AND     AL,5FH
F90E C3                     201     UQUIT:  RET
                            202     ;
F90F 52                     203     KEYIN:  PUSH    DX              ;keyboard device handler
F910 BAE2E2                 204             MOV     DX,KSTAT
F913 EC                     205     KEYLOOP:IN      AL,DX           ;check for keypress
F914 2480                   206             AND     AL,80H
F916 74FB                   207             JZ      KEYLOOP         ;no keypress..then wait for one
F918 5A                     208             POP     DX
F919 52                     209     KIN:    PUSH    DX
F91A BAE3E3                 210             MOV     DX,KDATA
F91D EC                     211             IN      AL,DX           ;else get the character
F91E 247F                   212             AND     AL,7FH          ;and strip parity
F920 5A                     213             POP     DX
F921 C3                     214             RET
                            215     ;
F922                        216     KEYSTAT:                        ;RETURN [AL]=0 IF NO KEYPRESS ELSE [AL]=0FFH
F922 B400                   217             MOV     AH,FALSE        ;prepare for false
F924 52                     218             PUSH    DX
F925 BAE2E2                 219             MOV     DX,KSTAT
F928 EC                     220             IN      AL,DX
F929 2480                   221             AND     AL,80H
F92B 7402                   222             JZ      KEXIT           ;return it if no keypress
F92D F6D4                   223             NOT     AH              ;otherwise make it TRUE
F92F 8AC4                   224     KEXIT:  MOV     AL,AH


MCS-86 MACRO ASSEMBLER     VID88

LOC  OBJ                   LINE     SOURCE

F931 5A                     225             POP     DX
F932 C3                     226             RET
                            227     ;
F933                        228     CTLCHEK:                        ;CHECK FOR CTL-S, CTL-Q AND CTL-C
F933 50                     229             PUSH    AX
F934 E8EBFF                 230             CALL    KEYSTAT         ;keypress?
F937 3C00                   231             CMP     AL,0
F939 7418                   232             JZ      CTLEXIT         ;no keypress so return
F93B E8DBFF                 233             CALL    KIN             ;if keypress then get the data
F93E 3C13                   234             CMP     AL,CTLS         ;check for ctl-s
F940 750D                   235             JNZ     CTLCCHEK        ;if not look for ctl-c
F942 E8CAFF                 236     KWAIT:  CALL    KEYIN           ;if ctl-s then wait for another keypress
F945 3C11                   237             CMP     AL,CTLQ         ;is it ctl-q
F947 740A                   238             JZ      CTLEXIT         ;YES..return
F949 3C03                   239             CMP     AL,CTLC         ;abort?
F94B 745A                   240             JE      ERR             ;YES
F94D EBF3                   241             JMP     KWAIT           ;otherwise wait some more
F94F                        242     CTLCCHEK:
F94F 3C03                   243             CMP     AL,CTLC         ;is it ctl-c
F951 7454                   244             JZ      ERR             ;YES..ABORT!
F953                        245     CTLEXIT:
```

```
F953 58                  246             POP     AX
F954 C3                  247             RET
                         248     ;
F955                     249     CIN:                    ;GET BYTE FROM CASSETTE
F955 52                  250             PUSH    DX
F956 BA6E6E              251             MOV     DX,CSTAT
F959                     252     CINLOOP:
F959 EC                  253             IN      AL,DX
F95A 2410                254             AND     AL,10H          ;cassette ready to read?
F95C 75FB                255             JNZ     CINLOOP         ;NO..wait
F95E BA6F6F              256             MOV     DX,CDATA        ;YES..
F961 EC                  257             IN      AL,DX           ;get the data
F962 5A                  258             POP     DX
F963 C3                  259             RET
                         260     ;
F964                     261     COUT:                   ;WRITE A BYTE TO CASSETTE
F964 52                  262             PUSH    DX
F965 50                  263             PUSH    AX
F966 BA6E6E              264             MOV     DX,CSTAT
F969                     265     COUTLOOP:
F969 EC                  266             IN      AL,DX
F96A 2420                267             AND     AL,20H          ;cassette ready for write?
F96C 75FB                268             JNZ     COUTLOOP        ;NO..wait
F96E 58                  269             POP     AX              ;get char back
F96F BA6F6F              270             MOV     DX,CDATA
F972 EE                  271             OUT     DX,AL           ;and send to tape
F973 5A                  272             POP     DX
F974 C3                  273             RET
                         274     ;
F975 50                  275     CRLF:   PUSH    AX
F976 E8BAFF              276             CALL    CTLCHEK         ;CHECK FOR ABORT
F979 B00D                277             MOV     AL,CR           ;SEND CR AND LF TO CONSOLE
F97B E85C03              278             CALL    VIDOUT

MCS-86 MACRO ASSEMBLER   VID88

LOC  OBJ                 LINE    SOURCE

F97E B00A                279             MOV     AL,LF
F980 E85703              280             CALL    VIDOUT
F983 58                  281             POP     AX
F984 C3                  282             RET
                         283     ;
F985                     284     BLANK:                  ;PRINT A BLANK, SAVE ALL REG
F985 51                  285             PUSH    CX
F986 B90100              286             MOV     CX,1            ;print 1 blank
F989 E80200              287             CALL    TABS
F98C 59                  288             POP     CX
F98D C3                  289             RET
                         290     ;
F98E                     291     TABS:                   ;PRINT # BLANKS IN CX..ON EXIT CX=0
F98E 50                  292             PUSH    AX
F98F B020                293             MOV     AL,' '
F991 E84603              294     TLOOP:  CALL    VIDOUT
F994 E2FB                295             LOOP    TLOOP
F996 58                  296             POP     AX
F997 C3                  297             RET
                         298     ;
F998                     299     PRINTMESS:              ;PRINT THE MESSAGE <-- [SI] ON CONSOLE
F998 50                  300             PUSH    AX      ;END OF MESSAGE IS A ZERO (0)
F999 AC                  301     PMESS:  LODS    DBYTE           ;get a byte
F99A 3C00                302             CMP     AL,0            ;check for end of message
F99C 7407                303             JE      PQUIT           ;quit if zero
F99E 56                  304             PUSH    SI              ;otherwise save message pointer
F99F E83803              305             CALL    VIDOUT          ;and display byte
F9A2 5E                  306             POP     SI
F9A3 EBF4                307             JMP     PMESS           ;print more message
F9A5 58                  308     PQUIT:  POP     AX
F9A6 C3                  309             RET
                         310     ;
F9A7 B02A                311     ERR:    MOV     AL,'*'          ;print error
F9A9 E82E03              312             CALL    VIDOUT          ;  message
F9AC BCFFF7              313             MOV     SP,STACKP       ;reinitialize stack
F9AF E983FE              314             JMP     START           ;and abort!
                         315     ;
F9B2                     316     BINOUT:                 ;OUTPUT [AL] AS EIGHT BINARY DIGITS (BITS)
F9B2 51                  317             PUSH    CX
F9B3 B90800              318             MOV     CX,8
F9B6                     319     BINOUT1:
F9B6 D0E0                320             SHL     AL,1            ;get the bit
F9B8 7209                321             JB      BOUT1           ;output a 1
F9BA 50                  322             PUSH    AX              ;otherwise..
F9BB B030                323             MOV     AL,'0'          ;output
F9BD E81A03              324             CALL    VIDOUT          ;a 0
F9C0 EB0790              325             JMP     BINEND          ;continue
F9C3 50                  326     BOUT1:  PUSH    AX
F9C4 B031                327             MOV     AL,'1'          ;output a 1
F9C6 E81103              328             CALL    VIDOUT
```

Listing 1 continued on page 352

Listing 1 continued:

```
F9C9 58                  329     BINEND: POP     AX
F9CA E2EA                330             LOOP    BINOUT1         ;do it eight times
F9CC 59                  331             POP     CX
F9CD C3                  332             RET


MCS-86 MACRO ASSEMBLER    VID88


LOC  OBJ                 LINE    SOURCE

                         333     ;
                         334     HEXOUT:                 ;OUTPUT [AL] AS 2 HEX DIGITS..ALL REG SAVED.
F9CE                     335             PUSH    AX
F9CE 50                  336             PUSH    CX
F9CF 51                  337             MOV     AH,AL           ;save AL
F9D0 8AE0                338             MOV     CL,4
F9D2 B104                339             SHR     AL,CL           ;shift AL right 4 places
F9D4 D2E8                340             POP     CX
F9D6 59                  341             CALL    HEXDIGOUT       ;output upper nibble
F9D7 E80700              342             MOV     AL,AH           ;restore AL (now we do lower nibble)
F9DA 8AC4                343             CALL    HEXDIGOUT
F9DC E80200              344             POP     AX
F9DF 58                  345             RET
F9E0 C3                  346     ;
                         347     HEXDIGOUT:              ;CONVERT NIBBLE TO ASCII HEX
F9E1                     348             AND     AL,0FH          ;mask upper 4 bits
F9E1 240F                349             ADD     AL,90H          ;tricky conversion...
F9E3 0490                350             DAA                     ;     but
F9E5 27                  351             ADC     AL,40H          ;     it
F9E6 1440                352             DAA                     ;     works!
F9E8 27                  353             CALL    VIDOUT          ;print the result
F9E9 E8EE02              354             RET
F9EC C3                  355     ;
                         356     HEXCHK:                 ;CHECK AL FOR VALID HEX DIGIT;CONVERT TO BIN
F9ED                     357             SUB     AL,'0'  ; IF INVALID RETURN WITH CARRY SET.
F9ED 2C30                358             JB      HRET            ;Error..not alphanumeric
F9EF 720E                359             CMP     AL,0AH          ;check for 0-9
F9F1 3C0A                360             CMC
F9F3 F5                  361             JNB     HRET            ;return o.k. if 0-9
F9F4 7309                362             SUB     AL,7            ;adjust for A-F
F9F6 2C07                363             CMP     AL,10
F9F8 3C0A                364             JB      HRET            ;return error if > F
F9FA 7203                365             CMP     AL,16
F9FC 3C10                366             CMC
F9FE F5                  367     HRET:   RET
F9FF C3                  368     ;
                         369     GETPARMB:               ;16 BIT HEX VALUE TO BX. BX IS SHIFT REGISTER, ACCEPTS LAST 4
FA00                     370                             ;ON ENTRY CX EQUALS NUMBER OF KEYPRESSES THAT CAN BE ACCEPTED
                         371                             ;ON EXIT AH CONTAINS TERMINATOR (I.E. CR, SPACE)
                         372                             ;UNLESS THE TERMINATOR IS INVALID (NOT EQUAL CR, SPACE OR ',')
                         373                             ;IN WHICH CASE AN ERROR IS GENERATED.
                         374     ;
FA00 BB0000              375             MOV     BX,0            ;clear BX
FA03 E8E6FE              376     LOOPB:  CALL    CONIN           ;get a character
FA06 3C30                377             CMP     AL,'0'          ;alphanumeric ?
FA08 7210                378             JB      BEXIT           ;NO...quit
FA0A 51                  379             PUSH    CX              ;YES...then
FA0B B104                380             MOV     CL,4            ;shift BX to
FA0D D3E3                381             SHL     BX,CL           ;make room for
FA0F 59                  382             POP     CX              ;latest addition
FA10 E8DAFF              383             CALL    HEXCHK          ;check for valid hex and convert to binary
FA13 7292                384             JB      ERR             ;if invalid then error!
FA15 02D8                385             ADD     BL,AL           ;otherwise add it in
FA17 E2EA                386             LOOP    LOOPB           ;keep looking


MCS-86 MACRO ASSEMBLER    VID88


LOC  OBJ                 LINE    SOURCE

FA19 C3                  387             RET
FA1A 3C20                388     BEXIT:  CMP     AL,' '          ;test for blank
FA1C 740B                389             JE      BGOOD
FA1E 3C2C                390             CMP     AL,','          ;..comma
FA20 7407                391             JE      BGOOD
FA22 3C0D                392             CMP     AL,CR           ;  or carriage return
FA24 7403                393             JE      BGOOD
FA26 E97EFF              394             JMP     ERR             ;if none of the above the ERROR
FA29 8AE0                395     BGOOD:  MOV     AH,AL           ;save terminator
FA2B C3                  396             RET
                         397     ;
FA2C                     398     GETPARMD:               ;16 BIT HEX VALUE TO DX. USE GETPARMB
FA2C 53                  399             PUSH    BX              ;save BX
FA2D E8D0FF              400             CALL    GETPARMB        ;get the parameter
FA30 8BD3                401             MOV     DX,BX           ;put it where it belongs
FA32 5B                  402             POP     BX              ;restore BX
FA33 C3                  403             RET
                         404     ;
```

```
FA34
FA34 51
FA35 B9FFFF
FA38 E8C5FF
FA3B 3C0D
FA3D 7406
FA3F E8EAFF
FA42 EB0390
FA45 8BD3
FA47 59
FA48 C3

FA49
FA49 52
FA4A 3BD3
FA4C 7303
FA4E E956FF
FA51 2BD3
FA53 8BCA
FA55 41
FA56 5A
FA57 C3

FA58

FA58 53
FA59 51
FA5A 52
FA5B 8AD0
FA5D B9FFFF
FA60 E89DFF
FA63 81F9FFFF
FA67 7502
FA69 8ADA
FA6B 8AC3
FA6D 5A
```

```
405     SETUP:                  ;GET PARMS IN BX AND DX.  ALL PURPOSE PARAMETER GETTER.
406             PUSH    CX              ;save CX
407             MOV     CX,0FFFFH       ;allow  64K keypresses
408             CALL    GETPARMB        ;get first parameter
409             CMP     AL,CR           ;check for carriage return
410             JE      SET1            ;if so [DX] defaults to [BX]
411             CALL    GETPARMD        ;otherwise get second parameter
412             JMP     SET2
413     SET1:   MOV     DX,BX
414     SET2:   POP     CX
415             RET
416     ;
417     CLENGTH:                        ;[CX]<--[DX]-[BX]+1, IF[BX]>[DX] THEN ERR
418             PUSH    DX
419             CMP     DX,BX           ;if [BX] > [DX]
420             JNB     CL1             ;then error!
421             JMP     ERR
422     CL1:    SUB     DX,BX           ;else determine difference
423             MOV     CX,DX           ;and put in CX
424             INC     CX              ;count = difference + 1
425             POP     DX
426             RET
427     ;
428     GETPARMAL:              ;[AL]<-- ASCII HEX FROM CONSOLE
429                             ;[AL] UNCHANGED IF NO PARAMETER ENTERED
430             PUSH    BX              ;save BX
431             PUSH    CX              ;save CX
432             PUSH    DX              ;save DX
433             MOV     DL,AL           ;save AL
434             MOV     CX,0FFFFH       ;64 keypresses allowed
435             CALL    GETPARMB        ;get the parameter
436             CMP     CX,0FFFFH       ;how many parameters entered?
437             JNE     GQUIT           ;if greater than zero then continue
438             MOV     BL,DL           ;if zero parms entered restore old value
439     GQUIT:  MOV     AL,BL           ;otherwise put it where it belongs
440             POP     DX              ;restore DX
```



```
LOC  OBJ                LINE    SOURCE

FA6E 59                  441            POP     CX              ;restore CX
FA6F 5B                  442            POP     BX              ;restore BX
FA70 C3                  443            RET
                         444    ;
FA71                     445    OUTBX:                  ;[BX] OUTPUT AS FOUR HEX DIGITS
FA71 50                  446            PUSH    AX
FA72 8AC7                447            MOV     AL,BH           ;output
FA74 E857FF              448            CALL    HEXOUT          ;  BH
FA77 8AC3                449            MOV     AL,BL           ;  and
FA79 E852FF              450            CALL    HEXOUT          ;  BL
FA7C 58                  451            POP     AX
FA7D C3                  452            RET
                         453    ;
                         454 +1 $EJECT
```



```
LOC  OBJ                LINE    SOURCE

                         455    ;
                         456    ;
                         457    ;
                         458    ;
                         459    ;   ************************************************
                         460    ;   *                                              *
                         461    ;   *       C O M M A N D   R O U T I N E S        *
                         462    ;   *                                              *
                         463    ;   ************************************************
                         464    ;
                         465    ;
FA7E                     466    FILL:                   ;FILL A BLOCK OF MEMORY WITH A CONSTANT
FA7E E8B3FF              467            CALL    SETUP           ;get start and end
FA81 E8C5FF              468            CALL    CLENGTH         ;compute the count
FA84 E8D1FF              469            CALL    GETPARMAL       ;and get the constant
FA87 8807                470    FLOOP:  MOV     M[BX],AL        ;...fill it...
FA89 43                  471            INC     BX
FA8A E2FB                472            LOOP    FLOOP
FA8C C3                  473            RET
                         474    ;
FA8D                     475    VERIFY:                 ;VERIFY EQUALITY OF TWO BLOCKS OF MEMORY
FA8D E8A4FF              476            CALL    SETUP           ;GET SOURCE START AND END
FA90 E8B6FF              477            CALL    CLENGTH         ;and compute the length
FA93 41                  478            INC     CX
```

Listing 1 continued on page 354

Listing 1 continued:

```
FA94 8BF3                479             MOV     SI,BX           ;save source in SI
FA96 51                  480             PUSH    CX
FA97 B9FFFF              481             MOV     CX,OFFFFH       ;64K keypresses allowed
FA9A E863FF              482             CALL    GETPARMB        ;get the destination
FA9D 59                  483             POP     CX
FA9E 8BFB                484             MOV     DI,BX           ;into DX
FAA0 E8D2FE              485             CALL    CRLF
FAA3 56                  486             PUSH    SI              ;save source
FAA4 BE90F8              487             MOV     SI,OFFSET VHEAD
FAA7 E8EEFE              488             CALL    PRINTMESS       ;print header
FAAA 5E                  489             POP     SI              ;restore source
FAAB                     490     VLOOP:
FAAB F3                  491       REPE  CMPS    DBYTE,DBYTE     ;do it!
FAAC A6
FAAD 83F900              492             CMP     CX,0            ;all done?
FAB0 7501                493             JNE     VERR            ;NO...error
FAB2 C3                  494             RET                     ; if done then return
                         495     ;
FAB3 8BDE                496     VERR:   MOV     BX,SI           ;get the source addr
FAB5 4B                  497             DEC     BX              ;adjust it
FAB6 E8BCFE              498             CALL    CRLF
FAB9 E8B5FF              499             CALL    OUTBX           ;output the addr
FABC E8C6FE              500             CALL    BLANK
FABF 8A07                501             MOV     AL,M[BX]        ;get what's there
FAC1 8AE0                502             MOV     AH,AL           ;save it in AH
FAC3 E808FF              503             CALL    HEXOUT          ;output the data
FAC6 E8BCFE              504             CALL    BLANK
FAC9 E8B9FE              505             CALL    BLANK
FACC 8BDF                506             MOV     BX,DI           ;get the destination addr
FACE 4B                  507             DEC     BX              ;adjust it
```



```
LOC  OBJ                 LINE    SOURCE

FACF E89FFF              508             CALL    OUTBX           ;display it
FAD2 8A07                509             MOV     AL,M[BX]        ;get the data
FAD4 E8AEFE              510             CALL    BLANK
FAD7 E8F4FE              511             CALL    HEXOUT          ;output the data
FADA E8A8FE              512             CALL    BLANK
FADD 32C4                513             XOR     AL,AH           ;determine bad bits
FADF E8D0FE              514             CALL    BINOUT          ;display in binary
FAE2 E84EFE              515             CALL    CTLCHEK         ;check for abort
FAE5 EBC4                516             JMP     VLOOP           ;continue
                         517     ;
FAE7                     518     MOVE:                   ;MOVE A BLOCK OF MEMORY
FAE7 E84AFF              519             CALL    SETUP           ;get start and end
FAEA 3C0D                520             CMP     AL,0DH          ;if not enough data
FAEC 7503                521             JNZ     M1
FAEE E9B6FE              522             JMP     ERR             ;then error!
FAF1 E855FF              523     M1:     CALL    CLENGTH         ;otherwise compute length
FAF4 53                  524             PUSH    BX              ;save start address
FAF5 E83CFF              525             CALL    SETUP           ;and get destination
FAF8 8BFB                526             MOV     DI,BX           ;[DI] <-- destination
FAFA 5B                  527             POP     BX
FAFB 8BF3                528             MOV     SI,BX           ;[SI] <-- source
FAFD F3                  529       REP   MOVS    DBYTE,DBYTE     ;...move it...
FAFE A4
FAFF C3                  530             RET
                         531     ;
FB00                     532     DUMP:                   ;DISPLAY MEMORY
FB00 E831FF              533             CALL    SETUP           ;get start and end
FB03 E843FF              534             CALL    CLENGTH         ;and compute length
FB06 E81900              535             CALL    NULINE2         ;set up console
FB09 8A07                536     DLOOP1: MOV     AL,M[BX]        ;get what's there
FB0B E8C0FE              537             CALL    HEXOUT          ;print it
FB0E E874FE              538             CALL    BLANK           ;and a blank
FB11 43                  539             INC     BX
FB12 F6C30F              540             TEST    BL,0FH          ;test for 16 byte boundary
FB15 7503                541             JNZ     DNEXT           ;if not then continue
FB17 E80300              542             CALL    NULINE          ;otherwise set up console for new line
FB1A E2ED                543     DNEXT:  LOOP    DLOOP1          ;continue
FB1C C3                  544             RET
                         545     ;
FB1D 83F901              546     NULINE: CMP     CX,1
FB20 7409                547             JE      NUQUIT
FB22                     548     NULINE2:
FB22 E850FE              549             CALL    CRLF            ;go to new line
FB25 E849FF              550             CALL    OUTBX           ;print address
FB28 E85AFE              551             CALL    BLANK           ;and a blank
FB2B C3                  552     NUQUIT: RET
                         553     ;
FB2C                     554     INPUT:                  ;INPUT FROM A PORT
FB2C E805FF              555             CALL    SETUP           ;get port address
FB2F E843FE              556             CALL    CRLF
FB32 8BD3                557             MOV     DX,BX
FB34 EC                  558             IN      AL,DX           ;read the port
FB35 E896FE              559             CALL    HEXOUT          ;print data in hex
FB38 E84AFE              560             CALL    BLANK           ;       and
```

```
MCS-86 MACRO ASSEMBLER    VID88

LOC  OBJ                  LINE     SOURCE

FB3B E874FE                561             CALL    BINOUT          ;       binary
FB3E C3                    562             RET
                           563     ;
FB3F                       564     OUTPUT:                         ;OUTPUT TO A PORT
FB3F E8F2FE                565             CALL    SETUP           ;get address
FB42 8AC2                  566             MOV     AL,DL           ;and data
FB44 FEC8                  567             DEC     AL              ;adjust data
FB46 8BD3                  568             MOV     DX,BX
FB48 EE                    569             OUT     DX,AL           ;output data
FB49 C3                    570             RET
                           571     ;
FB4A                       572     GOTO:                           ;EXECUTE A PROGRAM
FB4A E8E7FE                573             CALL    SETUP           ;get the address
FB4D FFE3                  574             JMP     BX              ;GO!!
                           575     ;
FB4F                       576     CWRITE:                         ;WRITE TO CASSETTE
FB4F E8E2FE                577             CALL    SETUP           ;get the range
FB52 E8F4FE                578             CALL    CLENGTH         ;compute the length
FB55 E81DFE                579             CALL    CRLF
FB58 E85500                580             CALL    CPROMPT
FB5B E80F00                581             CALL    CSTART
FB5E E85600                582             CALL    LENGTHOUT       ;tell length
FB61 8A07                  583     CLOOP:  MOV     AL,[BX]         ;get a byte
FB63 E8FEFD                584             CALL    COUT            ;output
FB66 43                    585             INC     BX              ;next byte
FB67 E8C9FD                586             CALL    CTLCHEK         ;check for abort
FB6A E2F5                  587             LOOP    CLOOP           ;continue
FB6C C3                    588             RET
                           589     ;
FB6D B03C                  590     CSTART: MOV     AL,3CH          ;start byte
FB6F E8F2FD                591             CALL    COUT
FB72 B0E6                  592             MOV     AL,0E6H         ;sync byte
FB74 E8EDFD                593             CALL    COUT
FB77 8AC5                  594             MOV     AL,CH           ;high length
FB79 E8E8FD                595             CALL    COUT
FB7C 8AC1                  596             MOV     AL,CL           ;low length
FB7E E8E3FD                597             CALL    COUT
FB81 C3                    598             RET
                           599     ;
FB82                       600     READ:                           ;READ FROM CASSETTE
FB82 E8AFFE                601             CALL    SETUP           ;get address
FB85 E8EDFD                602             CALL    CRLF
FB88 E82500                603             CALL    CPROMPT
FB8B E80F00                604             CALL    READINIT
FB8E E82600                605             CALL    LENGTHOUT       ;prompt when reading
FB91 E8C1FD                606     RLOOP:  CALL    CIN             ;get a byte
FB94 8807                  607             MOV     [BX],AL
FB96 43                    608             INC     BX              ;next byte
FB97 E899FD                609             CALL    CTLCHEK         ;check for abort
FB9A E2F5                  610             LOOP    RLOOP           ;continue
FB9C C3                    611             RET
                           612     ;
FB9D                       613     READINIT:
FB9D B010                  614             MOV     AL,10H          ;reset interface

MCS-86 MACRO ASSEMBLER    VID88

LOC  OBJ                  LINE     SOURCE

FB9F 52                    615             PUSH    DX
FBA0 BA6E6E                616             MOV     DX,CSTAT
FBA3 EE                    617             OUT     DX,AL
FBA4 5A                    618             POP     DX
FBA5 E8ADFD                619             CALL    CIN
FBA8 8AE8                  620             MOV     CH,AL           ;get high length
FBAA E8A8FD                621             CALL    CIN
FBAD 8AC8                  622             MOV     CL,AL           ;and low length
FBAF C3                    623             RET
                           624     ;
FBB0                       625     CPROMPT:                        ;CASSETTE PROMPT
FBB0 BEA8F8                626             MOV     SI, OFFSET CHEAD
FBB3 E8E2FD                627             CALL    PRINTMESS
FBB6 C3                    628             RET
                           629     ;
FBB7                       630     LENGTHOUT:                      ;OUTPUT RECORD LENGTH
FBB7 53                    631             PUSH    BX
FBB8 8BD9                  632             MOV     BX,CX           ;get the count
FBBA E8B4FE                633             CALL    OUTBX           ;output it
FBBD 5B                    634             POP     BX
FBBE C3                    635             RET
```

Listing 1 continued on page 356

Listing 1 continued:

```
FBBF
FBBF E8B3FD
FBC2
FBC2 B0E6
FBC4 E89DFD
FBC7 E858FD
FBCA 3C00
FBCC 74F4
FBCE E848FD
FBD1 C3

FBD2
FBD2 E85FFE
FBD5 E89DFD
FBD8 BE71F8
FBDB E8BAFD
FBDE E8A4FD
FBE1 E8CCFF
FBE4 E8B6FF
FBE7 E8CDFF
FBEA
FBEA E868FD
FBED 3A07
FBEF 7507
FBF1 43
FBF2 E83EFD
FBF5 E2F3
FBF7 C3
FBF8 50
FBF9 E879FD
FBFC E872FE
FBFF E883FD
```

```
636     ;
637     PUTSYNC:                        ;SEND SYNC STREAM TO CASSETTE
638             CALL    CRLF
639     SYNCLOOP:
640             MOV     AL,0E6H         ;sync character
641             CALL    COUT            ;send it
642             CALL    KEYSTAT         ;check for keypress
643             CMP     AL,0            ;zero = no keypress
644             JE      SYNCLOOP        ;so continue
645             CALL    KIN             ;ignore the keypress
646             RET                     ;and quit
647     ;
648     COMPARE:                        ;COMPARE INPUT FROM CASSETTE WITH MEMORY
649             CALL    SETUP
650             CALL    CRLF
651             MOV     SI, OFFSET COMHEAD      ;print header
652             CALL    PRINTMESS
653             CALL    BLANK
654             CALL    CPROMPT
655             CALL    READINIT
656             CALL    LENGTHOUT
657     COMLOOP:
658             CALL    CIN             ;get char from cassette
659             CMP     AL,[BX]         ;compare with memory
660             JNE     COMERR          ;not equal!! error
661     COM1:   INC     BX              ;if equal
662             CALL    CTLCHEK         ;check for abort
663             LOOP    COMLOOP         ;then continue checking
664             RET
665     COMERR: PUSH    AX
666             CALL    CRLF
667             CALL    OUTBX           ;if error..output memory address
668             CALL    BLANK
```



```
LOC  OBJ

FC02 8A07
FC04 8AF0
FC06 E8C5FD
FC09 E879FD
FC0C 58
FC0D E8BEFD
FC10 E872FD
FC13 32C6
FC15 E89AFD
FC18 EBD7

FC1A
FC1A E817FE
FC1D 53
FC1E 52
FC1F E853FD
FC22 BE86F8
FC25 E870FD
FC28 E84AFD
FC2B 03DA
FC2D E841FE
FC30 E852FD
FC33 5A
FC34 5B
FC35 2BDA
FC37 E837FE
FC3A C3

FC3B
FC3B E8F6FD
FC3E E808FE
FC41 E831FD
FC44 53
FC45 51
FC46 8A07
FC48 8AE0
FC4A F6D0
FC4C 8807
FC4E 8A07
FC50 F6D0
FC52 3AC4
FC54 750C
FC56 8827
FC58 43
FC59 E8D7FC
FC5C E2E8
FC5E 59
FC5F 5B
FC60 EBE2
```

```
LINE    SOURCE

669             MOV     AL,[BX]         ;get memory data
670             MOV     DH,AL           ;save it too
671             CALL    HEXOUT          ;output what's in memory
672             CALL    BLANK
673             POP     AX              ;restore cassette data
674             CALL    HEXOUT          ;output it
675             CALL    BLANK
676             XOR     AL,DH           ;determine bad bits
677             CALL    BINOUT          ;and print in binary
678             JMP     COM1            ;continue
679     ;
680     HEXMATH:                        ;COMPUTE SUM AND DIFFERENCE OF TWO HEX #'S
681             CALL    SETUP           ;get the numbers
682             PUSH    BX              ;save
683             PUSH    DX              ; them
684             CALL    CRLF
685             MOV     SI, OFFSET MHEAD
686             CALL    PRINTMESS       ;print the header
687             CALL    CRLF
688             ADD     BX,DX           ;sum
689             CALL    OUTBX
690             CALL    BLANK
691             POP     DX              ;restore
692             POP     BX              ; numbers
693             SUB     BX,DX           ;difference
694             CALL    OUTBX
695             RET
696     ;
697     NTEST:                          ;MEMORY TEST
698             CALL    SETUP           ;get start and end
699             CALL    CLENGTH         ;compute length
700             CALL    CRLF
701     MTEST1: PUSH    BX
702             PUSH    CX
703     MTLOOP: MOV     AL,[BX]         ;get what's there
704             MOV     AH,AL           ;save it
705             NOT     AL              ;complement
706             MOV     [BX],AL         ;and store it back
707             MOV     AL,[BX]         ;read it again
708             NOT     AL              ;re-complement
709             CMP     AL,AH           ;is it o.k.?
710             JNE     SHORT   TERR    ;if not then error!
711             MOV     [BX],AH         ;restore previous value
712     TNEXT:  INC     BX              ;next location
713             CALL    CTLCHEK         ;check for abort
714             LOOP    MTLOOP          ;continue
715             POP     CX
716             POP     BX
717             JMP     MTEST1          ;test forever
718     ;
```

```
FC62 E810FD              719    TERR:   CALL    CRLF            ;TELL USER ABOUT BAD MEMORY
FC65 E809FE              720            CALL    OUTBX           ;output bad address
FC68 E81AFD              721            CALL    BLANK           ;and a blank
FC6B 32C4                722            XOR     AL,AH           ;tell user which


MCS-86 MACRO ASSEMBLER   VID88


LOC  OBJ                 LINE   SOURCE

FC6D E85EFD              723            CALL    HEXOUT          ;bits are bad in hex...
FC70 E812FD              724            CALL    BLANK
FC73 E83CFD              725            CALL    BINOUT          ;and binary
FC76 EBE0                726            JMP     TNEXT           ; continue
                         727    ;
FC78                     728    ESUBST:                         ;SUBSTITUTE MEMORY WITH HEX DATA
FC78 E8B9FD              729            CALL    SETUP           ;get address
FC7B                     730    NUSLOOP:
FC7B E8F7FC              731            CALL    CRLF            ;and
FC7E E8F0FD              732            CALL    OUTBX           ;print it
FC81 B90800              733            MOV     CX,8            ;8 entries per line
FC84 E8FEFC              734    SLOOP:  CALL    BLANK
FC87 8A07                735            MOV     AL,M[BX]        ;get what's there
FC89 E842FD              736            CALL    HEXOUT          ;and print it
FC8C 50                  737            PUSH    AX              ;save it
FC8D B02D                738            MOV     AL,'-'          ;with a prompt
FC8F E84800              739            CALL    VIDOUT
FC92 58                  740            POP     AX              ;restore it
FC93 E8C2FD              741            CALL    GETPARMAL       ;get new data
FC96 EB0890              742            JMP     QTEST           ;check for quit
FC99 8807                743    SNEXT:  MOV     M[BX],AL        ;otherwise, put new data in memory
FC9B 43                  744            INC     BX              ;and continue
FC9C E2E6                745            LOOP    SLOOP
FC9E EBDB                746            JMP     NUSLOOP
FCA0 80FC20              747    QTEST:  CMP     AH,' '          ;if blank then
FCA3 74F4                748            JE      SNEXT           ;continue
FCA5 80FC0D              749            CMP     AH,0DH          ;if carraige return
FCA8 7403                750            JE      Q1              ;then we are done
FCAA E9FAFC              751            JMP     ERR             ;otherwise..error!
FCAD 8807                752    Q1:     MOV     M[BX],AL        ;save that last one!
FCAF C3                  753            RET
                         754    ;
FCB0                     755    AENTER:                         ;ENTER ASCII TEXT IN MEMORY
FCB0 B9FFFF              756            MOV     CX,0FFFFH       ;64K keypresses
FCB3 E84AFD              757            CALL    GETPARMB        ;get the entry address
FCB6 E8BCFC              758            CALL    CRLF
FCB9 E830FC              759    ELOOP:  CALL    CONIN
FCBC 3C04                760            CMP     AL,CTLD         ;done?
FCBE 7405                761            JE      EEXIT           ;YES
FCC0 8807                762            MOV     M[BX],AL        ;NO..put data in memory
FCC2 43                  763            INC     BX
FCC3 EBF4                764            JMP     ELOOP
FCC5 E8ADFC              765    EEXIT:  CALL    CRLF
FCC8 B040                766            MOV     AL,'@'
FCCA E80D00              767            CALL    VIDOUT
FCCD E8A1FD              768            CALL    OUTBX           ;output the ending address
FCD0 C3                  769            RET
                         770    ;
FCD1                     771    KTOGGLE:                        ;TOGGLE THE UPPER/LOWER CASE FLAG
FCD1 A052F4              772            MOV     AL,BYTE PTR M[UCFLAG]   ;get the flag
FCD4 F6D0                773            NOT     AL              ;toggle
FCD6 A252F4              774            MOV     BYTE PTR M[UCFLAG],AL   ;put flag back
FCD9 C3                  775            RET
                         776    ;


MCS-86 MACRO ASSEMBLER   VID88


LOC  OBJ                 LINE   SOURCE

                         777 +1 $EJECT


MCS-86 MACRO ASSEMBLER   VID88


LOC  OBJ                 LINE   SOURCE

                         778    ;
                         779    ;
                         780    ;
                         781    ;       ************************************************
                         782    ;       *                                              *
                         783    ;       *              V I D E O   D R I V E R         *
                         784    ;       *                                              *
```

Listing 1 continued on page 358

Listing 1 continued:

```
                        ;       *                                               *
                        ;       *          DRIVES  TDL  VDB  VIDEO  INTERFACE     *
                        ;       *                                               *
                        ;       *      converted from 8080 Assembler with CONV-86   *
                        ;       *                                               *
                        ;       ************************************************
                        ;
                        ;
                        ;       VIDEO DRIVER
                        ;
FCDA 50                 VIDOUT: PUSH    AX
FCDB 56                         PUSH    SI
FCDC 57                         PUSH    DI
FCDD E80400                     CALL    VIDEO
FCE0 5F                         POP     DI
FCE1 5E                         POP     SI
FCE2 58                         POP     AX
FCE3 C3                         RET
                        ;
                        ;******** CONVERTED CODE BEGINS HERE ********
                        ;
                        ;VDB DRIVER
00E1                    VD      EQU     0E1H
00E0                    VC      EQU     0E0H
00E0                    XRD     EQU     0E0H
00E1                    YRD     EQU     0E1H
00C0                    YWR     EQU     0C0H
00E2                    MRD     EQU     0E2H
0080                    MWR     EQU     80H
0088                    VMODE   EQU     88H
0098                    BMODE   EQU     98H
                        ;
                        ;
FCE4 53                 VIDEO:  PUSH    BX
FCE5 8B1E50F4                   MOV     BX,WORD PTR M[XY]
FCE9 247F                       AND     AL,7FH
FCEB 7403                       JZ      SHORT L_2
FCED E80600                     CALL    VOUT
FCF0                    L_2:
FCF0 891E50F4                   MOV     WORD PTR M[XY],BX
FCF4 5B                         POP     BX
FCF5 C3                         RET
                        ;
FCF6 3C20               VOUT:   CMP     AL,20H
FCF8 7303                       JAE     SHORT L_3
FCFA EB7490                     JMP     CNTL
FCFD                    L_3:


MCS-86 MACRO ASSEMBLER  VID88


LOC  OBJ                LINE    SOURCE

                          832           CMP     AL,7FH
FCFD 3C7F                 833           JNZ     SHORT L_4
FCFF 7501                 834           RET
FD01 C3                   835   L_4:
FD02                      836           OUT     VD,AL
FD02 E6E1                 837           DEC     BH
FD04 FECF                 838           JZ      SHORT L_5
FD06 7401                 839           RET
FD08 C3                   840   L_5:
FD09                      841           MOV     BH,80
FD09 B750                 842           DEC     BL
FDOB FECB                 843           JZ      SHORT L_6
FDOD 7401                 844           RET
FDOF C3                   845   L_6:
FD10                      846   V02:    INC     BL
FD10 FEC3                 847   ;
                          848   SCROLL: PUSH    BX
FD12 53                   849           PUSH    DX
FD13 52                   850           PUSH    CX
FD14 51                   851           MOV     AL,BMODE
FD15 B098                 852           OUT     VC,AL
FD17 E6E0                 853           XOR     AL,AL
FD19 32C0                 854           OUT     VC,AL
FD1B E6E0                 855           MOV     DX,0C150H
FD1D BA50C1               856           MOV     AL,DH
FD20 8AC6                 857   S1:     OUT     VC,AL
FD22 E6E0                 858           MOV     CH,DL
FD24 8AEA                 859           MOV     BX,VIDBUF
FD26 BB00F4               860   L1:     IN      AL,VD
FD29 E4E1                 861           MOV     M[BX],AL
FD2B 8807                 862           LAHF
FD2D 9F                   863           INC     BX
FD2E 43                   864           SAHF
FD2F 9E                   865           DEC     CH
FD30 FECD                 866           JNZ     L1
FD32 75F5                 867           MOV     AL,DH
FD34 8AC6
```

```
FD36 FEC8          868                  DEC     AL
FD38 E6E0          869                  OUT     VC, AL
FD3A BB00F4        870                  MOV     BX, VIDBUF
FD3D 8AEA          871                  MOV     CH, DL
FD3F 8A07          872          L2:     MOV     AL, M[BX]
FD41 E6E1          873                  OUT     VD, AL
FD43 9F            874                  LAHF
FD44 43            875                  INC     BX
FD45 9E            876                  SAHF
FD46 FECD          877                  DEC     CH
FD48 75F5          878                  JNZ     L2
FD4A FEC6          879                  INC     DH
FD4C 8AC6          880                  MOV     AL, DH
FD4E 3CD9          881                  CMP     AL, 0D9H
FD50 72D0          882                  JB      S1
FD52 8AEA          883                  MOV     CH, DL
FD54 B020          884                  MOV     AL, 20H
FD56 E6E1          885          S2:     OUT     VD, AL

MCS-86 MACRO ASSEMBLER    VID88

LOC  OBJ           LINE      SOURCE

FD58 FECD          886                  DEC     CH
FD5A 75FA          887                  JNZ     S2
FD5C 59            888                  POP     CX
FD5D 5A            889                  POP     DX
FD5E 5B            890                  POP     BX
FD5F B088          891          SETCAV: MOV     AL, VMODE
FD61 E6E0          892                  OUT     VC, AL
FD63 B050          893          SETCUR: MOV     AL, 80
FD65 2AC7          894                  SUB     AL, BH
FD67 E6E0          895                  OUT     VC, AL
FD69 B0D9          896                  MOV     AL, 25+0C0H
FD6B 2AC3          897                  SUB     AL, BL
FD6D E6E0          898                  OUT     VC, AL
FD6F C3            899                  RET
                   900          ;
FD70 3C0D          901          CNTL:   CMP     AL, CR
FD72 7422          902                  JZ      SHORT CCR
FD74 3C0A          903                  CMP     AL, LF
FD76 7415          904                  JZ      SHORT CLF
FD78 3C0C          905                  CMP     AL, FF
FD7A 741E          906                  JZ      SHORT CFF
FD7C 3C08          907                  CMP     AL, BS
FD7E 7401          908                  JZ      SHORT CBS
FD80 C3            909                  RET
FD81 B04F          910          CBS:    MOV     AL, 79
FD83 2AC7          911                  SUB     AL, BH
FD85 7901          912                  JNS     SHORT L_8
FD87 C3            913                  RET
FD88               914          L_8:
FD88 E6E0          915                  OUT     VC, AL
FD8A FEC7          916                  INC     BH
FD8C C3            917                  RET
FD8D FECB          918          CLF:    DEC     BL
FD8F 7503          919                  JNZ     SHORT L_9
FD91 E97CFF        920                  JMP     V02
FD94               921          L_9:
FD94 EBCD          922                  JMP     SETCUR
FD96 B750          923          CCR:    MOV     BH, 80
FD98 EBC9          924                  JMP     SETCUR
FD9A B098          925          CFF:    MOV     AL, BMODE
FD9C E6E0          926                  OUT     VC, AL
FD9E BBD007        927                  MOV     BX, 25*80
FDA1 32C0          928          CFF1:   XOR     AL, AL
FDA3 E6E1          929                  OUT     VD, AL
FDA5 9F            930                  LAHF
FDA6 4B            931                  DEC     BX
FDA7 9E            932                  SAHF
FDA8 8AC7          933                  MOV     AL, BH
FDAA 0AC3          934                  OR      AL, BL
FDAC 75F3          935                  JNZ     CFF1
FDAE BB1950        936                  MOV     BX, 256*80+25
FDB1 EBAC          937                  JMP     SETCAV
                   938          ;
----               939          ABS_0   ENDS


MCS-86 MACRO ASSEMBLER    VID88


LOC  OBJ                  LINE      SOURCE

  F80F                    940               END     INIT

ASSEMBLY COMPLETE, NO ERRORS FOUND
```