```
; This is a CPM program to test some of the hardware on the Z80
; IBM-PC S-100 keyboard converter board and modify some of the software.
; This code will reside at 100H in your S-100 system RAM. The actual
; 2716 EPROM code will reside at near the beginning of the code with
; most of the diagnostic code above the EPROM code.
;
; The EPROM code will reside at 0H in RAM on the board without the diagnostic
; code. To remove the diagnostic code set DEBUG FALSE.
;
; For diagnostics you will need two input ports and one outport to control the board.
; The port connections are shown on the S100Computers.com web site.
; They are called KEYCLEAR,KEYSTAT and KEYIN (see below).
; You will also need to use your "normal" CPM keyboard deviceto test this board.
; You cannot test this board and at the same time use it as the CPM keyboard
; input device.
;
;         V2.1    Everything except caps lock does not work
;         V2.2    Caps lock OK CTRL needs to be done
;         V2.3    CAPS,CTRL done
;         V2.4    F1,F2 function key strings added
;
FALSE     EQU     0
TRUE      EQU     NOT FALSE
;
LF        EQU     0AH
CR        EQU     0DH
BELL      EQU     07H
SPACE     EQU     20H
TAB       EQU     09H   ;TAB ACROSS (8 SPACES FOR SD-BOARD)
;
DEBUG     EQU     TRUE  ;Mods to the the EPROM code if we are debugging here
;
IF  NOT DEBUG
STACK       EQU   0FFFH ;Stack at top or RAM for real ROM code
ENDIF
;
CONT$A      EQU   0F5H  ;CTRL port for output to Video board keyboard input port
CONT$B      EQU   0F7H  ;CTRL port for IBM Keyboard input
CONT$C      EQU   0F9H  ;CTRL port for data port KEYCLEAR
CONT$D      EQU   0FBH  ;CTRL port for port KEYSTAT
;
IF        DEBUG
KEYOUT      EQU   01H   ;Data port for direct output to SD Systems video board
KEYIN       EQU   04H   ;Raw data port for IMB keyboard (Diagnostic port)
KEYCLEAR    EQU   07H   ;Pulsing bit 0 high clears 74LS161 Shift Regs
KEYSTAT     EQU   07H   ;Input bit 0 is zero if IBMPC Key char ready
                        ;to be read and translated
ELSE

KEYOUT      EQU   0F4H  ;Data port for output to Video board keyboard input port
KEYIN       EQU   0F6H  ;Data port for IMB keyboard input
KEYCLEAR    EQU   0F8H  ;Pulsing bit 0 high clears 74LS161 Shift Regs
KEYSTAT     EQU   0FAH  ;Input bit 0 is zero if IBMPC Key char ready
ENDIF
;
;------------------   ACTUAL EPROM CODE FOR THE BOARD -------------------------------------
;
; The memory location "CASEFLAG" will contain various bit flages to indicate
; the current ASCII translation table being used(upper/lower case etc). They
; are also used to turn on/off the board indicator LED's.
; (Note a register could be set aside instead of a RAM location, but I wanted
; to have expandabity for more complex translations later).
;
; Bit flags for CASEFLAG:-
```

```
;                 Bit 0  not used (used to clear 74LS161 shift regs)
;                 Bit 1  Strobe for data into Video board keyboard port
;                 Bit 2 = 1 if Upper Case Lock is ON
;                 Bit 3 = 1 if Shift key is currently HELD down
;                 Bit 4 - 1 if Ctrl key is currently HELD down
;                 Bit 5 = 1 if NUM LOCK is ON
;
; REMEMBER IX is not used for ANYTHING other than to point to CASEFLAG
;
;
IF        DEBUG             ;If debug/test version start at 100H so it can be loaded with CPM
          ORG     100H
;
Start:    LD      SP,STACK    ;need a CPM stack
          LD      HL,SIGNON
          CALL    PMSG
          CALL    SETUP_BUFFERS     ;Setup IX to point to character output buffer
          CALL    CLEAR       ;Clear the shift registers

          JP      EPROM       ;<--- JUMP TO EPROM CODE
;
ELSE
          ORG     0H          ;The EPROM code will start here
;
          DI                  ;Just in case
          LD      A,0FH       ;First setup the two Zilog PIO's
          OUT     (CONT$A),A  ;Mode 0 Output
          LD      A,03H
          OUT     (CONT$A),A

          LD      A,4FH       ;Mode 1 Input
          OUT     (CONT$B),A
          LD      A,03H
          OUT     (CONT$B),A

          LD      A,0FH       ;Mode 0 Output
          OUT     (CONT$C),A
          LD      A,03H
          OUT     (CONT$C),A

          LD      A,4FH       ;Mode 1 Input
          OUT     (CONT$D),A
          LD      A,03H
          OUT     (CONT$D),A

          IN      A,(KEYIN)   ;Clear out any junk
          IN      A,(KEYSTAT)

          LD      HL,800H           ;Next clear RAM. This will allow DEBUG monitor (see below)
RAM0:     XOR     A           ;show up any problem.
          LD      (HL),A            ;Zero RAM
          INC     HL
          LD      A,H
          CP      10H         ;Clear from 0800H to 0FFFH
          JR      NZ,RAM0
          JR      OVER
;
          ORG     38H         ;Reset location 0FFH (CPU jumps here if no RAM)
          LD      A,38H       ;Flag for no RAM (38 appears on HEX display, HALT LED comes on)
          OUT     (KEYOUT),A  ;Send 38H to HEX display to indicate a RAM problem
          HALT                ;No Strobe, so Keyboard does not see it. Stop Everything
;
OVER:     LD      HL,800H           ;Check RAM.
RAM1:     LD      A,(HL)            ;Is it 0.
          OR      A           ;Zero RAM
```

```
          JR      NZ,RAM_PROBLEM
          INC     HL
          LD      A,H
          CP      10H         ;Clear from 0800H to 0FFFH
          JR      NZ,RAM1
          LD      SP,STACK    ;Now setup a valid stack on the boards RAM (@ 0FFFH)
          JR      RAM_OK
;
RAM_PROBLEM:
          LD      A,10H
          OUT     (KEYOUT),A  ;Send 10H to HEX display to indicate a RAM problem
          HALT                ;No Strobe, so Keyboard does not see it. Stop Everything

RAM_OK:   LD      A,11H
          OUT     (KEYOUT),A  ;Send 11H to HEX display to indicate a reset went OK.
                              ;No Strobe, so Keyboard does not see it
          LD      A,80H
          CALL    DELAY_LONG  ;Check stack

          LD      A,12H
          OUT     (KEYOUT),A  ;Then send 12H to HEX display to indicate a reset went OK.
                              ;No Strobe, so Keyboard does not see it

          CALL    SETUP_BUFFERS     ;Setup IX to point to character output buffer
          CALL    CLEAR       ;Clear all shift registers

          LD      A,13H
          OUT     (KEYOUT),A  ;Then send 13H to HEX display to indicate we are ready to go.
                              ;No Strobe, so Keyboard does not see it
ENDIF

EPROM:
LOOP:     CALL    GETSTAT           ;The MAIN LOOP. Anything at IBM Keyboard
          CALL    Z,LOOP1
          CALL    CRTOUT            ;>>> IF <<<, there is something in the que, send it
          JR      LOOP        ;to the keyboard output port

LOOP1:    IN      A,(KEYIN)   ;Something at IBM keyboard port
IF        DEBUG
          CALL    DIAGNOSTIC  ;Display raw data on consol
ENDIF
          CP      0F0H        ;Is it an UP key
          JR      NZ,DOWNKY   ;Must be a down key stroke
          CALL    UPKEY       ;Special tratment for UP key scan codes
          JR      SKIP
DOWNKY:   LD      (DOWN_KEY),A      ;Store it
          CP      58H         ;Is it CAPS Lock key
          JP      Z,CAPSKEY
          CP      12H         ;Is it a SHIFT key
          JP      Z,SHIFTKEY
          CP      59H         ;Is it the other SHIFT key
          JP      Z,SHIFTKEY
          CP      14H         ;Is it the CTRL key
          JP      Z,CTRLKEY
          CP      77H         ;Is it the NUM LOCK key
          JP      Z,NUMKEY

          CALL    TRANSLATE   ;IBM char in A, return with ASCII in C

          LD      A,C
          OR      A           ;Don't send NULL characters (SD System Board
          JP      Z,SKIP            ;cannot seem to handle them!).
          CALL    CHARTOBUFFER      ;Just put it in the 256 byte OUTBUFFER
SKIP:     CALL    CLEAR
          RET
```

```
GETSTAT:   IN      A,(KEYSTAT)
           BIT     0,A         ;Bit 0 is Zero if a character is ready
           RET

SETUP_BUFFERS:
           XOR     A
           LD      (CASEFLAG),A      ;Will contain flags for LC, Shift, Ctrl etc.
           LD      IX,CASEFLAG ;IX will ALWAYS point to CASEFLAG
           EXX                 ;Z80 Alt Reg set will hold the "OUT" Buffer data
           LD      DE,OUTBUFFER      ;DE' End of que - Always
           LD      BC,OUTBUFFER      ;BC' Start of que - Always
           LD      (DE),A            ;Flag as nothing there
           EXX
           RET

CLEAR:     PUSH    AF          ;Save value
           LD      A,(IX+0)    ;Routine to clear 74LS161 shift registers
           OR      1H          ;Need to raise bit 0, see below
           OUT     (KEYCLEAR),A      ;LED will also lightup for bits in CASEFLAG
           CALL    DELAY       ;Delay a little (seems to be required!)
           LD      A,(IX+0)    ;Bits 0 & 1 will always be 0
           OUT     (KEYCLEAR),A
           POP     AF
           RET

DELAY:     LD      A,0FFH            ;Slight time delay for hardware
DL1:       DEC     A
           JP      NZ,DL1
           RET

DELAY_LONG:             ;Delay based on value in [A]
           PUSH    BC
           LD      B,A
DEL1:      CALL    DELAY
           DJNZ    DEL1
           POP     BC
           RET

NUMKEY:    BIT     5,(IX+0)    ;Is the NUMBER LOCK already set
           JP      Z,NUMSET
           RES     5,(IX+0)
           CALL    CLEAR
           RET                 ;Note a RET will drop the stack back to the main Loop
NUMSET:    SET     5,(IX+0)    ;set, so trans sees it as NUMLOCK
           CALL    CLEAR
           RET

SHIFTKEY:
           SET     3,(IX+0)    ;Set the SHIFT key, so TRANSLATE sees scan as upper case
           CALL    CLEAR       ;in the tables below
           RET

CTRLKEY:
           SET     4,(IX+0)    ;Set the CTRL key, so TRANSLATE sees scan as ctrl keys
           CALL    CLEAR       ;in the tables below
           RET

CAPSKEY:   BIT     2,(IX+0)    ;Is the CAPS LOCK already set
           JP      Z,CAPSET
           RES     2,(IX+0)
           RES     3,(IX+0)    ;Also do clear the regular shift key flags
           CALL    CLEAR
           RET                 ;Note a RET will drop the stack back to the main Loop
CAPSET:    SET     2,(IX+0)    ;set, flag to indicate CAPS LOCK key down
```

```
          CALL    CLEAR
          RET

UPKEY:                ;Because we are bypassing the Z80/PIO we need special
IF        DEBUG             ;code here for the Diagnostic version.
          LD      A,(DOWN_KEY)      ;It's an UPKEY (0F0H), what was the last key
ELSE
          CALL    CLEAR       ;Clear the register
UPKEY1:   CALL    GETSTAT           ;Wait to find which key we are shifting or ctrl-ing
          JR      NZ,UPKEY1   ;Note:- This is a better way. Fast typing misses keys
          IN      A,(KEYIN)   ;in the diagnostic DOWN_KEY mode.
ENDIF
          CP      14H         ;Was the SHIFT or CTRL held down until now
          JR      Z,CLEAR_CTRL      ;Note Alt keys are not used in this version
          CP      12H
          JR      Z,CLEAR_SHIFT
          CP      59H
          JR      Z,CLEAR_SHIFT
IF        DEBUG
          CP      0H          ;This if for situations where the SHIFT/CTRL is held
          JR      Z,CLEAR_SHIFT     ;down for multiple key strokes and then released.
          JR      Z,CLEAR_CTRL      ;or simply pressed and released (Note both cleared)
ENDIF

NOSHIFT:  XOR     A
          LD      (DOWN_KEY),A
          CALL    CLEAR
          RET               ;Note a RET will drop the stack back to the main Loop

CLEAR_SHIFT:
          RES     3,(IX+0)
          JR      NOSHIFT

CLEAR_CTRL:
          RES     4,(IX+0)
          JR      NOSHIFT
;
;
CHARTOBUFFER:               ;We get here knowing we have a character
          EXX               ;Get the alt reg's that contain the pointers
          LD      (DE),A
          INC     E           ;update pointer (loop 256 bytes)
          XOR     A
          LD      (DE),A            ;Flag for new end of que
          EXX
          RET

CRTOUT:   EXX               ;Send character to actual CRT board
          LD      A,(BC)            ;EXX to get to pointers
          CP      0H
          JP      Z,CRT1            ;nothing in que
          LD      A,(BC)
          INC     C           ;Advance pointer
          OUT     (KEYOUT),A  ;Show ASCII on Hex Display and send it
          LD      A,(IX+0)    ;Need to send strobe bit (raise bit1)
          OR      2H          ;Note bit 0 will always be 0 in CASEFLAG
          OUT     (KEYCLEAR),A      ;LEDs will stay the same
          CALL    DELAY
          CALL    DELAY
          LD      A,(IX+0)    ;Bits 0 & 1 will always be zero
          OUT     (KEYCLEAR),A
          CALL    DELAY
          CALL    DELAY
          CALL    DELAY
KP2:      IN      A,(KEYSTAT)
```

```
          BIT     1,A
          JP      Z,KP2       ;Wait until ACK from SD Systems board returns HIGH
CRT1:     EXX                 ;Back to "normal" register set
          RET

TRANSLATE:                    ;Convert PC Scan codes to ASCII characters
          CP      0E0H        ;Is it the special "prfix" keys . (Print Scr, Scroll lock
                              ;Pause, Ins, Del,...the 4 arrow keys). All these begin with
                              ;E0H.
          JP      NZ,F1_F12_KEYS ;If so just skip them for now, they are done on the number pad.
          CALL    CLEAR       ;anyway
          LD      C,0
          RET
                              ;The function keys and number pad require special treatment
F1_F12_KEYS:                  ;Translate the function keys into multi character strings.
          CP      05H         ;Function key F1 = "DIR *.*[ALL]"
          JR      NZ,NOTF1
          LD      HL,F1_STRING
F1_LOOP:  LD      A,(HL)
          OR      A
          JR      Z,F1_DONE
          CALL    CHARTOBUFFER
          INC     HL
          JR      F1_LOOP
F1_DONE:  CALL    CLEAR
          LD      C,0
          RET

NOTF1:    CP      06          ;Function key F2 = "PIP E:=A:*.*[V]"
          JR      NZ,NOTF2
          LD      HL,F2_STRING
F2_LOOP:  LD      A,(HL)
          OR      A
          JP      Z,F1_DONE
          CALL    CHARTOBUFFER
          INC     HL
          JR      F2_LOOP

NOTF2:                        ;<<<<  Add more strings if needed later
                              ;Note fall through to NUM_PAD below

                              ;We treat the Number pad keys seperately because
                              ;they are independent of the Shift & Ctrl keys
NUM_PAD:  BIT     5,(IX+0)    ;Special treatment for NUMLOCK Keys
          JP      Z,LOOKUP_TABLES
          CP      6CH         ;'7'
          JP      NZ,M1
          LD      C,'7'
          RET
M1:       CP      75H         ;'8'
          JP      NZ,M2
          LD      C,'8'
          RET
M2:       CP      7DH         ;'9'
          JP      NZ,M3
          LD      C,'9'
          RET
M3:       CP      6BH         ;'4'
          JP      NZ,M4
          LD      C,'4'
          RET
M4:       CP      73H         ;'5'
          JP      NZ,M5
          LD      C,'5'
          RET
```

```
M5:       CP      74H         ;'6'
          JP      NZ,M6
          LD      C,'6'
          RET
M6:       CP      69H         ;'1'
          JP      NZ,M7
          LD      C,'1'
          RET
M7:       CP      72H         ;'2'
          JP      NZ,M8
          LD      C,'2'
          RET
M8:       CP      7AH         ;'3'
          JP      NZ,M9
          LD      C,'3'
          RET
M9:       CP      70H         ;'0'
          JP      NZ,M10
          LD      C,'0'
          RET
M10:      CP      71H         ;'.'
          JP      NZ,M11
          LD      C,'.'
          RET
M11:      CP      79H         ;'+'
          JP      NZ,M12
          LD      C,'+'
          RET
M12:      CP      7BH         ;'-'
          JP      NZ,M13
          LD      C,'-'
          RET
M13:      CP      7CH         ;'*'
          JP      NZ,M14
          LD      C,'*'
          RET
M14:      CP      4AH         ;'/'
          JP      NZ,LOOKUP_TABLES ;<<<< Must be a "regular" keyboard key
          LD      C,'/'
          RET
;
; There are 3 possible lookup tables for translating the scan code to ASCII
; depending on the status of the CTRL and Shift keys
; (Alt key's could be added later).

LOOKUP_TABLES:
          LD      B,0
          LD      C,A
          BIT     4,(IX+0)    ;Is it a CTRL key
          JP      Z,NOTCTRL
          LD      HL,CTRLTBL
          ADD     HL,BC
          LD      C,(HL)
          RET
NOTCTRL:
          BIT     2,(IX+0)    ;Is Caps Lock on
          JP      NZ,UCASE

          BIT     3,(IX+0)    ;Is Shift key down
          JP      NZ,UCASE
          LD      HL,IBM1TBL  ;LowerCase (Default)
          ADD     HL,BC
          LD      C,(HL)
          RET
UCASE:    LD      HL,IBM2TBL  ;Upper case
```

```
          ADD     HL,BC
          LD      C,(HL)
          RET
;
IBM1TBL:          ;The "Normal" table
           ;00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f
     DEFB  0,'*',  0,'*','*','*','*','*',  0,'*','*','*','*',09H,'`',00H

          ;10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f
     DEFB  0,  0,  0,  0,  0,'q','1',  0,  0,  0,'z','s','a','w','2',0

           ;20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2a, 2b, 2c, 2d, 2e, 2f
     DEFB  0,'c','x','d','e','4','3',  0,  0,' ','v','f','t','r','5',0

           ;30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 3a, 3b, 3c, 3d, 3e, 3f
     DEFB  0,'n','b','h','g','y','6',  0,  0,  0,'m','j','u','7','8',0

           ;40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 4a, 4b, 4c, 4d, 4e, 4f
     DEFB  0,',','k','i','o','0','9',  0,  0,'.','/','l',';','p', '-',0

           ;50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 5a, 5b, 5c, 5d, 5e, 5f
     DEFB  0,  0,27H,  0,'[','=',  0,  0,  0,  0,0DH,']',  0,'\',   0,0

           ;60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 6a, 6b, 6c, 6d, 6e, 6f
     DEFB  0,  0,  0,  0,  0,  0,  0,08H,  0,  0,11H,  0,13H,10H,  0,  0,  0

           ;70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 7a, 7b, 7c, 7d, 7e, 7f
     DEFB 0BH,7FH,03H,15H,04H,05H,1BH,00H,'*',02H,18H,16H,0CH,17H,'*',0

           ;80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 8a, 8b, 8c, 8d, 8e, 8f
     DEFB  0,  0,  0,'*',  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0

IBM2TBL:          ;If the shift key or caps lock key is on
           ;00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f
     DEFB  0,'*',  0,'*','*','*','*','*',  0,'*','*','*','*',09H,'~',00H

           ;10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f
     DEFB  0,  0,  0,  0,  0,'Q','!',  0,  0,  0,'Z','S','A','W','@',0

           ;20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2a, 2b, 2c, 2d, 2e, 2f
     DEFB  0,'C','X','D','E','$','#',  0,  0,' ','V','F','T','R','%',0

           ;30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 3a, 3b, 3c, 3d, 3e, 3f
     DEFB  0,'N','B','H','G','Y','^',  0,  0,  0,'M','J','U','&','*',0

           ;40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 4a, 4b, 4c, 4d, 4e, 4f
     DEFB  0,'<','K','I','O',29H,'(',  0,  0,'>','?','L',':','P', '_',0

           ;50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 5a, 5b, 5c, 5d, 5e, 5f
     DEFB  0,  0,22H,  0,'{','+',  0,  0,  0,  0,0DH,'}',  0,'|',  0,0

           ;60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 6a, 6b, 6c, 6d, 6e, 6f
     DEFB  0,  0,  0,  0,  0,  0,  0,08H,  0,  0,11H,  0,13H,10H,  0,  0,  0

           ;70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 7a, 7b, 7c, 7d, 7e, 7f
     DEFB 0BH,7FH,03H,15H,04H,05H,1BH,00H,'*',02H,18H,16H,0CH,17H,'*',0

           ;80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 8a, 8b, 8c, 8d, 8e, 8f
     DEFB  0,  0,  0,'*',  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0


CTRLTBL:          ;If the CTRL key is pressed
           ;00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f
     DEFB  0,'*',  0,'*','*','*','*','*',  0,'*','*','*','*',09H,'~',00H
```

```
            ;10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f
     DEFB  0,  0,  0,  0,  0,11H,'!',  0,  0,  0,1AH,13H,01H,17H,'@',0

            ;20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2a, 2b, 2c, 2d, 2e, 2f
     DEFB  0,03H,18H,04H,05H,'$','#',  0,  0,' ',16H,06H,14H,12H,'%',0

            ;30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 3a, 3b, 3c, 3d, 3e, 3f
     DEFB  0,0EH,02H,08H,07H,19H,'^',  0,  0,  0,0DH,0AH,15H,'&','*',0

            ;40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 4a, 4b, 4c, 4d, 4e, 4f
     DEFB  0,'<',0BH,09H,0FH,49H,'(',  0,  0,'>','?',0CH,':',10H, '_',0

            ;50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 5a, 5b, 5c, 5d, 5e, 5f
     DEFB  0,  0,22H,  0,1BH,'+',  0,  0,  0,  0,0DH,1DH,  0,1DH,  0,0

            ;60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 6a, 6b, 6c, 6d, 6e, 6f
     DEFB  0,  0,  0,  0,  0,  0,08H,  0,  0,11H,  0,13H,  0,10H,  0,  0

            ;70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 7a, 7b, 7c, 7d, 7e, 7f
     DEFB 0BH,7FH,03H,15H,04H,05H,1BH,00H,'*',06H,12H,01H,0CH,0EH,'*',0

            ;80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 8a, 8b, 8c, 8d, 8e, 8f
     DEFB  0,  0,  0,'*',  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0
;
;
F1_STRING: DB      'DIR *.*[ALL]',CR,0           ;Show Directory
F2_STRING: DB      'PIP E:=A:*.*[V]',CR,0        ;Copy stuff across to memory disk

MSG0:      DEFM 'IBM-> ASCII KEYBOARD ROM Controller  John Monahan 15/2/2009'

IF NOT     DEBUG
            ORG   800H        ;Hardware RAM chip will start here
ENDIF

OUTBUFFER: DEFS    257
            DEFB  '$'
CASEFLAG:  DEFB    0H          ;Flags for Shift, Numlock etc
DOWN_KEY:  DEFB    0H          ;Diagnostic program needs to store previous char
;
;
;
IF         DEBUG
;
;-------------------------DIAGNOSTIC TESTING CODE -------------------------------
;
DIAGNOSTIC:
           LD      D,A         ;Store scan code in D
           CALL    CRLF
           LD      A,D
           CP      0F0H        ;Skip displaying Up key info
           JR      Z,NormalKey1
           CP      77H         ;Is it the NUM LOCK key
           JR      NZ,NotNum
           LD      HL,SCANCODENUM
           CALL    PMSG
           JP      NormalKey1
NotNum:    CP      58H         ;Is it CAPS Lock key
           JP      NZ,NotCaps
           LD      HL,SCANCODECAPS
           CALL    PMSG
           JP      NormalKey1
NotCaps:
           CP      12H         ;Is it a SHIFT key
           JP      NZ,NotShift1
           LD      HL,SCANSHIFTKEY
```

```
           CALL    PMSG
           JP      NormalKey1
NotShift1:
           CP      59H         ;Is it the other SHIFT key
           JP      NZ,NotShift2
           LD      HL,SCANSHIFTKEY
           CALL    PMSG
           JP      NormalKey1
NotShift2:
           CP      14H         ;Is it the CTRL key
           JP      NZ,NormalKey
           LD      HL,SCANCTRLKEY
           CALL    PMSG
           JP      NormalKey1
NormalKey:
           LD      HL,SCANCODE
           CALL    PMSG
           LD      A,D
           CALL    PACC        ;Show the scan code
           LD      HL,SCANCODE1
           CALL    PMSG
           LD      A,D
           CALL    ZBITS       ;Show the scan code in bits
           LD      HL,SCANCODE2
           CALL    PMSG
NormalKey1:
           LD      A,D
           RET
;
;
PMSG:      LD      A,(HL)      ;PRINT MESSAGE STRING in [HL] up to 0
           OR      A
           RET     Z
           LD      C,A
           CALL    CO
           INC     HL
           JP      PMSG
;
;Print the acccumulator value on CRT in HEX-ASCII
PACC:      PUSH    AF
           PUSH    BC
           PUSH    AF
           RRCA
           RRCA
           RRCA
           RRCA
           CALL    ZCONV
           POP     AF
           CALL    ZCONV
           POP     BC
           POP     AF
           RET

ZCONV:     AND     A,0FH       ;HEX to ASCII
           ADD     90H
           DAA
           ADC     40H
           DAA
           LD      C,A
           CALL    CO
           RET
;
;DISPLAY BIT PATTERN IN [A]
;
ZBITS:     PUSH    AF
```

```
          PUSH    BC
          PUSH    DE
          LD      E,A
          LD      B,8
BQ2:      SLA     E
          LD      A,18H
          ADC     A
          LD      C,A
          CALL    CO
          DJNZ    BQ2
          POP     DE
          POP     BC
          POP     AF
          RET
;
; Display a character on the CRT in my case it's an SD Systems 8024 Video board.
;
CO:       IN      A,(0)       ;console output (arrive with character in C)
          AND     A,04H       ;Note character is in C and A on return.
          JR      Z,CO
          LD      A,C
          OUT     (1),A
          RET
;
CRLF      LD      C,CR
          CALL    CO
          LD      C,LF
          CALL    CO
          RET
;
;
SIGNON:   DB      'Diagnostic Program for S-100 PC keyboard Converter Board V1.0',CR,LF
          DB      'Will test that the 74LS164 shift register circuitry is working OK',CR,LF
          DB      'Press any key on the PC keyboard',CR,LF,0
SCANCODE: DB      'Keyboard Scan code = ',0
SCANCODE1:DB      '  (',0
SCANCODE2:DB      ')  Translated ASCII= ',0
SCANCODE3:DB      CR,LF,0
SCANCODEUP: DB    'Scan code for Up key',CR,LF,0
SCANCODENUM: DB   'Will translate from Number Lock Table',CR,LF,0
SCANCODECAPS: DB 'Scancode for Shift Lock key',CR,LF,0
SCANSHIFTKEY: DB 'Scancode for Shift Key',CR,Lf,0
SCANCTRLKEY: DB 'Scancode for Ctrl key',CR,LF,0
;
          DS      20H
STACK     DB      0H
ENDIF
;
;END
```