```
;
; Hardware Debug Monitor for 8080/8085/Z80 processor family
;
; This simple monitor requires NO RAM of any sort. You can use it to
; poke around the system as long as the CPU, a ROM and the UART are
; working. (Hint: if the UART is not working, you can use the polling
; by HDM86 to track down its accessing and debug it).
;
; This monitor provides basic read/modify memory, as well as
; Loop Read/Write functions (useful when hardware debugging).
;
; Commands (must be UPPER-CASE):
;  Mxxxx      - Display memory at specified address
;               Displays one line, enter SPACE for another, RETURN to stop
;  Exxxx      - Edit memory at specified address
;               Prompts with location, enter two digit HEX value, then
;               SPACE to proceed to next or RETURN to stop
;  Gxxxx      - Go (Begin Execution) at specified address
;  Rxxxx      - Perform Loop-Read at specified address (RESET to stop)
;  Wxxxx dd   - Perform Loop-Write dd -> specified address (RESET to stop)
;
; Build with DDS XASM tools:
;       MACRO HDM80.MAC >HDM80.ASM
;       ASM85 HDM80 -t
;
; Dave Dunfield - March 28, 2005
;
        ORG     $0000
;       ORG     $F000
;
; Macro to read a character into A
; Modify this macro to perform character input on your hardware.
; No registers other than A may be used.
;
;
; Macro to write character in L
; Modify this macro to perform character output on your hardware
; No registers other than A & L may be used
;
;
; Macro to perform a SUBROUTINE call using one level stack in SP
; Even though HL is destroyed by RET, the use of SP instead of HL
; as the return address holding register allows HL to be used as
; temporary storage within the subroutine.
;
;
; Macro to perform a RETURN to address in SP
; Note: Destroys HL
;
;
```

```
; Initialize hardware
;
HINIT   MVI    A,$80
        OUT    $6B               ; SET DLAB FLAG
        MVI    A,12              ; = 1,843,200 / ( 16 x 9600 )
        OUT    $68               ; Set BAUD rate til 9600
        MVI    A,$00
        OUT    $69               ; Set BAUD rate til 9600
        MVI    A,$03
        OUT    $6B               ; Set 8 bit data, 1 stopbit

;
; Main loop - issue prompt & wait for command
;
TOP     LXI     SP,*+6           ; SP = return address
        JMP     LFCR             ; Branch to subroutine
        MVI     L,'>'            ; Prompt
putc2   IN      $6D              ; Read status
        ANI     %00100000        ; TX ready?
        JZ      putc2            ; Wait for it
        MOV     A,L              ; Get data
        OUT     $68
getc3   IN      $6D              ; Read status
        ANI     %00000001        ; RX ready?
        JZ      getc3            ; No, wait
        IN      $68              ; Read data
        OUT     $68
;
; Memory dump command
;
        CPI     'M'              ; Memory
        JNZ     EDIT             ; No, try next
        LXI     SP,*+6           ; SP = return address
        JMP     GETH             ; Branch to subroutine
        MOV     D,A              ; Set high
        LXI     SP,*+6           ; SP = return address
        JMP     GETH             ; Branch to subroutine
        MOV     E,A              ; Set low
MD1     LXI     SP,*+6           ; SP = return address
        JMP     LFCR             ; Branch to subroutine
        MOV     A,D              ; Get high address
        LXI     SP,*+6           ; SP = return address
        JMP     PUTH             ; Branch to subroutine
        MOV     A,E              ; Get low address
        LXI     SP,*+6           ; SP = return address
        JMP     PUTH             ; Branch to subroutine
        MVI     C,16             ; Display 16 bytes
MD2     LXI     SP,*+6           ; SP = return address
        JMP     SPACE            ; Branch to subroutine
        LDAX    D                ; Get data from memory
        INX     D               ; Advance to next
        LXI     SP,*+6          ; SP = return address
        JMP     PUTH            ; Branch to subroutine
        DCR     C               ; Reduce count
        JNZ     MD2             ; Display them all
        LXI     SP,*+6          ; SP = return address
        JMP     PAUSE           ; Branch to subroutine
        JMP     MD1             ; And proceed
; Substitute command
EDIT    CPI     'E'             ; Edit
        JNZ     go              ; No, try next
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     D,A             ; Set high address
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     E,A             ; Set low address
edi1    LXI     SP,*+6          ; SP = return address
        JMP     LFCR            ; Branch to subroutine
        MOV     A,D             ; Get high address
        LXI     SP,*+6          ; SP = return address
        JMP     PUTH            ; Branch to subroutine
        MOV     A,E             ; Get low address
        LXI     SP,*+6          ; SP = return address
        JMP     PUTH            ; Branch to subroutine
        LXI     SP,*+6          ; SP = return address
        JMP     SPACE           ; Branch to subroutine
        LDAX    D               ; Get data
        LXI     SP,*+6          ; SP = return address
        JMP     PUTH            ; Branch to subroutine
        MVI     L,'='           ; Prompt
putc19  IN      $6D             ; Read status
        ANI     %00100000       ; TX ready?
        JZ      putc19          ; Wait for it
        MOV     A,L             ; Get data
        OUT     $68
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        STAX    D               ; Store it
        INX     D               ; Next
        LXI     SP,*+6          ; SP = return address
        JMP     PAUSE           ; Branch to subroutine
        JMP     edi1            ; And get next
; Go (execute)
go      CPI     'G'             ; Go?
        JNZ     lread           ; No, try next
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     D,A             ; Set high address
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     E,A             ; Set low address
        XCHG                    ; HL = address
        PCHL                    ; PC = address
; Loop read
lread   CPI     'R'             ; Read (loop)?
        JNZ     lwrite          ; No, try next
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     D,A             ; Set high address
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     E,A             ; Set low address
        LXI     SP,*+6          ; SP = return address
        JMP     LFCR            ; Branch to subroutine
lr1     LDAX    D               ; Read the data
        JMP     lr1             ; And continue (forever)
; Loop write
lwrite  CPI     'W'             ; Write (loop)?
        JNZ     error           ; No, try next
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     D,A             ; Set high address
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     E,A             ; Set low address
        LXI     SP,*+6          ; SP = return address
        JMP     SPACE           ; Branch to subroutine
        LXI     SP,*+6          ; SP = return address
        JMP     GETH            ; Branch to subroutine
        MOV     B,A             ; Save
        LXI     SP,*+6          ; SP = return address
        JMP     LFCR            ; Branch to subroutine
        MOV     A,B             ; Restore data
lw1     STAX    D               ; Write
        JMP     lw1             ; And continue (forever)
;
; Error has occured - issue indicator and wait next command
;
ERROR   MVI     L,'?'           ; Error indicator
putc32  IN      $6D             ; Read status
        ANI     %00100000       ; TX ready?
        JZ      putc32          ; Wait for it
        MOV     A,L             ; Get data
        OUT     $68
        JMP     TOP             ; New command
;
; Output LFCR
;
LFCR    MVI     L,$0A           ; Line-feed
```

```
putc33  IN      $6D              ; Read status
        ANI     %00100000        ; TX ready?
        JZ      putc33           ; Wait for it
        MOV     A,L              ; Get data
        OUT     $68
        MVI     L,$0D            ; Carriage-return
XOUT    EQU     *                ; LABLE ADDRESS
putc34  IN      $6D              ; Read status
        ANI     %00100000        ; TX ready?
        JZ      putc34           ; Wait for it
        MOV     A,L              ; Get data
        OUT     $68
XRET    LXI     H,0              ; Get zero
        DAD     SP               ; Get address from SP
        PCHL                     ; PC = return address
;
; Wait for key SPACE=proceed, RETURN=end command
;
PAUSE   EQU     *                ; LABLE ADDRESS
getc36  IN      $6D              ; Read status
        ANI     %00000001        ; RX ready?
        JZ      getc36           ; No, wait
        IN      $68              ; Read data
        OUT     $68
        CPI     $0D              ; End?
        JZ      TOP              ; Yes - exit
        CPI     ' '              ; Continue?
        JNZ     PAUSE            ; Wait for it
        JMP     XRET
;
; Output a space
;
SPACE   MVI     L,' '            ; Get space
        JMP     XOUT             ; Output & return
;
; Output A in HEX
;
PUTH    MOV     H,A              ; Save for later
        RRC                      ; Shift
        RRC                      ; High
        RRC                      ; Into
        RRC                      ; Low
        ANI     %00001111        ; Save only LOW
        CPI     10               ; In range?
        JC      puth1            ; Yes, no adjust
        ADI     7                ; Adjust for alpha
puth1   ADI     '0'              ; Convert to ASCII
        MOV     L,A              ; Set output
putc37  IN      $6D              ; Read status
        ANI     %00100000        ; TX ready?
        JZ      putc37          ; Wait for it
        MOV     A,L             ; Get data
        OUT     $68
        MOV     A,H             ; Get data
        ANI     %00001111       ; Save only LOW
        CPI     10              ; In range?
        JC      puth2           ; Yes, no adjust
        ADI     7               ; Adjust for alpha
puth2   ADI     '0'             ; Convert to ASCII
        MOV     L,A             ; Set output
        JMP     XOUT            ; Output & return
;
; Get HEX character in A
;
GETH    EQU     *               ; LABLE ADDRESS
getc38  IN      $6D             ; Read status
        ANI     %00000001       ; RX ready?
        JZ      getc38          ; No, wait
        IN      $68             ; Read data
        OUT     $68
        SUI     '0'             ; Convert
        JC      ERROR           ;
        CPI     10              ; In range?
        JC      geth1           ; It's ok
        SUI     7               ; Additional convert
        CPI     10              ; In range?
        JC      error           ; No - error
        CPI     16              ; In range?
        JNC     error           ; No error
geth1   RLC                     ; Shift over
        RLC                     ; Shift over
        RLC                     ; Shift over
        RLC                     ; Shift over
        ANI     %11110000       ; Keep only top
        MOV     L,A             ; Save for later
getc39  IN      $6D             ; Read status
        ANI     %00000001       ; RX ready?
        JZ      getc39          ; No, wait
        IN      $68             ; Read data
        OUT     $68
        SUI     '0'             ; Convert
        JC      ERROR           ;
        CPI     10              ; In range?
        JC      geth2           ; It's ok
        SUI     7               ; Additional convert
        CPI     10              ; In range?
        JC      error           ; No - error
        CPI     16              ; In range?
        JNC     error           ; No error
geth2   ORA     L               ; Include high nibble
```

```
        JMP     XRET
```