```
; CodeDump
;
; Dump code starting at memory location CD_CurLoc to the screen


LinesShown      .equ     16

CodeDump:
        call    CD_RegDump              ; Dump the registers

CD_DispLoop:
        ld      hl,(CD_CurLoc)          ; Get the current location
        ld      b,LinesShown
CD_DispLoc:
        call    CD_DispHL
        call    CD_Space
        ld      de,CD_Buffer
        call    CD_Show4
        call    CD_Space
        ld      (de),a                  ; Put space in ASCII buffer too
        inc     de
        call    CD_Show4
        call    CD_Space
        call    CD_ShowAscii
        djnz    CD_DispLoc

        ld      hl,CD_Prompt
        call    CD_PrintString             ; Show Prompt
CD_CommandLoop:
        call    0FC6Ch                     ; Get key
        cp      '+'                     ; Is it next page
        jp      z,CD_NextPage
        cp      13                      ; Is it next page
        jp      z,CD_NextPage
        cp      '-'                     ; Is it previous page
        jp      z,CD_PrevPage
        cp      '!'                     ; Is it reset?
        jp      z,0fc30h
        cp      '@'                     ; Is it Smart Writer?
        jp      z,0FCE7h
```

```
        jp      CD_CommandLoop

CD_NextPage:                            ; +
        ld      hl,(CD_CurLoc)
        ld      de,LinesShown * 8
        add     hl,de
        ld      (CD_CurLoc),hl
        call    CD_RegDump1
        jp      CD_DispLoop

CD_PrevPage:                            ; -
        ld      hl,(CD_CurLoc)
        ld      de,LinesShown * 8
        or      a
        sbc     hl,de
        ld      (CD_CurLoc),hl
        call    CD_RegDump1
        jp      CD_DispLoop


CD_DispHL:
        ld      a,h                     ; Show the current address
        call    CD_ShowA
        ld      a,l
        call    CD_ShowA
        ret

CD_ShowAscii:
        push    bc
        ld      de,CD_Buffer
        ld      b,9
CD_SA_Loop:
        ld      a,(de)
        call    0FC33h
        inc     de
        djnz    CD_SA_Loop
        pop     bc
        ret

CD_Show4:
        push    bc
```

```
        ld      b,4
CD_S4_Loop:
        ld      a,(hl)                  ; Get value to display
        ld      (de),a                  ; Save in buffer for ASCII display
        inc     de
        call    CD_ShowA                ; Display the value as hex
        inc     hl
        djnz    CD_S4_Loop
        pop     bc
        ret

CD_ShowA:
        push    de
        call    CD_NumToHex
        ld      a,d
        call    0FC33h
        ld      a,e
        call    0FC33h
        pop     de
        ret

CD_Space:
        ld      a,32
        jp      0FC33h

; CD_RegDump
;
; Saves all the registers, clears screen and displays them. Clobbers all registers
;
; Use CD_RegDump1 to just display saved registers

CD_RegDump:
        ld      (CD_REG_BC),bc          ; Save the register values
        ld      (CD_REG_DE),de
        ld      (CD_REG_HL),hl
        ld      (CD_REG_SP),sp
        ld      (CD_REG_IX),ix
        ld      (CD_REG_IY),iy
        push    af
        pop     hl
        ld      (CD_REG_AF),hl
```

```
CD_RegDump1:
        ld      a,0ch
        call    0fc39h                  ; Clear Screen
        ld      hl,CD_BC
        call    CD_PrintString          ; Print Reg Display
        ld      hl,(CD_REG_BC)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CD_DE
        call    CD_PrintString             ; Print Reg Display
        ld      hl,(CD_REG_DE)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CD_HL
        call    CD_PrintString          ; Print Reg Display
        ld      hl,(CD_REG_HL)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CD_SP
        call    CD_PrintString          ; Print Reg Display
        ld      hl,(CD_REG_SP)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CRLF
        call    CD_PrintString

        ld      hl,CD_IX
        call    CD_PrintString          ; Print Reg Display
        ld      hl,(CD_REG_IX)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CD_IY
        call    CD_PrintString          ; Print Reg Display
        ld      hl,(CD_REG_IY)
        call    CD_DispHL               ; Print stored reg value

        ld      hl,CD_A
        call    CD_PrintString          ; Print Reg Display
        ld      a,(CD_REG_AF)
        call    CD_ShowA                ; Print stored reg value
```

```
        ld      hl,CD_F
        call    CD_PrintString          ; Print Reg Display

        ld      hl,CD_F_Disp            ; Flag display
        ld      a,(CD_REG_AF + 1)       ; Get the flag saved
        ld      b,a                     ; Put in b to work with

        ld      a,(hl)                  ; Get the Flag Display
        bit     7,b                     ; Sign set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      a,(hl)                  ; Get the Flag Display
        bit     6,b                     ; Zero set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      a,(hl)                  ; Get the Flag Display
        call    CD_R2                   ; Unused flag
        ld      a,(hl)                  ; Get the Flag Display
        bit     4,b                     ; Half Carry set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      a,(hl)                  ; Get the Flag Display
        call    CD_R2                   ; Unused flag
        ld      a,(hl)                  ; Get the Flag Display
        bit     2,b                     ; Parity Overflow set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      a,(hl)                  ; Get the Flag Display
        bit     1,b                     ; Subtraction set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      a,(hl)                  ; Get the Flag Display
        bit     0,b                     ; Carry set?
        call    z,CD_R2                 ; No
        call    nz,CD_R1                ; Yes
        ld      hl,CRLF
        call    CD_PrintString
        ret

CD_R1:
        and     011011111b              ; Turn off lower case
```

```
CD_R2:
       inc     hl                      ; Next flag
       jp      0FC33h                  ; Display the byte


CD_PrintString:
       push    af
       push    hl
CD_PS_1:
       ld      a,(hl)                      ; Get character
       or      a                           ; Is it a zero?
       jr      z,CD_PS_Exit                ; Yes exit
       call    0fc39h
       inc     hl
       jr      CD_PS_1
CD_PS_Exit:
       pop     hl
       pop     af
       ret

; Convert number value in A to ASCII hex in DE
;
CD_NumToHex:
       push    bc
       push    af
       ld      c,a                 ; a = number to convert
       call    CD_Num1
       ld      d,a
       ld      a,c
       call    CD_Num2
       ld      e,a
       pop     af
       pop     bc
       ret                         ; return with hex number in de
CD_Num1:
       rra
       rra
       rra
       rra
CD_Num2:
       or      0F0h
```

```
        daa
        add     a,0A0h
        adc     a,040h              ; Ascii hex at this point (0 to F)
        ret

CD_CurLoc:      .dw     0234h
CD_Buffer:      .db     0,0,0,0,32,0,0,0,0
CD_REG_BC       .dw     0
CD_REG_DE       .dw     0
CD_REG_HL       .dw     0
CD_REG_SP       .dw     0
CD_REG_IX       .dw     0
CD_REG_IY       .dw     0
CD_REG_AF       .dw     0

CD_BC:          .db     "BC:",0
CD_DE:          .db     " DE:",0
CD_HL:          .db     " HL:",0
CD_SP:          .db     " SP:",0
CD_IX:          .db     "IX:",0
CD_IY:          .db     " IY:",0
CD_A:           .db     "  A:",0
CD_F:           .db     " F:",0
CD_Prompt:      .db     13,10
                .text   "[+]Next -Previous !Reset @SW ?"
                .db     0
CD_F_Disp:      .text   "sz-h-pnc"
```