```
; hello.asm
;
; This is a basic hello world program in z80 for the ADAM computer.
; I will add comments so that you have an idea what it is doing but
; this is not meant as a tutorial on how to program in z80.
;
; William (Milli) Hicks 12/30/2017

; These equates are memory address where EOS routines reside. For
; details see the EOS programmers handbook

WRITEVDP        .equ  0FD20h          ; Write data to a VDP register
INITVRAM        .equ  0FD29h          ; Initialize VRAM tables
DEFAULTASCII    .equ  0FD38h          ; Copy the default 128 character font to VRAM
PUTASCII        .equ  0FD17h          ; Copy font data to VRAM
FILLVRAM        .equ  0FD26h          ; Fill VRAM with data
INITCONSOLE     .equ  0FC36h          ; Initialize the console settings
DISPREGULAR     .equ  0FC33h          ; Display character on screen
READKEYBOARD    .equ  0FC6Ch          ; Wait for a key press and return it

SMARTWRITER     .equ  0FCE7h          ; Exit to Smart Writer

SCREENFONT      .equ  0h              ; Where the font table is in VRAM
SCREENDATA      .equ  0800h           ; Where the screen data is in VRAM
FONTCOLORS      .equ  0bc0h           ; Where the font color tables are in VRAM

; Values used when talking to the VDP

VDPBORDER       .equ  0700h           ; Value used to set the border color in 32 col mode
VDP32COL        .equ  01c0h           ; Tell the VDP we are doing 32 column mode
VDP40COL        .equ  01d0h           ; Tell the VDP we are doing 40 column mode

; Text Color Equates

TRANS           .equ  0
BLACK           .equ  1
MGREEN          .equ  2
LGREEN          .equ  3
DBLUE           .equ  4
LBLUE           .equ  5
DRED            .equ  6
```

```
CYAN            .equ  7
MRED            .equ  8
LRED            .equ  9
DYELLOW         .equ  10
LYELLOW         .equ  11
DGREEN          .equ  12
MAGENTA         .equ  13
GRAY            .equ  14
WHITE           .equ  15
FRGRND          .equ  16

TEXTCOLOR       .equ  WHITE           ; Text color in 40 column mode. 32 column uses font colors
BACKCOLOR       .equ  DBLUE           ; Background color used in all modes

TRUE            .equ  1
FALSE           .equ  0

                                      ; Make this TRUE for 32 column, FALSE for 40 column
DO32COLUMN      .equ  FALSE

#if DO32COLUMN                        ; If DO32COLUMN = TRUE then use these defines

                                      ; Set the border color in 32 column mode to the same as
                                      ; the background
BORDERCOLOR     .equ  VDPBORDER + BACKCOLOR
SCREENSIZE      .equ  VDP32COL        ; Value to tell VDP for 32 column
COLUMNS         .equ  32
ROWS            .equ  24

#else                                 ; Otherwise use these for 40 column

                                      ; Set the background and foreground color for all
                                      ; text in 40 column mode
BORDERCOLOR     .equ  VDPBORDER + (TEXTCOLOR * 16) + BACKCOLOR
SCREENSIZE      .equ  VDP40COL        ; Value to tell VDP for 40 column
COLUMNS         .equ  32
ROWS            .equ  24

#endif
```

```
        .org    100h                ; Where in memory this program will reside

; Set up text mode so we can see something on the screen


        ld      bc,0000h            ; VDP Reg 0 (Text mode)
        call    WRITEVDP
        ld      bc,SCREENSIZE       ; VDP Reg 1 (32 column mode)
        call    WRITEVDP
                                    ; Setup initial VRAM tables
        ld      hl,SCREENFONT       ; Pattern Generator Address (the font data)
        ld      a,03h               ; 3 = Pattern generator
        call    INITVRAM
        ld      hl,SCREENDATA       ; Pattern Name Table Address (the screen data)
        ld      a,02h               ; 2 = Pattern name table
        call    INITVRAM
        call    DEFAULTASCII        ; Load default ASCII table
        ld      de,0400h            ; Copy the EOS characters to the VRAM
        ld      hl,SCREENFONT
        ld      bc,0080h
        call    PUTASCII
        ld      hl,FONTCOLORS       ; Pattern color table address
        ld      a,04h               ; 4 = Pattern color
        call    INITVRAM
                                    ; Set the border
        ld      bc,BORDERCOLOR
        call    WRITEVDP

        ld      c,TEXTCOLOR         ; Set the font colors for 32 column mode (even if 40 column)
        ld      b,BACKCOLOR
        ld      d,0
        ld      e,0
        ld      hl,FONTCOLORS
        add     hl,de               ; HL holds the VRAM address of the color group to change
        ld      de,32               ; Only changing one entry
        ld      a,c                 ; Put foreground in A
        rlc     a                   ; Multiply by 16
        rlc     a
        rlc     a
```

```
        rlc     a
        add     a,b                 ; Add the background
        call    FILLVRAM



        ld      a,32                ; Screen background character (space)
        ld      de,COLUMNS * ROWS   ; Number of spaces to fill
        ld      hl,SCREENDATA       ; Address of screen
        call    FILLVRAM
                                    ; 32 Column - Set the background
                                    ; 40 Column - Set the border color and 40 column text color

        ld      b,COLUMNS - 1       ; Number of columns
        ld      c,ROWS - 1          ; Number of rows
        ld      d,0                 ; home cursor to the
        ld      e,0                 ; Top Left
        ld      hl,SCREENDATA
        call    INITCONSOLE

; At this point the screen is all setup so we can actually now say "Hello World"

        ld      hl,Message          ; Where the "Hello World" message is in memory
Loop:
        ld      a,(hl)              ; Get next character to print to the screen
        cp      0                   ; Is it a Zero
        jp      z,Exit              ; Yes so we are done, exit
        call    DISPREGULAR         ; Go print it
        inc     hl                  ; Point to the next character
        jp      Loop                ; Go do more

Exit:
        call    READKEYBOARD        ; Wait for a key press
        jp      SMARTWRITER         ; All done exit to Smart Writer

Message:        .text "Hello World"
                .db  0
        .end
```