```
;
; display_hex_digit_at_pos.asm
;
; Created: 10/21/2020 2:58:57 PM
; Author : tyler ovenden
; 112122685
;


; Replace with your application code
.nolist
.include "m4809def.inc"
.list

; Replace with your application code
    start:
 ; configure I/O ports
    ldi r16, 0x00               ;change r17 to all 0s for input
    out VPORTA_DIR, r16         ;PORTA - all pins configured as inputs
    ldi r17, 0xFF               ;load r17 with all 1s
    out VPORTE_DIR, r17         ;sets PORTE as an output
    out VPORTD_DIR, r17         ;sets PORTF as an output
    out VPORTD_OUT, r17         ;sets PORTD as output

    main_loop:
    out VPORTD_OUT, r17
    sbis VPORTE_IN, 1      ;checks if flip flop is on, button is pushed
    rjmp main_loop         ;goes back to beginning of loop if button released
    rjmp take_in           ;goes to display if button pushed

    take_in:
    ldi r18, 0x00                     ;sets r18 to a blank register
    ldi r16, VPORTA_IN                ;loads r16 with switch inputs
    rcall reverse                     ;reverses r16
    mov r19, r16                     ;moves reversed number into new registert to get ↩
      bits representing display digits
    andi r19, 0x03                 ;ands r19 with 0000 0011 to get only first two bits↩

    lsr r16                       ;shifts r16 4 times to get only 4 bits
    lsr r16
    lsr r16
    lsr r16
    rcall hex_to_7seg               ;converts hex number to 7 segment display pattern
    ldi r20, 0xFF                   ;create a register representing first digit with  ↩
      everything off at first
    mov r21, r20                     ;create a register representing 2nd digit with   ↩
      everything off at first
    mov r22, r20                      ;create a register representing 3rd digit with  ↩
      everything off at first
    mov r23, r20                      ;create a register representing 4th digit with  ↩
      everything off at first
    sbi VPORTE_IN , 1               ;clears flip flop
    cpi r19, 0x00                      ;check if  switch is set to first digit
    breq first_digit                     ;branches to set first digit if r19 = 0
    cpi r19, 0x01                        ;check if  switch is set to 2nd digit
    breq second_digit                     ;branches to set 2nd digit if r19 = 1
    cpi r19, 0x02                         ;check if  switch is set to 3rd digit
    breq third_digit                      ;branches to set 3rd digit if r19 = 2
    cpi r19, 0x03                        ;check if  switch is set to 4th digit
    breq fourth_digit                     ;branches to set 4th digit if r19 = 3
    rjmp main_loop                     ;goes back to main loop

    ;***************************************************************************
;*
;* "reverses" - reverses a register
;*
;* Description: Reverses a register using two different registers.
;* shifts r16 then moving that shifted bit into r17 8 times to reverse
;*
;* Author:                        Tyler Ovenden
;* Version:                       1.0
;* Last updated:                  102120
;* Target:                        ATmega4809
;* Number of words:               11
;* Number of cycles:
;* Low registers modified:        none
;* High registers modified:       r16, r17
;*
;* Parameters: r16: input from switch
;* Returns: r16: reversed switch input, shifted 4 times to get only bits 7-4 from
  reversed bit
;*
;* Notes:
;*
;***************************************************************************

    reverse:
    lsr r16                         ;shifts r16 once putting msb in flag
    rol r17                         ;rotates r17 once placing carry bit from lsr into r17
    cpi r16, 0x00                   ;checks if r16 is all 0
    brne reverse                    ;if r16 is not 0 then repeat loop
    mov r16, r17                    ;moves reversed number in r17 to r16
    ret                             ;ends subroutine
    ;***************************************************************************
;*
;* "hex_to_7seg" - Hexadecimal to Seven Segment Conversion
;*
;* Description: Converts a right justified hexadecimal digit to the seven
;* segment pattern required to display it. Pattern is right justified a
;* through g. Pattern uses 0s to turn segments on ON.
;*
;* Author:                        Ken Short
;* Version:                       1.0
;* Last updated:                  101620
;* Target:                       ATmega4809
;* Number of words:              8
;* Number of cycles:             13
;* Low registers modified:       none
;* High registers modified:      r16, r18, ZL, ZH
;*
;* Parameters: r18: right justified hex digit, high nibble 0
;* Returns: r18: segment values a through g right justified
;*
;* Notes:
;*
;***************************************************************************
hex_to_7seg:
    andi r18, 0x0F              ;clear ms nibble
    ldi ZH, HIGH(hextable * 2)  ;set Z to point to start of table
    ldi ZL, LOW(hextable * 2)
    ldi r16, $00                ;add offset to Z pointer
    add ZL, r18
    adc ZH, r16
    lpm r18, Z                  ;load byte from table pointed to by Z
    ret

    ;Table of segment values to display digits 0 - F
    ;!!! seven values must be added - verify all values
hextable: .db $01, $4F, $12, $06, $4C, $24, $20, $0F, $00, $04, $08, $60, $31, $32,
  $30, $38

display:
out VPORTD_OUT, r20             ;sets 7 segment display value for first display digit
ldi r16, 0x7F                   ;sets value for transitors to display 1st digit
out VPORTC_OUT, r16             ;turns on 1st digit
out VPORTD_OUT, r21             ;sets 7 segment display value for 2nd display digit
ldi r16, 0xBF                    ;sets value for transitors to display 2nd digit
out VPORTC_OUT, r16             ;turns on 2nd digit
out VPORTD_OUT, r22             ;sets 7 segment display value for 3rd display digit
ldi r16, 0xDF                    ;sets value for transitors to display 3rd digit
out VPORTC_OUT, r16             ;turns on 3rd digit
out VPORTD_OUT, r23             ;sets 7 segment display value for 4th display digit
ldi r16, 0xBF                    ;sets value for transitors to display 4th digit
out VPORTC_OUT, r16              ;turns on 4th digit
sbis VPORTE_IN, 1                ;checks if flip flop output is 1, pushed down
rjmp display                     ;loops display if flip flop released
rjmp take_in                     ;restarts loop to take in switch values


first_digit:
mov r20, r18                     ;places hex digits into register reprsenting 1st
  display digit
rjmp display                     ;calls display for 7 segment display

second_digit:
mov r21, r18                    ;places hex digits into register reprsenting 2nd
```

```
  display digit
rjmp display                    ;calls display for 7 segment display

third_digit:
mov r22, r18                    ;places hex digits into register reprsenting 3rd     ↩
  display digit
rjmp display                     ;calls display for 7 segment display                ↩


fourth_digit:
mov r23, r18                     ;places hex digits into register reprsenting 4th    ↩
  display digit
rjmp display                   ;calls display for 7 segment display
```