```
AVRASM ver. 2.2.6  C:\Users\edwar\Desktop\380\Lab7\pushbutton_interrupt
  \pushbutton_interrupt\main.asm Thu Nov 03 15:07:54 2016

C:\Users\edwar\Desktop\380\Lab7\pushbutton_interrupt\pushbutton_interrupt\main.asm
  (28): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel\ATmega_DFP
  \1.1.130\avrasm\inc\m16def.inc'
C:\Users\edwar\Desktop\380\Lab7\pushbutton_interrupt\pushbutton_interrupt\main.asm
  (28): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel\ATmega_DFP
  \1.1.130\avrasm\inc\m16def.inc'


                                ;*
                                ;* Title:           Pushbutton Interrupt
                                ;* Author:          Edward Wang/ Yash Jain
                                ;* Version:         1.0
                                ;* Last updated:    11/02/16
                                ;* Target:          ATmega16 @ 1MHz
                                ;*
                                ;* DESCRIPTION
                                ;* Handles the interrupts from a pushbutton
                                /* when inputted from PD3 as well as
                                   grabs an input from the dip switch on
                                   pins PA0 - PA3. Performs the necessary
                                   table lookup and grabs the appropriate
                                   hex to 7 segment display value. This value
                                   is then push to display 0 while its
                                   original value is pushed onto display 1.
                                   When there is no interrupt, main_loop
                                   loops themux_display method to display
                                   alternatively display 0 and display1 with
                                   a delay.
                                */
                                ;*
                                ;* VERSION HISTORY
                                ;* 1.0 Original version
                                ;*******************************************
                                .list

                                .equ QCLR   = 0
                                ;flip-flop asynchronous clear input connected to PD0

                                .def dig0_seg = r0
                                .def dig1_seg = r1
                                .def digit_ON = r2

                                .cseg

                                reset:
                                   .org RESET
000000 c004                            rjmp start ;program starts
                                   .org INT1addr  ;INT0 interrupt vector
```

```
000004 c01d                             rjmp pushbutton_isr
                                start:
                                    ;Configure port B as an output port
000005 ef0f                         ldi r16, $FF        ;load r16 with all 1s
000006 bb07                         out DDRB, r16
                                   ;port B - all bits configured as outputs
000007 e000                        ldi r16, $00    ;initial count is 0
000008 bb08                         out PORTB, r16

                                    ;Configure port C as an output port
000009 9aa7                         sbi DDRC, 7    ;PINC7 configured as output
00000a 9aa6                         sbi DDRC, 6    ;PINC6 configured as output

                                   //Configure PORTD
00000b e001                        ldi r16, $01
00000c bb01                        out DDRD, r16
                                   //reset DFF
00000d 9890                        cbi PORTD, 0
00000e 9a90                        sbi PORTD, 0

                                   //Configure PORTA
00000f e400                        ldi r16, $40
000010 bb0a                        out DDRA, r16
000011 ef0f                        ldi r16, $FF
000012 bb0b                        out PORTA, r16

                                    ;Inital delay value
000013 e041                         ldi r20, 1        ; 32

                                   ;Initial value of digit_ON is 1
000014 e001                        ldi r16, 1
000015 0e20                        add digit_ON, r16


                                    ;Initialize stack pointer to allow
                                   ;subroutine calls
000016 e50f                         ldi r16, LOW(RAMEND)    ;load low
                                   ;byte of stack pointer
000017 bf0d                         out SPL, r16
000018 e004                         ldi r16, HIGH(RAMEND)   ;load high
                                   ;byte of stack pointer
000019 bf0e                         out SPH, r16

                                   /*
                                   ISC00/01 - manage trigger events
                                              for INT0
                                   ISC10/11 - manage trigger events
                                              for INT1
                                   00 - low level
                                   01 - any logic
                                   10 - falling edge
                                   11 - rising edge
```

```
                                   */
00001a e00c                        ldi r16, (1 << ISC11) | (1 << ISC10)
                                   //MCUCR: MCU Control Reg
                                   //contains config for ISCXX
00001b bf05                         out MCUCR, r16
                                   ;rising edge at INT1 requests interrupt
00001c e800                         ldi r16, 1<<INT1
                                   //GICR: General Interrupt Control Reg
                                   //contains interrupt enable bits
                                   //for INT0/1/2
00001d bf0b                         out GICR, r16

                                   //set global interrupt enable
00001e 9478                         sei

                                 main_loop:
00001f 940e 0045                   call mux_display
000021 cffd                        rjmp main_loop


                                 ;****************************************
                                 ;*
                                 ;* "pushbutton_isr" - Push button interrupt
                                 ;*
                                 ;* Description:at pushbutton interrupt,
                                 ;* get input from DIP switch, perform
                                 ;* table lookup from hextable and updates
                                 ;* display values.
                                 ;*
                                 ;* Author:                  Edward Wang/
                                 ;*                         Yash Jain
                                 ;* Version:                    1.0
                                 ;* Last updated:            0.1
                                 ;* Target:                  ATmega16@1MHz
                                 ;* Number of words:            12
                                 ;* Number of cycles:        21
                                 ;* Low registers modified:  ro, r1
                                 ;* High registers modified: none
                                 ;*
                                 ;*
                                 ;*
                                 ;****************************************
                                     ;INT0 interrupt service routine
                                 pushbutton_isr:
000022 930f                          push r16            ;save r16
000023 b70f                          in r16, SREG        ;save SREG
000024 930f                          push r16

                                   //call get_input subroutine
000025 b309                        in r16, PINA
                                   ;input switch values
000026 700f                         andi r16, $0F
```

```
                                ;force ms nibble to 0
000027 940e 0037                call hex_2_7seg

                                ;generate a negative pulse on pin 0
                                ;of port D
000029 9890                      cbi PORTD, QCLR
                                ;to clear the flip-flop
00002a 9a90                      sbi PORTD, QCLR

                                ;pop items in stack
00002b 910f                     pop r16
00002c bf0f                     out SREG, r16
00002d 910f                     pop r16
00002e 9518                     reti


                               ;Table of segment values to display digits 0 - F
00002f 4f01
000030 0612
000031 244c
000032 0f60
000033 0c00
000034 0008
000035 0131
000036 3830                    hextable: .db $01, $4F, $12, $06, $4C, $24, $60, $0F,↵
   $00, $0C, $08, $00, $31, $01, $30, $38

                               ;************************************
                               ;************************************
                               ;*
                               ;* "hex_2_7seg" -
                               ;Hexadecimal to Seven-Segment Table Lookup
                               ;*
                               ;* Description:
                               ;* Uses table lookup to convert a
                               ;hexadecimal value passed in r16 to the
                               ;* seven-segment pattern required to
                               ;display the hexadecimal value on a
                               ;* common anode display. The seven-segment
                               ;pattern is returned in r16.
                               ;*
                               ;* Author: Edward Wang, Yash Jain
                               ;* Version: 1.0
                               ;* Last updated: 10/26/16
                               ;* Target: ATmega16 @1Mhz
                               ;* Number of words: 18
                               ;* Number of cycles: 18
                               ;* Low registers modified: none
                               ;* High registers modified: r16
                               ;*
                               ;* Parameters:
                               ;* r16 - right justified hexadecimal value
```

```
                            ;to convert
                            ;*
                            ;* Returns:
                            ;* r16 - seven-segment pattern
                            ;*
                            ;* Notes:
                            ;* Values in the table are for a common
                            ;anode display. Complement these
                            ;* values when using a common cathode display
                            ;*
                            ;*************************************
                            ;*********************************
                            hex_2_7seg:
                               ;push r25 to stack(SREG)
000037 932f                    push r18
                               ;push r18
000038 e0f0                     ldi ZH, high (hextable * 2)
                               ;set Z to point to start of table
000039 e5ee                     ldi ZL, low (hextable * 2)
00003a e020                     ldi r18, $00
                               ;add offset to Z pointer
00003b 0fe0                     add ZL, r16
00003c 1ff2                     adc ZH, r18
00003d 9104                     lpm r16, Z
                               ;load byte from table pointed to by Z
00003e 940e 0042               call update_display_image
000040 912f                    pop r18
000041 9508                     ret
                               ;jump back to read switches again


                            ;*********************************************
                            ;***************************
                            ;*
                            ;* "update_display_image" -
                            ;Update Display Image
                            ;*
                            ;* Description:
                            ;* Copies the image of the segment pattern
                            ;for dig0 stored in r0 (dig0_seg)
                            ;* to the image of the segment pattern for
                            ;dig1 stored in r1 (dig1_seg).
                            ;* Then copies the segment pattern in
                            ;r16 to r0. This effectively, shifts
                            ;* what is displayed one digit to the left
                            ;when the multiplexed display is
                            ;* updated.
                            ;*
                            ;* Author: Yash Jain/ Edward Wang
                            ;* Version: 1.0
                            ;* Last updated: 10/26/16
                            ;* Target: ATmega16 @1Mhz
```

```
                            ;* Number of words: 2
                            ;* Number of cycles: 2
                            ;* Low registers modified: r0, r1 - new
                            ;segment patterns for dig0 and dig1
                            ;* High registers modified: none
                            ;*
                            ;* Parameters:
                            ;* r16 - new segment pattern for dig0
                            ;*
                            ;* Returns:
                            ;* r0, r1 - updated with new segment patterns
                            ;for dig0 and dig1
                            ;*
                            ;* Notes:
                            ;* Uses def and undef directives to provide
                            ;aliases for r0 and r1
                            ;* .def dig0_seg = r0
                            ;image of segment pattern for digit 0
                            ;* .def dig1_seg = r1
                            ;image of segment pattern for digit 1
                            ;***************************************
                            ;*********************************
                            update_display_image:
000042 2c10                    mov dig1_seg, dig0_seg
                               ;move dig0 to dig1
000043 2e00                    mov dig0_seg, r16
                                   ;call mux_display
000044 9508                     ret
                               ;jump back to main_loop


                            ;*********************************
                            ;***************************************
                            ;*
                            ;* "mux_display" -
                            ;Multiplexes Two-Digit Common Anode
                            ;Seven-Segment Display
                            ;*
                            ;* Description:
                            ;* Each time this subroutine is called,
                            ;it turns OFF the previous
                            ;* digit and turns ON the next digit of
                            ;a two-digit seven segment display.
                            ;* The segment values to be displayed
                            ;are taken from registers r1 and r0
                            ;* for digits dig1 and dig0, respectively
                            ;. The subroutine maintains a digit
                            ;* counter (r2) indicating which digit is
                            ;currently being displayed.
                            ;*
                            ;* To keep each digit ON for a longer time
                            ; requires a separate delay
```

```
                            ;* subroutine.
                            ;*
                            ;* Author: Yash Jain, Edward Wang
                            ;* Version: 0.0
                            ;* Last updated: 10/26/16
                            ;* Target: ATmega16 @1 Mhz
                            ;* Number of words:
                            ;* Number of cycles:
                            ;* Low registers modified: r2
                            ;* High registers modified: none
                            ;*
                            ;* Parameters: The segment values to be
                            ;displayed are passed in r0 - r2
                            ;* r0 - dig0_seg
                            ;* r1 - dig1_seg
                            ;*
                            ;* Returns:
                            ;* r2 - digit_ON, increments r2 to select
                            ;next digit turned ON
                            ;*
                            ;* Notes: 0s turn ON digits and 0s turn ON
                            ;segments
                            ;* The segments are a through g at PB6
                            ;through PB0 respectively.
                            ;* The digit driver pins are PC7 and PC6
                            ;for digits dig1 and dig0
                            ;*
                            ;* Uses def and undef directives to
                            ;provides aliases for
                            ;* r0, r1, and r2
                            ;* .def dig0_seg = r0;image of segment
                            ;pattern for digit 0
                            ;* .def dig1_seg = r1;image of segment
                            ;pattern for digit 1
                            ;* .def digit_ON = r2;lsb indicates
                            ;digit that is ON
                            ;*
                            ;****************************************
                            ;********************************
                            mux_display:
000045 930f                    push r16
000046 b70f                    in r16, SREG
000047 930f                    push r16
000048 932f                    push r18
000049 93af                    push r26
00004a 9423                    inc digit_ON
00004b 2d22                    mov r18, digit_ON
00004c 7021                    andi r18, $01 ;lsb is preserved
00004d fd20                    sbrc r18, 0   ;test if lsb=0
00004e 940e 0059               call display0
000050 ff20                    sbrs r18, 0   ;test if lsb=1
000051 940e 0060               call display1
```

```
000053 91af                         pop r26
000054 912f                         pop r18
000055 910f                         pop r16
000056 bf0f                         out SREG, r16
000057 910f                         pop r16
000058 9508                         ret
                                  display0:
000059 ba08                         out PORTB, dig0_seg
00005a 98af                         cbi PORTC, 7
00005b 9aae                         sbi PORTC, 6
                                    ;ldi r18, 5
00005c e604                         ldi r16, 100
00005d 940e 0067                    call var_delay
00005f 9508                         ret
                                  display1:
000060 ba18                         out PORTB, dig1_seg
000061 98ae                         cbi PORTC, 6
000062 9aaf                         sbi PORTC, 7
                                    ;ldi r18, 5
000063 e604                         ldi r16, 100
000064 940e 0067                    call var_delay
000066 9508                         ret

                                  ;***************************************************
                                  ;************************
                                  ;*
                                  ;* "var_delay" - Variable Delay - 0.1 ms increments
                                  ;*
                                  ;* Description:
                                  ;* Delays for a time equal to r16 * 0.1 ms when
                                  ;ATmega16 clocked at 1 MHz
                                  ;*
                                  ;* Author: “authors’ names”
                                  ;* Version: 1.0
                                  ;* Last updated: 10/09/16
                                  ;* Target: ATmega16 @ 1 MHz
                                  ;* Number of words:
                                  ;* Number of cycles: ~100 * r16
                                  ;* Low registers modified: none
                                  ;* High registers modified: none
                                  ;*
                                  ;* Parameters:
                                  ;* r16 - outer loop control variable
                                  ;*
                                  ;* Returns:
                                  ;* delay of 0.1ms * r16
                                  ;*
                                  ;* Notes:
                                  ;* Delay is designed for ATmega16 with 1 MHz
                                  ;clock
                                  ;*
                                  ;***************************************************
```

```
                            ;*************************

                            var_delay:     ; load outer loop count
                            outer_loop:
000067 e2a0                     ldi r26, 32  ;about 100 us
                            inner_loop:
000068 95aa                     dec r26
000069 f7f1                     brne inner_loop
00006a 950a                     dec r16
00006b f7d9                     brne outer_loop
00006c 9508                     ret


RESOURCE USE INFORMATION
------------------------

Notice:
The register and instruction counts are symbol table hit counts,
and hence implicitly used resources are not counted, eg, the
'lpm' instruction without operands implicitly uses r0 and z,
none of which are counted.

x,y,z are separate entities in the symbol table and are
counted separately from r26..r31 here.

.dseg memory usage only counts static data declared with .byte

"ATmega16" register use summary:
x  :   0 y  :   0 z  :   1 r0 :   3 r1 :   2 r2 :   3 r3 :   0 r4 :   0
r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   0
r13:   0 r14:   0 r15:   0 r16:  40 r17:   0 r18:  10 r19:   0 r20:   1
r21:   0 r22:   0 r23:   0 r24:   0 r25:   0 r26:   4 r27:   0 r28:   0
r29:   0 r30:   2 r31:   2
Registers used: 10 out of 35 (28.6%)

"ATmega16" instruction use summary:
.lds  :   0 .sts  :   0 adc   :   1 add   :   2 adiw  :   0 and   :   0
andi  :   2 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
brcc  :   0 brcs  :   0 break :   0 breq  :   0 brge  :   0 brhc  :   0
brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
brne  :   2 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
brvs  :   0 bset  :   0 bst   :   0 call  :   7 cbi   :   4 cbr   :   0
clc   :   0 clh   :   0 cli   :   0 cln   :   0 clr   :   0 cls   :   0
clt   :   0 clv   :   0 clz   :   0 com   :   0 cp    :   0 cpc   :   0
cpi   :   0 cpse  :   0 dec   :   2 eor   :   0 fmul  :   0 fmuls :   0
fmulsu:   0 icall :   0 ijmp  :   0 in    :   3 inc   :   1 jmp   :   0
ld    :   0 ldd   :   0 ldi   :  17 lds   :   0 lpm   :   2 lsl   :   0
lsr   :   0 mov   :   3 movw  :   0 mul   :   0 muls  :   0 mulsu :   0
neg   :   0 nop   :   0 or    :   0 ori   :   0 out   :  13 pop   :   7
push  :   7 rcall :   0 ret   :   6 reti  :   1 rjmp  :   3 rol   :   0
ror   :   0 sbc   :   0 sbci  :   0 sbi   :   6 sbic  :   0 sbis  :   0
sbiw  :   0 sbr   :   0 sbrc  :   1 sbrs  :   1 sec   :   0 seh   :   0
```

```
sei   :   1 sen   :   0 ser   :   0 ses   :   0 set   :   0 sev   :   0
sez   :   0 sleep :   0 spm   :   0 st    :   0 std   :   0 sts   :   0
sub   :   0 subi  :   0 swap  :   0 tst   :   0 wdr   :   0
Instructions used: 22 out of 113 (19.5%)

"ATmega16" memory use summary [bytes]:
Segment   Begin    End      Code   Data   Used    Size   Use%
----------------------------------------------------------------
[.cseg] 0x000000 0x0000da    196     16    212   16384   1.3%
[.dseg] 0x000060 0x000060      0      0      0    1024   0.0%
[.eseg] 0x000000 0x000000      0      0      0     512   0.0%

Assembly complete, 0 errors, 0 warnings
```