```
; Modified by: Sudhakar Ganti (Fall 2016), Tom Arjannikov (Fall 2019)
; button.asm
;
; This program that demonstrates reading the buttons
; This programs turns off the LED on pin 52 for
; a delay duration and turns it on again if
; "right" or "up" button is pressed. Otherwise
; LED is on continuously
;

        ;First initialize built-in Analog to Digital Converter
        ; initialize the Analog to Digital converter
        ldi r16, 0x87
        sts ADCSRA, r16
        ldi r16, 0x40
        sts ADMUX, r16

        ; initialize PORTB and PORTL for ouput
        ldi     r16, 0b10101010
        out DDRB, r16
        ldi     r16, 0b00001010
        sts DDRL, r16

main_loop:
        ldi r19, 0b00000010 ; turn on LED on pin 52
        out PORTB, r19

        call check_button   ; check to see if a button is pressed
        cpi  r24, 1         ; Register R24 is set to 1 if "right" or "up" pressed
        brne main_loop

        ldi r19, 0x00       ; turn off LED if "right" or "up" pressed
        out PORTB, r19
        call delay
        rjmp main_loop      ; Go back to main loop after a short delay


;
; The function below called check_button tests to see if the button
; UP or RIGHT has been pressed,
;
; on return, r24 is set to be: 0 if not pressed, 1 if pressed
;
; Uses registers:
;       r16
;       r17
;       r24
;
; This function could be made much better.  Notice that the a2d
; returns a 2 byte value (actually 10 bits).
;
; if you consider the word:
;        value = (ADCH << 8) +  ADCL
;
; then:
;       value > 0x3E8 - no button pressed
;
; otherwise:
;       value < 0x032 - right button pressed
;       value < 0x0C3 - up button pressed
;       value < 0x17C - down button pressed
;       value < 0x22B - left button pressed
;       value < 0x316 - select button pressed
;
; This function 'cheats' because ADCH is 0 when the right or up button is
; pressed, and non-zero otherwise. Hence this works only for these buttons.
; It needs to be modified to take care of all button presses.

;
; Below are the LCD keypad shield values for different buttons.
;
.equ RIGHT      = 0x032 ; the same for both LCD keypad board
;
; board v1.0
;.equ UP     = 0x0FA
;.equ DOWN   = 0x1C2
;.equ LEFT   = 0x28A
;.equ SELECT = 0x352
;
; If the following values don't work properly, uncomment the
; values under v1.0 and comment out the following set.
;
; board v1.1
.equ UP     = 0x0C3
.equ DOWN       = 0x17C
.equ LEFT       = 0x22B
.equ SELECT     = 0x316

;
; TODO: Modify the code below to check any button and set r24 accordingly
;

check_button:
        ; start a2d conversion
        lds     r16, ADCSRA       ; get the current value of SDRA
        ori r16, 0x40     ; set the ADSC bit to 1 to initiate conversion
        sts     ADCSRA, r16

        ; wait for A2D conversion to complete
wait:
        lds r16, ADCSRA
        andi r16, 0x40     ; see if conversion is over by checking ADSC bit
        brne wait          ; ADSC will be reset to 0 is finished

        ; read the value available as 10 bits in ADCH:ADCL
        lds r16, ADCL
        lds r17, ADCH

        clr r24
        cpi r17, 0
        brne skip
        ldi r24,1
skip:
        ret

;
; delay
;
; this function uses registers:
;
;       r20
;       r21
;       r22
;
delay:
;
; TODO: Write a delay loop.
;
        ret
```