Report

Laboratory 2

September 19, 2017

Author:
Caroline Nilsson *(cn222nd)*
Daniel Alm Grundström *(dg222dw)*
Term: HT 2017
Course: 1DT301 - Computer Technology I

Contents

1 Assignment 1

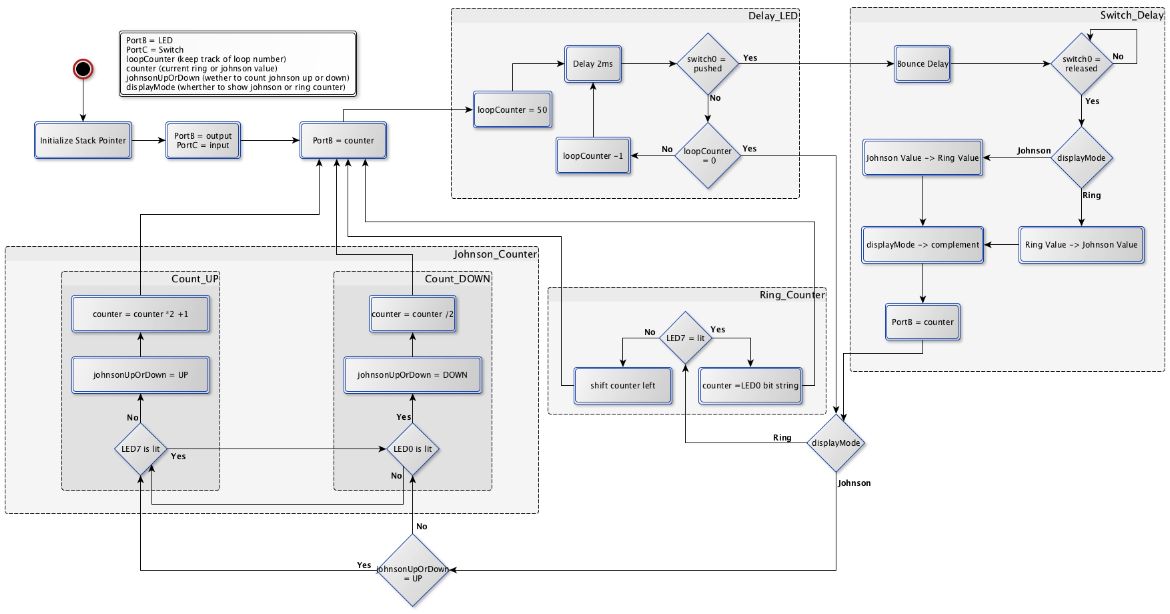

Figure 1: Switch between Johnson and Ring counter using switch0

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;    1DT301, Computer Technology I
;    Date: 2017-09-19
;    Author:
;                        Caroline Nilsson            (cn222nd)
;                        Daniel Alm Grundström       (dg222dw)
;
;    Lab number:         2
;    Title:              Subroutines
;
;    Hardware:           STK600, CPU ATmega2560
;
;    Function:           Counts up a counter and display it's value as either a
;                        Ring counter or Johnson counter. The display mode can
;                        be toggled between ring/johnson by pressing switch SW0.
;
;    Input ports:        PIN0 on PORTC
;
;    Output ports:       PIN2 on PORTB
;
;    Subroutines:        led_out            - Outputs counter to LEDs
;                        delay_led          - Delay to make changes to LEDs
;                                             visible. Also continuously checks
;                                             if switch is pressed.
;                        ring_counter       - Counts up ring counter
;                        johnson_counter    - Counts johnson counter up/down
;                        check_switch       - Checks if switch gets pressed
;                        on_switch_pressed  - Handles what should happen when
;                                             switch gets pressed
;                        delay_short        - Short delay of 2 ms used between
;                                             switch checks
;                        delay_switch       - Delay of 10 ms used after switch is
;                                             pressed down
;
;    Included files:     m2560def.inc
;
;    Other information:  N/A
;
;    Changes in program:
;                        2017-09-14:
;                        Implements flowchart design.
;
;                        2017-09-19:
;                        Refactors code by breaking down large subroutines
;                        into smaller subroutines.
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"

.def displayMode = r16                 ;determines whether to output ring or johnson
.def counter = r17                     ;keeps track of output value
.def dataDir = r18                     ;use to set input and output on PORTs
.def loopCounter = r19                 ;counts number of loops in delay led
.def johnUpOrDown = r20                ;whether to count johnson value up or down
.def complement = r21                  ;temp, to output counters complement
.equ UP = 0x01                         ;constant: value of up
```

```
.equ DOWN = 0x00                          ;constant: value of down
.equ JOHNSON = 0x00                       ;constant: Johnson display mode
.equ RING = 0xFF                          ;constant: Ring display mode
.equ SWITCH = PINC0                       ;constant: PIN of switch to check

;Initialize stack pointer
ldi r18, HIGH(RAMEND)
out SPH, r18
ldi r18, LOW(RAMEND)
out SPL, r18

;set PORTB to output
ldi dataDir, 0xFF
out DDRB, dataDir

;set PORTC to input
ldi dataDir, 0x00
out DDRC, dataDir

;initialize starting state
ldi displayMode, JOHNSON
ldi counter, 0x01
ldi johnUpOrDown, UP

main_loop:
    cpi displayMode, JOHNSON              ;if displaymode = johnson
        breq johnson1                     ;then jump to johnson branch

    ring1:                                 ;else jump to ring
        rcall ring_counter
        rjmp main_loop

    johnson1:
        rcall johnson_counter

    rjmp main_loop

;Outputs complement of the current value of counter to LEDs
led_out:
    mov complement, counter
    com complement
    out PORTB, complement
    ret

;Delay with continuous switch checking
delay_led:
    ldi loopCounter, 50

    loop_led:
        rcall delay_short
        rcall check_switch

        cpi loopCounter, 0                ;if loopcounter = 0
        breq delay_led_end                ;then jump to end

        dec loopCounter
        rjmp loop_led

    delay_led_end:
        ret

;Creates the ring counter by writing the complement of counter
;to PORTB and then increments the ring counter
ring_counter:
    sbis PORTB, PINB7                     ;if the 7th led is lit
        ldi counter, 0x01                 ;then set counter to one

    sbic PORTB, PINB7                     ;else
        lsl counter
                                  ;shift counter to the left
    rcall led_out
    rcall delay_led

    ret

;Creates the johnson counter by writing the complement of counter
;to PORTB and then checks wheter to count up or down
johnson_counter:
    cpi johnUpOrDown, UP                  ;if count up is active
    breq count_up                         ;then jump to count up

    rjmp count_down                       ;else jump to count down

    ;checks whether to continue to count up and
    ;increments the johnson value
    count_up:
        sbis PORTB, PINB7                 ;if the 7th led is lit
            rjmp count_down               ;then jump to count down

        ldi johnUpOrDown, UP
        lsl counter                       ;shift to the left
        inc counter
                                          ;add one

        rjmp end

    ;checks whether to continue to count down and
    ;decrease the johnson value
    count_down:
        sbic PORTB, PINB0                 ;if the right most led is not lit
            rjmp count_up                 ;then jump to count up

        ldi johnUpOrDown, DOWN
        lsr counter
                                ;shift to the right

    end:
        rcall led_out
        rcall delay_led

        ret


;Checks if the switch is pressed and in that case calls on_switch_pressed
check_switch:
    sbic PINC, SWITCH                   ;if switch is not pressed
        rjmp check_switch_end           ;then jump to end of subroutine

    switch_pressed_down:
        rcall delay_switch

        ;wait until button is released
        loop_switch:
            sbis PINC, SWITCH           ;if switch to the right most is still pressed
            rjmp loop_switch            ;then jump to loop switch

        ;When the button has been released we consider the switch pressed
        rcall on_switch_pressed

    check_switch_end:
    ret


;Handles what should happen when the switch gets pressed
on_switch_pressed:
    cpi displayMode, JOHNSON            ;if displaymode = johnson
    breq johnson_to_ring                ;then jump to johnson to ring

    cpi displayMode, RING               ;if displaymode = ring
    breq ring_to_johnson                ;then jump to ring to johnson

    ;convert ring value to johnson value
    ring_to_johnson:

        lsl counter
        dec counter

        rjmp switch_end

    ;convert johnson value to ring value
    johnson_to_ring:
        lsr counter
        inc counter

    switch_end:
        com displayMode
        ;TODO: needed? rcall led_out
        rcall led_out
                                        ;toogle displaymode between ring and johnson
    ret

;Delay for 2 ms
delay_short:
    ldi  r31, 13
    ldi  r30, 252
    L1:
        dec  r30
    brne L1
    dec  r31
    brne L1
    nop

    ret

;Delay 10 ms to avoid bouncing when switch is pressed
delay_switch:
    ldi  r31, 13
    ldi  r30, 252
L2: dec  r30
    brne L2
    dec  r31
    brne L2
    nop

    ret
```

2 Assignment 2

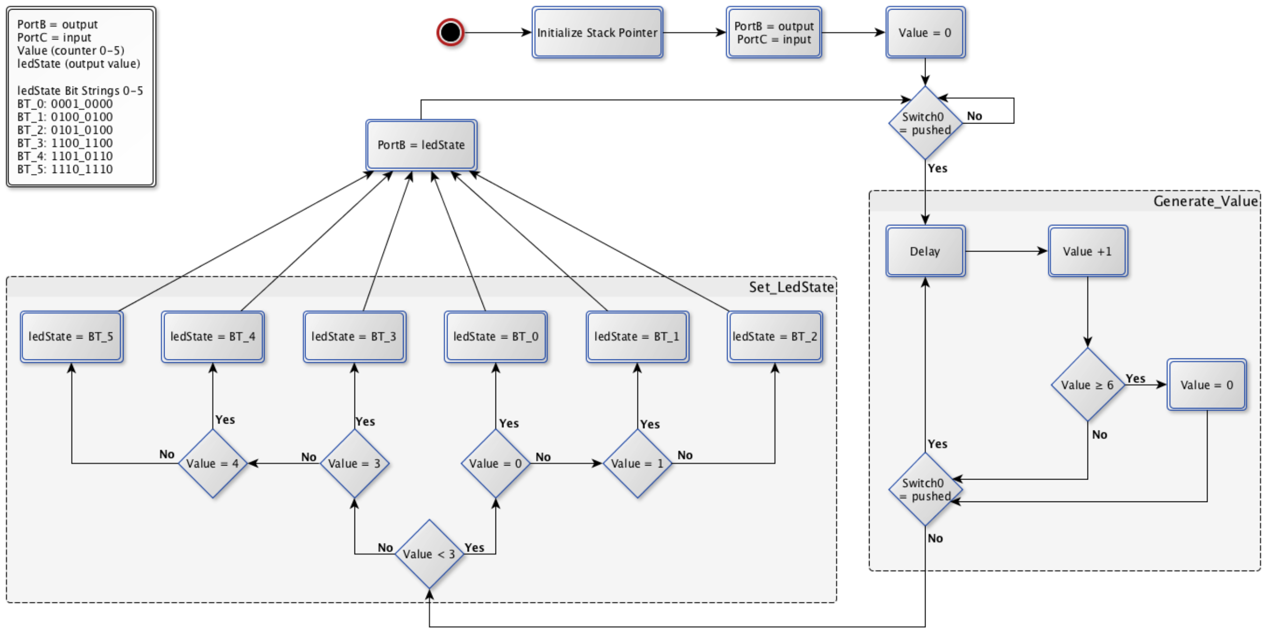

Figure 2: Simulating electronic dice

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;    1DT301, Computer Technology I
;    Date: 2017-09-19
;    Author:
;                          Caroline Nilsson            (cn222nd)
;                          Daniel Alm Grundström       (dg222dw)
;
;    Lab number:           2
;    Title:                Subroutines
;
;    Hardware:             STK600, CPU ATmega2560
;
;    Function:             Generates a random value between 1 and 6 when the user
;                          presses down switch SW0 and, when the user releases
;                          the switch, outputs a representation of a dice value
;                          to the LEDs
;
;    Input ports:          PIN0 on PORTC
;
;    Output ports:         PORTB
;
;    Subroutines:          generate_value    - Generate a pseudorandom value
;                                              between 1 and 6
;                          set_led_state     - Set value to output to LEDs
;                          delay_switch      - Delay of 10 ms used after switch is
;                                              pressed down
;
;    Included files:       m2560def.inc
;
;    Other information:    N/A
;
;    Changes in program:
;                          2017-09-14:
;                          Implements flowchart design.
;
;                          2017-09-19:
;                          Adds constants for LED dice states and comments.
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"

.def dataDir = r16
.def randomValue = r17
.def ledState = r18
.def complement = r19

.equ LED_DICE_1 = 0b0001_0000
.equ LED_DICE_2 = 0b0100_0100
.equ LED_DICE_3 = 0b0101_0100
.equ LED_DICE_4 = 0b1100_1100
.equ LED_DICE_5 = 0b1101_0110
.equ LED_DICE_6 = 0b1110_1110

ldi r16, HIGH(RAMEND)
```

```
out SPH, r16
ldi r16, LOW(RAMEND)
out SPL, r16

ldi dataDir, 0xFF
out DDRB, dataDir

ldi dataDir, 0x00
out DDRC, dataDir

ldi complement, 0xFF
out PORTB, complement

loop:
    sbic PINC, PINC0                    ;Wait until switch is pressed down
        rjmp loop

    rcall generate_value
    rcall set_led_state
    mov complement, ledState
    com complement
    out PORTB, complement

    rjmp loop

;Generate a pseudorandom value by repeatedly incrementing a counter for as long
;as the switch is pressed down
generate_value:
    ldi ledState, 0xFF                  ;Reset LEDs
    out PORTB, ledState
    start:

        rcall delay_switch              ;Delay to avoid bouncing effects

        inc randomValue
        cpi randomValue, 6
        brge reset_value
        rjmp end

    reset_value:
        ldi randomValue, 0

    end:
        sbis PINC, PINC0                ;If switch is still pressed down
            rjmp start                  ;   then jump to start

        ret

;Set LED output value to bit pattern representing different dice values
;depending of value of the pseudorandomly generated value
set_led_state:
    cpi randomValue, 3
    brlo less
    rjmp more

    less:
        cpi randomValue, 0
        breq one

        cpi randomValue, 1
        breq two

        rjmp three

        one:
            ldi ledState, LED_DICE_1
            rjmp end_led_state

        two:
            ldi ledState, LED_DICE_2
            rjmp end_led_state

        three:
            ldi ledState, LED_DICE_3
            rjmp end_led_state

    more:
        cpi randomValue, 3
        breq four

        cpi randomValue, 4
        breq five

        rjmp six

        four:
            ldi ledState, LED_DICE_4
            rjmp end_led_state

        five:
            ldi ledState, LED_DICE_5
            rjmp end_led_state

        six:
            ldi ledState, LED_DICE_6
            rjmp end_led_state

    end_led_state:
        ret

;Delay 10 ms to avoid bouncing effects when a switch is first pressed down
```

```
delay_switch:
    ldi   r31, 13
    ldi   r30, 252
    L1:
        dec   r30
    brne L1
    dec   r31
    brne L1
    nop

    ret
```

3 Assignment 3

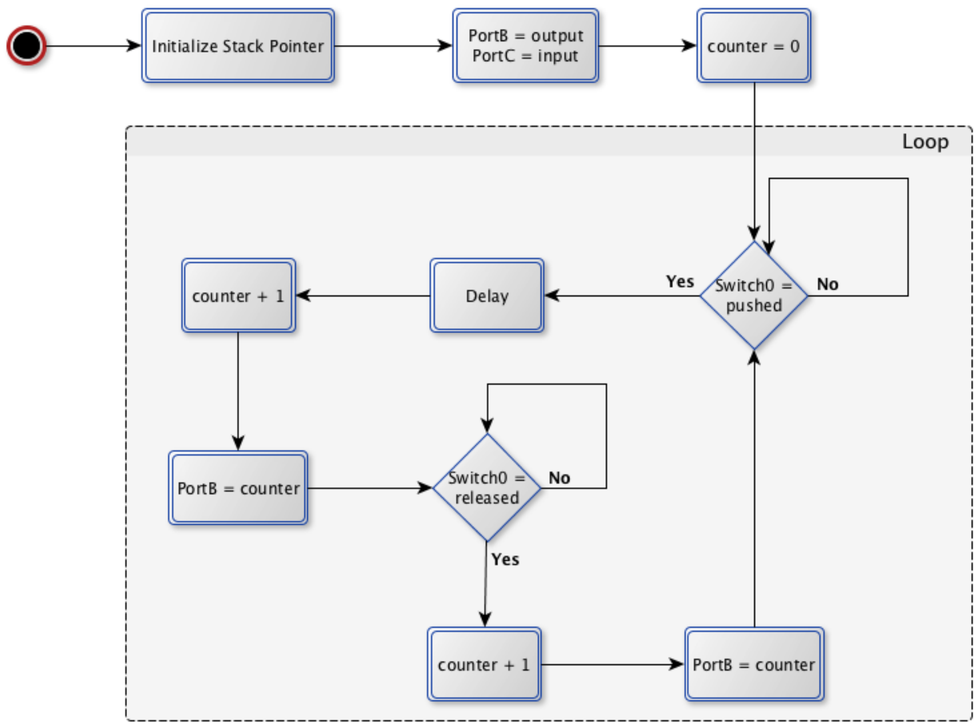

Figure 3: Change counter on switch0

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;    1DT301, Computer Technology I
;    Date: 2017-09-19
;    Author:
;                        Caroline Nilsson           (cn222nd)
;                        Daniel Alm Grundström      (dg222dw)
;
;    Lab number:         2
;    Title:              Subroutines
;
;    Hardware:           STK600, CPU ATmega2560
;
;    Function:           Counts number of times switch SW0 changes values, i.e.
;                        how many times the switch goes from 0 to 1 and 1 to 0.
;
;                        The counter is outputted to the LEDs in binary form
;                        each time the counter gets incremented.
;
;    Input ports:        PIN0 on PORTC
;
;    Output ports:       PORTB
;
;    Subroutines:        wait_for_switch_press    - Delays execution of the
;                                                   Program until SW0 is press
;                        on_switch_down           -
;                        wait_for_switch_release  -
;                        on_switch_up             -
;                        led_out                  -
;                        delay_switch             -
;
;    Included files:     m2560def.inc
;
;    Other information:  N/A
;
;    Changes in program:
;                        2017-09-14:
;                        Implements flowchart design.
;
;                        2017-09-19:
;                        Refactors the code by breaking smaller subroutines into
;                        multiple smaller ones. Adds comments
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"

.def dataDir = r16
.def counter = r17
.def complement = r18
```

```

ldi r16, HIGH(RAMEND)
out SPH, r16
ldi r16, LOW(RAMEND)
out SPL, r16

ldi dataDir, 0xFF
out DDRB, dataDir

ldi dataDir, 0x00
out DDRC, dataDir

ldi counter, 0x00
rcall led_out

main_loop:
    rcall wait_for_switch_press
    rcall on_switch_down

    rcall wait_for_switch_release
    rcall on_switch_up

    rjmp main_loop

;Pauses execution of program until SW0 is pressed down
wait_for_switch_press:
    loop:
        sbic PINC, PINC0                ;If SW0 is not pressed down
            rjmp loop                   ;    then continue waiting
    ret                                 ;return when SW0 gets pressed down

;Handles what should happen when SW0 gets pressed down
on_switch_down:
    rcall delay_switch                  ;Delay to avoid bouncing effects
    inc counter
    rcall led_out                       ;Output new counter value
    ret

;Pauses execution of program until SW0 is released
wait_for_switch_release:
    loop_2:
        sbis PINC, PINC0                ;If SW0 is still pressed down
            rjmp loop_2                 ;    then continue waiting
    ret                                 ;return when SW0 gets released

;Handles what should happen when SW0 gets released
on_switch_up:
    inc counter
    rcall led_out                       ;Output new counter value
    ret

;Outputs a binary representation of the current counter value to the LEDs
led_out:
    mov complement, counter
    com complement
    out PORTB, complement

;Delay of 10 ms. Used to avoid effects of switch bouncing
delay_switch:
    ldi  r31, 13
    ldi  r30, 252
    L1:
        dec  r30
    brne L1
    dec  r31
    brne L1
    nop

    ret
```

4 Assignment 4

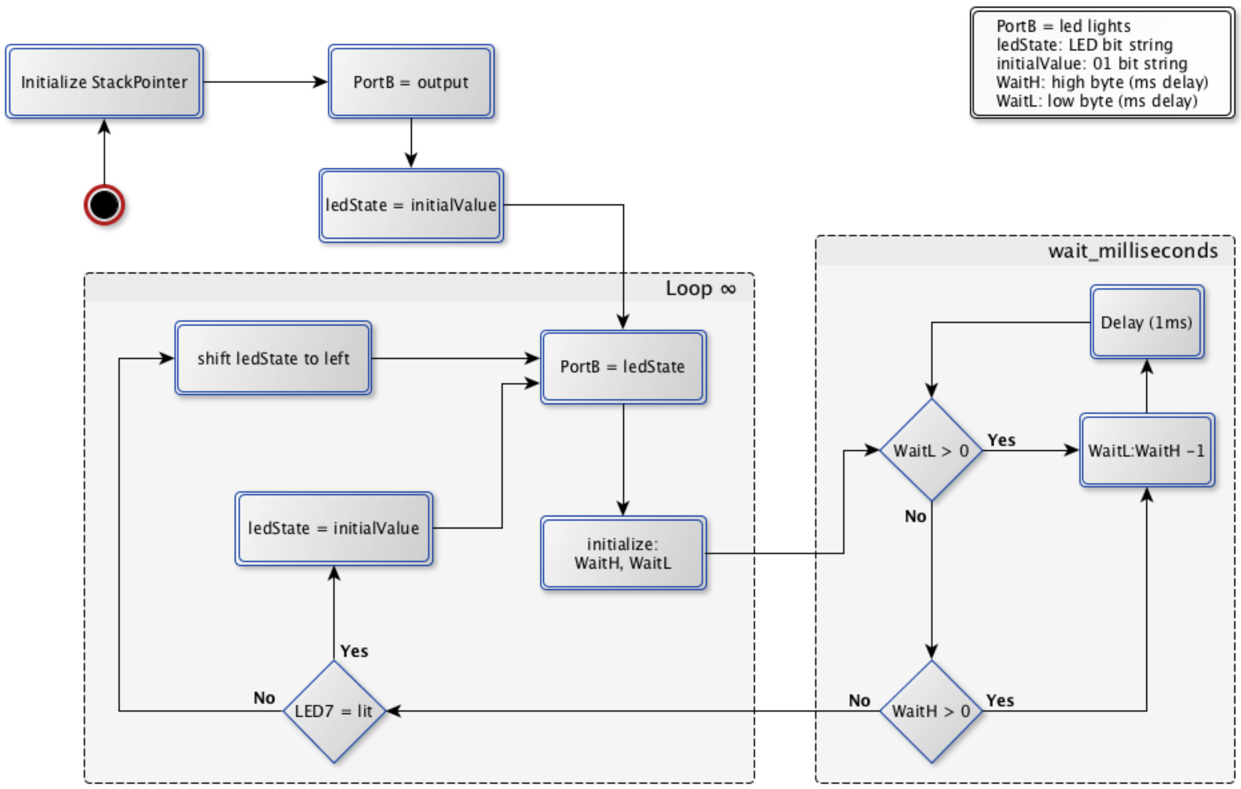

Figure 4: Ring counter with "wait-milliseconds" delay

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;    1DT301, Computer Technology I
;    Date: 2017-09-07
;    Author:
;                         Caroline Nilsson          (cn222nd)
;                         Daniel Alm Grundström     (dg222dw)
;
;    Lab number:          1
;    Title:               How to use the PORTs. Digital input /output.
;                         Subroutine call.
;
;    Hardware:            STK600, CPU ATmega2560
;
;    Function:            Repeatedly lights LEDs sequentially right to left.
;
;                         I.e:
;                         0000 0001 -> 0000 0010 -> 0000 0100 -> ... ->
;                         1000 0000 -> 0000 0001 -> 0000 0010 -> ...
;
;    Input ports:         N/A
;
;    Output ports:        PORTB
;
;    Subroutines:         delay - delays execution
;    Included files:      m2560def.inc
;
;    Other information:   Since a subroutine is used, the stack pointer must
;                         be initialized so the processor knows where in the
;                         code to jump when the subroutine returns.
;
;    Changes in program:
;                         2017-09-01:
;                         Implements flowchart design
;
;                         2017-09-04:
;                         Adds header, comments and some minor refactoring
;
;                         2017-09-07:
;                         Adjusts code to handle pull up resistor on PORTB.
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"
.def dataDir = r16
.def ledState = r17
.def complement = r18
```

```
.def waitH = r25
.def waitL = r24
.equ INITIAL_LED_STATE = 0x01


; Initialize SP, Stack Pointer
ldi r20, HIGH(RAMEND)                   ; R20 = high part of RAMEND address
out SPH,R20                             ; SPH = high part of RAMEND address
ldi R20, low(RAMEND)                    ; R20 = low part of RAMEND address
out SPL,R20                             ; SPL = low part of RAMEND address

; Set PORTB to output
ldi dataDir, 0xFF
out DDRB, dataDir

ldi ledState, INITIAL_LED_STATE         ; Set initial LED state

loop1:
    mov complement, ledState
    com complement
    out PORTB, complement                 ; Write state to LEDs

    ldi waitH, HIGH(1000)
    ldi waitL, LOW(1000)
    rcall wait_milliseconds             ; Delay to make changes visible

    sbis PORTB, PINB7
        ldi ledstate, INITIAL_LED_STATE
    sbic PORTB, PINB7
        lsl ledState                        ; Rotate LED state to the left
    rjmp loop1

wait_milliseconds:
    loop2:
        cpi waitL, 0x00
        breq low_zero
        rjmp wait

    low_zero:
        cpi waitH, 0x00
        breq high_zero
        rjmp wait

    high_zero:
        ret

    wait:
        sbiw waitH:waitL, 0x01

        ldi  r20, 2
        ldi  r19, 74
    L1: dec  r19
        brne L1
        dec  r20
        brne L1
        rjmp PC+1

        rjmp loop2
```