Report

Laboratory 1

September 19, 2017

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

Contents

1 Assignment 1

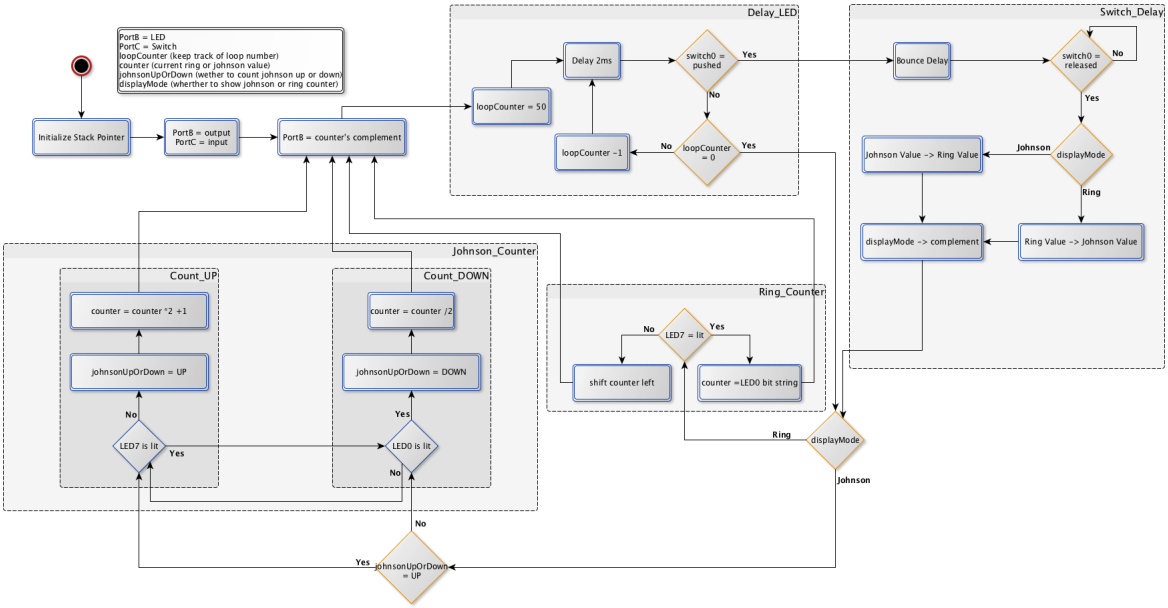

Figure 1: Switch between Johnson and Ring counter using switch0

```
1   .include "m2560def.inc"
2
3   .def displayMode = r16              ;determens wheter to output ring or johnson
4   .def counter = r17                  ;keeps track of output value
5   .def loopCounter = r19              ;counts number of loops in delay led
6   .def dataDir = r18                  ;use to set input and output on PORTs
7   .def johnUpOrDown = r22             ;whether to count johnson value up or down
8   .def complement = r23               ;temp, to output counters complement
9   .equ UP = 0x01                      ;constant: value of up
10  .equ DOWN = 0x00                    ;constant: value of down
11
12  ;TODO: define constants for johnson mode and ring mode
13
14  ;Initialize stack pointer
15  ldi r18, HIGH(RAMEND)
16  out SPH, r18
17  ldi r18, LOW(RAMEND)
18  out SPL, r18
19
20  ;set PORTB to output
21  ldi dataDir, 0xFF
22  out DDRB, dataDir
23
24  ;set PORTC to input
25  ldi dataDir, 0x00
26  out DDRC, dataDir
27
28  ;initialize starting state
29  ldi displayMode, 0x00
30  ldi counter, 0x01
31  ldi johnUpOrDown, UP
32
33  main_loop:
34
35      cpi displayMode, 0x00           ;if displaymode = johnson
36      breq johnson                    ;then jump to johnson branch
37
38      cpi displayMode, 0xFF           ;if displaymode = ring
39      breq ring                       ;then jump to ring
40
41      johnson:
42          rcall johnson_counter
43          rjmp main_loop
44
45      ring:
46          rcall ring_counter
47
48      rjmp main_loop
49
50  ;Creates the ring counter by writing the complement of counter
51  ;to PORTB and then increments the ring counter
52  ring_counter:
53      ; TODO: Move this to beginning of main_loop
54      mov complement, counter
55      com complement
56      out PORTB, complement
```

```
57      rcall delay_led
58
59      cpi displayMode, 0x00               ;if displaymode = johnson
60      breq ring_end                       ;then jump to end
61
62      sbis PORTB, PINB7                   ;if the 7th led is lit
63          ldi counter, 0x01               ;then set counter to one
64
65      sbic PORTB, PINB7                   ;else
66          lsl counter                     ;shift counter to the left
67
68      ring_end:
69      ret
70
71  ;Creates the johnson counter by writing the complement of counter
72  ;to PORTB and then checks wheter to count up or down
73  johnson_counter:
74      ; TODO: Move this to beginning of main_loop
75      mov complement, counter
76      com complement
77      out PORTB, complement
78      rcall delay_led
79
80      cpi displayMode, 0xFF               ;if displaymode = ring
81      breq end                            ;then jump to end
82
83      cpi johnUpOrDown, UP                ;if count up is active
84      breq count_up                       ;then jump to count up
85
86      rjmp count_down                     ;else jump to count down
87
88      ;checks whether to continue to count up and
89      ;increments the johnson value
90      count_up:
91          sbis PORTB, PINB7               ;if the 7th led is lit
92              rjmp count_down             ;then jump to count down
93
94          ldi johnUpOrDown, UP
95          lsl counter                     ;shift to the left
96          inc counter                     ;add one
97          rjmp end
98
99      ;checks whether to continue to count down and
100     ;decrese the johnson value
101     count_down:
102         sbic PORTB, PINB0               ;if the right most led is not lit
103             rjmp count_up               ;then jump to count up
104
105         ldi johnUpOrDown, DOWN
106         lsr counter                     ;shift to the right
107
108     end:
109         ret
110
111 ;Delay with continuous switch checking
112 delay_led:
113     ldi loopCounter, 50
114
115     loop_led:
116     ldi  r31, 13
117     ldi  r30, 252
118 L1: dec  r30
119     brne L1
120     dec  r31
121     brne L1
122     nop
123
124
125
126         sbis PINC, PINC0                ;if right most switch is pressed
127         rcall delay_switch              ;then jump to delay switch
128
129         cpi loopCounter, 0              ;if loopcounter = 0
130         breq delay_led_end              ;then jump to end
131
132         dec loopCounter                 ;subtract one
133
134         rjmp loop_led
135
136     delay_led_end:
137         ret
138
139 ;Delay to avoid bouncing when switch is pressed
140 delay_switch:
141     ldi  r20, 13
142     ldi  r21, 252
143 L2: dec  r21
144     brne L2
145     dec  r20
146     brne L2
147     nop
148
149     ;wait for button release
150     loop_switch:
151         sbis PINC, PINC0                ;if switch to the right most is still pressed
152         rjmp loop_switch                ;then jump to loop switch
153
154     cpi displayMode, 0x00               ;if displaymode = johnson
155     breq johnson_to_ring                ;then jump to johnson to ring
156
157     cpi displayMode, 0xFF               ;if displaymode = ring
```

```
158         breq ring_to_johnson                ;then jump to ring to johnson
159
160         ;convert ring value to johnson value
161         ring_to_johnson:
162             lsl counter
163             dec counter
164
165             rjmp switch_end
166
167         ;convert johnson value to ring value
168         johnson_to_ring:
169             lsr counter
170             inc counter
171
172         switch_end:
173             com displayMode                 ;toogle displaymode between ring and johnson
174             ret
```

2 Assignment 2

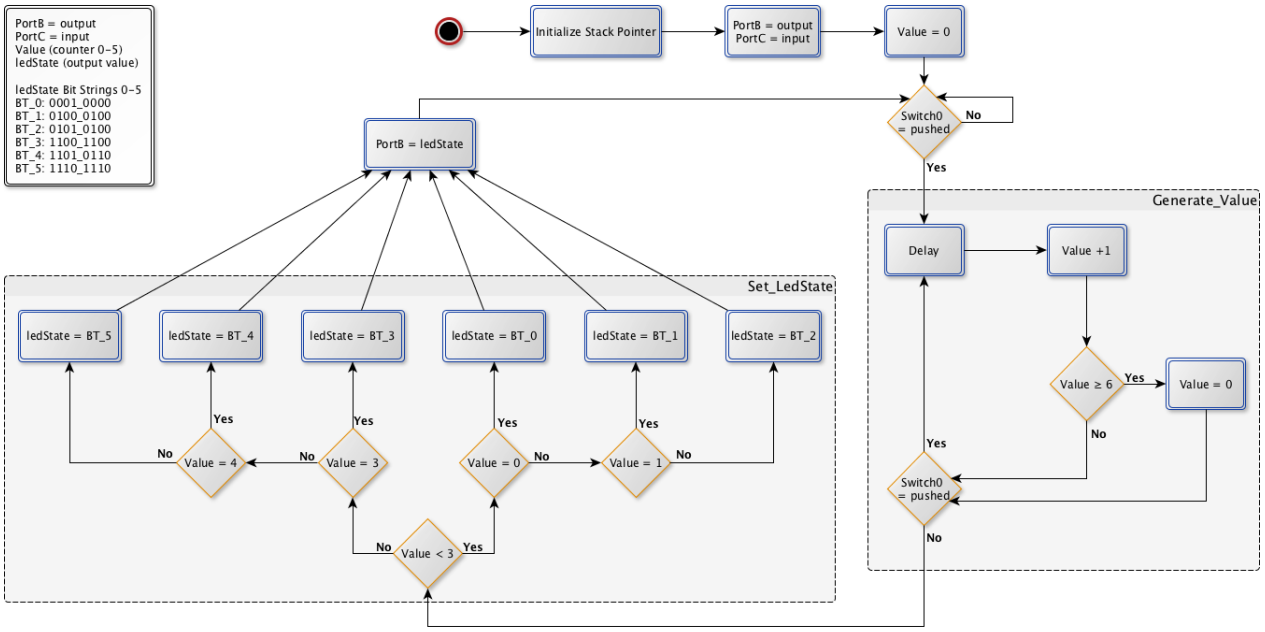

Figure 2: Simulating electronic dice

```
1   .include "m2560def.inc"
2
3   .def dataDir = r16
4   .def randomValue = r17
5   .def ledState = r18
6
7   ldi r16, HIGH(RAMEND)
8   out SPH, r16
9   ldi r16, LOW(RAMEND)
10  out SPL, r16
11
12  ldi dataDir, 0xFF
13  out DDRB, dataDir
14
15  ldi dataDir, 0x00
16  out DDRC, dataDir
17
18  loop:
19
20      sbis PINC, PINC0
21          rcall generate_value
22
23      rjmp loop
24
25  generate_value:
26
27      start:
28  ;       ldi  r20, 13
29  ;       ldi  r21, 252
30  ;L1:    dec  r21
31  ;       brne L1
32  ;       dec  r20
33  ;       brne L1
34  ;       nop
35
36      inc randomValue
37      cpi randomValue, 6
38      brge reset_value
39      rjmp end
40
41      reset_value:
42          ldi randomValue, 0
43
44      end:
45          sbis PINC, PINC0
46              rjmp start
47
48          rcall set_led_state
49          ret
50
51  set_led_state:
52      cpi randomValue, 3
53      brlo less
54      rjmp more
```

```
55
56          less:
57              cpi randomValue, 0
58              breq one
59              cpi randomValue, 1
60              breq two
61              rjmp three
62
63              one:
64                  ldi ledState, 0b0001_0000
65                  rjmp end_led_state
66              two:
67                  ldi ledState, 0b0100_0100
68                  rjmp end_led_state
69              three:
70                  ldi ledState, 0b0101_0100
71                  rjmp end_led_state
72          more:
73              cpi randomValue, 3
74              breq four
75              cpi randomValue, 4
76              breq five
77              rjmp six
78              four:
79                  ldi ledState, 0b1100_1100
80                  rjmp end_led_state
81              five:
82                  ldi ledState, 0b1101_0110
83                  rjmp end_led_state
84              six:
85                  ldi ledState, 0b1110_1110
86                  rjmp end_led_state
87          end_led_state:
88              out PORTB, ledState
89              ret
```

3 Assignment 3

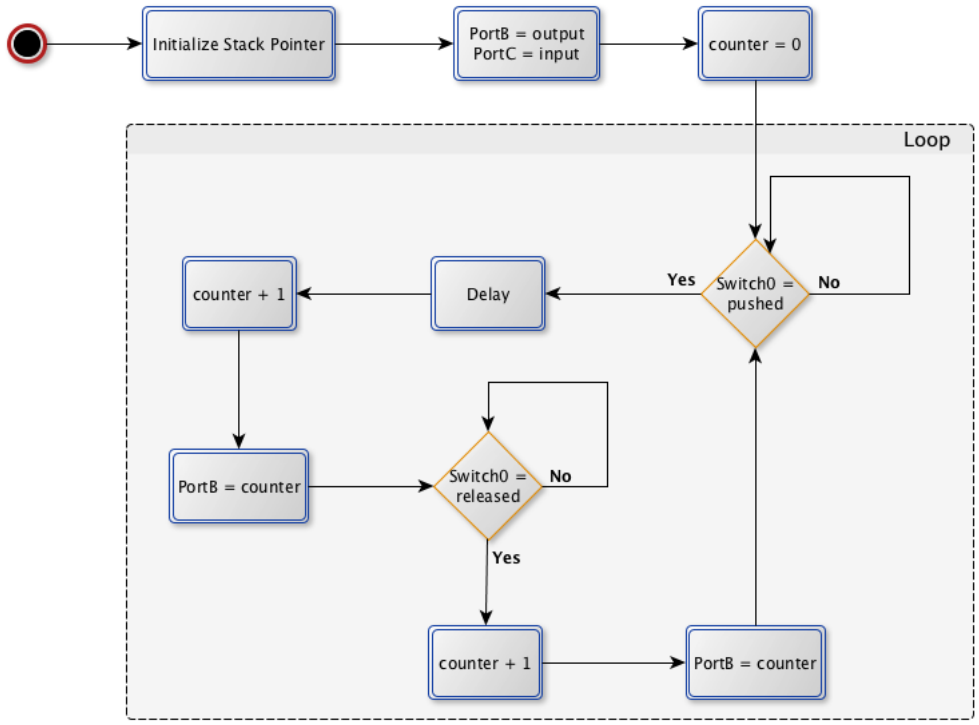

Figure 3: Change counter on switch0

```
1   .include "m2560def.inc"
2
3   .def dataDir = r16
4   .def counter = r17
5
6   ldi r16, HIGH(RAMEND)
7   out SPH, r16
8   ldi r16, LOW(RAMEND)
9   out SPL, r16
10
11  ldi dataDir, 0xFF
12  out DDRB, dataDir
13
14  ldi dataDir, 0x00
15  out DDRC, dataDir
16
17  ldi counter, 0x00
18
19  loop:
20      sbis PINC, PINC0
21          rcall switch_down
22      rjmp loop
23
24  switch_down:
25  ;       ldi  r20, 13
26  ;       ldi  r21, 252
27  ;L1:    dec  r21
28  ;       brne L1
29  ;       dec  r20
30  ;       brne L1
31  ;       nop
32
33      inc counter
34      out PORTB, counter
35
36      loop_2:
37          sbic PINC, PINC0
38              rjmp switch_released
39          rjmp loop_2
40
41      switch_released:
42          inc counter
43          out PORTB, counter
44          ret
```

4 Assignment 4