Report

Laboration 2

Subroutines.

Author: Amata ANANTAPRAYOON,
Adell TATROUS
Student ID: aa224iu,
at222ux
Semester: VT2018
Course: Computer Technology 1
Course code: 1DT301

Contents

1 Task 1: Switch between Johnson counter and ring counter

Write a program which switch between Ring counter and Johnson counter. You should not use Interrupt in this lab. The push button must be checked frequently, so there is no delay between the button is pressed and the change between Ring/Johnson. Use SW0 (PA0) for the button. Each time you press the button, the program should change counter.

1.1 flowchart

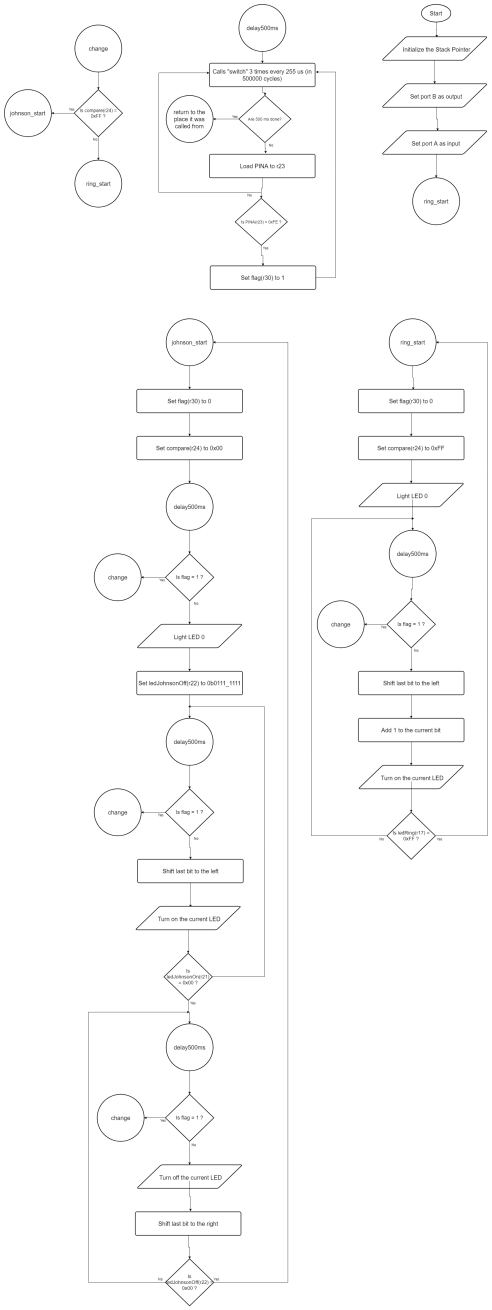

Figure 1: Flowchart for task 1

1.2 Assembly Program

,label=task1.asm

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2018-09-14
; Author:
; Amata Anantaprayoon (aa224iu)
; Adell Tatrous (at222ux)
;
; Lab number: 1
; Title: How to use the PORTs. Digital input/output. Subroutine call.
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Program which switch between Ring counter and Johnson counter when the user
;           press SW0
; Input ports: PORTA
;
; Output ports: PORTB.
;
; Subroutines:  delay500ms
;               switch
;               setFlag
;
; Included files: m2560def.inc
;
; Other information: We using SW1 instead of SW0
;
; Changes in program: 2018-09-20: Implementation
;                     2018-09-26: Comment the code + debug
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"
.def dataDir = r16
.def ledRing = r17
.def ledJohnsonOn = r21
.def ledJohnsonOff = r22
.def compare = r24
.def flag = r30


;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 port B as output
ldi dataDir, 0xFF
out DDRB, dataDir

;set port A as input
ldi dataDir, 0x00
out DDRA, dataDir
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

;RING counter
ring_start:
ldi flag, 0                     ; set flag to 0 (defualt)
ldi compare, 0xFF               ; set compare to 0xFF

;Lights LED0
ldi ledRing, 0b1111_1110
out PORTB, ledRing

ring_loop:

call delay500ms
cpi flag, 1                     ; check if flag is On (flag on = 1)
breq change                     ; if flag = 1 -> jump to change

lsl ledRing     ; shifts last bit to the left
inc ledRing     ; add 1 to the current bit
out PORTB, ledRing

cpi ledRing, 0xFF  ;compare ledOn with 0xFF
breq ring_start          ;IF ledOn = 0xFF jump to start

rjmp ring_loop

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

;Johnson counter

johnson_start:
ldi flag,0              ; set flag to 0 (defualt)
ldi compare , 0x00      ; set compare to 0x00

call delay500ms
cpi flag, 1             ; check if flag is On (flag on = 1)
breq change             ; if flag = 1 -> jump to change

;Lights LED0
ldi ledJohnsonOn, 0b1111_1110
out PORTB, ledJohnsonOn

ldi ledJohnsonOff, 0b0111_1111
```

```
96
97  forward:
98
99  call delay500ms
100 cpi flag, 1              ; check if flag is On (flag on = 1)
101 breq change              ; if flag = 1 -> jump to change
102
103 lsl ledJohnsonOn         ; shifts last bit to the left
104 out PORTB, ledJohnsonOn
105 cpi ledJohnsonOn, 0x00   ;compare ledOn with 0xFF
106 breq backward    ;IF ledOn = 0xFF jump to start
107
108 rjmp forward
109
110 backward:
111
112 call delay500ms
113 cpi flag, 1              ; check if flag is On (flag on = 1)
114 breq change              ; if flag = 1 -> jump to change
115
116
117 out PORTB, ledJohnsonOff
118 lsr ledJohnsonOff
119 cpi ledJohnsonOff, 0x00   ;compare ledOn with 0x00
120 breq johnson_start   ;IF ledOff = 0x00 jump to start
121
122 rjmp backward
123
124 ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
125
126 delay500ms:
127
128 ; Delay 500 000 cycles
129 ; 500ms at 1 MHz
130
131
132     ldi  r18, 3
133     ldi  r19, 138
134     ldi  r20, 86
135 L1:
136     dec  r20
137     brne L1
138     call switch
139     dec  r19
140     brne L1
141     call switch
142     dec  r18
143     brne L1
144     call switch
145     rjmp PC+1
146
147 ret
148
149 ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
150
151 switch:
152
153 in r23, PINA             ;load PINA to r23
154 cpi r23, 0xFE            ;check if r23 is equal to 0xFE
155 breq setFlag             ;if true -> call setFlag
156
157 ret
158 ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
159
160 setFlag:
161 ldi r30, 1               ;set flag on
162 ret
163
164 ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
165
166 change:
167 cpi compare, 0xFF        ;check compare with 0xFF
168 ;if ture means current state is on ring -> call Johnson
169 breq johnson_start
170 ;else call ring
171 call ring_start
```

2 Task 2: Electronic dice

Create an electronic dice.Number 1 to 6 should be generated randomly. You could use the fact that the time you press the button varies in length.

1 = 1110_1111, 2 = 0111_1101, 3 = 0110_1101, 4 = 0011_1001, 5 = 0010_1001, 6 = 0001_0001 (1 = LED off, 0 = LED On)

2.1 flowchart

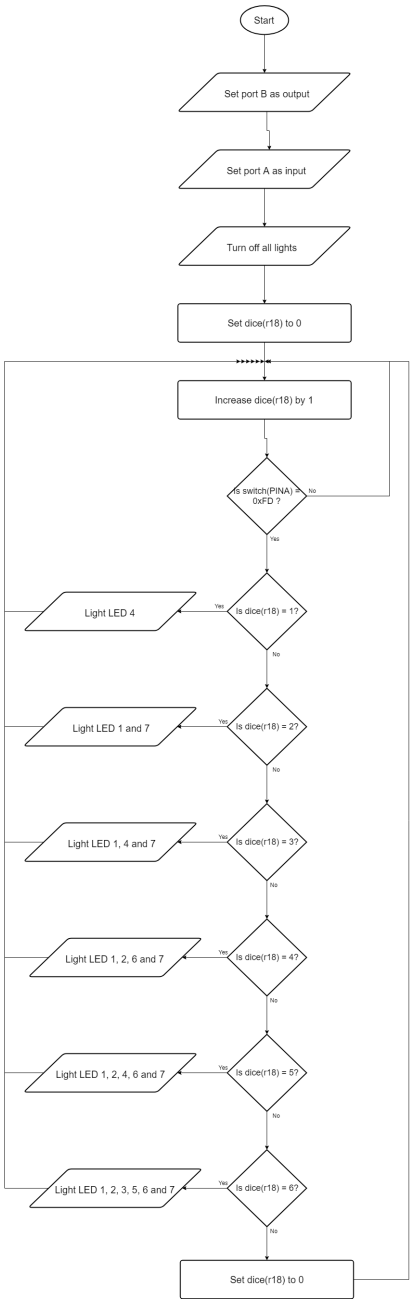

Figure 2: Flowchart for task 2

2.2 Assembly Program

,label=task2.asm

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2018-09-20
; Author:
; Amata Anantaprayoon (aa224iu)
; Adell Tatrous (at222ux)
;
; Lab number: 2
; Title: Subroutines
;
; Hardware: STK600, CPU ATmega2560
;
; Function: The program lights up "random" nummber of LED from 1-6 when
;           press SW1
;
; Input ports: On-board SW1 connected to PORTA
;
; Output ports: PORTB.
;
; Subroutines: N/A
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: 2018-09-20: Implementation
;                     2018-09-21: Edit + comment
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<



.include "m2560def.inc"
.def dataDir = r16
.def switch = r17
.def dice = r18
.def ledOn = r19

;set port B as output
ldi dataDir, 0xFF
out DDRB, dataDir

;set port A as input
ldi dataDir, 0x00
out DDRA, dataDir

;default setting
ldi ledOn, 0xFF
out PORTB, ledOn

;dice
ldi dice, 0

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

random:

inc dice

in switch, PINA             ;load PINA in switch
cpi switch, 0xFD            ;compare switch with 0xFD
breq stop                   ;if SW1 = press -> stop

rjmp random

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

;Reset dice to 0
reset:
ldi dice, 0
rjmp random

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

;Stop the "random"

stop:
cpi dice, 1
breq case_1

cpi dice, 2
breq case_2

cpi dice, 3
breq case_3

cpi dice, 4
breq case_4

cpi dice, 5
breq case_5

cpi dice, 6
breq case_6

rjmp reset
```

```
96
97  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
98
99  ;lights the correct LED
100
101 case_1:
102 ldi ledOn, 0b1110_1111
103 out PORTB, ledOn
104 rjmp random
105
106 case_2:
107 ldi ledOn, 0b0111_1101
108 out PORTB, ledOn
109 rjmp random
110
111 case_3:
112 ldi ledOn, 0b0110_1101
113 out PORTB, ledOn
114 rjmp random
115
116 case_4:
117 ldi ledOn, 0b0011_1001
118 out PORTB, ledOn
119 rjmp random
120
121 case_5:
122 ldi ledOn, 0b0010_1001
123 out PORTB, ledOn
124 rjmp random
125
126 case_6:
127 ldi ledOn, 0b0001_0001
128 out PORTB, ledOn
129 rjmp random
```

3 Task 3: Change counter

Write a program that is able to count the number of changes on a switch. As a change we count when the switch SW0 goes from 0 to 1 and from 1 to 0, we expect therefore positive and negative edges. We calculate the changes in a byte variable and display its value on PORTB.

3.1 flowchart

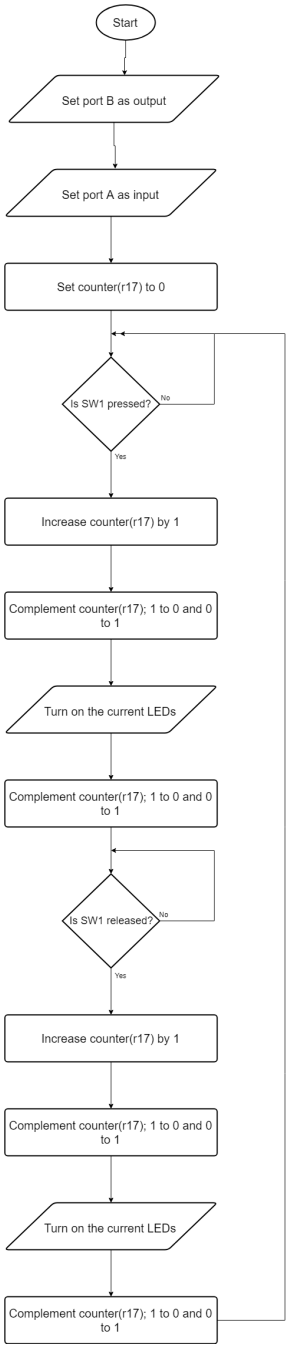

Figure 3: Flowchart for task 3

3.2 Assembly Program

,label=task3.asm

```
1   ;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
2   ; 1DT301, Computer Technology I
3   ; Date: 2018-09-21
4   ; Author:
5   ; Amata Anantaprayoon (aa224iu)
6   ; Adell Tatrous (at222ux)
7   ;
8   ; Lab number: 2
9   ; Title: Subroutines
10  ;
11  ; Hardware: STK600, CPU ATmega2560
12  ;
13  ; Function: LEDs should light up in binary form while incresing
14  ;           1 bit everytime we press and release SW1.
15  ;
16  ; Input ports: On-board SW1 connected to PORTA
17  ;
18  ; Output ports: PORTB.
19  ;
20  ; Subroutines: pressOn/Off
21  ;              increment
22  ;              waitPress/Release
23  ;              lightsOn
24  ;
25  ; Included files: m2560def.inc
26  ;
27  ; Other information: board is broken so we tried with SW1 instead of SW0
28  ;
29  ; Changes in program: 2018-09-21: Implementation
30  ;
31  ;
32  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
33  .include "m2560def.inc"
34  .def dataDir = r16
35  .def counter = r17
36  .def switchOn = r18
37  .def switchOff = r19
38  .def check = r20
39
40  ;set port B as output
41  ldi dataDir, 0xFF
42  out DDRB, dataDir
43
44  ;set port A as input
45  ldi dataDir, 0x00
46  out DDRA, dataDir
47
48  ldi counter, 0
49  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
50
51  main:
52      call waitPress
53      call PressOn
54
55      call waitRelease
56      call pressOff
57  rjmp main
58
59  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
60
61
62  pressOn:
63  rcall increment
64  rcall lightsOn
65
66  ret
67
68  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
69
70  pressOff:
71  rcall increment
72  rcall lightsOn
73
74  ret
75
76  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
77
78  increment:
79  inc counter                     ;increse counter by 1
80  ret
81
82  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
83
84  lightsOn:
85  com counter                     ;Ones Complement= 1->0, 0->1
86  out PORTB, counter              ;lights LEDs
87  com counter                     ;back to "normal" state
88  ret
89
90  ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
91
92  waitPress:
93  sbis PINA,PINA1                 ;skip next line when user dont press sw1
94  ret
95
```

```
96   rjmp waitPress
97
98   ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
99
100  waitRelease:
101  sbic PINA, PINA1                    ;skip next line when user releases sw1
102  ret
103
104  rjmp waitRelease
```

4 Task 4: Delay subroutine with variable delay time

Modify the program in task 5 in Lab 1 to a general delay routine that can be called from other programs. It should be named **wait_milliseconds**. The number of milliseconds should be transferred to register pair R24, R25.

4.1 flowchart

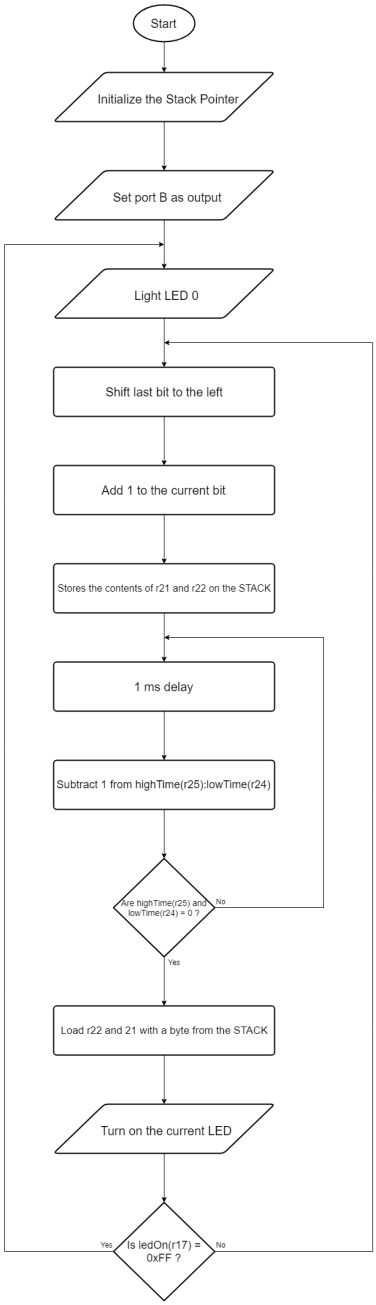

Figure 4: Flowchart for task 4

4.2 Assembly Program

,label=task4.asm

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2018-09-26
; Author:
; Amata Anantaprayoon (aa224iu)
; Adell Tatrous (at222ux)
;
; Lab number: 2
; Title: Subroutines
;
; Hardware: STK600, CPU ATmega2560
;
; Function: general delay routine that can be called from other
;           programs
;
; Input ports:N/A
;
; Output ports: PORTB
;
; Subroutines: wait_milliseconds
;
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: 2018-09-26: Implementation
;
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"
.def dataDir = r16
.def ledOn = r17
.def highTime = r25
.def lowTime = r24
.equ time = 5       ;<<<<<<<< edit the time here (in ms)

; 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 port B as output
ldi dataDir, 0xFF
out DDRB, dataDir

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

start:
;Lights LED0
ldi ledOn, 0b1111_1110
out PORTB, ledOn

loop:

ldi highTime, HIGH(time)
ldi lowTime, LOW(time)

lsl ledOn                            ; shifts last bit to the left
inc ledOn                            ; add 1 to the current bit
rcall wait_milliseconds
out PORTB, ledOn

cpi ledOn , 0xFF                     ;compare ledOn with 0xFF
breq start                           ;IF ledOn = 0xFF jump to start

rjmp loop

;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

wait_milliseconds:
;store data in r16 and r17 to the stack
push r21
push r22

;Delay for 1ms at 1 MHz (250us at 4 MHz)
L0:
ldi r21,2
ldi r22, 75

L1:
dec r22
brne L1
dec r21
brne L1
;repeat N times to gain N ms delay
sbiw highTime:lowTime ,1
brne L0

;return stored data from stack
pop r22
pop r21

ret
```