;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

; ED2022, Computer Technology I

; Lab 1, task 1

; Hardware: STK600, CPU ATmega2560

; Date: 2015-11-26

; Authors: Ivan Hussein

; Function:

; --------

; Light LED 2

; Used subroutines:

; Global subroutines (that can be used from other programs):

; Other information:

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

.INCLUDE "m2560def.inc"

.org 0x00

ldi r16, 0b00000010 ; sätter PIN2 som output

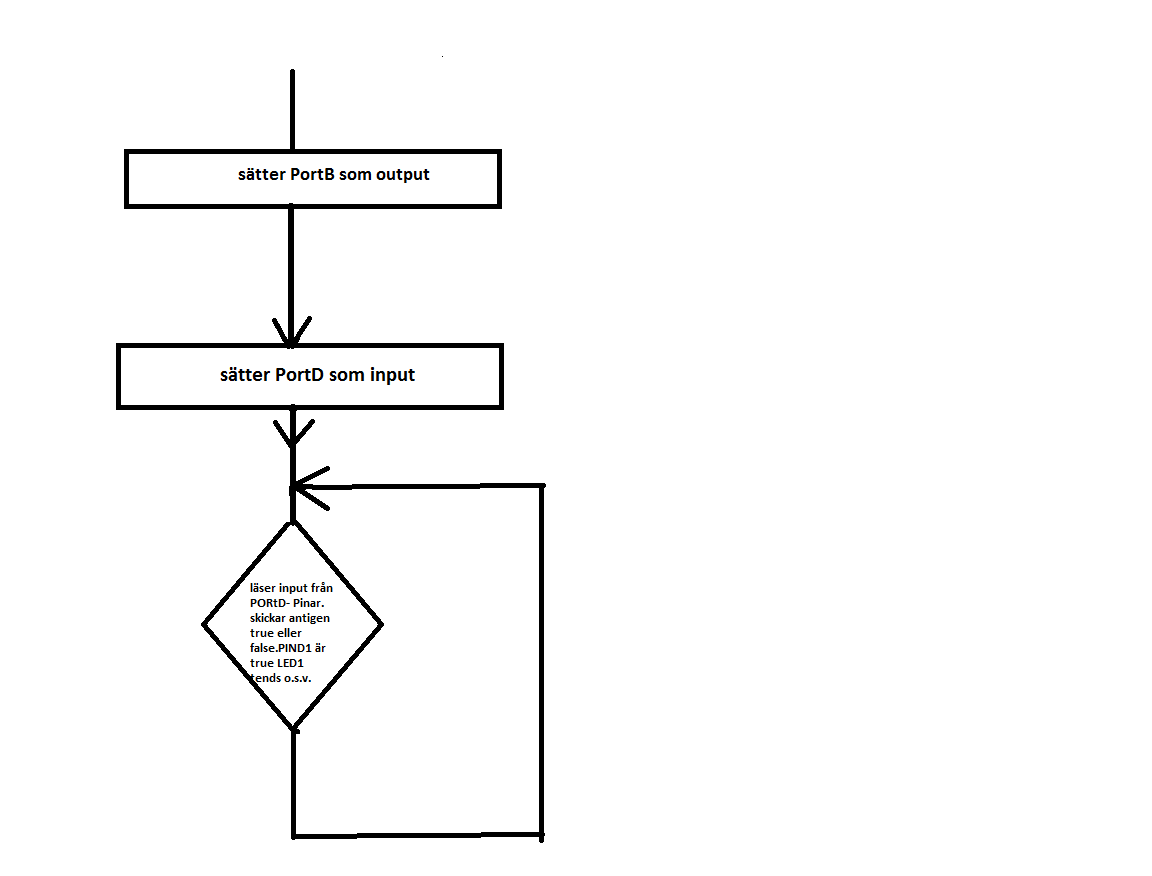
out DDRB, r16 ; PIN2 på PORTB är output andra PINAR är input

loop:

ldi r16,0b11111101 ; LED2 tends hela tiden i loopen.

out PORTB, r16

rjmp loop



;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

; ED2022, Computer Technology I

; Lab 1, task 2

; Hardware: STK600, CPU ATmega2560

; Date: 2015-11-26

; Authors: Ivan Hussein

; Function:

; --------

; Read the switches and light the corresponding LED.

; Example: When you press SW5, LED5 so should light.

; Used subroutines:

; Global subroutines (that can be used from other programs):

; Other information:

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

.INCLUDE "m2560def.inc"

.org 0x00

ldi r16, 0xff

out DDRB, r16 ; sätter PortB som output.

ldi r16, 0x00

out DDRD, r16 ; sätter PortD som input.

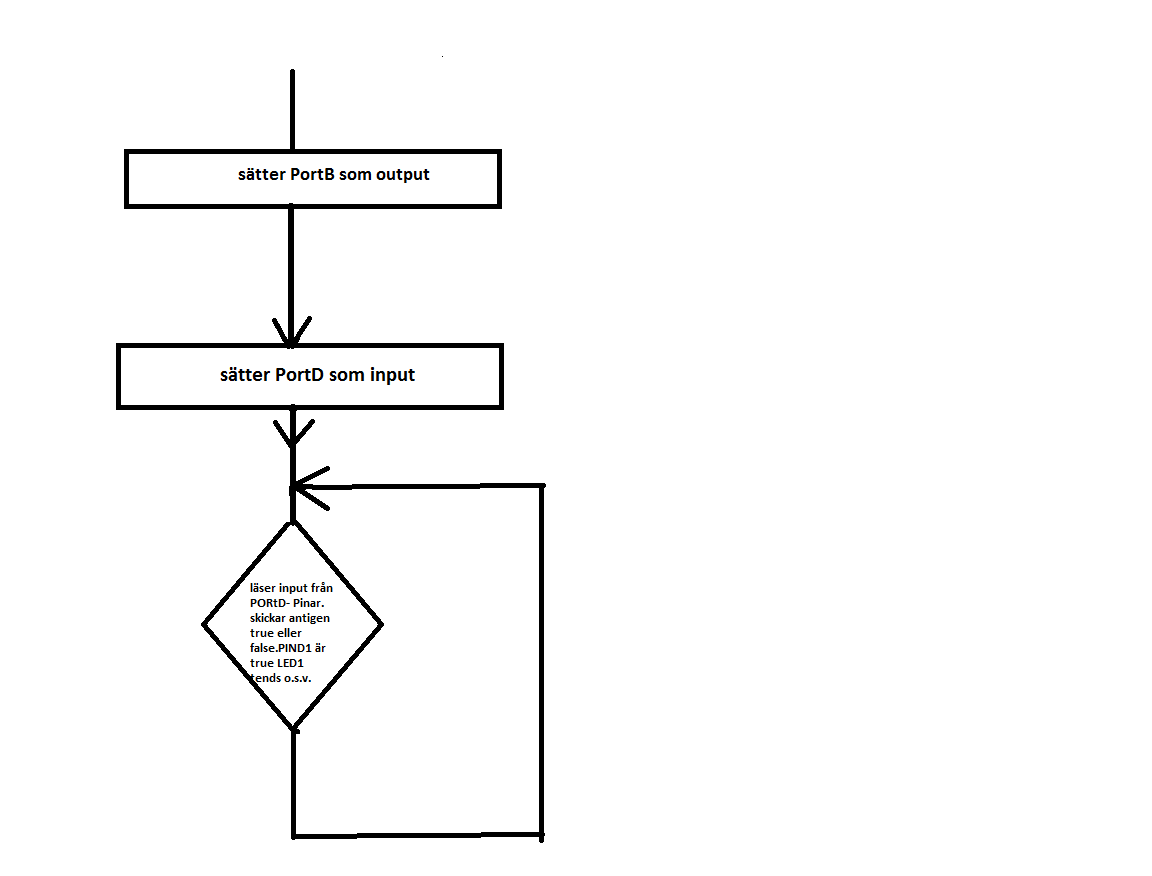
loop: ; början av loop.

in r16, PIND ; läser input från PORtD- Pinar.

andi r16, 0xff ; den gemför ETt-och-ETT är true, NOLL-och-NOLL är false. skickar antigen true eller false.

out PORTB, r16 ; True eller false skickar till output PORTEN, PORTB.

rjmp loop ; retunerar till loop.



;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

; ED2022, Computer Technology I

; Lab 1, task 3

; Hardware: STK600, CPU ATmega2560

; Date: 2015-11-26

; Authors: Ivan Hussein

; Function:

; --------

; When you press switch, SW5 the LED, LED0 turns ON.

; Used subroutines:

; Global subroutines (that can be used from other programs):

; Other information:

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

.INCLUDE "m2560def.inc"

.org 0x00

ldi r16, 0b00000001

out DDRB, r16 ; sätter PortB som output.

ldi r16, 0b11011111

out DDRD, r16 ; sätter PortD som input.

ldi r19, 0b00000001

loop:

in r17, PIND ; läser input från PORtD- Pinar.

cpi r17, 0b00100000 ; jmf med en konstan

breq ledon ; om du är lika hoppar den till ledon

cpi r17, 0b00100000 ; om de är olika hoppar den till ledoff

breq ledoff

ledoff:

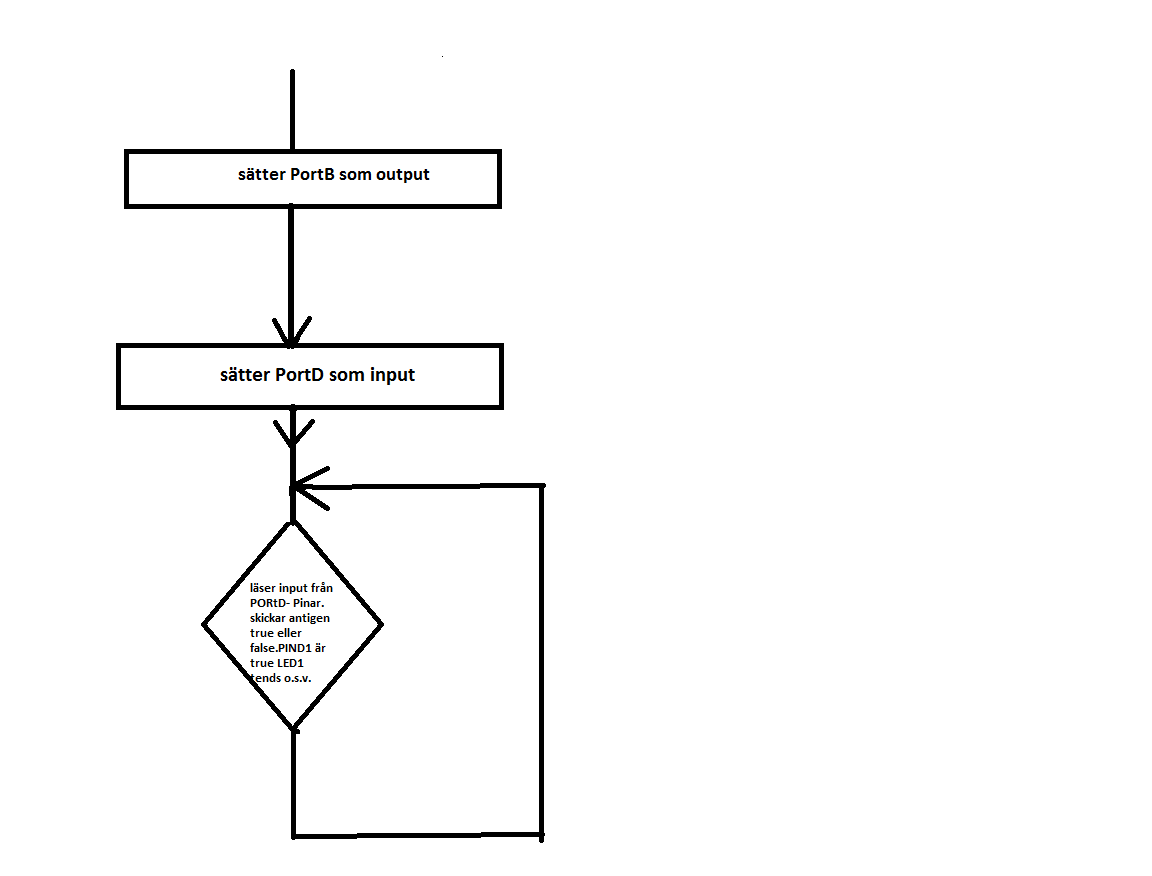
ldi r17, 0x10 ; här släker led

out PORTB, r17

jmp loop

ledon: ; här slår på led

out PORTB, r19

jmp loop

;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

; ED2022, Computer Technology I

; Lab 1, task 5

; Hardware: STK600, CPU ATmega2560

; Date: 2015-11-26

; Authors: Ivan Hussein

; Function:

; --------

; Ring Counter. LSL is used with a delay of approximately 0.5 sec in between each count.

; Used subroutines: Delay as a subroutine

; Global subroutines (that can be used from other programs):

; Other information: Stack Pinter

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

.include "m2560def.inc"

ldi r20, HIGH(0x45F) ; R20 = high part of RAMEND address ; nollställer r16 och intierar DDRA som output

out 0x3E,R20 ; SPH = high part of RAMEND address

ldi R20, low(0x45F) ; R20 = low part of RAMEND address

out 0x3D,R20 ; SPL = low part of RAMEND address

start: ; nollställer r16 och intierar DDRA som output

ldi r16, 0x00

ldi r17, 0xff

out DDRA, R17

loop: ; loppen startar

dec r17 ; r17 minskar

call Delay ; kallar Delay subrotine

out PORTA, r17 ; r17 skicka till output

inc r17 ; ökar r17

lsl r17 ; logical shift left. går mot vänster

cp r17, r16 ; jämför om de är inte samma så går den till loop

brne loop

jmp start ; om r17 och r16 är lika gå till start.

Delay: ; Delay subrutin

ldi r18, 255 ; r18 = limit value

ldi r19, 0 ; r19 = loop counter

ldi r23, 255

ldi r24, 0

del\_1:

del\_2:

inc r24 ; r19 = r19+ 1

cp r24, r23 ; compare r18 - 19

brne del\_2 ; if greater or equal, go back

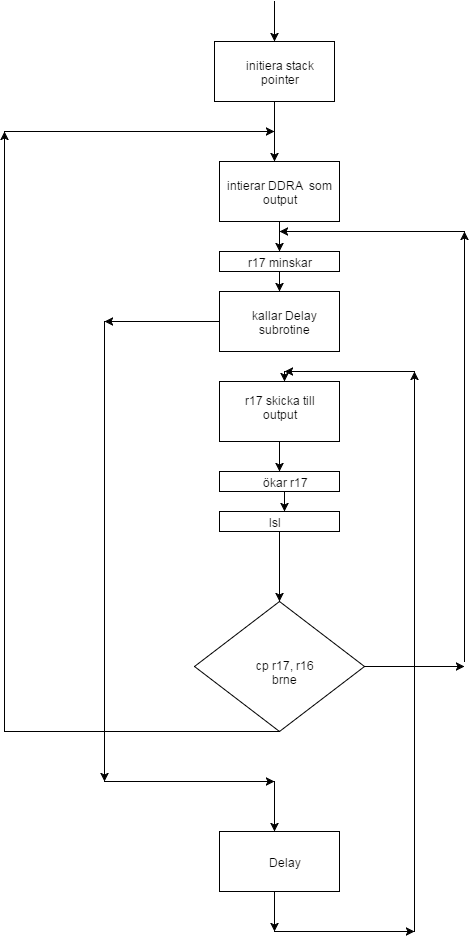
ldi r24, 0

dec r18

cp r19, r18

brne del\_1

ret



;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

; ED2022, Computer Technology I

; Lab 1, task 6

; Hardware: STK600, CPU ATmega2560

; Date: 2015-11-26

; Authors: Ivan Hussein

; Function:

; --------

; Johnson Counter. LSL and LSR are used with a delay of approximately 0.5 sec in between each

; count.

; Used subroutines: Delay as a subroutine

; Global subroutines (that can be used from other programs):

; Other information: Stack Pinter

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

.include "m2560def.inc"

ldi r20, HIGH(0x45F) ; R20 = high part of RAMEND address ; intiera stack pointer

out 0x3E,R20 ; SPH = high part of RAMEND address

ldi R20, low(0x45F) ; R20 = low part of RAMEND address

out 0x3D,R20 ; SPL = low part of RAMEND address

start: ; huvudprogram börjar

ldi r16, 0x00 ;

ldi r17, 0xff

out DDRA, r17 ; sätter DDRA som output

call loop\_mot\_v ; subrutine call, vänsträknare

ldi r16, 0xff

ldi r17, 0xff

out DDRA, r17

call loop\_mot\_h ; subrutine call, högräknare

slut:

cp r17, r16

brne slut ; jämför r16 med r17 om de är lika hoppar de till slut

jmp start ; om de är lika hoppar de till start

loop\_mot\_v: ; vänsträknare börjar

call Delay ; Delay call

lsl r17 ; en bit åt vänster i r17

out PORTA, r17 ; r17 i output

cp r17, r16 ; jämför r16 med r17

brne loop\_mot\_v ; om de inte lika hoppar till vänsträknare

loop\_mot\_h: ; högräknare börjar

cpi r17, 0b11111111 ; jämför om r17 me 0xFF

breq loop\_mot\_v ; om de lika hoppa till vänsträknare

call Delay ; Delay call

com r17 ; inverterar r17

lsr r17 ; en bit åt höger i r17

com r17 ; invera r17

out PORTA, r17 ; r17 på output

cp r17, r16 ; jämför r17 med r16

brne loop\_mot\_h ; om inte lika koppa till högrknare

Delay: ; Delay

ldi r18, 255

ldi r19, 0

ldi r23, 255

ldi r24, 0

del\_1:

del\_2:

inc r24

cp r24, r23

brne del\_2

ldi r24, 0

dec r18

cp r19, r18

brne del\_1

ret

