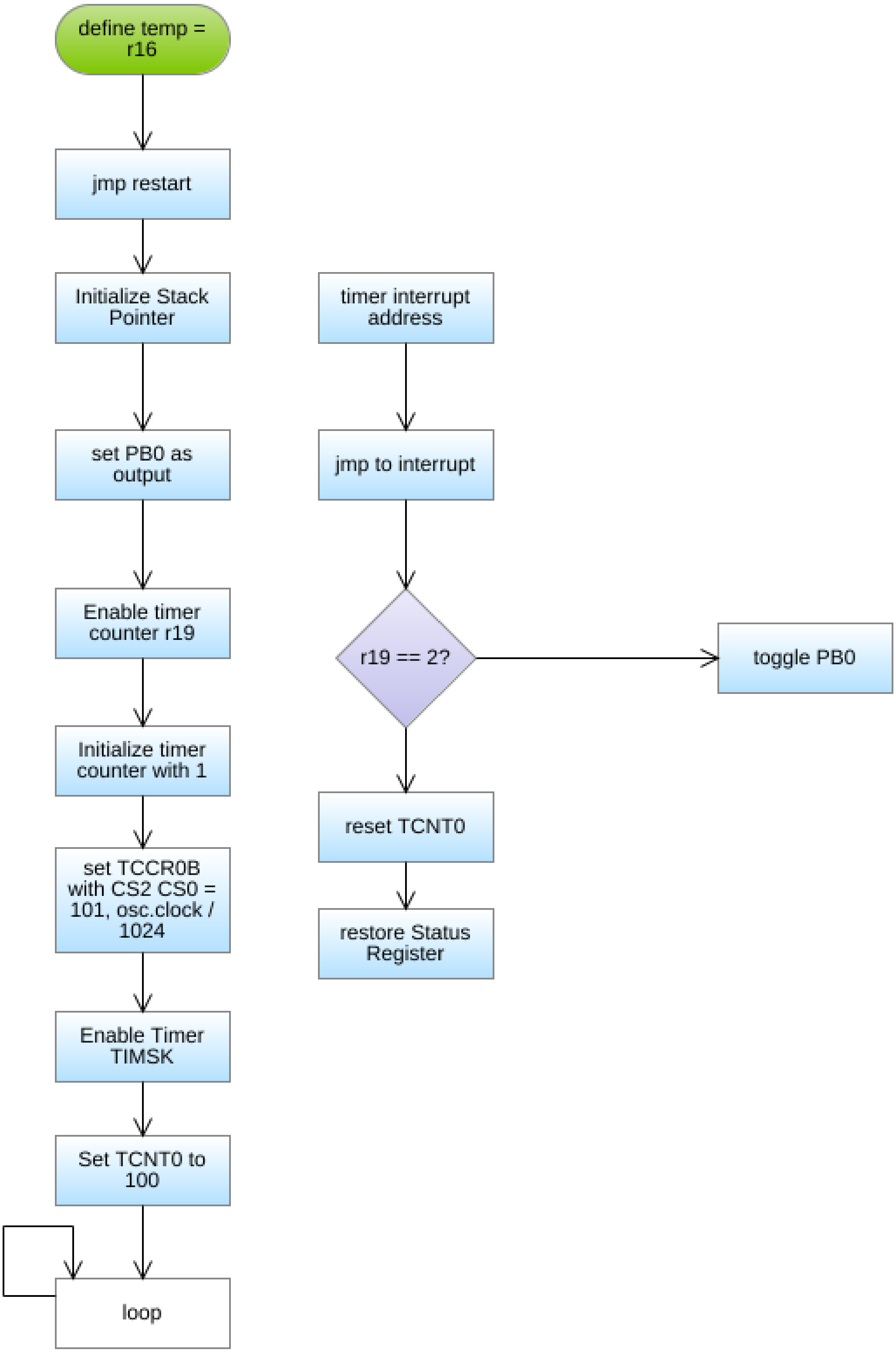
lab4

lh223ng och fe222pa

November 2021

1 task1

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2021-09-18
; Author:
; Student name Li Ang Hu
; Student name Fredric Eriksson Sep lveda
;
; Lab number: 4
; Title: task1
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Write a program in Assembly that creates a square wave. One LED should be conn
with the frequency 1 Hz. Duty cycle 50%. (On: 0.5 sec, Off: 0.5 sec.) Use the timer functio
create an interrupt with 2 Hz, which change between On and Off in the interrupt subroutine

; Input ports: No input
;
; Output ports: PORTB
;
; Subroutines: If applicable.
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: (Description and date)
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.def temp = r16

jmp restart
.org OVF0addr
        jmp timer0_int


restart:
ldi temp, LOW(RAMEND)
out SPL, temp
ldi temp, HIGH(RAMEND)
out SPH, temp
ldi temp, 0x01
out DDRB, temp
mov r6, temp ; store 0x1
ldi r19, 0; 4 timer counter


ldi r17, 0xff; turn LEDs off
ldi temp, 0x05 ; prescaler value to TCCR0B
out TCCR0B, temp ; CS2 CS0 = 101, osc.clock / 1024

ldi temp, (1<<TOIE0) ; Timer 0 enable flag, TOIE0
sts TIMSK0, temp ; to register TIMSK
ldi temp, 100 ; starting value for counter
out TCNT0, temp ; counter register
sei ; enable global interrupt



start:
rjmp start ; main loop


timer0_int:
push temp ; timer interrupt routine
in temp, SREG ; save SREG on stack
push temp


cpi r19, 2;
breq toggle
inc r19
; additional code to create the square output
ldi temp, 140
out TCNT0, temp; starting value for counter
pop temp
out SREG, temp ; restore SREG
pop temp ; restore register

reti ; return from interrupt

; toggle led
toggle:
        eor r18, r6
        out portb, r18
        ldi r19, 0
        ret
```

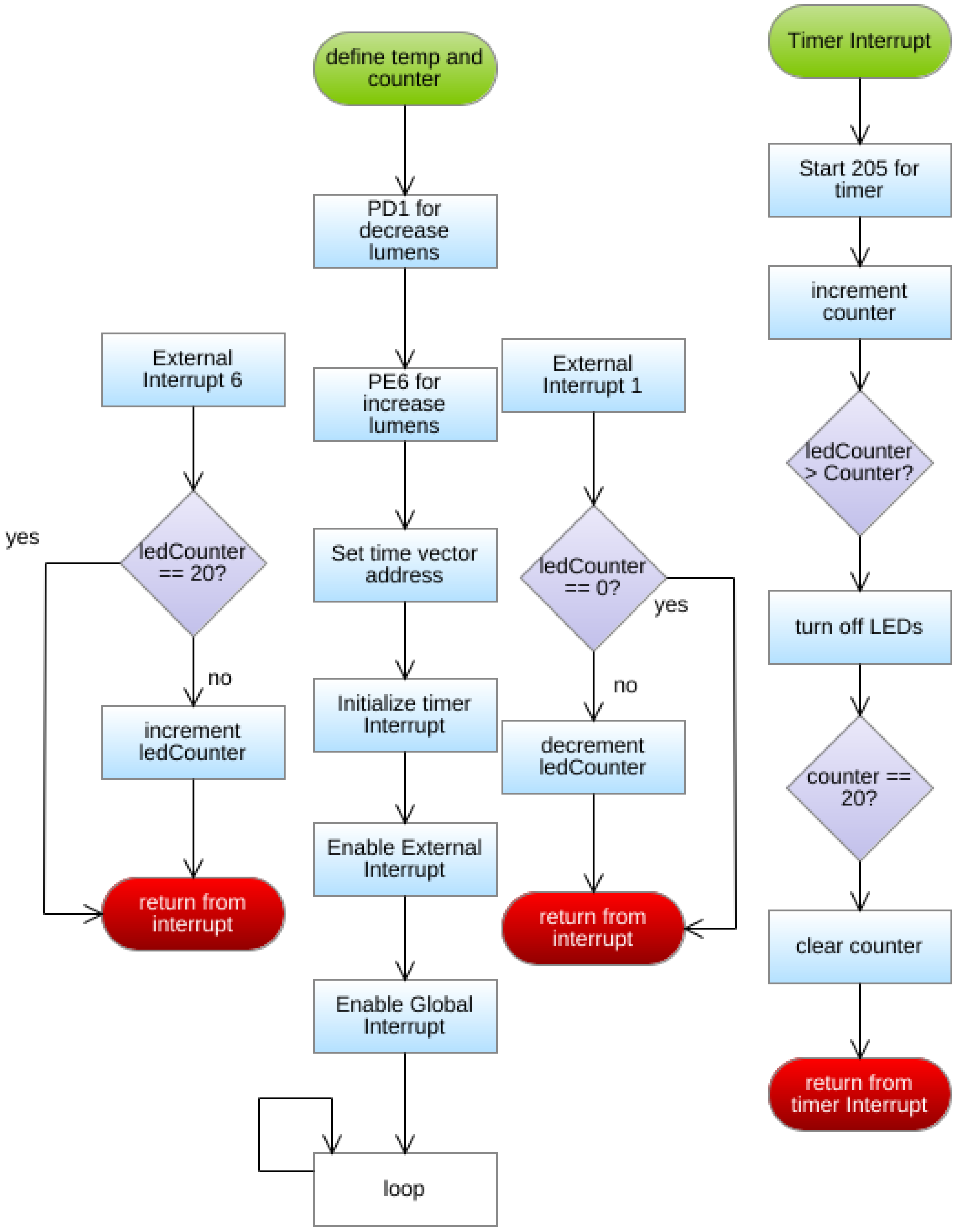


2 task2

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2021-09-18
; Author:
```

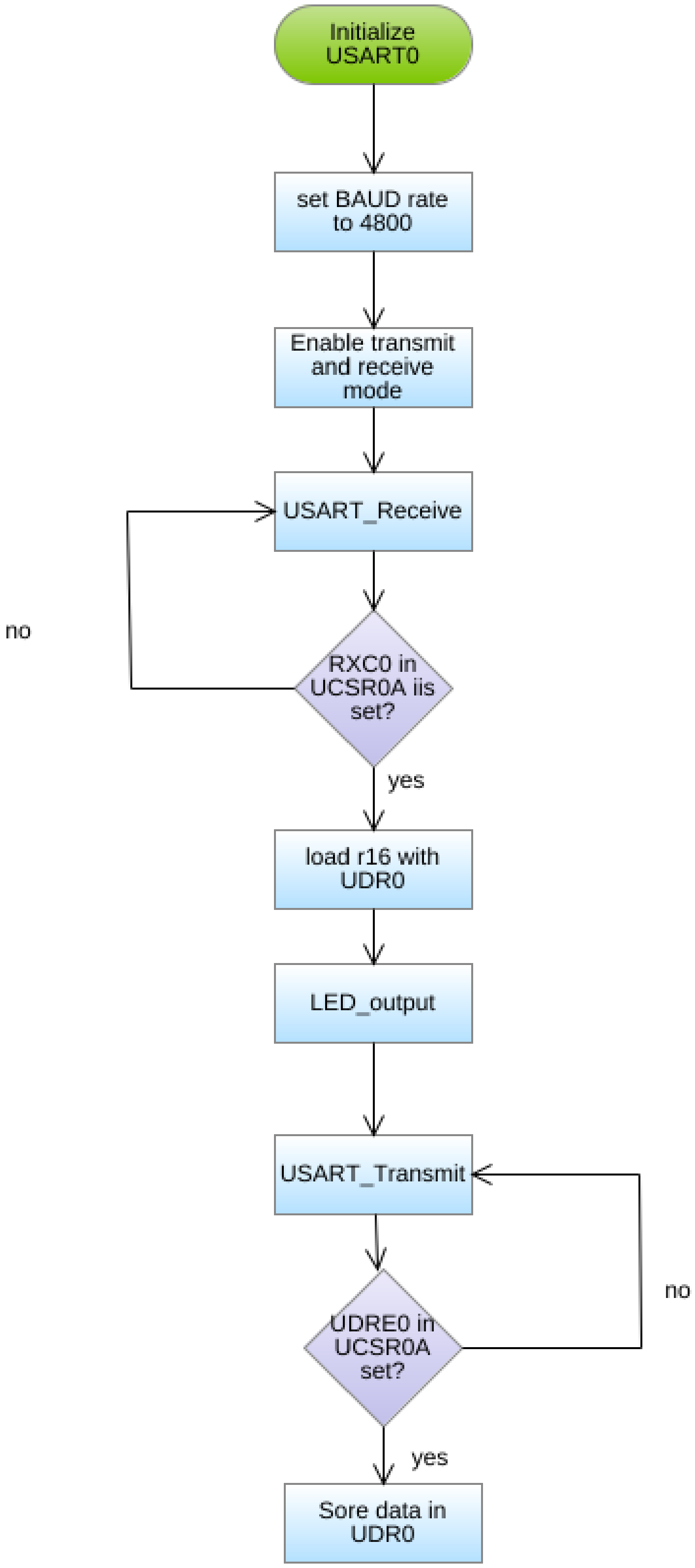
```
; Student name Li Ang Hu
; Student name Fredric Eriksson Sep lveda
;
; Lab number: 4
; Title: task2
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Modify the program in Task 1 to obtain Pulse Width Modulation (PWM). The freque
fixed, but the duty cycle should be possible to change. Use two push buttons to change the
cycle up and down. Use interrupt for each push button. The duty cycle should be possible t
from 0 % up to 100 % in steps of 5 %. Connect the output to an oscilloscope, to visualize
change in duty cycle

; Input ports: PORTD and PORTE
;
; Output ports: PORTB
;
; Subroutines: If applicable.
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: (Description and date)
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.def temp = r16
.def ledCounter = r17
.def counter = r18
.def LEDstate = r22
.def switch0 = r23
.def switch1 = r24

.equ timer = 205
.org 0x00
rjmp start

.org INT1addr ; PD1 for decrease lumens
rjmp increase

.org INT6addr ; PE6 for increase lumens
rjmp decrease

.org OVF0addr
jmp timer0_int

.org 0x72
start:

        ldi temp, 0b00111100                    ;set DDRB(0b00111100) as output
        out DDRB, temp

        ldi temp, 1<<CS00               ; No prescaler
        out TCCR0B, temp                ;
        ldi temp, (1<<TOIE0)    ; Timer 0 enable flag, TOIE0
        sts TIMSK0, temp                ; to register TIMSK

        ldi temp, timer                 ; starting value for counter
        out TCNT0, temp                 ;50MS

        ldi temp, (1<<INT1) | (1<<INT6) ; enable INT1 and INT6
        out EIMSK, temp

        ldi temp, 0b00001000    ; falling edge for INT1
        sts EICRA, temp

        ldi temp, 0b00100000    ; falling edge for INT6
        sts EICRB, temp

        sei                                              ; enable global interrupt

        clr counter
        ldi ledCounter,10

loop:
        rjmp loop

increase:
        cpi ledCounter,20               ;increases ledcounter until it is 20
        breq retiInc
        inc ledCounter
        retiInc:
        reti

decrease:                                       ;decreases ledcounter until it is 0
        cpi ledCounter,0
        breq retiDec
        dec ledCounter
        retiDec:
        reti

timer0_int:


        ldi temp, timer ; starting value for counter
        out TCNT0, temp

        inc counter

        cp ledCounter, counter  ;this achieves the PWM effect
        brge turn_off ; Branch if Greater or Equal, Signed
        clr temp
        out PORTB, temp ; turn leds on
        rjmp end

        turn_off:
        ser temp                        ;turns off the led (Set Register temp to 0xff)
        out PORTB, temp

        end:                            ;when this is reached timer0e_int nds
        cpi counter,20
        brne doNothing
        clr counter  ; Clear Register counter

        doNothing:
                nop

reti
```

3 task3 and task4 combined


define temp and counter
PD1 for decrease lumens
PE6 for increase lumens
Set time vector address
Initialize timer Interrupt
Enable External Interrupt
Enable Global Interrupt
loop
External Interrupt 6
ledCounter == 20?
yes
no
increment ledCounter
return from interrupt
External Interrupt 1
ledCounter == 0?
yes
no
decrement ledCounter
return from interrupt
Timer Interrupt
Start 205 for timer
increment counter
ledCounter > Counter?
turn off LEDs
counter == 20?
clear counter
return from timer Interrupt

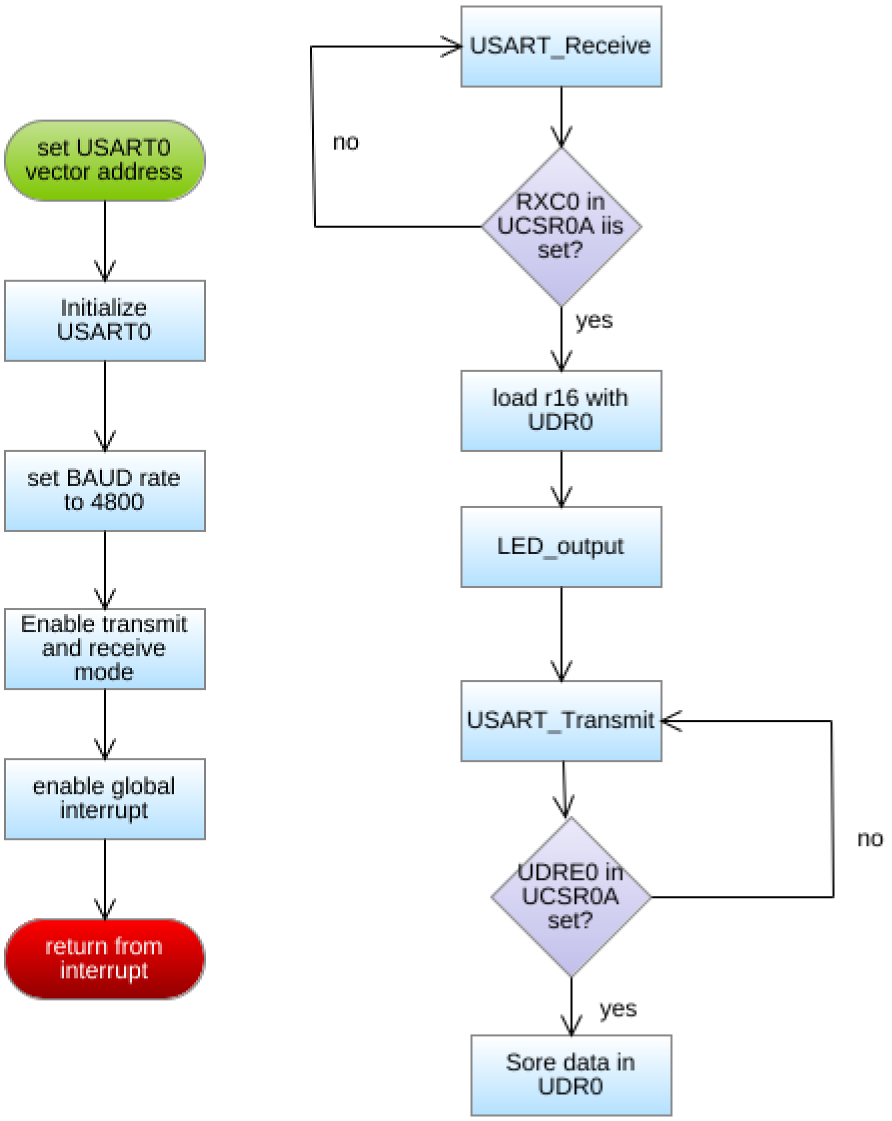
```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2021−09−18
; Author:
; Student name Li Ang Hu
; Student name Fredric Eriksson Sep lveda
;
; Lab number: 4
; Title: task3 and task4
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Write a program in Assembly that uses the serial communication port0 (RS232).
computer to the serial port and use a terminal emulation program. (Ex. Hyper Terminal) The
program should receive characters that are sent from the computer, and show the code on th
For example, if you send character A, it has the hex code $65, the bit pattern is 0110 010
should be displayed with LEDs On for each    one    . Use polled UART, which means that the
should be checked regularly by the program.
Serial communication, Wikipedia: https://en.wikipedia.org/wiki/Serial_communication

; Input ports: USART0
;
; Output ports: PORTB
;
; Subroutines: If applicable.
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: (Description and date)
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
USART_Init:
ldi r16, 0xff
out ddrb, r16
out portb, r16 ; turn leds off

; Set baud rate
ldi r16, 12             ; 4800 bps, Asynchronous Normal mode (U2Xn = 0) => UBBRR = 10^6/(48
sts UBRR0L, r16

; Enable receiver and transmitter
ldi r16, (1<<RXEN0)|(1<<TXEN0)
sts UCSR0B, r16

loop:

        USART_Receive:
        ; Wait for data to be received
        lds r17, UCSR0A
        sbrs r17, RXC0 ; Skip if Bit in Register Set
        rjmp USART_Receive
        ; Get and return received data from buffer
        lds r16, UDR0


        LED_output:       ;Show Data on LEDs
                com r16
                out PORTB, r16    ;Write character to PORTB
                com r16
```

```
USART_Transmit:
; Wait for empty transmit buffer
lds r17, UCSR0A
sbrs r17, UDRE0
rjmp USART_Transmit
; Put data (r16) into buffer, sends the data
sts UDR0, r16

rjmp loop          ;Return to loop
```



4 task5

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2021-09-18
; Author:
; Student name Li Ang Hu
; Student name Fredric Eriksson Sep lveda
;
; Lab number: 4
; Title: task5
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Do task 3 and 4, but use Interrupt instead of polled UART.

; Input ports: USART0
;
; Output ports: PORTB
;
; Subroutines: If applicable.
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: (Description and date)
;
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.org 0x00
jmp USART_Init

.org URXC0addr
jmp USART_Receive

/*.org UTXC0addr
jmp USART_Transmit*/

USART_Init:
; enable LEDs output PORTB
ldi r16, 0xff
out ddrb, r16
;out portb, r16 ; turn leds off

; Set baud rate
ldi r16, 12             ; 4800 bps, Asynchronous Normal mode (U2Xn = 0) => UBBRR = 10ˆ6/(48
sts UBRR0L, r16
/*ldi r16, (1<<U2X1)
sts UCSR1A, r16*/
; Enable receiver and transmitter
ldi r16, (1<<RXEN0)|(1<<TXEN0) | (1<<RXCIE0)
sts UCSR0B,r16

sei


loop:
        nop
        nop
        rjmp loop       ;Return to loop


USART_Receive:
        ; Wait for data to be received
        lds r17, UCSR0A
        sbrs r17, RXC0 ; Skip if Bit in Register Set
        rjmp USART_Receive
        ; Get and return received data from buffer
        lds r16, UDR0

LED_output:     ;Show Data on LEDs
        com r16
        out PORTB,r16    ;Write character to PORTB
        com r16

USART_Transmit:
        ; Wait for empty transmit buffer
        lds r17, UCSR0A
        sbrs r17, UDRE0
        rjmp USART_Transmit
        ; Put data (r16) into buffer, sends the data
        sts UDR0, r16

        reti
```


set USART0 vector address
Initialize USART0
set BAUD rate to 4800
Enable transmit and receive mode
enable global interrupt
return from interrupt
USART_Receive
no
RXC0 in UCSR0A iis set?
yes
load r16 with UDR0
LED_output
USART_Transmit
no
UDRE0 in UCSR0A set?
yes
Sore data in UDR0