Linnæus University

1DT301 – Lab 1

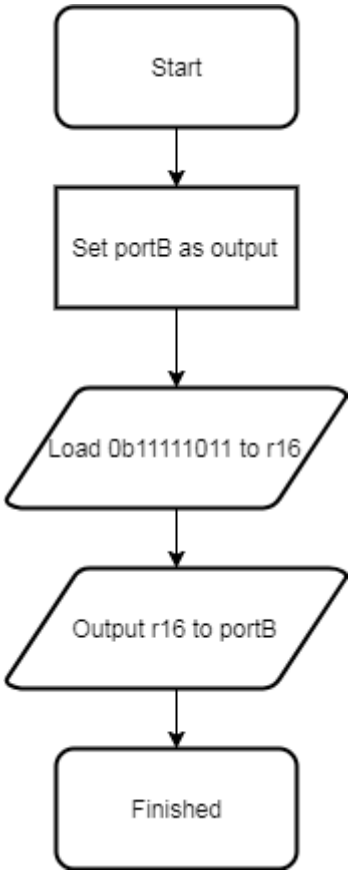
Authors : Matus Maruna & John Charo

Computer Technology 1
17HT-1DT301

TASK 1 :

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2016-09-05
; Author:
; Student  Matus Maruna
; Student  John Charo
; Lab number: 1
; Title: Lighting LED number 2
; Hardware: STK600, CPU ATmega2560
; Function: The function of this program is that it turns on LED2 on the board
; Input ports: None. The input was written in the program.
; Output ports: Port B is connected to the LEDs on the board
; Subroutines: None.
; Included files: m2560def.inc
; Other information:
; Changes in program: None.
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

.include "m2560def.inc"
ldi r16, 0xFF
out 0x04, r16 ; setting PORTB as output
ldi r16, 0b11111011 ; loading the value to be output to PORTB in order to turn on the second LED
out 0x05, r16 ; outputting the value to PORTB
```

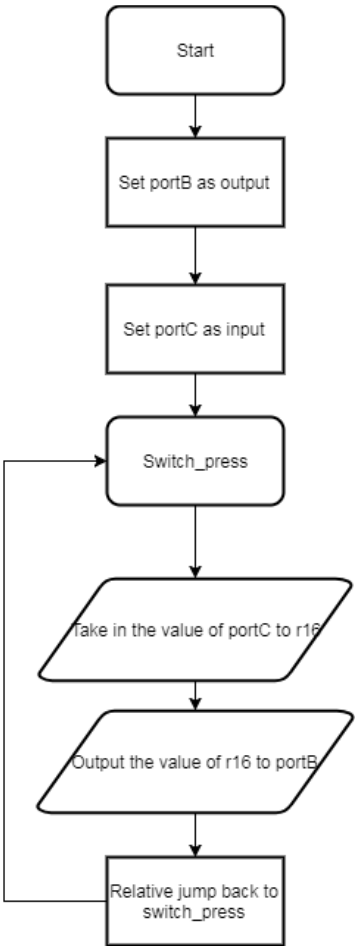
Summary of Task1 : We set portB as output, and then we load the value to register 16 and output r16 to portB. This value turns on LED2.



TASK 2 :

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2016-09-05
; Author:
; Student  Matus Maruna
; Student  John Charo
; Lab number: 1
; Title: Connecting switches with corresponding LEDs
; Hardware: STK600, CPU ATmega2560
; Function: The program turns on the LED when the corresponding switch below it is pressed on
; Input ports: Port A is connected to switches
; Output ports: Port B is connected to the LEDs on the board
; Subroutines: switch_press is a loop that listens for a switch on the board being pressed
; Included files: m2560def.inc
; Other information: None.
; Changes in program: None.
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
 .include "m2560def.inc"
ldi r16, 0xFF
out 0x04, r16 ; setting portB as output
ldi r16, 0x00
out 0x07, r16 ; setting portC as input
ldi r16, 0xFF
out 0x05, r16 ; Setting portB as output

switch_press:  ; loop to listen to the switches
in r16, 0x06  ; load in r16 the value from portC
out 0x05, r16 ; output the value of r16 to portB

rjmp switch_press
```

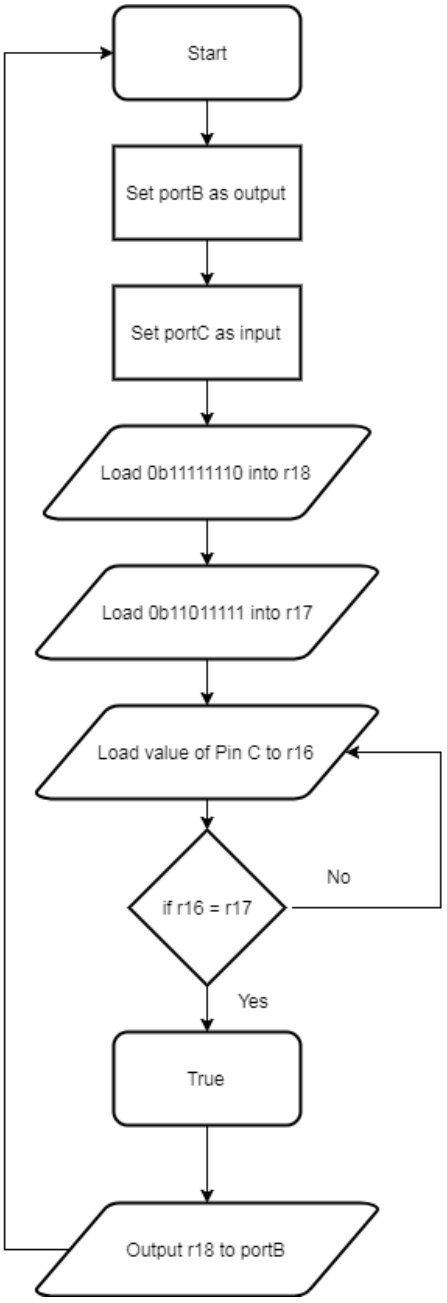
Summary of Task 2 : We set portB as output and portC as input. Then continuously set the value of portC to r16 and output r16 to portB. This way the corresponding button will light the corresponding LED.



TASK 3:

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2016-09-05
; Author:
; Student Matus Maruna
; Student John Charo
; Lab number: 1
; Title: Switch 5 turns on LED0
; Hardware: STK600, CPU ATmega2560
; Function: The function of this program is to turn on the LED0 when switch 5 is pressed. All other   switches are ignored
; Input ports: Port C is connected to switches
; Output ports: Port B is connected to the LEDs on the board
; Subroutines: switch_press5 is a loop that listens for a switch on the board that corresponds to ;switch number 5 being pressed
;            true is an outcome that turns on the first LED when the conditions inside switch_press5 are satisfied
; Included files: m2560def.inc
; Other information: None.
; Changes in program: None.
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

.include "m2560def.inc"

ldi r16, 0xFF
out 0x04, r16
ldi r16, 0x00
out 0x07, r16
ldi r16, 0xFF
out 0x05, r16
ldi r17,0b11011111 ; desired value for the switch press
ldi r18,0b11111110 ; desired value for the LED output
switch_press5: ; loop to listen to the switch press
in r16, 0x06 ; input the value from PortC which is connected to switches into r15
cp r16,r17 ; compare input with the desired input
breq true ;branch if equal to "true"
rjmp switch_press5
true: out 0x05,r18 ; output the desired output to portB
```

Task 3 Summary : This is similar to task 3 except we add a compare statement that checks if button 5 was pressed. If so then it outputs the desired value. If not the loop continues and does nothing.

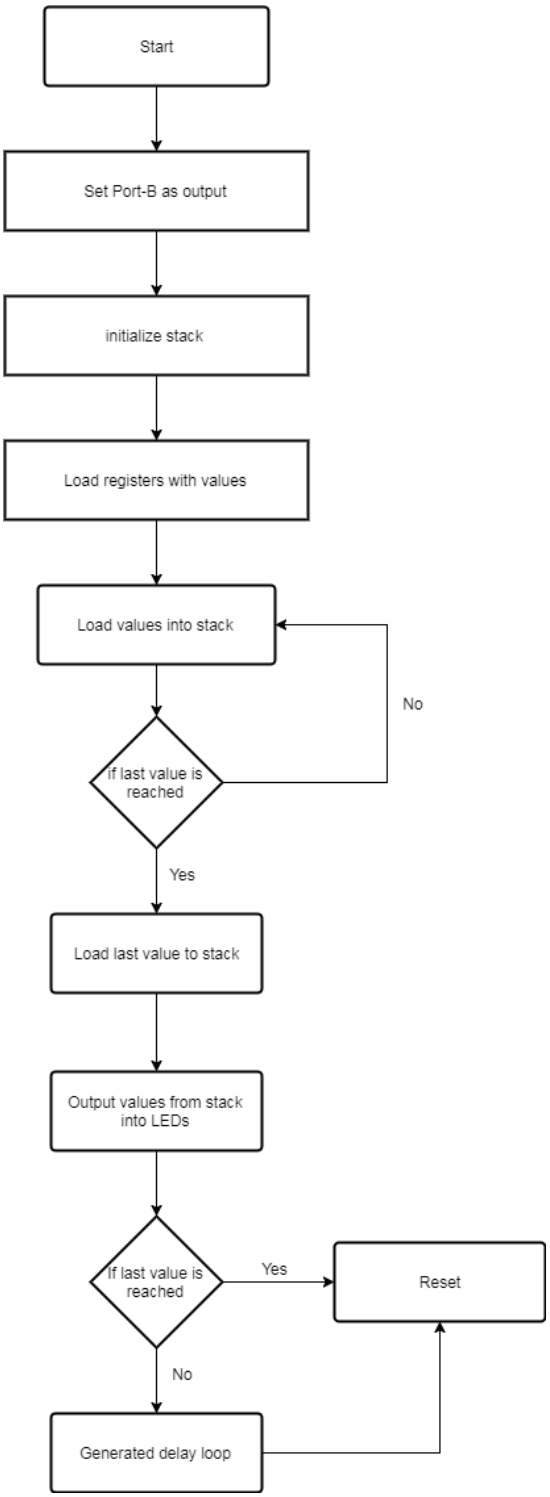


TASK 5:

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2016-09-05
; Author:
; Student Matus Maruna
; Student John Charo
; Lab number: 1
; Title: Infinite Ring counter
; Hardware: STK600, CPU ATmega2560
; Function: The function of this program is to turn on the LEDs in sequence by their order on the board
;         otherwise refered to as a johnson counter
; Input ports: None.
; Output ports: Port B is connected to the LEDs on the board
;
; Subroutines: "load" is an initializing sequence that loads the values needed to be output to PortB into a stack, these values are loaded
;           in reverse order that they going to be output, it will call "first" when reaching the last value (first to be output) which is pushed in
;           with the "first" subroutine
;           "ring" is a loop that outputs the values from the stack by poping them and then loading them onto PortB, it will call "reset" when done.
;           "reset" is called when the "johnson" subroutine reaches the last value and will reset the values so that "load" can be called again
;           "DELAY" is an automatically generated delay of 500ms
; Included files: m2560def.inc
; Other information: None.
; Changes in program: None.
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<


.include "m2560def.inc"
;inilizing output
ldi r16, 0xFF
out 0x04, r16
ldi r16, 0xFF
out 0x05, r16
; initlizing stack. Copy pasted from the assignment question
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

ldi r17, 0b10000000
ldi r16, 0b01111111
ldi r19, 0b00000001
ldi r25, 0b01111111

load:
mov r16,r17 ;move value of r17 to r16
com r16  ;inverse the value of r16 from 01111111 to 1000000
push r16 ; push the value of r16 to the bottom of the stack
lsr r17  ; move the 1 in r17 to the right by pushing in a 0
cp r17,r19 ; compare r17 to r19 so it knows when to stop
breq first ; branch to first when r17 is equal to r19
rjmp load ; loop back if not

first:
mov r16,r17 ;loop that will produce the last value of to be pushed into the stack and the first value to be output
com r16
push r16
rjmp ring

ring:
pop r16 ; pop a value out of the stack and into r16
out PORTB,r16 ; output the value of r16 to portB
cp r16,r25 ; compare r16 to r25 or the last value to be output
breq reset ; if last value is reached branched to reset
call DELAY ; if last value is not reached jump to superdelay
rjmp ring
```

```
; Generated by delay loop calculator
; at http://www.bretmulvey.com/avrdelay.html
;
; Delay 500 000 cycles
; 500ms at 1.0 MHz
DELAY:
   ldi  r18, 3
   ldi  r19, 138
   ldi  r20, 86
L1: dec  r20
   brne L1
   dec  r19
   brne L1
   dec  r18
   brne L1
   rjmp PC+1

ret


reset:
ldi r17, 0b10000000 ; resets r17 and r16 to their original values so that the loop can be called again
ldi r16, 0b01111111
call DELAY
rjmp load
```

Flowchart task 5: We have split the flowcharts into 6 parts so it's easier to follow, each flowchart represents a subroutine. The program starts by preloading values into the stack. Then outputting them from the stack with a delay in between. When it reaches the last value resets the variables so it can go from the start in an infinite loop.



TASK 6

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2016-09-05
; Author:
; Student Matus Maruna
; Student John Charo
;
; Lab number: 1
; Title: Infinite Ring counter
; Hardware: STK600, CPU ATmega2560
; Function: The function of this program is to execute the ring counter
; Input ports: None.
; Output ports: Port B is connected to the LEDs on the board
;
; Subroutines: "johnson" is a subroutine loop that pushes in values of r16 and preforms a logic shift on them to reflect the way
;             the values are going to be output. On completing the output, it branches to "backward" subroutine
;            "back_loop" is similar to "johnson and uses the values that were placed into stack in the "johnson" to output the LED flashes
;             backwards, when done it branches to "forward".
;            "forward" loads in the values needed for "johnson" and outputs the last value branching to "johnson" when done.
; Included files: m2560def.inc
; Other information: None.
; Changes in program: None.
;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
.include "m2560def.inc"

ldi r16, 0xFF ; setting portB as output
out 0x04, r16
ldi r16, 0xFF
out 0x05, r16
; initilizing the stack, copy pasted from the assignment question
ldi r24, HIGH(RAMEND) ; R20 = high part of RAMEND address
out SPH,R24 ; SPH = high part of RAMEND address
ldi R24, low(RAMEND) ; R20 = low part of RAMEND address
out SPL,R24 ; SPL = low part of RAMEND address
;Loading needed values into registers
ldi r16, 0b11111111
ldi r17, 0b00000000
ldi r20, 0b11111111

;Ring loop that lights LEDS from 0 to 7
johnson:
push r16 ; push the value to stack, saving it for later
ldi r19, 0 ; load 0 to r19 to clear out superdelay counter
cp r16,r17 ; compare r16 to the last value being output which is in r17
BREQ backward ; branch if equal to backward subroutine
lsl r16 ; pushing in 0 from the right shifting the bits 1 place to the left and removing leftmost bit.
out portB,r16 ; outputting the value to portB
call DELAY
rjmp johnson

forward: ; subroutine that switches and loads the values needed for the loop to go forward from 0 to 7
out portB, r16
ldi r16,0b11111111
call DELAY
rjmp johnson

back_loop: ; same as the front loop with minor adjustments
ldi r19, 0 ; loading a value to reset superdelay_back
cp r16,r20 ; compare r16 to last value to be output and branch if its the last value
BREQ forward ;
pop r16 ; pop a value from stack to r16, values previously saved in the front ring loop
out portB,r16 ; output the value into portB
call DELAY
rjmp back_loop
```

```
backward: ; subroutine that switches and loads the values needed for the loop to go backwards from 7 to 0
out portB, r16
ldi r16,0b00000000
call DELAY
rjmp back_loop

; Generated by delay loop calculator
; at http://www.bretmulvey.com/avrdelay.html
;
; Delay 500 000 cycles
; 500ms at 1.0 MHz
DELAY:
    ldi  r18, 3
    ldi  r19, 138
    ldi  r21, 86
L1: dec  r21
    brne L1
    dec  r19
    brne L1
    dec  r18
    brne L1
    rjmp PC+1
ret
```

Summary of Task6 “johnson” does the first part of the counter and outputs and stores variables going from LED0 to 7 using a delay in between flashes. “backward” subroutine is then called setting up the variables so that “back loop” can be called which outputs the variables from the stack created by “johnson”. When “back loop” is done “forward” subroutine is called and variables are loaded so “johnson” subroutine can start again.