



# Computer Technology I

## Lab. 2 : Subroutines



*Author:* LOIC GALLAND,  
LEONARDO PEDRO

*Supervisor:*

*Semester:* Autumn 2019

*Area:* Computer Science

*Course code:* 1DT301

## **Contents**

<b>1</b>	<b>Task 1 - Switch – Ring counter / Johnson counter</b>	<b>1</b>
<b>2</b>	<b>Task 2 - Electronic dice</b>	<b>4</b>
<b>3</b>	<b>Task 3 - Change counter</b>	<b>7</b>
<b>4</b>	<b>Task 4 - Delay subroutine with variable delay time</b>	<b>9</b>

## 1 Task 1 - Switch – Ring counter / Johnson counter

*Write a program which switch between Ring counter and Johnson counter. You should not use Interrupt in this lab. The pushbutton 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.*

[illegible]

```

        switch
    breq JC ;If they are =, go to Johnson Counter
    cp r17, r22 ;Check if all the lights are
        turned off
    breq RC_light ;
rjmp RC_loop
RC_light:
    rol r17 ;do a rol here because we are not supposed to
        see it appear.
    out portB, r17 ;light up the desired LEDs
    rjmp RC_loop ;go back to the loop to make it
        continue
rjmp RC

JC: ;Johnson Counter Code
    ldi r21, 0b11111110 ;r21 = to light up the LEDs
    ldi r22, 0b11111111 ;Desired condituon
    ldi r23, 0b00000000 ;Desired condition

my_loop1: ;Loop to do the going left part of the Johnson
    Counter
    out portB, r21 ;Light up the corresponding LEDs
    LSL r21 ;Logical shift to the left of R21
    CALL Delay1 ;Delay of 0.5s
    in r25, PINA ;Get the input from PINA
    cp r25, r24 ;Compare input and desired switch
    breq RC If equal go back to Ring Counter
    cp r21, r23 ;compare info with desired one
    breq light ;If equal go to "light" where it is
        going right.
rjmp my_loop1

light: ;initialisation process to be able to go right
    out portB, r23 ;Turn on all the lights
    CALL Delay1 ;Delay of 0.5s
    ldi r21, 0b10000000 ;Set up the first iteration to
        make sure it goes right correctly
    out portB, r21 ;output to PortB
    Second_loop: ;Action of going right here
        in r25, PINA ;check info from switches
        cp r25, r24 ;Compare switches with desired
            switch
        breq RC ;If equal go back to Ring Counter
        out portB, r21 ;Output it to r21
        ASR r21 ;Arithmetic Shift right to be able to
            shift the bits to the right.
        CALL Delay1 ;Delay of 0.5s
        cp r21, r22 ;compare info with desired one
        breq my_loop1 ;if equal go back my_loop1 and
            go right again
    rjmp Second_loop

rjmp JC
Delay1:
; Generated by delay loop calculator
; at http://www.bretmulvey.com/avrdelay.html
; Delay 1 950 500 cycles
; 500ms at 3.901 MHz
    ldi r18, 10
    ldi r19, 230

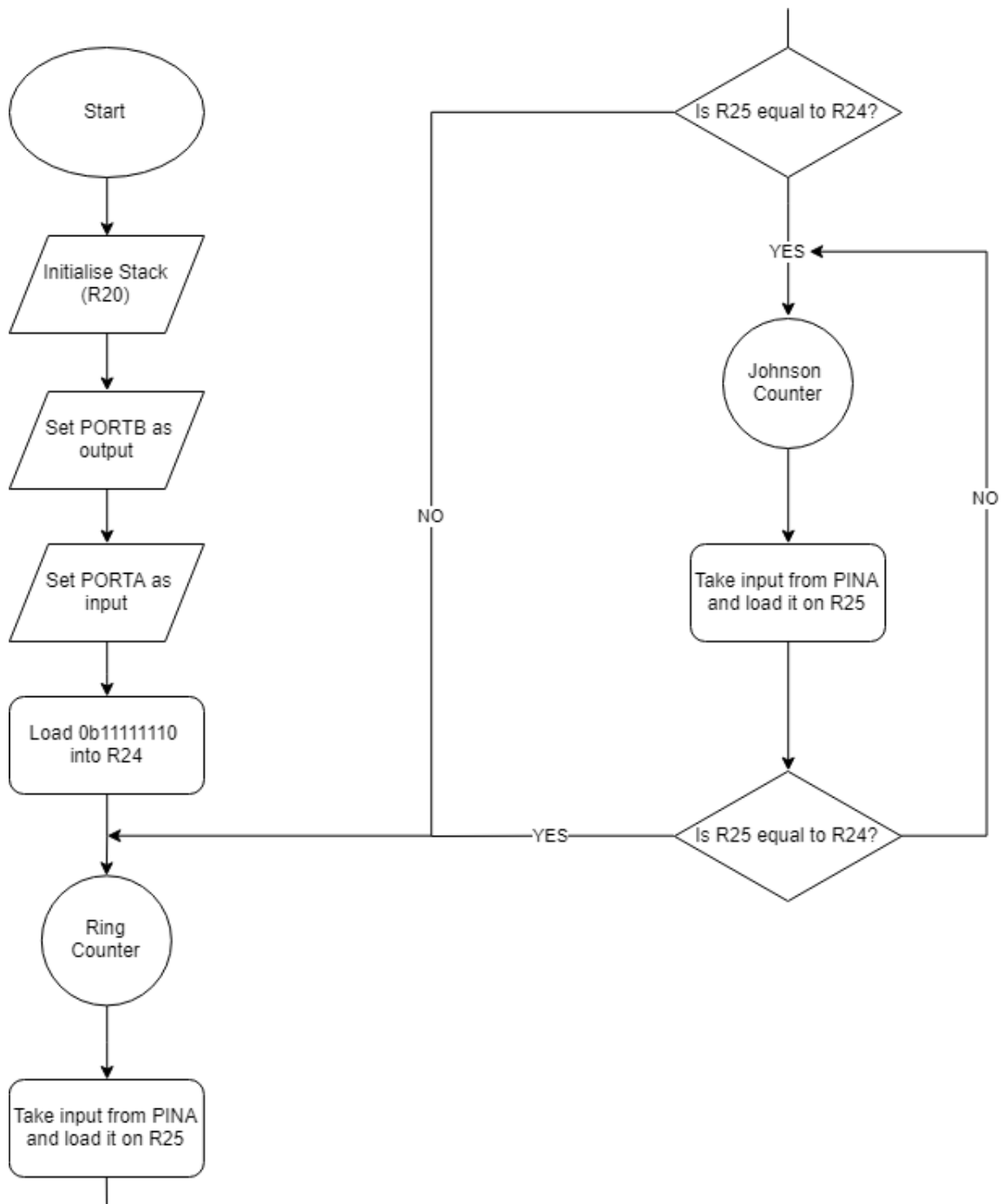
```

```

    ldi r20, 22
L1:  dec r20
    brne L1
    dec r19
    brne L1
    dec r18
    brne L1
RET

```

This is the flowchart of the task 1:



## 2 Task 2 - Electronic dice

*You should create an electronic dice. Think of the LEDs placed as in the picture below. The number 1 to 6 should be generated randomly. You could use the fact that the time you press the button varies in length.*

[illegible]

```

        in r17,PINA      ;Check the info from the switches
        cp r17,r25       ;Check if user has released the switches
        breq RD ;If yes go to RD loop
rjmp Listening_F         or_Switch_Release

reset:  ;to reset the counter
ldi r19,1
rjmp Main

RD:     ;This is where the random will happen
        cpi r19,1        ;When it will be sent to this loop
        breq ONE         ;It will stop on one of those numbers
        cpi r19,2        ;Depending on which number is
        breq TWO         ;r19 right now it will decide which
        cpi r19,3        ;number it will gets
        breq THREE
        cpi r19,4
        breq FOUR
        cpi r19,5
        breq FIVE
        cpi r19,6
        breq SIX
rjmp RD

ONE:     ;If r19 was equal to 1 it will come here
ldi r18,0b11101111      ;Load to r18 the corresponding binary code to
                        make it look like a 1
out PortB,r18           ;Output it to PORTB
rjmp Listening_For_Switch_Press ;and go back to the first loop
TWO:
ldi r18,0b10111011
out PortB,r18
rjmp Listening_For_Switch_Press
THREE:
ldi r18,0b10101011
out PortB,r18
rjmp Listening_For_Switch_Press
FOUR:
ldi r18,0b00111001
out PortB,r18
rjmp Listening_For_Switch_Press
FIVE:
ldi r18,0b00101001
out PortB,r18
rjmp Listening_For_Switch_Press
SIX:
ldi r18,0b00010001
out PortB,r18
rjmp Listening_For_Switch_Press

```

This is the flowchart of the task 2:



### 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.*

[illegible]

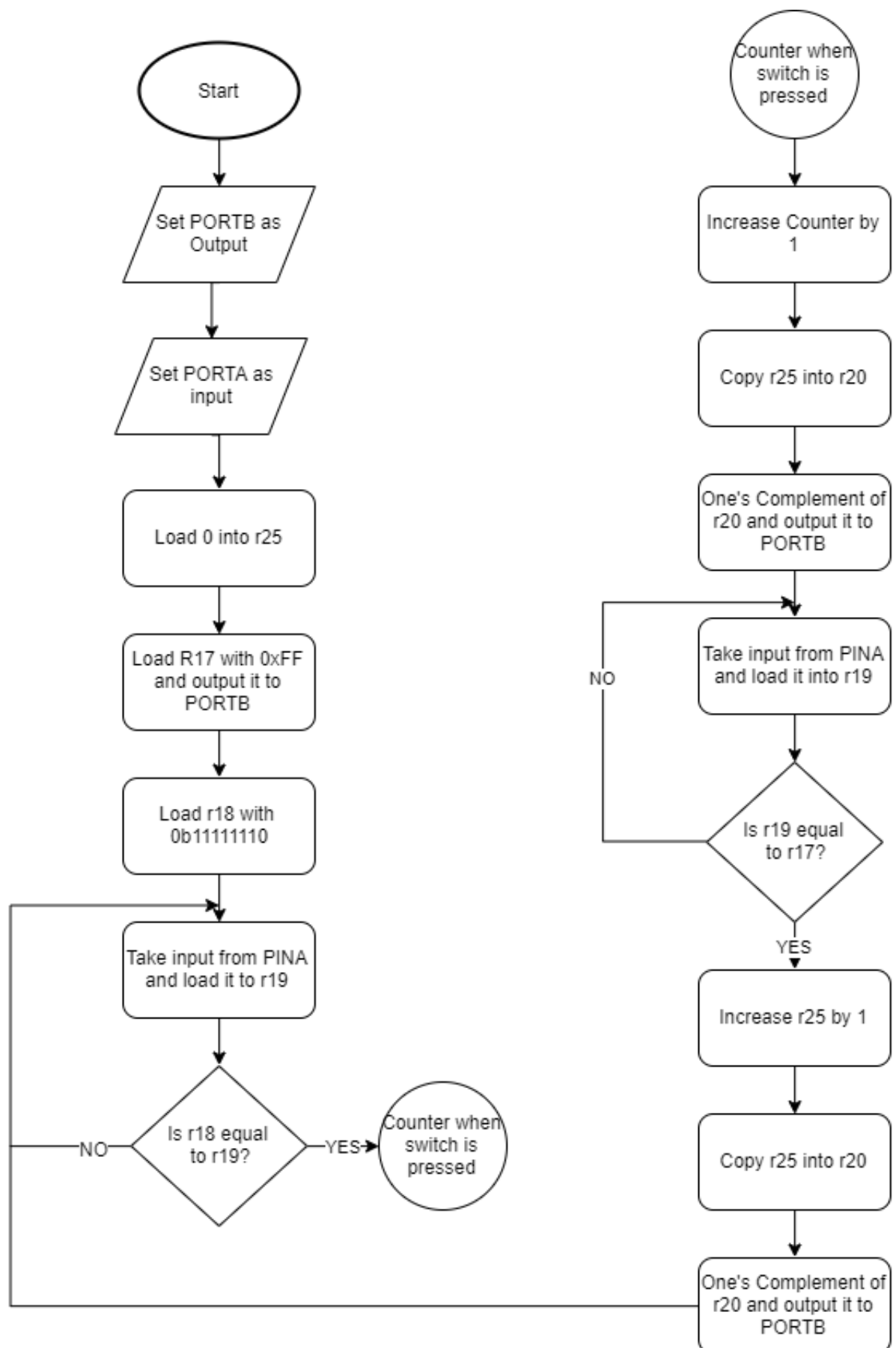
```

counter_when_switch_pressed:
    inc r25
    mov r20,r25
    com r20
    out portB,r20
    loop:
        in r19,PINA
        cp r19,r17
        breq counter_when_switch_released
    rjmp loop

counter_when_switch_released:
    inc r25
    mov r20,r25
    com r20
    out portB,r20
    rjmp my_loop

```

This is the flowchart of the task 3:



#### 4 Task 4 - Delay subroutine with variable delay time

[illegible]

```

        call ms_delay
        sbiw r25:r24,1
        cpi r25, high(0)
        breq reset
        rjmp wait_milliseconds
reset:
RET

ms_delay:
    ; Generated by delay loop calculator
    ; at http://www.bretmulvey.com/avrdelay.html
    ;
    ; Delay 1 000 cycles
    ; 1ms at 1 MHz

    ldi r18, 2
    ldi r19, 75
L1: dec r19
    brne L1
    dec r18
    brne L1
    rjmp PC+1
RET

```

This is the flowchart of the task 1: