



Computer Technology I

Lab. 1 : How to use the PORTs, Digital input/output, Subroutine call



Authors: LEONARDO PEDRO,
LOÏC GALLAND

Supervisor:

Semester: Autumn 2019

Area: Computer Science

Course code: 1DT301

Contents

1	Task 1	1
2	Task 2	3
3	Task 3	5
4	Task 5	7
5	Task 6	9

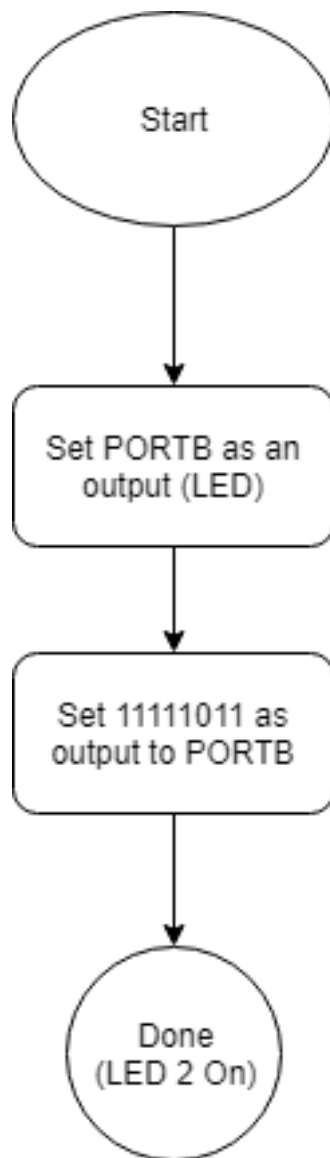
1 Task 1

Write a program in Assembly language to light LED 2. You can use any of the four ports, but start with PORTB. The program should be very short! How many instructions is minimum number?

[illegible]

To be able to light up the LEDs we need 4 lines of code. The first line is to store into the register r16 the value 0xFF. In the second line the register r16 is loaded to DDRB (Data Direction Register of port B). In the third line the desired binary code is stored into register r16. The binary number will determine which LED will light up. In the last line the register r16 is loaded into the PortB (Data Register of Port B).

This is the flowchart of the task 1:

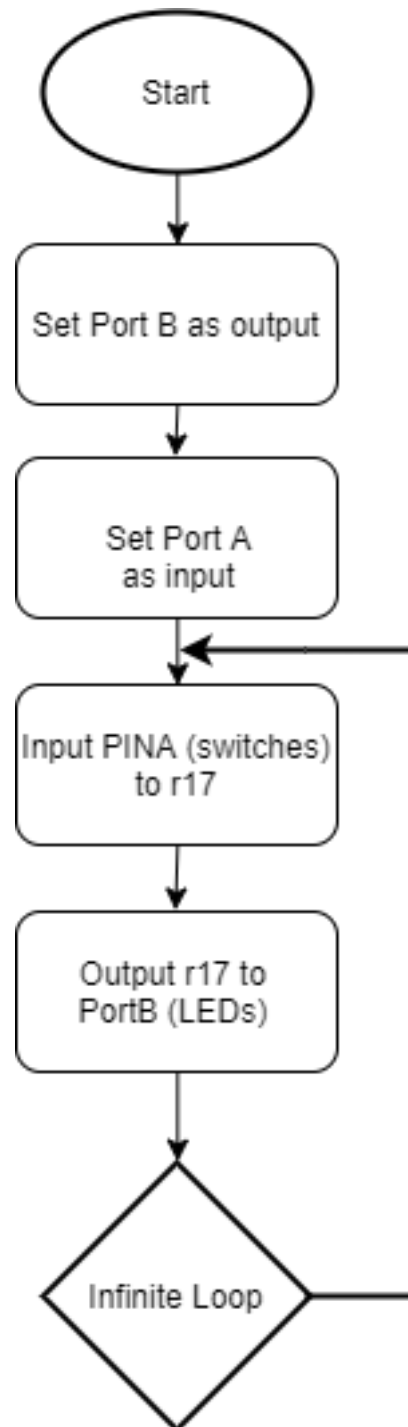


2 Task 2

Write a program in Assembly language to read the switches and light the corresponding LED. Example: When you press SW5, LED5 so should light. Make an initialization part of the program and after that an infinite loop.

[illegible]

This is the flowchart for Task 2:



3 Task 3

Write a program in Assembly language to read the switches and light LED0 when you press SW5. For all other switches there should be no activity.

```
; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 1DT301, Computer Technology I
; Date: 2019-09-09
; Author:
; Loic GALLAND
; Leonardo PEDRO
;
; Lab number: 1
; Title: How to use the PORTs. Digital input/output. Subroutine call.
;
; Hardware: STK600, CPU ATmega2560
;
; Function: Program to only light up LED number 0 if the switch number
           5 is pressed. If any other switch is pressed, nothing will happen
;
; Input ports: The Port A will be used as an input port in this Task
;
; Output ports: The portB is used as an output port
;
; Subroutines: If applicable.
; Included files: m2560def.inc
;
; Other information:
;
; Changes in program: (Description and date)
;
ldi r16, 0xFF          ;Setting up the data direction for Port B
out DDRB, r16          ;Set port B as output

ldi r16, 0x00          ;Setting up the data direction for Port A
out DDRA, r16          ;Set Port A as output

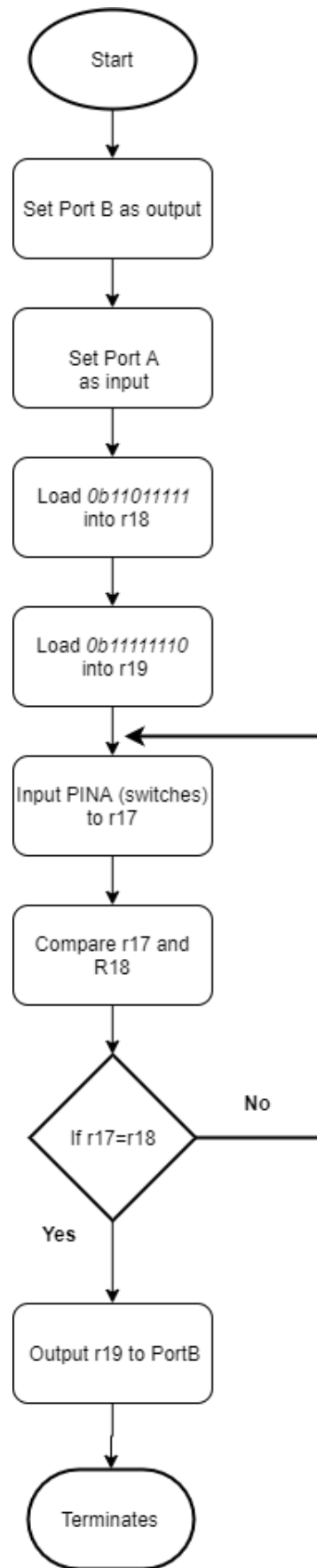
ldi r16, 0xFF          ;Turn off all the LEDs
out portB, r16

ldi r18, 0b11011111    ;Desired binary code for SWITCH number 5
ldi r19, 0b11111110    ;Binary code to light up LED0

my_loop:
    in r17, PINA        ;get the info from the switch
    cp r17, r18         ;compare switch info with desired one
    breq light          ;condition if r17=r18 go to the "light"
rjmp my_loop

light: out portB,r19     ;turns on the LED0
```

This is the flowchart for Task 3:

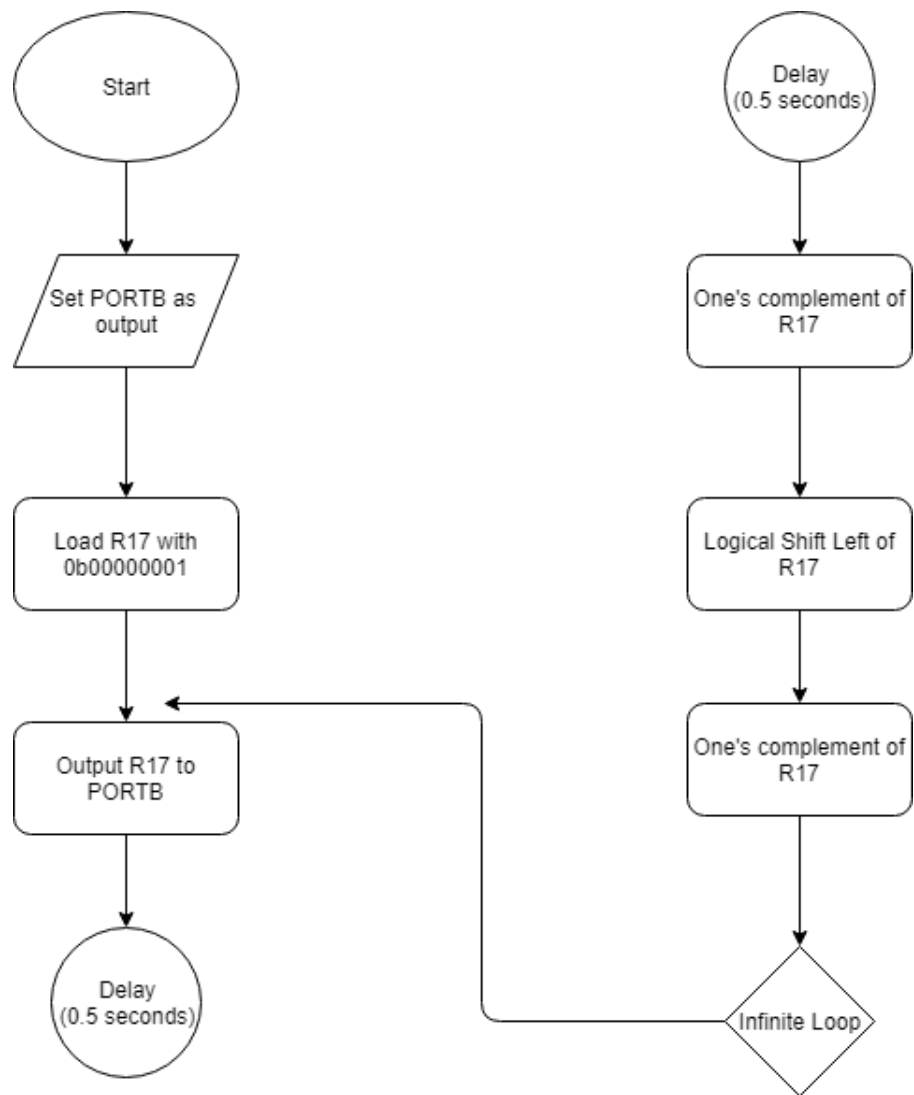


4 Task 5

Write a program in Assembly language that creates a Ring Counter. The values should be displayed with the LEDs. Use shift instructions, LSL or LSR. Make a delay of approximately 0.5 sec in between each count. Write the delay as a subroutine.

[illegible]

This is the flowchart of the task 5:



5 Task 6

Write a program in Assembly language that creates a Johnson Counter in an infinite loop.

[illegible]

```

light:
    out portB, r23 ;output r23 to portB
    CALL Delay ;Delay of 0.5s
    ldi r21, 0b10000000 ;initialize the LED to make it go right
    out portB, r21
    Second_loop:
        out portB, r21 ;output r21 to portB
        ASR r21 ;Shift all the bits to the right so here it
            would go from 0b1000 0000 to 0b1100 000 and so on
        CALL Delay ;we call the delay
        cp r21, r22 ;Compare the current status to
            know if it needs to start going the other way
            ;So for the first one we compare 0b1100
            0000 with 0b1111 1111
        breq my_loop ; if r21 == r22 which means all lights are
            turned off we go back to my_loop
        rjmp Second_loop ;if breq is not true then we repeat the loop
            of Second_loop

Delay:
;Generated by delay loop calculator
;at http://www.bretmulvey.com/avrdelay.html
;
;Delay 4 050 000 cycles
;500ms at 8.1 MHz

    ldi r18, 21
    ldi r19, 140
    ldi r20, 174
L1:    dec r20
    brne L1
    dec r19
    brne L1
    dec r18
    brne L1
    rjmp PC+1
RET

```

This is the flowchart of the task 6:

