

## Ζήτημα 1.1

Ο κώδικας assembly που παράγει ρυθμιζόμενες χρονικές καθυστερήσεις με ακρίβεια (π.χ. για 10msec):

```
.include "m328Pbdef.inc"

.equ FOSC_MHZ=16    ;MHz
.equ DEL_mS=10     ;mS
.equ F1=FOSC_MHZ*DEL_mS

reset:
    ldi r24,low(RAMEND)
    out SPL,r24
    ldi r24,high(RAMEND)
    out SPH,r24

    ser r24
    out DDRD,r24
    out PORTD,r24

    clr r26

loop1:
    ldi r24, low(F1-1)    ;F1-1 - we skip one delay_inner loop
    ldi r25, high(F1-1)  ;to modify number of cycles
    rcall delay0         ;and call delay0
    com r26
    out PORTD,r26
    rjmp loop1

delay0:
    ldi r23, 248         ;we add two more "nop" instructions
    nop                 ;and decrease the value of r23
    nop                 ;(responsible for ms-loop) by 1
loop0_in:
    dec r23
    nop
    brne loop0_in

delay_inner:
    ldi r23, 249         ; (1 cycle)
loop_inn:
```

## 1<sup>η</sup> εργαστηριακή άσκηση

```
dec r23           ; 1 cycle
nop               ; 1 cycle
brne loop_inn     ; 1 or 2 cycles

sbiw r24 ,1       ; 2 cycles
brne delay_inner  ; 1 or 2 cycles

ret               ;4 cycles
```

## 1<sup>η</sup> εργαστηριακή άσκηση

### Ζήτημα 1.2

Ο κώδικας assembly για τον υπολογισμό των λογικών συναρτήσεων  $F0 = (A' \cdot B' \cdot C' + D)'$   
 $F1 = (A' + C) \cdot (B' + D)'$  είναι ο ακόλουθος:

```
.include "m328PBdef.inc"

.def A = r16
.def B = r17
.def C = r18
.def D = r19
.def T = r20
.def F0 = r21
.def F1 = r22
.def counter = r23
.def step = r24

init:
    ldi r25, low(RAMEND)
    out SPL, r25
    ldi r25, high(RAMEND)
    out SPH, r25

    ldi A, 0x45
    ldi B, 0x23
    ldi C, 0x21
    ldi D, 0x01
    ldi step, 0x05
    ldi counter, 5

loop:
    clr T                ;clear the temporary, F0 and F1
    clr F0
    clr F1

    com A
    mov F0, A            ;F0 = A'
    mov F1, A            ;F1 = A'
    com A

    com B
    and F0, B            ;F0 = A'*B'
    mov T, B             ;T = B'
    com B

    com C
```

## 1<sup>η</sup> εργαστηριακή άσκηση

```
and F0, C      ;F0 = A'*B'*C'
com C
or F1, C        ;F1 = (A' + C)

or F0, D        ;F0 = (A'*B'*C' + D)
com D
or T, D         ;T = (B' + D')
com D

com F0          ;F0 = F0'
and F1, T       ;F1 = (A' + C)*(B' + D')

ldi T, 0x04     ;use as few registers as possible
inc A
inc B
inc B           ;A and B are increased with "inc"
add C, T        ;C and D are increased with "add"
add D, step     ;because that way less instructions are needed

dec counter
brne loop
```

Έπειτα από προσομοίωση στο MPLAB X, ο πίνακας που προκύπτει για τις τιμές των μεταβλητών A, B, C, D και των αποτελεσμάτων F0 και F1 για τις 5 επαναλήψεις του προγράμματος είναι ο παρακάτω:

A	B	C	D	F0	F1
0x45	0x23	0x21	0x01	0x66	0xBA
0x46	0x25	0x25	0x06	0x61	0xB9
0x47	0x27	0x29	0x0B	0x64	0xB8
0x48	0x29	0x2D	0x10	0x6D	0xBF
0x49	0x2B	0x31	0x15	0x6A	0xB6

## 1<sup>η</sup> εργαστηριακή άσκηση

### Ζήτημα 1.3

Ο κώδικας assembly για τον έλεγχο ενός βαγονέτου που κινείται συνεχώς, αρχικά από δεξιά προς τα αριστερά και στη συνέχεια αντίστροφα:

```
.include "m328PBdef.inc"

.equ F1 = 1*1500          ;1 MHz * 1500 ms
.equ F2 = 1*2000          ;1 MHz * 2000 ms

init:
    ldi r25,low(RAMEND)
    out SPL,r25
    ldi r25,high(RAMEND)
    out SPH,r25

    ser r26
    out DDRD,r26
    out PORTD,r26          ;Port D is output

    ldi r21, 0b00000001    ;initialize LED - starting from the right
    bst r21, 7             ;store MSB of led configuration
    out PORTD, r21
    call delayone

                                ;flag set to 1 indicates direction should be changed

goleft:
    lsl r21                ;rotate left
    bst r21, 7             ;update flag
    out PORTD, r21
    call delayone
    brtc goleft            ;while flag is not set, repeat
    call delaytwo          ;extra delay of two seconds
                                ;change direction

goright:
    lsr r21                ;rotate right
    bst r21, 0             ;update flag
    out PORTD, r21
    call delayone
    brtc goright           ;while flag is not set, repeat
    call delaytwo          ;extra delay of two seconds
    jmp goleft             ;change direction
```

## 1<sup>η</sup> εργαστηριακή άσκηση

```
;delay of 2 sec
delaytwo:
    ldi r24, low(F2-1)
    ldi r25, high(F2-1)
    call delay0
    ret

;delay of 1,5 sec
delayone:
    ldi r24, low(F1-1)
    ldi r25, high(F1-1)
    call delay0
    ret

delay0:
    ldi r23, 248
    nop
    nop
loop0_in:
    dec r23
    nop
    brne loop0_in

delay_inner:
    ldi r23, 249                ; (1 cycle)
loop_inn:
    dec r23                    ; 1 cycle
    nop                        ; 1 cycle
    brne loop_inn              ; 1 or 2 cycles

    sbiw r24 ,1                ; 2 cycles
    brne delay_inner           ; 1 or 2 cycles

    ret                        ;4 cycles
```