

Εργαστήριο Μικροϋπολογιστών: 1^η Εργαστηριακή Άσκηση (2022)

Ομάδα 64

Ιωάννου Κωνσταντίνος AM:031-19840

Μυτιληναίος Γιώργος AM:031-19841

Ζήτημα 1.1

```
1 .include "m328PDef.inc"
2
3 reset:
4     ldi r24,low(RAMEND)
5     out SPL,r24
6     ldi r24,high(RAMEND)
7     out SPH,r24
8
9     .equ msec= 845 // Εδώ δίνουμε πόσα ms θέλουμε να διαρκεί το πρόγραμμα μας
10
11 main:
12     rcall waitx
13     rjmp main
14
15
16
17
18 waitx:
19     nop
20     ldi r24,low(msec) //1 usec
21     ldi r25,high(msec) //1 usec
22     sbiw r24,1 //2 usec
23     breq LOOP // 1 or 2 , Αν θέλουμε 1 ms delay πάμε σε συγκεκριμένη περίπτωση
24     jmp wait11 //2 usec
25
26
27 loop2: // επαναλαμβάνεται όσα ms ζητάμε μείον μια φορά
28     rcall wait1m //3 usec +....
29     sbiw r24,1 //2 usec
30     brne loop2 //1 or 2
31     ret // 4 usec
```

```
32
33 wait4:
34     ret
35
36 wait1m: // for loop2 delay
37     ldi r26,98
38     loop1:
39         rcall wait4
40         dec r26
41         brne loop1
42     nop
43     nop
44     nop
45     nop
46     nop
47     nop
48     nop
49     nop
50     nop
51     ret
52
53 ; 10usec + 4usec*loop + 1 msec * loop -> 10 usec +1msec*loop αλλάζοντας την w
54 wait11: //987 usec ... for DC delay
55     ldi r26,98
56     loop11:
57         rcall wait4
58         dec r26
59         brne loop11
60     jmp loop2
```

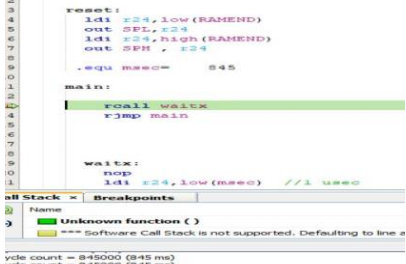
Ξεκινώντας αρχικοποιούμε την στοίβα ώστε να μην υπάρχει θέμα με τους καταχωρητές που χρησιμοποιούμε αν καλέσουμε τις εντολές ret,por,push .

Η waitx αφού φορτώσει την τιμή (εστω X) από τους r24 ,r25 πηγαίνει στην ρουτίνα wait11 η οποία διαρκεί ακριβώς 1 ms μείον τις καθυστερήσεις που έχουν δημιουργηθεί μέχρι αυτήν την στιγμή λόγω των εντολών .

Στην συνέχεια πηγαίνουμε στην ρουτίνα loop2 η οποία εκτελείται X-1 φορές όπου σε κάθε επανάληψη καλεί την ρουτίνα wait4 η οποία διαρκεί 1ms μείον τις καθυστερήσεις των εντολών στην loop2.

Έτσι ζητάμε X msec και συμπεριλαμβάνοντας τις καθυστερήσεις τόσο εκτός loop2 όσο και μέσα στο loop2 παίρνουμε ακριβώς X msec καθυστέρηση .

Για X = 845



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

.equ msec= 845

main:
    rcall waitx
    rjmp main

waitx:
    nop
    ldi r24,low(msec) //1 usec
    ldi r25,high(msec) //1 usec
    sbiw r24,1 //2 usec
    breq LOOP // 1 or 2 , Αν θέλουμε 1 ms delay πάμε σε συγκεκριμένη περίπτωση
    jmp wait11 //2 usec

loop2: // επαναλαμβάνεται όσα ms ζητάμε μείον μια φορά
    rcall wait1m //3 usec +....
    sbiw r24,1 //2 usec
    brne loop2 //1 or 2
    ret // 4 usec

wait4:
    ret

wait1m: // for loop2 delay
    ldi r26,98
    loop1:
        rcall wait4
        dec r26
        brne loop1
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    ret

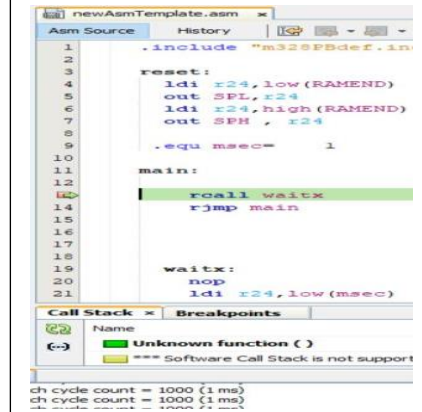
; 10usec + 4usec*loop + 1 msec * loop -> 10 usec +1msec*loop αλλάζοντας την w
wait11: //987 usec ... for DC delay
    ldi r26,98
    loop11:
        rcall wait4
        dec r26
        brne loop11
    jmp loop2
```

```

61
62
63 LOOP: //for 1 ms only
64     wait: // for loop2 delay
65     ldi r26,98
66     loop12: //
67     rcall wait4
68     dec r26
69     brne loop12
70
71     nop
72     nop
73     nop
74     nop
75     nop
76     nop
77     jmp main
78

```

Τέλος όταν θέλουμε απλώς να έχουμε καθυστέρηση 1ms καλούμε wait η οποία διαρκεί 1ms μείον τις καθυστερήσεις που έχουν δημιουργηθεί μέχρι αυτή την στιγμή.



Ζήτημα 1.2

```

1 ;F0 = (A' * B' + B'*D )'
2 ;F1 = (A+C) * (B+D')
3
4 .include "m328PBdef.inc"
5
6 .DEF A = r16
7 .DEF AN = r21
8 .DEF B = r17
9 .DEF C = r18
10 .DEF D = r19 ; Δήλωση καταχωρητών
11 .DEF BN = r22
12 .DEF CN = r23
13 .DEF DN = r24
14 .DEF temp = r28
15 .DEF check = r20
16 .DEF F0 = r30
17 .DEF F1 = r29
18
19 ;.cseg
20 ;.org 0 ; διεύθυνση εκκίνησης
21 .equ loops = 6
22
23
24 ldi check , loops
25
26 ;Αρχικές τιμές
27 ldi A, 0x55
28 ldi B, 0x43
29 ldi C, 0x22
30 ldi D, 0x02

```

```

31
32
33 Loop:
34 ;Δημιουργώ τα συμπληρώματα
35 mov AN,A
36 com AN
37 mov BN,B
38 com BN
39 mov CN,C
40 com CN
41 mov DN,D
42 com DN
43
44 ;F0 = (A' * B' + B'*D )'
45 clr F0
46 mov F0,AN
47 and F0,BN
48 mov temp,BN
49 and temp,D
50 or F0,temp
51 com F0
52
53 ;F1 = (A+C) * (B+D')
54 clr F1
55 mov F1,A
56 or F1,C
57 mov temp,B
58 or temp,DN
59 and F1,temp
60
61 DEC check
62 cpi check , 0
63 breq EXIT ; skip if check != 0

```

```

63
64   subi A , -0x02
65   subi B , -0x03
66   subi C , -0x04
67   subi D , -0x05
68
69   jmp Loop
70
71   EXIT:

```

Βάζουμε ένα brake point για κάθε επανάληψη :

A	B	C	D	F0	F1
0x55	0x43	0x22	0x02	0x57	0x77
0x57	0x46	0x26	0x07	0x56	0x76
0x59	0x49	0x2A	0x0C	0x59	0x7B
0x5B	0x4C	0x2E	0x11	0x4E	0x6E
0x5D	0x4F	0x32	0x16	0x4F	0x6F
0x5F	0x52	0x36	0x1B	0x56	0x76

Ζήτημα 1.3

```

7   reset:
8       ldi r24 , low(RAMEND)      ; initialize stack pointer
9       out SPL , r24
10      ldi r24 , high(RAMEND)
11      out SPH , r24
12      ser r24                    ; initialize PORTD for output
13      out DDSD , r24
14      ldi r26 , 1                ; set r26 as 00000001, output register
15      ser r27                    ; direction register (0:left or 1:right) [T
16
17   main:
18       out PORTD , r26 ; show leds
19       rcall WAIT
20
21       cpi r27 , 0
22       breq sleft ;jump if left else continue
23
24       set ;T flag 1
25       lsr r26
26       rjmp main
27
28   sleft:
29       clt ; T flag 0
30       lsl r26
31       rjmp main
32
33   check_boundaries:
34       cpi r26 , 0b00000001
35       breq change_direction
36       cpi r26 , 0b10000000
37       breq change_direction
38       ret
39
40   change_direction:
41       com r27
42       jmp wait15
43       jmp main
44
45
46   WAIT:
47       rcall check_boundaries
48       jmp wait05
49

```

Με το **set** ορίζουμε T Flag ως 1 όταν έχει κατεύθυνση προς τα δεξιά ενώ αντίστοιχα με την εντολή **clt** ορίζουμε T flag ως 0 όταν το led έχει κατεύθυνση προς τα αριστερά.

Χρησιμοποιούμε την waitx του ζητήματος 1.1 με ορίσματα 500 και 1500 ώστε να έχουμε delay περίπου 0.5 sec και 1.5 sec.

Δίνουμε αρχική τιμή 00000001 στον r26 και στην συνέχεια με lsr μετακινούμαστε ένα bit δεξιά με το κατάλληλο delay ενδιάμεσα(ομοίως με lsl για κατεύθυνση προς τα αριστερά) .Κάθε φορά γίνεται έλεγχος αν βρισκόμαστε το MSB ή στο LSB ώστε να αλλάξει η κατεύθυνση και ταυτόχρονα να αυξηθεί το delay σε 1.5 sec για τα άκρα.