



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΜ&ΜΥ
Εργαστήριο Μικροϋπολογιστών

6^η Εργαστηριακή Άσκηση
Ακ. έτος 2011-2012

Ομάδα C07:

Ελένη Ευαγγελάτου	A.M.: 03108050
Γρηγόρης Λύρας	A.M.: 03109687
Βασιλεία Φραγκιαδάκη	A.M.: 03108026

5 Φεβρουαρίου 2012

Άσκηση (i)

Στην άσκηση αυτή ζητείται στο χρονόμετρο που απεικονίζεται στην PORTA και υλοποιείται με ταχύτητα 100 msec ανά μέτρηση, όταν ενεργοποιείται εξωτερική διακοπή INT0, αν το PD0 είναι μηδέν, να απαριθμεί τις διακοπές και να εμφανίζει το πλήθος τους στην PORTB. Γι' αυτό το σκοπό ενεργοποιούμε με τις κατάλληλες εντολές τις διακοπές στο κυρίως πρόγραμμα και αρχικά βάζουμε το .org 0x002 που επιτρέπει την εξυπηρέτηση της διακοπής INT0 στον Atmega16.

Κυρίως κώδικας:

```
1 ;
2 ;part1.asm
3 ;
4 ; Created: 29/1/2012 11:50:22 ??
5 ; Author: Valia
6 ;
7
8 ;.include "m16def.inc"
9 .def temp1 = r24
10 .def temp2 = r25
11 .def metrhths = r26
12 .def check_PD0 = r22
13 .def state = r27
14 .def metr_diakopwn = r23
15
16 .org 0x000
17 rjmp main
18 .org 0x002
19     rjmp ISRO
20     reti
21
22 main:
23     ldi temp1,high(RAMEND)
24     out SPH,temp1 ; arxikopoihsh ths stoivas
25     ldi temp1,low(RAMEND)
26     out SPL,temp1
27
28     ldi temp1,(1<<ISC01) | (1 << ISC00) ;energopoihsh ths diakophs INTO
29     out MCUCR,temp1
30     ldi temp1,(1 <<INT0)
31     out GICR ,temp1
32     sei
33     clr temp1
34
35     ser metrhths
36     out DDRA,metrhths ;PORTA exodos
37
38     clr metrhths
39     out DDRD,metrhths ;PORTD eisodos
40
41 loop:
42     out PORTA,metrhths
43
44     ldi temp1,low(100)
45     ldi temp2,high(100) ;ka8ysterhsh 100 msec
46     rcall wait_msec
47     inc metrhths
48     rjmp loop
49
50 ISRO:
51     push metrhths
52     in state,SREG ;swzw thn katastash:D
53     push state
54     push check_PD0
55
56     in check_PD0,PIND ;elegxos PDO
57     ror check_PD0
58     brcs exit ;an PRDO = 0 metraei th diakoph
59
60     ser metrhths
61     out DDRB,metrhths ;exodos h PORTB
62
```

```

63         inc metr_diakopwn          ;auwanoume t metrthh
64
65         out PORTB, metr_diakopwn    ;emfanish tun diakopwn
66         ldi temp1, low(1000)
67         ldi temp2, high(1000)
68         ;rcall wait_msec
69
70     exit:
71         out SREG, state
72         pop check_PDO
73         pop state
74         pop metrthhs
75         reti
76
77     ;routines pou pragmatopoioun ka8ysterhsh tosa msec, oso to periewomeno tun r25-r24
78     wait_msec:
79         push temp1
80         push temp2
81         ldi temp1, low(998)
82         ldi temp2, high(998)
83         rcall wait_usec
84         pop temp2
85         pop temp1
86         sbiw temp1, 1
87         brne wait_msec
88
89         ret
90
91     wait_usec:
92         sbiw temp1, 1
93         nop
94         nop
95         nop
96         nop
97         brne wait_usec
98
99         ret

```

Άσκηση (ii)

Σε αυτό το μέρος πραγματοποιήσαμε ένα πρόγραμμα μέτρησης το οποίο διακόπτεται μόνο από την διακοπή INT1 (push button button PD3). Το πρόγραμμα μέτρησης εμφανίζεται στην σειρά led PA. Όταν έχουμε διακοπή πρέπει να διαβάζουμε από τα dip switches B και να μετράμε πόσα είναι ON. Το πόσα είναι ON φαίνεται στα leds PC. Μετά την διακοπή συνεχίζεται κανονικά η μέτρηση στα leds PA. Το πρόγραμμα μέτρησης είναι ακριβώς όπως το δεθέν της εκφώνησης. Αυτή την φορά απλά αναγνωρίζονται οι διακοπές INT1. Ο κώδικας εξυπηρέτησης της διακοπής βρίσκεται στην label ISR1. Το πόσα dip switches είναι ON υπολογίζεται ως εξής: διαβάζουμε από τον PINB, και κάνουμε shift δεξιά. Αν έχουμε carry, τότε αυξάνουμε κατά 1 το άθροισμα των switches που είναι on. Διαφορετικά ελέγχουμε το επόμενο ψηφίο. Αυτό γίνεται και για τα 8 bits. Στο τέλος, για ασφάλεια ώστε να έχουμε το αποτέλεσμα στα 4 least significant bits, κάνουμε μια bitwise λογική πράξη and με τον αριθμό 00001111.

Έπειτα δείχνουμε το αποτέλεσμα στα leds PC. Ο κώδικας μας:

Κυρίως κώδικας:

```

1     ;
2     ; part2.asm
3     ;
4     ; Created: 31/1/2012 7:58:21 ??
5     ; Author: Eleni
6     ;
7
8     .org 0x00
9         rjmp main
10    .org 0x04
11        rjmp ISR1
12        reti
13
14
15
16    main:
17        ldi r30, low(RAMEND)
18        out SPL, r30
19        ldi r30, high(RAMEND)
20        out SPH, r30

```

```

21
22      ;programma metrhshs opws dinetai
23      ser r26
24      out DDRA, r26
25      clr r26
26      ;====gia energopoihsh ths INT1 =====
27
28      ldi r24, (1<< ISC11)|(1 << ISC10) ;na prokaleitai me shma 8etikhs akmhsh
29      out MCUCR, r24
30      ldi r24, (1<< INT1)
31      out GICR, r24
32      sei
33
34
35      loop:
36          out PORTA, r26
37          ldi r24, low(100)
38          ldi r25, high(100)
39
40          rcall wait_msec
41          inc r26
42          rjmp loop
43
44      ;====rutina eksuphrethshs diakophs=====
45
46      ISR1:
47          push r26 ;swzei to periezomeno tou r26
48          in r26, SREG ;kai SREG
49          push r26
50
51          clr r27 ;clear ton r27 giati ekei 8a mazepsw to sum mou
52
53          clr r25
54          out DDRB, r25 ;den eimai sigourh gia ton r25
55          in r29, PINB
56
57          mov r28, r29 ;mhpus tuxon xreiatei
58
59      bit0:
60          lsr r28
61          brcs plus_one_0 ;an eww carry dhlash asso tote pros8etw ena sto sum dhladh ston register r27
62
63      bit1:
64          lsr r28
65          brcs plus_one_1
66
67      bit2:
68          lsr r28
69          brcs plus_one_2
70
71      bit3:
72          lsr r28
73          brcs plus_one_3
74
75      bit4:
76          lsr r28
77          brcs plus_one_4
78
79      bit5:
80          lsr r28
81          brcs plus_one_5
82
83      bit6:
84          lsr r28
85          brcs plus_one_6
86
87      bit7:
88          lsr r28
89          brcs plus_one_7
90          rjmp go_on
91
92      ;auksanei kata 1
93      plus_one_0:
94          inc r27
95          rjmp bit1
96
97      plus_one_1:
98          inc r27
99          rjmp bit2
100
101      plus_one_2:

```

```

96         inc r27
97         rjmp bit3
98
99 plus_one_3:
100        inc r27
101        rjmp bit4
102
103 plus_one_4:
104        inc r27
105        rjmp bit4
106
107 plus_one_5:
108        inc r27
109        rjmp bit6
110
111 plus_one_6:
112        inc r27
113        rjmp bit7
114
115 plus_one_7:
116        inc r27
117
118
119
120 ;telos plus_one
121
122 ;kanw AND me 0000 1111
123 ;set bit mask ston r16 gia na exw ari8mo mono sta 4 lsb, gia asfaleia , mporei k na mhn xreiazetai
124 go_on:
125        ldi r16, 0x0F
126        and r27, r16 ;bitwise and. To apotelesma apo8hkeuetai ston r27
127
128        ser r29
129        out PORTC, r29
130        out DDRC, r27 ;ta deixnw sta leds
131
132        reti
133
134 ;=====wait_msec=====
135 wait_msec:
136        push r24
137        push r25
138        ldi r24, low(998)
139        ldi r25, high(998)
140        rcall wait_usec
141        pop r25
142        pop r24
143        sbiw r24, 1
144        brne wait_msec
145
146        ret
147
148 wait_usec:
149        sbiw r24,1
150        nop
151        nop
152        nop
153        nop
154        brne wait_usec
155
156        ret

```

Άσκηση (iii)

Κυρίως κώδικας:

```

1 .org 0x000
2     rjmp main
3 .org 0x002
4     rjmp ISR0
5 .org 0x010
6     rjmp TIM1_OVF
7     reti
8
9 main:

```

```

10  reset:
11      ldi r24,high(RAMEND)
12      out SPH,r24
13      ldi r24,low(RAMEND)
14      out SPL,r24
15      ldi r24,2
16      ;#set A1 for output
17      ;#and A0 for input
18      out DDRA,r24
19      ;activate timer1
20      ;2 is TOIE1
21      ;avrStudio doesn't know about it
22      ldi r24,(1 << 2)
23      out TIMSK,r24
24      ldi r24,(1 << CS12)|(0<<CS11)|(1<<CS10)
25      out TCCR1B,r24
26      ;activate INTO
27      ldi r24,(1 << ISC01) | (1 << ISC00)
28      out MCUCR,r24
29      ldi r24,(1 << INTO)
30      out GICR,r24
31      sei
32
33  fun:
34      ;busy wait
35      ;for interrupt
36      in r26,PINA
37      ror r26
38      brcs ON
39      rjmp fun
40  ON:
41      rcall set_timer
42      rcall led_on
43      rjmp fun
44
45
46
47  ISRO:
48      push r26 ;swzei to periezomeno tou r26
49      in r26, SREG ;kai SREG
50      push r26
51      rcall debounce
52      rcall set_timer
53      rcall led_on
54      out SREG,r26
55      pop r26
56      sei
57      reti
58
59  TIM1_OVF:
60      rcall led_off
61      reti
62
63  debounce:
64      push r26
65      push r25
66      push r24
67  deb_loop:
68      ldi r26,(1<<INTF0)
69      out GIFR,r26
70      in r26,GIFR
71      rol r26
72      rol r26
73      ldi r24,0x05
74      ldi r25,0x00
75      rcall wait_msec
76      brcs deb_loop
77      pop r24
78      pop r25
79      pop r26
80      ret
81
82  set_timer:
83      push r24
84      ldi r24,0xa4

```

```

85     sts TCNT1H,r24
86     ldi r24,0x72
87     sts TCNT1L,r24
88     pop r24
89     ret
90
91     led_on:
92     push r24
93     ldi r24,2
94     out PORTA,r24
95     pop r24
96     ret
97
98     led_off:
99     push r24
100    ldi r24,0
101    out PORTA,r24
102    pop r24
103    ret
104
105    wait_usec:
106    sbiw r24,1
107    nop
108    brne wait_usec
109    ret
110
111    wait_msec:
112    push r24
113    push r25
114    ldi r24,low(998)
115    ldi r25,high(998)
116    rcall wait_usec
117
118    pop r25
119    pop r24
120    sbiw r24,1
121    brne wait_msec
122    ret

```