Άσκηση πέμπτη του Εργαστηρίου

Στην πέμπτη άσκηση εξετάζεται η οδήγηση ενδείκτων επτά τομέων (seven segment displays) με την χρήση πινάκων μετατροπής (Look-up tables).

Οι πίνακες είναι έννοιες που συναντιούνται γενικά στον προγραμματισμό, και χρησιμοποιούνται πολύ στα προγράμματα των μικροελεγκτών. Οι πίνακες είναι ομάδες συναφών δεδομένων . Για παράδειγμα τα τηλεφωνικά νούμερα του καταλόγου του κινητού τηλεφώνου θα μπορούσαν να βρίσκονται όλα σε ένα πίνακα, ενώ τα ονόματα σε κάποιο άλλο πίνακα. Υπάρχουν μονοδιάστατοι πίνακες, δισδιάστατοι, τρισδιάστατοι κοκ. Οι πίνακες κατά κανόνα υλοποιούνται στη μνήμη των συστημάτων. Ειδικότερα στους μικροελεγκτές δημιουργούνται είτε στη μνήμη προγράμματος (ROM) είτε στη μνήμη δεδομένων (RAM). Η διαφορά των δύο πινάκων προκύπτει από τις εγγενείς δυνατότητες της κάθε μνήμης. Ένας πίνακας στη μνήμη προγράμματος έχει στις θέσεις συγκεκριμένες τιμές οι οποίες μπορούν πλέον μόνο να διαβαστούν και όχι να αλλαχτούν κατά τη διάρκεια της εκτέλεσης του προγράμματος. Αντίστοιχα ένας πίνακας υλοποιημένος στη μνήμη RAM μπορεί να μην έχει αρχικές τιμές για τα στοιχεία του, αλλά αυτές είναι δυνατόν να μεταβάλλονται κατά την εκτέλεση του προγράμματος. Παράδειγμα ενός μονοδιάστατου πίνακα με μεταβλητή των 8 bits δίδεται παρακάτω.

```
int8 Leds[8] = {1,2,4,8,16,32,64,128};
```

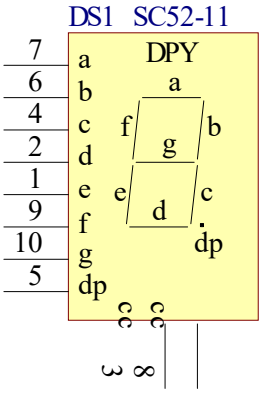
Ένα παράδειγμα πίνακα τεσσάρων διαστάσεων με μεταβλητή των 8 bits δίδεται παρακάτω.

```
int8 4D_table[4] [4] = {{01,02,03,04},
                        {05, 06,07,08},
                        {09,10,11,12},
                        {13,14,15,16}};
```

Ενδείκτες επτά τομέων (Seven Segment Displays)

Οι ενδείκτες επτά τομέων είναι ένα από τα πολύ χρησιμοποιούμενα περιφερειακά των μικροελεκτών και για την οδήγησή τους απαιτείται η χρήση των πινάκων μετατροπής. Οι ενδείκτες μπορεί να είναι ένας ή περισσότεροι και για τη περίπτωση των περισσοτέρων συνδεσμολογούνται σε διάταξη μήτρας για εξοικονόμηση γραμμών I/O.

Οι ενδείκτες 7 τομέων (seven segment displays), είναι συσκευές που έχουν πάνω τους LEDs με τέτοια διάταξη, ώστε να μπορούν να σχηματίζουν αριθμητικά νούμερα από 0 ως 9. Στο σχήμα που ακολουθεί δίνεται το σχηματικό διάγραμμα ενός ενδείκτη 7 τομέων.



Στον ενδείκτη αυτό υπάρχουν 8 LEDs, από τα οποία τα 7 είναι διατεταγμένα στο σχήμα **8** και ένα είναι η τελεία. Τα LEDs αυτά έχουν ονόματα όπως φαίνεται στο σχήμα. Τα ονόματα είναι a, b, c, d, e, f, g και DP. Οι ενδείκτες έχουν 10 pins από τα οποία τα 8 συνδέονται στα LEDs και τα υπόλοιπα 2 αποτελούν το κοινό ακροδέκτη. Συγκεκριμένα όπως φαίνεται τα pins 1, 2, 4, 5, 6, 7, 9, 10 συνδέονται αντίστοιχα με τα τμήματα των LEDs e, d, c, DP, b, a, f, g.

Ανάλογα με το κοινό ακροδέκτη οι ενδείκτες χωρίζονται σε δύο κατηγορίες.

1. Στους Ενδείκτες κοινής καθόδου
2. Στους Ενδείκτες κοινής ανόδου

Η λειτουργία της κάθε κατηγορίας περιγράφεται παρακάτω:

Στους ενδείκτες κοινής καθόδου οι δύο κοινοί ακροδέκτες (pins 3, 8) συνδέονται στη γείωση του κυκλώματος. Για να ανάψει για παράδειγμα το LED a του ενδείκτη πρέπει να δοθεί τάση στο pin 7, ενώ για να ανάψει το LED d του ενδείκτη πρέπει να δοθεί τάση στο pin 2. και παράλληλα γείωση στους κοινούς ακροδέκτες pins 3,8. Δίνοντας τάση στα κατάλληλα pins και ανάβοντας τα αντίστοιχα LEDs μπορούν να σχηματιστούν όλα τα ψηφία πάνω στον ενδείκτη. Έτσι για να σχηματιστεί το νούμερο 2 πρέπει να ανάψουν τα LEDs a, b, g, e, d ή να δοθεί τάση στα pins 7, 6, 10, 1, 2. Αν συνδεθούν τα 8 pins με το PORTB του μικροελεγκτή, τότε μπορεί να δίνει το πρόγραμμα του χρήστη τάσεις στα LEDs γράφοντας λογικό ‘1’ στα pins που χρειάζονται τάση. Παρακάτω συνοψίζονται οι τιμές που πρέπει να έχει το PORTB για να σχηματιστούν όλοι οι αριθμοί στον ενδείκτη.

Χαρακτήρας που σχηματίζεται	**PORTB (pin No.)**								**PORTB (HEX)**
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
	ΕΝΔΕΙΚΤΗΣ (pin No.)								
	5 (DP)	**10 (g)**	**9 (f)**	**1 (e)**	**2 (d)**	**4 (c)**	**6 (b)**	**7 (a)**	
0	‘0’	‘0’	‘1’	‘1’	‘1’	‘1’	‘1’	‘1’	0x3F
1	‘0’	‘0’	‘0’	‘0’	‘0’	‘1’	‘1’	‘0’	0x06
2	‘0’	‘1’	‘0’	‘1’	‘1’	‘0’	‘1’	‘1’	0x5B
3	‘0’	‘1’	‘0’	‘0’	‘1’	‘1’	‘1’	‘1’	0x4F
4	‘0’	‘1’	‘1’	‘0’	‘0’	‘1’	‘1’	‘0’	0x66
5	‘0’	‘1’	‘1’	‘0’	‘1’	‘1’	‘0’	‘1’	0x6D
6	‘0’	‘1’	‘1’	‘1’	‘1’	‘1’	‘0’	‘1’	0x7D
7	‘0’	‘0’	‘0’	‘0’	‘0’	‘1’	‘1’	‘1’	0x07
8	‘0’	‘1’	‘1’	‘1’	‘1’	‘1’	‘1’	‘1’	0x7F

9	‘0’	‘1’	‘1’	‘0’	‘1’	‘1’	‘1’	‘1’	0x6F
A	‘0’	‘1’	‘1’	‘1’	‘0’	‘1’	‘1’	‘1’	0x77
b	‘0’	‘1’	‘1’	‘1’	‘1’	‘1’	‘0’	‘0’	0x7C
c	‘0’	‘0’	‘1’	‘1’	‘1’	‘0’	‘0’	‘1’	0x39
d	‘0’	‘1’	‘0’	‘1’	‘1’	‘1’	‘1’	‘0’	0x5E
e	‘0’	‘1’	‘1’	‘1’	‘1’	‘0’	‘0’	‘1’	0x79
F	‘0’	‘1’	‘1’	‘1’	‘0’	‘0’	‘0’	‘1’	0x71

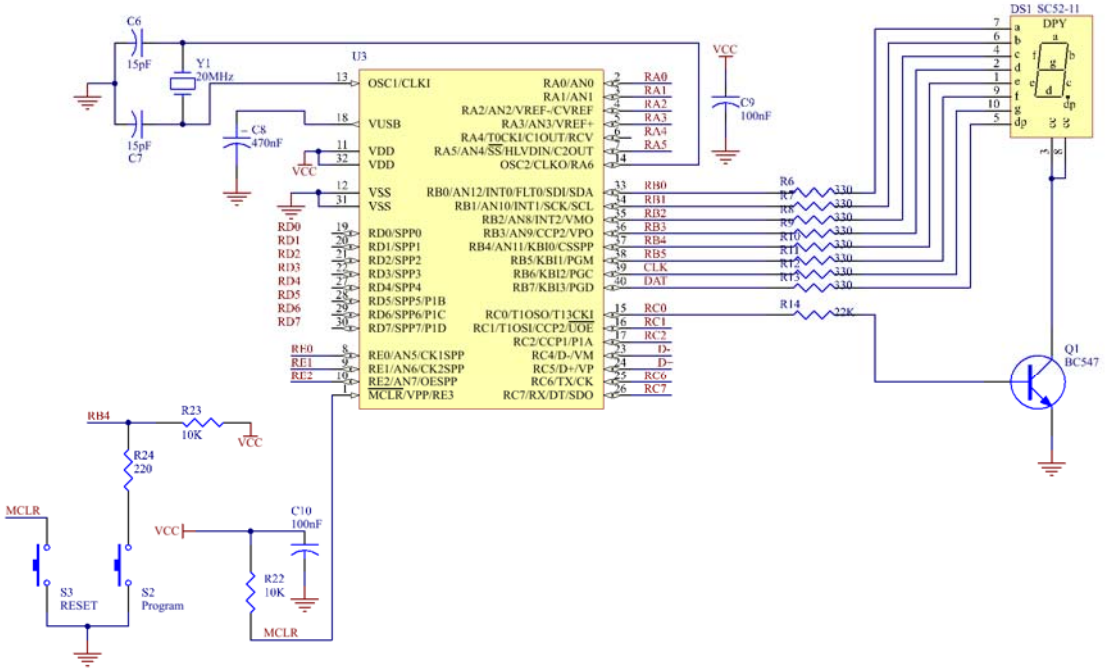
Όπως προκύπτει εύλογα από το παραπάνω πίνακα, αυτή η αντιστοίχηση και οδήγηση των LEDs, με τη βοήθεια του PIC γίνεται εύκολα με τη χρήση ενός πίνακα μετατροπής. Ο πίνακας μετατροπής θα μετατρέπει την αριθμητική τιμή που επιθυμεί ο χρήστης να προβληθεί στον ενδείκτη σε εκείνη τη δεκαεξαδική τιμή (τελευταία στήλη) που όταν τοποθετηθεί στη PORTB ανάβουν τα αντίστοιχα LEDs, σχηματίζοντας τελικά την επιθυμητή ένδειξη.

Αντίστοιχα για τους ενδείκτες κοινής ανόδου, τα κοινά σήματα (pins 3, 8) συνδέονται μόνιμα σε τάση, και τα pins των LEDs γειώνονται για να ανάψουν. Έτσι ο παραπάνω πίνακας θα λειτουργούσε μετατρέποντας απλά όλα τα ‘0’ σε ‘1’ και όλα τα ‘1’ σε ‘0’. Για την οδήγηση των τμημάτων και για το περιορισμό του ρεύματος τοποθετούνται εν σειρά αντιστάσεις που η τιμή τους υπολογίζεται λαμβάνοντας υπόψη τη τάση επαφής του κάθε led και το επιθυμητό ρεύμα λειτουργίας. Ενημερωτικά αναφέρεται ότι η τάση επαφής για τα κόκκινα leds είναι 1.8V για τα πράσινα και πορτοκαλί η τάση είναι 2.1V και για τα λευκά και μπλε 3V. Οπότε η τιμή της αντίστασης υπολογίζεται από την σχέση R= (5-τάση επαφής)/επιθυμητό ρεύμα. Για παράδειγμα αν οδηγείται ένα led κόκκινο και το επιθυμητό ρεύμα είναι 10mA τότε η τιμή της αντίστασης θα είναι R=(5-1.8)/10=0.32KΩ επιλέγεται η πλησιέστερη τιμή 330Ω. Όταν τα display συνδέονται σε διάταξη μήτρας τότε επειδή κάθε χρονική στιγμή ένα display είναι αναμμένο έχουμε διαμοιρασμό του χρόνου και σε αυτή την περίπτωση η τιμή των αντιστάσεων θα πρέπει να είναι μικρότερη ανάλογα με το πόσα leds είναι συνδεδεμένα. Μια άλλη λύση είναι να οδηγηθούν απευθείας χωρίς αντιστάσεις αρκεί τα leds να μπορούν να αντέξουν τα 20mA που μπορεί να δώσει η πόρτα του PIC. (πόρτα B).

Για να γίνει κατανοητή η οδήγηση ενός ενδείκτη δίδεται παρακάτω το πρόγραμμα οδήγησης. Στο παράδειγμα αυτό η μέτρηση του χρόνου γίνεται με χρονοκαθυστέρηση.

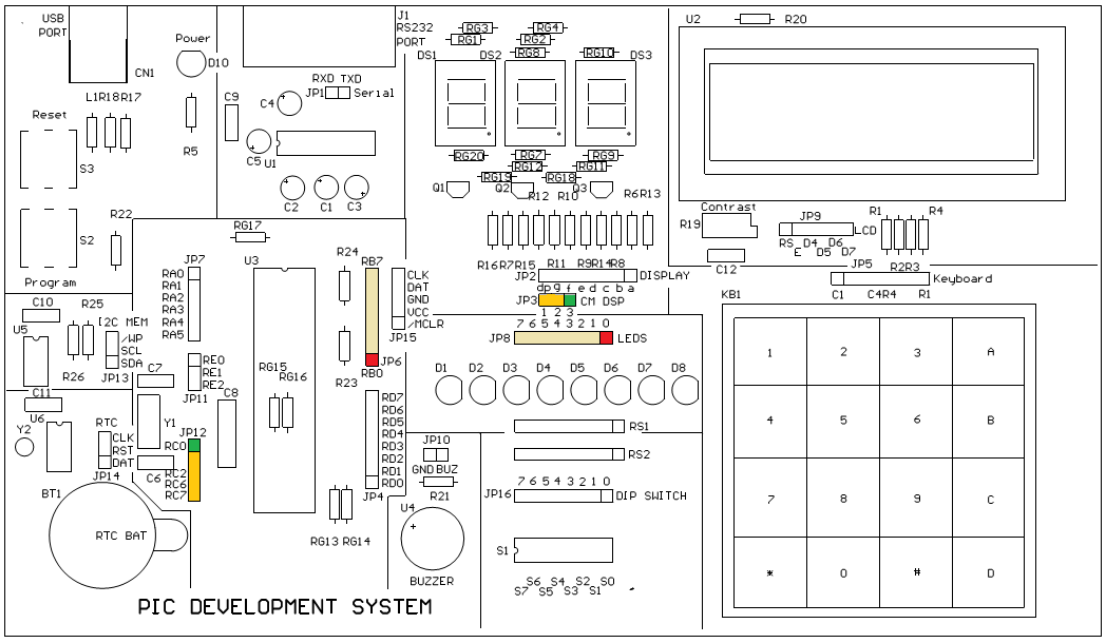
Να γραφεί πρόγραμμα που απαριθμεί σε ενδείκτη 7 τομέων της αναπτυξιακής πλακέτας του εργαστηρίου τους αριθμούς 0 έως 9 με ρυθμό 1 Hz. Υπενθυμίζεται ότι οι ενδείκτες της πλακέτας είναι κοινής καθόδου αλλά οι κάθοδοι συνδέονται μέσω τρανζίστορ στη γείωση γι’ αυτό για να ανάψει ένας ενδείκτης απαιτείται να τεθεί το κοινό του άκρο σε λογικό 1. Τα επτά τμήματα του ενδείκτη συνδέονται στην πόρτα B και το κοινό άκρο στο bit 0 της πόρτας C RC0.

Το σχηματικό του κυκλώματος.



ΑΣΚΗΣΗ 5_1

Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

```
#include <main.h>

#use standard_io ( A )   // Standard είσοδοι και έξοδοι   // Standard είσοδοι και έξοδοι
#use standard_io ( B)
#use standard_io ( C )
#byte PORTA         =0xF80  //ορισμός των θυρών με την θέση τους στην
#byte PORTB         =0xF81
#byte PORTC         =0xF82
#byte PORTD         =0xF83
#byte PORTE         =0xF84
int8 table[16] =  {   0b00111111, // Πίνακας μετατροπής
                      0b00000110,
                     0b01011011,
                     0b01001111,
                     0b01100110,
                     0b01101101,
                     0b01111101,
                     0b00000111,
                     0b01111111,
                     0b01101111,
                     0b01110111,
                     0b01111100,
                     0b00111001,
                     0b01011110,
                     0b01111001,
                     0b01110001};


// Δήλωση συναρτήσεων

void init (void);

// Κύριο πρόγραμμα

void main()
{
    int8  counter;
    init(); // Κλήση της ρουτίνας αρχικοποίησης

    while (1){
                for ( counter=0;counter<10;counter++){ // Μετρητής δευτερολέπτων
                PORTB = table[counter];
                delay_ms(1000);             // Χρονοκαθυστέρηση 1 sec
                }
            }
}

// Ορισμός συναρτήσεων
```

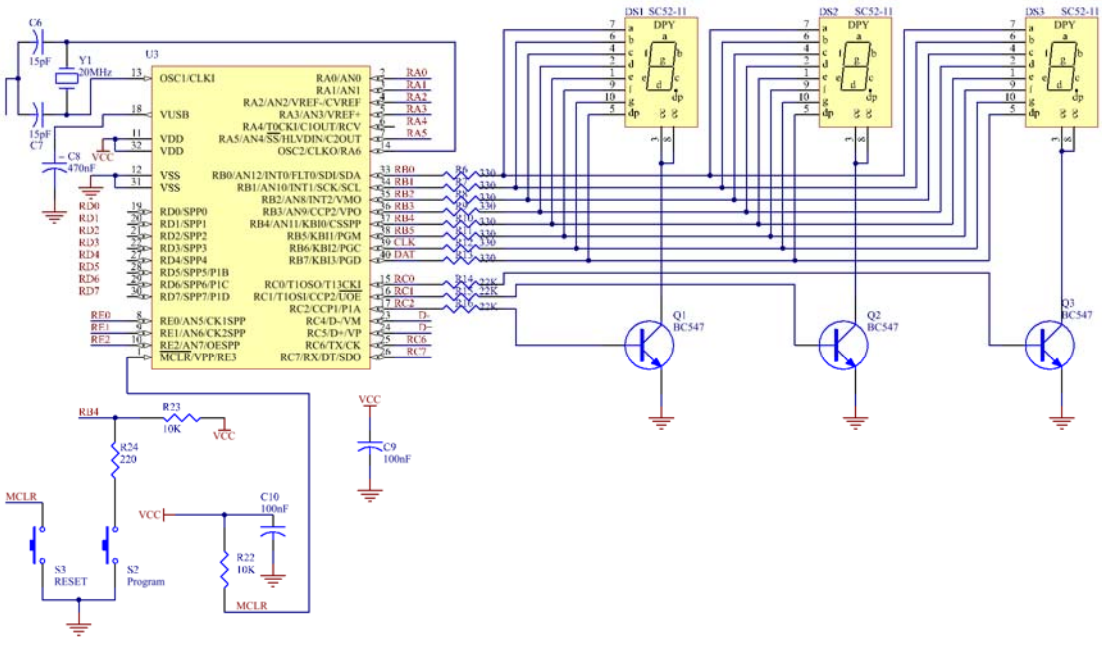
```
void init (void){
                    set_tris_b(0x00);     // Καθορισμός της πόρτας B ως έξοδος
                    set_tris_c(0x00);     // Καθορισμός της πόρτας C ως έξοδος
                    PORTB = 0;
                    PORTC = 1;
            }
```

Παρακάτω δίδεται ένα παράδειγμα της χρήσης των ενδείκτων επτά τομέων οδηγούμενων σε διάταξη μήτρας και η μέτρηση του χρόνου γίνεται με χρήση διακοπής από την υπερχείλιση του timer0.

Να γραφεί πρόγραμμα που να μετρά στους ενδείκτες επτά τομέων της αναπτυξιακής πλακέτας δευτερόλεπτα από την ένδειξη 0 μέχρι την τιμή 999 και μετά να ξεκινά ξανά από το 0. Στην άσκηση θα χρησιμοποιούν η μέθοδος της διακοπής για την μέτρηση του χρόνου και ο πίνακας μετατροπής για την ένδειξη στους ενδείκτες επτά τομέων της αναπτυξιακής πλακέτας.

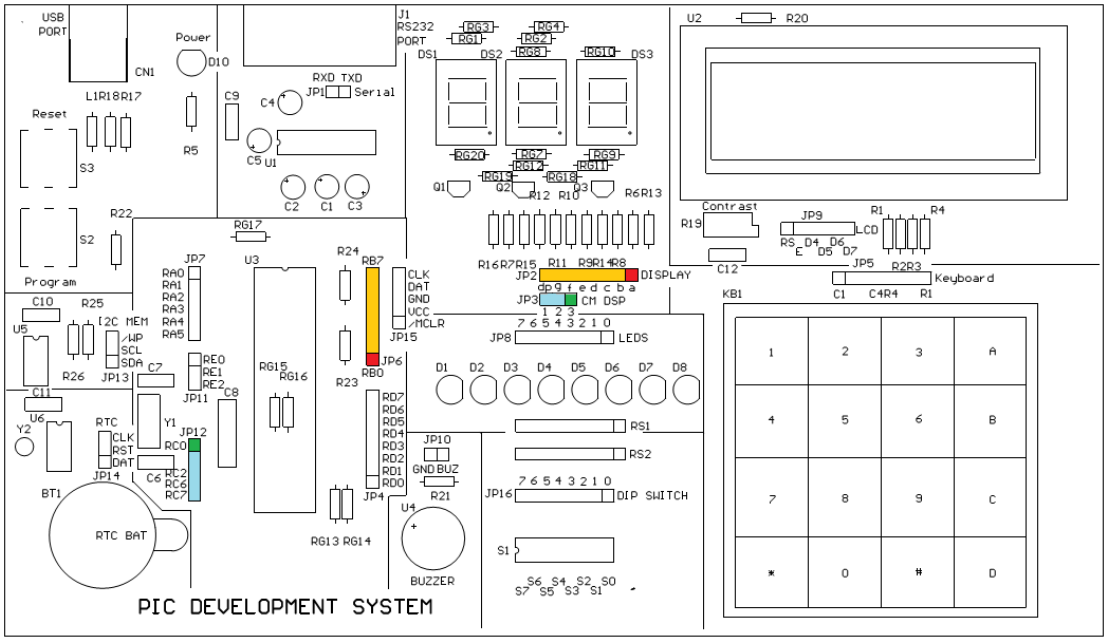
Υπενθυμίζεται ότι η πλακέτα περιέχει ενδείκτες κοινής καθόδου και είναι συνδεσμολογημένοι σε διάταξη μήτρας τα κοινά τους άκρα (οι κάθοδοι των leds) δεν είναι μόνιμα στη γείωση αλλά πηγαίνουν μέσω τρανζίστορς τα οποία είναι συνδεσμολογημένα σε συνδεσμολογία κοινού εκπομπού. Άρα για να ενεργοποιηθεί ένα display απαιτείται η οδήγηση της βάσης του τρανζίστορ με λογικό 1. Οι κοινοί τομείς των displays a,b,c,d,e,f,g,dp οδηγούνται από τη πόρτα B της πλακέτας (RB0,RB1,….RB7) αντίστοιχα. Τα κοινά άκρα οδηγούνται από τα pins της πόρτας C και συγκεκριμένα από το RC0 το λιγότερο σημαντικό display από το RC1 το μεσαίο και από το RC2 το περισσότερο σημαντικό.

Το σχηματικό του κυκλώματος.



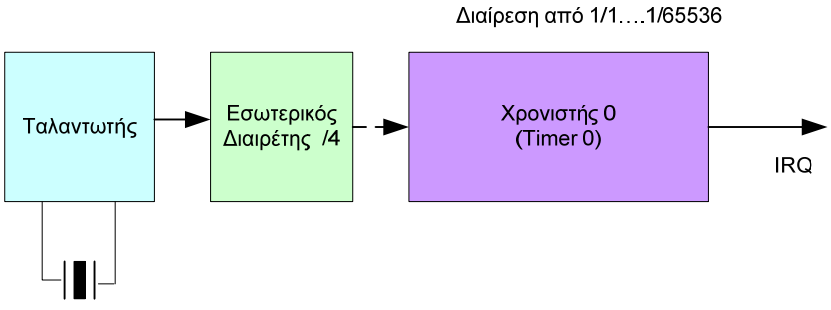
ΑΣΚΗΣΗ 5_2

Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

Για την μέτρηση του χρόνου θα χρησιμοποιηθεί η μέθοδος της διακοπής και επειδή σαν βάση χρόνου θα ληφθεί το δευτερόλεπτο θα πρέπει κάθε δευτερόλεπτο να αλλάζει η ένδειξη. Ωστόσο επειδή κάθε χρονική στιγμή ένας μόνο ενδείκτης είναι ενεργός για να μπορεί το μάτι του παρατηρητή να βλέπει όλους τους ενδείκτες αναμμένους θα πρέπει να ληφθεί χρόνος για το interrupt τέτοιος ώστε, καθώς κάθε φορά θα οδηγείται ένα display για τον χρόνο αυτό, να ξεγελά το μάτι του παρατηρητή. Ο χρόνος αυτός θα πρέπει να είναι τουλάχιστον T=1/ (3*66,66Hz)= 5 msec Δηλαδή συχνότητα εναλλαγής 66,66 Hz. Το μάτι με την συχνότητα αυτή θα τα βλέπει όλα αναμμένα. Καλό είναι στις περιπτώσεις υπολογισμού του χρόνου με την μέθοδο διακοπής να σχεδιάζεται το παρακάτω σχέδιο, που βοηθά στον υπολογισμό της αρχική τιμής του Hardware μετρητή (Timer0).



Για τον υπολογισμό του χρόνου των 5msec χρησιμοποιώντας το παραπάνω σχήμα και έχουμε

20,833333 nsec * 4 * 1 * (65536 – αρχική τιμή) =5000000 nsec => αρχική τιμή = 5536.

Η τιμή της διαίρεσης του Prescaler λαμβάνεται 1 (χωρίς την χρήση του με τον timer0) διότι ο επιθυμητός χρόνος είναι πολύ μικρός (5 msec).

Με όλες τις παραπάνω πληροφορίες μπορεί να γραφεί το πρόγραμμα χρησιμοποιώντας τις έτοιμες συναρτήσεις του C Compiler.

```
#include <main.h>

#use standard_io ( A )   // Standard είσοδοι και έξοδοι
#use standard_io ( B)    // Standard είσοδοι και έξοδοι
#use standard_io ( C )   // Standard είσοδοι και έξοδοι

#byte PORTA        =0xF80  //ορισμός των θυρών με τη θέση τους στη
#byte PORTB        =0xF81  // μνήμη
#byte PORTC        =0xF82
#byte PORTD        =0xF83
#byte PORTE        =0xF84

// Ορισμός των μεταβλητών (Variable definition )

int8  des=0;
int16 seconds=0;
int8 counter;
int8 table[16] =   {  0b00111111,  //Πίνακας μετατροπής για ενδείκτη 7 τομέων
                      0b00000110,  // Από τις 16 τιμές θα χρησιμοποιηθούν στην
                     0b01011011,  // άσκηση αυτή μόνο οι 10 πρώτες
                     0b01001111,
                     0b01100110,
                     0b01101101,
                     0b01111101,
                     0b00000111,
                     0b01111111,
                     0b01101111,
                     0b01110111,
                     0b01111100,
                     0b00111001,
                     0b01011110,
                     0b01111001,
                     0b01110001};

int8 dig[3] = {1,2,4};     // Πίνακας μετατροπής για την επιλογή των ψηφίων

// Δήλωση συναρτήσεων
```

```
void timer0_int(void);
void init (void);

// Κύριο πρόγραμμα  (Main program)

void main()
{
     init();

    while (1){              //  Περιμένει για πάντα  το Interrupt
               }
}

// Ορισμός συναρτήσεων  (Function definition)

// Ρουτίνα εξυπηρέτησης της διακοπής (Interrupt service routine)

#INT_TIMER0 HIGH   // Διακοπή με μεγάλη προτεραιότητα
void timer0_int(void){
        int16 mon,dec,eka;
        counter--; // Κάθε 200 * 5 msec  = 1 sec
        if (counter == 0){
                        seconds++; // Αυξάνει κατά 1  κάθε 1 sec
                        counter = 200;// Αρχική τιμή του counter
                                if (seconds > 999){
                                                seconds = 0;
                                               }
                        }
          eka = (int8) (seconds /100);  // Υπολογισμός εκατοντάδων του μετρητή
          dec = (int8) ((seconds - (100*eka))/10); // των δεκάδων
          mon = (int8) (seconds - (100 * eka) -(10 * dec));// των μονάδων
          set_timer0(5536); // Αρχική τιμή του Hardware μετρητή
          des = ++des%3;  // Modulo 3 μετρητής που λειτουργεί ως
                          //  δείκτης για τον πίνακα επιλογής
                          //  του ψηφίου που δείχνει κάθε στιγμή

                    PORTC = dig[des]; // Επιλογή ενός από τα 3 ψηφία
                    if (des==0){
                                PORTB = table[mon];// Δείξε τις μονάδες
                                }

                    if (des==1){
                                PORTB = table[dec];// Δείξε τις δεκάδες
                                }
                    if (des==2){
                                PORTB = table[eka];// Δείξε τις εκατοντάδες
```

```
                                        }
                        }
// Ρουτίνα αρχικοποίησης (Initialization routine)
void init (void){
                set_tris_b(0x00);     // Καθορισμός της πόρτας B ως έξοδος
                set_tris_c(0x00);     // Καθορισμός της πόρτας C ως έξοδος
                PORTB = 0;
                PORTC = 0;
                counter = 200;        // Αρχική τιμή του counter
                seconds = 0;          // Αρχική τιμή του μετρητή χρόνου
                des =0;               // Αρχική τιμή επιλογής ψηφίου
                SETUP_TIMER_0(T0_INTERNAL | T0_DIV_1 ); // Prescaler
                                                        //  /1
                set_timer0(5536);   //  Αρχική τιμή του  hardware μετρητή
                enable_interrupts(INT_TIMER0); // Ενεργοποίηση της
                                               // διακοπής του timer0
                enable_interrupts(GLOBAL);      // Ενεργοποίηση του γενικού
                                               // διακόπτη των διακοπών

        }
```

Ασκήσεις που θα πρέπει να ετοιμαστούν για το πέμπτο εργαστήριο

1. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής θα λειτουργεί ως χρονόμετρο με ανάστροφη φορά.

- Ο χρόνος θα αρχίζει να μετράτε σε δευτερόλεπτα ξεκινώντας από την τιμή 300 όταν το bit RD1 τεθεί σε λογικό 0 και για όσο χρόνο είναι σε λογικό 0.
- Αν κατά τη μέτρηση αλλάξει κατάσταση στο bit RD0 από λογικό «1» που ήταν αρχικά σε λογικό «0» τότε θα πρέπει να αποθηκεύεται σε βοηθητική μεταβλητή η οποία θα απεικονίζεται στο τέλος της μέτρησης.
- Η μέτρηση προς τα κάτω θα σταματά στην τιμή 280 και μετά από 5 sec θα απεικονίζεται η αποθηκευμένη μεταβλητή για 5 sec.
- Η αλλαγή κατάστασης του bit RD3 από λογικό «1» σε λογικό «0» οποιαδήποτε στιγμή κάνει reset στον μετρητή και τον επανατοποθετεί στην αρχική του τιμή δηλαδή το 300.
- Η παραπάνω διαδικασία να επαναλαμβάνεται συνεχώς. Να χρησιμοποιηθεί η μέθοδος της διακοπής από την υπερχείλιση του timer0 για την μέτρηση του χρόνου και οι πίνακες μετατροπής για την οδήγηση των ενδείκτων.

2, Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να λειτουργεί το σύστημα ως ρολόι πραγματικού χρόνου:
Το σύστημα θα λειτουργεί ως εξής:

- Αρχικά η ένδειξη θα είναι 12:00. Επειδή υπάρχουν μόνο 3 ενδείκτες η ένδειξη θα γίνεται σε δύο φάσεις. Για μισό δευτερόλεπτο θα απεικονίζεται η ώρα και για μισό τα λεπτά.

- Για να μπορεί να γίνεται ο διαχωρισμός της ένδειξης μεταξύ των δύο ενδείξεων θα πρέπει όταν η ένδειξη στους δύο δεξιότερους ενδείκτες είναι η ώρα τότε στον αριστερότερο ενδείκτη θα εμφανίζεται η ένδειξη Ω ενώ όταν η ένδειξη είναι τα λεπτά στον αριστερότερο ενδείκτη θα υπάρχει η ένδειξη Π.
- Να χρησιμοποιηθεί η μέθοδος της διακοπής από την υπερχείλιση του timer0 για την μέτρηση του χρόνου και οι πίνακες μετατροπής για την οδήγηση των ενδείκτων.
- Για την προ-τοποθέτηση στην αρχική τιμή της ώρα χρησιμοποιούνται οι διακόπτες RD0 και RD1 όταν το bit RD2 τεθεί σε λογικό «1» (time set). Ο διακόπτης RD0 αυξάνει την τιμή των λεπτών με ρυθμό ενός δεκάτου του δευτερολέπτου. Ενώ ο διακόπτης RD1 θα μειώνει την τιμή των λεπτών με τον ίδιο ρυθμό.

Και στις δύο ασκήσεις θα χρησιμοποιηθούν για τους ενδείκτες οι ίδιες πόρτες όπως το παράδειγμα που περιγράφεται στην παραπάνω λυμένη άσκηση.(Πόρτα Β για τους επτά τομής του ενδείκτη και πόρτα C για την επιλογή των ψηφίων).

Ερωτήσεις που θα πρέπει να απαντηθούν.

1. Ποια θα είναι η τιμή του x μετά την εκτέλεση του παρακάτω προγράμματος.

```
x=0;
y=2;
switch (y) {
case 0 : x+=1;
case 1 : x+=2;
case 2 : x+=4;
case 3 : x+=8;
}
```

2. Στην παρακάτω function και στη κλήση της ποια θα είναι η τιμή του x

```
int calculate (int x, int y) {
if(x>5)
        x=5;
return x*y;
}

x= calculate (1*2*3 , 4*5);
```

a.100
b.120
c. 4
d.20
e.Κανένα από τα παραπάνω δεν είναι σωστό στη C.

3. Τι αποτέλεσμα θα έχει ο παρακάτω κώδικας

```
int8 table[10] = {1,2};
for(int sum=0, int i=0;
i<sizeof(table); i++)
  sum+ = table[i];
```

```
cout << sum;
```

a.1
b.3
c.15
d.21
e.255

4. Ποια θα είναι η έξοδος από το παρακάτω πρόγραμμα.

```
int table[] = {1,2,3,4,5,6,7,8,9};
for(int i=1; i<=3 ; i++)
cout << table[ i*3 %9];
```

a.369
b.470
c.240
d.18241
e. Δεν είναι σωστό στη C

5. Ποια θα είναι η τιμή του x μετά την εκτέλεση του παρακάτω προγράμματος

```
int16 x;
int16 * p;
int16 a[5];
#locate a=0x100
a[0] = 3;
a[1] = 4;
p=a;
x=p+1;
```

a.3
b.4
c.0x101
d.0x102
e.Συντακτικό λάθος. Δεν είναι σωστό στη C