

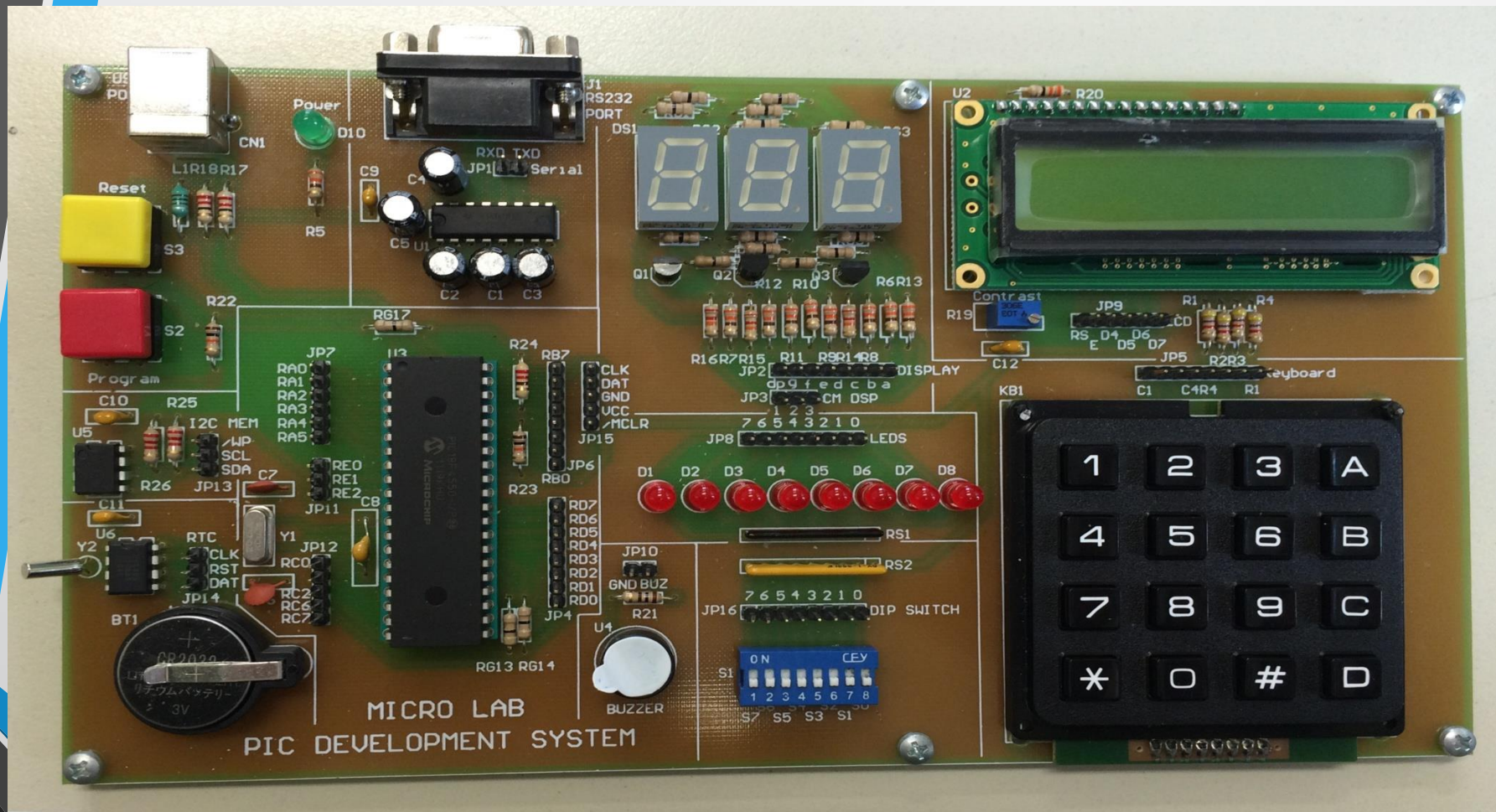
Ενσωματωμένα Συστήματα

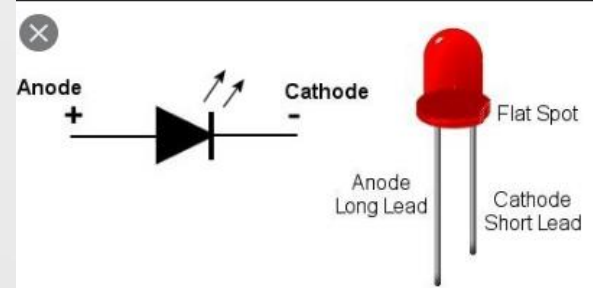
(6^ο εξάμηνο)

08-Seven Segment Displays

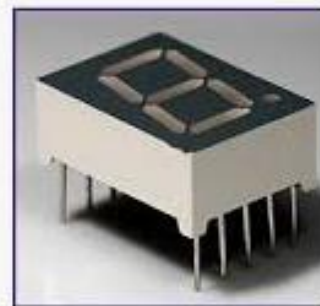
Διδάσκουσα: Παπαδοπούλου Μαρία
Επίκουρη Καθηγήτρια

Εμφάνιση μηνυμάτων σε ενδείκτες 7 τομέων



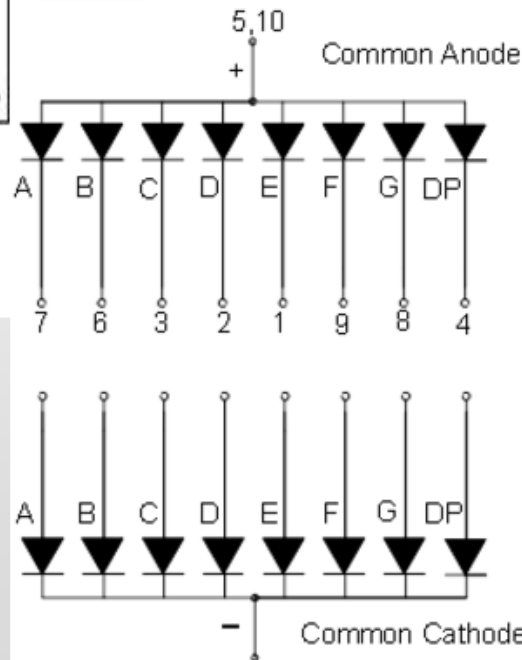
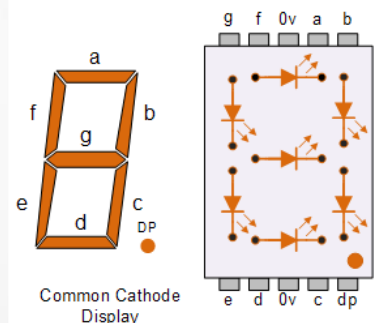
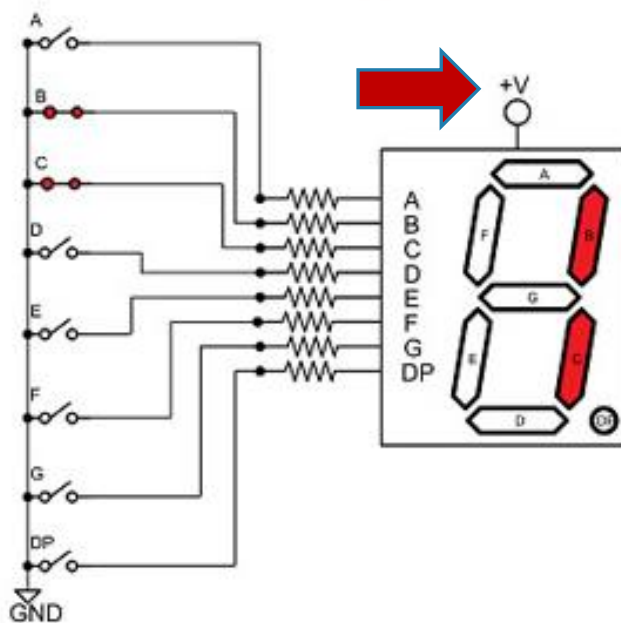
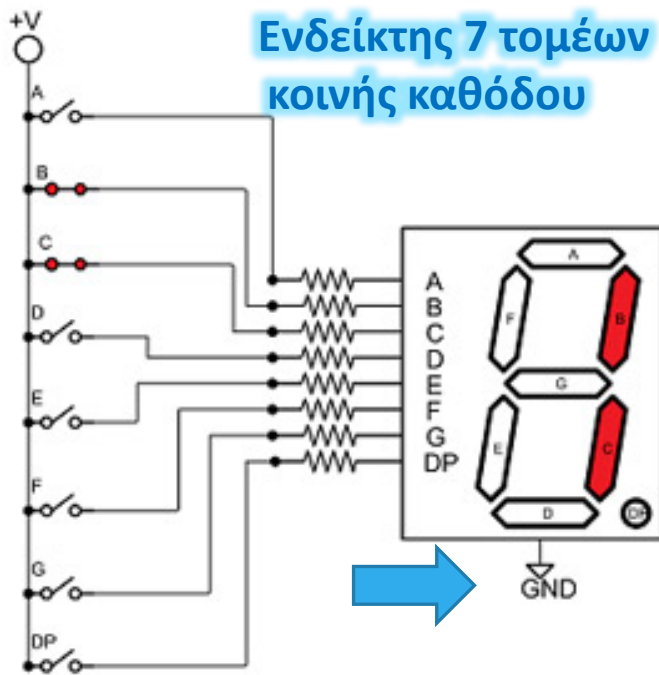



Ενδείκτης 7 τομέων Seven segment display




Ενδείκτης 7 τομέων
κοινής ανόδου

Ενδείκτης 7 τομέων
κοινής καθόδου



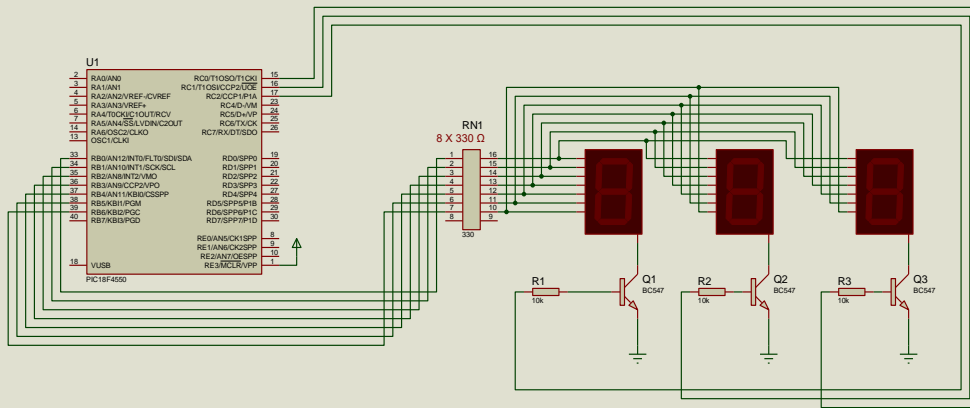
- Οι **κάθοδοι** των διόδων συνδέονται στη γη
- Για να ανάψει το  στέλνουμε:

00111111

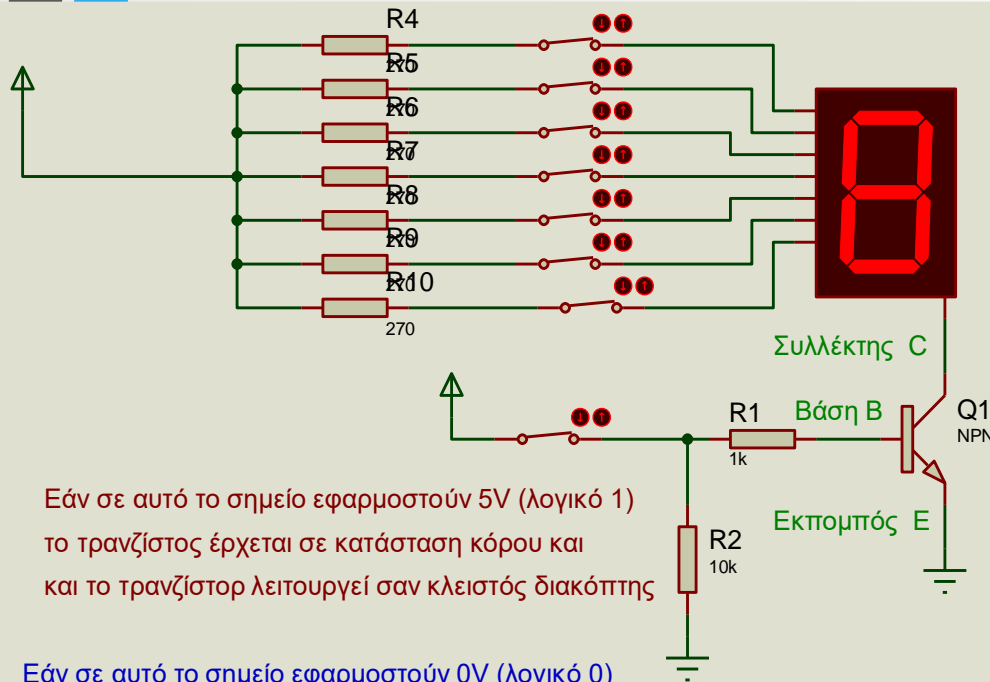
- Οι **άνοδοι** των διόδων συνδέονται στην τάση τροφοδοσίας
- Για να ανάψει το  στέλνουμε:

11000000

Άσκηση 3α. Να εμφανιστεί στους τρεις ενδείκτες 7 τομέων ο αριθμός **12.3**



Ενεργοποίηση και απενεργοποίηση του ενδείκτη 7 τομέων με εφαρμογή **λογικού 1** ή λογικού 0 στη βάση του τρανζίστορ



Εάν σε αυτό το σημείο εφαρμοστούν 5V (λογικό 1) το τρανζίστορ έρχεται σε κατάσταση κόρου και το τρανζίστορ λειτουργεί σαν κλειστός διακόπτης

Εάν σε αυτό το σημείο εφαρμοστούν 0V (λογικό 0) το τρανζίστορ έρχεται σε κατάσταση αποκοπής και το τρανζίστορ λειτουργεί σαν ανοικτός διακόπτης

κλειστός διακόπτης

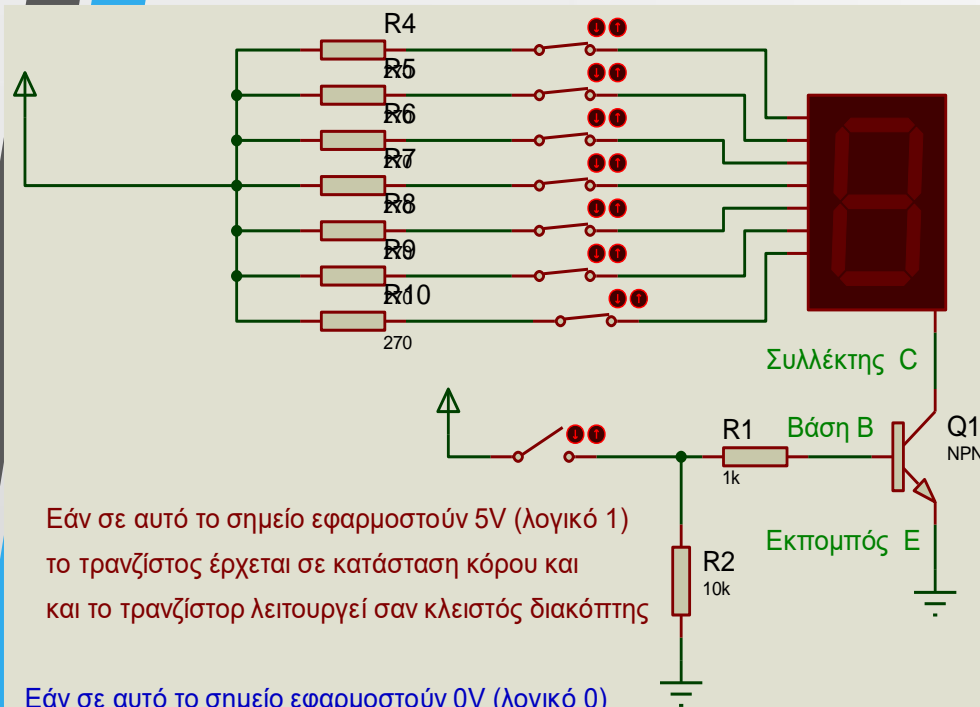
στον κλειστό διακόπτη
οι επαφές είναι ενωμένες

ανοικτός διακόπτης

στον ανοιχτό διακόπτη
οι επαφές δεν είναι ενωμένες

- Όταν εφαρμοστεί λογικό 1 στη βάση του τρανζίστορ το τρανζίστορ έρχεται σε κατάσταση κόρου και κλείνει την επαφή μεταξύ συλλέκτη C και εκπομπού E.
- Ο ακροδέκτης καθόδου του ενδείκτη 7 τομέων συνδέεται με τη γη

Ενεργοποίηση και απενεργοποίηση του ενδείκτη 7 τομέων με εφαρμογή λογικού 1 ή λογικού 0 στη βάση του τρανζίστορ



Εάν σε αυτό το σημείο εφαρμοστούν 5V (λογικό 1) το τρανζίστορ έρχεται σε κατάσταση κόρου και το τρανζίστορ λειτουργεί σαν κλειστός διακόπτης

Εάν σε αυτό το σημείο εφαρμοστούν 0V (λογικό 0) το τρανζίστορ έρχεται σε κατάσταση αποκοπής και το τρανζίστορ λειτουργεί σαν ανοικτός διακόπτης

κλειστός διακόπτης

στον κλειστό διακόπτη
οι επαφές είναι ενωμένες

ανοικτός διακόπτης

στον ανοικτό διακόπτη
οι επαφές δεν είναι ενωμένες

- Όταν εφαρμοστεί λογικό 0 στη βάση του τρανζίστορ το τρανζίστορ έρχεται σε κατάσταση αποκοπής και ανοίγει την επαφή μεταξύ συλλέκτη C-εκπομπού E.
- Αποκόπτεται η σύνδεση του ακροδέκτη καθόδου του ενδείκτη 7 τομέων με τη γη

Εμφάνιση του αριθμού **12.3** στους τρεις ενδείκτες 7 τομέων

Λύση

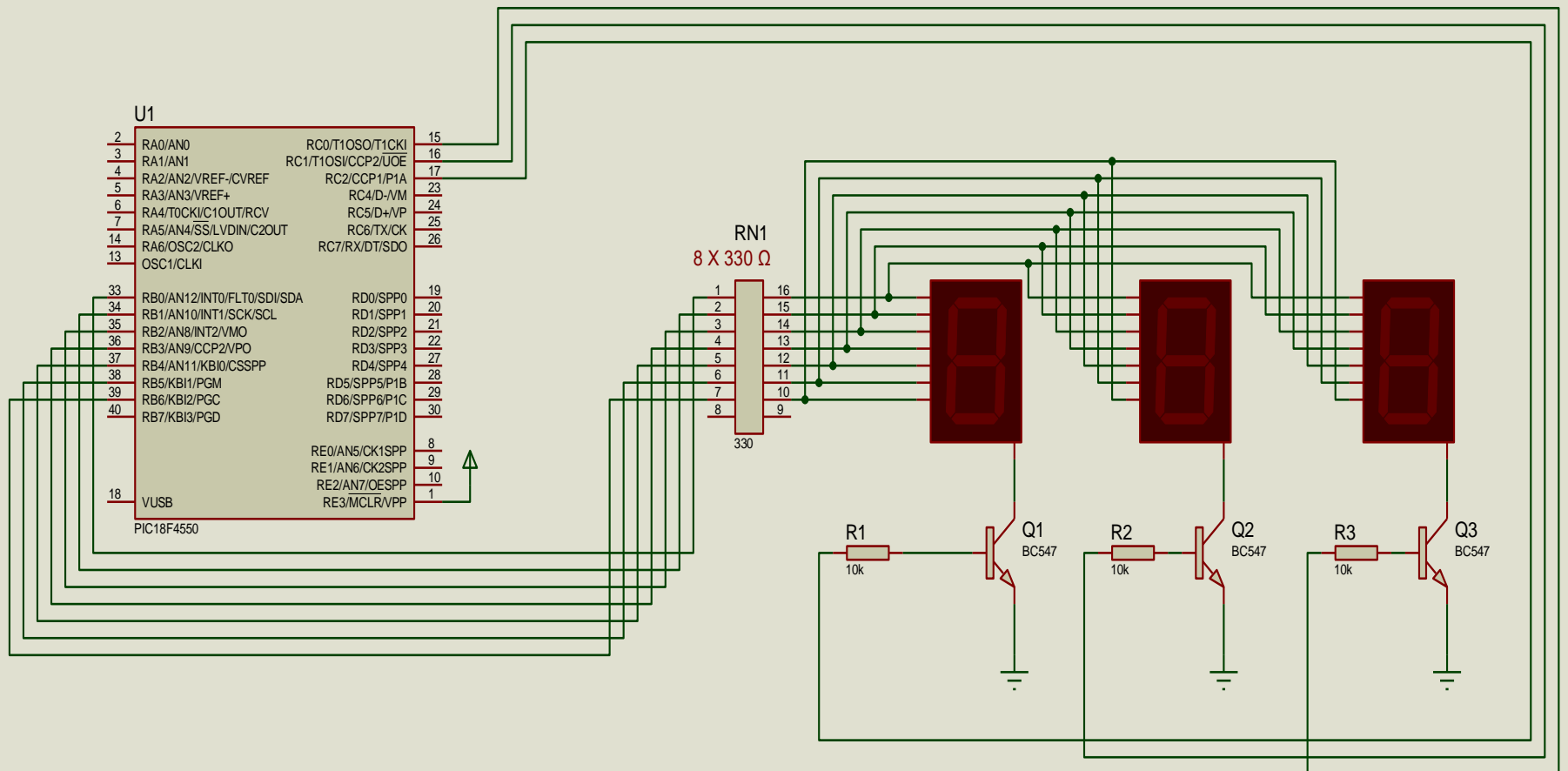
- Θα πρέπει να ενεργοποιηθεί ο δεξιός ενδείκτης ($C_0=1, C_1=0, C_2=0$).
- Στη συνέχεια να αποσταλεί στην πόρτα B ο κώδικας για την εμφάνιση του **3**
- Η παραπάνω κατάσταση να παραμείνει για 5 ms

- Θα πρέπει να ενεργοποιηθεί ο μεσαίος ενδείκτης ($C_0=0, C_1=1, C_2=0$).
- Στη συνέχεια να αποσταλεί στην πόρτα B ο κώδικας για την εμφάνιση του **2**.
- Η παραπάνω κατάσταση να παραμείνει για 5 ms

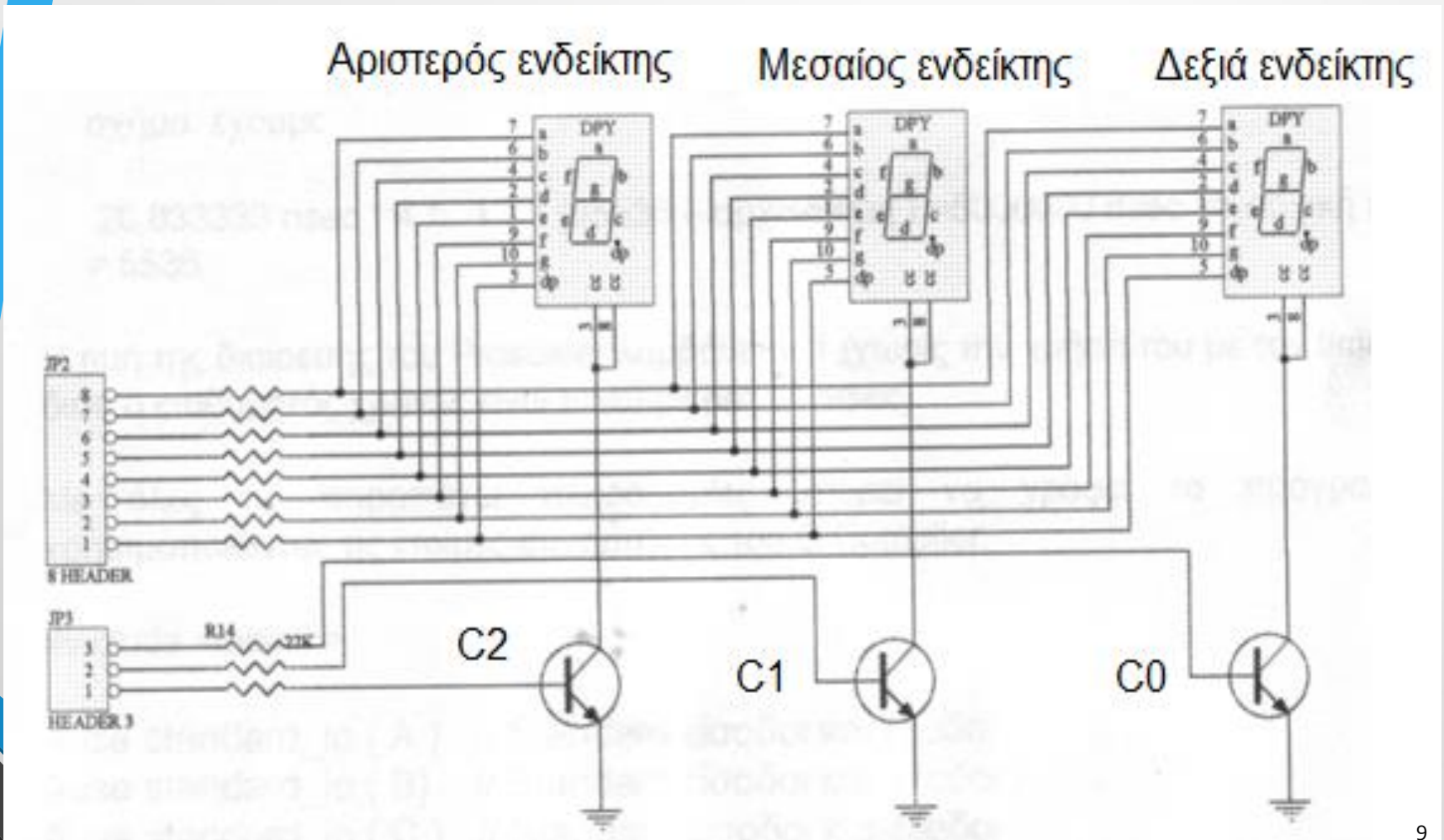
- Θα πρέπει να ενεργοποιηθεί ο αριστερός ενδείκτης ($C_0=0, C_1=0, C_2=1$).
- Στη συνέχεια να αποσταλεί στην πόρτα B ο κώδικας για την εμφάνιση του **1**
- Η παραπάνω κατάσταση να παραμείνει για 5 ms

- Τα παραπάνω βήματα θα επαναλαμβάνονται συνεχώς, άρα θα πρέπει να μπουν μέσα σε μια while(TRUE).

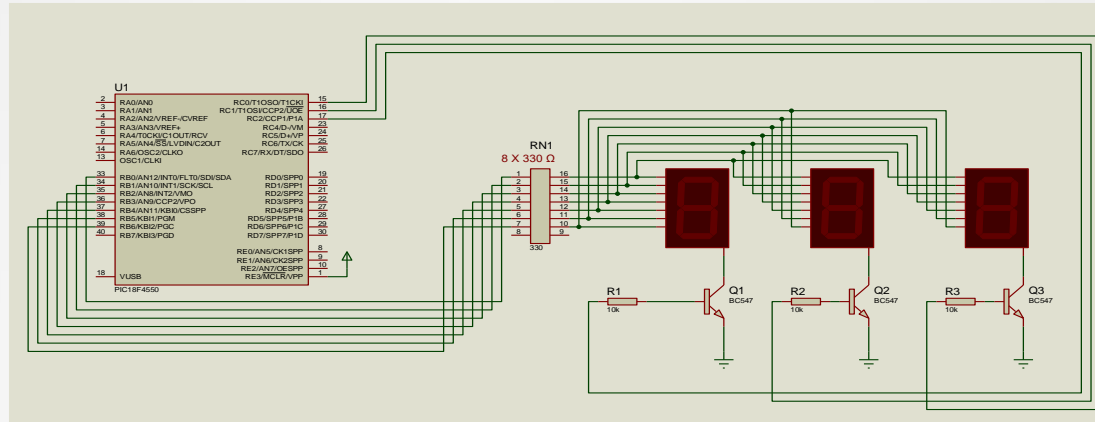
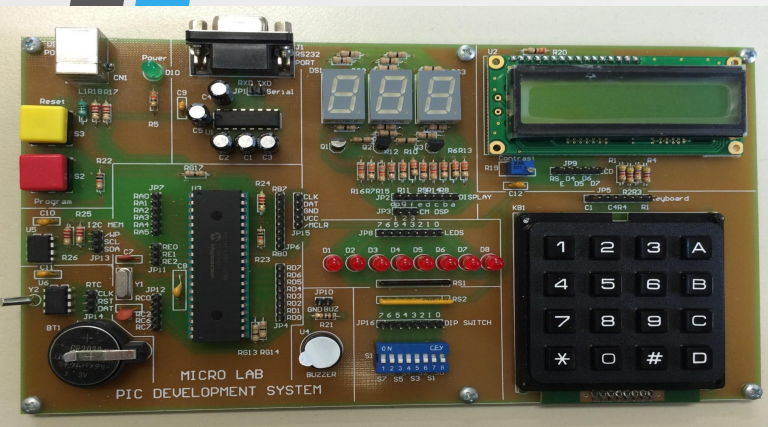
Να εμφανιστεί στους τρεις ενδείκτες 7 τομέων ο αριθμός 12.3



Να εμφανιστεί στους τρεις ενδείκτες 7
τομέων ο αριθμός **12.3**



Κώδικας για την εμφάνιση του κάθε ψηφίου



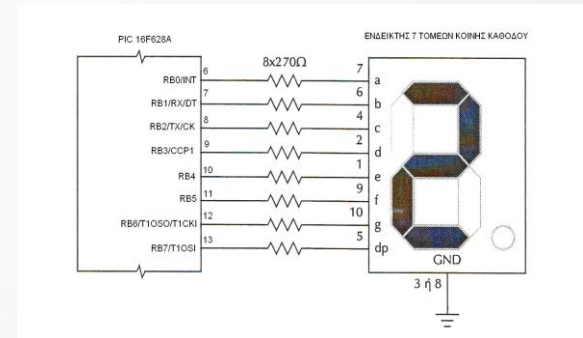
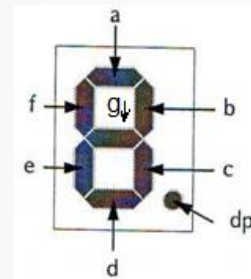
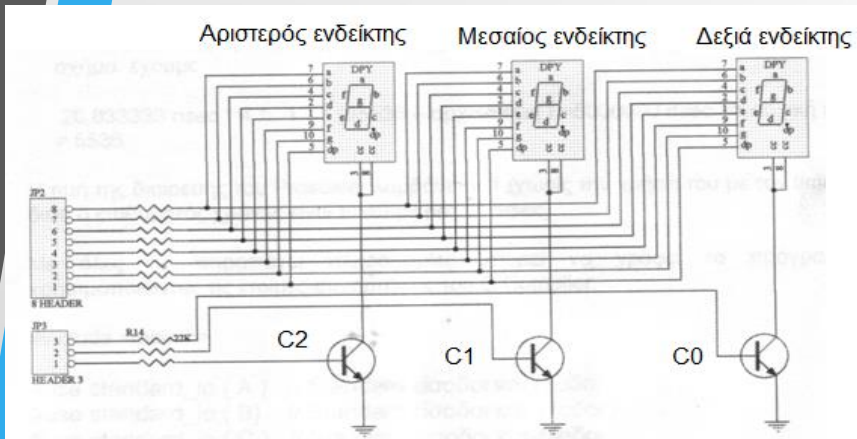
- Θα πρέπει να βρούμε τον κώδικα για την εμφάνιση του καθενός από τα παρακάτω ψηφία
1 2 3
- Για να εμφανιστεί το **1** πρέπει να είναι αναμμένοι οι τομείς b και c, επομένως πρέπει να αποσταλεί στην πόρτα B η τιμή 000 00110 (dp-g-f-e-d-**c-b-a**)
- Για να εμφανιστεί το **2**, πρέπει να είναι αναμμένοι οι τομείς a, b, g, e και dp, επομένως πρέπει να αποσταλεί στην πόρτα B η τιμή 1101 1011 (**dp-g-f-e-d-c-b-a**)
- Για να εμφανιστεί το **3** πρέπει να είναι αναμμένοι οι τομείς a, b, c, d, g, επομένως πρέπει να αποσταλεί στην πόρτα B η τιμή 01001111 (dp-g-f-e-**d-c-b-a**)

Πρόγραμμα εμφάνισης του 12.3

```
#include <main.h>
#byte PORTB =0xF81 //καθορισμός του καταχωρητή δεδομένων της πόρτας B
#byte PORTC =0xF82 //καθορισμός του καταχωρητή δεδομένων της πόρτας C
void main() {
    set_tris_b(0x00); // Καθορισμός της πόρτας B ως έξοδος
    set_tris_c(0x00); // Καθορισμός της πόρτας C ως έξοδος
    while (TRUE){    // οι κώδικες του κάθε ψηφίου στέλνονται κάθε 5 ms
                    // με ενεργοποίηση του αντίστοιχου ενδείκτη
        // Εμφάνιση του 3
        PORTC=0b00000001; //ενεργοποίηση του δεξιά ενδείκτη
        PORTB=0b01001111; //αποστολή του κώδικα για εμφάνιση του 3
        delay_ms(5);      //αναμονή για 5 ms
        //Εμφάνιση του 2.
        PORTC=0b00000010; // ενεργοποίηση του μεσαίου ενδείκτη
        PORTB=0b11011011; //αποστολή του κώδικα για εμφάνιση του 2.
        delay_ms(5);      //αναμονή για 5 ms
        // Εμφάνιση του 1
        PORTC=0b00000100; //ενεργοποίηση του αριστερά ενδείκτη
        PORTB=0b00000110; //αποστολή του κώδικα για εμφάνιση του 1
        delay_ms(5);      //αναμονή για 5 ms
    } // κλείνει η αγκύλη while(TRUE)
} // κλείνει η αγκύλη main
```

Άσκηση 3b. Ασκήσεις πάνω στους ενδείκτες 7 τομέων

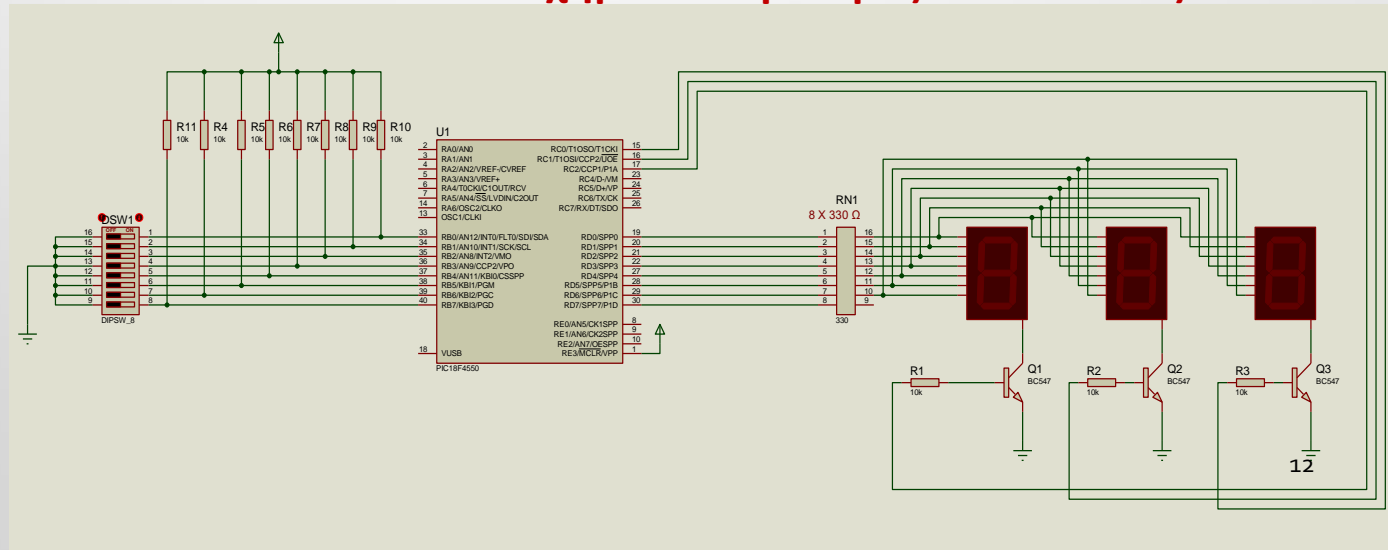
Να γραφεί πρόγραμμα με το οποίο διαβάζεται συνεχώς η τιμή της πόρτας B που χρησιμοποιείται σαν είσοδος και εμφανίζεται η τιμή της, στο δεκαδικό αριθμητικό σύστημα, στους 3 ενδείκτες 7 τομέων.



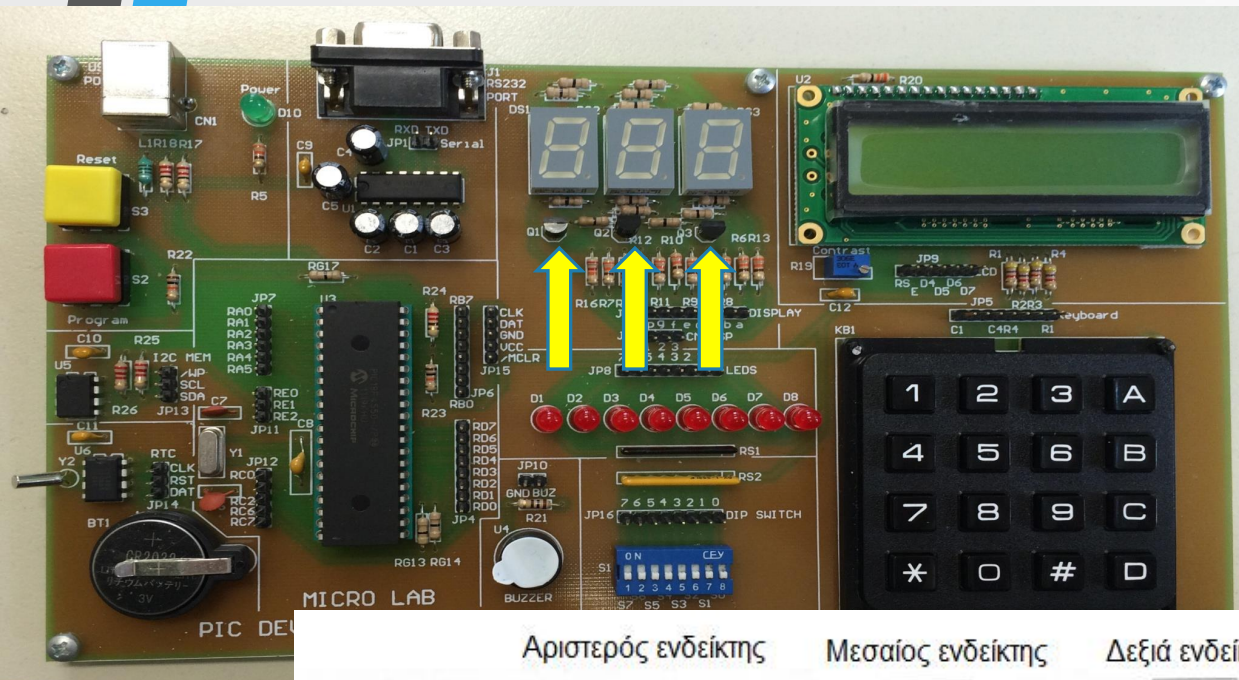
Προσοχή! Στην πόρτα B μπορούμε να σχηματίσουμε τιμές από 000 έως 255

Με τους ακροδέκτες C0, C1, C2 επιλέγουμε τον ενδείκτη 7 τομέων που θα ενεργοποιηθεί

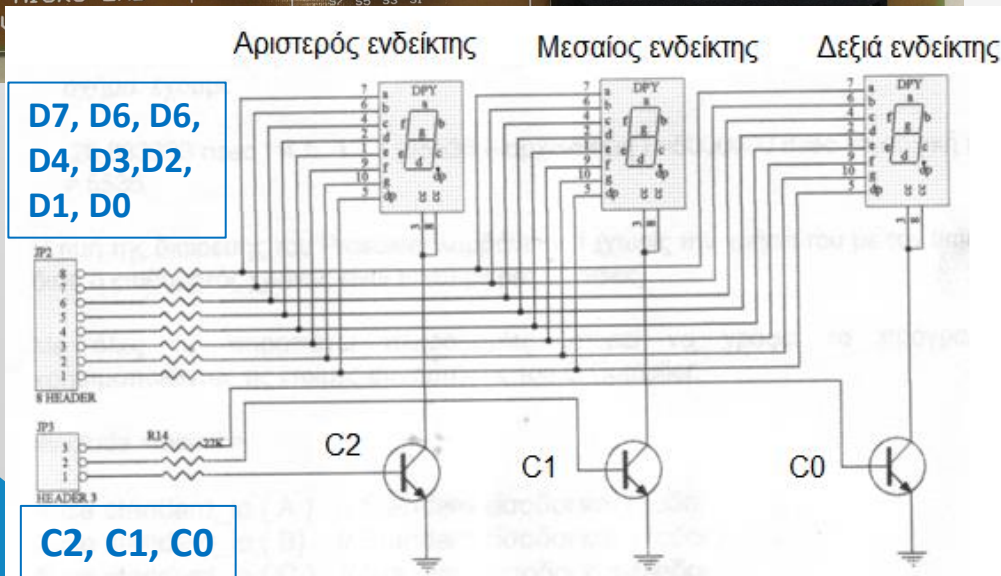
Από την πόρτα D στέλνουμε τον κώδικα του ψηφίου που θέλουμε να εμφανιστεί



Σύνδεση στην πλακέτα ανάπτυξης εφαρμογών του εργαστηρίου



Τα κίτρινα βέλη ↑ στη φωτογραφία της πλακέτας δείχνουν τα τρία τρανζίστορ με τα οποία επιλέγεται ο ενδείκτης που θα ενεργοποιηθεί δια μέσου των τιμών που τοποθετούνται στους ακροδέκτες C0, C1, C2



Αρχικές ρυθμίσεις

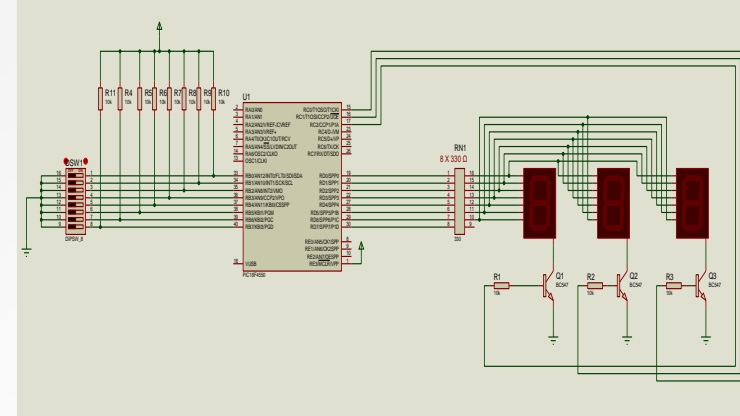
```
set_tris_b(0xFF); // Καθορισμός της πόρτας B ως εισόδου
set_tris_c(0x00); // Καθορισμός της πόρτας C ως εξόδου
set_tris_d(0x00); // Καθορισμός της πόρτας D ως εξόδου
```

```
int8 table[16] = { 0b00111111, // Πίνακας με του κώδικες για
                    // εμφάνιση σε ενδείκτη 7 τομέων
```

Ο πίνακας με όνομα table αποτελείται από 16 τιμές και κάθε μια τιμή είναι αριθμός των 8 bit

```
0b00000110, // των ψηφίων 0, 1, 2, ... 9, ... F.
0b01011011, // κώδικας για εμφάνιση του 2
0b01001111, // κώδικας για εμφάνιση του 3
0b01100110, // κώδικας για εμφάνιση του 4
0b01101101, // κώδικας για εμφάνιση του 5
0b01111101, // κώδικας για εμφάνιση του 6
0b00000111, // κώδικας για εμφάνιση του 7
0b01111111, // κώδικας για εμφάνιση του 8
0b01101111, // κώδικας για εμφάνιση του 9
0b01110111, // κώδικας για εμφάνιση του A
0b01111100, // κώδικας για εμφάνιση του B
0b00111001, // κώδικας για εμφάνιση του C
0b01011110, // κώδικας για εμφάνιση του D
0b01111001, // κώδικας για εμφάνιση του E
0b01110001; // κώδικας για εμφάνιση του F

int isodos; // μεταβλητή για αποθήκευση της τιμής εισόδου
int monades; // μεταβλητή για αποθήκευση των μονάδων
// της τιμής εισόδου
int decades; // μεταβλητή για αποθήκευση των δεκάδων
// της τιμής εισόδου
int ekatontades; // τιμή για αποθήκευση των εκατοντάδων της
// τιμής εισόδου
```



Παράδειγμα

- Για να εμφανιστεί σε έναν ενδείκτη η τιμή 8 θα πρέπει να σταλεί στην πόρτα B η τιμή 00111111, δηλαδή η δυαδική τιμή 0b00111111 είναι ο κώδικας για εμφάνιση του **8**
- Έστω ότι σχηματίσαμε στην πόρτα B την τιμή 1111 1110 η οποία στο δεκαδικό αριθμητικό σύστημα γράφεται: **254**
- Η μεταβλητή **ekatontades** θα πρέπει να πάρει την τιμή **2**.
- Η μεταβλητή **decades** θα πρέπει να πάρει την τιμή **5**.
- Η τιμή **monades** θα πάρει την τιμή **4**.

Πώς υπολογίζουμε τις εκατοντάδες, δεκάδες και μονάδες ενός αριθμού;

Οι μεταβλητές *isodos*, *ekatontades*, *decades*, *monades* είναι ακέραιες μεταβλητές (int)

```
isodos=PORTB;           // η isodos μπορεί να πάρει τιμή από 0 έως 255
ekatontades=isodos/100;  // υπολογισμός των εκατοντάδων της τιμής
                        // της πόρτας B (τιμή από 0-9)
decades=(isodos-ekatontades*100)/10;  // υπολογισμός των δεκάδων της τιμής
                                     // της πόρτας B (τιμή από 0-9)
monades =isodos-ekatontades*100-decades*10; //υπολογισμός των μονάδων της
                                     //τιμής της πόρτας B (τιμή από 0-9)
```

Παράδειγμα: isodos = 254

$$ekatontades = \frac{isodos}{100} = \frac{254}{100} = 2 \quad (\text{διότι η μεταβλητή } ekatontades \text{ είναι ακέραιη μεταβλητή})$$

$$decades = \frac{isodos - ekatontades \times 100}{10} = \frac{254 - ekatontades \times 100}{10} = \frac{254 - 2 \times 100}{10} =$$
$$= \frac{254 - 200}{10} = \frac{54}{10} = 5 \quad (\text{διότι η μεταβλητή } decades \text{ είναι ακέραιη μεταβλητή})$$

$$monades = isodos - ekatontades \times 100 - decades \times 10 =$$
$$= 254 - 2 \times 100 - 5 \times 10 = 254 - 200 - 50 = 4$$

Κύριο πρόγραμμα

```
while (TRUE){           // οι κώδικες του κάθε ψηφίου στέλνονται κάθε 5 ms
                        // με ενεργοποίηση του αντίστοιχου ενδείκτη

isodos=PORTB;
ekatontades=isodos/100;           // υπολογισμός των εκατοντάδων της τιμής της πόρτας B
decades=(isodos-ekatontades*100)/10; // υπολογισμός των δεκάδων της τιμής της πόρτας B
monades =isodos-ekatontades*100-decades*10; //υπολογισμός των μονάδων της τιμής της πόρτας B

//.....Εμφάνιση των εκατοντάδων .....
PORTC=0b000000100;           //ενεργοποίηση του αριστερά ενδείκτη
PORTD=table[ekatontades]; //αποστολή του κώδικα για εμφάνιση των εκατοντάδων
delay_ms(5);                 //αναμονή για 5 ms

//.....Εμφάνιση των δεκάδων .....
PORTC=0b000000010;           // ενεργοποίηση του μεσαίου ενδείκτη
PORTD=table[decades]; //αποστολή του κώδικα για εμφάνιση των δεκάδων
delay_ms(5);                 //αναμονή για 5 ms

//.....Εμφάνιση των μονάδων .....
PORTC=0b000000001;           //ενεργοποίηση του δεξιά ενδείκτη
PORTD=table[monades]; //αποστολή του κώδικα για εμφάνιση των μονάδων
delay_ms(5);                 //αναμονή για 5 ms

}                             // κλείνει η αγκύλη του while(TRUE)
}                             // κλείνει η αγκύλη του main
```

RC2=1, RC1=0, RC0=0

**Ενεργοποιείται ο ενδείκτης
των εκατοντάδων**

RC2=0, RC1=1, RC0=0

**Ενεργοποιείται ο ενδείκτης
των δεκάδων**

RC2=0, RC1=0, RC0=1

**Ενεργοποιείται ο ενδείκτης
των μονάδων**

Άσκηση 3c. Μετρητής δεκάτων του δευτερολέπτου σε 3 ενδείκτες 7 τομέων

RD0 μετάβαση από 1 σε 0 : Ξεκινάει ο μετρητής δεκάτων του δευτερολέπτου.

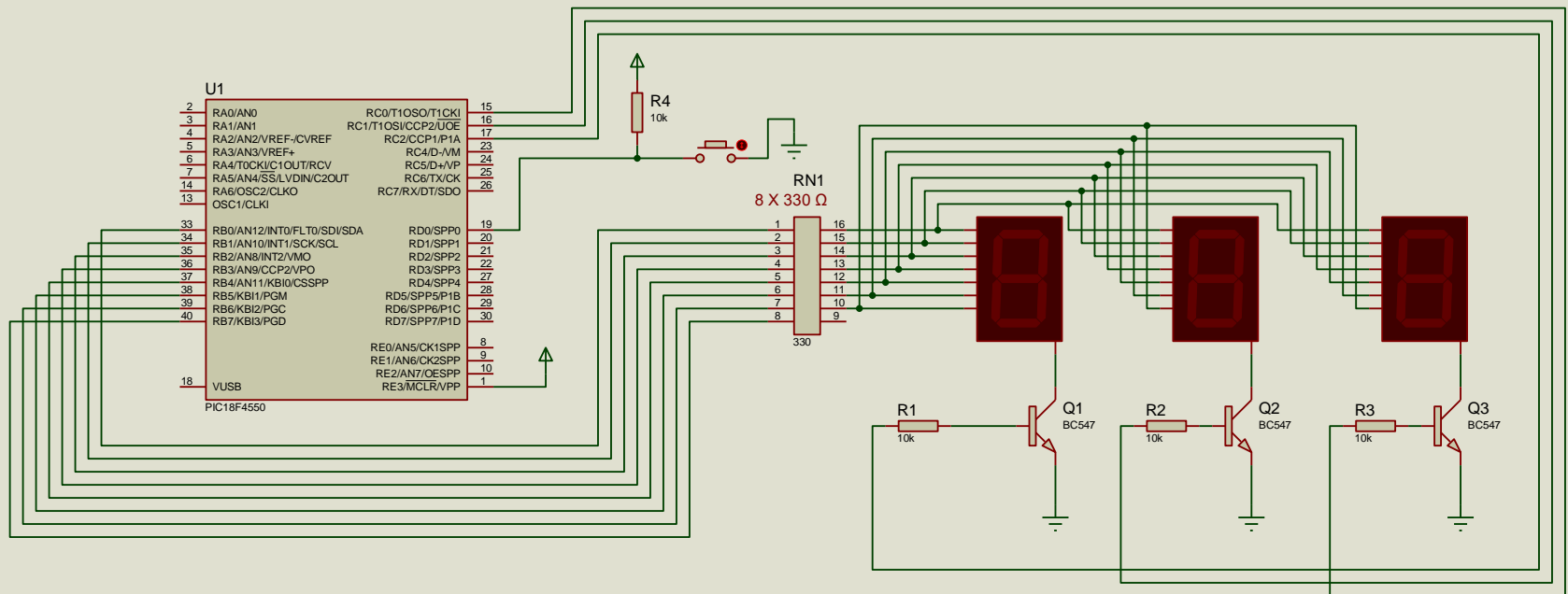
RD0 0: Εμφανίζεται στους ενδείκτες η τρεχούμενη τιμή της μέτρησης χρόνου(σε δέκατα του δευτερολέπτου)

RD0 1 : Εμφανίζεται η τιμή του μετρητή στους ενδείκτες κατά την στιγμή μετάβασης από 0 σε 1
(Ο μετρητής χρόνου συνεχίζει να μετράει αλλά δεν φαίνεται στους ενδείκτες)

RD0 0: Εμφανίζεται ξανά στους ενδείκτες η τρεχούμενη τιμή της μέτρησης χρόνου(σε δέκατα του δευτερολέπτου)

RD0 1 : Εμφανίζεται η τιμή του μετρητή στους ενδείκτες κατά την στιγμή μετάβασης από 0 σε 1
(Ο μετρητής χρόνου συνεχίζει να μετράει αλλά δεν φαίνεται στους ενδείκτες)

Οι παραπάνω αλλαγές στον ακροδέκτη RD0 μπορούν να επαναληφθούν με τα ίδια αποτελέσματα.



- Ο μετρητής ξεκινάει από το 000, φτάνει στο 999 και ξαναξεκινάει από το 000.
- Χρειάζεται συνολικά 100 sec (= 1000 x 0,1 sec) για να πάει από το 000 έως το 999.

Βασικές ιδέες για τη λύση του προβλήματος

1. Χρειαζόμαστε μια μεταβλητή, π.χ. με όνομα **tenthseconds** η οποία θα αυξάνει κατά 1 κάθε 0,1 δευτερόλεπτα

Η μεταβλητή αυτή θα παίρνει τιμές από 000 έως 999 και στη συνέχεια θα ξεκινάει πάλι από το 000

Την τιμή αυτή της μεταβλητής counter θα πρέπει να την εμφανίζουμε στους τρεις ενδείκτες 7 τομέων.

2. Για να γίνει αυτό θα πρέπει να υπολογίζονται οι εκατοντάδες, οι δεκάδες και οι μονάδες της μεταβλητής **tenthseconds** και να εμφανίζονται στον πρώτο αριστερά, στον μεσαίο και στον δεξιά ενδείκτη.

Επομένως θα χρειαστούν τρεις ακόμα μεταβλητές, που θα παίρνουν τιμές από 0 έως 9, στις οποίες θα καταγράφονται οι εκατοντάδες, δεκάδες και μονάδες της μεταβλητής **tenthseconds**.

Οι τρεις αυτές μεταβλητές θα μπορούσαν να ονομαστούν **eka**, **dec**, **mon**.

Οι τιμές αυτών των μεταβλητών θα χρησιμοποιηθούν για την εμφάνιση των τριών ψηφίων στους ενδείκτες 7 τομέων.

Σε κάθε χρονική στιγμή θα εμφανίζεται μόνο μια από αυτές τις τιμές σε έναν μόνο ενδείκτη, αλλά επειδή θα αναβοσβήνουν με μεγάλη ταχύτητα θα εμφανίζονται και οι τρεις ενδείκτες αναμμένοι.

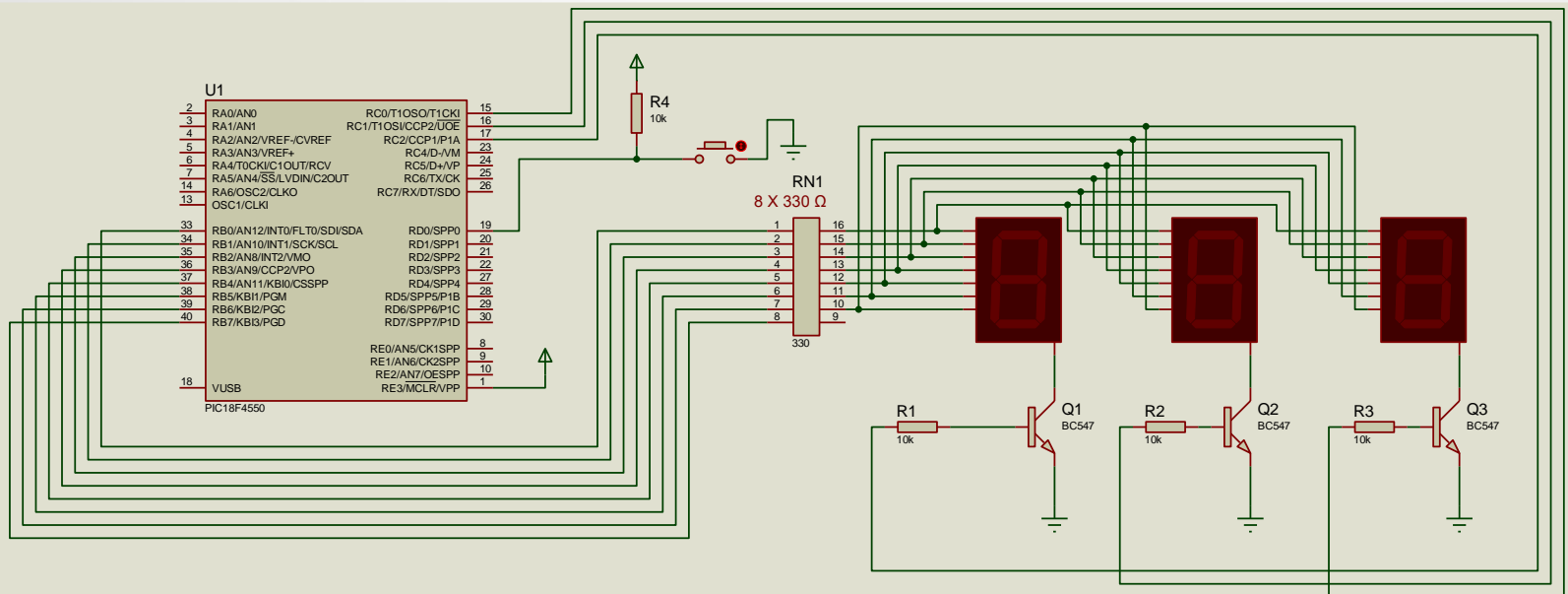


Θέλουμε να μετράμε δέκατα του δευτερολέπτου (0,1 sec). Ανά πόσο χρονικό διάστημα θα ρυθμίσουμε να γίνονται διακοπές (interrupts);

- Μπορούμε να ρυθμίσουμε να γίνονται διακοπές κάθε 5 ms, δηλαδή 200 διακοπές το δευτερόλεπτο.
- Αν γίνονται διακοπές κάθε 5 ms, ύστερα από 20 διακοπές θα έχει περάσει χρόνος $5\text{ms} \times 20 = 100\text{ms} = 0,1\text{sec}$
- Με μια μεταβλητή, **counter** μπορούμε να μετράμε 20 διακοπές και κάθε 20 διακοπές θα αυξάνουμε την τιμή της μεταβλητής **tenthseconds** κατά 1.
- Επιπλέον, σε κάθε διακοπή, δηλαδή κάθε 5 ms, θα εμφανίζουμε ένα από τα τρία ψηφία της μεταβλητής **tenthseconds**, δηλαδή μόνο τις εκατοντάδες, μόνο τις δεκάδες ή μόνο τις μονάδες. Οι τρεις αυτές τιμές αποθηκεύονται στις μεταβλητές **eka**, **dec**, **mon**.
- **APA**, κάθε ενδείκτης θα είναι αναμμένος μόνο για 5 ms και για τα επόμενα 10 ms θα είναι κλειστός

Θέλουμε κάθε 5 ms να ενεργοποιούμε μόνο έναν ενδείκτη και να εμφανίζουμε διαδοχικά τις εκατοντάδες τις δεκάδες και τις μονάδες. Πώς θα γίνει αυτό;

```
eka = (int8) (tenthseconds_stop /100); // Υπολογισμός των εκατοντάδων του μετρητή
dec = (int8) ((tenthseconds_stop - (100*eka))/10); // Υπολογισμός των δεκάδων του μετρητή
mon = (int8) (tenthseconds_stop - (100 * eka) - (10 * dec)); // Υπολογισμός των μονάδων
// του μετρητή
```



Εμφάνιση των εκατοντάδων δεκάδων και μονάδων

```
des = ++des%3;           // Μεταβλητή des παίρνει τις τιμές 0, 1, 2, 0, 1, 2, 0, 1, 2,...
                           // %3 σημαίνει modulo 3 (υπόλοιπο διαίρεσης με το 3)
PORTC = dig[des];        // Η πόρτα C παίρνει τιμές 0000 0001, 0000 0010, 0000 0100,...
                           // δηλαδή τις τιμές 1, 2, 4, 1, 2, 4, ... από τον πίνακα dig
                           // Η πόρτα C επιλέγει ποιος ενδείκτης θα ανάψει.

// Εμφάνιση μονάδων, δεκάδων, εκατοντάδων στους ενδείκτες
if (des==0) {             // PORTC=0000 0001(=dig[0]), επιλογή ενδείκτη μονάδων
    PORTB = table[mon];    // Αν des=0 ενεργοποιείται ο ενδείκτης των μονάδων
                           // PORTC=0000 0001(=dig[0])
}
if (des==1){              //PORTC=0000 0010(=dig[1]), επιλογή ενδείκτη δεκάδων
    PORTB = table[dec];    // Αν des=1 ενεργοποιείται ο ενδείκτης των δεκάδων
}
if (des==2) {             // PORTC=0000 0100(=dig[2]), επιλογή ενδείκτη εκατοντάδων
    PORTB = table[eka];    // Αν des=2 ενεργοποιείται ο ενδείκτης των εκατοντάδων
}
```

des	++des	++des%3
0	1	1
1	2	2
2	3	0
0	1	1
1	2	2
2	3	0
0	1	1

Η μεταβλητή **des** παίρνει διαδοχικά τις τιμές 0, 1, 2, 0, 1, 2, ... και
Η **PORTC** παίρνει διαδοχικά τις τιμές 0000 0001, 0000 0010, 0000 0100, ...

```
int8 dig[3] = {1,2,4};    // Πίνακας με τις τιμές 0000 0001, 0000 0010, 0000 0100 για ενεργοποίηση
                           // από τους ακροδέκτες της πόρτας C, με τη σειρά ενός κάθε φορά, από
                           // τους τρεις ενδείκτες κοινής καθόδου. Εφαρμόζονται 5 V στη βάση του
                           // αντίστοιχου τρανζίστορ και αυτό έρχεται στον κόρο συνδέοντας την κοινή
                           // κάθοδο του ενδείκτη με τη γη.
```

Δηλώσεις Άσκησης 3c. Μετρητής δεκάτων του δευτερολέπτου

```
#include <main.h>
```

```
#byte PORTB    =0xF81 //Ορισμός των θυρών με την θέση τους στην μνήμη.
```

```
#byte PORTC    =0xF82
```

```
#byte PORTD    =0xF83
```

Δηλώσεις μεταβλητών

```
int8 des=0;    //Μεταβλητή που χρησιμοποιείται από την πόρτα C για να ενεργοποιεί  
               //έναν κάθε φορά από τους ενδείκτες 1, 2, 3 αποστέλλοντας με την σειρά λογικό 1  
               //στις βάσεις των τρανζίστορ που συνδέουν τις καθόδους των τριών ενδεικτών  
               //προς την γη.
```

```
int16 tenthseconds,tenthseconds_stop=0; //Με την μεταβλητή tenthseconds μετράμε δέκατα  
                                         //του δευτερολέπτου.  
                                         //Την μεταβλητή tenthseconds_stop την χρησιμοποιούμε για  
                                         //να αποθηκεύσουμε την τιμή του χρόνου όταν ο  
                                         //ακροδέκτης RD0 μεταβεί από 0 σε 1.
```

```
int8 counter;    // μεταβλητή για να μετράμε 20 διακοπές που συμβαίνουν κάθε 5 ms  
                 //κάθε 20 διακοπές έχει περάσει ένα δέκατο του δευτερολέπτου(=100 ms)
```

```
int1 start=0;    // Η μεταβλητή start γίνεται 1 όταν ξεκινήσει ο μετρητής και παραμένει για πάντα 1
```

```
int1 stop=0;     // Η μεταβλητή stop παίρνει την τιμή 1 όταν γίνει RD0=1  
                 // Τότε σταματάει να μεταφέρεται η τρέχουσα τιμή του μετρητή στους ενδείκτες.  
                 // Οι ενδείκτες για stop =1 εμφανίζουν την τιμή κατά την στιγμή  
                 //μετάβασης του RD0 από 0 σε 1. Δηλαδή όταν stop=1 «παγώνουν» οι ενδείκτες.
```

Δηλώσεις Άσκησης 3c. Μετρητής δεκάτων του δευτερολέπτου

```
int8 table[16] = { 0b00111111, // Πίνακας με του κώδικες για εμφάνιση σε ενδείκτη 7 τομέων
                  0b00000110, // των ψηφίων 0, 1, 2, ... 9, ... F.
                  0b01011011,
                  0b01001111,
                  0b01100110,
                  0b01101101,
                  0b01111101,
                  0b00000111,
                  0b01111111,
                  0b01101111,
                  0b01110111,
                  0b01111100,
                  0b00111001,
                  0b01011110,
                  0b01111001,
                  0b01110001};

int8 dig[3] = {1,2,4}; // Πίνακας με τις τιμές 0000 0001, 0000 0010, 0000 0100 για ενεργοποίηση
                        // από τους ακροδέκτες της πόρτας C, με τη σειρά ενός κάθε φορά, από
                        // τους τρεις ενδείκτες κοινής καθόδου. Εφαρμόζονται 5 V στη βάση του
                        // αντίστοιχου τρανζίστορ και αυτό έρχεται στον κόρο συνδέοντας την κοινή
                        // κάθοδο του ενδείκτη με την γη.

// Δήλωση συναρτήσεων
void timer0_int(void);
void init (void);
```


Κύρια συνάρτηση και συνάρτηση αρχικοποίησης της Άσκησης 3c. Μετρητής δεκάτων του δευτερολέπτου

// Κύρια συνάρτηση

```
void main()
```

 $\{$

```
init(); //ρουτίνα αρχικοποίησης με αρχικές τιμές
```

```
while (1){
```

```
if ((INPUT(PIN_D0)==0) && (start==0)){ //Αυτή η συνθήκη ισχύει στο ξεκίνημα όταν RD0=0
                                     // και start=0
```

```
start=1; // Στο ξεκίνημα γίνεται start=1 και δεν αλλάζει ποτέ.
```

}

```
else if(start == 1 && INPUT(PIN_D0) == 1){//Αυτό συμβαίνει όταν γίνει RD0=1
```

```
stop = 1; // Αν η μεταβλητή stop είναι 1
          // οι ενδείκτες δεν εμφανίζουν
          // την τρέχουσα τιμή χρόνου αλλά
          // εμφανίζουν την τιμή χρόνου κατά την
          // προηγούμενη στιγμή μετάβασης
          // του ακροδέκτη RDO από 0 σε 1
```

}

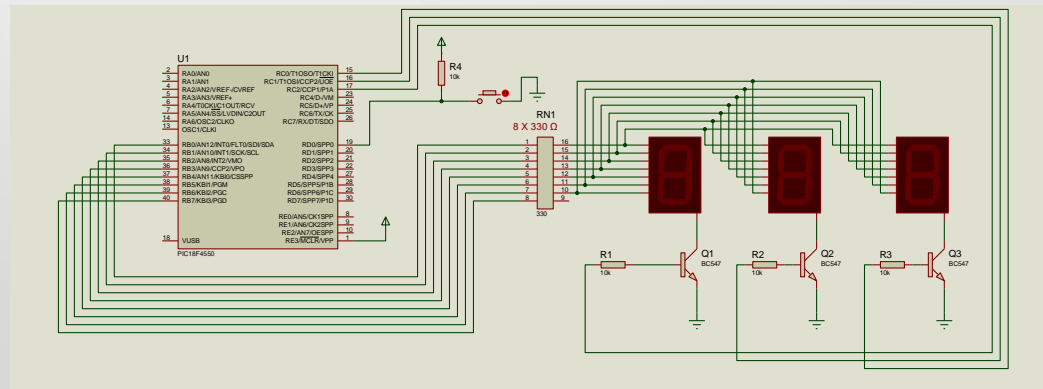
```
else { //Αυτό συμβαίνει όταν RD0=0
```

```
stop = 0;           // Οι ενδείκτες εμφανίζουν την
                    // τρέχουσα τιμή χρόνου.
```

}

```
// Κλείνει η while
```

```
} // Κλείνει η main
```



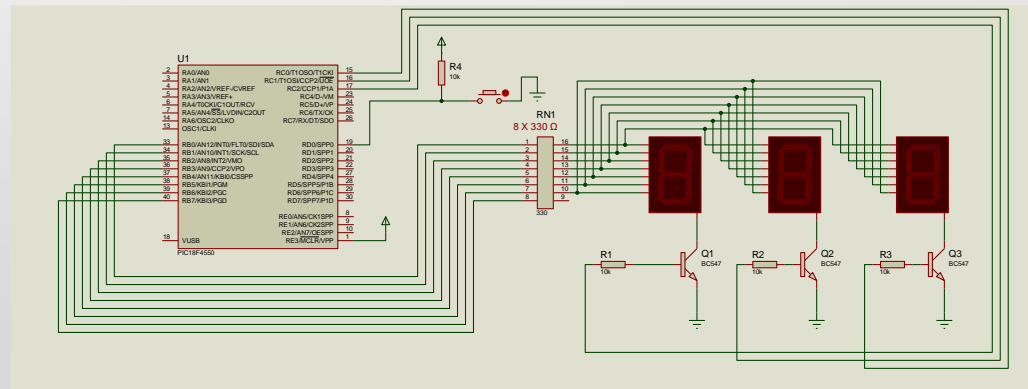
Η κύρια συνάρτηση και η συνάρτηση αρχικοποίησης της άσκησης 3c μετρητής δεκάτων του δευτερολέπτου

// Συνάρτηση αρχικοποίησης

void init(void){

```
    set_tris_b(0x00);    // Καθορισμός της πόρτας B ως έξοδος
    set_tris_c(0x00);    // Καθορισμός της πόρτας C ως έξοδος
    set_tris_d(0x0f);    // Καθορισμός της πόρτας C ως έξοδος
    PORTB = 0;           // Αρχική τιμή πόρτας B
    PORTC = 0;           // Αρχική τιμή πόρτας C
    counter = 20;        // Αρχική τιμή του counter
    tenthseconds = 0;    // Αρχική τιμή μετρητή δεκάτων δευτερολέπτου
    tenthseconds_stop = 0; // Αρχική τιμή της μεταβλητής tenthseconds_stop
    start=0; // Κατά την εκκίνηση start=0. Στη συνέχεια γίνεται 1 και παραμένει πάντα 1
    stop=0; // Αρχική τιμή μεταβλητής stop=0, δηλαδή ο μετρητής εμφανίζεται στους ενδείκτες.
    des =0; // Αρχική τιμή μεταβλητής που χρησιμοποιείται για επιλογή ενδείκτη από τη θύρα C
    SETUP_TIMER_0(TO_INTERNAL | TO_DIV_1 ); // Ρύθμιση prescaler=1
    set_timer0(5536);    // Αρχική τιμή του timer για διακοπές κάθε 5 ms
    enable_interrupts(INT_TIMER0); // Ενεργοποίηση της διακοπής του timer0
    enable_interrupts(GLOBAL); // Ενεργοποίηση του γενικού
    // διακόπτη των διακοπών
```

} // Κλείνει η αγκύλη της init()



Ρουτίνα διακοπών Άσκησης 3c. Μετρητής δεκάτων του δευτερολέπτου

#INT_TIMER0 HIGH // Διακοπή με μεγάλη προτεραιότητα από τον Timer0

void timer0_int(void){

int16 mon,dec,eka; // Μεταβλητές για αποθήκευση μονάδων, δεκάδων, εκατοντάδων
counter--; // Κάθε 20 * 5 msec = 0,1 sec

// Διακοπές συμβαίνουν κάθε 5 ms. Ο counter παίρνει την αρχική τιμή 20

// Κάθε 20 διακοπές έχουν περάσει 20X5ms=100ms=0,1 s.

if (counter == 0){ // κάθε φορά που μηδενίζεται ο counter έχουν περάσει 0,1 s

if (start == 1){ //Αν τρέχει ο μετρητής

tenthseconds++; // αυξάνεται κατά 1 κάθε 0,1 sec(=20 διακοπές)

}

counter = 20; //αν γίνουν 20 διακοπές, δηλαδή counter=0

//ο μετρητής διακοπών ξαναπαίρνει την τιμή 20

if (tenthseconds > 999){ // Ο μετρητής μετά την τιμή 999

tenthseconds = 0; // ξαναπαίρνει την τιμή 0.

}

}

if (stop == 0){

tenthseconds_stop = tenthseconds; // Όταν είναι stop=0(δηλαδή RD0=0)

//αποθηκεύεται συνεχώς η τρέχουσα

//τιμή του μετρητή στην

// μεταβλητή seconds_stop.

// Η ένδειξη του μετρητή θα τρέχει

// Όταν γίνει stop=1(δηλαδή RD0=1) η μεταβλητή

//tenthseconds κρατάει σταθερά την τιμή

//που είχε κατά την στιγμή της μετάβασης

//του ακροδέκτη RD0 από 0 σε 1

// Επομένως όταν RD0=1 η ένδειξη θα παραμένει

// παγωμένη(σταθερή)

}

eka = (int8) (tenthseconds_stop /100); //Υπολογισμός των εκατοντάδων του μετρητή

dec = (int8) ((tenthseconds_stop - (100*eka))/10); //Υπολογισμός των δεκάδων του

// μετρητή

mon = (int8) (tenthseconds_stop - (100 * eka) -(10 * dec)); //Υπολογισμός των μονάδων

// του μετρητή

set_timer0(5536); // Αρχική τιμή του timer0 για να έχουμε διακοπές κάθε 5 ms

des = ++des%3; //Μεταβλητή des παίρνει τις τιμές 0, 1, 2, 0, 1, 2, 0, 1, 2,...

//%3 σημαίνει modulo 3(υπόλοιπο διαίρεσης με το 3)

PORTC = dig[des]; //Η πόρτα C παίρνει τιμές 0000 0001, 0000 0010, 0000 0100,...

// δηλαδή τις τιμές 1, 2, 4, 1, 2, 4, ... από τον πίνακα dig

//Η πόρτα C επιλέγει σε ποιος ενδείκτης θα ανάψει.

// Εμφάνιση μονάδων, δεκάδων, εκατοντάδων στους ενδείκτες

if (des==0){ //PORTC=0000 0001(=dig[0]), επιλογή ενδείκτη μονάδων

PORTB = table[mon]; //Αν des=0 ενεργοποιείται ο ενδείκτης των μονάδων

//PORTC=0000 0001(=dig[0])

}

if (des==1){ //PORTC=0000 0010(=dig[1]), επιλογή ενδείκτη δεκάδων

PORTB = table[dec]; //Αν des=1 ενεργοποιείται ο ενδείκτης των δεκάδων

}

if (des==2){ //PORTC=0000 0100(=dig[2]), επιλογή ενδείκτη εκατοντάδων

PORTB = table[eka]; //Αν des=2 ενεργοποιείται ο ενδείκτης των εκατοντάδων

//PORTC=0000 0100(=dig[2])

}

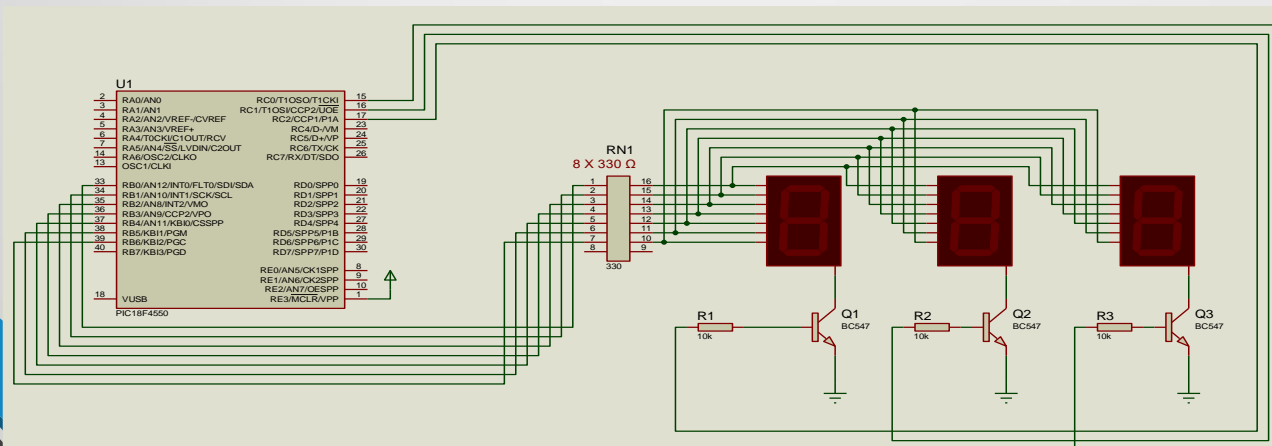
}

Άσκηση 3d. Δημιουργία ρολογιού πραγματικού χρόνου με ενδείκτες 7 τομέων

Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για το μικροελεγκτή PIC18F4550 με το οποίο οι 3 ενδείκτες 7 τομέων της αναπτυξιακής πλακέτας να λειτουργούν σαν ένα ρολόι πραγματικού χρόνου:

Το σύστημα θα λειτουργεί ως εξής:

- Αρχικά η ένδειξη θα είναι 12:00. Επειδή υπάρχουν 3 ενδείκτες, η ένδειξη θα γίνεται σε δύο φάσεις. Για ένα δευτερόλεπτο θα απεικονίζεται η ώρα και για ένα δευτερόλεπτο θα απεικονίζονται 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 στον αριστερότερο ενδείκτη να εμφανίζεται η ένδειξη Ω ενώ όταν η ένδειξη είναι τα λεπτά στον αριστερότερο ενδείκτη να υπάρχει η ένδειξη Π.
- Να χρησιμοποιηθεί η μέθοδος της διακοπής από την υπερχείλιση του timer0 για τη μέτρηση του χρόνου και οι πίνακες μετατροπής για την οδήγηση των ενδεικτών.



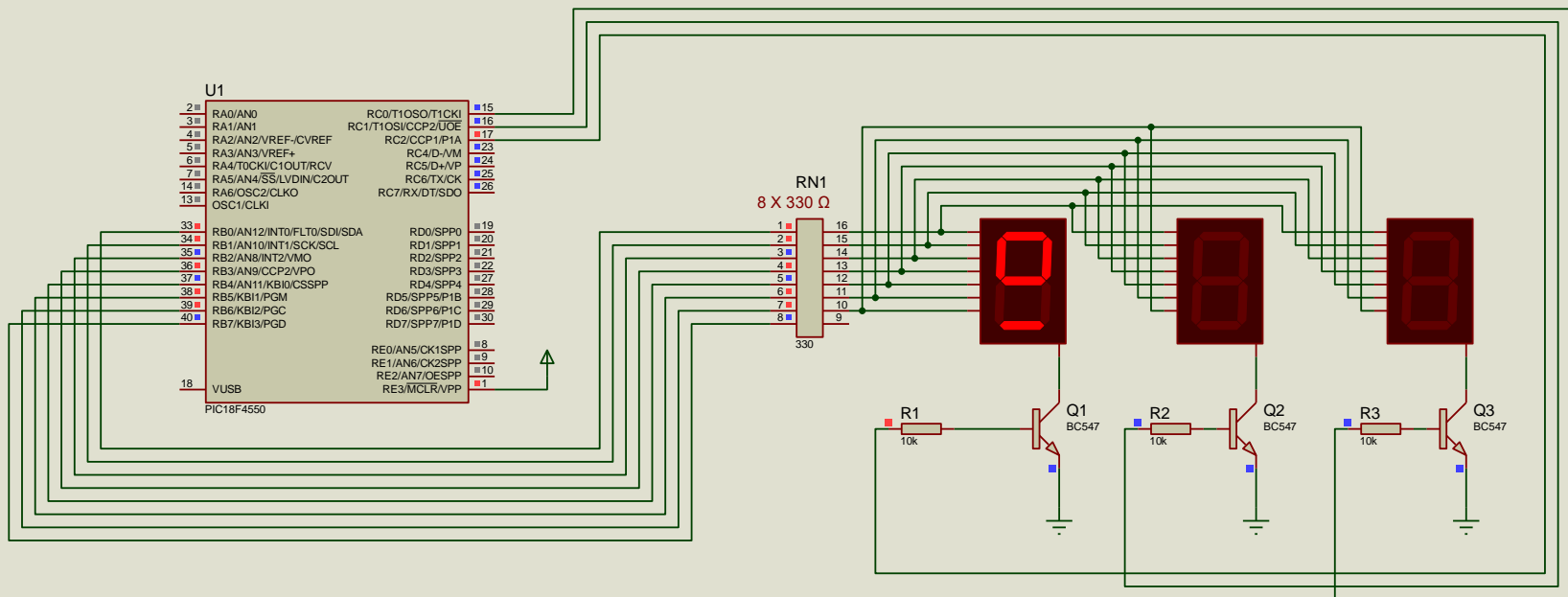
Για ένα δευτερόλεπτο εμφανίζεται η ώρα στους δύο δεξιά ενδείκτες

Αρχικά η ένδειξη θα είναι 12:00

Για ένα δευτερόλεπτο θα απεικονίζεται η ώρα και
για ένα δευτερόλεπτο θα απεικονίζονται τα λεπτά.

Όταν η ένδειξη στους δύο δεξιότερους ενδείκτες είναι η ώρα στον αριστερότερο ενδείκτη εμφανίζεται η ένδειξη Ω

Όταν η ένδειξη είναι τα λεπτά στον αριστερότερο ενδείκτη να υπάρχει η ένδειξη Π



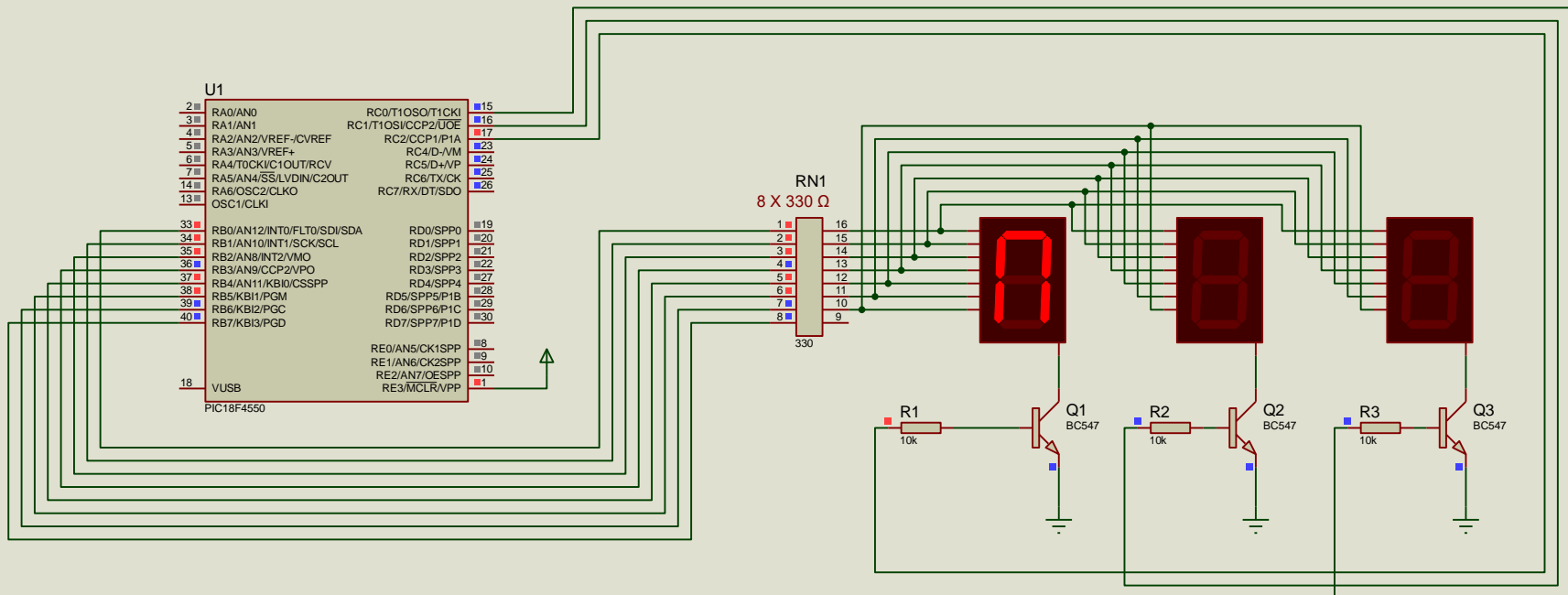
Για ένα δευτερόλεπτο εμφανίζονται τα λεπτά στους δύο δεξιά ενδείκτες

Αρχικά η ένδειξη θα είναι 12:00

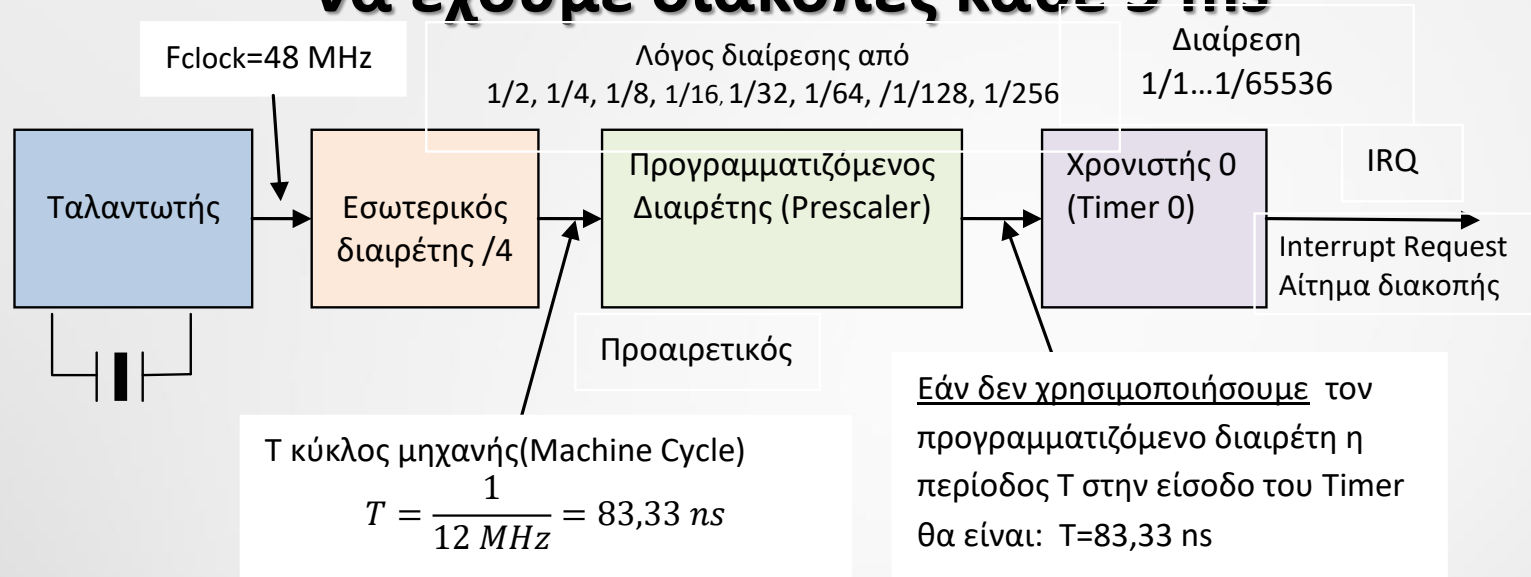
Για ένα δευτερόλεπτο θα απεικονίζεται η ώρα και για ένα δευτερόλεπτο θα απεικονίζονται τα λεπτά.

Όταν η ένδειξη στους δύο δεξιότερους ενδείκτες είναι η ώρα στον αριστερότερο ενδείκτη εμφανίζεται η ένδειξη Ω

Όταν η ένδειξη είναι τα λεπτά στον αριστερότερο ενδείκτη να υπάρχει η ένδειξη Π



Υπολογισμός της αρχικής τιμής του timer0 έτσι ώστε να έχουμε διακοπές κάθε 5 ms



`SETUP_TIMER_0(TO_INTERNAL | TO_DIV_1); // Ρύθμιση prescaler=1`

$$83,33 \text{ ns}(65536 - \text{αρχική τιμή του timer0}) = 5\,000\,000 \text{ ns} \Leftrightarrow$$

$$65536 - \text{αρχική τιμή του timer0} = \frac{5\,000\,000}{83,33} \Leftrightarrow$$

$$65536 - \text{αρχική τιμή του timer0} = 60\,000 \Leftrightarrow$$

$$\text{αρχική τιμή του timer0} = 60\,000 - 65536 \Leftrightarrow$$

$$\text{αρχική τιμή του timer0} = 5536$$

`set_timer0(5536); // Αρχική τιμή του μετρητή για να συμβαίνουν`

`// διακοπές κάθε 5 ms. Θα μπει μέσα στη ρουτίνα διακοπών.`

`// Στην αρχή.`

Διάγραμμα ροής του Προγράμματος ρολογιού

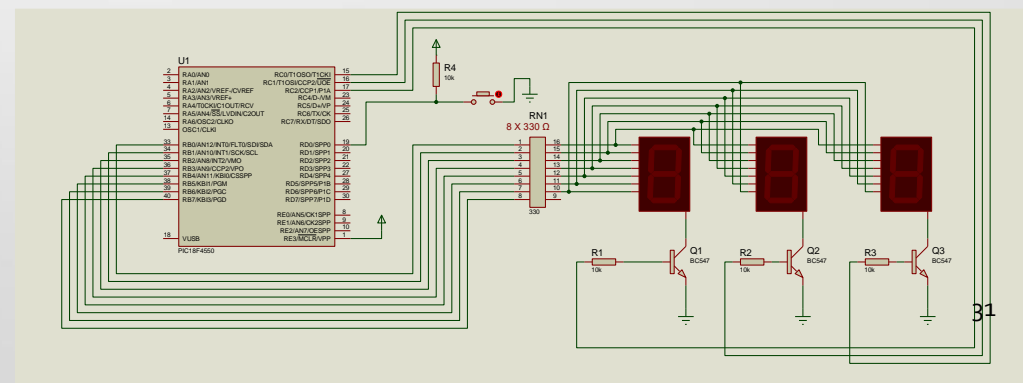
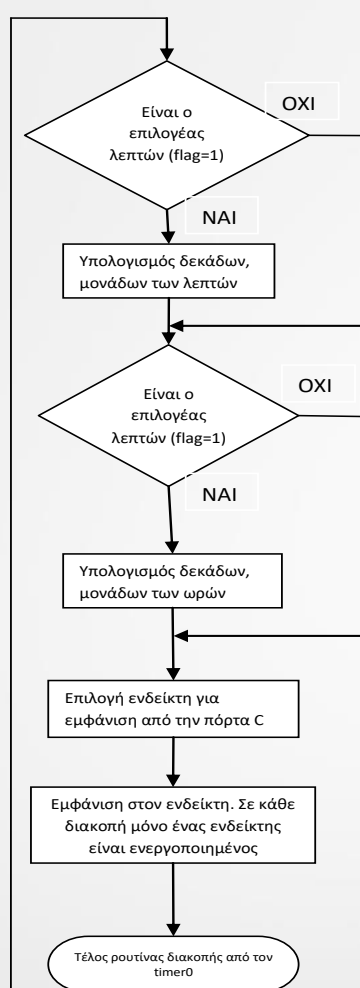
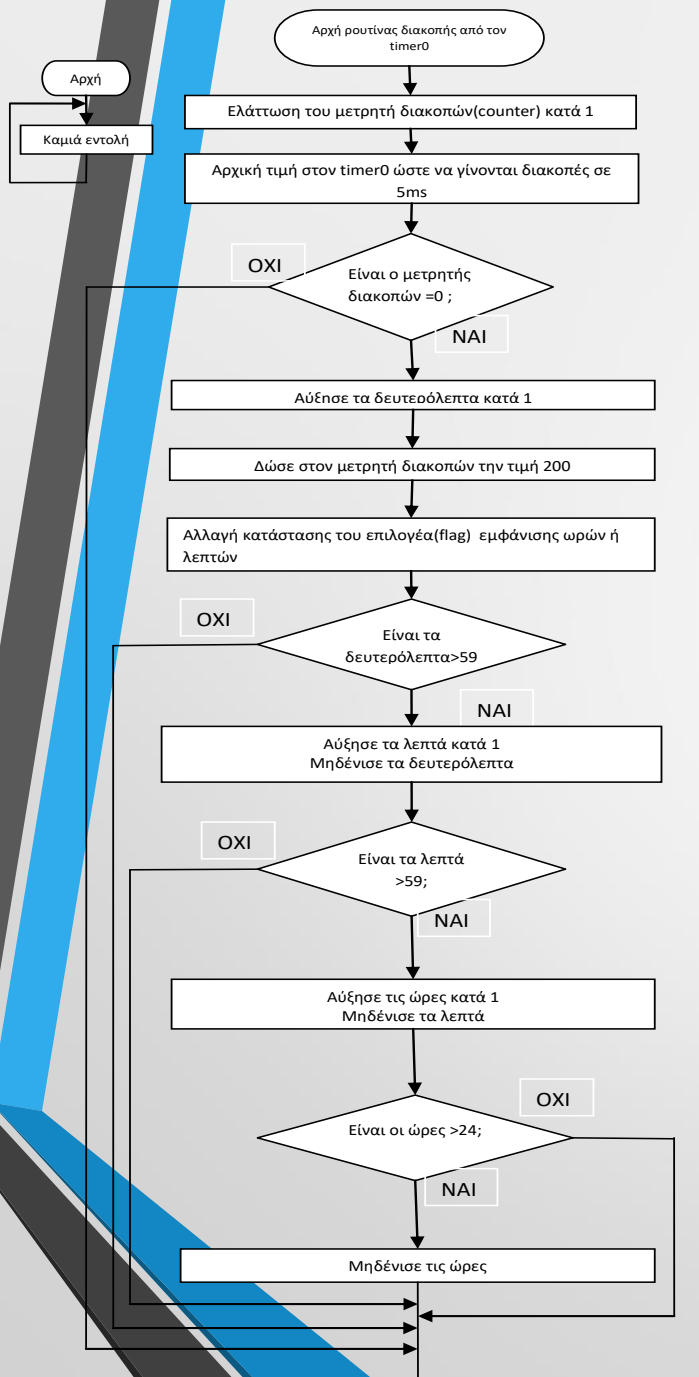
Διακοπές από τον timer0 κάθε 5 ms

Κάθε 200 διακοπές αυξάνει το περιεχόμενο της μεταβλητής **seconds** κατά 1

Μετά την τιμή 59 η μεταβλητή **seconds** παίρνει την τιμή 0 και αυξάνεται το περιεχόμενο της τιμής **minute** κατά 1

Μετά την τιμή 59 η μεταβλητή **minute** παίρνει την τιμή 0 και αυξάνεται το περιεχόμενο της τιμής **hour** κατά 1

Μετά την τιμή 23 η μεταβλητή **hour** παίρνει την τιμή 0



Βασικές ιδέες για την κατασκευή του ρολογιού

Θα υπάρχουν 3 μεταβλητές: **seconds**, **minute**, **hour** για να μετράνε δευτερόλεπτα, λεπτά και ώρες

1. Θα υπάρχει μια μεταβλητή **seconds** της οποίας η τιμή θα αυξάνεται κατά 1 κάθε δευτερόλεπτο.
Μπορούμε να ρυθμίσουμε να συμβαίνουν διακοπές κάθε 5 ms. Κάθε 200 διακοπές ($=200 \times 5 \text{ ms} = 1000 \text{ ms} = 1 \text{ s}$) θα αυξάνεται το περιεχόμενο της μεταβλητής **seconds** κατά 1.
2. Θα υπάρχει μια μεταβλητή **minute** με την οποία θα μετράμε τα λεπτά της ώρας.
Μετά την τιμή 59 η μεταβλητή **seconds** αντί να πάρει την τιμή 60 θα παίρνει την τιμή 0 και θα αυξάνεται η τιμή της μεταβλητής **minute** (η οποία μετράει τα λεπτά) κατά 1
3. Θα υπάρχει μια μεταβλητή **hour** με την οποία θα μετράμε τις ώρες.
Μετά την τιμή 59 η μεταβλητή **minute** αντί να πάρει την τιμή 60 θα παίρνει την τιμή 0 και θα αυξάνεται η τιμή της μεταβλητής **hour** (η οποία μετράει τις ώρες) κατά 1.
Μετά την τιμή 23 η μεταβλητή **hour** θα παίρνει την τιμή 0.

Τα παραπάνω διότι:

1 λεπτό της ώρας = 60 δευτερόλεπτα

1 ώρα = 60 λεπτά

Για 1 δευτερόλεπτο θα εμφανίζεται στους ενδείκτες τη τιμή της μεταβλητής **minute**

Για 1 δευτερόλεπτο θα εμφανίζεται στους ενδείκτες η τιμή της μεταβλητής **hour**

Εναλλαγή της εμφάνισης των ωρών ή των λεπτών κάθε 1 δευτερόλεπτο με τη βοήθεια της μεταβλητής **flag**

Η μεταβλητή **flag** θα αλλάζει τιμή από 0 σε 1 και από 1 σε 0 κάθε 1 δευτερόλεπτο.

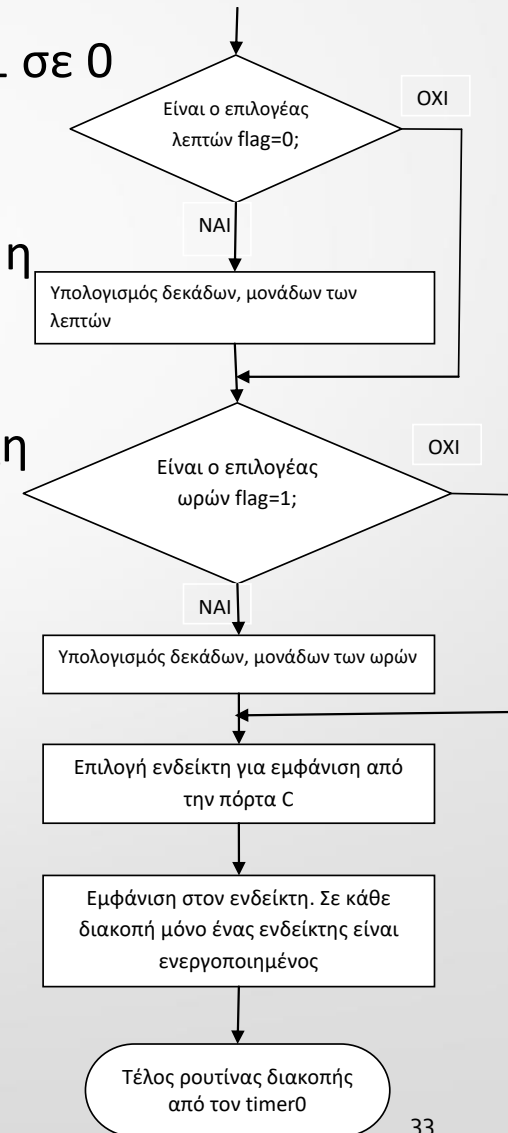
Κάθε 200 διακοπές ($=200 \times 5 \text{ ms} = 1 \text{ sec}$) θα εκτελείται η εντολή **flag = flag ^ 1**

Υπενθύμιση: \wedge είναι το σύμβολο για τη λογική πράξη του “Αποκλειστικού Η” (**Exclusive OR**)

Εάν **flag=0** θα υπολογίζονται οι δεκάδες και μονάδες της μεταβλητής **minute**

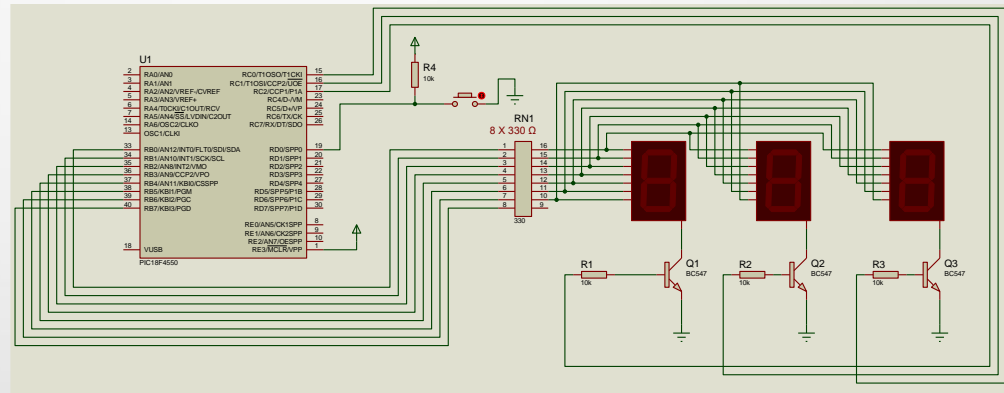
Εάν **flag= 1** θα υπολογίζονται οι δεκάδες και μονάδες της μεταβλητής **hour**

Θα εμφανίζονται στους 2 δεξιά ενδείκτες οι δεκάδες και μονάδες της μεταβλητής **minute** ή της μεταβλητής **hour**



Δηλώσεις πάνω από τη main(){ }

```
#include <main.h>
#byte PORTA    =0xF80 //ορισμός των θυρών με την θέση τους στην μνήμη του
                  //μικροελεγκτή
#byte PORTB    =0xF81
#byte PORTC    =0xF82
// Ορισμοί μεταβλητών
int8 des=0; //Μεταβλητή για επιλογή ενδείκτη 7 τομέων από τον πίνακα int8 dig[3]
μέσω
        // της πόρτας C
int8 seconds=0; //Μεταβλητή για την μέτρηση των δευτερολέπτων
int8 minute=0; //Μεταβλητή για την μέτρηση των ωρών
int8 hour=12; //Μεταβλητή για την μέτρηση ωρών
int8 counter=200; //Μεταβλητή για την μέτρηση διακοπών
int1 flag=0; //Μεταβλητή για επιλογή εμφάνισης ωρών ή λεπτών
        //Πίνακας για εμφάνιση των ψηφίων στους ενδείκτες
int8 table[16] = { 0b00111111, //0
                  0b00000110, //1
                  0b01011011, //2
                  0b01001111, //3
                  0b01100110, //4
                  0b01101101, //5
                  0b01111101, //6
                  0b00000111, //7
                  0b01111111, //8
                  0b01101111, //9
                  0b01101011, //Ω Σύμβολο για την ώρα
                  0b00110111, //Π Σύμβολο για τα λεπτά
                  0b00111001,
                  0b01011110,
                  0b01111001,
                  0b01110001};
int8 dig[3] = {1,2,4}; //Πίνακας για ενεργοποίηση ενός μόνο ενδείκτη από την
        //πόρτα C. Η πόρτα C εφαρμόζει 5V στην βάση ενός μόνον
        //από τα τρία τρανζίστορ που συνδέουν την κοινή κάθοδο των
        //ενδεικτών προς την γη.
        //1→0000 0001, 2→0000 0010, 4→0000 0100
```



```
// Δήλωση συναρτήσεων
void timer0_int(void);
void init (void);
```


Κύρια συνάρτηση main(){ } και ρουτίνα αρχικοποίησης init(){ }

// κύρια συνάρτηση

```
void main()
```

```
{
```

```
    init();
```

```
    while (1){          // Το κύριο πρόγραμμα δεν κάνει τίποτα.
```

```
                        // Περιμένει να συμβεί μια διακοπή
```

```
    }
```

```
}
```

// Ρουτίνα αρχικοποίησης

```
void init (void){
```

```
    set_tris_b(0x00);    // Καθορισμός της πόρτας B ως εξόδου
```

```
    set_tris_c(0x00);    // Καθορισμός της πόρτας C ως εξόδου
```

```
    PORTB = 0;           // αρχική τιμή της πόρτας B
```

```
    PORTC = 0;           // αρχική τιμή της πόρτας C
```

```
    counter = 200;       // Αρχική τιμή του counter
```

```
    seconds = 0;         // Αρχική τιμή του μετρητή δευτερολέπτων
```

```
    minute = 0;          // Αρχική τιμή του μετρητή λεπτών
```

```
    hour = 12;           // Αρχική τιμή του μετρητή ωρών
```

```
    des = 0;             // Αρχική τιμή της μεταβλητής για επιλογή ενδείκτη
```

```
    flag = 0;            // Αρχική τιμή της μεταβλητής για εμφάνιση ώρας ή λεπτών
```

```
    SETUP_TIMER_0(TO_INTERNAL | TO_DIV_1 ); // Ρύθμιση prescaler=1
```

```
    set_timer0(5536);     // Αρχική τιμή του timer0
```

```
    enable_interrupts(INT_TIMER0);    // Ενεργοποίηση
```

```
                                // διακοπών από τον timer0
```

```
    enable_interrupts(GLOBAL);    // Ενεργοποίηση του γενικού
```

```
                                // διακόπτη των διακοπών
```

```
}
```

Η ρουτίνα διακοπών από τον timer0 (εκτελείται κάθε 5 ms)

// Ρουτίνα διακοπών

#INT_TIMER0 HIGH // Διακοπή με μεγάλη προτεραιότητα

void timer0_int(void){

int16 mon,dec,eka; //μεταβλητές για εμφάνιση ψηφίων στους ενδείκτες 7 τομέων

set_timer0(5536); // αρχική τιμή του μετρητή για να συμβαίνουν

//διακοπές κάθε 5 ms

counter--; //Ο μετρητής ελαττώνεται κατά 1 και μηδινίζεται

// κάθε 200 * 5 msec = 1 sec

if (counter == 0){

seconds++; // Αυξάνεται ο μετρητής δευτερολέπτων κατά 1 κάθε

// 1 δευτερόλεπτο

counter = 200; // Αρχική τιμή του μετρητή διακοπών ώστε να μηδενίζεται

// μετά από 200 διακοπές δηλαδή μετά από 1 δευτερόλεπτο

//διότι 200 * 5 msec = 1 sec

flag^=1; // Σημαία που χρησιμοποιείται για να εμφανιστούν . Ισοδυναμεί με το flag=flag^1.

// Σημαία που χρησιμοποιείται για να εμφανιστούν οι ώρες ή τα λεπτά

if (seconds > 59){

seconds = 0; // μετά την τιμή 59 ο μετρητής δευτερολέπτων παίρνει

// την τιμή 0

minute++; //Αν περάσουν 60 δευτερόλεπτα αυξάνει ο μετρητής

// λεπτών κατά 1

if (minute > 59){

minute = 0;

hour++; //Αν περάσουν 60 λεπτά αυξάνει ο

// μετρητής ωρών κατά 1

}

if (hour > 24)

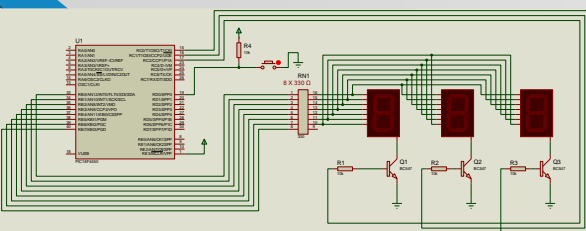
hour = 0; //Αν περάσουν 24 ώρες ο μετρητής ωρών

// γίνεται 0

}

}

}



if (flag == 0){

//αν το flag=0 τότε εμφανίζονται τα λεπτά

dec = (int8)minute / 10; //Υπολογίζονται οι δεκάδες των λεπτών

mon = minute - dec * 10; //Υπολογίζονται οι μονάδες των λεπτών

eka = 11; // Στο 11° στοιχείο το πίνακα int8 table[16] αντιστοιχεί το

// σύμβολο για τα λεπτά «Π»

// Το σύμβολο αυτό θα εμφανιστεί στον πρώτο από αριστερά

// ενδείκτη(των εκατοντάδων)

}

if (flag == 1){

//Αν το flag είναι 1 τότε εμφανίζονται οι ώρες

dec = (int8)hour / 10; // υπολογισμός των δεκάδων των ωρών

mon = hour - dec * 10; // υπολογισμός των μονάδων των ωρών

eka = 10; //Στο 10° στοιχείο το πίνακα int8 table[16] αντιστοιχεί το

// σύμβολο για την ώρα, «Ω»

// Το σύμβολο αυτό θα εμφανιστεί στον πρώτο από αριστερά

// ενδείκτη(των εκατοντάδων)

}

des = ++des%3; //Η μεταβλητή αυτή παίρνει διαδοχικά τις τιμές 0, 1, 2, 0, 1, 2, 0, 1, ...

PORTC = dig[des]; // ώστε να επιλέγονται από τον πίνακα dig[des] διαδοχικά οι τιμές

//0000 0001, 0000 0010, 0000 0100,0000 0001, 0000 0010, 0000 0100, ...

// και να ενεργοποιούνται με την σειρά οι ενδείκτες 7 τομέων

if (des==0){

PORTB = table[mon]; // εμφανίζονται οι μονάδες της ώρας ή των λεπτών

}

if (des==1){

PORTB = table[dec]; // εμφανίζονται οι μονάδες της ώρας ή των λεπτών

}

if (des==2){

PORTB = table[eka]; // εμφανίζεται ή ένδειξη ώρας «Ω» ή λεπτών «Π»

}

}

Η πόρτα C παίρνει κάθε 5 ms διαδοχικά τις τιμές:

0000 0001

0000 0010

0000 0100

0000 0001

.....

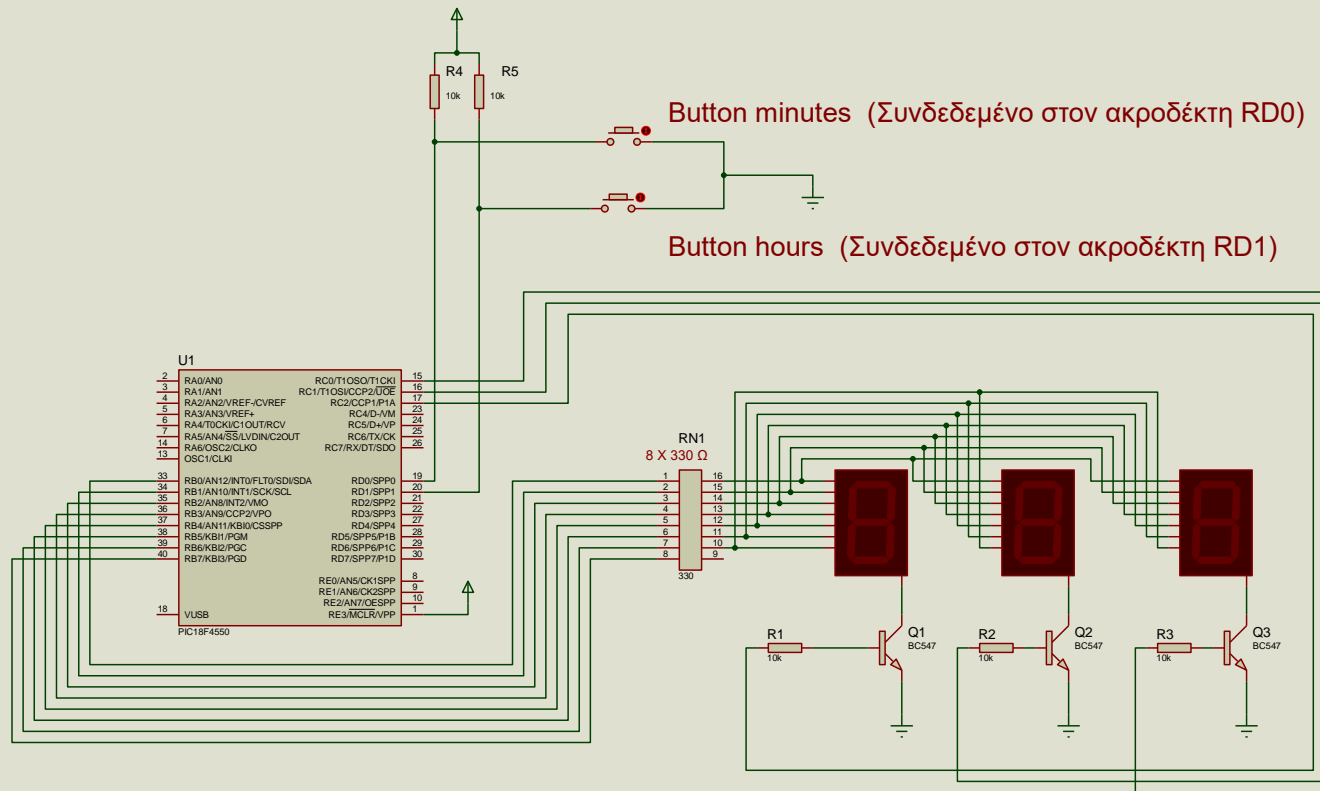
des	++des	++des%3
0	1	1
1	2	2
2	3	0
0	1	1
1	2	2
2	3	0
0	1	1

Τροποποίηση του προγράμματος ώστε να μπορούμε να ρυθμίσουμε την ώρα

Ρύθμιση ώρας

Όταν πατήσουμε το μπουτόν button minutes η μεταβλητή minute (και η αντίστοιχη ένδειξη) θα αυξάνεται κατά 1 κάθε δευτερόλεπτο

Όταν πατήσουμε το μπουτόν button hours η μεταβλητή hour (και η αντίστοιχη ένδειξη) θα αυξάνεται κατά 1 κάθε δευτερόλεπτο



Στην κατάλληλη θέση μέσα στη ρουτίνα διακοπών θα βάλουμε αυτές τις δύο εντολές
Μπορούμε επίσης να σταματήσουμε την αύξηση της μεταβλητής seconds όταν πατιέται τουλάχιστον ένα από τα μπουτόν.

```
if (input(PIN_D0)==0) {minute++;}  
if (input(PIN_D1)==0) {hour++;}
```