Άσκηση όγδοη του Εργαστηρίου

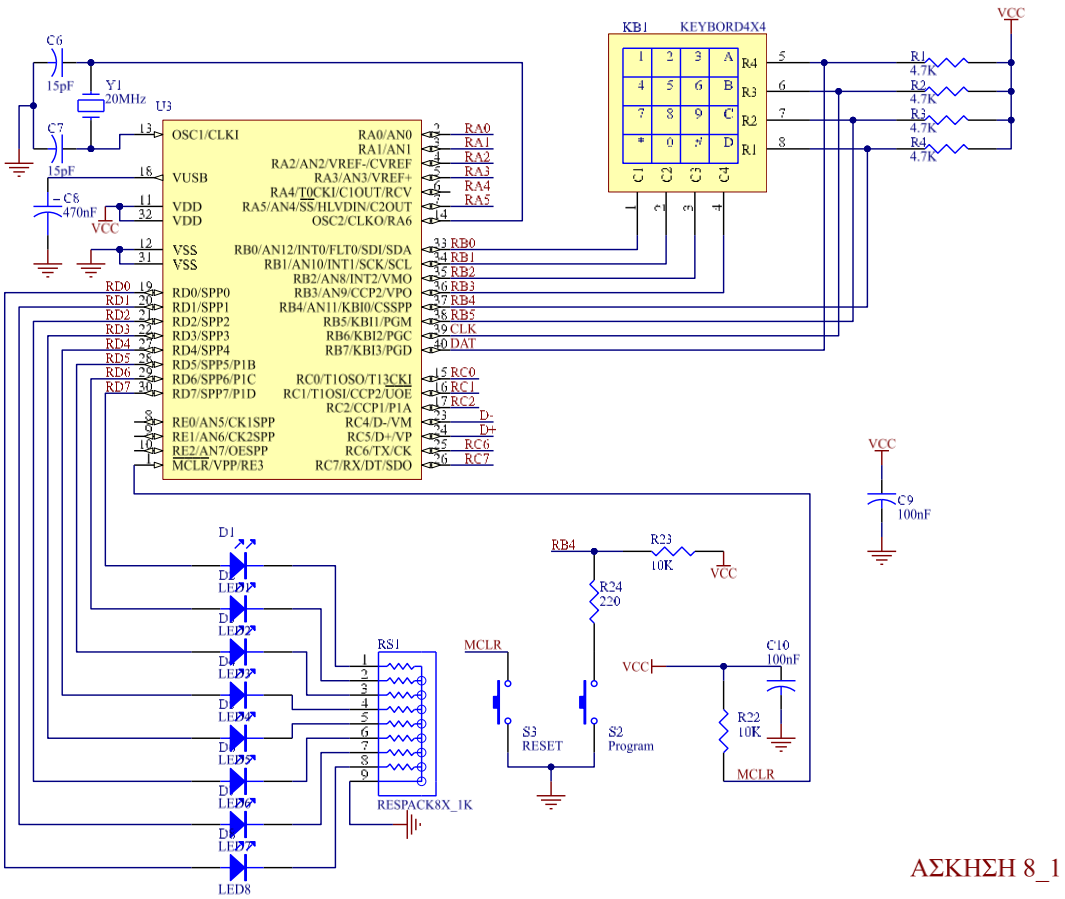
Στην όγδοη άσκηση εξετάζεται ο προγραμματισμός του μικροελεγκτή PIC18F4550 με τη προσθήκη πληκτρολογίου 16 πλήκτρων (4x4) και οθόνης LCD. Το πληκτρολόγιο ελέγχεται με τη μέθοδο polling.

Στη μέθοδο αυτή ένα λογικό 0 μετακινείται μεταξύ των στηλών των πλήκτρων και διαβάζονται οι γραμμές που είναι η επιστροφή των δεδομένων. Οι στήλες είναι έξοδοι του PIC και οι γραμμές είσοδοι. Αν δεν έχει πατηθεί κανένα πλήκτρο τότε όλες οι γραμμές είναι σε λογικό 1 μέσω εξωτερικών pull-up αντιστάσεων που υπάρχουν στην αναπτυξιακή πλακέτα. Η μεταβλητή k που επιστρέφει στη περίπτωση αυτή το πρόγραμμα έχει τιμή μηδέν. Όταν πατηθεί κάποιο πλήκτρο, τότε μια στήλη που είναι σε λογικό μηδέν μέσω του πλήκτρου που πατήθηκε, συνδέεται με μία γραμμή ,από όπου ο μικροελεγκτής διαβάζει σε αυτή τη γραμμή, το λογικό μηδέν και το πρόγραμμα επιστρέφει στη μεταβλητή k τη τιμή 1 και στη μεταβλητή kchar τη τιμή του πλήκτρου που πατήθηκε. Επειδή η τιμή αυτή δεν αντιστοιχεί στην αναγραφόμενη τιμή του πλήκτρου, χρησιμοποιείται ένα πίνακας μετατροπής για να πάρει ο χρήστης την επιθυμητή τιμή που είναι ο ASCII χαρακτήρας του πλήκτρου που πατήθηκε. Στο πρόγραμμα εφαρμογής χρησιμοποιείται μια έτοιμη συνάρτηση του CCS C compiler, και επαναληπτικά ελέγχεται η μεταβλητή k που επιστρέφει το πρόγραμμα (μέθοδος polling) και όταν η μεταβλητή αυτή γίνει 1 διαβάζεται η επιστρεφόμενη τιμή του πλήκτρου.

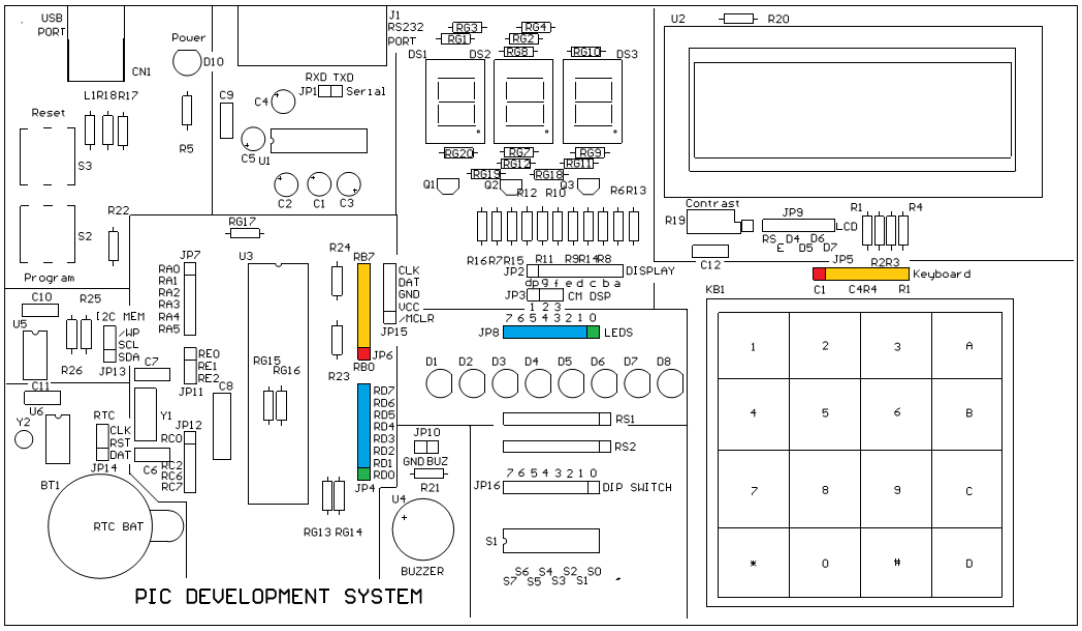
Παράδειγμα 1.

Στην αναπτυξιακή πλακέτα του εργαστηρίου συνδέουμε το πληκτρολόγιο 4X4 με το PIC18F4550 μέσω της πόρτας B. Κάθε φορά που πιέζεται κάποιο από τα 16 πλήκτρα ο αριθμός του πλήκτρου εμφανίζεται στα leds που είναι συνδεδεμένα στη πόρτα D. Στο πρόγραμμα αυτό χρησιμοποιούνται η έτοιμες συναρτήσεις του compiler CCS.

Το σχηματικό του κυκλώματος



Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

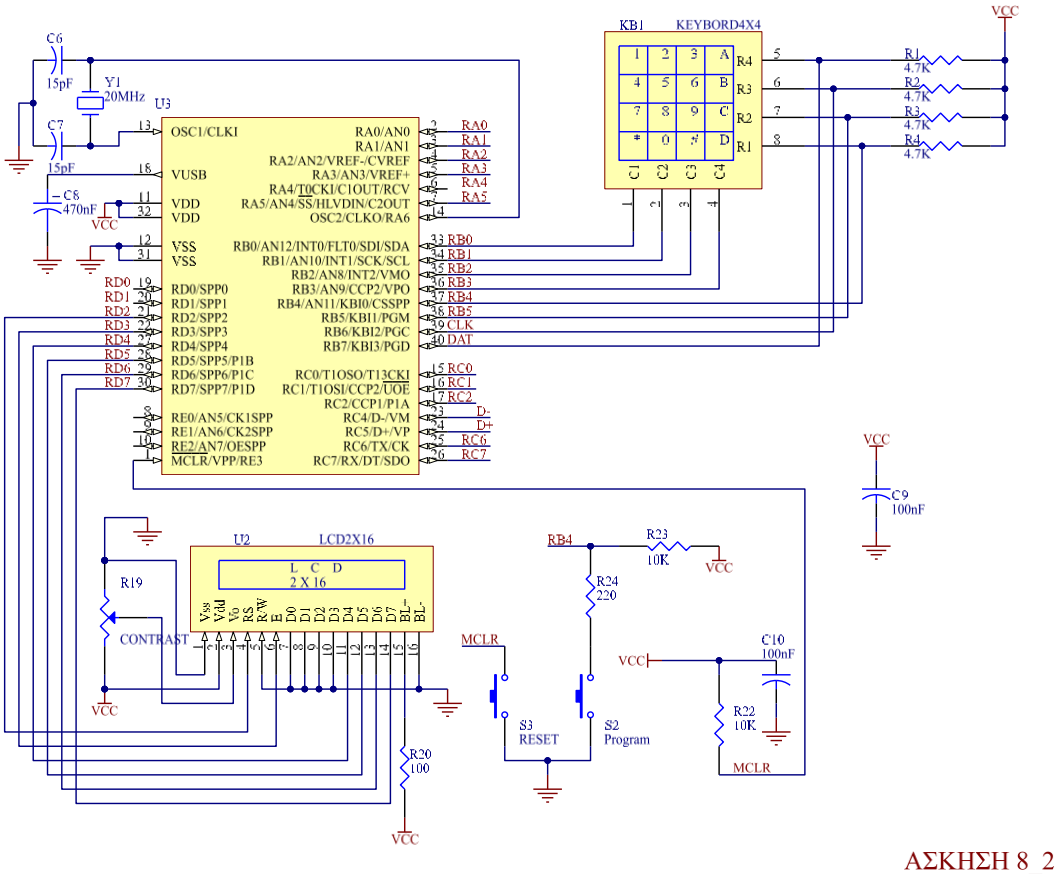
```
#include <main.h>
#include <keypad.h>
#byte PORTB=0xf81
#byte PORTD=0xf83


//-----------------Function Definition----------------------------
void init (void);
//----------------------------------------------------------------
void main()
{
char k;
init();
kbd_init();

while(TRUE)
  {
   k=kbd_getc();
   if(k!=0)                          // Check if key press
     {
      PORTD = k & 0x0f;
     }
  }

}
//---------------------------------------------------------
//-----------------Function declaration--------------
//---------------------------------------------------------
```

```
void init (void){
set_tris_d(0x00);
PORTD = 0;
}
//-----------------------------------------------------------
```

Παράδειγμα 2.

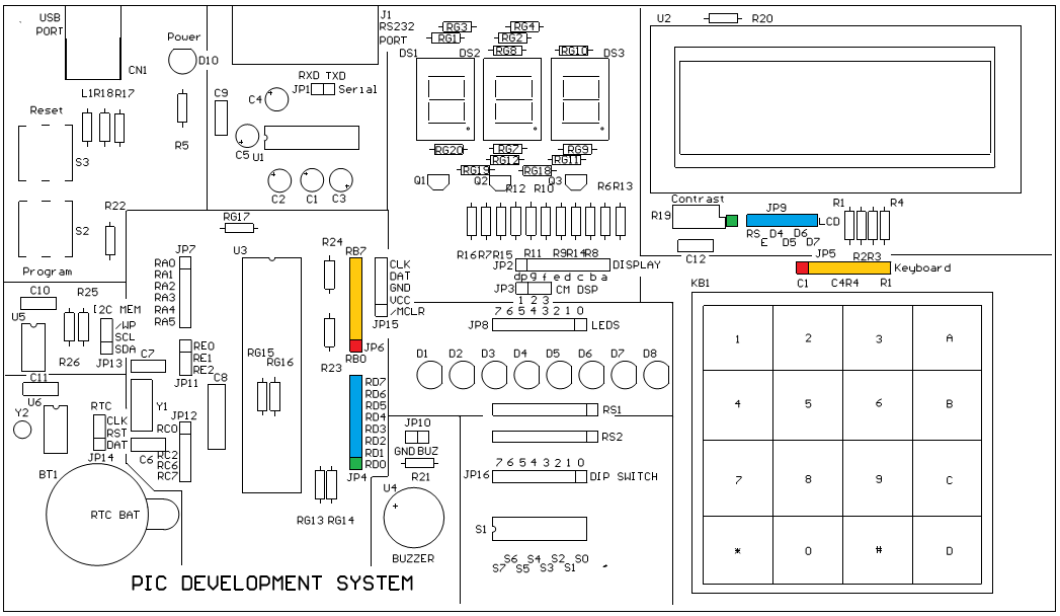
Στην αναπτυξιακή πλακέτα του εργαστηρίου συνδέουμε το πληκτρολόγιο 4X4 με το PIC18F4550 μέσω της πόρτας B και την οθόνη στη πόρτα D. Το πρόγραμμα περιμένει να εισαχθούν δύο αριθμοί του ενός ψηφίου από το πληκτρολόγιο και δείχνει ένα-ένα τους αριθμούς και το άθροισμα τους στην οθόνη Στο πρόγραμμα χρησιμοποιούνται η έτοιμες συναρτήσεις του CCS compiler.

Το σχηματικό του κυκλώματος



ΑΣΚΗΣΗ 8_2

Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

```
#include <main.h>
#include <keypad.h>
#include <flex_lcd.h>
#byte PORTB=0xf81
#byte PORTD=0xf83

//-------------------------------------------------
char state=1;
char N1=0;
char N2=0;
//------------------Function Definition-------------------------------
void init (void);
//--------------------------------------------------------------------
void main()
{
char k;
init();
lcd_init();
kbd_init();

  while(TRUE)
    {
     switch(state){
  case 1:
          k=kbd_getc();
         while (k!=0)
           {

            N1= k & 0b00001111;
```

```
            printf (lcd_putc, "%1d+", N1 );
             state = 2;
            break;
              }
        break;
   case 2:
        k=kbd_getc();
          while (k!=0)
          {
           N2 = k & 0b00001111;
           printf (lcd_putc, "%1d=", N2 );

                    N2 = N2 + N1;
          printf (lcd_putc, "%2d", N2 );
          N1=0;
           N2=0;
           state = 1;
              delay_ms(2000);
              printf (lcd_putc, "\f" );
          break;
         }

        }

     }
}
//-----------------------------------------------------------
//-----------------Function declaration----------

//-----------------------------------------------------------
void init (void){
set_tris_d(0x00);
}
//-----------------------------------------------------------
```

Οι συναρτήσεις που χρησιμοποιούνται στα παραπάνω προγράμματα

Η συνάρτηση keypad.h

```
//Keypad connection:
#define row0 PIN_B4
#define row1 PIN_B5
#define row2 PIN_B6
#define row3 PIN_B7
#define col0 PIN_B0
#define col1 PIN_B1
#define col2 PIN_B2
#define col3 PIN_B3
```

```
//------------------- Variable definition--------------------------
char const KEYS[4][4] =
{{'1','2','3','A'},
 {'4','5','6','B'},
 {'7','8','9','C'},
 {'*','0','#','D'}};


#define KBD_DEBOUNCE_FACTOR 33 // Set this number to apx n/333 where
// n is the number of times you expect
// to call kbd_getc each second

//------------------Function Definition----------------------------
short int ALL_ROWS (void);
void kbd_init(void);
char kbd_getc(void);
//-----------------------------------------------------------------
void kbd_init(void)
{
set_tris_b(0xF0);
output_b(0x0F);
}
//-----------------------------------------------------------
short int ALL_ROWS (void)
{
if(input (row0) & input (row1) & input (row2) & input (row3))
return (0);
else
return (1);
}

//-----------------------------------------------------------

char kbd_getc(void)
{
static byte kbd_call_count;
static short int kbd_down;
static char last_key;
static byte col;

byte kchar;
byte row;

kchar='\0';

if(++kbd_call_count>KBD_DEBOUNCE_FACTOR)
  {
     switch (col)
           {
            case 0:
```

```
        output_low(col0);
      output_high(col1);
      output_high(col2);
      output_high(col3);
      break;

      case 1:
      output_high(col0);
      output_low(col1);
      output_high(col2);
      output_high(col3);
      break;

       case 2:
      output_high(col0);
        output_high(col1);
      output_low(col2);
      output_high(col3);
      break;

       case 3:
      output_high(col0);
      output_high(col1);
      output_high(col2);
      output_low(col3);
      break;
       }

    if(kbd_down)
        {
         if(!ALL_ROWS())
             {
             kbd_down=false;
             kchar=last_key;
              last_key='\0';
               }
        }
else
    {
     if(ALL_ROWS())
      {
       if(!input (row0))
            row=0;
       else if(!input (row1))
            row=1;
       else if(!input (row2))
           row=2;
       else if(!input (row3))
            row=3;
```

```
                last_key =KEYS[row][col];
                kbd_down = true;
               }
            else
                  {
              ++col;
              if(col==4)
                   col=0;
                  }
             }
            kbd_call_count=0;
     }
         return(kchar);
}

//-------------------------------------------------
//-------------------------------------------------
//-------------------------------------------------
```

Η συνάρτηση flex_lcd.h

```
//-------------------------------------------------
#define LCD_DB4   PIN_D4
#define LCD_DB5   PIN_D5
#define LCD_DB6   PIN_D6
#define LCD_DB7   PIN_D7

#define LCD_E      PIN_D3
#define LCD_RS     PIN_D2
#define LCD_RW     PIN_D1
#define lcd_type 2        // 0=5x7, 1=5x10, 2=2 lines
#define lcd_line_two 0x40 // LCD RAM address for the 2nd line
int8 const LCD_INIT_STRING[4] =
{
 0x20 | (lcd_type << 2), // Func set: 4-bit, 2 lines, 5x8 dots
 0xc,                    // Display on
 1,                      // Clear display
 6                       // Increment cursor
 };
//=================================================================
==
void lcd_send_nibble(int8 nibble);
void lcd_send_byte(int8 address, int8 n);
void lcd_init(void);
void lcd_gotoxy(int8 x, int8 y);
void lcd_putc(char c);
```

```
//=============================================================
==
//-------------------------------------------------------------
void lcd_send_nibble(int8 nibble)
{
// Note:  !! converts an integer expression
// to a boolean (1 or 0).
 output_bit(LCD_DB4, !!(nibble & 1));
 output_bit(LCD_DB5, !!(nibble & 2));
 output_bit(LCD_DB6, !!(nibble & 4));
 output_bit(LCD_DB7, !!(nibble & 8));

 delay_cycles(20);
 output_high(LCD_E);
 delay_us(50);
 output_low(LCD_E);
}

//-------------------------------------
// This sub-routine is only called by lcd_read_byte().
// It's not a stand-alone routine.  For example, the
// R/W signal is set high by lcd_read_byte() before
// this routine is called.


//----------------------------------------
// Send a byte to the LCD.
void lcd_send_byte(int8 address, int8 n)
{
output_low(LCD_RS);

if(address)
   output_high(LCD_RS);
else
   output_low(LCD_RS);

 delay_cycles(10);

output_low(LCD_E);

lcd_send_nibble(n >> 4);
lcd_send_nibble(n & 0xf);
}

//----------------------------
void lcd_init(void)
{
int8 i;

output_low(LCD_RS);
```

```
output_low(LCD_E);

delay_ms(200);

for(i=0 ;i < 3; i++)
   {
    lcd_send_nibble(0x03);
    delay_ms(10);
   }

lcd_send_nibble(0x02);

for(i=0; i < sizeof(LCD_INIT_STRING); i++)
   {
    lcd_send_byte(0, LCD_INIT_STRING[i]);

    // If the R/W signal is not used, then
    // the busy bit can't be polled.  One of
    // the init commands takes longer than
    // the hard-coded delay of 60 us, so in
    // that case, lets just do a 5 ms delay
    // after all four of them.

    delay_ms(10);
    }

}

//---------------------------

void lcd_gotoxy(int8 x, int8 y)
{
int8 address;

if(y != 1)
   address = lcd_line_two;
else
   address=0;

address += x-1;
lcd_send_byte(0, 0x80 | address);
}

//-----------------------------
void lcd_putc(char c)
{
 switch(c)
   {
    case '\f':
```

```
      lcd_send_byte(0,1);
      delay_ms(4);
      break;

    case '\n':
       lcd_gotoxy(1,2);
       break;

    case '\b':
       lcd_send_byte(0,0x10);
       break;

    default:
       lcd_send_byte(1,c);
       break;
   }
}

//-----------------------------------------------
//-----------------------------------------------
```

Ασκήσεις που θα πρέπει να ετοιμαστούν για το όγδοο εργαστήριο

1. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για το μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής διαβάζει ένα τριών ψηφίων αριθμό και να καθυστερεί σε δέκατα του δευτερολέπτου το σήμα που εισάγεται από το RD1(λογικό 0 ή λογικό 1) και βγαίνει από το RD2. Υπ. τροποποιήστε τη λυμένη άσκηση 2 σε ότι αφορά την εισαγωγή του αριθμού.
2. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για το μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής θα λειτουργεί σαν ηλεκτρονική κλειδαριά. Ο κωδικός (password) είναι των 4 ψηφίων. Αν κατά την εισαγωγή του κωδικού υπάρξει λάθος τότε να μετρά τον αριθμό των λαθών που έγιναν και αν αυτός γίνει μεγαλύτερος από 2 τότε να θέτει το bit RE0 σε λογικό 0 από λογικό 1 που ήταν αρχικά για χρόνο 3 sec. Κάθε φορά που εισάγεται λανθασμένο ψηφίο ο κωδικός πρέπει να εισάγεται ξανά από την αρχή. Η εκτύπωση των σωστών αριθμών γίνεται στο LCD που συνδέεται στη πόρτα D. Το πληκτρολόγιο συνδέεται στη πόρτα B όπως στη λυμένη άσκηση 2.

Οι υπορουτίνες για το πληκτρολόγιο και για το LCD υπάρχουν στη σελίδα του μαθήματος της θεωρίας στο φάκελο χρήσιμα αρχεία. Προσοχή το LCD συνδέεται στην πόρτα D, οπότε μπορεί να χρειαστεί αλλαγή της πόρτας στην ρουτίνα flex_lcd.h ή lcd.h.

Ερωτήσεις που πρέπει να απαντηθούν

1. Να γραφεί πρόγραμμα που να διαβάζει από το πληκτρολόγιο ένα αριθμό του ενός ψηφίου και να τον ελέγχει. Αν ο αριθμός είναι άρτιος να θέτει το bit RC0 σε λογικό 1 διαφορετικά να το θέτει σε λογικό 0.

2. Να γραφεί πρόγραμμα που να διαβάζει από το πληκτρολόγιο δύο αριθμούς των δύο ψηφίων και αν η διαφορά τους είναι μεγαλύτερη από 6 τότε να θέτει το bit RC2 σε λογικό 1 διαφορετικά να το θέτει σε λογικό 0.
3. Να γραφεί πρόγραμμα που να λειτουργεί το σύστημα ως προγραμματιζόμενο χρονόμετρο. Η είσοδος του χρόνου εισάγεται από το πληκτρολόγιο σε sec με δύο ψηφία (χρόνος από 01 – 99 sec). Ο χρόνος θα πρέπει να εμφανίζεται στο LCD καθώς θα μειώνεται και όταν γίνει 0 τότε να θέτει το bit RC0 σε λογικό 1 για χρόνο 3sec και κατόπιν σε λογικό 0 και να περιμένει ξανά νέο χρόνο από το πληκτρολόγιο. Το πληκτρολόγιο συνδέεται στην πόρτα B και το LCD στη πόρτα D.