Άσκηση έβδομη του Εργαστηρίου

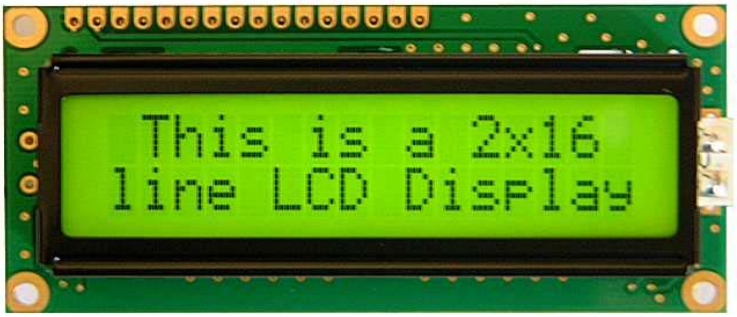
Στην έβδομη άσκηση εξετάζεται η οδήγηση έξυπνων οθονών LCD σε συνδυασμό με δεδομένα που εισάγονται από διακόπτες (dip-Switches).

Στις οθόνες LCD υπάρχουν πάρα πολλές επιλογές για τους χρήστες μικροελεγκτών. Γενικά υπάρχουν δύο τύποι ανάλογα με το τρόπο διασύνδεσης με το μικροελεγκτή. Οι οθόνες που έχουν δικό τους ενσωματωμένο μικροελεγκτή για τον έλεγχο τους («έξυπνες οθόνες») και αυτές που είναι μόνο ο ενδείκτης LCD (μόνο το τζάμι). Στη πρώτη περίπτωση για τη διασύνδεσή τους απαιτούνται από το μικροελεγκτή μερικές γραμμές I/O, το πόσες, εξαρτάται από το τύπο της οθόνης, ενώ στη δεύτερη περίπτωση, απαιτούνται πάρα πολλές γραμμές και πολύ πιο περίπλοκο πρόγραμμα για την οδήγησή τους, συνήθως οι κατασκευαστές μικροελεγκτών για τη περίπτωση αυτή ενσωματώνουν κυκλώματα, που βοηθούν το χρήστη να οδηγήσει αυτού του τύπου τις οθόνες (LCD drivers). Ένας άλλος τρόπος διαχωρισμού των οθόνων υγρών κρυστάλλων είναι ανάλογα με το τύπο των δεδομένων που μπορούν να απεικονίσουν έτσι προκύπτουν δύο τύποι οι αλφανουμερικές οθόνες (alphanumeric LCD) που μπορούν δείξουν μόνο γράμματα νούμερα και κάποιους επιπλέον χαρακτήρες όπως κάποιους χαρακτήρες στίξης κάποιους γραφιστικούς χαρακτήρες και τις περισσότερες φορές έχουν ένα κύριο σετ χαρακτήρων κάποιας γλώσσας και ένα βοηθητικό από κάποια άλλη γλώσσα. Ο δεύτερος τύπος είναι ο γραφιστικός τύπος (Graphics LCD) σε αυτές ο χρήστης μπορεί να δημιουργήσει οποιοδήποτε χαρακτήρα και γραφιστικά σχέδια και εικόνες. Στη περίπτωση αυτή το χαρακτηριστικό είναι η ανάλυση τους (resolution) δηλαδή ο αριθμός των εικονοστοιχείων (pixels) που μπορούν να απεικονίσουν όπως παράδειγμα 128X64 ο πρώτος αριθμός δίνει τον αριθμό των εικονοστοιχείων στην οριζόντια κατεύθυνση και ο δεύτερος στη κατακόρυφο. Στις γραφιστικές οθόνες υπάρχουν πάρα πολύ τύποι που είναι ανάλογοι με το τρόπο λειτουργίας τους όπως οι απλές γραφιστικές οθόνες (LCD), μονόχρωμες ή πολύχρωμες οθόνες LCD με οπίσθιο φωτισμό Led, οι οθόνες amoled και άλλες.

Στα μικρουπολογιστικά συστήματα μέχρι πρότινος οι οθόνες που χρησιμοποιούταν ήταν οι αλφανουμερικές οθόνες και σε κάποιες περιπτώσεις οι απλές γραφιστικές. Με την μείωση των τιμών των έξυπνων γραφιστικών οθονών όλο και περισσότεροι κατασκευαστές συσκευών ενσωματώνουν στις συσκευές τους οθόνες LCD TFT και LCD Amoled (Βλέπε κινητά τηλέφωνα, MP3 Players, οικιακές συσκευές κ.λ.π.)

Στην παρούσα ενότητα θα εξεταστούν οι «έξυπνες» αλφανουμερικές οθόνες LCD λόγω του διδακτικού περιεχομένου αλλά και της παρουσίας μιας τέτοιας οθόνης στην αναπτυξιακή πλακέτα του εργαστηρίου. Η οθόνη αυτή έχει όπως και οι περισσότερες οθόνες αυτού του τύπου ένα μικροελεγκτή της Hitachi τον HD44780 και είναι αυτός που χαρακτηρίζει το τρόπο επικοινωνίας με το μικροελεγκτή της πλακέτας (PIC18F4550). Γενικά υπάρχουν πάρα πολλές εταιρείες που παράγουν τέτοιου τύπου οθόνες που έχουν όλες τον ίδιο μικροελεγκτή ή παρόμοιο με το ίδιο ρεπερτόριο εντολών. Οι οθόνες αυτές χαρακτηρίζονται εκτός από το τύπο του μικροελεγκτή που περιέχουν και από τις δυνατότητες τους στην απεικόνιση. Ο αριθμός των γραμμών και αριθμός των χαρακτήρων που μπορούν να απεικονιστούν είναι δύο χαρακτηριστικά των οθονών αυτών.(2x16, 2x20, 4x16) Ένα επιπλέον χαρακτηριστικό είναι η ύπαρξη ή όχι του πίσω φωτισμού (Back Light). Το χρώμα του φωτισμού είναι ένα ακόμη χαρακτηριστικό των οθονών αυτών.

Η οθόνη που ενσωματώνεται στην αναπτυξιακή πλακέτα του εργαστηρίου είναι μια αλφανουμερική «έξυπνη» οθόνη με 2 σειρές και 16 χαρακτήρες σε κάθε γραμμή με ύπαρξη πίσω φωτισμού από Led κίτρινου-πράσινου χρώματος. Όπως αναφέρθηκε και παραπάνω περιέχει τον μικροελεγκτή της Hitachi HD44780 και για την επικοινωνία με το μικροελεγκτή της πλακέτας χρησιμοποιεί μια σειρά από εντολές που θα αναλυθούν παρακάτω. Στην οθόνη υπάρχει η δυνατότητα απεικόνισης λατινικών γραμμάτων αριθμών και κάποιων επιπλέον χαρακτήρων που εμφανίζονται στο σετ χαρακτήρων σε παρακάτω πίνακα. Οι ελληνικοί χαρακτήρες υπάρχουν σε κάποιες οθόνες σαν δεύτερο σετ χαρακτήρων αλλά όχι σε όλες τις οθόνες. Όπου δεν υπάρχουν μπορούν να δημιουργηθούν σαν επιπλέον προγραμματισμένοι χαρακτήρες δυνατότητα που δίνουν όλες οι οθόνες αυτού του τύπου.



Συνοπτικός πίνακας των εντολών του LCD σε δεκαεξαδική μορφή

LCD Command Codes

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

Στον CCS C Compiler για την οδήγηση των υγρών κρυστάλλων υπάρχουν έτοιμες συναρτήσεις που κάνουν το προγραμματισμό αυτών των οθονών πολύ εύκολη και δίδονται παρακάτω.

Η συνάρτηση flex LCD είναι μια από τις έτοιμες συναρτήσεις που ο χρήστης μπορεί να καλέσει με διάφορες παραμέτρους προκειμένου να τυπώσει ένα μήνυμα.
Οι εντολές δίδονται όπως τις εντολές εκτύπωσης της standard C σε οθόνη CRT.
Με το χαρακτήρα \f καθαρίζει την οθόνη και θέτει το κέρσορα στην αρχική θέση.
\n θέτει τον κέρσορα στην αρχή της δεύτερης γραμμής.
\b κάνει ολίσθηση του κέρσορα κατά μία θέση προς τα αριστερά

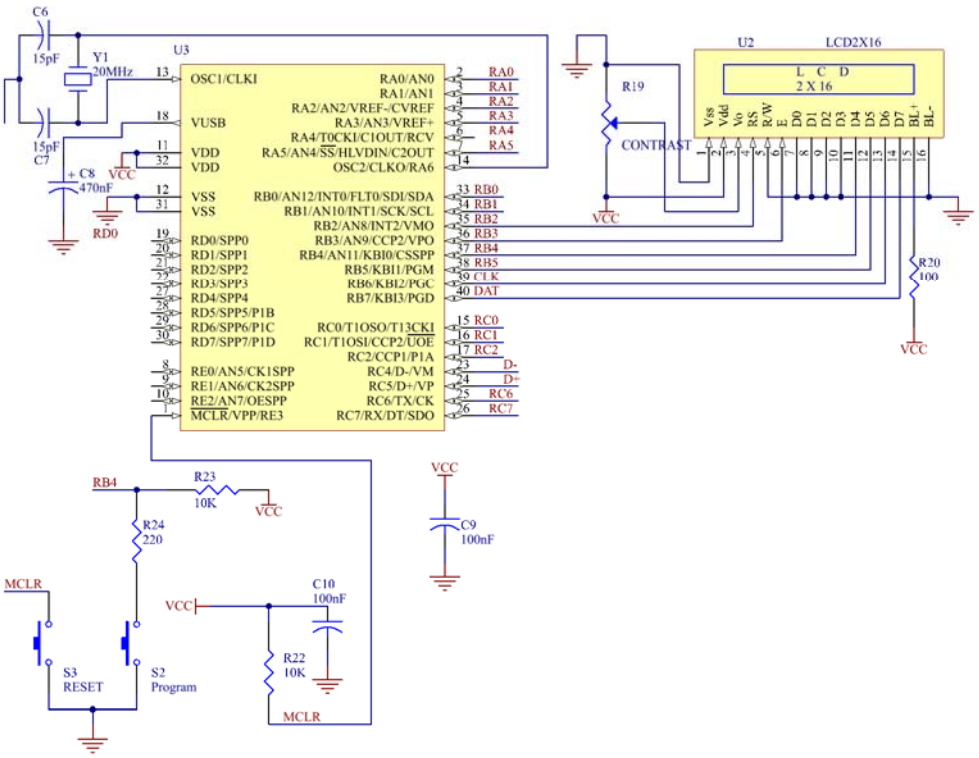
- \\a – To set cursor to the upper left
- \\f – To clear display and set cursor to upper left
- \\n – To go to start of next line
- \\b – To move back one position

Παράδειγμα της λειτουργίας με χρήση της flex LCD δίδεται παρακάτω.

Να γραφεί πρόγραμμα που να τυπώνει στη πρώτη γραμμή της οθόνης το μήνυμα MicroII Lab και στη δεύτερη γραμμή το μήνυμα LCD test program για χρόνο 2 sec και κατόπιν να σβήνει προς τα δεξιά το μήνυμα της δεύτερης γραμμής με ρυθμό 5Hz μέχρι να εξαφανισθεί όλο το μήνυμα της γραμμής αυτής.

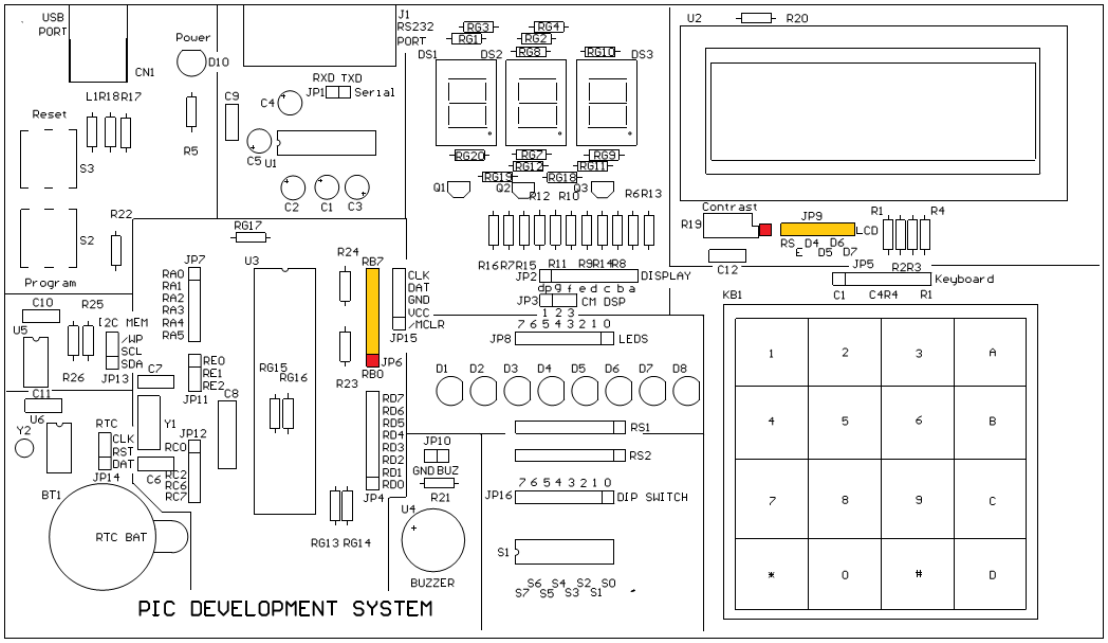
Λύση

Το σχηματικό του κυκλώματος



ΑΣΚΗΣΗ 7_1

Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

```
#include <main.h>

//============================
// flex_lcd.c
//============================

#define LCD_DB4   PIN_B4
#define LCD_DB5   PIN_B5
#define LCD_DB6   PIN_B6
#define LCD_DB7   PIN_B7

#define LCD_E     PIN_B3
#define LCD_RS    PIN_B2
#define LCD_RW    PIN_B1
```

```
//=======================================

#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, 5x7 dots
 0xc,                    // Display on
 1,                      // Clear display
 6                       // Increment cursor
 };

// Function definition

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);


//============================
//Main program
//============================

void main()
{
int8 i=0;
lcd_init();  //

lcd_putc("\f  MicroII Lab\n");
lcd_putc("LCD test program");
delay_ms(2000);
lcd_putc(" \n");
for(i=0;i<16;i++){
                    delay_ms(200);
                    lcd_putc("  \b");
                    }
while(1);

}

//-------------------------------------
//send a nibble to lcd
//-------------------------------------

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);
}

//-----------------------------------
// 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);
}

//----------------------------
//Init lcd
//----------------------------

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]);

        delay_ms(10);
        }

}

//-----------------------------
//Set cursor position
//-----------------------------

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);
}

//-----------------------------
//Send a char to lcd
//-----------------------------

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);
```

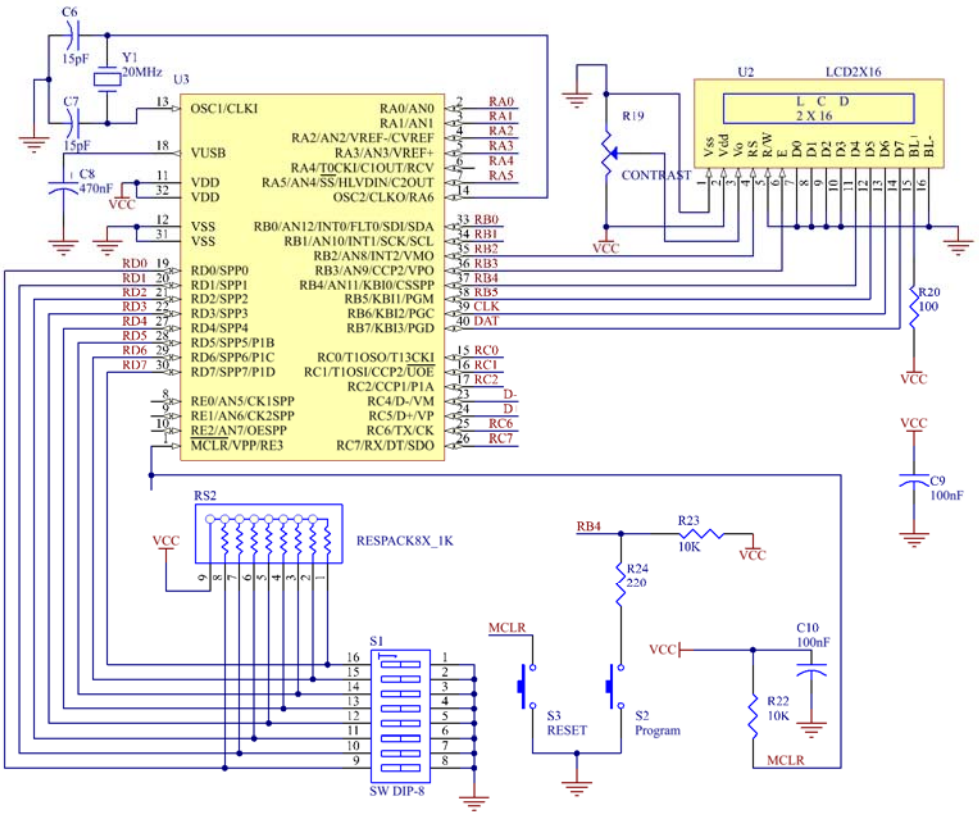
```
      break;

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

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

Ένας άλλος τρόπος οδήγησης είναι χρησιμοποιώντας προγράμματα της standard C και στέλνοντας της κατάλληλες εντολές. Ένα παράδειγμα δίδεται παρακάτω.

Παράδειγμα.

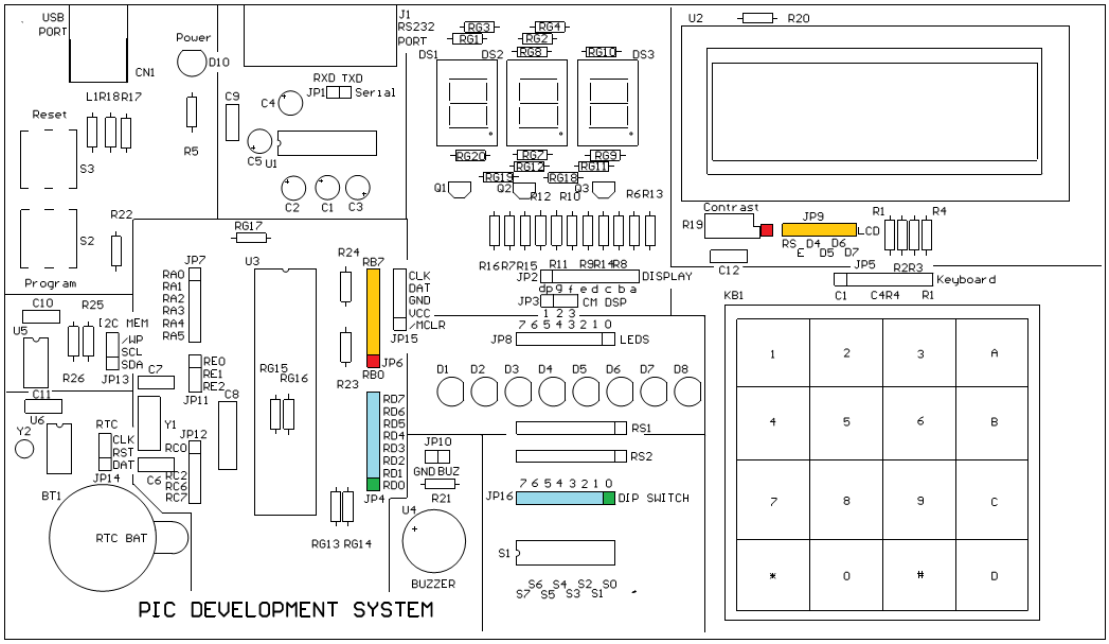
Να γραφεί πρόγραμμα που να τυπώνει στην οθόνη LCD τη κατάσταση των διακοπτών που θα συνδέονται στην πόρτα D.

Το σχηματικό του κυκλώματος



ΑΣΚΗΣΗ 7_2

Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα του παραδείγματος

```
#include <main.h>
#byte PORTB=0xf81
#byte PORTD=0xf83
// INPUTS-----------------------------------

#define sw1  (PORTD & 0x01)
#define sw2  ((PORTD & 0x02) >1)
#define sw3  ((PORTD & 0x04) >1)
#define sw4  ((PORTD & 0x08) >1)
#define sw5  ((PORTD & 0x10) >1)
#define sw6  ((PORTD & 0x20) >1)
#define sw7  ((PORTD & 0x40) >1)
#define sw8  ((PORTD & 0x80) >1)

//LCD  PIN definition----------------------------
```

```
#define LCD_D0  PIN_B4
#define LCD_D1  PIN_B5
#define LCD_D2  PIN_B6
#define LCD_D3  PIN_B7
#define LCD_RS  PIN_B2
#define LCD_EN  PIN_B3
//LCD  commands-------------------------
#define LINE_1  0x00
#define LINE_2  0x40
#define CLEAR_DISP  0x01

// Variable definition----------------------

int i=0;

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

void LCD_Init ( void );
void lcd_message(void);
void LCD_SetPosition ( unsigned int cX );
void LCD_PutChar ( unsigned int cX );
void LCD_PutCmd ( unsigned int cX );
void LCD_PulseEnable ( void );
void LCD_SetData ( unsigned int cX );
void pic_init (void) ;

// Main programm-----------------------------

void main()
{
 pic_init();
 lcd_init();
 lcd_message ();

  while (TRUE)
  {
        if (sw1 == 0) {
                LCD_SetPosition ( LINE_1 + 1 );
                  printf ( LCD_PutChar, "SW1 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_1 + 1 );
                printf ( LCD_PutChar, "SW1 OFF " );
                }
        if (!(sw2)) {
                LCD_SetPosition ( LINE_1 + 9 );
                  printf ( LCD_PutChar, "SW2 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_1 + 9 );
```

```
                printf ( LCD_PutChar, "SW2 OFF " );
                }
        if (!(sw3)) {
                LCD_SetPosition ( LINE_2 + 1 );
                printf ( LCD_PutChar, "SW3 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_2 + 1 );
                printf ( LCD_PutChar, "SW3 OFF " );
                }
    if (!sw4) {
                LCD_SetPosition ( LINE_2 + 9 );
                 printf ( LCD_PutChar, "SW4 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_2 + 9 );
                printf ( LCD_PutChar, "SW4 OFF " );
                }
        delay_ms(1000);
        if (!(sw5)) {
                LCD_SetPosition ( LINE_1 + 1 );
                  printf ( LCD_PutChar, "SW5 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_1 + 1 );
                printf ( LCD_PutChar, "SW5 OFF " );
                }
        if (!(sw6)) {
                LCD_SetPosition ( LINE_1 + 9 );
                  printf ( LCD_PutChar, "SW6 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_1 + 9 );
                printf ( LCD_PutChar, "SW6 OFF " );
                }
        if (!(sw7)) {
                LCD_SetPosition ( LINE_2 + 1 );
                printf ( LCD_PutChar, "SW7 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_2 + 1 );
                printf ( LCD_PutChar, "SW7 OFF " );
                }
    if (!sw8) {
                LCD_SetPosition ( LINE_2 + 9 );
                 printf ( LCD_PutChar, "SW8 ON  " );
                }
                else{
                LCD_SetPosition ( LINE_2 + 9 );
                printf ( LCD_PutChar, "SW8 OFF " );
```

```
                }
        delay_ms(1000);
         }
}
// Function definition--------------------------
// LCD FUNCTIONS --------------------------------

void LCD_Init ( void )
        // LCD initialize
   {
         LCD_SetData ( 0x00 );
        delay_ms ( 500 );       // wait enough time after Vdd rise
        output_low ( LCD_RS );
        LCD_SetData ( 0x03 );   // init with specific nibbles to start 4-bit mode
        LCD_PulseEnable();
        LCD_PulseEnable();
        LCD_PulseEnable();
        LCD_SetData ( 0x02 );   // set 4-bit interface
      LCD_PulseEnable();      // send dual nibbles hereafter, MSN first
        LCD_PutCmd ( 0x2C );    // function set (all lines, 5x7 characters)
        LCD_PutCmd ( 0x0C );    // display ON, cursor off, no blink
        LCD_PutCmd ( 0x01 );    // clear display
        LCD_PutCmd ( 0x06 );    // entry mode set, increment
        }

void lcd_message(void){
        // Scoll a message left
        LCD_PutCmd ( CLEAR_DISP );
        LCD_SetPosition ( LINE_1 + 1 );
        printf ( LCD_PutChar, "MIKRO II LAB" );
        delay_ms(1000);
        LCD_PutCmd ( 0x07 );
        for (i=0;i<13;i++){
                                LCD_SetPosition ( LINE_1 + 1 );
                                printf ( LCD_PutChar, " " );
                                delay_ms(400);
                                }
        delay_ms(1000);
        LCD_PutCmd ( CLEAR_DISP );
        LCD_PutCmd ( 0x06 );
        delay_ms(20);

}


void LCD_SetPosition ( unsigned int cX )
         {
   // Set cursor position
   LCD_SetData ( swap ( cX ) | 0x08 );
   LCD_PulseEnable();
```

```
   LCD_SetData ( swap ( cX ) );
   LCD_PulseEnable();
   }

void LCD_PutChar ( unsigned int cX )
   {
   // Send to LCD data in nibbles
   output_high ( LCD_RS );   // Send data to LCD in nibbles
   LCD_SetData ( swap ( cX ) );     // send high nibble
   LCD_PulseEnable();
   LCD_SetData ( swap ( cX ) );     // send low nibble
   LCD_PulseEnable();
   output_low ( LCD_RS );
   }

void LCD_PutCmd ( unsigned int cX )
   {
   // Send  to LCD commsnds in nibbles (4 bits operation)
   LCD_SetData ( swap ( cX ) );     // send high nibble
   LCD_PulseEnable();
   LCD_SetData ( swap ( cX ) );     // send low nibble
   LCD_PulseEnable();
   }

void LCD_PulseEnable ( void )
         // Set reset the Enable signal
   {
   output_high ( LCD_EN );
   delay_us ( 10 );
   output_low ( LCD_EN );
   delay_ms ( 5 );
   }

void LCD_SetData ( unsigned int cX )
   {
   output_bit ( LCD_D0, cX & 0x01 );
   output_bit ( LCD_D1, cX & 0x02 );
   output_bit ( LCD_D2, cX & 0x04 );
   output_bit ( LCD_D3, cX & 0x08 );
   }
// I/O definition----------------------------------
void pic_init (void) {
  SET_tris_d(0xff);              //Port D input
  SET_tris_b(0x00);             //Port B output
  }
```

Ασκήσεις που θα πρέπει να ετοιμαστούν για το έβδομο εργαστήριο

1. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής θα ελέγχει το bit RD1 και αν είναι σε λογικό

0 τότε να τυπώνει κείμενο στην Αγγλική γλώσσα που θα έχει τοποθετηθεί σε πίνακα με μέγιστο 16 χαρακτήρες σε κάθε γραμμή αφού κάνει έλεγχο για τα κενά μεταξύ των λέξεων έτσι ώστε να μη κόβονται οι εκτυπώμενες λέξεις στην οθόνη. Όταν γεμίζει μία γραμμή το κείμενο συνεχίζει στην επόμενη και κατόπιν η δεύτερη γραμμή θα γίνεται πρώτη και θα συνεχίζεται η εκτύπωση στη δεύτερη γραμμή μέχρι το τέλος του κειμένου. Τα δεδομένα και τα σήματα ελέγχου του LCD παρέχονται από τη πόρτα B όπως στις λυμένες ασκήσεις.

2. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για το μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής θα λειτουργεί ως μετρητής πραγματικού χρόνου. Στη πρώτη γραμμή του LCD θα τυπώνεται το μήνυμα Timer και στη δεύτερη γραμμή ο χρόνος σε ώρες λεπτά και δευτερόλεπτα. Η μέτρηση του χρόνου θα γίνεται με διακοπή από την υπερχείλιση του Timer0. Για την πρωτοποθέτηση της σωστής ώρας θα χρησιμοποιηθούν οι διακόπτες που συνδέονται στα pins RD1,RD2,RD3. Η λειτουργία της πρωτοποθέτησης (time setting) επιλέγεται με αλλαγή του RD3 από λογικό 0 σε λογικό 1, για την αύξηση των λεπτών το RD2 και για τη μείωση των λεπτών το RD1. Οι αλλαγές θα συμβαίνουν με αλλαγή των pins από λογικό 0 σε λογικό1.

3. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για το μικροελεγκτή PIC 18F4550 με το οποίο ο μικροελεγκτής που θα εκτυπώνει μήνυμα στην Ελληνική γλώσσα στις δύο γραμμές του LCD το οποίο θα ολισθαίνει προς τα αριστερά με ρυθμό 2 χαρακτήρες το δευτερόλεπτο.

Ασκήσεις που πρέπει να σταλούν.

1. Να γραφεί πρόγραμμα που δείχνει στο LCD στην πρώτη γραμμή και πρώτη στήλη τους πρώτους 16 χαρακτήρες του Αγγλικού αλφαβήτου.
2. Να γραφεί πρόγραμμα που να λειτουργεί τον PIC ως λογικό ενδείκτη (logic probe). Όταν η είσοδος RD1 είναι σε λογικό 0 να τυπώνεται στην πρώτη γραμμή πρώτη στήλη το μήνυμα Low ενώ αν η είσοδος είναι σε λογικό 1 το μήνυμα High. Το LCD συνδέεται όπως στις παραπάνω ασκήσεις.
3. Να γραφεί πρόγραμμα που να λειτουργεί το σύστημα σαν λογικό αναλυτή . Η συσκευή θα λειτουργεί ως εξής. Η είσοδος του αναλυτή θα είναι το bit RD0. Η είσοδος ελέγχεται κάθε 1msec αν η είσοδος είναι σε λογικό 1 και πριν ήταν σε λογικό 0 τότε στην οθόνη εμφανίζεται ο χαρακτήρας _I αν και πριν ήταν σε λογικό 1 τότε στην οθόνη στέλνεται ο χαρακτήρας – αν η είσοδος αλλάξει από λογικό 1 σε λογικό 0 τότε θα τυπώνεται στην οθόνη ο χαρακτήρας 7 που δείχνει αλλαγή από 1 σε 0 και συνεχίζει όσο η είσοδος είναι σε 0 να τυπώνεται ο χαρακτήρας _ που είναι για λογικό 0. Ο κύκλος επαναλαμβάνεται συνεχώς και η εικόνα θα πρέπει να μετακινείται προς τα αριστερά.