Course name: Practical App. CS II
Course Code: CSC20X
Lecturer: Dr. Ahmed ElShafee

Exam number: Final – model answer
Exam Date: 03/06/2014
Time Allowed: 120 minutes

Q1	Q2	Q3	Q4	Q5	Q6	Total
/7.5	/5	/10	/5	/7.5	/5	/40

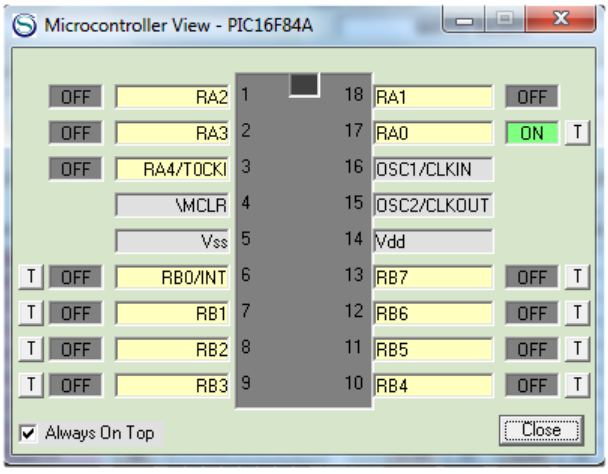
1. A 220V AC lamp is placed on the top of telecommunication tower in to alert plans during the night, this lamp is called Beacon Lamp.
You are required to build a control board uses 16F84A microcontroller to drive an 220 V AC beacon lamp. MC is connected to photocell (photo resistor "LDR") to surrounded detect light intensity.
Microcontroller will monitor the feedback from photocell (daylight = 0, night=1), it will turn off the beacon lamp in the presence of daylight, and beacon will flash (delay 1 sec) during the night.
Write a C program that implements Beacon Controller.
Consider the following

MC pin	Interfaced to
RA0	Photocell
RA1	Beacon Lamp

Status	MC input
Day light	0
Night	1

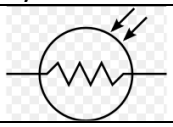
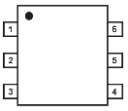
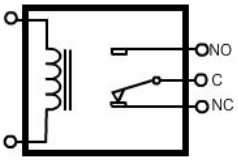
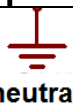
Status	MC output
Beacon on	1
Beacon off	0

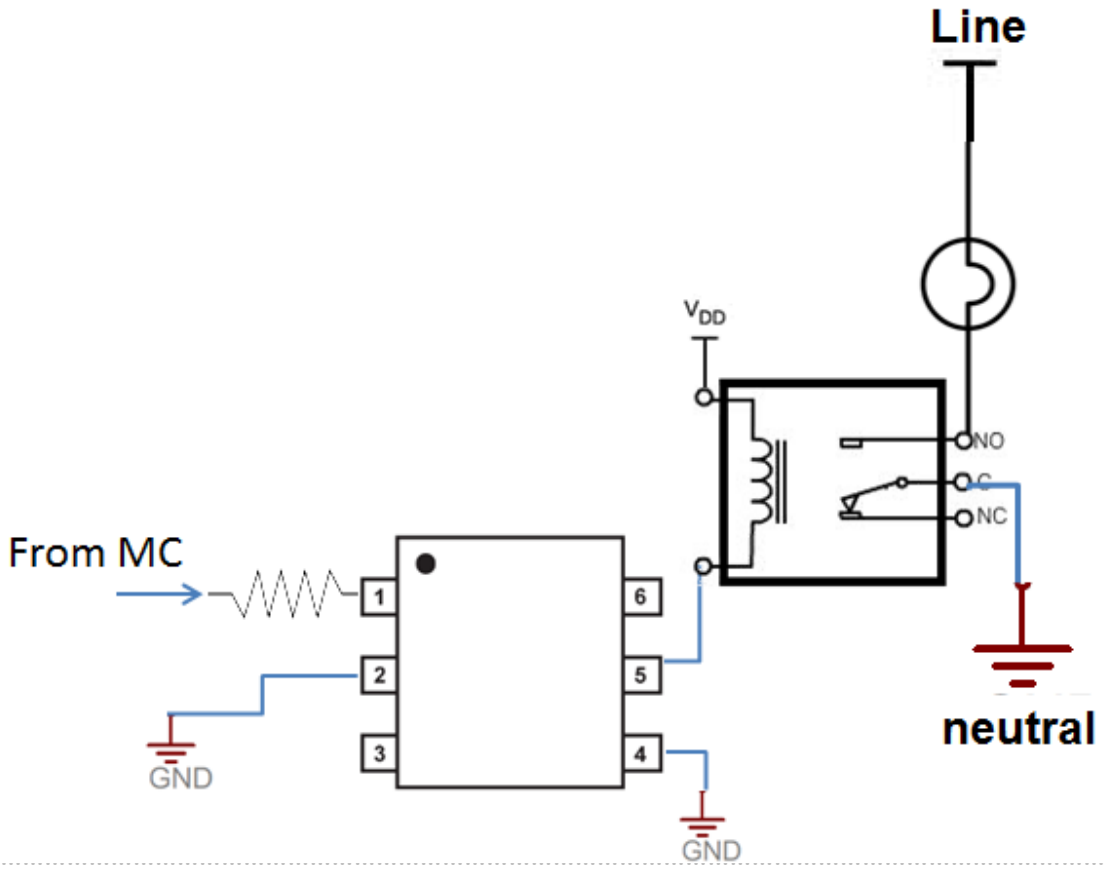
Note: you may use comments on page 2 to write your code, or you may write you own code from scratch on page 3

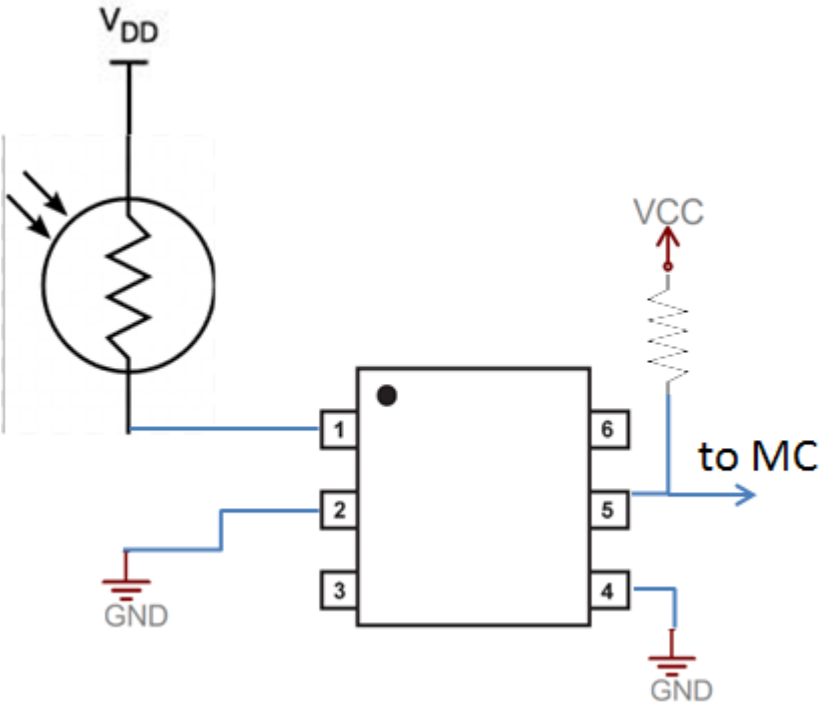


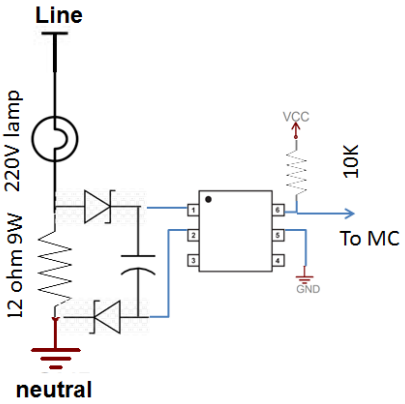
```
#include <xc.h>
#include <pic16f84a.h>
void  main(void) {
   // RA0 : PhotoCell
   // RA1 : Beacon
   // configure portA
   TRISA=0x01;
   //end less loop
   while(1)
   {
      // check photo cell for daylight
      if(RA0==0)
      {
         // turn beacon off
         RA1=0;
      }
      // check photo cell for night
      else
      {
         // turn photo cell on
         RA1=1;
         // delay
         _delay(1000);
         // turn photo cell off
         RA1=0;
         // delay
         _delay(1000);
      }
   }
}
```

2. build convenient interfaces to connect 220V AC lamp and photo cell to microcontroller from the following components

#	Components	Value	Symbol
1	Photo Cell	-	
2	Opto-Coupler Isolator	4N35	
3	Relay 12V	-	
4	Resistors	100 ohm	
5	Resistor	10K ohm	
6	Vcc	5V	
7	Vdd	12V	
8	Gnd	0V	
9	Lamp	220 Volt	
10	Line	-	
11	Neutral	-	


Line
VDD
NO
C
NC
From MC
1
2
3
6
5
4
GND
GND
neutral


VDD
VCC
1
2
3
6
5
4
to MC
GND
GND



3. faulty lamp is a major problem that faced the traditional Beacon controller, as controller can't detect that beacon lamp is fault, and not flashing.
You ☺ come up with small circuit connects in series with AC lamp, that can detect that lamp is on (healthy) or off (faulty).
So if beacon lamp is healthy, circuit will feedback 1 to MC,
And if beacon lamp is faulty, circuit will feed back 0 to MC.
Then You ☺ suggested to connect 2 beacon lamp instead of one, so if one becomes faulty, controller automatically switch to 2nd lamp, and feedback alarm (active high).
Write a C program that implements evolved Beacon Controller.
Consider the following

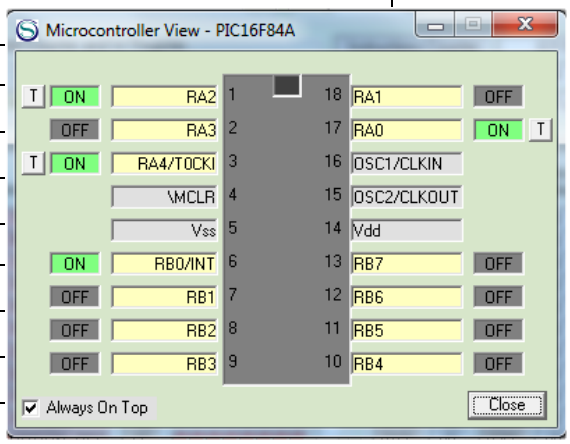
MC pin	Interfaced to
RA0	Photocell
RA1	Beacon Lamp 1
RA2	Beacon Lamp 1 Feedback
RA3	Beacon Lamp 2
RA4	Beacon Lamp 2 Feedback
RB0	Alarm

Status	MC input
Day light	0
Night	1

Status	MC output
Beacon is on	1
Beacon is off	0

Status	MC output
Beacon is faulty	1
Beacon is healty	0

Status	MC output
Alarm	1
No alarm	0



Note: you may use comments on pages 8,9 to write your code, or you may write you own code from scratch on pages 10,11

```
#include <xc.h>
#include <pic16f84a.h>
   // RA0 : PhotoCell
   // RA1 : Beacon 1
   // RA2 : Beacon 1 feedback
   // RA3 : Beacon 2
   // RA4 : Beacon 2 Feedback
   // RB0 : Alarm
void  main(void) {
   // initialize flags for beacon lamps status (faulty / healthy)
   int B1Faulty=0,B2Faulty=0;
   // configure port A
   TRISA=0xf5;
   // Configure Port B
   TRISB=0x00;
   //end less loop
   while(1)
   {
      // check photo cell for daylight
      if(RA0==0)
      {
         // turn beacon off
         RA1=0;
      }
      // check photo cell for night
      else
      {
         // check if beacon lamp one status is healthy (if)
         if(B1Faulty==0)
         {
            // turn on beacon lamp one
            RA1=1;
            // delay
            _delay(1000);
            // check beacon lamp one feedback
            if(RA2==1) // reverse
            {
               // set beacon lamp 1 status faulty
               B1Faulty=1;
               // activate alarm
               RB0=1;
            }
         }
```

```
        // check if beacon lamp 1 status is  faulty  and beacon 1 status in healthy(else if)
        else if((B1Faulty==1) & (B2Faulty==0))
        {
          // turn on beacon lamp 2
          RA3=1;
          // delay
          _delay(1000);
          // check beacon lamp 2 feed back
          if(RA4==1) // reverse
          {
            // // set beacon lamp 2 status faulty
            B2Faulty=1;
          }
        }
        // if beacon lamp 2 status is faulty
        else
        {
          // turn off all beacon lamps
          RA1=0;
          RA3=0;
          // set alarm on
          RB0=1;
        }
        // turn off beacon lamp 1,2
      RA1=0;
      RA3=0;
      // delay
      _delay(1000);
      }
   }
}
```

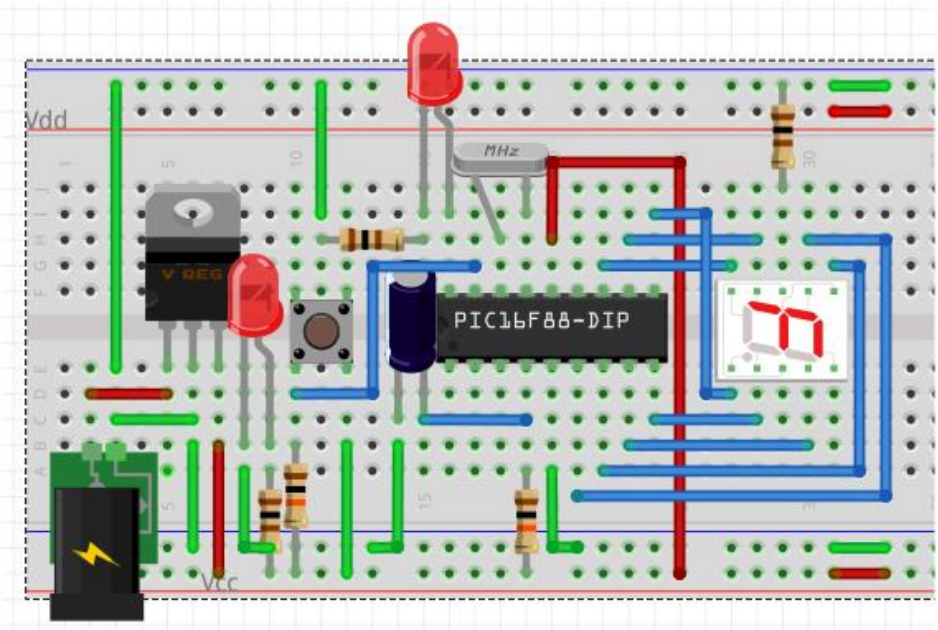
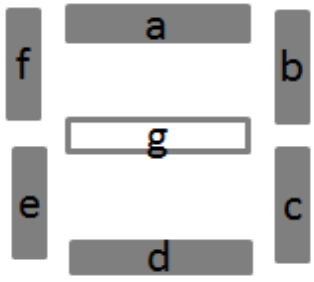
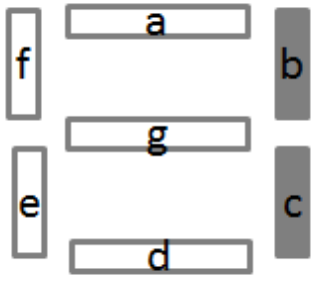
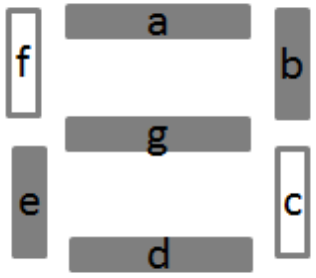
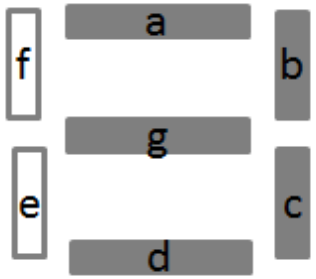
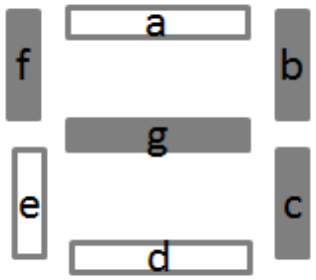
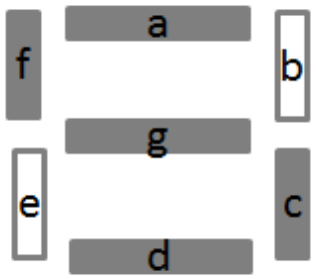
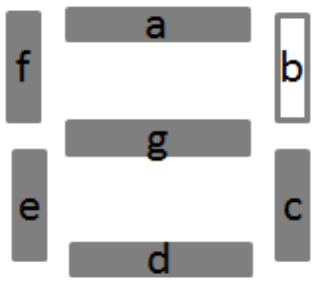
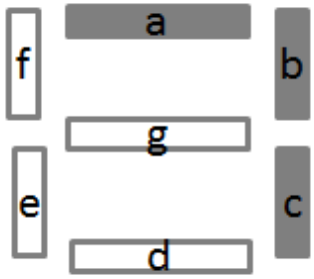
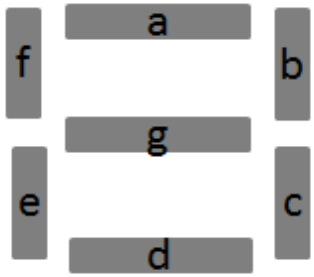
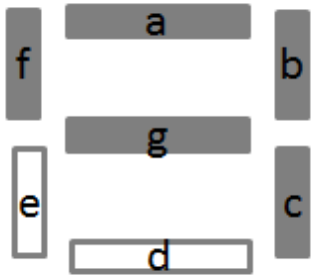
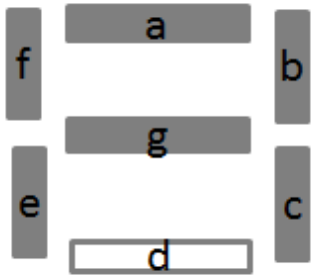
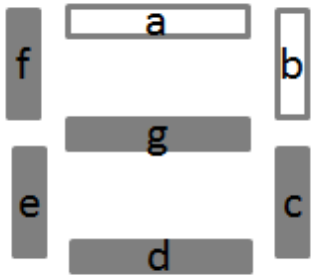


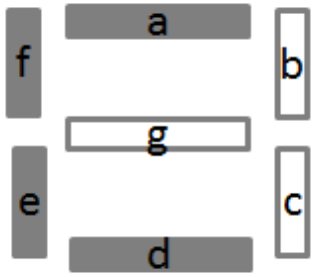
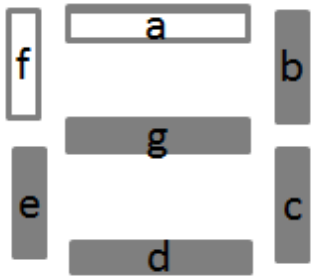
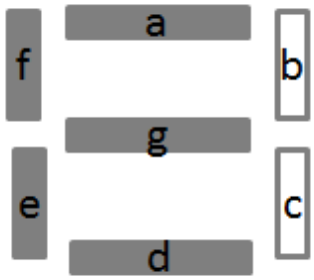
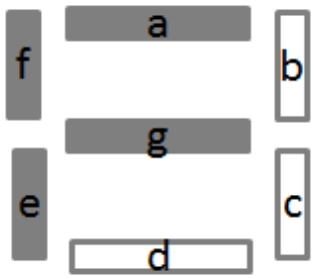
Figure shows a PIC16F84A connected to seven segments display. When power is connected to microcontroller, it starts counting up from 0x00 to 0x0f, then overflow and start over again. If you push press it counts down from 0x0f to 0x00 then overflow and start over.

4. complete the following table, symbol 0 is already solved

symbol	h	g	f	e	d	c	b	a	binary	hex	7segment
0			*	*	*	*	*	*	00111111	0x3f	
1						*	*		00000110	0x06	

2		*		*	*		*	*	01011011	0x5B	
3		*			*	*	*	*	01001111	0x4F	
4		*	*			*	*		01100110	0x66	
5		*	*		*	*		*	01101101	0x6d	
6		*	*	*	*	*		*	01111101	0x7C	

7						*	*	*	00000111	0x07	
8		*	*	*	*	*	*	*	01111111	0x7F	
9		*	*			*	*	*	01100111	0x67	
10		*	*	*		*	*	*	01110111	0x77	
11		*	*	*	*	*			01111100	0x7C	

12			*	*	*			*	00111001	0x39	
13		*		*	*	*	*		01011110	0x5E	
14		*	*	*	*		*	*	01111011	0x7B	
15		*	*	*				*	01110001	0x71	

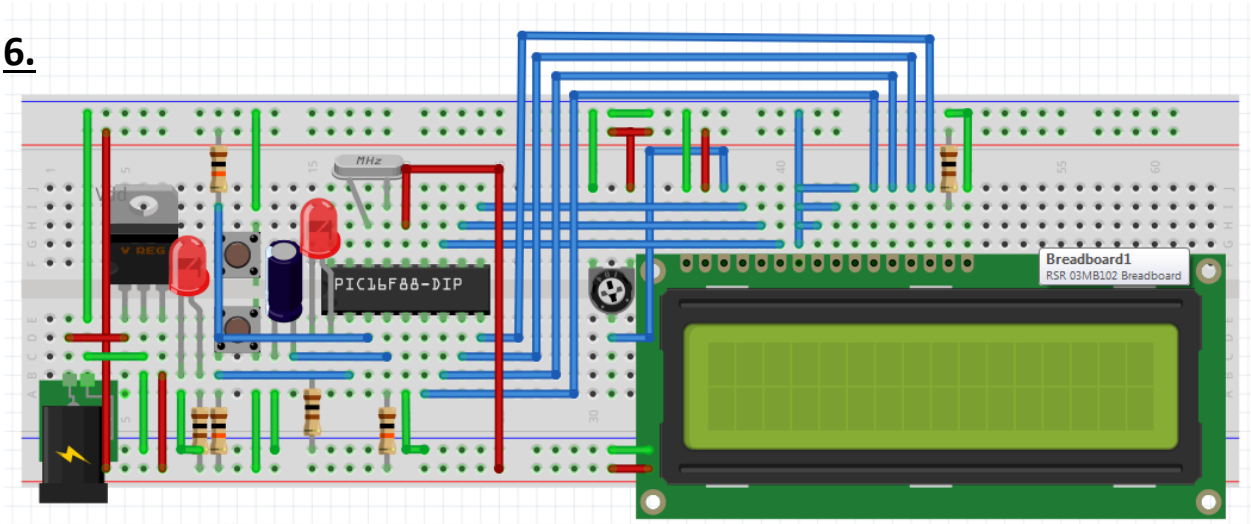
5. write a program in C language for press controlled 7 segments up down hex counter, use comments on page 16,17,18 or write the program from scratch on page 18,19

```
#include <xc.h>
#include <pic16f84a.h>
// portb <--> 7 segment
// RA0  press
void SevenSegmentDecoder(int number);
void main(void) {
   // configure port B
   TRISB=0x00;
   // configure port A
   TRISA=0b00011101;
   // initialize counter =0
   int x=0;
   // endless loop
   while(1)
   {
      // check RA0 if released
      if(RA0==0)
      {
         // count up with overflow
         if(x<15)
            x++;
         else
            x=0;
      }
      // check RA0 if pressed
      else
      {
         // count down with overflow
         if(x>0)
         {x=x-1;}
         else
         {x=15;}
      }
      // decode counter, send it to port b
      SevenSegmentDecoder(x);
      // delay
      _delay(10);
   }
}
```

```
void SevenSegmentDecoder(int number)
{
   // test counter
switch (number)
{
   // first condition send decoded data to port B
   case 0:
      PORTB=0b00111111;
   break;
   // 2nd  condition send decoded data to port B
   case 1:
      PORTB=0b00000110;
   break;
   // and so on
   case 2:
      PORTB=0b01011011;
   break;
   case 3:
      PORTB=0b01001111;
   break;
   case 4:
      PORTB=0b01100110;
   break;
   case 5:
      PORTB=0b01101101;
   break;
   case 6:
      PORTB=0b01111101;
   break;
   case 7:
      PORTB=0b00000111;
   break;
   case 8:
      PORTB=0b01111111;
   break;
   case 9:
      PORTB=0b01100111;
   break;
```

```
case 10:
      PORTB=0b01110111;
   break;
case 11:
      PORTB=0b01111100;
   break;
   case 12:
      PORTB=0b00111001;
   break;
case 13:
      PORTB=0b01011110;
   break;
case 14:
      PORTB=0b01111011;
   break;
case 15:
      PORTB=0b01110001;
   break;
   default:
      PORTB=0b00000000;
   break;
}
}
```

6.



The above picture show an LCD connected to PIC16F84A microcontroller using the following configuration

Change LCD Module Color Scheme	
LCD Type: 2x16	RS Line ---> PORTB, 4
Data Lines ---> PORTB	E Line ---> PORTB, 6
Interface: 4-bit, Low	R/W Line ---> PORTB, 5

You are required to write a program that display two different counters on LCD.
Each counter is displayed on one line of LCD.
Each counter is controlled using a press, that counter increment s after pressing its press for convenient period than updates counter on LCD screen.
Counter overflow after exceeding 100
Consider using the following library
Use comments on pages 21,22 or write code from scratch on pages 22,23,24

```
#include <xc.h>
#include <pic16f84a.h>
#define LCD_RS RB4
#define LCD_RW RB5
#define LCD_EN RB6

void display_string(char str[]);
void send_data(char dataout);
void display_char(char data_value);
void send_cmd(char cmdout);
void write_cmd(char cmd_value);
void INIT_LCD(void);
void INIT_HW(void);
```

Hex Code	Command to LCD instruction Register
01	Clear display screen
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line

```
#include <xc.h>
#include <pic16f84a.h>
#include "lcd.h"
//#include "config.h"
void delay_ms(int x);
void main ()
{
   //initialize counter 1
   int counter1=0;
   //initialize counter 2
   int counter2=0;
   INIT_HW() ;
   // config port A RA3 & RA4 (2 presses) binput, RA2 output (LED)
   TRISA = 0x18 ;
   RA2=1;
   INIT_LCD() ;
   // goto line 1
   write_cmd(0x80) ;
   // write welcome string on 1st line
   display_string("Welcome 2 press");
   // goto line 2
   write_cmd(0xC0) ;
   // write welcome string on 2nd line
   display_string("ctrolled counter");
   //delay_ms(1000);
   // delay 50 us
   _delay(50);
   // clear lcd
   write_cmd(0x01);
   while(1)
   {
      // check upper press is pressed
      if(RA3==1)
         // increment counter
         counter1=(counter1+1)%1000;
      // check if upper press is pressed
      if(RA4==1)
         // decrement down counter
         counter2=(counter2+1)%1000;
      // goto 1st line
      write_cmd(0x80) ;
// display counts (3 digits) of 1st counter
      display_string("Counter1 = ");
      char d3=(counter1/100)+'0';
      char d2=((counter1-(counter1/100))/10)+'0';
      char d1=(counter1%10)+'0';
```

```
// display counts (3 digits) of 1st counter
      display_string("Counter1 = ");
      char d3=(counter1/100)+'0';
      char d2=((counter1-(counter1/100))/10)+'0';
      char d1=(counter1%10)+'0';
      display_char(d3);
      display_char(d2);
      display_char(d1);
      // goto 2nd line
      write_cmd(0xC0) ;
      // display countes (3 digits) of 2nd counter
      display_string("Counter2 = ");
      d3=(counter2/100)+'0';
      d2=((counter2-(counter2/100))/10)+'0';
      d1=(counter2%10)+'0';
      display_char(d3);
      display_char(d2);
      display_char(d1);
      // toogle status of RA2 // led blinking
      RA2=~RA2;
      //delay_ms(500);
      // delay 50us
      _delay(50);
   }
}
```