Microcontroladores Laboratorio

Semana 7

Preguntas previas:

- En DD, el nivel de contraste en que valor de incremento/decremento empleo?
  - Puedes usar en incrementos de 10 en 10.
- Sobre el horario de TF

Exámenes Finales

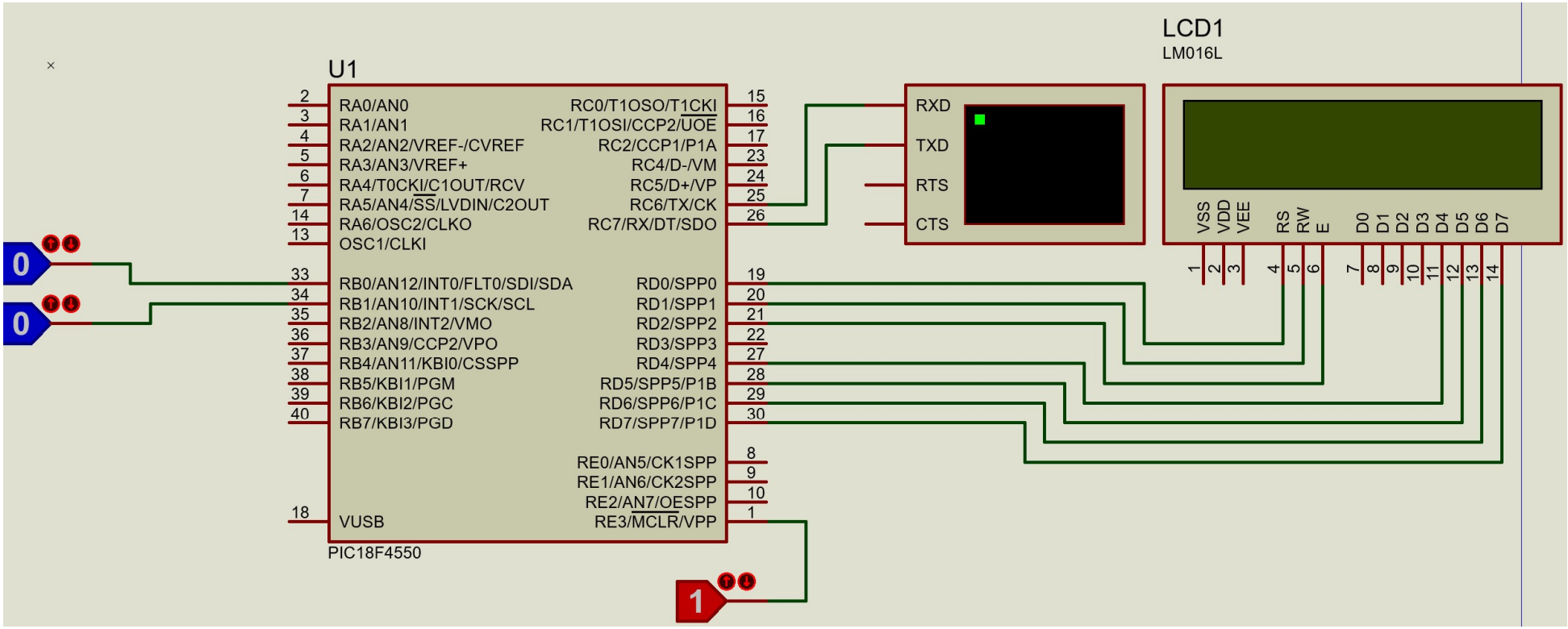
Tipo prueba	Fecha	Asignatura	Secc.	Grupo	Hora	Aula	Local	Medio	Módulo
TRABAJO FINAL	MI 03.03.2021	EL174 - MICROCONTROLADORES	EL51	01	15:00-18:00	UA-43	MO	SUS	RE

- Profesor no se puede simular el contraste y la luz de fondo del LCD en el Proteus
  - Usa el instrumento virtual de osciloscopio…

Agenda:

- Ingreso de cadena de caracteres por comunicación serial

Circuito de prueba



Especificaciones de la aplicación ejemplo:

- Se mostrará un menú a través del terminal serial al energizar el microcontrolador
- La velocidad de comunicación es 33600 8N1
- Se tendrá una opción para el ingreso de un mensaje (cadena de caracteres) desde el terminal serial y otra opción para poder visualizar el mensaje ingresado tanto en el mismo terminal como en un display LCD 2x16
- Al momento del ingreso de la cadena cada carácter ingresado hará que se muestre un asterisco en el terminal serial.

```c
#pragma config PLLDIV = 1          // PLL Prescaler
#pragma config CPUDIV = OSC1_PLL2// System Clock
#pragma config FOSC = XTPLL_XT  // Oscillator Sel
#pragma config PWRT = ON          // Power-up Time
#pragma config BOR = OFF          // Brown-out Res
#pragma config WDT = OFF          // Watchdog Time
#pragma config CCP2MX = ON        // CCP2 MUX bit
#pragma config PBADEN = OFF       // PORTB A/D Enab
#pragma config MCLRE = ON         // MCLR Pin Enabl
#pragma config LVP = OFF          // Single-Supply

#include <xc.h>
#include <stdio.h>
#include <string.h>
#define _XTAL_FREQ 48000000UL        //frecuencia

unsigned char menu_1[] = {"Bienvenidos a la aplic
unsigned char menu_2[] = {"Elija opcion:"};
unsigned char menu_3[] = {"(1) Ingrese mensaje"};
unsigned char menu_4[] = {"(2) Visualiza el mensa
unsigned char menu_7[] = {"(m) Visualizar el menu

unsigned char msg_ingreso[] = {"Ingrese mensaje
unsigned char msg_ingreso_ack[] = {"Mensaje ingre
unsigned char msg_emision[] = {"El mensaje ingres

unsigned char error[] = {"Tecla invalida, intente

unsigned char msg_input[32] = {0};
unsigned char menu_state = 0;

unsigned char num_char = 0;

void init_config(void){
    TRISDbits.RD6 = 0;
    TRISDbits.RD7 = 0;
    INTCONbits.GIE = 1;      //Interruptor global
    INTCONbits.PEIE = 1;     //Interruptor de inte
    PIE1bits.RCIE = 1;       //Habilitamos las int
}
```

```c
void EUSART_config(void){
    SPBRGH = 0;                 //Ignorado debido a q
    SPBRG = 21;                 //38600 baudios
    TRISCbits.RC6 = 0;          //Puerto RC6 como sal
    RCSTAbits.SPEN = 1;         //Encendemos el puert
    TXSTAbits.TXEN = 1;         //Encendemos el trans
    RCSTAbits.CREN = 1;         //Encendemos el recep
}

void EUSART_siguientelinea(void){
    TXREG = 0x0A;
    while(TXSTAbits.TRMT == 0);
    TXREG = 0x0D;
    while(TXSTAbits.TRMT == 0);
}

void EUSART_enviacadena(const unsigned char *vector){
    unsigned char cantidad = 0;
    cantidad = strlen(vector);
        for (unsigned char x=0;x<cantidad;x++){
            TXREG = vector[x];
            while(TXSTAbits.TRMT == 0);
        }
}

void EUSART_enviachar(unsigned char papa){
    TXREG = papa;
    while(TXSTAbits.TRMT == 0);
}

void vis_menu(void){
    EUSART_enviacadena(menu_1);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_2);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_3);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_4);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_7);
    EUSART_siguientelinea();
}
```

```c
void main(void) {
    init_config();
    EUSART_config();
    vis_menu();
    EUSART_siguientelinea();
    while(1);
}
```

```c
void __interrupt() RC_ISR(void){
    PIR1bits.RCIF = 0;
    if (menu_state == 0){
        switch(RCREG){
            case '1':
                EUSART_enviacadena(msg_ingreso);
                EUSART_siguientelinea();
                num_char = 0;
                menu_state = 1;
                break;
            case '2':
                EUSART_enviacadena(msg_emision);
                EUSART_siguientelinea();
                EUSART_enviacadena(msg_input);
                EUSART_siguientelinea();
                vis_menu();
                EUSART_siguientelinea();
                menu_state = 0;
                break;
            case 'm':
                vis_menu();
                EUSART_siguientelinea();
                menu_state = 0;
                break;
            default:
                EUSART_enviacadena(error);
                EUSART_siguientelinea();
                menu_state = 0;
        }
    }
    else{
        if  (RCREG == 0x0D){
            menu_state = 0;
            EUSART_enviacadena(msg_ingreso_ack);
            EUSART_siguientelinea();
            vis_menu();
            EUSART_siguientelinea();
        }
        else{
            msg_input[num_char] = RCREG;
            EUSART_enviachar('*');
            num_char++;
        }
    }
}
```

Implementación

Fin de la sesión