MANUAL TÉCNICO

LIBRERÍA PARA MANEJO DE LCD 2X16 EN MODO 4-BITS

FACULTAD DE INGENIERIA UAEM

MATERIA: SISTEMAS EMBEBIDOS

PROFESOR: BENJAMIN PEREZ CLAVEL

INTEGRANTES DEL EQUIPO

ULISES BECERRIL VALDÉS

JOSE BRYANT ALVAREZ MORALES

YANELA MICHELLE TRINIDAD CALIXTO

FECHA DE ENTREGA

VIERNES 07 DE JUNIO DEL 2024

Contenido

INTRODUCCIÓN

Este manual técnico detalla el desarrollo y uso de una librería para controlar un LCD de 2 filas por 16 caracteres (2x16) en modo de 4-bits con un microcontrolador PIC18F46K22. La librería se entrega en un archivo con extensión `.C` y es adaptable a otros modelos de microcontroladores mediante la definición de constantes simbólicas para los pines utilizados.

Para poder realizar la librería que aquí se explica fue necesario tener conocimientos del manejo adecuado de los registros del PIC, así como conocer a fondo las formas y secuencias de configuración de la pantalla LCD, es importante resaltar que el PIC es un dispositivo que tiene una velocidad de procesamiento mucho mayor a la de cualquier pantalla LCD y es necesario considerar estos tiempos de espera, es por ello que dentro de las funciones incluidas en esta librería se consideran delays en diversos puntos, los cuales asumimos para poder asegurar un correcto funcionamiento de la pantalla LCD.

Tambien es importante resaltar el hecho que de estas librerías están diseñadas específicamente para el PIC antes mencionado, además de que fueron construidas usando lenguaje C para el compilador XC8 de Microchip; pueden ser adaptadas a otro microcontrolador, pero es importante tomar en cuenta estas consideraciones.

REQUERIMIENTOS TÉCNICOS

HARDWARE

- Microcontrolador: PIC18F46K22
- LCD: Pantalla LCD de 2 filas por 16 caracteres (HD44780 o compatible)
- Conexiones:
- Datos: 4 bits (D4, D5, D6, D7)
- Control: RS, RW, E

SOFTWARE

- Compilador: MPLAB XC8: El MPLAB XC8 es un compilador C optimizado para microcontroladores PIC de 8 bits, desarrollado por Microchip Technology. Este compilador convierte el código fuente escrito en el lenguaje de programación C a un lenguaje de máquina que puede ser ejecutado por los microcontroladores PIC.

Características Principales

- Optimización del Código: Ofrece varias opciones de optimización que mejoran el rendimiento y reducen el tamaño del código.
- Compatibilidad: Soporta una amplia gama de microcontroladores PIC de 8 bits.
- Bibliotecas Estándar: Incluye bibliotecas estándar de C y bibliotecas específicas para el manejo de periféricos del microcontrolador.
- Compatibilidad con MPLAB X IDE: Se integra perfectamente con el entorno de desarrollo MPLAB X IDE para proporcionar una experiencia de desarrollo coherente y eficiente.
- Soporte Técnico: Microchip ofrece soporte técnico y actualizaciones periódicas del compilador.
- Entorno de Desarrollo: MPLAB X IDE

HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

- MPLAB X IDE: Entorno de desarrollo integrado utilizado para escribir y depurar el código.
- MPLAB XC8: Compilador C para microcontroladores PIC de 8 bits.
- Proteus: Software de simulación utilizado para validar la funcionalidad del LCD antes de la implementación física.

DESARROLLO

ESPECIFICACIONES DE PINES USADOS

Se definen los pines del microcontrolador que manejarán las señales RS, RW, E y el bus de datos. Estas definiciones se realizan mediante constantes simbólicas para facilitar la adaptación a otros microcontroladores:

```
#define D4 LATDbits.LATD0
#define D5 LATDbits.LATD1
#define D6 LATDbits.LATD2
#define D7 LATDbits.LATD3

#define RS LATDbits.LATD4
#define RW LATDbits.LATD5
#define E LATDbits.LATD6
```

DESCRIPCIÓN DE PINES

- D4-D7:** Pines de datos (4 bits de datos)
- RS:** Registro de Selección (0 = Comando, 1 = Datos)
- RW:** Lectura/Escritura (0 = Escritura, 1 = Lectura)
- E:** Habilitación (Activado en flanco de subida)

CONSTRUCCIÓN DEL CÓDIGO

PROTOTIPOS DE LAS FUNCIONES

Las funciones implementadas en la librería son las siguientes:

```
void start_LCD4b(void);
void clrs_LCD4b(void);
void wchar_LCD4b(unsigned char);
void wstring_LCD4b(char*);
void gotoxy_LCD4b(int, int);
void shift_L_LCD4b(int);
void shift_R_LCD4b(int);
void blinkc_LCD4b(bool);
void showc_LCD4b(bool);
void enviarNibble(unsigned char);
void enviarByte (unsigned char);
```

FUNCIÓN `MAIN`

Se incluye una función `main()` que demuestra el uso de la librería:

```c
void main(void)
{
    start_LCD4b();
    while(1)
    {
        //Demostración de funcionamiento
        //Escribir caracter
        wchar_LCD4b('H');
        wchar_LCD4b('o');
        wchar_LCD4b('l');
        wchar_LCD4b('a');
        wchar_LCD4b('!');
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);

        //Escribir cadena
        wstring_LCD4b("Ejemplo de");
        gotoxy_LCD4b(2,1);
        wstring_LCD4b("cadena");
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);

        //Cambio de posición del cursor
        wstring_LCD4b("Cursor en 2,8");
        gotoxy_LCD4b(2,8);
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);
        wstring_LCD4b("Cursor en 1,4");
        gotoxy_LCD4b(1,4);
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);

        //Desplazamiento de texto a la izquierda
        wstring_LCD4b("Desplazamiento de texto a la izquierda");
        shift_L_LCD4b(25);
        clrs_LCD4b();
        __delay_ms(500);
```

```
        //Desplazamiento de texto a la derecha
        wstring_LCD4b("Desplazamiento de texto a la derecha");
        shift_R_LCD4b(25);
        clrs_LCD4b();
        __delay_ms(500);

        //Cursor blink apagado
        wstring_LCD4b("Blink apagado");
        blinkc_LCD4b(false);
        __delay_ms(5000);
        clrs_LCD4b();
        wstring_LCD4b("Blink encendido");
        blinkc_LCD4b(true);
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);

        //Cursor apagado
        wstring_LCD4b("Cursor apagado");
        showc_LCD4b(false);
        __delay_ms(5000);
        clrs_LCD4b();
        wstring_LCD4b("Cursor");
        gotoxy_LCD4b(2,1);
        wstring_LCD4b("encendido");
        showc_LCD4b(true);
        __delay_ms(5000);
        clrs_LCD4b();
        __delay_ms(500);
    }//FIN WHILE

    return;
}//FIN MAIN
```

IMPLEMENTACIÓN DE FUNCIONES

Función `start_LCD4b`

Descripción:Inicializa el LCD con las configuraciones por defecto: canal de comunicación de 4 bits, pantalla con 2 líneas y 16 caracteres, cursor encendido y parpadeando.

Implementación:

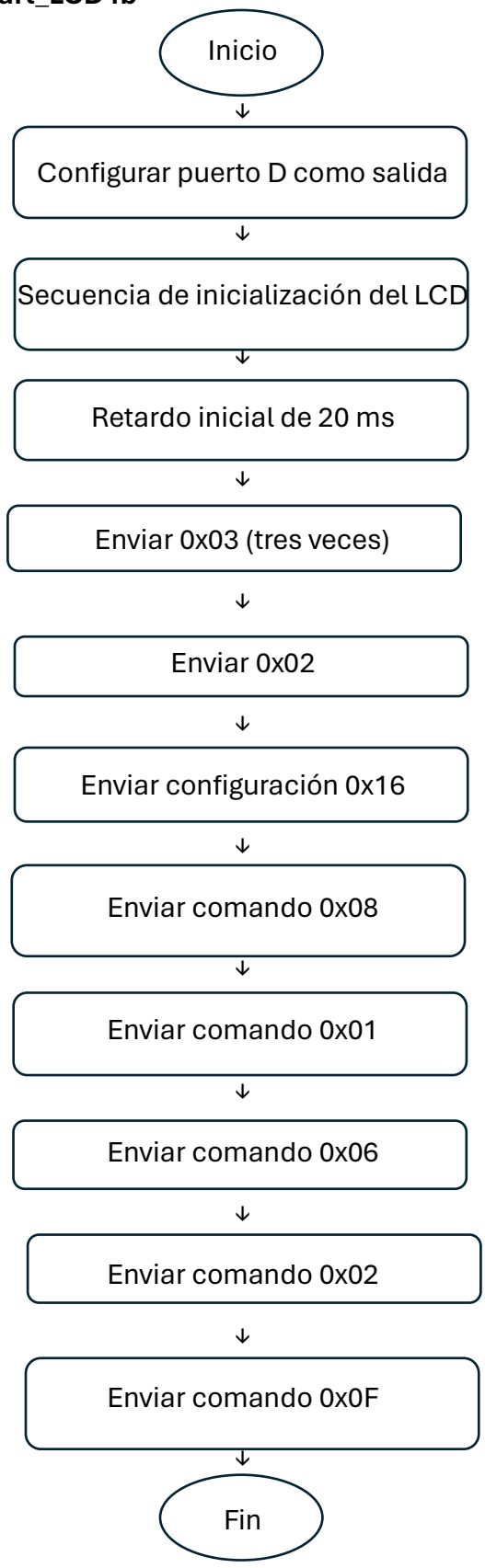
```
void start_LCD4b(){
    //CONFIGURACION DE PUERTOS
    TRISD = 0x00;   //PUERTO D SALIDA
    ANSELD = 0X00;  //PUERTO D SALIDA DIGITAL

    //COMANDOS PARA ENCENDER PANTALLA
    RS = 0;
    RW = 0;
    __delay_ms(20);
    enviarNibble(0x03);
    enviarNibble(0x03);
    enviarNibble(0x03);

    //SETEAR A 4BITS (SECUENCIA DE INICIALIZACION)
    enviarNibble(0x02); //4bits
    enviarByte(0x16);   //2 lineas y caracteres
    enviarByte(0x08);   //Apagar
    enviarByte(0x01);   //Limpiar
    enviarByte(0x06);   //Shift
    enviarByte(0x02);   //Return home

    //ENCENDER
    enviarByte(0x0F);
}//Fin start_LCD4b
```

Diagrama de Flujo: `start_LCD4b`



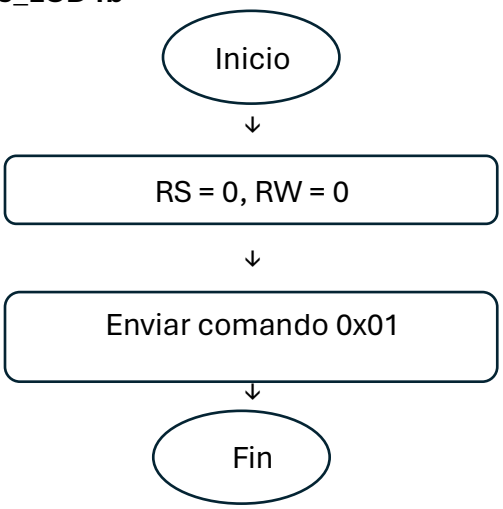
Función `clrs_LCD4b`

Descripción:Borra el contenido de la pantalla y regresa el cursor a la posición inicial (primera fila y primera columna).

Implementación:

```
void clrs_LCD4b(){
    RS = 0;
    RW = 0;
    enviarByte(0x01);
}//Fin clrs_LCD4b
```

Diagrama de Flujo: `clrs_LCD4b`



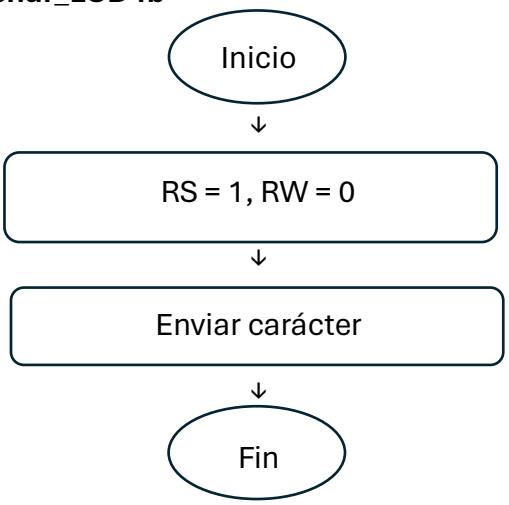
Función `wchar_LCD4b`

Descripción: Escribe un carácter en la pantalla del LCD.

Implementación:

```
void wchar_LCD4b(unsigned char c){
    RS = 1;
    RW = 0;
    enviarByte(c);
}//Fin wchar_LCD4b
```

Diagrama de Flujo: `wchar_LCD4b`



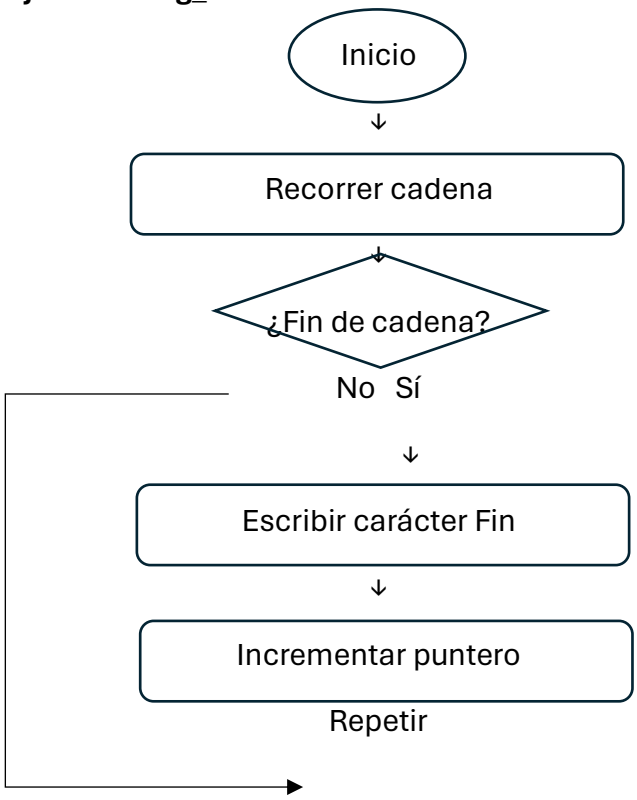
Función `wstring_LCD4b`

Descripción: Escribe una cadena de caracteres en la pantalla del LCD.

Implementación:

```
void wstring_LCD4b(char *cadena){
    for(int i = 0; cadena[i] != '\0'; i++){
        wchar_LCD4b(cadena[i]);
    }
}//Fin wstring_LCD4b
```

Diagrama de Flujo: `wstring_LCD4b`



Función `gotoxy_LCD4b`

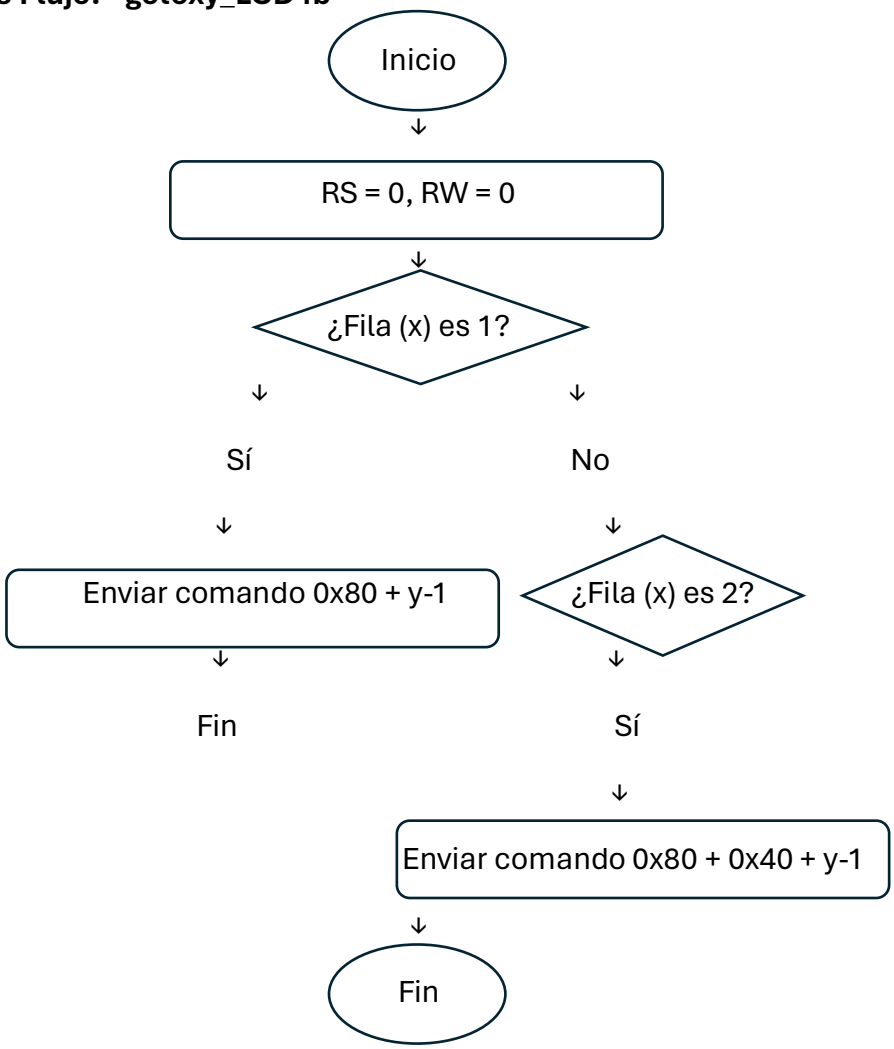
Descripción: Posiciona el cursor en la fila y columna especificadas.

Implementación:

```
void gotoxy_LCD4b(int x, int y){
    RS = 0;
    RW = 0;

    if(x == 1){
        enviarByte(0x80 + (0x00 + (y-1)));
    }else if(x == 2){
        enviarByte(0x80 + (0x40 + (y-1)));
    }
}//Fin gotoxy_LCD4b
```

Diagrama de Flujo: `gotoxy_LCD4b`



Función `shift_L_LCD4b`

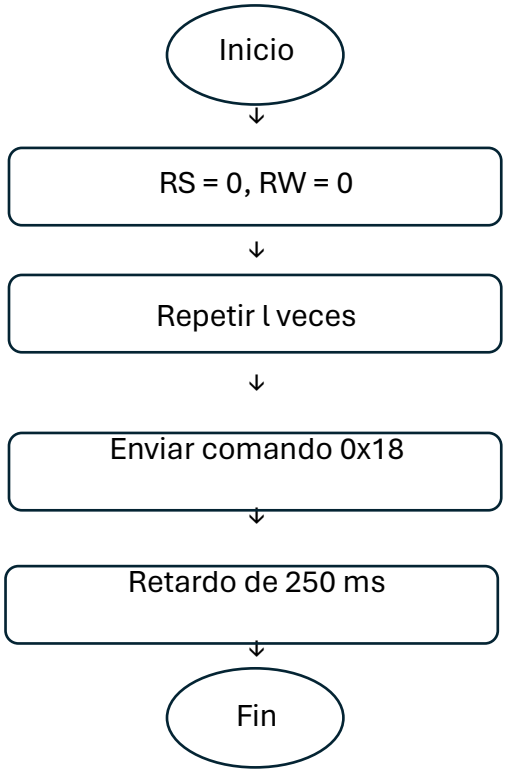
Descripción: Desplaza el texto de la pantalla hacia la izquierda.

Implementación:

```
void shift_L_LCD4b(int l){
    RS = 0;
    RW = 0;

    for(int i = 0; i<l; i++){
        enviarByte(0x18);
        __delay_ms(250);
    }
}//Fin shift_L_LCD4b
```

Diagrama de Flujo: `shift_L_LCD4b`



Función `shift_R_LCD4b`

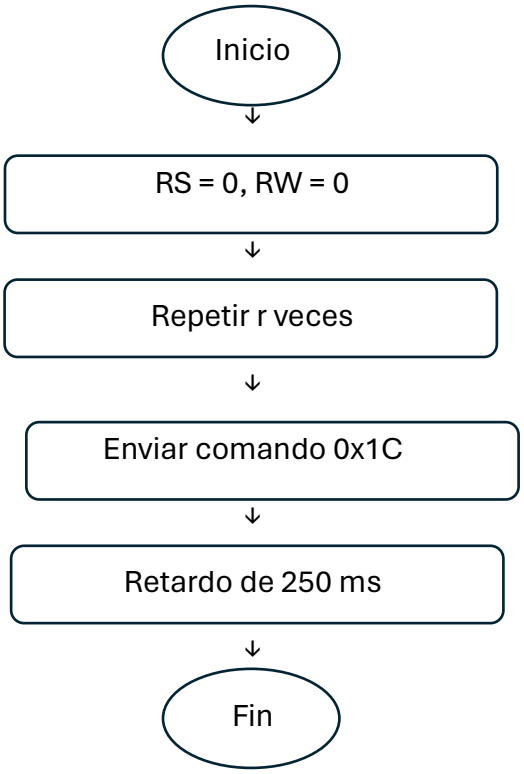
Descripción:Desplaza el texto de la pantalla hacia la derecha.

Implementación:

```
void shift_R_LCD4b(int r){
    RS = 0;
    RW = 0;

    for(int i = 0; i<r; i++){
        enviarByte(0x1C);
        __delay_ms(250);
    }
}//Fin shift_R_LCD4b
```

Diagrama de Flujo: `shift_R_LCD4b`



Función `blinkc_LCD4b`

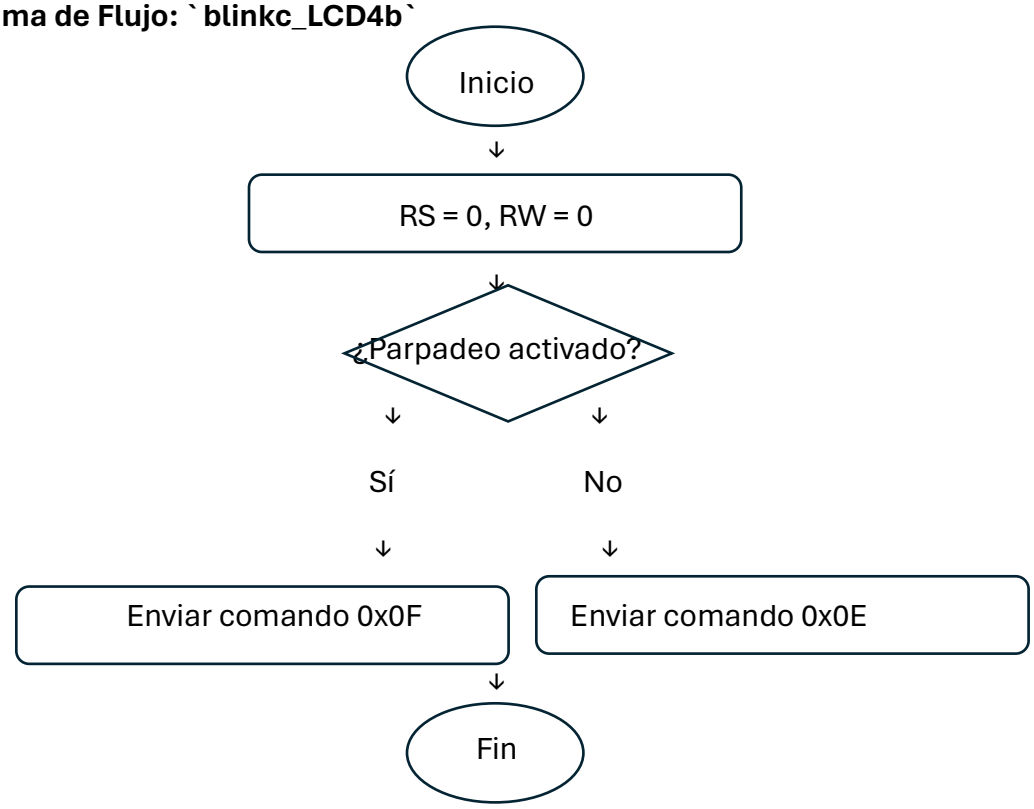
Descripción: Controla el parpadeo del cursor en la pantalla del LCD.

Implementación:

```
void blinkc_LCD4b(bool b){
    RS = 0;
    RW = 0;
    if(b){
        enviarByte(0x0F);
    }else{
        enviarByte(0x0E);
    }

}//Fin blinkc_LCD4b
```

Diagrama de Flujo: `blinkc_LCD4b`



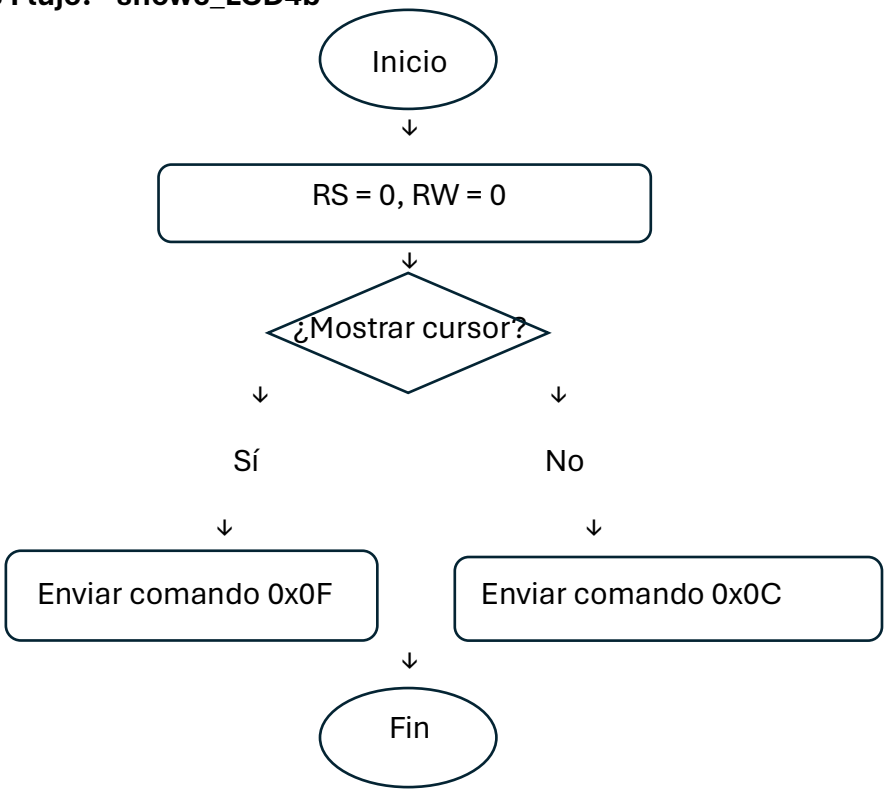
Función `showc_LCD4b`

Descripción: Controla la visibilidad del cursor en la pantalla del LCD.

Implementación:

```
void showc_LCD4b(bool b){
    RS = 0;
    RW = 0;
    if(b){
        enviarByte(0x0F);
    }else{
        enviarByte(0x0C);
    }
}//Fin showc_LCD4b
```

Diagrama de Flujo: `showc_LCD4b`



Función `enviarNibble`

Descripción:Envía un nibble (4 bits) al LCD.

Implementación:

```
void enviarNibble (unsigned char dato){
    D4 = (dato >> 0) & 0x01;
    D5 = (dato >> 1) & 0x01;
    D6 = (dato >> 2) & 0x01;
    D7 = (dato >> 3) & 0x01;

    E = 1;
    __delay_ms(30);
    E = 0;
    __delay_ms(30);
}//FIN ENVIAR NIBBLE
```

Función `enviarByte`

Descripción: Envía un byte al LCD, dividiéndolo en dos nibbles.

Implementación:

```
void enviarByte (unsigned char dato){
    enviarNibble(dato >> 4);
    enviarNibble(dato & 0x0F);
}//FIN ENVIAR BYTE
```

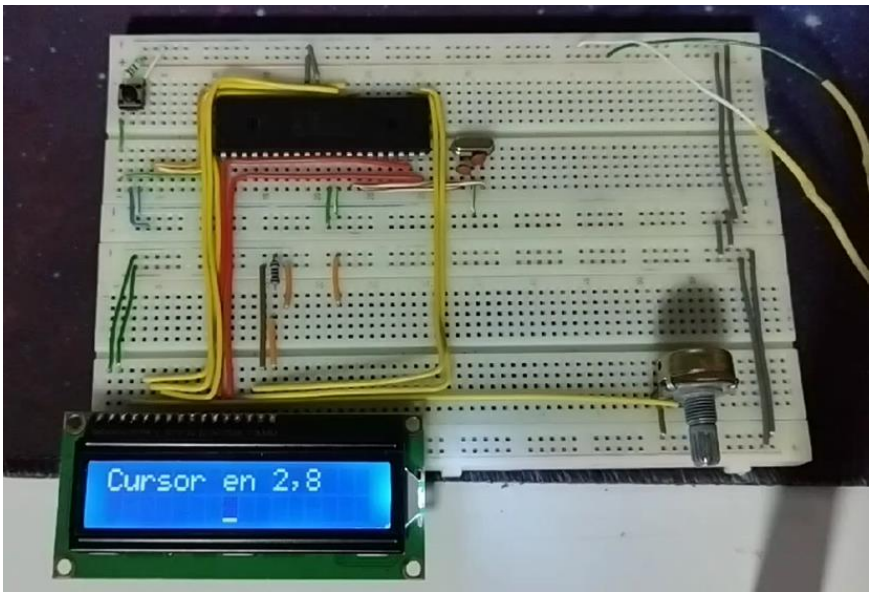
FUNCIONAMIENTO

El funcionamiento de la librería se basa en la comunicación de 4 bits con el LCD, donde los datos y comandos se envían en dos partes (nibbles) de 4 bits cada una. Las funciones de control y escritura permiten interactuar con el LCD de manera sencilla, manejando la configuración, escritura de caracteres y cadenas, posicionamiento del cursor y desplazamiento del texto.

Cada una de las funciones de la librería hace uso de a función ***enviarByte*** para poder comunicarse con la pantalla LCD, de esta manera reducimos la cantidad de código necesaria para poder implementar las librerías. Tambien es importante saber que las funciones que envían los bytes no hacen manipulación de los pines ***RS*** o ***RW*** por lo que es necesario hacer los cambios en estos pines según lo necesario.

Implementación

La librería ha sido probada con éxito en un entorno de simulación con Proteus y en un montaje físico utilizando un PIC18F46K22 y una pantalla LCD 2x16. Las pruebas incluyeron la escritura de caracteres y cadenas, el desplazamiento del texto, la configuración del cursor (parpadeo y visibilidad) y la limpieza de la pantalla.



CONCLUSIONES

La librería proporcionada ofrece una solución eficiente y fácil de usar para controlar pantallas LCD 2x16 en modo de 4-bits con microcontroladores PIC18F46K22. La estructura modular del código y el uso de constantes simbólicas para la configuración de pines permiten una fácil adaptación a otros modelos de microcontroladores. Esta librería puede ser expandida para incluir funciones adicionales según las necesidades del proyecto, como el manejo de caracteres personalizados o la integración con otros periféricos.

Este manual técnico proporciona una guía completa para la implementación y uso de la librería, asegurando un desarrollo eficiente y efectivo de proyectos que requieran interacción con pantallas LCD.

VIDEO DE IMPLEMENTACIÓN

Para poder demostrar el uso de esta librería hicimos uso de la función ***main*** mostrada al inicio de este documento, al realizar la ejecución obtuvimos los resultados mostrados en el siguiente video:

https://drive.google.com/file/d/1Sa70japPdp0MvVlQFGBjtO_fu8svMI4K/view?usp=sharing