PRÁCTICA 2

Índice

1. Objetivos y fases de la práctica

- Fase 1: Mandar texto a cualquier posición en un display tipo HD44780.
- Fase 2: Definir un nuevo dibujo en la memoria del display y visualizarlo en pantalla.
- Fase 3: Manejar los LED y el botón de la placa al igual que se hizo en la práctica 1.
- Fase 4: Sustituir en la librería del display LCD, que se proporciona en esta práctica, las llamadas a las funciones GPIO LL por funciones equivalentes GPIO HAL.

2. Introducción

La mayoría de los sistemas empotrados necesitan alguna interfaz para interactuar con el usuario. Ya sea una lavadora, un horno, un ascensor… todos requieren algún tipo de elemento de manipulación. Las interfaces más simples son los botones y las luces (LEDs), pero a veces se necesita mostrar más información, y en estos casos, las pantallas tipo display son la solución. En esta práctica, veremos cómo funcionan los displays más sencillos que se suelen usar con los microcontroladores. Hablaremos de dispositivos universales y fáciles de conseguir (aunque hay otros tipos de displays que son específicos para ciertos aparatos y mensajes, pero esos no los trataremos en esta práctica).

Además, vamos a introducir la placa de desarrollo que utilizaremos en casi todas las prácticas de este curso y el próximo. También queremos aclarar las diferencias entre las librerías para el uso de periféricos que proporciona STMicroelectronics. Estas diferencias se hacen evidentes al usar una u otra librería, especialmente cuando surge un problema y comprobamos que una no ofrece la funcionalidad que tiene la otra.

Los objetivos académicos de esta práctica son: estudiar y entender el funcionamiento de los displays tradicionales usados con los microcontroladores, conocer los aspectos

más básicos de la placa de desarrollo del curso y entender las diferencias entre las diversas librerías que los fabricantes proporcionan para manejar los periféricos en los microcontroladores tipo Cortex.

Para alcanzar estos objetivos académicos, planteamos los siguientes objetivos prácticos:

1. Enviar texto a cualquier posición en un display tipo HD44780.
2. Definir un nuevo dibujo en la memoria del display y visualizarlo en pantalla.
3. Manejar los LEDs y el botón de la placa, como se hizo en la práctica 1.
4. Sustituir en la librería del display LCD, proporcionada en esta práctica, las llamadas a las funciones GPIO LL por funciones equivalentes GPIO HAL.

Junto con este boletín de prácticas se adjuntan los siguientes archivos:

- Manual de la placa B-L475E-IOT01A
- Manual del Hitachi HD44780
- Librerías para manejar el HD44780 (dos archivos comprimidos en un zip).
- Documento sobre las librerías HAL y LL de STM de 2173 páginas, solo para consultar en caso de dudas.

3. Desarrollo de la práctica

▪ En la fase 1 hemos colocado el shield del display sobre la placa, y tras ello ha sido conectada nuestro PC a través de USB.

Luego, ya dentro del IDE, hemos configurado los pines del microcontrolador para que coincidan con los provistos por la librería LCD. Haciendo uso de la siguiente tabla:

Nombre actual	Arduino	Pin MCU	Tipo de puerto		Nombre nuevo que hay que poner
ARD_D10	D10	PA2	Output Push Pull	Low	Led_LCD
ARD_D9	D9	PA15	Output Push Pull	Low	E_LCD
ARD_D6	D6	PB1	Output Push Pull	Low	D6_LCD
ARD_D5	D5	PB4	Output Push Pull	Low	D5_LCD
ARD_D8	D8	PB2	Output Push Pull	Low	RS_LCD
ARD_D7	D7	PA4	Output Push Pull	Low	D7_LCD
ARD_D4	D4	PA3	Output Push Pull	Low	D4_LCD

Tras configurar los pines, debemos activar la librería LL. Para ello iremos a “Project Manager” → “Advanced Settings” → “GPIO” → “LL”.

Tras esto, procederemos a añadir los ficheros adjuntos a la práctica, “hd44780.c” en la carpeta src y “hd44780.h” en la carpeta inc de nuestro proyecto. Para finalizar,

deberemos añadir en “main.c”, en la sección de “/* USER CODE BEGIN Includes */” el siguiente código:

```
#include "hd44780.h"
```

Una vez añadido, vamos a realizar una prueba para comprobar si todos los pasos se han realizado correctamente. Para ello añadiremos en “main.c”, dentro del while(1) el siguiente código:

```
lcd_reset();
lcd_display_settings(1, 0, 0);
lcd_clear();


lcd_print("Hola Mundo");
```

Ejecutamos el código, nos daremos cuenta que la pantalla no se verá bien pues estará ennegrecida. Para solucionarlo añadiremos en “main.c”, dentro del while(1) y previo al anterior código,la siguiente línea:

```
LL_GPIO_SetOutputPin(GPIOA, Led_LCD_Pin);
```

Por último, vamos a mostrar en pantalla “dos muñecos haciendo gimnasia con brazos y piernas”. Para ello, vamos a añadir en “main.c”, dentro del while(1) el siguiente código:

```
moveToXY(1,7);
lcd_write(0,1);
moveToXY(1,8);
lcd_write(0,1);
HAL_Delay(1000);

moveToXY(1,7);
lcd_write(1,1);
moveToXY(1,8);
lcd_write(1,1);
HAL_Delay(1000);
```

▪ En la fase 2, queremos apagar un LED al pulsar un botón mientras los muñecos siguen haciendo gimnasia. Añadimos el siguiente código en "main.c", dentro del while(1) y después del código anterior:

```
if (LL_GPIO_IsInputPinSet(BUTTON_EXTI13_GPIO_Port, BUTTON_EXTI13_Pin)) {

        LL_GPIO_SetOutputPin(LED2_GPIO_Port, LED2_Pin);

} else {

        LL_GPIO_ResetOutputPin(LED2_GPIO_Port, LED2_Pin);
```

```
}
```

Esto hará que ambas funciones funcionen, pero no correctamente, pues los muñecos se moverán muy rápidos. Para evitar esto, añadimos este código en "stm32f1xx_it.c", dentro de "/* USER CODE BEGIN SysTick_IRQn 0 */".

Para la segunda parte de la Fase 2, que viene a ser casi la Fase 4 de la primera práctica, queremos que el botón sea intermitente. Para ello, añadimos el siguiente código en "stm32f1xx_it.c", dentro de "/* USER CODE BEGIN SysTick_IRQn 0 */"

```
static char estado = 0;

if (estado == 0) {

        LL_GPIO_ResetOutputPin(LED2_GPIO_Port, LED2_Pin);

        estado++;

} else if (estado == 32) {

        LL_GPIO_SetOutputPin (LED2_GPIO_Port, LED2_Pin);
        estado++;

} else if (estado == 64) {

        estado = 0;

} else {

        estado++;


}
```

Al usar interrupciones distintas, no tenemos por qué comentar el código del botón. El LED estará cambiando su valor intermitentemente,in.

▪ En la fase 3 repetiremos la Fase 1 de esta práctica, pero usando librerías HAL. Debemos ir al archivo de la librería "hd44780.c" y cambiar todas las funciones que hacen referencia a la librería LL por sus equivalentes en la librería HAL. Aquí están los fragmentos de código modificados, sin los comentarios:

```
void lcd_clock(void) {


        HAL_GPIO_WritePin(CLOCK_PORT, LCD_CLOCK, 1);
```

```
        HAL_Delay(1);
        HAL_GPIO_WritePin(CLOCK_PORT, LCD_CLOCK, 0);
        HAL_Delay(1);

}
void lcd_reset(void) {

        HAL_GPIO_WritePin(RS_PORT, LCD_RS, 0);
        HAL_GPIO_WritePin(LCD_PORT7, LCD_7, 0);
        HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 0);
        HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 0);
        HAL_GPIO_WritePin(LCD_PORT56, LCD_6, 0);
        HAL_GPIO_WritePin(CLOCK_PORT, LCD_CLOCK, 0);

        HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 1);
        HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 1);
        lcd_clock();
        lcd_clock();
        lcd_clock();

        HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 0);
        lcd_clock();

        HAL_GPIO_WritePin(Led_LCD_GPIO_Port, Led_LCD_Pin, 1);

}

void lcd_write(uint8_t byte, uint8_t rs) {

        if((byte >> 4 ) & 1)
          HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 1);
        else
                HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 0);

        if((byte >> 5 ) & 1)
                HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 1);
        else
                HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 0);

        if((byte >> 6 ) & 1)
                HAL_GPIO_WritePin(LCD_PORT56, LCD_6, 1);
        else
                HAL_GPIO_WritePin(LCD_PORT56, LCD_6, 0);

        if((byte >> 7 ) & 1)
                HAL_GPIO_WritePin(LCD_PORT7, LCD_7, 1);
        else
```

```
            HAL_GPIO_WritePin(LCD_PORT7, LCD_7, 0);

        if(rs)
            HAL_GPIO_WritePin(RS_PORT, LCD_RS, 1);
        else
            HAL_GPIO_WritePin(RS_PORT, LCD_RS, 0);

        lcd_clock();

        if(byte & 1)
                    HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 1);
        else
                    HAL_GPIO_WritePin(LCD_PORT4, LCD_4, 0);

        if((byte >> 1 ) & 1)
                    HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 1);
        else
                    HAL_GPIO_WritePin(LCD_PORT56, LCD_5, 0);

        if((byte >> 2 ) & 1)
                    HAL_GPIO_WritePin(LCD_PORT56, LCD_6, 1);
        else
                    HAL_GPIO_WritePin(LCD_PORT56, LCD_6, 0);

        if((byte >> 3 ) & 1)
                    HAL_GPIO_WritePin(LCD_PORT7, LCD_7, 1);
        else
            HAL_GPIO_WritePin(LCD_PORT7, LCD_7, 0);

        lcd_clock();

}
```

Para verificar que las modificaciones funcionan correctamente, añadimos en "main.c", dentro del while(1), el siguiente código:HAL_GPIO_WritePin(GPIOA, Led_LCD_Pin, GPIO_PIN_SET);

```
lcd_reset();
lcd_display_settings(1, 0, 0);
lcd_clear();


lcd_print("Hola Mundo");
```

4. Conclusión

Todas las fases se completaron con éxito siguiendo las instrucciones de la hoja de la práctica, aunque esta vez no fue tan “guiada” como la anterior.

Personalmente, me ha parecido una práctica interesante, ya que podemos ver visualmente lo que programamos gracias al LCD, y no es simplemente un texto que aparece en la consola, haciendo más ameno el proceso. En general, la práctica ha sido sencilla, aunque la parte de cambiar las librerías de LL a HAL, a pesar de ser algo no muy complicado, era un arduo trabajo ya que determinar qué líneas de código debían ser modificadas y cuáles no se me hizo largo.