Sistemas Basados en Microprocesador

Bloque 2 – Práctica 3

Interfaces de usuario: Display LCD

Parte I

Departamento de Ingeniería Telemática y Electrónica
Universidad Politécnica de Madrid

Introducción y objetivos de la práctica.

El objetivo de esta práctica es desarrollar las funciones básicas que permitan el manejo de un display gráfico de 128 x 32 pixels de cristal líquido (LCD) por la tarjeta STM-34F429ZI. Este display tiene interfaz SPI, está disponible en la tarjeta *mbed App Board.*

Una vez terminada la práctica debería contar con una serie de ficheros (*.c, *.h) que permitan el uso del display en prácticas posteriores. Por ello es de suma importancia que todo el código desarrollado esté lo suficientemente documentado y verificado.

Documentación

Además de los documentos empleados en las prácticas anteriores debe disponer de los siguientes:

- *mbed App Board* documentation and schematics:

  https://os.mbed.com/components/mbed-Application-Board/

  https://os.mbed.com/media/uploads/chris/mbed-014.1_b.pdf

- Display datasheet https://www.newhavendisplay.com/specs/NHD-C12832A1Z-FSW-FBW-3V3.pdf
- Display controller http://www.lcd-module.de/eng/pdf/zubehoer/st7565r.pdf
- CMSIS driver specification http://www.keil.com/pack/doc/CMSIS/Driver/html/index.html
- CMSIS SPI Interface:

  http://www.keil.com/pack/doc/CMSIS/Driver/html/group__spi__interface__gr.html

Proyecto 1. Inicialización del SPI y LCD

Cree un proyecto (P3_1) en Keil uVision para la tarjeta de desarrollo, utilizando CMSIS, que incluya una función (llamada **`LCD_reset`**) que permita configurar el interfaz SPI de acuerdo a las siguientes especificaciones.

- El proyecto se debe crear partiendo de cero y basarse en CMSIS, empleando el driver para gestionar el interface SPI (SP1).
- Debe configurar el SPI con los siguientes parámetros. Cada uno de estos parámetros tiene definida una etiqueta (en mayúscula y entre paréntesis) que ayuda a su configuración
  - Modo master (ARM_SPI_MODE_MASTER)
  - CPOL1 y CPHA1 (ARM_SPI_CPOL1_CPHA1)
  - Organización de la información MSB a LSB (ARM_SPI_MSB_LSB)
  - 8 bits de datos (ARM_SPI_DATA_BITS(8))
  - Frecuencia del SCLK, 20MHz

- Tres pines del GPIO se deben configurar como salida, puestas a nivel alto inicialmente, para ser usadas como señal de RESET, CS y A0 respectivamente. Los pines a emplear debe identificarlos tras analizar los esquemáticos de las placas.
- Se debe generar el pulso de reset para el LCD con la duración adecuada (analice los datos del manual) antes de proceder a su configuración.
- Después del reset debe haber un retardo de 1 ms antes de comenzar la secuencia de comandos de inicialización del LCD.

Para la inicialización de los dispositivos y antes de que la aplicación realice su funcionalidad suele ser necesario realizar esperas para que los dispositivos finalicen su configuración. Para obtener estas esperas (por ejemplo hasta que un LCD finalice su proceso de reset). en la asignatura microprocesadores se empleaban objetos tipo Timers, Tickers o Timeout. Aunque existe la función HAL_Delay, en esta práctica no está permitido su uso y será necesario la implementación de una función **`void delay (uint32_t n_microsegundos).`**

Dicha función debe utilizar el timer 7, en un modo básico sin interrupciones, esperando hasta que se active el flag que indica que ha llegado al valor final de cuenta. El seudocódigo asociado a esta función puede ser el siguiente:

```
void delay(uint32_t n_microsegundos)
{
  // Configurar y arrancar el timer para generar un evento
  // pasados n_microsegundos

  // Esperar a que se active el flag del registro de Match
  // correspondiente

  // Borrar el flag,

  // Parar el Timer y ponerlo a 0 para la siguiente llamada a
  // la función

}
```

Para la inicialización del LCD se recomienda seguir la siguiente secuencia dentro de la función **`LCD_reset`**:

1. Inicialización y configuración del driver SPI para gestionar el LCD
2. Configurar los pines de IO que sean necesarios y programar su valor por defecto
3. Generar la señal de reset

Adjunte en la carpeta del proyecto una captura del analizador, en la que se observen todas las señales involucradas el reset.

Responda a las siguientes preguntas:

1. ¿Cuál es la máxima frecuencia que puede utilizarse en la señal de reloj del display? ¿Dónde está indicada esta información?
2. ¿Quién y cómo se genera la señal de CS para el display?
3. ¿Por qué se ha programado el modo CPOL1 y CPHA1? Justifíquelo con el cronograma del display disponible en la hoja de catálogo.
4. ¿Qué duración debe tener el pulso de RESET? Para responder a esta pregunta debe buscar la información en la hoja de catálogo del display incluido en la información técnica del fabricante.

Proyecto 2. Funciones para la gestión del LCD

Cree un proyecto (P3_2) a partir del P3_1 sobre el que tiene que añadir las siguientes funciones en C. Para ello tiene que completar el código realizando las operaciones que sean necesarias.

LCD_wr_data: Función que escribe un dato en el LCD.

```
void LCD_wr_data(unsigned char data)
{
  // Seleccionar CS = 0;

  // Seleccionar A0 = 1;

  // Escribir un dato (data) usando la función SPIDrv->Send(…);

  // Esperar a que se libere el bus SPI;

  // Seleccionar CS = 1;


}
```

LCD_wr_cmd: Función que escribe un comando en el LCD.

```
void LCD_wr_cmd(unsigned char cmd)
{
  // Seleccionar CS = 0;

  // Seleccionar A0 = 0;

  // Escribir un comando (cmd) usando la función SPIDrv->Send(…);

  // Esperar a que se libere el bus SPI;

  // Seleccionar CS = 1;

}
```

Proyecto 3. Configuración del LCD

Basándose en el proyecto realizado en el proyecto 2 (P3_2) cree un nuevo proyecto (P3_3) que incluya una función denominada **`LCD_init`** que envíe al LCD la siguiente secuencia de operaciones:

`wr_cmd(0xAE);`	Display off
`wr_cmd(0xA2)`	Fija el valor de la relación de la tensión de polarización del LCD a 1/9
`wr_cmd(0xA0);`	El direccionamiento de la RAM de datos del display es la normal
`wr_cmd(0xC8);`	El scan en las salidas COM es el normal
`wr_cmd(0x22);`	Fija la relación de resistencias interna a 2
`wr_cmd(0x2F);`	Power on
`wr_cmd(0x40);`	Display empieza en la línea 0
`wr_cmd(0xAF);`	Display ON
`wr_cmd(0x81);`	Contraste
`wr_cmd(0x??);`	Valor Contraste
`wr_cmd(0xA4);`	Display all points normal
`wr_cmd(0xA6);`	LCD Display normal

Consulte y explique el significado de las diferentes operaciones del proceso de inicialización del display indicadas en la tabla anterior. Cuando pruebe el código verá que la pantalla del display no visualiza ninguna información porque todavía no se ha rellenado su memoria con información.
Realice un cronograma en un documento gráfico que refleje el comportamiento de las señales utilizadas la secuencia de inicialización del LCD.
Adjunte en la carpeta del proyecto una captura del analizador, en la que se observen todas las señales involucradas en la inicialización y compárelo con el análisis que ha realizado de manera teórica anteriormente. Utilice la facilidad que ofrece la herramienta PulseView para decodificar líneas que pertenecen a un bus SPI.

Proyecto 4. Escritura de datos en el LCD

Añada al proyecto anterior una rutina que permita copiar la información de un array de datos global (`buffer`) tipo unsigned char de 512 elementos a la memoria del display LCD. Cada uno de los bits de este array (4096) representa el estado de uno de los 128x32 píxeles de la pantalla.

```
void LCD_update(void)
{
    int i;
    LCD_wr_cmd(0x00);      // 4 bits de la parte baja de la dirección a 0
    LCD_wr_cmd(0x10);      // 4 bits de la parte alta de la dirección a 0
    LCD_wr_cmd(0xB0);      // Página 0

    for(i=0;i<128;i++){
        LCD_wr_data(buffer[i]);
        }


    LCD_wr_cmd(0x00);      // 4 bits de la parte baja de la dirección a 0
    LCD_wr_cmd(0x10);      // 4 bits de la parte alta de la dirección a 0
    LCD_wr_cmd(0xB1);      // Página 1

    for(i=128;i<256;i++){
        LCD_wr_data(buffer[i]);
        }

    LCD_wr_cmd(0x00);
    LCD_wr_cmd(0x10);
    LCD_wr_cmd(0xB2);      //Página 2


    for(i=256;i<384;i++){
        LCD_wr_data(buffer[i]);
        }

    LCD_wr_cmd(0x00);
    LCD_wr_cmd(0x10);
    LCD_wr_cmd(0xB3);      // Pagina 3


    for(i=384;i<512;i++){
        LCD_wr_data(buffer[i]);
        }

}
```

Pruebe la rutina desarrollada escribiendo en el display un cuadrado de 8x8 pixeles en la parte superior izquierda. Indique los datos que debe enviar. Para poder realizar la prueba basta con que complete el contenido de los datos del array `buffer` desde i=0 hasta i=7.

A continuación, diseñe una letra “A” con un tamaño de 8x8 pixeles y visualícela en la parte superior izquierda del display.

Finalmente realice alguna prueba adicional empleando las páginas que posee el LCD.