```
////////////////////////////////////////////////////////////////////////
//
//  Título:     DriverOLEDSSD1306
//  Autor:      Pablo Crespo de Pablos
//  fecha:      25/04/2020
//  version:    0.0
//
////////////////////////////////////////////////////////////////////////

#ifndef DriverOLEDSSD1306_c
#define DriverOLEDSSD1306_c

////////////////////////////////////////////////////////////////////////
// includes
#include "DriverOLEDSSD1306.h"
//#include ".h"


////////////////////////////////////////////////////////////////////////
// Inicialización de variables


////////////////////////////////////////////////////////////////////////
// Funciones privadas

bool InitialSequence( OLED_t * Oled){


    bool resultado = true;

    #define NUM_SetOff 1
    uint8_t SetOff[NUM_SetOff]={
        (0xAE),//10.1.12 Set Display ON/OFF (AEh/AFh)-
                 Apagar la pantalla
    };

    #define NUM_CONFIG_HARD 9
    uint8_t configHard[NUM_CONFIG_HARD]={

        (0xA1), //Invertir el eje X de pantalla, por si esta girada.
                //Puedes cambiarlo por 0xA0 si necesitas
                  cambiar la orientacion

        (0xC8), //Invertir el eje Y de la pantalla
                //Puedes cambiarlo por 0xC0 si necesitas
                  cambiar la orientacion
        (0x81), //Configurar el contraste
        (0x7F), //Este valor tiene que estar entre
                  0x00 (min) y 0xFF (max)
        (0xA6), //10.1.10 Set Normal/Inverse Display
                  (A6h/A7h)- Poner la pantalla en modo Normal

        (0xD5),(0x80),  //Establecer la velocidad del Oscilador

        (0x8D),(0x14)   //Activar el 'charge pump'
    };

    #define NUM_SCREEN_CONFIG 9
    uint8_t configScreen[NUM_SCREEN_CONFIG]={

        (0xA8),(0x1F),  // 10.1.11 Set Multiplex Ratio (A8h) -
```

```
    Establecer el maximo de filas a 0x3F = 63
    // es decir, ira de 0 a 63, por tanto tenemos 64 filas de
    pixeles
    // En nuestro caso 1F-31

    (0xD3),(0x00),  //Poner el offset a 0

    (0x40), //Poner el comienzo de linea a 0

    (0xDA), //Mapear los pines COM
    (0x40), //Al parecer, la unica configuracion que
    funciona con mi modelo es 0x12, a pesar de que en la
    documentacion dice que hay que poner 0x02
    // #PCPnote: aqui la única que funciona es esta
    pero tiene relaci□on el cmm 0xA8

    (0x20), //10.1.3 Set Memory I2Caddressing Mode (20h)
    (0x00)  //Horizontal I2Caddressing mode (A[1:0]=00b)
    };


#define NUM_COLyPAG 9
uint8_t configColpag[NUM_COLyPAG]={
//  (0xB0), //10.1.13 Set Page Start I2Caddress for Page
              I2Caddressing Mode (B0h~B7h)
    (0x21),(0x00),(0x7F),   // 10.1.4 Set Column I2Caddress
    (21h)- Columnas de 0 a 127 - triple byte command
    (0x22),(0x00),(0x03),   // 10.1.5 Set Page I2Caddress
    (22h) - Paginas de 0 a 3 - triple byte command
    (0xB0),(0x00),(0x10) //10.1.13 Set Page Start I2Caddress
    for Page I2Caddressing Mode (B0h~B7h)
};

#define NUM_SetOnScreen 2
uint8_t SetOnScreen[NUM_SetOnScreen]={

    (0xA4), //10.1.9 Entire Display ON (A4h/A5h) - Este comando
              ordena al chip que active el output de la pantalla
              en funcion del contenido almacenado en su GDDRAM

    (0xAF)  //10.1.12 Set Display ON/OFF (AEh/AFh)- Encender la
              pantalla
    };


/* Inicializar la pantalla, siguiendo las instrucciones
     exactas de la pag. 64 de la documentacion:   */

//Iniciar la comunicacion con la pantalla

// Apagamos la pantalla
Oled->data   = SetOff;
Oled->TXsize = NUM_SetOff;
resultado &= Oled->writeCommand(Oled);

// Configuramos hardware de la pantalla
Oled->data   = configHard;
Oled->TXsize = NUM_CONFIG_HARD;
resultado &= Oled->writeCommand(Oled);

// Configuramos l□a de la pantalla (l□as, COMs, I2Caddressing...)
```

```
    Oled->data   = configScreen;
    Oled->TXsize = NUM_SCREEN_CONFIG;
    resultado &= Oled->writeCommand(Oled);

    // configuramos el nmero donde empiezan las columnas y las filas
    Oled->data   = configColpag;
    Oled->TXsize = NUM_COLyPAG;
    resultado &= Oled->writeCommand(Oled);

    // Encendemos la pantalla
    Oled->data   = SetOnScreen;
    Oled->TXsize = NUM_SetOnScreen;
    resultado &= Oled->writeCommand(Oled);

return resultado;
}


//////////////////////////////////////////////////////////////////
uint8_t I2C_Scan(void) {

    static uint8_t dir;
    HAL_StatusTypeDef res;

    for(uint16_t i = 0; i < 128; i++) {
            res = HAL_I2C_IsDeviceReady(&hi2c1, i << 1, 1, 10);
            if(res == HAL_OK) {
                //Se ha encontrado algo, hay que comprobar que
                  es lo que buscamos...
                // De momento es todo que si
                dir = i << 1;
                return dir;
            }
    }
    return 0xFF;
    //Esta es la manera de indicar que no ha encontrado nada
 }


//////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_init  ( OLED_t * Oled){

    MX_I2C2_Init();
    return InitialSequence(Oled);
}


//////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_start ( OLED_t * Oled){ return true;}


//////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_stop  ( OLED_t * Oled){ return true;}


//////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_read  ( OLED_t * Oled){ return true;}
```

```
////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_writeCommand ( OLED_t * Oled){

    HAL_StatusTypeDef res;

    res= HAL_I2C_Mem_Write(
                Oled->i2c ,
                Oled->I2Caddress ,
                OLED_COMANDO ,
                1 ,
                (uint8_t *)Oled->data ,
                Oled->TXsize ,
                Oled->TimeOut);

    return (res == HAL_OK )? true:false;
}

////////////////////////////////////////////////////////////////
bool DriverOLEDSSD1306_writeData ( OLED_t * Oled){

    HAL_StatusTypeDef res;
    res= HAL_I2C_Mem_Write(
                Oled->i2c ,
                Oled->I2Caddress ,
                OLED_DATO ,
                1 ,
                (uint8_t *)Oled->data ,
                Oled->TXsize ,
                Oled->TimeOut);

    return (res == HAL_OK )? true:false;
}

////////////////////////////////////////////////////////////////////////
// Funciones pblicas

OLED_t * NewOLED(void){

    static OLED_t NewOLEDvar ={

        //data
        .i2c= & hi2c2,
        .I2Caddress = 0x78,
        //La dirección usual de las pantallas Oled
          basadas en el chip SSD1306 es 0x3C, = 0x78>>1

        .TimeOut = OLEDTIMEOUT,

        //method
        .init  = DriverOLEDSSD1306_init,
        .start = DriverOLEDSSD1306_start,
        .stop  = DriverOLEDSSD1306_stop,
        .read  = DriverOLEDSSD1306_read,
        .writeCommand = DriverOLEDSSD1306_writeCommand,
        .writeData    = DriverOLEDSSD1306_writeData

    };
    return &NewOLEDvar;
}
```

```
////////////////////////////////////////////////////////////////////
// Función principal


#endif  // final
```