Índice del documento

Departamento de Ingeniería Telemática y Electrónica

Universidad Politécnica de Madrid

2021-22

Sistemas Basados en Microprocesador

Integración y desarrollo

de una aplicación de

medida de iluminación

Alumno:

Pablo Martínez Lafarga

[1 Objetivos de la PRÁCTICA 2](#_Toc88129291)

[1.1 Resumen de los objetivos de la práctica realizada 2](#_Toc88129292)

[1.2 Acrónimos utilizados 2](#_Toc88129293)

[1.3 Tiempo empleado en la realización de la práctica. 2](#_Toc88129294)

[1.4 Bibliografía utilizada 2](#_Toc88129295)

[1.5 Autoevaluación. 2](#_Toc88129296)

[2 RECURSOS UTILIZADOS DEL MICROCONTROLADOR 3](#_Toc88129297)

[2.1 Diagrama de bloques hardware del sistema. 3](#_Toc88129298)

[2.2 Cálculos realizados y justificación de la solución adoptada. 3](#_Toc88129299)

[3 SOFTWARE 4](#_Toc88129300)

[3.1 Descripción de cada uno de los módulos del sistema 4](#_Toc88129301)

[3.2 Descripción global del funcionamiento de la aplicación. Descripción del autómata con el comportamiento del software (si procede) 4](#_Toc88129302)

[3.3 Descripción de las rutinas más significativas que ha implementado. 4](#_Toc88129303)

[4 DEPURACION Y TEST 5](#_Toc88129304)

[4.1 Pruebas realizadas. 5](#_Toc88129305)

# Objetivos de la PRÁCTICA

T

## Resumen de los objetivos de la práctica realizada

En primer lugar, se ha abordado el desarrollo de la practica de forma modular y escalonada, es decir, hemos ido realizando las diferentes funcionalidades, de forma individual, para así poder hacer un mejor rastreo de los posibles errores o de las irregularidades funcionales que se encuentren. Además de resultar de facilidad a la hora del desarrollo, también era una de las especificaciones de diseño. Cabe destacar que cada módulo se ha desarrollado como un hilo independiente. Esto es un arma de doble filo debido: puede resultar de utilidad a la hora de hacer una aplicación en tiempo real, pero un problema más que gestionar, cuando requieres que módulos específicos, tengan prioridades diferentes.

Teniendo todo esto en cuenta, el primer modulo a desarrollar fue el encargado de la gestión del LCD, de esta forma, más adelante podríamos ir comprobando el funcionamiento del resto de módulos de forma mas sencilla, es decir, representando en la pantalla LCD diferentes mensajes, al puro estilo “*debug”*.

Por inercia el segundo modulo a realizar resultó ser el de la hora, ya que, debido a las especificaciones, esta se tenia que representar en todo momento en la parte superior del LCD. También era lógico que este fuera el segundo modulo, ya que nos serviría para comprobar en cualquier momento, si la aplicación se había quedado bloqueada o no, ya que, en caso de llegar a un bloqueo, los segundos dejarían de actualizarse. Por otro lado, era de utilidad tener este modulo funcionando desde el principio, ya que seriamos capaces de ver si los futuros módulos, consumían demasiados recursos, hasta el punto de que el reloj no se actualizase en tiempo real, es decir, el segundero contase más lento de lo habitual.

El siguiente módulo, debía ser uno que nos permitiera ir interactuando en tiempo de ejecución con la aplicación, por tanto, había dos posibles módulos: comunicación o joystick. La primera intención fue la de proceder con el desarrollo del primero de ellos, pero tras un par de horas de avanzar muy lentamente debido a lo complejidad intrínseca del módulo, decidí aplazarlo, continuar con el joystick y más adelante retomar el módulo de comunicación.

Continué codificando el modulo RGB, el cual era prácticamente un descanso después de todos los dolores de cabeza generados a causa del joystick y del inacabado módulo de comunicación. Esto es debido a que este módulo, es lo que se diría “*straightforward”*.

Ya era hora de empezar a realizar la parte principal de la aplicación, es decir, todo lo relativo a la toma de medidas de luminosidad, y comunicación con el sensor correspondiente a través del protocolo I2C. Desarrollare más en profundidad el contenido de este módulo, más adelante.

Por último, decidí realizar el modulo principal, el cual se encarga del control del resto de módulos y de su interconexión. Llegados a este punto, y como habíamos estado realizando pequeños programitas para poder revisar las funcionalidades del resto de módulos, no resulto de mucha complejidad unirlo todo. Eso sí, en este punto se empezaron a ver una cantidad ingente de problemas, bugs, o funcionalidades parcial o totalmente incorrectas, todo ello derivado del haber realizado el resto de los módulos con una pobre visión del funcionamiento final de la aplicación. Este modulo junto con el de comunicación, fueron los dos que más tiempo consumieron, debido a que eran “la cinta americana” del resto de módulos.

## Acrónimos utilizados

|  |  |
| --- | --- |
| CMSIS | Cortex Microcontroller Software Interface Standar |
| HAL | Hardware Abstraction Layer |
| GPIO | General Purpose Input/Output |
| UART | Universal Asynchronous Receiver Transmitter |
| USART | Universal Synchronous/Asynchronous Receiver Transmitter |
| I2C / I2C | Inter-Integrated Circuit |
| SPI | Serial Peripheral Interface |
| RGB | Red – Green - Blue |
| LCD | Liquid Crystal Display |
| CS | Chip Select |
| MOSI | Master Output Slave Input |
| MISO | Master Input Slave Output |
| PWM | Pulse Width Modulation |
|  |  |
|  |  |

## Tiempo empleado en la realización de la práctica.

Debe realizar una descripción sencilla del tiempo que ha dedicado a la realización de las actividades relacionadas con la práctica.

|  |  |
| --- | --- |
|  | **[Tiempo empleado para realizar la práctica]:** El tiempo total empleado ha sido de x horas. |

|  |  |
| --- | --- |
| Modulo | Tiempo |
| LCD | 3 horas |
| CLOCK | 30 minutos |
| RGB | 1 hora y 30 minutos |
| Joystick | 5 horas |
| TSL2591 | 4 horas |
| COM | 6 horas y 30 minutos |
| PRINCIPAL | 10 horas y 30 minutos |

## Bibliografía utilizada

1. <https://www.keil.com/pack/doc/CMSIS/Driver/html/index.html>
2. <https://www.st.com/resource/en/user_manual/um1725-description-of-stm32f4-hal-and-lowlayer-drivers-stmicroelectronics.pdf>
3. <https://moodle.upm.es/titulaciones/oficiales/pluginfile.php/10030203/mod_resource/content/8/UM1725-STM32F4-HAL-AND-LL-DRIVERS-DESCRIPTION.pdf>

## Autoevaluación.

RA971 - Manejar temporizadores hardware para gestionar la temporización y sincronización de una aplicación. -> OBJETIVO CUMPLIDO: por ejemplo, en el modulo de reloj se necesita hacer una sincronización adecuada, así mismo como se requiere de una buena temporización por medio de un “*timer*”, para poder implementar el paso del tiempo en el reloj.

RA970 - Establecer y gestionar las comunicaciones entre dos sistemas utilizando diferentes interfaces. -> OBJETIVO CUMPLIDO: Se utilizan diversas interfaces en diferentes módulos. Por mencionar algunas: SPI en el modulo LCD, I2C en el módulo TSL2591, USART en el módulo COM y GPIO en el módulo RGB.

RA907 - Desarrollo de aplicaciones en grupos de trabajo. -> OBJETIVO PARCIALMENTE CUMPLIDO: A pesar de que, a lo largo de todo el curso, las distintas actividades eran individuales y no se ha permitido el trabajo en grupo, no se puede negar la existencia de una estrecha colaboración con diferentes compañeros.

RA736 - Interpretar las especificaciones de funcionamiento de un sistema basado en microcontrolador de mediana complejidad. -> OBJETIVO CUMPLIDO: Como se puede apreciar con esta última práctica, se han interpretado correctamente las especificaciones de esta. Posteriormente, dichas características se han desglosado y troceado en partes, más sencillas conceptual o funcionalmente, para así poder alcanzar el objetivo final de desarrollar la aplicación.

RA737 - Escribir el código necesario para desarrollar una aplicación basada en microcontrolador de mediana complejidad. -> OBJETIVO CUMPLIDO: Se puede ver como la aplicación final funciona de acuerdo con las especificaciones presentadas.

RA730 - Conectar un periférico a un microcontrolador utilizando interfaces basadas en protocolos estándar -> OBJETIVO CUMPLIDO: Se utilizan diversos protocolos de comunicación. Por mencionar algunas: SPI en el módulo LCD, I2C en el módulo TSL2591, USART en el módulo COM y GPIO en el módulo RGB.

RA733 - Aprender a manejar cualquier periférico de mediana complejidad de un microcontrolador a partir de la documentación proporcionada por el fabricante -> OBJETIVO CUMPLIDO: Se ha conseguido entender el “*datasheet*” del sensor proporcionado (TSL2591), paso necesario para realizar una correcta configuración de los registros de configuración del sensor.

RA968 - Manejar instrumentación electrónica específica para el desarrollo de sistemas basados en

Microprocesador. -> OBJETIVO CUMPLIDO: para realizar el proceso de “*debug”* en algunos módulos como el TSL2591, el cual usa un protocolo de comunicación I2C, ha sido esencial el uso de un analizador lógico.

RA738 - Elaborar el informe que justifica y describe la toma de decisiones adoptadas en el desarrollo de un

proyecto y defenderlo oralmente con precisión y detalle. -> OBJETIVO PREVISIBLEMENTE CUMPLIDO: Quedará a evaluación del profesorado, pero desde mi humilde punto de vista, considero que he realizado un buen proyecto, y que he sido capaz de defenderlo con esta memoria.

RA735 - Analizar la arquitectura software y hardware de sistemas basados en microcontrolador de mediana

complejidad. -> OBJETIVO CUMPLIDO: Esto es un paso esencial al principio de cualquier desarrollo de una aplicación con una complejidad mínima. Considero más que demostrada este conocimiento.

RA734 - Manejar entornos de CAD para la codificación, la compilación y la depuración de aplicaciones basadas en microcontrolador. -> OBJETIVO CUMPLIDO: El uso de “*Keil microvision”* es algo esencial para este proyecto. Se podría haber abordado el mismo sin el uso de dicho software y con uno similar.

# RECURSOS UTILIZADOS DEL MICROCONTROLADOR

## Diagrama de bloques hardware del sistema.

Graphical user interface

Description automatically generated

## Cálculos realizados y justificación de la solución adoptada.

En este punto debe describir como ha configurado cada uno de los recursos del microcontrolador, los cálculos que haya realizado y los valores programados en los registros más significativos.

# SOFTWARE

## Descripción de cada uno de los módulos del sistema

• PRINCIPAL: en este módulo, se gestionan todas las colas y variables globales de todos los demás

módulos. En él, se reciben los mensajes y envían los mensajes para la realización de cada gestión de

los módulos. Dicha gestión se realiza por medio de una maquina de estados (mostrada en el punto 3.2).

• LCD: gestiona tanto la configuración inicial de los pines necesarios para llevar a cabo una comunicación SPI con el “*display”*, como la posterior comunicación con el mismo. Toda la configuración inicial del LCD esta “*hard coded”*, por tanto, no se puede configurar sin modificar el código fuente cosas como si se quiere el “*display* “en modo normal o en modo invertido. La comunicación con en el exterior se realiza por medio de una cola de mensajes, en la cual se debe especificar tanto el mensaje de cada una de las dos líneas, como el desplazamiento inicial de cada una de ellas.

• JOYSTICK: es el encargo de procesar las interrupciones creadas por los distintos gestos del sistema. Además, se encarga de gestionar los rebotes de cada gesto, y a su vez determinar la duración de estos. Una pulsación se considera larga si excede los 1500ms pulsada, es decir 1,5s. Este módulo está desarrollado específicamente para esta aplicación, es decir, que solo implementa los gestos de arriba y abajo, aunque si se requiriera, no llevaría más de 5 minutos implementar el resto de los gestos. Se comunica con el exterior por medio de una cola de mensajes, en la cual se especifica, tanto el gesto que se ha producido, como la duración de este.

• RGB: gestiona el encendido y apagado de los leds. Todo se gestiona internamente y la única comunicación es por medio de una cola de mensajes, en la cual solo se especifica cual de los leds se quiere encender. Este módulo está diseñado específicamente para esta aplicación, esto quiere decir que no se puede configurar externamente que led se desea hacer que parpadee, ni se puede seleccionar el color del RGB. Por tanto, todas las funcionalidades de ese estilo están “*hard coded”*. El efecto de respiración del led RGB esta creado mediante un PWM, el parpadeo continuado del LED2 y el “*flick”* del LED3 están ambos diseñados mediante diversos “*timmers”*.

• COM: es el encargado de sacar por el canal serie de comunicación, los comandos que se mandan a la aplicación para realizar diversas funciones de configuración, toma de medidas y representación de estas. Se gestiona mediante el protocolo de comunicación USART. El funcionamiento interno es algo complejo ya que constantemente se tiene que estar cambiando de datos entre hexadecimal y ASCII, todo ello debido a las especificaciones técnicas. Muchas de las tramas a enviar y recibir combinan ambas codificaciones en la misma trama. Además, este módulo debe comprobar que todas las tramas cumplen un determinado formato conforme a especificaciones, y todas las tramas que no lo cumplan deben ser ignoradas. La comunicación con el exterior se hace por medio de una cola de mensajes en la cual se indica entre otras cosas, el código de la operación, y la información de la trama (es decir parámetros de configuración u otros elementos que van embebidos en las tramas).

• CLOCK: Mediante un timer virtual, se realiza, cada segundo, un incremento de la variable global “*seconds”,* y se comprueba si se debiese aumentar también las variables “*minutes*” y “*hours*”. A pesar de ser variables globales, se implementa por comodidad y estética, una función adicional para “*settear”* los valores de las tres variables. Este módulo implementa un reloj de formato 24 horas.

• MAIN: es el encargado de inicializar los diversos “*threads*” y ya cada uno de ellos de forma autónoma inician las colas de mensajes y demás sistemas de comunicación, como pudiera ser la configuración inicial de pines.

## Descripción global del funcionamiento de la aplicación. Descripción del autómata con el comportamiento del software (si procede)

En este punto debe explicar el funcionamiento completo de su aplicación. Debe aportar detalles de todos los elementos que forman la aplicación y de cómo interaccionan entre sí. Puede utilizar autómatas para describir el funcionamiento del software.

## Descripción de las rutinas más significativas que ha implementado.

En este punto debe enumerar y describir la funcionalidad de las rutinas más importantes que ha implementado.

# DEPURACION Y TEST

## Pruebas realizadas.

Descripción del método de prueba utilizado para comprobar que las rutinas funcionan adecuadamente. Resultados de los tests. Indicación explicita de si el test es correcto o incorrecto. En este punto debe hace especial hincapié en definir:

1. Cuál es el objetivo de la prueba, indicando los módulos implicados
2. Cuál es el proyecto de Keil que permite realizar la prueba
3. Cuáles son las condiciones de entrada que permiten ejecutar la prueba
4. Cuáles son los resultados que se esperan y cuáles son los realmente obtenidos.