UNIVERSIDAD
COMPLUTENSE
MADRID

Trabajo Final de Máster

Control del cromatismo de luminarias LED basado en PID a través del protocolo de control DALI

Autor:	Jesús López Chaves
Tutores:	Eva Besada Portas
	Diego Pedro Morales Santos
Curso:	2018 / 19
Convocatoria:	Junio

06 de Junio de 2019

Máster Ingeniería de Sistemas y Control

Nombre del Proyecto:	Control del cromatismo de luminarias LED basado en PID a través del protocolo de control DALI
Clase del Proyecto:	Tipo B**:** Proyecto específico propuesto por el alumno
Nombre del Estudiante:	Jesús López Chaves.
Nombre de los Directores:	Eva Besada Portas (UCM). Diego Pedro Morales Santos (UGR).

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Jesús López Chaves.

Granada, 6 de Junio de 2019

Resumen

En este proyecto se desarrolla un controlador PID para el control del cromatismo de una luminaria LED de tres colores (rojo, verde y azul). Para ello, inicialmente se caracteriza la respuesta dinámica y estacionaria del sistema, y posteriormente se procede a estudiar el mismo desde un enfoque multivariable para el desacoplo de las variables a controlar. A continuación, se cierra el lazo de control con un regulador PID, cuyos parámetros se sintonizan mediante un algoritmo de optimización heurístico. Por otra parte, se implementa un controlador basado en el estándar de comunicación DALI, empleando los microcontroladores de la familia XMC100 de Infineon Technologies, y su correspondiente interfaz gráfica para el ordenador. Para todo este trabajo, además, ha sido necesaria la elaboración de distintas librerías en C y MATLAB, así como en Python para la programación de la interfaz gráfica.

Palabras Clave

Controlador, PID, DALI, multivariable, XMC, interfaz gráfica, cromatismo, temperatura, brillo, optimización heurística, optimización con nubes de partículas, Qt, Python, C, MATLAB.

Listado de Siglas

- **LED** Diodo Emisor de Luz.
- **DALI** *Digital Addressable Lighting Interface*.
- **PWM** Modulación por Anchura de Pulso.
- **APP** Aplicación Informática.
- **IDE** Entorno de Desarrollo Integrado.
- **SMD** Dispositivo de Montaje Superficial.
- **RGB** Rojo, Verde, Azul.
- **ADC** Convertidor Analógico – Digital.
- **SSGM** Matriz de Ganancias en Estado Estacionario.
- **RGA** Matriz de Ganancias Relativas.
- **CG** Dispositivo de Control. Esclavo de la comunicación DALI.
- **PC** Ordenador Personal
- **UART** Transmisor-Receptor Asíncrono Universal
- **TX** Transmisor
- **RX** Receptor
- **CRC** Verificación de Redundancia Cíclica
- **RAM** Memoria de acceso aleatorio
- **ROM** Memoria de solo lectura
- **IoT** Internet de las cosas
- **CCU** Unidad de Captura y Comparación

- **ZnTe** Teleruro de Zinc

Contenido

1. Introducción 18
   1.1. Contextualización 18
   1.2. Motivación 20
   1.3. Objetivos 21
   1.4. Estado del Arte 22
   1.5. Memoria 23
2. Fundamentos Físicos 26
   2.1. Emisión Fotónica en Dispositivos LEDs 26
   2.2 Cromatismo 27
   2.3. Control de Corriente. 27
   2.4. Estándar DALI 28
      2.4.1. Red DALI 29
      2.4.1. Especificaciones del estándar DALI 30
3. Caracterización y Modelado del Sistema 33
   3.1. Presentación del Sistema 33
      3.1.1. Microcontrolador 34
      3.1.2. Fuente de Iluminación LED 34
      3.1.3. Drivers 34
      3.1.4. Sensor de Color RGB 35
   3.2. Respuesta Estacionaria del Sistema 35
   3.3. Respuesta Dinámica del Sistema 36
   3.4. Modelo en Simulink 38
4. Estudio del Control PID 41
   4.1. Desacoplo de Variables 41
      4.1.1. Estudio de las Dependencias 42
      4.1.2. Desacoplo Inverso 42
      4.1.3. Implementación en Simulink 44
   4.2. Problema de Optimización Heurística 46
      4.2.1. Variables de Decisión 46
      4.2.2. Restricciones 47

4.2.3. Funciones Objetivo 47
4.3. Algoritmo de Optimización 49
4.4. Resultados de la Optimización 52
4.4.1. Sistema continuo 52
4.4.2. Sistema discreto 53
..... 54
4.5. Operación y Resultados del Controlador PID 54
5. Controlador DALI 58
5.1. Transceptor DALI 58
5.1.1. Hardware del Transceptor DALI 58
5.1.2. Software del Transceptor DALI 60
5.2. Direccionamiento de Esclavos 60
5.2.1. Reconocimiento por Dirección Aleatorio 60
5.2.2. Búsqueda Binaria 61
5.3. Búsqueda de Esclavos 62
5.4. Comandos DALI 63
5.5. Desarrollo de los Esclavos DALI 64
5.6. Resultados de la Implementación del Maestro DALI 65
6. Diseño y Desarrollo de la Interfaz DALI 67
6.1. Protocolo de Comunicación PC – Maestro 67
6.1.1. Interfaz física 67
6.1.2. Desarrollo Software del protocolo 68
6.2. Interfaz Gráfica 72
6.2.1. Interfaz Serie 72
6.2.2. Interfaz Gráfica 73
7. Implementación Física 78
7.1. Maestro DALI 78
7.2. Interfaz DALI 79
7.4. Sensor RGB 81
7.5. LEDs RGB 82
7.6. Alimentación 82
7.8. Resultados del Sistema Real 83
8. Conclusiones y Futuros Trabajos 90

8.1. Conclusiones 90
8.2. Trabajos Futuros 91
Bibliografía 93

Índice de Figuras

Ilustración 1. Cine 4D con distinto cromatismo en la iluminación [1]. 18
Ilustración 2. Estadio Allianz Arena el 6 de septiembre del pasado año 2018. Francia contra Alemania [2]. 19
Ilustración 3. Cabina de avión con distintos ambientes gracias al cambio de color de la iluminación. [3]..... 19
Ilustración 4. Diagrama del sistema general. 25
Ilustración 5. Recombinación directa de portadores para la emisión de fotones y fonones [9]. 26
Ilustración 6. Colores primarios, secundarios y blanco [10]. 28
Ilustración 7. Curvas I-V para LEDs RGB [10]. 29
Ilustración 8. Ondas PWM. [12]. 29
Ilustración 9. Márgenes de alimentación del bus DALI [11]. 30
Ilustración 10. Trama de mensaje transmitida por el maestro. 31
Ilustración 11. Tramas maestro - esclavo y temporización de las mismas. Unidades en milisegundos. 31
Ilustración 12. Codificación Manchester [12]. 31
Ilustración 13. Sistema en lazo abierto. 33
Ilustración 14. Layout del driver rojo [15]. 34
Ilustración 15. Respuesta estacionaria de los tres canales. 36
Ilustración 16. Respuesta dinámica del sistema. 37
Ilustración 17. Sistema real frente al modelo del sistema. 38
Ilustración 18. Diagrama de bloques del modelo construido en Simulink. 39
Ilustración 19. Respuesta del sistema ante un escalón unitario en cada una de las tres entradas. 39
Ilustración 20. Red de desacoplo inverso 2x2 en configuración A [18]. 42
Ilustración 21. Diagrama en Simulink de la red de desacoplo. 45
Ilustración 22. Respuesta del sistema en lazo abierto con red de desacoplo. 45
Ilustración 23. Valor de las señales de control con la red desacoplo. 46
Ilustración 24- Respuesta dinámica con pendientes negativas indeseados. 49
Ilustración 25. Diagrama de flujo del algoritmo de nube de partículas [20]. 51
Ilustración 26. Valor medio y desviación estándar de las soluciones en 30 ejecuciones del algoritmo. 52
Ilustración 27. Respuesta dinámica de un canal modelo controlado. 53
Ilustración 28. Respuesta dinámica recogida para un canal modelo discreto. 54
Ilustración 29. Sistema global en lazo cerrado. 55
Ilustración 30. Respuesta dinámica del sistema en conjunto. 56
Ilustración 31. Respuesta dinámica global del sistema en tiempo discreto. 56
Ilustración 32. Modo de operación del sistema controlado. 57
Ilustración 33. Layout de la interfaz física del transceptor de DALI. 59
Ilustración 34. Montaje en baquelita de las dos resistencias para la limitación de corriente en la alimentación. 60

Ilustración 35. Proceso de direccionamiento en protocolo DALI. 61
Ilustración 36. Reconocimiento de esclavos en el bus DALI. 62
Ilustración 37. Diagrama de flujo de la búsqueda binaria y direccionamiento de esclavos. 63
Ilustración 38. Proceso de búsqueda de dispositivos [25]. 64
Ilustración 39. Transmisión a nivel de bits del mensaje 0xAA55. 66
Ilustración 40. Tramas a nivel de bits entre el maestro y esclavo. 66
Ilustración 41. Ejemplo de trama transmitida por UART [8]. 68
Ilustración 42. Diagrama de flujo del protocolo entre el ordenador y el maestro DALI. 69
Ilustración 43. Flujo de datos en el protocolo entre el ordenador y el maestro DALI. ... 70
Ilustración 44. Capturas de la interfaz serie para el ordenador. 73
Ilustración 45. Interfaz gráfica para el controlador DALI. 76
Ilustración 46. Archivo resultante de la interfaz gráfica. 77
Ilustración 47. Placa de desarrollo XMC 1400 BootKit [22]. 79
Ilustración 48. Placa DALI PHY [23]. 80
Ilustración 49. Kit de desarrollo " RGB LED Shield with XMC1202" [30]. 80
Ilustración 50. Sensor RGB TCS34725 [32] 81
Ilustración 51. LEDs seleccionados [14]. 82
Ilustración 52. Fuente de alimentación seleccionada [33]. 83
Ilustración 53. Inserción de salidas en la fuente de alimentación para el bus DALI y los LEDs. 84
Ilustración 54. Sistema en conjunto con todos los componentes. 84
Ilustración 55. Comparación entre la respuesta dinámica del sistema real y la del modelo. 85
Ilustración 56. Respuesta dinámica del modelo discreto en conjunto con T = 292 ms. . 86
Ilustración 57. Respuesta dinámica del sistema con el control PID ajustado para T = 292 ms 86
Ilustración 58. Respuesta dinámica del modelo continuo en conjunto. 87
Ilustración 59. Comparación entre la respuesta dinámica del sistema real y la del modelo con el PID optimizado para eliminar las interferencias entre canales. 87
Ilustración 60. Cambios de consigna del sistema real. 88

Índice de Tablas

Tabla 1. Ganancias en estado estacionario del sistema. 36
Tabla 2. Respuestas dinámicas del sistema. 37
Tabla 3. Valores de la red de desacoplo 3x3 [19]. 44
Tabla 4. Resumen de las características dinámicas del control PID óptimo encontrado. 53
Tabla 5. Resumen de las características dinámicas del control PID para tiempo discreto. 54
Tabla 6. Configuración de la interfaz UART para el protocolo de comunicación entre el PC y el maestro DALI. 68
Tabla 7. Ejemplos de tramas entre el maestro DALI y la interfaz. 71
Tabla 8. Ejemplos de trama entre la interfaz y el maestro DALI. 72
Tabla 9. Especificaciones del microcontrolador XMC 1400 [22]. 78
Tabla 10. Especificaciones del microcontrolador XMC 1300 [31] 81
Tabla 11. Resumen de las características más importantes del sensor RGB. 81

Capítulo 1

Introducción

Este capítulo tiene por objetivo realizar una presentación de este proyecto, pasando por su contextualización en el momento actual, explicando la motivación de dicha investigación, repasando brevemente el estado del arte que envuelve el desarrollo del mismo, y finalmente revisando la estructura de la presente memoria y del sistema.

1.1. Contextualización

No cabe duda de que el desarrollo tecnológico que la sociedad actual está experimentando no tiene precedentes. Miles de aplicaciones nuevas, a pequeña y gran escala, están abriendo nuevos paradigmas revolucionarios, cuyos frutos ya se pueden ir disfrutando en la actualidad. Desde el desarrollo de dispositivos de control de alta potencia, que posibilitan la gestión de energía de forma eficiente y rápida, pasando por el desarrollo de redes neuronales y la inteligencia artificial, hasta los pequeños microprocesadores con tecnología a escala nanométrica. Todos ellos son ejemplos actuales que muestran el profundo avance que están experimentando las bases sobre las que se fundamentan los sistemas de la sociedad actual.

En concreto, en el campo de la iluminación también se están desarrollando nuevos conceptos que posibilitan la creación de ambientes y espacios. Un ejemplo claro y sencillo puede verse en los cines 4D (Ilustración 1), donde la iluminación de la sala va acorde a la escena que se está proyectando en ese momento, haciendo que el espectador se sienta inmerso en la misma.

Ilustración 1. Cine 4D con distinto cromatismo en la iluminación [1].

Otro ejemplo más llamativo puede encontrarse en la ciudad de Múnich, en concreto, en el estadio Allianz Arena, donde se puede cambiar el cromatismo de la iluminación exterior del estadio de acuerdo con el partido que se está jugando (ver Ilustración 2).

Pero esto no solo se limita a aplicaciones de alta potencia. En iluminación de habitáculos, el control del cromatismo de las luminarias es especialmente importante. Por ejemplo, algunos aviones pueden cambiar el color del habitáculo de los pasajeros según se está despegando, en pleno vuelo, en el aterrizaje, o en función de la hora del día.

Ilustración 2. Estadio Allianz Arena el 6 de septiembre del pasado año 2018. Francia contra Alemania [2].

Ilustración 3. Cabina de avión con distintos ambientes gracias al cambio de color de la iluminación. [3]

Estos ejemplos ilustran una concepción de la iluminación, conocida como *Smart Lighting*, que va más allá del mero alumbramiento de elementos o habitaciones. Esta nace de la mano del desarrollo del Internet de las Cosas, o como popularmente se conoce IoT

(*Internet of Things*), cuyo objetivo es uno: facilitar las actividades cotidianas en distintos entornos y llevarlas a su máximo exponente mediante su intercomunicación a través de internet. Es decir, conseguir metas tan llamativas como que un cepillo de dientes alerte del nivel de caries excesivo en la boca a su dueño, o que el macetero de una planta detecte la temperatura ambiental y la humedad de la tierra, y abra la llave de riego para dicha planta.

En concreto, en el campo de la iluminación inteligente se persigue la capacidad de crear distintos ambientes de forma intuitiva, fácil y, en definitiva, accesible a todos los usuarios. Así, la idea sería cambiar el color o la temperatura de forma automática o remota, a través de una interfaz sencilla que garantice la fiabilidad y estabilidad del sistema.

1.2. Motivación

Dado que toda esta tecnología está en pleno desarrollo, aún quedan muchas líneas de investigación que merecen la pena explorar, y que pueden contribuir a la construcción de la tecnología que empieza hoy, y que se establecerá en un mañana.

Para dar cabida a todos estos sistemas, se hace evidente la necesidad tanto de un control del color, como de un estándar de comunicación que posibilite el control de todos los dispositivos drivers para las luminarias. En ese sentido, otra motivación es conseguir adaptar el control y la monitorización de las lámparas de una forma eficiente, flexible y fácil para el usuario

Para ello, la implementación del controlador persigue la consecución de una consigna de forma estable, además de la monitorización del color de la lámpara. Es posible que, aunque un sistema se haya configurado correctamente en su instante inicial, el color de este se desvíe de la consigna debido al calentamiento de las luminarias, al encendido de una fuente de luz posteriormente a la configuración, o cualquier otro motivo que altere la salida del sistema. En este caso, el controlador actuaría para eliminar el posible error producido. A fin de eliminar esta posible desviación en la salida o simplemente para la consecución de una consigna distinta, es necesario un control del color.

Se hace indispensable entonces el campo del control de sistemas, como principal herramienta para la consecución de un determinado valor de consigna en la salida de la fuente de iluminación, y todo el trasfondo sistemático y de modelado que la implementación del control conlleva.

Sin duda, una de las mejores vías para dar cabida a un amplio abanico de aplicaciones concretas, basadas todas en el mismo concepto, es a través del estándar DALI (*Digital Addressable Lighting Interface*). Este posibilita una cobertura flexible y eficiente, tanto en pequeños sistemas de iluminación como pueden ser mostradores de

perfumes, hasta en grandes centrales luminosas que dan vida a estadios o recintos para conciertos.

1.3. Objetivos

El principal objetivo de este proyecto, como ya se ha comentado, ha sido la implementación de un control del cromatismo de lámparas LED. Esto se articula en una serie de objetivos más concretos que a continuación se presentan, y sobre los que se desarrollan los capítulos posteriores.

- **Caracterización y modelado del sistema.** El primer objetivo de este proyecto nace de la necesidad de conocer en detalle el sistema en cuestión. En concreto, consiste en el desarrollo de un modelo matemático del sistema lo más fiable posible, que permita la predicción de las salidas del mismo ante cualquier entrada.

- **Sintonización del controlador:** Una vez conocido el sistema en detalle, el siguiente objetivo es controlarlo. Por ello, en este trabajo se ha propuesto un control simple, basado en un PID, con una red de desacoplo entre variables previa. El objetivo entonces es su sintonización. Para ello, se aplican distintas técnicas de optimización heurística para la obtención de los mejores parámetros.

- **Implementación de un controlador basado en la interfaz DALI:** Como ya se ha discutido, la interfaz DALI resulta idónea para dar cabida a un gran abanico de aplicaciones distintas. Se propone entonces la implementación de un controlador DALI que sea capaz de gestionar la interfaz física propia del estándar, así como su protocolo, el direccionamiento de los esclavos, etc.

- **Desarrollo de una interfaz gráfica para el controlador:** Este objetivo tiene su motivación en la búsqueda de simplificar el uso del controlador DALI y favorecer su expansión a todos los usuarios. Para ello se propone el desarrollo de una interfaz gráfica para la transmisión de comandos al maestro DALI.

- **Desarrollo de un protocolo de comunicación para el controlador DALI y el ordenador:** Este objetivo nace directamente del anterior y es crucial para la consecución del mismo. Para comunicar el controlador DALI con el ordenador se necesita un protocolo de transmisión de datos. Esto conduce a una nueva línea de trabajo para este proyecto.

- **Implementación del control en el sistema real:** Este objetivo culmina el proyecto y tiene como fundamento la implementación de una actividad global y en conjunto de todos los trabajos realizados para la consecución de los objetivos previos. Así, se desea tener un sistema en el que, a través de la interfaz gráfica se selecciona un valor de consigna en los colores rojo, verde y azul. Esta consigna se

envía a través del protocolo de comunicación implementado al maestro DALI. Este, tras tomar una lectura del valor actual del color, inicia el control PID, que de forma iterativa acerca el color de la lámpara al de consigna recibido, transmitiendo para ello comandos a los distintos esclavos anclados al bus DALI mediante este estándar.

- **Redacción de la memoria del proyecto:** El último objetivo marcado para la finalización del proyecto consiste en la redacción de esta memoria, que pretende recoger todo el trabajo realizado, y exponer los resultados de forma clara, concisa y concreta, justificando todas y cada una de las decisiones tomadas, asentando las bases para investigaciones futuras.

1.4. Estado del Arte

A continuación, se presentan algunos artículos de interés, donde se desarrollan algunos de los puntos de este proyecto, contextualizandolo así en el panorama científico actual.

- "*Modeling and Feedback Control of Color-tunable LED Lighting Systems*" - Sina Afshari, Sandipan Mishra, Agung Julius, Fernando Lizarralde, John T. Wen [4]. En este artículo se desarrolla un controlador por realimentación. Para ello se busca minimizar la función de coste relativa a el consumo de energía, sujeto a la calidad de la luz. El desarrollo se implementa en un espacio de color distinto al RGB, en concreto, el espacio XYZ, que tiene en cuenta la sensibilidad del ojo humano a la recepción de los colores primarios. Además, se estudia la temperatura de color, en relación con la calidad de la luz, para la minimización de la función comentada. Si bien en este artículo se desarrolla un sofisticado controlador, no se especifica de qué forma se lleva a cabo dicho control, ni se adapta ninguna interfaz para flexibilizar el controlador. Sin embargo, si que plantea una alternativa distinta a un controlador PID para el control del cromatismo de las luminarias.

- "*Neural Network Based Dimming Level Control of LED Network*" - A. P. Jaya Muruga Raja, V. Meenakshi [5]. En este artículo se presenta un control de la iluminación de redes LEDs basado en redes neuronales. Si bien el propósito de este artículo difiere del principal objetivo de este trabajo, si muestra una alternativa distinta al control PID implementado en el proyecto, basándose en redes neuronales, sintonizadas a través del algoritmo de "*backpropataion*", que cierran el lazo de control.

- "*Development and Research of Lighting System Based on DALI*" Huadong Li, Mingguang Wu, Yufang Zhong [6]. "*A Design of Embedded DALI Controller*" - Wei Wu, Mingguang Wu, and Yanpeng Liu: En estos dos artículos se desarrollan

un sistema de iluminación basado en DALI basado en distintos microcontroladores (ATMEGA8L y ARM7 MCU). En ellos se hace un repaso similar de los puntos relativos a la implementación del maestro DALI y la interfaz gráfica de este proyecto. Sin embargo, estos se limitan a su implementación, sin darle la aplicación del control de cromatismo que se desarrolla en este artículo.

- "*Low-cost implementations of LED intelligent lighting based on DALI*" - Ma Jian-she, Liu Hong-lei [7]: En este artículo se desarrolla un sistema de bajo coste basado en DALI para el control de balastros LEDs. A lo largo del artículo se desarrolla el hardware de la interfaz DALI para la transmisión de comandos a través de la codificación Manchester. Esta implementación sirve tanto para el maestro como para el esclavo. Este artículo desarrolla más detalladamente otra alternativa a la interfaz DALI que en este proyecto se ha empleado como un módulo ya desarrollado.

Tras esta breve revisión del estado del arte podemos concluir que los trabajos descritos dan soluciones parciales al problema de la iluminación inteligente, mientras que en este Trabajo Fin de Master se cubren múltiples aspectos de su problemática.

1.5. Memoria

Este proyecto se realiza bajo el módulo "Trabajo Final de Máster" del máster "Ingeniería de Sistemas y Control" y responde a 12 de los 60 créditos que posibilitan la realización del mismo. En esta memoria se recogen de forma resumida los trabajos realizados para este proyecto que han arrojado resultados satisfactorios o han servido de alguna forma a la consecución del mismo. La memoria puede dividirse en dos hilos temáticos bien diferenciados que se interconectan en los últimos capítulos. Por un lado, la parte del modelado y control del sistema, y por otro, la implementación del controlador DALI para la estandarización del control de cromatismo.

Para facilitar la comprensión de la relación entre los capítulos que constituyen esta memoria, se presenta la Ilustración 4, que muestra un diagrama general del sistema, donde, en recuadros aparecen los elementos físicos que lo constituyen, y en elipses los programas software desarrollados. Además, es importante resaltar que al final de cada capítulo se exponen los resultados obtenidos en el mismo, y se muestra el proceso de desarrollo, contrastándolo con la Ilustración 4.

En el segundo capítulo, se hace un repaso de los fundamentos físicos más importantes que intervienen en el desarrollo de este proyecto, así como una presentación de los sistemas que se van a desarrollar, y la justificación de dicho desarrollo.

A continuación, siguiendo la primera de las dos temáticas, se describe en el tercer capítulo el proceso de estudio del sistema, la obtención del modelo matemático del mismo y la elaboración de un simulador para su posterior uso. En el cuarto capítulo se emplea este modelo para la optimización de los parámetros del controlador PID del sistema, tanto

en tiempo continuo como en discreto, empleando técnicas de optimización heurística. Este modelado responde al recuadro azul celeste, junto con el sensor de color del diagrama general del sistema presentado en la Ilustración 4. Por su parte, el control PID corresponde a la elipse naranja

Por otro lado, en el capítulo cinco se describe el controlador DALI implementado. El objetivo de este capítulo no es entrar en detalle de la programación del mismo sino, más bien, centrarse en los procesos más conflictivos para la consecución de la versión estable del maestro DALI. A continuación, en el sexto capítulo se presenta la interfaz gráfica desarrollada, mostrando también el protocolo implementado para la comunicación de la interfaz con el maestro DALI. Análogamente al capítulo anterior, aquí se muestran las especificaciones más importantes del protocolo y la interfaz. En el diagrama general del sistema presentado en la Ilustración 4, los contenidos redactados en estos dos capítulos responden al recuadro amarillo y el dibujo del ordenador.

A continuación, en el capítulo siete se describe la implementación del sistema físico real, y de cada uno de los dispositivos que lo componen. Puede verse en este capítulo como claramente los anteriores hilos temáticos mencionados se complementan mutuamente, y dan lugar a un sistema global que incrementa el valor de la suma de sus partes. En la práctica, este capítulo muestra la implementación física del diagrama presentado en la Ilustración 4. Por último, se muestran los resultados obtenidos con el sistema real, su comparación con las simulaciones, y las correcciones propuestas.

Por último, en el octavo y último capítulo se muestran los objetivos estudiados en este proyecto, y se proponen algunas de las posibles líneas de investigación futuras que complementan y desarrollan más aún todos dichos objetivos.

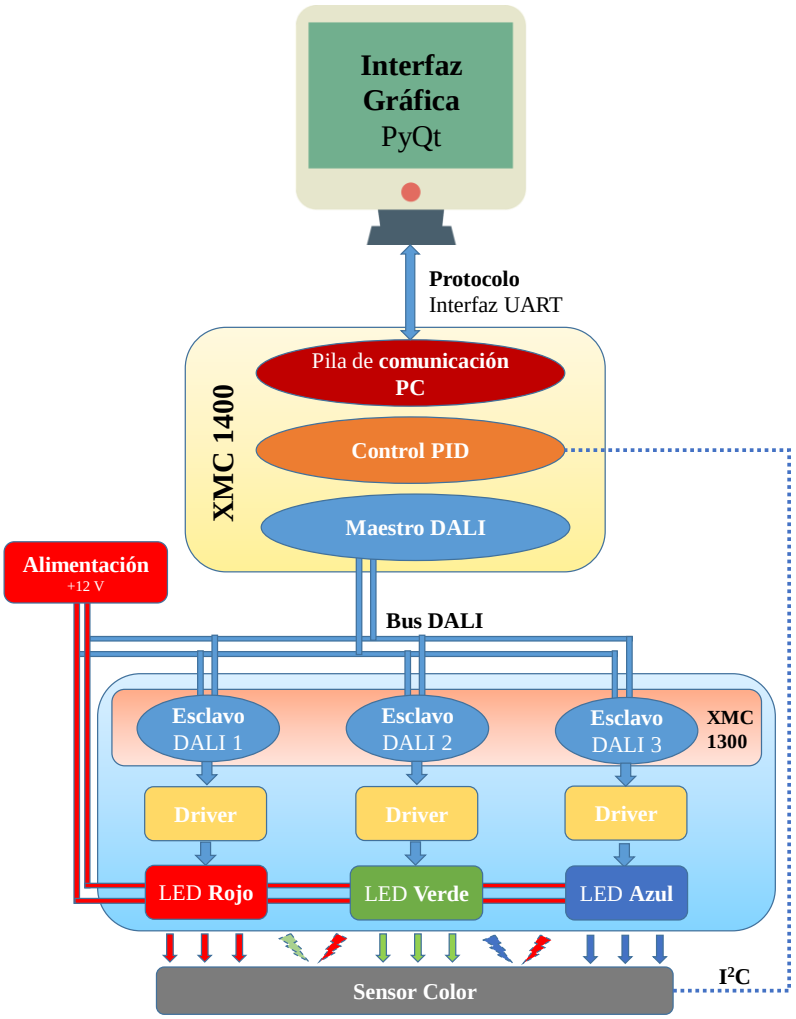


Ilustración 4. Diagrama del sistema general.

Capítulo 2

Fundamentos Físicos

A continuación, se muestra una breve introducción sobre los aspectos físicos que fundamentan algunos de los elementos del sistema, así como otros aspectos de interés.

2.1. Emisión Fotónica en Dispositivos LEDs

El funcionamiento de los dispositivos LEDs ("*Light-Emitting Diode*") [8] consiste básicamente en la recombinación directa de portadores (electrones con carga negativa, y huecos, con carga positiva) que libera una energía en forma de fotones (ver Ilustración 5 izquierda) y que, de acuerdo con la ley de Planck, se emite en forma de radiación electromagnética. Esta emisión, para ciertos materiales (por ejemplo, de teleruro de Zinc, ZnTe) coincide con las frecuencias visibles del espectro electromagnético.

Sin embargo, este proceso también libera energía en forma de fonones (modo cuantizado de vibraciones), que, en la práctica, tendrá consecuencias en el aumento de la temperatura del material cristalino (Ilustración 5 derecha).

Este aumento de la temperatura será uno de los responsables de posibles desviaciones en el cromatismo de la luz emitida por una lámpara constituida por varios LEDs de distinto color, ya que provocará un aumento de la corriente a través de cada semiconductor (mayor número de portadores en recombinación). Otros posibles factores de alteración del cromatismo pueden ser la oxidación de los materiales semiconductores o la aparición de nuevas fuentes de luz cercanas.

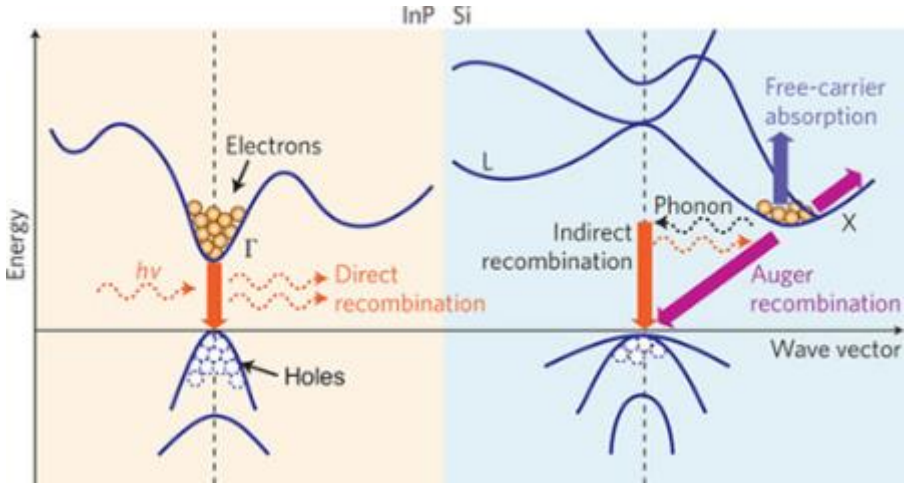


Ilustración 5. Recombinación directa de portadores para la emisión de fotones y fonones [9].

2.2 Cromatismo

El cromatismo de la luz hace referencia a la frecuencia de la onda electromagnética que la constituye (en la práctica, al color con la que dicha luz se observa). De aquí se derivan otros conceptos como la temperatura de color, de acuerdo con la radiación del cuerpo negro.

Como se ha comentado previamente, según el material del que está constituido el LED, la emisión del fotón en la recombinación de portadores tendrá una frecuencia determinada. Es decir, es posible obtener distintos colores, empleando diferentes materiales para la construcción de LED (siempre buscando que dicha recombinación sea lo más directa posible, evitando la emisión de fonones a la red). Sin embargo, no es posible obtener todos los colores del espectro visible, debido a los materiales que actualmente se emplean para la construcción de los LEDs.

Como solución a este problema se obtienen los diferentes colores del espectro visible a partir de la combinación en mayor o menor medida de los tres colores llamados primarios, cuya principal característica reside en tener un cromatismo que no puede ser obtenido a partir de otros colores, es decir, su valor es primario y, además, origen de otros muchos.

Estos colores primarios son el rojo, verde y azul. Así, por ejemplo, es posible obtener el color amarillo de la mezcla de un cromatismo verde y otro rojo. En definitiva, todas las longitudes de onda (frecuencias) de los colores del espectro visible pueden ser obtenidos a partir de la combinación de los colores primarios rojo, verde y azul. Por otro lado, merece especial mención el color blanco, que se obtendrá de la mezcla de los tres colores primarios, es decir, este poseerá todas las frecuencias del espectro visible. En la Ilustración 6 se muestra gráficamente este fenómeno.

Por último, es importante recalcar el hecho de que la intensidad de color dista del cromatismo del mismo. En la práctica, esto quiere decir que un mismo color puede ser obtenido con el mismo porcentaje relativo de los colores que lo constituyen. Por ejemplo, graduando la intensidad de los colores primarios de 0 a 100 (0 mínimo valor y 100 máximo), el color amarillo podrá ser obtenido con la combinación RGB ("*Red Green Blue*") de (100, 100, 0), o con la combinación (50, 50, 0), siendo esta última la mitad de intensa, pero con el mismo cromatismo (color amarillo).

2.3. Control de Corriente.

La Ilustración 7 muestra la curva característica I-V de LEDs RGB. De esta es posible apreciar como la corriente directa que circulará a través de un LED es una función exponencial de la tensión directa que soportan sus terminales. En concreto, dicha respuesta se modela matemáticamente con la ecuación 1

$$I = I_S\left(e^{\frac{V}{V_T}} - 1\right) \quad (1)$$

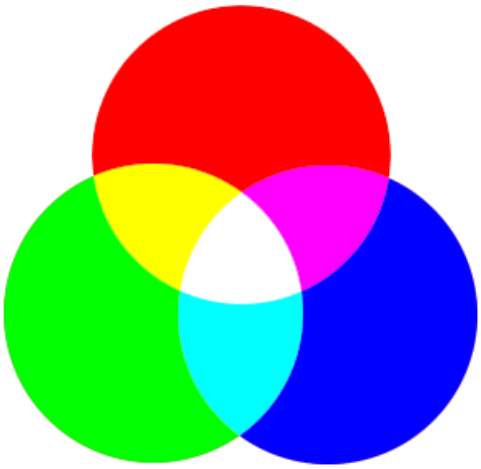


Ilustración 6. Colores primarios, secundarios y blanco [10].

Donde I_S responde a la corriente de saturación (corriente inversa cuando no hay tensión aplicada) y V_T es el voltaje térmico, dependiente del material.

En la práctica, cuanta más corriente circule a través del LED (o equivalentemente, cuanta más tensión soporten sus polos), más portadores se recombinarán en las zonas de decaimiento y mayor luz se emitirá. Una primera opción para controlar el cromatismo de una lámpara RGB entonces, sería modular el valor analógico de la corriente (tensión) que circula por cada uno de los tres canales (rojo, verde y azul), a fin de modificar el color global de dicha lámpara. Sin embargo, dada la respuesta exponencial, resulta considerablemente difícil dicho proceso, ya que es altamente no lineal, y pequeñas variaciones en la tensión de control implican grandes variaciones en la corriente, y, por tanto, en el color de la lámpara.

Como opción de control de color alternativa, se propone una modulación de la tensión de control en PWM ("Pulse Width Modulation"), donde, la tensión de control tiene un valor de amplitud constante, y se aplican pulsos a una frecuencia constante, pero con un ciclo de trabajo (o tiempo en alto de la señal, también llamado "*dutty cycle*") variable. Un ejemplo con distintos valores de ciclo de trabajo se muestra en laIlustración 8.

Filtrando posteriormente la onda (con filtros paso baja) es posible obtener el valor medio de dicha señal, cuyo valor dependerá por tanto del ciclo de trabajo de la misma. En resumen, es posible obtener una relación lineal entre el valor del ciclo de trabajo de la modulación PWM y la corriente que circula por el LED, es decir, el color de la lámpara.

2.4. Estándar DALI

DALI (*Digital Addressable Lighting Interface*) es un estándar de comunicación destinado a iluminación que está especificado por los estándares estándar IEC 62386 y IEC 60929.

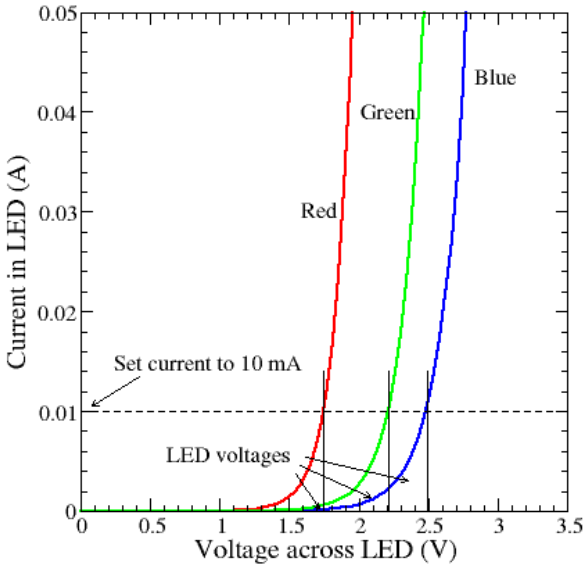


Ilustración 7. Curvas I-V para LEDs RGB [10].

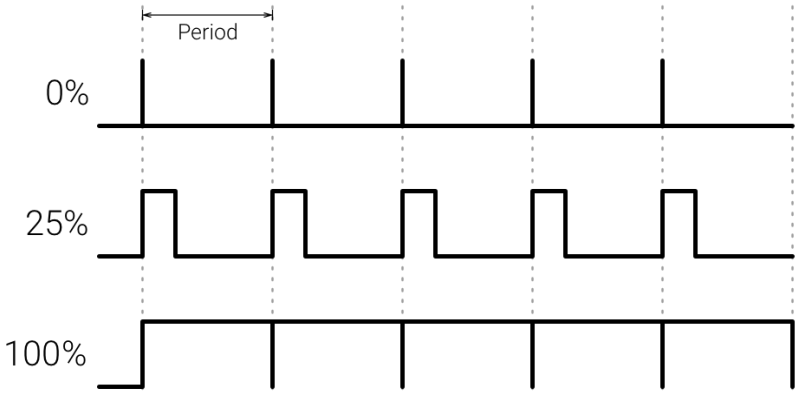


Ilustración 8. Ondas PWM. [12].

2.4.1. Red DALI

Una red DALI consta de un dispositivo maestro, un bus DALI, y hasta 64 dispositivos esclavos. A continuación, se describe más en detalle la función de cada uno de estos elementos:

- **Maestro DALI:** el dispositivo maestro será el encargado de transmitir diversas órdenes de control, realizar el escaneo de dispositivos esclavos, el direccionamiento de los mismos, y su agrupación en hasta 16 conjuntos distintos. Por otro lado, estará encargado también de recibir las ordenes deseadas por el usuario a través de distintas interfaces. Por ejemplo, a través de Bluetooth o Wifi, por puertos serie o con botones propios.

- **Bus DALI:** El bus DALI será el encargado de transmitir los mensajes enviados desde el maestro a los esclavos y viceversa. En él pueden estar anclados hasta 64 dispositivos esclavos.

- **Esclavos DALI:** Son los dispositivos que controlarán el circuito driver de las lámparas en cuestión. Estos recibirán las ordenes enviadas a través del bus DALI por el maestro y las ejecutarán. Puede haber hasta 64 esclavos en un mismo bus, todos ellos direccionados con una dirección única propia. También pueden agruparse los esclavos hasta en 16 grupos distintos.

2.4.1. Especificaciones del estándar DALI

A grandes rasgos, el estándar DALI realiza la comunicación entre los dispositivos a través de la propia alimentación del bus. Es por ello que este está constituido únicamente por dos hilos entre los que debe haber una tensión de como máximo 22.5 voltios. Generalmente se suelen emplear 16 voltios para alimentar el bus DALI, aunque, tal y como se muestra en la Ilustración 9, esta tensión puede reducirse hasta los 9.5 voltios para transmitir los bits correctamente. Cuando el bus DALI está en estado de reposo (no hay transmisión), la tensión entre los hilos es igual a la tensión de alimentación. Como se muestra en la Ilustración 9, un uno lógico será una transición entre el nivel bajo del bus (definido entre los márgenes de -6.5 V a 6.5 V) al nivel alto del bus (entre los márgenes de 9.5 V a 22.5 V). Un cero lógico será una transición en sentido inverso.

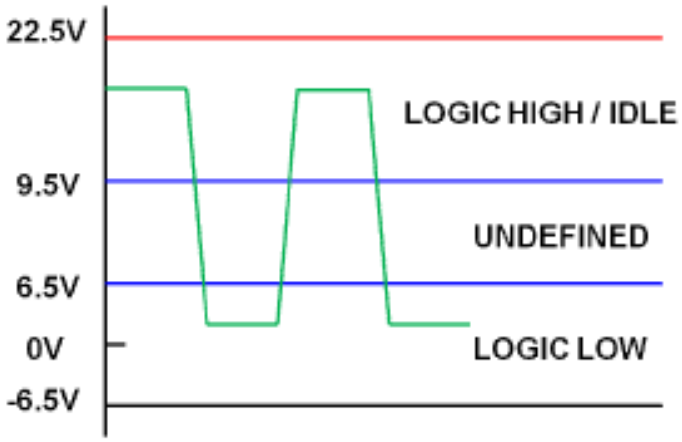


Ilustración 9. Márgenes de alimentación del bus DALI [11].

Por otro lado, el estándar DALI opera a una velocidad de transmisión igual a 1200 bits por segundos, es decir, la emisión de un bit ocupará un espacio temporal igual a 830 microsegundos. Una trama enviada por el máster consta de 2 bytes, donde se codifican el comando a ejecutar y la dirección del esclavo al que va dirigido el comando. Por su parte, los esclavos pueden responder a determinados comandos con un único byte. Todas las

tramas comienzan con un bit de inicio, igual a un 1 lógico. Un ejemplo de la trama enviada por el maestro puede verse en la Ilustración 10.

Ilustración 10. Trama de mensaje transmitida por el maestro.

En cuanto a la temporización, cabe destacar que entre dos tramas enviadas por el maestro debe existir un tiempo superior a 9.16 ms. Por su parte, entre una trama transmitida por el maestro y otra por el esclavo debe existir un tiempo superior a 2.92 ms pero inferior a 9.16 ms. Esto se ilustra en la Ilustración 11.

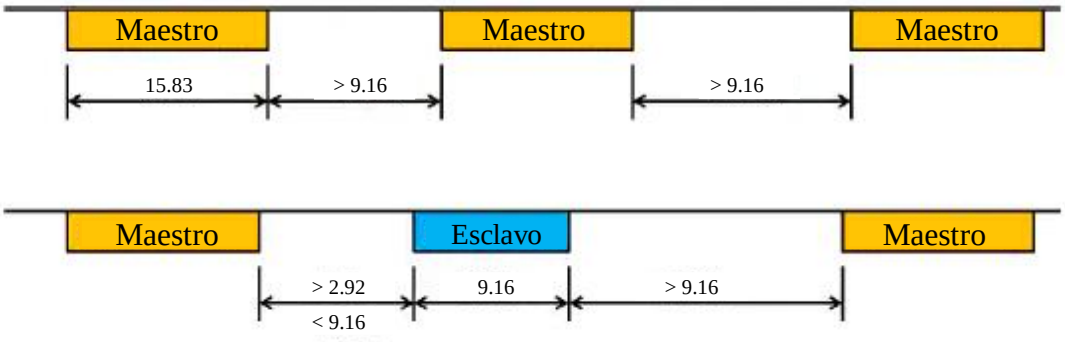


Ilustración 11. Tramas maestro - esclavo y temporización de las mismas. Unidades en milisegundos.

La emisión de bits se realiza de acuerdo al código Manchester, en el que se emplean las transiciones de estado alto y bajo (flancos de subida y bajada). En concreto, un 1 lógico será una transición de subida, es decir, de una tensión entre las líneas del bus DALI inferior a 6 voltios a una tensión superior a 9.5 voltios. Para transmitir un 0 lógico se empleará un flanco de bajada, proceso inverso al detallado previamente. En la Ilustración 12 puede verse un ejemplo a nivel de bits de esta codificación.

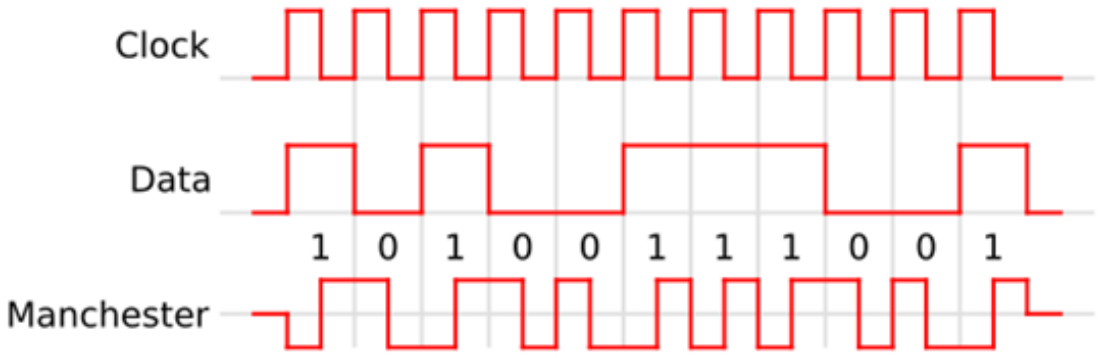


Ilustración 12. Codificación Manchester [12].

Por último, comentar que existen comandos estándar y comandos especiales. Los comandos especiales se envían como primer byte, y a continuación se envía la dirección del esclavo como segundo byte. En los comandos estándar, este orden se invierte. Los comandos especiales se identifican porque el primer bit comienza por un 1. Las

direcciones 64 direcciones disponibles para los esclavos se codifican de acuerdo con la expresión 1:

$$Addr_{cod} = 2 * Addr_{real} + 1 \quad (1)$$

Es decir, que únicamente se necesitarán 7 bits para direccionar los 64 esclavos posibles. Esto hace que, para los comandos estándar, el primer bit (el más significativo) siempre sea igual a 0 en el primer byte. Por lo que, si se recibe un 1 en el primer bit del byte correspondiente a las direcciones, se identifica ese comando como un comando especial.

Los puntos vistos en este capítulo son la base de los elementos con los que se trabajaran en el desarrollo de este proyecto, tal y como se verá en los capítulos posteriores.

Capítulo 3

Caracterización y Modelado del Sistema

Previamente al desarrollo del controlador del sistema es necesario obtener un modelo matemático formal que prediga la respuesta del mismo, a fin de poder realizar simulaciones en el ordenador.

3.1. Presentación del Sistema

En primer lugar, se procede a la presentación del sistema, y cada uno de los componentes que lo constituyen. En concreto, este consta de un microcontrolador de la familia XMC1000, una fuente de iluminación LED de tres colores (rojo, verde y azul), tres drivers para LED, y un sensor de color. En la Ilustración 13 se muestra un diagrama general del sistema. Se representan, además, las interferencias que existen entre los distintos canales de color.

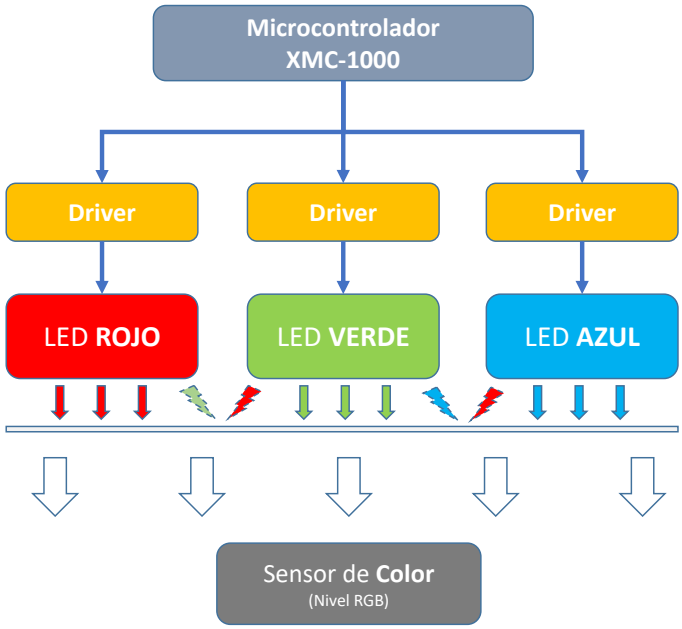


Ilustración 13. Sistema en lazo abierto.

A continuación, se explica, a modo de presentación, cada uno de los bloques que aparece en la Ilustración 13. Para una información más detallada sobre los mismo, se puede acudir al capítulo 7.

3.1.1. Microcontrolador

Para la gestión de la comunicación con el maestro a través del protocolo DALI, y la posterior generación de las señales PWM (modulación por ancho de pulso) para los drivers, se emplea un microcontrolador de la familia XMC 1000 de Infineon Technologies [13]. En concreto, se emplea el dispositivo XMC 1300, con 128 MB de memoria no volátil. La principal ventaja de este chip radica en las aplicaciones (APPs) que facilita Infineon en su entorno de desarrollo (IDE), denominado DAVE, para la programación del mismo.

3.1.2. Fuente de Iluminación LED

Por otro lado, se selecciona como fuente de iluminación una tira de 5 metros, equipada con 300 LEDs SMD (dispositivo de montaje superficial) de tres colores: rojo, verde y azul. La tensión de alimentación de los mismos es de 12 V [14].

3.1.3. Drivers

La función básica de los drivers será la de proporcionar una corriente continua a los LEDs, directamente dependiente del ciclo de trabajo de la señal PWM (salida del control del microcontrolador). Para ello, se emplea una configuración en convertidor DC/DC reductor (BUCK driver). En la Ilustración 14 se puede estudiar el esquema del circuito del driver rojo. La señal de control PWM, enviada por el microcontrolador será la señal etiquetada como "*GATE_RED*" en la ilustración. La bobina, representada por un rectángulo en rojo, será la encargada de integrar la tensión a lo largo de un periodo, proporcionando una corriente continua a los LEDs, con un pequeño rizado fruto de la señal PWM.

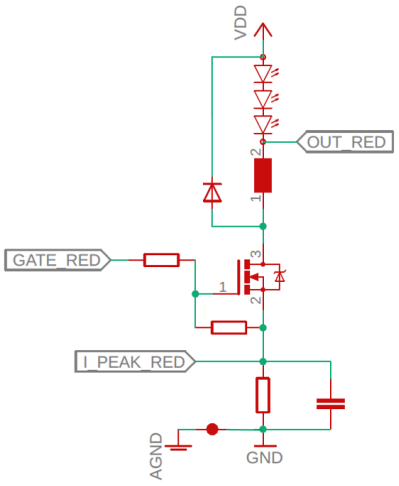


Ilustración 14. Layout del driver rojo [15].

Es interesante comentar que, a fin de poder modelar también la amplitud analógica de la corriente que circulará a través de los LEDs, se obtiene la tensión caída en una

resistencia de derivación, de muy bajo valor, en la fuente del transistor de potencia. Esta tensión será la entrada al terminal positivo de un comparador. En el terminal negativo del mismo habrá una tensión de referencia, que puede ser configurada en el microcontrolador. Cuando la tensión en la fuente del transistor supere la tensión de referencia, el comparador dará una señal lógica igual a 1 y se reseteará entonces el ciclo de la señal PWM, por lo que la corriente comenzará a disminuir su valor. De esta forma se modula la amplitud de la corriente, de forma independiente al ciclo de trabajo aplicado a la señal PWM de entrada a la puerta del transistor. Esto posibilita poder igualar la lectura del sensor para un mismo valor de ciclo de trabajo en cada una de las señales PWM de los drivers, haciendo que por el canal más ineficiente circule mayor corriente. En concreto, la tensión de referencia del comparador en el canal azul se configuró a 8 mA, en el canal verde a 25 mA, y, finalmente, a 60 mA, en el canal rojo.

3.1.4. Sensor de Color RGB

Por último, para cerrar el lazo de control del sistema, se incluye un sensor de color sensible al rojo, verde y azul. En concreto, se seleccionó el sensor TCS 34725 [16], consistente en una matriz de fotodiodos y cuatro ADC (convertidor analógico-digital). Es posible configurar el sensor con distintos tiempos de medida, correspondientes a la integración de las respectivas medidas. Además, es posible también configurar la ganancia de la medida. En concreto, para este proyecto se ha configurado el sensor con un tiempo de integración igual a 101 ms (o equivalentemente, a 42 medidas integradas, con un tiempo de conversión aproximado a 2.54 ms por cada medida), y una ganancia igual a 60. Las lecturas devueltas por el mismo necesitarán de 2 bytes ya que el valor máximo leído por el sensor es aproximadamente igual a 25000 unidades.

3.2. Respuesta Estacionaria del Sistema

Una vez presentado el sistema, el primer paso para caracterizar caracterizarlo es estudiar la respuesta estacionaria del mismo. Para ello, inicialmente se desarrolló un firmware en C para el microcontrolador, que permitiese la modificación del valor del ciclo de trabajo para cada una de las señales PWM de los tres canales, además de realizar lecturas del sensor. Seguidamente, se implementó un programa en MATLAB que tomaba 64 valores linealmente espaciados dentro del rango de valores [0, 4095] relativos al ciclo de trabajo de la señal PWM, y promediaba 20 medidas distintas del sensor para cada valor enviado, eliminando de esta forma posibles perturbaciones o ruido al realizar las medidas. Durante este proceso, únicamente se encendía e incrementaba la señal PWM de un canal, estando los otros dos completamente apagados. Las figuras devueltas pueden estudiarse en la Ilustración 15. En ellas se representa la lectura del sensor (eje Y) frente al valor de la señal PWM del driver (eje X).

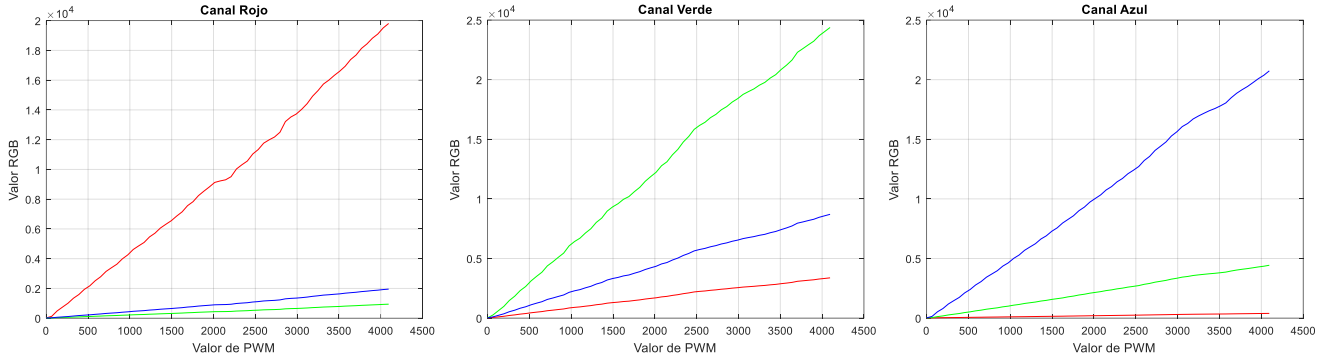


Ilustración 15. Respuesta estacionaria de los tres canales.

Como puede verse, cada uno de los canales, al no tener un cromatismo ideal, arroja una determinada cantidad de longitudes de onda correspondientes a los otros colores primarios, es decir, hay interferencias entre los canales. En concreto, existe una importante interacción entre el canal verde y el azul. Este hecho complicará de forma considerable el control, e implicará la inclusión de una red de desacoplo para la eliminación de estas interferencias.

Se obtiene así la matriz de ganancias en estado estacionario (SSGM), presentada en la Tabla 1.

Salida / Entrada	Rojo	Verde	Azul
Rojo	5.151	0.228	0.445
Verde	0.829	5.93	2.132
Azul	0.0967	1.074	5.068

Tabla 1. Ganancias en estado estacionario del sistema.

3.3. Respuesta Dinámica del Sistema

A continuación, se procede a modelar la respuesta dinámica del sensor. Para ello, se elaboró otro programa en MATLAB donde se recogían un total de 100 muestras de la lectura del sensor y se introducía un escalón en cada canal (por separado) de valor igual a 4095 (valor máximo). El resultado para los tres canales fue similar al mostrado en la Ilustración 16 con el canal verde. Como puede observarse, existe una dinámica en la respuesta del sistema, aunque esta es muy rápida. En esta figura se representa cada una de las muestras correspondientes a la lectura del sensor RGB (representado con círculos) frente al tiempo.

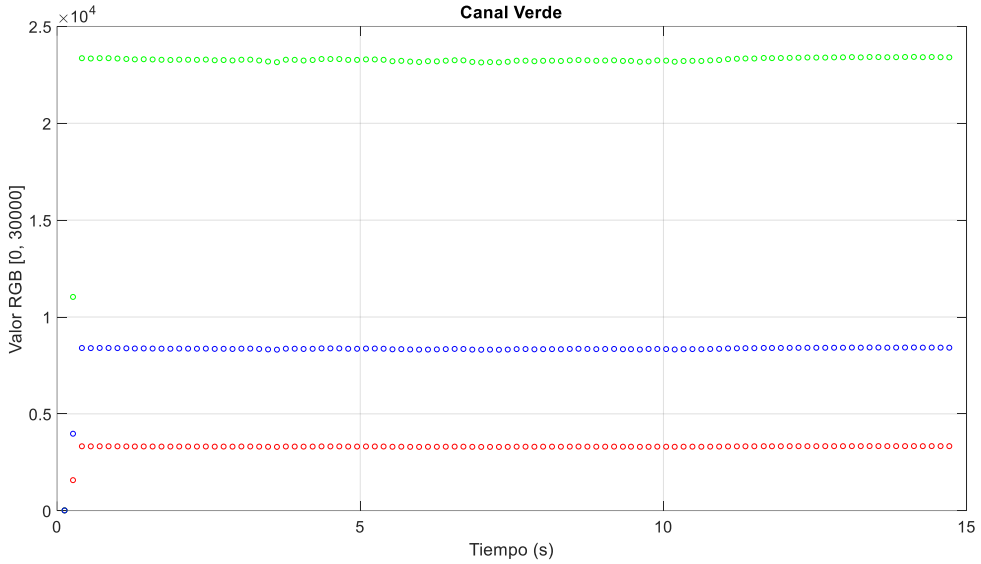


Ilustración 16. Respuesta dinámica del sistema.

Para modelar la respuesta del sistema entonces, se ajustó la función de transferencia de primer orden presentada en la expresión A, modificando las correspondientes ganancias. Así se obtuvieron las distintas dinámicas expresadas en la Tabla 2. El resultado de comparar la respuesta del sistema real (representación en círculos) con los modelos elaborados (representación en línea continua) puede observarse en la Ilustración 17.

$$G_1(s) = e^{-0.15s}\frac{K_{ij}}{\frac{s}{20}+1}; \qquad \forall i,j \in \{R,G,B\} \tag{A}$$

Entrada \ Salida	Rojo	Verde	Azul
Rojo	$e^{-0.15s}\frac{5.151}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{0.228}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{0.445}{\frac{s}{20}+1}$
Verde	$e^{-0.15s}\frac{0.829}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{5.93}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{2.132}{\frac{s}{20}+1}$
Azul	$e^{-0.15s}\frac{0.097}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{1.07}{\frac{s}{20}+1}$	$e^{-0.15s}\frac{5.068}{\frac{s}{20}+1}$

Tabla 2. Respuestas dinámicas del sistema.

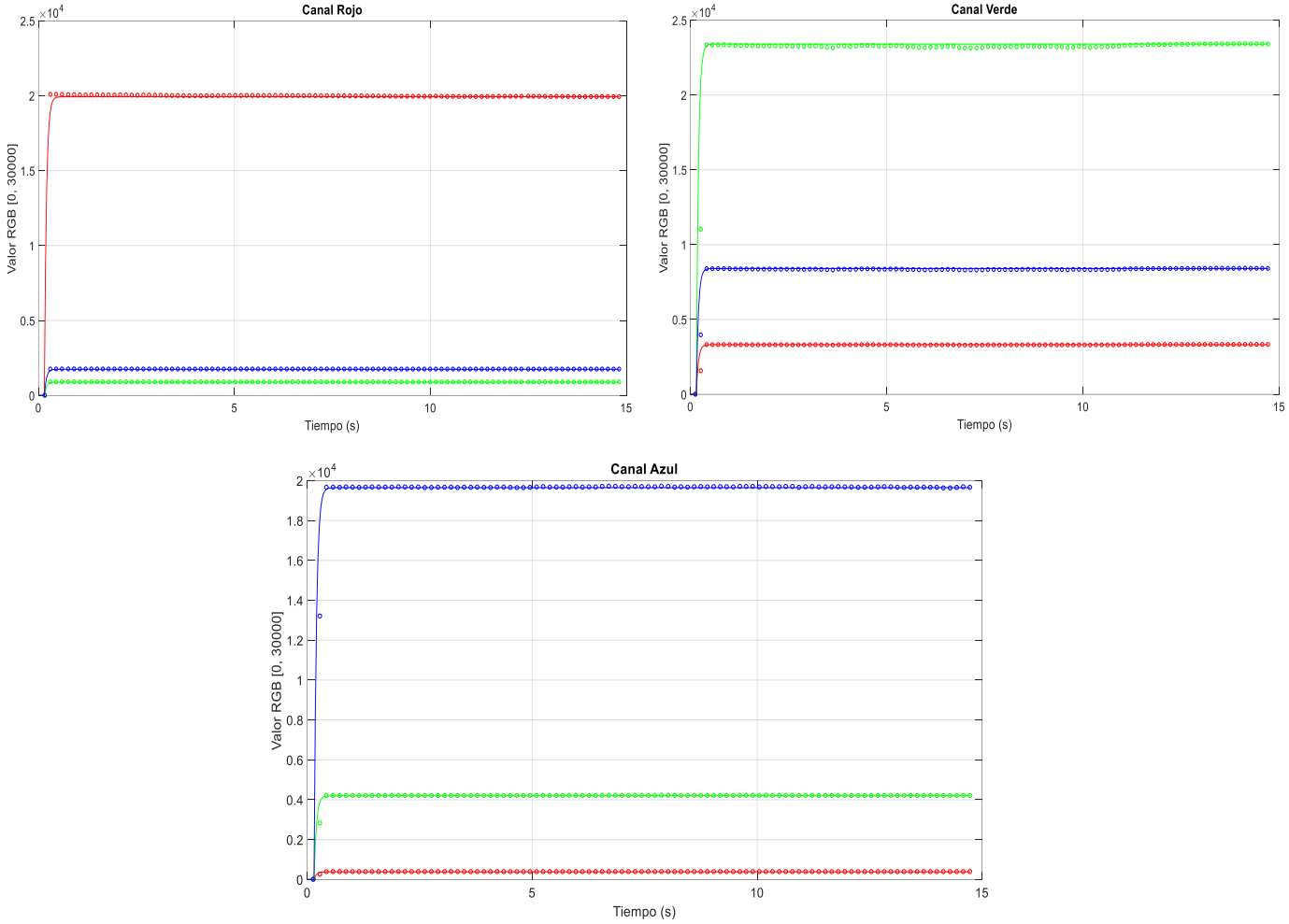


Ilustración 17. Sistema real frente al modelo del sistema.

3.4. Modelo en Simulink

Por último, es de interés presentar el modelo implementado en Simulink para la simulación del sistema durante la optimización de los parámetros del controlador por medio de un algoritmo de optimización heurística. La Ilustración 18 muestra una captura del modelo construido en Simulink. Además, en la Ilustración 19 puede estudiarse la respuesta del sistema ante un escalón en la entrada roja, a los 10 segundos, en la entrada verde a los 20 segundos, y finalmente en la entrada azul a los 30 segundos. Es evidente que el sistema no sigue la señal de referencia, ya que el valor al que se estabilizan los canales es muy distinto al de consigna (la unidad).

Es de interés destacar, tal y como se ha predicho anteriormente, el fuerte acoplamiento que existe entre el canal verde y el azul. En concreto, a los 20 segundos, cuando se cambia la consigna del canal verde, la lectura del canal azul pasa de 0.45 (la influencia sobre el color azul que tiene el canal rojo), a 2.58. Estas dependencias radican

en la incapacidad de obtener un color primario (rojo, verde o azul) totalmente perfecto con fuentes de iluminación reales.

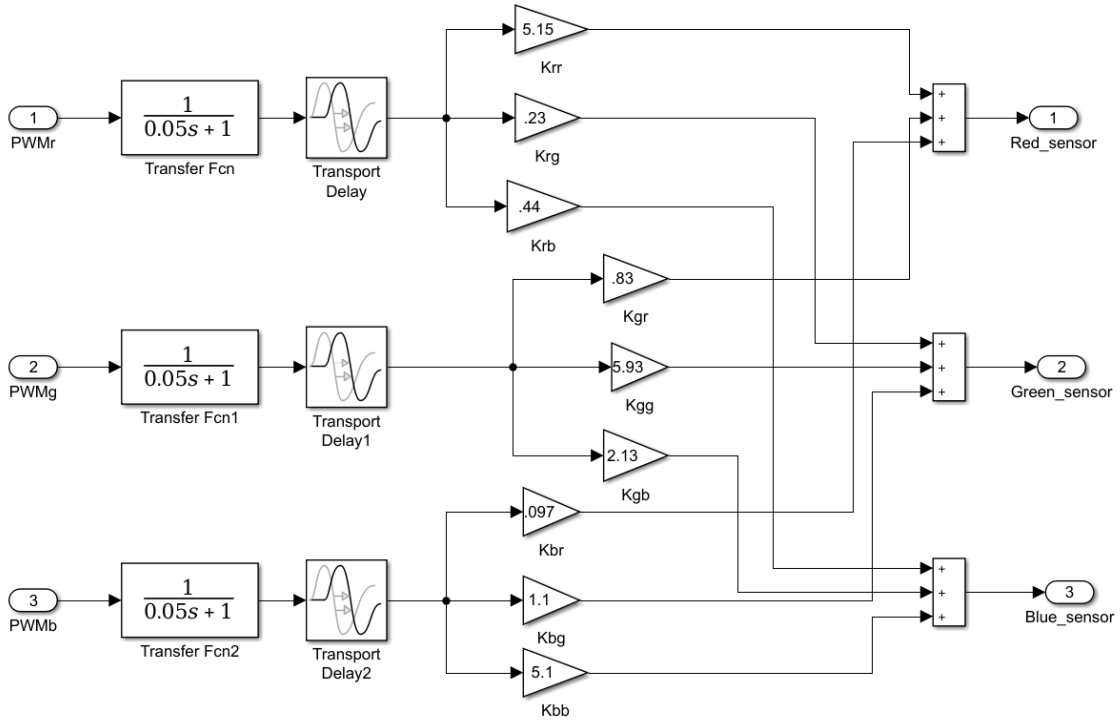


Ilustración 18. Diagrama de bloques del modelo construido en Simulink.

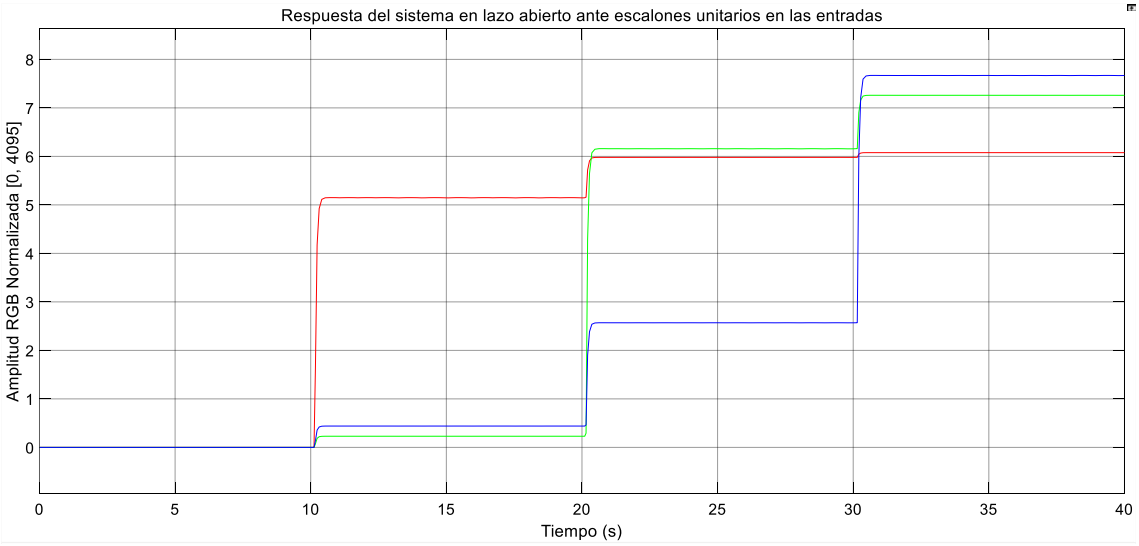


Ilustración 19. Respuesta del sistema ante un escalón unitario en cada una de las tres entradas.

Tras este estudio, se ha obtenido entonces un modelo matemático que es capaz de predecir las salidas del sistema real (lectura del sensor RGB) ante una entrada concreta (valor de PWM), es decir, un modelo equivalente al recuadro azul celeste del diagrama

presentado en la Ilustración 4, y al correspondiente sensor de color. Esto permite la simulación por computador del sistema, y el análisis del comportamiento del sistema cerrado antes de realizar las pruebas sobre el sistema real.

Capítulo 4

Estudio del Control PID

En este capítulo se expone el estudio realizado para el control del sistema a través de un control PID. En primer lugar, analizando el sistema desde un enfoque de control multivariable, se plantea la implementación de una red de desacoplo para la eliminación de las dependencias existentes entre las variables que se desean controlar. A continuación, se aplican técnicas de optimización heurística para la sintonización y optimización de los parámetros propios del controlador PID.

4.1. Desacoplo de Variables

Como se ha visto en la Ilustración 19, existe un fuerte acoplamiento entre las variables, especialmente entre la entrada verde y la azul. En este trabajo se propone la implementación de una red de desacoplo entre las variables del sistema para posibilitar el control de cada canal de forma independiente, como si fueran sistema totalmente aislados. Gracia a que, como se ha visto anteriormente, la respuesta dinámica de los tres canales puede aproximarse por un modelo matemático común, en el que únicamente se cambia la ganancia de cada planta, es posible estudiar el control de un canal mediante un control PID, y, posteriormente, modificar las ganancias para adaptar el mismo control a los demás canales. De esta forma, como se verá posteriormente, el problema de optimización ve reducida su complejidad al estudio con únicamente tres variables de decisión.

Es importante recalcar a favor de esta propuesta de control que, dado que los tres canales se aproximan por la misma dinámica, también lo hacen así sus dependencias. Esto implica que la red de desacoplo acaba siendo simplemente una red de ganancias, debido a la cancelación de las dinámicas de cada canal. Se simplifica así enormemente su implementación al no existir transitorios que tener en cuenta.

Sería también posible implementar tres controladores distintos, que tuvieran en cuenta las dependencias, pero ya sería necesaria la sintonización de tres controladores, y la optimización de nueve variables de decisión (tres por cada controlador PID), aumentando la complejidad del estudio.

El objetivo de la red, por lo tanto, es eliminar la respuesta de las salidas j ($\forall j \neq i$) cuando la entrada i cambia de consigna. Por ejemplo, suponiendo que el canal verde cambia su valor de consigna, se espera que tras modificar el mismo, la lectura del valor rojo o azul sea exactamente la misma que al principio.

4.1.1. Estudio de las Dependencias

Se procede a analizar el grado de dependencia entre las variables de entrada y de salida. La SSGM se ha mostrado en la Tabla 1. Tomando esta como punto de partida, se procede a estudiar la matriz de ganancias relativas, también llamada RGA (*Relative Gain Array*), para analizar y ponderar la interacción, a fin de identificar que entradas serán más óptimas para controlar cada una de las salidas [17]. Así, la matriz RGA obtenida es:

$$\Lambda = \begin{bmatrix} 1.006 & -0.006 & 0.001 \\ -0.004 & 1.086 & -0.082 \\ -0.002 & -0.08 & 1.081 \end{bmatrix} \qquad (B)$$

Tal y como puede estudiarse en la bibliografía [17], la proximidad a la unidad en la matriz RGA implica la bondad en el ajuste de la salida j a partir de la entrada i. En este caso entonces se confirma que la mejor entrada para controlar la salida roja es a su vez la roja. Análogamente ocurre con el canal verde y el azul.

Se definen a continuación los desacoplos entre la entrada roja y las salidas verde y azul, entre la entrada verde y la salida roja y azul, y entre la entrada azul y la salida roja y verde.

Por otro lado, se calcula el número de condición [17] para dar una medida cuantificable de las interacciones entre las variables. En concreto, este resulta en:

$$\gamma = 1.8995 \qquad (C)$$

Es decir, si bien el valor es razonablemente bueno para el desacoplo de variables, si insinúa una interacción importante.

4.1.2. Desacoplo Inverso

La Ilustración 20 muestra un diagrama de bloques de una red de desacoplo inverso en configuración A. La idea básica para elaborar esta red es invertir el efecto que tiene sobre la variable i las dependencias con el resto de variables.

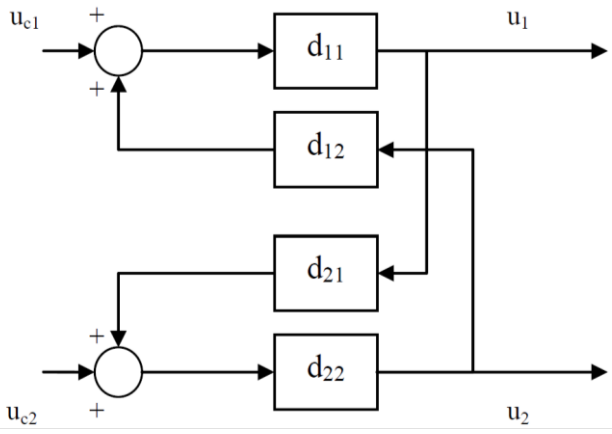


Ilustración 20. Red de desacoplo inverso 2x2 en configuración A [18].

El principal motivo de la selección de este tipo de desacoplo es la simplicidad que implica su construcción en los elementos de la red de desacoplo, como consecuencia de las funciones aparentes resultantes.

Para poder implementar la red de desacoplo inverso en configuración A, es necesario que se cumplan determinadas restricciones. Concretamente:

- Si existen retardos en las funciones de transferencia: $\tau_{12} \geq \tau_{11}$, $\tau_{13} \geq \tau_{11}$, $\tau_{21} \geq \tau_{22}$, $\tau_{23} \geq \tau_{22}$, $\tau_{31} \geq \tau_{33}$ y $\tau_{32} \geq \tau_{33}$.

- Ni g_{11} ni g_{22} ni g_{33} pueden tener ceros con parte real positiva.

- $grado(numerador\ g_{11}) + grado(denominador\ g_{12}) \geq grado(numerador\ g_{12}) + grados(denominador\ g_{11})$.

- $grado(numerador\ g_{11}) + grado(denominador\ g_{13}) \geq grado(numerador\ g_{13}) + grados(denominador\ g_{11})$

- $grado(numerador\ g_{22}) + grado(denominador\ g_{21}) \geq grado(numerador\ g_{21}) + grados(denominador\ g_{22})$.

- $grado(numerador\ g_{22}) + grado(denominador\ g_{23}) \geq grado(numerador\ g_{23}) + grados(denominador\ g_{22})$.

- $grado(numerador\ g_{33}) + grado(denominador\ g_{31}) \geq grado(numerador\ g_{31}) + grados(denominador\ g_{33})$.

- $grado(numerador\ g_{33}) + grado(denominador\ g_{32}) \geq grado(numerador\ g_{32}) + grados(denominador\ g_{33})$

Dado que, en el modelo de estudio, se ha verificado que la dinámica de todas las respuestas puede modelarse de acuerdo a una única fórmula laplaciana (Tabla 2), todas las condiciones aquí presentadas se cumplen, garantizando entonces la posibilidad de implementación de la red.

Imponiendo que la salida de la red sea igual a la entrada en aquellas variables que han sido seleccionadas para controlar las respectivas salidas, se obtienen las expresiones mostradas en la primera columna de la Tabla 3, extendidas para el caso de un sistema con tres entradas y tres salidas [19]. Además, en esta se muestra el resultado de cada elemento que constituye la red.

Elementos de la Red de Desacoplo	Valores Reales para el Sistema	Proceso Aparente Final
$d_{11} = 1$	$d_{11} = 1$	$q_{11} = g_{11}$
$d_{12} = -\frac{g_{12}}{g_{11}}$	$d_{12} = -0.161$	$q_{12} = 0$
$d_{13} = -\frac{g_{13}}{g_{11}}$	$d_{13} = -0.0188$	$q_{13} = 0$
$d_{21} = -\frac{g_{21}}{g_{22}}$	$d_{21} = -0.038$	$q_{21} = 0$
$d_{22} = 1$	$d_{22} = 1$	$q_{22} = g_{22}$
$d_{23} = -\frac{g_{23}}{g_{22}}$	$d_{23} = -0.181$	$q_{23} = 0$
$d_{31} = -\frac{g_{31}}{g_{33}}$	$d_{31} = -0.088$	$q_{31} = 0$
$d_{32} = -\frac{g_{32}}{g_{33}}$	$d_{32} = -0.421$	$q_{32} = 0$
$d_{33} = 1$	$d_{33} = 1$	$q_{33} = g_{33}$

Tabla 3. Valores de la red de desacoplo 3x3 [19].

4.1.3. Implementación en Simulink

Una vez obtenido el desacoplo inverso, el siguiente paso es construir el modelo en Simulink y simularlo. En la Ilustración 21 se muestra el resultado de dicho montaje, donde el bloque de la derecha (LED/Sensor) encapsula el comportamiento del sistema representado en la Ilustración 18. Es importante comentar que, para evitar errores de simulación, es necesario añadir un retardo (mínimo, por ejemplo, de 10 ms). Esto evita que se produzcan lazos algebraicos (es decir, que una misma señal sirva para su propia definición, como ocurre por ejemplo en u(k)=Ku(k))

Por otro lado, en la Ilustración 22 se muestra la simulación de la respuesta dinámica del sistema con la red de desacoplo tras introducir un escalón unitario en cada una de las tres entradas. El primero a los 10 segundos en la entrada roja, el segundo a los 20 en la entrada verde, y finalmente el último en la entrada azul a los 30 segundos.

Si bien el sistema aún no está controlado correctamente (los valores estacionarios siguen siendo distintos de las consignas unitarias introducidas), es directo observar que

se ha eliminado la interacción existente entre los tres canales, quedando únicamente unas pequeñas oscilaciones de muy baja amplitud cuando acontece el cambio de consigna en los otros canales.

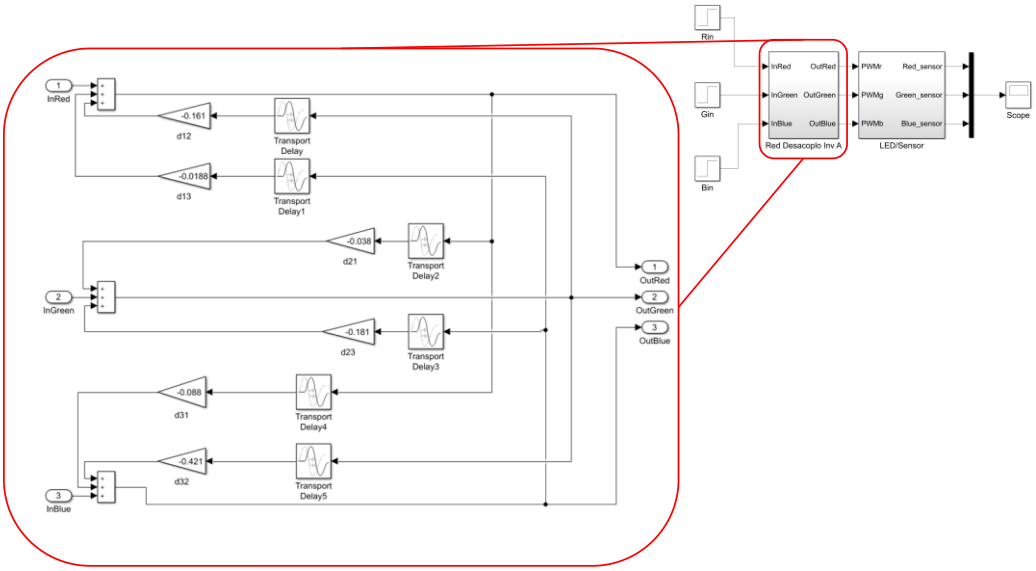


Ilustración 21. Diagrama en Simulink de la red de desacoplo.

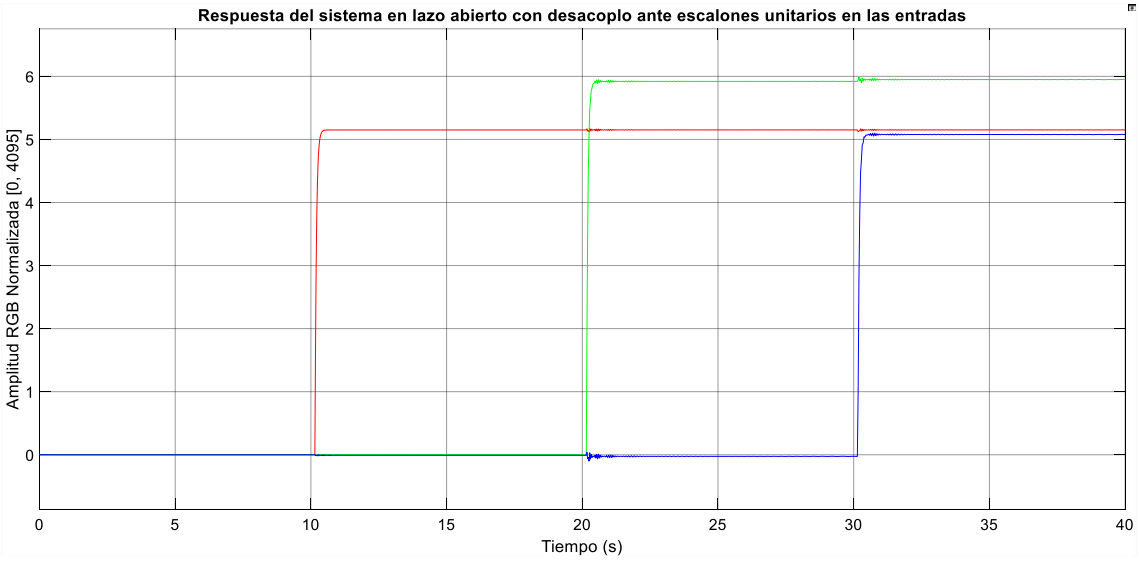


Ilustración 22. Respuesta del sistema en lazo abierto con red de desacoplo.

Aunque el resultado mostrado en la Ilustración 22 es altamente satisfactorio, la red de desacoplo diseñada arroja un serio problema en las señales de control. Dicho problema se pone de manifiesto Ilustración 23, que muestra el valor de las señales de control (PWM) generadas por la red cuando se producen cambios de consigna. Como puede observarse, a los 10 o 20 segundos, las señales de control del canal verde y azul se estabilizan en valores negativos, algo que, en el sistema real, con las señales de control PWM saturadas entre 0 y 4095, es imposible. Esto es debido a la imposibilidad de obtener colores primarios puros, sin longitudes de onda correspondientes a otros colores, con

fuentes de iluminación reales. Debido a esto, será necesario asegurar una "cantidad mínima" de cada color primario en las consignas exigidas al control PID, para asegurar su correcto funcionamiento.

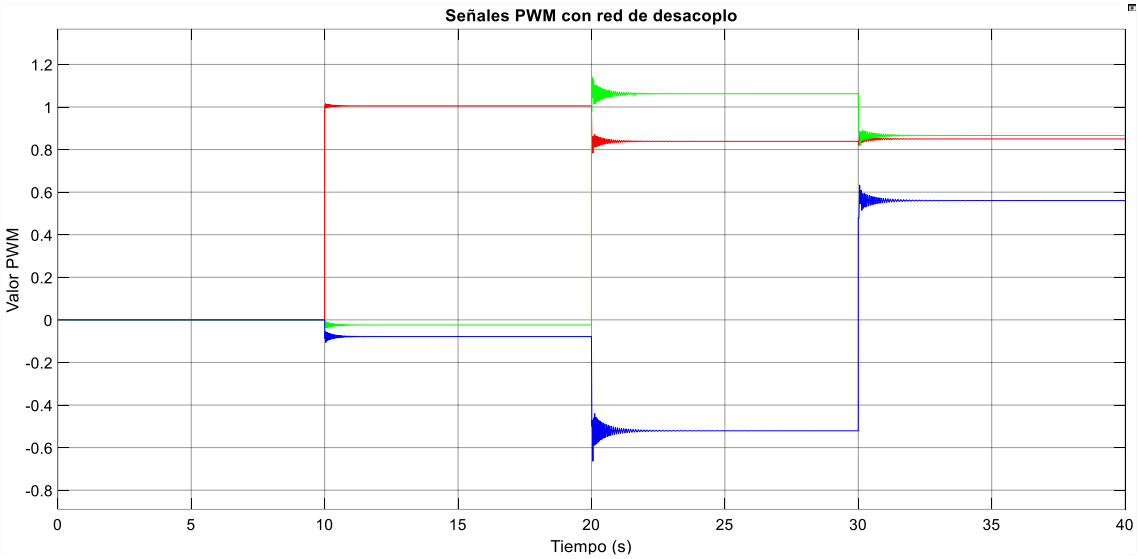


Ilustración 23. Valor de las señales de control con la red desacoplo.

4.2. Problema de Optimización Heurística

Una vez modelado el sistema, y planteado el desacoplo de las variables, se procede a implementar el control PID para la consecución del color deseado. Para ello, se enfoca la sintonización del controlador PID como un problema de optimización, aplicando técnicas heurísticas para la obtención de resultados aceptables.

El objetivo principal de esta búsqueda será encontrar los valores óptimos de los coeficientes K_P, K_I y K_D del controlador PID para la obtención de una respuesta dinámica adecuada. Para ello se eligen los parámetros de la respuesta dinámica que se desean optimizar, que constituyen las diferentes funciones objetivo del problema. Además, se imponen una serie de restricciones sobre las variables de decisión (coeficientes del controlador PID) que limitan la región factible de dichas variables.

4.2.1. Variables de Decisión

Como se ha comentado anteriormente, las variables de decisión serán los parámetros propios del controlador PID, es decir, los coeficientes que ponderan la proporcionalidad, la integración y la derivación del error sobre la nueva señal de control, de acuerdo con la combinación lineal de componentes mostrada en la ecuación 2.

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \qquad (2)$$

4.2.2. Restricciones

Las restricciones se imponen, en este caso, atendiendo a las simulaciones del modelo. Se restringen los valores factibles de las variables de decisión a aquellos valores que simplemente permiten la simulación del modelo en lazo cerrado. Se impone entonces:

1. **Limitación del dominio de la constante proporcional K_P:** La variable de decisión relativa a la constante proporcional del controlador PID únicamente puede tomar valores entre 0 y 2.

2. **Limitación del dominio de la constante integral K_I:** Por su parte, la constante integral únicamente puede tomar valores entre 1 y 10.

3. **Limitación del dominio de la constante derivativa K_D:** Finalmente, se limitó el espectro de la variable derivativa entre 0 y 0.005. Como puede verse, esta es la más restringida, ya que también era la que más alteraba las simulaciones y, por lo tanto, la ejecución del algoritmo.

4.2.3. Funciones Objetivo

Por último, también se modelan las funciones objetivo, concerniendo los parámetros que se quieren optimizar de la respuesta dinámica del sistema en lazo cerrado. A continuación, se presentan cada una de las cuatro funciones objetivo, motivando además su selección.

- **Minimizar el tiempo de subida.**

La primera función objetivo persigue alcanzar lo más pronto posible el valor deseado (consigna) de intensidad en el canal a controlar. En concreto, se tomará como tiempo de subida al tiempo que tarda la salida en ir desde el 10% del valor de consigna hasta el 90%.

- **Minimizar la energía de la señal de control.**

La siguiente función objetivo busca obtener unas señales de control primeramente realizables, es decir, que no sobrepasen el máximo de 4095 restringido por el driver de los LEDs para configurar el ciclo de trabajo del PWM. Para lograrlo se premiará aquellos valores del PID que minimicen la energía de la señal de control aplicada a los LEDs, entendiendo esta como la integral del cuadrado de la propia señal. En concreto, para medir la energía se empleará la expresión 3:

$$E = \int_0^t u^2 dt \tag{3}$$

Que, si bien eleva al cuadrado el valor de la señal de control, sirve para estimar que sistema empleará más energía.

- **Minimizar el sobredisparo.**

No es deseable que existan "relámpagos" de luz consecuencia de un sobredisparo al cambiar la consigna del cromatismo de la lámpara. Por ello, se pide minimizar posibles picos en la respuesta dinámica controlada del sistema. De esta forma, además, se consigue una respuesta sobreamortiguada, ya que la dinámica presentará un comportamiento creciente. Este objetivo está estrechamente relacionado con el primero, ya que, a menor tiempo de subida, mayor probabilidad de obtener un sistema subamortiguado, es decir, de que exista sobredisparo.

- **Minimizar posibles cambios de pendiente**

Finalmente se añade este cuarto y último objetivo, fruto de las respuestas dinámicas obtenidas tras correr el algoritmo de optimización. En la Ilustración 24 se muestra una de las respuestas teóricamente óptimas para el controlador PID antes de añadir este objetivo. Como puede verse, no existe sobredisparo alguno, sin embargo, existen unas fluctuaciones en la subida que, al igual que el sobredisparo, son altamente indeseable. Por lo tanto, se incluyó este objetivo en el problema.

Básicamente lo que se estudiará es si la respuesta tiene, en algún instante, una pendiente (derivada) negativa. En caso afirmativa se sumará el valor de esta derivada para cada una de las muestras, y, posteriormente, se dividirá su valor final entre el número de muestras con pendiente negativa (derivada ponderada). Este valor será el parámetro que medirá los indeseados cambios de pendiente. Además, dado que en las sobreelongaciones también existen pendientes negativas, esta función objetivo también minimizará el posible sobredisparo de la respuesta dinámica en cuestión.

A fin de poder implementar este planteamiento como un problema mono-objetivo, se considera una suma ponderada de los cuatro objetivos aquí presentados. En esta ponderación, los pesos aplicados deben tener en cuenta:

a. Por un lado, el valor de fondo de escala que estas funciones objetivo pueden tener, ya que, en caso de que no exista la misma escala en los valores de las funciones objetivo, la función con una mayor amplitud pesará más (inevitable e indeseablemente) en la ponderación.

b. Un segundo concepto a tener en cuenta en la selección de estos pesos es la importancia que cada una de estas funciones objetivo tendrá en la optimización. Las funciones objetivo cuya optimización es más deseable deberán pesar más, para que el algoritmo tienda a buscar soluciones que las optimicen.

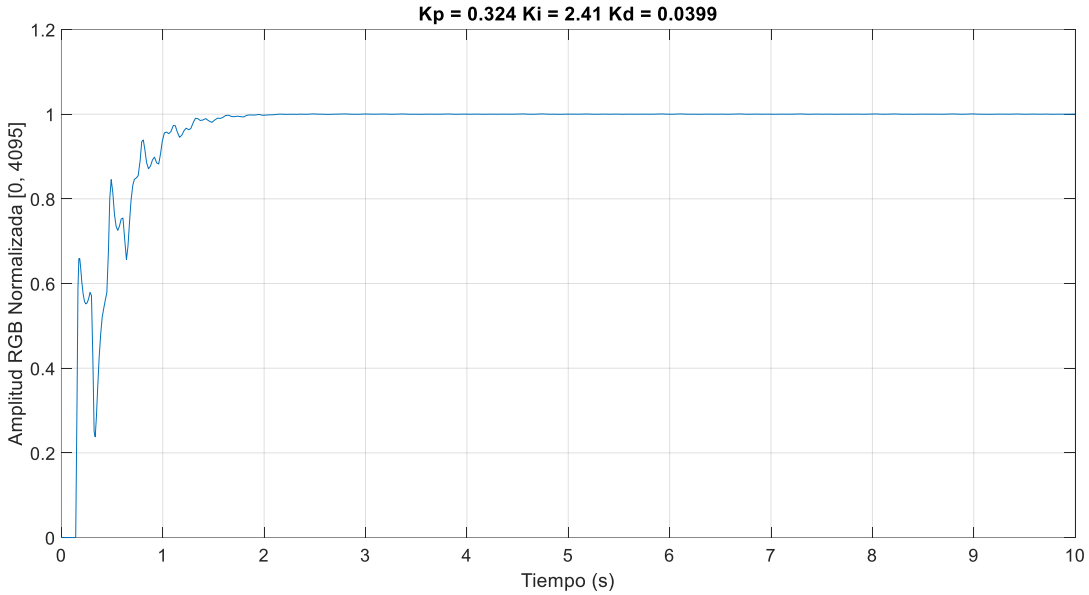


Ilustración 24- Respuesta dinámica con pendientes negativas indeseados.

En el algoritmo desarrollado se ha intentado, en primera instancia, ajustar los pesos de cada función objetivo para relativizar a la unidad sus valores correspondientes. Posteriormente, la idea ha sido ponderar el peso deseado a cada función. Aunque han sido muchos los valores probados para los distintos pesos, finalmente se optó por pesar el tiempo de subida de la respuesta con 1, la energía del sistema con 0.1 (ya que su valor de salida para soluciones estables suele rondar la decena), el sobredisparo con 0.5 (ya que puede tener también valores altos en las respuestas dinámicas recogidas, y además no se desea que pese demasiado en la función objetivo final), y 10 para los cambios de pendiente (ya que esta es una característica totalmente indeseable en las soluciones y además su valor de salida suele ser bastante bajo en comparación con el resto).

4.3. Algoritmo de Optimización

Para la optimización del problema planteado se probaron varios algoritmos, concretamente el algoritmo de temple simulado, un algoritmo genético y el algoritmo de nubes de partículas. Fue este último el que devolvió un resultado más adecuado, ya que alcanzó las mismas soluciones óptimas devueltas por los otros, en un menor tiempo de computo. Es por ello que en la presente memoria se procede a presentar en líneas general este algoritmo. Para el ajuste de los parámetros del algoritmo de nubes de partículas se realizó un estudio estadístico, en el que se compararon las curvas medias de evolución de la función objetiva ponderada, y se determinó la parametrización que devolvía habitualmente una solución mejor en un tiempo razonable.

La idea básica de este algoritmo es mover cada una de las soluciones de la población actual hacia el máximo local observado por dicha población y el máximo global observado por el algoritmo. Más concretamente, para cada partícula (solución del

problema) calcula una velocidad de avance (en la que tiene en cuenta las direcciones previamente mencionadas) y se la aplica a la posición actual de la partícula.

La Ilustración 25 muestra el diagrama de flujo del algoritmo de nube de partículas. En ella se observa que el primer paso es la generación de la población inicial. Esta generación se crea de forma aleatoria, siguiendo una distribución uniforme en todo el espacio factible. Por su parte, las velocidades iniciales de cada partícula se inicializan a 0.

A continuación, sigue la evaluación de la suma ponderada de las funciones objetivo, para cada una de las partículas presentes en la población actual. Hecho esto, se define como solución local ($x_{l,i}$) de cada partícula la solución generada en la inicialización y como solución global del algoritmo (x_g) la mejor solución de la población inicial.

Seguidamente se procede a la actualización de la velocidad de cada partícula de acuerdo con la expresión mostrada en 4:

$$v_i = w \cdot u_w \cdot v_i + c_l \cdot u_l \cdot \Delta(x_{l,i} - x_i) + c_g \cdot u_g \cdot \Delta(x_g - x_i) \quad (4)$$

En concreto, la velocidad de la partícula i será igual a la suma ponderada de su velocidad anterior, más la diferencia entre la posición de dicha partícula con la posición de su mínimo local, más la diferencia entre la posición de dicha partícula con la posición de su mínimo global. Además los pesos w, c_l y c_g ponderan la importancia de estas variables sobre la nueva velocidad. Por último, se incluyen otros coeficientes generados aleatoriamente (u_w, u_l y u_g) para modificar estos pesos en cada iteración e introducir aleatoreidad en el recorrido del espacio de búsqueda por el algoritmo de nube de partículas.

Tras esto, se actualiza la posición de cada partícula, de acuerdo con la expresión presentada en 5:

$$x_i = x_i + v_i \quad (5)$$

A continuación, una vez que esto se ha realizado para todas las soluciones de la población actual, se realiza el proceso de mutación, en el que, para cada uno de los dígitos que describen la posición de cada una de las soluciones, se calcula una probabilidad, y si esta es inferior a 0.02, se modifica de forma aleatoria, dentro del rango permitido, el valor de dicho dígito.

Además, en cada iteración, se generan 10 soluciones de forma aleatoria, siguiendo una distribución uniforme. Estas 10 soluciones son conocidas como “inmigrantes” y competirán con la población actual.

El siguiente paso es evaluar la población de nuevas soluciones y los inmigrantes, y encontrar el mínimo local. Si dicho mínimo es inferior (más óptimo) que el global, se almacena esta solución como mejor solución global. Terminada esta comprobación, se vuelve a empezar.

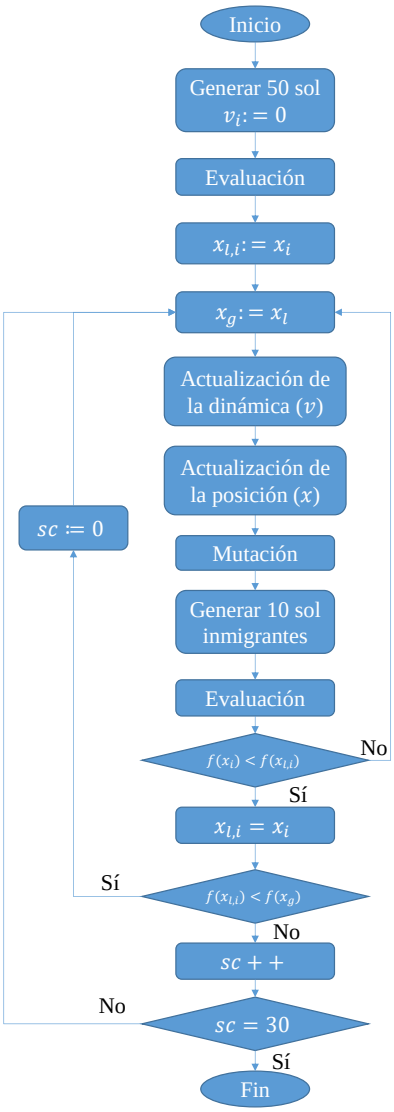


Ilustración 25. Diagrama de flujo del algoritmo de nube de partículas [20].

Este proceso se repetirá tantas veces como sean necesarias, hasta que se alcance una solución global que sea imbatible durante un total de 30 iteraciones. En este caso se concluye que dicha solución es la óptima y finaliza el algoritmo.

Por último, es de interés mostrar el gráfico de la Ilustración 26, donde se muestra el valor medio de la función ponderada del problema sobre 30 ejecuciones del algoritmo de nube de partículas, para cada una de las iteraciones. Además, se muestra también la desviación estándar sobre dicha media. Es directo comprobar como, por un lado, la desviación estándar disminuye y se estabiliza tras aproximadamente 45 iteraciones. Por otro lado, el valor medio de la función objetivo es estrictamente decreciente a lo largo de todas las iteraciones de cada ejecución. Esto demuestra el correcto funcionamiento del algoritmo.

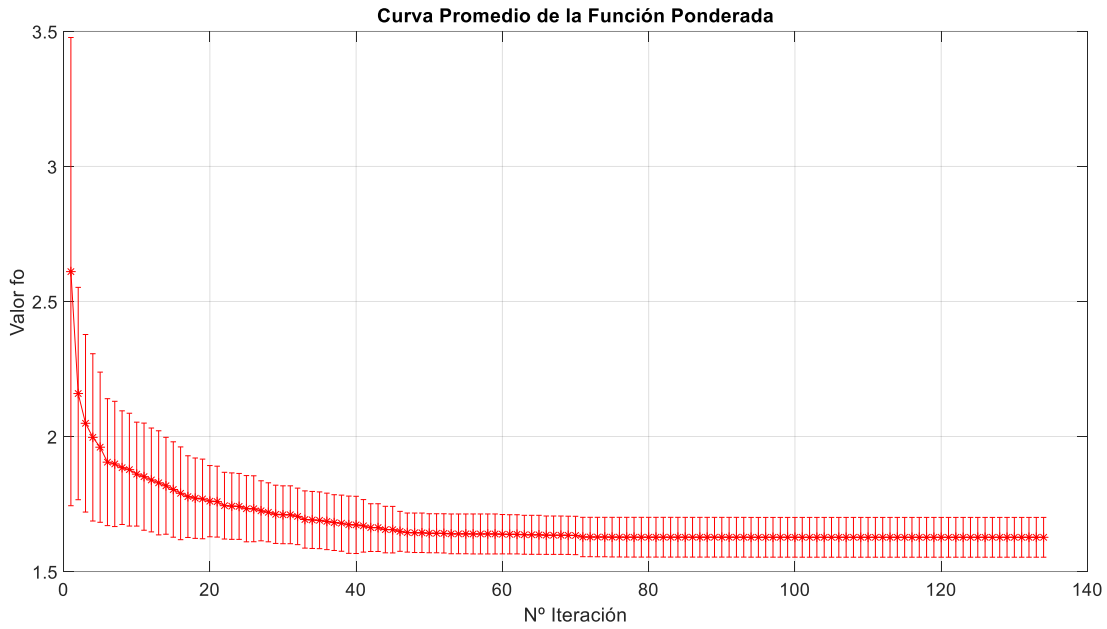


Ilustración 26. Valor medio y desviación estándar de las soluciones en 30 ejecuciones del algoritmo.

4.4. Resultados de la Optimización

A continuación, se procede a detallar los resultados recogidos tras correr el algoritmo de nube de partículas. Es conveniente recalcar que, si bien inicialmente las simulaciones se realizaron en tiempo continuo, el sistema real se controlará en tiempo discreto. Por lo tanto, se optimizarán los valores del PID para las dos configuraciones (continuo y discreto).

4.4.1. Sistema continuo

La Ilustración 27 muestra una de las soluciones óptimas recogidas en el algoritmo de nube de partículas para el sistema continuo. Los parámetros recogidos en esta respuesta son presentados en la Tabla 4.

Es directo comprobar que la respuesta dinámica obtenida para este control PID es considerablemente buena. En apenas medio segundo se alcanza el valor unitario de consigna, casi sin sobredisparo, y con un valor de energía razonable, en comparación con otras soluciones devueltas.

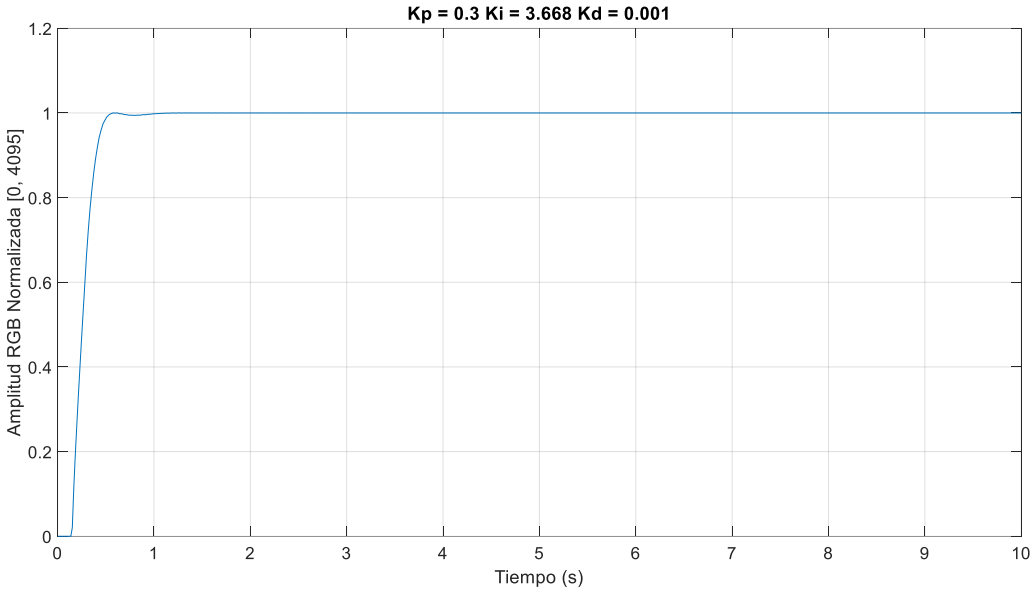


Ilustración 27. Respuesta dinámica de un canal modelo controlado.

Parámetro	Valor
K_P	0.3
K_I	3.668
K_D	0.001
Tiempo de Subida (s)	0.5096
Energía señales de control	9.995
Sobredisparo (%)	0.0152
Pendiente negativa	0.0025
Valor función objetivo	1.567
Tiempo de Computo (s)	968.87

Tabla 4. Resumen de las características dinámicas del control PID óptimo encontrado.

4.4.2. Sistema discreto

Como se ha adelantado anteriormente, el control real de sistema se realizará en tiempo discreto, ya que la recogida de datos por parte del sensor requiere de un periodo de 101 ms. Además, dado que para enviar los comandos de control del maestro DALI al esclavo DALI se necesitan aproximadamente 17 ms, y se enviarán en total 6 en cada iteración, el periodo del sistema asciende entonces a 204 ms. Para modelar este comportamiento en Simulink, simplemente se configuraron los controladores PID de cada canal a tiempo discreto, empleando una aproximación trapezoidal. Además, los retardos de la red de desacoplo también se discretizaron, empleando el periodo comentado igual a 204 ms. A continuación, se volvió a correr el algoritmo de nube de partículas para este sistema en tiempo discreto, obteniéndose la respuesta mostrada en la Ilustración 28. Los datos recogidos para esta respuesta dinámica se presentan en la Tabla 5.

En este caso, el sistema empeora su respuesta, como puede verse por el valor de la función objetivo de la solución encontrada. En general, esta ha sido la tónica en todas las optimizaciones realizadas en tiempo discreto. Sin embargo, cabe destacar el correcto funcionamiento del sistema sintonizando el control PID con los parámetros mostrados en la Tabla 5.

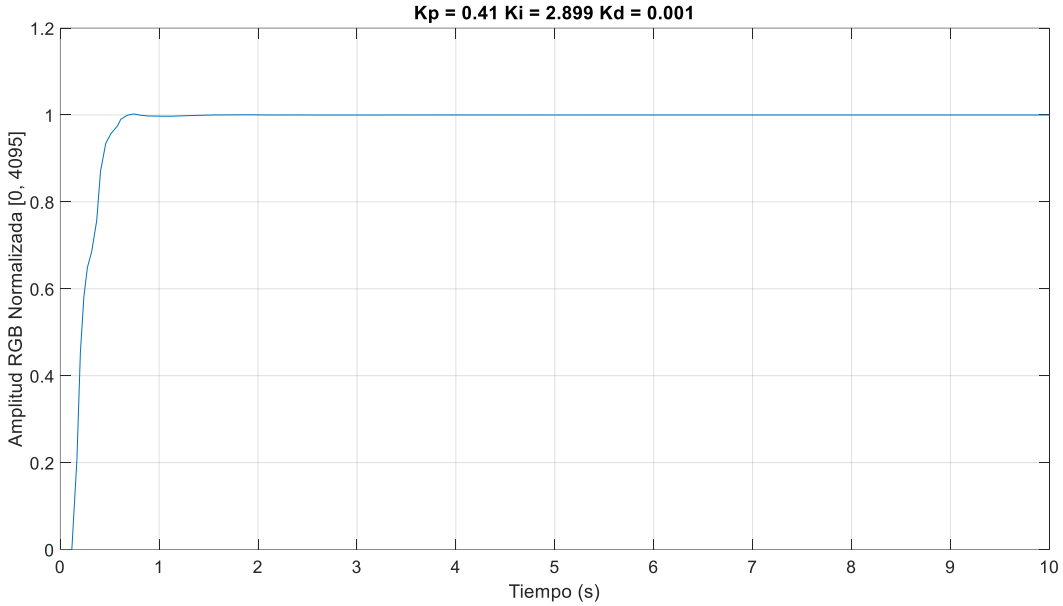


Ilustración 28. Respuesta dinámica recogida para un canal modelo discreto.

Parámetro	Valor
K_P	0.41
K_I	2.899
K_D	0.001
Tiempo de Subida (s)	0.6123
Energía señales de control	9.9941
Sobredisparo (%)	0.2298
Pendiente negativa	0.0017
Valor función objetivo	1.7598
Tiempo de Computo (s)	968.87

Tabla 5. Resumen de las características dinámicas del control PID para tiempo discreto.

4.5. Operación y Resultados del Controlador PID

Volviendo a la perspectiva multivariable de este problema, dado que la estrategia es controlar cada uno de los canales de forma independiente tras haber realizado el correspondiente desacoplo, la estructura de control entonces será DK_D. De hecho, como se vio anteriormente, la dinámica de los tres canales se puede aproximar por la misma respuesta, cambiando únicamente las ganancias en estado estacionario de cada uno. Es

por ello que, incluyendo como parámetros de realimentación el inverso de cada ganancia, se normalizan las salidas de los tres canales y se pueden estudiar estos como si fuera uno solo. En la Ilustración 29 puede verse esta construcción de forma más clara, dado el diagrama de bloques de Simulink. Este mismo modelo ha sido el empleado para la evaluación de las soluciones en el algoritmo de optimización heurística.

La respuesta dinámica del sistema completo con los parámetros de los controladores presentados en la Tabla 4 se muestra en la Ilustración 30. Puede verse en esta figura como los controladores actúan correctamente, devolviendo una respuesta adecuada. Además, la red de desacoplo asegura el aislamiento entre variables cuando cambian las consignas.

En tiempo discreto, la respuesta conjunta de los tres canales, de acuerdo con los parámetros presentados en la Tabla 5 se muestra en la Ilustración 31. Es directo comprobar que, en este caso, la respuesta empeora levemente, principalmente como consecuencia de la inexactitud en la red de desacoplo, ya que, para calcular el valor actual de desacoplo se emplea el valor de consigna anterior y no el actual como idealmente se realizaría en tiempo continuo.

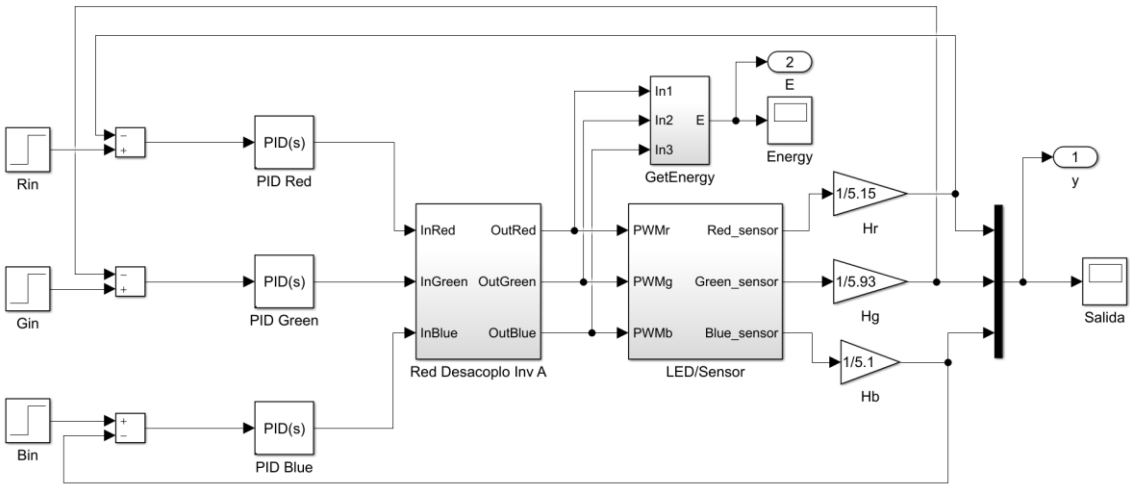


Ilustración 29. Sistema global en lazo cerrado.

Por último, cabe destacar como será la operación de este controlador PID cuando se implemente en el sistema real. El objetivo principal de este controlador es alcanzar valores de consigna de forma rápida, efectiva y estable. Sin embargo, es posible también que, tras haber establecido un valor de la lámpara concreto, este se desvíe debido al calentamiento de los LEDs, a la aparición de otra fuente de color posteriormente, o a cualquier otra razón. En este momento también sería interesante que el controlador modifique los valores de las señales que se aplican a los LEDs, de forma que el sistema se adapte a las nuevas circunstancias.

La Ilustración 32 muestra un posible ejemplo de control del sistema, donde, a la derecha aparece un color seleccionado en un círculo en negro, dentro del espacio de color *CIE 1931* [21]. A continuación, se activa el controlador PID. Tras sucesivas iteraciones,

mostradas con cruces blancas, el color actual de la lámpara medido por el sensor se vuelve acercar cada vez más al color deseado hasta que finalmente el error existente entre el mismo y el color de la luminaria es prácticamente despreciable.

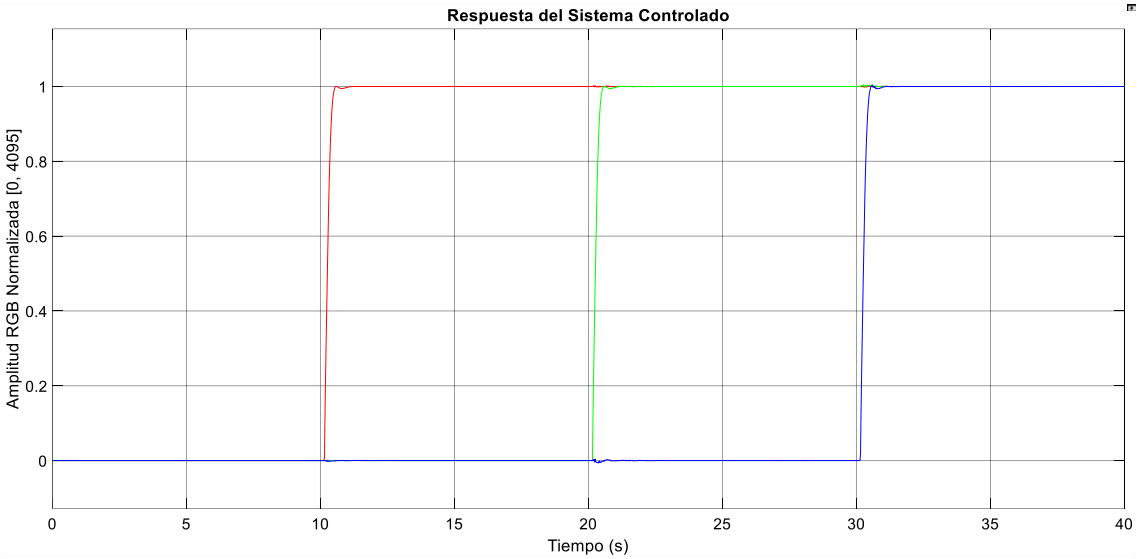


Ilustración 30. Respuesta dinámica del sistema en conjunto.

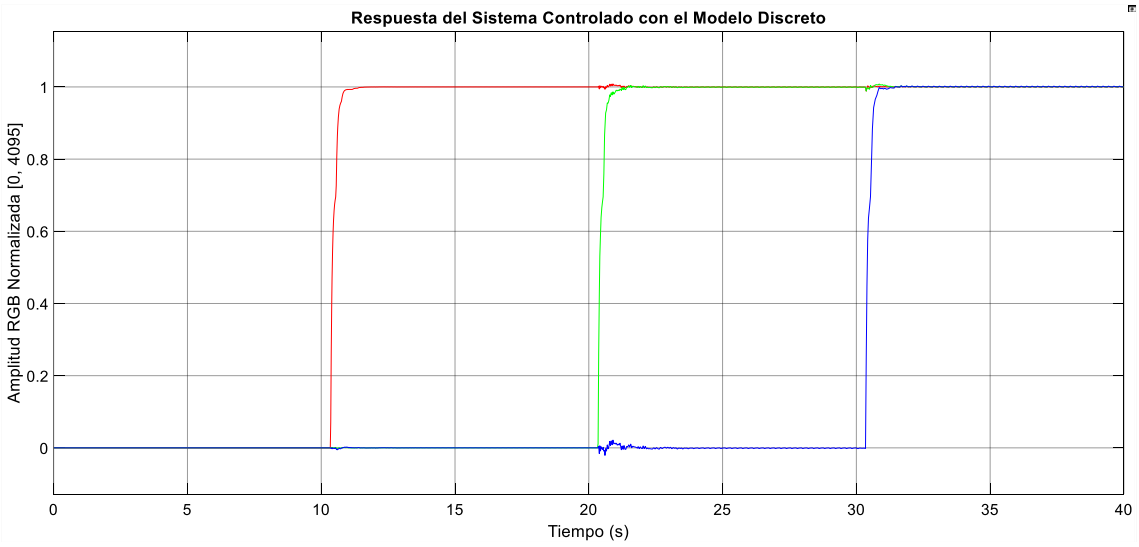


Ilustración 31. Respuesta dinámica global del sistema en tiempo discreto.

Con este algoritmo de optimización se ha conseguido sintonizar un controlador PID que permite modificar la consigna del color de la lámpara presentada en el recuadro azul celeste del diagrama de la Ilustración 4 de forma controlada. Además, en los siguientes capítulos se implementará el controlador DALI, que permitirá flexibilizar este control para un número de lámparas distinto, permitiendo articular una red de luminarias que puede dar vida a distintas aplicaciones.

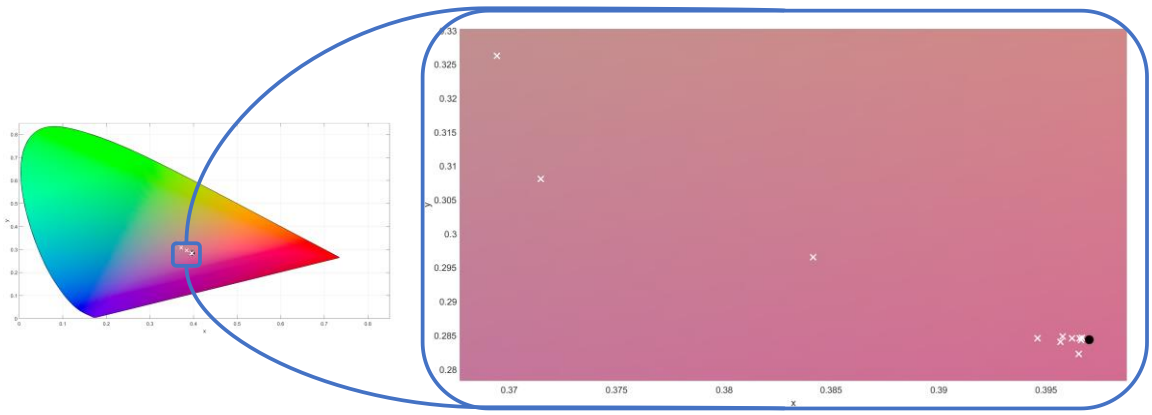


Ilustración 32. Modo de operación del sistema controlado.

Capítulo 5

Controlador DALI

En este capítulo se procede a desarrollar en detalle el proceso de implementación del controlador maestro DALI.

Es importante resaltar que el software desarrollado para este fin es adaptable a toda la familia de microcontroladores XMC1000 de Infineon Techonologies. En concreto, para este proyecto se ha escogido el modelo 1400 en su correspondiente placa de desarrollo *BootKit* [22]. Por otro lado, como interfaz hardware para DALI se ha empleado la placa "DALI PHY" [23] comercializada por este mismo fabricante.

Dado que la finalidad de este trabajo no es la exposición minuciosa del estándar de control DALI, a continuación, se irán presentando algunos de los detalles de interés, que han sido necesarios para la realización de este proyecto.

5.1. Transceptor DALI

El estándar DALI usa la codificación Manchester para la transmisión de mensajes, es decir, en lugar de definirse un bit por la diferencia de potencia existente entre dos hilos (generalmente 5 voltios significan un 1 y 0 voltios un 0), en la codificación Manchester son las transiciones las que definen los bits. En concreto, una transición de 0 a 5 voltios implica un 1 y viceversa para un 0. Un ejemplo de esta codificación es presentado en la Ilustración 12.

Para la gestión a nivel binario de la interfaz, se partió de la librería "DALIXVR" presente en el software "*DALI Stack*" de Infineon Technologies [24]. A continuación, se procede a presentar un poco más en detalle esta interfaz, tanto a nivel hardware como software.

5.1.1. Hardware del Transceptor DALI

Como se ha adelantado antes, la interfaz hardware de este transceptor se materializa en la placa "DALI PHY" [23]. La Ilustración 33 muestra el esquemático de esta interfaz.

Inicialmente se emplea un puente de diodos para eliminar la polaridad en el bus DALI.

Dado que el estándar DALI emplea la propia línea de alimentación, a 16 voltios, es necesario un desacoplamiento entre la parte lógica y el bus DALI. Es por ello que se

emplean optoacopladores (fotodiodo más foto transistor) para permitir la recepción y envío de bits de forma aislada.

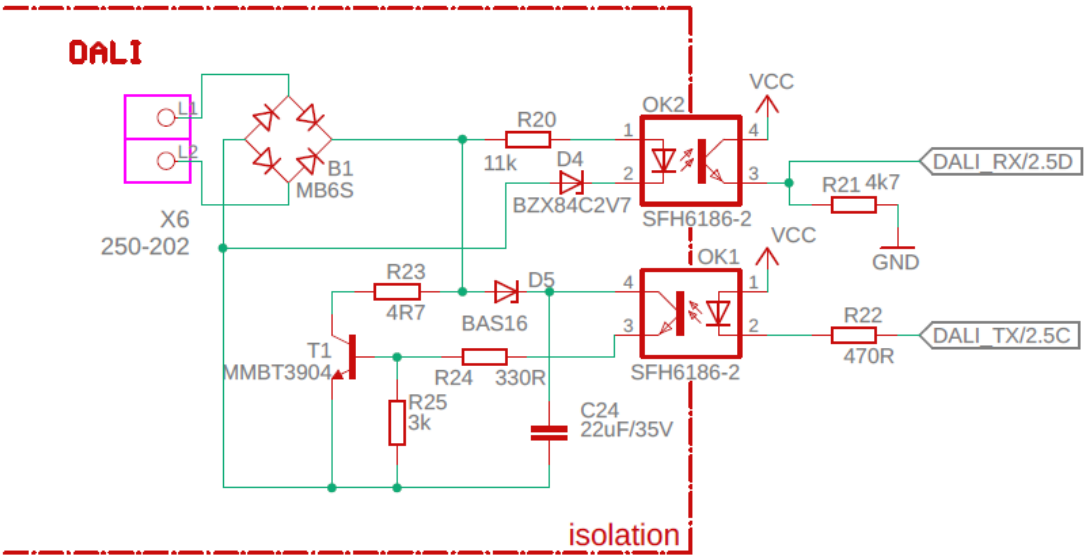


Ilustración 33. Layout de la interfaz física del transceptor de DALI.

La recepción de bits acontecerá cuando se hayan cortocircuitado las líneas del bus DALI, que inicialmente y en estado de reposo están a 16 voltios. Cuando la tensión entre los hilos cae, el fotodiodo del optoacoplador OK2 dejará de emitir y por lo tanto el fototransistor parará su conducción, estableciendo una alta impedancia entre la alimentación y el pin DALI_RX. Es decir, se habrá puesto un 0 en este pin.

Por otro lado, cuando se desea enviar bits a través de esta interfaz, se pondrá un 0 (0 voltios) en el pin DALI_TX. Esto implicará que el fotodiodo del optoacoplador OK1 comenzará a emitir, haciendo que circule una corriente entre el colector y el emisor del fototransistor. Esta corriente, debido al divisor de tensión formado en el emisor del fototransistor, generará una tensión en la base de transistor T1, haciendo que este entre en conducción y generando un camino de baja impedancia entre las líneas del bus DALI.

Es importante remarcar que, cuando se desea escribir un cero en el bus DALI, en realidad lo que se está haciendo es poner una tensión prácticamente igual a 16 voltios entre los polos de un diodo, con una baja resistencia. Como se sabe, la curva característica de los diodos es una exponencial, es decir que los diodos estarían exigiendo una corriente excesivamente elevada. Es por ello necesario aumentar la resistencia para limitar dicha corriente. Este es el motivo por el que se incluyen dos resistencias de 100 ohmios (Ω) en la alimentación del bus DALI. La Ilustración 34 muestra el montaje en baquelita de las dos resistencias para dicho propósito en la fuente de alimentación seleccionada.

5.1.2. Software del Transceptor DALI

En cuanto a la parte software de este transceptor, únicamente será necesario un bloque que gestione los eventos en el microcontrolador, dos pines y un temporizador. La idea básica entonces es registrar cuando se produce una transición en el bus DALI, para registrar el evento y llamar a una función (*callback*) para que ejecute la máquina de estados que gestiona tanto la transmisión como la recepción de mensajes, según el estándar, a través de los pines DALI_TX y DALI_RX. Por su parte, el temporizador se emplea para gestionar la temporización entre los sucesivos bits.

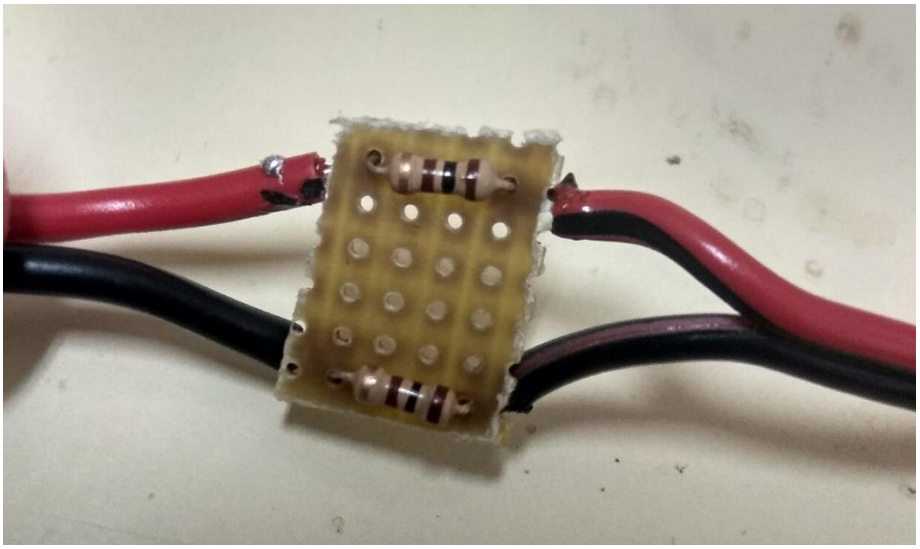


Ilustración 34. Montaje en baquelita de las dos resistencias para la limitación de corriente en la alimentación.

5.2. Direccionamiento de Esclavos

El proceso de direccionamiento de los esclavos (o *control gears*) mediante el estándar DALI puede verse, de forma muy resumida, en la Ilustración 35. La idea básica para el direccionamiento inicial de los esclavos es realizar una búsqueda binaría sobre unas direcciones aleatorias generadas por los propios esclavos para encontrar aquel que tenga la dirección más baja, para posteriormente asignarle una dirección corta y sacarlo del proceso de búsqueda. A continuación, se procede a estudiar un poco más en detalle cada uno de los subprocesos presentados en esta ilustración.

5.2.1. Reconocimiento por Dirección Aleatorio

Inicialmente, se realiza un reconocimiento de los esclavos que ya poseen una dirección corta. Para ello, se emplean las direcciones aleatorias de 3 bytes que los mismos deberán tener almacenadas, haciendo entonces que el proceso se divida en tres pasos. La idea es interrogar a cada dirección corta por su dirección aleatoria, a fin de obtener respuesta y concluir con la existencia de un esclavo ya direccionado en el bus DALI. Este

proceso se realizará para las 64 direcciones cortas que pueden ser direccionadas en el bus DALI. El diagrama de flujo del proceso se presenta en la Ilustración 36.

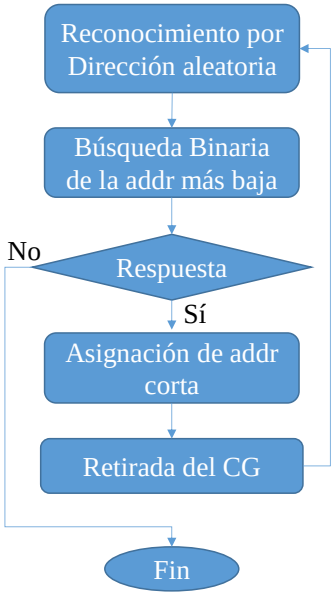


Ilustración 35. Proceso de direccionamiento en protocolo DALI.

5.2.2. Búsqueda Binaria

A continuación, tras el proceso de reconocimiento inicial, comienza el direccionamiento como tal. Para ello, se emplean direcciones aleatorias de 3 bytes (24 bits) generados por el propio esclavo de forma aleatoria, usando como semilla el número de identificación del chip XMC (único y distinto para cada microcontrolador). Dado que este está formado por tres bytes, la búsqueda binaria de la dirección aleatoria se realiza en tres pasos, buscando primero el byte más alto, después el medio y por último el bajo.

Por lo tanto, el primer paso es enviar una petición de generación aleatoria de direcciones. Tras esto, comienza el bucle de direccionamiento principal. En él, el primer paso es hacer un reconocimiento de los esclavos que aún no están direccionados. A continuación, se interroga a estos por una determinada dirección. Si se recibe respuesta en el bus, significa que al menos un esclavo posee una dirección (o parte de la dirección aleatoria) que es más baja que la dirección buscada, por lo tanto, en consecución con la búsqueda binaria, se exigirá que la siguiente dirección a buscar esté en la mitad superior de la tabla (direcciones más bajas) y se vuelve a realizar la búsqueda. En caso de que, tras la interrogación por la dirección buscada, no se reciba respuesta por parte de los esclavos, significará que la dirección buscada es superior a las direcciones aleatorias de los esclavos, así que la siguiente dirección a buscar será en la mitad inferior de la tabla (direcciones más altas).

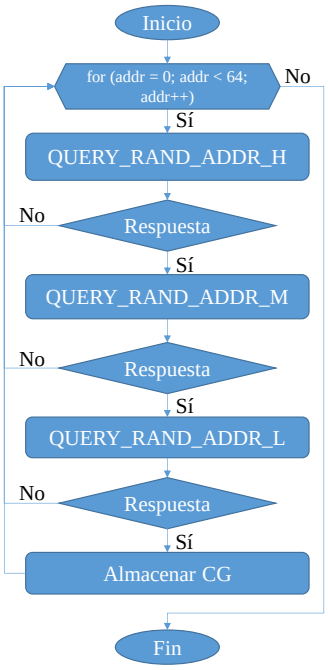


Ilustración 36. Reconocimiento de esclavos en el bus DALI.

Este proceso continúa, acotando el rango de la dirección menor, hasta que en dicho rango solo existe una dirección (la diferencia entre el límite superior del rango y el inferior es uno, es decir, entre A_{mayor} y A_{menor} en la Ilustración 37), es decir, la dirección que se está buscando es la misma que la dirección del esclavo en cuestión, por lo tanto, el siguiente paso será asignarle a este esclavo una dirección corta y eliminarlo del ciclo de direccionamiento, para que en las posteriores iteraciones no conteste a las interrogaciones. Así finaliza una iteración y comienza nuevamente la siguiente.

El proceso se repite hasta que ningún esclavo conteste en el chequeo inicial, momento en el que se concluye que todos los esclavos están direccionados, o, en su caso, que no existen esclavos en el bus DALI. La Ilustración 37 muestra el diagrama de flujo de todo este proceso.

5.3. Búsqueda de Esclavos

Otro de los procesos del estándar DALI implementados es la búsqueda y reconocimiento de dispositivos de control. Es posible que exista ya algún dispositivo con la dirección corta asignada de un direccionamiento anterior, almacenada en una memoria no volátil. En el caso de que todos los dispositivos presentes en el bus DALI hayan ya sido direccionados previamente, existe un proceso para la búsqueda y registro de estos esclavos, más ligero que el direccionamiento en sí, y que es igualmente válido para que la interfaz DALI y el propio maestro DALI puedan escanear el bus DALI, y tener un registro de los dispositivos conectados.

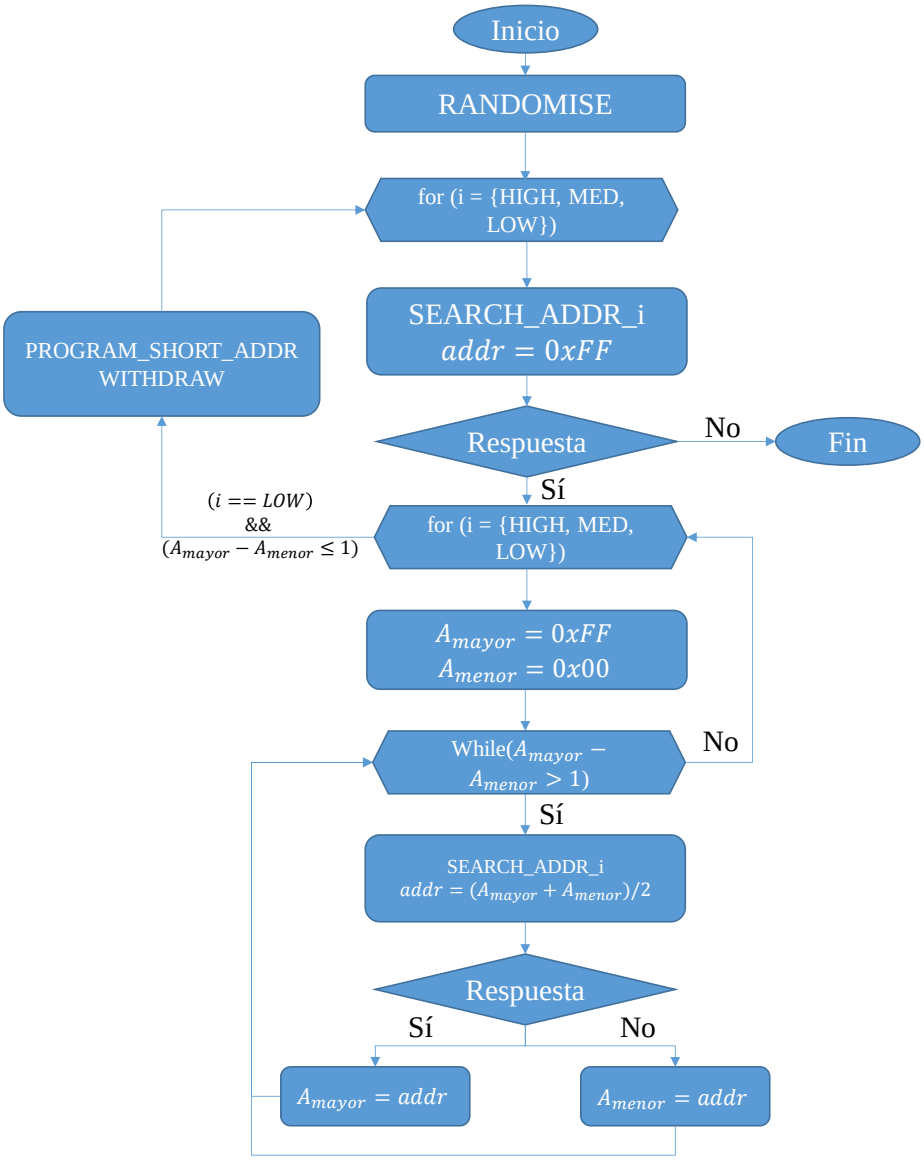


Ilustración 37. Diagrama de flujo de la búsqueda binaria y direccionamiento de esclavos.

Este proceso es muy sencillo y se esquematiza en la Ilustración 38. Como puede observarse, la idea básica es preguntar a cada una de las 64 direcciones cortas en el bus DALI por el estado real de la lámpara. Si algún dispositivo contesta, se demuestra así su existencia y se registra el mismo. A continuación, se pregunta si este pertenece a algún grupo, inicialmente a los grupos de 0 a 7, y posteriormente a los grupos de 8 a 15. Si en algún caso, el dispositivo contesta afirmativamente, se registran estos grupos [25].

5.4. Comandos DALI

Por último, queda hablar de los comandos DALI. Una vez direccionados los esclavos conectados al bus DALI, es posible enviarles comandos específicos de control

de iluminación a cada esclavo concreto. En este caso, el proceso es trivial, ya que, en la mayoría de los casos, el proceso se limita a enviar el comando específico y la dirección deseada.

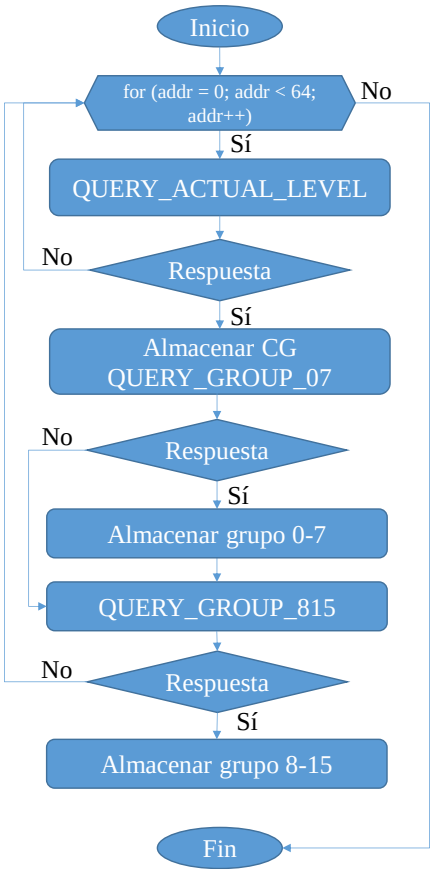


Ilustración 38. Proceso de búsqueda de dispositivos [25].

Cabe destacar el proceso para configurar un nivel de intensidad concreto en una de los esclavos. Para ello es necesario enviar dos comandos, el primero que habilita el control del esclavo en cuestión. En el segundo se debe mandar el nivel de intensidad deseado. Este proceso será el empleado para el control PID de la lámpara RGB.

5.5. Desarrollo de los Esclavos DALI

Por último, para este proyecto se ha desarrollado un firmware adaptado al microcontrolador XMC1300 de Infineon Technologies que gestiona el control de tres canales distintos como tres esclavos independientes. Si bien el dispositivo maestro de la comunicación DALI se ha desarrollado y depurado con el firmware facilitado en la página web de Infineon [26], para el control PID del cromatismo de la lámpara a través del valor de intensidad de tres canales RGB se desarrolló un firmware muy sencillo que emula el comportamiento de tres esclavos en lo que respecta al direccionamiento y búsqueda de esclavos, y gestiona la intensidad de cada canal.

En este firmware, al igual que en el maestro, se incluye la librería del transductor DALI para la gestión de la interfaz física. A continuación, se crean tres sistemas de control para los drivers que controlarán cada uno de los canales de luz LED.

Por último, se va chequeando el registro de gestión de eventos del transductor, que cambiará de estado cuando se reciba un nuevo comando. Llegado el caso, se procesa dicho comando, y se ejecuta la correspondiente orden de control en cada uno de los canales.

5.6. Resultados de la Implementación del Maestro DALI

Finalmente, se muestra la Ilustración 39 donde se transmite el mensaje 0xAA55. Como puede verse, la trama comienza en con un uno, siendo este el bit de inicio propio del protocolo DALI. Posteriormente acontecen las transiciones para ir generando los bits de la trama en los flancos de subida (para los 1) y de bajada (para los 0), de acuerdo con la codificación Manchester. Puede verse, además, como la trama ocupa un intervalo de tiempo aproximadamente igual a los 14 milisegundos, casando con la velocidad de transmisión igual a 1200 bits por segundo propia del estándar DALI.

Otro ejemplo de tramas puede observarse en la Ilustración 40, donde el maestro envía el comando QUERY_MISSING_SHORT_ADDRESS (0x96) a todos los posibles esclavos anclados al bus DALI (*BROADCAST*, igual a la dirección 0xFF), y se recibe una respuesta de los esclavos igual a 0xFF. Esto acontece dos veces.

A continuación, se proporciona el enlace a un vídeo donde puede verse como el maestro inicia el proceso de búsqueda de esclavos en el bus DALI, posteriormente realiza un direccionamiento y, a continuación, envía distintas órdenes de control a los tres esclavos encontrados, que ejecutan dichas órdenes. Además, en paralelo, el maestro va comunicándose con la interfaz gráfica, y esta va actualizando sus objetos. El vídeo puede encontrarse aquí [27]. En este caso, el dispositivo esclavo, tiene cargado el firmware proporcionado por Infineon Technologies en su página web [24].

Tras esta implementación del maestro, se consigue el control de hasta 64 esclavos, a través del estándar DALI, que engloba el direccionamiento, la búsqueda de esclavos en el bus y los distintos comandos de control. De esta forma se permite la articulación de una red de luminarias esclavas, que pueden actuar como una fuente de color, que puede ser controlada a través del PID implementado en el capítulo previo.

Aunque se permitir la articulación de una red de luminarias, sin embargo, el funcionamiento de una red sale del ámbito del proyecto y el análisis de los resultados en los capítulos siguientes se centrara en uno de los nodos de la red.

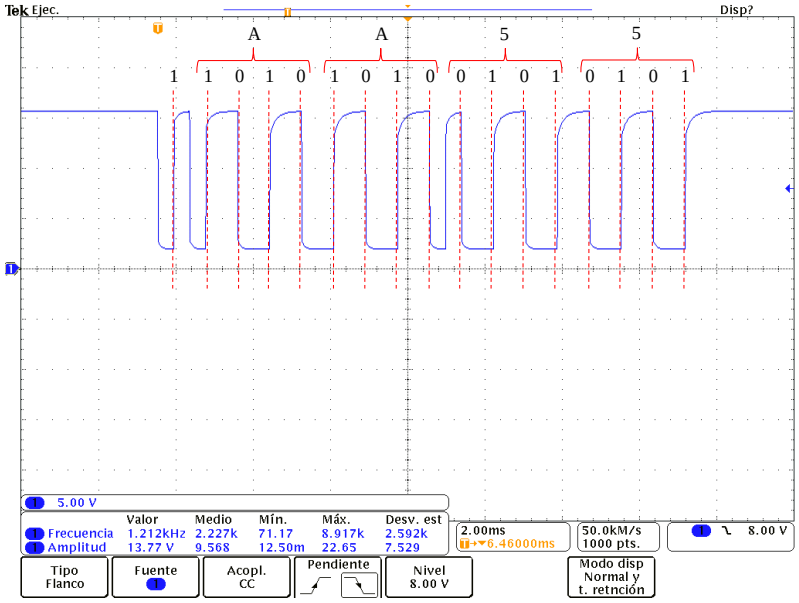


Ilustración 39. Transmisión a nivel de bits del mensaje 0xAA55.

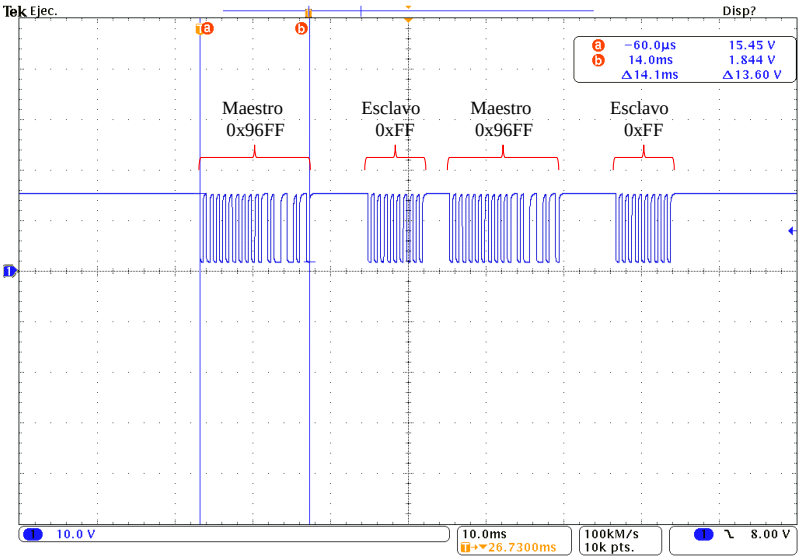


Ilustración 40. Tramas a nivel de bits entre el maestro y esclavo.

Capítulo 6

Diseño y Desarrollo de la Interfaz DALI

En este capítulo, se estudia y detalla el diseño y desarrollo de la interfaz implementada para el controlador DALI. Inicialmente se describe el protocolo de comunicación entre el maestro DALI y el ordenador. A continuación, se detalla la interfaz serie desarrollada. Posteriormente y como último paso, se desarrolla la interfaz gráfica para el usuario.

6.1. Protocolo de Comunicación PC – Maestro

Para la comunicación entre el ordenador y el maestro DALI se ha desarrollado un pequeño protocolo basado en la interfaz física de UART. El objetivo de este protocolo, aparte de garantizar la comunicación entre los dos dispositivos mencionados, es la detección y corrección de errores, y la comprobación de la conexión del maestro. A continuación, se desarrolla el protocolo implementado, inicialmente desde un enfoque físico para después estudiar en detalle su concepción software.

6.1.1. Interfaz física

Como se ha adelantado previamente, para la implementación física de este protocolo se ha empleado la interfaz UART, o comúnmente conocida como puerto serie. La comunicación UART es una implementación alámbrica basada en dos hilos TX y RX, intercambiados entre el transmisor y el receptor (el cable TX del primer dispositivo va al RX del segundo y viceversa). La interfaz usa codificación binaria y una frecuencia de transmisión que puede ser reconfigurable, pero exige que todos los dispositivos a comunicar estén en sincronía, es decir, operen a la misma frecuencia.

La comunicación UART divide los mensajes a transmitir en bytes que se irán enviando de forma secuencial a través del hilo de transmisión. Por su parte, el dispositivo receptor recibirá el mensaje fraccionado por el pin de recepción y volverá a ensamblar el mensaje para su posterior gestión. En general, los procesos de transmisión y recepción se realizan mediante la gestión de interrupciones.

Entre los parámetros que se pueden configurar en la comunicación UART destacan la frecuencia de transmisión, el bit de paridad, el bit de comienzo y final de byte o el modo de operación (*full-duplex* cuando los dos dispositivos van a enviar y recibir por líneas separadas o *half-duplex* cuando los dispositivos solo emplean una línea para enviar y recibir). Todos estos parámetros deben ser igualmente configurados en los dispositivos a comunicar para el correcto intercambio de información.

En la Ilustración 41 puede observarse un ejemplo de trama enviado por UART.

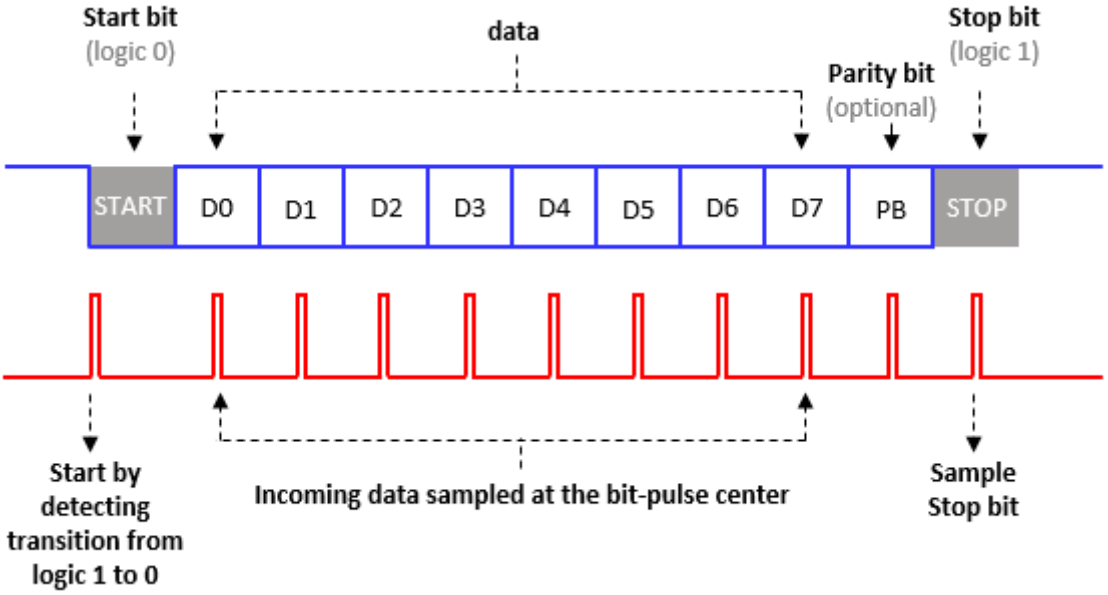


Ilustración 41. Ejemplo de trama transmitida por UART [8].

Como puede observarse, se envía un bit de inicio en la transición de un 1 lógico a un 0. A continuación, en cada pulso de reloj, se envía un bit (por ello es crucial que los dispositivos a comunicar estén configurados con la misma frecuencia). Posteriormente es posible enviar un bit de paridad para la detección de errores. Por último, se envía uno o dos bits de parada para indicar que se finaliza el byte.

En concreto, para la comunicación entre el maestro y el ordenador, se ha empleado la configuración mostrada en la Tabla 6:

Parámetro	Valor
Modo de Operación	*Full-Duplex*
Frecuencia de Transmisión	9600 baudios
Longitud de datos	8 bits
Bit de Parada	1 bit de parada
Paridad	*Off*

Tabla 6. Configuración de la interfaz UART para el protocolo de comunicación entre el PC y el maestro DALI.

6.1.2. Desarrollo Software del protocolo

Se concibe este protocolo desde una perspectiva bidireccional. Por un lado, el controlador DALI envía al ordenador los comandos propios del estado del proceso, datos de las luminarias, datos del bus etc; y este, tras interpretarlos, muestra en pantalla la información referida en dicho comando. Por otro lado, el ordenador enviará al controlador DALI las funciones o configuración que el usuario desea para el esclavo en cuestión.

Cada una de las tramas constará de dos bytes de un información, más un byte de redundancia cíclica, CRC-8, codificado con el polinomio 0x07 ($x^2 + x + 1$). Con este último byte de redundancia únicamente se pretende garantizar que la información ha sido correctamente recibida por el dispositivo en cuestión. La idea básica es dividir el mensaje a transmitir entre el polinomio (empleando para ello la operación XOR) y enviar el resto en un nuevo byte para que el receptor repita la operación y compruebe que el resto de la operación en efecto es el mismo que el recibido en el byte de CRC. Tras cada recepción, si el mensaje ha sido correctamente recibido, se envía un reconocimiento al emisor (ya sea el maestro DALI o el PC).

En la Ilustración 43 se muestra el diagrama de flujo para el correcto establecimiento de la operación del protocolo en la interfaz. Básicamente se persigue comprobar que el dispositivo a comunicar es un maestro DALI, y posteriormente, la interfaz se pone en espera. Si se realiza una petición de transmisión, se envía el comando desde la interfaz y se recibe la confirmación. En caso de recibir una "no confirmación" (NACK) por parte del maestro DALI, se reenvía el comando. Por otro lado, si la interfaz recibe un comando se comprueba la correcta recepción del mismo y se actúa en consecuencia, de forma análoga a como se ha explicado en la transmisión.

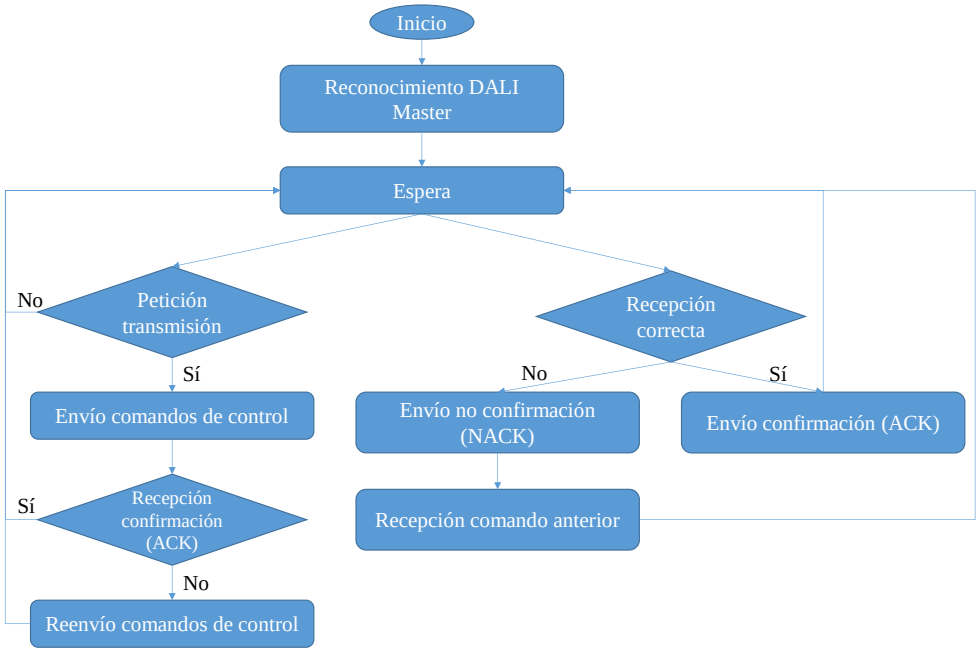


Ilustración 42. Diagrama de flujo del protocolo entre el ordenador y el maestro DALI.

Para reconocer que existe un dispositivo serie conectado a los terminales serie del PC, y que este es un maestro DALI, la interfaz en el PC inicialmente comprueba si hay puertos serie disponibles, y los lista. Posteriormente se pide seleccionar el puerto deseado. Seguidamente, se interroga al maestro por la versión de firmware que este tiene (a modo de *ping*). A continuación, si esta es 51 o 60 (versión 5.1 o versión 6.0), se supone que el microcontrolador conectado a través del puerto serie es un maestro DALI y se verifica la conexión.

Para enviar una trama desde el controlador DALI hasta el ordenador se emplearán 2 bytes, siguiendo el propio protocolo DALI. El primero hace referencia a la dirección del esclavo al que va dirigido el comando, el segundo al comando en sí. Para referenciar un esclavo concreto al que previamente ya se le ha asignado una dirección corta, está se codificará de acuerdo con la ecuación 2.

De tal forma que la dirección corta 0x00 será referenciada con 0x01, la 0x01 se referenciará con la 0x03, la 0x02 con la 0x05 etc. Además, como se ha dicho anteriormente, se suma un último byte de redundancia cíclica a cada trama, formándose así paquetes de 3 bytes.

Finalmente, cuando el maestro DALI recibe un mensaje por parte de la interfaz (o viceversa), se envía un nuevo paquete de 3 bytes con una confirmación de recepción (ACK). En la Ilustración 43 se muestra el diagrama de flujo datos del protocolo.

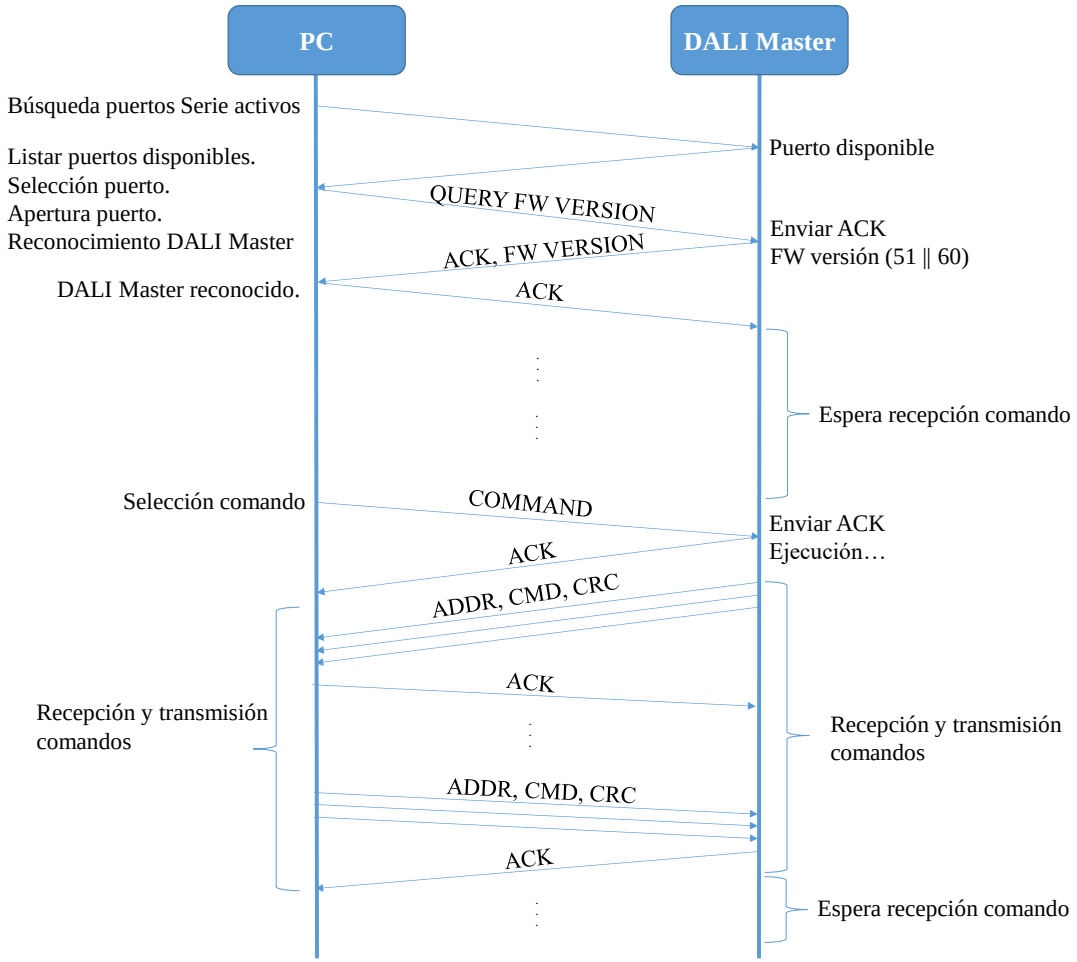


Ilustración 43. Flujo de datos en el protocolo entre el ordenador y el maestro DALI.

Por último, cabe destacar los distintos tipos de mensajes que se intercambiarán entre la interfaz DALI y el maestro.

En concreto, el controlador DALI le enviará a la interfaz los siguientes tipos de mensajes, de acuerdo al comando que estos incluyen:

- **Comandos Estándar:** El controlador le enviará a la interfaz los mismos comandos que envía al esclavo en la comunicación DALI con el mismo formato (dirección del esclavo en el primer byte y comando en el segundo), pero incluyendo el byte de redundancia cíclica.

- **Comandos Especiales:** Nuevamente se repite el mismo formato de envió que en el protocolo DALI. Estos comandos especiales, se reconocen por tener los cuatro primeros bits iguales a 1010 (0xA) o 1011 (0xB). Además, al igual que en el protocolo DALI, en este caso el primer byte hace referencia al comando, y el segundo a la dirección.

- **Respuesta de Esclavos:** Otro detalle de interés que se desea visualizar en la interfaz son las respuestas que los esclavos DALI dan a los comandos enviados por el maestro. Para ello, cuando se reciben estas respuestas en el maestro, estas se envían a la interfaz DALI con los cuatro primeros bits iguales a 1000 (0x8).

- **Medida del Color:** Por último, el maestro DALI deberá enviar la lectura tomada del sensor. Para ello se enviará un primer byte con los tres primeros bits iguales a 1001 (0x9). Posteriormente se enviarán los tres bytes correspondientes a el nivel de intensidad de cada color.

Más ejemplos de mensajes enviados desde el maestro DALI a la interfaz pueden encontrarse en la Tabla 7.

Comando	Tipo	Dirección	Trama
Query missing short address	Estándar	Broadcast	0xFF963C
Query Random Address (H)	Estándar	0x0B	0x0BC2D7
Compare	Especial	--	0xA900A5
Search Address (M)	Especial	0xEB	0xB3EBEF
Program Short Address	Especial	0x00	0xB70123
Withdraw	Especial	--	0xAB008F
Respuesta “00”	Respuesta	--	0x8000B6

Tabla 7. Ejemplos de tramas entre el maestro DALI y la interfaz.

Por otro lado, la interfaz gráfica enviará también peticiones al maestro DALI. En este caso, dado que el abanico de peticiones que la interfaz puede enviarle al maestro es mucho más reducido, se distinguen únicamente dos tipos distintos de comandos:

- **Comandos Estándar:** Al igual que en el caso anterior, se envían comandos de control al maestro DALI desde la interfaz, en el mismo formato que en el protocolo DALI (añadiendo el byte de CRC).

- **Procesos:** La interfaz le enviará la petición de ejecución de procesos (búsqueda de esclavos, direccionamiento etc) mediante la máscara 1000 (0x8) en los cuatro primeros bits. Así, la búsqueda de esclavos será la trama 0x80, el direccionamiento la trama 0x81, la petición del firmware 0x82, la petición de borrado de direcciones 0x83, la petición del control PID 0x84 y la lectura del sensor RGB 0x85.

Más ejemplos de mensajes que la interfaz DALI envía al maestro se pueden ver en la Tabla 8.

Comando	Tipo	Dirección	Trama
Recall Max Level	Estándar	0x00	0x01050E
Step Up	Estándar	0x01	0x030336
Step Down & Off	Estándar	0x02	0x050754
Down	Estándar	0x00	0x01021B
Search Gears	Proceso	--	0x008089
Addressing	Proceso	--	0x00818E
FW Version	Proceso	--	0x008287
Delete Address	Proceso	Broadcast	0xFF8345
PID Control	Proceso	--	0x008495
Read Sensor	Proceso	--	0x008592

Tabla 8. Ejemplos de trama entre la interfaz y el maestro DALI.

6.2. Interfaz Gráfica

A continuación, una vez detallado el protocolo que se emplea para la comunicación, se procede a mostrar la interfaz gráfica desarrollada, partiendo de la interfaz serie que inicialmente se implementó. Por último, se repasan algunos de los detalles más importantes de su programación.

6.2.1. Interfaz Serie

Inicialmente, para la programación y depuración del protocolo, así como del maestro DALI, se desarrolló una interfaz serie utilizando el lenguaje de programación *Python*. Esta interfaz mostraba mensajes en el terminal serie del IDE de *Python*. Algunas capturas tomadas de esta interfaz serie se muestran en la Ilustración 44.

Esta interfaz lanza un menú de opciones y permite la selección de cada una de estas a través de la inclusión de su correspondiente número identificativo. Sin embargo, esta interfaz ya reconocía y clasificaba los comandos recibidos del maestro DALI, y sirvió de base para el posterior desarrollo de la interfaz gráfica.

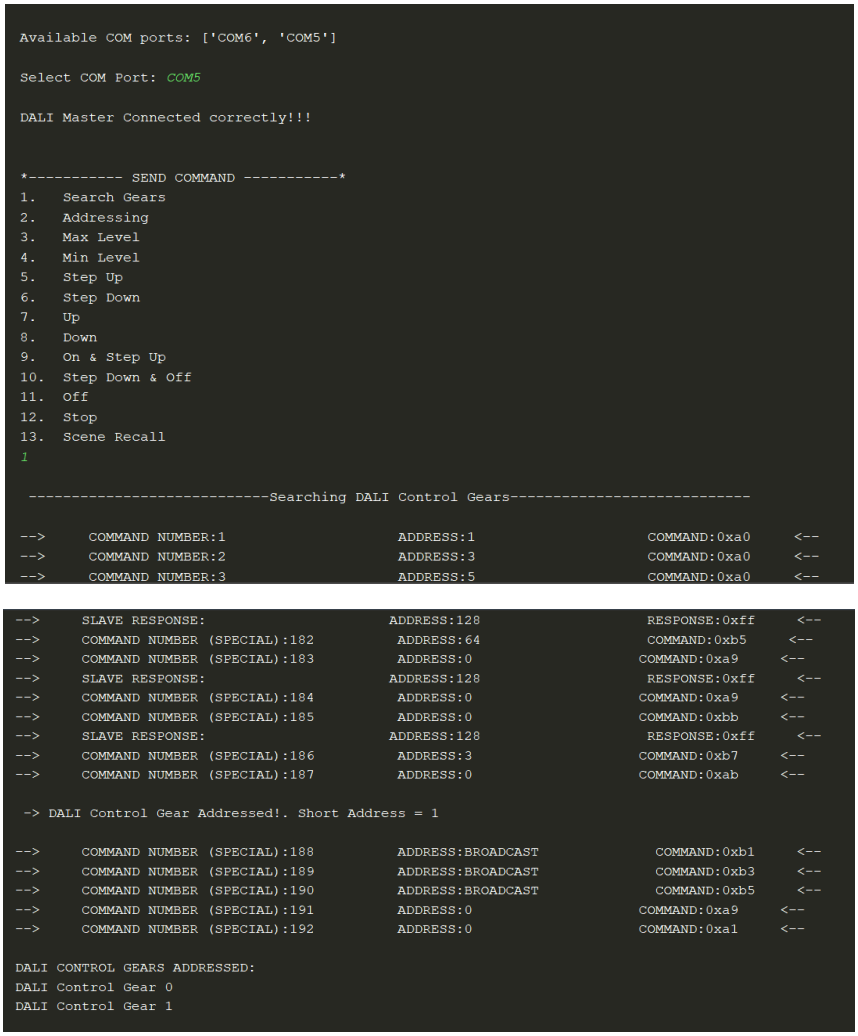


Ilustración 44. Capturas de la interfaz serie para el ordenador.

6.2.2. Interfaz Gráfica

El siguiente y último paso para dar por terminado el diseño del controlador DALI fue el desarrollo de una interfaz gráfica. Para ello, se empleó el software de diseño *Qt* [28], que permite la programación mediante objetos en C++. Sin embargo, *Python* tiene su propia librería para la programación de interfaces desarrolladas en *Qt*, llamada *PyQt* [29], libre y gratuita. De esta forma, únicamente fue necesario el entorno de diseño de interfaces *Qt Designer* también facilitado en la instalación de *PyQt*.

En la Ilustración 45 se muestran algunas capturas de la interfaz gráfica desarrollada. Esta cuenta con distintos objetos (o *widgets*) que han sido programados a través de eventos y funciones (*signals & slots*), propias en *PyQt*.

- **Cuadros de Disposición:** Sirven para ordenar el espacio en la aplicación y redimensionar los objetos que hay en su interior, cuando el tamaño de la

ventana de la aplicación cambia. Además, también existen otros objetos incluidos en la interfaz para mejorar su visualización, como los espaciadores verticales y horizontales.

- **Botones:** Estos objetos se incluyen para realizar una función determinada cuando se pulsan. En concreto, generan un evento que llama a la función (*Callbacks*) a la que están asociado. Aparecen con el número 1 en la Ilustración 45.

- **Etiquetas:** Estas tienen una utilidad meramente informativa. Cabe destacar que existen algunas etiquetas cuyo mensaje puede ser reconfigurado durante la ejecución de la interfaz, como por ejemplo la marcada en 2 en la Ilustración 45. Otras en cambio, mantienen su mensaje durante toda la ejecución.

- **Recuadro de Selección:** En este recuadro se muestran los puertos serie del PC que están disponibles, a fin de facilitar la selección de aquel que tiene un maestro DALI. Aparece con el número 3 en la Ilustración 45.

- **Tabla de Mensajes:** En esta tabla se muestran los comandos y las direcciones enviadas y recibidas con el maestro DALI, además de las respuestas de los esclavos DALI. Aparece con el número 4 en la Ilustración 45.

- **Cuadros de Texto:** En estos cuadros de texto se permite insertar caracteres que serán recogidos por el programa de control de la interfaz. Por ejemplo, el valor de consigna del control PID. Véase el recuadro 5 en la Ilustración 45.

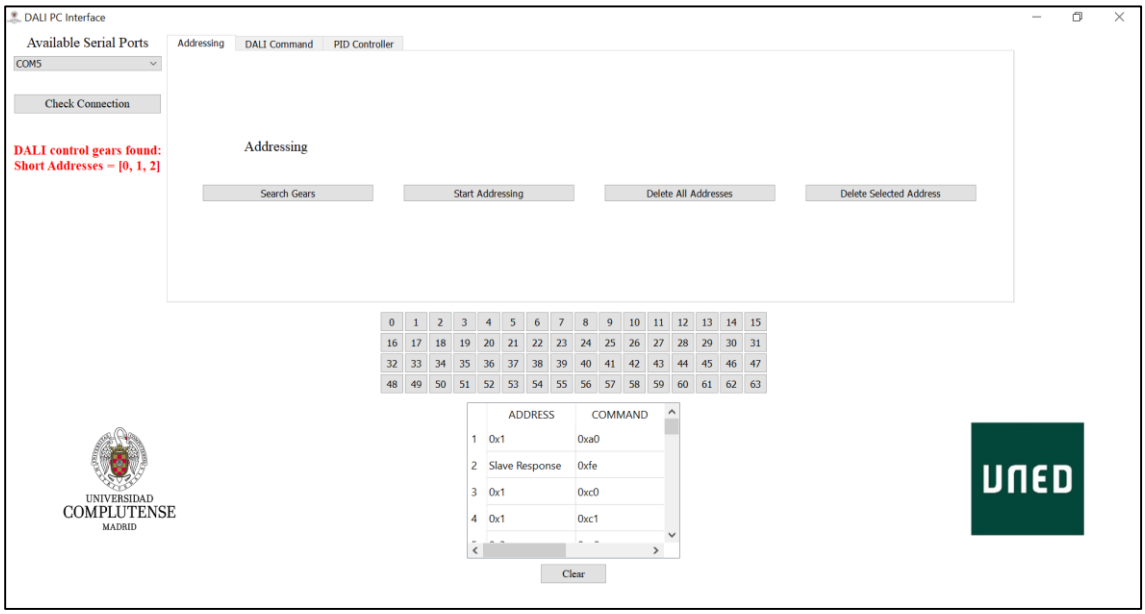
- **Imágenes:** También se han incluido imágenes en la interfaz, con fin meramente embellecedor. En el número 6 de la Ilustración 45 puede verse un ejemplo

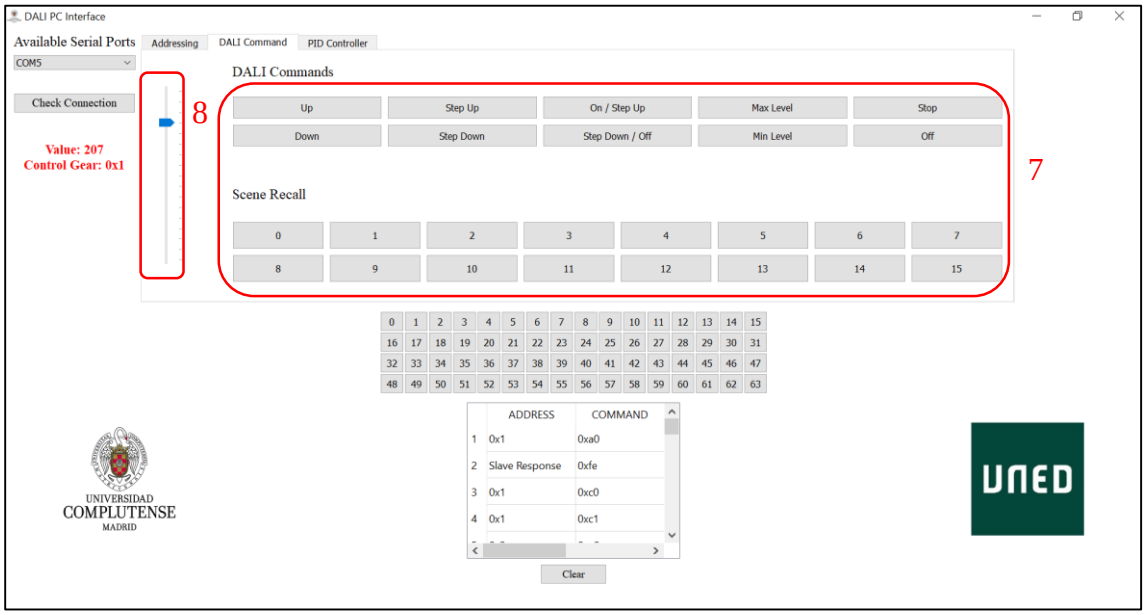
- **Campo de Paneles:** Este objeto se incluyó con la idea de insertar distintos paneles de control, de acuerdo con la función específica que se realiza en cada uno de ellos. Así pueden verse tres paneles, el primero relativo a las funciones de direccionamiento, el segundo, relativo a comandos DALI y el tercero y último, con la opción del control PID y la lectura del sensor RGB.

- **Deslizador Vertical:** Este objeto se incluyó para poder modificar el valor de intensidad de cada canal concreto de forma más sencilla y eficaz. Esta escalado entre 0 y 255. Puede verse con el número 8 en la Ilustración 45

- **Comunicación Serie:** Este objeto, al igual que los cuadros de disposición, no es visible en la Ilustración 45, aunque es vital para la correcta operación de la interfaz. Es el encargado de gestionar toda la comunicación a través del terminal UART del ordenador, usando el protocolo descrito anteriormente.

Cabe destacar que la interfaz y el firmware del maestro DALI están en sincronía, es decir, la interfaz transmite mensajes cuando el maestro DALI está en espera de recepción y viceversa. Además, ambos llevan la cuenta del estado del bus DALI, así como de los esclavos. Esto ha sido especialmente dificultoso en los procesos, donde el maestro DALI envía mensajes a la interfaz que muestran el estado del proceso. Finalmente se ha resuelto este problema de sincronía implementando máquinas de estados que materializan los diagramas de flujo presentados en las ilustraciones de este capítulo y el anterior. La idea básica es reconocer que mensajes se reciben del maestro DALI, y cambiar el estado actual, hasta finalmente acabar el proceso.





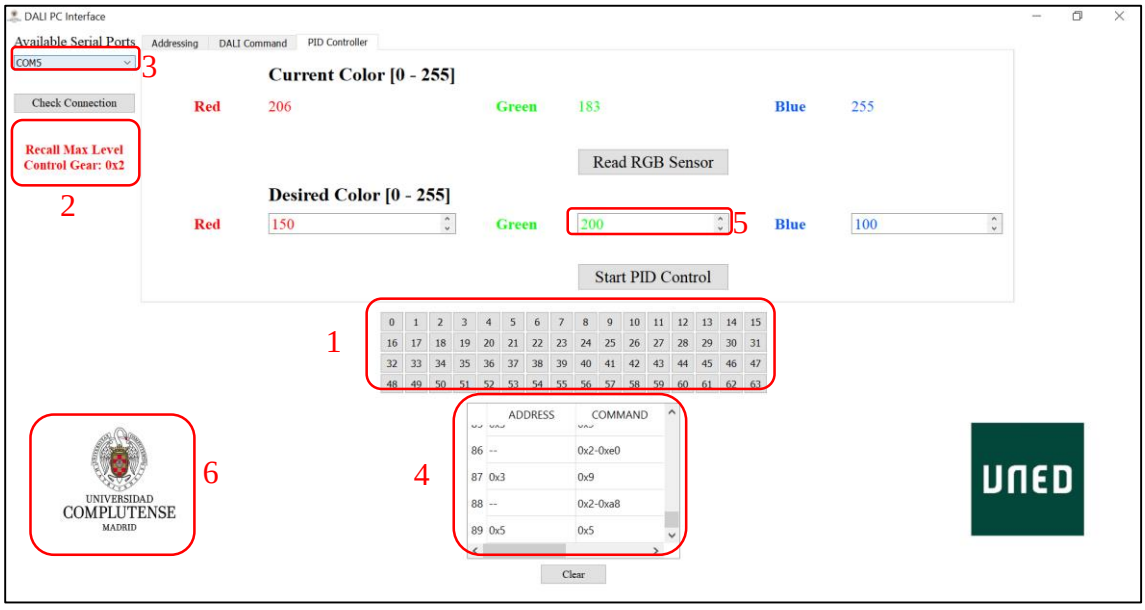


Ilustración 45. Interfaz gráfica para el controlador DALI.

Finalmente, a fin de facilitar la ejecución de la interfaz, se empleó la librería "*PyInstaller*" para generar un ejecutable *.exe* para correr la misma en Windows, sin necesidad de tener preinstalado el intérprete de Python ni ninguna de sus librerías. Además, las imágenes de la interfaz se codificaron en base64 para incluirlas en el propio archivo sin necesidad de tenerlas en la carpeta. El archivo resultante se muestra en la Ilustración 46.

Ilustración 46. Archivo resultante de la interfaz gráfica.

Gracias a la implementación del protocolo y la interfaz gráfica se consigue facilitar la comunicación con el maestro DALI, permitiendo así su utilización por parte de cualquier usuario. Además, esta interfaz gráfica sirve también para identificar en qué estado se encuentra la comunicación DALI y los dispositivos que en ella intervienen. Queda así concluido el bloque amarillo relativo al maestro DALI, presentado en el diagrama de bloques de la Ilustración 4, así como el bloque del dibujo del ordenador, que se corresponde a la interfaz gráfica desarrollada, y el protocolo de comunicación entre ambos dispositivos.

Capítulo 7

Implementación Física

En este capítulo se procede a mostrar las principales características del montaje real sistema. Para ello, se mostrará un breve repaso de cada uno de los componentes, sus especificaciones más importantes, la programación de los microcontroladores y los cambios realizados al hardware.

7.1. Maestro DALI

En primer lugar, es de interés hablar del dispositivo que ejercerá la función de maestro DALI y de controlador PID, es decir, el encargado de gestionar por un lado la comunicación a través de la interfaz DALI, y por otro de implementar el control PID del cromatismo de la lámpara.

Como bien se ha dicho anteriormente, el firmware de gestión del protocolo DALI puede ser cargado sobre cualquier microcontrolador de la familia XMC 1000, tras una pequeña adaptación previa, debido a la frecuencia principal del reloj de cada microcontrolador. Concretamente, para este proyecto se ha empleado el microcontrolador XMC 1400, que, a grandes rasgos, tiene las siguientes especificaciones:

Especificación	Tipo / Valor
Nombre del microcontrolador	XMC 1400 VQFN64
Tipo microcontrolador	32-bit ARM Cortex-M0 CPU Core
Frecuencia de Reloj	48 MHz
Interrupciones	64 nodos de interrupción
Memoria RAM	16 Kbytes
Memoria ROM	8 Kbytes
Interfaces de Comunicación	UART, IIC, SPI, IIS etc
Entradas y Salidas	56 (12 entradas analógicas)
Tensión de alimentación	Entre 1.8 y 5.5 V

Tabla 9. Especificaciones del microcontrolador XMC 1400 [22].

Además, se selección la placa de desarrollo XMC 1400 BootKit para la implementación de este proyecto. Esta cuenta con un circuito de puerto serie y depurador, que se ha empleado para la comunicación UART con la interfaz gráfica. Una imagen de esta placa puede verse en la Ilustración 47.

Cabe destacar que, para la implementación del software del maestro DALI, únicamente es necesario una unidad de captura y comparación CCU (*Compare and Capture Unit*), cuya función es la monitorización de señales. En este caso, esta unidad se emplea para detectar cuando se va a recibir un mensaje (el bus DALI tiene una diferencia

de potencial baja). Además, se necesita un temporizador, un nodo de interrupciones, una entrada digital y una salida digital.

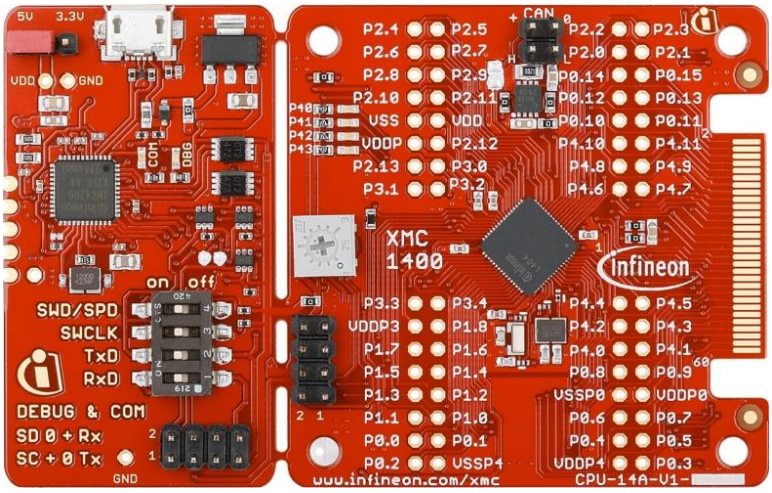


Ilustración 47. Placa de desarrollo XMC 1400 BootKit [22].

Por otra parte, el firmware del maestro DALI también tiene incorporada la librería de control del sensor RGB, desarrollada en C para este proyecto. Dicha librería se ocupa de la configuración de los parámetros del sensor, de la gestión de peticiones de lectura, y la recepción de datos por parte del mismo.

Por último, el firmware del maestro DALI también implementa la librería de control PID, con los parámetros óptimos encontrados en el algoritmo de optimización heurística. En dicha librería se implementa en código lo mostrado en el diagrama de bloques del sistema en bucle cerrado (ver Ilustración 29). El control PID se activará a petición de la interfaz gráfica, y se supondrá finalizado cuando el error entre el valor leído por el sensor RGB y la consigna sea inferior a un 1 % para los tres colores.

7.2. Interfaz DALI

La interfaz física de DALI ya ha sido presentada en detalle en el apartado 5.1.1. Además, en este apartado se adelantó la interfaz empleada: La placa DALI PHY de Infineon Technologies [23]. Esta tiene dos versiones, una primera, directamente adaptada para las placas de desarrollo de los microcontroladores de la familia XMC 1000, y una segunda donde se permite la selección de los pines de transmisión y recepción DALI, además de facilitar otros dos pines para alimentar su bus. Esta segunda ha sido la empleada en este proyecto. Una imagen de esta interfaz puede verse en la Ilustración 48.

Cabe destacar que esta misma interfaz física se emplea tanto en el maestro DALI como en el esclavo, con la única salvedad de que, en el esclavo, esta interfaz ya está integrada en la propia placa de circuito impreso.

7.3. LED Driver

A continuación, actuando como esclavos DALI y como drivers de los tres canales de LEDs RGB, se empleó una versión superior de la placa "*RGB LED Shield with XMC 1202*" mostrada en la Ilustración 49, con apariencia similar. Este kit dispone de tres drivers para LEDs. Cada uno de los drivers actuará como un esclavo DALI independiente, por lo que una misma placa e interfaz DALI servirá para simular el comportamiento de los tres esclavos DALI. Además, el kit empleado en el trabajo está equipado con la interfaz hardware de DALI. Por otro lado, cuenta con un microcontrolador XMC 1300, con las especificaciones mostradas en la Tabla 10.

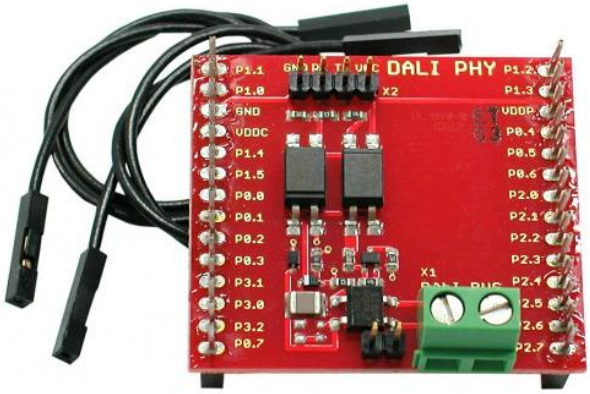


Ilustración 48. Placa DALI PHY [23].

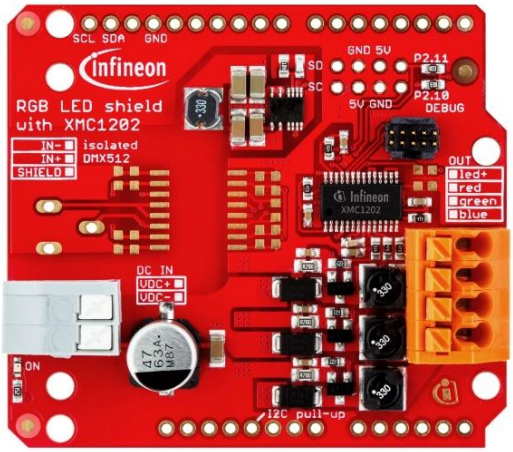


Ilustración 49. Kit de desarrollo " RGB LED Shield with XMC1202" [30]

Como puede verse, este microcontrolador dispone de nueve canales de control de brillo y color, también llamados BCCU (*Brightness Color Control Unit*). Estas unidades básicamente se encargan de la gestión de las señales PWM y los comparadores que controlan los drivers de los LEDs, tal y como se explicó en el apartado 3.1.

Especificación	Tipo / Valor
Nombre del microcontrolador	XMC 1400 T038
Tipo microcontrolador	32-bit ARM Cortex-M0 CPU Core
Frecuencia de Reloj	32 MHz
Memoria RAM	16 Kbytes
Memoria ROM	8 Kbytes
Interfaces de Comunicación	UART, IIC, SPI, IIS etc
Entradas y Salidas	Hasta 38 (12 entradas analógicas)
Canales de control de LED	9 canales
Tensión de alimentación	Entre 1.8 y 5.5 V

Tabla 10. Especificaciones del microcontrolador XMC 1300 [31]

7.4. Sensor RGB

Para la lectura del nivel de iluminación RGB se empleó el sensor TCS34725, que, como ya se adelantó en el capítulo 3, consta de una matriz de fotodiodos y cuatro bloques ADC de 16 bits para la conversión a digital del valor de intensidad. La comunicación con este sensor se realiza a través del estándar de I^2C. Cabe recordar que el sensor permite la selección de la ganancia en la medida, así como el tiempo de integración (o equivalentemente, el número de muestras integradas en una sola medida). En concreto, para este trabajo se configuró una ganancia igual a 60 y un tiempo de integración igual a 101 ms. En la Tabla 11 se muestran las características más importantes de este sensor:

Parámetros	Valor
Tensión de Alimentación	3 V
Temperatura de Operación	[-30, 70] ℃
Corriente de Alimentación	0.235 mA
Tiempo de Conversión del ADC	2.54 ms
Tiempo de Integración Configurado	101 ms
Ganancia Configurada	60

Tabla 11. Resumen de las características más importantes del sensor RGB.

Una imagen del sensor puede ser vista en la Ilustración 50.

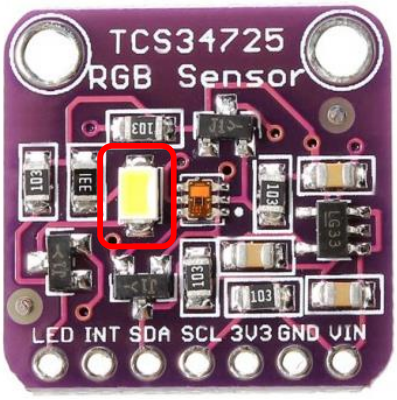


Ilustración 50. Sensor RGB TCS34725 [32] .

Este sensor está pensado para dar un valor de la RGB de la propia luz que el emite, al ser reflejada sobre un objeto, mostrando así el color del mismo. Es decir, la placa del sensor está dotada de un pequeño LED (recuadrado en rojo en la Ilustración 50) que emite luz blanca neutra y que permite conocer el color de un objeto midiendo su reflejo por el sensor RGB. Este LED ha sido retirado de la placa para este proyecto, ya que se desea que el propio sensor tome una lectura de la luz que ya de por si emiten los LEDs.

7.5. LEDs RGB

Para la fuente de iluminación LED se ha seleccionado una tira de LEDs SMD de 5 metros (300 LEDs) con tensión de alimentación a 12V y corriente máxima de 6 A. Por su parte, la temperatura de operación barre un rango de -25 ℃ a 50 ℃. La vida útil de los mismos es aproximadamente igual a 50000 horas.

Ilustración 51. LEDs seleccionados [14].

Estos LEDs tienen cuatro entradas. La primera es de alimentación, donde se debe colocar el terminal positivo de la salida de la fuente. Las otras tres entradas restantes son los terminales negativos de los respectivos canales rojo, verde y azul.

Para el testeo del correcto funcioamiento del sistema, se implementó un prototipo en el que los LEDs estaban depositados en el interior de un tarro de vidrio transparente, y en cuya tapa fue posteriormente colocado el sensor RGB para la lectura de datos.

7.6. Alimentación

Por último, para alimentar el bus DALI y los LEDs se seleccionó una fuente de alimentación reconfigurable, con tensión de salida igual a 12, 15, 16, 18, 19, 20 y 24 voltios y corriente de salida igual a 4 – 4.5 amperios. Una imagen de este cargador puede ser vista en la Ilustración 52.

Con este cargador se alimentaron los LEDs y el bus DALI. Si bien las especificaciones hablan de una tensión de alimentación distinta para cada elemento, lo cierto es que se probó tanto la operación del sistema alimentando los LEDs a 16 V (tal y como se requiere para el bus DALI), como alimentando el bus DALI a 12 V (según se recomienda para los LEDs). El resultado en ambos casos fue satisfactorio así que, a fin

de evitar posibles rupturas de los LEDs durante la operación y tomando como un mal menor el hecho de que pudiera haber algún bit erróneo en la comunicación DALI (hecho que nunca se ha observado), se prefirió utilizar una tensión de alimentación igual a 12 V para ambos elementos. Esto además queda justificado por los márgenes de operación en la alimentación del bus DALI, pudiéndose tener una tensión entre los hilos del bus hasta de 9.5 voltios, tal y como puede verse en la Ilustración 9.

Ilustración 52. Fuente de alimentación seleccionada [33].

Cabe destacar que, para poder alimentar tanto el bus DALI como los LEDs, se realizaron unos pequeños cambios en la placa del cargador. En concreto, se quitó el cable de salida que esta traía de fábrica y se equipó con dos salidas distintas en paralelo, una para alimentar los LEDs y otra para alimentar en bus DALI. En concreto, la salida de alimentación del bus DALI cuenta también con las dos resistencias de 100 ohmios necesarias para la limitación de la corriente (ver Ilustración 34). Una imagen del proceso y el acabado puede verse en la Ilustración 53.

Por último, se muestra la Ilustración 54 , donde puede verse el sistema en conjunto, con todos los bloques que lo componen interconectados.

Con esta implementación física queda materializado el diagrama de bloques presentado en la Ilustración 4.

7.8. Resultados del Sistema Real

A continuación, se muestran los resultados obtenidos tras la recogida de datos con el sistema real, y su comparación con las simulaciones en MATLAB del modelo detallado en el capítulo 3.

Para la obtención de la respuesta dinámica del sistema real ya controlado por el PID, se introdujo en la interfaz un valor de consigna igual a 100 para los tres canales. Dado que este valor es relativo a un máximo de 255, este entonces se corresponde a una consigna igual a 1600 en el control PID.

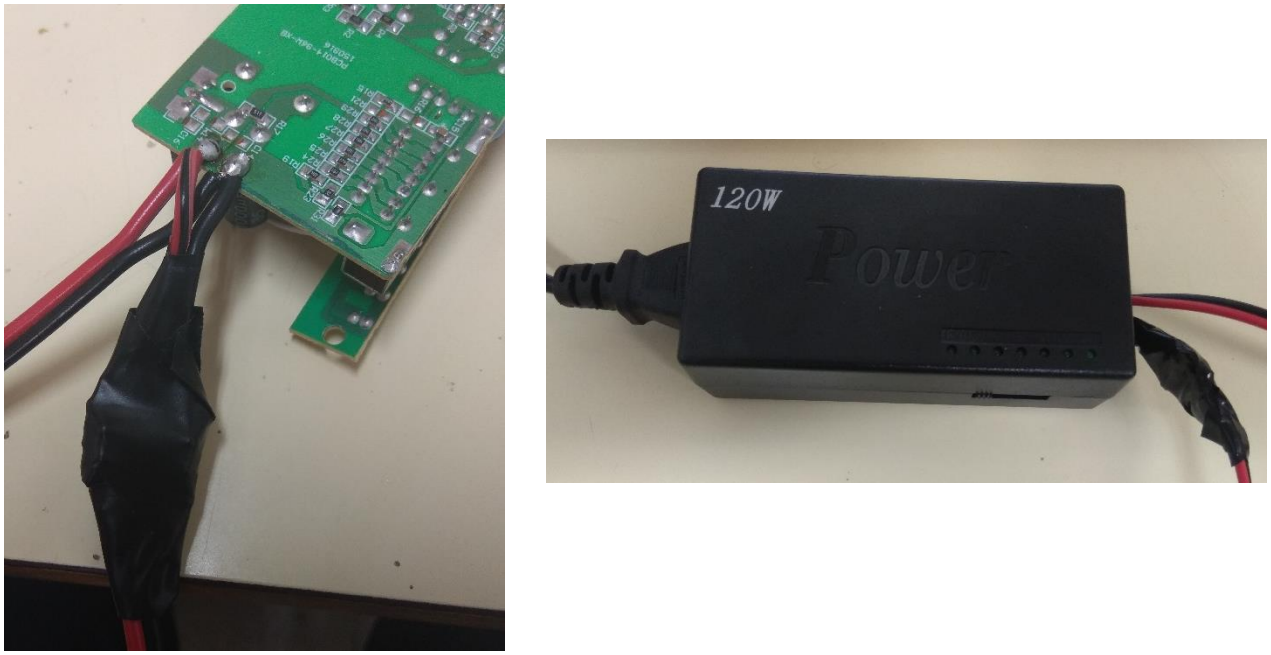


Ilustración 53. Inserción de salidas en la fuente de alimentación para el bus DALI y los LEDs.

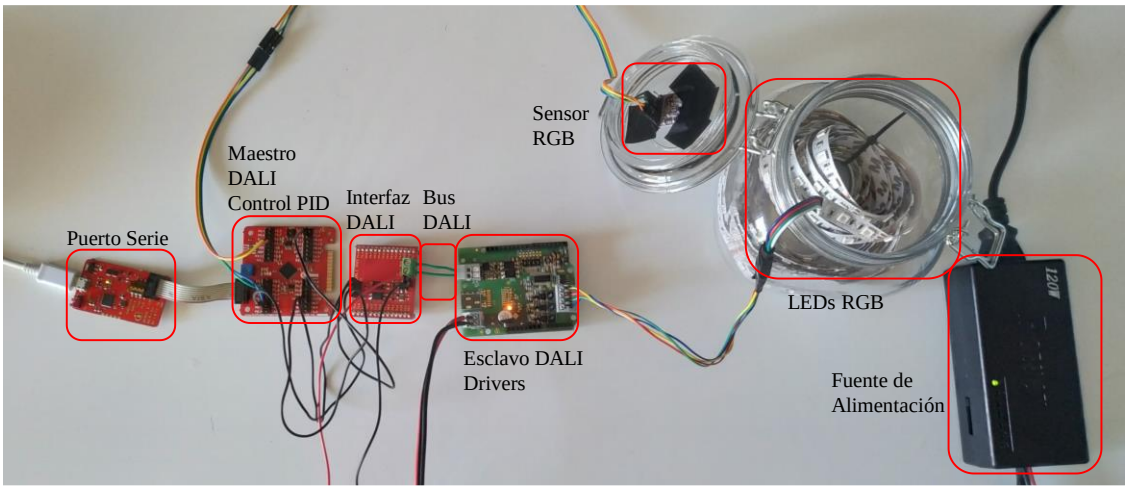


Ilustración 54. Sistema en conjunto con todos los componentes.

A continuación, se tomaron muestras 100 de la lectura del sensor, y se representó el resultado, obteniéndose las gráficas de la Ilustración 55. En ellas puede verse como la respuesta dinámica del canal rojo se asemeja a la simulación del modelo. Sin embargo, no ocurre así con los canales verde y azul, que se alejan de la simulación del modelo obtenido, presentando unas sobre-elongaciones inesperadas. Tras estudiar posibles razones, se observó que el tiempo entre muestras medido en la recogida de datos del sistema real revelaba un periodo superior al seleccionado para el modelo del sistema discreto, fruto del tiempo de integración del sensor, y del retardo en la transmisión de los

6 comandos DALI. Concretamente, el periodo de modelo es de 204 ms, mientras que el sistema real tiene un periodo aproximado de 292 ms. Estos 88 ms de diferencia, posiblemente producidos por el tiempo de procesamiento de los microcontroladores, podrían ser el motivo de que la respuesta dinámica del sistema real se alejara de la esperada en el modelo. Es por ello que a continuación, se probó a simular el modelo en Simulink con el periodo real, de 292 ms, obteniéndose el resultado mostrado en la Ilustración 56.

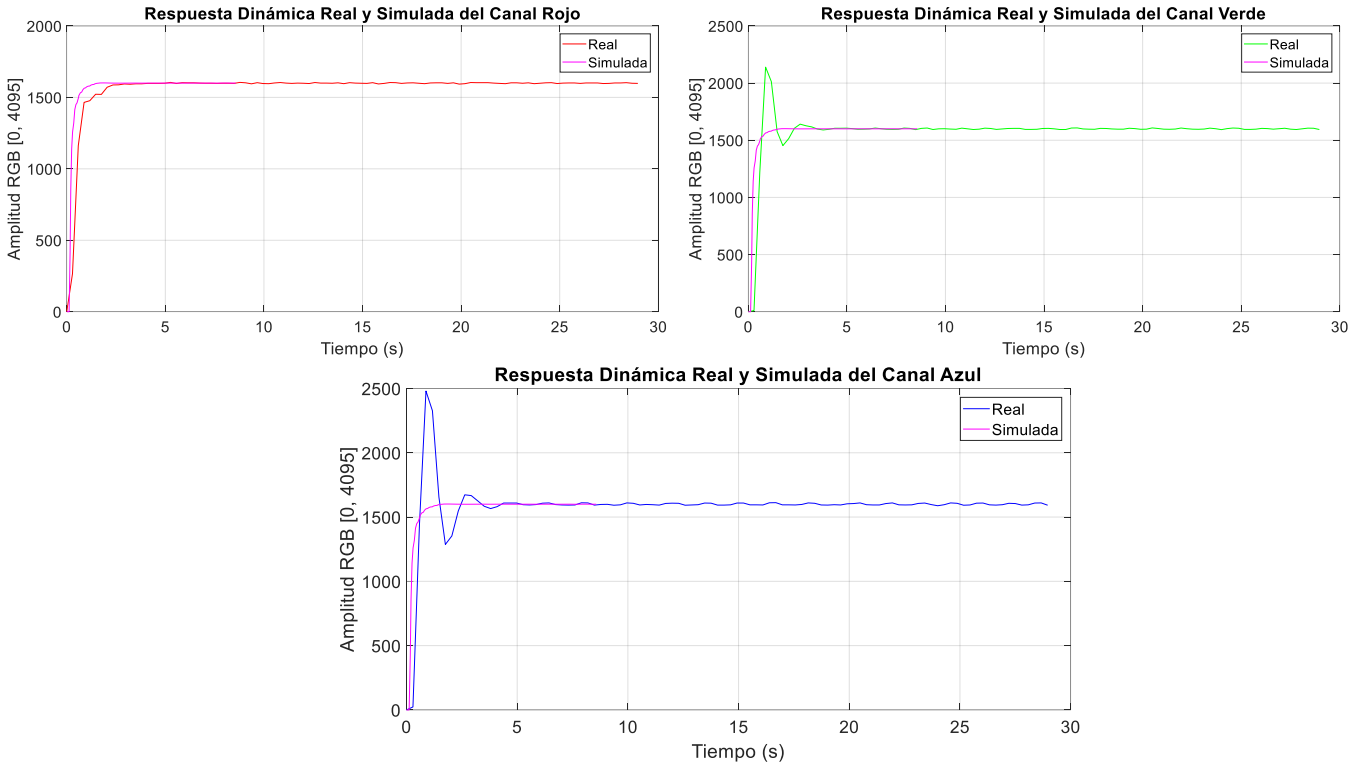


Ilustración 55. Comparación entre la respuesta dinámica del sistema real y la del modelo.

Como puede verse en la Ilustración 56, con este nuevo periodo también aparecen unas oscilaciones en la respuesta dinámica de la simulación, y que se parecen a las obtenidas en el sistema real.

A continuación, entonces, se reajusto el modelo matemático discreto de Simulink con el nuevo periodo obtenido, y se volvió a lanzar el algoritmo de nube de partículas para obtener unos parámetros del control PID que devolvieran una respuesta dinámica más aceptable. Así se obtuvieron los parámetros K_P, K_I y K_D iguales a 0.409 , 2.566 y 0.001 respectivamente, que devolvieron la respuesta dinámica de la Ilustración 57. Sin embargo, tras volver a recoger datos del sistema real, se obtuvieron unas gráficas similares a las mostradas en la Ilustración 55, con las sobre-elongaciones de los canales verde y azul.

Esto revela que el fuerte acoplamiento existente entre el canal verde y azul no puede ser eliminado directamente con la red de desacoplo, debido a que para calcular el

elemento en el instante de tiempo t_i se usa el valor de las consignas en el instante anterior. Dado que inicialmente las consignas son 0, en el primer instante la red de desacoplo está desactivada.

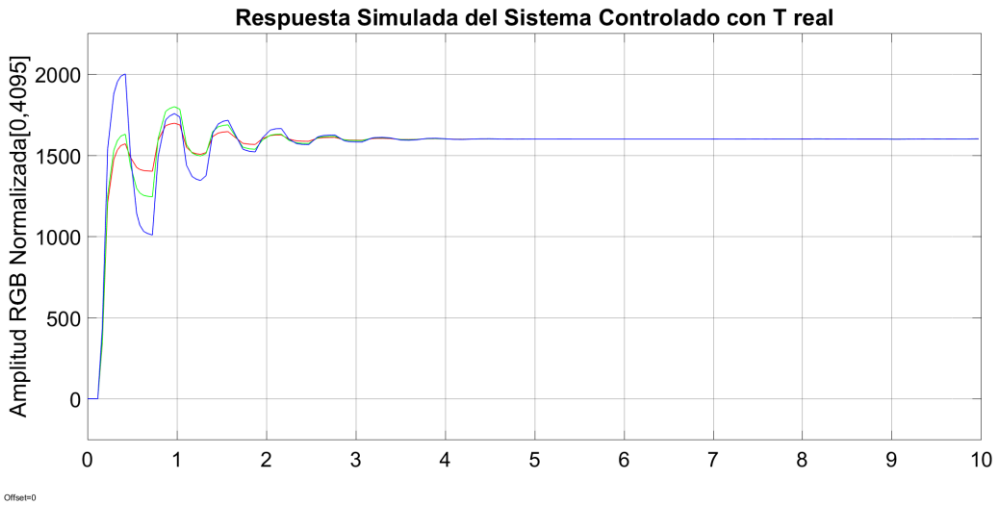


Ilustración 56. Respuesta dinámica del modelo discreto en conjunto con T = 292 ms.

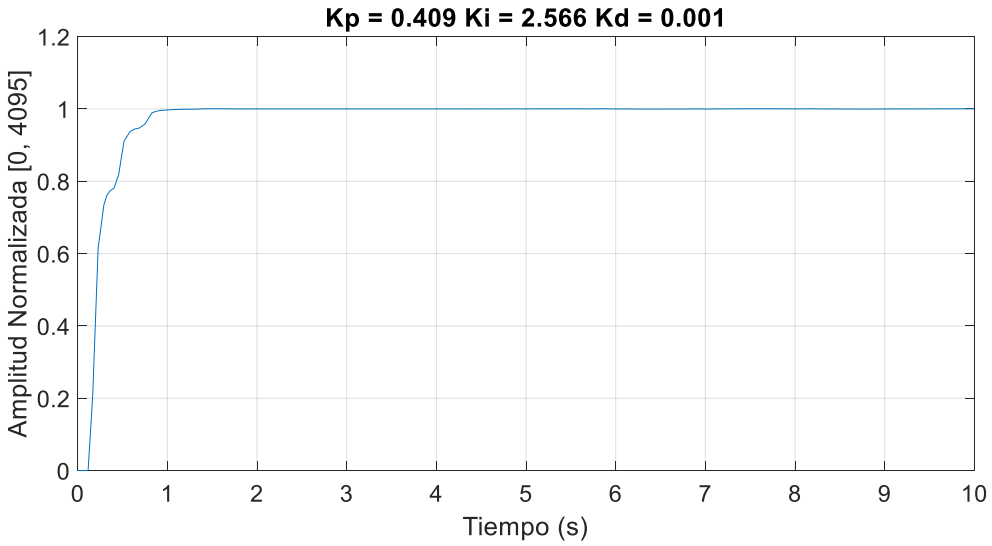


Ilustración 57. Respuesta dinámica del sistema con el control PID ajustado para T = 292 ms

De hecho, si se simula la misma respuesta mostrada en la Ilustración 56 (para el sistema discreto) en el sistema continuo, se observa que, en este caso, los tres canales responden con la misma dinámica, tal y como es de esperar. Esto se muestra en la Ilustración 58.

Por último, en un intento por sintonizar el controlador PID de forma que enmendara las dependencias en tiempo continuo, se volvió a lanzar el algoritmo de nube de partículas, esta vez teniendo en cuenta las salidas de los tres canales (y no una como modelo), sumando posteriormente el valor de la suma ponderada de objetivos para cada

canal. Se obtuvo así un control PID con parámetros $K_P = 0.212, K_I = 2.031$ y $K_D = 0.001$.

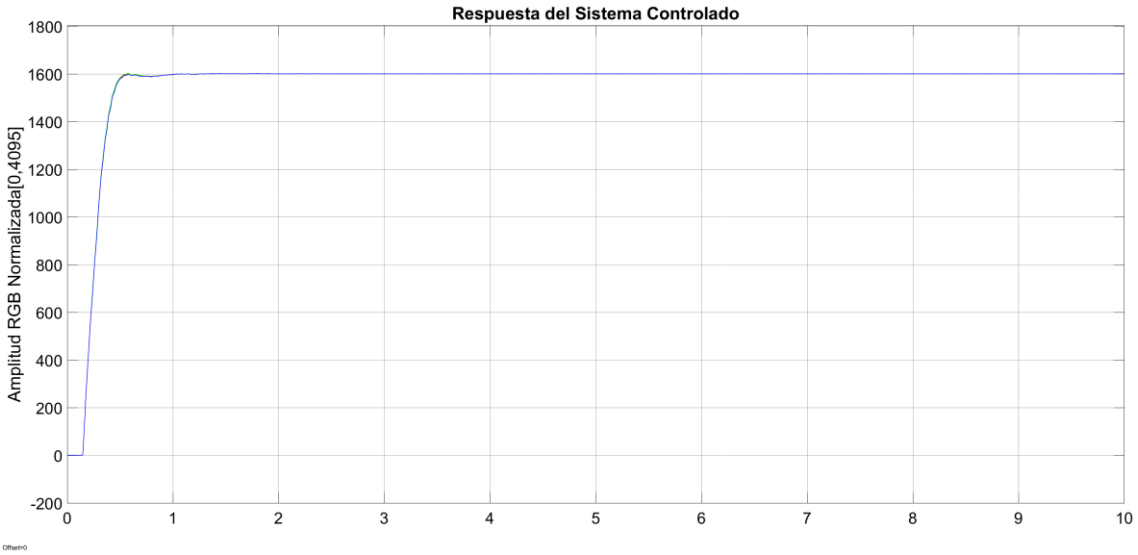


Ilustración 58. Respuesta dinámica del modelo continuo en conjunto.

Tras esta última sintonización del PID, se recogieron nuevamente muestras de la lectura del sensor y se obtuvieron las respuestas dinámicas mostradas en la Ilustración 59, en donde se comparan con el modelo del sistema.

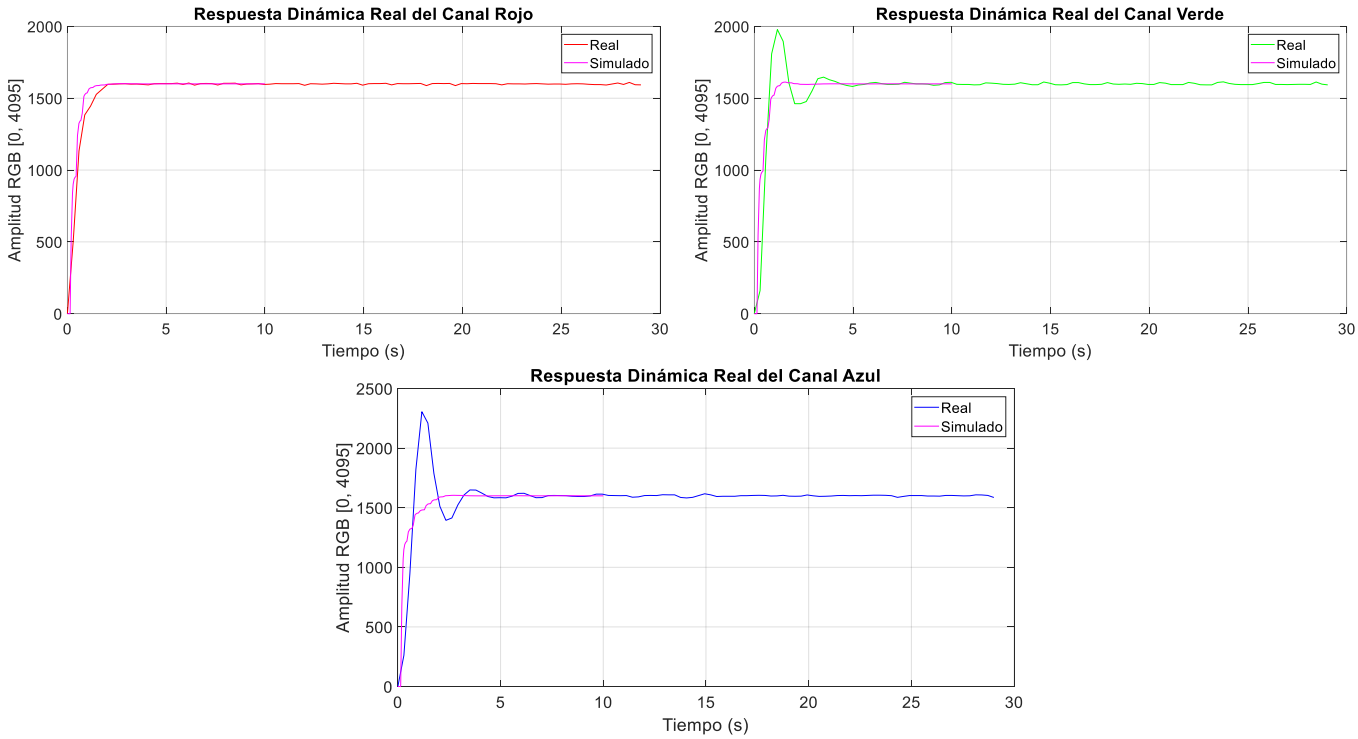


Ilustración 59. Comparación entre la respuesta dinámica del sistema real y la del modelo con el PID optimizado para eliminar las interferencias entre canales.

Si bien en esta nueva respuesta dinámica se consiguió reducir la amplitud de las sobre-elongaciones, estas seguían apareciendo.

Ante estos resultados se concluye que el modelo matemático del sistema en tiempo continuo se asemeja razonablemente bien a los datos recogidos, pero al introducir el controlador en tiempo discreto aparece una sobre-elongación en los canales verde y azul, fruto de las dependencias entre estos, que perturban la respuesta esperada. Por lo tanto, para mejorar el controlador será necesario determinar la causa de la discrepancia e incluirla en el modelo del sistema/controlador para poder equiparar la respuesta simulada y la real, y optimizar de nuevo los parámetros del controlador PID, tal y como ya se ha hecho durante esta fase de experimentación final.

Cabe destacar que estas discrepancias entre el modelo discreto y los datos recogidos del sistema real no impiden en ningún caso que los controladores PID obtenidos controlen perfectamente el sistema, alcanzando los valores de consigna deseados.

De hecho, aquí [34] se enlaza un vídeo donde puede verse como, tras realizar un direccionamiento de los esclavos anclados al bus DALI, se repiten cinco cambios de consigna distintos y posteriormente se realizan lecturas del sensor RGB. Así se comprueba cómo se alcanza correctamente el valor de consigna para los cinco cambios. Este experimento queda representado en la Ilustración 60.

En concreto, la primera consigna introducida es equivalente a un valor de 1600 para los tres canales. A continuación, se modifica la consigna del canal rojo a 3200. Posteriormente la consigna de este canal baja a 1440 y la del canal verde asciende a 3200. Seguidamente, el canal rojo baja nuevamente a 1280, el canal verde ve reducida su consigna a un valor de 1440 y la consigna del canal azul aumenta a 3200. Finalmente, las consignas de los tres canales se configuran a 3200.

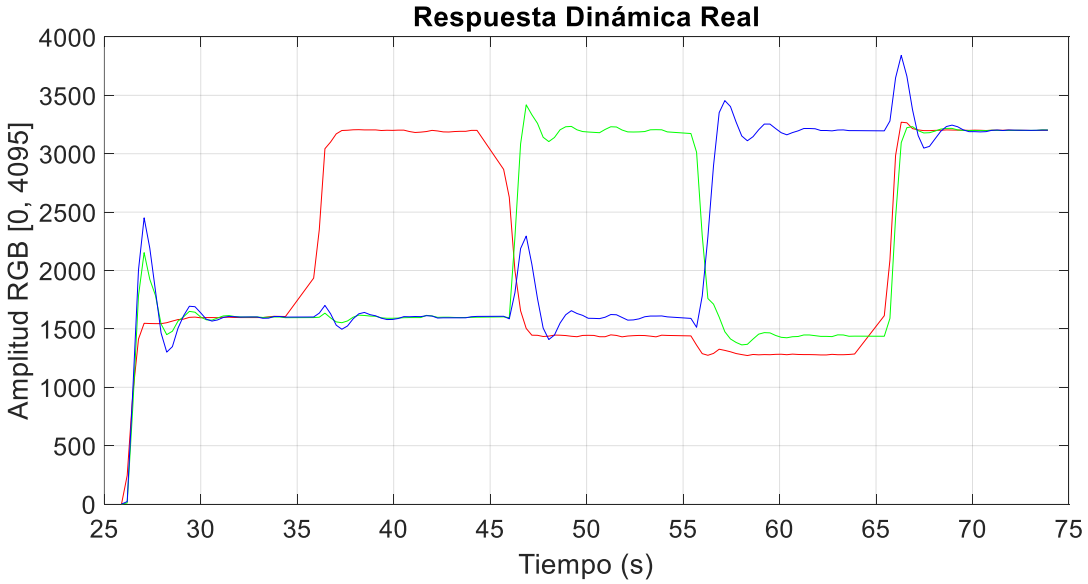


Ilustración 60. Cambios de consigna del sistema real.

Capítulo 8

Conclusiones y Futuros Trabajos

En este capítulo se muestra, a modo de síntesis, un resumen de los objetivos cumplidos en este proyecto. Posteriormente, se muestras posibles líneas de trabajo futuras.

8.1. Conclusiones

En cuanto a los objetivos conseguidos, se concluye en:

- **Caracterización del Sistema:** En primer lugar, se ha procedido a caracterizar la respuesta, tanto dinámica como estacionaria, de un sistema constituido por una luminaria LED de tres canales (rojo, verde y azul), y un sensor sensible a estos, todo ello aplicado desde un enfoque multivariable.

- **Implementación del Control PID:** A continuación, se ha implementado el controlador PID para el control del cromatismo de la luz del sistema descrito anteriormente. Para la sintonización y optimización del mismo, se han empleado técnicas heurísticas.

- **Desarrollo del Controlador DALI:** Por otro lado, se ha desarrollado un controlador basado en el estándar de iluminación DALI sobre la familia de microcontroladores XMC1000 de Infineon Technologies.

- **Implementación de la Interfaz Gráfica:** Por último, a fin de facilitar el uso del controlador, así como su depuración, se ha diseñado y programado una interfaz gráfica para ordenador, basada en Python y el entorno de desarrollo Qt.

- **Desarrollo del protocolo de comunicación:** Se ha desarrollado un protocolo de comunicación basado en la interfaz UART para la comunicación del ordenador donde se está ejecutando la interfaz con el maestro DALI. Este protocolo de comunicación cuenta con importantes funcionalidades como el escaneo de maestros DALI a través de un código *ping* o la detección de errores usando bytes de redundancia cíclica.

- **Implementación de las librerías:** Para la consecución de todos estos puntos, se han desarrollado distintas librerías, tanto en Python (para la librería de la interfaz gráfica), como en C (para las librerías del controlador DALI, del sensor de color

RGB y del controlador PID), como en MATLAB y Simulink (para la obtención del modelo formal del sistema y la optimización de los parámetros del PID).

8.2. Trabajos Futuros

Por último, se plantean una serie de propuestas para complementar este proyecto a fin de mostrar posibles vías de desarrollo del mismo. Estas propuestas han surgido durante la realización del proyecto.

- **Estudio de un control centralizado para el cromatismo:** La primera línea de investigación propuesta consiste en estudiar una posible implementación del control de cromatismo de las luminarias mediante una estructura de control centralizado (estructura K), donde se tuvieran en cuenta las dependencias existentes entre los canales y se actuara en consecuencia.

- **Desarrollo de un algoritmo de optimización para cualquier tipo de control:** En este proyecto se ha trabajado con unos LEDs concretos de tres colores, y se ha desarrollado un programa de optimización basado en técnicas heurísticas para el mismo control propuesto. Otra posible línea de desarrollo podría ser la implementación de un optimizador para otro tipo controladores.

- **Desarrollo de control de temperatura de color:** Otro de los posibles trabajos a desarrollar podría ser el control de la temperatura de color, así como la intensidad de la luz. En esta posible línea de desarrollo podrían traducirse las consignas de temperatura de color e intensidad en valores RGB mediante un cambio del espacio del color, y aplicar el controlador PID aquí desarrollado para obtener la consigna deseada.

- **Conexión del maestro DALI de forma inalámbrica:** Otra propuesta interesante puede ser conectar el maestro DALI de forma inalámbrica, a través de un módulo Bluetooth o Wifi, para realizar su control de forma remota.

- **Desarrollo de una placa de circuito impreso para el maestro DALI:** En este proyecto, para la implementación del maestro DALI se han empleado distintos módulos interconectados mediante cables. Esta propuesta sugiere el desarrollo de una placa de circuito impreso específica para el maestro DALI, que integre la fuente de alimentación, la interfaz DALI, el sensor de color RGB, y por último el microcontrolador que gestione el protocolo, la interfaz, la comunicación con el sensor e implemente el control PID. La fuente de alimentación podría ser desarrollada para equiparla con doble salida, a fin de alimentar los LEDs y el bus DALI de forma independiente. Por otro lado, también sería interesante el desarrollo de un sensor de color específico para este dispositivo. Además, de

forma paralela al control del maestro DALI a través de la interfaz gráfica, se podría equipar la placa de circuito impreso del maestro DALI con algunos botones para la ejecución de determinados comandos DALI sin necesidad de estar conectado este a un ordenador.

- **Desarrollo de una fuente de iluminación LED fija:** Uno de los problemas secundarios en este proyecto ha sido las pequeñas variaciones que existían en las ganancias en estado estacionario del sistema, fruto de las distintas posiciones que tomaban los LEDs en el interior del tarro, que provocaban desviaciones en la lectura del sensor RGB. Un paso más avanzado en este proyecto podría contemplar el desarrollo de una fuente de iluminación LED donde estos estuvieran fijos, y se contara además con un difusor que homogenizara la luz resultante.

- **Pruebas del sistema en una red de luminarias:** En este proyecto se ha trabajado con un nodo que simula tres esclavos distintos. Sin embargo, el estándar DALI permite desplegar una red de hasta 64 esclavos. Otra posible línea de trabajo futura podría ser realizar pruebas del sistema controlado con una red de esclavos, donde seguramente aparecerían retardos en la comunicación DALI que habría que tener en cuenta. Además, dependiendo de la aplicación a desarrollar, si esta pretende cubrir una extensión grande, sería necesario la inclusión de más de un sensor, y, por lo tanto, de más de un lazo de control. En este caso, quizás sería interesante agrupar los esclavos relativos a las zonas de operación en diferentes grupos (aprovechando la ventaja de que DALI permite la creación de hasta 16 agrupaciones distintas), y, en cada zona, incluir un sensor y un lazo de control distintos para garantizar que el color de cada zona es el deseado.

Bibliografía

[1] Cines Multiplex, «YouTube,» [En línea]. Available: https://www.google.com/search?biw=1745&bih=852&tbm=isch&sa=1&ei=_mvyXJmfOI6tgweS05bADA&q=cine+4d&oq=cine+4d&gs_l=img.3..35i39j0l5j0i67l2j0l2.16504.16602..16741...0.0..0.71.135.2.....0....1..gws-wiz-img.1YvHbZtq0o0#imgrc=2n6e_UPDPrw4_M:.

[2] D. D. l. Heras, «twitter,» [En línea]. Available: https://twitter.com/David_Heras/status/1037763612008427520.

[3] Relaxing and useful sounds, noises and videos, «YouTube,» [En línea]. Available: https://www.google.com/search?biw=1745&bih=852&tbm=isch&sa=1&ei=EXDyXLeiJcTkgweiiLmQAw&q=cabina+pasajeros+avion&oq=cabina+pasajeros+avion&gs_l=img.3..0.9180.13246..13502...1.0..0.93.1554.23.....0....1..gws-wiz-img.....35i39j0i67j0i5i30j0i8i30.rSvCqf895.

[4] S. M. A. J. F. L. J. T. W. Sina Afshari, «Modeling and Feedback Control of Color-tunable LED Lighting System».

[5] V. M. A. P. Jaya Muruga Raja, «Neural Network Based Dimming Level Control of LED Network».

[6] M. W. Y. Z. Huadong Li, «Development and Research of Lighting System».

[7] L. H.-l. Ma Jian-she, «Low-cost implementations of LED intelligent lighting based on DALI».

[8] Soliton, «UART Protocol Validation Service,» [En línea]. Available: https://www.solitontech.com/uart-protocol-validation-service/.

[9] Recombinación Directa, «http://augertoolimage.blogspot.com,» [En línea]. Available: https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwj70fzHvPDgAhWsxoUKHTH-BtIQjRx6BAgBEAU&url=http%3A%2F%2Faugertoolimage.blogspot.com%2F2013%2F09%2Fauger-recombination.html&psig=AOvVaw2saeQ8Vq6sEdzpYFlxAEyt&ust=1552063310370596.

[10] I-V RGB LEDs, «Chegg,» [En línea]. Available: https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwje8PTt1PDgAhUMzoUKHawiAesQjRx6BAgBEAU&url=https%3

A%2F%2Fwww.chegg.com%2Fhomework-help%2Fquestions-and-answers%2Fgiven-blue-led-45v-battery-want-make-led-light-need-10-ma-current--.

[11] Mouser Electronics, «Digitally Addressable Lighting Interface,» [En línea]. Available: https://www.mouser.es/applications/lighting-digitally-addressable/?utm_medium=email&utm_source=november2013&utm_campaign=electronicos-mx&utm_content=article.

[12] «Ingeniería de Redes. Código Manchester,» [En línea]. Available: http://ingk3lmyrd.blogspot.com/p/codigo-amnchester.html.

[13] Infineon Technologies, «32-bit XMC1000 Industrial Microcontroller ARM® Cortex®-M0,» [En línea]. Available: https://www.infineon.com/cms/en/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-cortex-m/32-bit-xmc1000-industrial-microcontroller-arm-cortex-m0/.

[14] Ebay, «Ultra Brillante 3528/5050 60/300 Cuentas 1/5M RGB LED Tira Flexible de la Luz 3,» [En línea]. Available: https://www.ebay.es/itm/Ultra-Brillante-3528-5050-60-300-Cuentas-1-5M-RGB-LED-Tira-Flexible-de-la-Luz-3/173042977034?ssPageName=STRK%3AMEBIDX%3AIT&var=471806829614&_trksid=p2057872.m2749.l2649.

[15] Infineon Technologies, «RGB LED Lighting Shield with XMC1202,» [En línea]. Available: https://www.infineon.com/cms/en/product/evaluation-boards/kit_led_xmc1202_as_01/.

[16] ams Datasheet , «TCS3472 Color Light-to-Digital Converter with IR Filter,» [En línea]. Available: https://ams.com/documents/20143/36005/TCS3472_DS000390_2-00.pdf/6e452176-2407-faaf-a590-d526c78c7432.

[17] F. M. García, «Tema 2: Análisis de sistemas multivariables.».

[18] F. M. García, «Tema 4: Control por desacoplo.».

[19] J. G. Jurado, «Diseño de sistemas de control multivariable por desacoplo con controladores PID.».

[20] A. H. O. G. P. R. E. BESADA, «OPTIMIZACIÓN HEURÍSTICA BASADA EN TECNICAS EVOLUTIVAS».

[21] Wikipedia, «Espacio de color CIE 1931,» [En línea]. Available: https://es.wikipedia.org/wiki/Espacio_de_color_CIE_1931.

[22] Infineon Technologies, «KIT_XMC14_BOOT_001,» [En línea]. Available: https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc14_boot_001/.

[23] Infineon Technologies, «DALI PHY for XMC™ Boot Kits Board User Manual».

[24] Infineon Techonologies, «DALI 2.0 Software Download for XMC™ 1000 Microcontrollers,» [En línea]. Available: https://infineoncommunity.com/2018Mar_GLOB_GATED_DALI_Software_ID941.

[25] D. I. I. Aliance, «DALI Quick Start Guide,» [En línea]. Available: https://www.digitalilluminationinterface.org/data/downloadables/6/4/dali-quick-start-guide_public-v1_april-2018.pdf.

[26] Infineon Technologies, «DALI 2.0 Control Gear Stack library».

[27] J. L. Chaves, «Google Drive - Control DALI,» [En línea]. Available: https://drive.google.com/file/d/1tTUZizWaDC3eOGqw2LaT_ePUvK3nEvJW/view.

[28] T. Q. Company, «Qt,» [En línea]. Available: https://www.qt.io/.

[29] P. S. Foundation, «PyQt5 5.12.2,» [En línea]. Available: https://pypi.org/project/PyQt5/.

[30] Infineon Technologies, «RGB LED Lighting Shield with XMC1202,» [En línea]. Available: https://www.infineon.com/cms/en/product/evaluation-boards/kit_led_xmc1202_as_01/.

[31] Infineon Technologies, «XMC1300 - Reference Manual,» [En línea]. Available: https://www.infineon.com/dgdl/Infineon-xmc1300-AA_rm-UM-v01_01-EN.pdf?fileId=5546d46255dd933d0155e3175a5f77c7.

[32] Hacktronics India, «TCS34725 RGB Color Sensor with IR filter and White LED,» [En línea]. Available: https://hacktronics.co.in/color-recognition/tcs34725-rgb-color-sensor-with-ir-filter-and-white-led.

[33] Ebay, «Cargador universal para el portátil.,» [En línea]. Available: https://www.ebay.es/itm/96W-15-16-18-19-20-22-24-VOLT-Universal-Laptop-Car-Airplane-Adaptor/143198856909?hash=item21575126cd:g:jOgAAOSwZnpcUbOB.

[34] J. L. Chaves, «Google Drive - Respuesta real del sistema controlado.,» [En línea]. Available:

https://drive.google.com/file/d/1I16ejFLRkYNIsMNJQXoWIDIol9cqfA-V/view.

[35] EL DIODO LED Funcionamiento, «asesor ped,» [En línea]. Available: https://www.youtube.com/watch?v=C3L-ON4D9rw.

[36] Colores RGB, «nochesdecode.com.ar,» [En línea]. Available: https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjp-cmHyfDgAhVt5eAKHUPMBVwQjRx6BAgBEAU&url=http%3A%2F%2Fnochesdecode.com.ar%2F2012%2F12%2Fgenerar-color-aleatorio-en-hex%2F&psig=AOvVaw2oxQXXRozgs6IwHs-7eqrN&ust=1552066307681387.

[37] Modulación PWM, «B105 lab,» [En línea]. Available: https://www.google.com/url?sa=i&source=images&cd=&ved=2ahUKEwj4uZOX2fDgAhUJ1uAKHVMRCEcQjRx6BAgBEAU&url=http%3A%2F%2Felb105.com%2Fstm32f4-configurar-el-pwm-con-timers%2F&psig=AOvVaw3IouoK0hEqj5HIjdoig5vZ&ust=1552070998183796.

[38] Universidad de Granada, «YouTube,» [En línea]. Available: https://www.youtube.com/watch?v=C3L-ON4D9rw.

[39] «http://augertoolimage.blogspot.com,» [En línea]. Available: http://augertoolimage.blogspot.com/2013/09/auger-recombination.html.

[40] «nochesdecode.com.ar,» [En línea]. Available: http://nochesdecode.com.ar/2012/12/generar-color-aleatorio-en-hex/.