

Índice general

Índice de figuras	vii
Índice de tablas	ix
1. Introducción	1
1.1. Estado del Arte	1
1.1.1. Introducción a historia de los ledes	1
1.1.2. Primeros tipos de ledes	2
1.1.3. Avances en diseños modernos	2
1.1.4. Ledes de direccionamiento individual	3
1.2. Problemática	4
1.3. Justificación	4
1.4. Objetivos	5
1.4.1. Objetivo general	5
1.4.2. Objetivos específicos	5
1.5. Hipótesis	5
1.6. Metodología	5
1.6.1. Especificaciones de diseño	5
1.6.2. Creacion de esquemáticos	6
1.6.3. Simulación eléctrica	6
1.6.4. Montaje	6
1.6.5. Pruebas experimentales	6
2. Marco teórico	7
2.1. Ledes de direccionamiento individual	7
2.2. Microcontrolador ESP8266	10
2.3. Módulo de comunicación Bluetooth	12
2.4. Aplicación en Android	13
3. Métodos	15
3.1. Especificación de propósitos y requerimientos	15
3.2. Especificación de procesos	16
3.3. Modelo de dominio	17
3.4. Modelo de información	18
3.5. Especificación de servicios	18
3.6. Especificación de los niveles IoT	19

3.7. Especificación de la vista funcional	19
4. Resultados	21
4.1. Especificación de la vista operacional	21
4.2. Disolución de las limitaciones de diseño	22
4.2.1. Consumo de las luces	23
4.2.2. Frecuencia de actualización	23
4.2.3. Cableado	24
4.3. Implementación de los dispositivos y componentes	24
4.3.1. Esquema	25
4.3.2. Construcción	27
4.3.3. Programación del microcontrolador	29
4.3.4. Explicación del código	31
4.4. Desarrollo de la aplicación	35
5. Pruebas	41
5.1. Microcontrolador y ledes	41
5.2. Uso de la aplicación	42
6. Conclusiones	45
Bibliografía	49

Índice de figuras

2.1. Dispositivo LED RGB con circuito integrado [1]	8
2.2. Esquema de montaje para un LED WS2812b. El puerto DIN es para la comunicación serial [2]	8
2.3. Esquema de montaje para un LED WS2812b con una placa Arduino [3].	9
2.4. Encapsulado de un LED RGB 5050.	9
2.5. Dos tiras de LEDs de direccionamiento individual conectadas en serie [4].	10
2.6. Chip del microcontrolador ESP8266 [5].	11
2.7. Placa con el microcontrolador ESP8266 montado [5].	11
2.8. Módulos Bluetooth para microcontroladores [6].	12
2.9. Diagrama para la conexión del módulo Bluetooth HC-05 con una placa ESP8266 [7].	12
3.1. Especificación de procesos para el sistema de control del SunBoard Led System.	16
3.2. Modelo de dominio del SunBoard Led System.	17
3.3. Modelo de información del SunBoard Led System.	18
3.4. Especificación de servicios del SunBoard Led System.	19
3.5. Especificación de servicios del SunBoard Led System.	19
3.6. Vista funcional del SunBoard Led System.	20
4.1. Vista operacional del SunBoard Led System.	22
4.2. Diagrama del diseño eléctrico de los componentes del SunBoard.	24
4.3. Esquemático del SunBoard.	26
4.4. Parte frontal del prototipo.	27
4.5. Parte trasera del prototipo.	28
4.6. Microcontrolador conectado al prototipo.	28
4.7. Adaptador de corriente del prototipo.	29
4.8. Conexión de la tira de ledes del prototipo.	29
4.9. Diagrama de funcionamiento del programa del SunBoard.	30
4.10. Menú principal de la aplicación.	36
4.11. Barra de funciones de la aplicación.	36
4.12. Guardando una nueva rutina en la aplicación.	37
4.13. Cargando una rutina en la aplicación.	37
4.14. Menú de configuración de la aplicación.	38
4.15. Opciones extra en el menú de configuración de la aplicación.	38

Índice de figuras

4.16. Opciones para la ejecución de demos en la aplicación.	39
4.17. Menú para el mapeo de posiciones..	40
5.1. Prueba de encendido del microcontrolador y el led indicador.	41
5.2. Comprobando la conexión Bluetooth con el dispositivo.	42
5.3. Prueba de funcionamiento del Modo SunBoard.	42
5.4. Prueba del funcionamiento de las rutinas.	43
5.5. Comprobando el funcionamiento de los demos.	43
5.6. Ejecución de distintas instrucciones y su salida por el puerto serial. .	44

Índice de tablas

Índice de tablas

Nomenclature

Bluetooth Protocolo de comunicación inalámbrico para transmisión de voz y datos entre distintos dispositivos.

Chipset Conjunto de circuitos integrados construidos en base a la arquitectura de un procesador.

FPS *Frames per second*, fotogramas por segundo, es la cantidad de veces que se actualiza una imagen cada segundo.

IDE Integrated Development Environment. Entorno de desarrollo integrado.

IoT Internet of Things, Internet de las Cosas en español.

RGB Red Green Blue. Modelo de color basado en la síntesis aditiva, en la que es posible interpretar colores del espectro visible al ojo humano por medio de la mezcla aditiva de 3 colores primarios.

1

Introducción

El proyecto que se presenta es un sistema interactivo para el control de una matriz de luces de 10x20 unidades, su manejo se realiza por medio de una aplicación móvil. El diseño de este dispositivo se basa en el uso de un microcontrolador Arduino y tiras de diodos emisores de luz (LED) multicolor RGB diseñadas con el circuito integrado WS2811.

1.1. Estado del Arte

La propuesta parte de la idea de controlar una gran cantidad de diodos emisores de luz (LED, de sus siglas en Inglés) individualmente por medio de un solo dispositivo. Debido a su eficiencia, bajo costo y facilidad de uso, este tipo de luces se adecúan a la aplicación que se pretende dar en este proyecto. La elección e implementación parte en base a un panorama de diseño que se presentó desde el momento en que este tipo de dispositivos se inventaron; gracias a su basta cantidad de aplicaciones que ya existen, la búsqueda de una opción adecuada para este proyecto se vuelve intuitiva. Sin embargo, para comprenderlo adecuadamente, primeramente hace falta indagar cronológicamente el proceso de invención y mejora del led.

1.1.1. Introducción a historia de los ledes

Los aparatos capaces de manipular la corriente eléctrica han sido de gran interés desde el mismo momento en que se empezó a utilizar la electricidad para el beneficio del hombre, pues las características y necesidades de cada objeto que consume energía cambian considerablemente, ya sea por su tamaño o por el trabajo que realiza.

Sean generadores o un simple interruptor, la realidad es que existen una gran cantidad de componentes que son utilizados para manipular una corriente eléctrica, cada uno con distintos efectos muy particulares. Con el tiempo, este tipo de técnicas han ido evolucionando para aprovechar las características de la electrónica, y así conseguir resultados más precisos. Aunque si nos remontamos al principio, a su forma más básica, nos encontramos con la electroluminiscencia, fenómeno óptico en el cual la electricidad ocasiona que un objeto emita luz [8]. La electroluminiscencia fue descubierta en 1907 por el británico Henry Joseph Round [9].

A principios del siglo XX, los diodos de vacío se usaban frecuentemente en las radios y en otras tareas que necesitaban rectificadores, pero fue hasta 1950 que se consiguió diseñar un diodo semiconductor que soportaba altos niveles de corriente,

1. Introducción

reemplazando rápidamente al diodo de vacío. Curiosamente, ambos tipos de diodos fueron desarrollados de manera separada en aproximadamente el mismo tiempo [10]. Alrededor de 1927 se reportó la construcción de un tipo de diodo capaz de emitir luz, sin embargo, fueron las investigaciones realizadas por Kurt Lehovec et al. las que finalmente consiguieron interpretar el mecanismo de estos dispositivos en 1951, esto se realizó empleando cristales de carburo de silicio con un generador de impulsos y con una fuente de alimentación de corriente [11].

1.1.2. Primeros tipos de ledes

Procedente a los descubrimientos de un nuevo tipo de dispositivos capaces de emitir luz, en 1955 ya se hablaba de la emisión infrarroja del arseniuro de galio y de otras aleaciones de semiconductores. Fue así como en 1961 se descubrió un tipo de radiación infrarroja de 900 nm procedente de un diodo túnel construido por medio del sustrato de arseniuro de galio, demostrando finalmente dos años después la posibilidad de generar luz eficientemente con semiconductores de este tipo (W. T. Matzen, 1963) [12].

Los descubrimientos anteriormente mencionados permitieron que en 1962 James R. Biard y Gary Pittman, dos investigadores que trabajaban en la compañía tecnológica Texas Instruments, produjeran una patente titulada "Semiconductor Radiant Diode". Este trabajo describía cómo la aleación de zinc encontrada en los diodos de unión n-p permite la emisión de luz infrarroja de manera eficiente [13]. Con la patente concedida por la Oficina de Patentes y Marcas de Estados Unidos, se consideró a los diodos emisores de luz de arseniuro de galio como los primeros productos de uso práctico de este tipo. El primer led comercial vino en 1962 por parte de la misma compañía que desarrolló la patente (Texas Instruments), este dispositivo era capaz de emitir una luz de 890 nm. No obstante, el primer led con luz visible al ojo humano fue desarrollado en 1962 por Nick Holonyak Jr. (considerar que el primero solamente emitía luces infrarrojas) [14]; subsecuentemente, George Crawford, un estudiante de Holonyak, inventó el primer led amarillo, además de mejorar diez veces la luminosidad del primer led rojo y rojo-naranja [15]. Más tarde se empezó a tomar ventaja de distintos materiales semiconductores para desarrollar ledes de mayor intensidad y eficiencia, los cuales tuvieron fuertes aplicaciones en telecomunicaciones, de la misma manera en que se desarrollaron versiones adaptadas específicamente dedicadas a este ámbito.

1.1.3. Avances en diseños modernos

Los primeros ledes de espectro visible al ojo humano normalmente fueron usados como reemplazo para distintos tipos de fuentes de luz de baja intensidad, como las lámparas incandescentes. Debido a su todavía alto costo, hasta finales de la década de los 70's este tipo de luces eran mayormente usadas en equipamiento de pruebas costoso para laboratorio, lo cual decrementaba considerablemente la posibilidad para más aplicaciones potenciales [16]. Los primeros ledes usaban un empaquetado metálico similar al de los transistores, en conjunto a una ventanilla de cristal para permitir ver la luz.

A partir de 1962, dos multinacionales estadounidenses se embarcaron por separado en la investigación y desarrollo de ledes de mayor eficiencia y menor costo. Entre estas investigaciones tenemos los trabajos realizados por el equipo de Howard C. Borden y Gerald P. Phigini bajo la filial de Hewlett-Packard en el que exponían el desarrollo de una pequeña pantalla hecha con semiconductores capaz de mostrar números del 0 al 9, la cual sólo necesitaba de señales tan pequeñas como las de un circuito integrado y solamente de 5 voltios para funcionar [17]. Por otro lado, y como parte de una colaboración con la compañía Monsanto, se realizaron más investigaciones sobre el arseniuro de galio y otros materiales para desarrollar los que se llamarían los "primeros ledes usables" [18]. Fruto de todos estos trabajos, en esa misma década HP lanzó su indicador numérico, mientras que Monsanto una lámpara led, convirtiéndose así en los primeros productos que llevarían a la producción masiva este tipo de dispositivos. Así mismo, con los avances realizados en su proyecto y las constantes mejoras, el indicador numérico de HP se convirtió en toda una revolución, lo que terminó sentando las bases para futuros dispositivos de este tipo [19].

Más tarde se vieron más avances en ledes de distintos colores, como el azul, el cual usaba nitruro de galio dopado con magnesio; fue desarrollado por Herb Maruska y Wally Rhines en 1972, dos estudiantes de doctorado de la Universidad de Stanford [20], hasta el día de hoy este tipo de material continua siendo la base para de los ledes azules comerciales. Varios años más tarde, un led azul de alta intensidad fue expuesto por Shuji Nakamura, un investigador de la Corporación Nichia [21].

Con el alcance al primer diseño eficiente del led azul, rápidamente vino el desarrollo del primer led blanco. Inicialmente, y a pesar de que es posible generar una luz blanca con los colores individuales de los ledes rojo, verde y azul, los resultados no son satisfactorios ya que se están utilizando tres distintas ondas de luz de baja amplitud. Aunque en un principio eran caros e inefficientes, los ledes de color blanco se han vuelto prácticos, convenientes y de bajo costo. Actualmente existen los llamados ledes RGB (del Inglés, Red Green Blue), los cuales son capaces de mostrar una gran cantidad de colores del espectro de luz visible con tan solo combinar la intensidad de cada color en un solo encapsulado. Este tipo de ledes ya se utilizaban en 1995 en aplicaciones con pantallas [22].

1.1.4. Ledes de direccionamiento individual

Las matrices de ledes se han estado utilizando desde hace mucho tiempo, tal como se pudo ver con el numerador de Hewlett-Packard. En estos días y debido a su alta accesibilidad a todo el público, se pueden ver variados diseños de matriciales innovadores, como el cubo para mostrar animaciones en 3 dimensiones diseñado por el usuario chr en 2019 [23], el cual no es más que el resultado de varios de años de perfeccionamiento de las técnicas con multiplexores en ledes. Este tipo de diseños presentan una gran versatilidad y velocidad para mostrar imágenes, funcionando como una pantalla de baja resolución. No obstante, a pesar de la apartente ventaja que presentan los multiplexores, para la aplicación que se busca en este proyecto no es la solución más práctica. Como se puede observar en el diseño del usuario chr, este tipo de configuración requiere de una gran cantidad de salidas individuales hacia cada led, lo cual termina convirtiéndose en un sistema complicado de instalar

y mantener debido a la ingente cantidad de cableado que aumenta exponencialmente así se vaya añadiendo más unidades al modelo.

En busca de una técnica que permitiera conectar muchos ledes individuales sin requerir de conexiones externas complejas, se crearon los **ledes de direccionamiento individual**. Este tipo de dispositivos se empezaron a ver tan tempranamente como en 1985 gracias al trabajo de P. Deimel et al., quienes presentaron el diseño de un circuito con doce ledes capaces de ser direccionados de manera individual [24]. Más tarde aparecieron diseños de mayor rendimiento y capacidades de aplicación, como el demostrado por H. X. Zhang et at. en 2006 con su artículo sobre diodos emisores de luces ultravioleta de hasta 120 unidades [25]. Finalmente, y con el rápido avance de esta tecnología, nos encontramos con uno de los dispositivos más comunes y utilizados en materia de ledes comerciales, el circuito integrado WS2811 [26]. Este circuito integrado permite controlar ledes RGB de manera individual, a la vez que sus salidas y entradas se conectan a otras unidades del mismo dispositivo en serie, creando de esta manera un circuito expandible en el que cada pieza es casi completamente independiente para recibir enviar datos en serie.

1.2. Problemática

La creación de este proyecto nace de la necesidad de ofrecer una alternativa barata a productos comerciales con las mismas funcionalidades, particularmente una pared interactiva de ejercicios de escalado que utiliza ledes para crear patrones de rutinas de entrenamiento [27].

El producto comercial mencionado y otros de similar naturaleza ofrecen las partes necesarias para su instalación, particularmente el módulo de control y las luces que se usan en la pared. Su instalación es sencilla, y ofrecen las herramientas adecuadas para que el cliente empiece a utilizarlo rápidamente con tan solo conectar las tiras de ledes al dispositivo que se comunica a una aplicación móvil para su manejo. La gran desventaja de este aparato es su precio; llegando a costar hasta 500 dólares estadounidenses, y sumando al precio de los materiales para la instalación de la pared, éste no es un producto fácilmente accesible para cualquier personas.

Lo que se requiere, entonces, es un dispositivo de bajo costo que sea capaz de ofrecer las mismas funcionalidades que el módulo de control explicado anteriormente, así como la aplicación de teléfono móvil que permita controlarlo de manera sencilla.

1.3. Justificación

Analizando las funciones del módulo de control, se deduce rápidamente que es posible replicarlo con la tecnología que se encuentra disponible al público. Este dispositivo se basa en el uso de ledes de direccionamiento individual y un microcontrolador para resolver las instrucciones que recibe de la aplicación móvil.

Actualmente existen una gran cantidad de herramientas para diseñar y programar un circuito de esta naturaleza, así mismo, las piezas tienen una alta disponibilidad en el mercado de electrónicos, ofreciendo muchas versiones de distinta calidad y

efectividad, por lo que el precio de toda su construcción es potencialmente mucho menor al que piden los creadores de la pared de escalado interactiva.

Por otro lado, gracias a lo común que son este tipo de aplicaciones, hay repositorios públicos en Internet que facilitan considerablemente la programación de los ledes de direccionamiento individual, por lo que la etapa de programación se ve considerablemente reducida.

1.4. Objetivos

1.4.1. Objetivo general

Crear un módulo de bajo costo y fácil uso para el control de una matriz de ledes RGB que utilicen circuitos integrados para permitir el direccionamiento individual.

1.4.2. Objetivos específicos

Para lograr el objetivo específico general es necesario realizar los siguientes objetivos:

- Diseñar un módulo de control que sea capaz de manipular la intensidad y color de una tira de ledes de hasta 200 unidades.
- Diseñar el algoritmo y programa para un microcontrolador que manipule hasta 200 ledes de direccionamiento individual.
- Implementar un protocolo de comunicación entre una aplicación móvil y el módulo de control.
- Diseñar una aplicación móvil capaz de indicar al módulo de control qué unidades de la matriz deben apagarse y encenderse.

1.5. Hipótesis

Este módulo de control tiene un costo reducido en comparación a productos comerciales de similar naturaleza, sin sacrificar las funciones básicas que permiten satisfacer con lo ofrecido por el producto original. La calidad de construcción y efectividad del dispositivo permitirá conectarlo a tiras de ledes largas sin requerir de una configuración complicada.

1.6. Metodología

Para este proyecto se utilizará la metodología de Diseño de electrico propuesto por Carlos J. Jiménez Fernández [28], la cual se basa en las siguientes etapas:

1.6.1. Especificaciones de diseño

Este paso consiste las especificaciones de diseño del circuito, describiendo las funcionalidades que se proyectan, así como considerar otros datos que definirán el resultado final, como las dimensiones, consumo de potencia, entre otros.

1.6.2. Creacion de esquemáticos

Subsecuenta a las especificaciones, un método común es el de diseño de transistores y operacionales por medio de esquemáticos, lo cual no es más que una manera gráfica de describir las conexiones de cada uno de los componentes involucrados en el diseño del circuito. Esto debe incluir las fuentes de alimentación, y todos los conectores de entrada y salida.

1.6.3. Simulación eléctrica

Para comprobar el funcionamiento eléctrico del circuito, es necesario realizar simulaciones para detectar errores que se pudieron haber presentado en la etapa de diseño. En este paso, además, es extremadamente importante que los resultados se comprueben detallamente para que se adecúen a todo lo expuesto en las especificaciones. De no cumplirse adecuadamente con esta etapa, se vuelve a las especificaciones de diseño para realizar los cambios adecuados.

1.6.4. Montaje

Tal como indica su nombre, en esta etapa se construye y monta el prototipo diseñado, utilizando cada una de las piezas mencionadas en la etapa de diseño.

1.6.5. Pruebas experimentales

Finalmente, se procede a comprobar el funcionamiento general del circuito por medio de pruebas experimentale exhaustivas que determinen la calidad de los resultados, comparándolos cons especificados al principio del proceso de diseño. De no obtener los resultados esperados, se vuelve a la etapa de simulación eléctrica para comparar las discrepancias encontradas.

2

Marco teórico

La teoría del funcionamiento de este proyecto empieza, principalmente, con las características que aporta el circuito integrado WS2811, el cual forma parte de una familia dispositivos para la construcción y diseño de ledes de direccionamiento individual.

Antes de explicar los conceptos introducidos, y como se explicó antes, es necesario comprender la necesidad que se presenta: controlar una matriz de 10 columnas y 20 filas. Ya que la cantidad de dispositivos es enorme (200 unidades), se vuelve imposible conectarlos individualmente a un solo microcontrolador sin tener que diseñar sistemas multiplexores complejos, lo cual se complica aún más considerando las cuestiones para la alimentación de corriente y la cantidad de pines que tienen los diodos RGB (Red Green Blue, lo cual se refiere a un estándar de color).

Por otro lado, Arduino es una de las plataformas de desarrollo más populares para microcontroladores que existen actualmente, por lo que existen una gran diversidad de librerías y componentes compatibles. En base a esto se considera que cualquier placa compatible con Arduino es una opción viable.

El desarrollo de este sistema se basa en 3 componentes principales:

1. Ledes de direccionamiento individual.
2. Microcontrolador que funcione con la plataforma Arduino.
3. Aplicación móvil para Android.

2.1. Ledes de direccionamiento individual

En esta sección se explica el funcionamiento detrás de los ledes de direccionamiento individual.

Precisamente, los LEDs de direccionamiento individual dan una solución al problema que se presenta. Este tipo de luces contienen un pequeño chip controlador, el cual tiene entradas y salidas para comunicación en serie (Figura 2.1), permitiendo conectar una gran cantidad de piezas en una sola línea o tira. De esta manera se pueden controlar una gran cantidad de luces utilizando sólo unos pocos cables, en vez de necesitar una conexión para cada una [1] [29].

El chip WS2811 es uno de los más populares actualmente (superado por el WS2812b). Únicamente tiene 3 cables entre cada LED (voltaje, tierra y datos), además de que el circuito integrado normalmente va dentro del encapsulado, lo cual lo vuelve compacto.



Figura 2.1: Dispositivo LED RGB con circuito integrado [1].

Para su funcionamiento, el WS2811 almacena 3 bytes de información (24 bits), uno para cada uno de los colores RGB, es decir, 256 niveles de intensidad para un total de más 16 millones de combinaciones. Los datos que sobran en un chip son enviados al siguiente por medio de su salida. Esto vuelve cada LED independiente del siguiente, por lo que incluso si se divide en fragmento una tira seguirá funcionando sin problemas. El montaje con una placa Arduino es muy sencillo y similar entre esta familia de dispositivos, requiriendo nada más que un par de conexiones para controlar una tira de LEDs (mirar las figuras 2.2 y 2.3) [2] [3].

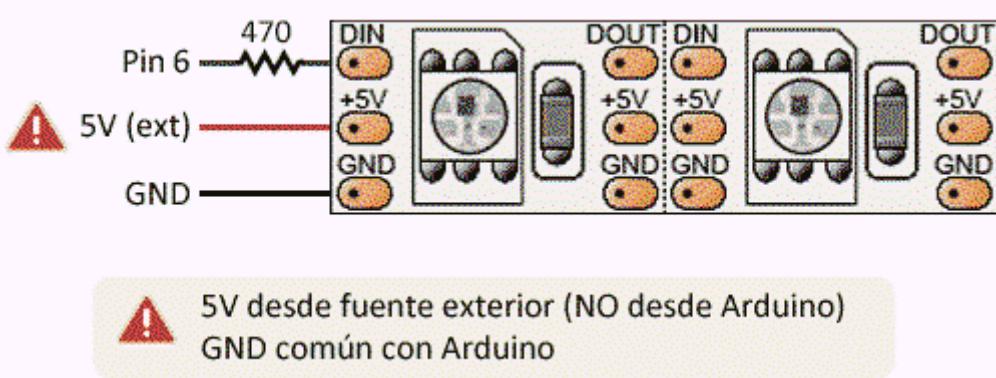


Figura 2.2: Esquema de montaje para un LED WS2812b. El puerto DIN es para la comunicación serial [2]

Otra de las ventajas de estos dispositivos es su popularidad en el mercado. A través de tiendas en Internet, es posible comprarlos a precios tan bajos como \$15 US por 50 piezas [?]. Por otro lado, existen diversas librerías que facilitan muchísimo su uso, evitando tener que preocuparse por comprender a fondo su funcionamiento, una de ellas es FastLED. Disponible para muchos de los microcontroladores más comunes, esta librería permite controlar la mayoría de los chips para LEDs más comunes, a la vez que ofrece funciones complejas pero sencillas de usar (según se requiera) [30].

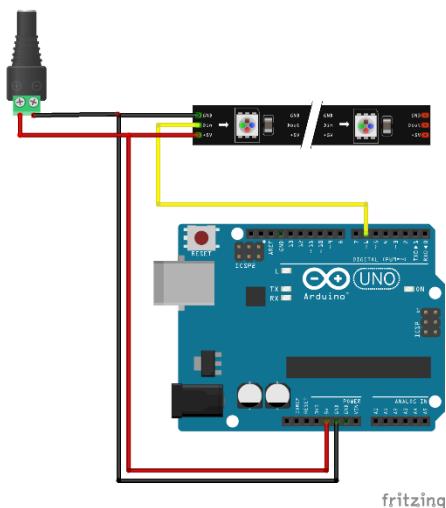


Figura 2.3: Esquema de montaje para un LED WS2812b con una placa Arduino [3].

Otro punto importante que cabe resaltar, son las diferencias de las luces con WS2811 y WS2812 (las cuales son las más populares actualmente). Esto es una cuestión del empaquetado. Las tiras que utilizan el WS2812 son planas con todas las luces apuntando en la misma dirección (como si fueran una cinta adhesiva) ya que utilizan un LED RGB 5050, el cual tiene esa forma particular, cada pieza separada por una distancia específica; el LED WS2812 utiliza el mismo circuito integrado que el WS2811. Por otro lado, las tiras del WS2811 son similares al encapsulado de LED tradicional (Figura 2.4) [31].



Figura 2.4: Encapsulado de un LED RGB 5050.

Finalmente, otro punto muy importante de estos dispositivos es su capacidad de escalado, es decir, la facilidad que tienen para conectarse muchos al mismo tiempo. La mayoría de las versiones comerciales vienen con un puerto de entrada y otro de salida en los extremos de las tiras, permitiendo “encadenar” varias con tan solo unir la salida de uno y la entrada de otro, tal como se puede observar en la Figura 2.5.



Figura 2.5: Dos tiras de LEDs de direccionamiento individual conectadas en serie [4].

Cada una de las características descritas vuelven al WS2811 en una opción viable para diseñar luces con muchísimas piezas sin la necesidad de cableados complicados o una programación difícil de manejar, a la vez que se mantiene un precio relativamente bajo. Así mismo, el tipo de encapsulado de este tipo de LEDs se adecúa a las necesidades del proyecto presente, ya que se necesita separar cada pieza varios centímetros para ajustarse al tamaño de una pared de escalado, lo cual es alrededor de 20 cm para la pared de entrenamiento de escalado comercial.

2.2. Microcontrolador ESP8266

Es posible encontrar una gran variedad de productos dentro del mercado de microcontroladores, cada uno variando en características, tamaños y precios.

Empezando por las más comunes para prácticas de laboratorio, están las placas Arduino, las cuales están construidas sobre los chip Atmega [32], con Arduino UNO y Arduino MEGA. Sin embargo, estos microcontroladores normalmente incluyen muchos conectores y funciones que probablemente no se utilicen pues están hechos para utilizarse en prototipos de laboratorio, siendo el factor más importante el tamaño.



Figura 2.6: Chip del microcontrolador ESP8266 [5].

Aunque es posible adquirir solamente el chip del microcontrolador (Figura 2.6), también existen placas integradas de tamaño disminuido, como Arduino Mini y, particularmente, el ESP8266 (Figura 2.7). Este último es uno de los microcontroladores más populares y de bajo costo en el mercado, el cual ronda solamente los \$2 dólares estadounidenses.

El ESP8266 viene integrado con un módulo WIFI, por lo que normalmente se usa como módulo de comunicación, a pesar de esto también es posible darle la función de microcontrolador. Se alimenta con 3.3v y cuenta con un procesador Tenisilica Xtensa LX106 de 80 Mhz y una memoria RAM de 64 KB para datos y otros 96 KB para datos. Así mismo, cuenta con pines dedicados para los protocolos de comunicación UART y las interfaces SPI y I2C. Esto último es importante para el uso de módulos externos como el de comunicación Bluetooth [33].



Figura 2.7: Placa con el microcontrolador ESP8266 montado [5].

Finalmente, la plataforma de desarrollo Arduino ofrece compatibilidad con este chipset, por lo que es posible usar casi cualquier variedad del ESP8266 en conjunto a las librerías FastLED para facilitar el manejo de los ledes de direccionamiento individual. Gracias a esto, la codificación se vuelve muy similar al que se utilizaría en placas Arduino.

2.3. Módulo de comunicación Bluetooth

Una parte importante para el control del sistema es el tipo de comunicación a utilizar, ya que de esto depende la efectividad y facilidad con la que se transmiten los datos con la aplicación de que utiliza el usuario.

Un protocolo de comunicación muy conocido y que está disponible en muchos dispositivos inteligentes es el Bluetooth, el cual está especialmente diseñado para dispositivos de bajo consumo que requieran corto alcance de emisión. Precisamente, esto se adecúa a las necesidades del proyecto.

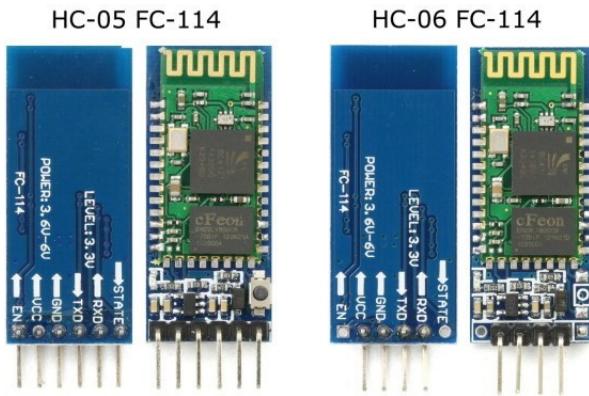


Figura 2.8: Módulos Bluetooth para microcontroladores [6].

Existen varios módulos de comunicación Bluetooth en el mercado para utilizar con placas Arduino u otro tipo de microcontroladores (Figura 2.8), incluso algunas lo tienen incluido [6]. Módulos como el HC-05 no requieren de ninguna configuración complicada pues cada una de sus terminales se puede conectar directamente a los puertos de, por ejemplo, una placa con el chip ESP8266 (Figura 2.9). Como se basa en comunicación serial, es compatible prácticamente con cualquier controlador.

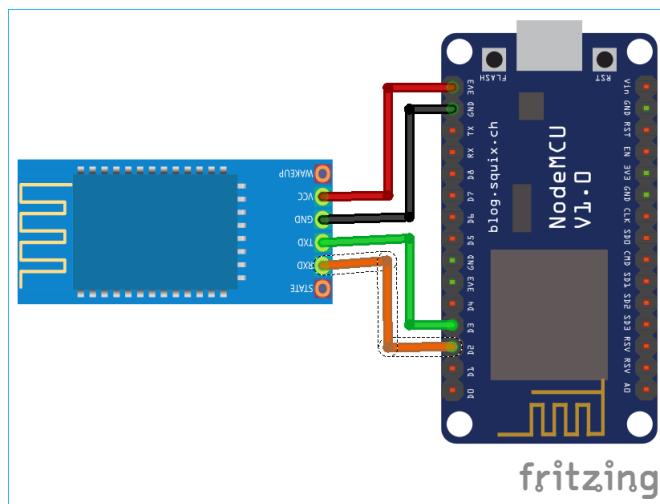


Figura 2.9: Diagrama para la conexión del módulo Bluetooth HC-05 con una placa ESP8266 [7].

2.4. Aplicación en Android

Diseñar una aplicación móvil que sea eficiente y fácil de usar puede ser un reto considerable si no se le dedica el tiempo adecuado. Para facilitar la solución de este problema existen distintas herramientas (IDE), entre las que destaca el MIT APP Inventor, una plataforma de desarrollo para el sistema operativo Android, el cual se basa en un entorno visual para simplificar el proceso de programación y diseño. Este software, desarrollado por Google Labs, siendo gratuito permite cubrir un gran número de necesidades en un dispositivo móvil. Además de su simplicidad de uso, permite distribuir las aplicaciones creadas fácilmente [34].

2. Marco teórico

3

Métodos

El marco metodológico es la explicación de los mecanismos utilizados para el análisis de nuestra problemática de investigación. Por lo general, se trata del tercer capítulo del proyecto y es el resultado de la aplicación, sistemática y lógica, de los conceptos y fundamentos expuestos en el marco teórico. Es importante comprender que la metodología de la investigación es progresiva, por lo tanto, no es posible realizar el marco metodológico sin los fundamentos teóricos que justifican el estudio de los conceptos que rodean el proyecto.

En este capítulo se especifica la aplicación de los conceptos vistos en el Marco Teórico, esto es, la composición de cada una de las partes del sistema basado en el propósito general y específico del proyecto. Se desglosan detalles técnicos genéricos, además de los flujos de funcionamiento y uso del sistema general y los subsistemas que lo componen, ya sean físicos o programados.

Partes de este capítulo están basadas en la metodología expuesta por los autores Arshdeep y Vijay (2014) en su libro 'Internet of Things: A Hands-On Approach' [35].

3.1. Especificación de propósitos y requerimientos

En esta sección se describen el propósito del sistema, así como su comportamiento y lo que requiere para lograrlo (recolección de datos, análisis, manejo del sistema, etc.).

- **Propósito:** Un módulo de control para el apagado y encendido de luces led de manera remota por medio de una aplicación móvil.
- **Comportamiento:** El módulo de control debe ejecutar una prueba de funcionamiento automática de todos los ledes al encenderse. Después de realizar la prueba, debe esperar a conectarse con un dispositivo, para finalmente esperar a recibir instrucciones por comunicación inalámbrica desde la aplicación móvil. El módulo de control provee interruptor de encendido y un botón de reinicio.
- **Requerimientos para el mantenimiento del sistema:** Además de la prueba de funcionamiento, el módulo de control debe tener indicadores para encendido y la transmisión de datos con la aplicación móvil.
- **Requerimiento para el análisis de datos:** No se realiza análisis de datos.
- **Requerimientos para la ejecución de la aplicación:** La aplicación se debe ejecutar localmente en un dispositivo móvil y enviar instrucciones al módulo

de control, el cual es un dispositivo independiente.

- **Requerimientos de seguridad:** Para conectarse al módulo de comunicación inalámbrica se requiere de autenticación por medio de un código.

3.2. Especificación de procesos

Esta sección es para explicar el flujo del funcionamiento general del sistema.

Primero se realiza la prueba de funcionamiento inicial, en la que se trata de encender en secuencia cada uno de los ledes conectados al módulo de control. Independientemente de si prendieron todas las luces, el sistema las apaga todas y procede a esperar la conexión con la aplicación móvil hasta que se complete. Enseguida trata de recibir instrucciones, las cuales consisten en un indicador de posición para un led y la intensidad de luminosidad que se le debe aplicar, no es necesario especificar .^apagado.^o .^encendido^oza que la intensidad mínima se considera apagado y cualquier otra mayor es encendido. Finalmente el sistema aplica los valores recibidos al led especificado y vuelve a esperar más instrucciones (mirar la Figura 3.1).

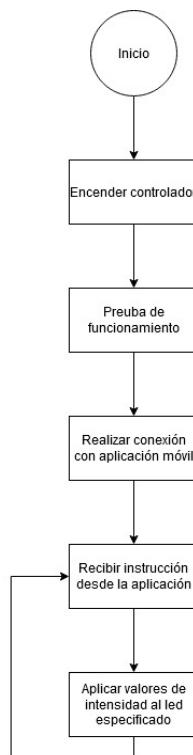


Figura 3.1: Especificación de procesos para el sistema de control del SunBoard Led System.

No hay un final específico al flujo ya que se basa en el tipo de funcionamiento perpetuo de los sistemas empotrados, en el que siempre está realizando procesos hasta que se corta el suministro de corriente.

3.3. Modelo de dominio

Este apartado es para describir el modelo de dominio del sistema. Este modelo define los atributos de los objetos que conforman el sistema, y la relación entre ellos. Es una representación abstracta de los conceptos, objetos y entidades, independientemente de la tecnología o plataformas utilizadas.

A continuación se describen cada uno de los elementos mostrados en el diagrama de la Figura 3.2.

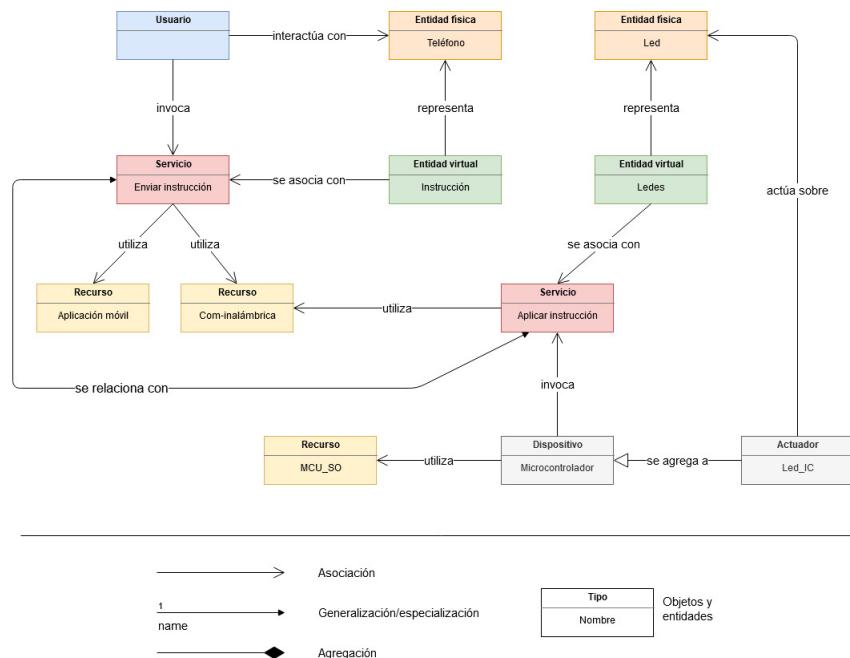


Figura 3.2: Modelo de dominio del SunBoard Led System.

Entidades físicas:

- **Led:** Unidad perteneciente al conjunto de luces que controla el usuario.
- **Teléfono:** El dispositivo con el que interactúa el usuario para controlar el funcionamiento del microcontrolador.

Entidades virtuales:

- **Ledes:** Representación virtual de los ledes dentro del sistema.
- **Instrucción:** Es la representación de la información que envía el usuario al microcontrolador. Contiene los datos necesarios para modificar el estado de los ledes.

Dispositivos:

- **Microcontrolador:** El dispositivo que manipula las luces de acuerdo a la información recibida desde el módulo de comunicación.
- **Led_IC:** Actuador que manipula directamente la intensidad de los ledes y se conecta al Microcontrolador.

Recursos:

- **MCU_SO:** El sistema operativo del microcontrolador.
- **Aplicación móvil:** El sistema que trabaja sobre el teléfono para la interacción con el usuario.

- **Com-inalámbrica:** Recurso de red para transmitir información de manera inalámbrica.

Servicios:

- **Enviar instrucción:** La acción de enviar la información de los ledes que se están intentando apagar/encender.
- **Aplicar instrucción:** La acción de recibir la información sobre los ledes que se desean apagar/encender y aplicar dicha información a un led en particular.

3.4. Modelo de información

El modelo de información define la estructura de toda la información en la estructura del sistema, particularmente, los atributos de las entidades virtuales, sus relaciones, etc. Este modelo no describe cómo la información expuesta se almacena o se representa.

En el caso de este proyecto, solamente se utiliza una entidad virtual para representar las luces, para las cuales sólo se manipula la intensidad lumínica. A pesar de que, idealmente, se utilizan luces RGB, en donde se especifican atributos para cada uno de los 3 colores, en el caso de este modelo no es necesario más que especificar que se está trabajando con la intensidad de las luces y un identificador, además de instrucciones que contienen esta información (observar la Figura 3.3).

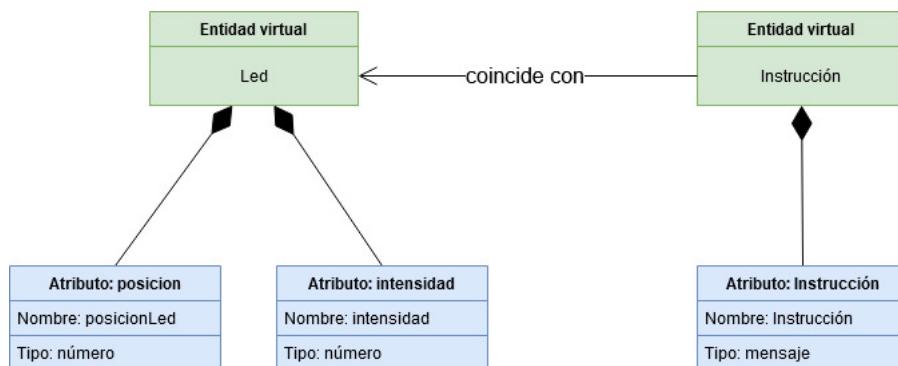


Figura 3.3: Modelo de información del SunBoard Led System.

3.5. Especificación de servicios

Esta parte de la metodología de diseño consiste en definir los servicios dentro de un sistema IoT (Internet of Things), tipos de servicios, entradas y salidas, *endpoint*, rutinas de servicios, pre-condiciones y la forma en que afectan. La especificación de servicios de SunBoard se puede observar en la figura 3.4.

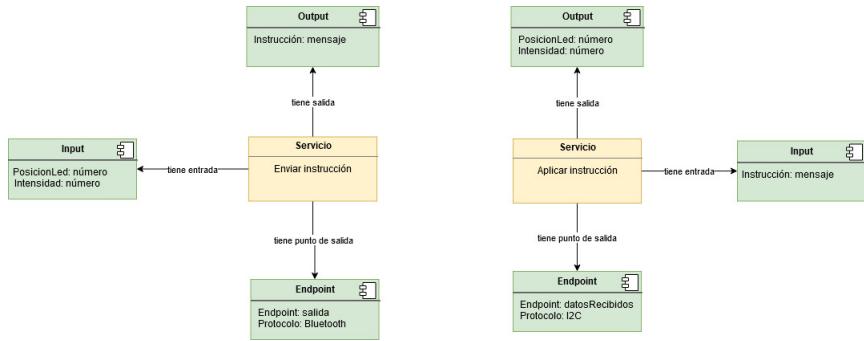


Figura 3.4: Especificación de servicios del SunBoard Led System.

3.6. Especificación de los niveles IoT

Dentro de los sistemas IoT se definen niveles para explicar la implementación de las distintas partes de dicho sistema. Estas partes se encargan de distintas tareas específicas (comunicaciones, seguridad, protocolos, etc.) y se comunican entre ellas. Los niveles IoT pueden ser tanto locales como en la nube, es decir, almacenándose en servidores externos. En el caso del SunBoard, esto es relativamente sencillo: la información viaja de manera local desde la aplicación que utiliza el usuario, se transmite por protocolos de comunicación hasta llegar al controlador que interpreta esta información y la aplica al dispositivo (ledes), tal como se observa en la figura 3.5.

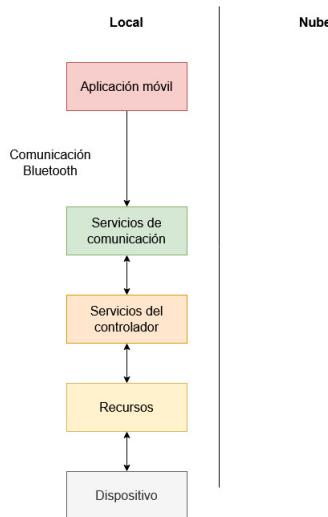


Figura 3.5: Especificación de servicios del SunBoard Led System.

3.7. Especificación de la vista funcional

En esta sección se definen las funciones del sistema agrupadas de distintas maneras (*Functional Groups* o FG). Cada grupo ofrece características necesarias para la interacción con instancias de conceptos definidos en el Modelo de Dominio, así como ofrecer información relacionada con estos conceptos. Revisar la figura 3.6.

3. Métodos

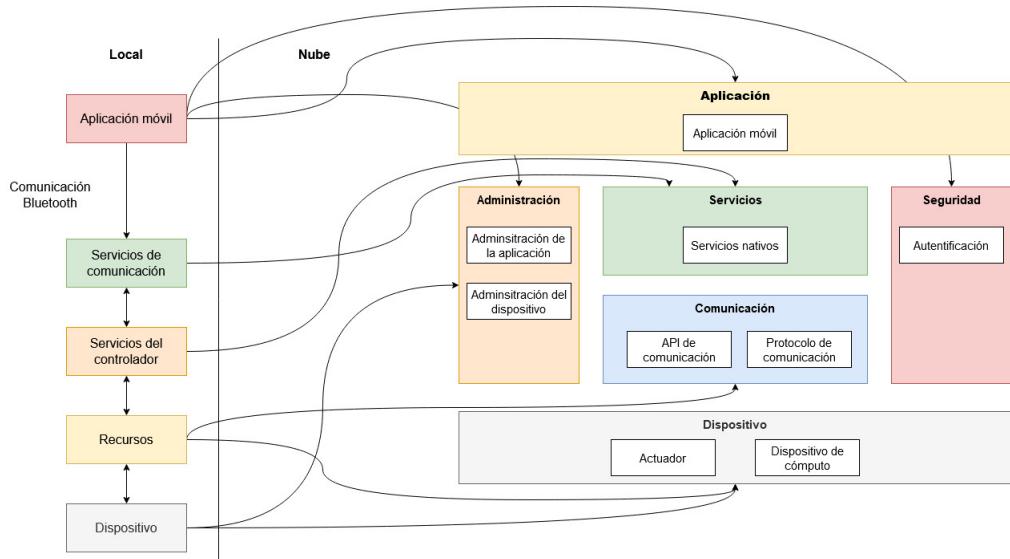


Figura 3.6: Vista funcional del SunBoard Led System.

4

Resultados

Concordé a los diagramas mostrados en la metodología y a la información del marco teórico, el siguiente paso es especificar los diseños y recursos a los que se llegó para utilizar en el proyecto, para finalmente mostrar el resultado de sus implementación y la construcción del dispositivo, así como la programación y recopilación de datos obtenidos de los prototipos creados, sean satisfactorios o no.

En esta sección se presentan, pues, la explicación y discusión de cada una de las partes que conforman el SunBoard Led System, incluyendo esquemáticos, gráficas, y/o tablas para dar explicación a procedimientos o experimentos realizados. Con este capítulo se culmina lo visto en los anteriores dos.

4.1. Especificación de la vista operacional

Continuando con la metodología de diseño vista en el Capítulo 3, en este paso se definen varias opciones pertinentes a un sistema IoT y su implementación, esto es, la especificación de dispositivos, recursos para el almacenamiento de información, aplicaciones, software para servicios, etc.

La figura 4.1 muestra los elementos que se utilizan para el SunBoard Led System:

4. Resultados

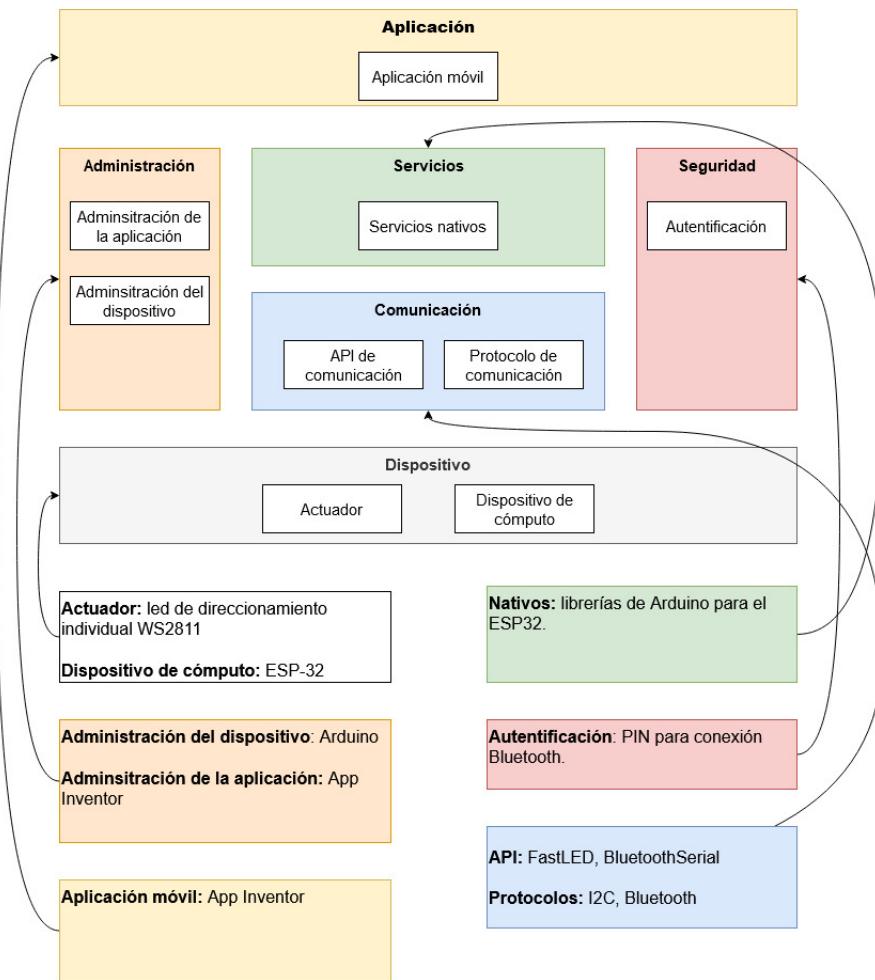


Figura 4.1: Vista operacional del SunBoard Led System.

- **Dispositivos:** Microcontrolador ESP-32 (dispositivo de cómputo), ledes de direccionamiento individual WS2811 (dispositivo y actuador).
- **Comunicación:**
 1. APIs de comunicación: BluetoothSerial, FastLED
 2. Protocolos de comunicación: Bluetooth, I2C
- **Servicios:** librerías nativas de Arduino para el chip ESP32.
- **Aplicación:** Aplicación móvil hecha por medio del software de diseño App Inventor.
- **Seguridad:** Autentificación en la comunicación Bluetooth por medio de PIN.
- **Administración:** Arduino (administración del dispositivo) y App Inventor (administración de la aplicación).

4.2. Disolución de las limitaciones de diseño

Antes de continuar con las especificaciones del dispositivo, cabe explicar varios puntos críticos para asegurar la calidad del prototipo, particularmente en cada una de las limitantes que se presentan entre los distintos componentes y cómo éstos

interactúan entre ellos, como ancho de banda para transmisión de dato y consumo eléctrico.

4.2.1. Consumo de las luces

De acuerdo a la información expuesta en el portal de Pololu.com [4] a la hoja de datos del WS2811 [12], cada led a máxima intensidad de brillo consume alrededor de 60 mA con 5 V de alimentación. Esto se debe a que, internamente, cada encapsulado consiste en 3 luces, una para cada color (rojo, verde y azul), con un consumo máximo de 20 mA cada una.

Si consideramos una tira de 50 piezas, el consumo de corriente máximo es de 3 amperios. Esto es bastante considerable ya que se estará utilizando hasta 200 ledes al mismo tiempo, lo que nos daría un consumo de hasta 12 amperios en total. Una fuente de alimentación común y compacta normalmente ofrece hasta 2 amperios, lo cual se alega bastante de lo que se necesita. Utilizar una fuente más grande trae consigo otra serie de problemas, como el costo y el tamaño.

El funcionamiento de este tipo de dispositivos se basa en la modulación por ancho de pulsos (PWM) para alimentarse, lo cual es, en pocas palabras, controlar el porcentaje de tiempo en el que la corriente fluye. Al reducir la frecuencia del PWM, se reduce la cantidad de corriente consumida (ya que disminuye la cantidad de tiempo efectiva en que se suministra energía). Para permitir el uso de fuentes baratas, el controlador debe limitar el brillo de los ledes.

4.2.2. Frecuencia de actualización

Otra cuestión es la velocidad de actualización. En primera, los ledes de este tipo no permanecen estáticos todo el tiempo, sino que se actualizan constantemente con nuevos valores cierta cantidad de veces por segundo. *Frames per second* (FPS) es el término más común para referirse a la cantidad de veces que se actualiza una imagen cada segundo. Este proceso es tan rápido que el ojo humano no puede percibir el cambio, al menos desde 30 FPS en adelante, por debajo de este valor se empieza a notar el parpadeo de la luz.

Se requieren 3 bytes de información por led [26], a mayor cantidad de piezas mayor es la cantidad de información que se requiere por segundo. Ahora, en una situación hipotética en la que se están usando 1000 ledes con una frecuencia mínima de 30 FPS, se estará escribiendo la información de 30,000 ledes cada segundo (90,000 bytes), esto es alrededor de 16.6 μ s por led. De acuerdo a sus especificaciones, el chip WS2812 le toma 30 μ s para mover 3 bytes de información, esto significa que cada segundo se puede actualizar más de 33,333 de estos dispositivos; sin embargo, si se considera que el procesador del controlador pasa 50 % del tiempo calculando las animaciones, se tiene que se pueden actualizar alrededor de 16,666 ledes cada segundo. Esto no es una limitación para el enfoque de este proyecto (solamente 200 ledes), por lo que no se requiere realizar un cambio sustancial.

4.2.3. Cableado

Sin embargo, algo que sí podría ocasionar problemas es el tamaño de las tiras. Entre más grande sea una línea de ledes, mayor es la pérdida de voltaje en sus extremos. Por ejemplo, una tira de 60 luces tiene una caída de 0.8 V, mientras que una de 150 pierde 2.0 V en su salida [4]. Es por esto que no se recomienda tiras mayores a 180 ledes, para las cuales es mejor utilizar fuentes de alimentación separadas para cada una. Una pérdida de voltaje ocasiona menor intensidad de brillo y malfuncionamiento, por lo que es importante solucionar esto.

Si se considera una sola fuente que es capaz de alimentar toda la tira completa, entonces se podrían **conectar en paralelo** partes más pequeñas, mientras que la línea de datos sigue siendo en serie.

Finalmente, una recomendación común en este tipo de circuitos es conectar un resistor de entre 100 ohm y 500 ohm entre el microcontrolador y la tira de LEDs para reducir el ruido de la señal de comunicación. También es posible conectar un capacitor entre las líneas de tierra y voltaje de la fuente para reducir aún más el ruido que pueda causar una mala fuente de alimentación.

4.3. Implementación de los dispositivos y componentes

El funcionamiento general del sistema se compone de varios elementos físicos y digitales que permiten la manipulación por parte del usuario. Este apartado es para describir la relación entre dichos elementos y constituye como la parte 9 de la metodología expuesto en el capítulo 3. Con lo expuesto hasta ahora, es posible presentar un diagrama para simplificar la configuración de los componentes, tal como se muestra en la figura 4.2.

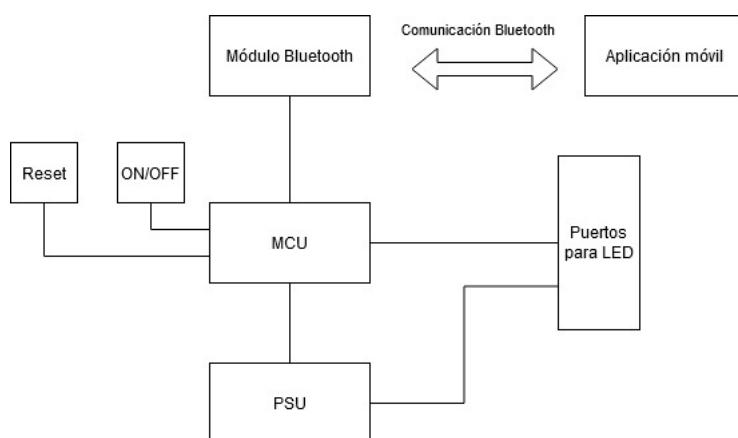


Figura 4.2: Diagrama del diseño eléctrico de los componentes del SunBoard.

- **MCU:** Es en donde ocurre la mayoría de la lógica del sistema (microcontrolador). Recibe información desde el módulo de conexión inalámbrica y se conecta directamente a los ledes.

- **Tiras de ledes:** Son los arreglos de ledes que conforman la matriz completa. Deberían ser externas al encapsulado del módulo de control.
- **PSU:** Fuente de alimentación. Se conecta por separado al microcontrolador y a las tiras de ledes. Esto es necesario ya que el consumo de estos últimos dos dispositivos es considerablemente distinto, por lo que es necesario aislarlos o podría dañarse el microcontrolador.
- **Módulo Bluetooth:** El dispositivo de enviar y recibir instrucciones por medio del protocolo de comunicación Bluetooth.
- **Interfaz de control:** Constituye la aplicación móvil por medio de la cual el usuario puede manipular el prendido y apagado de los ledes.
- **Reset:** Botón para reiniciar el microcontrolador.
- **ON/OFF:** Botón para controlar el paso de la corriente a todo el módulo de control.

Aunque en el diagrama el módulo Bluetooth se presenta como un componente externo, la realidad es que existen microcontroladores con este tipo de funciones ya implementadas en el chip, tal como ocurre con el ESP-32, el cual, además, cuenta con compatibilidad con WIFI [36].

4.3.1. Esquema

Dentro del diseño del esquema que se muestra en la figura 4.3, se puede observar que se agregaron varios componentes no mencionados anteriormente. El primero y más notable es el *barrel jack*, el cual es necesario para poder conectar una fuente de alimentación externa, de esta forma se asegura el tamaño compacto del dispositivo, por supuesto, es importante también asegurarse de que la fuente sea de 5V y 3A como mínimo.

Entre las terminales del interruptor de encendido y la fuente de corriente se conectó un capacitor de 100 μ F. Esto es necesario para evitar el ruido. De la misma manera se colocó un diodo emisor de luz (led) para que funcione como indicador de encendido. El botón reset, por otro lado, nos permite reiniciar el microcontrolador directamente en caso de que ocurra un problema.

Como se puede observar a la derecha del esquema, hay 3 pares de conectores. Un par va a voltaje, otro a tierra y un solo conector va directo a uno de los piensos del microcontrolador. Estas terminales son para las tiras del LED WS2811. Hay dos pares de conectores de alimentación ya que se pretende que se conecten dos tiras de 100 ledes cada una, de esta forma se evita la caída de voltaje que ocasionaría una sola tira de 200 piezas. Se está usando solamente una entrada para datos debido a que las dos tiras se pueden interconectar en serie.

Una parte importante para el mantenimiento del dispositivo ante fallas es utilizar indicadores visuales, particularmente ante dos situaciones: cuando hay corriente eléctrica y cuando se están transmitiendo datos. Tal como se hizo con el indicador de encendido, hay otro led para indicar cuándo hay transferencia de datos de manera inalámbrica (por Bluetooth).

4. Resultados

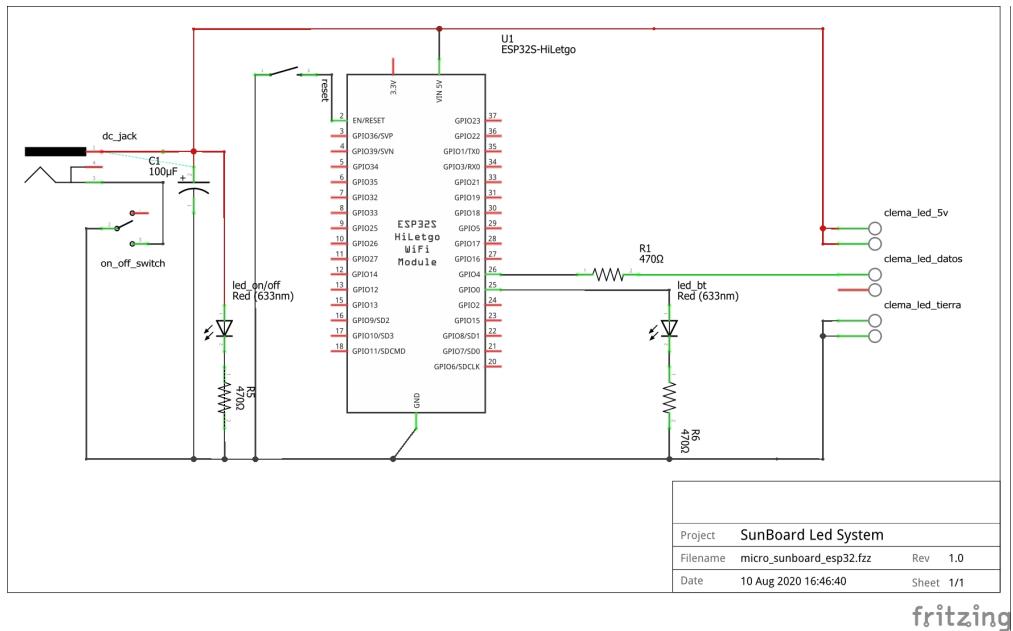


Figura 4.3: Esquemático del SunBoard.

A continuación se mencionan cada uno de los componentes incluidos en el esquemático.

- **ESP32:** Es el bloque de procesamiento central del dispositivo, específicamente, el modelo ESP32 DevKit 1 de Espressif [37].
- **Clemas:** Funcionan como conectores para dos tiras de ledes de 50 piezas cada una. El conector de datos va en serie con ambas tiras.
- **C1:** Un capacitor de 100 micro faradios que sirve como filtro para la fuente de alimentación.
- **R1:** Resistor de 470 ohms que funciona como filtro en el conector de datos para las tiras de ledes.
- **dc_jack:** Conector para la fuente de alimentación.
- **on_off_switch:** Interruptor para controlar el paso de la corriente en todo el dispositivo.
- **reset:** Botón para reiniciar el dispositivo.
- **led_on/off:** Led para indicar cuando el aparato esté encendido.
- **R6:** Resistor de 470 ohms para el led indicador de encendido.
- **led_bt:** Led para indicar cuando el aparato esté transmitiendo información a través de comunicación Bluetooth.
- **R5:** Resistor de 470 ohms para el led indicador de transferencia de información por Bluetooth.

Con todo lo mostrado, este controlador puede apagarse y prenderse fácilmente, reiniciarse sin desconectarlo, conectarse con tiras de LEDs externas y comunicarse por Bluetooth con cualquier otro dispositivo (sea una computadora o un teléfono móvil).

4.3.2. Construcción

Con el propósito de realizar pruebas sencillas y rápidas, se optó por un diseño simplificado del esquema presentado anteriormente. Con esto, se omitió el uso de conectores externos y de luces indicadoras más allá de las que ofrece la tira de ledes, la cual, en este caso, fue solamente de 50 piezas por razones prácticas. Esto es, pues, un prototipo.

Concordando a lo anteriormente mencionado, para una tira de 50 ledes solamente se requieren de 3 amperios, algo que facilita considerablemente la implementación. Cabe mencionar que este acercamiento aún así permite el uso de cada una de las funciones que se exponen en este documento, por lo que no supone ningún impedimento para comprobar de lleno la teoría expuesta en la metodología de diseño.



Figura 4.4: Parte frontal del prototipo.

El SunBoard es un sistema que permite controlar las luces en una pared para entrenamientos de escalado; es así, pues, como en la figura 4.4 se muestra la parte frontal de la tabla en la que se colocaron cada uno de los 50 ledes en una matriz de 7x7 (el led restante se encuentra detrás, tal como se muestra en la figura 4.5).

4. Resultados



Figura 4.5: Parte trasera del prototipo.

El microcontrolador (el ESP32) se encuentra detrás, conectado a la tira de ledes y a la fuente de corriente, el cual es un adaptador de 5 voltios y 3 amperios (mirar las figuras 4.6 y 4.7).



Figura 4.6: Microcontrolador conectado al prototipo.

Como se vio en el esquema, la tira de ledes y el microcontrolador se conectan a la fuente de corriente en paralelo, esto es para evitar una sobrecarga sobre este último, ya que los ledes pueden consumir bastante corriente cuando están todo activos al mismo tiempo. Sin embargo, es muy importante unir los cables de polaridad negativa de ambos componentes para evitar funcionamientos inesperados (tierra común). Esto se puede observar tanto en la figura 4.6 como en la figura 4.8, con el cable que sobresale por debajo de la placa del microcontrolador.

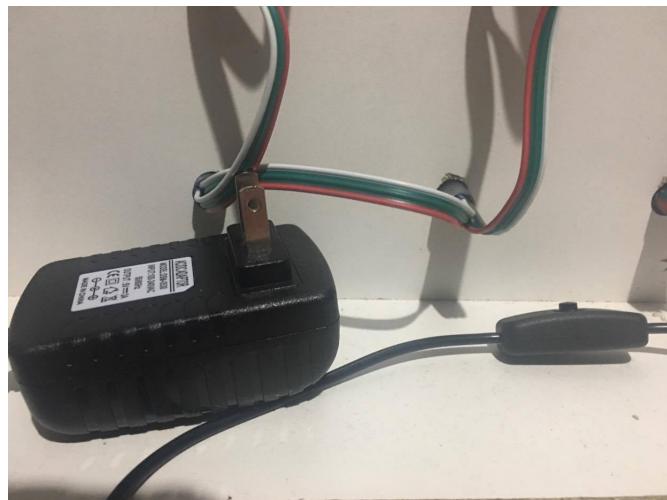


Figura 4.7: Adaptador de corriente del prototipo.



Figura 4.8: Conexión de la tira de ledes del prototipo.

4.3.3. Programación del microcontrolador

La programación de este dispositivo se realiza completamente sobre la plataforma **Arduino**, con la cual es posible utilizar librerías para la compatibilidad con el chip

4. Resultados

ESP32, así como **FastelLED** para el uso sencillo y efectivo de los LED WS2811 [30], además de las librerías básicas para el uso de la comunicación Bluetooth. Estas dos primeras librerías son repositorios hechos por la comunidad.

Para comprender todo el proceso que sigue el programa instalado en el microcontrolador, es necesario visualizar la estructura general, la cual consiste en distintas modos de funcionamiento con propósitos específicos. Esto se realiza para implementar pequeños programas distintos bajo el mismo sistema y cambiar entre ellos sin problema alguno.

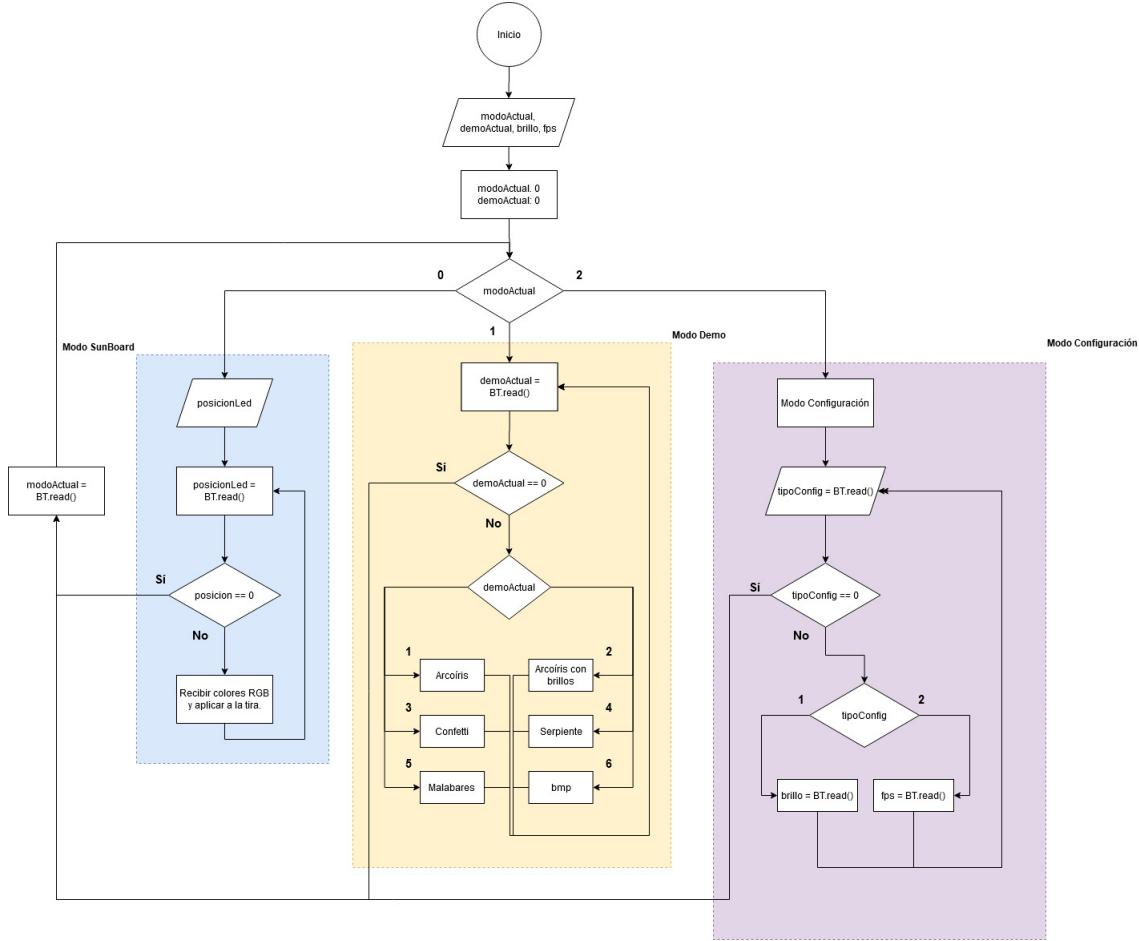


Figura 4.9: Diagrama de funcionamiento del programa del SunBoard.

Existen 3 modos principales (mirar la figura. 4.9):

1. **SunBoard:** Función normal del sistema. Permite encender ledes de manera individual con colores propios. En cada ciclo, se leen 4 números: la posición de un led y los valores de sus tres colores (RGB).
2. **Demo:** Patrones (secuencias de colores) predefinidos para probar todos los ledes conectados al dispositivo. Cuando se cambia a este modo, el primer número leído indicará cuál de los 6 patrones almacenados se ejecutará.
3. **Configuración:** Permite cambiar variables internas del programa. Al momento de ejecutarse, este modo espera recibir un indicador y el valor de la variable correspondiente: el brillo de los ledes o la frecuencia de actualización (FPS) que se utilizará para mostrar los colores.

Como se puede observar, hay una serie de variables declaradas al principio del diagrama, las cuales se detallan a continuación:

- **modoActual:** Indica el modo que se está ejecutando actualmente. Este valor se inspecciona una vez antes de entrar a un nuevo modo.
- **brillo:** La luminosidad general de los ledes.
- **fps:** La frecuencia de actualización de las luces cuando se estén ejecutando el Modo Demo.

Así mismo, cada modo tiene sus variables internas que tienen funciones específicas:

- **Modo SunBoard :: posicionLed:** El número de led en la tira al que se le va a aplicar un color nuevo (el color negro se considera apagado).
- **Modo Demo :: demoActual:** Indica el demo que se está ejecutando actualmente.
- **Modo Configuración :: tipoConfig:** Indica la variable a modificar; 1 para el brillo, 2 para los FPS.

Finalmente, "BT.read().es una representación de la función que lee datos desde la comunicación Bluetooth.

El SunBoard Led System puede cambiar fácilmente desde un modo a otro con tan solo recibir una serie de instrucciones por comunicación inalámbrica. Siempre que se está ejecutando cualquier modo, el programa revisa constantemente si un dato recibido es el comando particular para cambiar de modo; de recibir dicha instrucción, el próximo valor recibido indicará el siguiente modo al que se cambiará.

Como se puede observar, el programa en general nunca llega a un fin concreto, este se debe a la naturaleza del microcontrolador, el cual ejecuta todas sus funciones bajo un ciclo perpetuo, por lo queq la única manera de detenerlo es apagado el dispositivo.

Por supuesto, este diagrama es una explicación simplificada del código, ya que no se está tomando en cuenta varios aspectos que se tienen que codificar dentro del programa para hacerlo funcionar. Por ejemplo, en cada uno de los modos se está leyendo una y otra vez valores desde la comunicación Bluetooth, mientras que la realidad es que en el programa sólo se realiza esta rutina sólo en el momento que se van a recibir datos, de otra forma se salta el paso por completo.

4.3.4. Explicación del código

Con la librería FastLED, los valores de los colores se almacenan en una variable del tipo “CRGB”, la cual no es más que un conjunto de 3 bytes, uno para cada color. Dentro de un arreglo de objetos CRGB se almacenan los valores para cada uno de los ledes conectados al controlador, cada posición en este arreglo es una pieza. FastLED también ofrece la posibilidad de definir el consumo máximo las luces de acuerdo a un amperaje y voltaje predefinido.

Al principio del código del programa se definen las variables y constantes más importantes para lo antes explicado. Cabe notar la declaración del arreglo de los ledes y de la variable **SerialBT**, la cual se encarga de manejar los datos recibidos por Bluetooth.

```
#define FREC_BT_INFO      30 // la frecuencia con la que se
                                recibe informaci n por BT
```

4. Resultados

```
#define NUM_LEDS      50 // cantidad de ledes conectados
#define DATA_PIN       4 // pin de datos para la tira de
                      ledes
#define PSU_V          5 // voltaje de la fuente de
                      alimentacion
#define PSU_A          3000 // miliamperes de la fuente de
                      alimentacion
int BRIGHTNESS =      96;
int FRAMES_PER_SECOND = 120;

CRGB leds[NUM_LEDS]; // arreglo con la informacion para cada uno
                      de los ledes
BluetoothSerial SerialBT; // variable para el control de la
                      informacion por medio de BT
```

En la función **setup**, la primera parte del código que se ejecuta, se inicializan las librerías FastLED y BluetoothSerial pasa sus respectivas variables, además de **Serial** para enviar datos por medio de comunicación serial, la cual sirve para realizar pruebas mostrando información de lo que está haciendo el microcontrolador (cabe mencionar que se omite todo el código relacionado a esto por efectos prácticos). También se puede notar que se asigna un valor a la posición 0 del arreglo de ledes y el uso de la función **FastLED.show**; dicha función es la encargada de hacer visible cada uno de los colores asignados a las luces, con lo cual, es este caso, se está usando como indicador de que el aparato está activo siempre que se prende por primera vez.

```
void setup() {
    FastLED.addLeds<WS2811, DATA_PIN, RGB>(leds, NUM_LEDS); // 
    inicializar el control de los ledes
    FastLED.setMaxPowerInVoltsAndMilliamps(PSU_V, PSU_A); // ajustar
    la corriente maxima que se puede consumir para los ledes
    FastLED.setBrightness(BRIGHTNESS);
    leds[0] = CRGB::Red;
    FastLED.show();
    Serial.begin(115200);
    SerialBT.begin("SunBoard"); // inicializar la comunicacion BT y
    asignar el nombre
    //prueba(); // realizar secuencia de prueba inicial
}
```

Antes de continuar, cabe recordar lo mencionado al principio de esta sección acerca de la estructura del programa, pues la siguiente pieza de código es la principal representación de ello:

```
// Lista de demos. El primer demo es el de la funcion normal del
// dispositivo.
typedef void (*ListaFunciones[])();
ListaFunciones demos = {rainbow, rainbowWithGlitter, confetti,
                        sinelon, juggle, bpm};
ListaFunciones modos = {sunboard, demo, configurar};

int demoActual = 0; // indice del demo actual
uint8_t matiz = 0; // rotating "base color" used by many of the
                  patterns
int modoActual = 0; // indice del modo actual
```

Estas variables contienen direccionamientos a una serie de funciones importantes dentro del programa. Hay dos arreglos: **demos** y **modos**, en donde el primero se ejecuta en el Modo Demo que se mencionó anteriormente, mientras que el segundo divide la principales funciones del sistema. Las siguientes tres variables, **demoActual**, **matiz** y **modoActual** son variables de control para llevar seguimiento sobre el estado actual del sistema; particularmente, **matiz** se usa dentro del Modo Demo para variar el color de las luces de manera dinámica [38].

```
void loop() {
    // cargar el modo que se est    ejecutando actualmente
    modos[modoActual]();
}
```

La función **loop** se ejecuta de manera perpetúa durante todo el tiempo que el dispositivo está activo, por lo que no tiene fin. En el caso del SunBoard, esta función no tiene más que una sola línea, aunque es evidente que esto se realiza para llamar el modo que se esté ejecutando actualmente, los cuales, como se observó en el diagrama de la figura 4.9, se encierran momentáneamente en su propio ciclo.

Al principio, el modo que se ejecuta por defecto es el Modo SunBoard, el modo normal del sistema.

```
/*
    Modo de funcionamiento normal del SunBoard en el que recibe
    instrucciones para
    prender y apagar ledes individuales.
*/
void sunboard() {
    while (true) {
        if (SerialBT.available()) {
            int pos = SerialBT.read();
            // posicion 0 significa un cambio de modo
            if (pos == 0) {
                cambiarModo();
                break; // salir del modo actual
            } else {
                int r = SerialBT.read();
                int g = SerialBT.read();
                int b = SerialBT.read();
                if (pos < NUM_LEDS && pos >= 0) leds[pos] = CRGB(r, g, b);
                FastLED.show();
            }
        }
        delay(FREC_BT_INFO);
    }
}
```

La lógica de la función **sunboard** es sencilla: se revisa constantemente si hay datos de entrada por comunicación Bluetooth (para esto sirve la función **SerialBT.available**), para que, llegado el momento, se lea un número entero de 8 bytes hacia la variable **pos** para la posición de un led en particular. De recibirse un 0, se pasa a cambiar de modo (tal como se explicó anterieramente), de lo contrario se reciben 3 números más, los cuales constituyen los colores rojo, verde y azul de un objeto tipo **CRGB** para almacenar la información de un led. Finalmente, los

4. Resultados

valores recibidos se muestran llamando la función **FastLED.show**.

```
void cambiarModo() {
    int numRecibido = SerialBT.read();
    if (numRecibido >= 0 && numRecibido < ARRAY_SIZE(modos)) {
        modoActual = numRecibido;
        ...
        if (modoActual == 1) {
            numRecibido = SerialBT.read();
            if (--numRecibido >= 0 && numRecibido < ARRAY_SIZE(demos)) {
                demoActual = numRecibido;
            } else {
                demoActual = 0;
            }
        }
        ...
    }
}
```

A pesar de todo el código que la función **cambiarModo** tiene, la lógica se resume en recibir un número por Bluetooth y comprobar que sea válido (que no se salga de los valores permitidos dentro del arreglo de funciones para los modos).

De todas, la función para el Modo Demo es la más complicada de todas, pues involucra una serie de cálculos muy rápidos que hace el programa para mostrar colores en cada uno de los patrones distintos.

```
void demo() {
    while (true) {
        // cargar el demo que se est    ejecutando actualmente
        demos[demoActual]();
        ...
        // actualizaciones periódicas
        EVERY_N_MILLISECONDS(20) {
            matiz++; // actualiza el matiz (color base), esto da el
                      aspecto del arco iris
        }
        EVERY_N_MILLISECONDS(FREC_BT_INFO) {
            // recibir instrucciones por BT para cambiar modo (si hay)
            if (SerialBT.available()) {
                int numRecibido = SerialBT.read();
                if (numRecibido == 0) {
                    cambiarModo();
                    break;
                }
            }
        }
    }
}
```

El modo demo realiza 3 pasos en todo su ciclo: calcular posiciones, ajustar matiz y revisar si hay entrada de datos por Bluetooth. Como se puede observar en el código, la función **demo** primeramente ejecuta una función que esté almacenada en el arreglo de demos, en donde se calculan las posiciones y colores de acuerdo al patrón seleccionado; luego, la función **EVERY_N_MILLISECONDS** ejecuta un fragmento de código cada cierta cantidad de tiempo, en este caso es para ajustar el matiz; finalmente, se utiliza la misma función de antes para ejecutar rutinas de actualización por medio de instrucciones recibidas por Bluetooth (cambiar de modo o cambiar de demo).

Como se puede observar, la función **demo** encierra todo su código dentro de un **while** perpétuo, el cual solamente se detiene cuando se recibe un 0 desde la conexión

Bluetooth. De esta forma, el Modo Demo en cada ciclo ejecuta estas 3 tareas rápidamente, permitiendo que los cambios de estado sin interrumpir ninguna parte.

```
void configurar() {
    while (true) {
        if (SerialBT.available()) {
            ...
            switch (tipoConfig) {
                case 1:
                    BRIGHTNESS = SerialBT.read();
                    FastLED.setBrightness(BRIGHTNESS);
                    break;
                case 2:
                    FRAMES_PER_SECOND = SerialBT.read();
                    break;
            }
            ...
        }
    }
}
```

Por último, tenemos la función **configurar**, en la que lo único que se espera es recibir datos para cambiar variables dentro del programa, particularmente 2: el brillo y los fotogramas por segundo. Esta sección del código, como las otras, revisa constantemente si hay una entrada de datos, cuando los recibe comprueba si el número recibido no es un 0 (lo cual significaría un cambio de modo), de lo contrario se recibe otro número más para indicar qué variable se va a cambiar, siendo el último número recibido el valor que se asignará.

El resto del código es para el cálculo de los patrones incluidos con el sistema. Si se desea conocerlos más a detalle, se puede consultar la documentación y código fuente de la librería FastLED, en donde vienen incluidos estos demos [39].

4.4. Desarrollo de la aplicación

El último paso para la metodología de diseño IoT es el desarrollo de la aplicación móvil que el usuario utilizará para comunicarse con el controlador. Esta aplicación debe permitir seleccionar cualquiera de los 200 ledes individualmente para apagarlos/prenderlos y cambiarles el color.

En general, la aplicación permite acceder a cada una de las funciones del sistema expuestas hasta ahora, de manera sencilla y rápida, sin tener que preocuparse por el funcionamiento interno del código y la comunicación con el microcontrolador.

Es así, pues, como este software permite:

1. Prender y apagar ledes de manera individual, seleccionado su color y posición en la tabla.
2. Ejecutar cada uno de los 6 demos disponibles .
3. Configurar el brillo y la frecuencia de actualización.

Esta aplicación móvil funciona en dispositivos móviles Android, y está desarrollada con el MIT App Inventor, una plataforma de programación visual para el mencionado sistema operativo.

4. Resultados

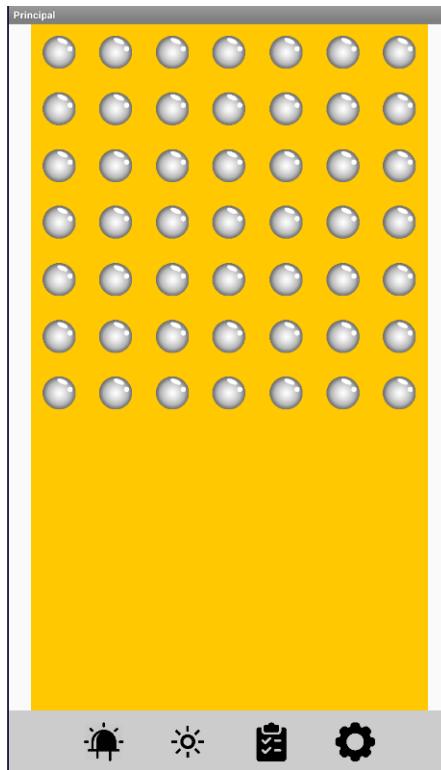


Figura 4.10: Menú principal de la aplicación.

Tal como se muestra en la figura 4.10, el menú principal se divide en dos elementos principales: el **panel de ledes** y la **barra de funciones**. Dentro del panel aparecen todos los ledes que se pueden controlar en la tabla del prototipo, cuando se presiona cualquier de ellos se le asigna un color y la información se envía al microcontrolador para que se aplique al led correspondiente.



Figura 4.11: Barra de funciones de la aplicación.

Los iconos de la barra inferior permiten realizar una serie de funciones rápidas (figura 4.11). De izquierda a derecha, son las siguientes:

- **Leds:** Apaga todos los ledes inmediatamente, también permite acceder al menú principal desde el **menú de configuración**.
- **Color:** Permite seleccionar un color para asignarles a los ledes del panel. Cuando se presiona aparece una pequeña rueda, de entre la cual se puede elegir un color.
- **Rutinas:** Permite guardar y cargar patrones de ledes que se hayan hecho con el panel de ledes (las cuales se llaman rutinas).
- **Configuración:** Accede al menú de configuración.

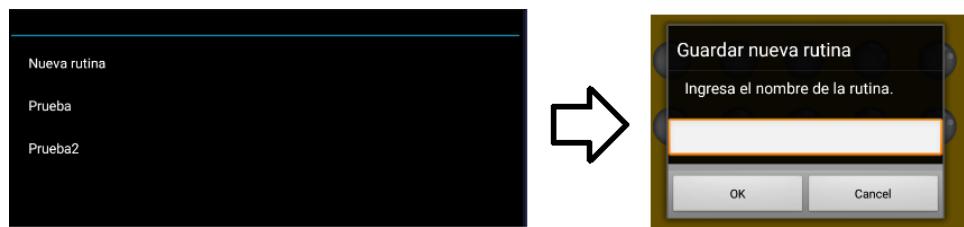


Figura 4.12: Guardando una nueva rutina en la aplicación.

El botón para rutinas constituye una función particular de esta aplicación para ser utilizada en los ejercicios de entrenamiento que se han estado mencionando a lo largo del proyecto. En este menú se pueden guardar el patrón actual de ledes y asignarles un nombre para identificarlo; cuando se quieran volver a utilizar, simplemente se seleccionan desde este menú (mirar las figuras 4.12 y 4.13).

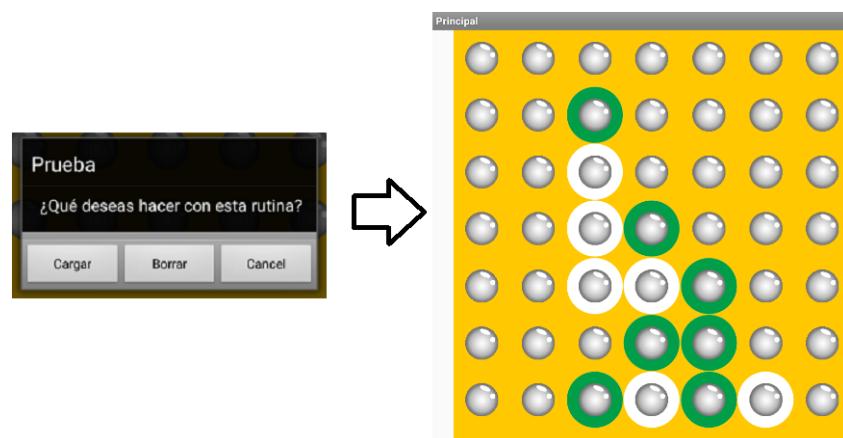


Figura 4.13: Cargando una rutina en la aplicación.

Antes de empezar a utilizar el dispositivo, primero es necesario realizar la conexión por Bluetooth. Esto se puede hacer en el menú de configuración.

4. Resultados



Figura 4.14: Menú de configuración de la aplicación.

Este menú está dividido en varias partes, siendo la primera en la que se puede seleccionar el dispositivo con el que la aplicación se conectará. Naturalmente, es necesario seleccionar el que corresponda al SunBoard. Una vez esté listo el dispositivo, se presiona el botón **Actualizar** para cargar la lista de aparatos que están en el rango del teléfono móvil y se selecciona de entre la lista. Una vez se recibe la notificación de que la conexión se realizó exitosamente, se puede volver al menú principal presionando el ícono del led en la barra inferior (es la única función que se tiene la barra mientras se esté en el menú de configuración).



Figura 4.15: Opciones extra en el menú de configuración de la aplicación.

Las siguientes opciones constituyen funciones experimentales y no es necesario

modificarlas para utilizar el SunBoard con normalidad. Las opciones que la aplicación nos permite modificar son las siguientes (figura 4.15):

- **Dimensiones:** El panel de ledes por defecto es de 7x7 para ajustarse al diseño del prototipo, pero si se desea se puede modificar a uno *más pequeño*, disminuyendo cada alguna de las dos medidas.
- **Posiciones:** Ya los índices de los ledes en el panel de la aplicación seguramente no coinciden con la forma en que están colocados en la tabla del prototipo, es necesario asignarle una posición a cada led *individualmente*. Para esto la aplicación ofrece 3 opciones: colocando uno por uno, escribirlos todos en un campo de texto, o usar el **mapeo interactivo**. Esta última opción se detalla más adelante.
- **Otras configuración:** El modo debug permite mostrar la salida de ciertas acciones en la aplicación, ya que se implementó para efectos de pruebas durante el desarrollo, no se recomienda activarlo ya que podría causar muchos problemas. Por otro lado, es posible modificar el brillo máximo de todos los ledes.

Al final del menú de configuración se encuentran las opciones para ejecutar los demos que se mencionaron anteriormente (figura 4.16). El uso es sencillo: se selecciona una frecuencia de actualización, un demo de entre la lista, y se presiona el botón **Ejecutar** para que el dispositivo empiece a mostrarlo.



Figura 4.16: Opciones para la ejecución de demos en la aplicación.

Finalmente, está el menú para el **mapeo de posiciones**. En un principio, los ledes en el panel del menú principal están ordenados de izquierda a derecha, de arriba hacia abajo, lo cual no coincidió con la forma en que colocó la tira de ledes en el prototipo. Con este menú se pueden asignar las posiciones de manera sencilla y rápida; en orden, se coloca el índice de los ledes en el panel con su correspondiente posición en la tabla del prototipo. Como se observa en la figura 4.17, la tabla recorre las posiciones de la matriz de manera alternante, de izquierda a derecha.

4. Resultados

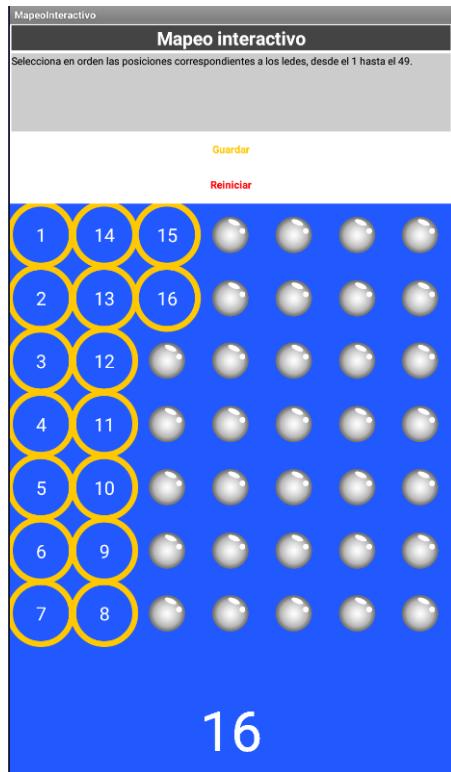


Figura 4.17: Menú para el mapeo de posiciones..

5

Pruebas

Para demostrar el funcionamiento general de lo implementado en el capítulo 4, se procede a presentar los resultados de las pruebas realizadas a lo largo del diseño e implementación del SunBoard Led System.

La versión que se muestra es la de un mero prototipo y, tal como se mencionó en el capítulo anterior, se omitieron pequeñas partes del diseño para efectos prácticos, lo cual no tiene por qué limitar el resultado final, pues cada una de las funciones expuestas hasta ahora se lograron poner a prueba.

5.1. Microcontrolador y ledes

La base del diseño de este dispositivo parte de controlar una gran cantidad de ledes con un solo conector, por lo que es muy importante comprobar que este funcione de manera de manera correcta antes de siquiera seguir con la implementación de funciones complicadas.

Como se detalló durante la explicación del código, al prenderse por primera vez el microcontrolador, éste enciende el primer led de la tira para indicar que el aparato está activo. La placa del ESP32 también cuenta con una luz propia para indicar que se está recibiendo corriente (observar la figura 5.1).

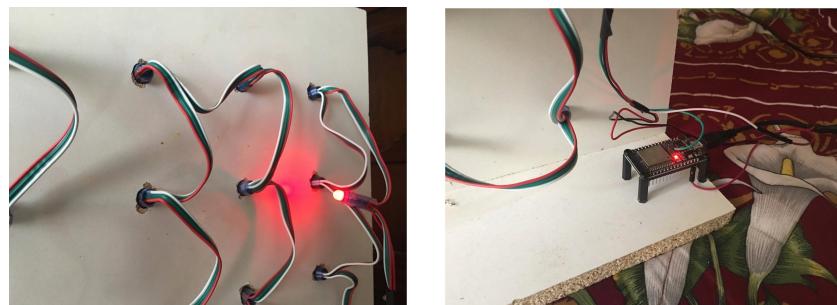


Figura 5.1: Prueba de encendido del microcontrolador y el led indicador.

Evidentemente, esto no demuestra del todo que la tira de ledes funcione correctamente, pues aún haría falta comprobar todas las demás piezas en la tira. Sin embargo, esto como mínimo debería dar la idea sobre dos puntos muy importantes: el microcontrolador está funcionando y se puede comunicar con los circuitos integrados de las luces. Así mismo, ésta es una rutina que se lleva a cabo cada vez que se prende el aparato, por lo que sirve como un indicador minimalista.

Otra parte de comprobar la integridad del dispositivo es ver que éste sea capaz de iniciar la comunicación Bluetooth. Esto se puede comprobar con cualquier dispositivo que tenga integrado este protocolo, sólo hace falta ver si el Sunboard aparece en la lista de dispositivos disponibles y si es posible vincularlo. Como se observa en la figura 5.2, esto se comprobó con una computadora, en la que, efectivamente, el SunBoard aparece en la lista de dispositivos y es posible vincularlo. Con lo que se pudo concluir que la comunicación Bluetooth funciona correctamente.

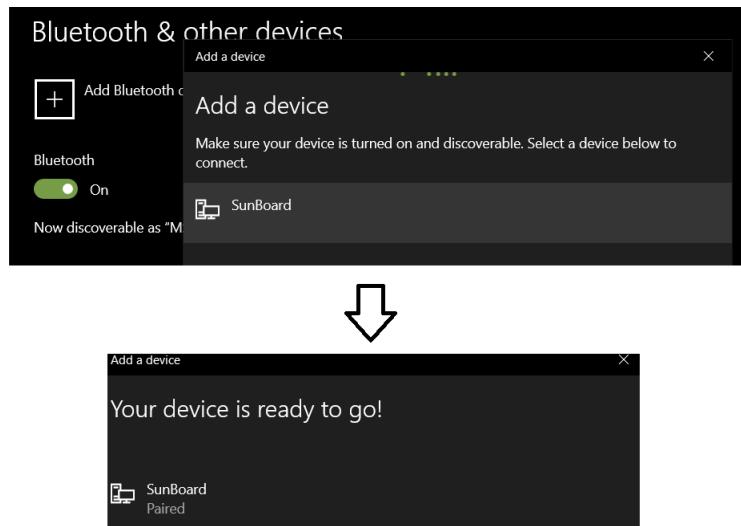


Figura 5.2: Comprobando la conexión Bluetooth con el dispositivo.

5.2. Uso de la aplicación

Una vez se consiguió vincular un dispositivo móvil con el SunBoard, el siguiente paso fue ejecutar la aplicación, conectarse y tratar de encender cualquier led con cualquier color. Esto permitió comprobar dos cosas: que el microcontrolador es capaz de recibir e interpretar instrucciones por Bluetooth y que la tira de ledes funciona completamente ya que permite encender cada una de sus luces (figura ??).

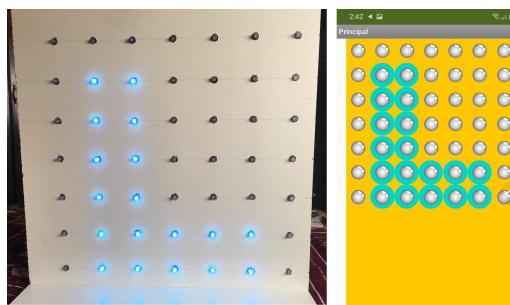


Figura 5.3: Prueba de funcionamiento del Modo SunBoard.

Continuando con el menú principal, se guardaron y cargaron varias rutinas para comprobar que la aplicación las guardara correctamente y le enviará la información

al dispositivo (figura 5.4). Esto resultó ser efectivo, ya que en todo momento, el estado del panel en la aplicación se reflejaba en la tabla.

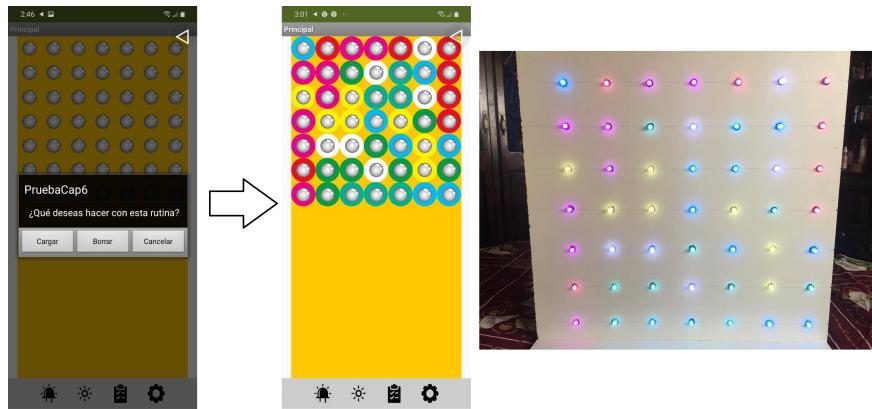


Figura 5.4: Prueba del funcionamiento de las rutinas.

Con el funcionamiento normal (Modo SunBoard) del dispositivo comprobado, se procedió a tratar de ejecutar cualquier de los demos disponibles en el menú de configuración. Al tratar de realizar esto, la aplicación le indica al microcontrolador que cambie de modo y ejecute un demo en particular, por lo que el estado del panel es sobreescrito inmediatamente. En la figura 5.5 se muestran varias fotografías de la ejecución de los demos, aunque probablemente no se puedan apreciar del todo de este modo.

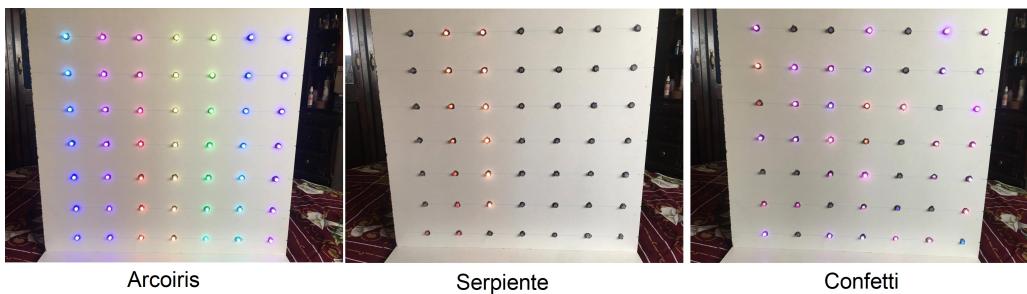


Figura 5.5: Comprobando el funcionamiento de los demos.

Para reforzar las pruebas realizadas, se programó el microcontrolador para que mostrar por el puerto serial el resultado del código que se esté ejecutando, tal como se puede observar junto a varias de las imágenes mostradas hasta ahora. Esto es algo que se mencionó brevemente en el Capítulo 4.

5. Pruebas

The figure displays four separate terminal windows, each enclosed in a black-bordered box, showing the output of different commands sent to a microcontroller. The windows are arranged in a 2x2 grid.

- Top Left:** Shows the initial state after entering mode 0. It includes the command "Entrando al Modo SunBoard".

```
Modo seleccionado: 0
Posicion: 1
RGB: 0,0,0
Posicion: 14
RGB: 0,0,0
```
- Top Right:** Shows the state when LED 1 is turned on with white color.

```
Posicion: 1
RGB: 255,255,255
```
- Bottom Left:** Shows the state while executing demos for Arcoiris and BMP.

```
Modo seleccionado: 1
Demo seleccionado: 0
Modo seleccionado: 1
Demo seleccionado: 5
```
- Bottom Right:** Shows the state during configuration loading for brightness and FPS.

```
Modo seleccionado: 2
Tipo de config 1, valor = 96
Tipo de config 2, valor = 120
```

Figura 5.6: Ejecución de distintas instrucciones y su salida por el puerto serial.

En la figura 5.6 se resume una serie de pruebas y su correspondiente mensaje por la salida serial del dispositivo, reforzando así que la comunicación se está realizando de manera correcta entre la aplicación y el microcontrolador.

6

Conclusiones

A través de lo expuesto hasta ahora en los primeros capítulos, y con los resultados de las pruebas vistas en el Capítulo 5, se encontró que los ledes de direccionamiento individual ofrecen una gran gama de posibilidades sin requerir de una instalación complicada. El enfoque primario de este proyecto fue crear un dispositivo que sea capaz de apagar y encender una gran cantidad de luces desde una aplicación móvil, lo cual se consiguió exitosamente sin la necesidad de una lógica compleja gracias al uso de las librerías y recursos que ya se encuentran disponibles.

Anteriormente se mostró un prototipo con una construcción sencilla y que aun así fue capaz de realizar comunicaciones inalámbricas y enviar datos a las tiras de ledes con el chip WS2811 sin requerir de una configuración complicada. Todo esto demuestra claramente que el diseño de un dispositivo de este tipo no supone un gran costo ni un tiempo de desarrollo desorbitado, posicionándose así como una alternativa adecuada para el MoonBoard, un proyecto comercial de similar naturaleza pero con precio mucho mayor.

Los resultados obtenidos en cada una de las pruebas dan una clara idea de que las funciones implementadas en este proyecto no son más que una pequeña parte de lo que estos dispositivos son capaces de hacer con la programación y el uso de componentes adecuados.

La cantidad de herramientas y recursos disponibles bajo la plataforma de desarrollo Arduino demuestra que ésta fue una elección adecuada para las necesidades de este proyecto, pues su uso e implementación no causaron dificultades mayores durante el desarrollo.

El diseño que se tomó en la aplicación móvil en conjunto al programa del microcontrolador fueron hechos de tal manera que no se necesitara una gran cantidad de acciones por parte del usuario para manipular el estado de los ledes, no obstante, esto no significa que no podrían manejarse otro tipo de sistemas de entrada, ya sean comandos de voz o algún otro sensor externo conectado directa o indirectamente al controlador.

Continuando con lo anterior, la evolución natural de un proyecto como éste es con el uso de más métodos de entrada para manipular los ledes (sensores, audio, etc.) y una construcción más robusta que permita un fácil transporte y uso tiras de ledes aún mayores (tal como se expuso en el esquema del Capítulo 4). El prototipo fue, pues, nada más que un rápido acercamiento al verdadero potencial de un proyecto de esta índole.

En resumen, con este proyecto se presentó una alternativa viable y de bajo costo a un producto comercial del mismo tipo; se realizaron las pruebas pertinentes y se analizaron los resultados obtenidos para comprobar la efectividad del prototipo

6. Conclusiones

construido, concluyendo satisfactoriamente con el objetivo inicial.

Nomenclature

Bluetooth Protocolo de comunicación inalámbrico para transmisión de voz y datos entre distintos dispositivos.

Chipset Conjunto de circuitos integrados construidos en base a la arquitectura de un procesador.

FPS *Frames per second*, fotogramas por segundo, es la cantidad de veces que se actualiza una imagen cada segundo.

IDE Integrated Development Environment. Entorno de desarrollo integrado.

IoT Internet of Things, Internet de las Cosas en español.

RGB Red Green Blue. Modelo de color basado en la síntesis aditiva, en la que es posible interpretar colores del espectro visible al ojo humano por medio de la mezcla aditiva de 3 colores primarios.

Bibliografía

- [1] Individually addressable leds.
- [2] Luis Llamas. Conectar arduino con paneles y tiras led rgb ws2812b (neopixel), Apr 2018.
- [3] Mitchell Graham. Ws2812 / neopixel addressable leds: Arduino quickstart guide - tutorial, Feb 2019.
- [4] Pololu - addressable rgb 60-led strip, 5v, 2m (ws2812b).
- [5] Esp8266 todo lo que necesitas saber del módulo wifi para arduino, Jun 2020.
- [6] Bluetooth en arduino, Nov 2016.
- [7] Abhiemanyu Pandit. Bluetooth module interfacing with esp8266: Controlling an led, May 2019.
- [8] Jed Margolin. The road to the transistor, Aug 2004.
- [9] Henry Joseph Round. A note on carborondum, Feb 1907.
- [10] M. Guarnieri. Trailblazers in solid-state electronics [historical]. *IEEE Industrial Electronics Magazine*, 5(4):46–47, 2011.
- [11] K. Lehovec. New photoelectric devices utilizing carrier injection. *Proceedings of the IRE*, 40(11):1407–1409, 1952.
- [12] W T Matzen. *Semiconductor Single-Crystal Circuit Development*. Texas Instruments Incorporated, 1963.
- [13] James R Biard and Gary E Pittman. Semiconductor radiant diod, Dec 1966.
- [14] Jr. Holonyak, Nick and S. F. Bevacqua. Coherent (visible) Light Emission from Ga(As_{1-x}P_x) Junctions. *Applied Physics Letters*, 1(4):82–83, December 1962.
- [15] T. S. Perry. M. george craford [biography]. *IEEE Spectrum*, 32(2):52–55, 1995.
- [16] H. Walter Yao. *Light-emitting diodes: research, manufacturing, and applications V: : 24-25 January 2001, San Jose, USA*. SPIE, 2001.
- [17] Howard C Borden and Gerald P Phigini. Solid-state displays. *Hewlett-Packard Journal*, 20(6), Feb 1969.
- [18] Bernhard Kramer. *Advances in solid state physics*. Springer-Verlag, 2003.
- [19] Charles House and Raymond Price. *The HP Phenomenon Innovation and Business Transformation*. Stanford University Press, 2009.
- [20] Neel V Patel. Nobel shocker: Rca had the first blue led in 1972, Oct 2014.
- [21] Shuji Nakamura, Takashi Mukai, and Masayuki Senoh. Candela-class high-brightness ingan/algan double-heterostructure blue-light-emitting diodes. *Applied Physics Letters*, 64(13):1687–1689, 1994.
- [22] Yu Ri Song, Chang-Sub Won, Hyungkeun Ahn, Deuk-Young Han, Yun-Seok Choi, and Seok-Jun Lym. The study on optimal design and optical properties of led module for full-color displays. In *Proceedings of 5th International Conference*

- on *Properties and Applications of Dielectric Materials*, volume 2, pages 956–959 vol.2, 1997.
- [23] Chr. Led cube 8x8x8, May 2019.
 - [24] P. Deimel, J. Cheng, S. Forrest, P. Hu, R. Huntington, R. Miller, J. Potopowicz, D. Roccasecca, and C. Seabury. Individually addressable monolithic 1×12 light-emitting diode array. *Journal of Lightwave Technology*, 3(5):988–991, 1985.
 - [25] H. X. Zhang, E. Gu, C. W. Jeon, Z. Gong, M. D. Dawson, M. A. A. Neil, and P. M. W. French. Microstripe-array ingan light-emitting diodes with individually addressable elements. *IEEE Photonics Technology Letters*, 18(15):1681–1683, 2006.
 - [26] WorldSemi. Ws2811.
 - [27] Welcome to training on the moonboard, climb on the same problems as other climbers from around the world.
 - [28] Carlos J. Jiménez Fernández. Metodología de diseño electrónico dentro de prácticas obligatorias de laboratorio. *Pixel-Bit. Revista de Medios y Educación*, page 19–27, Jul 2010.
 - [29] Serial communication, May 2020.
 - [30] Daniel Garcia. Fastled/fastled.
 - [31] Hans Luijten. Arduino - controlling a ws2812 led strand with neopixel or fastled, Jan 2015.
 - [32]
 - [33] 0a-esp8266ex datasheet en - espressif.
 - [34] App inventor, May 2020.
 - [35] A. Bahga and V. Madisetti. *Internet of Things: A Hands-On Approach*. Arsheep Bahga & Vijay Madisetti, 2014.
 - [36] Luis Llamas. Esp32, el "hermano mayor" del esp8266 con wifi y bluetooth, Apr 2018.
 - [37] Esp32-devkitc v4 getting started guide¶.
 - [38] FastLED. Fastled/fastled.
 - [39] FastLED. Fastled/fastled.