Taller de Proyecto I (E0306)
2022

Informe Final

Sistema Centralizado de Control de Iluminación de un Ambiente

Ingeniería en Computación
22/02/2023

Kleinubing, Hernán (01614/6)
Palacio, Constantino Agustín (01806/2)

FACULTAD DE INGENIERÍA
UNIVERSIDAD NACIONAL DE LA PLATA

Tabla de contenido

1. Introducción

En la actualidad, las tecnologías de las comunicaciones están avanzadas a un nivel sin precedentes. Dentro del campo de la electrónica, la integración de millones de componentes dentro de los conocidos *chips* ha permitido encapsular una gran capacidad de cómputo dentro de un dispositivo de hardware de pequeño tamaño y precio relativamente bajo. Particularmente relevantes son los microcontroladores (MCU), dispositivos que poseen una unidad de procesamiento, memorias de cálculo y almacenamiento, y periféricos de comunicación dentro de un único circuito integrado.

Esta clase de dispositivos electrónicos pueden ser utilizados para una gran variedad de propósitos, entre los que destaca la domótica.

Formalmente, puede definirse a la domótica como el conjunto de sistemas encargados de automatizar diversos procesos en una vivienda y brindar así servicios de comunicaciones, seguridad y monitoreo. Algunos dispositivos dentro de este campo son los termostatos, persianas eléctricas, medidores y reguladores de consumo eléctrico, alarmas de incendios, cámaras de vigilancia y sistemas de regulación de luminaria.

En particular, un dispositivo de control de luminaria es capaz de realizar una o varias de las siguientes tareas:

- Encendido automático de algunas o todas las luces de un ambiente basándose en la luz exterior, sensores de movimiento, temporizadores, etc.
- Regulación de la intensidad de una lámpara según preferencias del usuario o nivel de luminosidad del ambiente

Dichas tareas pueden ser realizadas a través de diversos tipos de tecnologías. Pueden adquirirse módulos con sensores de movimiento y luminosidad que se acoplan a una instalación eléctrica existente, cuyas detecciones regulan el encendido/apagado de las luces vinculadas a ellos. Es posible comprar módulos de regulación de intensidad (*dimmers*), que consisten en perillas que limitan la corriente que circula por la lámpara. Estos módulos también se conectan directamente a las lámparas, como se muestra en la figura 0.

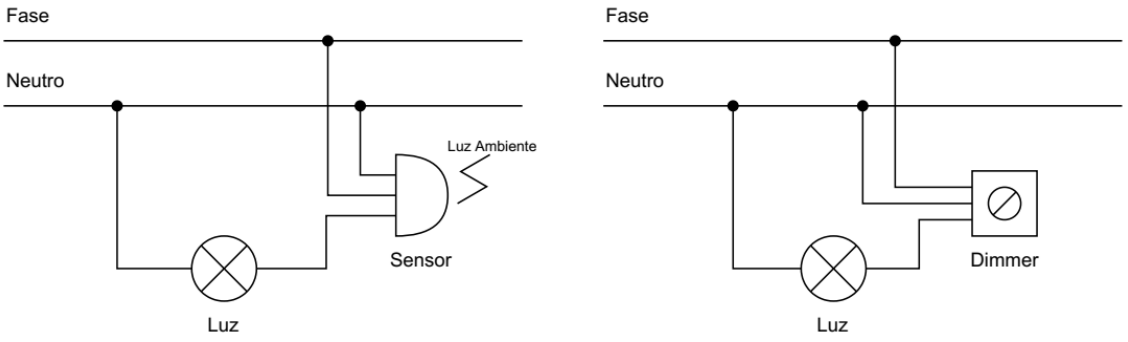


Figura 0. Esquemas de conexionado de un sensor y un regulador de intensidad.

En la figura 0 se aprecian las conexiones necesarias para agregar funcionalidades de encendido automático (izquierda) y de control de intensidad (derecha). Ambas funcionalidades deben instalarse por separado y, en caso de desearse que una misma lámpara se encienda automáticamente a una intensidad configurable, la instalación resultaría demasiado compleja para un usuario estándar. Asimismo, tener en cuenta que este tipo de instalaciones no son para luces de LED, sino lámparas incandescentes o de LED con alimentación de 220V/10A, por lo que errores o descuidos en la instalación podrían provocar daños al equipo, la instalación del ambiente o el operador.

Otro tipo soluciones incorpora tecnología de conexión WiFi para controlar la iluminación. Un ejemplo puede conseguirse comercialmente bajo el nombre Macroled Smart SSX2-WIFI [1] y permite encender y apagar las luces por medio de una aplicación móvil con reconocimiento de voz. La instalación de este dispositivo es relativamente sencilla comparada con la tradicional, y combina reguladores de intensidad, encendido selectivo y otras funciones que serían muy difíciles de instalar si no se usara esta forma de solución integrada. En la figura 1 puede observarse el diagrama de conexión del sistema Macroled Smart. Notar que es de complejidad comparable con conectar una llave de luz estándar o un regulador de intensidad.

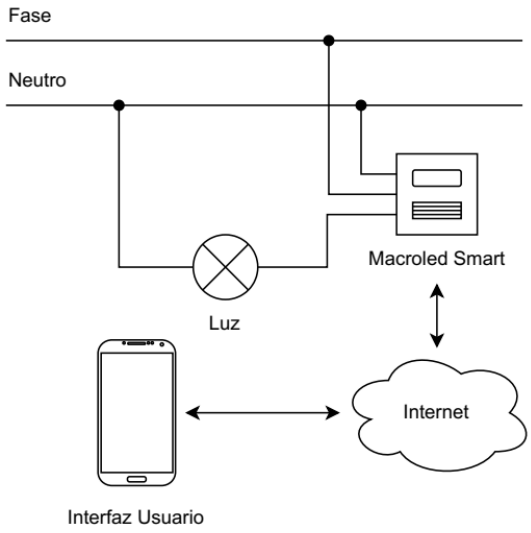


Figura 1. Esquema de conexión de solución integrada Macroled Smart.

Una solución integrada menos sofisticada que la presentada aquí es la que se propone en el presente proyecto: un sistema centralizado de control de encendido e intensidad basado en la placa de desarrollo EDU-CIAA-NXP.

La motivación para la realización del proyecto podría condensarse en una situación factible en la realidad. Supóngase que en una casa habitan cuatro personas, cada una de las cuales es responsable de apagar las luces cada vez que deja una habitación. Considerar también que la casa podría contar con luces en el patio delantero o trasero y que alguno de los integrantes de la unidad doméstica debe encenderlas cuando oscurece por las noches. Si una o varias personas olvidaran apagar las luces, éstas quedarían encendidas y se estaría desperdiciando energía, además de reducir la vida útil de las lámparas. Si olvidasen encender las luces exteriores, podría considerarse una falla en la seguridad del hogar. La implementación del sistema propuesto simplificaría las tareas de control por parte de los usuarios finales y podría lograr una reducción en el consumo de energía eléctrica.

2. Objetivos

2.1 Objetivos Principales

El objetivo principal del proyecto es diseñar e implementar un dispositivo capaz de controlar la iluminación de un ambiente desde una consola central. El mismo estará basado en la placa de desarrollo EDU-CIAA-NXP y utilizará entradas manuales (pulsadores y conmutadores) para el control del sistema. A su vez, se contará con entradas "inteligentes" (sensores de luminosidad y movimiento) para regular el encendido automático de las luces basándose en el momento del día (día/noche) y el movimiento dentro del ambiente. El siguiente diagrama de bloques ilustra el sistema descrito en términos generales. En él se obviaron conexiones, controles y salidas relacionadas con el estado del sistema, como el indicador de encendido y botón de reinicio entre otros.

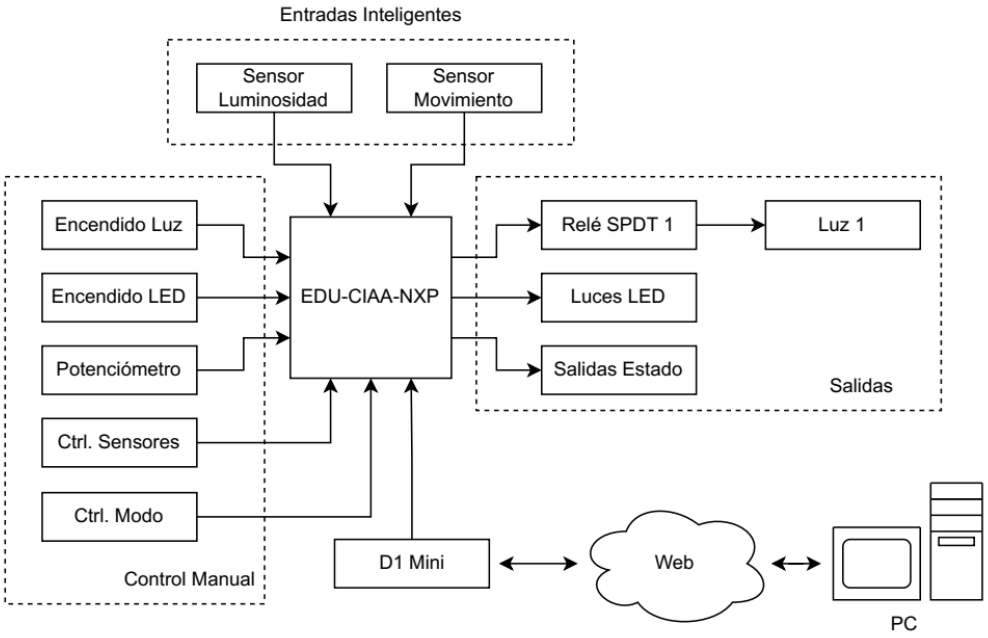


Figura 2. Diagrama de bloques del sistema.

El sistema posee una salida a una luz doméstica de 220V y una luz LED, modelada como una única conexión. Aquí se conectaría una "tira" de luces LED, cuya intensidad será controlada a través de un potenciómetro conectado a uno de los puertos del periférico ADC del MCU.

Las entradas de control del sistema son botones que alternarían el estado de las luces entre encendido y apagado. Por defecto (al encender el sistema), las luces estarían apagadas. Estas entradas se conectarán a los pines de propósito general (entrada/salida digital) del MCU.

El estado de la luz activaría/desactivaría el relé, que cerraría o abriría el circuito entra la lámpara y la entrada de 220V.

La interfaz de usuario incluirá también un conmutador para habilitar/deshabilitar los sensores del sistema, así se evitaría que las luces se encendieran de forma automática cuando se desea que permanezcan apagadas o viceversa. En la figura 2 se indica como "control de sensores".

Respecto a los sensores, se conectará el sensor de luminosidad a una de las entradas del periférico ADC del MCU. Idealmente, se conectará el sensor de movimiento a un pin de propósito general. Esta última conexión depende del tipo de sensor de movimiento que se vaya

a emplear en el proyecto. Si el sensor devuelve una respuesta digital (+3.3V si hay movimiento, 0V si no lo hay), se hará la conexión descrita anteriormente. De retornar un valor analógico, el sensor de movimiento deberá conectarse a una entrada del ADC.

Al sistema también se le agregará conexión WiFi a través de un módulo ESP8266, para poder realizar las tareas de control y monitoreo de forma remota desde un navegador web. Esta conexión no sería implementada directamente sobre el poncho por ser demasiado compleja, sino que se conectaría a una bornera ubicada en el poncho.

Se proveerá una entrada de selección de modo, donde se puede elegir entre utilizar la interfaz de usuario WiFi o la botonera. Por defecto, la interfaz WiFi se encuentra desactivada y debe pulsarse este botón para activarla.

2.2. Objetivos Secundarios

A continuación se incluye un listado de posibles mejoras o ampliaciones que podrían agregarse al sistema una vez cumplidos los objetivos principales.

2.2.1. Conexión USB

Se propone una conexión mediante USB a una PC para control del dispositivo sin la necesidad de utilizar los controles manuales. Se haría uso de un software de emulación de terminales para establecer la comunicación por protocolo RS232 a través del periférico UART del MCU. De realizarse esta mejora, deberá tenerse en cuenta que el puerto USB también provee alimentación a la placa EDU-CIAA-NXP.

2.2.2. Soporte de Luces LED Configurables

En el mercado existen luces LED a las que se les puede configurar el nivel de intensidad y color, además de encender los LEDs de forma individual. Si bien es sencillo conectar una de estas luces al sistema base y modificar el programa para que aproveche sus funcionalidades adicionales, las lámparas exceden el presupuesto del proyecto.

2.2.3. Configuración de la tira LED con sensor de luminosidad

Pueden agregarse al circuito de forma sencilla los componentes necesarios para permitir, además del control manual a través de una llave y el potenciómetro, que la intensidad de la tira LED se controle con el sensor de luminosidad ya existente en el diseño, junto con el código correspondiente. Esto se presenta como mejora en esta etapa del diseño, aunque podría llevarse a cabo en una etapa posterior.

3. Análisis de Requerimientos

3.1. Requerimientos Funcionales

- Detección del movimiento del ambiente.
- Detección de la luminosidad del ambiente.
- Alternar el estado de las luces con botonera conectada a entradas digitales de la placa
- Encendido de las luces por conexión Relé.
- Utilizar el periférico ADC0 para lectura de una entrada analógica y generación de una señal digital (PWM) que controle la intensidad de la luz LED.
- LEDs que indiquen el estado del dispositivo (encendido/apagado).
- Botón encendido general del dispositivo.
- Utilizar una entrada digital de la placa para conexión a selector de modo
- Conexión WiFi para controlar el dispositivo.

3.2. Requerimientos No Funcionales

- Utilización de placa de desarrollo EDU-CIAA-NXP con Firmware v3.
- Utilización del formato "poncho" para diseño y fabricación del PCB.
- Programación en lenguaje C.
- Fecha límite para el proyecto: diciembre 2022.

4. Diseño del Hardware

En la figura 2 (ver *Objetivos Principales*) se incluye un diagrama de bloques del sistema a partir del cual se diseñó el hardware del poncho, el cual se utilizará en esta sección para organizar la descripción del poncho.

4.1. Entradas Inteligentes

Los sensores PIR (movimiento) y LUX (luminosidad) se conectan a una bornera ubicada en el poncho a través de cables que permitirán al usuario localizar dichos sensores en una ubicación que le resulte conveniente. Por ejemplo, el sensor PIR podría ubicarse en la entrada de la habitación y el sensor LUX en el lado exterior de la ventana.

El sensor PIR consiste en un módulo estándar para microcontrolador debido a su complejidad de implementación y conveniente esquema de conexiones (ver figura 3). El módulo posee tres conexiones, dos de las cuales representan la alimentación (+5V y tierra). La tercera conexión (PIR_IN en el esquemático) se utiliza para transmitir los datos desde el módulo del sensor hacia el poncho en forma de señal digital (0: nivel 0V; 1: nivel 3.3V). Desde el poncho, esta señal entraría en el microcontrolador a través de la puerta de E/S GPIO4.

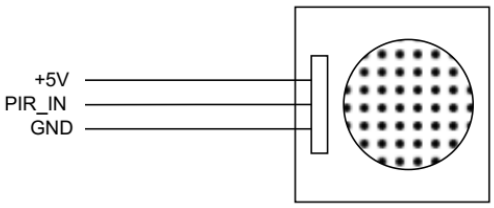


Figura 3. Terminales del sensor HC-S501 (PIR) utilizado.

El módulo posee además dos potenciómetros para regular la distancia máxima de detección de movimiento y el tiempo entre mediciones. En la hoja de datos del módulo se incluye una sección que explica cómo configurar el sensor.

El sensor LUX consiste en un LDR (resistor foto-dependiente o *photoresistor*), un componente pasivo que varía el valor de su resistencia con el grado de luminosidad captado. Este sensor se conecta a una bornera en el poncho mediante dos cables. La bornera se conecta a un divisor de tensión para que el pin analógico de la EDU-CIAA (CH1) pueda recibir un valor de tensión proporcional a la luminosidad captada por el LDR en el rango 0V-3.3V, donde 3.3V representa un grado de luminosidad alto y 0V representa un grado muy bajo de luminosidad.

4.2. Control Manual

La sección de control manual consiste en una botonera a través de la cual el operador puede encender o apagar las diferentes funciones del sistema. Cada uno de los botones que la componen contiene un circuito de manejo de rebote para que los cambios sean estables, el cual está formado por un resistor y un capacitor[1]. Con estos componentes se arma un circuito RC para inducir un retardo de 10ms que se combina con un retardo por software, también de 10ms.

Se eligió este valor de retardo porque resultaría imposible que una persona accionara un botón en un tiempo menor al indicado.

[1] En el informe anterior se utilizaron dos resistores y un capacitor, combinando un resistor de *pull-up* con un filtro RC pasa bajos. Esto no es necesario para un sistema como que se está desarrollando.

Utilizando la herramienta [2] se produce el siguiente gráfico, que muestra el tiempo que le toma a la señal pasar del cero lógico (no hay acción) al uno (pulsación).

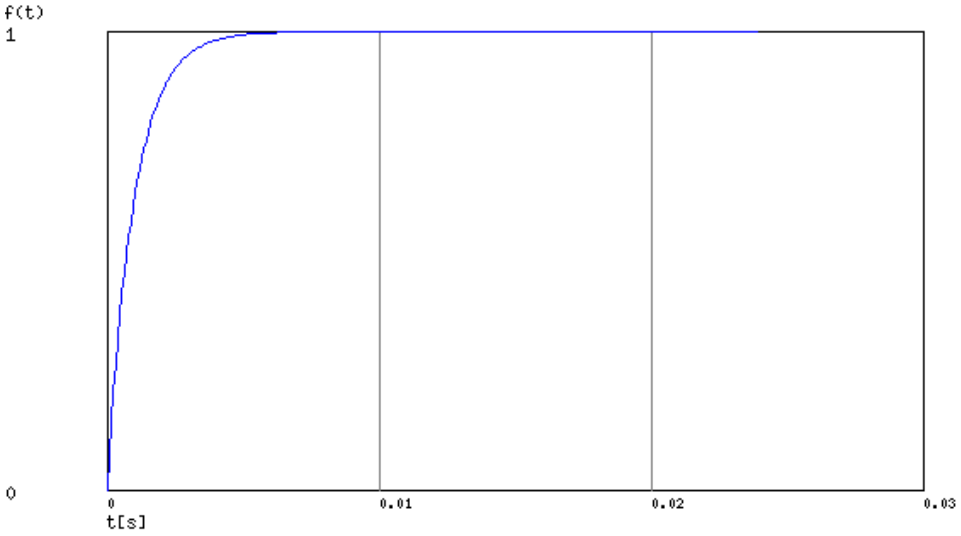


Figura 4. Tiempo de respuesta del botón con circuito de control de rebote.

El diseño de este circuito se puede mejorar, como se indica en [3], pero resulta suficiente a los efectos del sistema a desarrollar. El circuito resultante para uno de los pulsadores se detalla en la siguiente figura. También se incluye en el esquemático completo al final de este informe. Notar que el terminal de la EDU-CIAA indicado es a modo ilustrativo. Cada botón en el esquemático tiene indicado el terminal de la placa a dónde se conecta.

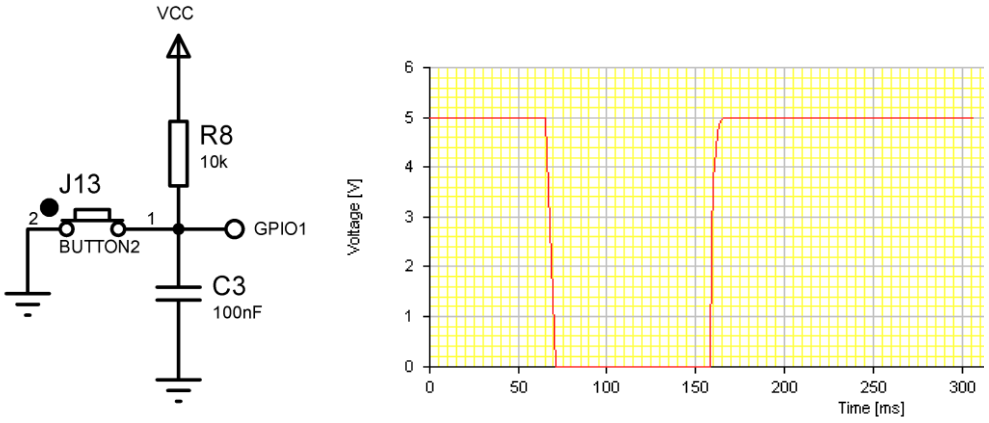


Figura 5. (a) Circuito esquemático para control de rebote de un botón. **(b)** Gráfico de señal de salida hecha en [11]. Notar que la transición bajo-alto se completa en 10ms.

Por otra parte, el potenciómetro utilizado para el control del brillo de la luz LED se conecta directamente al terminal CH2 de la placa. Los terminales de alimentación se conectan a los terminales +3.3V y a la tierra analógica (GNDA) en la EDU-CIAA siguiendo el esquema de la figura:

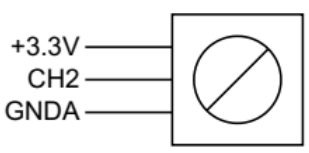


Figura 6. Esquema de conexión del potenciómetro a los terminales de la EDU-CIAA.

El potenciómetro es un modelo genérico de 10kΩ elegido por formar parte de un kit electrónico con el que ya se contaba. El terminal CH2 del periférico ADC recibe la señal del potenciómetro y la convierte en un valor numérico de 10 bits (rango 0...1023), el cual es utilizado por el firmware para generar un nivel de brillo que será aplicado sobre las luces LED.

En la figura 6 se utiliza una representación alternativa al modelo circuital del potenciómetro porque es más fácil ver las conexiones que se harán en el elemento real. Se ilustra el componente visto "de frente" o con el mando hacia arriba.

4.3. Salidas

Las salidas del sistema son todas señales lógicas provenientes de los terminales GPIO de la placa EDU-CIAA. Pueden representar el estado de los diversos componentes (encendido o apagado) así como controlar el estado de otros componentes del sistema (encendido o apagado de luz LED y luz 220V).

Las salidas de estado se conectan cada una a un LED de la placa EDU-CIAA. En iteraciones anteriores se había optado por utilizar un circuito serie resistor-LED para cada salida de estado, pero se decidió usar los componentes presentes en la placa para liberar espacio en el poncho para el trazado de pistas de circuito.

Originalmente se necesitaban cinco salidas de estado, pero se redujo el número de indicadores a tres: dos para el estado de los sensores y uno para el estado de la interfaz WiFi. En estas salidas, el LED apagado denota un '0' lógico (componente desactivado) y el LED encendido representa un '1' lógico (componente activado). Como indicador de alimentación del poncho, se aprovecha el incluido en la EDU-CIAA. El utilizar los componentes presentes en la EDU-CIAA permite no solo ahorrar espacio y tiempo de fabricación del poncho, sino que también reduce el consumo energético del sistema.

Por otra parte, las salidas de luces se modelan de una forma distinta. En el caso de la luz 220V, el terminal GPIO se conecta a la base de un transistor a través de un resistor de 2.2kΩ, el cual es parte de un circuito que evita que, cuando se pasa del estado 1 a 0 lógico, se produzca un pico de corriente en el inductor del relé. La siguiente figura ilustra la conexión del relé a un terminal GPIO de la EDU-CIAA.

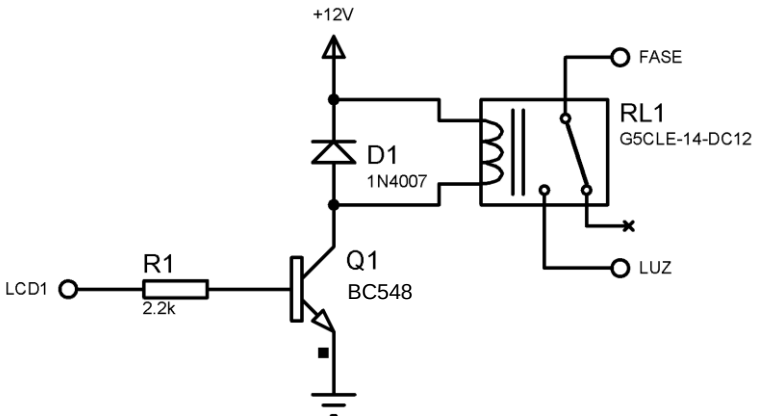


Figura 7. Conexión salida GPIO a relé de 12Vcon circuito de protección.

Los demás terminales del relé se conectan de tal forma que, cuando se haga circular corriente a través de la bobina (estado 1 lógico) el terminal indicado como FASE se conecte al indicado como LUZ, encendiéndose la luz 220V. Si se corta la circulación (estado 0 lógico), entonces el circuito se abrirá y la luz se apagará. El diodo D1 y el transistor Q1 de la figura 7 son el circuito de protección ante las respuestas de la bobina ante cambios en la circulación de corriente. Q1 se utiliza para poder entregar a la bobina del relé la corriente necesaria para activarlo, la cual no

puede ser entregada por el GPIO del MCU. En este caso, además permite alimentar la bobina del relé con 12V, lo cual no sería posible desde el GPIO del MCU. Notar que el terminal NC (*normally closed*) del relé no está conectado porque no se necesita hacer nada cuando no circula corriente por la bobina. En ese caso, podría utilizarse un relé de tipo SPST (*single pole, single throw*) en vez del SPDT (*single pole, double throw*) empleado. Sin embargo, los relés SPDT son más comunes y están presentes en los módulos de prueba para microcontroladores que se utilizaron para validar el diseño del sistema.

La conexión a la tira de LED es similar en principio. Como no se puede utilizar un relé por la alta frecuencia de alternancia entre niveles lógicos (PWM alterna la señal a velocidades muy superiores a las que el relé puede soportar), se hace uso del integrado ULN2003AN, un arreglo de siete transistores Darlington [5]. Podrían utilizarse otros métodos, como un transistor individual en vez de un integrado. Se eligió usar el circuito integrado por simplicidad. Los motivos para este diseño de circuito se explican en más detalle en la sección sobre la alimentación del circuito. La siguiente figura muestra las conexiones entre el terminal digital de la placa, el integrado ULN2003AN y las luces LED. El conector J5 denota la conexión de la tira de LED y no a la tira en sí. Su modelo de circuito es más complejo y no es necesario a efectos de diseño del sistema, puesto que este componente se encuentra armado y simplemente se conecta al poncho.

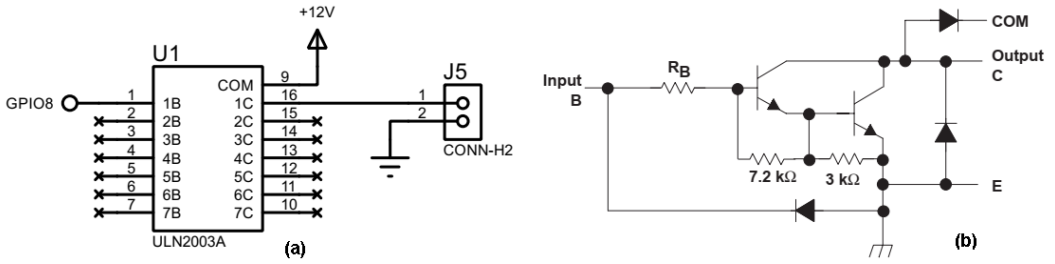


Figura 8. (a) Conexiones tira de LED. **(b)** Par Darlington dentro del ULN2003AN.

Un terminal GPIO de la EDU-CIAA se conecta a una de las entradas del ULN2003AN y una de las salidas se conecta al terminal de alimentación de la tira de LED. La alternancia entre el nivel alto y bajo en el pin digital de la placa harán que la entrada de alimentación de la tira de LED alterne a la misma frecuencia y con el mismo ciclo de trabajo. Esta alternancia a una frecuencia mayor a 25Hz, con diversos ciclos de trabajo, será percibida por el ojo humano como diferentes niveles de intensidad de luz o brillo.

4.4. Conexión WiFi

La conexión WiFi se modela mediante una bornera de cuatro terminales donde se conectará el módulo D1-mini. Es un módulo estándar para microcontroladores que provee conectividad WiFi a través del integrado ESP8266. Sólo se requieren las conexiones de alimentación a 3.3V y los terminales de comunicación RX/TX que se conectan a las entradas RX/TX (cruzadas) de la EDU-CIAA, siguiendo el diagrama siguiente:

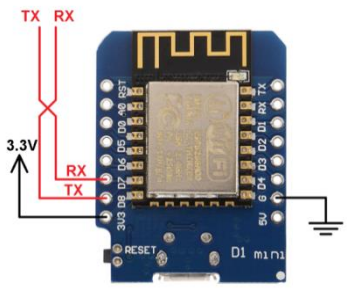


Figura 9. Conexiones módulo WiFi Wemos D1 mini

Para comunicarse con el módulo se hace uso de la UART-232 de la placa. El protocolo de comunicación utilizado es RS-232 a una tasa de transmisión de 9600 bps.

4.5. Alimentación

En primera instancia se decidió alimentar al poncho con los puertos de alimentación de la placa y alimentar al sistema entero a través de uno de los puertos USB de la EDU-CIAA. Sin embargo, la tira de LED conseguida requiere una alimentación de 12V. Con esto en consideración, la alimentación puede resolverse de varias maneras.

Primeramente puede alimentarse a la EDU-CIAA y toda la lógica de 5V y 3.3V mediante la conexión USB y utilizar una fuente separada para la tira de LED. Esto resulta conveniente desde el punto de vista de diseño porque simplemente puede agregarse un conector de alimentación al poncho para alimentar las luces y un circuito integrado ULN2003AN para controlar el brillo de las luces mediante una señal PWM. Desde el punto de vista de consumo, esta solución no impondría ninguna restricción sobre la EDU-CIAA al estar físicamente desconectada de la tira de LED.

A pesar de su simplicidad, la solución anterior complejizaría la interfaz de usuario al agregar cables de alimentación al circuito. Podría diseñarse una fuente de alimentación de 12V que, mediante reguladores de tensión, se bajasen los niveles a 3.3V y 5V para los componentes del poncho y la placa EDU-CIAA. La interfaz seguiría siendo simple por utilizar un único cable de alimentación, pero el diseño del sistema sería más complejo. Al tratarse de un prototipo, se optó por utilizar una fuente separada para la tira de LED con el integrado ULN2003AN, que funciona como una llave: un nivel lógico alto en una de las entradas permite la circulación de corriente por la salida correspondiente y un nivel bajo corta dicha circulación.

El relé de 12V consume 71mA aproximadamente. El consumo de la tira de LED puede analizarse observando el circuito que la compone. En [12] se incluye una tabla que muestra el consumo por cada metro de luces LED. En el caso particular de la tira que se usará en el proyecto, los LED son de tipo 5050 y hay 60 LEDs por cada metro de luces. Para estas características, el consumo es de 1.2A/m y la potencia es de 14.4W/m. Usándose un fragmento de medio metro de longitud, se obtiene un consumo de 0.6A y una potencia de 7.2W.

Respecto a la alimentación de los demás componentes, la EDU-CIAA provee cuatro puertos de alimentación: dos de 3.3V y dos de 5V, donde todos poseen un fusible configurable para protección, los cuales soportan una corriente máxima de 300mA [6]. El sensor de luminosidad y el módulo WiFi requieren alimentación de 3.3V, donde estos dispositivos tienen requerimientos de corriente de 5mA y 75mA[2], respectivamente. Otros componentes conectados a la línea de 3.3V incluyen al potenciómetro y los botones de control. El consumo de estos componentes puede considerarse pequeño.

El potenciómetro, el sensor LUX, el indicador de alimentación y los botones se conectarían a una misma entrada de alimentación de 3.3V pues su consumo sumado no supera el máximo soportado por el fusible de dicha entrada. El módulo WiFi utiliza su propia entrada de alimentación en la EDU-CIAA para mayor seguridad, pues posee un pico de consumo de 400mA, el cual no debería producirse pero puede ocurrir.

[2] 75mA es la corriente promedio. El módulo WiFi tiene un consumo pico de 400mA y un consumo en modo *stand-by* de 40µA.

4.6. Diseño y Fabricación del PCB

El circuito impreso del poncho se diseñó con dos limitaciones en mente. La primera es la separación de los conectores que lo conectarán a la EDU-CIAA-NXP, pues debe encajar sobre dicha placa sin que haya fuerza de contacto excesiva para evitar daños en el poncho y/o en la placa. Esta distancia es de 68.5mm[3]. La segunda restricción del diseño es el tamaño del circuito impreso. Se decidió utilizar una placa de 86x137mm para que el poncho tenga exactamente el mismo tamaño que la EDU-CIAA para que el montaje del mismo sea cómodo y compacto.

Otros aspectos considerados durante el diseño del poncho se relacionan con el ancho de las pistas, tamaño de *pads* y cantidad de conexiones manuales (puentes) a utilizar.

4.6.1. Pistas

Para las pistas se decidió utilizar un tamaño de 1mm para las señales de hasta 5V, con una separación de 0.2" (5.08mm) para que no haya problemas en el proceso de impresión, puesto que se va a hacer de forma manual. Suponiendo un espesor de pista de 6mils[4] y un aumento de temperatura de 5ºC respecto de la temperatura ambiente, se hace uso de la fórmula en [13] para calcular la máxima corriente que puede circular por las pistas del poncho:

$$I = k\Delta T^{0.44} A^{0.725}$$

En esta fórmula, *I* es la corriente que circula por la pista, ΔT es el incremento en la temperatura respecto de la ambiental y *A* es el la sección (área transversal) de la pista en mils cuadrados. La constante *k* depende de si la pista es interior o exterior al PCB. Las pistas interiores sólo existen en circuitos impresos multicapa, por lo tanto $k = 0.048$.

La sección se calcula como el producto entre el espesor y el ancho de la pista. Pasando ambos valores a mils[5] y multiplicando, se obtiene:

$$A = \varepsilon \cdot W = 6mils \cdot 1mm = 6mils \cdot 39.3701mils = 236.2206mils^2$$

Reemplazando en la fórmula original:

$$I = 0.048 \cdot 5^{0.44 ºC} \cdot 236.2206^{0.725} mils^2 = 5.122A$$

El valor de corriente obtenido es bastante alto comparado con los requerimientos de corriente de los componentes que se alimentan de 3.3V y 5V, por lo que no debería haber problemas de sobrecalentamiento en las pistas de 1mm.

Para la alimentación de 12V y el circuito de 220V se utilizaron pistas del doble y el triple del ancho, respectivamente. Repitiendo los cálculos realizados para las pistas de 1mm:

$$A_{2mm} = 6mils \cdot 2mm = 472.4412mils^2$$

$$I_{2mm} = k\Delta T^{0.44} A_{2mm}^{0.725} = 0.048 \cdot 2.03 \cdot 472.4412^{0.725} = 8.465A$$

$$A_{3mm} = 6mils \cdot 3mm = 708.66mils^2$$

$$I_{3mm} = k\Delta T^{0.44} A_{3mm}^{0.725} = 0.048 \cdot 2.03 \cdot 708.66^{0.725} = 11.358A$$

[3] Las dimensiones fueron dadas por la cátedra, pero están disponible en [15].
[4] 6mils equivalen a 0.152mm. Es el tamaño estándar para el espesor de las pistas de un circuito impreso.
[5] 1mm equivale a 39.37mils.

En ambos casos, los valores de corriente son mayores de los máximos esperados. Para el circuito de 220V, el valor máximo de corriente obtenido de 11.358A es mayor a la corriente estándar de 10A proveniente de la red de distribución eléctrica[6].

Para todos los tipos de pistas se intentó ubicar los componentes de forma tal que la longitud de la pista sea mínima y/o lo más directa posible, para mayor simplicidad y comodidad de análisis del circuito a la hora de realizar pruebas o trabajos de mantenimiento o actualización.

4.6.2. Puentes

El diseño de la placa intenta minimizar la cantidad de puentes lo más posible. Un puente es un fragmento de material conductor que conecta dos pistas desde el lado de los componentes por no haber espacio suficiente para conectarlas directamente en el lado de soldadura. En el software de diseño de PCB Proteus se modela a un puente como una pista en la cara de los componentes, sin embargo, en la implementación real será un cable o material conductor similar.

4.6.3. Plano de Tierra

Una adición al circuito es el plano de tierra, que simplifica notablemente el diseño del poncho teniendo en cuenta el reducido tamaño del PCB. Consiste en un relleno de cobre asociado a los terminales de tierra de todos los componentes de la placa. En el diseño se tuvo especial cuidado de dejar un espaciado de 0.2" entre las pistas y el plano de tierra para evitar conexiones accidentales debido al proceso de impresión y transferencia del diseño a la placa. Se usó un ancho de pista de 1mm para las conexiones de los *pads* al plano de tierra y se instruyó al software para que no se permitiera la formación de "islas" en el relleno.

Se define a una "isla" en este contexto como una zona de relleno de cobre que no está físicamente conectada a nada. Se forman cuando se hace pasar una pista por una zona muy estrecha y no queda el suficiente espacio físico debido a las reglas de diseño para que las zonas del relleno de cobre estén completamente unidas. Para evitar la formación de islas se decidió por limitar la cantidad de componentes en el poncho y eliminar algunas de las conexiones innecesarias, como terminales de E/S no usados o conexiones a tierra redundantes. Por esta razón se decidió también hacer simplificaciones en el circuito esquemático, como eliminar el filtro RC de los pulsadores y usar un circuito RC simple para retardos o reorganizar las conexiones para que las pistas quedaran prolijas en el PCB.

Se decidió terminar el relleno de cobre a la salida de la lógica de control del relé para asegurar la completa desconexión del circuito de 220V.

4.6.4. Pads

Los *pads* que se utilizan en el diseño del circuito impreso corresponden al elemento C80-40 de diseño en Proteus. Este *pad* por defecto tiene un agujero es de 40mils y una corona de cobre es de 80mils. Si bien el mínimo establecido por la cátedra es C70-30 (30mils de agujero y 70mils de corona de cobre), se consideró usar un tamaño mayor para evitar problemas de transferencia en el proceso de impresión y planchado del poncho. También se evitarían problemas en la etapa de perforación de los *pads*, pues de usar un tamaño menor se corre el riesgo de dañar o destruir completamente el *pad* al perforar, especialmente considerando que el tamaño de la mecha que se utilizará es mayor a 0.7mm (aproximadamente 28mils).

[6] En [14] se menciona que el valor máximo de corriente para aparatos de iluminación y consumo debe ser de 10A. Los interruptores y tomacorrientes que se consiguen comercialmente se menciona el mismo valor de corriente máxima soportada.

Para los componentes que sean más susceptibles al estrés mecánico (borneras, relés, etc.), se optó por *pads* de mayor tamaño para hacer de soporte para el componente mientras brindan conectividad al circuito. El tamaño elegido es C120-60 (corona de 120mils).

Para el zócalo del circuito integrado y el transistor se optó por utilizar un *pad* ovalado, para mayor separación entre conexiones y así evitar problemas con la transferencia del diseño a la placa de circuito, pues las conexiones de estos componentes son adyacentes y podrían formarse puentes debido a irregularidades en el proceso de planchado.

4.6.5. Reglas de Diseño y Ubicación de los Componentes

Con el objetivo de garantizar las distancias mínimas entre componentes, pistas y conexiones se decidió modificar el conjunto de reglas de diseño estándar (DRC) provistas por el programa de diseño Proteus. Las mismas son:

- Espacio entre *pads* (*pad-pad clearance*): 20mils
- Espacio entre un *pad* y una pista (*pad-trace clearance*): 20mils
- Espacio entre pistas (*trace-trace clearance*): 20mils
- Espacio entre gráficos (*graphics clearance*): 15mils
- Espacio de montaje (*edge-slot clearance*): 20mils

Además, se decidió ubicar los componentes teniendo en cuenta su función y la longitud de las pistas utilizadas en su conexión. Las borneras se ubicaron en los bordes del poncho para facilitar la conexión y desconexión de componentes externos. Particularmente, las borneras del sensor de luminosidad (LUX) y la conexión a la tira de LED se ubicaron lejos del extremo para aislar aún más el circuito de 220V.

Los botones se ubicaron en el extremo inferior para facilitar la interacción con el usuario y la conexión del módulo WiFi es tal que el mismo se monta directamente sobre el poncho sin tener que realizar ninguna conexión adicional.

El conector de entrada del potenciómetro se diseñó teniendo en cuenta que el mismo se ubicará sobre un posible panel de operación en vez de directamente sobre el poncho, por cuestiones de espacio y robustez requerida por este componente.

Asimismo se han dejado conexiones de alimentación auxiliares (5V, 3.3V y tierra) para facilitar la expansión, prueba y/o mantenimiento del circuito. En el conector de la placa EDU-CIAA-NXP también se dejaron algunos terminales de tierra extra para brindar un apoyo y conexión robustos al poncho y demás componentes.

Los componentes discretos y el zócalo del integrado se ubicaron teniendo en cuenta sus conexiones. Durante el diseño se buscó minimizar la cantidad de puentes utilizados y este objetivo fue clave para la ubicación de estos componentes. Otros objetivos para la ubicación de los componentes incluyen la reducción de espacio utilizado en el poncho (para una eventual reducción en el tamaño de la placa terminada) y la reducción en el número de *pads*. Al ser una placa de fabricación casera, un menor número de perforaciones reducirían el tiempo de fabricación y ahorraría esfuerzo, así como un menor tamaño de la placa reduciría el tiempo de atacado del cobre.

En la figura 11 se incluye una ampliación a página completa del diseño del poncho terminado. Al final de este informe se incluye la misma imagen en escala 1:1, la cual será impresa en papel fotográfico y utilizada para la fabricación del circuito. Notar que en la siguiente imagen se incluyen todas las capas del PCB solapadas: serigrafía, capa de componentes, capa de soldadura (con texto) y capas de perforación. En la imagen del anexo 11.3 se incluyen las capas en hojas separadas para facilitar la lectura.

Tener en cuenta además que las pistas del lado de los componentes representan los puentes que deben hacerse en el circuito terminado, no son pistas de un circuito impreso bicapa.

4.6.6. Fabricación del PCB

La fabricación de la placa de circuito impreso se llevó a cabo en el laboratorio ATEI[7] durante las dos primeras semanas del mes de diciembre de 2022. Se partió del diseño aprobado entonces, el cuál fue corregido siguiendo los errores y dificultades señaladas en la sección 7.2.1.

Se usó el proceso de transferencia térmica para fabricar el PCB. Este consiste en imprimir en escala 1:1 la cara de soldadura del diseño sobre una hoja de papel ilustración. Se prepara previamente una placa de sustrato fenólico virgen con las dimensiones exactas del poncho y se ubica de forma centrada el diseño sobre la placa, con el lado impreso contra la placa. Luego, se calienta utilizando una plancha para ropa y se quita el excedente de papel con agua. Se hacen las correcciones debidas con una fibra indeleble.

Después, en el área de mecanizado del ATEI, se perforó un agujero para sujetar la placa durante el proceso de atacado químico. Una vez completado el atacado, se perforaron los pads de acuerdo a las dimensiones de los componentes. En esta etapa, como se explica en 7.2.1, uno de los agujeros del conector del módulo WiFi no estaba alineado con los demás, lo que provocó que debido al estrés mecánico en la alineación del conector, se soltara la pista de alimentación del módulo. Luego, se cubrió la placa con una capa de resina (flux) y, una vez seca, se procedió al montaje y soldadura de los componentes. La siguiente figura muestra la placa terminada de ambos lados. Notar que aún no se corrigió la conexión de la tira de LED mencionada en 7.2.1.

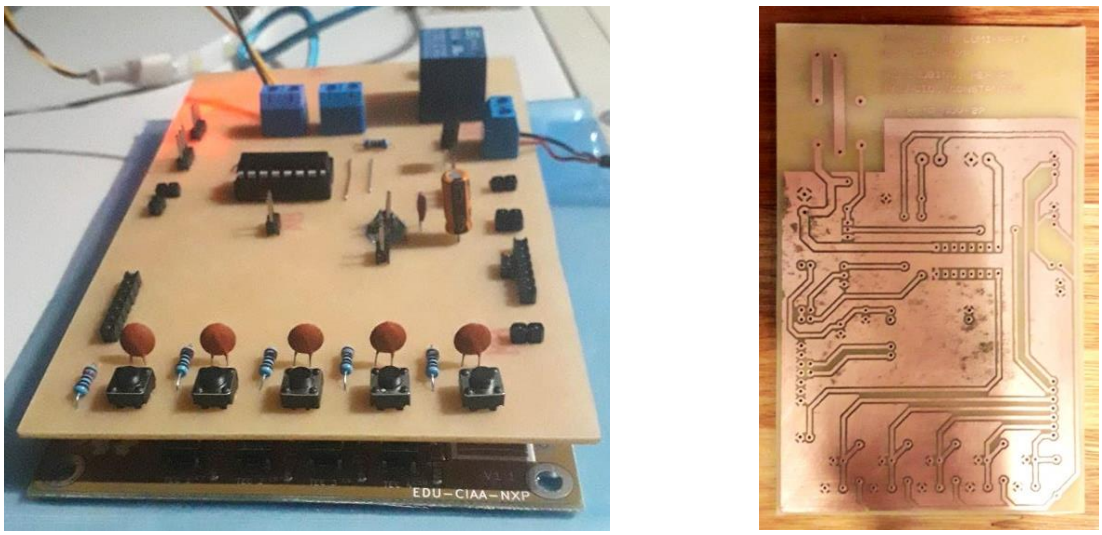


Figura 10. Placa de circuito impreso terminada: (izq.) componentes; (der.) soldaduras

[7] Departamento de Electrotecnia, Facultad de Ingeniería.

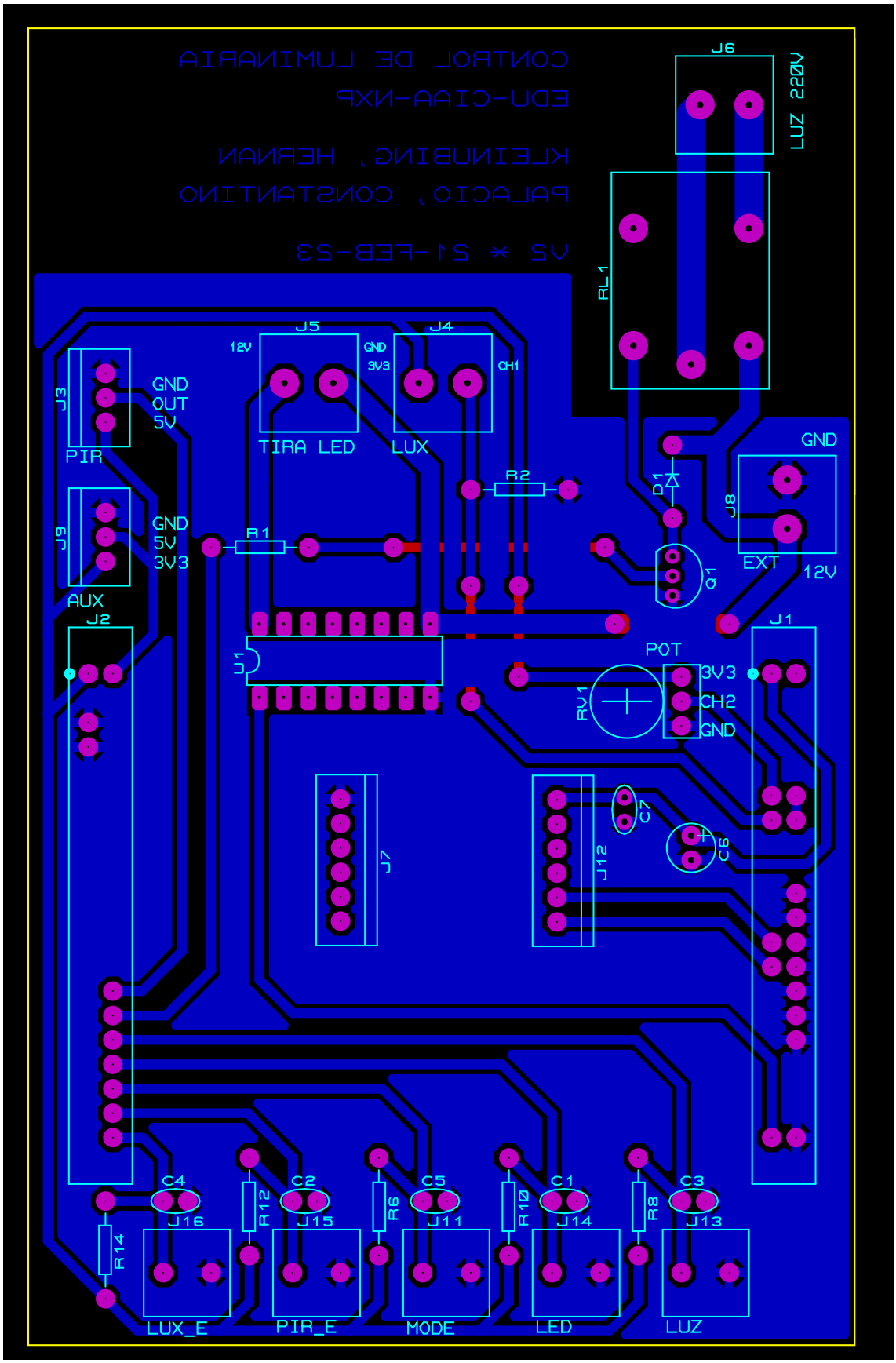


Figura 11. Diseño del poncho terminado.

5. Diseño del Firmware, Simulación y Depuración

Al igual que en el diseño del hardware, el diseño del firmware se realizó de forma modular respetando el diagrama de bloques de la figura 2. Las definiciones de puertos de E/S se hacen en un archivo de código compartido por todos los módulos de programación (`puertos.h`).

Todas las unidades de programa utilizan la librería sAPI (`sapi.h`) del firmware versión 3 para facilitar el control de las funciones de la placa EDU-CIAA-NXP.

La arquitectura de software utilizada se basa en un bucle infinito (super-loop) que encapsula todas las funciones de control. Dentro de éste, se hace una revisión periódica de las entradas de control (100ms) y se actualiza el estado del relé (500ms). El manejo de los retardos es no bloqueante, es decir, se pueden realizar otras tareas hasta que pase el tiempo especificado.

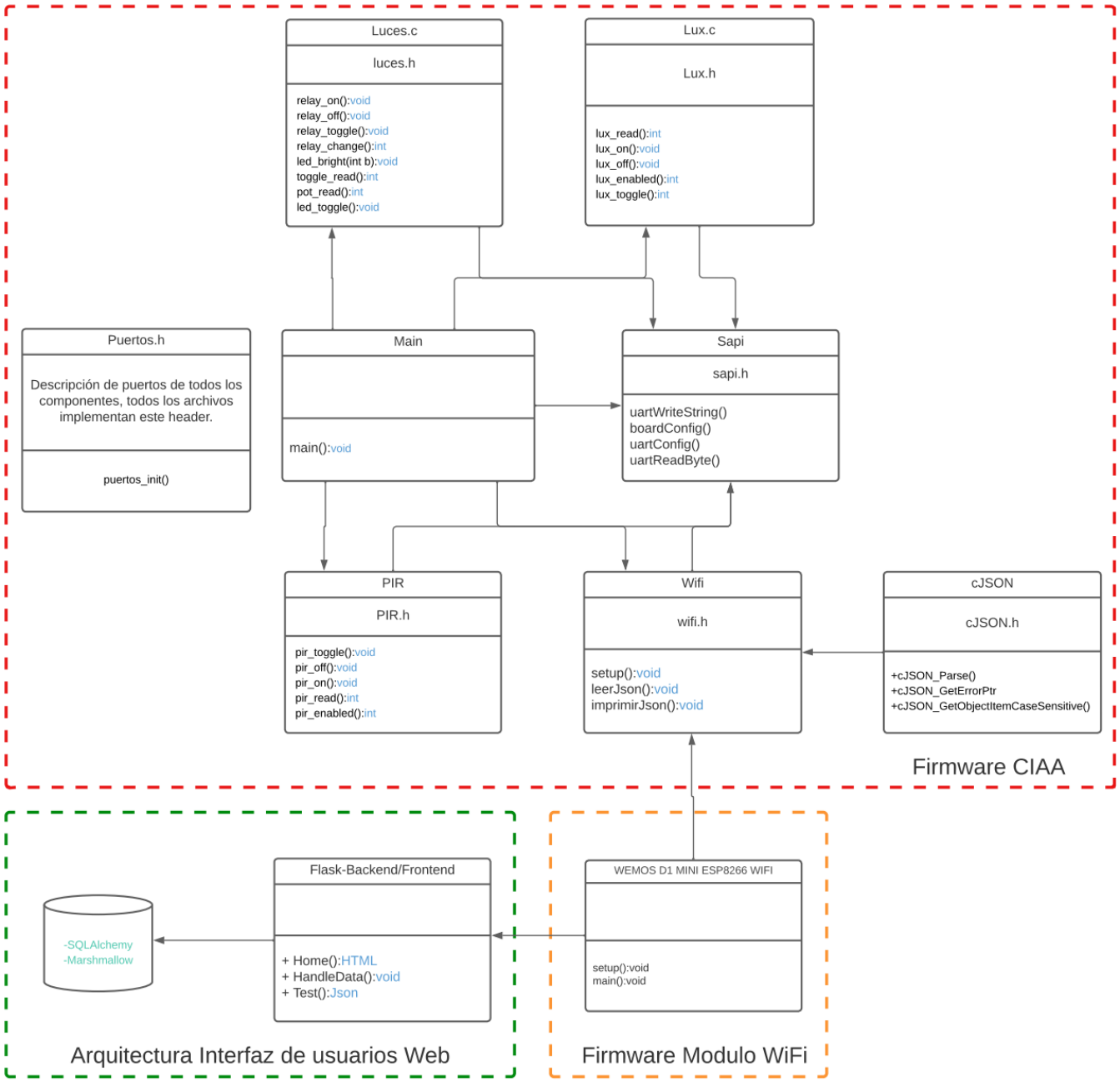


Figura 12. Diagrama de módulos de programación que componen el sistema.

Durante el desarrollo del proyecto no se tuvo en consideración el uso del modo de bajo consumo del MCU. Se lo consideró inicialmente para ser implementado en una versión futura del prototipo.

A continuación se presenta el pseudocódigo del programa principal y del módulo que se encarga de inicializar los puertos y periféricos de la placa. Los nombres de cada módulo se indican en negrita y cada función se indica subrayando su nombre. Esta convención se mantiene para cada pseudocódigo presentado en el informe. El programa completo se encuentra disponible en el repositorio Web indicado en 11.6.

```
Programa principal {
	inicialización de placa y periféricos
	configuración de retardos
	bucle infinito {
		si pasaron 100ms {
			si el modo WiFi está activado {
				leer controles Web y actualizar periféricos
			} sino {
				leer controles manuales y actualizar periféricos
			}
		}
		si pasaron 500ms {
			actualizar estado del relé
		}
	}
}

Módulo puertos {
	función puertos_init {
		inicializar puertos GPIO y periféricos
	}
}
```

5.1. Manejo de los Sensores

Para el control de los sensores se optó por utilizar un bucle que leyera el estado de las entradas de ambos sensores cada 100ms. Este mismo bucle se utiliza para leer las entradas de control en general y actualizar los estados de los componentes del sistema.

Cada uno de los sensores posee su propio controlador (`pir.h` y `lux.h`) para modularizar la solución al problema y facilitar la detección y corrección de errores en la programación así como la modificación del código para incluir nuevas funcionalidades. Las funciones previstas para todos los sensores son:

- Activación del sensor
- Consulta de estado actual (activado/desactivado)
- Lectura de entrada de control y ajuste del estado
- Lectura de entrada de datos del sensor y retransmisión al programa principal

Estas funciones son invocadas desde el programa principal para luego decidir qué debe hacerse con las luces en base a las respuestas del sensor. A continuación se presenta el pseudocódigo de los módulos de control de los sensores:

```
	Módulo lux {
		función lux_on {
			habilitar sensor de luminosidad
		}
```

```
        función lux_off {
                deshabilitar sensor de luminosidad
        }

        función lux_toggle {
                si se pulsó el botón de control de sensor LUX {
                        conmutar estado
                }
                actualizar indicador de estado
        }

        función lux_enabled {
                devolver el estado actual del sensor
        }

        función lux_read {
                si el sensor está activado {
                        leer y devolver un valor del ADC
                } sino {
                        devolver un valor de corte (dato inválido)
                }
        }
}

Módulo pir {
        función pir_off {
                deshabilitar sensor de movimiento
        }

        función pir_on {
                habilitar sensor de movimiento
        }

        función pir_toggle {
                si se presionó el botón de control del sensor PIR {
                        conmutar estado del sensor
                }
                actualizar indicador de estado
        }

        función pir_enabled {
                devolver estado actual del sensor
        }

        función pir_read {
                si el sensor está activado {
                        leer sensor
                        devolver valor leído
                } sino {
                        devolver "apagado"
                }
        }
}
```

5.2. Luces

Este módulo (`luces.h`) se encarga del control de las salidas de luces LED y 220V. Para la luz 220V posee funciones de encendido/apagado, consulta de estado y lectura de entrada de control. Para la tira de LED posee funciones de consulta de estado, lectura de entrada analógica (potenciómetro), lectura de terminal de control y ajuste de brillo mediante PWM.

Las funciones de lectura de entradas de control y consulta se invocan cada 100ms, cuando se realiza la consulta de todas las entradas de control del sistema. Las funciones de actualización de las salidas de luz se invocan desde el programa principal cada 500ms y simplemente escriben un valor en la salida correspondiente dependiendo del estado actual. Para el caso de la luz 220V, si el estado cambió respecto de la lectura anterior, escribe el estado actual y sino no cambia nada.

Para la tira de LED, siempre escribe un valor de *duty cycle* recibido como argumento, aun cuando no ha cambiado desde la lectura anterior. Debe tenerse en cuenta que la salida a LED sólo se actualiza si el componente se encuentra activado.

El pseudocódigo del módulo de control de las luces se presenta a continuación:

```
Módulo luces {
	función relay_change {
		devolver si hubo o no cambios en el estado del relé
	}

	función relay_off {
		apagar el relé
	}

	función relay_on {
		encender el relé
	}

	función relay_toggle {
		actualizar estado y conmutar relé
	}

	función toggle_read {
		si se presionó el botón de control del relé {
			conmutar estado
		}
		devolver estado actual
	}

	función led_bright {
		encender/apagar indicador de estado de luz LED
		configurar brillo de luz LED (PWM)
	}

	función led_on {
		poner estado de luz LED en 'encendido'
	}

	función led_off {
		poner estado de luz LED en 'apagado'
	}
	función led_toggle {
		si se presionó el botón de control de luz LED {
			conmutar estado de luz LED
		}
		actualizar salida de estado
	}

	función pot_read {
		leer y devolver un valor del ADC
	}
}
```

5.3. WiFi

Es posible controlar el proyecto a través de una comunicación Web, para eso utilizamos un módulo WiFi que se comunica usando la comunicación UART. Para modificar los valores del proyecto se pone a disposición una aplicación Web con botones que activan o desactivan las funciones del proyecto.

5.3.1. Poncho/EDU-CIAA

- WiFi: El propósito de esta librería es resolver la comunicación UART con el módulo WiFi, llevar la información recibida a un formato que nos permita utilizar esa información en el módulo rector.
- cJSON: Librería que utilizamos para hacer la interpretación de la información que recibe la CIAA a través del módulo WIFI

5.3.2. Módulo WiFi Wemos D1 Mini

- Esp02s. Dentro del módulo WiFi se utiliza un firmware encargado de:
  - Establecer la conexión WiFi entre el módulo y el Router WiFi.
  - Establecer la comunicación Serial entre el módulo y la CIAA.
  - Obtener la información almacenada en la base de datos remota.
  - Transmitir la información obtenida de la base de datos a la CIAA en formato JSON.
- Librerías
  - WiFiClient: Se usa para hacer consultas de tipo “Get” al endpoint /test
  - SoftwareSerial: Usada para establecer una comunicación serial virtual con la CIAA.
  - ESP8266WiFiMulti: Librería que se encarga de la conexión entre el módulo y el router WiFi.
  - Arduino: Librerías arduino manejo de pin y delay.

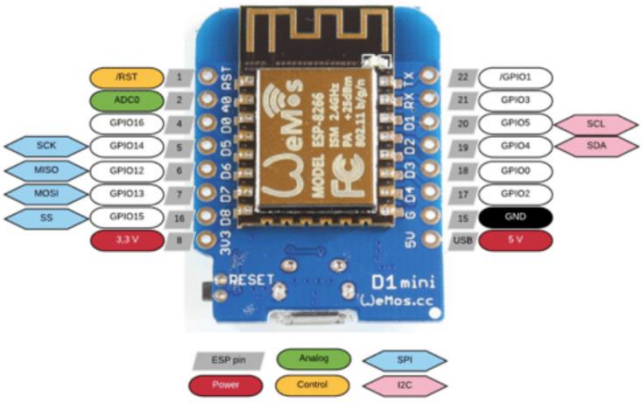


Figura 13. Módulo WiFi Wemos D1 Mini.

A continuación se incluye el pseudocódigo del módulo `wifi.h`, del módulo para conmutar entre entrada manual e interfaz Web; y del firmware del módulo Wemos. La función `campo_json` no es una única función, sino que representa un conjunto de funciones idénticas, una para cada campo de datos JSON.

```
Módulo mode {
	función mode_off {
		desactivar control WiFi
	}


	función mode_on {
		activar control WiFi
	}

	función mode_toggle {
		si se presionó el botón de control de modo {
			conmutar estado
		}
		actualizar indicador de estado
	}
}

Módulo wifi {
	Función campo_json {
		leer del JSON el valor del campo
		si es 'verdadero' {
			encender la salida de estado
			devolver 'verdadero'
		} sino {
			apagar la salida de estado
			devolver 'falso'
		}
	}

	Función leer_json {
		reservar memoria para la información recibida (buffer)
		mientras no se llegue al final o no se lea CR {
			si se presionó el botón de control de modo {
				modificar el estado del modo
				terminar función
			}
			si hay un dato en la UART-232 {
				agregar el dato al final del buffer
				incrementar puntero de fin de buffer
			}
		}
		si el tamaño del buffer supera al máximo {
			terminar buffer con carácter nulo
			decodificar información
			si hay error {
				imprimir notificación
				liberar buffer
				devolver 'falso'
			}
		}
		liberar buffer
		devolver 'verdadero'
	}

	Función imprimir_json {
		imprimir información decodificada por UART
	}

	Función wifi_setup {
		configurar UART-232 a 9600bps
	}
}
```

```
Firmware Wemos {
	Función Setup {
		iniciar comunicación serie
		conectar con modem WiFi
	}

	Función Loop {
		repetir {
			realizar consulta a backend
			si no hay error {
				obtener JSON de la respuesta
				enviar JSON por comunicación serie
			}
			esperar antes de repetir
		}
	}
}
```

5.4. Backend/Frontend

5.4.1. Backend

Se utiliza una librería de Python llamada Flask, la misma contiene una interfaz REST para manejar la interacción con el usuario como también la comunicación WiFi con el módulo Wemos D1 mini ESP8266.

5.4.1.1. Endpoints

- `http://{url}/`
  Este endpoint es la ventana principal, donde se obtiene el input del usuario que luego se almacena en la base de datos.
- `http://{url}/test`
  Este endpoint es el que consume el módulo WiFi para obtener la información desde la base de datos

5.4.2. Frontend

Para poder brindar una mejor experiencia de usuario se utilizó la librería Bootstrap que brinda componentes ya definidos para usar con HTML, también se agregaron métodos en JavaScript que permite modificar valores en la base de datos sin la necesidad de refrescar la página**.**

5.4.3. Base de Datos

Para poder persistir la información que ingresa el usuario utilizamos una base de datos, hacemos uso de las librerías:

- SQLAlchemy: Facilita el guardado y la recuperación de la información.
- Marshmallow: Facilita la descripción de esquemas que luego son usados para la transformación a formato JSON.

5.5. Interfaz de Usuario

La interfaz de usuario contiene botones, los cuales permiten al usuario activar o desactivar los componentes del poncho a través de un clic. Para la base de datos se utiliza JavaScript, lo que evita tener que recargar la página con cada operación. Esto ahorra tiempo y brinda una mejor experiencia de usuario.

Otro aspecto de la interfaz es el uso de una paleta de colores de alto contraste, que facilita la lectura de la información.

El flujo de datos desde la interfaz web hacia la EDU-CIAA se ilustra en la figura 15.

Figura 14. Interfaz de usuario WiFi.

La figura 14 muestra la interfaz de usuario web. Incluye todos los controles disponibles en la botonera: los botones activan/desactivan componentes del sistema y la barra deslizante ajusta la intensidad de la tira de LED.

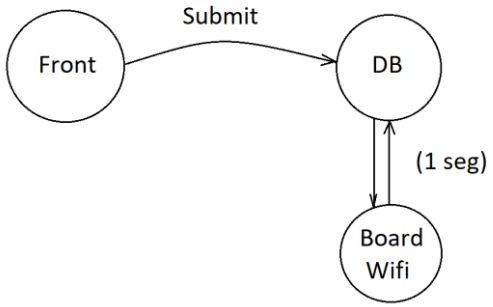


Figura 15. Flujo de datos desde la interfaz de usuario a la placa EDU-CIAA.

5.6. Pruebas de Funcionamiento

Para simplificar las pruebas de funcionamiento del sistema, normalmente hechas sobre un circuito prototipo en forma de *protoboard*, se fabricó una placa de circuito preliminar sobre una placa de prototipos de 5x10cm. En dicha placa se incluyeron los botones, salidas de estado y conexiones para el sensor PIR y módulo de relé Arduino. La figura 16 ilustra la placa de pruebas con el sensor PIR y el módulo de relé usado en las pruebas. Tener en cuenta que el circuito del módulo de relé es funcionalmente equivalente al propuesto en el esquemático del sistema.

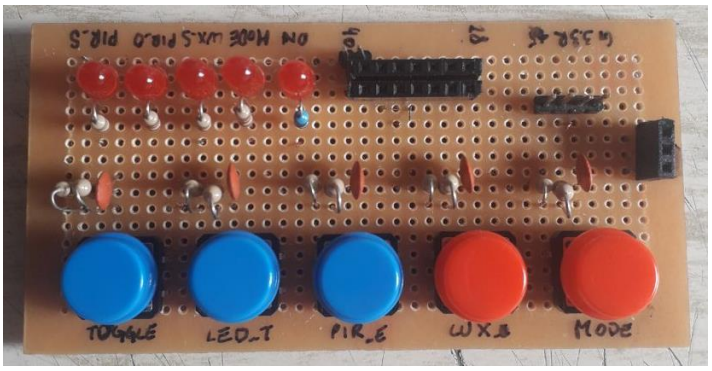


Figura 16. Placa de pruebas.

La placa de pruebas está basada en el esquemático presentado en el Informe de Avance I, por lo que las salidas de estado no son usadas a excepción del indicador de alimentación y la entrada del sensor PIR. Posee además un conjunto de conectores para el intercambio de señales con la EDU-CIAA-NXP. Un diagrama de conexionado se incluye en el apéndice 9.5. Notar que no todas las conexiones son utilizadas, pues los requerimientos de entrada y salida del sistema han cambiado desde el informe anterior.

5.6.1. Conexión de componentes

En las figura 17 se puede ver al módulo de relé y al sensor PIR conectados a la placa de pruebas del sistema. Esta placa se diseñó de forma tal que el sensor PIR encajase directamente en ella en vez de tener que conectarlo con un cable.

Para el módulo de relé se incluyeron cuatro terminales, de los cuales se usan tres. El cuarto es para la alimentación de 5V requerida por el módulo de cuatro canales (cuatro relés). En el caso del circuito real a usar en el poncho, se utilizaría un esquema de conexión similar, pero con 12V en lugar del de 5V que incluyen los módulos de microcontrolador estándar.

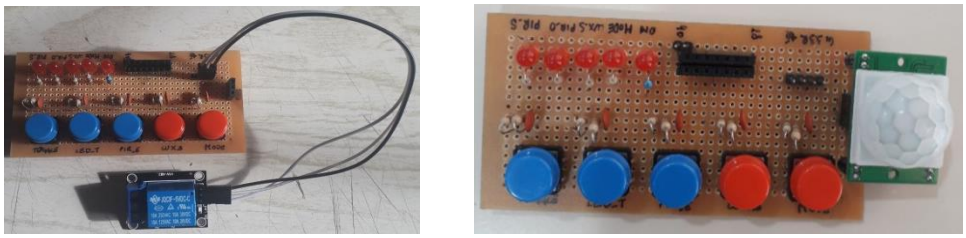


Figura 17. (a) Conexión de módulo de relé. **(b)** Conexión de sensor PIR.

Por cuestiones de espacio en la placa de pruebas no se han incluido los periféricos analógicos ni las conexiones de WiFi. Éstos se conectarán directamente a la EDU-CIAA para hacer las pruebas.

5.6.2. Conexión a EDU-CIAA-NXP

Respetando el diagrama de conexiones se completa la interfaz entre la placa de pruebas y la placa de desarrollo EDU-CIAA-NXP. Luego, se compila y ejecuta el programa escrito sobre el hardware y se procede a probar el funcionamiento del sistema. El sistema de pruebas completamente conectado se ilustra en la figura 18. Notar que no se incluyeron ni el sensor LUX ni el potenciómetro. Esto se debe a que el funcionamiento de estos componentes ya se probó por separado. Las pruebas que se realizadas en las secciones 5.6.3 a 5.6.6 son sobre la integración entre la botonera y la interfaz WiFi.

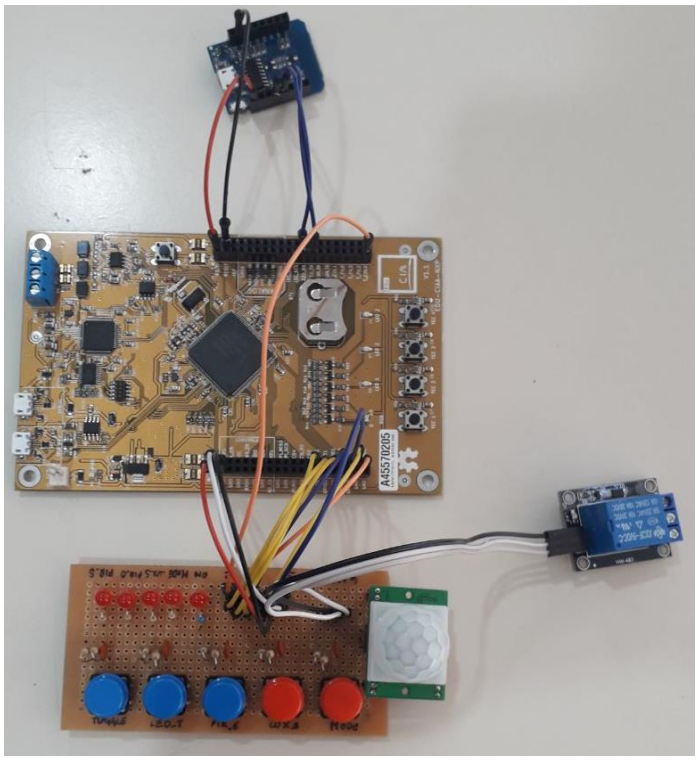


Figura 18. Sistema de prueba conectado.

Las entradas de la placa de pruebas son botones conectados a un circuito anti-rebote como el descrito en el informe anterior: un resistor de *pull-up* y un filtro RC pasa bajos. Accionando cada una de ellas puede verse su efecto en el comportamiento del sistema. Para encender o apagar la luz o la tira de LED, se acciona TOGGLE o LED_TOGGLE, respectivamente.

5.6.3. Pruebas de integración módulo WiFi

Se realizaron pruebas de integración del módulo WiFi con la aplicación web, en la misma se conectó el módulo WiFi D1 mini a la red WiFi y se consumió el servicio levantado de forma remota. En este caso se obtuvo una respuesta con los valores correspondientes a los parámetros del proyecto en formato JSON de forma correcta, dichos datos se corresponden con el valor almacenado en la Base de Datos (ver Figura 19).

Figura 19. Respuesta servicio remoto en formato JSON.

5.6.4. Prueba integración módulo WiFi con EDU-CIAA

Se probó la comunicación entre el módulo WiFi D1 mini con la placa EDU-CIAA, la misma se da a través de una conexión por UART. El módulo WiFi recibe un dato del backend y se lo comunica a la EDU-CIAA, la última espera a que llegue el dato y lo imprime a través de la UART USB (ver Figura 20). Se comprobó que la información no contiene errores sintácticos y se mantiene por períodos de tiempo prolongado.

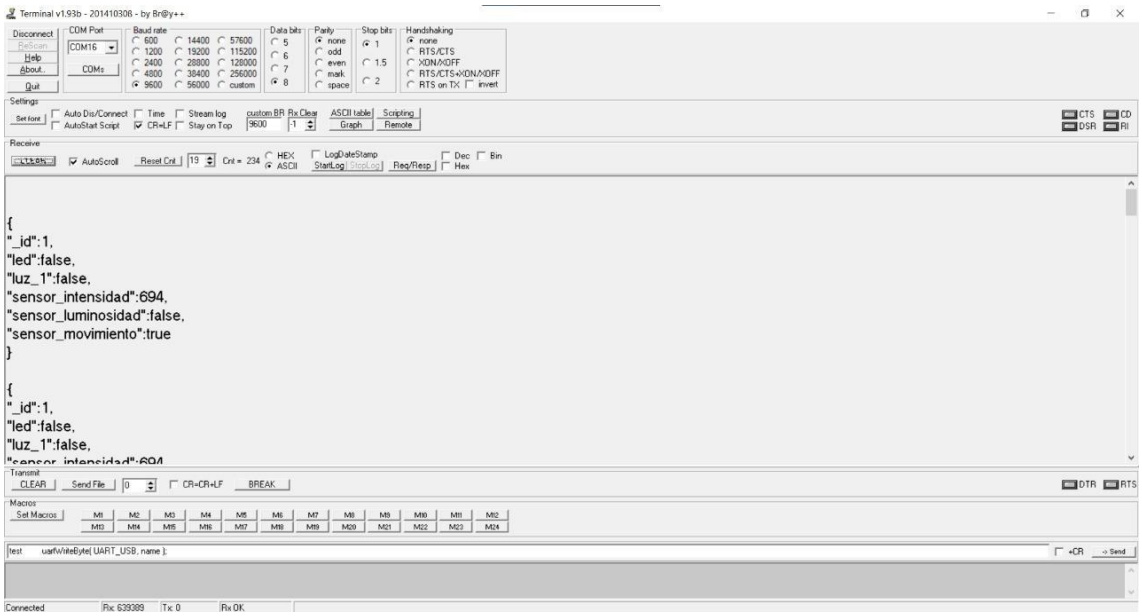


Figura 20. Comunicación entre módulo WiFi y EDU-CIAA.

5.6.5. Persistencia de información en la base de datos

Se comprobó la correcta modificación de los datos desde la web de control (ver Figura 14) y el backend, en dicha prueba se activan y desactivan los botones disponibles y luego se comprobó en la base de datos la correcta modificación y persistencia en el tiempo.

5.6.6. Test de integración

Se realizó un test de integración, que consistía en tener todos los periféricos conectados a la placa. Luego, se activaron y desactivaron desde la web de control. Se pudo observar el encendido/apagado de los componentes del sistema, así como la variación en la intensidad de la luz LED.

También se probó el cambio de modo desde WiFi a manual, en el mismo se probaron los botones que manejan el encendido y apagado de los periféricos, resultando satisfactoria.

6. Ensayos y Mediciones

Pruebas realizadas (ver video):

Prueba	Resultado	Comentario
Software		
Persistencia en la base de datos	Ok	
Activación de botones sin refresco de página	Ok	
Hardware		
Funcionamiento manual		
Luz	Ok	Encendido/Apagado
Tira de Led	Ok	- Encendido/Apagado - Intensidad con potenciómetro.
Sensor de movimiento	Ok	Encendido/Apagado y tiempo de refresco.
Sensor de luminosidad	Ok	Encendido/Apagado
Funcionamiento WiFi		
Luz	Ok	Encendido/Apagado
Tira de Led	Ok	- Encendido/Apagado - Intensidad con barra de intensidad.
Sensor de movimiento	Ok	Encendido/Apagado y tiempo de refresco.
Sensor de luminosidad	Ok	Encendido/Apagado
Prueba de tiempo prolongado		
Estado no se vea modificado.	Ok	
Proyecto responda a nuevas acciones luego del tiempo determinado	Ok	
Comunicación serial	Ok	Comunicación JSON desde aplicación web hacia módulo WiFi y luego EDU CIAA

6.1. Software

6.1.1. Persistencia en la base de datos

Para verificar el correcto funcionamiento de la base de datos se realizaron las siguientes pruebas:

1. Modificación de los valores en tiempo real verificando que la respuesta del endpoint '/test' se corresponda al valor modificado.
2. Definir los valores a un estado conocido, apagar la aplicación y volver a encender. Verificar que los valores previamente definidos sean los mismos.

6.1.2. Activación de botones sin refresco de página

a. Verificar el valor definido en la variable a comprobar haciendo uso del endpoint '/test', hacer clic sobre el botón que modifica dicho valor y por último comprobar en el endpoint mencionado que el valor se modificó sin que la página se actualice.

6.2. Hardware

6.2.1. Funcionamiento manual

Las pruebas de funcionamiento manual del proyecto se realizaron con todos los dispositivos conectados a la vez, se realizaron las pruebas de funcionamiento del dispositivo utilizando los botones físicos.

a. Luz
    i. Encendido/Apagado.
b. Tira de LED
    i. Encendido/Apagado.
    ii. Intensidad con potenciómetro.
c. Sensor de movimiento
    i. Sensor activado.
    ii. Sensor desactivado.
d. Sensor de luminosidad
    i. Sensor activado
    ii. Sensor desactivado.

6.2.2. Funcionamiento WiFi

Las pruebas de funcionamiento utilizando únicamente la comunicación WiFi se realizaron con todos los dispositivos conectados a la vez. Se realizaron las pruebas del funcionamiento del dispositivo utilizando los botones virtuales:

a. Luz
    i. Encendido/Apagado.

b. Tira de LED
    i. Encendido/Apagado.
    ii. Intensidad con potenciómetro.

c. Sensor de movimiento
    i. Sensor activado.
    ii. Sensor desactivado.
    iii. Tiempo de actualización de lectura del sensor.

d. Sensor de luminosidad
    i. Sensor activado.
    ii. Sensor desactivado.

6.3. Prueba de tiempo prolongado

En un principio el proyecto no respondía a los estímulos del usuario después de un tiempo prolongado, se investigó el problema y se concluyó que era debido a un desborde de memoria.

Se realizó una prueba de tiempo sobre el proyecto final, que consistía en dejar encendido el proyecto durante un día y luego verificar que:

- El estado no se vea modificado.
- El proyecto responda a nuevas acciones.

Se pudo comprobar el funcionamiento por dos días seguidos sin fallas.

7. Conclusiones

En conclusión, el proyecto ha cumplido con los requerimientos y objetivos establecidos, excepto en cuanto a la fecha límite, que se retrasó hasta febrero de 2023. Durante el desarrollo del proyecto, se presentaron algunos problemas y dificultades, como errores en el esquemático del circuito impreso, pero estos fueron resueltos con correcciones y ajustes. La prueba de tiempo prolongado demostró un buen funcionamiento sin fallas durante dos días consecutivos. En general, el proyecto ha cumplido con los objetivos principales y podría ser ampliado en el futuro.

7.1. Grado de cumplimiento de Requerimientos y objetivos

A lo largo del desarrollo del proyecto se ha cumplido con todos los requerimientos detallados en el punto 3 de este informe con la excepción de, posiblemente, la fecha límite de diciembre de 2022. Si bien el hardware y el software del proyecto fueron desarrollados en su totalidad para la fecha indicada por la cátedra, el informe final no se completó hasta febrero de 2023.
En cuanto a los objetivos principales detallados en el punto 2, se ha cumplido con todos ellos. Los restantes objetivos secundarios podrían ser tenidos en cuenta en una posible extensión futura del proyecto.

7.2. Problemas y dificultades en el desarrollo

7.2.1. Diseño Esquemático y Fabricación del Circuito Impreso

Se han presentado problemas y/o dificultades durante las etapas de diseño y construcción del prototipo. En particular en la etapa de confección del esquemático del poncho, se cometieron errores que produjeron problemas en la etapa de fabricación del circuito impreso. El primero fue el uso de un componente incorrecto. El esquemático hacía referencia a un transistor 2N3904 (Q1) cuando en realidad se iba a usar un BC548. Afortunadamente, la única diferencia entre ambos era que las conexiones de emisor y colector estaban invertidas. Una vez corregido este error en el esquemático no tuvo necesidad de cambiar el circuito impreso, pues solo basta con rotar el componente antes de soldar.

El segundo error cometido se relaciona con el circuito integrado usado para el control de las luces LED. No se tuvo en cuenta que la configuración interna de los pares Darlington era “open collector”. En la fabricación del PCB tuvo que hacerse una corrección en unas pistas para que el circuito LED funcionase, que puede verse en la siguiente figura:

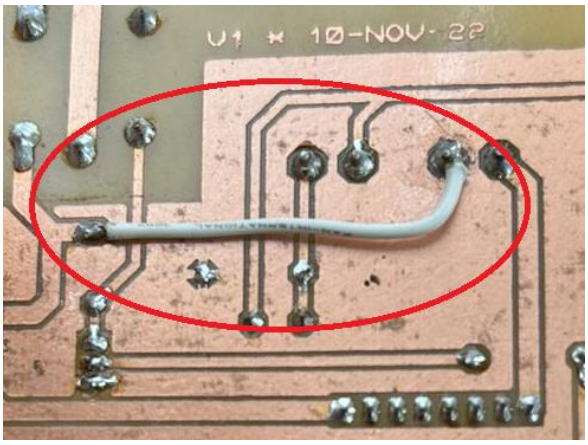


Figura 21. Rectificación del circuito de LED

Un tercer problema surgió en la etapa de perforación de la placa. Un pad no estaba perfectamente alineado con los demás. Esto causó que, al insertar el módulo WiFi el pad y la pista a la que se conectaba se aflojaran y levantaran del sustrato del PCB. Como aún había conexión a ese pad se optó por una solución rápida que consiste en fijar el componente con pegamento para reducir el estrés mecánico a la conexión. Una solución permanente hubiera sido rehacer el PCB.

Todas estas correcciones se han agregado al diseño del PCB y se actualizaron las figuras y anexos correspondientes. Para el problema de esta sección se agregaron pads adicionales para soldar conectores adicionales que cumplirán con la función de refuerzo mecánico de las conexiones del módulo WiFi.

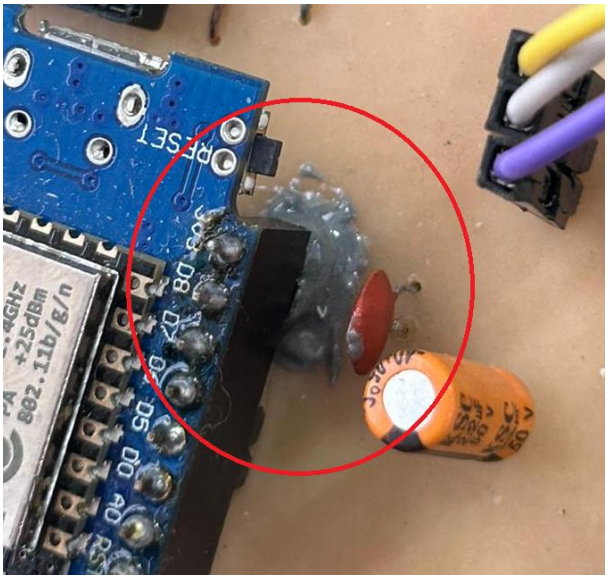


Figura 22. Reparación de conexión dañada.

7.2.2. Comunicación WiFi del proyecto contra aplicación Web

En principio, cuando hubo que resolver la comunicación con la aplicación Web, se planteó la comunicación con un ESP8266 básico pero como el desarrollo de la comunicación se complejizó demasiado como para ser utilizada en un prototipo, se optó por utilizar el módulo Wemos mini. Éste ha permitido el uso librerías de comunicación previamente desarrolladas, lo cual disminuyó el tiempo de desarrollo considerablemente.

7.2.3. Comunicación UART (módulo WiFi contra EDU-CIAA)

Al comienzo de programar la comunicación entre las dos placas nos encontramos con el problema que el paquete de información almacenada en la base de datos llegaba correctamente al módulo WiFi pero fallaba intermitentemente en la comunicación entre el módulo y la EDU-CIAA, la primer idea de solución fue desarrollar un mecanismo de suma de comprobación entre las dos placas, de ese modo podríamos descartar las tramas que tenían errores. Sin embargo la solución fue más simple, el problema se resolvió modificando la velocidad de comunicación a 9600 baudios, no se encontraron problemas de comunicación en las pruebas de tiempo prolongado.

7.2.4. Comunicación transversal

A la hora de establecer la comunicación transversal entre todas las partes del proyecto, se decidió usar el formato JSON ya que es un formato conocido por los integrantes del grupo y fácil de entender. El problema principal era resolver la lectura del JSON para el firmware, lo cual se resolvió utilizando una librería de interpretación de JSON [16].

7.3. Actividades realizadas

Las actividades realizadas por ambos integrantes del grupo reflejan lo indicado en la tabla de división de tareas, en mayor o menor medida. Pueden existir discrepancias en el trabajo realizado entre diciembre de 2022 y febrero de 2023, pues solo uno de los miembros contó con la EDU-CIAA y el poncho para realizar trabajos adicionales.

Las pruebas detalladas en el punto 6 fueron llevadas a cabo por Hernán por ser él quien contó con las herramientas durante el receso. Se dispuso que así fuera porque las pruebas sobre el hardware ya habían sido realizadas durante las clases en el laboratorio.

Sin tener en cuenta el trabajo realizado durante el verano, ambos integrantes invirtieron una cantidad similar de horas de trabajo en el proyecto, indicadas en el cronograma.

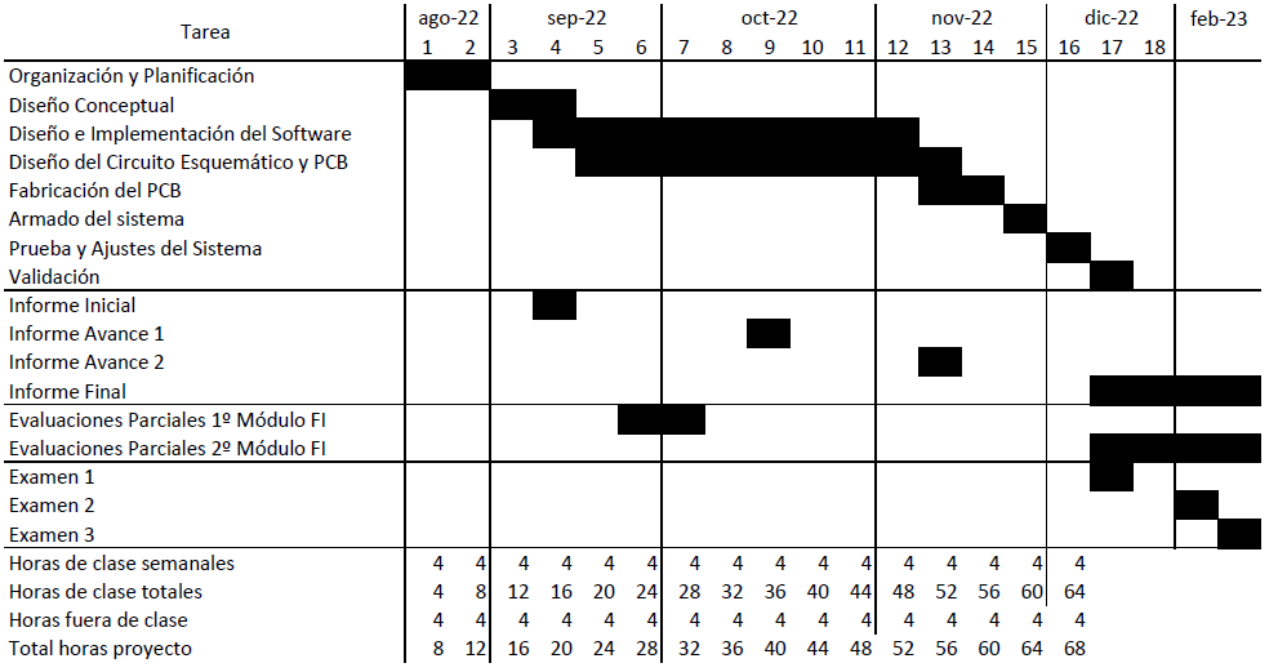
7.4. Análisis del presupuesto

En la sección 11.1 se muestra el listado de materiales utilizados para la construcción del prototipo, donde también se incluye el precio de cada uno de los componentes del proyecto. No se tuvo en cuenta a la placa de desarrollo ni la fuente de alimentación entre los gastos del proyecto por haber sido piezas proporcionadas por la cátedra, sin embargo, se los incluye como referencia.

Durante el desarrollo del proyecto no se discutió el presupuesto explícitamente, sino que se fue decidiendo qué componentes usar en base a su costo unitario. Además, algunos de los componentes usados en la construcción del poncho ya se tenían a disposición.

8. Cronograma

A continuación se muestra un cronograma tentativo para la realización del proyecto. En él se incluyen las etapas de realización del proyecto, entregas y evaluaciones de la materia, y las semanas de evaluación de la Facultad de Ingeniería para el segundo módulo. En la parte inferior se contabilizan las horas de clase semanales y el total de horas de clase semanal invertido en el proyecto.



Tarea	ago-22		sep-22				oct-22					nov-22				dic-22			feb-23
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
Organización y Planificación																			
Diseño Conceptual																			
Diseño e Implementación del Software																			
Diseño del Circuito Esquemático y PCB																			
Fabricación del PCB																			
Armado del sistema																			
Prueba y Ajustes del Sistema																			
Validación																			
Informe Inicial																			
Informe Avance 1																			
Informe Avance 2																			
Informe Final																			
Evaluaciones Parciales 1º Módulo FI																			
Evaluaciones Parciales 2º Módulo FI																			
Examen 1																			
Examen 2																			
Examen 3																			
Horas de clase semanales	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4			
Horas de clase totales	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64			
Horas fuera de clase	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4			
Total horas proyecto	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64	68			

9. División de Tareas

Tarea	Encargado
Armado del poncho	Constantino
Diseño del circuito esquemático	Constantino
Diseño del poncho	Constantino
Informe de Avance 1	Hernán
Informe de Avance 2	Constantino
Informe Final	Hernán/Constantino
Programación Control LED Analógico	Constantino
Programación Controles Botonera	Constantino
Programación Sensor de Luminosidad	Constantino
Programación Sensor de Movimiento	Constantino
Programación Comunicación JSON	Hernán
Programación Módulo WiFi	Hernán
Frontend Web	Hernán
Backend Web	Hernán
Integración Base de Datos	Hernán
Pruebas de Funcionamiento	Hernán/Constantino

10. Bibliografía

[1] Ficha técnica y manual del Macroled Smart SSX2-WIFI. Disponibles al 15/10/2022 en: https://macroled.com.ar/producto/smart/interruptores/interruptor-smart.

[2] OKAWA Electric Design – Low-Pass Filter Design Tool. Disponible al 15/10/2022 en: http://sim.okawa-denshi.jp/en/CRtool.php.

[3] Maxfield, Max. *Ultimate Guide to Switch Debounce (Part 3)*. Disponible al 15/10/2022 en: https://www.eejournal.com/article/ultimate-guide-to-switch-debounce-part-3/.

[4] Hoja de datos de sensor HC-S501.

[5] Hoja de datos de integrado ULN2003AN (Tiger). Disponible al 16/10/2022 en: https://pdf1.alldatasheet.com/datasheet-pdf/view/1136523/TGS/ULN2003AN.html.

[6] Hoja de datos del fusible MF-MSMF030 (Bourns). Disponible al 16/10/2022 en: https://pdf1.alldatasheet.com/datasheet-pdf/view/156717/BOURNS/MF-MSMF030.html.

[7] Hoja de datos del relé SRD05VDCSLC (Songle). Disponible al 16/10/2022 en: https://pdf1.alldatasheet.com/datasheet-pdf/view/1131944/SONGLERELAY/SRD05VDCSLC.html.

[8] Hoja de datos del microcontrolador LPC435xx/3x/2x/1x (rev. 5.3, 15/03/2016).

[9] Hoja de datos del módulo de relé de 4 canales para Arduino (Handson Technology).

[10] Hoja de datos del fotorresistor (RS Components).

[11] Software de simulación de circuitos Livewire, versión 1.11 (28/10/2004).

[12] Kessler, L. *Todo sobre Tiras LED*. Disponible al 17/11/2022 en: https://afinidadelectrica.com/2020/05/04/todo-sobre-tiras-de-leds/.

[13] Norma IPC-2221A (Mayo 2003). Sección 6.2: *Requerimientos de Material Conductivo*.

[14] Sobrevila, Marcelo. *Instalaciones Eléctricas*. 2º edición. Marymar. 1987.

[15] Brengi, Diego. *Dimensiones de la EDU-CIAA.*

[16] Librería de interpretación JSON. Disponible al 21/02/2023 en: https://github.com/DaveGamble/cJSON.

[17] Sitio de compra en línea de Electrocomponentes S.A., fabricantes de la placa de desarrollo EDU-CIAA-NXP. Disponible al 21/02/2023 en: https://www.electrocomponentes.com/tienda/desarrollo/computadora-industrial-abierta-argentina-ciaa-nxp-edu.

11. Anexos

11.1. Lista de Materiales (BOM)

El siguiente listado de materiales fue generado automáticamente por Proteus y no incluye los elementos que se modelaron mediante conexiones externas, ni los ajenos al poncho, como la placa de desarrollo, las luces LED o la fuente de alimentación. Éstos se enumeran a continuación y se incluye su costo unitario:

- Módulo Wemos D1 mini ($1101.00)
- Módulo Sensor HC-S501 ($764.47)
- Tira de LED ($566.19)
- Placa de desarrollo EDU-CIAA-NXP ($22679.00 [17], suministrada por la cátedra)
- Fuente de alimentación 12V (brindada por la cátedra)
- Luz 220V de prueba ($160.00)
- Placa para circuito impreso simple faz de 10x15cm ($860.00)

Los campos del listado de materiales son:

- Cantidad: número de componentes de un cierto tipo usados en el poncho
- Referencia: nombre de cada parte individual
- Valor
- Costo unitario
- Encapsulado

Los componentes están agrupados en categorías:

- Capacitores
- Resistores
- Circuitos Integrados
- Transistores
- Diodos
- Genéricos

Bill Of Materials for EDU-CIAA Luminaria

Design Title	EDU-CIAA Luminaria
Author	Constantino Palacio
Document Number	1
Revision	3
Design Created	martes, 21 de febrero de 2023
Design Last Modified	martes, 21 de febrero de 2023
Total Parts In Design	34

7 Capacitors

Quantity	References	Value	Unit Cost	PCB Package
6	C1-C5,C7	100nF	$10,00	CAP10
1	C6	2.2uF	$10,00	ELEC-RAD10
Sub-totals:			$70,00	

7 Resistors

Quantity	References	Value	Unit Cost	PCB Package
1	R1	2.2k	$5,00	RES40
6	R2,R6,R8,R10,R12,R14	10k	$5,00	RES40
Sub-totals:			$35,00	

1 Integrated Circuits

Quantity	References	Value	Unit Cost	PCB Package
1	U1	ULN2003A		DIL16
Sub-totals:			$0,00	

1 Transistors

Quantity	References	Value	Unit Cost	PCB Package
1	Q1	BC548	$25,00	TO92
Sub-totals:			$25,00	

1 Diodes

Quantity	References	Value	Unit Cost	PCB Package
1	D1	1N4007		DIODE30
Sub-totals:			$0,00	

17 Miscellaneous

Quantity	References	Value	Unit Cost	PCB Package
1	J1	CONN_CUSTOM4		CONN_CUSTOM4
1	J2	CONN_CUSTOM5		CONN_CUSTOM5

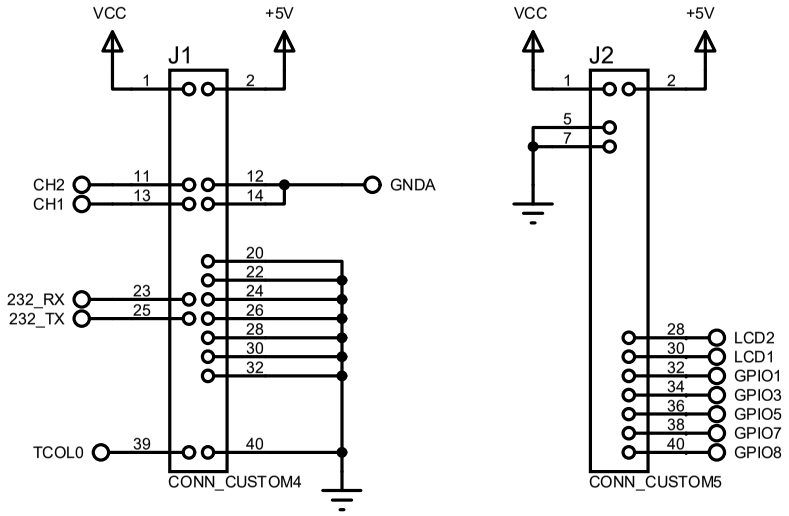
2	J3,J9	SIL-100-03		SIL-100-03
4	J4-J6,J8	TBLOCK-I2	$100,00	TBLOCK-I2
2	J7,J12	SIL-100-06		SIL-100-06
5	J11,J13-J16	BUTTON2	$85,00	BOTON3
1	RL1	G5CLE-14-DC12		RLY-OMRON-C4
1	RV1	10k	$170,00	PRE-THUMB
Sub-totals:			$995,00	

Totals: $1.125,00

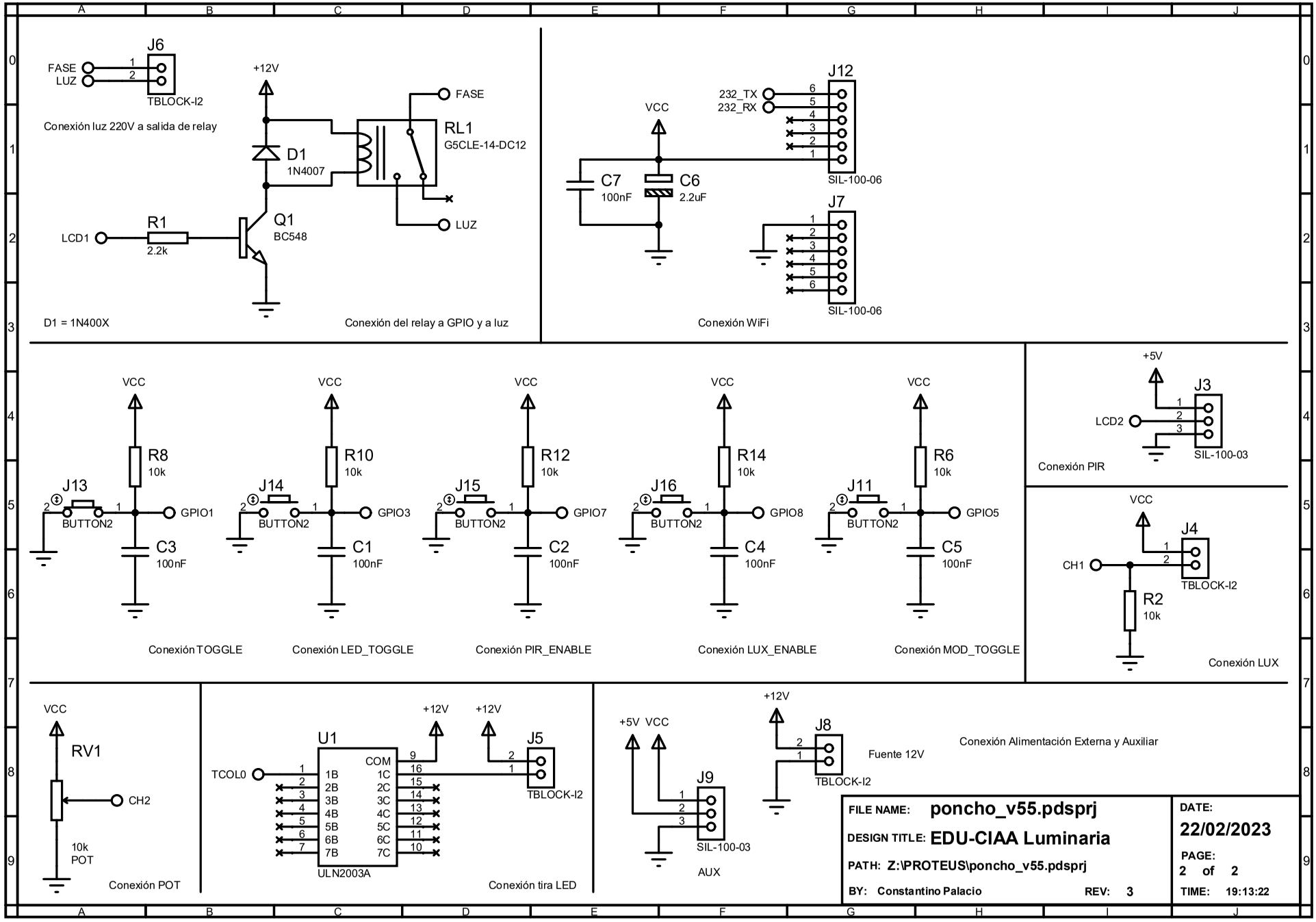
miércoles, 22 de febrero de 2023 9:07:01

11.2. Circuito Esquemático Completo

A continuación se reproduce el esquemático del poncho. El mismo se divide en secciones según la función que cumpla en el circuito.



FILE NAME: poncho_v55.pdsprj	DATE: 22/02/2023
DESIGN TITLE: EDU-CIAA Luminaria	PAGE: 1 of 2
PATH: Z:\PROTEUS\poncho_v55.pdsprj	
BY: Constantino Palacio REV: 3	TIME: 19:13:22

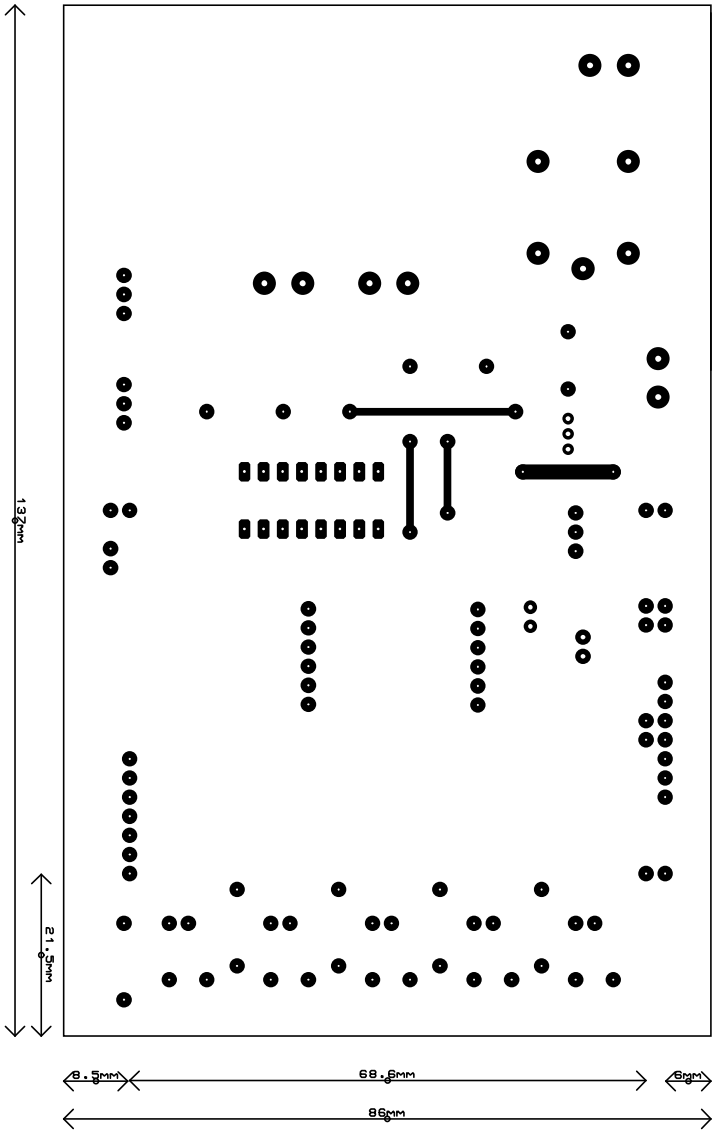

J6
FASE
LUZ
TBLOCK-I2
Conexión luz 220V a salida de relay
+12V
FASE
RL1
G5CLE-14-DC12
D1
1N4007
LUZ
R1
2.2k
LCD1
Q1
BC548
D1 = 1N400X
Conexión del relay a GPIO y a luz
VCC
J12
232_TX
232_RX
C7
100nF
C6
2.2uF
SIL-100-06
J7
SIL-100-06
Conexión WiFi
+5V
J3
LCD2
SIL-100-03
Conexión PIR
VCC
R8
10k
J13
BUTTON2
GPIO1
C3
100nF
Conexión TOGGLE
R10
10k
J14
BUTTON2
GPIO3
C1
100nF
Conexión LED_TOGGLE
R12
10k
J15
BUTTON2
GPIO7
C2
100nF
Conexión PIR_ENABLE
R14
10k
J16
BUTTON2
GPIO8
C4
100nF
Conexión LUX_ENABLE
R6
10k
J11
BUTTON2
GPIO5
C5
100nF
Conexión MOD_TOGGLE
J4
CH1
TBLOCK-I2
R2
10k
Conexión LUX
RV1
CH2
10k
POT
Conexión POT
U1
TCOL0
COM
1B
2B
3B
4B
5B
6B
7B
1C
2C
3C
4C
5C
6C
7C
ULN2003A
+12V
J5
TBLOCK-I2
Conexión tira LED
+5V
J9
SIL-100-03
AUX
J8
Fuente 12V
TBLOCK-I2
Conexión Alimentación Externa y Auxiliar
FILE NAME: poncho_v55.pdsprj
DESIGN TITLE: EDU-CIAA Luminaria
PATH: Z:\PROTEUS\poncho_v55.pdsprj
BY: Constantino Palacio
REV: 3
DATE: 22/02/2023
PAGE: 2 of 2
TIME: 19:13:22

11.3. Diseño de PCB

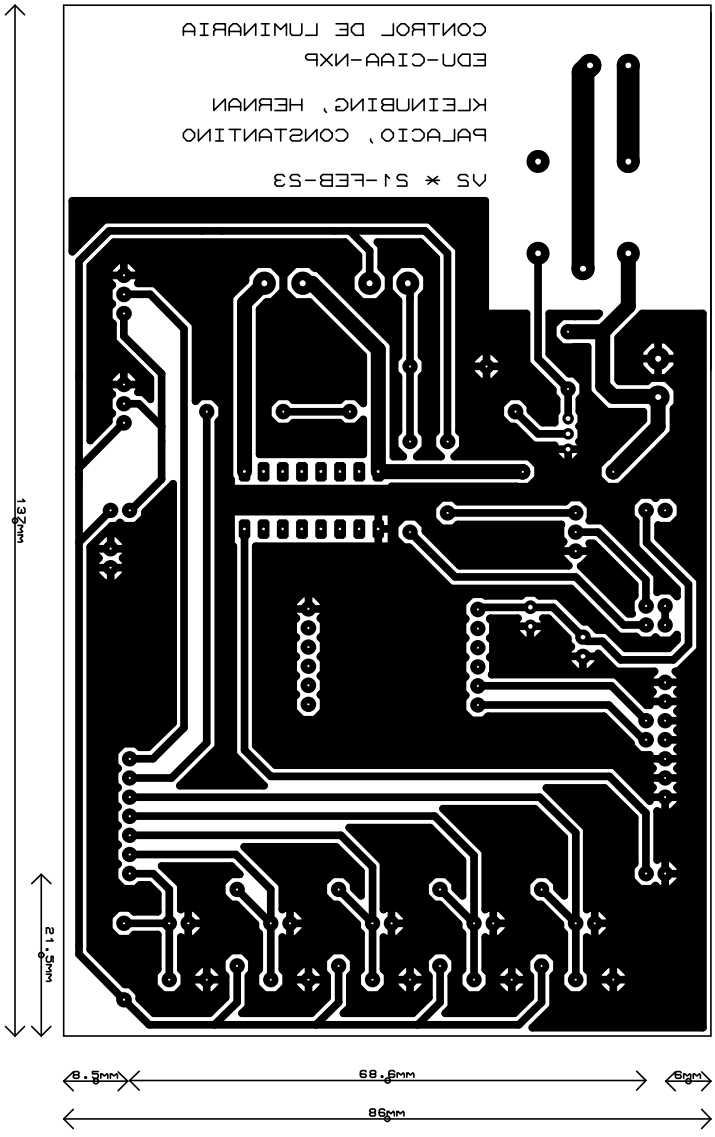
En las siguientes páginas se reproduce el diseño final del PCB en escalado 1:1. La capa de soldadura puede usarse directamente para fabricar el poncho.

Se incluyen medidas en cada capa para la ubicación de los conectores de la EDU-CIAA-NXP, distancia considerada clave a la hora de diseñar la placa. Además, se incluyen dimensiones generales del poncho.

La medida desde el borde hasta cada uno de los conectores no es simétrica, pues estos componentes no lo son. Esta medida es desde el borde del PCB hasta el terminal más cercano.

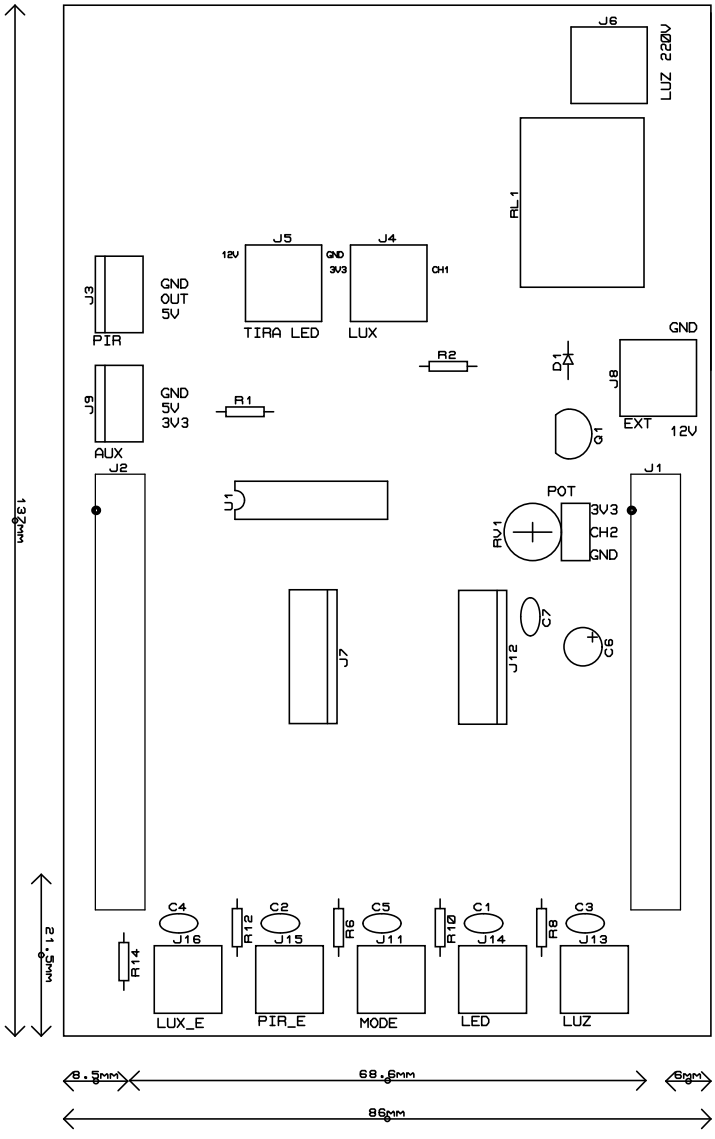


Capa de Componentes



Capa de Soldaduras

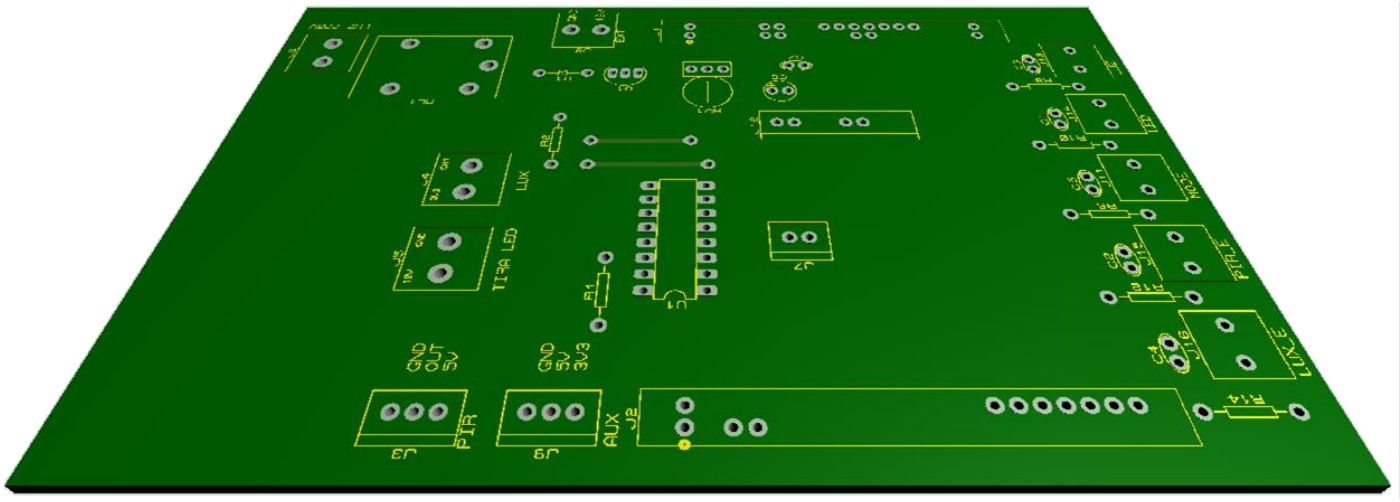
Esta capa no debe imprimirse en espejo. Imprimiéndola como está resulta en el poncho correcto.

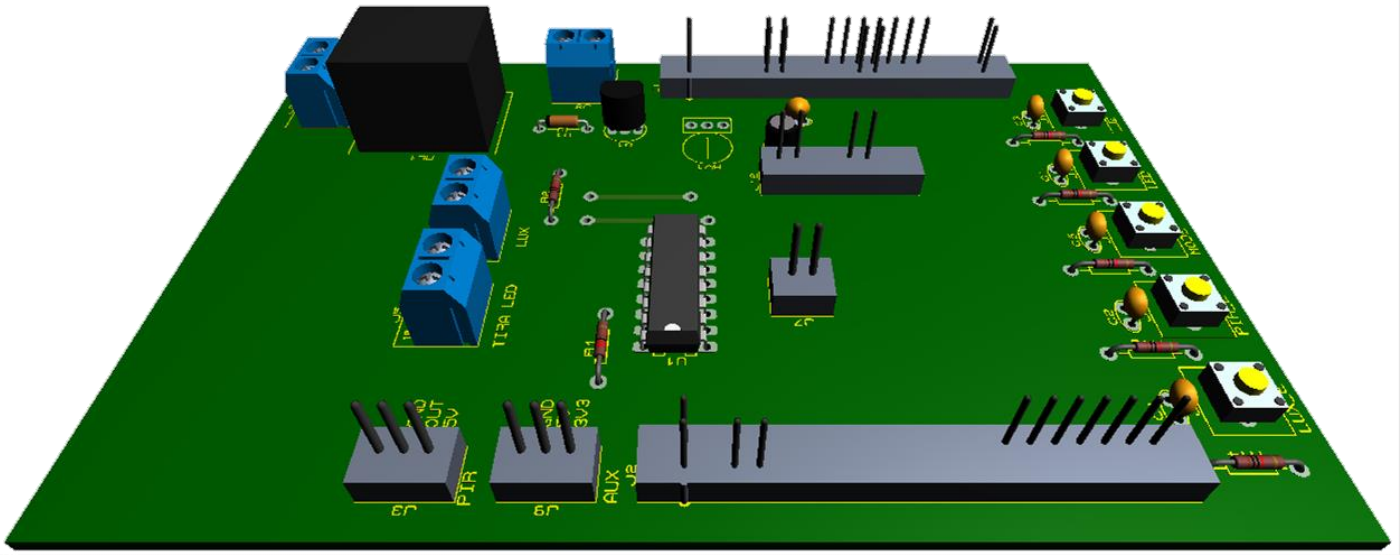

J6
LUZ 220V
RL1
J5
12V
GND
3V3
J4
CH1
J3
GND
OUT
5V
PIR
TIRA LED
LUX
R2
D1
GND
J8
EXT
12V
J9
GND
5V
3V3
AUX
R1
Q1
J2
J1
U1
POT
AV1
3V3
CH2
GND
J7
J12
C7
C6
C4
C2
C5
C1
C3
J16
J15
J11
J14
J13
R12
R6
R10
R8
R14
LUX_E
PIR_E
MODE
LED
LUZ
132mm
21.5mm
8.5mm
68.6mm
6mm
86mm

Serigrafía

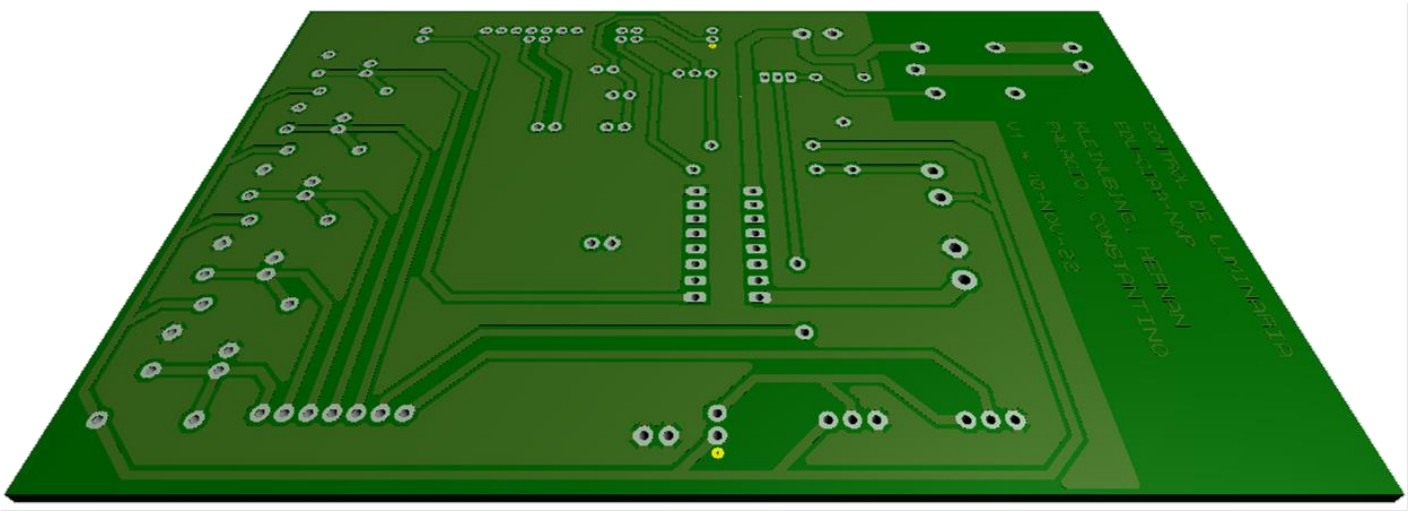
11.4. Modelado en Tres Dimensiones

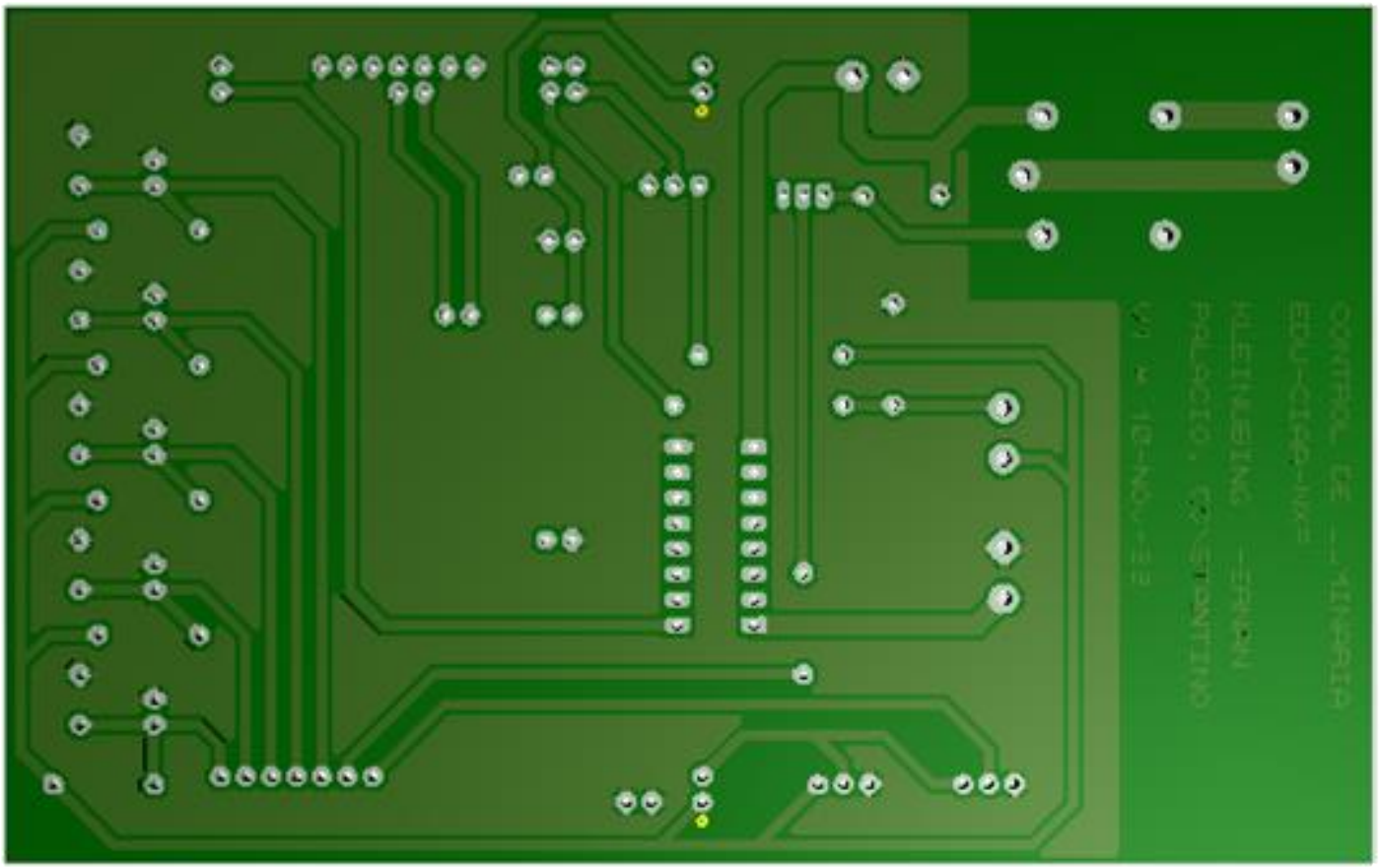
Vista del lado de los componentes sin poblar y poblada. Notar que los conectores de la EDU-CIAA se conectarán del lado de las pistas y no de los componentes. Tampoco existía un modelo en tres dimensiones de la conexión del potenciómetro, pero se conectará un *header* de 3 terminales.





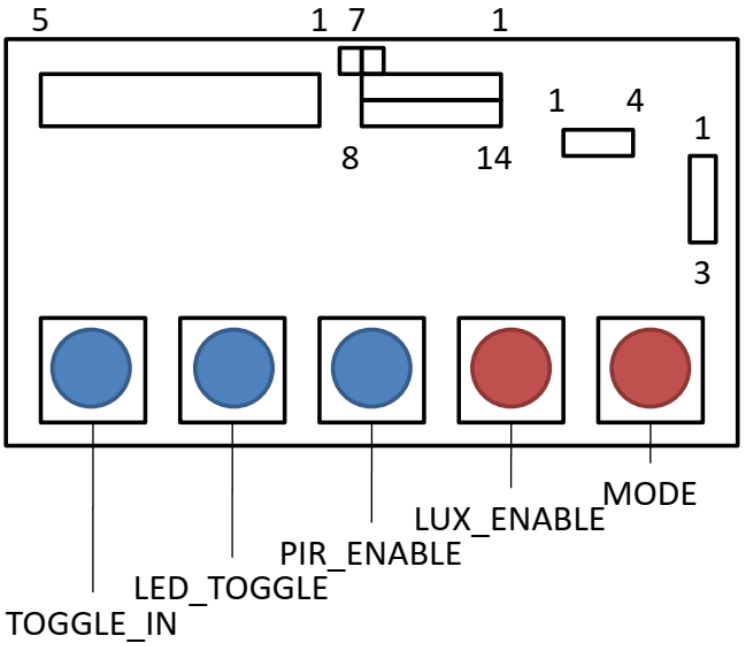
Vista del lado de las soldaduras (pistas) y vista superior de la misma cara.





11.5. Esquema de Conexionado de Placa de Pruebas

El siguiente diagrama muestra las conexiones necesarias para realizar la interfaz entre la placa de pruebas presentada en la sección 5, la placa de desarrollo EDU-CIAA-NXP, el sensor PIR y el módulo de relé.



Relé	
1	+5V
2	IN
3	+3,3V
4	GND

PIR	
1	GND
2	OUT
3	+5V

LEDs	
1	ON
2	MODE
3	LUX_STAT
4	PIR_OUT
5	PIR_STAT

EDU-CIAA-NXP		
1	GPIO5	MODE_TOGGLE
2	NC	MODE_STAT
3	GPIO7	PIR_ENABLE
4	NC	PIR_STAT
5	NC	LUX_STAT
6	GPIO8	LUX_ENABLE
7	TCOL0	LED_OUT
8	LCD1	RELAY_OUT
9	GPIO1	TOGGLE
10	LCD2	PIR_IN
11	GPIO3	LED_TOGGLE
12	+3,3V	
13	GND	
14	+5V	

Versión 2. 23-NOV-22.

De izquierda a derecha y de arriba abajo, las conexiones son: LEDs de estado (5 salidas), conector auxiliar (dos terminales macho), conector principal EDU-CIAA-NXP (dos filas de 7 terminales hembra), salida relé (4 terminales macho) y sensor PIR (3 terminales hembra colocados verticalmente). El significado de cada terminal de cada conexión (*pin-out*) se indica mediante las tablas junto a la figura. El sentido de lectura de los terminales (su número) se indica sobre la representación gráfica de la placa. El único conector que no se señalizó fue el auxiliar porque no está conectado a nada, sino que se incluyó para una posible ampliación.

Los terminales del conector EDU-CIAA-NXP indicados con "NC" no son usados por el firmware actual, pero están físicamente conectados a las salidas LED de la placa.

11.6. Enlaces a Recursos Externos

1. **Video con pruebas realizadas**
   https://drive.google.com/drive/folders/1prSXtEPx6hGWiNM0BgKefvPdsSo8t0Jh?usp=sharing

2. **Repositorio firmware del proyecto**
   https://github.com/Constantino-Palacio/proyecto_taller1

3. **Repositorio Aplicación web y módulo WiFi**
   https://github.com/kleinher/TallerDeProyectoI