Diseño e implementación de un sistema de control remoto y recopilación de datos para el robot secador de café.

Jeffrey Steven Muñoz Muñoz

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño e implementación de un sistema de control remoto y recopilación de datos para el robot secador de café.

Trabajo de graduación presentado por Jeffrey Steven Muñoz Muñoz
para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2022

Vo.Bo.:

(f) _____

M.Sc. Pablo Mazariegos

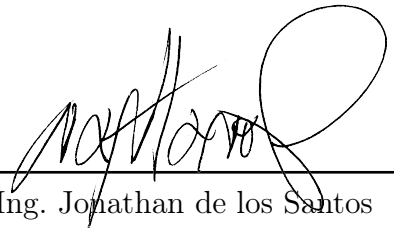
Tribunal Examinador:

(f) _____

M.Sc. Pablo Mazariegos

(f) _____

Ing. Kurt Kellner



(f) _____

Ing. Jonathan de los Santos

Fecha de aprobación: Guatemala, 6 de enero de 2022.

Prefacio

Este trabajo de graduación está dedicado a mi madre Dinora y mi hermano Kevin, sin el apoyo y consejos de ellos no habría podido lograrlo. También quiero agradecer a mis compañeros Arnulfo, Jorge, Julio e Ignacio, que a lo largo de la carrera siempre fueron fuente de apoyo e inspiración para poder seguir adelante. Por último quiero agradecer a mis profesores Kurt y Silvio quienes no solo me enseñaron cosas importantes en el aspecto académico sino que también en el aspecto profesional ayudándome a siempre buscar excelencia y mejora continua en mi vida diaria.

Índice

Lista de figuras

Lista de cuadros

Resumen

En este trabajo de graduación se realizaron e implementaron dos nuevos sistemas para el proyecto robot secador de café. Uno de estos nuevos sistemas fue el de control remoto para permitir que el robot pudiera ser manejable por un operario y así poder transportarlo desde el área de almacenamiento hasta el área de trabajo sin ninguna complicación. El segundo de los sistemas se planteó para que fuera un sistema de recolección de datos en tiempo real para poder desplegarlos en una aplicación web y móvil, pero que también permitiera realizar el almacenamiento de estos en una base de datos para poder consultarlos en un futuro de ser deseado.

A través del prototipo a escala del control remoto se implementaron 8 comandos de movimiento: movimiento de avance, movimiento de retroceso, giro a la izquierda, giro a la derecha, cambio de velocidad y cambio de modo automático y manual. Para el prototipo de pruebas de campo se implementaron 4 comandos: avance individual de cada motor, retroceso individual de cada motor, inclinación hacia arriba de las aspas e inclinación hacia abajo de las aspas. También se logró un alcance ininterrumpido de 15 metros al momento de realizar la prueba de distancia dentro de un área totalmente despejada. Para la construcción del control se logró un diseño que permitió almacenar todos los componentes utilizados (PCB, fuente de poder y módulo de radiofrecuencia) sin que afectara directamente la ergonomía del control o que causara algún daño al operario al momento de utilizarlo.

En el sistema de recopilación de datos se logró implementar de manera totalmente funcional el protocolo MQTT utilizando un módulo ESP32 para la publicación de los datos recolectados. Se utilizó una RaspBerry Pi 3 b+ como un servidor remoto de manera que esta contenía los servicios necesarios como lo fueron el broker Mosquitto, Node-Red con sus respectivas librerías para la creación de interfaces gráficas y la suscripción a los tópicos deseados. MariaDB que permitió tener una base de datos en la cual poder almacenar todos los datos recolectados y PHPMyAdmin para poder administrar mejor dicha base de datos. También se logró desarrollar de manera exitosa la interfaz gráfica para el despliegue de los datos en tiempo real y que esta fuera consultable desde un dispositivo móvil. En cuanto al almacenamiento de datos se logró una conexión exitosa de manera que al utilizar Node-Red se pudieran almacenar los datos de obtenidos en una base de datos dedicada únicamente a este fin y fuese posible desplegar todos los datos almacenados dentro de tablas ordenadas.

Abstract

The study developed throughout this document presents the creation and implementation of two new systems for the undergoing project "robot secador de café", the first of the new systems was about creating a remote controller that could control the robot to be able to transport it between the storage area and the working area without creating any kind of accidents that could affect the user or directly the robot. The second system was devised so it could collect data in real time and show it in a user interface that was accessible from the web or any mobile devise and that it could store all that collected data inside a dedicated data base, so it was possible to consult it in the future.

The first new system didn't present any complications with the radio frequency communication that was used, and it was possible to transmit the eight important commands that were established, also it was possible to accomplish an uninterrupted range of 15 meter in a clear open area when the distance test was realized. When constructing the controller, a design that could store every important component for the functionality was accomplished and this also includes a design that didn't affect the ergonomics or that it could hurt the user directly.

For the system dedicated to collecting data it possible to implement the MQTT protocol in a functional way with the help of an ESP32 module that was in charge of the publication in the specific topics of all the collected data and a RaspBerry Pi 3 b+ was used as remote server that contained all the necessary services like the broker Mosquitto, Node-Red with all of the libraries that were necessary for the creation of the user interfaces and the subscription of the desired topics in order to receive the data, MariaDB enabled the use of data bases so it was possible to storage all the received data and PHPMyAdmin was very useful so it was possible to administrate better all the data bases that were created. It was also possible to develop the user interface that was able to display the behavior of the collected data in real time and it was accessible either from the web or a mobile device. Regarding data storage, a successful connection was achieved so that by using Node-Red, the data obtained could be stored in a database dedicated solely to this purpose and it would be possible to display all the data stored within the created tables.

CAPÍTULO 1

Introducción

En Guatemala se ha acostumbrado a que una de las maneras más comunes y útiles para poder secar los granos de café dentro de las fincas es el esparcir los diferentes lotes de grano en el suelo y utilizar un rastrillo para poder mover los lotes de café . Con base en esto se esta desarrollando un robot el pueda realizar esta tarea de forma automática y así poder crear una manera mucho más eficiente de poder realizar este proceso reduciendo el error humano y asegurando una buena calidad del café.

En la presente investigación se desarrollaron dos partes importantes para el correcto funcionamiento de este robot, la primera parte fue un control remoto que permitiera mover el robot a voluntad y que pueda enviar los comandos necesarios para que el robot ejecute diferentes acciones. La segunda parte que se presentará es la implementación de un sistema de recopilación de datos para el robot el cual estará hecho para que tenga una aplicación móvil y una aplicación web las cuales podrán mostrar datos en tiempo real sobre el robot.

El control remoto fue diseñado y fabricado utilizando diferentes estándares para el diseño de controles remotos y así poder asegurar que este cumpliera con los requerimientos de ergonomía y seguridad del usuario que se suelen presentar en las diferentes normativas. El principal medio de comunicación entre el robot y el control remoto es radiofrecuencia y para lograr esto se implemento el módulo transceptor nRF24L01+PA. El cual permite tener una comunicación a una velocidad considerablemente rápida y además tiene un alcance de aproximadamente un kilómetro de distancia (esto si no hay obstáculos en entre los transceptores) por lo que se consideró más que factible utilizarlo en esta aplicación ya que las distancias a las que se encontrara el control y el robot no serán muy grandes.

Para las aplicaciones móviles y web se implemento el protocolo MQTT para la transmisión de datos ya que es un protocolo que provee una comunicación muy segura y estable. Para poder realizar el manejo de los datos utilizando este protocolo se utilizo el servicio Node-red que permite tener un buen manejo de los datos transmitidos ya que este servicio se especializa en simplificar el manejo de datos utilizando maneras mas gráficas de crear relaciones y funciones. Mientras tanto para poder crear la interfaz gráfica de las aplicaciones se utilizo la librería Node-red Dashboard que permite agregar los componentes necesarios de

manera que únicamente será necesario crear las conexiones requeridas con el resto de componentes para poder obtener los resultados deseados al contrario de tener que desarrollar cada uno de los componentes de manera individual.

CAPÍTULO 2

Antecedentes

2.1. Dualshock 4 (control para consola PlayStation 4)

Ya que uno de los componentes principales de este proyecto es el diseño e implementación de un control remoto este debe cumplir con varios requisitos para que pueda ser considerado ergonómicamente correcto y seguro para el uso. Para esto se tomó como una referencia confiable el diseño del control Dualshock 4 que es el control principal para la consola PlayStation 4, esta decisión fue tomada utilizando como referencia el artículo científico "The Ergonomic Development of Video Game Controllers" [1] en el cual se toman 4 de los controles mas populares en la historia y cada uno es sometido a varias pruebas para poder determinar si estos cumplen con los requisitos de antropometría estándar que existen en la industria. Al final del estudio se determinó que el Dualshock 4 es el control que posee la mayor ergonomía por lo que este fue seleccionado como referencia para este proyecto.

Figura 1: Control Dualshock 4 de la consola PlayStation 4

2.2. La aplicación Car-Sharing que se basa en HiveMQ para una conectividad confiable

BMW Mobility Services es un grupo empresarial dentro de BMW Group que esta desarrollando nuevas soluciones para la movilidad de la población urbana. Ya que se estima que para el 2030 aproximadamente el 60 por ciento de la población mundial viva en ciudades BMW Mobility Services se esta centrando en poder crear servicios de movilidad que se integren de manera fácil al estilo de vida de la población urbana [2].

Uno de estos servicios es un producto para el uso compartido de automóviles para operaciones de flotas. La idea principal de este servicio es poder permitir que los operadores de flotas puedan administrar de manera remota una flota de vehículos, emitir señales directas a vehículos individuales y poder recopilar datos de estos mismos vehículos. En 2014 BMW Mobility Services comenzó a utilizar HiveMQ como plataforma de mensajería para su servicio de auto compartido, hoy en día la plataforma de HiveMQ admite a más de 80,000 clientes del software para poder generar mas de 90,00 mensajes por minuto [2].

Un criterio clave para los servicios del uso de automóviles compartidos es poder tener una disponibilidad generalizada ya que los clientes tienen la expectativa de poder acceder al servicio para poder utilizar un automóvil de manera oportuna y confiable. Lastimosamente esto no es posible en todas las ciudades en las que se ha implementado el servicio ya que existen diferentes factores que no permiten una buena accesibilidad tales como el clima y la cobertura [2].

La industria automotriz utilizaba un patrón muy tradicional para poder establecer la comunicación remota con un vehículo utilizando el concepto de un gatillo. Este concepto funcionaba de manera que cuando se quería iniciar la comunicación del vehículo con la nube este se realizaba por medio de un mensaje de texto que creaba una llamada de atención al sistema encargado del cliente para que este pudiera iniciar una sesión HTTP y así poder conectarse con la nube. Pero el utilizar este concepto tiene algunos desafíos ya que el mensaje de texto es un servicio muy impredecible y el servicio HTTP es demasiado lento además de tener costos relativamente altos para la cantidad de mensajes y el tamaño de estos [2].

Para poder cumplir con la visión de crear una aplicación exitosa para poder compartir automóviles BMW Mobility Services optó por buscar otra alternativa para sus problemas de mensajería. El diseño de MQTT, el enfoque que este tiene hacia suscripción/publicación y los altos niveles de calidad de servicio que se utilizan logran satisfacer los requisitos de confiabilidad y eficiencia para una aplicación de esta índole por lo que en 2012 HiveMQ comenzó a desarrollar junto a BMW la plataforma HiveMQ MQTT para permitir a las empresas poder conectar millones de dispositivos de forma fiable y escalable utilizando el protocolo MQTT [2].

En 2014 BMW Mobility Services decidió implementar de manera permanente el protocolo MQTT dentro de su servicio de automóviles compartidos y para esto la compañía buscó un socio industrial que pudieras apoyarlos en este proyecto y dicho socio sería HiveMQ. Actualmente el servicio se compone de cientos de micro servicios acoplados escritos en Java que se ejecutan en un clúster de Kubernetes que se encuentran alojados en los servicios de web de Amazon. Normalmente HiveMQ maneja una carga de aproximadamente 90,000 tran-

sacciones por minutos que provienen de mas de 80,000 clientes que se encuentran conectados de manera simultánea [2].

BMW Mobility Services ha logrado extender el uso de HiveMQ como una plataforma interna de intercambio de datos para sus sistemas de backend. Con la lógica de publicar/suscribir que utiliza el protocolo MQTT un servicio de backend puede suscribirse a los datos que publican los vehículos por medio del broker de HiveMQ para poder realizar un análisis posterior de los datos lo cual permite que el proceso para la integración de los datos sea más inmediato y escalable [2].

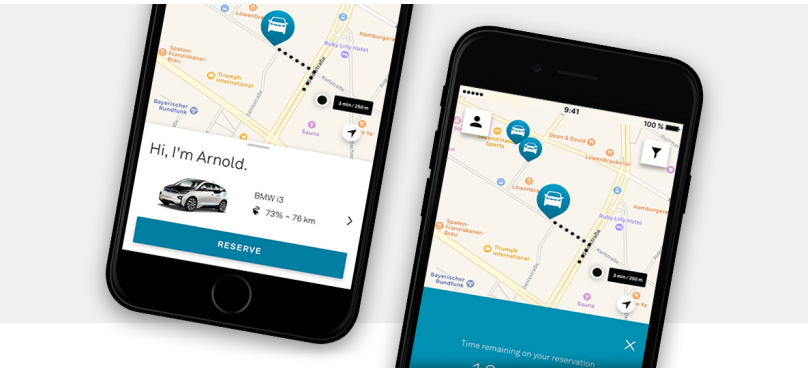


Figura 2: Aplicación Drive Now de BMW

CAPÍTULO 3

Justificación

La idea para este proyecto surgió de las observaciones que se realizaron al estado actual del robot secador de café, las cuales se centraron en las mejoras que este podría recibir para tratar que la primera versión de este pueda implementarse de manera efectiva en un ambiente laboral. Entre estas mejoras se decidió añadir dos nuevos "sistemas" al robot para que este pudiera brindar mas funciones y diferentes maneras de uso para el operario.

El primero de los nuevos sistemas que se decidió implementar fue el de control remoto ya que antes de añadir esta idea el plan para que los operarios pudieran transportar el robot desde su lugar de almacenamiento hasta el área de trabajo era mover este por medio de un trolley, pero con esta opción se correría el riesgo que se pudiera dañar el robot al momento de montarlo y desmontarlo en el trolley. Debido a esto se consideró que un control remoto permitirá a los operarios desplazar el robot de una manera segura tanto para ellos como para el robot y de esta manera evitar accidentes que puedan afectar las labores diarias de la finca.

El segundo de los nuevos sistemas es un sistema de recolección de datos el cual permitirá a los operarios conocer el estado en tiempo real del robot con datos como nivel de batería, tiempo de uso, ciclos de trabajo completados, entre otras cosas. Este sistema también permitirá conocer el historial del robot para poder saber cuántos días ha trabajado y la cantidad de horas que ha trabajado estos días así como los días restantes para que se le realice mantenimiento preventivo al robot y así evitar una falla durante el ciclo de trabajo.

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Implementar un sistema de control remoto que utilice un control físico y un sistema de recopilación de datos a través de Wi-Fi por medio de una aplicación web y móvil para el robot secador de café.

4.2. Objetivos específicos

- Implementar un sistema control remoto compacto y ergonómico que utilice radio frecuencia (RF) para poder manejar el robot.
- Crear una aplicación móvil y para computadora que permita recopilar, desplegar y exportar datos sobre el estado del robot tal como nivel de la batería, ciclos de carga, cantidad de procesos realizados, progreso de proceso actual, entre otros.
- Crear un sistema de comunicación vía Wi-Fi para la transmisión de los datos recopilados y poder consultar estadísticas del robot tales como fechas de uso, horas totales de uso, cronograma de mantenimiento preventivo, entre otros.

CAPÍTULO 5

Alcance

El alcance de este proyecto es poder tener dos nuevos sistemas para el robot secador de café los cuales son un sistema de control remoto que permita controlar el robot con distintos comandos de movimiento o acción y que utilice como su principal medio de comunicación la radiofrecuencia. El segundo sistema es un sistema de recopilación de datos que haga uso del protocolo MQTT para la transmisión de los datos permitiendo que estos puedan ser desplegados de manera gráfica e intuitiva en la web o en un dispositivo móvil y también teniendo la capacidad de almacenar los datos transmitidos dentro de una base de datos dedicada para poder consultarlos en un futuro si así se desea.

El control remoto estará diseñado tomando en cuenta normas especializadas para la ergonomía y seguridad de mandos lo cual permitirá asegurar que será cómodo de utilizar y también seguro para el operario, para su fabricación se utilizara el material PLA que será impreso en 3D lo cual permitirá tener una carcasa resistente y que también es fácil de fabricar lo cual permitirá crear una nueva pieza en caso sea necesario realizar una corrección o ajuste físico.

La recopilación de datos se realizará con un módulo ESP32 que se encontrara conectado por medio de WiFi a un servidor local instalado dentro de una RaspBerry Pi 3 b+ para la transmisión de los datos utilizando el protocolo MQTT, la recepción de los datos se trabajará utilizando el servicio Node-Red que se especializa en manejo de datos en sistemas que utilicen el protocolo MQTT. Este servicio permitirá que se puedan recibir los deseados únicamente y junto a la librería Node-Red Dashboard se podrá crear una interfaz gráfica que permita desplegar los datos de una manera interactiva y fácil de interpretar que podrá ser utilizada ya sea desde la web o en un dispositivo móvil si así se desea.

Dentro del sistema de recopilación de datos se utilizará el servicio de MariaDB para poder crear bases de datos dedicadas para los datos que se estén manejando y adicional a esto se utilizara PHPMyAdmin para poder manejar dichas bases de datos de una manera más fácil permitiendo un acceso más rápido al igual que una mejor visualización de las diferentes tablas en las que estarán los datos almacenados. Adicional a esto se utilizará una librería de comunicación de Node-Red para poder crear alertas por medio de correo electrónico en

dado caso ocurra un cambio drástico y no esperado en los datos recopilados.

CAPÍTULO 6

Marco teórico

6.1. Comunicación inalámbrica

Específicamente hablando, la comunicación inalámbrica es la transmisión de señales entre dos puntos en el espacio sin utilizar una conexión física entre ellos, este tipo de comunicación ha esta disponible para los humanos desde el inicio de los tiempos. La mayoría de las personas se comunican entre si utilizando su voz sin necesidad de utilizar equipamiento adicional utilizando las cuerdas vocales y el sistema auditivo humano [3].

En el sentido más general, un sistema de transmisión consiste en un transmisor, un receptor y un medio de transmisión para poder asegurar que el mensaje pueda moverse entre los dos puntos deseados. Tomando esto como base se puede decir que la combinación de las cuerdas vocales y el sistema auditivo humano se considera un "transceptor" ya que este es capaz de transmitir y recibir una señal que en este caso se encuentra en forma de sonido, mientras que el aire entre el transmisor y el receptor se considera el medio por el cual se mueve la señal. Para que una señal pueda entenderse en su totalidad ambos extremos del sistema deben utilizar el mismo "código" para el mensaje que en este caso se considera que ambas personas hablen el mismo idioma en el que se está transmitiendo el mensaje [3].

Debido a que los humanos siempre han tenido la necesidad de extender la distancia a la cual pueden moverse sus mensajes, esto ha provocado que se desarrollen varios sistemas de transmisión de señales y comunicación tales como palomas mensajeras, el telégrafo, radio, sistemas satélitales, etc. Este protocolo se centrará en dos sistemas de transmisión de datos creados por el humano que son la radio frecuencia y la comunicación Wi-Fi [3].

6.2. Comunicación RF (Radio Frecuencia)

Una señal de radio frecuencia (RF) se refiere a una señal inalámbrica electromagnética utilizada como una forma de comunicación en el ámbito de electrónicos inalámbricos. Las

ondas de radio son una forma de radiación electromagnética que se posee radio frecuencias identificadas que se encuentran en el rango de 3kHz hasta 300 GHz [4].

La frecuencia hace referencia a la tasa de oscilación de la onda. La propagación de una onda RF ocurre a la velocidad de la luz y no necesita de un medio específico para poder viajar. Las ondas RF también ocurren de manera natural de los rayos del sol, rayos eléctricos y las estrellas en el espacio que irradian ondas RF conforme estas envejecen. El hombre se comunica utilizando ondas de radio creadas de manera artificial que pueden oscilar en frecuencias especificas escogidas por el hombre. La comunicación RF es utilizada en muchas industrias como la televisión, sistemas de radas, plataformas de red móvil y de computadora, controles remotos y muchas más.

Mientras que componentes tales como mezcladoras, filtros, amplificadores de potencia se pueden clasificar de acuerdo a su rango de operación, estos no pueden ser categorizados según los estándares inalámbricos que utilizan ya que únicamente por la capa física (PHY) que utilizan [4].

6.3. Comunicación Wi-Fi

Wi-Fi es una tecnología de red inalámbrica que permite a varios dispositivos tales como computadoras, teléfonos celulares y otros dispositivos inteligentes interactuar con el internet. Permite a todos estos dispositivos intercambiar información entre sí de manera que se crea una red. Normalmente la conexión a internet ocurre por medio de un router. Cuando un dispositivo se conecta a una red Wi-Fi este se está conectando red manera inalámbrica a un router el cual crea el vínculo entre la red y el internet [5].

En el lado técnico, el estándar IEEE 802.11 define que los protocolos que activan la comunicación con un dispositivo inalámbrico que utiliza Wi-Fi, esto incluye dispositivos como routers inalámbricos y puntos de acceso inalámbricos. Pero ¿qué es un punto de acceso inalámbrico? Un punto de acceso inalámbrico (AP por sus siglas en inglés) permite que los que dispositivos inalámbricos se conecten a una red inalámbrica. Lo que un punto de acceso inalámbrico hace a la red es muy similar a lo que un amplificador hace con una guitarra eléctrica. Este punto de acceso toma el ancho de banda que emite el router principal y lo estira de manera que muchos dispositivos puedan conectarse a la red estirada desde distancias mucho mayores. Eso no es todo, un punto de acceso inalámbrico también es capaz de proveer información valiosa sobre los dispositivos que se encuentran conectados a la red local, puede proveer seguridad pro activa y muchos otros usos prácticos dentro de la red [5].

6.4. Módulo nRF24L01

Para este proyecto se estará utilizando un modulo nRF24L01 el cual es un chip transceptor que opera a 2.4 GHz con un motor de protocolo de banda base embebido (Enhanced ShockBurst) diseñado para aplicaciones inalámbricas de ultra bajo consumo energético. Este chip está diseñado para operar en las frecuencias de banda ISM que se encuentran entre 2.4 GHz y 2.4835 GHz. Tomando esto en cuenta el modulo nRF24L01 es muy poco exigente

al momento de ser implementado en un sistema ya que únicamente requiere de un microcontrolador y algunos componentes externos para poder funcionar dentro de un sistema de radio simple [6].

El motor de protocolo de banda base embebido utilizado se basa en comunicaciones de paquete y es capaz de soportar varios modos de uso que van desde operación manual hasta protocolos de operación autónoma. Los sistemas FIFO (First-in, First-out) internos aseguran un flujo de datos continuo entre el radio y el microcontrolador utilizado. En este caso el radio utilizado es una modulación GFSK y este permite que el usuario configure canales de frecuencia, potencia de salida y la tasa a la que se moverá la data a través del aire [6].

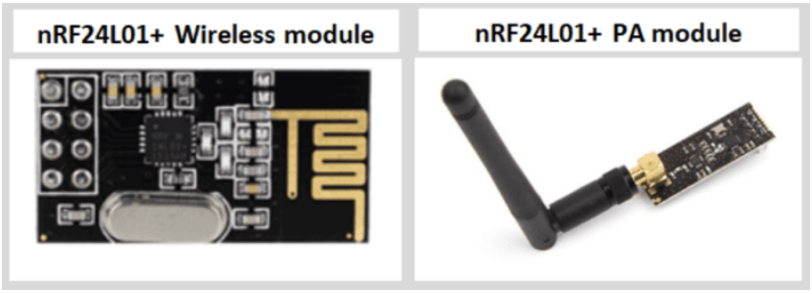


Figura 3: Módulo nRF24L01 (izquiera), módulo nRF24L01 con antena amplificadora (derecha)

6.5. ATmega328P

El ATmega328 es un microcontrolador creado por Atmel que pertenece a la serie megaAVR, este microcontrolador se usa comúnmente dentro de proyectos que tengan un bajo consumo energético y un bajo costo monetario y uno de los ejemplos más comunes para esto se encuentra en la plataforma Arduino ya que este es el microcontrolador principal para las plataformas Arduino UNO y Arduino NANO [7].

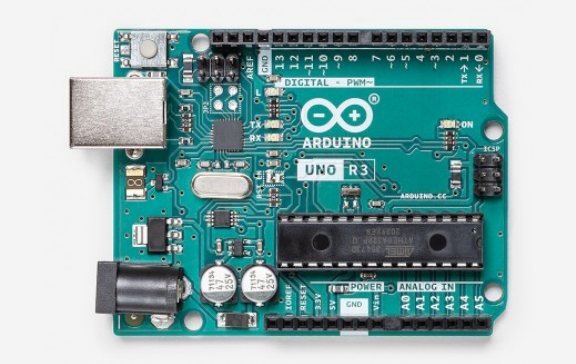


Figura 4: Arduino UNO R3.0 utilizando un microcontrolador ATmega328P

Este microcontrolador posee varias características que le han permitido convertirse en uno de los microcontroladores mas populares del mercado, algunas de estas características

son que posee 1KB de EEPROM (electrically erasable programmable read only memory) lo cual le permite que si la corriente eléctrica del microcontrolador de removida aun así puede almacenar datos y puede proveer de resultados después de volver a conectar la corriente eléctrica. Otras características muy importantes son que la arquitectura RISC es bastante avanzada, bajo consumo de poder, 6 pines PWM, contadores de tiempo real que poseen un oscilador separado, puertos USART programables, entre otras [7].

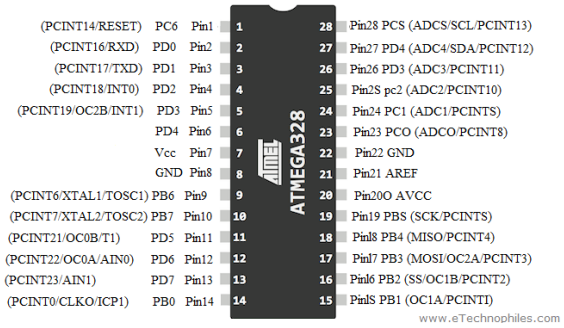


Figura 5: Pinout del microcontorlador ATmega328p

6.6. JavaScript

JavaScript es un lenguaje de programación que permite implementar características complejas en páginas web, por lo que cada vez que una página web hace algo más que solo desplegar información de manera estática es muy probable que JavaScript este implementado en esa página. JavaScript es una de las partes mas importantes hoy en día en la tecnología de web [8].

El lado del cliente de JavaScript consiste en algunas características de programación comunes que permiten al usuario hacer cosas como:

- Almacenar valores útiles dentro de variables.
- Operaciones en pedazos de texto o "strings" para unir varios strings en uno solo o tareas parecidas.
- Ejecutar ciertas partes del código como respuesta a un evento ocurrido en la página web.

Lo que es mucho mas interesante es la funcionalidad incluida en el lado del cliente del lenguaje ya que este tiene lo que se conoce como interfaces de programación de aplicaciones (APIs por sus siglas en inglés) que a grandes rasgos pueden describirse como conjuntos de códigos de construcción hechos y listos para usar los cuales permiten al desarrollador hacer las implementaciones de esos códigos mucho más fáciles dentro del proyecto. Las APIs puedes dividirse en dos categorías que son APIs de buscador y APIs de terceros [8].

Las APIs de buscador están hechas en los buscadores y son capaces de realizar tareas complicadas como mostrar información de una red local de computadoras. Algunos ejemplos de diferentes APIs de buscador son:

- Modelo de objetos de documento (DOM) APIs permiten manipular la parte de HTML y CSS de la página permitiendo modificar, remover o crear estas partes de maneras dinámicas para cambiar la apariencia de la página. Los mensajes emergentes son ejemplos muy comunes de la implementación de DOM.
- Las APIs de geolocación permiten extraer información geográfica y usarla de manera que se pueda crear interpretar como locaciones en tiempo real.
- Las APIs de audio y video permite al usuario reproducir archivos de audio y video dentro de la página web, también permiten extraer el video de una cámara web y desplegarlo en tiempo real dentro de la página.

Mientras tanto las APIs de terceros son desarrolladas de manera externa por lo que son menos especializadas en comparación a las APIs de buscador y estas también pueden incorporarse en proyectos ya existentes como funciones externas. Este tipo de APIs tiene una fortaleza muy grande en comparación al resto ya que mientras no sea necesaria una solución que no sea demasiado especifica las APIs de terceros son mas eficientes y son una solución más segura, también permiten ahorrar una cantidad de tiempo muy grande ya que como se pueden usar funciones de otros proyectos ya no es necesario crear muchas cosas de cero. Un ejemplo de esto sería una función de mapas en una aplicación, como ya existen diversas aplicaciones que pueden realizar esta función de una manera muy eficiente no seria necesario crear la función de cero para implementarla en una en una aplicación [8].

6.7. React JS

React.js es una librería de carácter de fuente abierta de JavaScript que es popularmente utilizada para crear aplicaciones y así poder manejar la capa visual de aplicaciones web y móviles. React permite a los usuarios crear componentes reusables para la interfaz del usuario. React permite a los programadores crear páginas web extensas que pueden cambiar su información sin necesidad de recargar la página cada vez que esto ocurra. Uno de los propósitos principales de React es permitir y facilitar la creación de aplicaciones y de esta manera permitir que estas sean rápidas, simples y fáciles de mejorar o cambiar [9].

Hoy en día React.js es muy utilizado a nivel global ya que presenta una barrera muy pequeña para poder ser utilizado en comparación a otras opciones que existen en el mercado. Algunas de las cosas que permiten a React disminuir las limitantes es que se comporta de manera declarativa lo cual significa que es muy fácil crear interfaces de usuario interactivas. Al utilizar código declarativo se vuelve mucho más predictivo, simple de entender y fácil de cambiar o arreglar. Otra de las grandes razones es que es una librería basada en componentes, esto significa que se pueden construir componentes que actúan de manera independiente en la aplicación y así poder crear interfaces de usuario bastante complejas de manera que los componentes independientes no se afecten entre sí [9].

6.8. Internet de las cosas (IoT)

Cualquiera que haya apuntado un navegador web en un motor de búsqueda o alguna red social conocer el poder que tiene el internet para conectar a millones de personas entre si o para brindar información en cantidades extremadamente altas. Aun así, con el aumento del desarrollo de varios dispositivos inteligentes el internet seguirá evolucionando para poder incluir lo que algunos llamas internet de las cosas (Internet of Things o IoT por sus siglas en inglés) que es un concepto que se basa en que billones de dispositivos inteligentes estarán interconectados entre si midiendo, moviendo y actuando en respuesta a alguna cosa o de manera independiente toda la información que se crea y maneja en el diario vivir del ser humano

Se planea que el internet de las cosas se expanda al punto en el que no sea únicamente personas interactuando con dispositivos sino que también incluya dispositivos interactuando entre si para llegar a formar algo parecido a un sistema nervioso a nivel global en el cual se pueda hacer miles de cosas como doctores examinando pacientes de manera remota utilizando dispositivos inteligentes de medicina y poder conocer el estado del paciente en tiempo real para dar diagnósticos acertados o que las empresas de energía puedan monitorear miles de kilómetros de tuberías de aceite y combustible de manera que si llegara a ocurrir algún incidente se pueda dar una respuesta inmediata sin importar la distancia [10].

Para poder crear este futuro se han creado tres pilares que ayudaran a transformar al planeta en un planeta inteligente, estos pilares son:

- Instrumentación: la información se debe poder capturar en cualquier lugar y momento, esto se puede llegar a lograr con la implementación de varios sensores remotos en los dispositivos inteligentes.
- Interconectividad: la información se debe poder mover de un punto de almacenamiento a otro punto donde pueda ser consumida sin ningún problema.
- Inteligencia: la información se debe poder procesar, analizar y actuar sobre ella para poder lograr un máximo valor de conocimiento y uso.

Gracias a los avances tecnológicos que se han hecho a lo largo de los años estos tres pilares ya están en un estado bastante avanzado, por ejemplo, la instrumentación se cumple de manera muy efectiva y algunos ejemplos de esto son las nuevas tiendas de Amazon que se basan en el uso de etiquetas RFID (Radio-frequency identification) y sincronización con el teléfono del cliente para poder realizar los cobros de los productos comprados. Mientras tanto el pilar de interconectividad ha evolucionado de maneras abismales ya que a lo largo de los años se han creado ecosistemas de servicios y dispositivos que permiten una rápida interconectividad entre si como por ejemplo el ecosistema mas poderoso del mundo que es el que ha creado Apple con todos sus productos de manera que ya hecho posible que el compartir información, sincronizar dispositivos y muchas más otras cosas sean extremadamente fáciles de realizar y todo esto también va de la mano con el ultimo pilar planteado que es el pilar de inteligencia ya que para que todo este proceso de interconectividad sea posible es necesario que los dispositivos que se encuentran dentro del ecosistema sean inteligentes en su totalidad para poder analizar los cambios dentro del ecosistema y adaptarse a este [10].

6.9. MQTT

MQTT es un protocolo estándar de mensajería de OASIS que se utiliza para el internet de las cosas. Está diseñado como un transporte de mensajería de publicación y suscripción liviano que es ideal para poder conectar dispositivos remotos con una huella de código pequeña y un ancho de banda de red mínimo. Hoy en día MQTT se utiliza en una amplia variedad de industrias como la automotriz, telecomunicaciones, entre otras [10].

El protocolo MQTT está construido sobre varios conceptos básicos que se centran en mantener un buen envió de datos mientras que se trata de que estos se mantengan de la manera mas liviana posible. Algunos de los conceptos básicos que se utilizan dentro del protocolo MQTT son:

- Publicar/suscribir: uno de los principios básicos del protocolo MQTT es que uno de los extremos de la comunicación publique un mensaje mientras que otro(s) se puedan suscribir a un tópico específico en el cual se encuentre dicho mensaje, esto comúnmente se conoce como un modelo de publicación/suscripción. Los clientes pueden suscribirse a algún tópico de su interés lo cual permitirá que reciban mensajes relacionados a dicho tópico y de igual manera cuando un cliente publica un mensaje este debe tener un tópico al cual otro cliente pueda suscribirse.
- Tópicos/suscripciones: como ya se mencionó en el protocolo MQTT los mensajes son publicados dentro de tópicos (estos son similares a un área de interés para el cliente). Los clientes por su parte se suscriben para poder recibir mensajes específicos que se encuentren dentro del tópico o tópicos a los cuales están suscritos. Las suscripciones a los tópicos pueden ser explícitas, las cuales limitan los mensajes recibidos únicamente a los que se publican dentro del tópico al cual se está suscrito o pueden utilizar un "designador comodín" el cual permite recibir mensajes de una variedad de tópicos disponibles.
- Niveles de calidad de servicio: MQTT define principalmente tres niveles de calidad de servicio (QoS por sus siglas en ingles) para el envío de mensajes. Cada uno de estos niveles tienen designado por el servidos un nivel de esfuerzo mayor que el anterior para poder asegurar que los mensajes si sean enviados. El utilizar niveles de calidad altos ayuda a asegurar una conexión para él envió de los mensajes de mucha más confianza, pero al costo de tener un consumo de ancho de banda mucho mayor lo cual puede afectar al tiempo de envió de los mensajes.
- Mensajes retenidos: al utilizar MQTT todos los mensajes que han sido enviados a los suscriptores actuales son guardados dentro del servidos. Esto se hace ya que si un nuevo suscriptor entra a un tópico especifico entonces todos los mensajes que han sido retenidos pueden ser enviados al nuevo suscriptor para asegurar que cuente con la misma cantidad de información que el resto de los suscriptores.
- Sesiones limpias y conexiones duraderas: cuando un cliente de MQTT se conecta al servidor este crea una "bandera de sesión limpia" y la coloca como verdadera. En el caso de la bandera de sesión limpia se encuentre en verdadero todas las suscripciones del cliente son eliminadas cuando este se desconecte del servidor. Mientras que si la bandera se encuentra en falsa entonces la conexión se trata como que fuera duradera

y esto permite que las suscripciones del cliente se mantengan sin importar que este se desconecte del servidor, cuando esto ocurre todos los mensajes que llegan luego de la desconexión con un nivel de calidad alto son guardados para envió al cliente cuando la conexión con este se reestablezca.

- Deseos: cuando un cliente se conecta al servidor este puede informarlo que este tiene un "deseo", o un mensaje que debe publicarse a un tópico o tópicos en específico en el caso de una desconexión inesperada del servidor.

6.9.1. Cliente

Cuando se habla de clientes la mayoría de las veces se refiere a los suscriptores y a los publicadores dentro del entorno de MQTT. Estas etiquetas de suscriptor y publicador sirven para poder identificar cuando es que un cliente está enviando o recibiendo mensajes (un cliente puede ser un suscriptor y un publicador a la vez). Un cliente de MQTT puede ser cualquier dispositivo que va desde un microcontrolador hasta un servidor a gran escala que utiliza las librerías de MQTT disponibles y se conecta directamente a un broker poder medio de la red [11].

6.9.2. Broker

El complemento de los clientes para que la comunicación pueda suceder es el broker. Este es el responsable de recibir, filtrar, determinar las diferentes suscripciones de los clientes y enviar todos los mensajes a sus respectivos tópicos para que los clientes puedan recibirlos. El broker también mantiene los datos de la sesión de todos los clientes que utilizan sesiones duraderas. Otra de las grandes responsabilidades que tiene este componente es el poder autenticar y autorizar que un cliente pueda entrar a la sesión actual, usualmente el broker facilita la autenticación personalizada para diferentes clientes y también facilita la integración con sistemas de "backend" para distintas aplicaciones. La implementación del broker es bastante importante ya que en la mayoría de los casos este es el componente que se encuentra conectado directamente a internet y esto significa que el sistema de autenticación de clientes que este utilice puede ayudar a que clientes indeseados no puedan entrar en las sesiones donde se maneje información sensible [11].

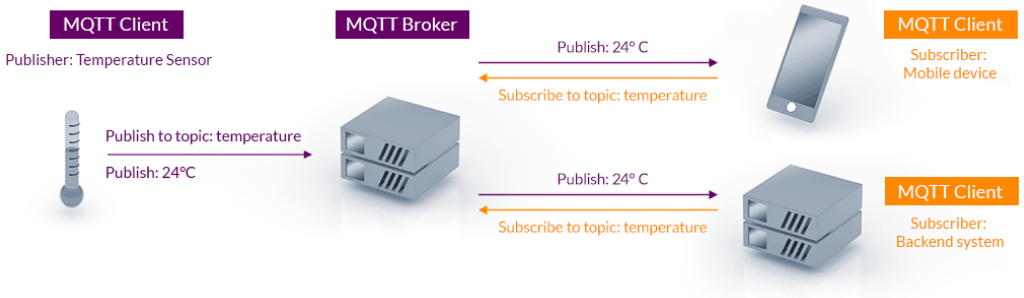


Figura 6: Diagrama de comunicación del protocolo MQTT

6.10. Bases de datos

Una base de datos se puede definir como una colección organizada y estructurada de información y/o datos que típicamente son recaudados utilizando medios electrónicos en una computadora. Las bases de datos típicamente están controladas por un sistema de manejo de bases de datos (DBMS por sus siglas en inglés), esto junto a la data recolectada y las aplicaciones especificas que se asocian con el manejo de bases de datos se conoce comúnmente como un sistema de bases de datos [12].

Hoy en día la información dentro de las bases de datos se modela de manera que se crean filas y columnas que crean una serie de tablas, esto permite que el acceso a la información de cada una de estas tablas sea más rápido y eficiente. Al tomar esto en cuenta se puede hacer el argumento que una base de datos es muy parecida a una hoja de calculo de Microsoft Excel ya que en ambos casos la información puede almacenarse de la manera anteriormente descrita, ¿pero qué es lo que diferencia una base de datos de una hoja de cálculo? Algunas de las diferencias principales entre estos dos métodos de almacenamiento de datos son:

- La manera en que la data es manipulada y almacenada.
- Quiénes tienen acceso directo a la información almacenada.
- Cuánta información puede ser almacenada dentro de cada método.

Las hojas de cálculo están diseñadas para que únicamente un usuario manipule la información almacenada o en algunos casos un grupo pequeño de personas. Por el contrario, las bases de datos están hechas para permitir que múltiples usuarios puedan acceder simultáneamente a la información y que este acceso se mantenga seguro. También permiten grandes cantidades de almacenamiento, ideal para organizaciones grandes [12].

6.10.1. ¿Qué es un programa de bases de datos?

Un programa de bases de datos es utilizado principalmente para crear, editar y mantener archivos de bases de datos facilitando la creación de entradas, ediciones y actualizaciones de datos. Normalmente el programa también maneja el almacenamiento de los datos, la creación de las respectivas copias de seguridad y el reporte de múltiples accesos. Los programas de bases de datos tienen la característica distintiva que simplifican la experiencia del usuario cuando este requiera de almacenar o acceder datos. Normalmente estos programas cuentan con una interfaz gráfica que permite una mejor visualización de la información almacenada [12].

6.10.2. Tipos de bases de datos

Existen muchos tipos diferentes de bases de datos, y para poder determinar cual es la base de datos ideal para una organización es necesario conocer como se planea utilizar la información [12].

Bases de datos relacionales: los objetos dentro de una base de datos relacionales están organización en conjuntos de tablas compuestas por filas y columnas. Estas bases de datos proveen de mejor y mas eficiente manera de acceder a la información almacenada.

Bases de datos orientadas en objetos: la información de estas bases de datos se encuentra representada como objetos, como sucede normalmente en la programación orientada a objetos.

Bases de datos distribuidas: una base de datos distribuida consiste en dos o mas archivos localizados en diferentes sitios. La base de datos puede ser accedida en múltiples computadoras localizadas en la misma área física o distribuidas en diferentes redes.

Depósito de datos: es un repositorio principal de datos que está diseñado para poder realizar análisis y consultas de los datos de una manera rápida.

Bases de datos NoSQL: una base de datos NoSQL o no relacionable, permite el almacenamiento y manipulación de los datos de una manera no estructurada o semi estructurada.

Bases de datos de fuente abierta: estas bases de datos proporcionan el código fuente del sistema utilizado.

Bases de datos en la nube: estas bases de datos pueden contar o no con una estructura especifica para almacenar los datos, estos residen en una plataforma computacional que puede ser privada, publica o híbrida. Los dos tipos más comunes de bases de datos en la nube son las tradicionales de las bases de datos como un servicio (DBaaS por sus siglas en inglés), en estas últimas todas las tareas administrativas y de mantenimiento son realizadas por el proveedor del servicio.

Bases de datos de documentos/JSON: están diseñadas para almacenar, retraer y manejar información de manera orientada a documentos, esta es una de las mejores maneras de manejar información en formato JSON.

Bases de datos autónomas: este tipo especifico de bases de datos se caracteriza porque están situadas en la nube y utilizan aprendizaje automático para automatizar la seguridad, copias de seguridad, actualización y muchas otras rutinas de administración de una base de datos convencional.

CAPÍTULO 7

Control remoto

7.1. Diseño del control remoto

El diseño del control remoto inició en la fase de bosquejo ya que era importante conocer la forma óptima que este control debía tener y las características con las que debía cumplir para que fuerza capaz de realizar la tarea deseada. En esta fase se determinó que el control tendría un pad direccional (D-pad) y botones que corresponderían a las acciones que se desea que haga el robot. Las acciones que serían realizadas por el robot serian de cambio de velocidad y cambio de modo de uso (automático o manual). El pad direccional se encargaría de los distintos movimientos del robot que son avance, retroceso, giro a la izquierda y giro a la derecha.

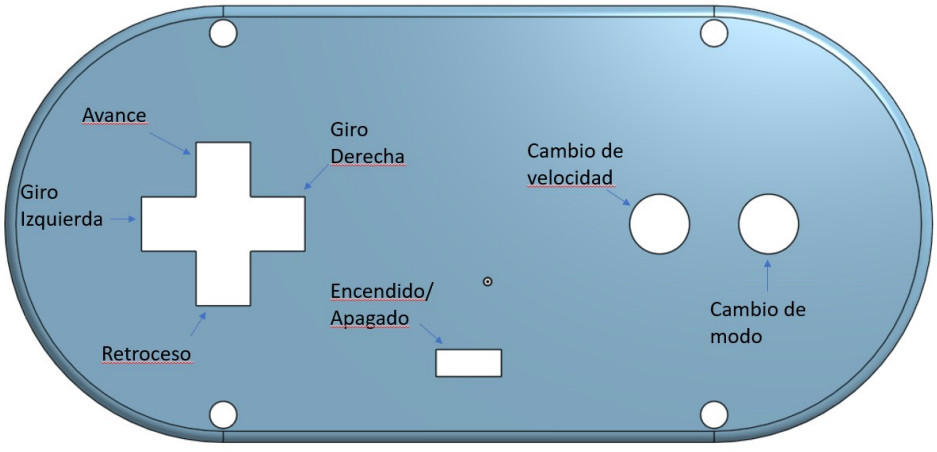


Figura 7: Layout y funcionamiento de cada uno de los botones dentro del control remoto

Para poder diseñar de manera correcta y eficiente el control remoto se hizo uso de la norma: "Seguridad de las máquinas: Requisitos ergonómicos para el diseño de dispositivos de información y mandos parte 3: Mandos UNE-EN 894-3:2001+A1" la cual brinda diferentes características que debe cumplir un mando para que pueda ser considerado como ergonómicamente correcto. Algunas de las características que debe tener el mando para poder cumplir con la norma son el tipo de agarre que este debe poseer, tamaño de los botones con relación al tamaño del mando y la fuerza que debe poder soportar cada una de las piezas. Seguido de esto se implementará la norma DIN 33402 para poder identificar las dimensiones antropométricas que se relacionan con las dimensiones del mando para asegurar la seguridad del usuario.

La norma UNE-EN 894-3:2001+A1 proporciona una serie de formularios para poder identificar las diferentes funciones y características de cada una de las partes principales del mando. Las partes principales que tendrá el mando en este caso son el pad direccional, los botones de acción y el cuerpo principal. Estos formularios están hechos de manera que se pueda identificar el grado de requisito que es necesario para cada una de las partes y de ser necesario conocer la fuerza aproximada a la cual estará sometida cada una de las piezas. Estos formularios proporcionan una tabla de descriptiva para poder visualizar de mejor manera estas características.

Codigo	Grado de requisito	Fuerza
0	Despreciable	<10 N
1	Bajo	>=10 a <25 N
2	Medio	>=25 a <50 N
3	Alto	>=50 a <80 N
4	Muy Alto	>=80 a <120 N

Cuadro 1: Tabla proporcionada para determinar grados de requisito para diseño de mandos

Para poder llenar los formularios correspondientes al pad direccional y los botones de acción se utilizaran los datos obtenidos en el artículo científico "Size, strength and physical exposure differences between adult and child computer users" [13] en el cual se realizaron mediciones de la fuerza promedio que puede ejercer un niño y un adulto al oprimir el botón de un ratón de computadora y así poder determinar si existe una diferencia significativa al momento de utilizar equipo estándar y equipo diseñado para niños. Según los resultados obtenidos en dicha investigación se determinó que la fuerza promedio que puede ejercer un adulto promedio es de 4.5 lbs o 20.017 N aproximadamente. Utilizando este dato se puede catalogar a estas piezas con un requisito bajo de diseño.

D-Pad					
Descripción de la información	Grado de requisito				
	0	1	2	3	4
Requisitos generales de la tarea					
Precisión				x	
Velocidad				x	
Fuerza		x			
Requisitos específicos de la tarea					
Examen visual	x				
Examen táctil		x			
Accionamiento involuntario	x				
Fricción			x		
Uso de guantes	x				
Facilidad de limpieza			x		

Cuadro 2: Formulario utilizado para determinar características necesarias para el diseño del pad direccional

Características del movimiento						
Tipo de movimiento	Lineal			Rotativo		
Eje de movimiento	x	y	z	x	y	z
Dirección del movimiento	+/-	+/-	+/-	+/-	+/-	+/-
Continuidad del movimiento	Continuo			Discreto		
Características de prensión						
Tipo de prensión	Contacto		Pinza		Agarre	
Parte de la mano que aplica fuerza	Dedo			Mano		
Método de aplicación de la fuerza	Normal			Tangencial		

Cuadro 3: Formulario utilizado para determinar características necesarias para el diseño del pad direccional

Botones					
Descripción de la información	Grado de requisito				
	0	1	2	3	4
Requisitos generales de la tarea					
Precisión				x	
Velocidad			x		
Fuerza		x			
Requisitos específicos de la tarea					
Examen visual	x				
Examen táctil		x			
Accionamiento involuntario	x				
Fricción			x		
Uso de guantes	x				
Facilidad de limpieza			x		

Cuadro 4: Formulario utilizado para determinar características necesarias para el diseño de los botones de acción

Características del movimiento						
Tipo de movimiento	Lineal			Rotativo		
Eje de movimiento	x	y	z	x	y	z
Dirección del movimiento	+/-	+/-	+/-	+/-	+/-	+/-
Continuidad del movimiento	Continuo			Discreto		
Características de prensión						
Tipo de prensión	Contacto	Pinza		Agarre		
Parte de la mano que aplica fuerza	Dedo			Mano		
Método de aplicación de la fuerza	Normal			Tangencial		

Cuadro 5: Formulario utilizado para determinar características necesarias para el diseño de los botones de acción

Luego de determinar el grado de requisito necesario para la fuerza que debe soportar cada una de estas piezas se le otorgó un grado de requisito a la precisión que debe tener la pieza y la velocidad de respuesta con la que debe contar, en el caso del pad direccional la velocidad de respuesta se determinó que debe ser alta ya que esto es lo que moverá el robot y si la velocidad de respuesta de esta pieza no es lo suficientemente buena podría ocasionar un accidente. Mientras tanto para los botones de acción se colocó un grado de requisito medio debido a que estos pueden funcionar sin que el robot este en movimiento por lo que no es critico que tengan una respuesta tan inmediata para las acciones que realizaran.

El siguiente juego de características es el de requisitos específicos de la tarea, en este juego de características se utilizaron los mismos grados de requisito para ambas piezas ya que ambas cumplirán con tareas similares dentro del mando. Aquí se le proporcionó un grado de requisito despreciable al examen visual, accionamiento involuntario y al uso de guantes para las piezas ya que no es significativo que la estética de las piezas sea perfecta y tampoco se quiere que estas piezas se acciones de manera involuntaria. En el examen táctil se utilizó el grado de requisito bajo ya que únicamente se quiere que las piezas puedan tocarse sin causar ninguna incomodidad al operario, el grado de requisito otorgado a la fricción es medio ya que cuando el operario este utilizando el mando no se quiere que los dedos se resbalen tal fácilmente de las piezas, pero tampoco se quiere que sean lo suficientemente ásperos como para lastimar al operario. Por último la facilidad de limpieza de igual manera tiene un grado de requisito medio ya que no se quiere que el operario intente limpiar el mando por su cuenta debido a que podría dañar los componentes internos, pero si se quiere que la persona designada del mantenimiento del mando pueda limpiarlo sin ningún problema y sin dañar ningún componente de este.

Las características del movimiento son el siguiente grupo de requisitos que se debía catalogar y dentro de este grupo se encuentra el tipo de movimiento que tendrá la pieza, eje del movimiento de la pieza, dirección del movimiento y la continuidad del movimiento. Ya que en este caso el pad direccional puede considerarse como un juego de botones direccionales entonces se le asigna un tipo de movimiento lineal al igual que a los botones de acción y para poder asignar el eje del movimiento se planteó que el eje “z” seria el eje en el cual los objetos puedan subir o bajar por lo que este es el eje perfecto para asignar dentro de los formularios, de la mano con esto se sabe que los botones se accionaran únicamente cuando estos vayan hacia abajo por lo que la dirección del movimiento de las piezas es en la parte negativa del eje “z”. Por último, ya que cada vez que se quiere que las piezas funcionen estas

deben accionarse se toma esto como un movimiento discreto dentro del último requisito de este juego de características.

El último juego de características que se otorgan dentro de los formularios son las características de prensión para cada una de las piezas, dentro de estas se encuentra el tipo de prensión, la parte de la mano que aplicará la fuerza a cada pieza y el método de la aplicación de la fuerza. El tipo de prensión que tendrán tanto el pad direccional como los botones de acción es únicamente de contacto ya que estos se deben oprimir para poder ser accionados, tomando esto en cuenta se asignan los dedos como la parte principal de la mano que aplicará la fuerza a las piezas. Ya que la acción que se debe realizar para poder accionar las piezas es oprimir la fuerza será aplicada de manera normal a las mismas.

Para poder diseñar correctamente el cuerpo principal del mando se utilizará el mismo formulario provisto por la norma UNE-EN 894-3:2001+A1, pero en este caso los datos necesarios para poder utilizar de manera correcta este formulario serán extraídos del artículo científico "The Influence of Hand Tool Design on Hand Grip Strength: A Review" [14] para poder otorgar el grado de requisito necesario a los diferentes requisitos. En este articulo científico se tomaron medidas del tamaño de la herramienta que se podría utilizar ya fuerza promedio que necesita un obrero para poder sostenerla manteniendo un agarre sólido de esta. Aquí se determinó que para una herramienta cuyas medidas se encuentren entre 25.4mm y 38.1mm la fuerza de agarre necesaria es aproximadamente de 34.8 N, al comparar estas medidas con las que se tenían planeadas inicialmente para el cuerpo principal del mando que serian de aproximadamente 30mm entonces el grado de requisito que se le otorgará a la fuerza que debe soportar el mando es un grado medio, por otra parte en el requisito de la precisión se le asigna un grado bajo ya que esta será una pieza estática por lo que la única precisión que se necesita en este caso es que se pueda sostener de manera que permita un fácil acceso a los botones y el pad direccional. Mientras tanto el último requisito que es el de la velocidad es despreciable por la misma razón que es una pieza estática.

Cuerpo del Mándo					
Descripción de la información	Grado de requisito				
	0	1	2	3	4
Requisitos generales de la tarea					
Precisión		x			
Velocidad	x				
Fuerza			x		
Requisitos específicos de la tarea					
Examen visual		x			
Examen táctil					x
Accionamiento involuntario	x				
Fricción			x		
Uso de guantes	x				
Facilidad de limpieza			x		

Cuadro 6: Formulario utilizado para determinar características necesarias para el diseño del cuerpo principal del mando

Características del movimiento						
Tipo de movimiento	Lineal			Rotativo		
Eje de movimiento	x	y	z	x	y	z
Dirección del movimiento	+/-	+/-	+/-	+/-	+/-	+/-
Continuidad del movimiento	Continuo			Discreto		
Características de prensión						
Tipo de prensión	Contacto		Pinza		Agarre	
Parte de la mano que aplica fuerza	Dedo			Mano		
Método de aplicación de la fuerza	Normal			Tangencial		

Cuadro 7: Formulario utilizado para determinar características necesarias para el diseño del cuerpo principal del mando

De igual manera que en las piezas anteriores los siguientes requisitos a los cuales es necesario otorgarle un grado son los requisitos específicos de la tarea, en estos los únicos requisitos que son despreciables son el de accionamiento involuntario debido a lo explicado anteriormente y el uso de guantes ya que la pieza como tal no será peligrosa al ser sujetada con las manos. El examen visual tiene un grado bajo ya que no la estética no es un factor muy importante en esta etapa del proyecto, contrario a esto el examen táctil tiene un grado de requisito muy alto debido a que el mando debe estar hecho de un material que no dañe las manos del operario y que también le permita sujetarlo de una manera cómoda. El requisito de fricción tiene un grado medio por la misma razón que no se quiere que el material del que este hecho dañe las manos del operario pero si que le permita mantener un agarre firme de la pieza y por último la facilidad de limpieza también tiene un grado de requisito medio ya que no se desea que el operario pueda intentar limpiarlo por su cuenta ya que podría dañar la pieza pero si se quiere que la misma tenga cierta facilidad de limpieza para la persona encargada del mantenimiento del dispositivo.

Las características del movimiento de esta pieza no se aplican ya que como se ha mencionado es una pieza estática por lo que no tendrá ningún tipo de movimiento. En el caso de las características de prensión se asignaron dos criterios para el requisito del tipo de prensión, estos siendo el de contacto y el de agarre ya que ambos se cumplirán al momento que el operario tome el mando. Para la parte que aplicará la fuerza se asignó que será la mano la parte principal que asignará la fuerza y por último el método de aplicación de la fuerza también será normal debido a la manera en que el mando se deberá sujetar para que el operario lo pueda utilizar.

7.2. Comunicación por radiofrecuencia

Para poder desarrollar la parte de la comunicación inalámbrica por medio de radio frecuencia se utilizarán los módulos NRF24L01+PA ya que estos ofrecen un mayor alcance para la transmisión de los datos de manera que este factor puede ser descartado como una posible fuente de error en el funcionamiento de estos. Para comenzar con su implementación se creó una pequeña prueba en la cual la transmisión de los datos sería en una sola dirección de manera que uno de los módulos siempre actuaria como transmisor y el otro modulo siempre actuaría como receptor y para esto se utilizara la librería de NRF24 que existe den-

tro de Arduino. En esta prueba únicamente se enviarían los datos de un potenciómetro del transmisor al receptor y se verificaría que estos están siendo recibidos utilizando el puerto serial de receptor.

Una vez se tuvo esta prueba funcionando de manera correcta se aumento el nivel de complejidad para que ahora ambos módulos tuvieran la capacidad de enviar y recibir datos, esta prueba se realizó de manera sencilla utilizando dos LED y dos botones para cada uno de los módulos y la meta fue el poder encender una LED de un color en uno de los módulos y que este enviará una señal al otro módulo para que también encendiera la LED del mismo color que este tenía por lo que al final cada uno de los módulos debería ser capaz de encender las dos LED que tienen y que el módulo contrario encendiera la LED que hacía juego. De igual manera se utilizó el puerto serial de los módulos para poder verificar que la información se estaba enviando de manera correcta por lo que cada vez que se encendía uno de los LED se enviaba el mensaje con el color del LED que se encendió y este se desplegaba en el puerto serial de ambos módulos. Esta sería la última prueba que se realizaría para comprobar no solo el buen comportamiento de los módulos, sino que también para poder comprender de mejor manera como se comportan los módulos.

Luego de haber completado ambas pruebas del funcionamiento de los módulos se comenzaría a trabajar de manera más directa en las funciones que debe tener el control remoto para el robot y estas funciones son el poder moverse accionando los dos motores laterales que tendrá el robot, poder cambiar la velocidad que tendrán los motores y poder alternar entre los modos manuales y automáticos para el manejo del robot. Inicialmente se utilizará una pequeña simulación de los motores y los cambios de modo y velocidad haciendo uso de LEDS de manera que se tienen dos LEDS que representarán la activación de los dos motores, un LED que se utilizará con un toggle para poder simular que el robot se encuentra en modo automático o en modo manual y, por último, un LED RGB el cual representara la velocidad de los motores alternando de colores según se oprima el botón dentro del control remoto.

Inicialmente se realizaba él envió de datos de manera puntual por lo que se asignaba una variable a cada uno de los botones que se utilizarían y se enviaban por medio de radiofrecuencia cada una de estas variables, al inicio esto no representaba un problema ya que se comenzó enviando los comandos para el movimiento direccional de los motores y estos eran únicamente cuatro movimientos, pero una vez se comenzaron a incorporar más variables para las funciones restantes los módulos no eran capaces de mantener el ritmo de envío de datos por lo que estos fallaban y confundían los datos enviados creando diferentes activaciones involuntarias y no deseadas. Para poder corregir esto se optó por crear un arreglo de datos que serían enviados como un solo "paquete" en el cual se guardaría cada uno de los comandos dentro de una posición diferente de arreglo y de igual manera para poder leer cada uno de los comandos guardados se revisará únicamente la posición deseada del arreglo por lo que no hay riesgo de crear una confusión en el código que provoque las acciones involuntarias que se estaban obteniendo anteriormente al intentar enviar cada una de las variables de manera individual.

El código en cuestión está dividido en dos partes: la primera fue utilizada únicamente en el lado del emisor de datos y la segunda fue utilizada en el lado del receptor de datos. En la primera parte del código es donde se realiza la lectura de todos los comandos para luego asignar los valores de cada lectura a una variable que sería colocada en una posición del arreglo de datos. Para poder establecer una comunicación entre ambas partes (receptor

y emisor) fue necesario utilizar un código especial que se comparta de manera que actuó como una contraseña que ambas partes deben tener para saber que se deben conectar entre sí. Luego de definir dicho código se utiliza la función "radio.write" que está incluida dentro de la librería de NRF24 que se utiliza, en esta función se envían como parámetros el arreglo de datos que se está utilizado y el tamaño en bytes de este mismo arreglo.

Ahora en la parte del receptor es donde se definen todas las salidas del control remoto que en este caso ya que se están simulando las salidas en esta parte del código se configuran las LEDS que representan los comportamientos del motor y otros aspectos del robot. Al tener bien configuradas todas las salidas se procede a utilizar el mismo código de comunicación que se estableció en la parte del emisor para asegurar que ambos módulos estarán conectados entre si y se utiliza la función "radio.read" que también está incluida en la librería de manera que los parámetros que se usaran serán exactamente los mismos que se utilizaron en la función "radio.write" que son el arreglo de datos y el tamaño en bytes de este arreglo.

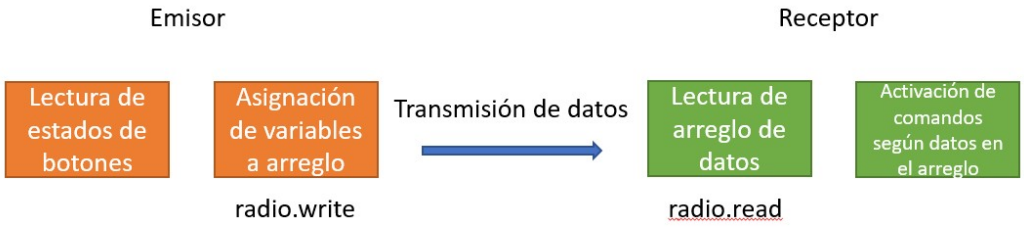


Figura 8: Diagrama de funcionamiento de comunicación por radiofrecuencia

7.3. Iteración No.1

La construcción del control remoto está compuesta de dos partes que son el diseño CAD de la carcasa del control remoto y el diseño de la PCB. Se comenzó realizando el diseño CAD de la carcasa del control remoto ya que al tener esta completa se podrá saber las dimensiones y el espacio disponible para poder crear el diseño más efectivo de la PCB.

Para el poder realizar el diseño de la carcasa se utilizaron los formularios llenados anteriormente para asegurar la buena ergonomía de esta, también se utilizaron algunas referencias de controles comerciales para crear algunas de las partes. Una de las referencias principales que se utilizó fue la forma general del control ya que para esto se tomó como base un control de Super Nintendo (SNES) debido a que este es tomado comúnmente por los críticos como uno de los diseños más cómodos de sujetar debido a la falta de lados rectos lo cual no provoca ninguna incomodidad en las palmas de las manos. También se tomó como referencia la implementación del pad direccional que tiene Sony en los controles Dualshock ya que aquí no utiliza una sola pieza en forma de cruz sino que se crea una cruz utilizando botones individuales y se decidió adoptar esta opción para poder minimizar la cantidad de movimientos accidentales que podrían ocurrir al momento de utilizar esta parte del control y también permite seguir cumpliendo con los formularios utilizados ya que se necesita que esta parte tenga un grado de precisión alto al momento de utilizarlo.

El control remoto está dividido en tres partes las cuales son la tapadera, el cuerpo principal y el almacenamiento de la batería. Se optó por crear esta división de las partes para

poder facilitar la manufactura del control en una impresora 3D ya que el intentar disminuir la cantidad de partes fusionándolas podría provocar que se creen geometrías complejas lo cual crearía un riesgo de fallo al momento del proceso de impresión y también afectaría el proceso de armado y desarmado volviéndolo más complicado para todos lo cual derrotaría el propósito de asignarle un grado de requisito medio al requisito de facilidad de limpieza. El cuerpo principal del control tiene una altura de 25 mm y posee 4 agujeros de 3 mm de diámetro, estos 4 agujeros tienen el propósito de alinearse con la tapa del control como se puede observar en la Figura 7. El cuerpo principal también posee un agujero en la parte baja donde se colocará la pieza para el almacenamiento de la batería y también se agregó una pequeña base con una elevación de 10 mm para poder sujetar la PCB con los componentes y así mantenerla asegurada dentro del control.

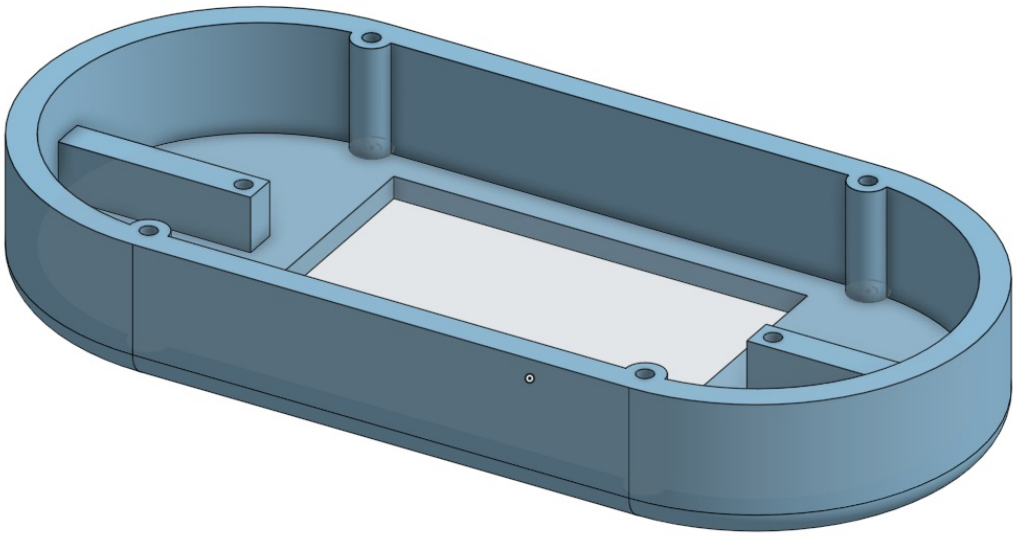


Figura 9: Modelo CAD del cuerpo principal diseñado para el control remoto

La tapadera del control tiene los 4 agujeros necesarios para poder utilizar los tornillos para mantener sujeta esta pieza con el cuerpo principal como se mencionó anteriormente. También posee los agujeros necesarios para poder ubicar los componentes que formaran parte del pad direccional y los botones de acción que se utilizarán para controlar el robot. Por último se tiene el compartimento para la batería que será una pieza con geometría simple que estará sujeta a la parte baja del cuerpo principal, este compartimento también será utilizado para poder ubicar el módulo NRF24L01 de manera que tiene un agujero por el cual saldrá la antena y de esta manera no se obstruirá la señal de ninguna manera.

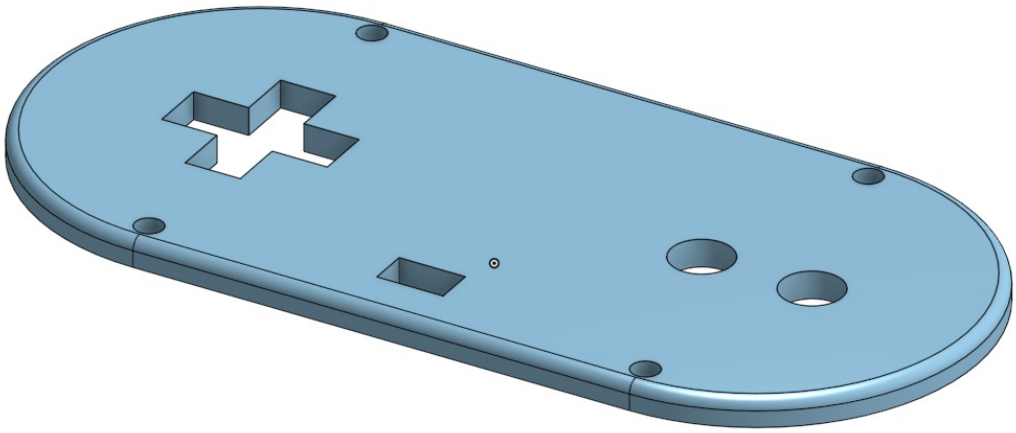


Figura 10: Modelo CAD de la tapadera diseñada para el control remoto

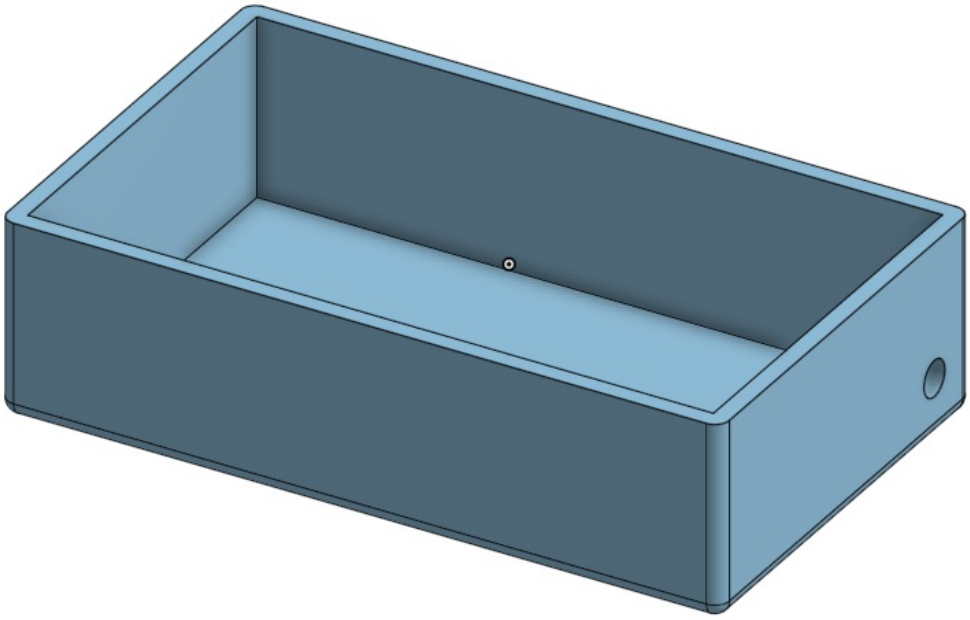


Figura 11: Modelo CAD del compartimento de almacenamiento de baterías diseñado para el control remoto

Al terminar el diseño CAD del control remoto se procedió a utilizar una impresora 3D equipada con plástico PLA para poder fabricar la carcasa. Con la carcasa completamente fabricada se realizaron pruebas de ensamble para verificar que todos los componentes encajaran correctamente.

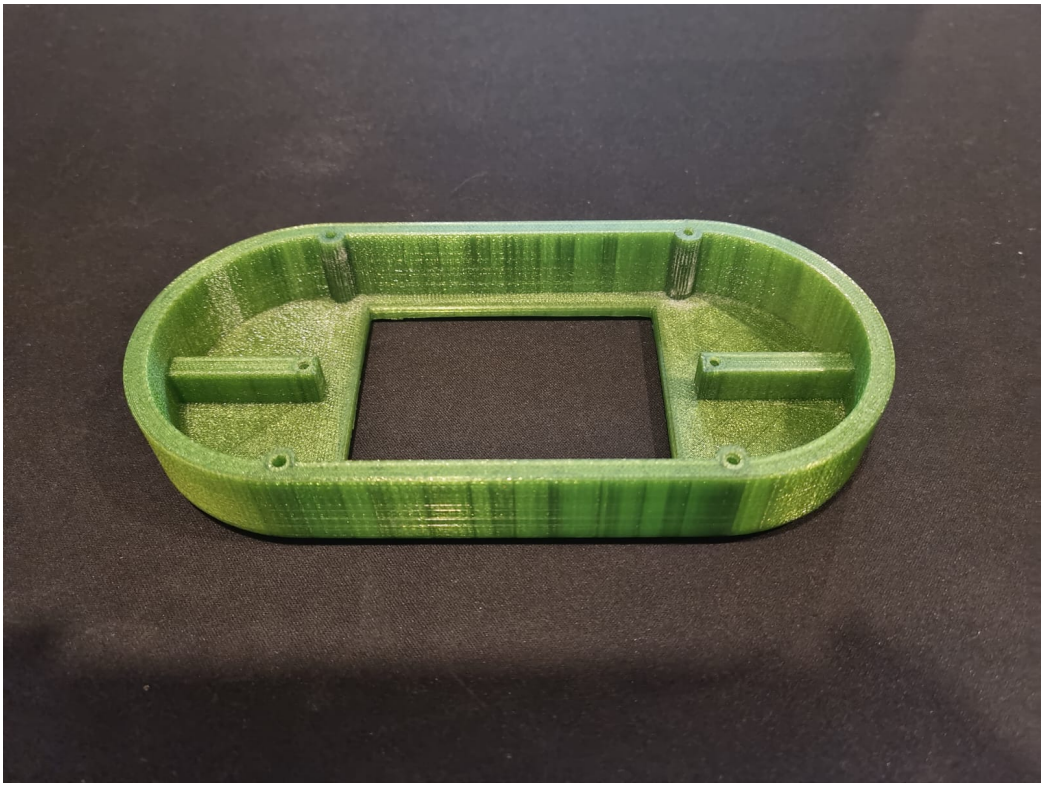


Figura 12: Cuerpo principal fabricado para el control remoto

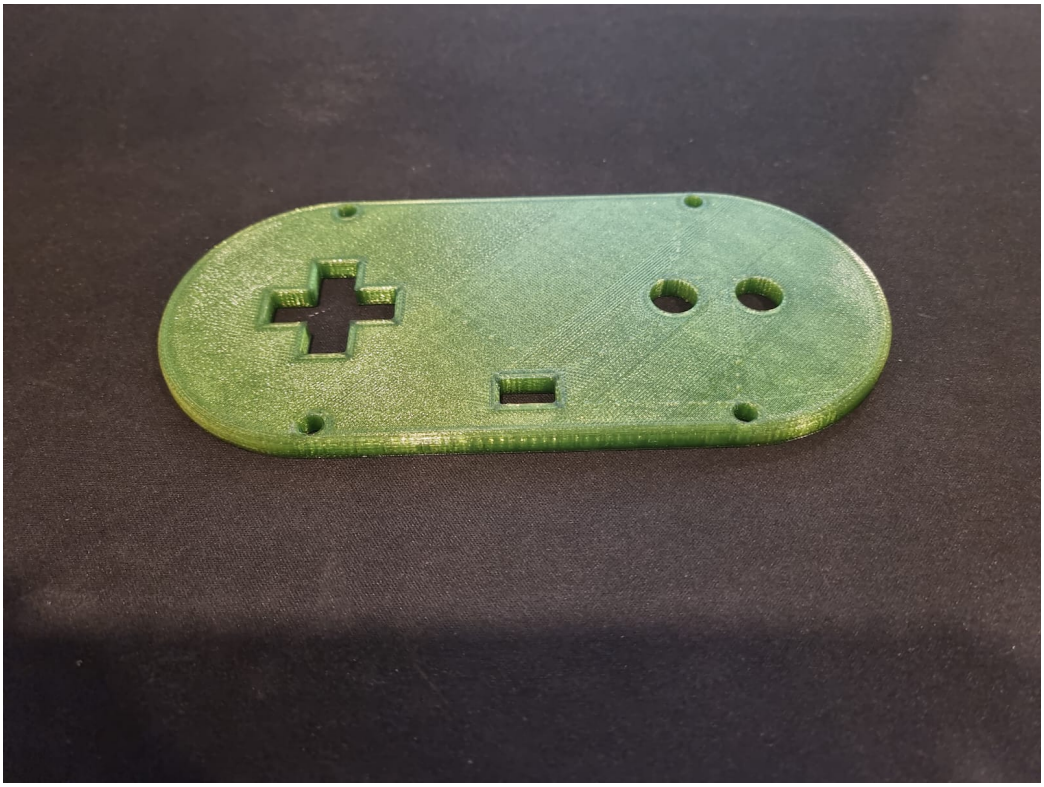


Figura 13: Tapadera fabricada para el control remoto

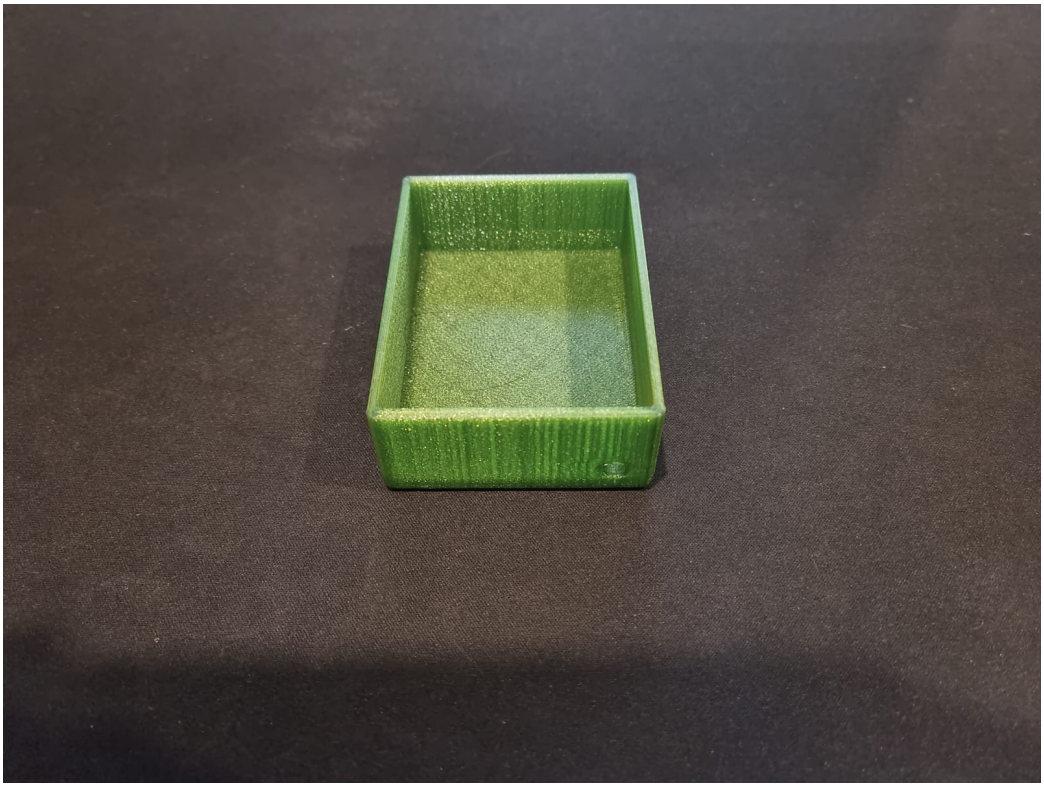


Figura 14: Base y almacenamiento de componentes fabricada para el control remoto

Lo siguiente fue comenzar a realizar los esquematicos y diseño de PCB para el circuito principal del control remoto. El diseño del esquemático se comenzó creando los componentes a utilizar dentro del circuito por lo que se creó una librería especifica. La librería contendría todos los componentes que en este caso serían resistencias, pulsadores, borneras, porta chips, headers hembra, switches y reguladores de voltaje. Para cada uno de estos componentes se creó el símbolo correspondiente dentro de la librería creada anteriormente, una vez se terminaron de crear todos los símbolos se creó otra librería que contendría lo que se conoce como un footprint para cada componente, estos footprints permitirán tener una vista más especifica de serán los componentes para utilizar una vez estén situados dentro de la PCB. En este paso se hizo uso de dos sitios web para poder obtener los footprints necesarios para cada uno de los componentes.

El primer sitio web utilizado fue DigiKey el cual es un sitio especializado en la venta de componentes electrónicos, este sitio se utilizó principalmente ya que ofrecen varios componentes electrónicos estándar junto con sus respectivos footprints creados por los fabricantes para este tipo de usos por lo que se tendrá la certeza que los footprints utilizados serán los correctos. El segundo sitio web utilizado fue SnapEDA el cual se especializa en proveer modelos 3D de varios componentes electrónicos y aunque estos no son totalmente necesarios para la fabricación de la PCB estos nos permitirán visualizar de mejor manera como se vera el producto final una vez se ensamble por completo la PCB. Haciendo uso de SnapEDA se obtuvo el modelo 3D de los componentes que serían utilizados en la placa final y estos se agregaron a cada uno de los footprints seleccionados de DigiKey de manera que dentro de la librería creada se tuvieran los footprints de cada uno de los componentes con su respectivo

modelo 3D.

La creación de ambas librerías antes de comenzar a crear los respectivos diseños para el esquemático y la PCB permitirán un mejor flujo de trabajo ya que todo estará listo para utilizar y con esto dicho se procedió a comenzar con el diseño del esquemático del circuito. El circuito por utilizar está compuesto por 6 pulsadores de los cuales 4 serán utilizados para el pad direccional y 2 para los botones de acción, 6 resistencias que serán utilizadas para cada uno de los pulsadores, 1 switch SPDT (Single Pole Double Throw por sus siglas en inglés) que permitirá encender el control remoto, un microcontrolador ATMega 328PU que permitirá realizar la correcta programación del control remoto para cada una de las funciones que este debe realizar, por último se agrego un regulador de voltaje LM7805 para poder asegurar que el voltaje que se utilizara en el circuito es de 5 volteos.

Adicional se deben agregar algunos componentes extras para que el microcontrolador incluido pueda funcionar de manera correcta que en este caso es un cristal oscilador de 16MHz que debe estar conectado a los pones 9 y 10, dos capacitores cerámicos de 22pF conectando uno de estos al pin 9 y tierra mientras que el otro debe estar conectado al pin 10 del microcontrolador y de igual manera a tierra, una resistencia de 10k Ohm debe conectarse a voltaje y al pin numero 1 y por último se debe agregar un capacitor electrolítico de 10uF de manera que la pata positiva estará conectada a voltaje y la pata negativa estará conectada a tierra.

El siguiente paso por realizar es crear el diseño de la PCB a fabricar y para esto se comenzó exportando el esquemático finalizado y ya que anteriormente se le agrego su respectivo footprint y modelo 3D a cada uno de los componentes utilizados esto permitirá visualizar de mejor manera el posicionamiento de los componentes dentro de la placa. Para poder darle la forma correcta a la placa se utilizó el cuerpo principal del control remoto creado anteriormente y se trazó el contorno del interior de esta pieza dejando un margen de 2mm para que la placa pudiera ser colocada sin ningún problema, luego se procedió a alinear de manera correcta las piezas del pad direccional y los botones de acción para que al momento de colocarlos en la placa estos quedaran posicionados de manera correcta con la tapadera del control. Al terminar de realizar el contorno se procedió a exportar este como un archivo DXF y así poder importarlo al creador de PCB de Altium Designer y poder darle la forma deseada a la placa.

Una vez se tenga la forma de la placa deseada y todos los componentes importados se procedió a organizar todos los componentes dentro de la placa para asegurar que estos estén en los lugares designados al momento de crear el contorno del cuerpo principal y también para asegurar que no se creen tantas complicaciones al momento de crear las conexiones entre componentes. Ya finalizada la organización de los componentes se realizará un paso extra para asegurar que no se creen problemas al momento de conectar todo y esto es establecer las reglas a utilizar dentro del diseño de la placa.

Se comenzó por establecer las reglas del grosor que tendrá cada uno de los tracks para las conexiones, estos grosores se dividen en mínimos, preferidos y máximos de manera que a estos se les asignaron tamaños de 0.5mm, 1mm y 1.5mm respectivamente, pero en esta placa únicamente se utilizaron tracks de 1mm de tamaño ya que con este tamaño se podrá soldar sin ninguna preocupación de dañar alguna parte al momento de utilizar un cautín para soldar.

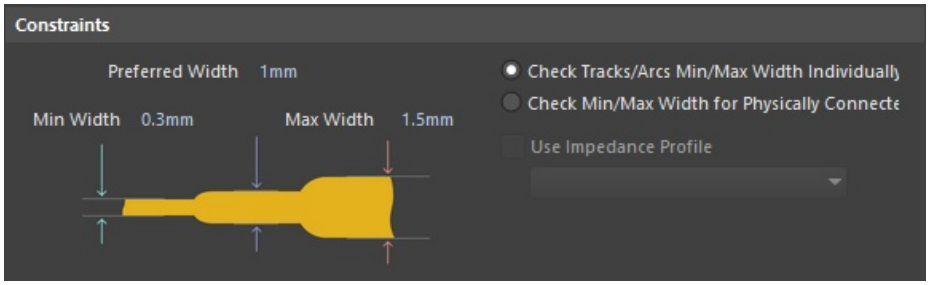


Figura 15: Regla establecida para los grosores mínimos, preferidos y máximos a utilizar en Altium Designer

La siguiente regla que se estableció fue la de distancia mínima entre conexiones (esto incluye vías, tracks y pads de soldadura) ya que es importante asegurar que estas no se estarán tocando entre si para poder evitar un contacto no deseado y también para reducir el riesgo de dañar una conexión al momento de soldar ya que si se encuentran muy cercanas entre sí es posible dañar varias conexiones. En este caso la distancia mínima que se colocó para conexiones es de 0.3mm, se utilizó esta distancia ya que permite juntar algunas conexiones sin que estas provoquen un problema, pero también que permitan aprovechar el espacio lo más posible en caso sea necesario colocar varias conexiones en un espacio reducido.

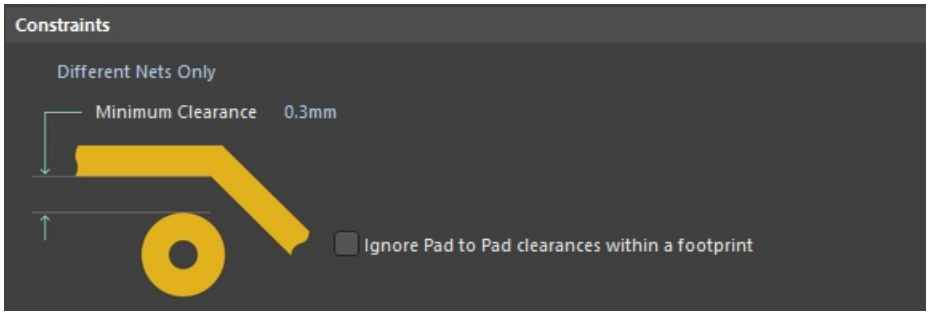


Figura 16: Regla establecida para la distancia mínima entre conexiones dentro de Altium Designer

Con las reglas principales establecidas se comenzó a realizar la conexión de los diferentes componentes, por la manera en que está diseñado el control remoto será necesario utilizar las dos caras de la placa por lo que en la cara superior se colocarán los botones para el pad direccional y los botones de acción, el microcontrolador ATMega 328P-U, los componentes extras para el funcionamiento del microcontrolador y el switch para el encendido y apagado del control. Mientras tanto en la cara inferior se colocarán el resto de los componentes que son las borneras para la conexión de la batería al circuito, el regulador de voltaje LM7805, los headers hembra para conectar el módulo NRF24L01 y el último componente extra para el funcionamiento del microcontrolador.

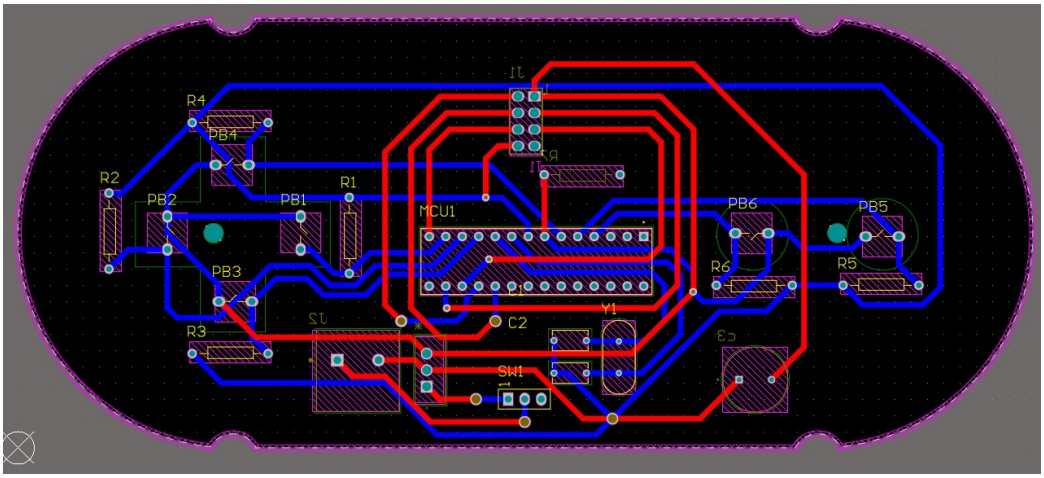


Figura 17: Conexiones realizadas en la cara superior de la placa representadas con el color rojo

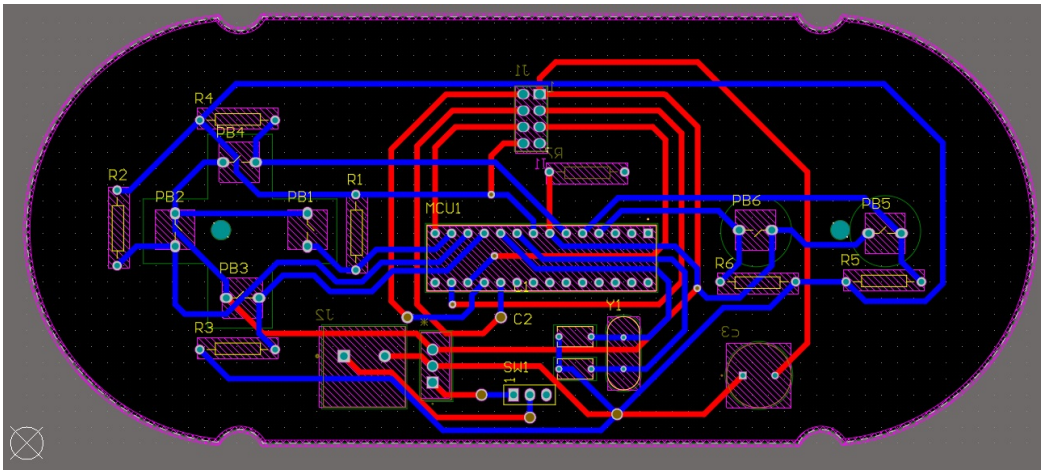


Figura 18: Conexiones realizadas en la cara inferior de la placa representadas con el color azul

Luego de terminar las conexiones de todos los componentes se utilizó la revisión de reglas que tiene Altium Designer para poder verificar que no hubiera errores en el diseño o las conexiones realizadas y una vez no se tengan errores de diseño se crearán los archivos de fabricación de la placa que son los archivos Gerber y los archivos NC Drill. Para los archivos Gerber se seleccionó que estos estuvieran en milímetros con un formato de 4:2 que tiene una resolución de 0.01mm, a pesar de que este formato es el que tiene la resolución as baja se opto por utilizarlo ya que la placa diseñada no es lo suficientemente complicada como para necesitar una precisión de fabricación muy alta. Por último se deben seleccionar todas las capas utilizadas en el diseño y se deben crear los archivos. Para la creación de los archivos NC Drill de igual manera se utilizarán milímetros y el mismo formato utilizado para los archivos Gerber (4:2) debido a que tampoco es necesario tener una resolución demasiado alta para esta parte de la fabricación ya que no se están utilizando agujeros muy pequeños o precisos. Una vez generados los archivos necesarios ya es posible enviarlos para que se inicie la fabricación de la placa utilizando la máquina CNC disponible.

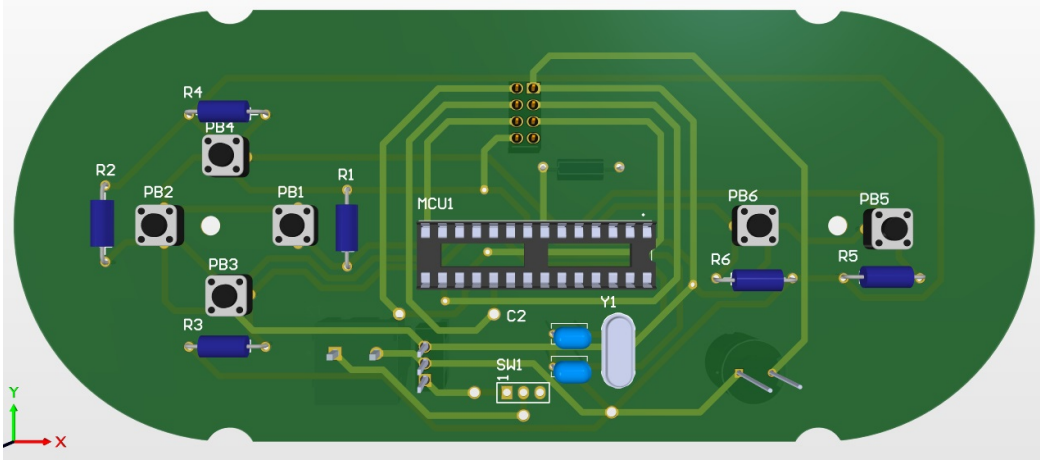


Figura 19: Cara superior de la placa terminada con todos los componentes colocados en sus repectivos lugares

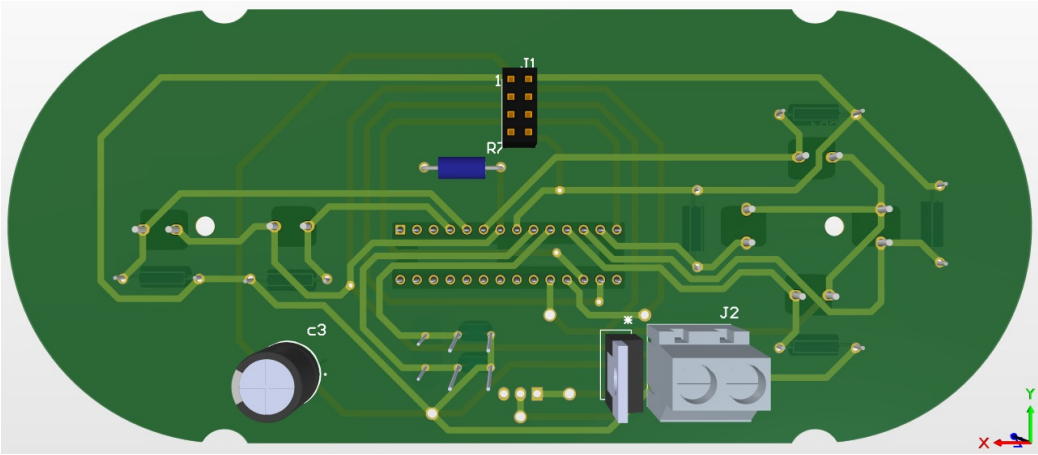


Figura 20: Cara inferior de la placa terminada con todos los componentes colocados en sus repectivos lugares

Luego de recibir la placa final se procedió a finalmente soldar cada uno de los componentes en su respectiva cara y asegurarse que todos los componentes tuvieran la continuidad correcta y para esto último se utilizó un multímetro con el cual se verificó cada uno de los tracks, vías y pads de la placa para asegurar que sí estaban conectados y soldados correctamente.

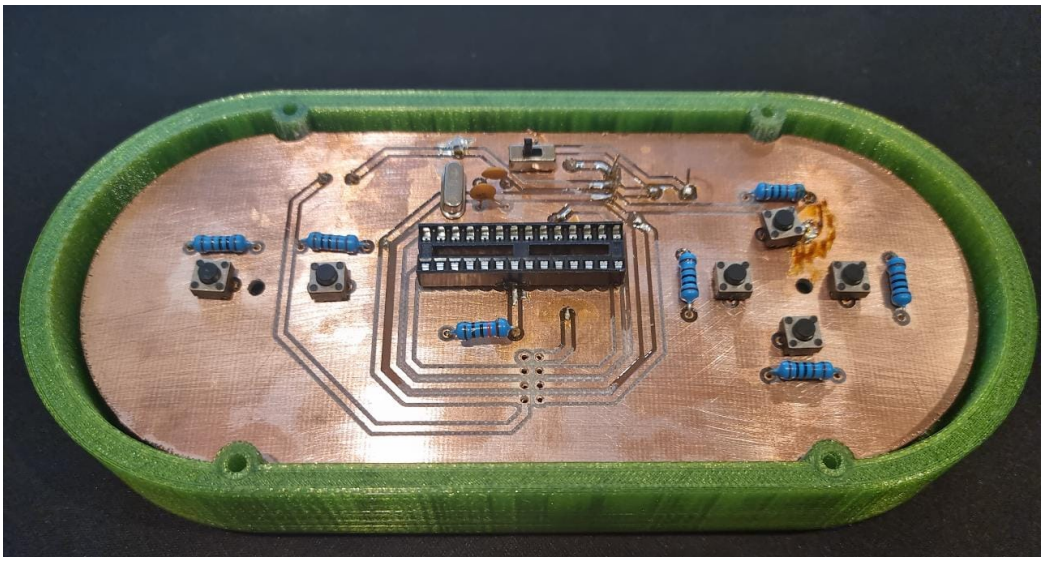


Figura 21: Cara superior de la placa fabricada y soldada

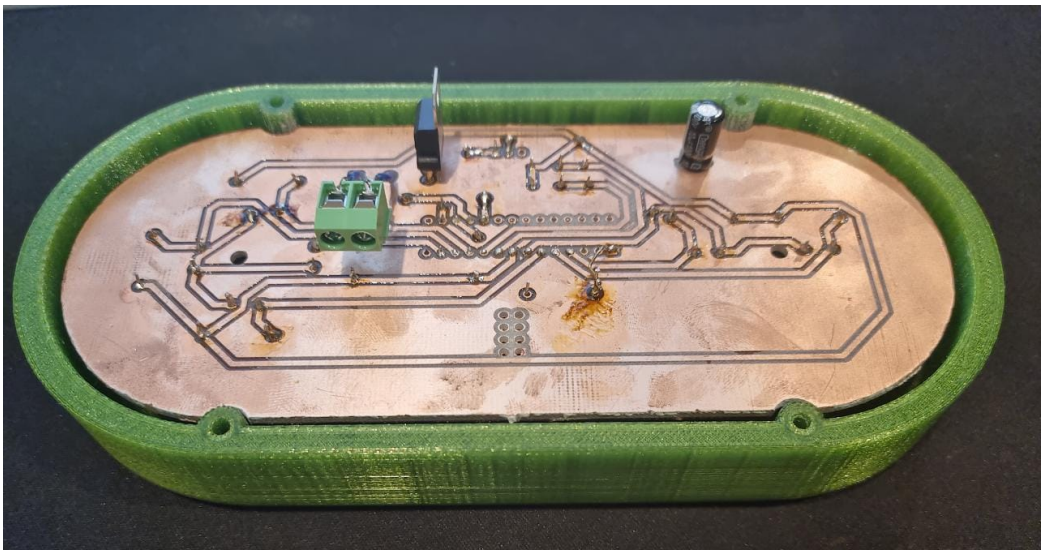


Figura 22: Cara inferior de la placa fabricada y soldada

Al finalizar la soldadura de la placa se procedió a realizar pruebas de ensamblaje para el control remoto y durante estas pruebas se pudo determinar que el tamaño del compartimento en el cual estarían ubicadas las baterías y el módulo NRF24L01 no era lo suficientemente grande como para poder almacenar de una manera segura ambos componentes por lo que se hizo un pequeño rediseño para aumentar el tamaño y para poder ubicar de mejor manera los componentes.

Figura 23: Nuevo compartimento para baterías y módulo de radiofrecuencia

Para poder verificar que los grados de requisito asignados a los botones fueron los correctos se realizo una verificación de fuerzas. En esta verificación se utilizó una balanza sobre la cual se colocó uno de los botones que se utilizaría y se aplico fuerza hasta que esta estuviera presionado. Luego se registró el dato mostrado en la balanza el cual se encontraba en gramos. Esta prueba se realizó 30 veces para poder obtener luego un promedio del peso mostrado en la balanza para poder hacer una conversión que permitiría obtener lbs fuerza y Newtons. El resultado obtenido fue de 2.57 N de manera que en la práctica los botones tenían un grado de requisito despreciable o con código 0. Por lo que el grado de requisito asignado que en este caso fue Medio con código 2 es bueno ya que se asegura que los botones no fallarán debido a la fuerza aplicada.

Figura 24: Prueba de fuerza realizada a los botones del control remoto

Seguido de esto se comenzaron a hacer pruebas de funcionamiento para el transmisor y receptor del control remoto, estas pruebas fueron realizadas con el control remoto ensamblado para el transmisor y para el receptor se utilizó la pequeña plataforma montada con LEDS y un Arduino UNO. Al momento de comenzar con las pruebas se verificó que los módulos funcionarán de manera correcta utilizando un pequeño programa de diagnóstico el cual mostraría si los módulos están funcionando de manera correcta haciendo uso de registros que tendrán un valor en hexadecimal para demostrar que esa función específica del módulo está trabajando de manera correcta, si el valor de este registro es 0x00 esto significa que el módulo no está funcionando de manera correcta.

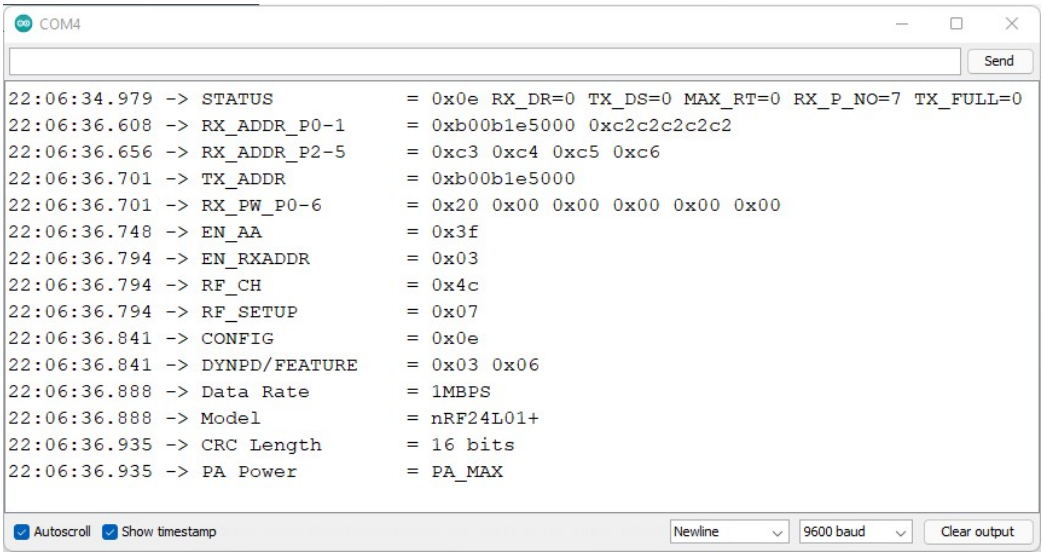


Figura 25: Diagnóstico de módulo NRF24L01 mostrando buen funcionamiento

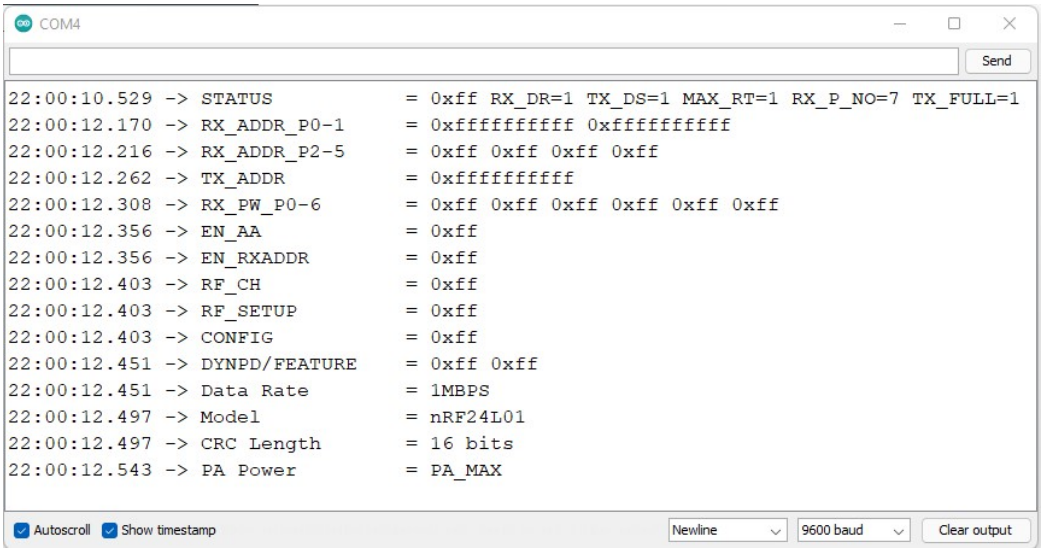


Figura 26: Diagnóstico de módulo NRF24L01 mostrando mal funcionamiento

Después de asegurar el correcto funcionamiento de los módulos se procede a realizar la prueba de comunicación para poder determinar si el control funciona como transmisor de manera correcta, al momento de comenzar con la prueba y encender el control se logró enviar correctamente un comando de movimiento que se viera desplegado en la plataforma del receptor, luego de esto el módulo NRF24L01 tuvo una falla y se dañó permanentemente. Después de la prueba fallida de comunicación se hizo una revisión al control remoto para poder determinar qué fue lo que provocó el fallo en el módulo de radiofrecuencia, en esta revisión de logro encontrar un problema con el regulador de voltaje implementado en el control remoto ya que uno de los pads sobre los cuales se encontraba soldado estaba despegado lo cual creó un mal contacto en el circuito. Este problema provocó que el módulo de radiofrecuencia recibiera un voltaje mucho más alto a su voltaje de uso máximo provocando

que este se quemará.

7.4. Iteración No.2

La segunda iteración del control remoto se comenzó haciendo ajustes al esquemático del circuito planteado inicialmente para poder tomar algunas medidas para proteger mejor el módulo de radiofrecuencia y asegurar que otros componentes operen de manera correcta dentro del control remoto. Estos ajustes se comenzaron agregando un regulador de voltaje de 3.3 V al circuito para que este pudiera alimentar directamente al módulo de radiofrecuencia, esto se implemento ya que el voltaje que emitirá el regulador de voltaje será exclusivo para el uso del módulo por lo que si este fallara no afectaría a otros componentes o al ATMega328P implementado.

Luego de esto se ajustaron los valores de resistencia utilizados para los botones del control remoto, ya que es posible variar el consumo de energía del módulo de radio frecuencia se utilizó un valor de resistencia de 4.7K ohm para poder estar en un punto medio entre 1k ohm y 10k ohm que permita asegurar que no ocurrirá ya sea un pulldown fuerte el cual dejaría que una gran cantidad de corriente pasara directamente al dispositivo cuando el pulsador se activa lo cual podría provocar un sobrecalentamiento en el módulo o un pulldown débil en el cual el valor de resistencia es tan alto que el valor de voltaje recibido al momento de activar el pulsador es demasiado bajo como para que funcione el dispositivo.

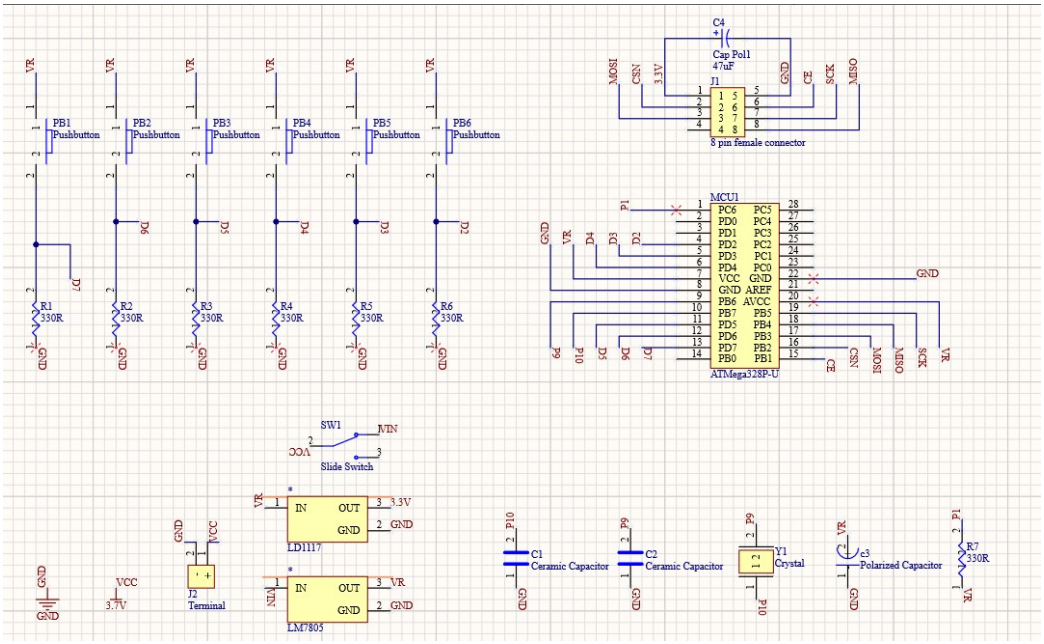


Figura 27: Esquemático con nuevos ajustes para iteración No.2

Para poder asegurar que estos nuevos ajustes al esquemático del control remoto funcionaran correctamente se volvió a construir el circuito en una breadbord, pero esta vez utilizando directamente el ATMega328p en vez de un Arduino UNO para poder cargar el código del transmisor.

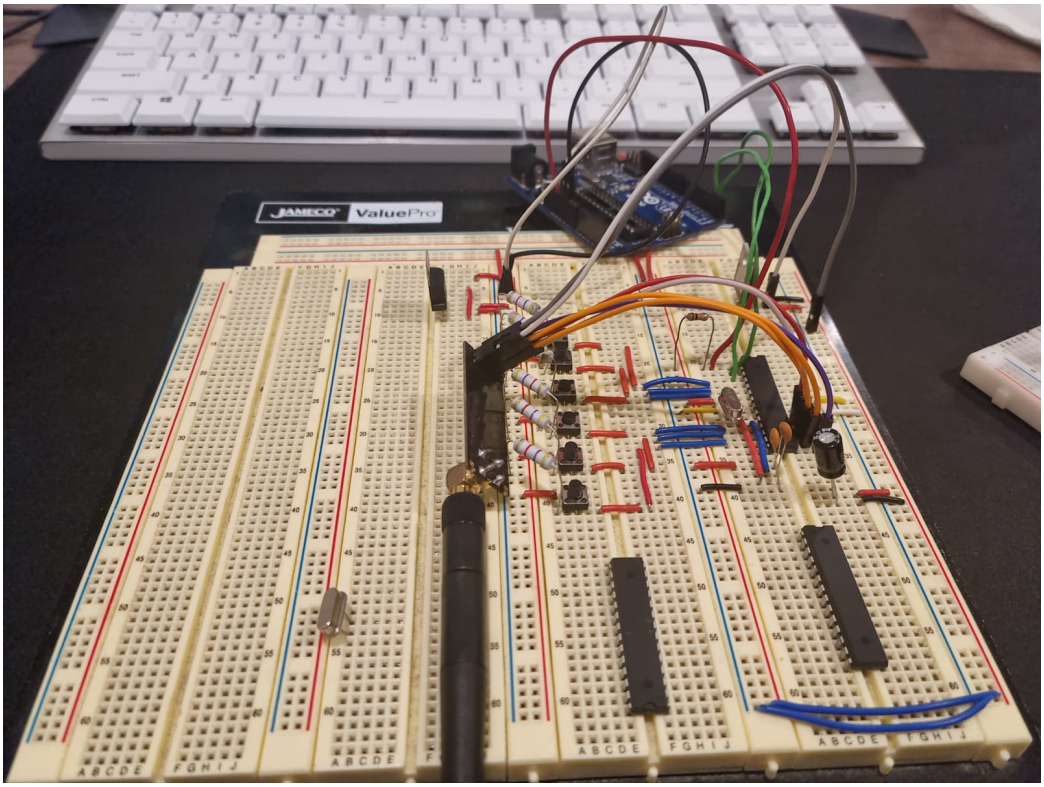


Figura 28: Circuito montado en breadboard para pruebas de ATMega328P de manera independiente

Una vez complemente armado el circuito del control remoto se procedió a realizar pruebas de funcionamiento, pero desafortunadamente estas no mostraban buenos resultados ya que no era posible entablar comunicación entre el receptor y el emisor por lo que se comenzaron a realizar distintas pruebas tanto al circuito como a cada uno de los componentes principales para asegurar que estos estaban trabajando de manera correcta

Primero se verificó que el ATMega328P que se estaba utilizando trabajará de manera correcta y para esto se retiró del circuito del control remoto y se implementó en un circuito construido específicamente para probar códigos de ejemplo de Arduino y así verificar que este chip estuviera trabajando como se espera por lo que se utilizó el código de ejemplo "Blinky" proporcionado en el ide de Arduino, pero este no funcionó al momento de cargarlo al chip de manera independiente así que se utilizó un chip diferente que ya había sido utilizado en un proyecto y con este funcionó de manera correcta el código por lo que llegó a la conclusión que el chip que se utilizó originalmente no poseía un bootloader para poder cargar los códigos desde el ide de Arduino.

Al encontrar un ATMega328P que tuviera el bootloader se procedió a verificar que los módulos de radiofrecuencia funcionaran correctamente ya que se comenzaron a utilizar diferentes debido a que uno de los que se utilizó originalmente se fue dañado durante las pruebas de la primera iteración del control remoto. Para poder comenzar con la verificación se comenzó por utilizar un código simple de transmisión de datos que enviará únicamente la palabra "Hello World" desde el transmisor y que está fuera recibida y desplegada en el monitor serial del receptor. Inicialmente esta prueba no funcionaba por lo que se utilizó el

mismo código de diagnóstico utilizado anteriormente para poder verificar que los módulos funcionaban de manera correcta y en los resultados del diagnóstico no se obtuvo ningún resultado anormal por lo que se siguieron haciendo pruebas utilizando distintos Arduinos y cambiando de posición los módulos. Luego de hacer distintas pruebas se encontró la razón por la cual el código no funcionaba correctamente y esto era debido a que uno de los módulos estaba ligeramente dañado por lo que únicamente era capaz de recibir datos, pero no de enviarlos por lo que se cambió de posición los módulos y se volvió a probar el código de ejemplo para la comunicación, pero esta vez sí se obtuvo un buen resultado y todo se desplegaba de manera correcta en el monitor serial del receptor.

Ya que los componentes principales para el funcionamiento del control remoto funcionan correctamente se procedió a realizar las pruebas de comunicación entre el transmisor y el receptor utilizando el circuito montado en breadboard. Debido a las pruebas que se realizan con anterioridad a cada uno de los componentes fue mucho mas fácil realizar la prueba de comunicación y en esta se obtuvieron los resultados esperados ya que el transmisor y el receptor se comunicaban de manera correcta y era posible enviar los comandos necesarios para el funcionamiento del control. Para poder verificar que este código y control remoto funcionaria correctamente en el robot principal se implementó en un pequeño robot de prueba que poseía las mismas características que el robot original, el código en este robot se cambio para que al momento de recibir los comandos de movimiento en lugar de encender luces LED se activarán los drivers de los motores utilizados y que se pudiera desplegar en el monitor serial cada uno de los movimientos que se realizaban.

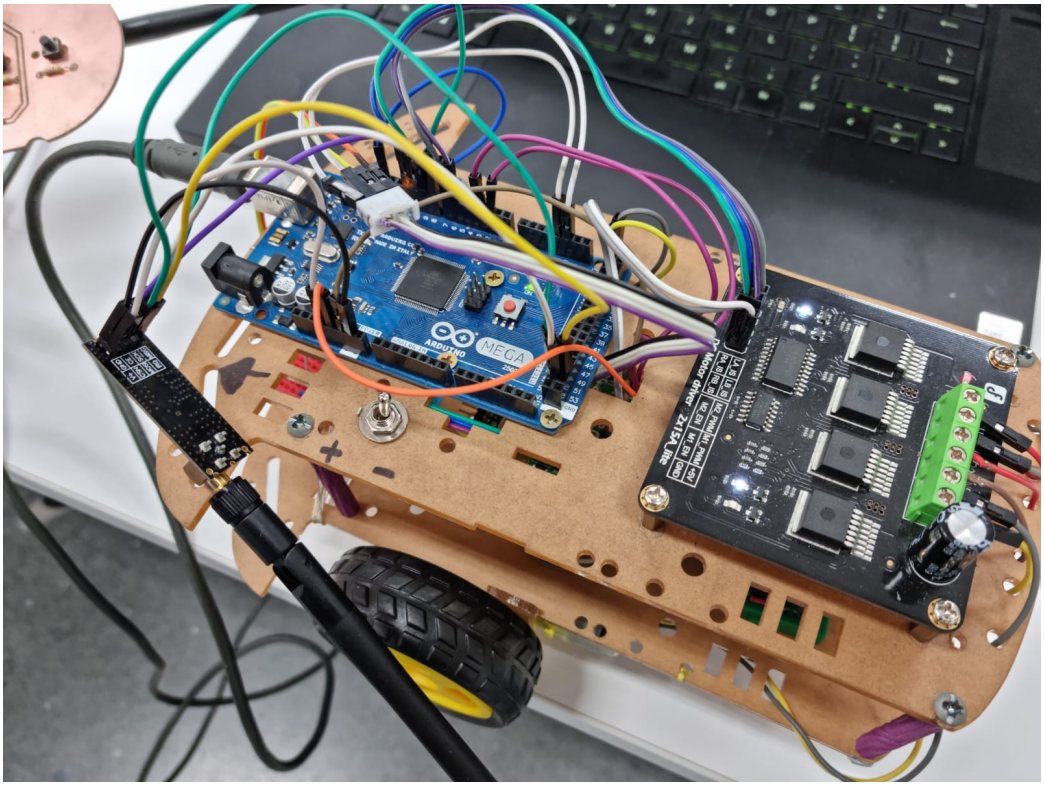


Figura 29: Conexión de módulo de radiofrecuencia a robot

Una vez asegurado el correcto funcionamiento del control remoto utilizando el ATMega328P de manera independiente se procedió a modificar el diseño de la PCB según los ajustes realizados al esquemático y también se realizaron algunos cambios en el diseño de los tracks ya que algunos de estos no estaban colocados de manera óptima en el prototipo anterior. Al terminar los ajustes necesarios para la adición de componentes y el cambio de los tracks se crearon los archivos de fabricación y se comenzó con la fabricación de la placa.

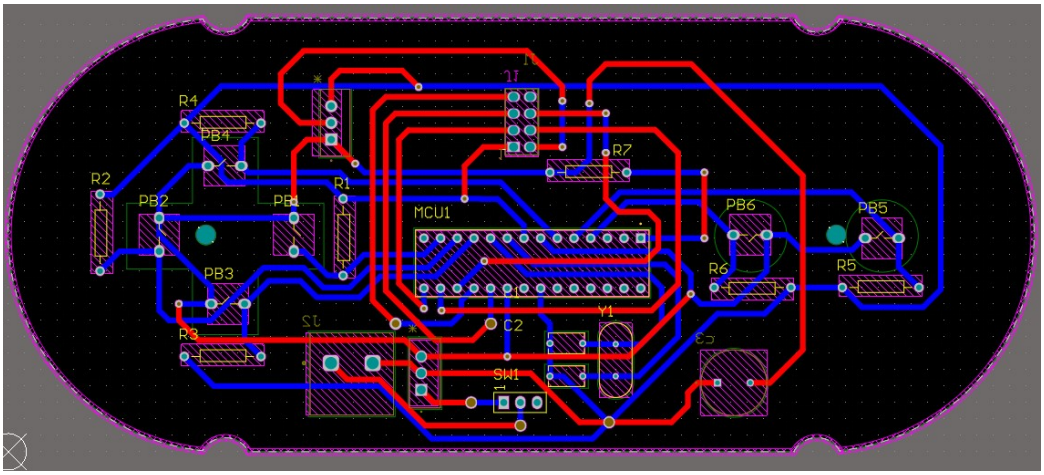


Figura 30: Cara superior de PCB para iteración No.2 del control remoto

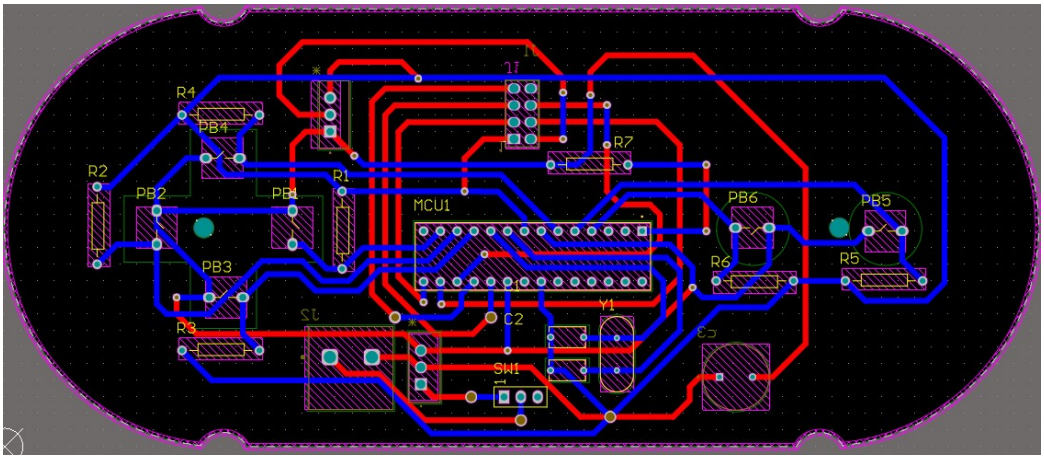


Figura 31: Cara inferior de PCB para iteración No.2 del control remoto

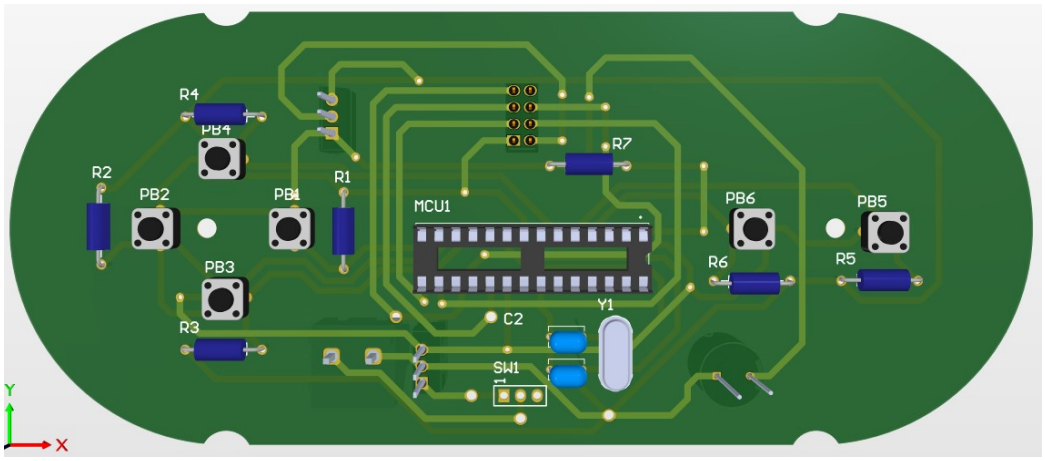


Figura 32: Cara superior de modelo 3D de PCB para iteración No.2 del control remoto

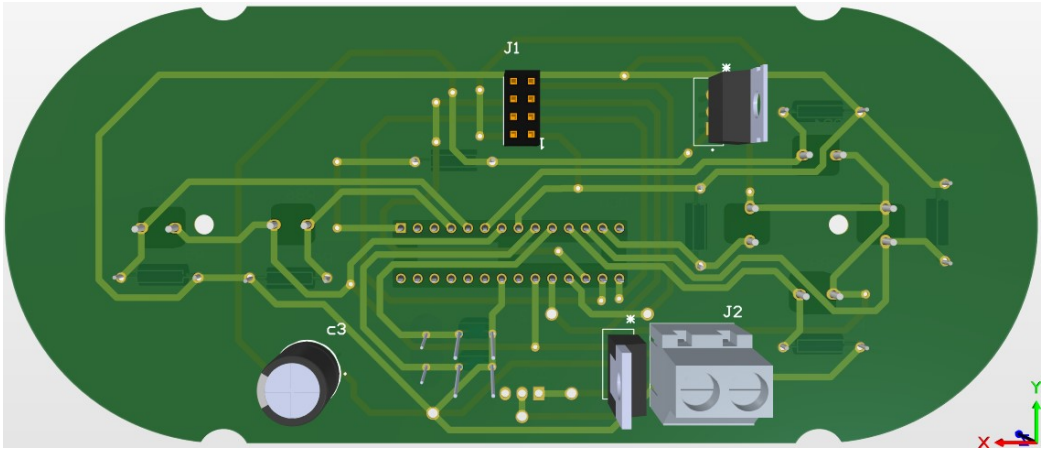


Figura 33: Cara inferior de modelo 3D de PCB para iteración No.2 del control remoto

Al momento de recibir la placa terminada se revisaron las continuidades de las tracks para estar seguros de que estos están colocados correctamente y que es seguro proceder a soldar por lo que este fue el siguiente paso a realizar y para este se siguieron los mismos pasos utilizados en la primera iteración para asegurar que todo estaba soldado correctamente en la placa.

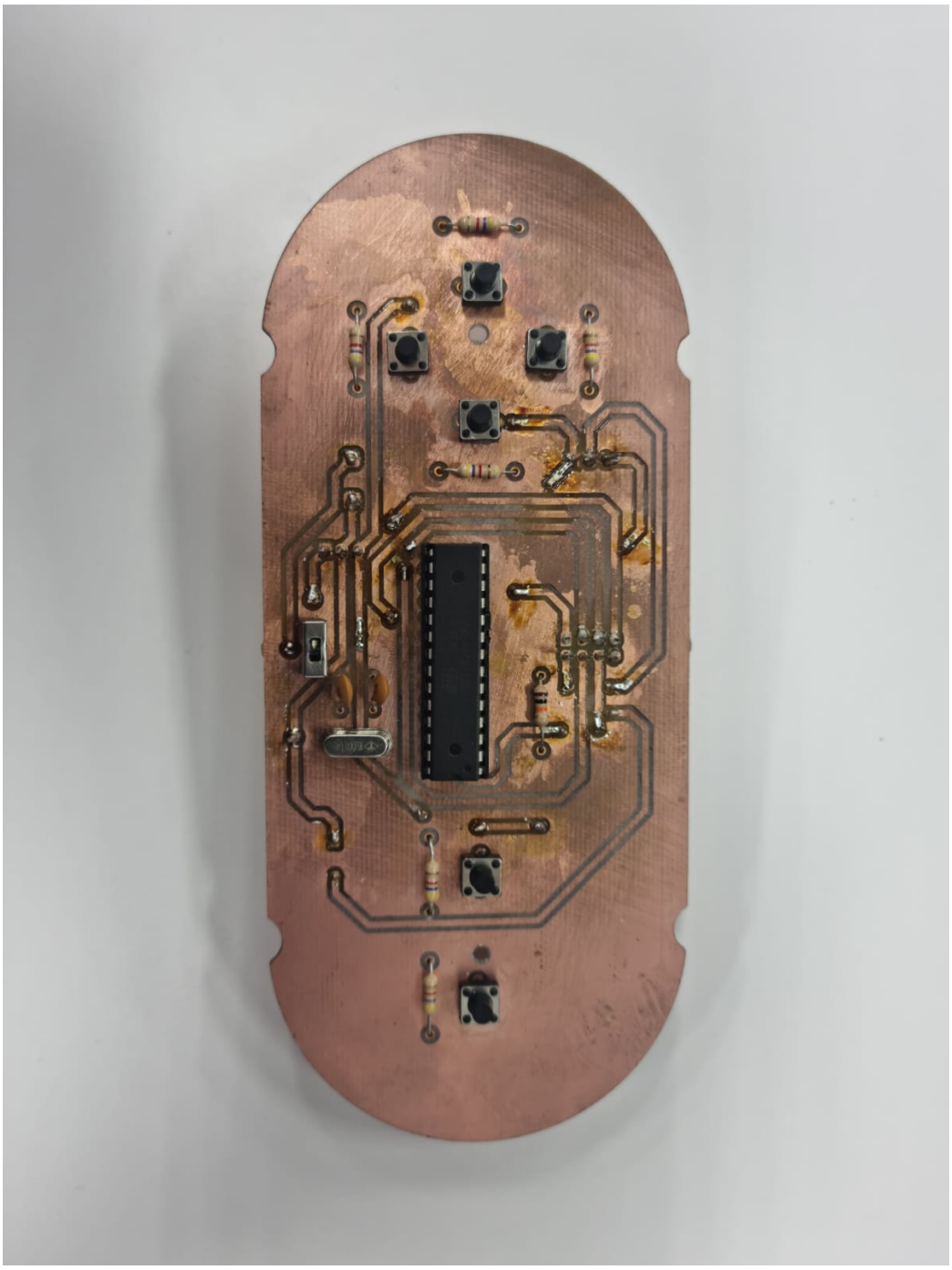


Figura 34: Cara superior de iteración No.2 fabricada y soldada

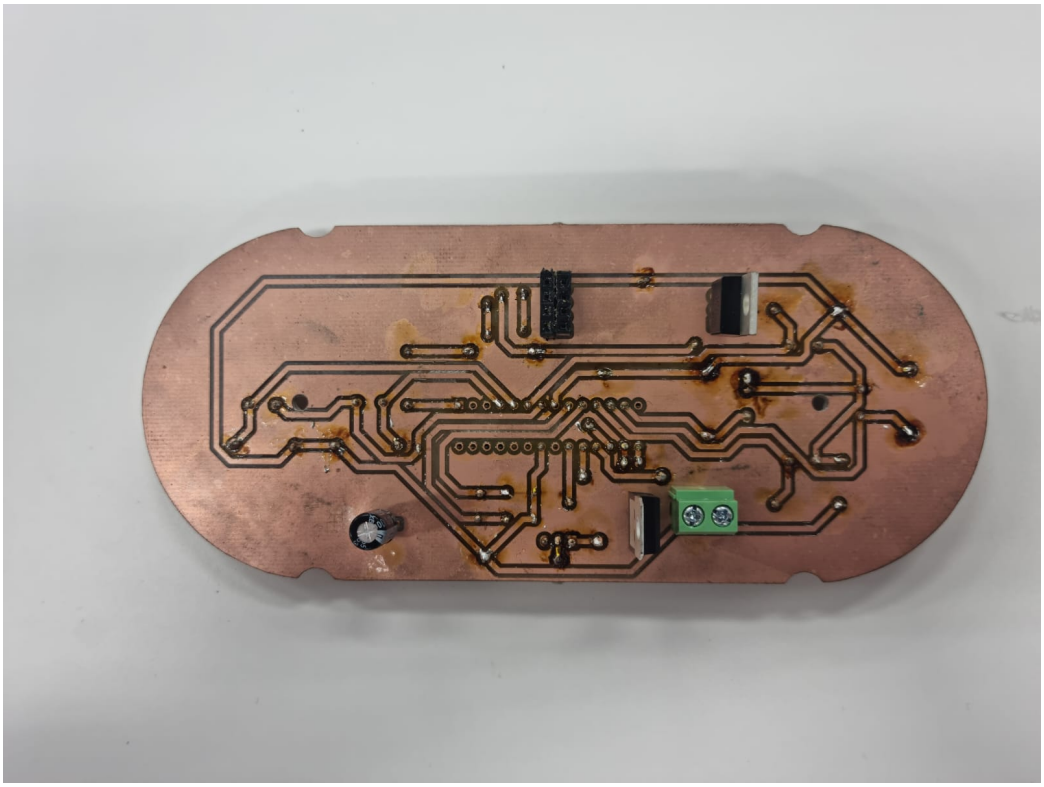


Figura 35: Cara inferior de iteración No.2 fabricada y soldada

Las primeras pruebas de funcionamiento que se realizaron fueron con la plataforma de LEDS y Arduino UNO utilizada anteriormente. Al encender el control remoto se intentaron enviar comandos de movimiento a la plataforma y estos fueron recibidos exitosamente. Se cambió el código principal para poder desplegar dentro del monitor serial los cambios de velocidad de que se realizaban y que también se desplegara una secuencia de bits indicadores para cada una de las velocidades que se tienen.

Figura 36: Monitor serial desplegando velocidad lenta en la prueba

Figura 37: Monitor serial desplegando velocidad rápida en la prueba

Figura 38: Monitor serial desplegando velocidad súper rápida en la prueba

Ya que los resultados obtenidos en la primera prueba fueron favorables se comenzaron a realizar más pruebas de funcionamiento. La segunda prueba realizada consistió en poder enviar los comandos de movimiento correctos al robot y que estos fueran desplegados en el monitor serial. Al realizar estas pruebas se observó que el orden de los botones había cambiado respecto al orden original por lo que se realizaron los ajustes en el código para poder obtener correctamente los comandos asociados a cada botón.

A	B	C
12:24:59.511 -> Velocidad: 100	12:25:01.328 -> Velocidad: 160	12:24:57.690 -> Retrocediendo
12:24:59.651 -> Retrocediendo	12:25:01.468 -> Retrocediendo	12:24:57.690 -> Velocidad: 220
12:24:59.651 -> Velocidad: 100	12:25:01.468 -> Velocidad: 160	12:24:57.830 -> Retrocediendo
12:24:59.789 -> Retrocediendo	12:25:01.609 -> Retrocediendo	12:24:57.830 -> Velocidad: 220
12:24:59.789 -> Velocidad: 100	12:25:01.609 -> Velocidad: 160	12:24:57.969 -> Retrocediendo
12:24:59.928 -> Retrocediendo	12:25:01.703 -> Retrocediendo	12:24:57.969 -> Velocidad: 220
12:24:59.928 -> Velocidad: 100	12:25:01.750 -> Velocidad: 160	12:24:58.109 -> Retrocediendo
12:25:00.023 -> Retrocediendo	12:25:01.843 -> Retrocediendo	12:24:58.109 -> Velocidad: 220
12:25:00.023 -> Velocidad: 100	12:25:01.843 -> Velocidad: 160	12:24:58.204 -> Retrocediendo
12:25:00.162 -> Retrocediendo	12:25:01.980 -> Retrocediendo	12:24:58.251 -> Velocidad: 220
12:25:00.162 -> Velocidad: 100	12:25:01.980 -> Velocidad: 160	12:24:58.343 -> Retrocediendo

Figura 39: Monitor serial desplegando movimiento de retroceso a velocidad mínima (Figura A), movimiento de retroceso a velocidad media (Figura B) y movimiento de retroceso velocidad máxima (Figura C)

A	B	C
12:25:06.299 -> Avanzando	12:25:08.348 -> Avanzando	12:25:11.646 -> Avanzando
12:25:06.299 -> Velocidad: 100	12:25:08.348 -> Velocidad: 160	12:25:11.646 -> Velocidad: 220
12:25:06.391 -> Avanzando	12:25:08.489 -> Avanzando	12:25:11.738 -> Avanzando
12:25:06.391 -> Velocidad: 100	12:25:08.489 -> Velocidad: 160	12:25:11.738 -> Velocidad: 220
12:25:06.533 -> Avanzando	12:25:08.628 -> Avanzando	12:25:11.878 -> Avanzando
12:25:06.533 -> Velocidad: 100	12:25:08.628 -> Velocidad: 160	12:25:11.878 -> Velocidad: 220
12:25:06.673 -> Avanzando	12:25:08.765 -> Avanzando	12:25:12.016 -> Avanzando
12:25:06.673 -> Velocidad: 100	12:25:08.765 -> Velocidad: 160	12:25:12.016 -> Velocidad: 220
12:25:06.813 -> Avanzando	12:25:08.904 -> Avanzando	12:25:12.158 -> Avanzando
12:25:06.813 -> Velocidad: 100	12:25:08.904 -> Velocidad: 160	12:25:12.158 -> Velocidad: 220

Figura 40: Monitor serial desplegando movimiento de avance a velocidad mínima (Figura A), movimiento de avance a velocidad media (Figura B) y movimiento de avance velocidad máxima (Figura C)

A B

```
12:25:16.803 -> Cruzando derecha 12:25:23.183 -> Cruzando izquierda
12:25:16.803 -> Velocidad: 80    12:25:23.183 -> Velocidad: 80
12:25:16.944 -> Cruzando derecha 12:25:23.323 -> Cruzando izquierda
12:25:16.944 -> Velocidad: 80    12:25:23.323 -> Velocidad: 80
12:25:17.082 -> Cruzando derecha 12:25:23.418 -> Cruzando izquierda
12:25:17.082 -> Velocidad: 80    12:25:23.418 -> Velocidad: 80
12:25:17.223 -> Cruzando derecha 12:25:23.555 -> Cruzando izquierda
12:25:17.223 -> Velocidad: 80    12:25:23.555 -> Velocidad: 80
12:25:17.315 -> Cruzando derecha 12:25:23.694 -> Cruzando izquierda
12:25:17.315 -> Velocidad: 80    12:25:23.694 -> Velocidad: 80
12:25:17.455 -> Cruzando derecha 12:25:23.833 -> Cruzando izquierda
12:25:17.455 -> Velocidad: 80    12:25:23.833 -> Velocidad: 80
```

Figura 41: Monitor serial desplegando movimiento de giro a la derecha (Figura A) y movimiento de giro a la izquierda (Figura B)

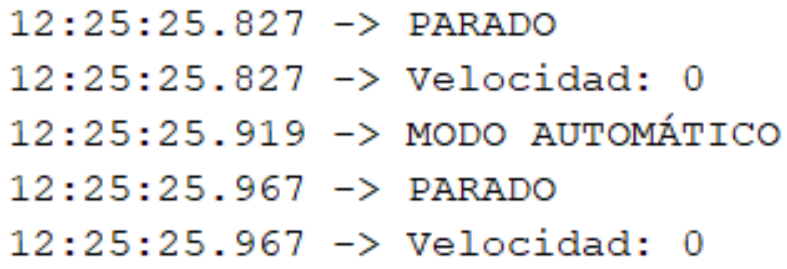


Figura 42: Monitor serial desplegando cambio de modo a automático

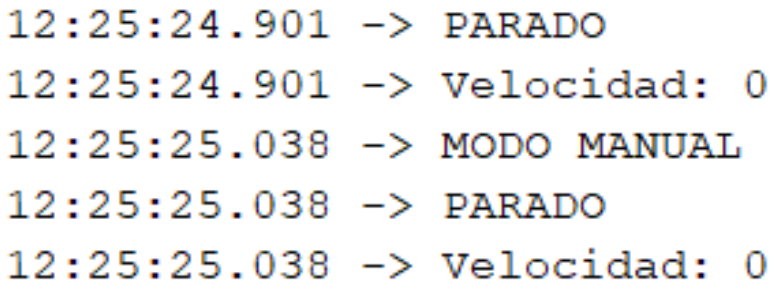


Figura 43: Monitor serial desplegando cambio de modo a manual

Al observar que los resultados obtenidos en las pruebas eran los esperados se procedió a realizar la implementación directa del control en el robot. Para esta implementación se utilizó el mismo código que fue utilizado en la segunda prueba y ahora en el robot se activaron los motores para que este pudiera ser manejado con el control. Al realizar esta implementación se logro mover el robot correctamente según los comando enviados.

Figura 44: Implementación del control remoto en el robot

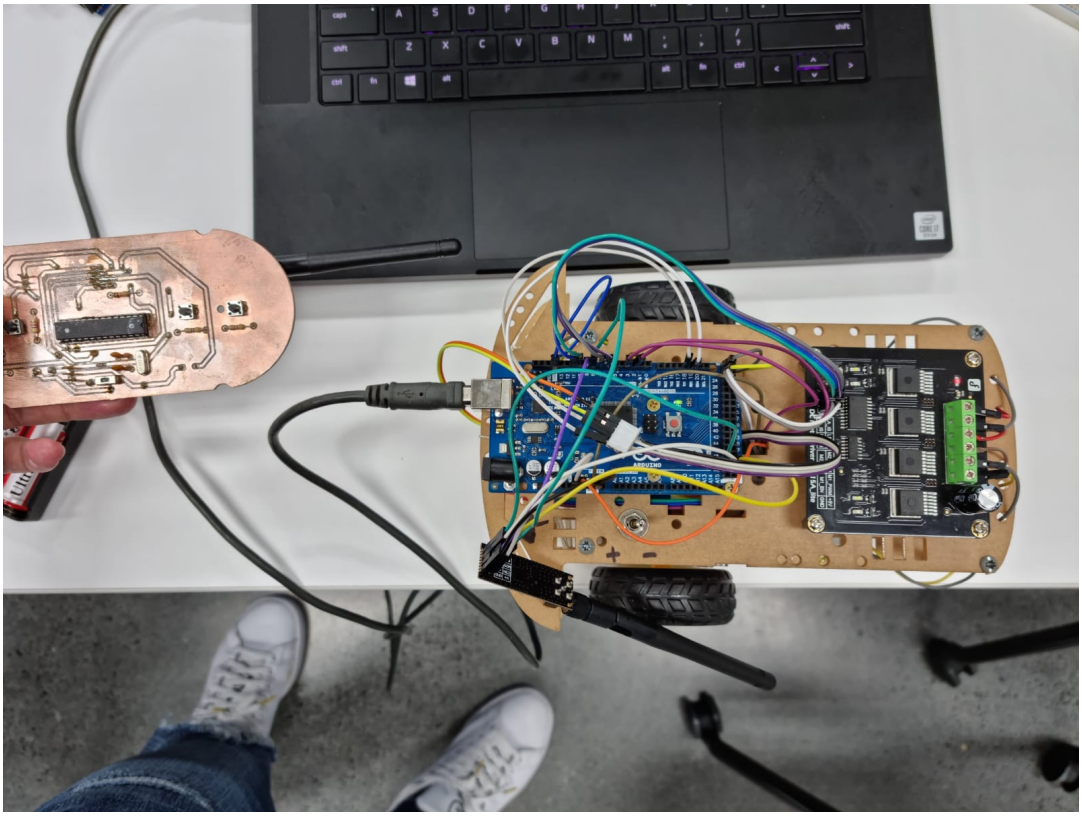


Figura 45: Control remoto y robot conectados exitosamente

Durante esta prueba de implementación también se utilizaron los comandos para los cambios de velocidad, lo cuales funcionaban correctamente ya que fue posible observar como el robot cambiaba su velocidad según la velocidad actual que se tenia dentro del ciclo.

Al tener el control remoto totalmente funcional se procedió a hacer cambios en el modelo CAD de la carcasa. Estos cambios contenían principalmente cambios de medidas en la tapadera para algunos componentes como el pad direccional, botones de acción y switch de encendido. También se cambiaron las medidas de los agujeros para los tornillos de ensamble ya que con base en una pequeña prueba de impresión que se realizó los agujeros verticales tienden a quedar un poco más flojos para la medida del tornillo. Tomando esto en cuenta se ajustaron los agujeros para que los tornillos pudieran entrar con facilidad pero que no fueran totalmente flojos.

Figura 46: Tapadera fabricada para prototipo No.2 de control remoto

El cuerpo del mando fue lo segundo que se modificó y en este la modificación principal que se realizó fue un cambio en los soportes de la PCB. Ya que estos eran totalmente uniformes algunas piezas soldadas tenían interferencia con la pieza provocando que no se pudiera atornillar completamente la placa. Esto se resolvió ajustando la medida de la pieza para que únicamente la parte en la que utilizarían los tornillos fuera alta y el resto de la pieza tuviera una medida menor para no perder el soporte y evitar que la pieza pudiera romperse al momento de atornillar la placa. Otro cambio que se realizó fue en los agujeros de ensamble del cuerpo principal ya que estos se ajustaron a las medidas que se usaron en la tapadera para que tuvieran un buen juego.

Figura 47: Cuerpo principal fabricado para prototipo No.2 de control remoto

Por último el compartimento de las baterías y el módulo de radiofrecuencia fue el que sufrió la mayor cantidad de cambios. El más significativo fue la adición de pestañas de sujeción para poder atornillar el compartimento al cuerpo principal. Esto permitió mantener una buena integridad estructural del control remoto sin perder una de las características de diseño asignadas que fue la facilidad de limpieza. Estas pestañas tienen agujeros para tornillos que se alinearon con agujeros de la misma medida creados en el cuerpo principal del mando. También se agregó una pequeña ranura en la parte lateral para que la antena del módulo de radiofrecuencia pudiera salir sin problemas. El último cambio que se realizó fue un pequeño ajuste de medidas par que las baterías no tuvieran tanto espacio de movimiento y así evitar problemas con los cables o que dañen el módulo de radiofrecuencia.

Figura 48: Compartimento de baterías y módulo de radiofrecuencia fabricado para prototipo No.2 de control remoto

CAPÍTULO 8

Sistema de recopilación de datos

8.1. Prototipo No.1 - React.JS, Node.JS y Mosca

Inicialmente se planteó la estructura que debe tener esta parte del proyecto por lo que es necesario conocer la información principal del robot y poder determinar las necesidades que tendrá la aplicación. Con base en las necesidades que se logren determinar se debe crear un bosquejo de cómo será la aplicación, este será diseñado únicamente en papel y lápiz ya que es necesario poder tener una idea básica de cómo se vera la aplicación y que componentes deben integrarse para que esta pueda funcionar. Seguido de esto se estará creando un "mockup" de la interfaz gráfica que será utilizada en la aplicación.

Figura 49: Mockup de aplicación en pantalla de datos actuales

Figura 50: Mockup de aplicación en pantalla de datos totales

Figura 51: Mockup de aplicación en pantalla de mantenimiento

Una vez se logró establecer de manera concreta que es lo que estará implementando dentro de la aplicación se hizo uso de la librería de JavaScript React, esta librería permitió crear toda la interfaz gráfica de la aplicación de manera reactiva por lo que al momento de detectar un cambio esta creará el render automáticamente y se podrá visualizar el cambio de manera rápida. Una vez se tiene el proyecto de React inicializado dentro del editor de código se comenzó por crear las pantallas necesarias para las diferentes secciones que tendría la aplicación que en este caso será una pantalla de datos actuales, una de datos totales y una pantalla de mantenimiento. Una vez creadas estas tres ventanas se comenzó a trabajar en una barra de navegación para poder movilizarse entre las tres pantallas creadas inicialmente y para esto se hizo uso del framework llamado Bootstrap, el cual facilita la creación de componentes otorgando partes generales de un componente para que puedan ser utilizadas para poder crear un componente que sea visualmente más complejo y que también sea adaptable lo cual significa que podrá cambiar su tamaño para adaptarse al tamaño de la ventana que se esta usando por lo cual no se tendrán problemas visuales con el componente.

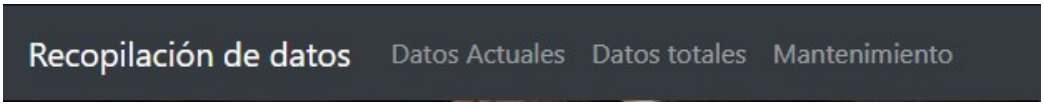


Figura 52: Barra de navegación en tamaño de ventana normal

Figura 53: Barra de navegación en tamaño de ventana reducido

Al ya tener creado el componente de la barra de navegación se procedió a utilizar otra librería llamada "React router" la cual permitiría crear los caminos necesarios para que cada uno de los botones dentro de la barra de navegación pueda trasladar al usuario a su respectiva página. La ventaja de utilizar esta librería es que los caminos que permite crear son bastante simples de configurar por lo que únicamente fue necesario tener los archivos creados para cada una de las pantallas y con esto se utilizaron las distintas funciones que posee la librería para asignar a cada botón su respectiva pantalla y que estos pudieran trasladarse a las mismas.

El segundo componente principal que se creo fue el pie de página que contendría cualquier tipo de información estática que se desee desplegar, en este caso se colocó la información de contacto. De igual manera que con la barra de navegación se hizo uso de Bootstrap ya que esta también permite crear el componente del pie de página de manera que no será necesario fabricarlo desde cero y también es posible crear el componente de manera que este sea adaptable al tamaño de la ventana. En este pie de página se crearon dos columnas en las cuales se puede ingresar información, pero únicamente se hizo uso de una de las columnas para colocar la información de contacto personal. Con el pie de página terminado y estilizado de la manera deseada únicamente fue necesario importar este componente a cada una de las pantallas que se crearon para que todas pudieran tener su respectivo pie de página.

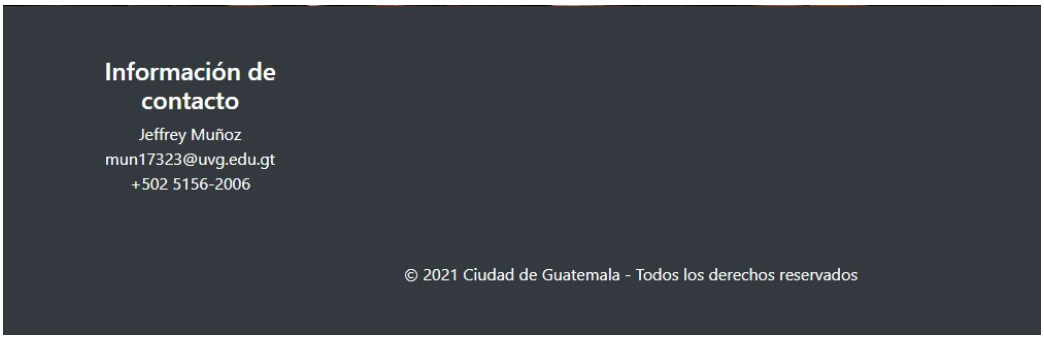


Figura 54: Pie de página con información de contacto

Con los dos componentes delimitantes de la altura de la pantalla se continuó trabajando en las pantallas creadas y lo primero fue crear un contenedor de contenido general para cada una de las pantallas. Este contenedor actuaria como la parte principal de cada pantalla y en la cual se ubicarán los diferentes componentes que se deseen usar para cada pantalla, este contenedor se creó para que tuviera propiedad fluida que es el mismo tipo de propiedad que tiene los componentes adaptables pero especifico para contenedores ya que si se usa una

imagen de relleno para el contenedor como fue en este caso la propiedad de fluido permitirá que la imagen también se adapte al tamaño de la ventana sin que esta se distorsione y se vuelva irreconocible.

Ya que el contenedor principal para cada una de las pantallas estaba listo se creó un sistema de red el cual facilitaría la ubicación de los componentes en cada página de manera que estarían ubicados en una "cuadrícula" con distintos espacios. Para la página de datos actual el sistema de red que se creó fue de dos columnas y en la primera columna se crearon tres filas del mismo tamaño para poder ubicar tres componentes distintos mientras que en la segunda columna se crearon únicamente dos filas en las cuales la primera tenía el mismo tamaño que las filas de la primera columna y la segunda fila tenía el tamaño de dos de estas filas, esta decisión se tomó ya que en la última fila se ubicarían dos componentes que están relacionados entre sí por lo que sería mas fácil trabajarlos si se encuentran dentro del mismo espacio del sistema de red.

En la segunda pantalla que es la pantalla de datos totales el sistema de red es mucho más simple ya que únicamente se creó una cuadrícula de dos columnas con una fila cada una que tendrán el mismo tamaño, se debe a que en esta página no se planean colocar varios componentes. Por último la pantalla de mantenimiento tiene una cuadrícula que también consta de dos columnas, pero la primera columna tiene únicamente una fila que es de tamaño adaptable mientras que la segunda columna tiene dos filas del mismo tamaño que también son adaptables por lo que si la primera columna comienza a aumentar mucho su tamaño estas filas en la segunda columna también aumentaran su tamaño, pero de manera igualdad para mantener una proporción. Una vez los sistemas de red fueron creados exitosamente se comenzó a programar el primer componente general para colocar dentro de las cuadrículas de cada una de las pantallas, este componente se conoce como una tarjeta y su principal función dentro del proyecto fue actuar como un contenedor independiente el cual se podrá configurar para que más componentes puedan ser insertados en este y también para hacer la modificación de estos componentes más sencilla y eficiente.

Ya que las tarjetas actúan como contenedores la configuración de estas es la misma que la del contenedor general de cada página para que pueda ser fluida. En este caso se agregaron propiedades específicas según los componentes que se deseaban incluir en cada una por lo que la primera propiedad fue la imagen y con esta propiedad será posible agregar una pequeña imagen a la tarjeta con el fin de tener una descripción visual sobre qué cosas estarán dentro de cada una de las tarjetas, la segunda propiedad fue de título de tarjeta y esta fue configurada para que al momento de no tener un título específico en la tarjeta se despliegue el mensaje "Title goes here". Como tercera propiedad se agrego el cuerpo de la tarjeta para que se pudiera desplegar cualquier texto deseado y de igual manera que en la propiedad del tí tulo se agregó un mensaje que indicaría que no hay texto que desplegar en el cuerpo de la tarjeta.

Estas tres son las propiedades generales que tendrán todas las tarjetas, adicional a esto se crearon más propiedades especificas según el uso que se le fuera a dar a la tarjeta en cada una de las pantallas. La primera propiedad específica que se creó fue la implementación de una barra de progreso que sería utilizada para poder desplegar el nivel de batería que posee el robot actualmente, para esta propiedad se creo otro componente nuevo utilizando Bootstrap el cual sería una barra de progreso que cambiaría según el valor que se asigne por lo que se ajustó para que pudiera desplegar un dato que iría disminuyendo cada segundo

para poder verificar que el componente se comporta de la manera esperada.

La segunda propiedad específica que se creó fue la de un contador de tiempo el cual sería implementado para poder llevar un control del tiempo que el robot ha estado en funcionamiento. Esta propiedad es bastante peculiar ya que de igual manera que con la barra de progreso se tuvo que crear un componente externo, pero en este caso dicho componente externo estaría compuesto de más componentes externos como un reloj y botones. Este componente externo fue configurado para que tuviera cuatro botones principales que serial el botón de inicio, el botón de pausa, el botón de resumir y el botón de reseteo, también tendría un componente de texto que desplegaría el valor del tiempo que esta contando actualmente el reloj. Cada uno de estos botones fue configurado para que pudieran actuar únicamente en un evento de click y pudieran interactuar con el intervalo del contador para inicializarlo, detenerlo o resetearlo.

Después de terminar de crear las propiedades específicas que se habían planeado se comenzó a llenar cada uno de los sistemas de red que estaban en las distintas pantallas utilizando las tarjetas creadas anteriormente y ya que las tarjetas fueron creadas de manera general fue posible personalizar cada una de las tarjetas para que tuviera el contenido que se necesitaba según la información que se deseaba desplegar en las tarjetas.

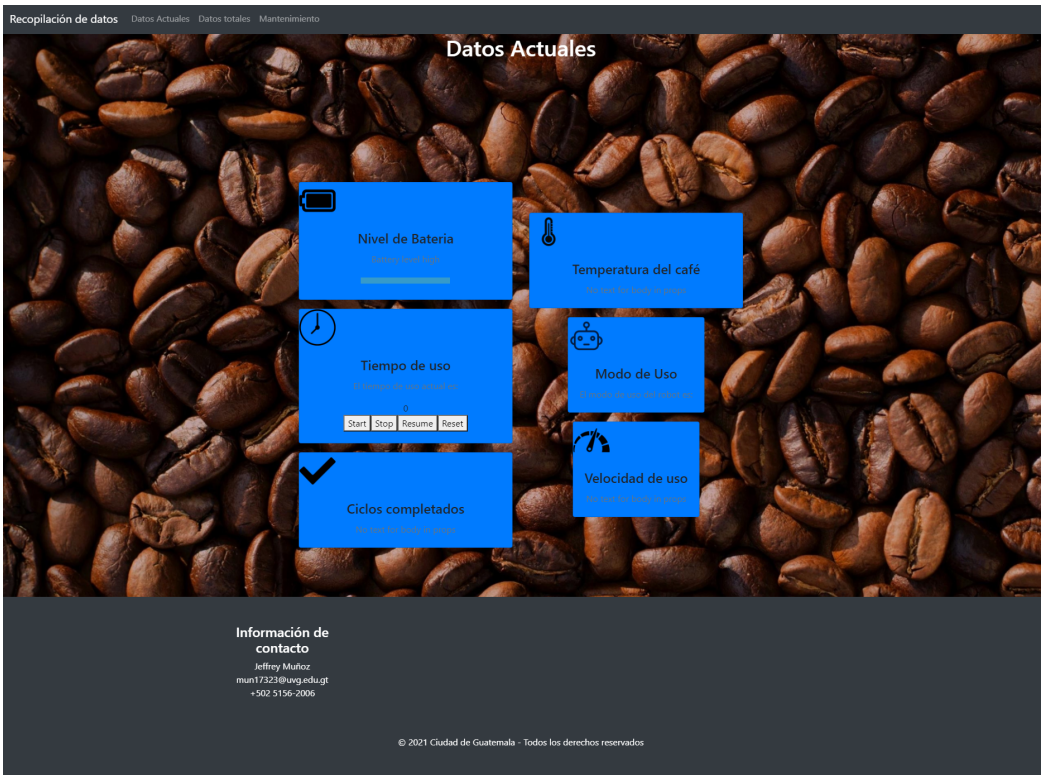


Figura 55: Pantalla de datos actuales con tarjetas ubicadas en el sistema de red

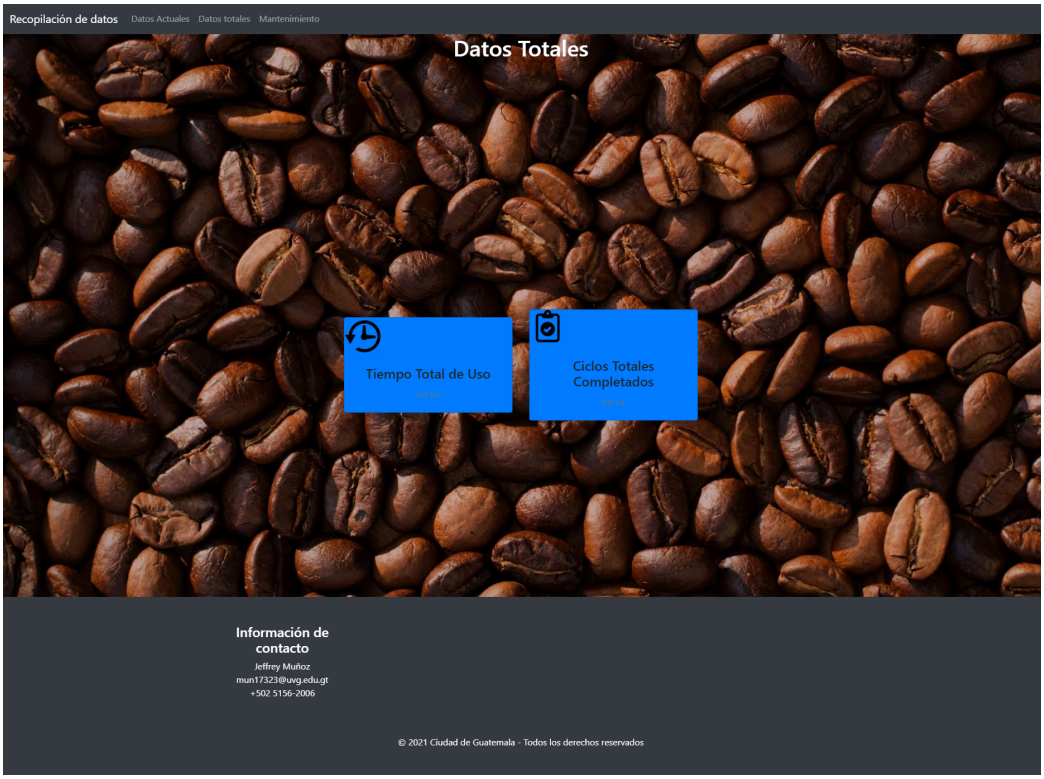


Figura 56: Pantalla de datos totales con tarjetas ubicadas en el sistema de red

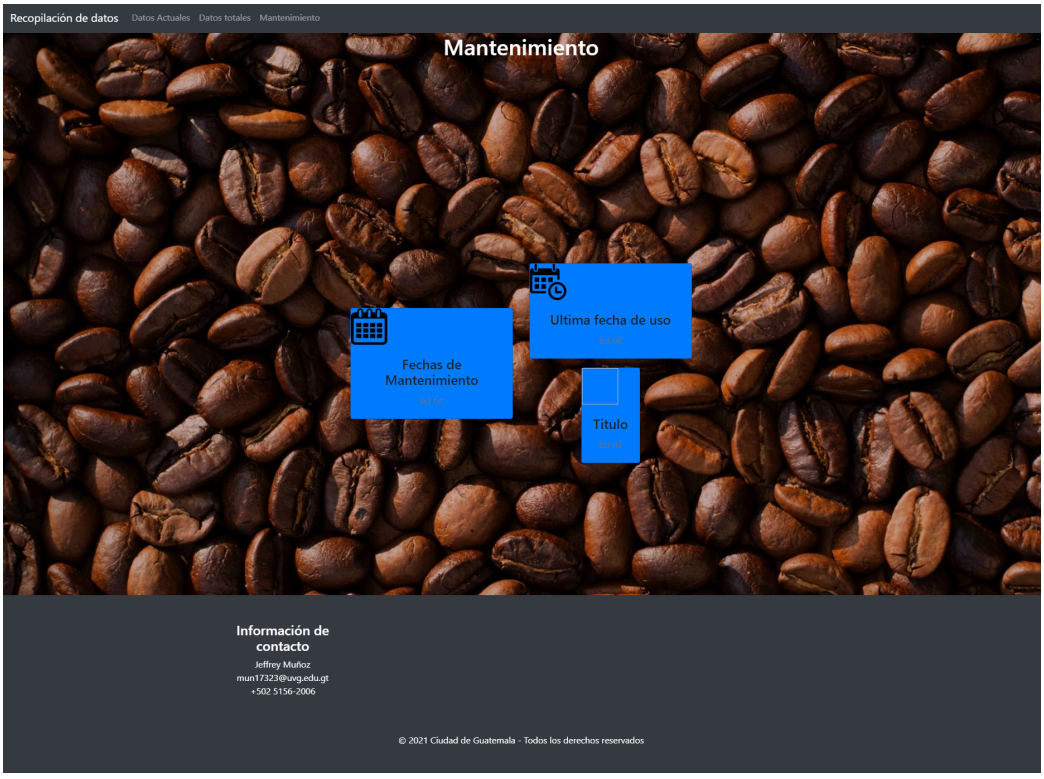


Figura 57: Pantalla de mantenimiento con tarjetas ubicadas en el sistema de red

Con el frontend terminado se comenzó a trabajar en la comunicación por Wi-Fi para el sistema de recopilación de datos. Ya que para el sistema se utilizaría el protocolo de comunicación MQTT fue necesario establecer un broker el cual permitiría la suscripción y publicación de los datos recopilados. Para poder realizar la conexión entre el frontend y el backend de la aplicación se decidió utilizar Node.JS para poder tener un broker basado en JavaScript. Esto se debe a que el routeo de los datos hacia los componentes del frontend sería más fácil debido a que la mayoría de librerías utilizadas para este proceso se encuentran hechas en JavaScript.

Para el broker basado en JavaScript se utilizó Mosca el cual es un broker gratuito y público para el uso de aplicación que implementen MQTT. Por lo que luego de crear un nuevo proyecto de Node.JS se instaló la librería que posee las funciones específicas para poder conectarse al servidor del broker. Luego de la instalación se creó el archivo principal del broker para poder ingresar la congracian necesaria para el acceso al servidor y para poder verificar la conexión de los clientes con el broker. Para la configuración de acceso al servidor se utilizó la función "mosca.Serverz únicamente se le asignó el puerto al cual se estaría conectando. Luego se utilizaron dos eventos para poder verificar cuándo se logró la conexión exitosa con el servidor, cuándo identifica que un cliente nuevo se ha conectado al broker y cuándo un mensaje ha sido publicado exitosamente.

Con la configuración del broker terminada se creó un nuevo archivo para poder crear un cliente encargado de publicar mensajes al broker. Para este cliente se utilizó el módulo MQTT con el cual se creó la conexión con el broker utilizando la dirección en la cual esta creado y la puerta con la que fue configurado. En esta prueba inicial se utilizó un Arduino UNO junto a un potenciómetro para poder publicar el valor del potenciómetro al broker por lo que fue necesario instalar el modulo de "serialport"dentro del proyecto. Una vez instalado el módulo se creó una constante que leería el puerto serial del Arduino. Con esta constante creada se usó un evento el cual una vez detecte la conexión correcta con el broker enviaría un mensaje a un tópico específico. En este caso el mensaje que se envió fue el valor del potenciómetro leído por medio del puerto serial dentro del tópico llamado "topic test".

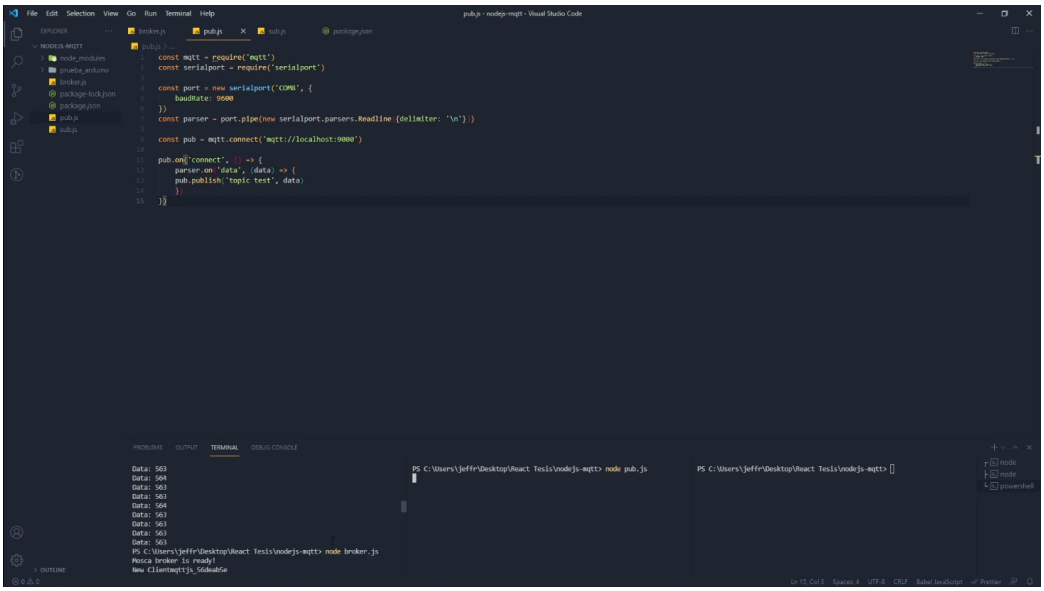


Figura 58: Prueba de conexión de broker y funcionamiento de cliente publicador

Al tener completo el cliente publicador de los datos lo siguiente fue crear el cliente suscriptor para que este pueda recibir los datos publicados. De igual manera que con el cliente encargado de publicar se utilizó el módulo MQTT y también se utilizaron las configuraciones de conexión con el broker. Para poder comenzar a recibir los mensaje publicados se creó otro evento de suscripción el cual al momento de entablar una conexión con el broker se suscribiría al mismo tópico en el cual se están enviando los datos en el cliente publicador. Para poder desplegar los mensajes recibidos se utilizó un evento nuevo que permitiría imprimir los mensajes recibidos del tópico al cual se está suscrito.

Con el broker y los clientes funcionando correctamente se procedió a implementar una base de datos para poder almacenar los mensajes publicados y recibidos correctamente. El servicio de base de datos que se utilizó fue MySQL debido a que ya existe un módulo especifico para Node.JS que permite utilizar distintas funciones para realizar operaciones de la base de datos. Por lo que lo primero que se realizó fue crear la base de datos dentro de MySQL llamada "mqttnode", dentro de esta base de datos se creo una tabla llamada arduino en la cual estarían los datos de las lecturas del potenciómetro. A esta tabla se le asignaron las propiedades de un identificador entero que no sea no nulo y además que se auto incremente cada vez que un nuevo dato es insertado en la tabla. Los datos de la tabla también se configuraron para que fueran enteros debido a que las lecturas del potenciómetro se realizarían únicamente con datos enteros.

Luego de tener la base de datos y la tabla configurada se realizaron los cambios dentro del cliente suscriptor para poder agregar el módulo de MySQL y poder utilizar sus funciones. Lo primero fue crear una constante con las propiedades para la conexión con la base de datos creada. Las propiedades necesarias fueron: la locación del host, el puerto utilizado por la base de datos, el usuario de MySQL, la contraseña del usuario y por último la base de datos a la cual se desea accesar. Con la conexión a la base de datos realizada exitosamente se agregó una función del módulo MySQL al evento de impresión de datos para poder realizar una consulta a la base de datos y poder insertar el último mensaje recibido dentro de la

tabla creada en la base de datos.

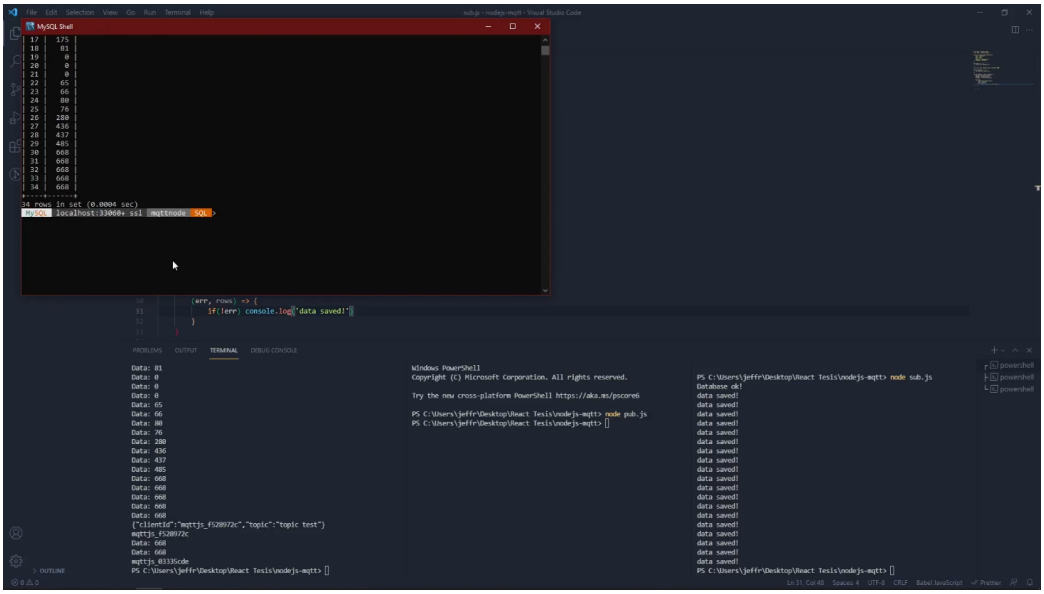


Figura 59: Funcionamiento del broker junto a los clientes publicadores y suscriptores

Ya que el broker y los clientes funcionaban correctamente utilizando la lectura del puerto serial de un Arduino UNO se comenzaron a realizar las pruebas para la conexión de un ESP32. Este microcontrolador debía conectarse con el broker como un cliente publicador para poder enviar los datos recopilados de los sensores. Al momento de realizar las pruebas de conexión se tuvieron problemas de inclusión para el microcontrolador. Entre estos se encontraban problemas de compilación por falta de funcionalidad para compatibilidad. Por lo que se realizo una investigación para poder solucionar los problemas encontrados.

Durante esta investigación se encontró que el mantenimiento al servicio Mosca se había descontinuado aproximadamente en 2017. Dentro de la investigación se encontró que el servicio no poseía la funcionalidad necesaria para la conectividad Wi-Fi deseada. Tambien se encontro otro servicio de broker llamado Mosquitto que junto a otro servicio llamado Node-Red ofrece una manera más eficiente de poder manejar el protocolo MQTT. Ya que Node-Red está hecho con el protocolo MQTT como su primera prioridad este servicio presento una opción más eficiente y con menor complejidad para la implementación dentro del proyecto haciendo uso del protocolo. Esto se debe a que una de las diferencias principales entre Node.JS y Node-Red en términos de MQTT se encuentra en el desempeño máximo que estos puede tener. Node.JS tiene un desempeño máximo más alto en comparación al que tiene Node-Red. Pero debido a que la aplicación dentro del proyecto es principalmente la de transmisión de datos simples. La complejidad que presenta la configuración del broker y demás componentes de Node.JS no se considera óptima. Debido a esto se optó por realizar el cambio de servicios para comenzar el sistema de recopilación de datos nuevamente utilizando Node-Red y Mosquitto dentro de una Raspberry Pi que actuaría como un servidor.

8.2. Prototipo No.2 - Node-Red

Esta sección del proyecto constará de tres partes importantes para poder funcionar las cuales son el módulo ESP32 para la transmisión de datos por medio de WiFi, una Raspberry Pi que actuara como servidor remoto para poder montar el broker a utilizar que en este caso será Mosquitto y también para poder almacenar los datos recopilados dentro de una base de datos, por último se utilizará Node-Red y Node-Red Dashboard para poder recibir los datos del broker y poder mapear estos a una interfaz gráfica desarrollada utilizando Node-Red Dashboard.

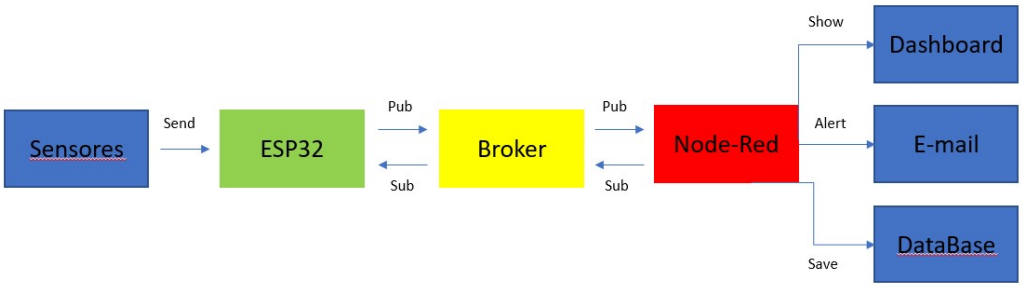


Figura 60: Diagrama de funcionamiento de la segunda iteración del sistema de recopilación de datos

Se comenzó configurando el ESP32 de manera que primero se instaló fueron las librearías de ESP32 de Arduino para poder agregar este módulo a la lista de boards disponibles dentro del IDE de Arduino y así poder cargar los códigos sin ningún problema, también se instaló la librería Pubsubclient la cual permite utilizar funciones de mensajería MQTT para poder hacer todas las acciones necesarias como la conexión con el broker, suscripción y publicación de mensajes en tópicos específicos. Una vez se tenían estas librerías instaladas se procedió a probar el ESP32 para verificar que esta funciona de manera correcta por lo que se cargó un código de ejemplo que permite realizar un escaneo de las redes disponibles a las cuales se puede conectar el módulo, una vez se determinó que el módulo puede escanear de manera correctas las redes disponibles en el área se procedió a realizar la conexión con la red en particular que se utilizaría para el sistema de recopilación de datos.

Una vez se tenía bien configurado el módulo WiFi se procedió a utilizar la Raspberry Pi para poder instalar las aplicaciones necesarias, la primera fue el broker Mosquitto el cual es el principal responsable de recibir todos los mensajes, filtrarlos y decidir quiénes son los clientes suscritos a los respectivos tópicos para poder recibir los mensajes publicados en estos tópicos. Para la instalación de Mosquitto se utilizó la terminar de la Raspberry Pi en la cual se ingresó el comando “sudo apt install -y mosquitto mosquitto-clients” para poder instalar la última versión disponible de Mosquitto y también para poder instalar el cliente de Mosquitto que permitirá realizar pruebas de funcionamiento del broker. Una vez la instalación del broker es completada se utilizó otro comando para poder asegurar que el broker inicializará automáticamente como un servicio propio de la Raspberry Pi cada vez que este se vuelva a encender, el comando utilizado fue “sudo systemctl enable mosquitto.service”.

Para poder verificar que el broker funciona de manera correcta se realizó una prueba de publicación y suscripción muy simple en la cual se utilizaron dos terminales de Raspberry Pi de las cuales una actuaría como suscriptor y la otra terminal actuaría como publicador

de un mensaje dentro de un tópico. En la terminal que tendría el rol de suscriptor se utilizó el comando ‘‘`mosquitto_sub -d -t testT`’’ para que esta terminal estuviera suscrita al tópico “testT”. En la segunda terminal que sería la que actuaría como publicador se utilizo el comando ‘‘`mosquitto_pub -d -t testT -m ‘‘Hello world!’”’` para poder hacer que esta terminal publique el mensaje “Hello world!” dentro del tópico “testT”. Se observó que el mensaje fue publicado y recibido exitosamente por lo que se continuó trabajando con el resto de los servicios de la Raspberry Pi ya que el broker funciona de manera correcta.

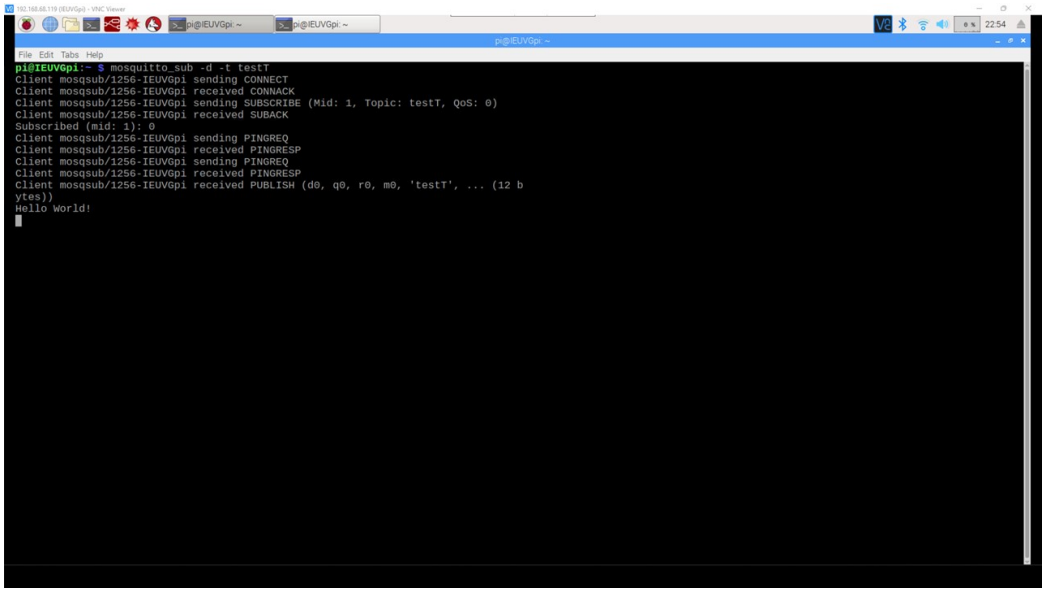


Figura 61: Terminal con rol de suscriptor para prueba de funcionamiento de broker Mosquitto

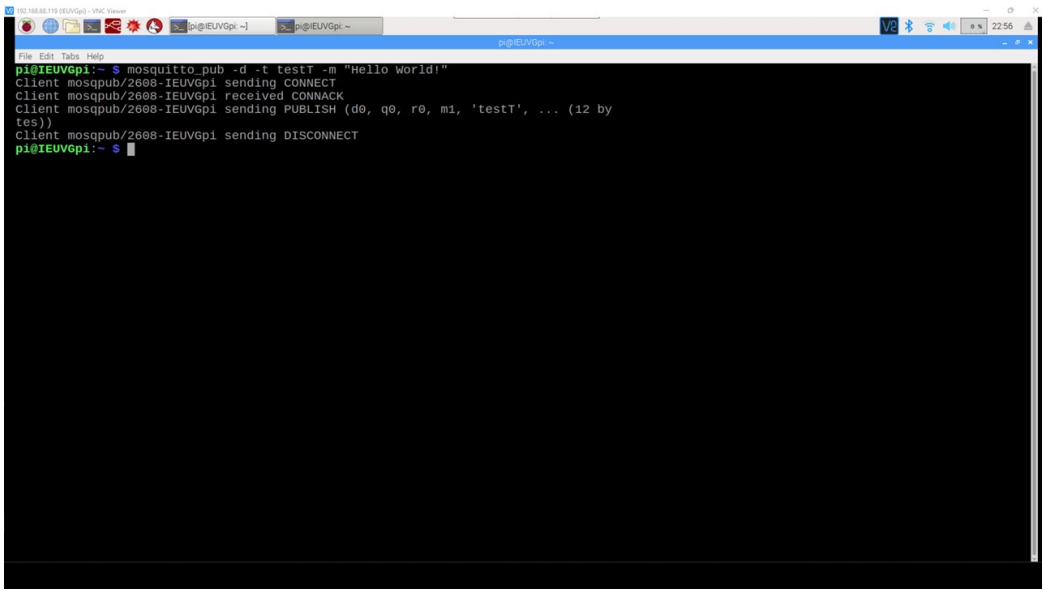


Figura 62: Terminal con rol de publicador para prueba de funcionamiento de broker Mosquitto

Seguido de esto se instaló Node-Red para poder hacer los flujos de cada uno de los

tópicos de manera que estos recibieran los datos de su respectivo tópico. Para poder realizar la instalación de Node-Red dentro de la Raspberry Pi se utilizó la terminal en la cual se ingresó el comando "bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)", este comando permite descargar e instalar Node.js, npm y Node-Red desde el repositorio de Github de Node-Red Org. Después de haber finalizado la instalación de Node-Red se configuró este para que de igual manera que el broker se inicializará como un servicio de la Raspberry Pi cada vez que esta se encienda y para eso se uso otra terminal en la cual se ingresó el comando "sudo systemctl enable nodered.service" y por último se hizo un reboot de la Raspberry Pi par que todos los cambios realizados a los servicios pudieran surtir efecto.

Ya que el broker utiliza la dirección IP de la Raspberry Pi como dirección propia para poder entablar conexiones con los clientes que publicarán mensajes en tópicos y los clientes que estarán suscritos a los tópicos se tomó la decisión de configurar una dirección IP estática en la Raspberry Pi para asegurar que esta siempre tendrá la misma dirección y por lo tanto el broker también por lo que no será necesario volver a realizar las configuraciones a la dirección de los clientes. Para poder realizar esta configuración fue necesario verificar la dirección IP actual de la Raspberry Pi, para esto se utilizó la terminal junto al comando "ifconfig" el cual mostraría la dirección IP actual dentro de la red a la cual se está conectado.

Después de obtener la dirección IP actual que se posee dentro la de red a la cual se está conectado es necesario verificar el "Gateway" que se tiene ya que será necesario ingresarlo dentro de la configuración final para la dirección IP estática, para poder consultar esto se ingresa el comando "sudo route -n" dentro de la misma terminal utilizada anteriormente. Una vez se tengan estas dos direcciones se pueden realizar los cambios en la configuración dhcpcd para asignar una dirección IP estática a la Raspberry Pi y para esto se uso el comando "sudo nano /etc/dhcpcd.conf" dentro de la terminal, esto abrirá el archivo de configuración en el cual se deben ingresar las siguientes configuraciones:

```
Interface wlan0

static ip_address=''dirección ip actual''

static routers=''gateway actual''

static domain_name_servers=8.8.8.8 8.8.4.4
```

Al ingresar estas configuraciones únicamente es necesario guardar los cambios realizados al archivo y realizar un reboot a la Raspberry Pi par que estas puedan tomar efecto. Al volver a inicializar la Raspberry se verificó la dirección IP para poder saber que en efecto las configuraciones funcionaron correctamente.

El siguiente paso que se realizó fue inicializar el editor de flujo de trabajo de Node-Red para poder instalar la librería Node-Red Dashboard la cual permite utilizar los nodos necesarios para poder crear la interfaz gráfica en la cual se desplegaran recopilados por los sensores.

Dentro del servicio Node-Red lo primero fue utilizar los nodos de MQTT para poder conectarse al broker utilizando la dirección IP con la cual está configurado. Una vez la conexión es exitosa se procedió a insertar los nodos de entrada de datos para MQTT para

poder hacer las lecturas de los datos que serán publicados, cada uno de estos nodos puede configurarse para suscribirse únicamente a un tópico en específico por lo que fue necesario utilizar tres nodos para poder realizar las suscripciones a los tres tópicos que se planean publicar desde el ESP32.

Para poder el funcionamiento de la transmisión de datos se utilizó un potenciómetro para enviar el valor actual de voltaje y también se usó un sensor DHT11 el cual permitirá leer datos de temperatura y humedad de la habitación los cuales fueron transmitidos por medio de Wi-Fi. Una vez se configuró de manera correcta el sensor y el potenciómetro se procedió a realizar las lecturas de cada uno de estos asignándole variables floats a cada valor que se deseaba leer y se utilizó la función "client.publish" que incluye la librería Pubsubclient para poder enviar el valor de la lectura a su respectivo tópico. Para poder enviar este valor y que fuese posible realizar el despliegue correcto en otra aplicación fue necesario enviarlo de manera que era un string por lo que se implementó la función "dtostrf" la cual permitió transformar un float a un string y seleccionar la cantidad de números que se desean desplegar después del punto decimal. Se implementó el uso del monitor serial de Arduino para poder desplegar en este los datos publicados cada vez que se cumpla el tiempo de espera entre publicaciones, por el lado de Node-Red se agregaron los nodos de "debuggin" para poder desplegar los valores recibidos en la pantalla de "debuggin" que trae incluido el servicio. Una vez se logró desplegar en la pantalla los valores recibidos de las suscripciones se sabe que está funcionando de manera correcta tanto para la publicación como la suscripción de los datos con el protocolo.

Con el correcto funcionamiento de la comunicación se comenzó a utilizar el dashboard de Node-Red para poder hacer una interfaz gráfica dedicada a desplegar los datos recibidos de una mejor manera por lo que para comenzar con esta interfaz gráfica se crearon dos grupos principales que serán valores actuales y valores recibidos. En el grupo de valores actuales se agregaron todos los componentes que únicamente desplieguen los valores recibidos más actuales en forma de válvulas o barras de progreso, mientras que en el grupo de valores recibidos se implementaron gráficas lineales las cuales permitieron observar los cambios de los valores recibidos a través del tiempo en forma de gráficas lineales y también buscar un momento del tiempo para poder conocer el valor exacto que se recibió en ese momento.

Para poder asignarle los valores correctos a cada uno de los componentes de la interfaz gráfica se utilizaron los nodos de gráficas, válvulas y barras de progreso que se proporcionan en la librería descargada. Estos nodos únicamente se conectaron a los receptores de datos en Node-red que en este caso son los nodos de MQTT que se utilizaron anterior y únicamente restaba configurar la apariencia que se deseaba que tuvieran estos componentes dentro de la interfaz gráfica.

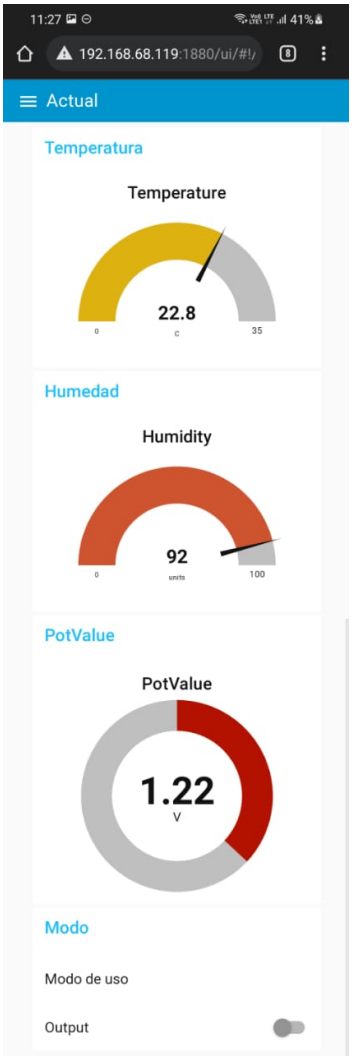


Figura 63: Página de despliegue de datos actuales en interfaz gráfica

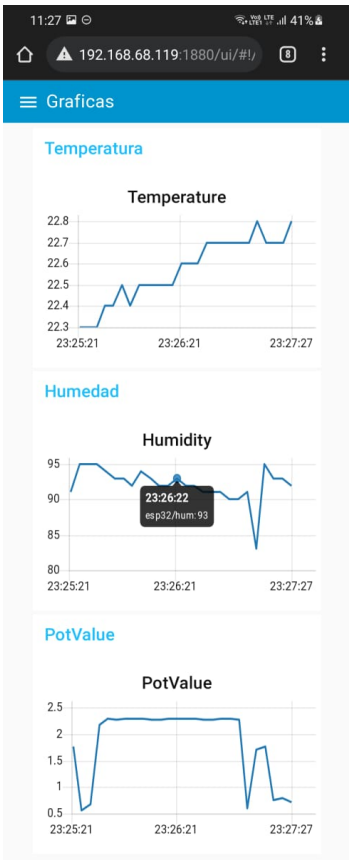


Figura 64: Página de despliegue de gráficas en interfaz gráfica

Al lograr el resultado visual deseado de la interfaz gráfica se procedió a trabajar en la configuración de la base de datos para que todos los valores que sean recibidos dentro de Node-Red se pudieran almacenar para su futura consulta. Este proceso comenzó instalando MariaDB dentro de la Raspberry Pi que se estaba utilizando ya que esta sería el receptor de los datos enviados por el módulo ESP32, luego se instaló PHPMyAdmin para poder tener un administrador de las bases de datos creadas y de esta manera se podría manejar de una manera más fácil los cambios que sean necesarios o también se puede exportar los datos de las bases de datos para poderlos analizar fuera de línea.

Una vez instalados estos servicios se utilizó PHPMyAdmin desde la web para poder crear una nueva base de datos dedicada exclusivamente al almacenamiento de los datos enviados por MQTT a Node-Red y dentro de esta base de datos se creó la tabla necesaria para ordenar los datos almacenados. Ahora del lado de Node-Red fue necesario instalar una nueva librería que permitiera el uso de nodos dedicados para la comunicación con bases de datos MySQL, con la nueva librería instalada se incluyó el nodo específico que se necesita para poder realizar la conexión con la base de datos recién creada de manera que únicamente fue necesario incluir el nombre de la base de datos, la tabla específica a la cual se desea conectar y los datos de ingreso (usuario y contraseña) para asegurarse que no cualquier individuo pueda establecer comunicación con la base de datos. El último paso que se realizó fue incluir un pequeño código que contendría las funciones necesarias para poder insertar datos dentro de la tabla correcta y revisar dentro de PHPMyAdmin que los datos se estuvieran almacenando de

manera correcta dentro de la tabla creada.

id ▲ 1	time	temp	humidity
1	2021-10-04 11:52:41	0	69
2	2021-10-04 11:52:41	23.3	0
3	2021-10-04 11:52:46	0	78
4	2021-10-04 11:52:46	23.2	0
5	2021-10-04 11:52:51	0	77
6	2021-10-04 11:52:51	23.3	0
7	2021-10-04 11:52:56	0	78
8	2021-10-04 11:52:56	23.2	0
9	2021-10-04 11:53:01	0	77
10	2021-10-04 11:53:01	23.3	0
11	2021-10-04 11:53:06	0	77
12	2021-10-04 11:53:06	23.3	0
13	2021-10-04 11:53:11	0	77
14	2021-10-04 11:53:11	23.3	0
15	2021-10-04 11:53:16	0	77
16	2021-10-04 11:53:16	23.3	0
17	2021-10-04 11:53:51	0	69
18	2021-10-04 11:53:51	23.3	0
19	2021-10-04 11:53:56	0	77
20	2021-10-04 11:53:56	23.3	0
21	2021-10-04 11:54:01	0	78
22	2021-10-04 11:54:01	23.3	0
23	2021-10-04 11:54:06	0	78
24	2021-10-04 11:54:06	23.3	0
25	2021-10-04 12:15:06	74	0

Figura 65: Base de datos dentro de PHPMyAdmin para almacenamiento de datos de temperatura y humedad recolectados por sensor

id ▲ 1	time	temp	humidity
676	2021-11-03 23:32:53	27.3	70
677	2021-11-03 23:32:58	27.3	72
678	2021-11-03 23:32:58	27.1	72
679	2021-11-03 23:33:03	27.1	72
680	2021-11-03 23:33:03	26.9	72
681	2021-11-03 23:33:08	26.9	72
682	2021-11-03 23:33:08	26.8	72
683	2021-11-03 23:33:13	26.8	73
684	2021-11-03 23:33:13	26.7	73
685	2021-11-03 23:33:18	26.7	74
686	2021-11-03 23:33:18	26.5	74
687	2021-11-03 23:33:23	26.5	74
688	2021-11-03 23:33:23	26.4	74
689	2021-11-03 23:33:28	26.4	75
690	2021-11-03 23:33:28	26.2	75
691	2021-11-03 23:33:33	26.2	76
692	2021-11-03 23:33:33	26.1	76
693	2021-11-03 23:33:38	26.1	75
694	2021-11-03 23:33:38	25.9	75
695	2021-11-03 23:33:43	25.9	76
696	2021-11-03 23:33:43	25.8	76
697	2021-11-03 23:33:48	25.8	77
698	2021-11-03 23:33:48	25.8	77
699	2021-11-03 23:33:53	25.8	78
700	2021-11-03 23:33:53	25.6	78

Figura 66: Base de datos dentro de PHPMyAdmin para almacenamiento de datos de temperatura y humedad recolectados por sensor

Lo último que se agregó dentro de Node-Red fue una función extra para poder obtener notificaciones por medio de correo electrónico cada vez que ocurra un cambio de datos muy repentino y no deseado. Para poder lograr esto se descargó una nueva librería de comunicación la cual provee nodos para dos tipos de comunicación distintos que son correo electrónico y mensajes de texto. En esta ocasión se utilizó la opción de correo electrónico ya que Node-Red únicamente permite enviar y recibir mensajes de texto con una compañía telefónica en específico la cual no está disponible en Guatemala. Para poder enviar los correos electrónicos únicamente fue necesario crear un pequeño código que contuviera la dirección de correo electrónico a la cual se desea enviar, el cuerpo del correo electrónico y el servicio que se utilizaría que en este caso fue Gmail. Por último se implemento un nodo de salida para correos electrónicos el cual recibiría el código creado anteriormente y realizaría la conexión con el servicio seleccionado para poder habilitar él envió de correos.

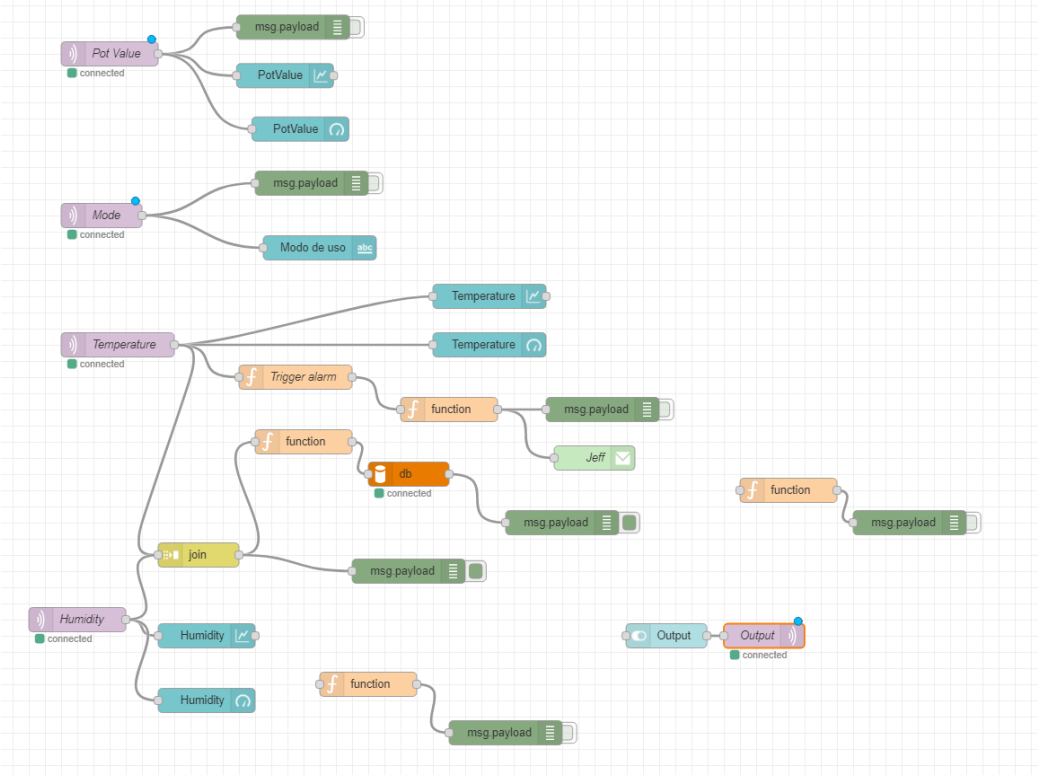


Figura 67: Flujo de trabajo dentro de Node-Red

CAPÍTULO 9

Preparación para pruebas de campo

El poder verificar que el robot principal pueda ser controlado de manera correcta es uno de los objetivos principales del proyecto. Para poder comenzar a utilizar el robot fue necesario construir dos nuevas piezas que permitirían mover el robot y también realizar cableado de una manera efectiva. En este caso las piezas principales que se crearon para esta etapa de pruebas fueron un nuevo control remoto rediseñado para poder mover individualmente cada uno de los motores y también para poder mover las aspas del robot. La segunda pieza que se creó fue una placa para poder montar sobre el Arduino MEGA que controlara al robot y esto se realizó con el propósito de facilitar todo el cableado interno de los componentes del robot.

9.1. Rediseño de control remoto

Para poder realizar el rediseño del control remoto fue necesario tomar en consideración la condición actual del robot principal. Esto se debe a que los motores del robot no tendrían encoders por lo que no seria posible que ambos motores arrancarán al mismo tiempo utilizando un solo botón del control remoto. Para poder arreglar esto se cambió el layout principal de los botones para que cada motor tuviera un botón de avance y uno de retroceso, también se agregaron dos botones nuevos que servirían para poder subir o bajar las aspas del robot.

Ya que los cambios necesarios para el funcionamiento del nuevo control eran principalmente respecto al layout del prototipo No.2 del control remoto fue posible reutilizar el circuito principal de este. Los cambios principales que se realizaron fueron en términos de diseño de PCB ya que fue necesario hacer cambios a las posiciones de los tracks para que estos se ajustarán a la nueva forma del control.

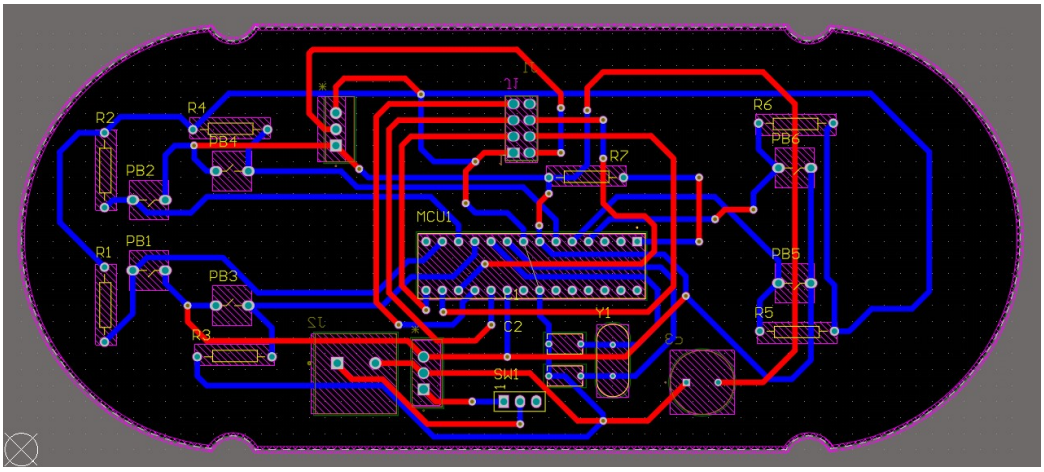


Figura 68: Cara superior de PCB para rediseño de control remoto

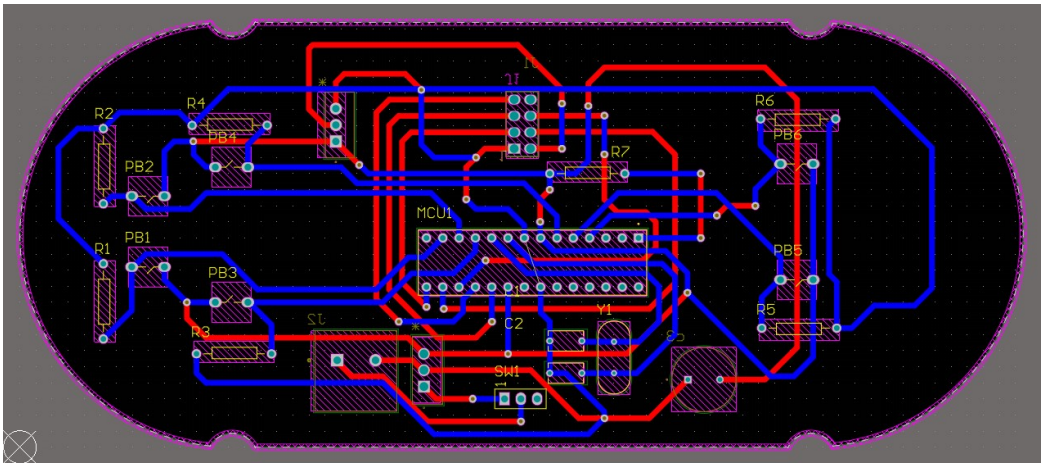


Figura 69: Cara inferior de PCB para rediseño de control remoto

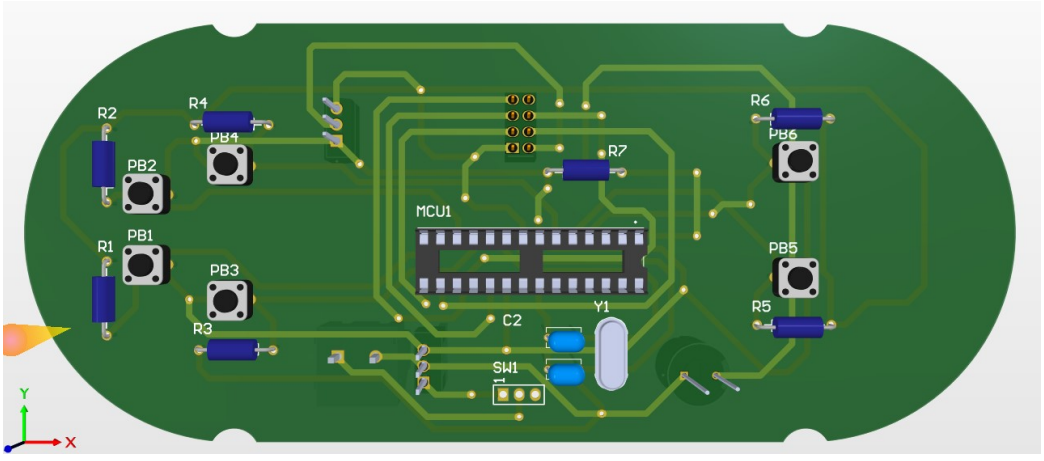


Figura 70: Cara superior de modelo 3D de PCB para rediseño de control remoto

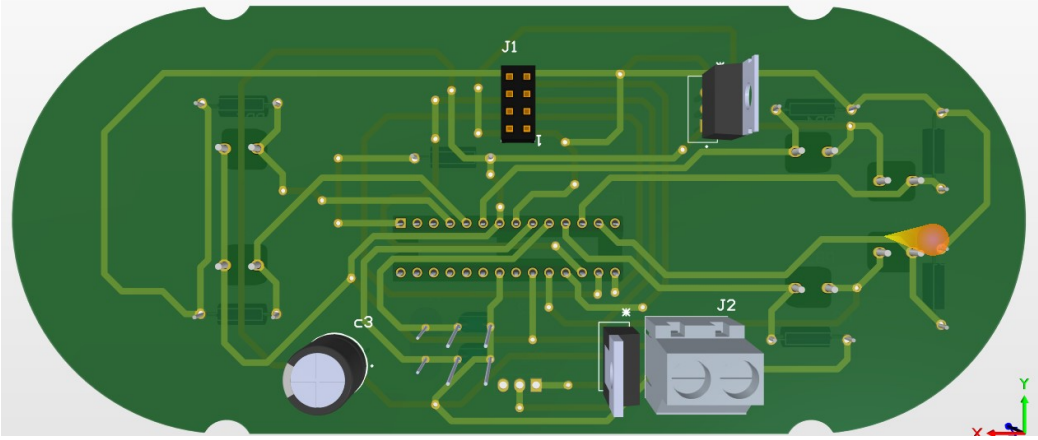


Figura 71: Cara inferior de modelo 3D de PCB para rediseño de control remoto

El nuevo layout esta hecho de manera que los botones que se encuentran a la derecha del control serán los encargados de mover el motor derecho del robot. Mientras que los botones que están en la parte del medio del control están encargados de control el avance y retroceso del motor izquierdo. El último par de botones que se encuentra en el extremo izquierdo del control serán los encargados de levantar o bajar las aspas del robot.

Una vez verificado el correcto rediseño de la nueva PCB se procedió a enviarla para su fabricación. Una vez recibida la nueva placa el proceso para la soldadura y verificación de la correcta conexión de los componentes fue el mismo que se realizó para los dos prototipos anteriores.

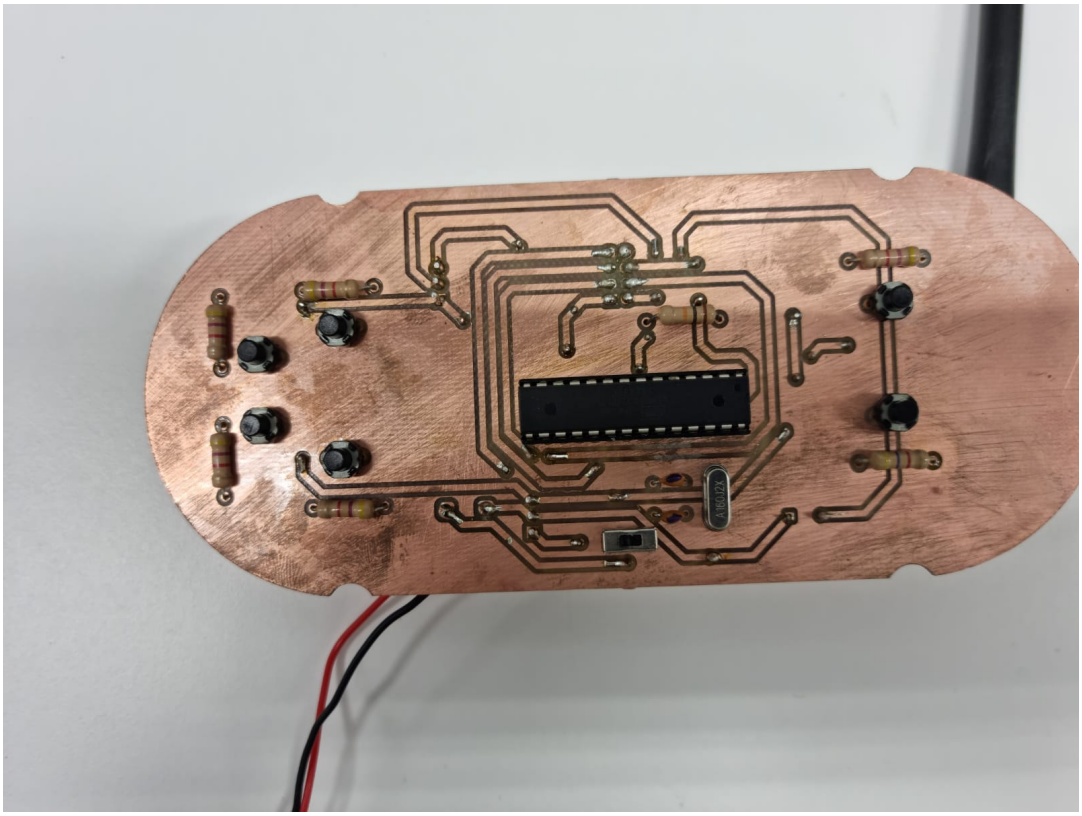


Figura 72: Cara superior de rediseño de control remoto fabricada y soldada

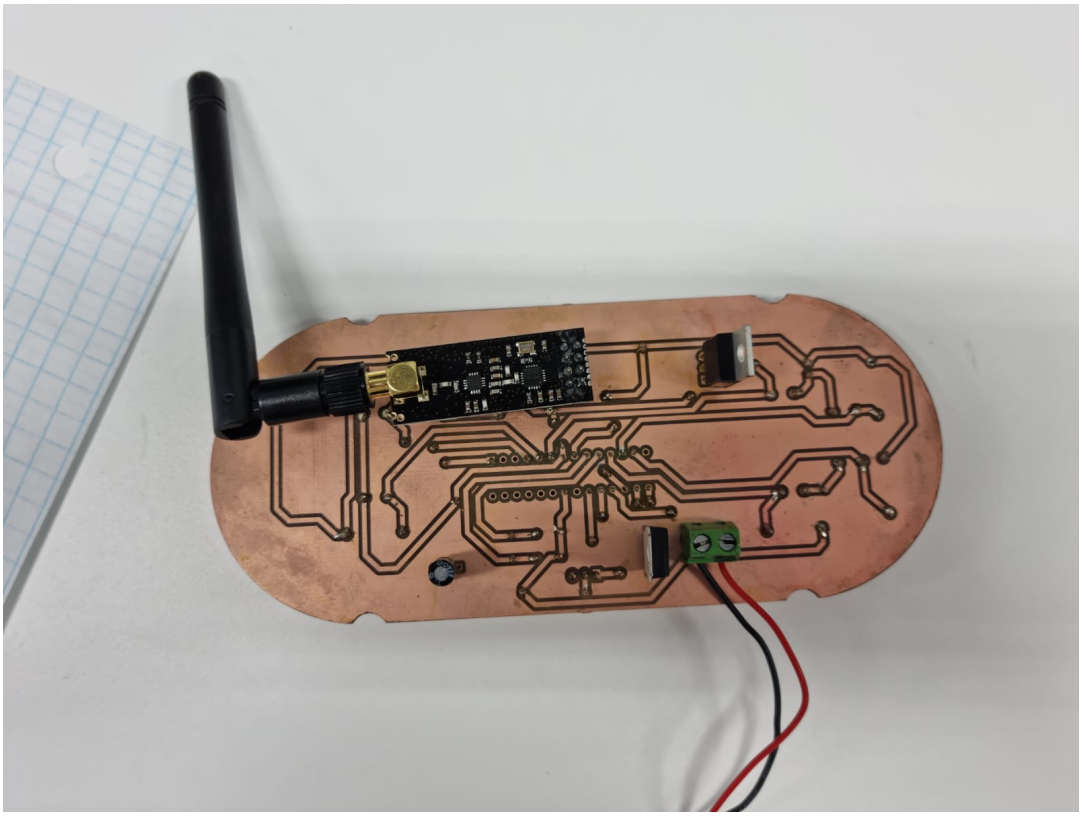


Figura 73: Cara inferior de rediseño de control remoto fabricada y soldada

Ya que se tenía verificada la correcta soldadura y conexión de los componentes se procedió a realizar pruebas de funcionamiento. La primera prueba que se realizó fue puramente de conexión entre el módulo de radiofrecuencia que tiene el control y el módulo de radiofrecuencia que se encuentra dentro del robot principal. Esta prueba consistió en enviar un arreglo de 5 posiciones con los números del 1 al 5 en ASCII para que estos fueran desplegados. Para poder verificar que los datos estaban siendo recibidos por el robot se implemento el monitor serial para poder desplegar todos los datos recibidos por el robot.

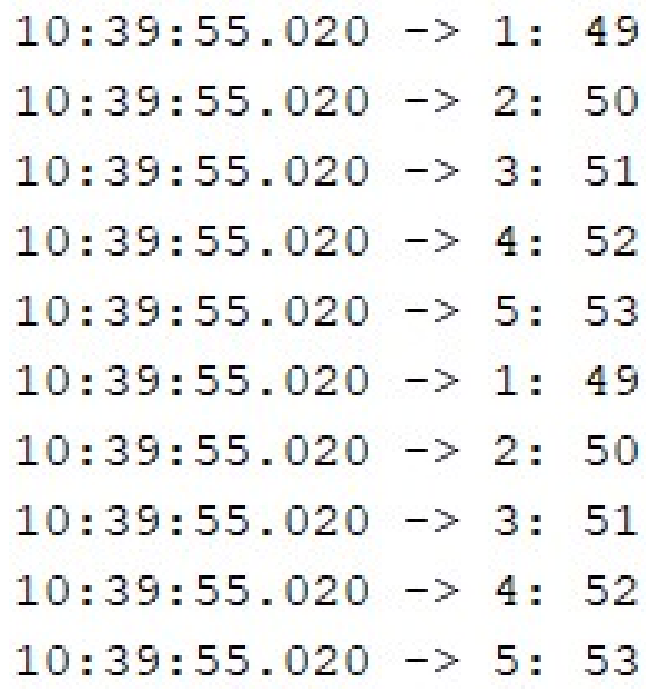


Figura 74: Datos recibidos y desplegados en el monitor serial del control remoto rediseñado

Con la primera prueba terminada se comenzó a realizar el código principal para la transmisión de los comandos. En este nuevo código que utilizaría un arreglo de 5 posiciones en las cuales la primera y última posición tendrían caracteres indicando el inicio y el fin del arreglo. Para el dato que se encuentra en la segunda posición se estaría enviando el estado actual del motor izquierdo, para el dato de la tercera posición el estado actual del motor derecho y para el dato que se encuentra en la cuarta posición se enviaría el comando de subida o bajada de las aspas.

Los estados de cada uno de los motores y las aspas se determinaron utilizando condicionales dentro de código de manera que al oprimir cada uno de los botones estos enviarían uno de tres datos. En el caso de los motores si se oprime el botón de avance se enviaría un valor de 20, si es el botón de retroceso se envía un valor de 0 y si no se oprime nada se envía un valor de 10. La misma lógica fue utilizada para los botones de accionamiento de las aspas pero en este caso el botón de subida envía un valor de 2, el de bajada de 1 y si no se oprime un botón se envía el valor de 0.

Para poder verificar que el código fue hecho correctamente se volvió a utilizar el monitor serial del IDE de Arduino para poder desplegar los valores recibidos. En este caso se lograron recibir los valores deseados ya que al no oprimir ningún botón dentro del control se obtenían los valores de 10 para ambos motores y 1 para las aspas. De igual manera al oprimir cada uno de los botones del control remoto se estaban obteniendo los respectivos valores de 20 para el avance, 0 para el retroceso, 2 para subida de aspas y 0 para bajada de aspas.

Al tener completamente asegurado el funcionamiento tanto del nuevo control remoto como del código realizado se decidió implementar el sistema de control remoto dentro del robot principal. Para comenzar con esta implementación primero se cableo toda la parte

electrónica del robot con lo cual se podría revisar dentro del driver principal de los motores si estos se comportarían de la manera deseada.

Con la verificación del driver completada se procedió a terminar de construir el robot para poder moverlo utilizando el control remoto. La primera prueba que se hizo con el robot totalmente construido fue únicamente de movimiento dentro del laboratorio para poder verificar que todos los botones funcionaran correctamente. En esta prueba se logró mover el robot satisfactoriamente, pero los botones de avance y retroceso se encontraban invertidos por lo que sería necesario cambiar esto dentro del código principal.

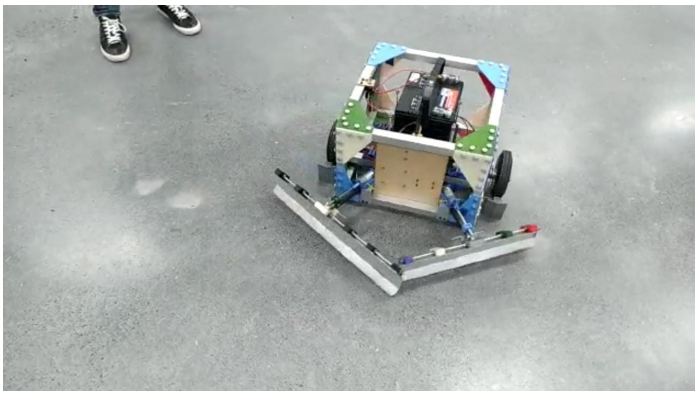


Figura 75: Prueba de movimiento de robot principal dentro de laboratorio

La segunda prueba realizada consistió en utilizar el robot para poder probar su función de movimiento de café para su secado. Por lo que para esta prueba se intentó replicar las condiciones en las cuales estaría trabajando el robot dentro de la finca y para esto se utilizó una módica cantidad de café el cual pudiera mover. Se creó una pequeña capa de café de aproximadamente 5 centímetros en el concreto para que el robot pudiera moverlo y se realizo la primera corrida del robot por este café. En esta primera corrida no se logró movilizar mucho café ya que las aspas se encontraban muy altas por lo que no se podía alcanzar la mayoría del café en el suelo.

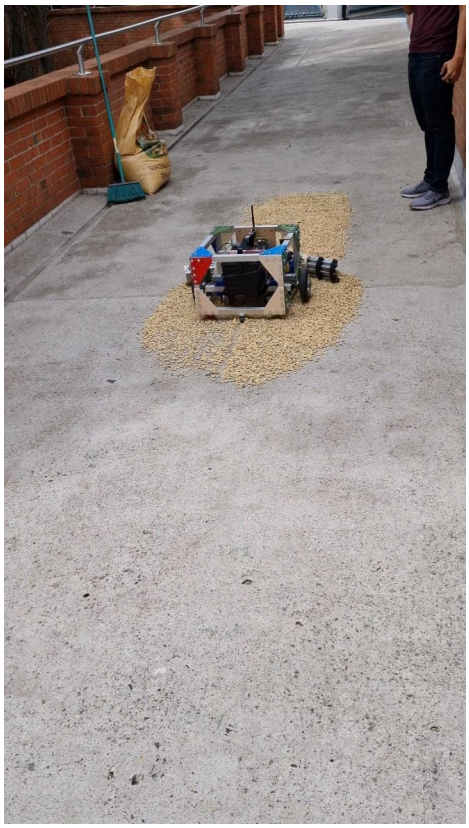


Figura 76: Primera corrida de prueba de movimiento de café con poco café desplazado

Para la segunda corrida del robot se bajo un poco la altitud de las aspas para que estas tuvieran mas contacto con el suelo y que una menor cantidad de café no hiciera contacto. En esta segunda corrida los resultados fueron mucho mas favorables ya que la cantidad de café que no tenía contacto con las aspas disminuyó considerablemente en comparación a la primera corrida. Estos resultados fueron considerados como satisfactorios ya que el objetivo principal del control remoto se cumplió exitosamente al poder controlar las funciones establecidas para el robot principal.

Figura 77: Primera corrida de prueba de movimiento de café con mucho café desplazado

9.2. PCB para cableado interno

La creación de esta pieza se hizo principalmente para poder facilitar todo el cableado interno que debe llevar el robot principal. Para esto se comenzó verificando cuales eran las conexiones necesarias para cada uno de los componentes y así poder agregar componentes nuevos que facilitarían dichas conexiones. Para esto se creó un pequeño diagrama el cual ayudaría a saber que componentes deben ir en la placa y también que voltajes son necesarios para que cada uno de estos componentes pueda operar correctamente.

Al terminar el diagrama se determinó que sería necesario agregar un puente H L293D para poder controlar los motores de las aspas, dos buses de conectores hembra para poder conectar el módulo de radiofrecuencia y el driver. También se notó que sería necesario utilizar varios pines del Arduino MEGA por lo que se optó por agregar varios conectores macho en la forma de los conectores hembra del Arduino para que estos cazaran formando un sistema parecido al que implementan los shields de Arduino.

Tomando esto en cuenta fue necesario crear la placa para que pudiera operar con una entrada de 12 volteos que serían suministrados por la batería principal del robot y que pudiera entregar 5 volteos y 3.3 volteos para cada uno de los componentes. Para poder lograr esto se utilizó el pin "VIN"que posee el Arduino MEGA para ser alimentado con 12 volteos. Debido a que el Arduino MEGA tiene sus propios reguladores de voltaje no fue necesario agregar reguladores extra por lo que se usaron los pines de "5Vƨ "3.3V"que posee el Arduino.

Para poder recibir los comandos del control remoto se implementó un bus de conectores hembra para colocar el modulo de radiofrecuencia. Para que el módulo pudiera funcionar correctamente fue necesario colocar ocho pines de los cuales cinco ya tenían una función fija que en este caso fueron los pines de "SCK", "MOSI", "MISO", "GNDƨ por último el pin de "3.3V". Los últimos dos pines útiles fueron asignados a los pines 48 y 49 del Arduino para que pudieran cumplir con la función de CNS Y CE.

El driver para los motores que sería utilizado en el robot necesitaba 6 pines del Arduino para poder funcionar correctamente y recibir las señales de los motores. Para estos 6 pines se implementó un bus de 6 conectores hembra para poder conectar el driver a la placa para luego realizar las conexiones a los pines del Arduino que fueran necesarios. El driver necesitaba un voltaje de alimentación de 5 volteos por lo que se uso uno de los pines de 5 volteos que posee el Arduino MEGA a igual que con el pin de tierra. Ya que este driver sería utilizado para operar dos motores cada uno de estos motores tenía su propio pin de PWM al igual que si pin ENABLE por lo que se utilizaron los pines PWM del 8 al 11 para estas conexiones.

El puente H implementado fue usado para los motores de las aspas que en este caso fueron dos por lo que en este entrarían las señales de cada una de las patas de los dos motores. Luego se diseño la placa para que enviaría las señales de salida para cada una de las terminales de los motores a dos terminales de tornillo en las cuales estarían conectados los motores. Para las conexiones de las señales de entrada de los motores se utilizaron los pines digitales del 14 al 17 del Arduino MEGA y se alimentó el chip con 5 volteos provenientes de uno de los pines respectivos.

Por último se realizaron las conexiones para los pines necesarios del módulo DMW1001 que tendría el robot. Este módulo únicamente necesitaría 5 volteos, tierra, una conexión al pin TX y una conexión al pin RX del Arduino. El Arduino MEGA cuenta con tres pines TX y con tres pines RX pero en este caso únicamente se utilizó el primer par de estos pines siendo estos los pines 18 y 19. Con estas conexiones terminadas fue posible comenzar con el diseño del esquemático de la placa y para esto se utilizó Altium Designer debido a que facilitaría la creación del esquemático y de la placa en general.

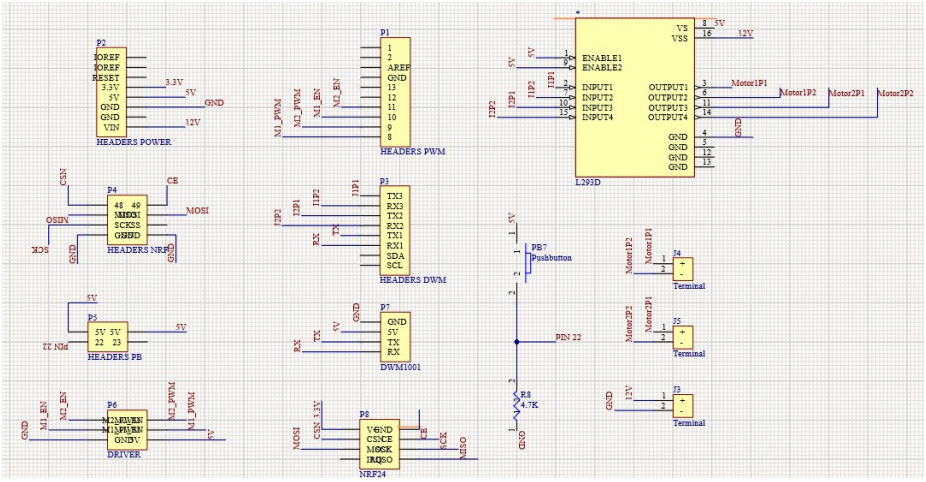


Figura 78: Esquematico para placa de cableado interno

Con el esquemático terminado se pudo proceder con el diseño de la placa por lo que únicamente fue necesario importar los componentes utilizado en el esquemático para poder organizarlos dentro de la placa. Para el diseño de la placa fue necesario tener las medidas exactas del Arduino MEGA ya que los conectores macho que se utilizaron debían estar completamente alineados con los conectores hembra del Arduino. Al haber colocado los conectores macho en las posiciones correctas según las medidas utilizadas se prosiguió con

la organización del resto de los componentes. Las terminales de tornillos utilizadas fueron colocadas en la parte baja de la placa para poder minimizar la distancia entre estas y los motores ya que el tamaño del cable de los motores no era muy largo.

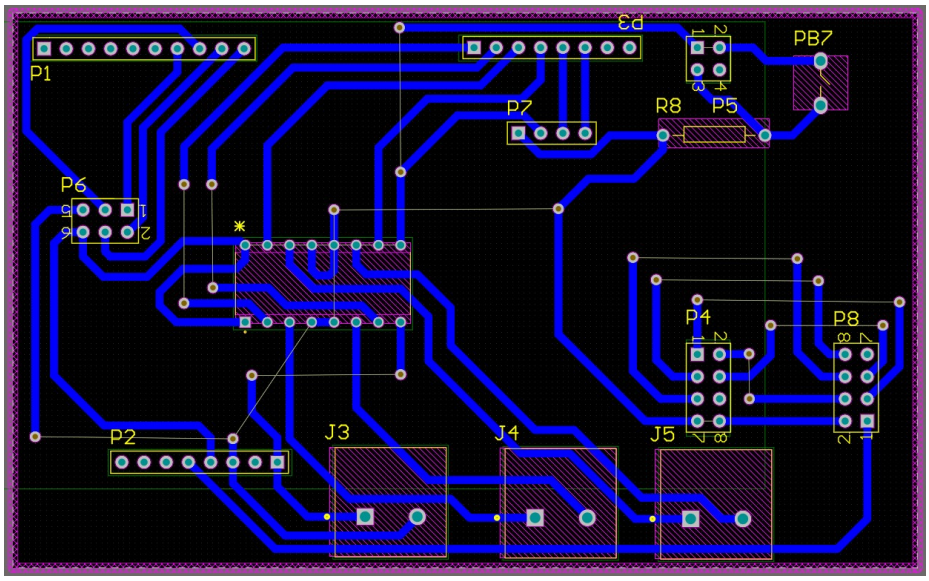


Figura 79: Diseño de placa para cableado interno

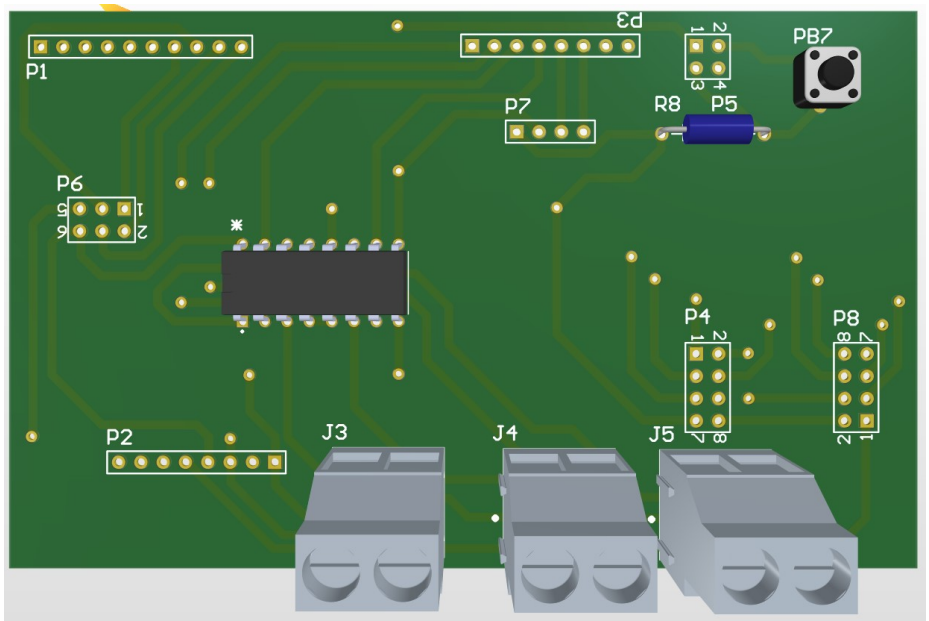


Figura 80: Modelo 3D de placa para cableado interno

Al verificar que las conexiones creadas dentro del diseño de la placa fueron las correctas se crearon los archivos de fabricación para poder comenzar con la fabricación de la placa. Al tener la placa fabricada se siguieron los mismos pasos que en el control remoto para su construcción. Por lo que al terminar la soldadura de los componentes se verificó la continuidad de los componentes utilizando un multímetro.

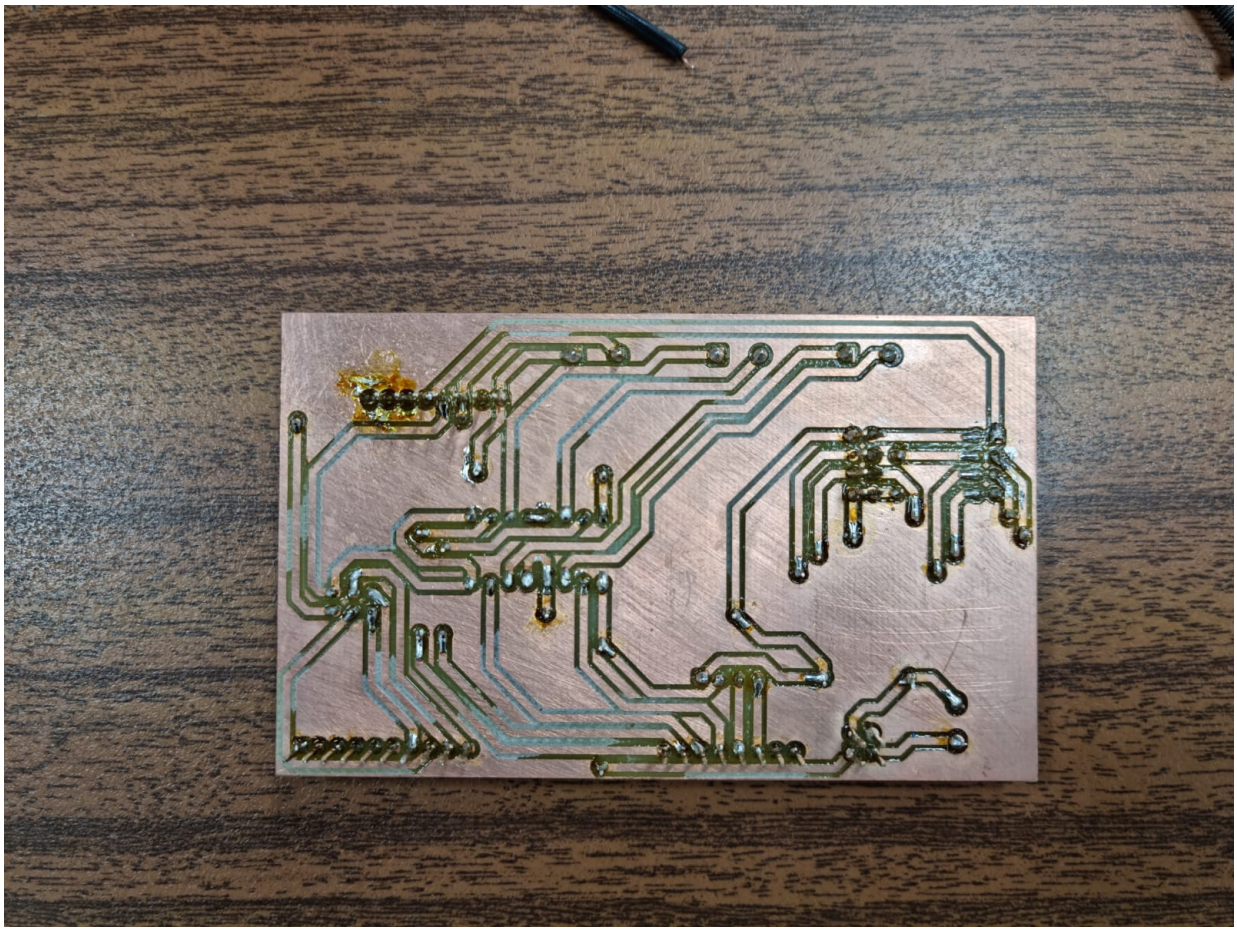


Figura 81: Cara inferior de placa para cableado interno fabricada

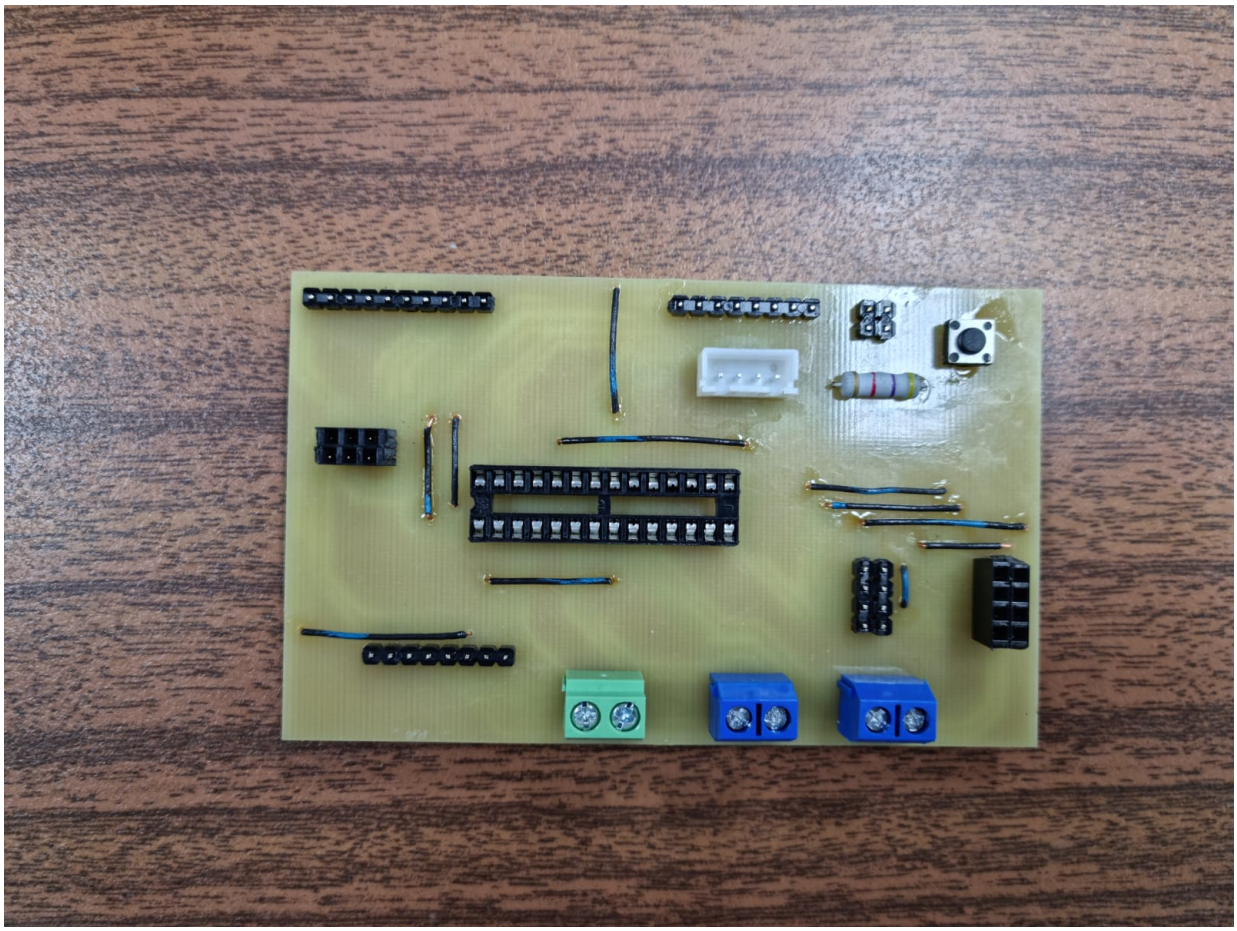


Figura 82: Cara superior de placa para cableado interno fabricada

Antes de comenzar con en ensamblado de la parte electrónica del robot se realizaron varias verificaciones con la placa para estar seguros que esta funcionaria. Primero se verificó que la placa pudiera encajar correctamente con el Arduino MEGA y al lograr una buena conexión sin necesidad de forzar la placa se conecto el módulo DWM1001, el driver y el modulo de radiofrecuencia en sus respectivos buses de conectores. Esto para poder verificar que estas partes funcionaban correctamente al utilizar la placa en vez de conectarlos al Arduino directamente.

Una vez se tuvieron todos los componentes conectados correctamente en sus respectivos buses se alimento la placa con 12 volteos y se comenzaron las pruebas. La primera prueba que se realizó fue la misma prueba mencionada en la subsección del control remoto para la verificación del recibimiento de datos, pero esta vez utilizando la nueva placa. Al confirmar el correcto funcionamiento del modulo de radiofrecuencia se agrego el driver para verificar que este también estaba funcionando correctamente con la nueva placa. Por lo que al recibir un comando del control remoto se debían encender las luces del driver según el comando, para los comandos de avance se encenderían las luces azules, para el comando de retroceso se encenderían las luces blancas y si no se recibe comando ninguna luz se encendería.

Con los resultados obtenidos en la última prueba se determinó que la placa funcionaba

correctamente por lo que se comenzó con el ensamblaje del robot principal. Al tener el robot completamente ensamblado se realizaron las pruebas de funcionamiento del robot mencionadas en la subsección del control remoto con lo cual se obtuvieron resultados favorables y se pudo cumplir con el objetivo principal de la elaboración de la placa.

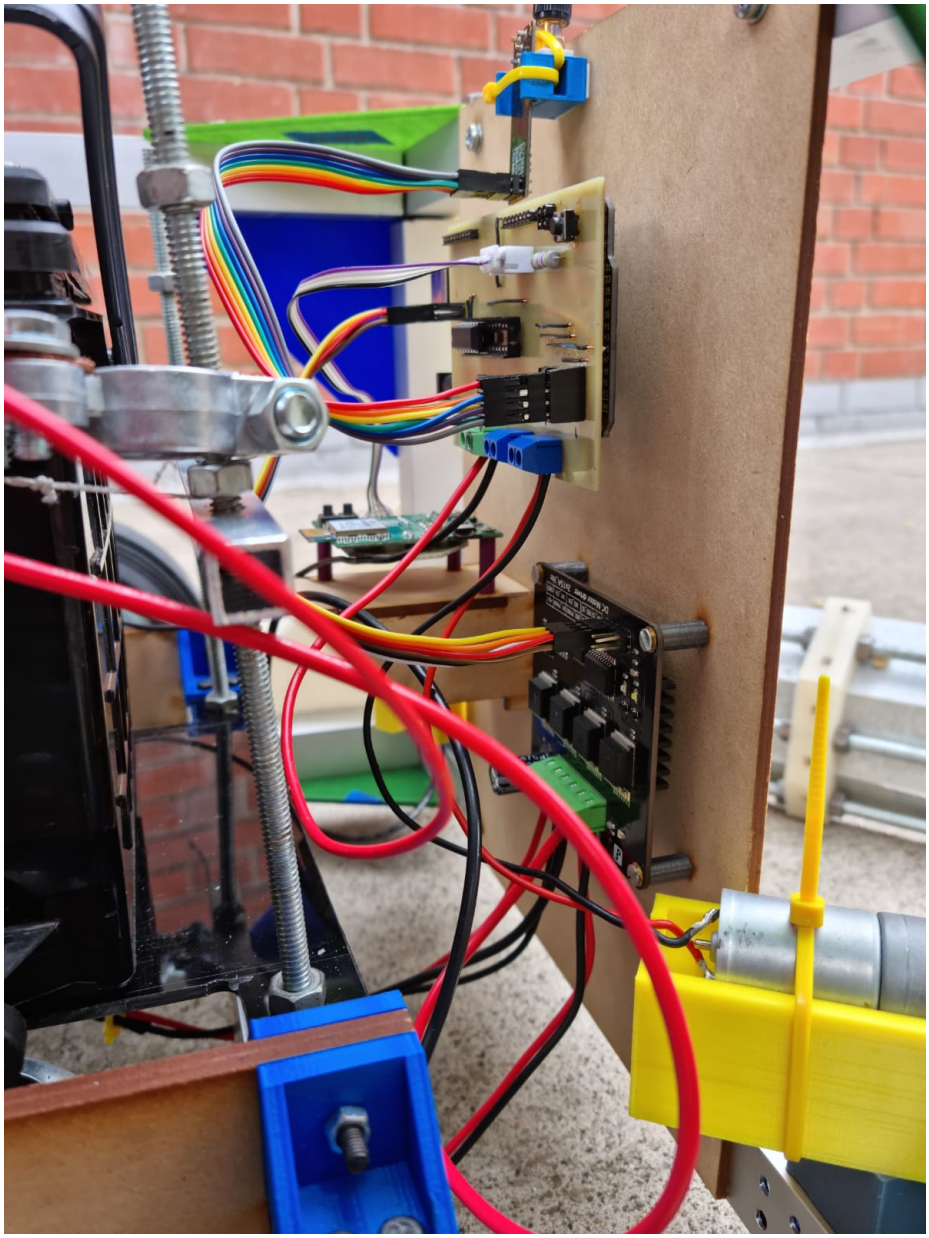


Figura 83: Montaje de placa dentro del robot principal

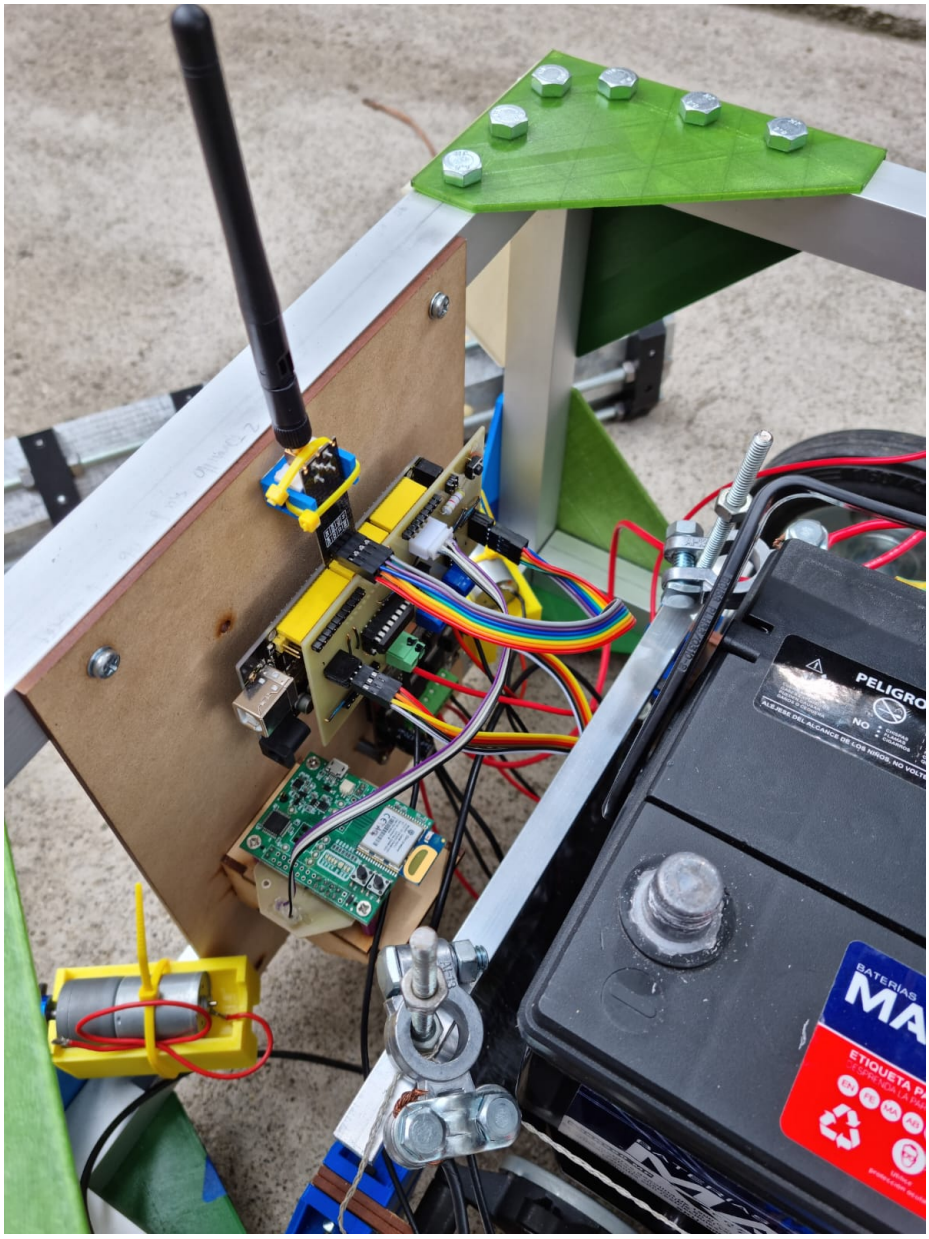


Figura 84: Montaje de placa dentro del robot principal

CAPÍTULO 10

Conclusiones

- Se construyeron e implementaron dos controles remotos compactos utilizando módulos de radio-frecuencia.
- Se implementó una aplicación web dinámica/fluida utilizando el servicio Node-Red en la que se visualiza información relevante del robot.
- Se implementó un sistema de comunicación MQTT vía Wi-Fi entre un microcontrolador ESP32 y un broker Mosquitto corriendo en una Raspberry Pi.
- Se implementó una PCB para la interconexión de módulos de la electrónica interna del robot secador de café.

CAPÍTULO 11

Recomendaciones

Durante la realización de este trabajo de graduación hubo múltiples inconvenientes con los módulos de comunicación RF. Uno de los principales problemas fue que los módulos se dañaban con mucha facilidad. Uno de ellos incluso dejó de transmitir datos y únicamente era capaz de recibirlos. Se recomienda adquirir módulos de un fabricante con alta calidad así como manejar los módulos con una banda antiestática en todo momento. Otro problema encontrado durante la implementación de los módulos se dio al realizar pruebas de alcance. Ya que fue necesario tener una línea de vista directa entre los módulos para asegurar que la señal no se vería afectada.

Al momento de realizar las pruebas de campo con el control remoto se podría realizar una encuesta a los operarios para poder obtener retroalimentación sobre este. Según estos resultados se recomienda rediseñar el control remoto para que implementar mejoras que ayuden a su funcionamiento. Algunas de las mejoras que se recomiendan implementar en la parte electrónica son la inclusión de una pantalla o LEDs para despliegue de información del estado del control. También se recomienda implementar un módulo de carga que utilice tecnología micro-USB o USB-C. Para la parte física del control se recomienda investigar diferentes procesos industriales de manufactura para poder fabricar la carcasa del control. De esta manera se podrá tener un control más industrial que pueda ser implementado en más pruebas de campo sin problemas.

Durante la realización del sistema de recopilación de datos se logró crear la aplicación móvil y web, pero estas se vieron limitadas por la librería utilizada dentro de Node-Red. Se recomienda la implementación de otras librerías como UiBuilder dentro de Node-Red o el uso de otros frameworks para desarrollo de frontend de aplicaciones tales como React.JS o Angular. También se recomienda dentro de esta implementación la adición de opciones en las aplicaciones para la descarga directa de estadísticas en formatos populares como csv o pdf.

Dentro de la fase de pruebas de campo del proyecto se realizó la implementación de una PCB para interconexión de módulos del robot. Debido a escases de tiempo el diseño realizado no fue el más óptimo en términos de uso de espacio y conexiones. Se recomienda rediseñar

esta PCB para que mejorar estos defectos y también para poder asegurar la buena conexión con el resto de los módulos. Ya que al momento de realizar el montaje se tuvieron problemas con las terminales de tornillo debido a que estas eran muy pequeñas para el calibre de cable que era necesario usar debido a los requisitos de corriente. También se tuvo el problema de la distancia de conexión de los motores a las terminales de tornillo. Para poder realizar estas conexiones se tuvo que mover el robot completo para tener acceso a estas terminales. En base a esto se recomienda que dentro del rediseño de la PCB se tenga en cuenta esto para facilitar el acceso entre los motores y las terminales de tornillo.

CAPÍTULO 12

Bibliografía

[1] R. Bhardwaj, "The Ergonomic Development of Video Game Controllers," *Journal of Ergonomics*, vol. 07, n.º 04, ene. de 2017. DOI: 10.4172/2165-7556.1000209.

[2] T. H. Team, *Car-Sharing Application relies on HiveMQ for Reliable Connectivity.* dirección: https://www.hivemq.com/case-studies/bmw-mobility-services/.

[3] R. Sobot, *Wireless Communication Electronics Introduction to RF Circuits and Design Techniques*. Springer International Publishing, 2021.

[4] *RF Wireless Technology.* dirección: https://www.mouser.com/applications/rf-wireless-technology/.

[5] *What Is Wi-Fi? - Definition and Types.* dirección: https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html.

[6] *nRF24L01 single chip transceiver*, mar. de 2008.

[7] *atmega328p datasheet*, 2015.

[8] *JavaScript.* dirección: https://developer.mozilla.org/en-US/docs/Web/JavaScript.

[9] *React – A JavaScript library for building user interfaces.* dirección: https://reactjs.org/.

[10] V. Lampkin, W. T. Leong, L. Olivera, S. Rawat, N. Subrahmanyam, R. Xiang y M. Keen, *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry*. IBM, 2012.

[11] HiveMQ. dirección: https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment/.

[12] *What is a database?* Dirección: https://www.oracle.com/database/what-is-database/.

[13] J. M. Blackstone y P. Johnson, "Size , strength and physical exposure differences between adult and child computer users," 2006.

[14] I. Halim, R. Z. Radin Umar, M. S. Syed Mohamed, N. Ahmad, V. Padmanathan y A. Saptari, "The Influence of Hand Tool Design on Hand Grip Strength: A Review," *International Journal of Integrated Engineering*, vol. 11, n.º 6, sep. de 2019. dirección: https://publisher.uthm.edu.my/ojs/index.php/ijie/article/view/3802.