Integrantes del Proyecto:

- Adell, Nicolas Fabian
- De Blasi, Luca
- Diaz Melion, Danilo Sebastian
- Gil Soria, Ian Lucas
- Montenegro, Luciano Nahuel
- Sojka, Santiago Alejandro

Índice

1. Introducción

BIAS es un módulo aplicable a una silla de ruedas que permite a los usuarios dirigirla mediante el sus pensamientos, eliminando la necesidad de controles físicos tradicionales y optimizando la experiencia de movilidad. Este proyecto representa un avance significativo en el campo de la movilidad asistida, con el objetivo de mejorar la independencia y la calidad de vida de las personas con movilidad reducida.

Este sistema se basa en la avanzada tecnología de interfaz cerebro-computadora (BCI), la cual captura las señales neuronales del usuario y las traduce en comandos precisos para maniobrar la silla de ruedas sin esfuerzo. Además, el diseño modular y adaptable del sistema permite su integración en diversos modelos de sillas de ruedas y su personalización según las necesidades específicas de cada usuario.

La seguridad es una prioridad fundamental en este desarrollo, por lo que el módulo incorpora múltiples capas de redundancia y verificación de señales, asegurando que la silla de ruedas responda de manera precisa y confiable a las intenciones del usuario.

Aspiramos a revolucionar la forma en que las personas con movilidad reducida interactúan con su entorno, brindándoles una herramienta que amplía su capacidad para controlar su vida de manera más segura e independiente.

1.1. Resumen del proyecto

Este proyecto se centra en el desarrollo de una silla de ruedas controlada mediante señales cerebrales. El sistema se compone principalmente de los siguientes componentes: la silla de ruedas, un dispositivo EEG (electroencefalograma) y dos motores que permiten al usuario moverse en la dirección deseada.

El funcionamiento del sistema es el siguiente: el usuario se sienta en la silla y se coloca el EEG en la cabeza. A continuación, el usuario piensa en la dirección en la que desea moverse, y los motores responden en consecuencia.

El EEG tiene la función de leer las señales cerebrales del usuario. Estas señales se clasifican en las categorías de alpha, beta, gamma, delta y theta, y luego se someten a un proceso de filtrado para eliminar cualquier ruido. Posteriormente, las señales filtradas son analizadas por una Inteligencia Artificial (IA) desarrollada por nuestro equipo, la cual identifica patrones para determinar si el usuario desea moverse o detenerse.

Además, la silla está equipada con un sistema de emergencia que se activa en caso de lecturas erróneas o la detección de obstáculos. Este sistema de emergencia está compuesto por sensores infrarrojos y ultrasónicos que detectan la presencia de obstáculos frente a la silla.

1.2. Motivación

Como estudiantes de séptimo año, buscamos desarrollar un proyecto con el fin de cumplir con el requerimiento horario de practicas profesionalizantes. Nuestra intención inicial fue encontrar el proyecto ideal en cuestiones de impacto social, enfocándonos en la mejora de la calidad de vida de aquellas personas con discapacidades. Bajo este marco, y tras una extensa investigación, proponemos crear un módulo aplicable a una silla de ruedas controlado mediante la actividad cerebral del usuario, utilizando tecnología de interfaces cerebro-computadora (BCI), para brindarles mayor autonomía, independencia y participación social. Creemos que este proyecto tiene el potencial de transformar la vida de las personas con movilidad motora reducida debido a enfermedades como ELA, esclerosis múltiple o cuadriplejia, permitiéndoles controlar su propio movimiento y mejorar significativamente su calidad de vida.

1.3. Objetivos

El objetivo de este proyecto es desarrollar una silla de ruedas motorizada que pueda ser controlada por la actividad cerebral del usuario. Esto beneficiaría a las personas con discapacidades motoras severas para lograr independencia y movilidad, mejorando su calidad de vida y participación en la

sociedad. De esta manera, estas personas las cuales no pueden caminar o tienen movilidad reducida, pueden gozar de la utilización de una silla de ruedas controlada por señales cerebrales sin la utilización de ambas manos y meramente por sus pensamientos, dándole libertad y autonomía al usuario.

2. ¿Quiénes somos?

Somos el grupo BIAS (Brain Intelligence Artificial System), conformado por seis estudiantes de la especialidad de Aviónica del séptimo año, segunda división, comisión C, de la Escuela Técnica N° 7 IMPA "Taller Regional Quilmes". Nuestro equipo se encuentra comprometido en el desarrollo de un módulo adaptable para sillas de ruedas, para así poder brindarles mas independencia y movilidad.

2.1. Contactos

En este apartado se detallarán los contactos a los integrantes del proyecto, adjuntando enlaces a sus redes:

2.1.1. Adell Nicolás

Instagram
LinkedIn
Gmail

2.1.2. De Blasi Luca

Instagram
LinkedIn
Gmail

2.1.3. Diaz Melión Danilo

Instagram
LinkedIn
Gmail

2.1.4. Gil Soria Ian

Instagram
LinkedIn
Gmail

2.1.5. Montenegro Luciano

Instagram
LinkedIn
Gmail

2.1.6. Sojka Santiago

Instagram
LinkedIn
Gmail

3. Desarrollo del Proyecto

En este capítulo se expondrá en detalle la metodología empleada para avanzar en el desarrollo del proyecto, incluyendo las fuentes de inspiración y los proyectos similares que se tomaron como referencia para el diseño de nuestros circuitos y la programación de los sistemas. Además, se incluirán imágenes ilustrativas de estos proyectos de referencia, con el fin de ofrecer una comprensión visual más completa. Al finalizar el capítulo, se presentarán y explicarán los resultados obtenidos, evaluando su eficacia en relación con los objetivos planteados.

3.1. Metodología

La metodología aplicada en este proyecto se basa en la organización del equipo en dos grupos especializados. El primer grupo se encarga del desarrollo de los programas, siendo ejemplos de esto los filtros virtuales, programas de recepción y procesado, mientras que el segundo se dedica a la implementación de los motores, sistemas de emergencia y filtros. Esta división del trabajo permite una utilización más eficiente del tiempo, al abordar simultáneamente diversas áreas del proyecto, lo que resulta en un avance más significativo en periodos de tiempo más reducidos.

3.2. Historia del Arte

En este capitulo se repasa todos los conceptos previamente vistos mediante los cuales pudimos empezar a conceptualizar la idea del proyecto.

3.2.1. Neuralink

Uno de los casos que nos permitió aproximarnos a la conceptualización del proyecto fue el chip cerebral estilo implante de Neuralink, denominado Telepathy. Este dispositivo, desarrollado por la compañía de Elon Musk, tiene como objetivo facilitar la vida de las personas mediante la optimización de herramientas y su adaptación al cuerpo humano a través del implante. El chip crea una conexión entre el ser humano y dispositivos electrónicos, permitiendo, por ejemplo, controlar una puerta con traba eléctrica, desbloquear un teléfono móvil, e incluso monitorear el estado de ánimo de las personas.

Este ejemplo nos ofreció una perspectiva sobre la viabilidad de desarrollar un dispositivo, que, aunque no tan complejo, pudiera ser controlado por el propio cuerpo humano. A partir de esta premisa, iniciamos una investigación para explorar las posibilidades y opciones disponibles basadas en esta idea.

3.2.2. Ryan Lopez EEG

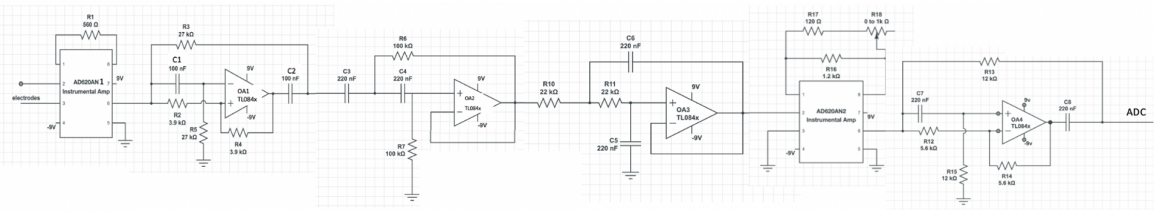
Si bien nuestro motivo y objetivo principal era crear un dispositivo original e innovador, también nos propusimos investigar proyectos con cierta similitud. En este proceso, encontramos el repositorio de un estudiante que desarrolló un sistema capaz de detectar señales alfa y, en función de las reacciones o pensamientos del usuario, permitir el control a través de un test de concentración. Este test consistía en que el usuario debía mantener un estado de concentración mientras estudiaba un documento para un examen, en lo que se podría considerar un simulacro.

El sistema desarrollado por Ryanlopezz activaba una alarma si los niveles de concentración del usuario comenzaban a disminuir, basándose en un espectro de señales alfa transmitidas por un conjunto de electrodos EEG dorados conectados a la cabeza del estudiante. Además, el sistema realizaba otras dos pruebas: una de ellas consistía en escribir en código Morse mediante pensamientos, enviando "pulsos de concentración", y la otra permitía controlar el movimiento de un personaje en el famoso videojuego "Flappy Bird", ajustando el impulso para subir o bajar.

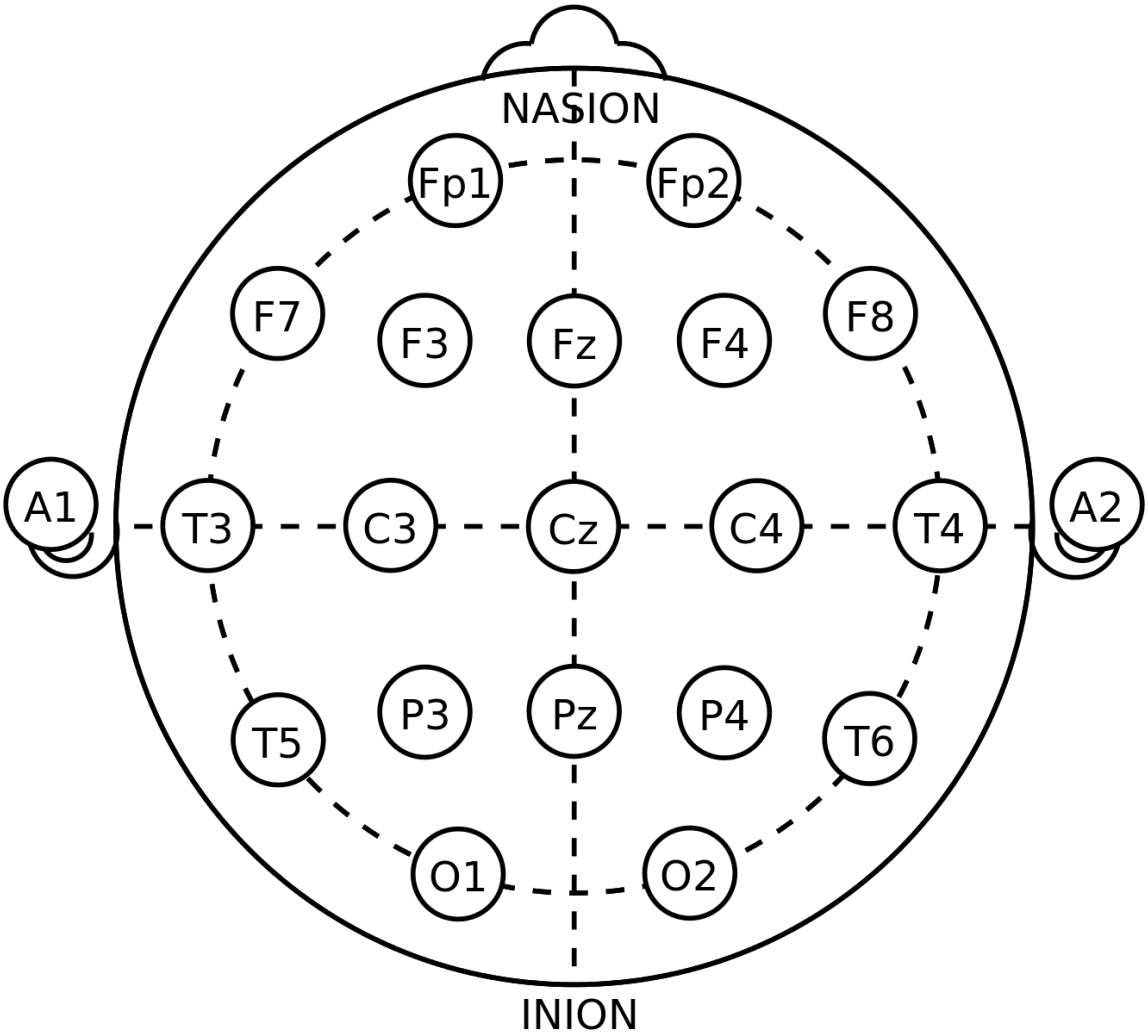
En el siguiente link se adjunta la página de GitHub de Ryan Lopez en donde se puede observar más a detalle los programas y archivos de su EEG.

Ryan Lopez GitHub

Gracias al mismo proyecto realizado por este usuario, pudimos desarrollar nuestro propio circuito para el sistema de filtrado, ya que íbamos a requerir otras especificaciones con diferentes funciones, tales como la entrada de señal: ya que esta debía ser mayor a la designada, al nosotros trabajar no solo con el espectro radioelectrico de las señales Alpha, sino con las principales señales que el cerebro produce al momento de realizar un pensamiento especifico; Por otro lado nosotros íbamos a requerir no solo un sistema de filtros EEG sino 4 placas del mismo circuito, debido a la posibilidad de maniobrar controlando los 3 ejes de los motores de la silla de ruedas



Por otro lado nos dio una primera vista de la localización de los botones EEG como irían colocados en la cabeza

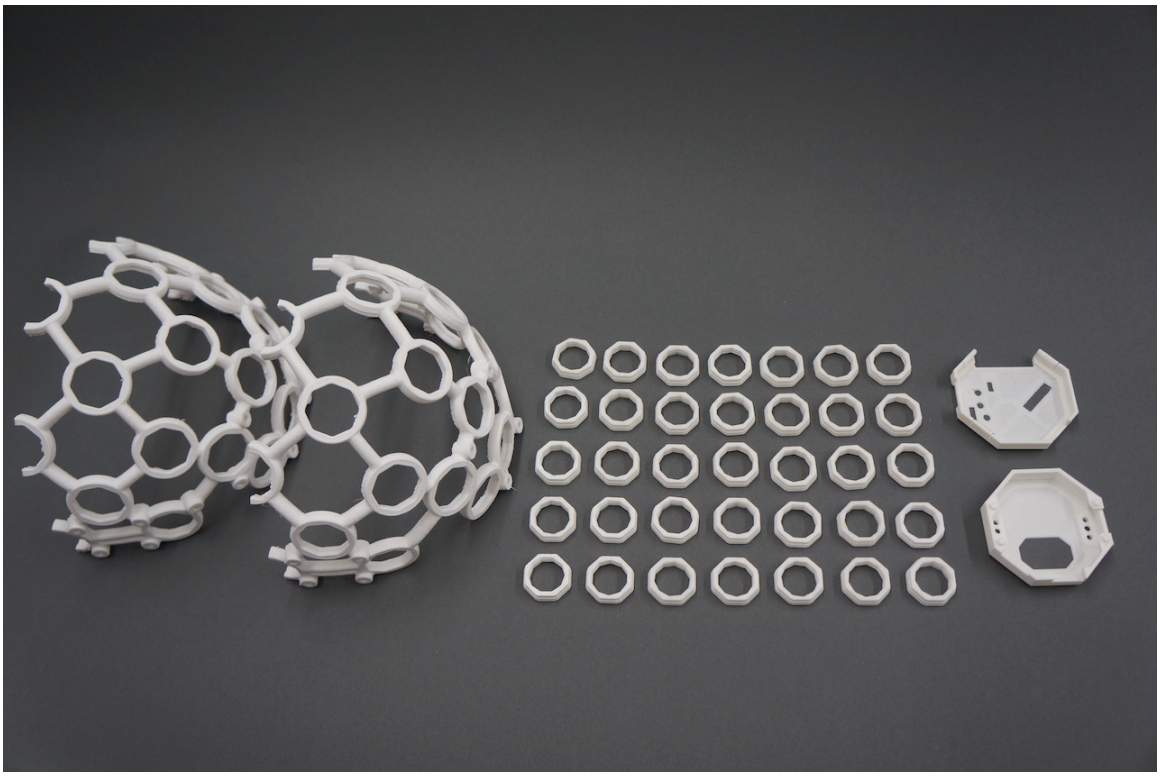


Así mismo para el estudio del aérea de medicina y saber como el cerebro humano se encarga de transmitir ciertas señales eléctricas gracias a la actividad cerebral tuvimos en cuenta los actuales dispositivos de eneflografos, como se componen y actúan gracias ala actividad cerebral, bajo un

especto radioelectrico determinado adjunto a la clasificación de las ondas transmitidas por el cerebro humano

3.2.3. OpenBCI

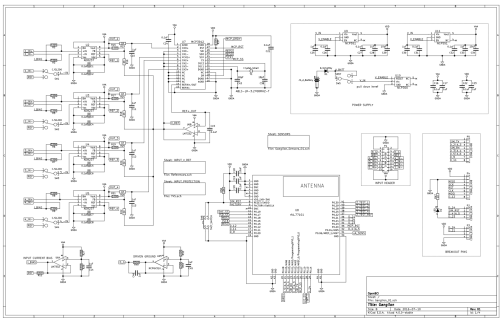
En nuestra búsqueda de conexiones para un microcontrolador y formas de desarrollar nuestro propio proyecto, descubrimos un sitio web llamado OpenBCI, una compañía especializada en la creación de dispositivos que integran tecnología con la interfaz del cuerpo humano. Esta empresa ha desarrollado un encefalógrafo casero, proporcionando dimensiones y parámetros estructurales que nos permitieron diseñar nuestra propia vincha/casco. Este dispositivo incorpora electrodos EEG, adaptados a la cabeza del usuario, como parte fundamental de nuestro proyecto "BIAS".



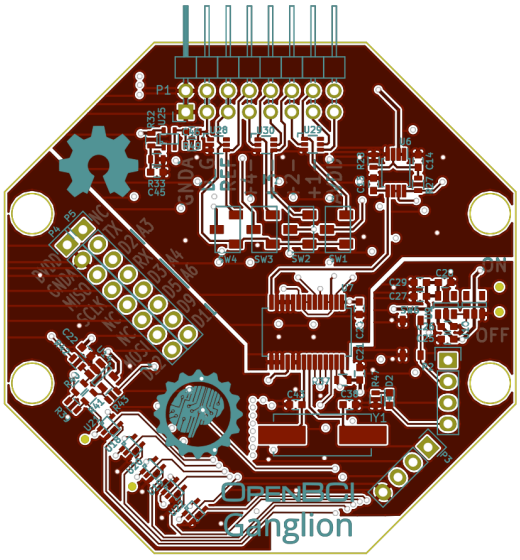
Además, al ser una compañía especializada en la creación de módulos de microcontroladores, nos proporcionaron orientación en cuanto a cómo realizar las conexiones entre la Raspberry Pi 4 que utilizamos y el sistema de filtrado EEG.

En los siguientes textos se adjuntan enlaces para ver la página oficial de OpenBCI y documentación de referencia correspondiente.

Página oficial de OpenBCI
Documentación

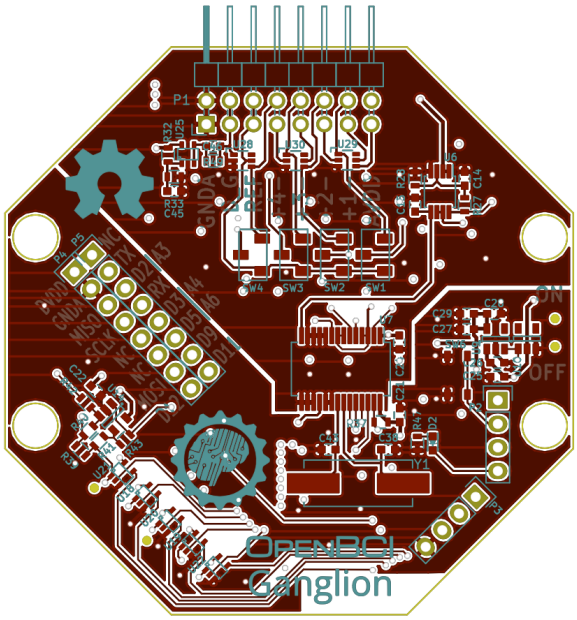


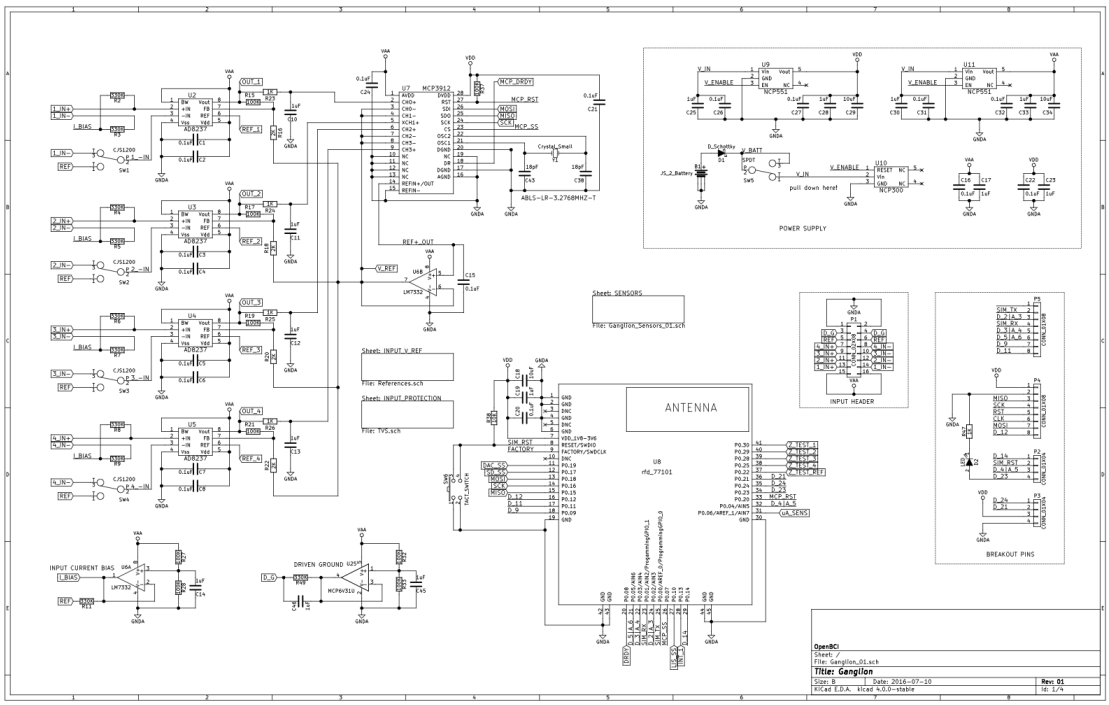
(a) Diseño Esquemático



(b) Diseño PCB

Figura 1: Diseño de microcontrolador de Open BCI





3.2.4. Autodesk Instructables

A partir de esta página, obtuvimos los primeros circuitos necesarios para el EEG, lo que nos permitió avanzar de manera significativa en el desarrollo del proyecto. Esta fuente de información fue fundamental para sentar las bases técnicas y nos brindó las herramientas necesarias para continuar con el diseño y la implementación del sistema.

4. Desarrollo Teórico

En este capítulo se describirán y desarrollarán todos los aspectos teóricos relacionados, justificando la elección de los componentes utilizados. Se abordarán teorías como la Jaula de Faraday y el sistema 10-20, además de ofrecer una explicación detallada sobre los tipos de señales empleadas, entre otros temas relevantes.

4.1. Distinción de alimentaciones

4.2. ¿Que es un electroencefalograma?

Un electroencefalograma (EEG) es una prueba médica que registra la actividad eléctrica del cerebro. Esta actividad se mide a través de electrodos colocados en el cuero cabelludo, que detectan las señales eléctricas generadas por las neuronas mientras se comunican entre sí. Estas señales eléctricas se representan como ondas cerebrales en una gráfica que muestra diferentes frecuencias y patrones.

El EEG se utiliza comúnmente para:

- **Diagnóstico de trastornos neurológicos:** Como la epilepsia, convulsiones, trastornos del sueño, encefalopatías o daño cerebral.

- **Monitoreo del estado cerebral:** En procedimientos quirúrgicos, cuidados intensivos o investigaciones sobre coma y muerte cerebral.
- **Investigación neurológica y cognitiva:** Para estudiar la actividad cerebral en respuesta a estímulos externos, emociones o en procesos de toma de decisiones.
- **Interfaz cerebro-computadora (BCI):** Donde se usa para traducir la actividad cerebral en comandos para controlar dispositivos externos, como en tu proyecto de silla de ruedas controlada por EEG.

Las ondas cerebrales se clasifican en varios tipos según su frecuencia: delta, theta, alpha, beta, y gamma, cada una asociada a diferentes estados de conciencia y actividades cerebrales.

4.3. ¿Que es un EEG?

Un electroencefalógrafo es un dispositivo médico que se utiliza para registrar y medir la actividad eléctrica del cerebro. Su función principal es captar las señales eléctricas generadas por las neuronas cerebrales y transformarlas en gráficos de ondas cerebrales, que se pueden analizar para evaluar la función cerebral.

4.3.1. Componentes que tiene un EEG:

- **Electrodos:** Pequeños sensores que se colocan en el cuero cabelludo para captar los cambios de voltaje producidos por la actividad cerebral.
- **Amplificador:** Como las señales eléctricas del cerebro son muy débiles, el electroencefalógrafo incluye amplificadores que aumentan la señal para que sea lo suficientemente fuerte como para ser registrada.
- **Conversor analógico-digital:** Convierte las señales eléctricas captadas en un formato digital para su procesamiento por computadora.
- **Software de procesamiento:** El dispositivo utiliza un software que transforma las señales en gráficos de ondas cerebrales que se presentan en tiempo real en una pantalla o se registran para análisis posterior.
- **Pantalla/Gráficos:** El electroencefalógrafo muestra las señales eléctricas en forma de ondas, conocidas como ondas cerebrales (Delta, Theta, Alpha, Beta, Gamma), lo que permite a los médicos interpretar la actividad cerebral.

4.4. Sistema 10-20

El sistema 10-20 es un método internacionalmente estandarizado para la colocación de electrodos en el cuero cabelludo durante un electroencefalograma (EEG). Este sistema asegura que los electrodos se coloquen de manera uniforme y precisa en relación con los puntos anatómicos del cráneo, permitiendo una medición consistente y reproducible de la actividad eléctrica cerebral.

El nombre "10-20"proviene de las distancias porcentuales entre los electrodos: Los electrodos se colocan en posiciones que están al 10 % o al 20 % de la distancia total entre puntos anatómicos clave del cráneo. Puntos anatómicos clave de referencia:

- **Nasion:** El punto entre la frente y la nariz.
- **Inion:** El punto más prominente en la parte posterior de la cabeza.
- **Puntos preauriculares:** Justo por encima de los oídos, a cada lado de la cabeza.

Las posiciones de los electrodos se distribuyen sobre la cabeza tomando como referencia estos puntos para cubrir todo el cuero cabelludo de forma equidistante.

4.4.1. ¿Para qué sirve?

El sistema 10-20 se utiliza para garantizar una colocación precisa y consistente de los electrodos en los estudios de EEG. Algunos de sus beneficios incluyen:

- **Estandarización global:** Permite que los estudios de EEG sean comparables en diferentes laboratorios y entre diferentes pacientes.
- **Cobertura completa del cerebro:** El sistema asegura que se cubran adecuadamente diferentes áreas del cerebro (frontal, parietal, occipital, temporal).
- **Detección de anomalías:** Facilita la identificación de regiones del cerebro donde pueden estar ocurriendo fenómenos anormales, como descargas epilépticas, actividad lenta anormal o asimetrías en las ondas cerebrales.

4.4.2. ¿Como se aplica?

En un estudio de EEG, se colocan electrodos en el cuero cabelludo siguiendo estas posiciones predeterminadas en el sistema 10-20:

1. Puntos y nomenclatura: Cada posición de electrodo tiene una nomenclatura que se utiliza para identificar las regiones del cerebro de las que se está registrando la actividad.

- F: Frontal (F3, F4, Fz, etc.)
- C: Central (C3, C4, Cz, etc.)
- P: Parietal (P3, P4, Pz, etc.)
- O: Occipital (O1, O2)
- T: Temporal (T3, T4, etc.)
- Los números impares (1, 3, 5, 7) se asignan al hemisferio izquierdo, mientras que los números pares (2, 4, 6, 8) se asignan al hemisferio derecho. La letra "zïndica posiciones centrales (a lo largo de la línea media de la cabeza).

4.4.3. Pasos para aplicar el sistema 10-20

1. **Medición del cráneo:** Se mide la distancia desde el nasion al inion y desde un punto preauricular al otro. Luego, se calculan los puntos intermedios en un 10 o 20 de esas distancias para determinar las ubicaciones exactas de los electrodos.
2. **Colocación de los electrodos:** Se colocan electrodos en los puntos marcados, asegurando una cobertura uniforme de la cabeza. Esto puede hacerse usando una gorra de EEG con posiciones ya marcadas o mediante la medición manual.
3. **Registro de la actividad:** Una vez colocados los electrodos, se inicia el registro de la actividad cerebral. Los electrodos miden las diferencias de potencial eléctrico entre las regiones del cerebro cubiertas por los electrodos.

4.4.4. Aplicaciones del sistema 10-20

- **Diagnóstico clínico:** El sistema se usa comúnmente en estudios de EEG para diagnosticar condiciones neurológicas como la epilepsia, trastornos del sueño, y daño cerebral.
- **Investigación:** En estudios de neurociencia, el sistema 10-20 facilita la comparación de la actividad cerebral entre sujetos, ya que la ubicación de los electrodos está estandarizada.
- **Interfaces cerebro-computadora (BCI):** En proyectos de BCI, se utiliza el sistema 10-20 para colocar electrodos en áreas específicas del cerebro que controlan movimientos o funciones específicas.

4.5. Comunicación UART

El UART (Universal Asynchronous Receiver-Transmitter) es un periférico de hardware utilizado para la comunicación serie. Su principal función es permitir la transmisión y recepción de datos de manera asíncrona, lo que implica que no requiere de una señal de clock compartida entre el transmisor y el receptor. A continuación, se destacan las características principales de este tipo de comunicación:

- **Comunicación serie:** Los datos se transmiten de forma secuencial, bit a bit, a través de una línea de transmisión (TX) y se reciben mediante una línea de recepción (RX).
- **Asincronía:** Al no utilizar una señal de clock constante, ambos dispositivos deben estar configurados con la misma velocidad de transmisión (tasa de baudios) para asegurar la correcta recepción de los datos.
- **Formato de datos:** Cada paquete de datos transmitido generalmente incluye un bit de inicio (indicando el comienzo de la transmisión), entre 5 y 9 bits de datos, un bit opcional de paridad (para detección de errores), y uno o dos bits de parada.
- **Velocidad de transmisión:** La velocidad de transmisión de la información, conocida como tasa de baudios, se refiere al número de bits transmitidos por segundo.

La implementación de la comunicación UART en nuestro proyecto se utiliza para la recepción y procesamiento de señales. Específicamente, en la RP2040 Zero, enviamos las lecturas de los cuatro canales del ADC (Conversor Analógico-Digital) en formato JSON durante un período de tiempo definido y con una frecuencia de muestreo específica a la Raspberry Pi 4.

4.6. Gráfico de Fourier

El gráfico de Fourier está basado en la Transformada de Fourier (TF), que es una operación matemática que descompone una señal en sus componentes sinusoidales de diferentes frecuencias. Este método es especialmente útil para analizar señales periódicas y no periódicas.

La Transformada de Fourier dicta que:

Dada una señal *x(t)* en el dominio del tiempo, la transformada de Fourier *X(f)* se define como:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

Figura 2: Formula de Transformada de Fourier

Donde:

- $x(t)$ es la señal en el dominio del tiempo.
- $X(f)$ es la señal transformada en el dominio de la frecuencia.
- f es la frecuencia (Hz).
- j es la unidad imaginaria
- $e^{-j2\pi ft}$ es la función compleja que representa la descomposición en sinusoides.

El gráfico de Fourier representa la magnitud o la fase de la transformada de Fourier. Este gráfico tiene dos componentes importantes:

1. **Espectro de Magnitud:** Se representa gráficamente la magnitud de cada frecuencia presente en la señal. En el eje horizontal está la frecuencia *f*, y en el eje vertical está la magnitud *|X(f)|*, que indica cuánta energía o potencia de la señal está concentrada en esa frecuencia específica.

$$|X(f)| = \sqrt{Re(X(f))^2 + Im(X(f))^2} \tag{1}$$

2. **Espectro de Fase:** Indica el desfase de cada frecuencia con respecto a la señal original. En el eje horizontal también está la frecuencia *f*, y en el eje vertical está el ángulo de fase $\theta(f), quepuedecalcularsecomo$:

$$\theta = tan^{-1}\left(\frac{Im(X(f))}{a}\right) \tag{2}$$

4.6.1. ¿Para que sirve?

El grafico de Fourier es una herramienta que sirve para analizar señales en el dominio de la frecuencia.

1. **Análisis de frecuencias:**
   - **Descomposición de señales complejas:** Permite descomponer una señal en sus componentes de frecuencia individuales. Esto es útil cuando quieres saber qué frecuencias están presentes en una señal y su importancia relativa.
   - **Identificación de frecuencias dominantes:** El gráfico te muestra cuáles son las frecuencias que más contribuyen a la señal original, lo que es esencial para entender fenómenos periódicos y patrones.
2. **Filtrado de señales:**
   - **Diseño de filtros:** Si necesitas eliminar ruido o componentes no deseados de una señal, el gráfico de Fourier ayuda a identificar las frecuencias del ruido o interferencia. Luego puedes diseñar un filtro que elimine esas frecuencias.
   - **Filtrado de bandas:** Puedes aplicar filtros pasa-bajas, pasa-altas o de banda (pasa-banda o elimina-banda) a una señal. El gráfico te muestra exactamente qué frecuencias están siendo afectadas.
3. **Compresión de datos:**
   - **Eliminación de información redundante:** En algunas señales (como imágenes o audio), las frecuencias más altas o muy bajas pueden no ser necesarias. Usando el gráfico de Fourier, puedes identificar esas frecuencias y eliminarlas para comprimir los datos sin perder información relevante.
   - **Compresión de audio e imágenes:** Muchos algoritmos de compresión (como JPEG o MP3) utilizan la Transformada de Fourier o versiones discretas de ella para eliminar componentes de alta frecuencia que son menos perceptibles para el ser humano.
4. **Análisis de vibraciones y ondas:**
   - **Detección de fallos en maquinaria:** En ingeniería mecánica y de control, el gráfico de Fourier es utilizado para analizar vibraciones y detectar problemas como desbalances, resonancias, o desgastes en partes móviles. Las frecuencias anómalas pueden señalar fallos.
   - **Análisis de señales acústicas:** Permite estudiar el espectro de frecuencias en señales de audio. Por ejemplo, para identificar las características de sonidos musicales o voces.
5. **Procesamiento de señales EEG:**

- Para el análisis de señales cerebrales (EEG), el gráfico de Fourier es vital para identificar y separar las diferentes bandas de frecuencias asociadas con diferentes actividades cerebrales (como las ondas delta, theta, alpha, beta y gamma).
- **Monitoreo de actividad cerebral:** En neurociencia, el análisis de Fourier se utiliza para analizar patrones de actividad cerebral, detectar anormalidades y realizar interfaces cerebro-máquina (como el control de la silla de ruedas en tu proyecto).

6. **Análisis de señales en telecomunicaciones:**
   - **Ancho de banda:** El gráfico de Fourier permite determinar el ancho de banda necesario para transmitir una señal sin pérdida de información.
   - **Modulación y demodulación:** En sistemas de comunicaciones, la modulación de señales se realiza en el dominio de la frecuencia. El gráfico de Fourier te permite visualizar cómo una señal ha sido modulada para su transmisión y cómo se puede demodular para su interpretación.
7. **Análisis de imágenes:**
   - **Reconocimiento de patrones:** En procesamiento de imágenes, la transformada de Fourier se utiliza para identificar patrones repetitivos o estructuras de frecuencia en una imagen. Es útil en tareas como la detección de bordes o compresión de imágenes.
   - **Eliminación de ruido:** Permite eliminar patrones específicos de ruido en imágenes filtrando frecuencias indeseadas.

4.6.2. ¿En que lo aplicamos?

Se emplea el Análisis de Fourier para identificar el espectro de frecuencias de las señales cerebrales, permitiendo la diferenciación de las ondas alpha, beta, gamma, delta y theta. Posteriormente, se aplica la Transformada Inversa de Fourier para reconstruir las señales en el dominio temporal, recuperando su forma original.

Una vez que las señales han sido diferenciadas por bandas de frecuencia, el algoritmo de machine learning procede a extraer las características (features) de cada tipo de onda a partir de una representación en el dominio de la frecuencia y del tiempo. Estas características son utilizadas para realizar la predicción del movimiento deseado en la silla de ruedas.

4.6.3. ¿Que es una feature?

Una feature es una propiedad medible o una característica distintiva de un fenómeno. En el contexto de este proyecto, las features se refieren a las propiedades específicas de cada señal cerebral, clasificadas por tipo de onda (alpha, beta, gamma, delta y theta). Estas propiedades son extraídas para capturar la información relevante que permitirá al sistema de aprendizaje automático identificar patrones en las señales y realizar predicciones relacionadas con el control de la silla de ruedas.

4.7. Filtros digitales

Un filtro digital es una herramienta utilizada en el procesamiento de señales para modificar o mejorar ciertos aspectos de una señal digital, eliminando o atenuando componentes no deseados, como ruido, o destacando componentes útiles. Los filtros digitales operan sobre señales discretizadas (muestreadas) y utilizan algoritmos matemáticos para realizar diversas transformaciones.

Existen diferentes tipos de filtros digitales, según su función:

- **Filtro pasa bajas:** Permite el paso de frecuencias bajas y atenúa las frecuencias altas.
- **Filtro pasa altas:** Permite el paso de frecuencias altas y atenúa las bajas.
- **Filtro pasa banda:** Permite el paso de un rango específico de frecuencias.

- **Filtro elimina banda (Notch):** Atenúa un rango específico de frecuencias, como en el caso de eliminar interferencias a 50-60 Hz.

Los filtros digitales pueden ser FIR (Finite Impulse Response) o IIR (Infinite Impulse Response):

- Los filtros FIR tienen una respuesta finita y son más estables, pero pueden necesitar más recursos de procesamiento.
- Los filtros IIR tienen una respuesta infinita y suelen ser más eficientes en términos de recursos, pero pueden ser menos estables si no se diseñan correctamente.

Los filtros digitales son ampliamente usados en aplicaciones como procesamiento de audio, señales biológicas (como EEG), imágenes y telecomunicaciones.

4.7.1. ¿Para que los necesitamos si ya tenemos por Hardware?

La combinación de filtros físicos (implementados en hardware) con filtros digitales (implementados en software) presenta una serie de ventajas significativas, incluso si ya se han instalado filtros físicos en el sistema. Aunque los filtros basados en hardware permiten un filtrado efectivo de las señales EEG, los filtros digitales proporcionan una flexibilidad y capacidades adicionales que no siempre son posibles de lograr mediante hardware exclusivamente. A continuación, se detallan las razones por las cuales puede ser beneficioso incorporar filtros digitales a la salida de los filtros físicos:

1. **Ajuste Preciso y Mayor Flexibilidad**

   Los filtros de hardware están limitados a sus parámetros iniciales, ya que una vez construidos con componentes específicos (como resistencias, condensadores, etc.), sus características clave, tales como la frecuencia de corte y el ancho de banda, son difíciles de modificar. En contraste, los filtros digitales permiten ajustar fácilmente parámetros como la frecuencia de corte o la configuración de filtrado sin necesidad de realizar cambios físicos. Esto resulta especialmente útil cuando, tras el análisis de la señal, es necesario ajustar el rango de paso de un filtro o eliminar una frecuencia no contemplada inicialmente. Además, los filtros digitales permiten realizar modificaciones rápidas y sin intervención física, optimizando la eficiencia del proceso.

2. **Compensación de las Limitaciones Intrínsecas del Hardware**

   Los filtros físicos, aunque efectivos, no son perfectos. Su desempeño puede verse afectado por tolerancias en los componentes o por factores ambientales, como la temperatura y el envejecimiento de los mismos. Los filtros digitales, por su parte, pueden corregir o complementar las imperfecciones de los filtros de hardware. Por ejemplo, si un filtro físico no consigue eliminar completamente el ruido de 50 Hz o presenta desviaciones mínimas en las frecuencias de corte, un filtro digital puede ajustarse con precisión para corregir estas deficiencias, mejorando así la calidad general del filtrado.

3. **Filtrado Adaptativo en Tiempo Real**

   Los filtros físicos son estáticos, lo que significa que su comportamiento es constante y no varía según las condiciones de la señal. En cambio, un filtro digital ofrece la posibilidad de implementar técnicas de filtrado adaptativo, es decir, que el filtro ajuste sus características en función de las condiciones cambiantes de la señal. Esto es especialmente beneficioso en entornos donde las condiciones son variables, como cuando la interferencia de 50 Hz no está presente de manera constante. Un filtro digital adaptativo puede activarse solo cuando detecta interferencias, optimizando el procesamiento de la señal en tiempo real. De esta manera, el sistema EEG se ajusta de manera dinámica, lo que puede incrementar la fiabilidad del procesamiento.

4. **Procesamiento Avanzado y Filtrado de Alta Precisión**

   La complejidad de los filtros físicos está limitada por los componentes que los conforman. Sin embargo, en el ámbito digital es posible implementar filtros de mayor complejidad y orden,

permitiendo un filtrado más preciso y específico. Los filtros digitales de mayor orden ofrecen transiciones más nítidas entre las bandas de paso y de rechazo. Por ejemplo, con un filtro digital, es posible eliminar una banda estrecha de ruido sin afectar las frecuencias adyacentes de interés, algo difícil de lograr mediante componentes físicos convencionales.

5. **Filtrado en Múltiples Etapas**

   El uso de filtrado digital permite procesar la señal en diferentes etapas, lo que resulta en un análisis más profundo y detallado de las señales EEG. Tras el paso por los filtros físicos, se pueden aplicar diferentes tipos de filtros digitales para aislar aspectos específicos de la señal que no fueron manejados por el filtrado físico. Por ejemplo, se pueden implementar filtros pasa banda específicos para resaltar ciertas bandas de ondas cerebrales, como las ondas alpha o beta, o utilizar técnicas avanzadas como el Análisis de Componentes Independientes (ICA) para separar señales superpuestas o interferencias no deseadas. El uso de filtros adaptativos también permite ajustar continuamente la respuesta del sistema, optimizando el rendimiento del procesamiento de señales.

6. **Posprocesamiento y Análisis Selectivo**

   El filtrado digital permite realizar un análisis detallado de las señales EEG en una etapa posterior, ya sea sobre los datos almacenados o sobre la señal adquirida en tiempo real. Este posprocesamiento es fundamental para detectar patrones específicos o eliminar artefactos que no fueron suficientemente atenuados en la etapa inicial de filtrado físico. Por ejemplo, los artefactos relacionados con el movimiento o las señales de baja frecuencia, como la respiración, pueden requerir un ajuste adicional después de la adquisición de la señal. Además, un análisis más exhaustivo en tiempo real puede detectar elementos que los filtros de hardware no lograron capturar completamente.

7. **Redundancia y Seguridad**

   Incorporar tanto filtros físicos como digitales en un sistema de procesamiento de señales EEG añade una capa extra de seguridad. Si un filtro físico falla o no cumple completamente su función, el filtro digital actúa como un sistema de respaldo, asegurando que el ruido y las interferencias no afecten el análisis de la señal. Esta redundancia es clave en entornos críticos donde se necesita garantizar la integridad y pureza de los datos EEG para obtener resultados confiables y precisos.

4.8. Cantidad de canales

4.9. Gel conductor

El gel conductor neutro empleado en los electroencefalogramas (EEG) es una sustancia formulada para optimizar la conductividad eléctrica entre los electrodos y el cuero cabelludo del paciente. Su principal objetivo es minimizar la resistencia entre los electrodos y la piel, garantizando una transmisión eficiente y sin interferencias de las señales eléctricas cerebrales hacia los electrodos.

4.9.1. Cómo aplicar el gel conductor

En los electrodos de copa dorada, el procedimiento consiste únicamente en rellenar la cavidad de la copa con el gel conductor, posteriormente colocarlo sobre una zona del cuero cabelludo previamente limpiada con algodón, mantenerlo en posición durante unos segundos y eliminar cualquier exceso de gel, en caso de haberlo, con un pequeño trozo de papel. Utilizamos este gel en el proyecto con el fin de obtener la mejor calidad posible en las señales, además de generar un leve efecto de succión que contribuye a mantener los electrodos firmemente adheridos.

4.10. Jaula de Faraday

Una jaula de Faraday es, esencialmente, una estructura metálica diseñada para proteger su interior de los campos eléctricos estáticos. Su nombre proviene del físico Michael Faraday, quien construyó la primera de estas estructuras en 1836. Las jaulas de Faraday se utilizan para proteger su contenido de descargas eléctricas y del ruido electromagnético, debido a que, en su interior, el campo eléctrico es nulo. Su principio de funcionamiento se basa en las propiedades de un conductor en equilibrio electrostático, lo que impide que cualquier carga eléctrica atraviese la estructura. Por esta razón, se emplean para proteger dispositivos sensibles a las cargas eléctricas. Este fenómeno es conocido como apantallamiento eléctrico.

4.10.1. ¿Por qué utilizamos una Jaula de Faraday?

Dado que trabajamos con señales altamente susceptibles al ruido debido a su frecuencia y voltaje, como es el caso de las señales captadas por el EEG, es fundamental mantenerlas lo más aisladas posible de interferencias externas. Además, contamos con una fuente importante de ruido electromagnético: el motor. Por esta razón, es esencial posicionar los componentes lo más alejados posible entre sí y utilizar una Jaula de Faraday en la sección destinada a la recepción y procesamiento de las señales.

La implementación de este elemento se realiza mediante una caja metálica que será ubicada en la parte superior trasera de la silla de ruedas, es decir, en la zona de apoyo de la espalda, para mantenerla lo más distante posible de los motores.

4.11. ¿Qué es un circuito de Offset?

Un circuito de offset se utiliza para agregar un voltaje constante a una señal, desplazando su nivel de referencia sin cambiar la forma de la señal. Este tipo de circuito es común en sistemas de procesamiento de señales, donde se requiere ajustar el punto de referencia de una señal para que coincida con el rango de entrada de otro dispositivo o componente.

Ejemplo de Funcionamiento:

Una señal de entrada, como una señal de corriente alterna (AC), puede presentar un rango de voltaje que oscila entre valores positivos y negativos, por ejemplo, una onda senoidal entre -5V y 5V.

Si se desea desplazar esta señal para que su rango esté comprendido entre 0V y 10V, se debe aplicar un voltaje de offset de 5V.

El circuito de offset sumará este valor constante de 5V a la señal, desplazando el rango completo de la misma hacia valores positivos, sin alterar su forma original.

4.11.1. Utilización de Offsets

En la sección de procesamiento y recepción, se implementó un circuito de offset para adaptar las señales, ya que la RP2040 Zero no es capaz de procesar señales con componentes negativos. Este ajuste permite una correcta recepción y, en consecuencia, un procesamiento completo de las señales.

4.12. Sistema de emergencia

La silla de ruedas está equipada con un sistema de emergencia basado en tecnología de ultrasonido, que se encuentra instalado en las direcciones principales del dispositivo. Este sistema tiene como propósito prevenir colisiones con objetos o cualquier tipo de obstáculo, protegiendo tanto al usuario como a la integridad de la silla. Cuando los sensores de ultrasonido detectan la presencia de un obstáculo cercano, se activa un indicador visual mediante un LED en la dirección del objeto, acompañado de una señal acústica generada por un buzzer. Simultáneamente, el sistema detiene el movimiento de la silla de ruedas de manera automática y procede a un tiempo de espera de unos segundos para evaluar si es seguro continuar el desplazamiento.

4.13. Tipos de alimentaciones

El proyecto tiene distintas tensiones de alimetación, pero la mayoría viene de la batería de 24V. La Raspberry Pi 4 se alimenta con 5V, por lo que debe pasar por un circuito regulador de tensiones conformado por un L7815 y luego por un L7805 L7805.

4.14. Sistema de control

4.15. Puentes H

Un Puente H es un circuito utilizado para invertir el sentido de giro de un motor y para separar su etapa de potencia con la de control. Su nombre viene de la forma gráfica que tiene el circuito. Se construye con 4 interruptores, pueden ser mecánicos o transistores. En la siguiente imágen se puede ver la forma de un puente H.

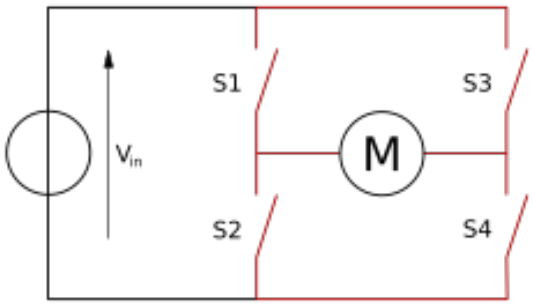


Figura 3: Puente H Genérico por medio de llaves mecánicas

En este ejemplo, cuando la llave S1 y S4 se cierran, correrá corriente por esta rama, haciendo que el motor funcione y tenga un sentido de giro. Cuando la llave S2 y S3 se cierran, correrá corriente por esta rama también, pero con sentido de giro invertido. En este circuito, S1 y S2 nunca pueden estar cerradas al mismo tiempo, porque esto generaría un corto circuito, lo mismo con las llaves S3 y S4.

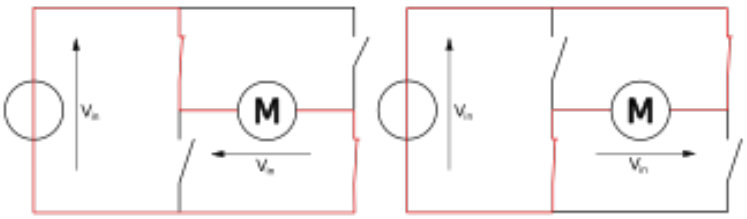


Figura 4: Puente H Operando

4.15.1. Puente H de potencia

Un puente H de potencia es un tipo de circuito en el que la etapa de potencia, que incluye el motor, los transistores y está conectada a altas tensiones y corrientes, se encuentra separada de la etapa de control, la cual generalmente opera con voltajes más bajos. Esta separación se realiza para proteger la etapa de control de las altas tensiones, que podrían dañar o quemar sus circuitos. Podemos utilizar nuestro puente H como ejemplo para ilustrar este concepto:

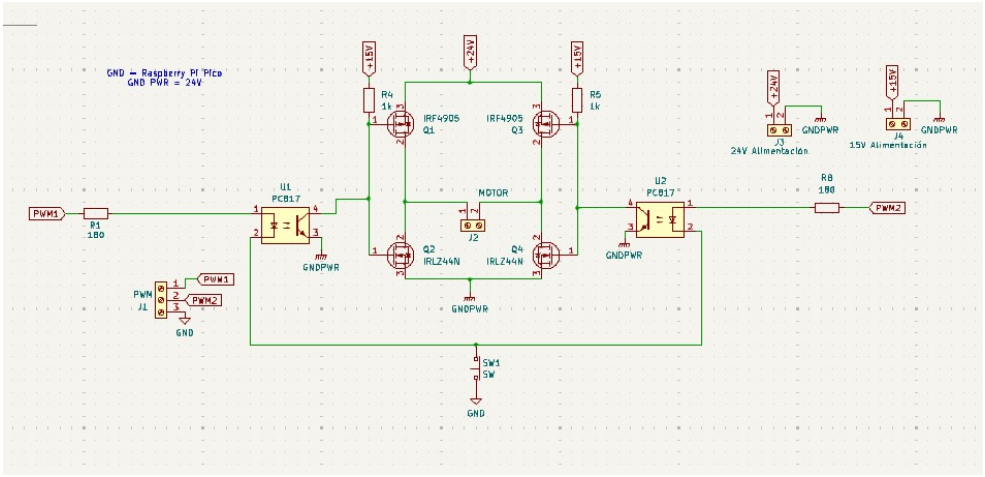


Figura 5: Nuestro Puente H de potencia

En nuestro puente H, hemos separado las ramas del motor de las de control, ya que el motor opera con 24V DC y tiene una corriente eficaz de 3,5A. Para controlar el motor de manera eficiente, hemos optado por un diseño que incluye cuatro transistores MOSFET de potencia, los cuales pueden soportar picos de hasta 74A y 82A, garantizando que puedan manejar la corriente del motor de manera continua. Además, los transistores son controlados mediante señales PWM, las cuales atraviesan un optoacoplador para minimizar el riesgo de que la corriente del motor regrese a la placa de control y cause daños. De ésta manera, podemos controlar el giro del motor de manera segura y regulada.

4.16. Sistema de recepción de señales

4.17. Control por PWM

La modulación por ancho de pulso (PWM, por sus siglas en inglés Pulse Width Modulation) es una técnica empleada para regular la cantidad de energía suministrada a una carga, como un motor o un LED, mediante la variación del ciclo de trabajo de una señal pulsante. En lugar de modificar el voltaje de forma continua, PWM ajusta el tiempo durante el cual la señal permanece en estado alto (encendida) y bajo (apagada) dentro de un período fijo. Esto depende de los siguientes parámetros:

- **Frecuencia:** Se refiere a la cantidad de ciclos de encendido y apagado que ocurren en un segundo, y se mide en Hertz (Hz).
- **Ciclo de trabajo:** Es el porcentaje de tiempo que la señal se mantiene en estado alto durante un ciclo. Por ejemplo: Un ciclo de trabajo del 0 % implica que la señal está siempre apagada. Un ciclo de trabajo del 50 % significa que la señal está encendida la mitad del tiempo y apagada la otra mitad. Un ciclo de trabajo del 100 % indica que la señal permanece siempre encendida. En nuestro proyecto, aplicamos la técnica de PWM en el control de los motores. Dado que utilizamos dos motores, si estos operaran a su máxima potencia, la silla de ruedas se desplazaría a una velocidad excesiva. Por esta razón, empleamos un ciclo de trabajo del 25 % para limitar la velocidad de avance.

5. Desarrollo Técnico

En este capítulo se proporcionará una descripción detallada sobre el funcionamiento de los distintos circuitos que conforman el dispositivo, así como su interacción en conjunto para cumplir con los objetivos del proyecto. Se explicará de manera clara y concisa cómo se utiliza el dispositivo, incluyendo instrucciones específicas para su correcto manejo y operatividad.

Además, se incluirán datos técnicos relevantes, tales como especificaciones de los componentes, características del diseño, y cualquier otro detalle que sea esencial para la comprensión del sistema. También se abordarán aspectos importantes acerca de las partes constitutivas del dispositivo, su función dentro del esquema general, y las consideraciones necesarias para su mantenimiento y optimización.

5.1. Descripción del funcionamiento

El usuario se sienta en la silla de ruedas y se coloca el dispositivo EEG. El EEG capta las señales cerebrales (alfa, beta, gamma, delta y theta) y las somete a un proceso de filtrado por hardware que incluye un filtro notch de 50 Hz, un filtro pasa-bajo de 100 Hz, un filtro pasa-alto de 0,5 Hz y un segundo filtro notch de 50 Hz. Posteriormente, las señales pasan por un circuito de compensación de offset para poder realizar la lectura por parte de la RP2040 Zero ya que la misma no puede leer valores negativos. Esta unidad se encarga de leer los datos mediante cuatro canales de ADC y enviarlos a la Raspberry Pi 4 por protocolo UART, donde se aloja el programa principal para el funcionamiento de la silla de ruedas.

Entre estos códigos se encuentran el control de motores, los filtros digitales, el procesamiento de señales y la inteligencia artificial (IA). Una vez las señales llegan a la Raspberry Pi 4, se filtran digitalmente para eliminar el ruido y mejorar la calidad de los datos. Tras el filtrado, las señales se envían a la IA, que tiene la función de reconocer los patrones cerebrales y determinar la dirección en la que el usuario desea moverse. Una vez el patrón ha sido identificado, se envía una señal a los motores para dirigir la silla hacia la dirección deseada (adelante, atrás, izquierda o derecha).

Además, el proyecto incluye un sistema de emergencia diseñado para garantizar la seguridad del usuario. Este sistema se activa si se detecta un error en la lectura de las señales o si hay un obstáculo en la trayectoria deseada. Su objetivo es prevenir accidentes y aumentar la seguridad del usuario al utilizar la silla de ruedas.

El sistema de emergencia utiliza sensores de ultrasonido para detectar la presencia de objetos a menos de 20 cm de la silla. En caso de detección, el sistema frena los motores y activa un LED en la dirección del objeto, además de un buzzer que emite una señal sonora durante un breve período para alertar al usuario sobre la activación del sistema.

5.2. Funcionamiento del EEG

El sistema de control de nuestra silla de ruedas se basa en el reconocimiento de señales cerebrales, lo que requiere un dispositivo especializado para captarlas: el electroencefalograma (EEG). Hemos desarrollado un EEG en forma de una cofia equipada con electrodos, los cuales se colocan siguiendo el protocolo conocido como “Sistema 10-20”. Este sistema estándar de ubicación define puntos específicos en el cuero cabelludo donde se deben colocar los electrodos para obtener lecturas óptimas de la actividad cerebral.

Los electrodos actúan como sensores pasivos que detectan las ondas cerebrales producidas por la actividad eléctrica del cerebro. Estas ondas se manifiestan como pequeñas fluctuaciones en el voltaje, que son captadas por los electrodos cuando se colocan sobre la superficie del cuero cabelludo del usuario. Es importante destacar que estos electrodos no penetran en el cráneo ni interactúan directamente con las neuronas. En cambio, registran los potenciales eléctricos generados por la actividad cerebral subyacente que se propagan hasta la superficie del cuero cabelludo.

Debido a que las señales captadas por los electrodos son de muy baja amplitud, requieren ser amplificadas antes de su procesamiento. Este paso no solo amplifica la señal cerebral, sino también el ruido presente en el entorno o generado por el propio cuerpo. Para abordar este problema, se realiza un proceso de filtrado que elimina el ruido y permite aislar las frecuencias relevantes de las ondas cerebrales.

Las frecuencias de las ondas cerebrales detectadas son las siguientes:

- Ondas alfa: 8-12 Hz
- Ondas beta: 12-30 Hz

- Ondas gamma: 30-100 Hz
- Ondas delta: 0,5-4 Hz
- Ondas theta: 4-8 Hz

Estas ondas cerebrales están asociadas con diferentes estados mentales y actividades cerebrales, por ejemplo, las ondas alfa se observan comúnmente en estados de relajación, mientras que las ondas beta se relacionan con un estado de alerta y concentración. Las ondas gamma, por otro lado, están vinculadas a procesos cognitivos de mayor complejidad, como la percepción y la conciencia. Las ondas delta y theta se asocian con fases profundas del sueño y la meditación.

5.3. Funcionamiento de los filtros HardWare

En un sistema de EEG, después de amplificar las señales captadas por los electrodos, se utilizan varios filtros para limpiar y mejorar la calidad de las señales antes de procesarlas o analizarlas. Estos filtros tienen un propósito importante: eliminar el ruido y enfocarse en las señales cerebrales relevantes.

5.3.1. Filtro Notch de 50 Hz

- **Propósito:** Atenuación del ruido inducido por la red eléctrica de 50 Hz, común en muchos sistemas de distribución de energía.
- **Funcionamiento:** El filtro Notch es un filtro de rechazo de banda estrecha, diseñado para eliminar una frecuencia específica (en este caso, 50 Hz) sin afectar significativamente las componentes de frecuencia cercanas o las señales EEG de interés. Las señales de EEG, que suelen estar en el rango de 0,5 a 40 Hz, pueden contaminarse por la interferencia de la red, por lo que este filtro resulta esencial para su eliminación sin alterar la información relevante.

5.3.2. Filtro pasa bajos de 100 Hz

- **Propósito:** Reducción de las componentes de alta frecuencia que no son relevantes para el análisis EEG.
- **Funcionamiento:** Este filtro pasa bajos permite el paso de frecuencias inferiores a 100 Hz, atenuando aquellas que exceden este umbral. Las señales cerebrales de interés generalmente se sitúan por debajo de los 40 Hz, aunque algunas pueden extenderse hasta 100 Hz. Este filtro elimina frecuencias más altas, que usualmente corresponden a interferencias electromagnéticas, ruido de dispositivos electrónicos o artefactos musculares (EMG).

5.3.3. Filtro pasa altos de 0,5 Hz

- **Propósito:** Eliminación de las componentes de muy baja frecuencia, incluyendo artefactos de corriente continua (DC) y ruido de baja frecuencia no relacionado con la actividad cerebral.
- **Funcionamiento:** El filtro pasa altos atenúa las señales con frecuencias por debajo de 0,5 Hz, permitiendo únicamente el paso de frecuencias superiores. La señal EEG relevante generalmente comienza en torno a los 0,5 Hz, mientras que frecuencias menores suelen estar asociadas a artefactos como el desplazamiento de los electrodos, fluctuaciones en la impedancia de los mismos, o interferencias lentas que no contienen información útil sobre la actividad cerebral.

5.4. Funcionamiento de los filtros digitales

Una vez que las ondas cerebrales son captadas y sometidas a un proceso inicial de filtrado, son transferidas a una Raspberry Pi 4. En este dispositivo se ejecuta un programa, desarrollado íntegramente por nuestro equipo, que continúa con el procesamiento y refinamiento de la señal. El motivo por el cual se aplica un filtrado adicional es que, luego de la etapa de filtrado inicial, la señal pasa por un segundo amplificador. Este amplificador, además de amplificar las señales cerebrales de interés, también amplifica el ruido residual presente en las mismas. Para contrarrestar este efecto, empleamos filtros digitales implementados por software, los cuales ofrecen una mayor precisión y control en la eliminación de interferencias no deseadas.

Dentro del programa, se han implementado los siguientes filtros:

- **Filtro Notch de 50 Hz:** Este filtro está diseñado para eliminar las interferencias generadas por la red eléctrica, las cuales operan a una frecuencia de 50 Hz. Dado que estas interferencias pueden superponerse con la señal cerebral y distorsionar su interpretación, su eliminación es esencial para obtener un registro más fiel de la actividad cerebral.
- **Filtro Pasa-Bajo de 100 Hz:** El objetivo de este filtro es retener las frecuencias cerebrales más relevantes, es decir, aquellas que se encuentran por debajo de los 100 Hz, mientras que elimina el ruido de alta frecuencia, que suele ser resultado de interferencias externas o artefactos no relacionados con la actividad cerebral. Este rango incluye frecuencias relacionadas con ritmos cerebrales importantes como las ondas alfa, beta, y gamma.
- **Filtro Pasa-Alto de 0,5 Hz:** Este filtro se encarga de remover las señales de baja frecuencia que no aportan información útil sobre la actividad cerebral, como los artefactos causados por movimientos lentos o variaciones de corriente. Solo se permite el paso de las frecuencias EEG relevantes, garantizando que las señales reflejen con mayor precisión la actividad neural de interés.

Es importante señalar que el uso de estos filtros garantiza que la señal procesada esté lo más limpia posible de interferencias y artefactos, mejorando la precisión del sistema para detectar y procesar patrones cerebrales. La implementación de filtros mediante software, además, permite ajustar dinámicamente las configuraciones según las necesidades específicas del sistema y del entorno operativo, optimizando así el rendimiento de la silla de ruedas controlada por señales cerebrales.

5.5. Funcionamiento del Puente H

El circuito utiliza dos señales PWM (PWM1 y PWM2) para controlar los transistores MOSFET en configuración de puente H, lo que permite invertir la polaridad del voltaje aplicado al motor, controlando así la dirección de rotación.

La velocidad del motor puede controlarse ajustando el ciclo de trabajo (duty cycle) de las señales PWM, que modulan la cantidad de energía entregada al motor.

Este circuito es ideal para aplicaciones donde se requiere un control preciso de dirección y velocidad de motores DC.

5.5.1. Descripción de los Componentes

1. **U1 y U2 (PC817 - Optoacopladores):** Aíslan eléctricamente el circuito de control (de baja tensión) del circuito de potencia (de alta tensión). Esto protege al microcontrolador de posibles picos de voltaje y ruidos eléctricos provenientes del circuito de potencia. Cada optoacoplador tiene un LED interno que, al ser activado por una señal PWM, enciende un fototransistor interno, permitiendo que la señal de control pase al lado de potencia.
2. **Q1 y Q3 (IRF4905 - MOSFET canal P):** Actúan como interruptores en la parte superior del puente H. Controlan la alimentación positiva hacia el motor. Estos MOSFET se activan cuando la señal PWM1 o PWM2 está en bajo (0V) y se desactivan cuando está en alto (señal de 5V), debido a la naturaleza de los MOSFET de canal P.

3. **Q2 y Q4 (IRF2805 - MOSFET canal N):** Actúan como interruptores en la parte inferior del puente H. Conectan el motor a tierra cuando están activados. Estos MOSFET se activan cuando la señal PWM1 o PWM2 está en alto (5V) y se desactivan cuando está en bajo (0V).

4. **R1 y R2 (Resistencias de 180 ohms):** Limitan la corriente de entrada hacia los LEDs internos de los optoacopladores (U1 y U2) para protegerlos de daños.

5. **R3 y R4 (Resistencias de 1k ohms):** Están conectadas en serie con los MOSFET de canal P (Q1 y Q3). Limitan la corriente de la compuerta y aseguran un apagado rápido del MOSFET al detenerse la señal.

6. **J1 y J2 (Conectores de entrada PWM):** Reciben las señales PWM de control provenientes de un microcontrolador o una fuente de señal PWM.

7. **J3 y J4 (Conectores de alimentación):** Proveen la alimentación al circuito, uno con 24V para el motor y otro con 15V para los circuitos de control.

8. **SW1 (Interruptor):** utilizado para habilitar o deshabilitar el circuito completo, conectando o desconectando GND.

5.6. Funcionamiento del Sistema de Emergencia

6. Circuitos usados

En este capítulo se incluirán imágenes de los esquemas de los circuitos, junto con su representación gráfica en formato virtual de los circuitos impresos, así como fotografías de los circuitos completos una vez finalizados.

Esquemático de los filtros

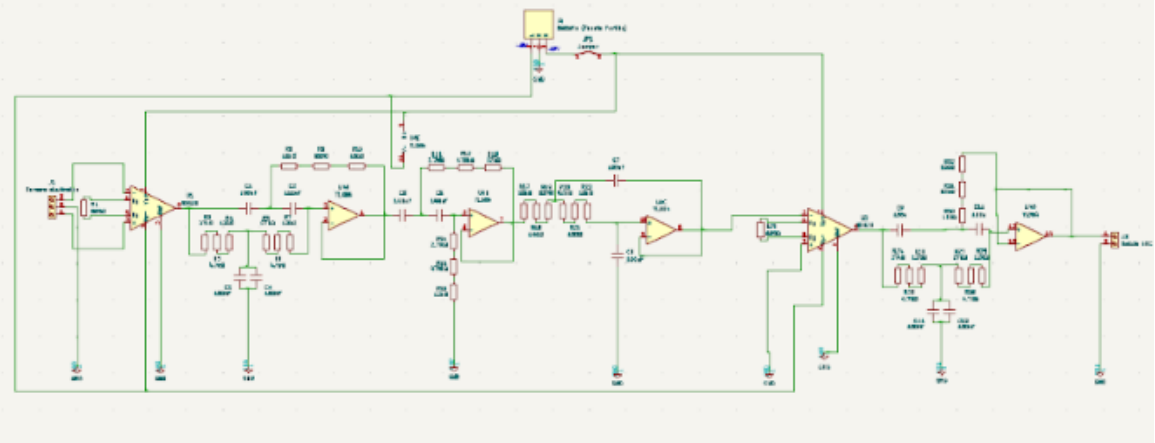


Figura 6: Esquemático de los filtros del EEG

6.1. Esquemático del sistema de control

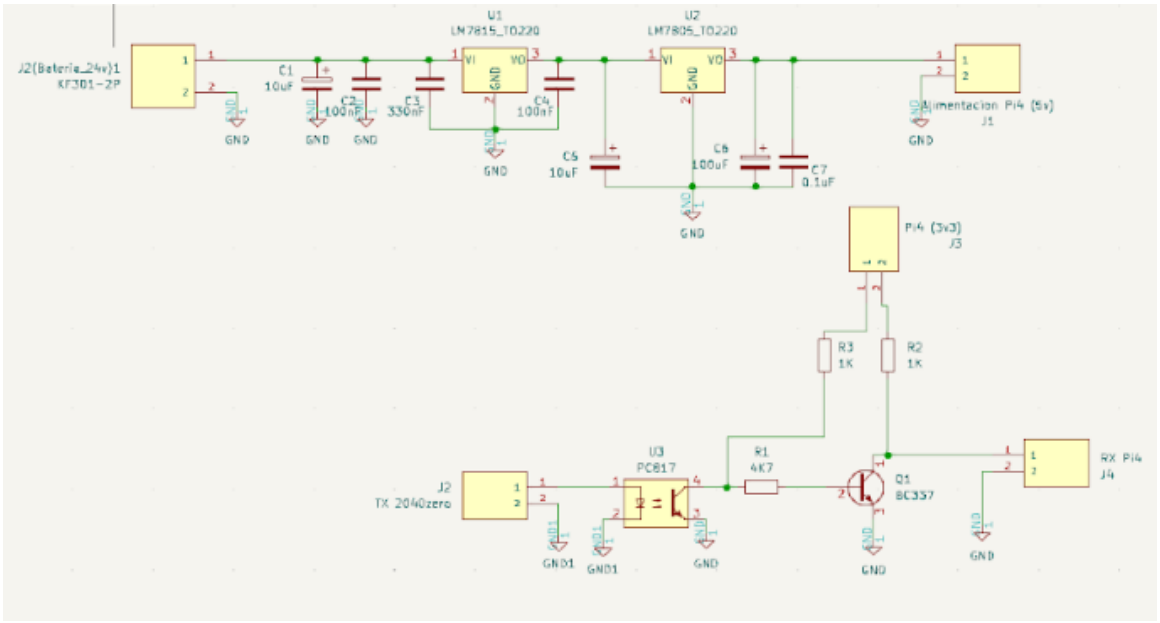


Figura 7: Esquemáticos del sistema de control

6.2. Esquemático del sistema Offset

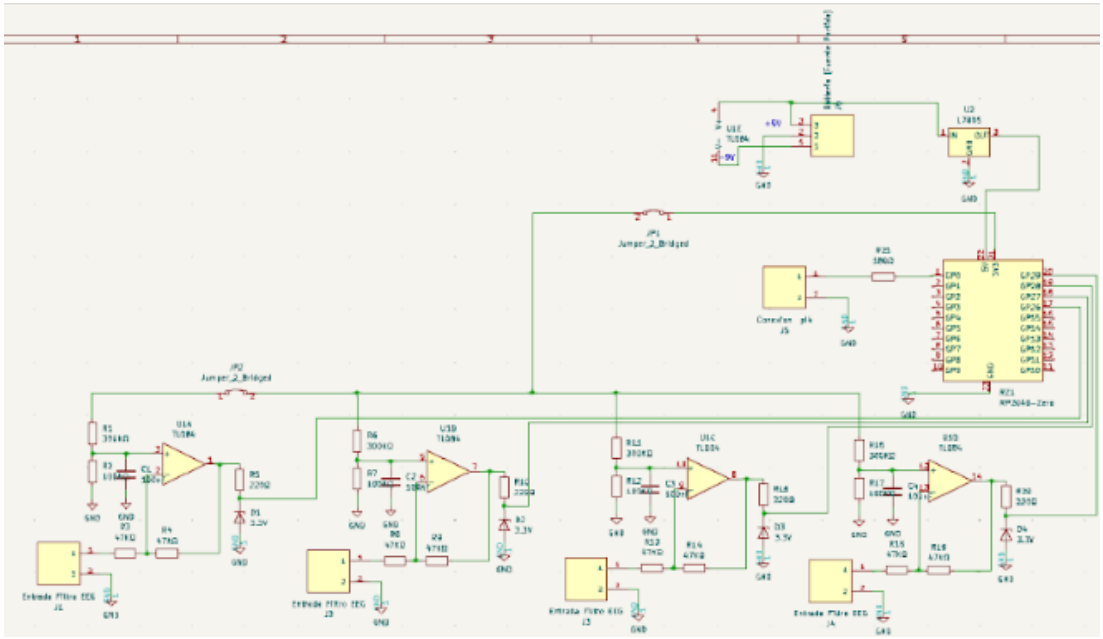


Figura 8: Esquemático del sistema de Offset

6.3. Esquemáticos de los Puentes H

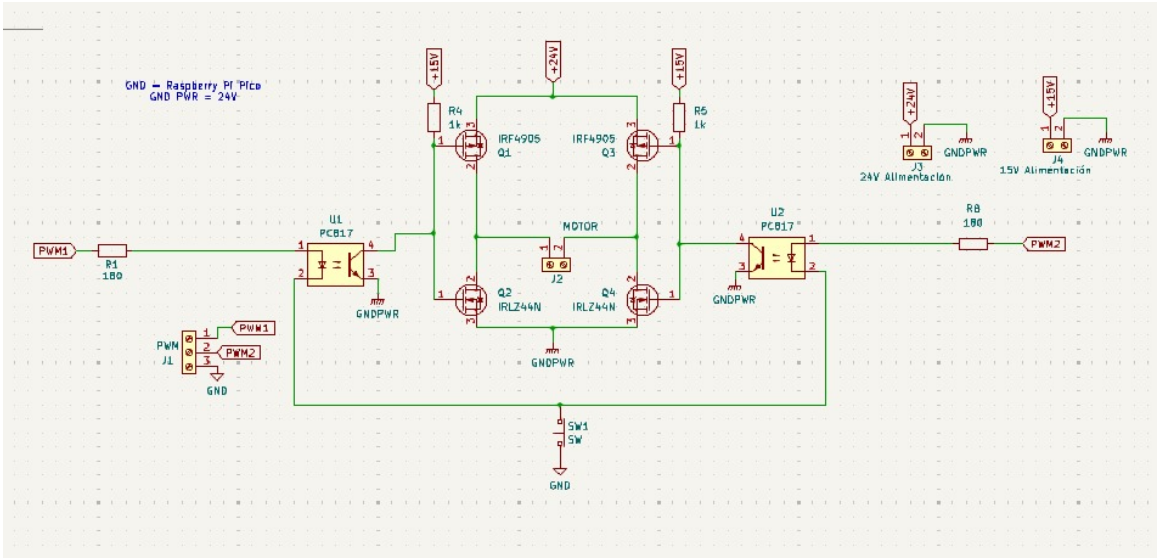


Figura 9: Puente H (Un Motor)

6.4. Esquemático del sistema de emergencia

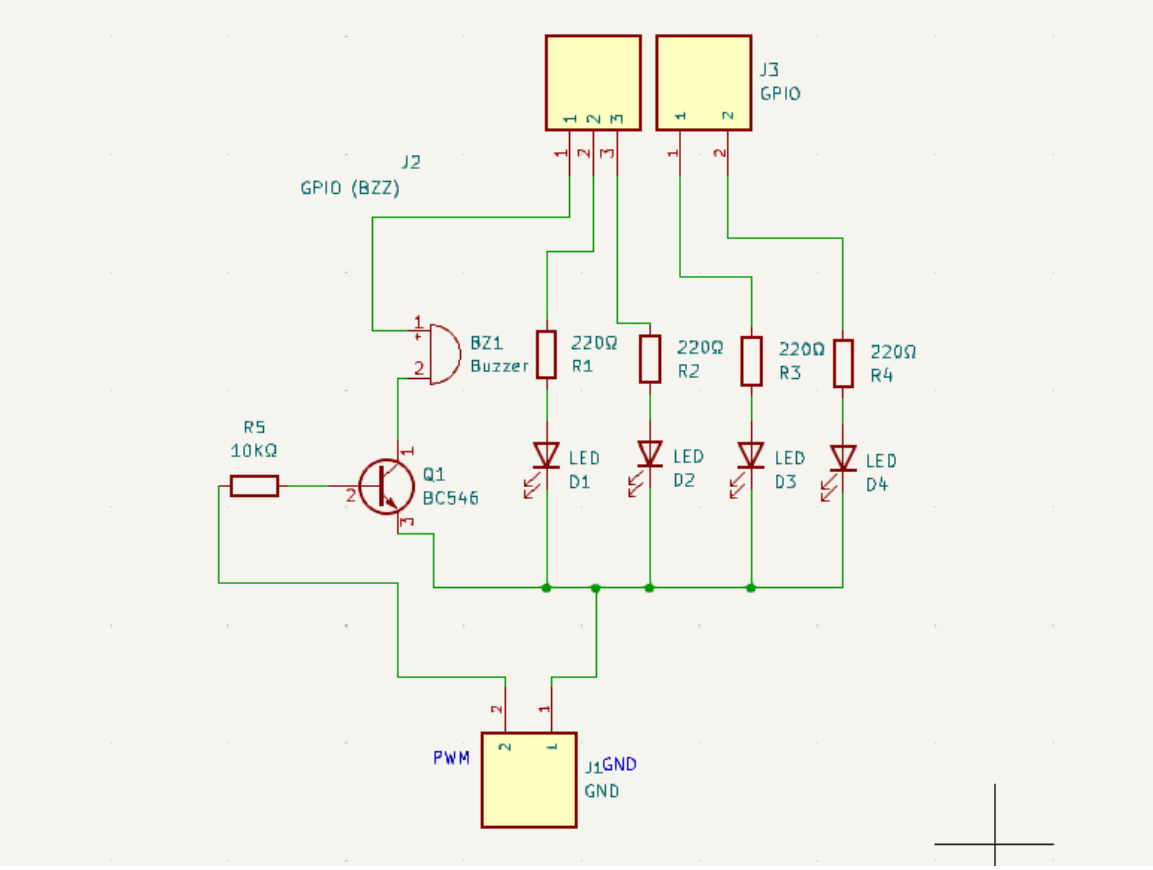
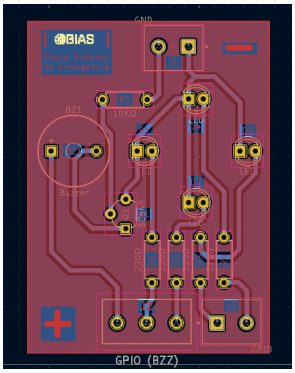
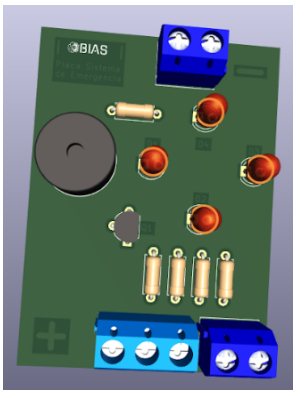


Figura 10: Esquemático del sistema de emegencia



(a) PCB Del sistema de emergencia



(b) Modelo 3D Del sistema de emergencia

6.5. Lista de materiales

Los programas desarrollados para el proyecto, tales como los filtros digitales, sistemas de inteligencia artificial y otros códigos relacionados, se almacenan en los siguientes dispositivos:

- Raspberry Pi 4
- RP2040-zero

6.5.1. EEG

1. TL084 (x4)
2. AD620 (x8)
3. Capacitores 100nF (x48)
4. Resistencias 100 (x4)
5. Resistencias 120 (x16)
6. Resistencias 560 (x8)
7. Resistencias 820 (x16)
8. Resistencias 1,5k (x8)
9. Resistencias 4,7k (x16)
10. Resistencias 12k (x8)
11. Resistencias 15k (x8)
12. Resistencias 27k (x16)
13. Resistencias 470k (x8)
14. Resistencias 2,7M (x8)
15. Electrodos (x9)

6.5.2. Placa OffSet

1. L7805 (x1)
2. TL084 (x4)
3. Jumpers (x2)
4. Diodos Zenner 3,3V (x4)
5. Capacitores 100nF (x4)
6. Resistencias 220 (x4)
7. Resistencias 50k (x8)
8. Resistencias 100k (x8)

6.5.3. Sistema de Emergencia

1. Transistor BC546 (x1)
2. Buzzer 5V (x1)
3. LEDs Rojos (x4)
4. Resistencias 220 (x4)
5. Resistencia 4,7k (x1)

6.5.4. Puentes H

1. Transistores IRF4905 (x4)
2. Transistores IRF2805 (x4)
3. Optoacopladores PC817 (x4)
4. Motores 24V 575W (x2)
5. Resistencias 100 (x4)
6. Resistencias 180 (x4)
7. Resistrencias 1k (x4)
8. Resistencias 4,7k (x4)

6.5.5. Placa reguladora de tensión

1. LM7815 (x1)
2. Capacitor 100F (x1)
3. Capacitor 330F (x1)
4. Capacitor 470F 35V (x1)
5. Capacitor 470F 25V (x1)

7. Conclusiones

En esta sección se presentará un análisis detallado de los conocimientos adquiridos a lo largo del desarrollo del proyecto, así como las conclusiones a las que hemos llegado durante este proceso. Se incluirán las conclusiones finales obtenidas, así como una evaluación de las principales limitaciones que afectaron el progreso del trabajo. Además, se ofrecerá un resumen de los resultados alcanzados, destacando los aspectos más relevantes y significativos. Finalmente, se abordarán las perspectivas futuras del proyecto, delineando los pasos a seguir para continuar su desarrollo y mejorarlo en el futuro.

7.1. Resumen de Resultados

7.2. Limitaciones

Durante la ejecución de este proyecto, nos enfrentamos a diversas limitaciones, algunas de las cuales fueron ocasionadas por circunstancias personales de cada integrante del equipo. Como es común en todo proyecto, al principio experimentamos dificultades en la organización y carecíamos de ciertos conocimientos técnicos, los cuales fuimos adquiriendo a lo largo del año.

El tiempo dedicado al proyecto quizá no fue suficiente, considerando la magnitud y la ambición del mismo. Además de estos desafíos iniciales, una de las principales limitaciones fue el prolongado tiempo de espera para obtener los materiales necesarios, los cuales provenían principalmente de la cooperadora. Dado que esta era nuestra principal fuente de herramientas y componentes, nuestro progreso estuvo estrictamente limitado por los tiempos de adquisición manejados por la institución.

Asimismo, al estar creando algo "nuevo", nos vimos obligados a adoptar un enfoque basado en la prueba y error, ya que la información disponible sobre los temas específicos que requeríamos era escasa. Esta situación también representó un obstáculo significativo, ya que corregir nuestros errores consumió una parte considerable de nuestro tiempo.

Finalmente, al tratarse de un proyecto desarrollado en el ámbito escolar, también nos vimos limitados por el tiempo disponible en la escuela. No pudimos dedicar todo nuestro tiempo al proyecto, ya que debíamos atender otras asignaturas y compromisos académicos.

7.3. Trabajo Futuro

El proyecto presenta un alto potencial de rentabilidad en el futuro, sustentado en su carácter innovador y las posibilidades que ofrece su desarrollo continuo. Al tratarse de un invento único en su categoría, se prevé que, con el tiempo y el trabajo adicional necesario, este dispositivo pueda alcanzar un nivel de madurez suficiente para ser comercializado a gran escala.

La inversión en tiempo y recursos adicionales permitirá perfeccionar sus características técnicas, optimizar su funcionalidad, y adaptar el producto a las necesidades de un mercado en expansión. Una vez finalizado y refinado, el proyecto podrá captar la atención de un público más amplio, lo que incrementará su viabilidad comercial y su posicionamiento en el mercado.

En caso de que el desarrollo del proyecto no se complete en su totalidad, este podrá ser aprovechado por la institución educativa, permitiendo que otros grupos de estudiantes continúen su avance. De esta manera, el proyecto no solo servirá como un recurso valioso para futuras generaciones, sino que también fomentará la colaboración y el aprendizaje continuo dentro de la comunidad académica, garantizando su evolución y finalización.

7.4. Informacion Adicional

La silla desarrollada en este proyecto es un prototipo, es decir, puede contener fallas de todo tipo, y es sujeto a modificaciones a futuro. Se recomienda su uso siempre acompañado, en zonas amplias libres de obstáculos. No se debe usar en lugares húmedos o si el equipo está mojado, y la inteligencia artificial está adaptada a las muestras de una persona del equipo del proyecto, por lo que para ser usada por alguien mas se deberán muestrear los patrones de movimiento de esta misma.

7.4.1. Programas Utilizados

Para el desarrollo del proyecto, hemos requerido hacer uso de los siguientes porogramas, con sus enlaces principales:

-Github

-Visual Studio Code

-Thonny

-Overleaf

-Pycharm

-Termius

-Kicad

-Livewire

-LTSpice

-GitHub Desktop

7.4.2. Agradecimientos

Queremos expresar nuestro más profundo agradecimiento a los profesores que nos han brindado su invaluable apoyo en la realización de este proyecto:

- Fabrizio Carlassara: Su asistencia fue crucial en la programación de la mayor parte del proyecto, así como en la creación de circuitos y esquemáticos, la búsqueda de materiales y el desarrollo de los filtros necesarios.
- Sergio Medina: Nos brindó su experiencia en la presentación y comercialización del proyecto, además de colaborar en la programación, la búsqueda de materiales y la planificación de los puentes H.
- Carlos Bianco: Su apoyo fue esencial en las pruebas y acondicionamiento de los motores, proporcionándonos herramientas y baterías para dichas pruebas, además de colaborar en la búsqueda de materiales.
- Daniel Espósito: Nos proporcionó motores y materiales, además de enseñarnos a construir los puentes H para los motores y nos ayudó a solucionar problemas con las pistas de los circuitos para las altas corrientes.
- Federico Solomiewicz: Colaboró principalmente en el desarrollo de los filtros, proporcionándonos herramientas para el proyecto y estando siempre disponible para resolver cualquier duda que surgiera.
- Juan Carlos Ruiz: Su colaboración fue esencial en las pruebas de los motores para la silla de ruedas, además de ayudarnos principalmente en la creación de los puentes H para los motores y sus circuitos impresos.
- Ana (Neurologa):

8. Referencias

En este apartado se presentarán de manera detallada las referencias consultadas al inicio del proyecto, las cuales fueron fundamentales para la conceptualización y desarrollo de la idea final. Estas fuentes proporcionaron el marco teórico y práctico necesario para orientar la dirección del trabajo, influyendo en las decisiones tomadas durante el proceso de diseño y ejecución.

8.1. Brain Computer Interface

La primera referencia utilizada en este proyecto es un informe técnico titulado "Brain Computer Interface", escrito por María Madrid Sobrino, el cual aborda el desarrollo de una interfaz cerebro-computadora (BCI) basada en señales EEG, específicamente empleando potenciales evocados visuales de estado estacionario (SSVEP).

Este documento resultó fundamental para nuestra investigación, ya que abarca los siguientes aspectos relevantes:

- Introducción a las interfaces cerebro-computadora (BCI): Una interfaz cerebro-computadora (BCI) es un sistema que permite a los usuarios comunicar sus intenciones al entorno externo sin necesidad de utilizar mecanismos fisiológicos naturales, como los nervios periféricos o los músculos. Estas interfaces están diseñadas principalmente para mejorar la comunicación de personas con discapacidades severas, aunque también tienen aplicaciones en la rehabilitación neurológica, en la industria de los videojuegos, entre otros. El informe distingue entre dos tipos de BCI: los invasivos, en los cuales los electrodos se implantan directamente en el cerebro, y los no invasivos, donde los electrodos se colocan en la superficie del cráneo. Los BCIs no invasivos, basados en la electroencefalografía (EEG), son los más ampliamente utilizados debido a su practicidad y menor riesgo.
- BCI basado en SSVEP: El informe describe un sistema BCI basado en SSVEP (Potenciales Evocados Visuales de Estado Estacionario), que utiliza la respuesta cerebral a estímulos visuales repetitivos. Cuando un estímulo visual parpadea a una frecuencia constante, el cerebro

genera señales EEG a la misma frecuencia del estímulo. Se destaca la alta efectividad de estos sistemas, que presentan una elevada relación señal-ruido y requieren poco entrenamiento por parte del usuario. Sin embargo, se señala que algunos estímulos visuales pueden provocar fatiga o, en ciertos casos, ataques epilépticos.

La referencia detalla el uso de la plataforma BCI2000 para la adquisición de datos, procesamiento de señales y presentación de estímulos. El sistema fue desarrollado empleando C++ y Matlab, lo que permitió la integración de un módulo de procesamiento de señales en Matlab. El sistema desarrollado incluye una aplicación que presenta 16 opciones visuales organizadas en cuatro grupos, permitiendo al usuario seleccionar una imagen específica enfocando su atención en ella. El método de clasificación empleado se basa en la frecuencia de los estímulos visuales, y la integración con BCI2000 facilita la ejecución de un sistema de lazo cerrado, donde las señales cerebrales controlan directamente el proceso de selección de imágenes.

Esta referencia fue clave para establecer las bases teóricas y técnicas necesarias para la implementación de un sistema de control basado en señales cerebrales, aplicable a nuestro proyecto de silla de ruedas.

8.2. Artificial Intelligence for Real-Time Decoding of Motor Commands from ECoG of Disabled Subjects for Chronic Brain-Computer Interfacing

La segunda referencia utilizada corresponde a la tesis doctoral titulada .ªrtificial Intelligence for Real-Time Decoding of Motor Commands from ECoG of Disabled Subjects for Chronic Brain-Computer Interfacing", presentada por Maciej Sliwowski en la Université Grenoble Alpes en 2022.

Esta tesis aborda el uso de inteligencia artificial (IA) para la decodificación en tiempo real de comandos motores a partir de señales de electrocorticografía (ECoG) en sujetos con discapacidades motoras. El objetivo principal es mejorar la comunicación y el control mediante interfaces cerebro-computadora (BCI). Los sistemas BCI ofrecen una herramienta de asistencia crucial para pacientes con tetraplejia, proporcionando un medio de interacción directa entre el cerebro y el entorno, compensando la pérdida de funciones motoras.

El enfoque principal del trabajo se centra en los BCI basados en ECoG, que presentan un equilibrio óptimo entre los sistemas intracorticales, más invasivos, y los métodos no invasivos, como el EEG, que suelen ser menos precisos. La tesis estudia específicamente la decodificación de movimientos continuos de traslación de la mano en un paciente tetrapléjico, logrando avances importantes en la interpretación de las señales cerebrales para aplicaciones prácticas en la rehabilitación motora.

Este trabajo es relevante para nuestro proyecto ya que aborda la integración de IA en la interpretación de señales neuronales, lo que resulta fundamental para desarrollar sistemas de control eficientes en tiempo real, como es el caso de nuestra silla de ruedas controlada por la mente. Además, su enfoque en la ECoG aporta perspectivas valiosas para mejorar la precisión y la respuesta del sistema BCI en sujetos con discapacidades motoras severas.

8.3. Controlled Wheelchair Based on Brain Computer Interface using Neurosky Mindwave Mobile 2

El artículo presenta el desarrollo de una silla de ruedas controlada mediante una interfaz cerebro-computadora (BCI) utilizando el dispositivo Neurosky Mindwave Mobile 2 para la adquisición de señales EEG. El propósito de este sistema es mejorar la movilidad y la calidad de vida de pacientes post-ictus, permitiéndoles controlar la silla de ruedas a través de sus ondas cerebrales.

Este artículo resultó valioso para nuestras primeras pruebas, dado que inicialmente contábamos con un dispositivo Neurosky Mindwave Mobile 2. Sin embargo, debido a limitaciones en la programación y procesamiento de señales, decidimos descartar esta opción y desarrollar nuestro propio sistema EEG.

El sistema BCI permite a los usuarios controlar dispositivos mediante señales cerebrales, las cuales son registradas utilizando electroencefalografía (EEG). El Neurosky Mindwave Mobile 2 es

un dispositivo portátil de bajo costo diseñado para registrar dichas señales EEG. En el contexto de este proyecto, las señales cerebrales se utilizan para controlar una silla de ruedas eléctrica, que originalmente funcionaba a través de un joystick.

El módulo de control del joystick fue reemplazado por un controlador basado en microcontroladores, que procesa las señales cerebrales capturadas y las clasifica utilizando un software desarrollado en Matlab. Este proceso permite que las señales EEG controlen el motor de la silla de ruedas en tiempo real.

En la metodología utilizada se emplea el método de motor imagery, el cual consiste en registrar y analizar las ondas cerebrales asociadas a la imaginación de movimientos motores, tales como caminar o mover las manos. Las señales capturadas por el dispositivo Neurosky se procesan en Matlab y se clasifican en cinco clases de movimiento: avanzar, retroceder, girar a la derecha, girar a la izquierda y una posición neutral.

El sistema incorpora la funcionalidad eSense, una característica del dispositivo que mide los niveles de atención y meditación del usuario, contribuyendo a la precisión del control basado en señales cerebrales.

8.4. Brain–computer interface for electric wheelchair based on alpha waves of EEG signal

El presente estudio aborda el desarrollo de una interfaz cerebro-computadora (Brain-Computer Interface, BCI) destinada al control de una silla de ruedas eléctrica mediante la detección y análisis de ondas alfa generadas por el cerebro, capturadas a través de un electroencefalograma (EEG). El principal objetivo de este sistema es proporcionar asistencia a personas que padecen enfermedades neurológicas graves que afectan severamente su capacidad de movilidad. La solución propuesta destaca por su simplicidad operativa, dado que emplea un número reducido de electrodos y no requiere un extenso período de entrenamiento para los usuarios.

Este estudio fue utilizado como referencia en el desarrollo de nuestro propio proyecto, el cual comparte un enfoque similar en la implementación de un sistema BCI para el control de una silla de ruedas. A partir de la revisión detallada de sus métodos y resultados, adaptamos y optimizamos el diseño de nuestros propios circuitos.

8.5. A Neural Network Based Brain–Computer Interface for Classification of Movement Related EEG

La tesis elaborada por Pontus Forslund y presentada en Linköping en diciembre de 2003, trata sobre el diseño de una interfaz cerebro-computadora (BCI) basada en electroencefalografía (EEG), enfocada en la clasificación de movimientos de la mano en dos dimensiones. El objetivo de dicho proyecto es desarrollar un sistema de comunicación intuitivo para individuos con discapacidades motoras graves.

En este estudio, se registraron señales EEG de un sujeto mientras operaba un joystick en cuatro direcciones. Las señales capturadas fueron procesadas mediante un modelo autorregresivo para extraer características relevantes, que luego fueron empleadas como entradas para una red neuronal artificial, diseñada para clasificar la dirección de los movimientos.

Este estudio ha sido utilizado como referencia para nuestro proyecto, específicamente en lo que respecta al reconocimiento de señales EEG y su asignación a los comandos de control del movimiento de la silla de ruedas controlada mediante señales cerebrales.

9. Lista de códigos

9.1. Códigos principales

9.1.1. bias.py

```python
from bias_dsp import FilterBias, ProcessingBias
from bias_reception import ReceptionBias
import numpy as np
from bias_graphing import GraphingBias
from bias_motors import MotorBias
from bias_ai import AIBias

class BiasClass:
    # Constructor
    def __init__(self, n, fs, channels, port, baudrate, timeout):
        # Define propieties for the class
        self._n = n
        self._fs = fs
        self._number_of_channels = channels
        self._duration = self._n / self._fs
        self._port = port
        self._baudrate = baudrate
        self._timeout = timeout
        self._commands = ["forward", "backwards", "left", "right", "stop", "rest"]
        self._samples_trainig_command = 100

        # Create objects as propieties in order to apply the rest of the code in Bias class
        self._biasReception = ReceptionBias(self._port, self._baudrate, self._timeout)
        self._biasFilter = FilterBias(n=self._n, fs=self._fs, notch=True, bandpass=True, fir=False, iir=False)
        self._biasProcessing = ProcessingBias(n=self._n, fs=self._fs)
        self._biasGraphing = GraphingBias(graph_in_terminal=True)
        self._biasMotor = MotorBias(echo_forward=18, trigger_forward=17, echo_backwards=23, trigger_backwards=22, echo_right=5, trigger_right=6,
                                    echo_left=25, trigger_left=24, led_forward=16, led_backwards=20, led_left=21, led_right=26, buzzer=12, motor1_in1=13,
                                    motor1_in2=19, motor2_in1=7, motor2_in2=8)
        self._biasAI = AIBias(self._n, self._fs, self._number_of_channels, self._commands)

    def train_ai_model(self, save_path, saved_dataset_path):
        self._biasAI.collect_and_train(reception_instance=self._biasReception, filter_instance=self._biasFilter,
                                       processing_instance=self._biasProcessing,
                                       samples_per_command=self._samples_trainig_command, save_path=save_path,
                                       saved_dataset_path=saved_dataset_path, real_data=True)

    def app_run(self):
        while True:
            # Receive eeg data
            signals = self._biasReception.get_real_data(channels=self._number_of_channels, n=self._n)

            # Graph signals
            for ch, signal in signals.items():
                t = np.arange(len(signals[ch])) / self._fs
                self._biasGraphing.graph_signal_voltage_time(t=t, signal=np.array(signal), title="Signal {}".format(ch))

            # Apply digital filtering
            filtered_data = self._biasFilter.filter_signals(signals)

            # Calculate the time vector
            t = np.linspace(0, self._duration, self._n, endpoint=False)

            # Graph signals
            for ch, signal in filtered_data.items():
                # Graph filtered signal
                self._biasGraphing.graph_signal_voltage_time(t=t, signal=signal,
```

```
    title="Filtered Signal {}".format(ch))

            # Process data
            times, eeg_signals = self._biasProcessing.process_signals(filtered_data)

            # Plot 4 signals with its resepctive bands
            for ch, signals in eeg_signals.items():
                # Plot the interpolated signals
                for band_name, sig in signals.items():
                    self._biasGraphing.graph_signal_voltage_time(t=times[ch], signal=
    sig, title=f"{band_name.capitalize()} interpolated. {ch}")

            # Plot
            self._biasGraphing.plot_now()

            #command = self._biasAI.predict_command(eeg_data=eeg_signals)
            #self._biasMotor.move_if_possible(command)
```

9.1.2. app.py

9.2. Inteligencia artificial

9.2.1. bias_ai.py

9.3. Filtrado y procesamiento de señales

9.3.1. bias_dsp.py

9.3.2. bias_graphing.py

```
import matplotlib.pyplot as plt
import plotext

class GraphingBias:
    # Constructor
    def __init__(self, graph_in_terminal):
        self._graph_in_terminal = graph_in_terminal

    # Graph signal in function of time
    def graph_signal_voltage_time(self, t, signal, title):
        if not self._graph_in_terminal:
            # Plot given signal in th time domain
            plt.figure(figsize=(12, 6))
            if signal.ndim == 1:
                plt.plot(t, signal)
            else:
                for i in range(signal.shape[0]):
                    plt.plot(t, signal[i])
            # Set graph paramters
            plt.title(title)
            plt.xlabel("Time [s]")
            plt.ylabel("Magnitude")
            plt.grid()

        else:
            plotext.clear_data()
            plotext.plot(t, signal)
            # Set graph parameters
            plotext.title(title)
            plotext.xlabel("Time [s]")
            plotext.ylabel("Magnitude")
            plotext.show()

    def graph_signal_voltage_frequency(self, frequencies, magnitudes, title):
        if not self._graph_in_terminal:
            # Plot given signal in the frquency domain
            plt.figure(figsize=(12, 6))
            plt.plot(frequencies, magnitudes)
```

```
            # Set graph parameters
            plt.title(title)
            plt.xlabel("Frequency [Hz]")
            plt.ylabel("Magnitude")
            plt.grid()

        else:
            plotext.clear_data()
            plotext.plot(frequencies, magnitudes)
            # Set graph parameters
            plotext.title(title)
            plotext.xlabel("Frequency [Hz]")
            plotext.ylabel("Magnitude")
            plotext.show()

    def plot_now(self):
        if not self._graph_in_terminal:
            plt.tight_layout()
            plt.show()
        else:
            pass

'''
#import matplotlib

#matplotlib.use('Agg')  # Use the non-GUI backend

#import matplotlib.pyplot as plt
import plotext as plt

def graph_signal_voltage_time(t, signal, title, filename):
    # Plot given signal in th time domain
    #plt.figure(figsize=(12, 6))
    if signal.ndim == 1:
        plt.plot(t, signal)
    else:
        for i in range(signal.shape[0]):
            plt.plot(t, signal[i])
    plt.title(title)
    plt.xlabel("Time [s]")
    plt.ylabel("Magnitude")
    plt.grid()
    #plt.savefig(filename)  # Save the plot as an image
    #plt.close()  # Close the plot to free memory
    #print(f"Plot saved to {filename}")

def graph_signal_voltage_frequency(frequencies, magnitudes, title):
    # Plot given signal in the frquency domain
    #plt.figure(figsize=(12, 6))
    plt.plot(frequencies, magnitudes)
    plt.title(title)
    plt.xlabel("Frequency [Hz]")
    plt.ylabel("Magnitude")
    plt.grid()

def plot_now():
    plt.tight_layout()
    plt.show()
'''
```

9.4. Motores

9.4.1. bias_motors.py

```
from gpiozero import DistanceSensor, PWMLED, Buzzer, PWMOutputDevice
from gpiozero.pins.pigpio import PiGPIOFactory
import time

```

```
def main():
    # Define motor instance
    biasMotor = MotorBias(echo_forward=18, trigger_forward=17, echo_backwards=23, trigger_backwards=22, echo_right=5, trigger_right=6,
                          echo_left=25, trigger_left=24, led_forward=16, led_backwards=20, led_left=21, led_right=26, buzzer=12, motor1_in1=13,
                          motor1_in2=19, motor2_in1=7, motor2_in2=8)
    while True:
        # Get command
        command = input("Enter command (forward/left/backwards/right/stop): ").strip()
        biasMotor.move_if_possible(command)

class MotorBias:
    def __init__(self, echo_forward, trigger_forward, echo_backwards, trigger_backwards, echo_right, trigger_right,
                 echo_left, trigger_left, led_forward, led_backwards, led_left, led_right, buzzer, motor1_in1,
                 motor1_in2, motor2_in1, motor2_in2):

        # Configurar pin factory in order to use pigpio
        factory = PiGPIOFactory()

        # Configure ultrasonic sensors and LEDs
        self._ultrasonic_forward = DistanceSensor(echo=echo_forward, trigger=trigger_forward, pin_factory=factory)
        self._ultrasonic_backwards = DistanceSensor(echo=echo_backwards, trigger=trigger_backwards, pin_factory=factory)
        self._ultrasonic_right = DistanceSensor(echo=echo_right, trigger=trigger_right, pin_factory=factory)
        self._ultrasonic_left = DistanceSensor(echo=echo_left, trigger=trigger_left, pin_factory=factory)

        # Configure LEDs
        self._led_forward = PWMLED(led_forward, pin_factory=factory)
        self._led_backwards = PWMLED(led_backwards, pin_factory=factory)
        self._led_left = PWMLED(led_left, pin_factory=factory)
        self._led_right = PWMLED(led_right, pin_factory=factory)

        # Configure buzzer
        self._buzzer = Buzzer(buzzer)

        # GPIO Pin setup for Motor 1
        self._motor1_in1 = PWMOutputDevice(motor1_in1, frequency=50, pin_factory=factory)
        self._motor1_in2 = PWMOutputDevice(motor1_in2, frequency=50, pin_factory=factory)

        # GPIO Pin setup for Motor 2
        self._motor2_in1 = PWMOutputDevice(motor2_in1, frequency=50, pin_factory=factory)
        self._motor2_in2 = PWMOutputDevice(motor2_in2, frequency=50, pin_factory=factory)

    def move_if_possible(self, command):
        try:
            # Move forward
            if command == "forward":
                distance = self._ultrasonic_forward.distance * 100
                # Maximum distance of 20 cm
                if distance < 20:
                    # Forward is blocked
                    self._led_forward.on()
                    self._buzzer.on()
                    print(f"Obastacle forward: {distance:.1f} cm. Blocked movement.")
                else:
                    # Do the movement
                    self._led_forward.off()
```

```
                    self._buzzer.off()
                    self.move_forward(25)
            # Move backwards
            elif command == "backwards":
                distance = self._ultrasonic_backwards.distance * 100
                # Maximum distance of 20 cm
                if distance < 20:
                    # Backwards is blocked
                    self._led_backwards.on()
                    self._buzzer.on()
                    print(f"Obstacle backwards: {distance:.1f} cm. Blocked movement."
)
                else:
                    # Do the movement
                    self._led_backwards.off()
                    self._buzzer.off()
                    self.move_backward(25)
            # Turn left
            elif command == "left":
                distance = self._ultrasonic_left.distance * 100
                # Maximum distance of 20 cm
                if distance < 20:
                    # Left is blocked
                    self._led_left.on()
                    self._buzzer.on()
                    print(f"Obstacle on the left: {distance:.1f} cm. Blocked movement
")
                else:
                    # Do the movement
                    self._led_left.off()
                    self._buzzer.off()
                    self.turn_left(25)
            # Turn right
            elif command == "right":
                distance = self._ultrasonic_right.distance * 100
                # Maximum distance of 20 cm
                if distance < 20:
                    # Right is blocked
                    self._led_right.on()
                    self._buzzer.on()
                    print(f"Obstacle on the right: {distance:.1f} cm. Blocked
movement.")
                else:
                    # Do the movement
                    self._led_right.off()
                    self._buzzer.off()
                    self.turn_right(25)
            # Brake
            elif command == "stop":
                # Make all parameters off
                self.brake()
                self._led_forward.off()
                self._led_backwards.off()
                self._led_left.off()
                self._led_right.off()
                self._buzzer.off()
            else:
                print("Invalid command")

            time.sleep(1)
            self.brake() # Stop after each command
            self._led_forward.off()
            self._led_backwards.off()
            self._led_left.off()
            self._led_right.off()
            self._buzzer.off()

        except KeyboardInterrupt:
```

```
            print("Program stopped by user")

    # Configure speed of motor depending on PWM
    def set_motor_speed(self, motor_in1, motor_in2, speed, invert=False):
        # Define positive speed
        if speed > 0:
            motor_in1.value = ((100.0 - speed) / 100.0) if invert else speed / 100.0
            motor_in2.value = 0
        # Define negative speed
        elif speed < 0:
            motor_in1.value = 0
            motor_in2.value = ((100.0 - abs(speed)) / 100.0) if invert else abs(speed) / 100.0
        # If it's zero brake
        else:
            motor_in1.value = 0
            motor_in2.value = 0

    # Move wheelchair forward
    def move_forward(self, speed):
        self.set_motor_speed(self._motor1_in1, self._motor1_in2, speed, invert=True)
        self.set_motor_speed(self._motor2_in1, self._motor2_in2, speed, invert=True)

    # Move wheelchair backwards
    def move_backward(self, speed):
        self.set_motor_speed(self._motor1_in1, self._motor1_in2, -speed, invert=True)
        self.set_motor_speed(self._motor2_in1, self._motor2_in2, -speed, invert=True)

    # Turn wheelchair left
    def turn_left(self, speed):
        self.set_motor_speed(self._motor1_in1, self._motor1_in2, -speed, invert=True)
        self.set_motor_speed(self._motor2_in1, self._motor2_in2, speed, invert=True)

    # Turn wheelchair right
    def turn_right(self, speed):
        self.set_motor_speed(self._motor1_in1, self._motor1_in2, speed, invert=True)
        self.set_motor_speed(self._motor2_in1, self._motor2_in2, -speed, invert=True)

    # Brake wheelchair
    def brake(self):
        self.set_motor_speed(self._motor1_in1, self._motor1_in2, 0, invert=True)
        self.set_motor_speed(self._motor2_in1, self._motor2_in2, 0, invert=True)

if __name__ == "__main__":
    main()
```

9.5. Recepción de señales

9.5.1. bias_reception.py

```
import serial
import time
import numpy as np
import json
from bias_graphing import GraphingBias
from signals import random_signal

def main():
    # Set constants
    n = 1000
    fs = 500
    number_of_channels = 4
    # Receive data
    biasReception = ReceptionBias()
    signals = {}
    signals = biasReception.get_real_data(channels=number_of_channels, n=n)

    # Graph signals
```

```
    biasGraphing = GraphingBias(graph_in_terminal=True)
    for ch, signal in signals.items():
        t = np.arange(len(signals[ch])) / fs
        biasGraphing.graph_signal_voltage_time(t=t, signal=np.array(signal), title="
    Signal {}".format(ch))

class ReceptionBias:
    # Constructor
    def __init__(self, port='/dev/serial0', baudrate=115200, timeout=1):
        self._port = port
        self._baudrate = baudrate
        self._timeout = timeout

    # Get the data from the RP2040 Zero
    def get_real_data(self, channels, n):
        # Initialize serial communication
        self._ser = self.init_serial(self._port, self._baudrate, self._timeout)
        try:
            # Capture the signal from the UART
            real_eeg_signals = self.capture_signals(channels=channels, n=n)
            return real_eeg_signals
        finally:
            self._ser.close()

    def capture_signals(self, channels, n):
        # Initialize variables
        signals = {f'ch{ch}': [] for ch in range(channels)}
        start_time = time.time()
       # Loop until we have enough samples
        while len(signals['ch3']) < n:
            if self._ser.in_waiting > 0:
                try:
                    # Read one line to detect \n character
                    data = self._ser.readline().decode('utf-8').strip()
                    eeg_data = self.process_data(data)
                    # Make an array with the data
                    if eeg_data:
                        for ch in range(channels):
                            signals[f'ch{ch}'].extend(eeg_data[f'ch{ch}'])
                except Exception as e:
                    print("Can't be decoded")


        # Check the time it takes to read
        elapsed_time = time.time() - start_time
        print(f"elapsed time: {elapsed_time}")
        # Ensure all signals have the correct length
        for ch in range(channels):
            signals[f'ch{ch}'] = signals[f'ch{ch}'][:n]

        return signals

    '''
    def combine_signals(self, signals):
        combined_signal = np.mean([signals[f'ch{ch}'] for ch in range(len(signals))],
     axis=0)
        return combined_signal
    '''

    # Initialize serial communication
    def init_serial(self, port, baudrate, timeout):
        return serial.Serial(port, baudrate, timeout=timeout)

    # Load JSON data
    def process_data(self, data):
        try:
            json_data = json.loads(data)
            print(json_data)
```

```
            return json_data
        except json.JSONDecodeError as e:
            print(f"Error decoding JSON: {e}")
            return None

if __name__ == "__main__":
    main()
```

9.5.2. CMakeLists.txt

9.5.3. pico_sdk_import.cmake

9.5.4. reception.c

9.6. Página Web

El sitio web ofrece una breve descripción sobre nosotros, nuestros objetivos, y proporciona enlaces a nuestras redes sociales, donde podrán contactarnos o conocer más acerca de nuestro trabajo.

Página Web

9.6.1. Lenguaje utilizado en la página web y su código

El desarrollo de la página web se realizó utilizando los lenguajes HTML y CSS.
Aquí se adjuntan los códigos de la página web:

Index.css

```
*{
    box-sizing: border-box;
}

html{
    scroll-behavior: smooth;
}

body{
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0%;
    background-image:linear-gradient(to right, #0f3443 0%,#34e69f 100%);
    color: white;
}

h1{ font-size: 3.5em;}
h2{ font-size: 2.7em;}
h3{ font-size: 2em;}
p{ font-size: 1.25em;}
ul{ list-style: none;}
li{ font-size: 1.25em;}

button{
    font-size: 1.25em;
    font-weight: bold;
    padding: 10px 30px;
    border-radius: 5px;
    border: 2px solid rgba(0,0,0,0.3);
    box-shadow: 2px 2px 10px rgba(0,0,0,0.5);
    color: white;
    background-color:grey;
}

button:hover{
    background-color: rgb(101, 33, 165);
}

```

```
.container{
    max-width: 1400px;
    margin: auto;
}

.color-acento{ color:blueviolet; }

header{
    background-color: rgb(14, 26, 52);

}

.social-icons {
    display: grid;
    font-size: 3rem;
    grid-template-columns: repeat(5, 1fr);
    grid-auto-flow: column;
    grid-auto-columns: 1fr;
    align-items: center;
    padding-right: -100px;
  }

header .logo{
    margin: 0;
    padding: 25px 30px;
    font-weight: bold;
    color: blueviolet;
    font-size: 1.6em;
    height:15vh;

}

header .container{
    display: flex;
    flex-direction: column;
    align-items: center;
}

header nav{
    display: flex;
    flex-direction: column;
    text-align: center;
    padding-bottom: 25px;
}

header a{
    padding: 5px 12px;
    text-decoration: none;
    font-weight: bold;
    color:rgb(183, 219, 251);
}

header a:hover{
    color: blueviolet;
}

#hero{
    display: flex;
    align-items: center;
    justify-content: center;
    text-align: center;
    flex-direction: column;
    height: 90vh;
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
```

```
    ,url("he.jpg");
    background-repeat: no-repeat;
    background-size: cover;
    background-position: center center;
}

#hero h1{
    color:rgb(183, 219, 251);
}

#hero button{
    font-size: 1.75em;
}

#somos-proya .container{
    text-align: center;
    padding: 200px 12px;
}

#nuestros-programas{
    background-color: rgb(14, 26, 52);
    color:rgb(183, 219, 251);
    text-align: center;
}

#nuestros-programas .container{
    padding: 150px 12px;
}

#nuestros-programas h2{
    margin-top: 0;
    font-size: 3.2em;
}

#nuestros-programas p{
    display: none;
}

#nuestros-programas .carta{
    background-position: center center;
    background-size: cover;
    padding: 50px 0px;
    margin: 30px;
    border-radius: 15px;
}

.carta:first-child{
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/front-end.jpg");

}

.carta:nth-child(2){
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/full-stack.jpg");
}

.carta:nth-child(3){
    background-image: linear-gradient(
        0deg,
```

```
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/python.jpg");

}

.carta:nth-child(4){
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/full-stack.jpg");
}
.carta:nth-child(5){
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/python.jpg");

}

.carta:nth-child(6){
    background-image: linear-gradient(
        0deg,
        rgba(0,0,0,0.5),
        rgba(0,0,0,0.5)
    )
    ,url("media/full-stack.jpg");
}

#caracteristicas{
    background-image:linear-gradient(to right, #0f3443 0%,#34e69f 100%);
    color:rgb(183, 219, 251);
    text-align: center;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    flex-wrap: wrap;
}

#caracteristicas .container{
    padding: 150px 12px;
}

#caracteristicas h2{
    margin-top: 0;
    font-size: 3.em;
    color:rgb(183, 219, 251);
    text-align: center;
}

#caracteristicas p{
    display: none;
}

#caracteristicas .carta{
    background-position: center center;
    background-size: cover;
    padding: 50px 0px;
    margin: 30px;
    border-radius: 15px;
}

```

```
#caracteristicas .container{
    text-align: center;
    padding: 250px 12px;
    width: 100vw;
}

#contacto{
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    text-align: center;
    background-color: rgb(14, 26, 52);
    color:rgb(183, 219, 251);
    height: 80vh;
}

#contacto h2{
    font-size: 9vw;
}

#contacto button{
    font-size: 5vw;
}

footer{
    background-color: white;
}

footer p{
    margin: 0;
    padding: 12px;
    color:  rgb(14, 26, 52);
}

footer .container{
    height: 150px;
    display: flex;
    justify-content: center;
    align-items: flex-end;
}

.fab.fa-linkedin {
    color: rgb(183, 219, 251);
  }

  .fab.fa-instagram {
    color: rgb(183, 219, 251);
  }

  .fab.fa-google{
    color: rgb(183, 219, 251);
  }

  .fas.fa-music{
    color: rgb(183, 219, 251);
  }

@media (min-width: 850px){
    header{
        position: fixed;
        width: 100%;
    }

    header .container{
        flex-direction: row;
        justify-content: space-between;
    }
```

```
311
312     header nav{
313         flex-direction: row;
314         padding-bottom: 0;
315         padding-right: 20px;
316     }
317
318     #hero h1{
319         font-size: 5em;
320     }
321
322     body{
323         color:  rgb(14, 26, 52);
324     }
325     #somos-proya .container{
326         display: flex;
327         justify-content: space-evenly;
328     }
329
330     #somos-proya .texto{
331         width: 50%;
332         max-width: 600px;
333         text-align: initial;
334         padding-left: 30px;
335         display: flex;
336         flex-direction: column;
337         justify-content: center;
338     }
339
340     #somos-proya h2{
341         margin-top: 0px;
342         font-size: 4em;
343     }
344
345     #somos-proya .img-container{
346         background-image: url("Fabi.png");
347         background-size: cover;
348         background-position: center center;
349         height: 500px;
350         width: 400px;
351     }
352
353     #nuestros-programas .programas{
354         display: flex;
355         justify-content: center;
356     }
357
358     #nuestros-programas p{
359         display: block;
360         margin-bottom: 30px;
361     }
362
363     #nuestros-programas h2{
364         font-size: 4em;
365     }
366
367     #nuestros-programas h3{
368         margin-top: 0;
369     }
370
371     #nuestros-programas .carta{
372         padding: 50px;
373         background-size: 100% 150px;
374         background-repeat: no-repeat;
375         background-position-y: 0;
376         background-color: rgb(50, 50, 50);
377         box-shadow: 2px 2px 10px rgba(0,0,0,0.5);
378     }
```

```
379
380     #caracteristicas .programas{
381         display: flex;
382         flex-wrap: wrap;
383         justify-content: center;
384     }
385
386     #caracteristicas p{
387         display: block;
388         margin-bottom: 30px;
389     }
390
391     #caracteristicas h2{
392         font-size: 4em;
393     }
394
395     #caracteristicas h3{
396         margin-top: 0;
397     }
398
399     #caracteristicas .carta{
400         padding: 50px;
401         background-size: 100% 150px;
402         background-repeat: no-repeat;
403         background-position-y: 0;
404         background-color: rgb(50, 50, 50);
405         box-shadow: 2px 2px 10px rgba(0,0,0,0.5);
406         max-height: 80vh;
407         max-width: 60vh;
408     }
409
410     .carta:first-child{
411         background-image: linear-gradient(
412             0deg,
413             rgba(0,0,0,0.5),
414             rgba(0,0,0,0.5)
415         )
416         ,url("media/front-end-cropped.jpg");
417
418     }
419
420     .carta:nth-child(2){
421         background-image: linear-gradient(
422             0deg,
423             rgba(0,0,0,0.5),
424             rgba(0,0,0,0.5)
425         )
426         ,url("media/full-stack-cropped.jpg");
427     }
428
429     .carta:nth-child(3){
430         background-image: linear-gradient(
431             0deg,
432             rgba(0,0,0,0.5),
433             rgba(0,0,0,0.5)
434         )
435         ,url("media/python-cropped.jpg");
436
437     }
438
439     #caracteristicas{
440         background-image: url("media/background-2.jpeg");
441         background-repeat: no-repeat;
442         background-size: 500px 400px;
443         background-position: calc(100vw - 500px) 120px;
444     }
445
446     #caracteristicas .container{
```

```
        text-align: initial;
    }

    #caracteristicas ul{
        margin-left: 100px;
    }

    #contacto h2{
        font-size: 5em;

    }

    #contacto button{
        font-size: 2em;
    }

    footer .container{
        justify-content: flex-end;
    }
    #contacto a{
        font-size: 2rem;
        text-align: left;
    }
}

@media (min-width: 1200px) {
    #caracteristicas{
        background-position-x: calc(100vw - 800px);
    }
}
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>PROYECTO BIAS</title>
    <link rel="stylesheet" href="index.css">
    <link
    rel="stylesheet"
    href="https://use.fontawesome.com/releases/v5.8.2/css/all.css"/>
</head>
<body>
    <header>
        <div class="container">
         <img class="logo" src="LogoBIAS.png">
            <nav>
                <a href="#somos-proya"> Que  es BIAS?</a>
                <a href="#nuestros-programas"> Por  qu  nuestro proyecto?</a>
                <a href="#caracteristicas"> Quienes  Somos?</a>
                <a href="#contacto">Contactanos</a>
            </nav>
        </div>
    </header>

    <section id="hero">
        <h1>Bienvenido al <br>Proyecto BIAS</h1>
    </section>

    <section id="somos-proya">
        <div class="container">
            <div class="img-container"></div>
            <div class="texto">
                <h2>  Qu  es BIAS?</h2>
                <p>BIAS es un proyecto innovador busca transformar la vida de
    personas con discapacidades motoras mediante sillas de ruedas controladas por
    se ales EEG. <br> Un dispositivo capta la actividad cerebral y la traduce en
```

```
    comandos, permitiendo a los usuarios dirigir la silla con sus pensamientos.
    Avanzar, girar o detenerse es posible, todo con la fuerza de la mente.<br> Esta
    tecnolog a promete mayor independencia y una mejor calidad de vida para quienes
    enfrentan barreras de movilidad. Un futuro donde la mente lidera el camino est
    m s cerca que nunca.</p>
33              </div>
34          </div>
35      </section>
36
37      <section id="nuestros-programas">
38          <div class="container">
39              <h2> Por  qu  nuestro proyecto?</h2>
40              <div class="programas">
41                  <div class="carta">
42                      <h3>Inclusi n</h3>
43                      <p><br><br><br>Buscamos la igualdad de oportunidades para
    aquellos que m s lo necesitan y generar una independencia nunca lograda hasta
    este momento, todo en vista de conseguir una vida m s plena e igualitaria.</p>
44                  </div>
45                  <div class="carta">
46                      <h3>Igualdad</h3>
47                      <p><br><br><br>Creemos que la igualdad es la base de una mejor
    sociedad no solo para garantizar que todos reciban iguales oportunidades si no
    para no perder gente talentosa que, de otra forma, no tendr a la oportunidad de
    desarrollar su potencial</p>
48                  </div>
49                  <div class="carta">
50                      <h3>Sentar un Precedente</h3>
51                      <p><br><br>Este proyecto busca crear una alternativa realista
    para la creaci n de una silla de ruedas EEG econ mica y accesible, un objeto el
     cual en este pa s no se comercializa</p>
52                  </div>
53              </div>
54          </div>
55      </section>
56
57      <section id="caracteristicas">
58          <div class="container">
59              <h2>  Quienes  Somos?</h2>
60              <div class="programas">
61                  <div class="carta">
62                      <h3>D az Meli n, Danilo</h3>
63                      <p><br>Dise o de Esquem ticos. Dise o estructural. Creaci n
    de P gina Web. Marketing y B squeda de Sponsors. Construcci n de Planos.
    Dise o de sistema de movimiento. Programaci n IA  <a href="https://instagram.
    com/_danilodiaz">
64                          <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:danilodiaz934@gmail.com">
65                              <i class="fab fa-google"></i></a><a href="https://www.
    linkedin.com/in/danilodiazmelion/">
66                                  <i class="fab fa-linkedin"></i></a> </p></div>
67                  </div>
68                  <div class="carta">
69                      <h3>Sojka, Santiago</h3>
70                      <p><br><br><br>Soldadura de PCBs. Redes Sociales. Edici n de
    videos. Creaci n de Contenido. Dise o de sistema de movimiento. Investigaci n.
     Programaci n IA. Programaci n de Conexi n para el Sensor EEG.<a href="https
    ://instagram.com/sojkaa.sant">
71                          <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:santiagosojka@gmail.com">
72                              <i class="fab fa-google"></i></a><a href="https://www.
    linkedin.com/in/santiago-sojka-817198271/">
73                                  <i class="fab fa-linkedin"></i></a> </p></p></div>
74                  </div>
75                  <div class="carta">
76                      <h3>Montenegro, Luciano</h3>
77                      <p><br>Conversi n de Esquem ticos a PCBs. Dise o  estructural.
     Dise o sistema de movimiento. Programaci n IA. Creaci n de contenido. <a href
```

```
    ="https://instagram.com/luchito_.montenegro">
78                      <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:nias.project.impa@gmail.com">
79                          <i class="fab fa-google"></i></a><a href="https://www.
    linkedin.com/in/luciano-montenegro-3215aa304/">
80                              <i class="fab fa-linkedin"></i></a> </p></p></div>
81              </div>
82              <div class="carta">
83                  <h3>De Blasi, Luca</h3>
84                  <p><br><br><br>Programaci n del sistema de movimiento.
    Construcci n de Planos. Dise o estructural. Programaci n IA.
85                      <a href="https://instagram.com/luca.deblasii">
86                      <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:nias.project.impa@gmail.com">
87                          <i class="fab fa-google"></i></a><a href="https://www.
    linkedin.com/in/luca-de-blasi-31164b304/">
88                              <i class="fab fa-linkedin"></i></a> </p></p></div>
89              </div>
90              <div class="carta">
91                  <h3>Adell, Nicol s Fabi n</h3>
92                  <p><br><br>Programaci n del Sistema de Movimiento. Preparaci n
    del Github. Redes Sociales. Programaci n IA. Programaci n de Conexi n para el
    Sensor EEG.<a href="https://instagram.com/nicolas.adell">
93                      <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:nias.project.impa@gmail.com">
94                          <i class="fab fa-google"></i></a><a href="http://www.
    linkedin.com/in/nicolas-adell-354508297">
95                              <i class="fab fa-linkedin"></i></a> </p></p></div>
96              </div>
97              <div class="carta">
98                  <h3>Gil Soria, Ian</h3>
99                  <p><br><br><br>Marketing y B squeda de Sponsors. Dise o de
    Esquem ticos y PCBs. Programaci n IA. Encargado de Carpeta de Campo y
    Documentaci n T cnica.<a href="https://instagram.com/ian_gilsooor">
100                     <br><br><div class="social-icons"><i class="fab fa-instagram"
    ></i> </a><a href="mailto:nias.project.impa@gmail.com">
101                         <i class="fab fa-google"></i></a><a href="https://www.
    linkedin.com/in/ian-lucas-gil-soria-a8090b2a8?utm_source=share&utm_campaign=
    share_via&utm_content=profile&utm_medium=android_app ">
102                             <i class="fab fa-linkedin"></i></a> </p></p></div>
103             </div>
104         </div>
105     </div>
106 </section>
107
108 <section id="contacto">
109     <h2>Contactanos</h2>
110     <div><a href="https://instagram.com/proyecto.bias">
111         <i class="fab fa-instagram"></i> </a> <p>@proyecto.bias</p></div>
112
113         <a href="mailto:bias.project.impa@gmail.com">
114             <i class="fab fa-google"></i></a><p>bias.project.impa@gmail.com</p>
115         <a href="https://tiktok.com/@proyecto_bias">
116             <i class="fas fa-music"></i></a> <p>@proyecto.bias (TikTok)</p>
117
118 </section>
119
120 <footer>
121     <div class="container">
122         <p>&copy; Proyecto BIAS 2024</p>
123     </div>
124 </footer>
125 </body>
126 </html>
```