Universidad Tecnológica Nacional

Proyecto Final

Guante de control remoto electrónico

Autores:

- *Berardi, Nahuel Matías*
- *Puig, Walter Ezequiel*
- *Ronchi, Sergio Juan Carlos*

Director:

- *Vicario, Sebastián*

Proyecto final presentado para cumplimentar los requisitos académicos para acceder al título de Ingeniero Electrónico en la

Facultad Regional Paraná

Diciembre de 2022

Declaración de autoría:

Yo/nosotros declaro/declaramos que el Proyecto Final “Guante de control remoto” y el trabajo realizado son propio/s. Declaro/declaramos:

- Este trabajo fue realizado en su totalidad, o principalmente, para acceder al título de grado de Ingeniero Electrónico, en la Universidad Tecnológica Nacional, Regional Paraná.
- Se establece claramente que el desarrollo realizado y el informe que lo acompaña no han sido previamente utilizados para acceder a otro título de grado o pre-grado.
- Siempre que se ha utilizado trabajo de otros autores, el mismo ha sido correctamente citado. El resto del trabajo es de autoría propia.
- Se ha indicado y agradecido correctamente a todos aquellos que han colaborado con el presente trabajo.
- Cuando el trabajo forma parte de un trabajo de mayores dimensiones donde han participado otras personas, se ha indicado claramente el alcance del trabajo realizado.

Firmas:

-
-
-

Fecha: 30/12/2022

Agradecimientos:

En primer lugar, deseamos expresar nuestro agradecimiento a Sebastián Vicario, nuestro profesor guía en este Proyecto Final. Él ha sabido acompañarnos y guiarnos en este trayecto tan importante de nuestra carrera, poniendo su tiempo a nuestra disposición, y brindando su vasto conocimiento y experiencia, sin dejar de lado su constante buen humor. ¡Gracias Sebastián!

A la comunidad educativa de la Universidad Tecnológica Nacional, Facultad Regional Paraná que nos brindaron el lugar y las herramientas necesarias para desarrollar este proyecto.

Berardi, Nahuel Matías

Puig, Walter Ezequiel

Ronchi, Sergio Juan Carlos

Universidad Tecnológica Nacional

Abstract

Facultad Regional Paraná

Ingeniero en Electrónica

Guante de control remoto electrónico

Berardi, Nahuel Matías

Puig, Walter Ezequiel

Ronchi, Sergio Juan Carlos

Abstract:

This article seeks for the incorporation of new technologies in remote applications that require performing repetitive tasks through a peripheral based on an electronic control glove.

The project consists in the elaboration of an electronic control glove equipped with movement sensors, a data storage system and a tactile user interface. It is sought that it be able to control other electronic devices in real time and on the other hand that can create, transmit and reproduce movement routines that are activated by a sensor of the receiving device. To achieve this, not only the glove is enough, it is necessary to implement a control

module that commands the receiving device, which in this case is a robotic arm, which is why an appropriate circuit must be made for its control.

For its elaboration, the communication and control system between both devices is based on the ESP32 wireless module, the data storage is done through SD memory reader modules, the mechanical system of the glove and the robotic arm were elaborated by means of 3D printing, Movement capture was done with gyro sensors and the module with accelerometer and gyroscope MPU6050. Finally, the user interface consists of a touch LCD screen controlled by an ATMEGA328P on the glove and a 2x16 LCD screen on the receiver plate incorporated into the arm.

The result obtained is a system that allows the user to remotely control the robotic arm in real time and store movement routines in the SD memory so that they can be reproduced when one of the sensors connected to it is activated. However, as the system is similar for any type of receiver, it has the potential to control other devices such as drones, robots, toys, and more.

Keywords:

Data

Memory

Movement

Routine

Wireless

Resumen:

Este artículo busca la incorporación de nuevas tecnologías en aplicaciones remotas y que requieran realizar tareas repetitivas mediante un periférico basado en un guante de control electrónico.

El proyecto consiste en la elaboración de un guante de control electrónico inalámbrico dotado de sensores de movimiento, un sistema de almacenamiento de datos y una interface de usuario táctil. Se busca que sea capaz de controlar otros dispositivos electrónicos en tiempo real y por otro lado que pueda crear, transmitir y reproducir rutinas de movimientos que sean accionadas mediante la activación de algún sensor del dispositivo receptor. Para lograr esto no solo basta con el guante, sino que es necesario implementar un módulo de control que comande el dispositivo receptor, el cual en este caso es un brazo robótico, es por esto que se debe realizar un circuito acorde para su control.

Para su elaboración, el sistema de comunicación y control entre ambos dispositivos está basado en el módulo inalámbrico ESP32, el almacenamiento de datos se hace mediante módulos de lectores de memoria SD, el sistema mecánico del guante y del brazo robótico fueron elaborados mediante impresión 3D, la captura de movimientos se realizó con sensores de giro y con un módulo que incorpora un acelerómetro y un giroscopio en un solo chip (MPU6050). Finalmente, la interface de usuario está constituida por una pantalla LCD táctil controlada por un ATMEGA328P en el guante y una pantalla LCD de 2x16 en la placa receptora incorporada al brazo.

El resultado obtenido es un sistema que le permite al usuario controlar remotamente en tiempo real el brazo robótico y almacenar rutinas de movimiento en la memoria SD para que sean reproducidas al activarse uno de los sensores conectados al mismo. Sin embargo, como el sistema es similar para cualquier tipo de receptor, tiene el potencial para controlar otros dispositivos como drones, robots, juguetes y más.

Palabras Clave:

Datos

Inalámbrico

Memoria

Movimiento

Rutina

Índice:

Lista de Figuras:

Lista de Tablas:

Lista de Abreviaciones y Símbolos

Lista de abreviaciones

LiPo	Litio y polímero
SCM	Supply Chain Management (Administración de la cadena de suministro)
SD	Secure Digital
LCD	Liquid Cristal Display (pantalla de cristal líquido)
PCB	Printed Circuit Board (placa de circuito impreso)
HMI	Human-Machine Interface (interfaz humano-maquina)
n°	Número
IMU	Inertial Measurment Units
MEMS	MicroElectroMechanical Systems
[°/s]	Grados por segundo
MPU	Multiple Process Unit (Unidad de proceso múltiple)
HTML	HyperText Markup Language (Lenguaje de Marcas de Hipertexto)
CSS	Cascading Style Sheets (hojas de estilo en cascada)
ROM	Read Only Memory (‘memoria de solo lectura’)
SDHC	Secure Digital High Capacity
MAC	Media Access Control
I2C	inter integrated circuits
ms	mili-segundos
ADC	Analog to Digital Converter (Conversor Analógico Digital)

Lista de símbolos

a	Aceleración
dV	Diferencia de velocidad
dt	Diferencia de tiempo
F	Fuerza
m	Masa
θ_X	Angulo de inclinación con respecto al eje X
θ_Y	Angulo de inclinación con respecto al eje Y
a_X	Acelerómetro del eje X
a_Y	Acelerómetro del eje Y
a_Z	Acelerómetro del eje Z
θ_{X_0}	Angulo con respecto al eje X anterior
θ_{Y_0}	Angulo con respecto al eje Y anterior
ω_X	Velocidad angular con respecto al eje X
ω_Y	Velocidad angular con respecto al eje Y
Δt	Intervalo de tiempo
$a_{(PWM)}$	Valor de la aceleración para luego generar la PWM
$a_{\max}$	Máximo valor del acelerómetro
$a_{\min}$	Mínimo valor del acelerómetro
k	Constante de ajuste
g_{x}	Valor de la velocidad angular sobre el eje X leído del giroscopio
g_{actual}	Valor de la velocidad angular en la medición actual leído del giroscopio
g_{anterior}	Valor de la velocidad angular en la medición anterior leído del giroscopio
t_{actal}	Valor del tiempo de la medición actual
t_{anterior}	Valor del tiempo de la medición anterior

Dedicado a:

Un trabajo de desarrollo es también fruto del reconocimiento y del apoyo vital que nos ofrecen las personas que nos estiman, sin las cuales no tendríamos la fuerza y energía que nos anima a crecer como personas y como profesionales.

Es por esto que deseamos dedicarle este informe a aquellas personas que han estado con nosotros a lo largo de todos estos meses brindándonos su apoyo de manera incondicional.

Berardi Nahuel:

Este proyecto lo dedico principalmente a mi mamá y mi papá, quienes me han brindado su apoyo y contención a lo largo de estos años, gracias a eso pude dedicarme por completo a esta etapa de mi vida. Hago una especial dedicatoria a mi hermana, que siempre está presente para motivarme y consolarme cuando lo necesito, es la persona que me apoya para seguir adelante, sin importar el tipo de problema que enfrente.

Finalmente, y no menos importante, a mis amigo y seres queridos por motivarme a continuar ¡gracias a todos!

Puig Walter:

Mi dedicatoria va principalmente para mi familia quienes me han brindado su amor y apoyo a lo largo de estos años para permitirme dedicarme al 100% a superar esta etapa de mi vida. Le hago una dedicatoria especial a Sol quien sin saberlo me ha sabido contener en las malas y me ha prestado un oído siempre que necesite hablar con alguien. Finalmente, y no menos importante a mi equipo de trabajo con los cuales hemos trabajado codo a codo para finalizar este proyecto y cumplir nuestras metas, ¡gracias a todos!

Ronchi Sergio:

A mis padres, Azucena y Pedro, que desde el Cielo están celebrando este momento.

A mi esposa Patricia, y a mis hijas Delfina y María Paz, nada podría funcionar sin ellas…

¡A mis amigos y seres queridos, gracias por motivarme siempre a seguir!

Capítulo 1: Introducción

1.1. Fundamentación del proyecto.

El proyecto propuesto es un guante de transmisión y control de datos, cuya finalidad es brindarle al usuario la posibilidad de crear fácilmente un conjunto de rutinas de trabajo que sean accionables mediante sensores o pulsadores sin la necesidad de programar cada uno de los movimientos que llevará a cabo el dispositivo electrónico que se desee controlar.

Inicialmente el proyecto fue planteado para aplicarse en sistemas de producción que requieran de un conjunto de rutinas de movimiento repetitivas en el tiempo y que sean accionadas por un sensor. Sin embargo, también se extiende a la posibilidad de controlar dispositivos de manera remota y en tiempo real, lo cual brinda la funcionalidad adicional de poder utilizarlo en aplicaciones de alto riesgo, incluso como un medio de entretenimiento (para controlar dispositivos como drones, vehículos de control remoto y periféricos para consolas o PC).

Para esto se propone la creación de un guante de control que estará dotado de varios sensores que detectarán los movimientos realizados por el usuario y un sistema de transmisión de datos que enviará esta información hacia el dispositivo electrónico que se desee controlar. La idea es que el guante le permita al usuario tanto la creación, como la gestión y el guardado de distintas rutinas de movimientos que deben ser leídas e interpretadas por otros dispositivos electrónicos para ejecutar una o varias acciones.

Para lo anterior, es imprescindible que el guante permita el control de dispositivos en tiempo real de forma remota, esto con la finalidad de llevar a cabo tareas manualmente o bien para darse una idea de cómo se comportarán algunas de las rutinas que se deseen implementar.

1.2. Público objetivo

El público objetivo de este proyecto está orientado a toda industria o institución que se vea en necesidad de controlar remotamente en tiempo real o mediante una rutina de movimientos algún dispositivo electrónico. Sin embargo, según la orientación que tome el proyecto en futuras versiones, podría tener demanda en distintas áreas enfocadas al entretenimiento.

Este prototipo brinda facilidades para realizar trabajos que emplean rutinas de movimientos repetitivos como los que se llevan a cabo en cadenas de producción, en dispositivos de maniobras (cocinero inteligente, cama de masaje inteligente, asistentes de laboratorio, etc.), permitiéndole al usuario la programación de las rutinas necesarias para ejecutar el trabajo deseado. A su vez también permite el control de dispositivos en tiempo real, lo cual permite al usuario utilizarlo en una amplia gama de aplicaciones que pueden ir desde su uso en trabajos potencialmente peligrosos hasta sistemas de entretenimiento.

1.3. Pruebas de concepto

Las prestaciones podrían garantizar, en general, un trabajo más eficiente. Cuando se lo usa en tiempo real de forma remota, un operario que efectúa un trabajo de riesgo, puede maniobrar en una situación de estrés mucho menor que si estuviera realizando un trabajo riesgoso de manera presencial, lo cual lleva a realizar mejor la tarea.

Para el caso en el que se realizan tareas repetitivas, el proceso se vuelve más económico al no requerir de un empleado que efectúe el trabajo manualmente y más confiable ya que reduce el margen de error por fatiga.

1.4. Análisis de mercado

1.4.1. Guante somatosensorial basado en arduino.

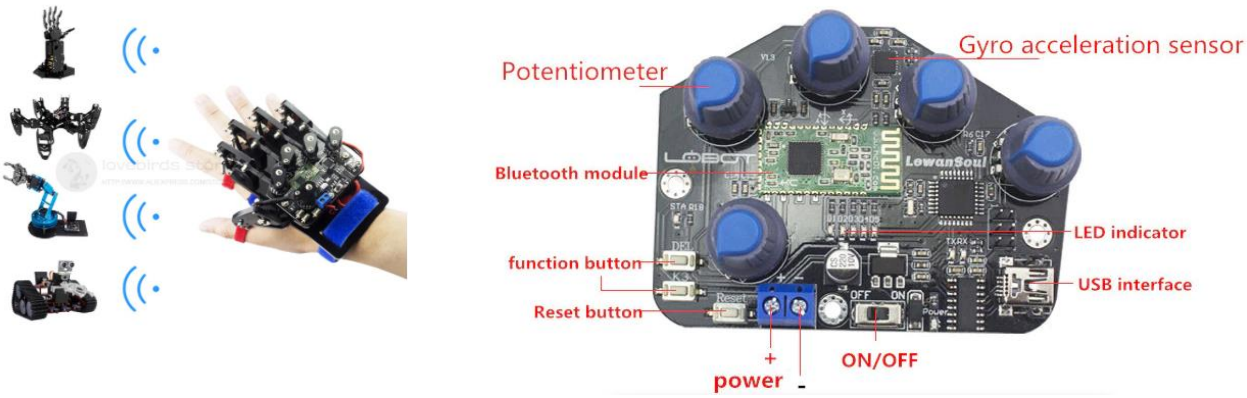


Figura 1: Guante somato sensorial

Se trata de un guante somato sensorial pensado para la comunicación y control de distintos tipos de dispositivos. Este modelo comercial permite el control de los dispositivos mostrados en la imagen mediante la aplicación de gestos predefinidos para tal fin.

Características:

- Guante mecánico de código abierto basado en Arduino; Modelo de chip único: ATmega328P
- Esquema abierto y código fuente;
- El módulo Bluetooth 4,0 integrado y el sensor de aceleración;
- Proporciona un gran número de rutinas de control que pueden controlar la palma del robot, el brazo robótico, el robot biónico, el coche inteligente

Parámetros del producto:

Modelo de microcontrolador: Arduino

Batería: 7,4 V 2S lipo-batería (integrada)

Tamaño: 147[mm] x 113[mm]

Peso: 0,26[kg]

Precio aproximado: 100 USD

El guante se usa para controlar el robot, el módulo Bluetooth del robot debe trabajar en HC-08. Basado en el Protocolo Bluetooth 4,0.

1.4.2. DEXTAROBOTICS DEXMO haptic feedback exoskeleton gloves

Este modelo lee los movimientos que se llevan a cabo con el guante capturando hasta 11 grados de libertad y además su tecnología le brinda al usuario la sensación de tacto cuando se lo utiliza en entornos de simulación de realidad virtual.

Figura 2: Dextarobotics dexmo

Para conseguir este efecto, los guantes utilizan un sistema que aplica un torque justo en la dirección contraria a la cual se efectúa el movimiento de la mano, es decir, para conseguir que tengas la sensación de que tus dedos chocan con algo, un mecanismo tira de ellos hacia atrás justo en el momento preciso.

Dexmo es el guante de retroalimentación de fuerza más portátil del mundo para investigadores, empresas y consumidores. Proporciona un toque natural e intuitivo que da un toque íntimo del mundo de RV verdaderamente inmersivo.

Figura 3: Dextarobotics dexmo

Dexmo es el primer guante portátil de retroalimentación de fuerza inalámbrica de dos manos disponible comercialmente del mundo, con captura de movimiento de mano y retroalimentación de fuerza para una experiencia interactiva de retroalimentación de fuerza más emocionante. Tiene aplicaciones en el área aeroespacial, formación industrial, investigación educativa, rehabilitación médica, modelado de simulación, socialización de juegos y más.

Características:

- **Habilidad de capturar movimientos:** Dexmo captura un rango completo de movimientos de la mano del usuario y posee dos grados de libertad en la captura de movimiento de los dedos, permitiendo capturar los movimientos de flexión y abducción de los mismos.
- **Fuerza de respuesta variable:** Le permite al usuario sentir el tamaño y la forma de cualquier objeto digital. Los motores detienen la rotación de los dedos de acuerdo al avatar digital con la que tiene interacción la mano dentro del mundo digital. Dexmo

también es capaz de generar una fuerza de salida variable, permitiendo al usuario detectar la diferencia en la rigidez de los objetos y detectar por ejemplo la diferencia entre una piedra y una pelota.

- **Simulación de múltiples capas de rigidez:** Al combinar el control de software con la capacidad de retroalimentación de fuerza variable, los desarrolladores pueden describir fácilmente la rigidez de un objeto a través de las librerías de Dexmo y lograr una sensación háptica compleja. Cuando hay un cambio en la rigidez, se pueden lograr sensaciones como presionar un botón y romper un huevo.
- **Mejora de la seguridad:** Dexmo está diseñado para ser seguro. La función de control de salida del par motor controla la fuerza ejercida por el dedo del usuario y desactiva todas las funciones del motor en circunstancias anormales. Además, la salida de torsión máxima de cada módulo de retroalimentación de fuerza de cada dedo está sujeta a 3[kg.cm] (0,3[N.m]). Un dedo humano puede proporcionar un torque promedio de 7[kg.cm] (0.7[N.m]). Entonces Dexmo nunca causará lesiones.
- **Soporte de plataforma:** Dexmo viene con su propio SDK, Lib Dexmo. Junto con el complemento de Unity que escribimos, Dexmo funciona con Oculus CV1, HTC Vive, PSVR, Hololens y potencialmente con cualquier otra solución de VR/MR compatible con el desarrollo de Unity.
- **Comunicación inalámbrica:** Comunicación inalámbrica con módulos de 2.4[GHz] con protocolos de comunicación optimizados, con latencia de 20-50[ms] y rangos de funcionamiento de 5 metros.
- **Ligero y portátil:** Gracias al diseño liviano de los motores, Dexmo pesa solo 320[g], mientras que los productos existentes que ofrecen funcionalidades similares generalmente pesan 10 veces más. Permite a los usuarios usar Dexmo cómodamente e ignorar su presencia cuando exploran en realidad virtual.
- **Hasta 6 horas de duración de la batería:** con los confiables y potentes módulos de batería LiPo de 2000[mAh], Dexmo puede funcionar de forma inalámbrica durante 6 horas con un uso normal.

  Precio aproximado: 800 USD

1.4.3. Guantes de control de movimiento Keywish Autos Smart

Figura 4: Guantes de control de movimiento Keywish Autos Smart

El dispositivo de guante de movimiento puede calcular la dirección de inclinación de la palma de 0 a 360 grados en tiempo real. De acuerdo con estos dos parámetros, los usuarios pueden controlar cualquier dispositivo bluetooth, que es adecuado para otros autos inteligentes SCM u otros hogares inteligentes.

Características:

- Guante deportivo de código abierto basado en Arduino; Modelo de chip único: ATmega328P
- Esquema y código de fuente abiertos;
- Posee un módulo Bluetooth y un sensor de aceleración;
- Proporciona una determinada cantidad de rutinas de control que pueden controlar el coche inteligente.

Parámetros del producto:

Modelo de microcontrolador: Arduino

Batería: 9[V]

Precio aproximado: 40 USD

1.5. Conclusiones

En el mercado existe una interesante variedad de guantes con características bien definidas y que se pueden orientar a aplicaciones diferentes. Existen prototipos que proponen una aplicación de los mismos como juguetes educativos, juegos interactivos o entretenimiento. El diseño implementado en este proyecto busca desde un inicio incorporar una tecnología que, si bien ya existe, no se ve implementada en los campos de interés mencionados. La intención de este proyecto es sentar las bases para diseñar y construir una herramienta orientada al área industrial y de automatización, sin embargo, por lo visto hasta ahora la naturaleza del diseño de este prototipo permite darle una gran flexibilidad y con ello la capacidad de poder ampliar su público objetivo y su área de aplicación.

Capítulo 2: Desarrollo

2.1. Introducción.

El proyecto consiste en un guante al cual se le implementan un conjunto de módulos que le permitirán al usuario la gestión y control de rutinas que se guardan y modifican dentro de una memoria SD. El diagrama en bloques del guante es el siguiente:

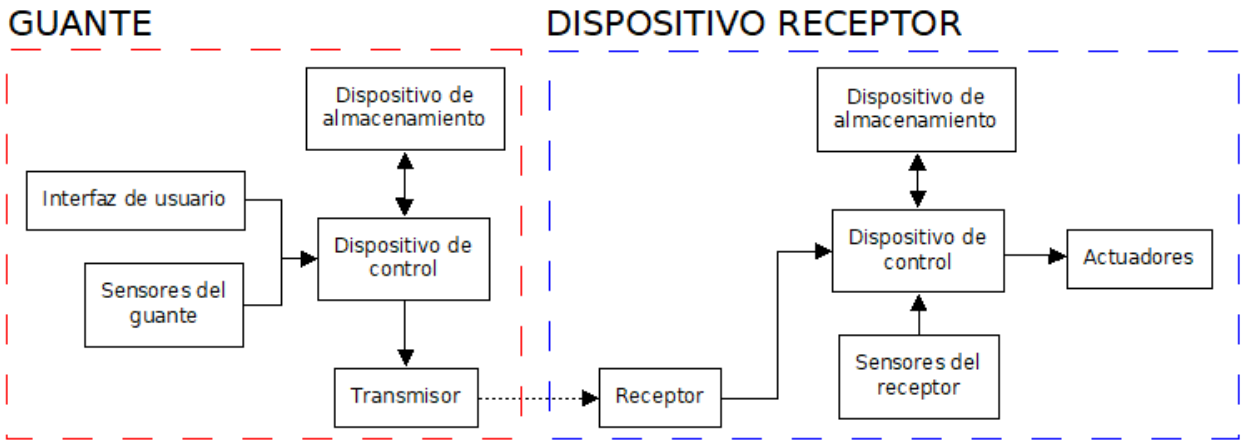


Figura 5: diagrama en bloques del guante

En términos generales, los bloques funcionan de la siguiente forma:

- Interfaz de usuario: El usuario interactúa con las opciones disponibles y crea un código acorde, el cual, será enviado al dispositivo de control para efectuar la tarea solicitada.
- Sensores del guante: Brindan información acorde a la posición de la mano y los dedos.
- Dispositivo de almacenamiento: Guarda la información acorde a los diferentes tipos de configuraciones y a las rutinas almacenadas.
- Dispositivo de control (guante): Gestiona todos los bloques, ejecuta las tareas solicitadas y evalúa posibles problemas o conflictos para prevenir errores.
- Emisor y receptor: Establecen el medio por el cual se podran comunicar el guante y el dispositivo receptor.
- Dispositivo de control (receptor): Despues de recibir una orden desde el guante, gestiona la memoria y los datos para poner en funcionamiento los actuadores propios del dispositivo receptor en caso de que así se requiera. Adicionalmente, detecta de manera constante los sensores propios conectados en el receptor, que al activarse ejecutarán la rutina vinculada al

mismo. Los datos dentro de la rutina deben ser modificados acorde al tipo de aplicación para lo cual se los utilice.

- Sensores del receptor: Puede tratarse de cualquier tipo de sensor o pulsador lógico. Cada sensor tiene asignado un número y al activarse el circuito de control debe reproducir la rutina vinculada al sensor activado.
- Actuadores (receptor): Son propios del dispositivo receptor conectado al dispositivo de control.

2.2. Funcionamiento general.

El guante posibilita la creación de hasta nueve rutinas, las cuales deben estar vinculadas a un número de sensor. Por ejemplo, si se crea la rutina uno y se la vincula al sensor tres, entonces cuando en el receptor se active el sensor tres se ejecutará la rutina uno desde la memoria SD del dispositivo receptor. En la siguiente imagen se observa dentro de la memoria SD que se han creado las rutinas 1, 2, 4 y 6.

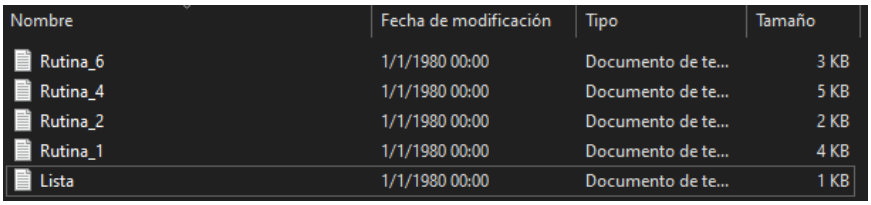


Figura 6: Contenido de la memoria SD

La memoria SD cuenta con un archivo llamado Lista.txt en el cual se listan los pares (n° rutina , n° sensor) y al crearse una rutina, por ejemplo la rutina 1, se crea un archivo 'Rutina_1.txt' en la cual almacenará los valores que lea en los sensores al momento de su creación. Al abrir el archivo Lista.txt podemos observar a que número de sensor están vinculadas las rutinas.

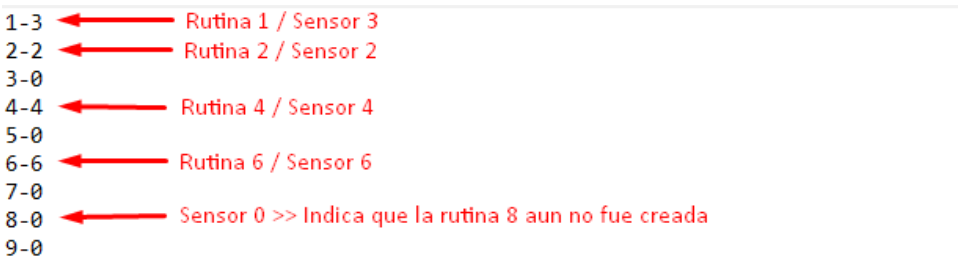


Figura 7: Contenido del archivo Lista.txt

Si abrimos los archivos de las rutinas, veremos los valores sensados durante su creación.

El guante permite realizar las siguientes operaciones:

1- Reproducción de datos en tiempo real: En este modo el guante lee los sensores y transmite sus valores al dipositivo receptor a una determinada frecuencia. El receptor por su parte, interpreta estos valores y ejecuta una acción acorde a los valores recibidos.

2- Transmisión de rutina: Se envia la configuración (n° rutina , n° sensor) y los valores leídos desde la memoria SD del emisor hacia la memoria SD del receptor.

3- Crear rutina: Crea una rutina y la almacena en la memoria SD del guante. Para ello modificará el par (n° rutina , n° sensor) y leerá los sensores del guante a una frecuencia determinada para almacenar esos valores en la memoria SD.

4- Cambiar sensor: Cambia el número del sensor vinculado a una rutina ya creada dentro de la memoria SD en el archivo Lista.txt, posteriormente transmite un código al dispositivo receptor para que haga lo mismo en su correspondiente memoria SD.

5- Borrar rutina: Elimina la rutina indicada. Para ello, primero actualiza el archivo Lista.txt haciendo que el valor del sensor vinculado a la rutina sea cero, luego elimina el archivo de la rutina indicada, y finalmente envía una orden al receptor para que haga lo mismo en su respectiva memoria SD.

6- Reproducir rutina: Esta modalidad permite seleccionar una rutina ya creada dentro de la memoria SD del guante y reproducirla en tiempo real una vez en el receptor sin guardarla en la memoria del mismo.

2.3. Interfaz de usuario:

Este bloque está compuesto por la pantalla LCD táctil ILI9341 y el microcontrolador ATMEGA328P. En principio se habia considerado utilizar una plantilla de pulsadores con una pantalla LCD de 2x16, pero la opción de la pantalla táctil nos otorga las siguientes prestaciones superadoras:

- Permite implementar una gran cantidad de botones en un espacio compacto, ahorrando en este caso la implementación de un PCB con al menos 15 pulsadores y una pantalla LCD de 2x16 o mayor.
- Brinda una interfaz más amigable con el usuario, permitiendo la visualización de la configuración realizada y el estado actual del guante.
- HMI más elegante, compacta y profesional.
- Totalmente controlable con el microcontrolador ATMEGA328P.

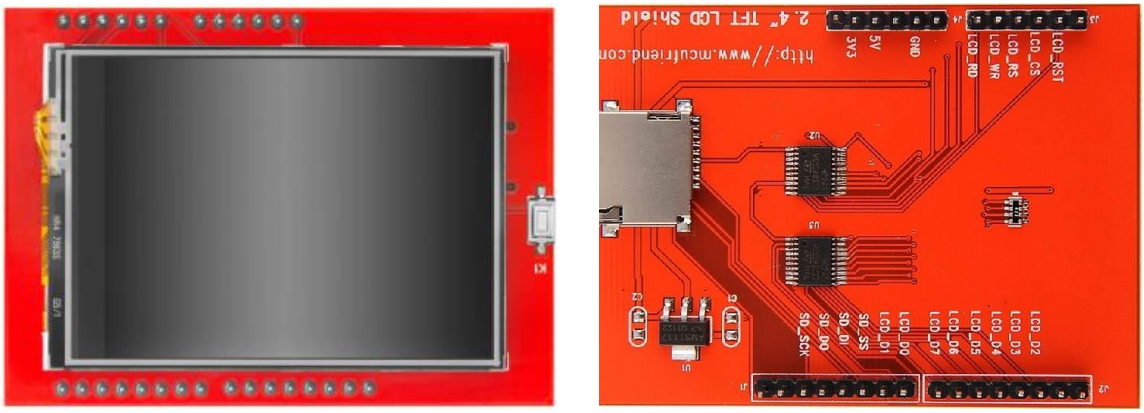


Figura 8: Pantalla LCD táctil ILI9341 vista de frente y reverso

Figura 9: Atmega328P

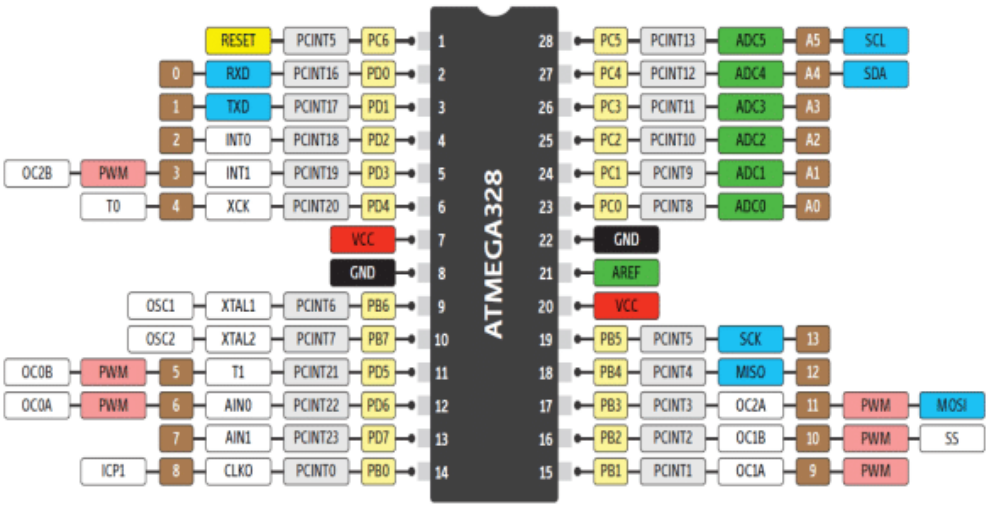


Figura 10: Pinout del Atmega328P

El esquema propuesto para la HMI es el siguiente:

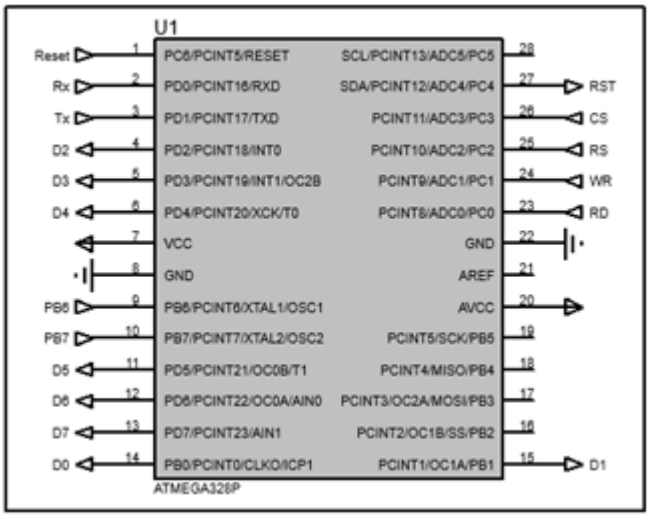


Figura 11: Conexionado del Atmega328P

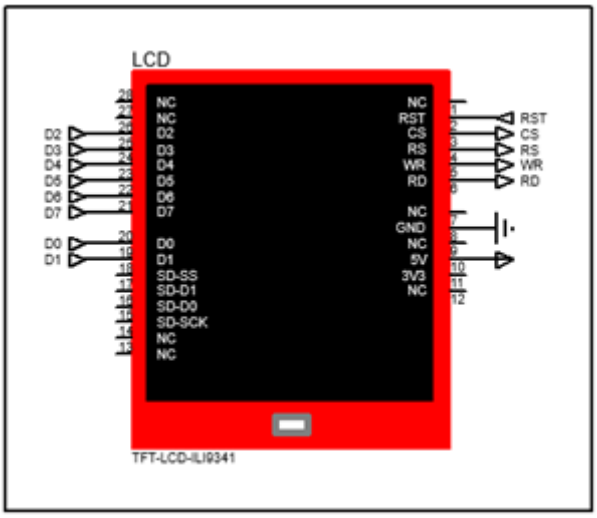


Figura 12: Conexionado de la pantalla LCD ILI9341

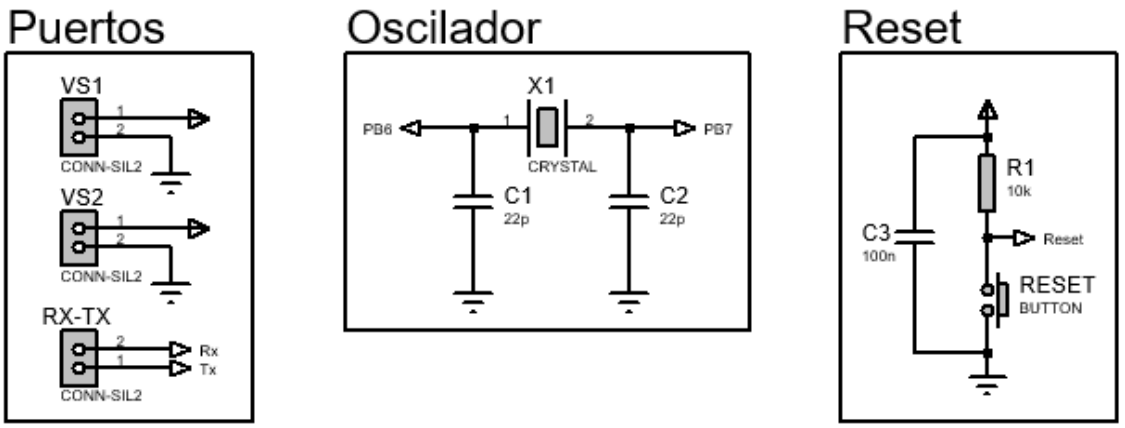


Figura 13: Puertos de alimentación, oscilador y botón de reset del HMI

El esquema es una conexión básica del microcontrolador con el circuito oscilador y el botón de reset. También cuenta con tres puertos, de los cuales dos de ellos son de alimentación (uno de entrada y uno de salida) y el otro es para comunicar mediante protocolo UART el ATMEGA328P con el ESP32 del bloque de control. Finalmente se conecta el microcontrolador con la pantalla LCD táctil, los pines RST, CS, RS, WR y RD se usan para reiniciar la pantalla y que la misma permita visualizar la información, adicionalmente con la ayuda del resto de los pines (D0 a D7) logramos controlar la componente táctil de la pantalla, que entre otras cosas retorna las coordenadas X, Y y la presión ejercida sobre la misma, permitiendo saber en qué zona de la pantalla se ejerce presión.

Una vez conocido el esquema implementado para la HMI, queda conocer el software que se implementa sobre el mismo. Para esto se presenta el menú de la pantalla:

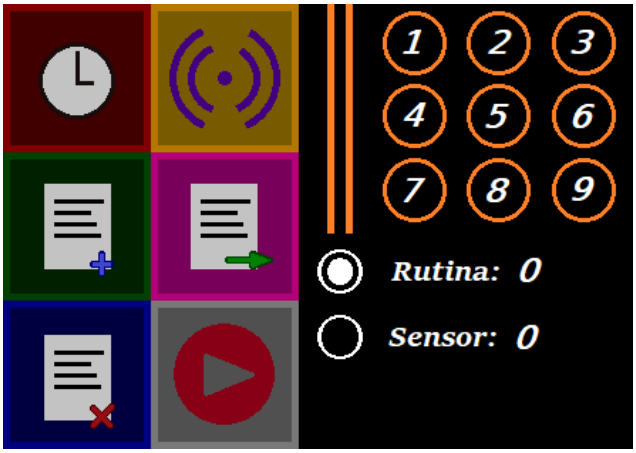


Figura 14: Diseño de la interfaz del HMI

En la mitad izquierda de la pantalla se hallan los botones correspondientes a las diferentes tareas disponibles, mientras que a la derecha tenemos un teclado numérico y un selector de rutina y de sensor, que nos permiten indicar sobre qué conjunto se realizará la tarea indicada. Para efectuar una tarea en concreto se deben seleccionar (de ser necesario) el número de la rutina y del sensor que se involucran en la tarea y posteriormente pulsar dos veces el icono correspondiente. Durante el primer pulso el botón se iluminará y en el segundo quedará marcado con un recuadro rojo indicando que la tarea esta activa.

Figura 15: Estados de los botones de la interfaz del HMI

Si no se presiona por segunda vez el botón, este se “apaga”. A continuación se presenta una tabla que indica el tipo de datos necesario para cada tarea.

Tarea	¿Requiere rutina?	¿Requiere sensor?
1 – Reproducción en tiempo real	NO	NO
2 – Transmitir rutina	SI	NO
3 – Crear nueva rutina	SI	SI
4 – Cambiar sensor	SI	SI
5 – Borrar rutina	SI	NO
6 – Reproducir rutina	SI	NO

Finalmente se indica la tarea asociada a cada uno de los iconos presentados en la pantalla:


Reproducción en tiempo real


Crear nueva rutina


Borrar rutina


Cambiar número de sensor


Transmitir rutina


Reproducir rutina

La lógica dentro del comportamiento de la HMI queda detallada en el siguiente diagrama:

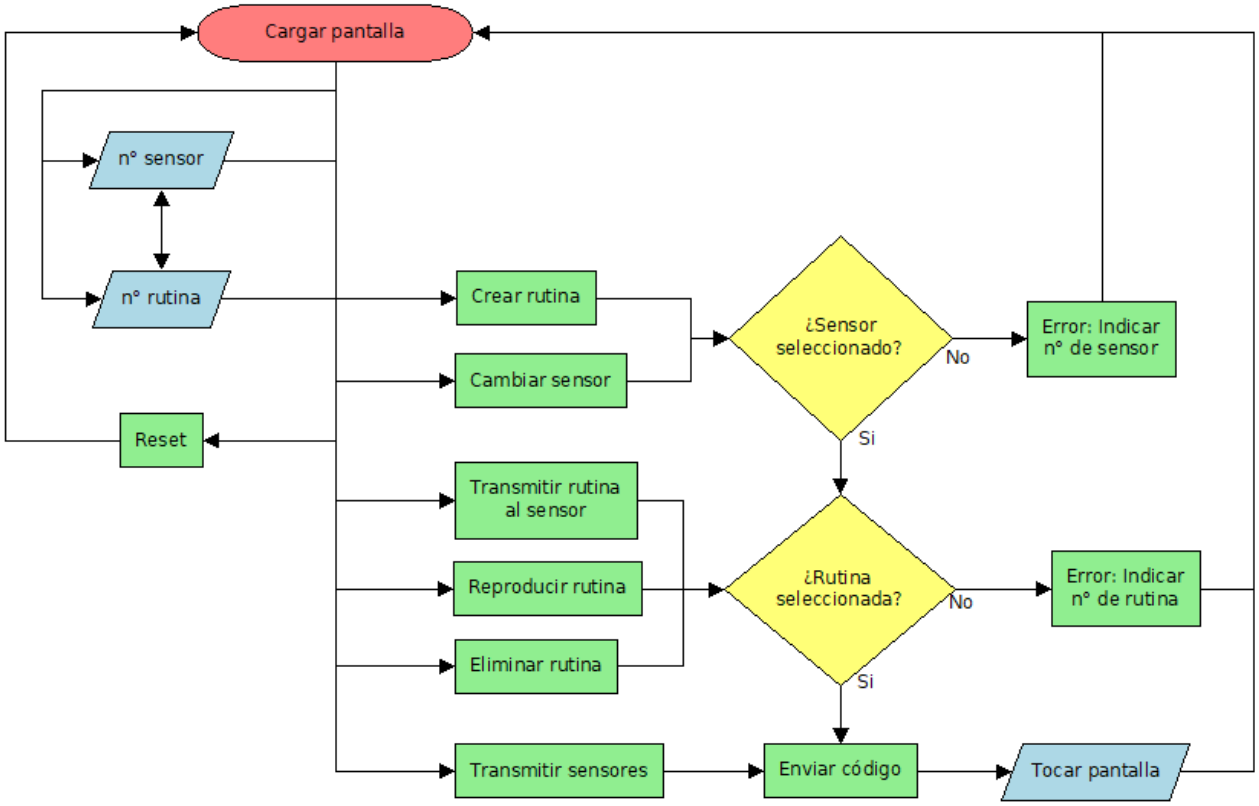


Figura 16: Diagrama de flujo de la interfaz del HMI

Inicialmente se carga la pantalla del menú principal. Desde este ese punto el usuario puede optar por modificar el número de sensor, el número de rutina, el botón de reset o bien presionar dos veces alguno de los botones de tareas. Luego de intentar activar alguna tarea, se evalúa si se han llenado los campos necesarios para cada una de ellas y en caso afirmativo envia un código acorde mediante comunicación I2C hacia el circuito de control. En caso contrario se podra visualizar en la pantalla un mensaje indicando el campo que falta completar para efectuar la tarea solicitada. Para finalizar la tarea solicitada, basta con volver a presionar la pantalla en cualquier lugar, esto enviará un código de finalización hacia el circuito de control y reiniciará la pantalla LCD.

El código enviado consiste en tres números separados por guiones, el primero corresponderá a la tarea que se desea ejecutar, el segundo es el número de la rutina involucrada y el tercero el número del sensor. La única excepción sera el código de cierre para indicar que se desea terminar con la tarea solicitada. Este último código sera 0-0-0.

2.4. Sensores del guante.

La función principal de los sensores en este proyecto es el de capturar los movimientos que el usuario efectúe al mover la mano o los dedos. Por un lado, para detectar la flexión en los dedos las dos mejores dos opciones disponibles son resistencias flexibles y módulos de sensores de giro. Se pretende analizar cada opción y justificar la elección tomada.

Resistencias flexibles:

Pros:

- Son mucho más simples de manipular y se pueden colocar directamente sobre el guante para que funcionen.

Contras:

- Requieren una conexión en serie de otra resistencia en caso de que se desee leer la flexión por variación de tensión y no de corriente.
- Son considerablemente más costosas que los sensores de giro.
- Son algo frágiles, por lo cual pueden quebrarse por el desgaste al usarlas.

Figura 17: Resistencia flexible

Módulos de sensores de giro:

Pros:

- Considerablemente más económicos que las resistencias flexibles.
- No requieren componentes externos, el valor de interés puede obtenerse directamente de la salida del sensor.
- Compactos, precisos y con un buen rango de funcionamiento.

Contras:

- Debe suministrarse 3.3[V] a cada uno de los dispositivos para que funcionen.
- Debe considerarse que no tienen tope, por lo cual, al girar y sobrepasar su valor máximo, conmutará al valor mínimo y viceversa.
- Requieren implementar un mecanismo externo para lograr rotar su eje al realizar el movimiento de los dedos.

Figura 18: Sensor de giro

Vistas las dos propuestas, se optó por el sensor de giro. La razón principal es que si bien tienen la desventaja de requerir del diseño de un mecanismo externo que permita la rotación del eje del sensor al flexionar un dedo, esto puede tomarse como una posibilidad para escalar el proyecto y diseñar a futuro un mecanismo más complejo y que permita el uso de más sensores de giro, permitiendo por ejemplo la lectura del movimiento lateral de los dedos y aumentando así los grados de libertad y la cantidad de señales que pueden interpretarse. El diseño y la impresión 3D de los módulos para conectar los sensores se detallan al final del capítulo.

Ya analizados los sensores de los dedos, queda pendiente medir la posición y rotación de la mano en el espacio con respecto al codo. Para ello también optamos por dos modelos. El primero y más sencillo es simplemente conectando algunos potenciómetros en las zonas próximas al codo y la muñeca, así el movimiento y giro de los mismos generarán las variaciones que deseamos medir para saber que estamos efectivamente flexionando el codo o bien rotando la muñeca. La segunda opción, que finalmente fue la adoptada en este proyecto, pese a resultar más compleja, pero mucho más interesante es la utilización de un módulo que cuenta internamente con un sistema de giroscopios y acelerómetros con los cuales podamos saber exactamente como se está produciendo el movimiento de la mano en el espacio.

El módulo escogido es el MPU6050. Es una unidad de medición inercial o IMU (Inertial Measurment Units) de 6 grados de libertad (DoF), este dispositivo combina un acelerómetro de 3 ejes con un giroscopio de 3 ejes. El acelerómetro mide la aceleración gravitacional y el giroscopio la velocidad rotacional que experimenta el módulo. Este

dispositivo se usa para determinar la orientación de un objeto en movimiento en el espacio (para nuestro caso particular será la mano).

La comunicación del módulo es por I2C, lo que le permite trabajar con la mayoría de microcontroladores disponibles en el mercado. Los pines SCL y SDA tienen una resistencia pull-up en la placa para una conexión directa al microcontrolador.

El acelerómetro mide la fuerza gravitacional y puede detectar pequeños movimientos o vibraciones.

En física, la aceleración es la variación de la velocidad por unidad de tiempo es decir razón de cambio en la velocidad respecto al tiempo: a=dV/dt.

Por otro lado, la segunda ley de Newton indica que, en un cuerpo con masa constante, la aceleración del cuerpo es proporcional a la fuerza que actúa sobre él mismo: a=F/m.

Este segundo concepto es utilizado por los acelerómetros para medir la aceleración. Los acelerómetros internamente tienen un MEMS (MicroElectroMechanical Systems) que de forma similar a un sistema masa resorte permite medir la aceleración.

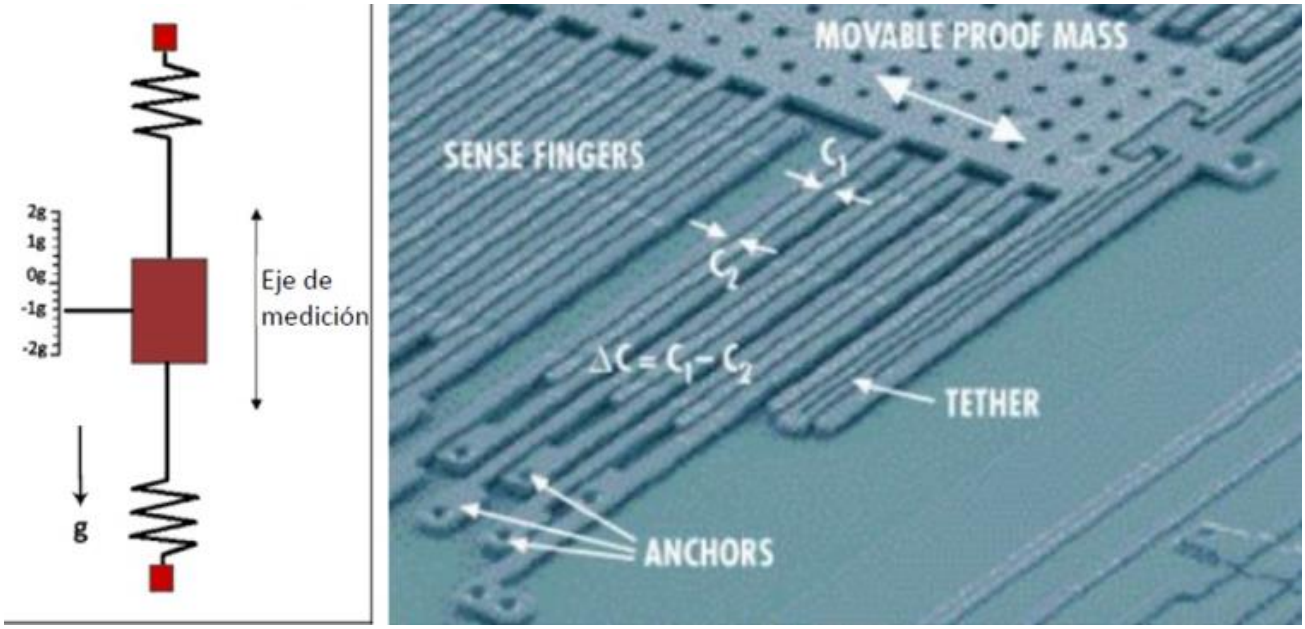


Figura 19: a) Principio de funcionamiento del Acelerómetro, b) Arquitectura interna del Acelerómetro

A pesar de que no exista movimiento, el acelerómetro siempre estará sensando la aceleración de la gravedad. Esto es particularmente útil, ya que al integrar la aceleración nos permite medir de forma indirecta la velocidad del movimiento y la posición del objeto.

Para que un eje en particular tome valores máximos, éste debe estar en paralelo al vector fuerza de gravedad. Si alguno de los ejes apunta en el mismo sentido que la fuerza de gravedad (hacia abajo) adquiere un valor de -9.81, en cambio si apunta en sentido opuesto (hacia arriba) adquiere un valor de 9.81.

Si el eje es perpendicular a la dirección de la gravedad, su acelerómetro será 0.

El giroscopio por otro lado, mide la velocidad angular sobre cada uno de los ejes del MPU6050 en [°/s]. En física la velocidad angular es la tasa de cambio del desplazamiento angular sobre unidad de tiempo o, en otras palabras, que tan rápido gira un cuerpo sobre su propio eje.

Este dato es particularmente útil para detectar el sentido de movimiento de un objeto en el espacio. Si el sensor está estático, entonces todos los giroscopios medirán cero.

Los giroscopios utilizan un MEMS (MicroElectroMechanical Systems) para medir la velocidad angular usando el efecto Coriolis.

El MPU6050 tiene tres ejes sobre la placa: los ejes x, y, z. Los mismos se usan para representar el sentido positivo del acelerómetro y el giroscopio vinculado a cada eje. Se presenta una imagen representativa de esto último:

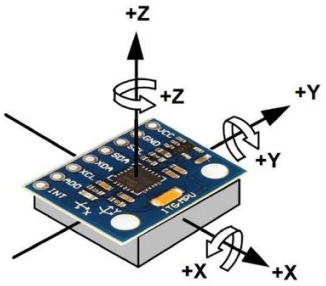


Figura 20: Orientación de los ejes del MPU6050

La implementación del MPU6050 requiere conocer el funcionamiento general que se ha explicado previamente pero también necesita una calibración previa para corregir errores de offset generados por inclinaciones no deseadas (como el de la soldadura del chip MPU a su módulo o del montaje del módulo al circuito) y errores de medición acumulativos como los que se generan en los giroscopios en ciclos muy largos de mediciones. Todos estos errores se corrigen calibrando por código algunas variables internas del MPU6050 con la ayuda de las librerías que se diseñaron para el mismo.

Para la lectura y adquisición de datos, nos interesa obtener los ángulos de inclinación y de giro que presenta el MPU6050.

Para calcular el ángulo de inclinación, asumiremos que estamos en un plano X-Z e inclinamos el MPU6050 un ángulo θ, dicho ángulo se calcula de la siguiente forma:

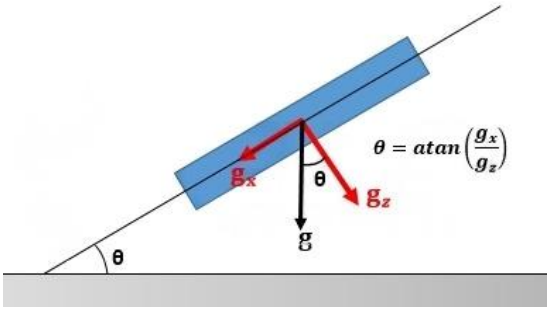


Figura 21: Cálculo de la componente del acelerómetro

Lo anterior nos sirve para calcular el ángulo en un plano 2D, pero, para calcular los ángulos de inclinación en un espacio 3D, tanto en X como en Y, usamos las siguientes fórmulas:

$$\theta_x = \tan^{-1}\left(\frac{a_x}{\sqrt[2]{{a_Y}^2 + {a_z}^2}}\right)$$

$$\theta_y = \tan^{-1}\left(\frac{a_y}{\sqrt[2]{{a_x}^2 + {a_z}^2}}\right)$$

Tener en cuenta que el resultado esta dado en radianes.

También es importante considerar que, si sólo trabajamos con el acelerómetro, éste es susceptible a las aceleraciones producto del movimiento del MPU o a fuerzas externas, pero en tiempos largos el ángulo no acumula errores.

Para el cálculo de los ángulos de rotación, se trabaja con los valores obtenidos de los giroscopios. El giroscopio nos entrega la velocidad angular, y para calcular el ángulo actual necesitamos integrar la velocidad y conocer el ángulo inicial. Esto lo hacemos usando la siguiente fórmula:

$$\theta_x = \theta_{x_0} + \omega_x \Delta t$$

$$\theta_y = \theta_{y_0} + \omega_y \Delta t$$

Tener en cuenta que θx es el ángulo que gira el eje X sobre su propio eje. En la siguiente imagen se observa que la velocidad angular es perpendicular al plano de rotación.

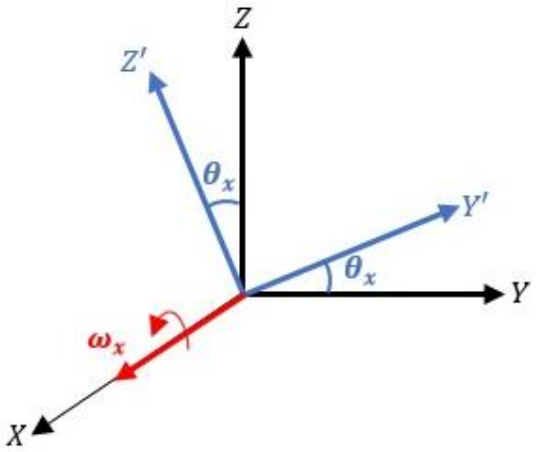


Figura 22: Variación de los ejes del acelerómetro cuando se lo rota sobre un eje

Es muy importante considerar que esta medida no es exacta ya que produce un error acumulativo en el tiempo (incluso varía cuando no se mueve el ángulo), o si se gira cierto ángulo y luego se regresa a la posición original el ángulo que medimos no es el inicial. Esto se debe a que al integrar la velocidad angular y sumar el ángulo inicial hay un error producto de la mala medición del tiempo o del ruido en la lectura del MPU. Este error, por más pequeño que sea, se va acumulando en cada iteración y creciendo, y es conocido como DRIFT.

Para disminuir el drift existen varios métodos, la mayoría aplican filtros para eliminar el ruido de las lecturas del sensor. También se pueden usar otros sensores como magnetómetros o acelerómetros y con los ángulos calculados con éstos mejorar el cálculo del giroscopio.

Un filtro muy usado para esto es el filtro de complemento, el cual es de fácil implementación y combina el ángulo calculado por el giroscopio con el ángulo calculado por el acelerómetro. La ecuación para calcular el ángulo usando el filtro de complemento es:

$$ángulo = 0.98\left(ángulo + \boldsymbol{\omega_{giroscopio}}\, dt\right) + 0.02(\boldsymbol{ang_{acelerómetro}})$$

De esta forma el ángulo del acelerómetro está pasando por un filtro pasa bajos, amortiguando las variaciones bruscas de aceleración; y el ángulo calculado por el giroscopio tiene un filtro pasa altos teniendo gran influencia cuando hay rotaciones rápidas.

Podemos probar también con otros valores diferentes a 0.98 y 0.02 pero siempre deben de sumar 1.

Conocidas las bases de la programación del MPU resta realizar las conexiones eléctricas correspondientes a la alimentación, a las líneas de comunicación con el microcontrolador y a los sensores de giro de cada uno de los dedos. El esquema es el siguiente:

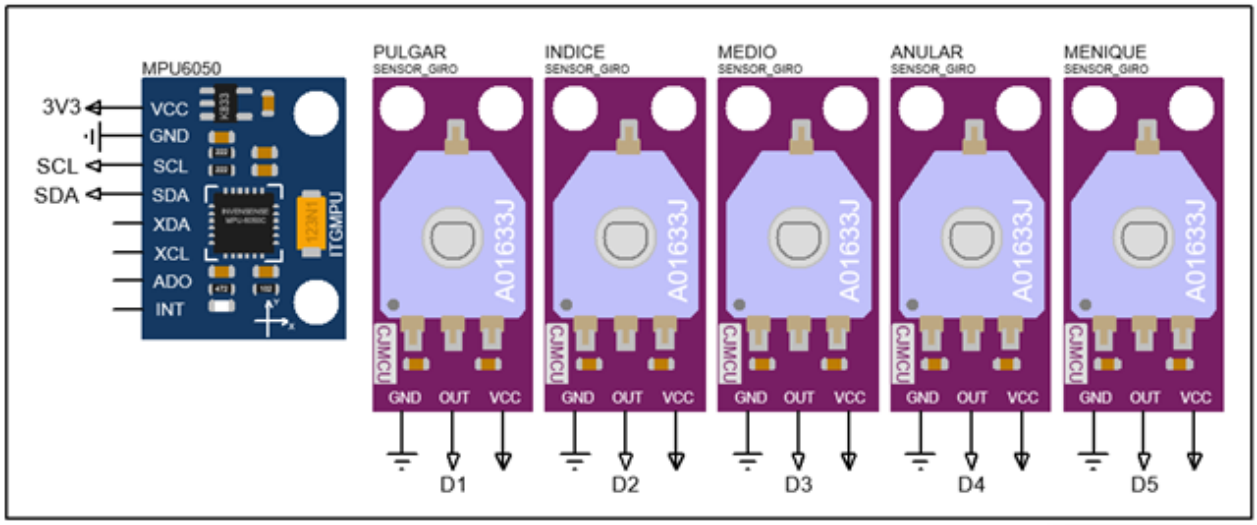


Figura 23: Conexionado del MPU y de los sensores de giro

2.5. Dispositivo de almacenamiento:

A la hora de evaluar las alternativas en cuanto a dispositivos de almacenamiento, surgen las alternativas de optar por un servidor web o bien por una memoria.

Por un lado, una página de internet (servidor web) permite guardar archivos o datos a través de internet. Este servicio es administrado por un proveedor informático a través de versiones gratuitas o por una tarifa comercial. La capacidad de almacenamiento en línea dependerá de lo que el usuario requiera (en caso de ser un servicio pago), o bien si es gratuito según lo considere el proveedor (en el caso de servidores gratuitos).

Pros de utilizar un servidor web:

- Permite compartir los archivos entre los usuarios sin tener que ocupar espacio de almacenamiento en dispositivos físicos de forma segura.

- Puede realizar copias de seguridad con la intención de preservar información importante y recuperarlos posteriormente sin complicaciones.
- No requiere montajes de ningun tipo de dispositivo extra.

Contras:

- Se necesita conocimientos profundos de HTML, CSS y de javascript para poder desarrollar la página web y poder hacer una interfaz amigable con el usuario o bien contratar un servicio para desarrollar páginas web.
- Al ser un sistema basado en la nube, es indispensable contar con acceso a internet para poder acceder a los archivos. Si el internet que se esta utilizando es lento, probablemente tenga inconvenientes a la hora de querer ver o descargar archivos almacenados. Y sencillamente, si no tienes internet, no puedes acceder de ninguna manera.
- Requiere contratar un servicio de hosting que provea el dominio de la pagina web. Además al ser un sistema que requiere bastante mantenimiento, los proveedores de este servicio pueden agregar costos según el volumen de subidas o descargas de la nube.

Por otro lado, existe la alternativa de implementar una memoria SD como dispositivo de almacenamiento, una tarjeta SD (Secure Digital) es una tarjeta de memoria flash extraíble de dimensiones reducidas generalmente utilizadas para almacenar información digital, como programas y archivos. Las tarjetas SD se utilizan comúnmente en teléfonos celulares y en otros dispositivos portátiles para ampliar el almacenamiento de la memoria ROM

Pros de utilizar tarjetas de almacenamiento:

- Son pequeñas y casi no ocupan espacio.
- Formato universal compatible con todos los dispositivos.
- Intercambiables entre dispositivos.
- Alta velocidad de acceso a la información.
- Precios asequibles para capacidades intermedias.
-

Contras:

- Si se pierde o se daña la tarjeta se pierde la información.
- Las de mayor capacidad son excesivamente caras.
- Existen más de ocho tipos diferentes de tarjetas con usos distintos.

Los requisitos necesarios para la implementación de un servidor web en comparación con las prestaciones que ofrece hace que no tenga sentido su implementación, razón por la cual se optó por la implementación de las memorias SD. Las opciones que se encuentran en el mercado prácticamente no varían en cuanto a las prestaciones, por lo cual no es demasiado importante cual de los módulos adaptadores de memoria se elija, los que se implementan en esta aplicación son:

Figura 24: Modulo lector de memoria SD transmisor

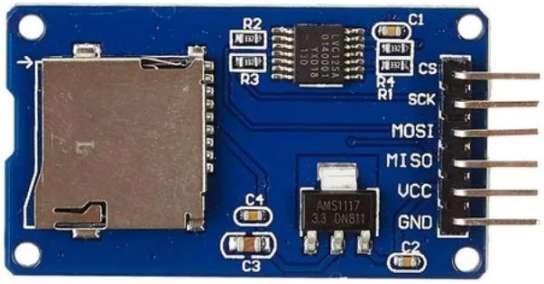


Figura 25: Modulo lector de memoria SD receptor

Los módulos permitirán conectar al dispositivo de control una tarjeta de memoria SD, de forma que éste pueda almacenar grandes cantidades de datos. Esto es ideal para realizar el registro de los datos de los sensores medidos (dataloggers).

Cada módulo está diseñado para acceder a la memoria SD mediante una interfaz SPI estandar, lo cual lo hace muy versátil ya que nos permite usarlo con una amplia variedad de microcontroladores y tarjetas de desarrollo. Son compatibles con tarjetas Micro SD, tarjetas Micro SDHC (tarjeta de alta velocidad) y poseen un regulador de voltaje de 5V o 3,3V.

La memoria requiere ser formateada con formato FAT32 para funcionar con el módulo.

Tiene un total de seis pines: VCC, MOSI, MISO, SCK, CS, GND. MOSI es el pin por el que se transmiten datos hacia el otro módulo, SCK es el pin que rige la velocidad a la que se transmite cada bit, CS es el pin que señala que esta seleccionado el chip, MISO es el pin por el que se reciben los datos desde el otro módulo.

En la memoria SD por un lado se crea el archivo de texto “Lista.txt” donde se guarda la informacion que vincula las rutinas creadas con sus respectivos sensores.

El otro tipo de archivos almacenados en la memoria SD son las rutinas “Rutina_X.txt” donde se encuentran registrados los valores de los sensores del guante, muestreados a una frecuencia dada.

El orden en el que se realiza el almacenamiento es siempre el mismo: primero los acelerómetros (en el eje X, en Y y en Z), luego los giroscopios (en el eje X, en Y y en Z) y por último los valores de los sensores de los dedos del D1 al D5 (pulgar, indice, medio, anular y meñique).

Ya conociendo las caracteristicas y el funcionamiento general del módulo y el rol que cumple dentro del proyecto se procede a mostrar el esquema de conexión del mismo.

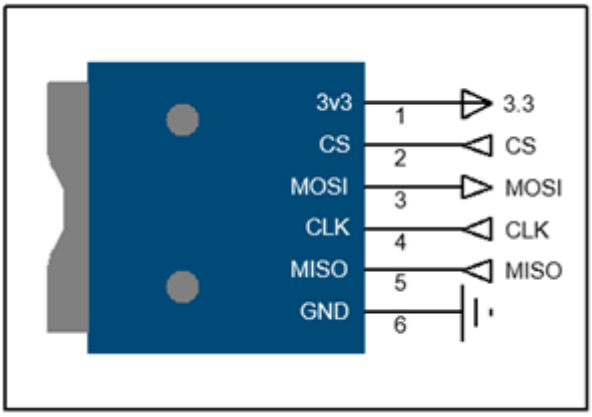


Figura 26: Conexionado del módulo de memoria SD

2.6. Transmisor/receptor

Como los dispositivos ESP32 cuentan con una antena para conexión inalámbrica, además de cumplir con el rol de circuito de control, también cuentan con los medios para cumplir el rol de circuito transmisor y receptor. Para la comunicación entre dos dispositivos ESP32, estos modelos cuentan con un protocolo de comunicación llamado ESPNOW.

ESPNOW es un protocolo de comunicación entre varios dispositivos creados por Espressif. Su funcionamiento requiere de emparejamiento de los dispositivos, por lo cual el “maestro” requerirá la dirección MAC del resto de dispositivos, pero una vez hecha la conexión será automática.

ESP32 soporta las siguientes características:

- Comunicación unicast encriptada y sin encriptar.
- Se pueden mezclar clientes con encriptación y sin encriptación.
- Permite enviar hasta 250-bytes de carga útil.
- Se pueden configurar callbacks para informar a la aplicación si la transmisión fue correcta.
- Largo alcance, pudiendo superar los 200[m] en campo abierto.

Pero también tiene sus limitaciones, las cuales son:

- El número de clientes con encriptación está limitado. Esta limitación es de 10 clientes para el modo Estación, 6 como máximo en modo punto de acceso o modo mixto.
- El número total de clientes con y sin encriptación es del 20.
- Sólo se pueden enviar 250 bytes como máximo.

Para esta aplicación solo requerimos una conexión maestro-esclavo unidireccional y dadas las características de transmisión, este dispositivo es óptimo y logra cumplir con todas las prestaciones requeridas. Para una conexión maestro-esclavo, el dispositivo maestro requiere la dirección MAC del ESP32 receptor para realizar el enlace de conexión.

Cada vez que se transmita un dato, parpadea alguno de los leds indicadores. Si el dato se envía correctamente y llega al dispositivo receptor, se le envía un breve pulso al led indicador verde. En cambio, si el dato se transmite, pero no llega a destino, se le envía un pulso al led indicador rojo indicando que la comunicación está fallando.

Lo último por mencionar es que se usa un protocolo propio de comunicación y se separa la transmisión de una tarea en tres fases.

Durante la primera fase (de apertura), la estructura de datos solo contiene la información referente a la orden solicitada, esto es, el código enviado por el guante (tarea-rutina-sensor) y además un booleano que indica que estamos en la fase 1. En esta fase, el ESP32 receptor realiza los preparativos necesarios para las demás. Esta es la única fase requerida para las tareas 4 (cambiar sensor) y 5 (borrar rutina).

Para la segunda fase (de trabajo), se transmite la información ya sea de los sensores o de los archivos de las rutinas. Durante esta fase, el ESP32 receptor guardará los datos en la memoria SD receptora o bien accionará los actuadores del receptor.

Por último, en la tercera fase (de cierre), el ESP32 receptor finaliza las configuraciones necesarias en los archivos y vuelve a su rutina normal.

Tras haber completado cualquiera de las órdenes de forma exitosa, los leds indicadores van a parpadear de forma alternada indicando la finalización de la tarea solicitada.

2.7. Dispositivo de control

Si bien en el mercado existen una gran variedad de microcontroladores que nos permitan realizar las tareas de gestión y control que necesitamos para conectar todos los bloques ya expuestos, como por ejemplo el Raspberry pi, el Stm32f411 o el Black Pill, en este proyecto se implementa como dispositivo de control (tanto en el emisor como en el receptor) el ESP32, ya que nos brinda todas las prestaciones necesarias para cumplir con las tareas establecidas, las siguientes son:

- Alimentación de 3.3[V] a 5[V] (los puertos de entrada soportan 3.3[V])
- Puertos de entrada analógicos suficientes para conectar los sensores de giro
- Puertos para comunicación UART (conexión con el ATMEGA328P y la pantalla TFT-LCD)
- Puertos para comunicación I2C (conexión con el sensor MPU6050)
- Puertos para comunicación SPI (conexión con el módulo lector de memoria SD)
- Comunicación inalámbrica entres dispositivos ESP32 (protocolo ESPNOW)

Se presenta a continuación el pinout del modelo del esp32 que se implementará tanto en la placa del circuito emisor como en la placa del circuito receptor.

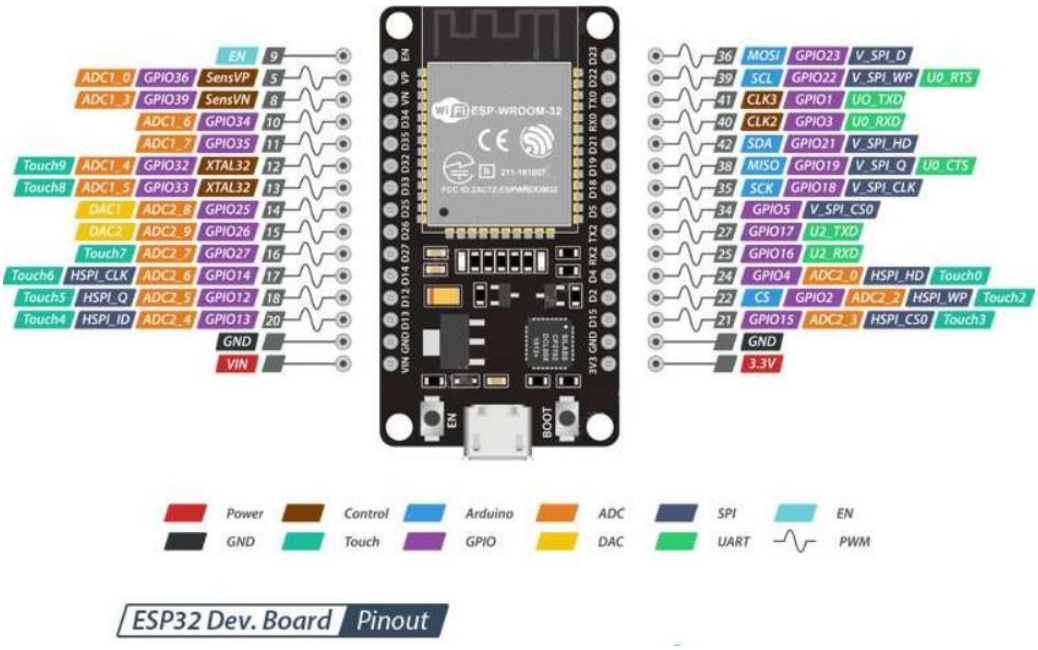


Figura 27: Pinout del ESP32 transmisor

A partir de este punto es necesario diferenciar entre el comportamiento del ESP32 del circuito emisor y el ESP32 del circuito receptor, ya que cada uno desempeña tareas diferentes, primero se analiza el comportamiento del ESP32 emisor y se presenta el esquemático y el diagrama de flujo del mismo para analizar su comportamiento:

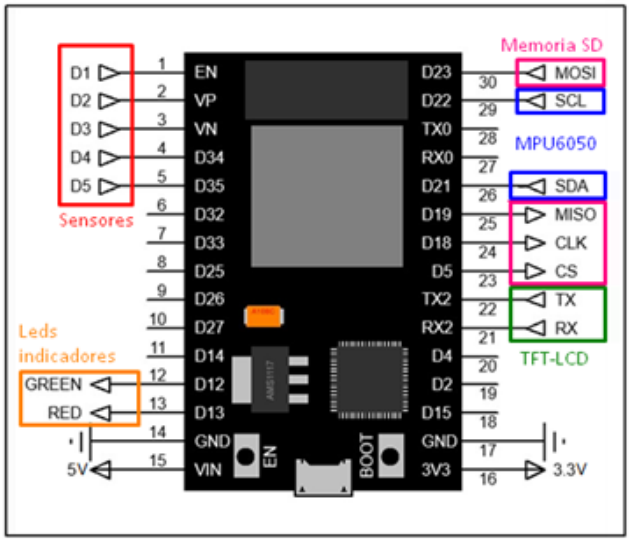


Figura 28: Conexionado del ESP32 del transmisor

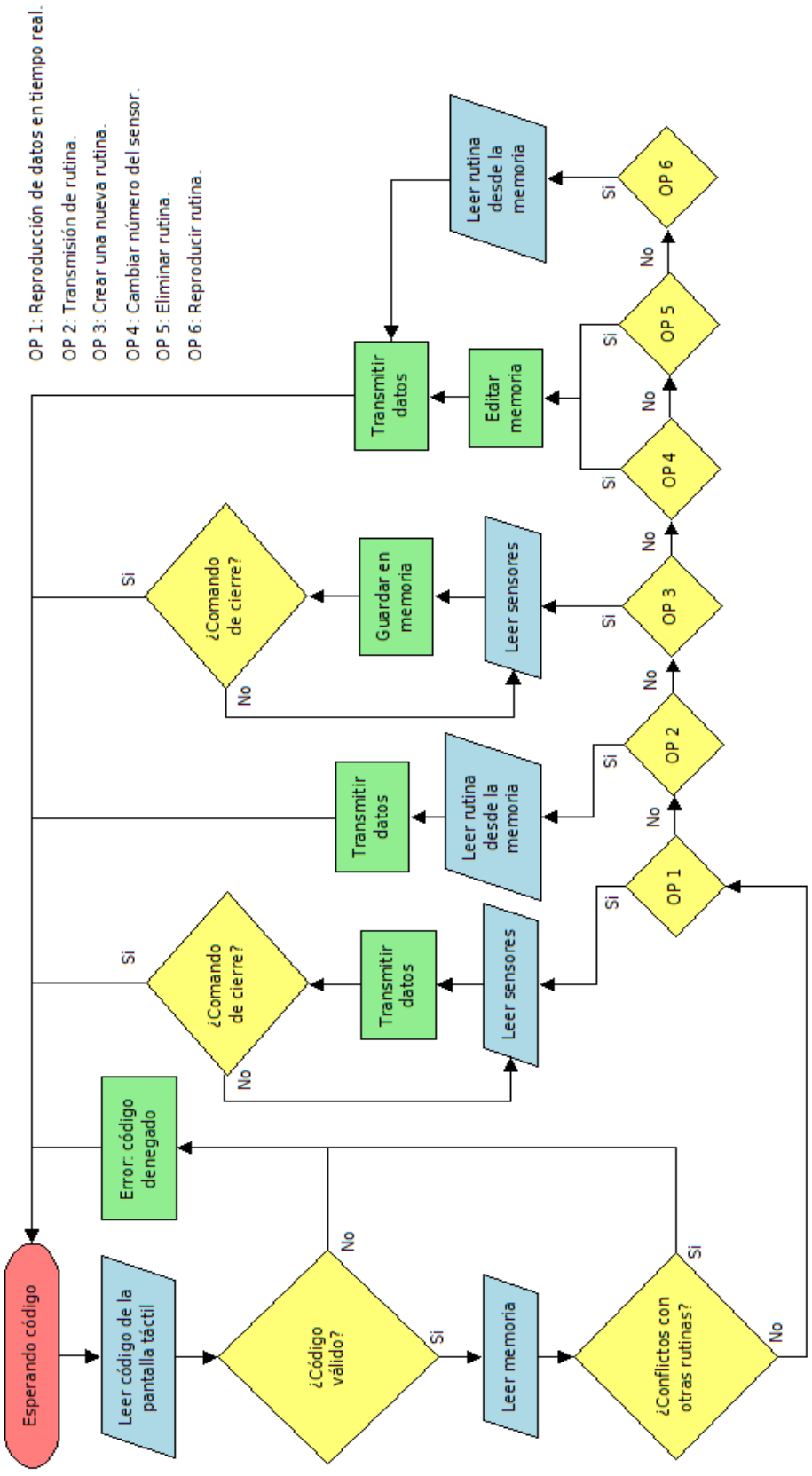


Figura 29: Diagrama de flujo del comportamiento del ESP32 transmisor

El ESP32 espera en todas las iteraciones a que el ATMEGA328P le envíe una orden por el puerto serial, si este no llega en una iteración del programa simplemente la finaliza. Cuando el ESP32 recibe una orden, la primera tarea a ejecutar es comprobar la sintaxis y valores del código, a fin de descartar errores, si el código no es válido y no está dentro de lo esperado, se lo ignora y finaliza la iteración. Si en cambio el código es válido, se evalúa entonces que la orden no entre en conflicto con los datos de la memoria, los casos que generan conflictos son:

- Intentar crear una rutina cuyo número de rutina ya fue creado previamente.
- Intentar crear una rutina con un número de sensor que ya esté en uso.
- Intentar transmitir o reproducir una rutina que no ha sido creada.
- Intentar cambiar el número del sensor a uno que ya esté en uso.
- Intentar cambiar el número del sensor de una rutina que no ha sido creada.
- Intentar borrar una rutina que no existe.

En caso de que el mensaje no sea válido o la orden entre en conflicto, los leds indicadores parpadearán tres veces indicando la que la orden fue negada.

Si la orden se toma en cuenta, se guardan los valores del código (tarea-rutina-sensor). Se lee el número de la tarea y según el valor del mismo, se ejecuta una rutina acorde.

Si la tarea es 1 (transmisión de datos en tiempo real) el ESP32 lee todos los sensores, almacena sus valores en una estructura de datos y la transmite en intervalos de 100[ms]. Esto se repite indefinidamente hasta que el usuario vuelve a tocar la pantalla táctil y envía el código de cierre de la tarea.

Si la tarea es 2 (transmisión de rutina) el ESP32 abre desde la memoria SD el archivo de la rutina indicada en la orden, lee su contenido tomando los valores de a once datos a la vez, los almacena en la estructura de datos y la transmite.

Si la tarea es 3 (crear rutina) lo primero que hace el ESP32 es actualizar el archivo Lista.txt, actualizando el par rutina/sensor de la nueva rutina creada, luego se crea un nuevo archivo en la memoria SD cuyo nombre será Rutina_x.txt, siendo x el número de la rutina indicado en la orden. Para finalizar, se leen los sensores y se guardan los valores dentro

del archivo creado, esto último se repite en intervalos de 100[ms] hasta que el usuario vuelve a tocar la pantalla táctil para enviar el código de cierre de la tarea.

Si la tarea es 4 (cambiar sensor) se actualiza el archivo Lista.txt con los nuevos valores del par rutina/sensor indicado en la orden. Finalmente se cargan en la estructura de datos los valores (tarea-rutina-sensor) y se los transmite para actualizar el archivo Lista.txt de la memoria SD del dispositivo receptor.

Si la tarea es 5 (borrar rutina) se actualiza el archivo Lista.txt cambiando el número del sensor de la rutina indicada por un cero (esto indica que la rutina no existe) y posteriormente borrando el archivo de rutina indicada. Finalmente, al igual que en el caso anterior, se cargan en la estructura de datos los valores (tarea-rutina-sensor) y se los transmite para actualizar el archivo Lista.txt de la memoria SD del dispositivo receptor.

Si la rutina es 6 (reproducir rutina) se lee el archivo de la rutina indicada, se cargan los valores dentro de la estructura de datos y se la envía en intervalos de 100[ms]. Este ciclo continúa hasta que se haya leído completamente el archivo.

Por otro lado, necesitamos analizar el esquema de control del circuito receptor. Se presenta entonces el esquemático y el diagrama de flujo del mismo.

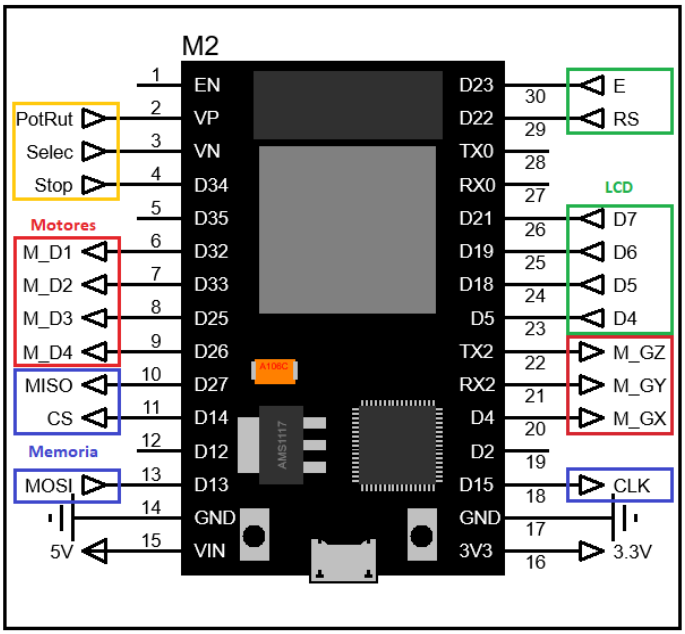


Figura 30: Conexionado del ESP32 receptor

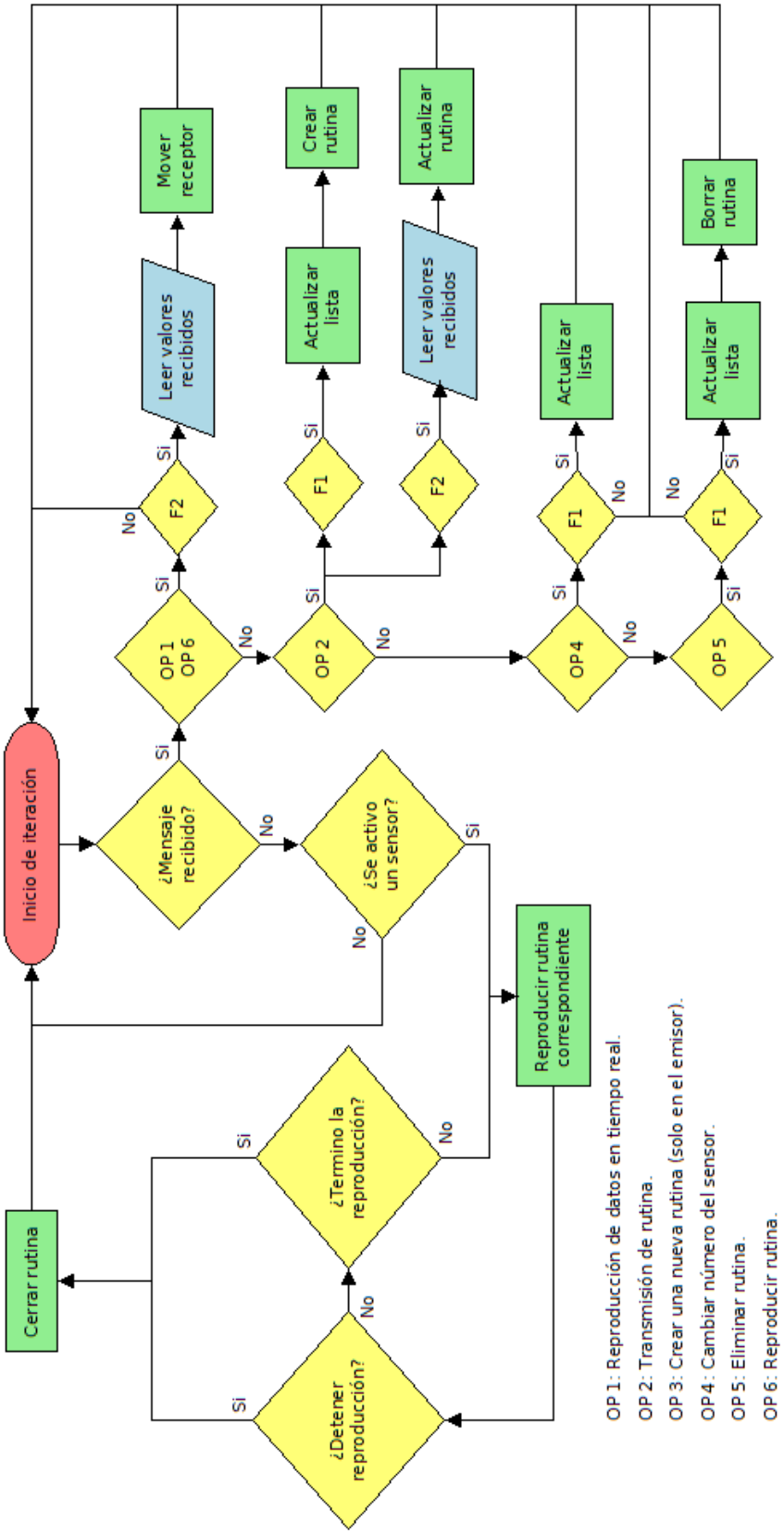


Figura 31: Diagrama de flujo del comportamiento del ESP32 receptor

El dispositivo de control (ESP32) del circuito receptor cumple esencialmente con las tareas de lectura de sensores y de detección de datos transmitidos.

En primer lugar y como se mencionó previamente, el ESP32 debe poder detectar los datos que son transmitidos desde el guante e implementar una rutina en función de la orden recibida.

Si ejecutamos la tarea 1 o 6 que son la transmisión de datos en tiempo real y la reproducción de una rutina determinada, el receptor (en ambos casos) recibe la orden y se configura para recibir y procesar los datos adecuadamente. Posteriormente recibirá paquetes de datos, los cuales pueden provenir de la lectura directa de los sensores del guante (tarea 1) o de los datos guardados en la memoria del guante (tarea 6). En cualquier caso, el proceso es el mismo, se leen los datos recibidos, se ajustan y filtran los valores que se salgan de los parámetros preestablecidos y se lo procesa para transformar el dato numérico en una señal PWM capaz de controlar los motores del brazo robótico. En la fase final se indica que se ha finalizado la transmisión de datos y que el ESP32 puede volver a su rutina normal, esperando una nueva transmisión de datos o que se active un sensor.

La tarea 2 (transmitir rutina) nos indica que el conjunto de datos leídos debe ser guardado en memoria, por lo cual se guarda el conjunto sensor/rutina dentro del archivo Lista.txt y luego se crea un archivo donde se almacenarán los datos de la rutina que se desea crear. En la siguiente etapa los datos recibidos son guardados directamente en el archivo creado. En la fase final se cierra el archivo creado y se retorna nuevamente a la rutina de espera.

En la tarea 4 (cambiar sensor) se edita el archivo Lista.txt, en esta instrucción se cambia el par sensor/rutina según lo que se indique en la pantalla.

La tarea 5 (eliminar rutina) edita el archivo Lista.txt eliminando el par sensor/rutina indicados. Posteriormente se elimina el archivo que contenía la rutina que se quiere borrar.

2.8. Procesamiento de la información (acelerómetros/giroscopios/sensores)

La información con la cual se trabaja consiste en datos obtenidos a partir de medidas realizadas con los sensores de giro y los valores de acelerómetro y giroscopio obtenidos del MPU6050. Cada uno de ellos debe ser procesado de forma diferente al resto y se debe adaptar para que el valor obtenido pueda ser transformado en una señal de PWM capaz de controlar correctamente los servomotores.

Antes de analizar cada caso, es importante conocer las características de la señal PWM que deseamos obtener, la misma debe tener una frecuencia de 50[Hz] (periodo de 20[ms]) y una resolución de 8 bit (varía entre 0 y 255). El duty de trabajo de la señal PWM necesario para mover el servomotor tiene que tener un valor mínimo de 1[ms] y un valor máximo que puede variar entre 2 y 3 [ms]. Con esto en cuenta debemos procesar toda la información de los sensores para obtener sus valores equivalentes para crear la señal PWM, de manera similar a como se haría en un mapeo de datos.

Acelerómetros: Los acelerómetros varían (idealmente) entre un rango de -9.81 a 9.81 y representan la aceleración del dispositivo en sus respectivos ejes. En la realidad, estos valores no son exactos y pueden presentar variaciones debido a factores físicos en la construcción del sensor por lo cual se les debe hacer pruebas para determinar su rango y posteriormente obtener los valores máximo y mínimo dentro de los cuales se desee variar la señal PWM. Estos valores nos ayudarán a acotar el rango de valores permitidos para así descartar posibles malas lecturas y evitar un exceso de movimiento por parte de los servomotores. Para convertir la señal otorgada por el acelerómetro en un valor aplicable en la generación de una señal PWM usamos la expresión:

$$a_{(PWM)} = 13 + k\frac{a_{(max)} - a}{a_{(max)} - a_{(min)}}$$

Donde a es el valor obtenido del acelerómetro y k es una constante con la cual podremos modificar el valor máximo que obtendremos para generar la señal PWM.

$$a = a_{(min)} \rightarrow \quad a_{(PWM)} = 13 + k$$

$$a = a_{(max)} \rightarrow \quad a_{(PWM)} = 13$$

Giroscopios: Los giroscopios miden la velocidad angular que experimenta el MPU6050 en cada uno de sus ejes, por ejemplo, el giroscopio $\boldsymbol{g_x}$ retorna la velocidad angular sobre el eje x (tomando el eje x como eje de giro). Este dato por si solo sirve para determinar la velocidad de movimiento del brazo, pero no para determinar su rotación total.

Para obtener la rotación realizada por el servomotor se debe integrar el dato obtenido del giroscopio. Para esto se guarda en cada iteración los valores de tiempo y velocidad angular. Posteriormente la posición angular se mapea para transformar la rotación total leída en el valor necesario para generar la señal PWM que necesitamos para controlar el servomotor.

$$\theta = k(g_{actual} - g_{anterior})(t_{actual} - t_{anterior})$$

Donde k es una constante con la que configuramos la sensibilidad de respuesta del servomotor y θ posteriormente es mapeado para que varie entre los valores precisados para generar la señal PWM (13 a 26-39).

Sensores de giro: Los sensores de giros se comportan de manera similar a un potenciómetro convencional. El procesamiento es similar al que se les da a los acelerómetros, primero determinamos el rango en el cual varia cada uno de ellos al mover los dedos y con estos valores acotamos los valores válidos para evitar posibles errores de lectura (esto dependerá del diseño de los brazos usados como eje de rotación en el guante). Con todos estos datos obtenidos se aplican las mismas fórmulas usadas en el acelerómetro reemplazando los valores de los acelerómetros por los valores de los sensores de giro.

2.9. Sensores del receptor

Este bloque corresponde a los sensores que serán los encargados de disparar las diferentes rutinas que se hayan guardado previamente en la memoria SD.

Antes de continuar es muy importante aclarar lo siguiente: el tipo de receptor del guante puede variar y el mismo podría ser un auto a control remoto, un dron, un periférico de videojuegos, una simulación de PC o como en nuestro caso un brazo robótico. Cada receptor tendrá sensores y actuadores diferentes (en tipo y cantidad) los cuales brindan información y trabajan con un tipo de señal concreta, por lo que, según sea el tipo de receptor que se desee controlar, se debe construir un circuito diferente y el código del bloque de control (ESP32) debe adaptarse para cada caso concreto.

Sin embargo, el diagrama de bloques presentado inicialmente se mantiene constante en cada caso previsto.

El tipo de receptor implementado no tiene valor comercial, sino que fue pensado como un modelo de demostración para exponer de la manera más sencilla posible las características buscadas en este proyecto.

Aclarado lo anterior, para el caso presentado en este proyecto no se utilizarán sensores para disparar las rutinas guardadas en la memoria SD, en cambio, se implementa una pantalla LCD 2x16 en conjunto con dos pulsadores y un potenciómetro, los cuales le permitirán al usuario seleccionar el sensor deseado y simular su disparo.

El esquema de conexión implementado es el siguiente:

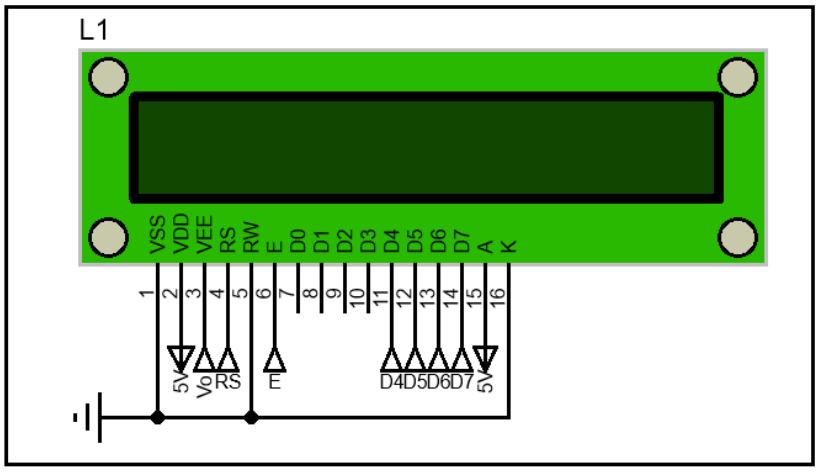


Figura 32: Conexionado de la pantalla LCD receptora

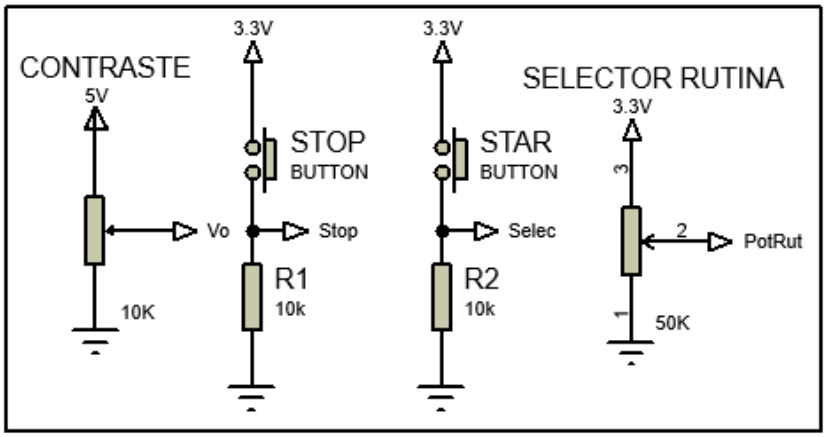
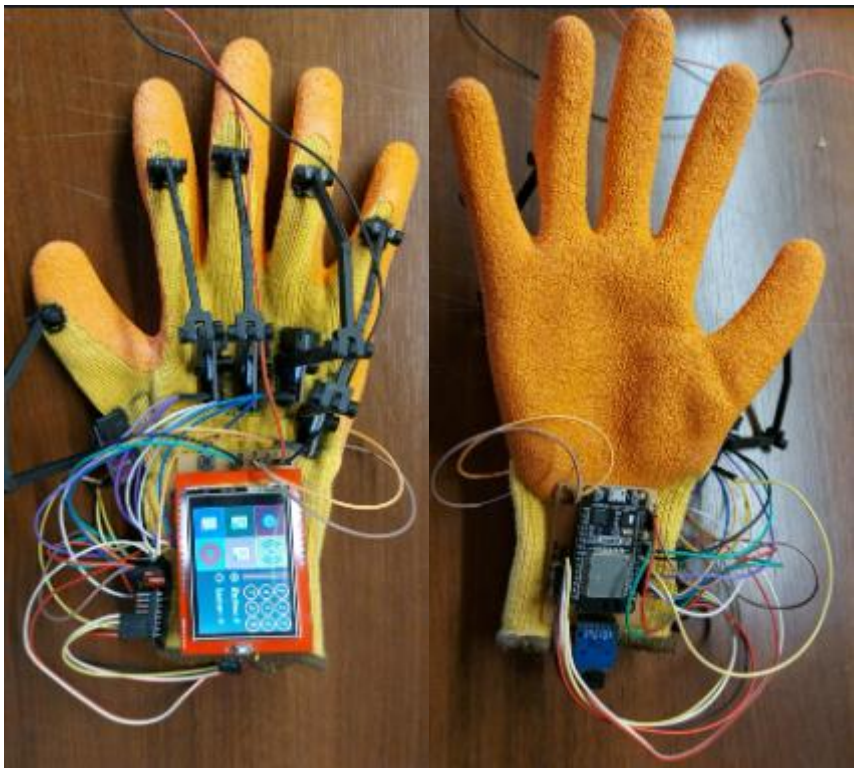


Figura 33: Conexionado de los botones de start, stop, el potenciómetro selector y el trimpot de contraste

El potenciómetro nos permite elegir el número del sensor que deseamos disparar; el pulsador de start inicia la rutina que esté vinculada al sensor seleccionado previamente; el pulsador de stop detiene forzosamente la reproducción de una rutina y la pantalla LCD es el medio por el cual podemos mostrarle la información necesaria al usuario.

2.10. Resultado final



2.11. Actuadores

Los actuadores utilizados en el proyecto son servomotores estándar. Éstos se pueden controlar para girar a posiciones angulares de hasta ± 90 grados desde el reposo. La precisión angular de cada servo afecta un poco la precisión con la que se pueden mover los dedos. Cabe aclarar que en este sistema se han utilizado servomotores relativamente económicos para mantener un bajo costo. El uso de servos de mayor calidad, por supuesto, aumentaría la fuerza y la precisión de todos los movimientos, pero costaría significativamente más.

Los actuadores corresponden a toda la sección del circuito receptor que es conectada a los puertos del ESP32 que entregan la señal PWM.

Lo primero que hay que conocer es el servomotor sobre el cual se desea operar. El mismo es el servomotor **MG996R** cuyas especificaciones de operación se detallan a continuación:

- Corriente de operación: 500mA a 900mA (6V)
- Corriente de parada: 2.5A (6V)
- Voltaje de operación: 4.8V a 7.2V
- Frecuencia de operación de la señal PWM: 50Hz
- Amplitud de la señal PWM: 4.8V a 7.2V

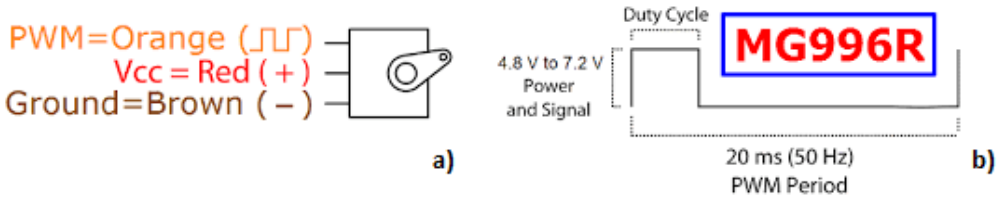


Figura 34: a) Conexionado del servo b) Características de la señal PWM

Figura 35: Imagen del servomotor MG996R

Debido al gran consumo que presentan los 7 servomotores operando al mismo tiempo, se ha optado por alimentar el circuito receptor con una fuente de tensión que sea capaz de proveer los valores especificados de tensión y corriente.

Se presenta la problemática de que la señal PWM otorgada por el ESP32 cuenta con una amplitud de 3.3V por lo cual se hace necesaria la aplicación de un conversor de nivel que sea capaz de amplificar el voltaje al menos hasta los 4.8V. Para esto se implementó un módulo basado en el chip YF08E el cual es un conversor de tensión capaz de amplificar una señal de 3.3V hasta los 5V y cuenta con la cantidad suficiente de puertos como para permitir trabajar con todos los servomotores.

Figura 36: Imagen del conversor de niveles lógicos

Lo último por analizar en este bloque es el brazo robótico que juega el rol de dispositivo receptor. Como el mismo posee una complejidad considerable se lo analizará a detalle en la siguiente sección.

2.12. Introducción

Se presenta un diseño imprimible en 3D para un brazo mecánico, el que será electrónicamente accionado y controlado por los datos que llegan del guante de control, ya sea en tiempo real, o con una rutina pregrabada.

El rápido crecimiento y avance de la industria de la impresión 3D nos permite realizar un prototipo con una calidad aceptable, para realizar los movimientos de manera bastante lineal, es decir, copiando tolerablemente los movimientos del guante.

2.13. Fabricación

Para crear este brazo es necesario tener un sistema mecánico bien diseñado que imite la funcionalidad del brazo humano lo mejor posible. Entre otras cosas el diseño mecánico implica cómo se accionan las articulaciones y los tipos de fuerzas presentes en el sistema.

Este brazo en particular usa el diseño de tendón artificial. Éstos son una forma viable de accionar manos biónicas. Los tendones pueden soportar una alta línea de fuerza, que no se estiran cuando se tensan. Estas líneas se conectan a los dedos y son tensados por motores en el antebrazo. Tirar de los tendones hace que los dedos se abran y cierren.

Los motores eléctricos que accionan estos tendones deben estar completamente alojados dentro del dispositivo, para disminuir el largo innecesario de los tendones, y de esta manera minimizar la posibilidad de problemas en los movimientos.

Evaluamos con el grupo que el diseño 3d de un brazo imprimible con una impresora 3D hogareña nos iba a demandar mucho tiempo, y no hacíamos uso de los conocimientos adquiridos durante la carrera de Ingeniería Electrónica. Por esto, decidimos adquirir un modelo que ofrece la empresa australiana Mahdi Designs, cuya licencia es educativa, y lo adaptamos a nuestras necesidades. En este caso en particular, diseñamos solamente las dos piezas que forman el movimiento rotacional sobre el eje Z (Pieza “Engranaje Codo” y pieza “Base Codo”).

Para imprimir las piezas de este brazo se usó una impresora 3D marca Artillery, modelo SideWinder X1.

2.14. Dedos

Cada dedo consta de tres componentes impresos individuales unidos entre sí con pasadores de polipropileno. En un principio usamos ejes metálicos, pero evaluando el peso del brazo terminado, decidimos cambiar el material de los mismos, para disminuir peso, y de esta manera, también disminuir el consumo de corriente necesario para mover el brazo. El tendón artificial se enrolla alrededor de la punta interior del dedo para crear un punto de bloqueo del tendón. Este tendón pasa por canales dentro del dedo para formar un bucle cerrado. Cuando se tira del tendón, se aplican fuerzas de rotación a todas las articulaciones y el dedo se curva hacia arriba.

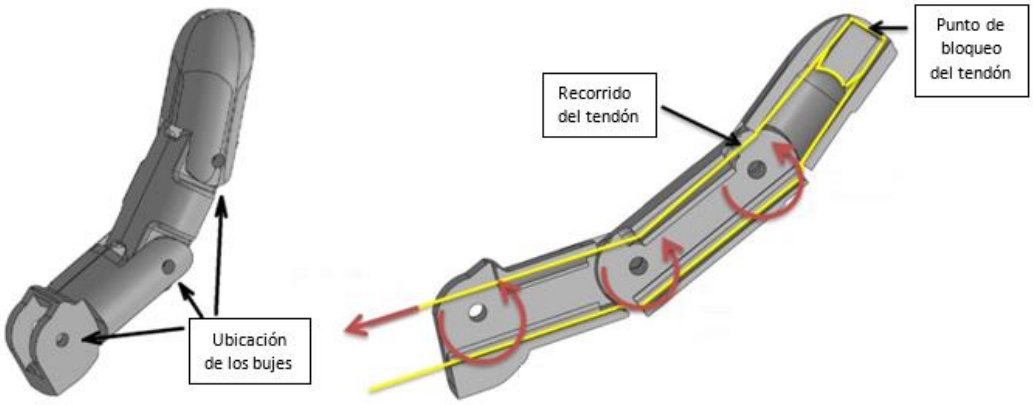


Figura 37: Funcionamiento de la flexión del dedo

El punto de bloqueo del tendón es fundamental para que al tensar el tendón tire de la punta del dedo y hace que todas las articulaciones giren. Si el tendón no se bloqueara, simplemente se deslizaría cuando tensa y el dedo no se movería. Para abrir el dedo desde una posición cerrada, se aplica tensión al otro extremo del tendón.

Se ha utilizado tanza de nylon, ya que ofrece un estiramiento mínimo cuando se tensa. Probamos también usar hilo encerado, pero notamos que, al ser más grueso, ofrecía resistencia por rozamiento en todo su recorrido.

Los tendones de la mano humana biológica funcionan de manera similar.

Sin embargo, hay más tendones biológicos unidos a diferentes huesos, lo que permite un control más preciso de los dedos.

2.15. Palma

Cada dedo se conecta a la palma mediante bujes de polipropileno. La parte inferior de la palma incorpora parte del mecanismo de rotación de la muñeca que se analiza a continuación.

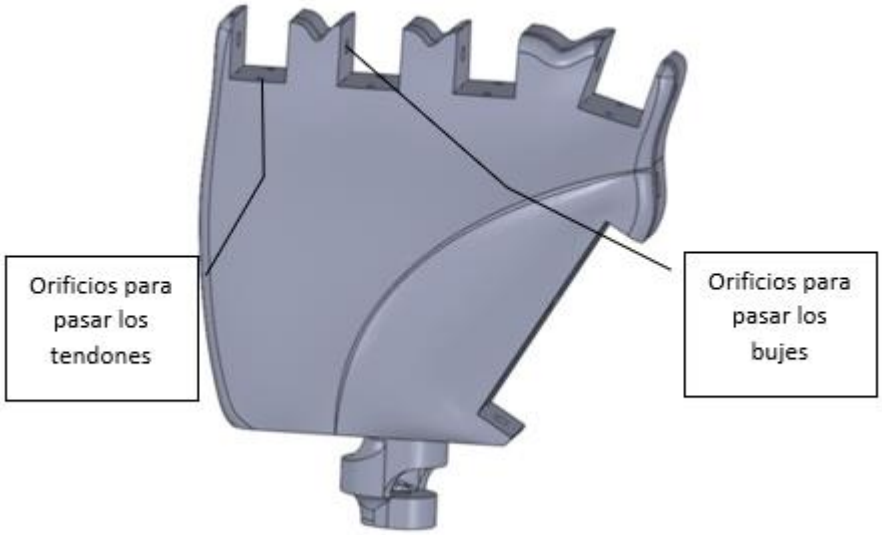


Figura 38: Imagen de la palma

2.16. Muñeca

El movimiento de rotación de la muñeca se logró encajando a presión la sección de la palma directamente sobre el eje del servo en el antebrazo como se muestra a continuación.

Figura 39: Imagen del conexionado del servomotor de la muñeca con la palma

Una abertura a través del punto de pivote de esta articulación permite que los tendones pasen desde los dedos hasta el antebrazo. Esto permite ± 90 grados de rotación sobre la muñeca.

El desafío con este diseño es proporcionar suficiente fuerza. Hubo que dejar una gran abertura alrededor de la base del cilindro para que los tendones pasaran a medida que la muñeca gira 180°.

Esta gran abertura concentra la tensión alrededor de la pequeña sección transversal cerca de la base de la palma.

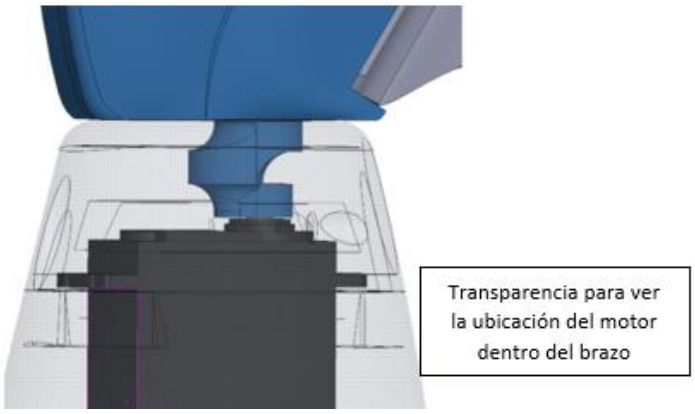


Figura 40: Ubicación del servomotor de la muñeca dentro de la impresión 3D

Los tendones interfieren entre sí a medida que pasan a través de la pequeña abertura de la muñeca hacia los motores en el antebrazo. La rotación de la muñeca de 180° hace que los tendones se tuerzan y se superpongan, lo que es indeseable pero no se puede evitar. Este tipo de diseño no permitiría que la muñeca rotara contínuamente, ya que causaría que los tendones se retuerzan y se vayan enredando.

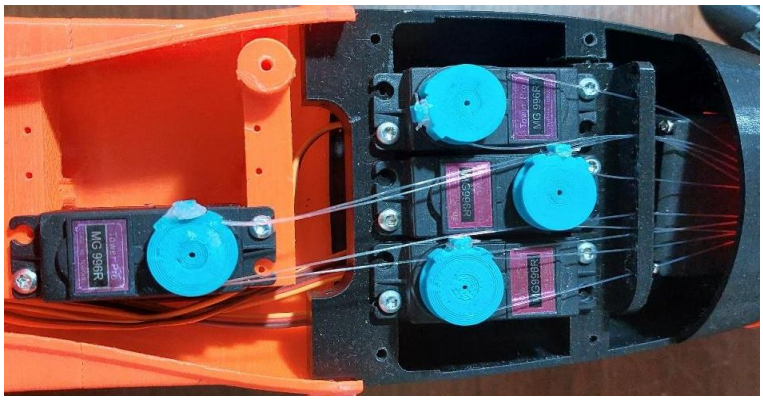


Figura 41: Detalle de los tendones en el antebrazo

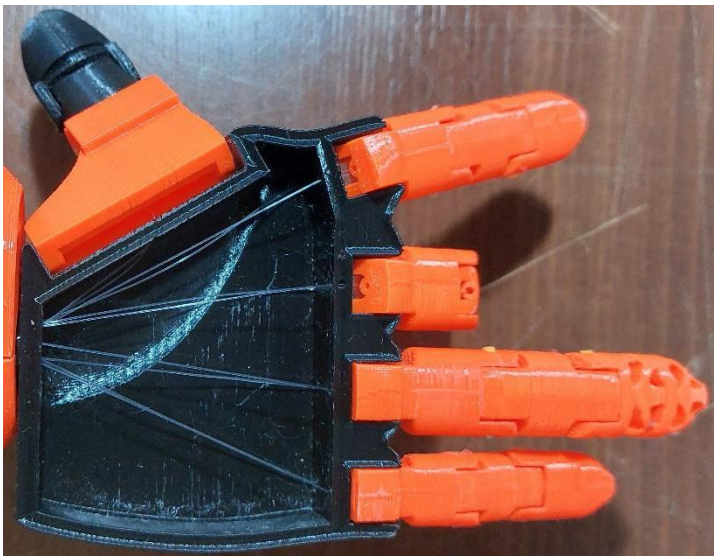


Figura 42: Detalle de los tendones en la mano

2.17. Sistema de manejo

Los tendones se envuelven alrededor de las bobinas en el servo impresas en 3D personalizadas creando un bucle cerrado que se muestra debajo. A medida que el servomotor gira en un sentido, tira del tendón y cierra el dedo. Para que abra el dedo, el motor se gira en la dirección opuesta.

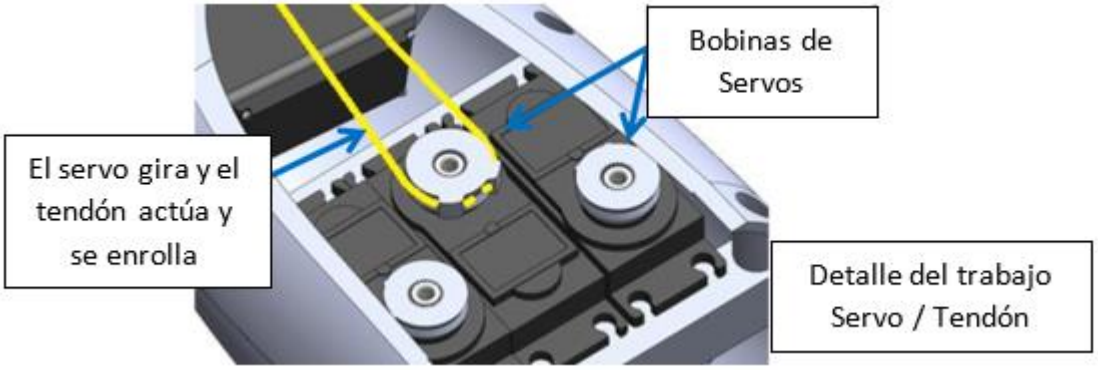


Figura 43: Detalle del trabajo servomotor/tendón

La imagen a continuación muestra el tendón artificial para el dedo índice. Todos los demás tendones se han omitido para mayor claridad. Los dedos pulgar, índice y medio están conectados a servomotores. El espacio interior del brazo es limitado. El anular y el dedo meñique han sido atados al mismo servo, lo que significa que se abren y cierran en tándem.

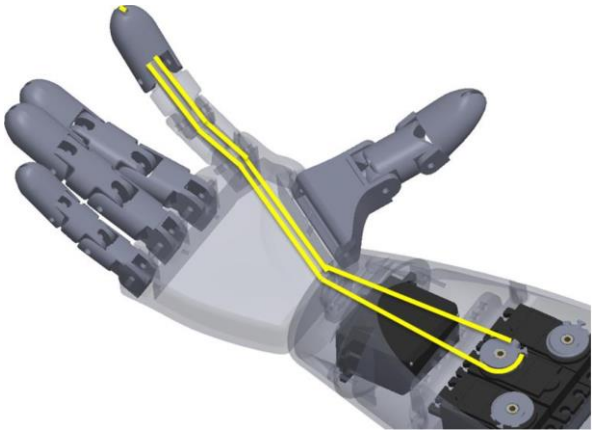


Figura 44: Recorrido del tendón

2.18. Antebrazo

Para la impresión del mismo, tuvo que dividirse en partes separadas que luego se ensamblaron con tornillos.

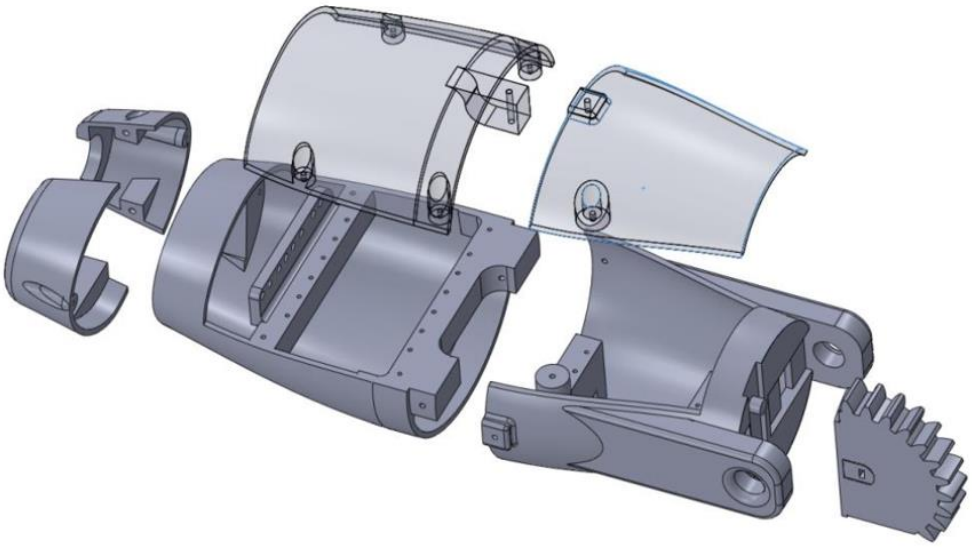


Figura 45: Piezas del brazo

Las dos grandes secciones del antebrazo podrían imprimirse en 3D como una sola pieza sin afectar el montaje del dispositivo. Sin embargo, la impresora 3D no es lo suficientemente grande para imprimir un objeto de este tamaño. Las dos piezas se imprimieron por separado y luego se aseguraron juntas con tornillos.

La sección de engranajes que se ve en la imagen de arriba es parte del mecanismo de rotación del codo. Es crucial que la calidad de este engranaje sea lo más precisa posible en cuanto a las dimensiones. Imprimiendo el engranaje por separado, y luego pegándolo al brazo, obtenemos un engranaje impreso de mayor calidad.

2.19. Codo

El actuador de codo siempre debe mover el peso del antebrazo encima de cualquier carga. El par mínimo requerido para levantar el antebrazo sin carga es de aproximadamente 13,5[kg.cm]. El servo TowerPro que se utiliza proporciona 10[kg.cm] de par. Con el fin de levantar el brazo usando un solo servo se tuvo que implementar un sistema de engranajes. Los engranajes nos permiten generar más par (fuerza de giro) a costa de perder velocidad. El sistema de engranajes diseñado aumenta el par del servo.

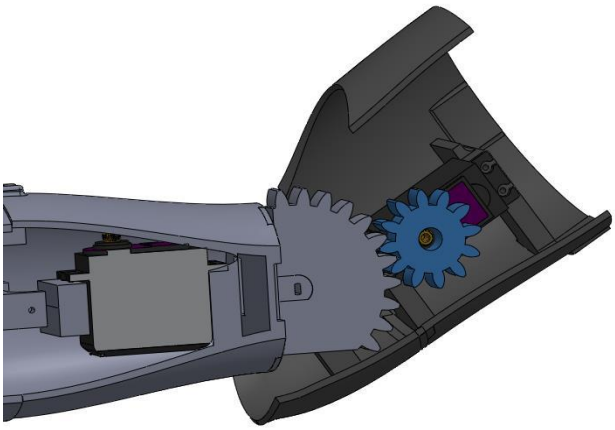


Figura 46: Sistema de engranajes del codo

Para nuestro proyecto, anexamos un par de piezas necesarias para realizar el giro alrededor del eje Z, para ello se diseñaron las siguientes piezas:

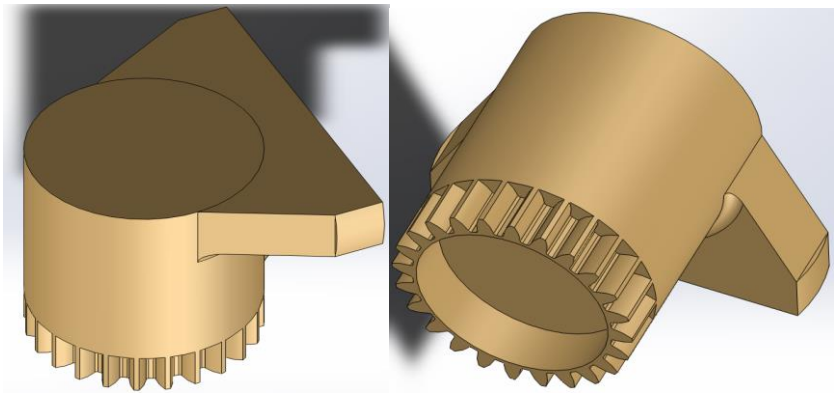


Figura 47: Imagen de la pieza del “engrane codo”

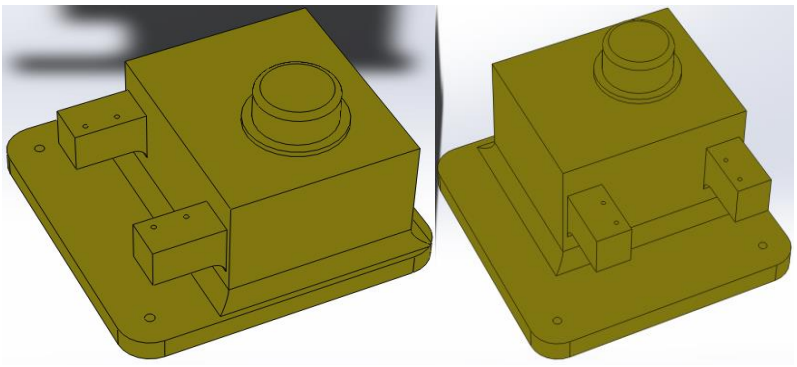


Figura 48: Imagen de la pieza “base codo”

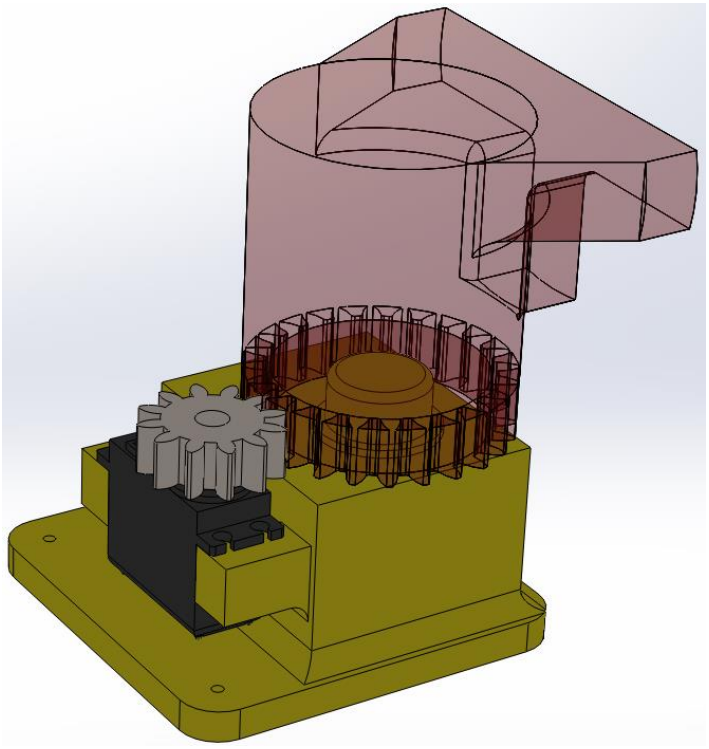


Figura 49: Ensamblaje de ambas piezas, más el engranaje y el servomotor

Dichas piezas fueron diseñadas con el software Solidworks, pudiendo lograr el encastre perfecto con el brazo, y obteniendo una buena calidad de diseño.

Para unir ambas piezas, se usó un rodamiento marca ABR, modelo 6205. Se escogió este modelo principalmente por el diámetro exterior (52[mm]), ya que ocupaba bien toda el área de encastre, y nos aseguramos que cumpla con la capacidad de carga estática y dinámica (en este caso está sobredimensionado). Lo vemos en la siguiente fotografía:

Figura 50: Rodamiento ABR 6205 RS

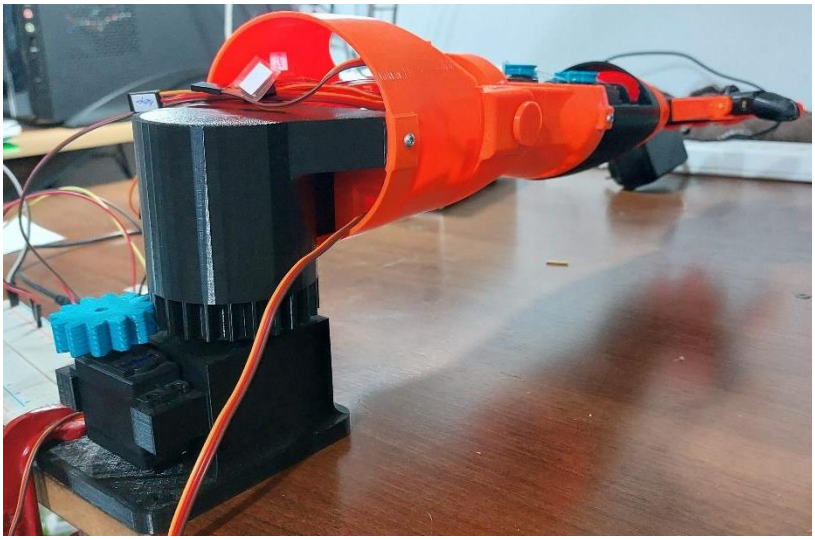


Figura 51: Ensamble de las piezas que conforman el “hombro”

El diseño final consta de 35 componentes individuales impresos en 3D.

Capítulo 3: Resultados

3.1. Guante de control remoto

Las prestaciones buscadas en este proyecto abarcan desde el control remoto de un dispositivo receptor en tiempo real hasta la posibilidad de crear, gestionar y reproducir un conjunto de rutinas vinculadas a determinados sensores del dispositivo receptor.

Sin embargo, además de esto se han tenido en cuenta otros detalles como la comodidad del usuario, la reutilización del código elaborado para extrapolar el funcionamiento a otros tipos de receptores y la sencillez del manejo del dispositivo.

En cuanto a las prestaciones técnicas del guante, el circuito es alimentado con una fuente externa de 5[V] y 700[mA].

El comportamiento de las memorias SD en ambos circuitos es idéntico, la gestión de los pares sensor/rutina que abarcan la creación, cambio de par y eliminación de rutinas, se maneja mediante el archivo “Lista.txt” y los datos de cada rutina concreta son almacenados en un archivo exclusivo para la misma.

La transmisión de la información es llevada a cabo mediante el protocolo ESP-NOW propia de los dispositivos ESP32. Además, se le incorpora al guante un par de leds indicadores que nos brindan información sobre el estado de la transmisión de datos, sobre posibles errores en los módulos del guante y sobre los permisos necesarios para ejecutar determinadas acciones. Los patrones programados en los leds son los siguientes:

- Falla en la lectura de la memoria SD: Parpadeo del led rojo de 1[Hz] de frecuencia con duty del 10%.
- Falla en la lectura del MPU6050: Parpadeo del led rojo de 1[Hz] de frecuencia con duty del 90%.
- Transmisión exitosa: Por cada paquete de datos enviado, parpadea el led verde.
- Transmisión fallida: Por cada paquete de datos enviado, parpadea el led rojo.
- Operación inválida: Ambos ledes parpadean tres veces.
- Operación finalizada: Ambos ledes parpadean alternadamente durante 2 segundos

Por otro lado, la pantalla táctil funciona de manera fluida y suave. Además, la interface presentada en la pantalla se ha simplificado al máximo, permitiendo de este modo al usuario un control práctico e intuitivo del guante.

La estructura impresa en 3D para conectar los sensores de giro con los dedos se diseñó y desarrolló de manera tal de que el usuario no las sienta en lo más mínimo mientras opera con el guante.

Finalmente, el código desarrollado fue pensado para su reutilización con otros dispositivos receptores y se prestó énfasis en omitir por completo el delay en las funciones clave de lectura y operación de datos del programa, logrando en su proceso un funcionamiento fluido, robusto y veloz.

El resultado final, según lo planteado en un inicio, es un guante capaz de leer la flexión de todos los dedos, su posición y movimientos relativos en el espacio. El mismo opera en tiempo real o gestiona las rutinas de la memoria SD según lo designe el usuario en las tareas disponibles en la interface de la pantalla táctil. El guante es cómodo, permite una buena movilidad y puede funcionar con uno o varios dispositivos en simultáneo, el único cambio que se debe realizar es sobre la codificación del mismo ya que se debe especificar el código MAC del ESP32 que jugara el rol de receptor en el otro circuito.

3.2. Modulo receptor

En cuanto al receptor, que incluye solo el circuito de control, se alimenta con una fuente de PC con 5[V] capaz de suministrar un gran amperaje debido al alto consumo de los servomotores.

El dispositivo receptor desarrollado fue modificado especialmente para propósitos de exposición, por lo cual en lugar de usar sensores o pulsadores para ejecutar las rutinas guardadas en la memoria SD, se emplea un menú con una pantalla LCD de 2x16 comandada con el controlador HD44780, un potenciómetro que permite simular el número del sensor que se desea activar y un par de pulsadores los cuales pueden iniciar la rutina vinculada al sensor seleccionado y detener forzosamente dicha rutina.

3.3. Brazo robótico

El brazo robótico presentado, fue impreso en una impresora 3D hogareña. Se asemeja a un brazo humano, en los movimientos de los dedos, la muñeca y el codo. La articulación que correspondería al hombro, es un movimiento rotatorio limitado a un solo eje. El movimiento de las articulaciones está efectuado por servomotores, que enrollan los tendones correspondientes a cada articulación. El control de estos motores está

comandado por el módulo receptor antes descripto. La alimentación de estos servomotores se efectuó con una fuente de 5[VCC].

3.4. Notas especiales

El dispositivo receptor desarrollado fue modificado especialmente para propósitos de exposición y su único valor real, (al menos en cuanto a ejecución de rutinas) es meramente didáctico. Esto puede no suponer ningún problema si lo que se busca es exclusivamente el control remoto del dispositivo receptor, sin embargo, si se desea trabajar con las modalidades que emplean las memorias SD, es necesario que el receptor cuente con algún sensor o pulsador que sea capaz de ejecutar el evento para la reproducción de las rutinas almacenadas en ellas.

Capítulo 4: Análisis de Costos

4.1. Análisis de costos

En esta sección se analizan los costos de producción de las distintas partes que componen el guante y el brazo robótico. Los costos dados en todas las tablas están estimados en dólares.

Las piezas del brazo robot fueron hechas en su totalidad con PLA+ de la marca Grilon, se estima aproximadamente un consumo de 1281 gramos de filamento y tardo uno total de 159 horas con 22 minutos

Tabla 1: Tiempos y costo de la impresión del brazo

Pieza	Gramos	Tiempo (hs)	Tiempo (min)	Cantidad	min totales
01_biceps	170	19	16	1	1156
02_Upper bicep cap	94	10	13	1	613
03_Locking Pins	4		36	2	72
04_Pin Lock	1		12	2	24
05_Small Gear	10	1	29	2	178
06_Index finger	19	3	3	1	183
07_Middle finger	18	3	13	1	193
08_Pinky finger	10	1	50	1	110
09_Ring finger	18	3	10	1	190
10_Thumb finger	27	3	54	1	234
11_Mid Forearm Large	127	13	49	1	829
12_Mid Forearm Small	39	4	54	1	294
13_Upper Forearm Small	18	2	9	1	129
14_wrist clearance A y B	2		12	1	12
15_Wrist Forearm Section A	19	2	11	1	131
16_Wrist Forearm Section B	9	1	16	1	76
17_Palm_Optimised Connector	68	9	42	1	582
18_Palm Finger Cutouts Back	29	2	55	1	175
19_Forearm Gear Split	29	3	10	1	190
20_Upper Forearm Large	187	20	33	1	1233
21_Upper Forearm Small	18	2	8	1	128
base codo	181	21	52	1	1312
engranaje codo	180	22	30	1	1350
22_Horn	4		42	4	168
	1281	**144**	**659**	**30**	**9562**

Teniendo en cuenta el costo estimado de trabajo de un ingeniero Junior ($289819 o U$D956,5), obtenemos un valor de servicio de unos USD 5,98 la hora. A continuación, se

muestran las tablas del costo de los materiales y la fabricación de cada una de las unidades que componen el producto, incluyendo en las mismas, el valor del servicio prestado:

Tabla 2: Costos de fabricación del guante

Materiales	Costo x Unidad	Cantidad	Total
Guante	4,57	1	4,57
Sensor de giro	6,85	5	34,24
ESP32	12,41	1	12,41
Atmega328P	17,71	1	17,71
MPU6050	5	1	5
Pantalla TFT	7,96	1	7,96
PLA	10,19	0,07 kg	0,71
Lector de memoria SD	2,23	1	2,23
Memoria SD 2GB	2,36	1	2,36
Horas de impresión	0,01	3	0,04
		Total	**87,33**

Tabla 3: Costos de fabricación del brazo robot

Material	Costo X Unidad	Modelo	Cantidad	Total
Servomotores	2,95	m90s	5	14,73
Servomotores	5,48	996r	1	5,48
PLA+	20,91		1,281 Kg	26,79
Horas de impresión	0,02		158,366	2,37
			Total	**51,60**

Tabla 4: Costos de fabricación del circuito receptor

Material	Costo X Unidad	Cantidad	Total
Placa	1,59	1	1,59
ESP32	12,41	1	12,41
Jumpers hembra	1,27	1	1,27
Jumpers macho	1,27	1	1,27
Potenciómetro	0,96	1	0,96
Botones	0,14	2	0,29
Trimpot	0,40	1	0,40
Display LCD	15,47	1	15,47
Nivelador lógico	1,59	1	1,59
Lector de memoria SD	2,23	1	2,23
Memoria SD 2GB	2,36	1	2,36
		Total	**39,84**

Al costo obtenido de las tablas anteriores que representan el coste de producción por unidad, se les debe incorporar el costo agregado de las horas invertidas en el diseño y desarrollo del guante.

El costo de producción de una unidad es de U$D178,77 considerando solo la materia prima, a esto se le agrega un costo de producción de 4 horas de trabajo por cada uno de nosotros, resultando en un costo agregado de U$D70,12 para un total de U$D248,89.

Al precio del producto le agregamos el costo del tiempo que invertimos en la investigación, diseño y desarrollo que es aproximadamente de dos meses de trabajo con jornada completa para tres ingenieros junior, agregando un costo de inversión de U$D5806,45.

El costo agregado será amortizado en la venta de las primeras 1000 unidades, aumentando el costo por unidad en U$D5,6 y dando un total de U$D254,5.

4.2. Plan de Venta:

4.2.1. Público objetivo.

Es una herramienta de automatización orientada al área industrial, sin embargo, este prototipo permite darle una gran flexibilidad y con ello la capacidad de poder ampliar su público objetivo y su área de aplicación hacia las áreas del entretenimiento y la educación.

4.2.2. Objetivos y plan de acción.

Ya que este es un nuevo producto en el mercado no posee fuerza de venta, entonces el primer objetivo planteado es el de dar a conocer el producto para obtener nuevos clientes. Para darnos a conocer y conseguir nuevos clientes es mediante el uso de redes sociales con las cuales es posible armar un perfil de nuestra empresa donde podemos colocar la descripción del producto, ingresar nuestro sitio web, ingresar nuestro número de contacto, administrar las interacciones con los clientes, publicar sobre novedades de nuestra empresa, impulsar tus publicaciones y hacer anuncios. Todos los anuncios que se van a mostrar van a tener la dirección de nuestra tienda propia la cual también va a ser un punto de venta donde vamos a atender a los clientes y donde posteriormente vamos a sacar prototipos pequeños para probar incursionar en el área del entretenimiento. El objetivo es que nuestro producto llegue a los dueños de empresas que requieran controlar procesos de automatización mediante la implementación de robots o brazos robóticos. Otro tipo de cliente pueden ser los centros educativos con orientación técnica, en las cuales el producto pueda a estimular el interés de los alumnos por la programación y la electrónica al poder ver físicamente una aplicación del mismo.

Otro objetivo es el de recuperar el costo de las inversiones realizadas en la etapa de diseño. La medida tomada es el de amortizar el costo de inversión y agregarlo al costo de venta de las primeras 1000 unidades.

Posteriormente, debemos mejorar los procesos de diseño y construcción del guante, de manera tal que nos permita reducir los costos de inversión por unidad y de esta forma abaratar costos y ser más competitivos en el mercado.

Lo último por el momento es desarrollar nuevos prototipos que permitan que el producto pueda aplicarse en otros mercados, ya que tiene muchísimo potencial y presenta muchas prestaciones.

Capítulo 5: Discusión y Conclusión.

5.1. Resultados obtenidos.

Las prestaciones generales obtenidas principalmente en el guante logran cumplir satisfactoriamente con los requisitos que se le han impuesto desde un inicio. Cumple con los protocolos mínimos de seguridad y el diseño final resulta cómodo y práctico a comparación de las versiones anteriores desarrolladas.

La codificación, junto a los módulos utilizados, brindan las bases fundamentales para el funcionamiento íntegro del dispositivo. Aun así, no se descartan posibles mejoras en cuando a la selección de los módulos, siempre buscando minimizar el consumo, reducir el tamaño o simplificar el funcionamiento general del guante.

En cuanto al dispositivo receptor que se conecta al brazo robótico, como se mencionó antes, cuenta con los medios necesarios para brindarle al usuario un control total en cuanto al disparo y ejecución de las rutinas almacenadas, no obstante, se vuelve a recalcar, es impráctico para todo propósito excepto como dispositivo de demostración y pruebas, que es lo desarrollado en este proyecto.

5.2. Objetivos propuestos vs objetivos hallados.

En un principio el propósito del proyecto consistía en la elaboración de un sistema de guardado y reproducción de datos que pueda ser aplicado en medios industriales de producción. Los resultados obtenidos a lo largo de su elaboración nos hacen pensar que mediante unos pequeños ajustes, este dispositivo es capaz de cumplir con las tareas para el cual fue diseñado en un principio, sin embargo, la misma naturaleza del proyecto demostró tener el potencial de ser usado en otros ámbitos para los que no fue pensado en un inicio y esto mismo le brinda el potencial de escalarlo y mediante pequeños ajustes incursionar en otras áreas que puedan ser explotadas mediante el uso de un periférico de estas características, principalmente en las áreas de educación y entretenimiento ya que puede ser conectado de manera muy sencilla a una gran variedad de dispositivos receptores.

5.3. Comparar con otros productos existentes a modo debate. Remarcando prestaciones y costo.

En el mercado podemos encontrar algunos modelos de guantes orientados exclusivamente al control en tiempo real de algunos dispositivos, tal es el caso del guante de control de movimiento Keywish Autos Smart o el guante somatosensorial basado en

Arduino que están pensados en el control de robots, drones y otros tipos de juguetes y suelen tener costos que oscilan alrededor de los U$D100. El modelo de Dexta Robotics en cambio está fuertemente orientado a videojuegos de realidad virtual ya que incorpora motores en su guante los cuales permiten dar la sensación de estar tocando un objeto virtual, este nivel de tecnología eleva mucho su complejidad y su costo, lo cual lo hace oscilar alrededor de los U$D800.

A comparación de los demás modelos, el aquí desarrollado es el único que integra las funcionalidades para añadir, guardar, manipular y controlar las rutinas de movimiento, ya que como se marcó al inicio, su orientación fue inicialmente para tareas relacionadas al área de producción industrial.

5.4. Remarcar problemas hallados y soluciones implementadas (breve).

Los problemas principales en el diseño vienen dados por las limitaciones de los módulos disponibles y la programación e intercomunicación entre todos ellos para que todo el conjunto trabaje de forma armónica. Esto requiere la conexión de las memorias y la pantalla LCD mediante los puertos SPI, la conexión de la pantalla táctil con el puerto UART y la conexión del MPU6050 mediante los puertos I2C.

El ESP32 tiene dos puertos de salidas ADC, pero al usar el protocolo de comunicación ESPNOW para conectar los dispositivos emisor y receptor se anula el puerto 1, limitando el número de entradas analógicas disponibles, esto ha requerido algunos cambios en el diseño en cuanto al conexionado y dio como resultado la incorporación de los sensores de giro posicionados en los nudillos de guante.

Afortunadamente con el dispositivo ESP32, tanto en el circuito emisor como receptor, se han podido interconectar todos los módulos requeridos y en ambos casos ocupando casi la totalidad de los puertos disponibles en ambos dispositivos.

En cuanto a la codificación, la complejidad recayó sobre la estructura del mismo. El código debía ser configurado para que puedan trabajar todos los módulos, los cuales en algunos casos requerían usar los mismos puertos. También debía diseñarse de manera tal que sea fluido, fácil de leer, mantener y mejorar. Además, debía responder precisamente y de manera rápida, por lo que instrucciones como “delay” que dejan en pausa todo el programa estaban prohibidas en ciertas situaciones y obligó a mejorar en nivel de codificación de los dispositivos. Finalmente, el código se preparó para operar bajo ciertas

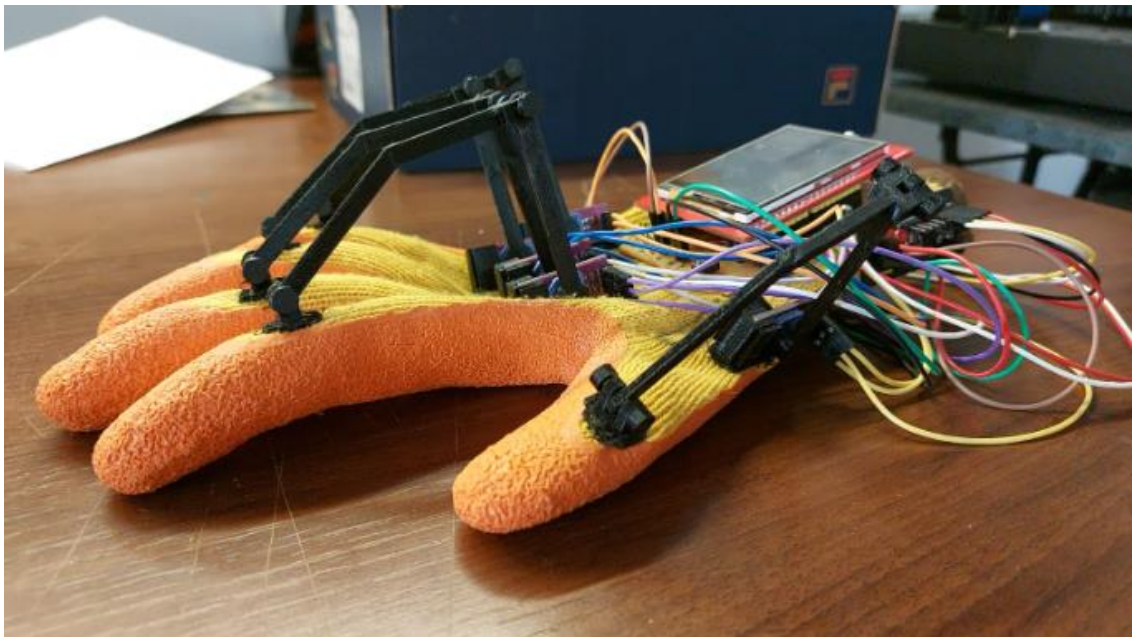
circunstancias de error, generando patrones que ayuden a detectar que tipo de problema puede llegar a estar experimentando el dispositivo bajo algún imprevisto.

Por el lado del diseño físico del guante, el principal inconveniente es la disposición de los módulos de los sensores de giro, ya que, al montarse sobre un guante, cuando realizamos movimientos con los dedos los nudillos se mueven demasiado y esto en algunas ocasiones ha generado atascos y molestias. Lo mismo ocurre con los brazos que conectan el eje de estos sensores con los dedos. Los primeros diseños eran incomodos, proporcionaban un rango de movimiento limitado y los encastres que mantenían todo en su lugar se soltaban o rompían. Tanto los diseños como la disposición de los sensores en el guante, fueron evolucionando de a poco hasta obtener el resultado presentado el cual es cómodo, permite un amplio rango de movimiento sin molestias y no produce atascos.

Cuando construimos el brazo robótico, nos encontramos con varios problemas. Como el mismo fue impreso con una impresora 3D hogareña, varios de los problemas iniciales fueron a causa de una incorrecta calibración de dicha impresora. En la parte interna de las falanges de los dedos, se encuentran unas cavidades por donde pasan los tendones. Por esto, dichas cavidades debían estar muy bien realizadas para no obstaculizar el movimiento. Debido a esto, se tuvieron que reimprimir varias piezas hasta lograr una buena calidad que no obstaculice el movimiento de los tendones.

También en este punto, habíamos seleccionado como tendón, un hilo encerado, que era bien firme, y no se estiraba. Pero al ser de un grosor considerable (1[mm] aprox.) aumentaba la fricción en el movimiento antes descripto. Le dimos una solución a este problema cambiando el hilo encerado por tanza de nylon, que tiene un diámetro mucho menor, y se desliza mucho mejor por las partes del brazo.

Otro punto que se vio afectado por la calidad de la impresión, fue la unión entre falanges, y la unión de las falanges con la palma de la mano. Estas uniones son encastres entre piezas, con un eje que permite la correcta rotación. Como estas uniones no tienen casi separación entre piezas, cualquier desprolijidad en la impresión, hacía que aumentara el rozamiento y se trabaran las piezas, ocasionando fallas en los movimientos. Solucionamos este problema reimprimiendo las piezas que ocasionaban problemas, y también hicimos un trabajo de post procesado, lijando cuidadosamente los sectores de encastre para reducir la fricción. Esto mejoró considerablemente la calidad de los movimientos.



5.5. Posibles mejoras al proyecto.

A lo largo del desarrollo se han planteado muchas formas de mejorar el producto final, sobre todo en lo que respecta al guante ya que es la parte que permanece inalterable, independientemente del tipo de receptor que se le conecte.

La principal mejora que se ha planteado es el diseño de una base impresa en 3D que reemplace por completo el guante y en la cual se puedan acoplar directamente todos los módulos, esto particularmente es un desafío considerable ya que el diseño debe ser capaz de adaptarse a distintos tamaños de manos y en todos ellos se debe sentir cómodo, por lo tanto, el diseño debería poder ajustar su tamaño mediante algún mecanismo móvil que le permita adaptarse a cada usuario. Otra opción es algo más plano que pueda sujetarse con abrojos como propone el modelo del guante somatosensorial basado en Arduino presentado al inicio del informe.

Otra mejora al guante puede ser usar baterías en vez de una fuente de alimentación. Se puede reducir el consumo para lograr que el equipo sea portable.

Analizando las mejoras del brazo robótico, también se puede reducir el consumo. Haciendo uso de motores de mejor eficiencia y utilizando celdas de Li-Po, podría lograrse que no sólo el guante, sino incluso el brazo, puedan ser equipos alimentados con baterías sin ningún inconveniente. Lógicamente esto tendría un costo asociado muy diferente al planteado en este proyecto.

También se puede mejorar la construcción del brazo robótico. Tanto los servomotores empleados, como el diseño y manufactura del mecanismo y cada una de las piezas que lo conforman, son puntos a mejorar. Si bien en este prototipo funcional el diseño adquirido cumplió los objetivos de base, un equipo comercial podría hacer uso de motores con mayores prestaciones y sensores con mayor precisión y vida útil. También podría construirse con piezas mecanizadas, engranajes de acero, etc., que marcarían una gran diferencia en el uso final que podría darse al dispositivo (mayor precisión en los movimientos, más fuerza, más velocidad, más resistencia, posibilidad de trabajar en atmosferas donde haya más temperatura, humedad, etc.).

También es posible mejorar el procesamiento de datos del MPU6050 o reemplazarlo directamente por un dispositivo que pueda realizar mejor dicha tarea, ya que los valores obtenidos a partir del giroscopio del mismo presentan errores de offset en rutinas prolongadas.

Finalmente, si se lograse reducir lo suficiente el consumo de todo el conjunto, podría llegar a integrarse una batería portable para alimentar el guante en lugar del cargador de 5V que se ha implementado. Lamentablemente, en nuestro diseño no ha sido posible debido a una pobre respuesta del ESP32 al no recibir la cantidad adecuada de energía de parte de la batería, demandando así una segunda unidad y, en consecuencia, el área disponible del guante para su colocación.

5.6. Nuevos desarrollos en base a este proyecto.

Este proyecto, el guante concretamente, presenta las bases para desarrollar un gran conjunto de receptores que pueden estar orientados a la industria, a tareas de alto riesgo, a la educación o al entretenimiento.

Si bien inicialmente el proyecto nace intentando dar una alternativa a la gestión de rutinas en tareas industriales, resulta fácilmente extrapolable a tareas educativas o de exposición como es el caso aquí planteado. También se puede usar en sistemas de entretenimiento al permitir el control en tiempo real de dispositivos como autos a control remoto, drones, periféricos para videojuegos (como propone el guante de Dexta Robotics).

En cualquier aplicación concreta, el diseño se puede compactar y mejorar aún más, ya que el diseño de este artículo está preparado para realizar tanto tareas en tiempo real como tareas de gestión y control de rutinas, que se entiende que no necesariamente es aprovechable por todos los receptores nombrados.

Capítulo 6: Literatura Citada.

Links:

Jrowberg (2021). Constantes de gravedad corregidas para el procesamiento FIFO para que coincidan con los valores 2g [Online]. Available: https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050

Jrowberg (2021). Eliminar llamadas extra de transmisión de inicio/fin durante las lecturas [Online]. Available: https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/I2Cdev

Adafruit (2020). Adafruit MPU605 [Online]. Available: https://github.com/adafruit/Adafruit_MPU6050

Randomnerdtutorials. *ESP32 with MPU-6050 Accelerometer, Gyroscope and Temperature Sensor (Arduino)* [Online]. Available: https://randomnerdtutorials.com/esp32-mpu-6050-accelerometer-gyroscope-arduino/

Naylampmechatronics. *Tutorial MPU6050, acelerómetro y giroscopio* [Online]. Available: https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html

Randomnerdtutorials. *Primeros pasos con ESP-NOW (ESP32 con Arduino IDE)* [Online]. Available: https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/

José Guerra Carmenate. *ESP32 Wifi y Bluetooth en un solo chip* [Online]. Available: https://programarfacil.com/esp8266/esp32/

Aprendiendoarduino (2016). *Aprendiendo Arduino* [Online]. Available: https://aprendiendoarduino.wordpress.com/tag/mosi/

Soloelectronicos (2022). *Como gestionar una SD desde un ESP32* [Online]. Available: https://soloelectronicos.com/2022/08/21/como-gestionar-una-sd-desde-un-esp32/

Álvaro Benito Herranz, José Manuel Villadangos Carrizo (2019). *Desarrollo de aplicaciones para IoT con el módulo ESP32* [Online]. Available:

https://ebuah.uah.es/dspace/bitstream/handle/10017/35420/TFG_Benito_Herranz_2019.pdf?sequence=1&isAllowed=y

Randomnerdtutorials. *ESP32: Guía para el Módulo de Tarjeta MicroSD usando Arduino IDE* [Online]. Available: https://randomnerdtutorials.com/esp32-microsd-card-arduino/

Circuitschools (2022). *Interfaz de módulo LCD 16X2 con ESP32 con y sin I2C* [Online]. Available: https://www.circuitschools.com/interfacing-16x2-lcd-module-with-esp32-with-and-without-i2c/

Rafael Lozano (2021). *Esp32 y pantalla lcd i2c* [Online]. Available: https://www.taloselectronics.com/blogs/tutoriales/esp32-y-pantalla-lcd-i2c

Randomnerdtutorials (2019). *Cómo usar LCD I2C con ESP32 en Arduino IDE (compatible con ESP8266)* [Online]. Available: https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/

Naylampmechatronics. *Tutorial pantalla TFT táctil con Arduino* [Online]. Available: https://naylampmechatronics.com/blog/26_tutorial-pantalla-tft-tactil-con-arduino.html

Luisllamas (2016). *Conectar Arduino a una pantalla TFT de 1.4" a 3.2* [Online]. Available: https://www.luisllamas.es/conectar-arduino-a-una-pantalla-tft/

Electronicavm (2015). *TFT LCD Touch 2.4» Shield para Arduino UNO* [Online]. Available: https://electronicavm.wordpress.com/2015/03/05/tft-lcd-touch-2-4-shield-para-arduino-uno/comment-page-2/

Luisllamas (2016). *Determinar la orientación Arduino y el IMU MPU-6050* [Online]. Available: https://www.luisllamas.es/arduino-orientacion-imu-mpu-6050/

Juan Diego Bernal Iñiguez (2014). *Diseño, construcción e implementación de un sistema de captura de movimiento para análisis ergonómico de riesgo laboral de extremidades superiores* [Online]. Available: https://dspace.ups.edu.ec/bitstream/123456789/7508/1/UPS-CT004422.pdf

JUAN MARCOS LUNA JARAMILLO (2019). *Desarrollo de un sistema de adquisición para la evaluación del balance corporal en base a la medida del movimiento del tronco* [Online]. Available: https://tesis.pucp.edu.pe/repositorio/bitstream/handle/20.500.12404/15817/LUNA_JARAMILLO_JUAN_DESARROLLO_SISTEMA_ADQUISICI%C3%93N.pdf?sequence=1

Miguel Angel Mexicano Aguilar, Brenda Ramírez Solís, Giovanni Meza Meza (2016). *Sensores inerciales: acelerómetro MPU6050* [Online]. Available: https://www.academia.edu/24324242/Sensores_inerciales_aceler%C3%B3metro_MPU6050_Investigaci%C3%B3n_4

Hetpro-store. *Acelerometro MPU-6050* [Online]. Available: https://hetpro-store.com/TUTORIALES/acelerometro-mpu-6050-6dof/

ALBUJA SÁNCHEZ BYRON MAURICIO, SÁNCHEZ GÓMEZ ELIZABETH VERÓNICA (2018). *Diseño e implementación en una tarjeta embebida de un algoritmo de control por modos deslizantes (SMC) para los angulos de roll, pitch y yaw de un hexarotor DJI F550* [Online]. Available: https://bibdigital.epn.edu.ec/bitstream/15000/19419/1/CD-8808.pdf

FERNANDO ALBERTO ALVARADO CLAVIJO (2011). *Mano robótica inalámbrica* [Online]. Available: https://dspace.ups.edu.ec/bitstream/123456789/1681/22/UPS-GT000238.pdf

Edison Herrera, Wellington Zambrano, Franklin Silva. *Diseño e implementación de un guante sensorizado para el control teleoperado de un prototipo de brazo robótico para aplicaciones de manejo de materiales peligrosos* [Online]. Available: https://www.academia.edu/21924150/Dise%C3%B1o_e_implementaci%C3%B3n_de_un_guante_sensorizado_para_el_control_teleoperado_de_un_prototipo_de_brazo_rob%C3%B3tico_para_aplicaciones_de_manejo_de_materiales_peligrosos

Empresa Mahdi Designs (Sydney, Australia): https://mdesigns.space/

Imágenes:

Figura 8: https://cdn.shopify.com/s/files/1/1509/1638/products/Front_58520f11-a7a9-47bb-a199-583a1988c316_1024x.jpg?v=1602515037

Figura 8: https://aws1.discourse-cdn.com/arduino/original/4X/4/c/1/4c1b522d85425fb6a2b757c58bf514b334f5a90c.jpeg

Figura 9: https://upload.wikimedia.org/wikipedia/commons/9/9b/ATMEGA328P-PU_%282%29.jpg

Figura 10: https://www.researchgate.net/profile/Mohammed-Therib/publication/312372376/figure/fig10/AS:450882367168521@1484510141868/Atmel-MCU-ATmega328-and-the-Arduino-pin-out.png

Figura 17: https://www.gizmojo.com.ar/system/photos/images/000/001/927/large/08606-03-L.jpg

Figura 18: https://http2.mlstatic.com/D_NQ_NP_992106-MLA28488203594_102018-O.webp

Figura 19: https://naylampmechatronics.com/img/cms/Blog/Tutorial%20MPU6050/MEMs%20acelerometro.jpg

Figura 20: https://naylampmechatronics.com/img/cms/Blog/Tutorial%20MPU6050/Ejes%20MPU6050.jpg

Figura 21: https://naylampmechatronics.com/img/cms/Blog/Tutorial%20MPU6050/angulo%20inclinacion%202D.jpg

Figura 22: https://naylampmechatronics.com/img/cms/Blog/Tutorial%20MPU6050/rotacion%20MPU6050.jpg

Figura 24: https://ae01.alicdn.com/kf/H739fa2c3b7ec431b92be995a7646a3dcD/M-dulo-de-tarjeta-microSD-TF-m-dulo-de-memoria-para-Arduino-ARM-AVR-2-piezas.jpg_Q90.jpg_.webp
-https://es.aliexpress.com/item/32673631024.html

Figura 25: https://articulo.mercadolibre.com.ar/MLA-826424338-modulo-micro-sd-card-5v-con-adaptador-3v3-pic-arduino-_JM

Figura 27: https://descubrearduino.com/wp-content/uploads/2020/06/ESP32-pinout.jpg

Figura 34: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ0WLJszUpChF9ki4I8OTJ9DnZSQR7Q7TSSNw&usqp=CAU

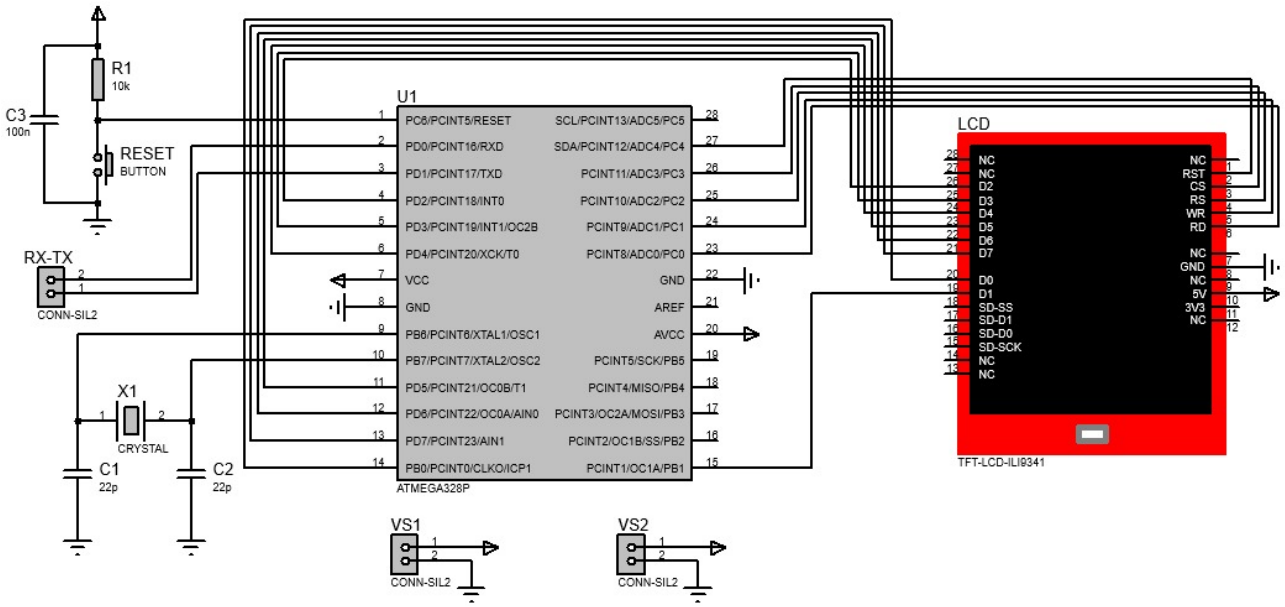
Figura 35: https://www.todomicro.com.ar/motores-y-drivers-motores/234-servomotor-mg996r-digital-10kg-torque.html

Figura 36: https://http2.mlstatic.com/D_NQ_NP_848740-MLM31239273401_062019-O.webp

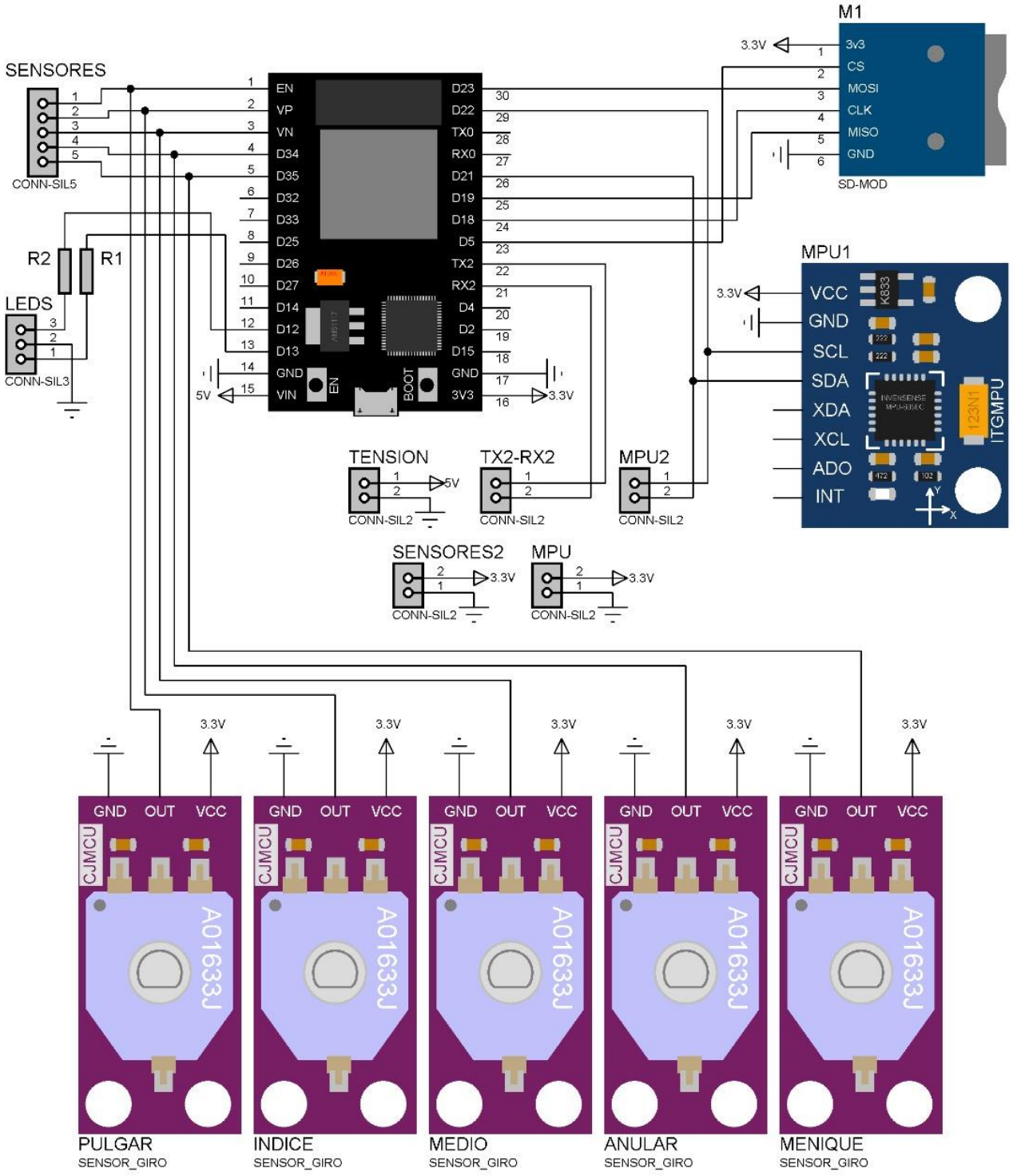
Anexo

Esquemáticos

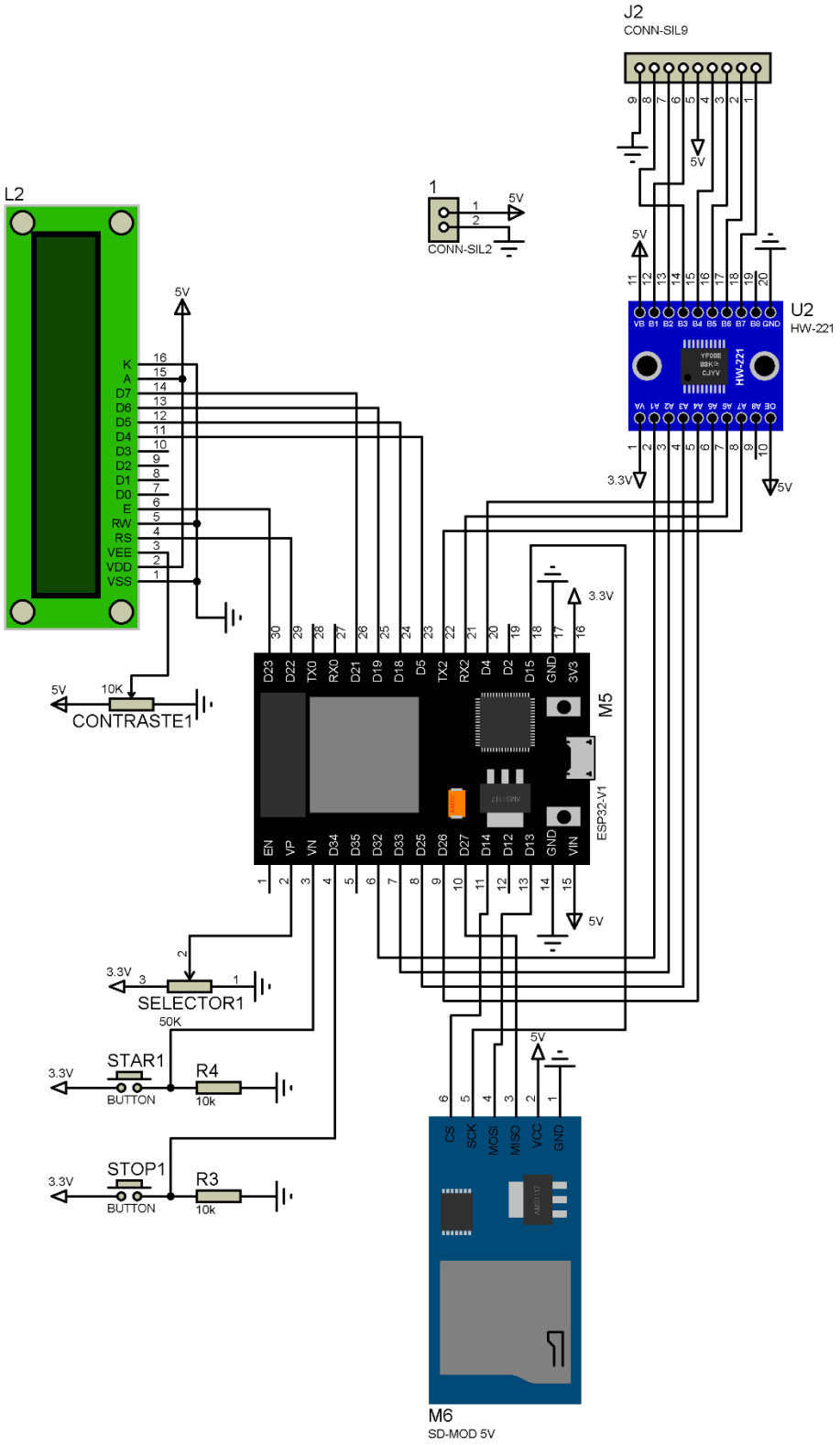
Esquemático interface HMI



Esquemático de control del brazo



Esquemático de control del receptor



PCB

Circuito transmisor

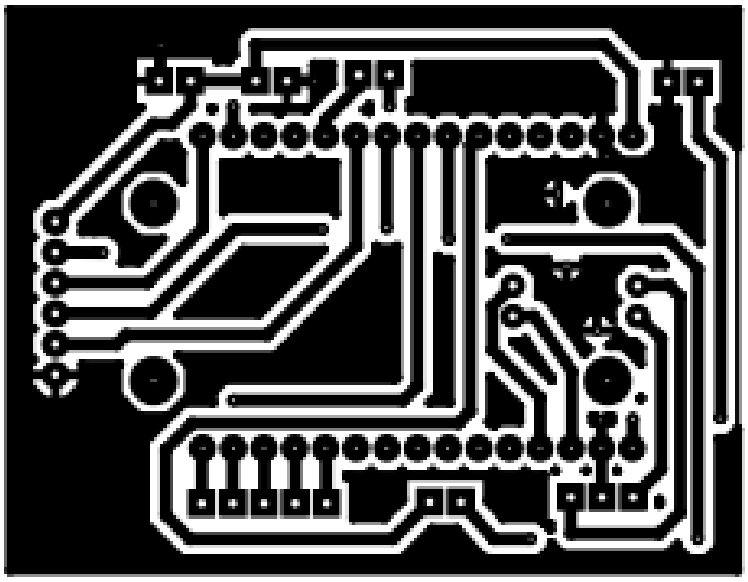


Imagen del circuito transmisor por atrás

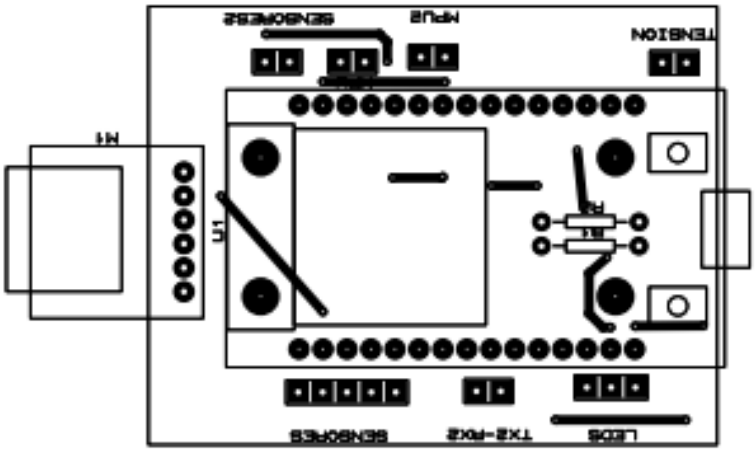


Imagen del circuito transmisor por el frente

Circuito interface HMI

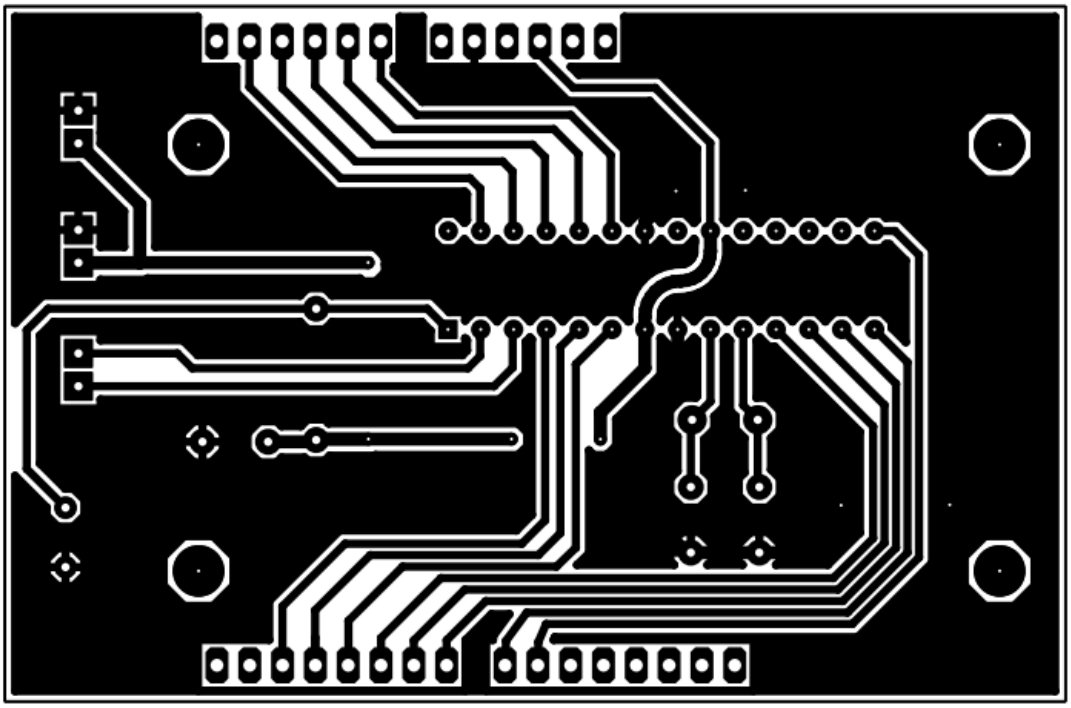


Imagen del circuito de la interface HMI por atrás

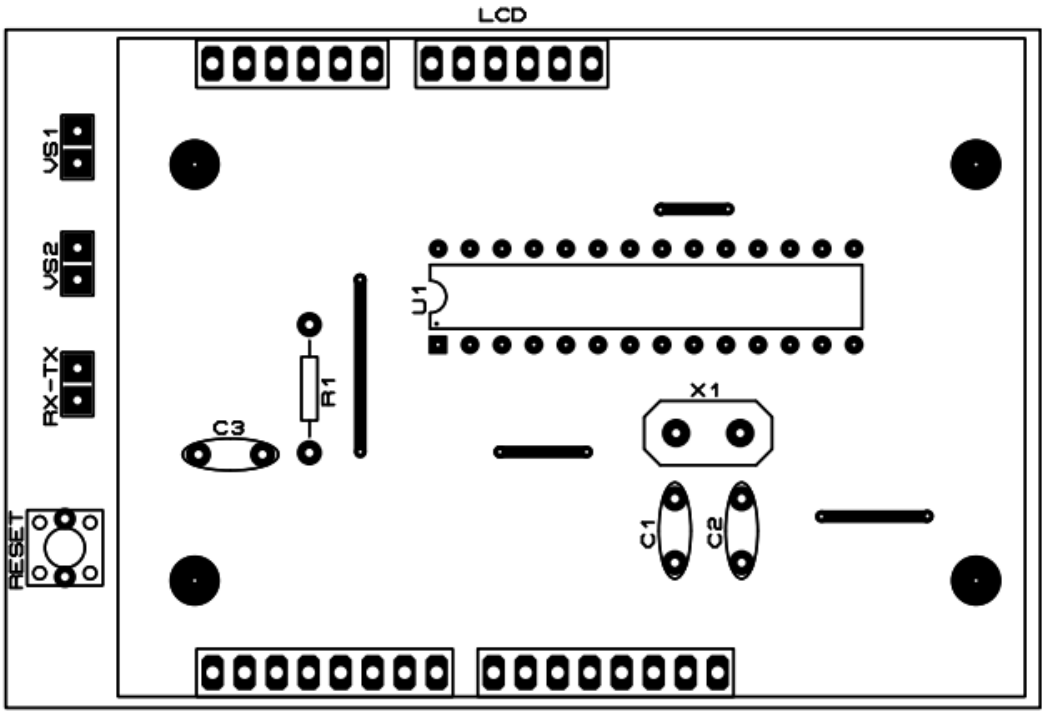


Imagen del circuito de la interface HMI por el frente

Circuito receptor

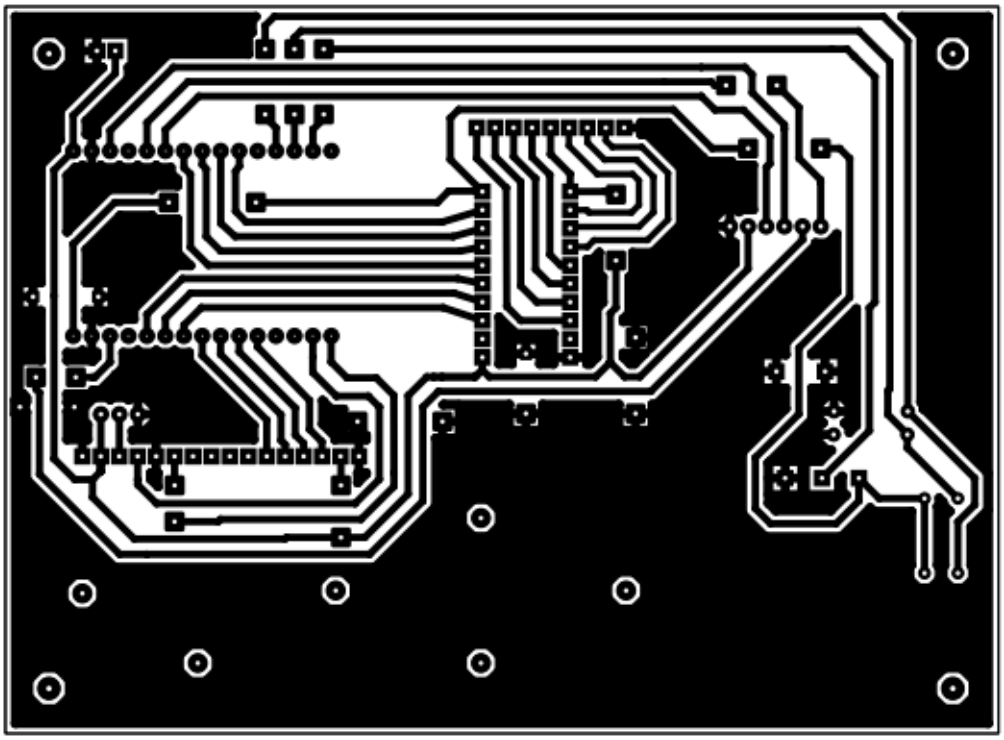


Imagen del circuito receptor por atras

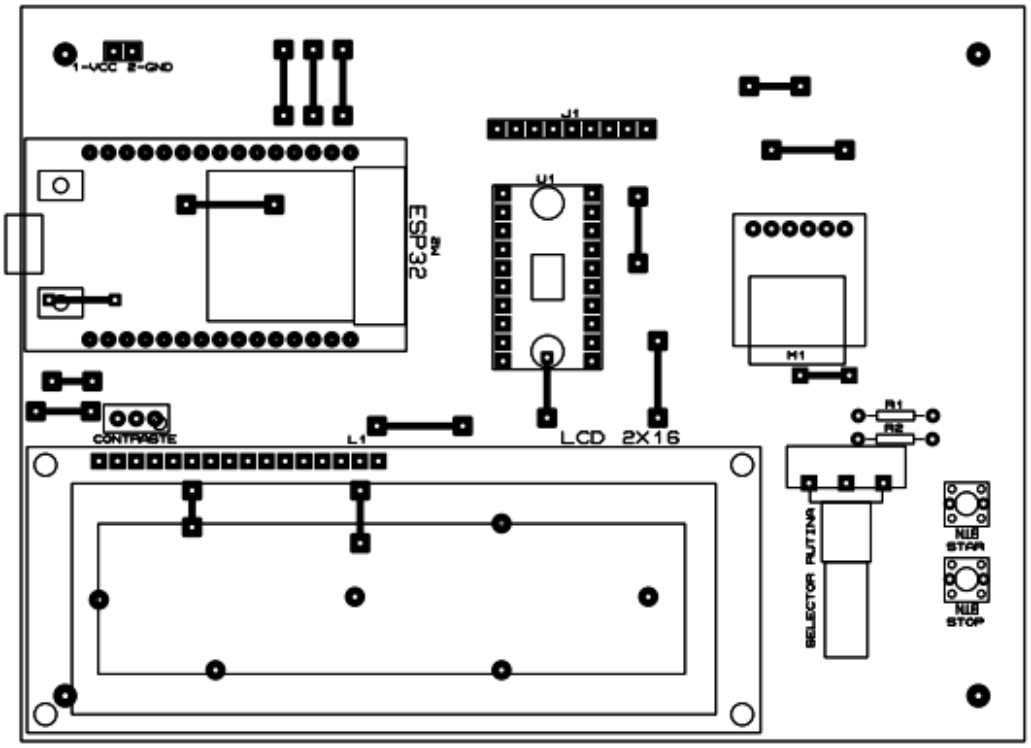


Imagen del circuito receptor por el frente