Universidad Católica de Santa María

Facultad de Ciencias e Ingenierías Físicas y Formales

Escuela Profesional de Ingeniería Electrónica

Diseño e implementación de un controlador numérico aplicado a una mano robótica

Tesis presentada por la Bachiller:

Rivera Delgado, Daniela Alessandra

ORCID: 0000-0003-1282-6427

para optar el Título Profesional de Ingeniero Electrónico con Especialidad en Automatización y Control

Asesor:

Mgtr. Zegarra Gago, Henry Christian

ORCID: 0000-0002-0177-2710

Arequipa - Perú

2026

UCSM-ERP

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

INGENIERIA ELECTRONICA

CON ESPECIALIDAD EN AUTOMATIZACIÓN Y CONTROL

TITULACIÓN CON TESIS

DICTAMEN APROBACIÓN DE BORRADOR

Arequipa, 22 de Diciembre del 2025

Dictamen: 014836-C-EPIE-2025

Visto el borrador del expediente 014836, presentado por:

2015801142 - RIVERA DELGADO DANIELA ALESSANDRA

Titulado:

DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR NUMÉRICO APLICADO A UNA MANO ROBÓTICA

Nuestro dictamen es:

APROBADO

Titulo Profesional/Titulo de Segunda Especialidad/Grado Académico a optar:

INGENIERO ELECTRONICO CON ESPECIALIDAD EN AUTOMATIZACIÓN Y CONTROL

29424254 - MALAGA CHAVEZ CESAR EDUARDO
DICTAMINADOR

29364669 - QUISPE YAUYO JUAN MEDARDO
DICTAMINADOR

29295487 - URRUTIA ESPINOZA MARIO WILLIAM
DICTAMINADOR

Diseño e implementación de un controlador numérico aplicado a una mano robótica

INFORME DE ORIGINALIDAD

9%	9%	1%	3%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	idoc.pub Fuente de Internet	3%
2	Submitted to Universidad Católica de Santa María Trabajo del estudiante	1%
3	www.itc.mx Fuente de Internet	1%
4	sitenordeste.com Fuente de Internet	1%
5	repositorio.utm.mx Fuente de Internet	1%
6	tesis.ucsm.edu.pe Fuente de Internet	1%
7	repositorio.utn.edu.ec Fuente de Internet	1%
8	repositorio.uide.edu.ec Fuente de Internet	1%

DEDICATORIA

En primer lugar, a mis padres Lucy y Víctor Hugo quienes me apoyaron de manera incondicional a lo largo de toda mi etapa universitaria. Son el motor que impulsa cada uno de mis logros.

A mi hermana Angela quien, a pesar de su corta edad, se ha convertido en una fuente constante de inspiración para mí. Su determinación, esfuerzo y capacidad para lograr todo lo que se propone han sido un ejemplo y una motivación permanente durante este camino.

A mi novio Bruno, su compañía constante, palabras de aliento y confianza en mí fueron fundamentales para seguir adelante, incluso en los momentos más difíciles, y lograr cumplir con su ayuda este objetivo.

Su apoyo constante, sus consejos y el ejemplo que me brindaron acompañaron cada esfuerzo y cada decisión tomada. Este logro es también reflejo de ese acompañamiento que dio sentido al recorrido y permitió llegar hasta aquí.

AGRADECIMIENTOS

En primer lugar, agradezco a mis padres por acompañarme en cada paso, por su paciencia, confianza y por el esfuerzo que siempre realizaron para que pudiera alcanzar este objetivo. Este logro es tanto suyo como mío.

A mi novio, gracias por su comprensión, paciencia y motivación durante este proceso; gracias por apoyarme a lograr este objetivo.

Expreso también mi sincero agradecimiento a mi asesor de tesis ing. Henry Zegarra, por su guía, dedicación y conocimientos compartidos, los cuales fueron clave para el adecuado desarrollo y culminación de este trabajo académico.

Finalmente, agradezco a todas aquellas personas que, de una u otra forma, me presionaron y alentaron a no desistir y a cumplir este objetivo, contribuyendo a que hoy este logro sea una realidad.

RESUMEN

En el presente trabajo se desarrolla e implementa un sistema de control numérico para una mano robótica impresa en 3D, capaz de replicar los movimientos de una mano humana mediante el uso de un guante sensorizado. El sistema propuesto emplea un microcontrolador Arduino MEGA 2560 como núcleo de procesamiento, el cual adquiere señales analógicas provenientes de potenciómetros ubicados en cada dedo del guante y las convierte en señales de control para servomotores que accionan la mano robótica.

El software fue diseñado bajo una arquitectura modular basada en una máquina de estados finitos, permitiendo una gestión clara y eficiente de los diferentes modos de operación, entre los que se incluyen el modo espejo, calibración individual por dedo, grabación y reproducción de movimientos, prueba de servomotores y configuración avanzada del sistema. La calibración personalizada permite adaptar el funcionamiento del sistema a distintos usuarios, mejorando la precisión y la usabilidad.

Asimismo, se incorporaron técnicas de suavizado e interpolación para garantizar movimientos más fluidos y naturales, reduciendo el desgaste mecánico de los actuadores. La utilización de la memoria EEPROM del microcontrolador posibilitó el almacenamiento persistente de parámetros de calibración y secuencias de movimiento, permitiendo la operación autónoma del sistema sin necesidad del guante.

Los resultados obtenidos demuestran que el sistema desarrollado es funcional, confiable y escalable, integrando de manera efectiva conceptos de adquisición de señales, procesamiento embebido y control de actuadores. El proyecto constituye una base sólida para futuras aplicaciones en áreas como la robótica asistiva, la rehabilitación y la teleoperación.

Palabras clave: control numérico, guante sensorizado, mano robótica

ABSTRACT

In this work, a numerical control system is developed and implemented for a 3D-printed robotic hand capable of replicating human hand movements through a sensorized glove. The proposed system uses an Arduino MEGA 2560 microcontroller as the core processing unit, responsible for acquiring analog signals from potentiometers located on each finger of the glove and translating them into control signals for servo motors that actuate the robotic hand.

The software was developed using a modular architecture based on a finite state machine, which enables an organized and efficient management of the system's operating modes. These modes include real-time mirror control, individual finger calibration, motion recording and playback, servo motor testing, and advanced configuration of operational parameters. Individual calibration allows the system to adapt to different users, improving precision and overall usability.

Furthermore, smoothing and interpolation algorithms were implemented to achieve fluid and natural movements, reducing abrupt transitions and mechanical stress on the actuators. The use of the microcontroller's EEPROM memory enables persistent storage of calibration data and recorded motion sequences, allowing autonomous operation of the robotic hand without the sensorized glove.

The results obtained demonstrate that the developed system is functional, reliable, and scalable, effectively integrating signal acquisition, embedded processing, and actuator control. This project provides a solid foundation for future applications in areas such as assistive robotics, rehabilitation, and teleoperation.

Key words: numeric control, sensorized glove, robotic arm

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

INTRODUCCION

El desarrollo de sistemas robóticos capaces de interactuar de manera natural con el ser humano ha adquirido una relevancia creciente en los últimos años, especialmente en aplicaciones relacionadas con la asistencia, la rehabilitación y la automatización. En este contexto, las manos robóticas representan uno de los mayores desafíos dentro de la robótica, debido a la complejidad de los movimientos humanos y a la necesidad de lograr un control preciso, coordinado y adaptable a distintos usuarios.

Uno de los principales problemas en el diseño de manos robóticas radica en la dificultad para capturar de forma eficiente los movimientos de la mano humana y reproducirlos de manera fiel en un sistema electromecánico. Muchos desarrollos existentes requieren sistemas de sensado costosos, algoritmos complejos o configuraciones poco intuitivas para el usuario final, lo que limita su accesibilidad y aplicabilidad práctica.

La motivación de esta investigación surge de la necesidad de desarrollar una solución funcional, de bajo costo y fácil implementación, que permita replicar los movimientos de una mano humana de forma intuitiva y confiable.

Para abordar este problema, se diseñó e implementó un sistema de control basado en un microcontrolador, el cual se encarga de adquirir las señales analógicas provenientes de un guante sensorizado, procesarlas y generar señales para el control de servomotores integrados en una mano robótica impresa en 3D. El sistema incorpora una arquitectura de software modular basada en una máquina de estados finitos, que permite gestionar distintos modos de operación. Asimismo, se implementó una interfaz de usuario, facilitando la configuración y el uso del sistema sin requerir dispositivos externos.

Como resultado de este trabajo, se logró desarrollar un prototipo funcional de mano robótica capaz de reproducir de manera proporcional los movimientos de una mano humana, con un desempeño estable y repetible. La calibración individual por dedo permitió adaptar el rango de movimiento del sistema a diferentes usuarios, mejorando la precisión y la comodidad de uso. Además, la implementación del almacenamiento de movimientos demostró la viabilidad de operar el sistema de forma autónoma, sin necesidad del guante sensorizado. En conjunto, los resultados obtenidos validan la factibilidad técnica de la propuesta y demuestran su potencial

aplicación en áreas como la robótica asistiva, la enseñanza y la investigación en sistemas embebidos.

CAPÍTULO I

1. PLANTEAMIENTO METODOLÓGICO

1.1 Descripción del problema

La constante búsqueda de un incremento en la productividad, una calidad uniforme en los productos y la disminución de los costos de mano de obra han impulsado al uso de la automatización en la industria. En los sistemas automatizados, un grupo importante son aquellos aplicados a implementar movimientos repetitivos, este tipo de sistemas cumplen un rol importante en la producción en masa, ya que aportan calidad uniforme en los productos, exactitud extrema en las operaciones y la disminución de los costos de mano de obra. Estas tareas propias del proceso de producción pueden ser realizadas por máquinas-robots diseñadas para llevar a cabo dichas tareas, reemplazando al ser humano o contribuyendo en las tareas que estos desarrollan. Estas máquinas pueden ser diseñadas específicamente para ciertas tareas o tener un diseño general y ser programadas dependiendo de la tarea que se quiera realizar, esto corresponde fundamentalmente al área de la robótica y los sistemas CAD/CAM (diseño asistido por computadora / manufactura asistida por computadora). En la industria existen diversos tipos de robots, la elección dependerá de las necesidades y especificaciones que se tengan.

El área de la robótica industrial está en constante desarrollo, por lo que muy seguido se observan nuevas máquinas en el mercado, con mejoras mecánicas y el aporte de la inteligencia artificial. La variedad de aplicaciones que poseen estos manipuladores es muy diversa. Este es un campo muy amplio en el que convergen distintas áreas como la cinemática, la dinámica, el control, lenguajes de programación, sensórica, electrónica de potencia entre otros; por lo que se le considera interdisciplinaria (Del Valle,2018)..

El uso de la robótica en la industria no solo busca optimizar sino también cuidar la salud del ser humano, al reducir los riesgos a los que se exponen al realizar ciertas tareas, también se reduce la aparición de lesiones a largo plazo al disminuir la carga laboral de las personas; por ejemplo, al reemplazar a una persona que cargaba paquetes muy pesados por una máquina.

El desarrollo de sistemas robóticos, a manera de prótesis o mecanismos autónomos, requiere de manera paralela la aplicación de técnicas de control apropiadas, que garanticen el trabajo eficiente; dentro de estas técnicas tenemos el control numérico, eficiente al momento de implementar tareas repetitivas con mecanismos de alta precisión (Hossian et al,2020).

Por las razones expuestas el presente trabajo de investigación propone el diseño e implementación de un controlador numérico para operar un sistema robótico, que pueda realizar tareas autónomas, con la finalidad de reemplazar tareas específicas realizadas por el hombre.

1.2 Solución planteada

Diseñar e implementar un controlador basado en técnicas de control numérico, aplicado a un sistema robótico, que pueda realizar tareas autónomas con la finalidad de reemplazar o ayudar al ser humano en tareas específicas, peligrosas y repetitivas elegidas por el usuario.

1.3 Alcances de la investigación

El presente proyecto consiste en diseñar e implementar un controlador numérico, para el manejo de un prototipo del sistema robótico de al menos cuatro grados de libertad, compuesto por un brazo robótico y un elemento terminal a modo de mano, el cual realizará tareas específicas susceptibles de ser modeladas numéricamente.

1.4 Objetivos

1.4.1 Objetivo principal

Diseñar un controlador para manejar un sistema robótico de al menos cuatro grados de libertad utilizando técnicas de control numérico.

1.4.2 Objetivos específicos

- Modelar numéricamente los movimientos a ser implementados.
- Diseñar los algoritmos para implementar la técnica de control y el manejo del sistema, a partir de la selección del microcontrolador más adecuado.
- Diseñar e imprimir las piezas en 3D para implementar el sistema robótico basándose en diseño asistido por computadora.
- Seleccionar los actuadores y sensores más adecuados para el proyecto y diseñar los sistemas de acondicionamiento respectivos.
- Ensamblar los distintos componentes del sistema robótico para que este pueda responder a las especificaciones iniciales de diseño elaborado.

1.5 Variables

1.5.1 Variable independiente

Sistema robótico, definido a través de sus indicadores:

- grados de libertad
- funcionalidad dinámica (patrón de movimiento)
- área de trabajo

1.5.2 Variable dependiente

Control numérico, definido como el conjunto de herramientas, métodos, algoritmos y sistemas, usado en procesos de fabricación y uso de naturaleza repetitiva, se manejan los siguientes indicadores:

- algoritmo de control
- dimensionalidad (2D, 3D)

1.6 Estado del arte

El control numérico es el término usado para describir aquellas máquinas que son controladas por una serie de instrucciones formadas por números y letras del alfabeto. En la actualidad este tipo de control lo vemos aplicado en diferentes máquinas de manufactura, con más frecuencia en tornos CNC (Control Numérico Computarizado) y fresadoras CNC. Las instrucciones en estas son valores numéricos conocidas como código G o simplemente código numérico, el cual indica las operaciones con las que deben operar los elementos finales de control de una máquina CNC. Por ejemplo, un torno CNC es un torno operado mediante control numérico por computadora el cual mecaniza piezas de revolución, mientras que una fresadora CNC se utiliza en mecanizados por arranque de viruta mediante el movimiento de una fresa, la cual es una herramienta rotativa de varios filos de corte (Úbeda et al, 2011). El control numérico también es aplicado en máquinas de impresión 3D, las cuales están alcanzando mayor popularidad hoy en día.

En el caso de sistemas robóticos, actualmente existen seis empresas principales que se dedican a su fabricación para la industria. La primera es Kawasaki Robotics, esta empresa tiene más de 50 años de experiencia y cuenta con más de 210000 robots enviados a todo el mundo con una gran variedad de aplicaciones en diversas industrias. Cuenta con variedad de modelos dependiendo de las cargas que se vayan a soportar o de la aplicación que se le vaya a dar (Kawasaki Robotics, 2023). Los robots de la serie R son de los más conocidos ya que al ser de uso general establecen un punto de referencia para otros robots industriales de pequeño a mediano trabajo. Poseen un diseño compacto, veloces y con un rango de trabajo que los hacen ideales para una amplia gama de aplicaciones en diversas industrias (Small-Medium Payload Robots, 2023).

La segunda empresa de desarrollo de robótica industrial es FANUC, que cuenta con más de cien modelos divididos en nueve series de robots que abarcan varias aplicaciones e industrias. De fáciles utilización e integración, con una carga útil de hasta 2.3 toneladas y alcance máximo de 4.7 metros (serie M-2000), debido a los sistemas inteligentes de gestión de energía, son capaces de proporcionar un mejor rendimiento con la mínima energía posible (FANUC,2023). Debido a sus características son usados en la industria automotriz en sistemas de elevación.

Otra experiencia de desarrollo de robots industriales es KUKA, que desarrolla numerosas variantes de robots industriales con diferentes capacidades de carga y alcances. Desarrollan desde la unidad lineal hasta el efector final, el software adecuado y unidades de control innovadores (KUKA,2023). Las aplicaciones son áreas higiénicamente sensibles debido a su cuerpo de acero inoxidable (serie DELTA), robots con un diseño ultra compacto sin contornos de interferencia y óptimo rendimiento para cualquier posición de montaje y espacios reducidos (modelo KR 4 AGILUS) hasta llegar a modelos de manejo de cargas de hasta 1.3 toneladas con distancias de hasta 6.5 metros y alta precisión (modelo KR 1000 TITAN de 6 ejes).

Aporte importante es la de la empresa Yaskawa Electric que trabaja en el desarrollo de la robótica, siendo una de las más importantes ya que tienen más de 500000 robots trabajando en todo el mundo, en aplicaciones en soldadura, paletización, pintura y manipulación (Yaskawa,2023). Así, por ejemplo, la serie más usadas es la GP, que consta de una serie de robots industriales polivalentes de 6 ejes, manejando carga útil de entre 4 y 600 Kg, con diferentes alcances, con altas tasas de protección IP, grasas y acabados de pintura para fines

específicos, por lo que son muy usados en automatización, ensamblaje, manejo de materiales, cuidado de máquinas, remoción de material o inspección de calidad.

Podemos citar también a Universal Robots, marca de Cobots. Encargada de la fabricación de bazos robóticos colaborativos de 6 ejes, flexibles y de fácil uso; con vents que superan las 50000 unidades. El producto de mayor uso es el Cobot UR10e con un 25% más de capacidad de carga útil, mejor rendimiento y productividad (Universal Robots,2023).

La última empresa que marca el desarrollo de la robótica es ABB (Asea Brown Boveri); líder en robótica, automatización de máquinas y servicios digitales, ofreciendo soluciones innovadoras. Así por ejemplo el modelo IRB 1300, robot industrial de 6 ejes con versiones IP67, Foundry Plus 2 y sala limpia ISO 4, por lo que es usado en entornos de producción hostiles y libres de contaminación, donde permite aumentar la productividad, mejorar la calidad del producto y reducir los tiempos de ciclo en una variedad de industrias (ABB,2023).

1.7 Antecedentes

Luego de la revisión del estado del arte, se han encontrado las siguientes investigaciones nacionales e internacionales

1.7.1 Antecedentes internacionales

Título: Prototipo de robot cartesiano utilizando Control Numérico Computarizado para la manipulación de botellas P.E.T. en el proceso de paletizado.

Autores: Roberto Xavier Paccha Medina y Josué Vicente Triviño Sánchez

Tipo: Tesis de grado (2023)

Resumen: En esta tesis los autores presentan el diseño y fabricación de un robot cartesiano dispuesto sobre una estructura metálica cuadrada, este robot tiene grados de libertad con respecto a los tres ejes (X,Y,Z), la estructura robótica tiene la funcionalidad del paletizado de botellas PET de 330ml, la finalidad es mejorar el nivel de productividad de unidades de 16 botellas en una caja de madera, la solución propuesta es de bajo costo, para garantizar su rentabilidad y evitar los errores propios de los operarios humanos, asimismo se busca garantizar la calidad del producto. El robot propuesto entonces puede realizar tareas repetitivas sin parar,

el control empleado es el control numérico por computador y un PLC (controlador lógico programable) como controlador (Paccha&Triviño,2023)

Título: Diseño y desarrollo de un prototipo de robot cuadrúpedo didáctico

Autores: César Alejandro De La Paz Arteaga, Francisco Javier Díaz Ruiz, Manuel Antonio Arenas Méndez.

Tipo: Artículo (2021)

Resumen: El artículo presenta un prototipo de robot cuadrúpedo con fines didácticos, la electrónica de control se implementa sobre una tarjeta de desarrollo Teensy 4.0 con el microcontrolador NXP-MIMXRT1062DVL6A, la comunicación se hace usando tecnología bluetooth, este módulo es de naturaleza autónoma. La dinámica del robot se implementa sobre ocho servomotores MG90s Tower Pro. El diseño del sistema embebido se hizo usando el software CadSoft EAGLE y técnicas de diseño asistido por computadora SolidWorks y ha sido elaborada mediante maquinado por control numérico por computadora. El algoritmo de control del robot se implementa sobre la plataforma Arduino. La fabricación de las piezas se hizo por impresión 3D en material PLA. El robot además puede ser controlado por una aplicación móvil desarrollada en el entorno MIT App Inventor 2 (De a Paz et al,2021)

Título: Diseño e implementación de un mecanismo basado en el control numérico por computadora para la automatización de una máquina duplicadora de llave

Autor: Jhoan Fernando Patiño Carrillo, Jonathan Chitiva Muñoz

Tipo: Tesis de grado, 2022

Resumen: En esta investigación se muestra el diseño e implementación de la automatización de una máquina duplicadora de llaves a través del control numérico por computadora para el manejo de un robot manipulador, con la finalidad de copiar y generar patrones aleatorios para el cambio de patrones de llaves. En el proceso se modelaron los diferentes movimientos de corte para poder ser reproducidos. La fabricación de la máquina se hizo teniendo en cuenta la confiabilidad, garantía y posterior mantenimiento de la misma. Se

programaron los diferentes algoritmos de control para las funciones de corte (Patiño&Chitiva,2022).

Título: Diseño, construcción y programación de una máquina de control numérico aplicada al prototipado rápido de modelado por deposición fundida de material para el laboratorio de mecatrónica de la Universidad Internacional del Ecuador

Autor: Galo Xavier Maldonado Toro

Tipo: Tesis de grado, 2012

Resumen: En este trabajo de grado se muestra el diseño y proceso de construcción de una máquina de prototipado rápido (proceso de fabricación de libre forma) usada para el modelado por deposición fundida de material, mediante la integración y control de sistemas mecatrónicos, para la elaboración de prototipos plásticos a pequeña escala en el Laboratorio de Mecatrónica de la Universidad Internacional del Ecuador, utilizando el control numérico como técnica de control base (Maldonado,2012).

Título: Control numérico en una máquina para rehabilitación de tobillos

Autor: José Santana Camilo, Andrés Blanco Ortega, Edgar Antúnez Leyva, Andrea Magadán Salazar y Fabio Gómez Becerra.

Tipo: Artículo, 2017

Resumen: En este artículo los autores proponen el uso del control numérico para el control de una máquina de rehabilitación de tobillos cuya estructura está basada en una mesa en torno a un sistema XY. El control numérico propuesto aquí es el utilizado fundamentalmente en máquinas herramientas para proporcionar movimientos precisos en el maquinado de piezas, es decir cuyo funcionamiento está basado en patrones. Esta investigación muestra tanto la estructura del sistema robótico es decir el diseño del mismo, así como la propuesta del control numérico que se encargue de establecer los movimientos suaves y complejos necesarios en la rehabilitación específica de tobillos (Santana et al,2017).

1.7.2 Antecedentes nacionales

Título: Diseño de un sistema de control embebido para prótesis transradial funcional

Autor: Magno Parra Farfán.

Tipo: Tesis, 2025

Resumen: En esta tesis el autor desarrolla un sistema de control embebido inteligente, con la finalidad de manejar una prótesis transradial a partir de señales electromiográficas (EMG) que sirven para definir movimientos deseados de la mano en tiempo real. El sistema embebido se soporta en un microcontrolador Raspberry Pi Zero 2 W, usando un algoritmo propio PUCP-IArm, en los resultados el autor reporta tiempos de respuesta inferiores a 0.4seg.entre la captura de la señal y la generación de la acción en la prótesis.

Título: Diseño e implementación de un sistema embebido portátil para la adquisición y procesamiento de señales electromiográficas del antebrazo

Autor: Julio Eduardo Reátegui Pinazo.

Tipo: Tesis, 2024

Resumen: En esta trabajo de investigación el autor presenta el diseño e implementación de un sistema embebido que utiliza la lectura a través de ocho sensores de señales electromiográficas (EMG) procedentes del antebrazo, para implementar el reconocimiento de patrones asociados a cinco posibles movimientos de la mano, el autor refiere que los resultados muestran una precisión aproximada de 94.8 %.

Título: Estudio para el desarrollo de un prototipo de prótesis robótica de mano controlada por señales musculares para manejar vehículos automáticos

Autor: Renzo José Saldaña Suárez.

Tipo: Tesis, 2020

Resumen: En esta tesis el autor propone un prototipo de prótesis robótica de mano, cuyo movimiento es controlado a partir de la medición de señales mioeléctricas (EMG), analizando movimientos de dedos y muñeca, es decir estudiando y seleccionando un conjunto de movimientos relevantes.

Título: Desarrollo y control de una mano robótica con visión artificial para la emulación de movimientos del ser humano

Autor: Fabio L. Fiestas Cobeñas & César A. Tesén Bardales.

Tipo: Tesis, 2024

Resumen: En este trabajos los autores desarrollan una investigación que utiliza técnicas de visión artificial utilizando OpenCV y Mediapipe), sobre un sistema embebido desarrollado a partir del microcontrolador Arduino, para que a partir de un conjunto de servomotores, se pueda emular movimientos humanos reales en una mano robótica.

CAPÍTULO II

2. MARCO TEÓRICO

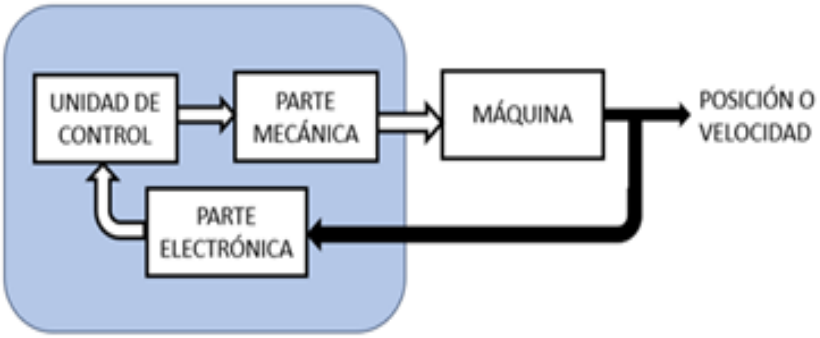
2.1 Control numérico

El control numérico (CN) es una técnica de control utilizada especialmente en la automatización de máquinas herramienta, con la finalidad de controlar su operación, secuencia de movimientos, a partir de instrucciones codificadas en forma numérica, la secuencia de estas instrucciones recibe el nombre de programas de CN, los que permiten especificar parámetros como velocidad, posición, trayectorias y funciones auxiliares de forma precisa y que puede reproducirse en cualquier momento (Groover, 2014).

El sistema que implementa el CN, requiere tres tipos de elementos para gobernar el movimiento de los ejes y precisa de diferentes elementos que se pueden agrupar en tres grupos (Lamikiz,2011):

- Parte mecánica: compuesta por actuadores como servomotores, husillos y otros, encargados de establecer el movimiento continuo sobre los ejes definidos, su respuesta debe ser rápida para corregir la posición de cada eje continuamente.
- Parte electrónica: compuesta por sensores encargados de medir la posición y velocidad sobre cada eje, esto permite al CN calcular las posiciones y gobernar a la parte mecánica.
- Parte algorítmica: programa definido por el usuario, este software es leído e interpretado por el CN, para establecer las órdenes correctas, reside en una unidad de control

Figura 1.- *Sistema con CN*



Fuente: Elaboración propia

Un ejemplo de esta estructura se muestra en la Figura 2.

Figura 2.- *Sistema en base a CN*

Fuente: (Lamikiz,2011)

Son ventajas del control numérico:

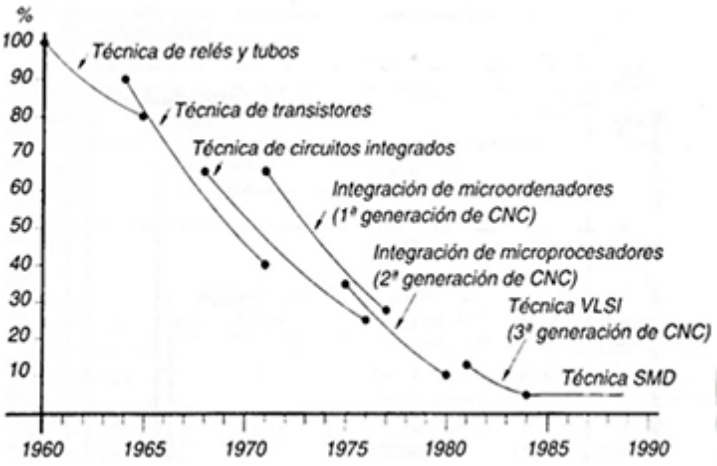
- La posibilidad de automatizar todos los movimientos de una máquina, lo que posibilita el trabajo permanente sin operario en la estación y se excluye los errores humanos en el funcionamiento.
- Se mejora la precisión y velocidad de los movimientos, independientemente de las veces que el proceso se repita.
- El control CNC es altamente flexible, dado que cambios en los movimientos de la máquina, solo es necesario cambiar de programa.

En la Figura 3 se muestra como los costos de la implementación de estas técnicas de control se han ido reduciendo. Las áreas de aplicación más comunes de esta técnica de control son: centros de mecanizado y torneado, máquinas de medición por coordenadas, robots, rectificadoras, prensas y máquinas de deformación, máquinas de cortes, medición 3D, máquinas de bobinar, máquinas de oxicorte, máquinas de electroerosión, cizallas, entre las más importantes.

Hay que notar entonces que las aplicaciones se pueden dividir en dos grandes categorías:

- Aplicaciones con máquina herramienta: como taladrado, laminado, torneado, etc.
- Aplicaciones sin máquina herramienta, como ensamblaje, trazado e inspección.

Figura 3.- *Evolución de los costos de implementación del control numérico*



Fuente: (Lamikiz,2011)

Hay diferentes criterios para clasificar los sistemas basados en CN (Control Numérico, sf):

a) Según el sistema de referencia: dependiendo del sistema de referencia estándar para especificar las posiciones relativas de la máquina herramienta con respecto al trabajo a realizar, se tiene:

- Fijos: el origen se localiza en una posición respecto a la mesa de trabajo (esquina inferior izquierda de la mesa de trabajo) y las posiciones se localizan a lo largo de ejes XY positivos y relativos a ese origen.
- Flotantes: el origen del sistema está en cualquier posición de la mesa de trabajo. El programador decide donde está situado, según se necesite, por ejemplo, si se necesita simetría podría estar en el centro de la mesa.

b) Según el control de las trayectorias: pueden ser

- CN punto a punto: lento y sencillo
- CN paraxial: rápido pero complejo, controla la posición final
- CN continuo o de contorneado: más común al manejar cada eje independientemente y en simultáneo, hay un control en cada punto
- Control numérico punto a punto

c) Según el tipo y naturaleza de los accionamientos:

- Hidráulicos

- Eléctricos
- Neumáticos.

d) Según el bucle de control

- En bucle cerrado: a través de sensores se mide el valor de la salida, y se compara en todo instante con un valor de referencia para proporcionar la señal de control
- En bucle abierto: sin realimentación.

e) Según la tecnología de control:

- Control Numérico Básico (CNB) la función de control es implementada por un circuito electrónico específico, usando lógica cableada, trabaja sin memoria, ejecuta un programa de forma secuencial. Tienen gran volumen y difícil mantenimiento.
- Control Numérico Computarizado (CNC): usando uno o varios microprocesadores, incluyen memorias de semiconductores para almacenamiento de programas y datos, asimismo unidades de entrada y salida para interactuar con operarios. Aportan fiabilidad, inmunidad ante ruidos y las bondades de la programación.
- Control Numérico Adaptativo (CNA): el controlador detecta en tiempo real el estado de la máquina y optimiza las velocidades y las posiciones; para ello, hace uso de múltiples sistemas sensoriales. Es necesario cuando hay una geometría variable, complejidad en las operaciones, variaciones en los materiales o desgaste en las herramientas.

2.2 Robótica

La Administración Nacional de Aeronáutica y el Espacio (NASA), define la robótica como el estudio de los robots, y define los robots como máquinas con un alto nivel de autonomía que se emplean para realizar trabajos humanos. Existen robots con una independencia total y otros que requieren de un operario para funcionar (Blog de Ingeniería, sf).

Surge como la intersección de la ciencia, ingeniería y tecnología, ya que usa conocimiento científico, computacional e informático, para posibilitar el diseño, desarrollo, programación, producción y aplicación de los robots. Por ello usa saberes de diferentes ingenierías, como la mecánica, eléctrica, electrónica y biomédica; y ciencias, como física, matemática, química y las ciencias computacionales. Las áreas de aplicación de los robots son diversas:

- Procedimientos médicos innovadores
- Seguridad al suplantar al humano
- Situaciones catastróficas
- Compañía
- Funciones laborales y domésticas
- Productividad y eficiencia en sistemas automáticos

En cuanto a su clasificación, los robots se distinguen por su geometría (tamaño, forma y materiales), el nivel de autonomía, inteligencia, funcionalidad y capacidad, es decir estructura y funcionamiento o la cronología de los robots.

2.2.1 Tipos de robot según su cronología

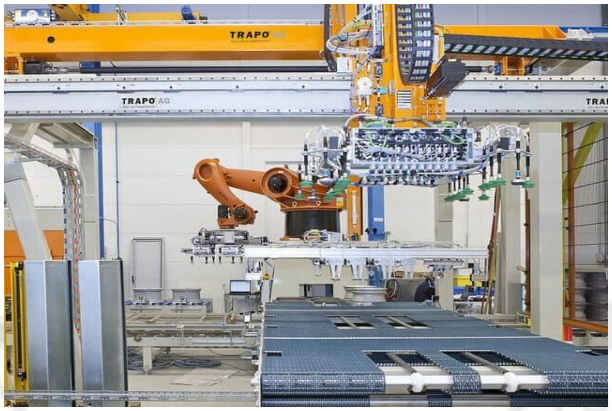
- Robots de manipulación o de primera generación: con un factor mecánico prioritario, aquí están los robots de manipulación, con control secuencial fijo, diseñados para una tarea fija, como se muestra en la Figura 4.

Figura 4.- *Robot de primera generación*

Fuente: (Blog de Ingeniería,sf).

- Robots de aprendizaje o de segunda generación: repiten una secuencia de movimientos previamente ejecutada por un operador. Predominantemente mecánicos, realimenta y almacena información del exterior. Usan un control en base a secuencias numéricas, muy usados en la industria automotriz

Figura 5.- *Robot de segunda generación*



Fuente: (Blog de Ingeniería,sf).

- Robots con control sensorizado o de tercera generación: computadoras que ejecutan órdenes según un programa hacia el manipulador para realizar una tarea.

El uso de sensores da información sobre el entorno, dimensionando el espacio para ajustar la estrategia de control, como se ve en la Figura 6.

Figura 6.- *Robot de tercera generación*

Fuente: (Blog de Ingeniería,sf).

- Robots inteligentes o de cuarta generación: utilizan sensores más complejos para controlar procesos en tiempo real, con múltiples funcionalidades. La Figura 7 muestra a Pepper de SoftBank Robotics, que entiende emociones humanas. Usan lógica difusa y redes neuronales para aprender directamente del entorno, sin requerir modelos matemáticos o abordando conocimientos imprecisos o subjetivos.

Figura 7.- *Robot de cuarta generación*

Fuente: (Blog de Ingeniería,sf).

- Robots 5G o de quinta generación: desarrollos actuales basados en inteligencia artificial, usan la llamada arquitectura de subsunción, incorporan biotecnología y nanotecnología.

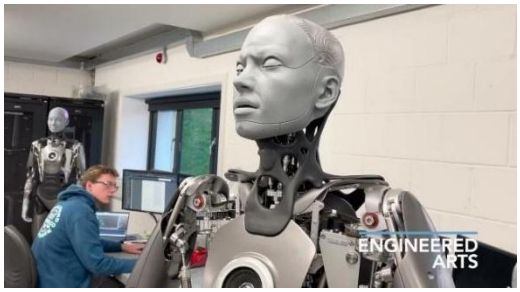
Figura 8.- *Robot de quinta generación*

Fuente: (Blog de Ingeniería,sf).

2.2.2 Tipos de robots según su estructura

- Robots poliarticulados: tienen una posición estática y una estructura para mover sus terminales, en un espacio limitado y según uno o más sistemas de coordenadas, tienen escaza libertad y autonomía. Destinados a la productividad, incluyen robots manipuladores, industriales y cartesianos, que ocupan una zona de trabajo amplia, actuando sobre objetos con un plano simétrico vertical o para reducir el espacio en el suelo.
- Robots móviles: son máquinas automáticas con alta capacidad de desplazamiento de forma autónoma y en diversos entornos. Requieren un sistema locomotor rodante y usan sensores para el desplazamiento. Se usan para transportar piezas u objetos en una cadena productiva.
- Robots androides y ginoides: antropomorfos o de forma humana, reproducen el comportamiento cinemático del ser humano, como Sophia o Mesmer.

Figura 9.- *Robot antropomorfo*



Fuente: (Blog de Ingeniería,sf).

- Robots zoomórficos: imitan la estructura y forma de animales, replicando su cinemática y habilidades motoras.
- Robots híbridos

2.1. Cinemática y dinámica de robots manipuladores

Los robots manipuladores de cadena cinemática abierta constan de dos elementos eslabones y articulaciones. Los primeros de naturaleza mecánica conforman la estructura del robot, mientras las articulaciones son las uniones de los eslabones para formar la cadena cinemática. Esta se denomina abierta cuando el conjunto de articulaciones y eslabones termina en un punto donde el último eslabón no se encuentra conectado a ningún otro. Los articulaciones hacen que los eslabones tengan un movimiento relativo entre sí, pueden ser rotacionales o traslacionales.

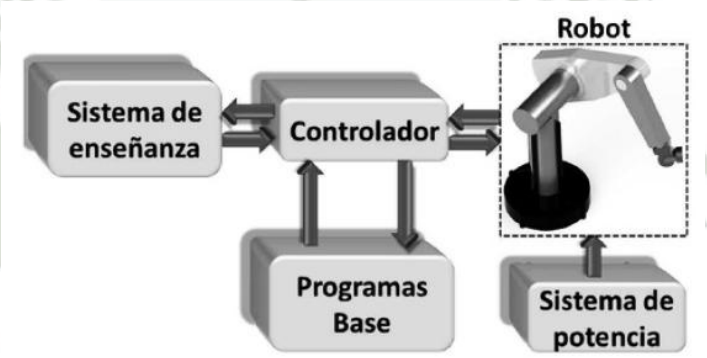
Dado un eslabón arbitrario l_i existe una articulación i-ésima que une las articulaciones l_i y l_{i+1} rota o se desplaza con respecto al eje z_i, entonces para una articulación siempre habrá una variable que está cambiando con respecto en el tiempo, esta se denomina θ_i (rotacional) o d_i (desplazamiento) y en torno a las cuales se modela la cadena (Miranda,2016). En la Figura 10 se muestran articulaciones y los grados de libertad (GDL) respectivos; donde GDL es el conjunto de variables independientes que se requieren para definir con precisión la posición y orientación del elemento final del robot. Los eslabones de los robots se mueven gracias a los actuadores, los que son hidráulicos, neumáticos o eléctricos, según se necesiten. Si las articulaciones son independientes, su número determina el GDL.

Figura 10.- *Articulaciones y GDL*

Fuente: adaptado de (Miranda,2016)

Los elementos de un sistema robótico se muestran en la Figura 11.

Figura 11.- *Elementos de un sistema robótico*



Fuente: adaptado de (Miranda,2016)

Aquí se tiene:

a) Sistema de potencia: contiene la fuente de alimentación y amplificadores para alimentar los actuadores.

b) Efector final: elemento que implementa la tarea que tenga el robot.

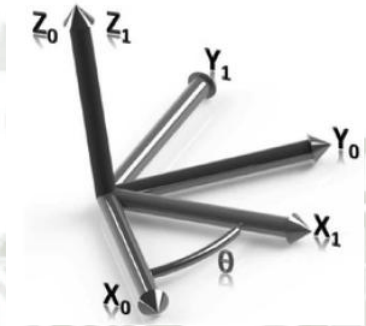
c) Sensores: miden las señales necesarias para controlar al robot manipulador.

d) Controlador: dispositivo computacional que implementa algoritmos para que el robot desarrolle sus tareas, puede considerar sistemas de aprendizaje basados en técnicas de inteligencia artificial.

El modelamiento de la cinemática del robot es importante porque permite implementar

diferentes esquemas de control, la herramienta más usada son las matrices de rotación, que muestra un cambio en la orientación con relación al mismo punto de referencia donde $\{0\} = [x0, y0, z0]$ es la orientación inicial y $\{1\} = [x1, y1, z1]$ es la orientación final originada por una rotación θ, como se muestra en la Figura 12.

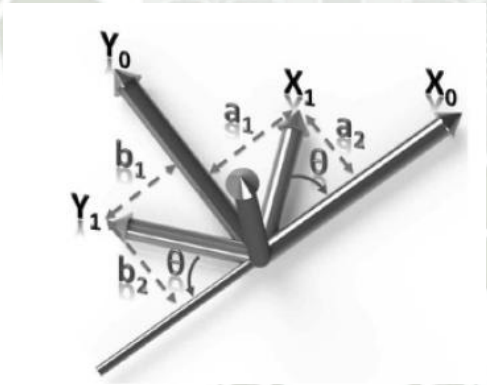
Figura 12.- *Rotación Θ de un robot con respecto al eje z (bidimensional)*



Fuente: adaptado de (Miranda,2016)

En la Figura 13 se identifican los componentes de cada vector.

Figura 13.- *Componentes de los vectores $x0, x1, y0, y1$ de la Figura 11*



Fuente: adaptado de (Miranda,2016)

Siendo x_1^0 la orientación del eje x del sistema {1} con respecto al eje x del sistema {0}, y y_1^0 tiene el mismo significado para el eje y

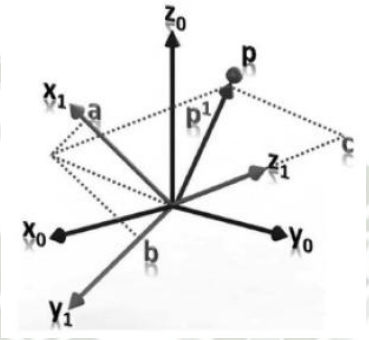
$$x_1^0 = \begin{bmatrix} \cos\theta \\ sen\theta \end{bmatrix}, \qquad y_1^0 = \begin{bmatrix} -sen\theta \\ \cos\theta \end{bmatrix} \tag{1}$$

La matriz de rotación será

$$R_1^0 = \left[x_1^0, y_1^0 \right] = \begin{bmatrix} \cos\theta & -sen\theta \\ sen\theta & \cos\theta \end{bmatrix} \tag{2}$$

El caso tridimensional se muestra en la Figura 14, donde la matriz de rotación se define en (3)

Figura 14.- *Rotación tridimensional*



Fuente: adaptado de (Miranda,2016)

$$R_1^0 = \begin{bmatrix} (x1,x0) & (y1,x0) & (z1,x0) \\ (x1,y0) & (y1,y0) & (z1,y0) \\ (x1,z0) & (y1,z0) & (z1,z0) \end{bmatrix} \tag{3}$$

En (4) se muestran las tres matrices básicas que se obtienen al rotar un ángulo θ con respecto a los ejes x, y y z respectivamente

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -sen\theta \\ 0 & sen\theta & \cos\theta \end{bmatrix} \quad R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & sen\theta \\ 0 & 1 & 0 \\ -sen\theta & 0 & \cos\theta \end{bmatrix} \quad R_{z,\theta} = \begin{bmatrix} \cos\theta & -sen\theta & 0 \\ sen\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Y se denomina composición de rotaciones a la aplicación secuencial de matrices de rotación con respecto a los ejes (x, y, z) necesarias para alcanzar un desplazamiento o rotación desde un punto inicial a un punto final en el espacio, a través de los actuadores incluidos en cada articulación del robot manipulador. Asimismo, en (5) se dan las propiedades más importantes de estas matrices de rotación

$$R_{x,0} = I \qquad R_{x,\alpha} R_{x,\beta} = R_{x,\alpha+\beta} \qquad [R_{x,\alpha}]^{-1} = R_{x,-\alpha} \tag{5}$$

Las composiciones de rotación más comunes son:

- Representación Roll-Pitch-Yaw (RPY): con respecto a ejes fijos.
- Ángulo de Euler Z-Y-Z: con dos rotaciones secuenciales sobre dos de los ejes.
- Representación eje-ángulo: eje arbitrario.
- Cuaterniones: usa un conjunto de parámetros definidos a partir de los ejes X-Y-Z.

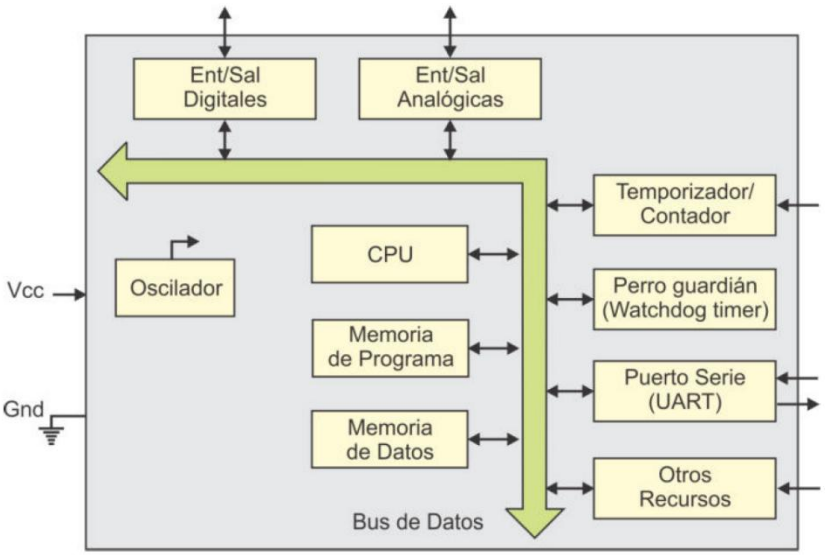
Finalmente hay que indicar que el movimiento de un robot es la combinación de transformaciones generadas por rotaciones y traslaciones.

2.2. Microcontrolador

Se define un microcontrolador (Santiago,2021) o MCU (Micro Controller Unit) como un circuito integrado de gran escala de integración (VLSI very large scale integration) que contiene una Unidad Central de Procesamiento CPU, memoria para código y datos, temporizadores, fuentes de interrupción y otros recursos para el desarrollo de aplicaciones específicas. Aunque un microcontrolador tiene más recursos que un microprocesador, es limitado en:

- Velocidad de procesamiento
- Capacidad de direccionamiento
- Tamaño de datos

Figura 15.- *Estructura de un microcontrolador*

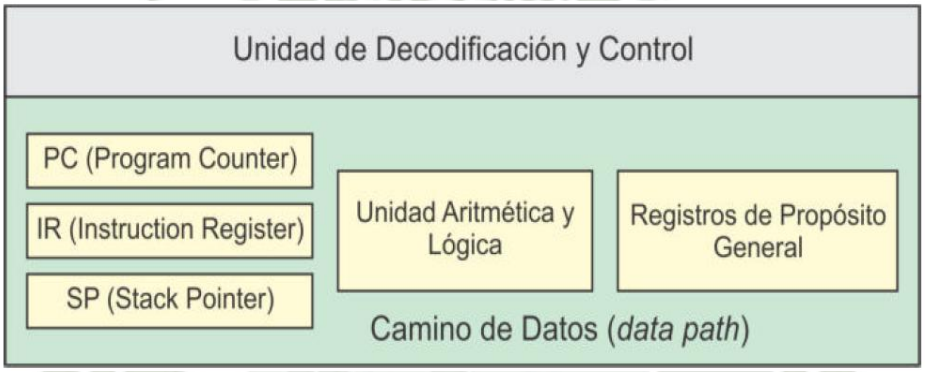


Fuente: (Santiago,2021)

Un microprocesador es de propósito general, mientras que el microcontrolador es de propósito específico para desarrollar sistemas embebidos, su estructura se muestra en la Figura 15.

a) CPU: administra actividades y opera sobre los datos al ejecutar programas desde la memoria de código. Ejecutar una instrucción requiere: leerla, decodificarla y ejecutarla. Las instrucciones pueden ser: aritméticas, lógicas, de transferencia de datos y de bifurcaciones y pertenecen a un repertorio o juego donde les corresponde un código binario único (opcode). En la Figura 16 se muestra los elementos de una CPU.

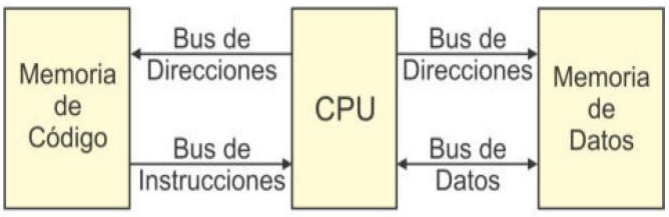
Figura 16.- *Elementos de una CPU*



Fuente: (Santiago,2021)

b) Sistema de memoria: alineado a la arquitectura Harvard se tiene dos espacios de almacenamiento separados, para datos e instrucciones. Se utilizan memorias no volátiles EPROM o similares

Figura 17.- *Sistema de memoria*



Fuente: (Santiago,2021)

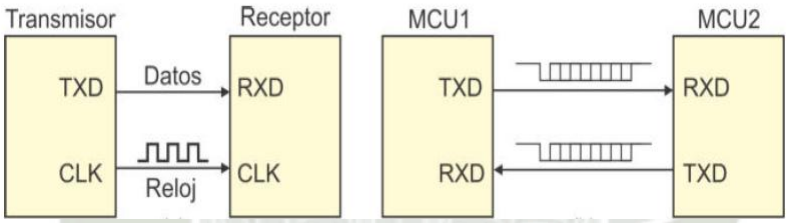
c) Oscilador: circuito RC o en base a cristal, que genera la frecuencia de trabajo con la que se toman las instrucciones y se ejecutan.

d) Temporizador/contador: como temporizador maneja funciones periódicas y como contador permite manejar cantidades de eventos. Es un registro de n-bits que se incrementa en cada ciclo o evento, hasta la ocurrencia de un desbordamiento y posterior reinicio. Existe la posibilidad de precargas, esto permite generar división de frecuencia y pulsos de anchos variables (PWM, pulse width modulation)

e) Temporizador guardián (WDT watchdog timer): temporizador que al desbordar genera el reinicio de todo el sistema, funcionalidad que se utiliza en situaciones anómalas.

f) Puerto serie: en base a un UART (Universal Asynchronous Receiver and Transmitter), para comunicación con sistemas externos, usando protocolos estándares, puede ser síncrona o asíncrona. Como se muestra en la Figura 18.

Figura 18.- *Transmisión serial síncrona y asíncrona*



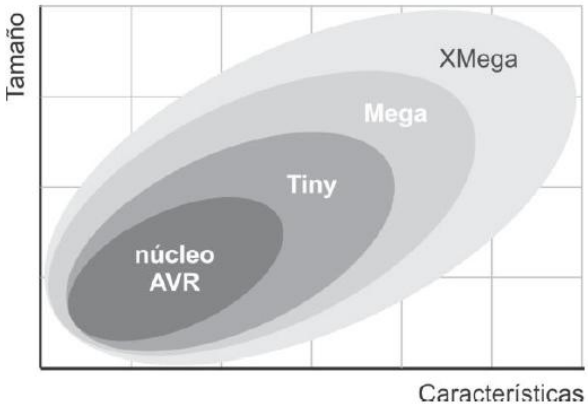
Fuente: (Santiago,2021)

g) Entradas y salidas digitales y analógicas: los microcontroladores incluyen puertos para su comunicación con el exterior, en el caso de las analógicas tienen incorporados los conversores analógico-digital ADC y digital-analógico DAC

2.4.1 Plataforma Arduino

En base a un microcontrolador AVR, procesador RISC (Reduced Instruction Set Computer) de arquitectura Harvard, nace en 1992 en Noruega y tomó su forma comercial en la empresa Atmel (Santiago,2021), este núcleo es la base de más de 50 microcontroladores, organizados en tres categorías: Tiny, Mega y XMega según la cantidad de recursos. Estos sistemas embebidos soportan programación en Java y un ambiente parecido a C++, sobre una IDE (Integrated Development Environment) que genera sketch con extensión .ino siendo open source.

Figura 19.- *Categorías de microcontroladores AVR*

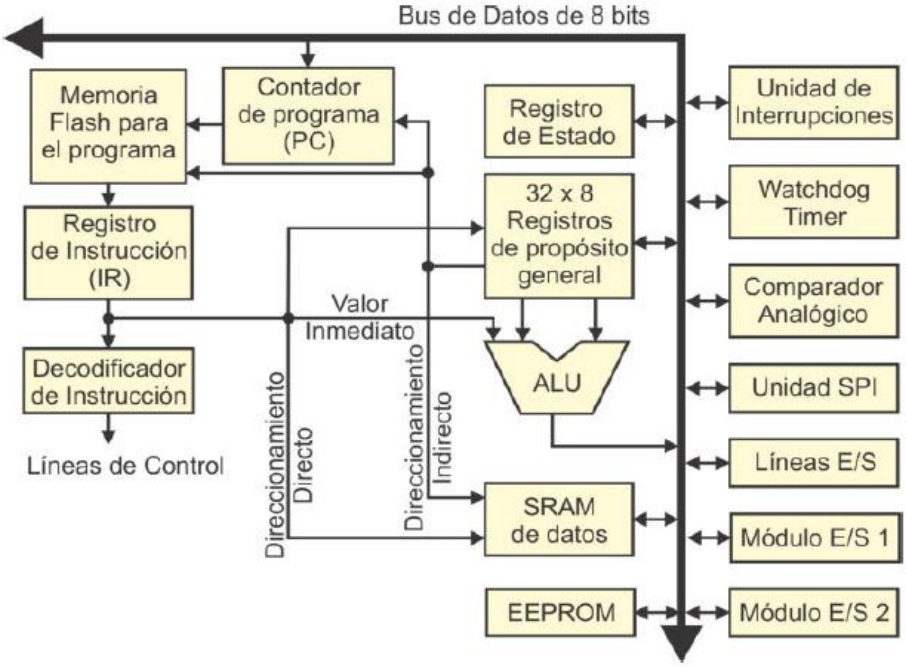


Fuente: (Santiago,2021)

La simplicidad permite proyectos en gadgets, robots, automatización, vehículos, etc.

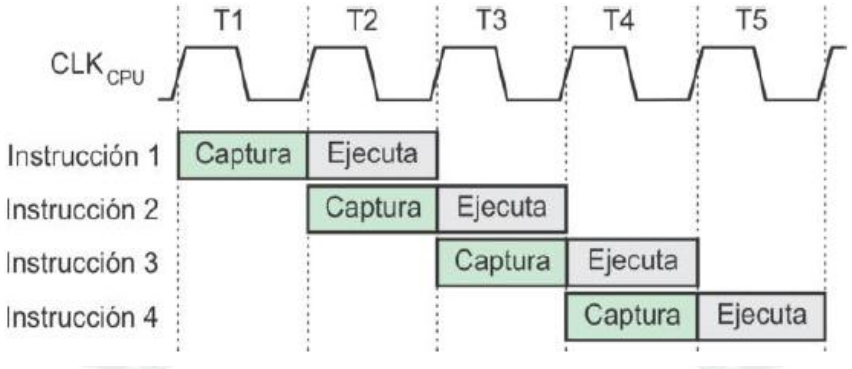
Un núcleo AVR tiene la estructura mostrada en la Figura 20, donde se observa la ALU, memoria SRAM de datos, EEPROM para el firmware, memoria flash para instrucciones, 32 registros generales, registro de estado, registro de instrucciones, decodificador de instrucciones, unidad de manejo de interrupciones, timer y módulos de entrada/salida. Como se muestra en la Figura 21, las fases de captura de instrucciones y ejecución de estas se realiza en paralelo.

Figura 20.- *Núcleo AVR*



Fuente: (Santiago,2021)

Figura 21.- *Paralelismo en el Núcleo AVR*



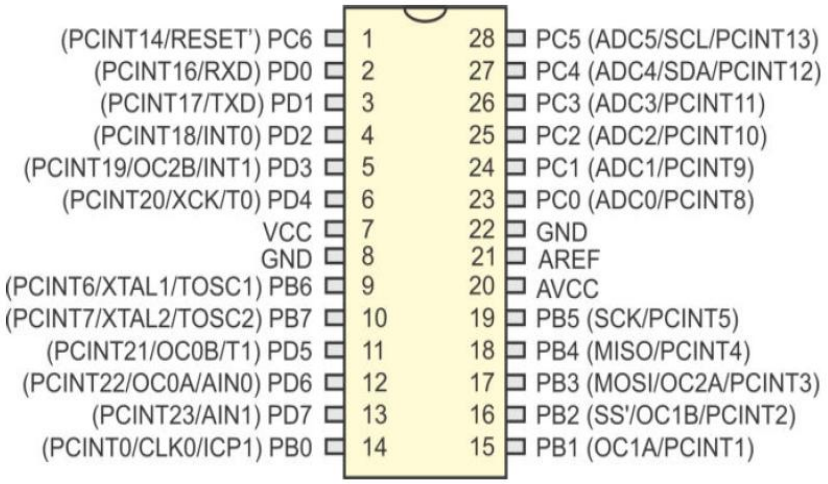
Fuente: (Santiago,2021)

En esta investigación se utiliza un Arduino Mega, el cual está basado en el ATMega328P, que presenta las siguientes características (Santiago,2021):

- Memoria de datos: 2 kbytes de SRAM y 1 kbyte de EEPROM.
- Terminales de entrada/salida: 23.
- Frecuencia máxima de trabajo: 20 MHz.
- Voltaje de alimentación: 1.8 a 5.5 Volts.
- Temporizadores: 2 de 8 bits y 1 de 16 bits.
- Canales PWM: 6.
- Fuentes de interrupción: 26.
- Interrupciones externas: 2 individuales y 3 por puerto.
- Canales de conversión analógico/digital: 6 de 10 bits en encapsulado
- PDIP, 8 en encapsulados TQFP o MLF.
- Facilidades de Reloj de Tiempo Real.
- Interfaz SPI como Maestro o Esclavo.
- Transmisor/Receptor Universal Síncrono/Asíncrono (USART).
- Interfaz serial de dos hilos (TWI) compatible con I2C.
- Programación In System.
- Oscilador interno configurable.
- Watchdog timer.

La distribución de señales en el patillaje se muestra en la Figura 22.

Figura 22.- *ATMega328P*



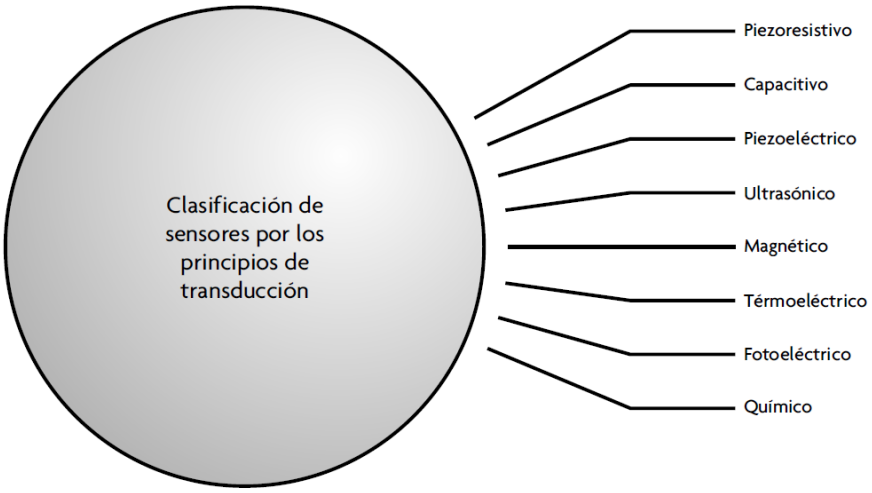
Fuente: (Santiago,2021)

2.3. Sensores y Transductores

Un transductor es un dispositivo que convierte una variable física en otra que tiene un dominio diferente; la diferencia entre un sensor y un transductor es que el transductor sólo cambia el dominio de la variable, en cambio un sensor proporciona una salida útil proporcional a la señal original para ser usada como entrada a un sistema de procesamiento de información para implementar alguna acción sobre el sistema. De forma general, los transductores se dividen en dos ramas: transductores de entrada, usados en un sistema de sensado para medir una variable física cuyo valor será procesado, y transductores de salida, cuando es parte de un sistema de actuación. Un sensor es sólo un transductor de entrada (Corona et al, 2014).

El funcionamiento del transductor/sensor se basa en un principio físico de transformación de energía, dentro de los más importantes tenemos: piezoresistivo (resistencia eléctrica-deformación), capacitivo (movimiento-cambio de capacitancia), piezoeléctrico. (presión-carga eléctrica), ultrasónico (energía mecánica de la onda-energía eléctrica), magnético (campo magnético-corriente eléctrica), térmico (energía térmica-energía eléctrica), fotoeléctrico (señal de luz-señal eléctrica), resistivo (desplazamiento-resistencia).

Figura 23.- *Clasificación de los sensores según el principio de transducción*



Fuente: (Corona,2014)

2.5.1 Clasificación de los sensores según el tipo de variable medida

Los sensores también se pueden clasificar según la variable medida, pero además un mismo sensor puede medir varias variables físicas, esta se muestra en la Tabla 1, esta clasificación es la más usada.

Tabla 1.- *Clases de sensores según variable medida*

Clasificación	Sensores
Según variable física	De posición, velocidad y aceleración
	De nivel y proximidad
	De humedad y temperatura
	De fuerza y deformación
	De flujo y presión
	De color, luz y visión
	De gas y pH
	Biométricos
	De corriente

Fuente: (Corona,2014)

2.5.2 Características de un sensor

Dentro de las características más importantes que deben ser consideradas al momento de elegir un sensor se tiene:

a) Características estáticas: (Corona,2014)

- Sensitividad: entrada mínima que provoca una salida detectable. La representación gráfica se denomina curva de salida, y la pendiente de la tangente a esta curva es la sensitividad del sensor.
- Rango: intervalo entre el valor mínimo y el máximo de la variable física a medir.
- Precisión: grado de repetitividad de una medida.
- Exactitud: diferencia máxima entre la salida del sensor y el valor real de la variable medida, expresada de forma porcentual.
- Linealidad estática: desviación entre la curva dada por el fabricante y la curva de salida real, expresada de forma porcentual en (6).

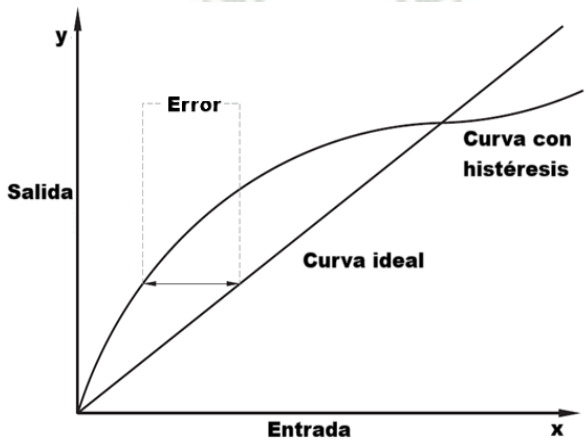
$$\%no\_linealidad = \frac{desviación\_máxima}{valor\_máximo\_a\_escala\_completa} x100 \quad (6)$$

- Offset: corrimiento en el eje de la curva de salida, se calcula como la salida que se da cuando el valor real de la variable es cero.
- Resolución: cambio más pequeño en la variable física que es posible registrar.
- Error estático: error en la medición, debido a errores en la lectura.

b) Características dinámicas:

- Tiempo de respuesta: tiempo desde que la variable sensada presenta un cambio y el sensor lo registra, depende del tipo de magnitud y el sensor utilizado.
- Histéresis: capacidad del sensor para seguir a la curva de salida ideal.

Figura 24.- *Histéresis*



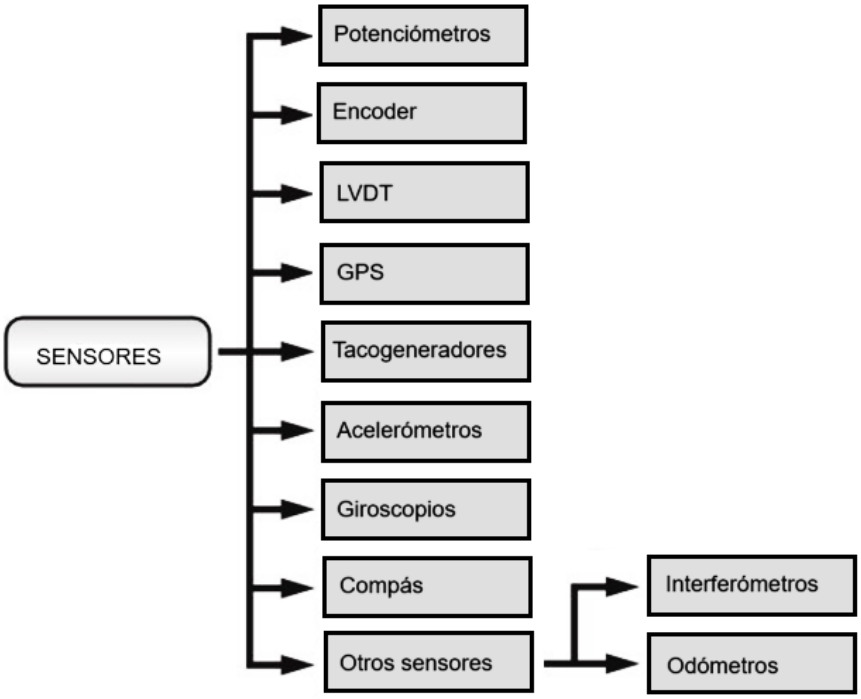
Fuente: (Corona,2014)

- Linealidad dinámica: capacidad para seguir correctamente la curva de salida de la hoja de datos cuando la variable física cambia rápidamente. Es el grado de distorsión del sensor.
- Error dinámico: causado por las cargas inducidas en el sensor debido a los aparatos de medición y a la forma de uso del sensor.

2.5.3 Sensor de desplazamiento/posición potenciómetrico

Hay un conjunto de sensores para medir el desplazamiento, velocidad y aceleración, tanto lineales como rotacionales, lo que se muestra en la Figura 25.

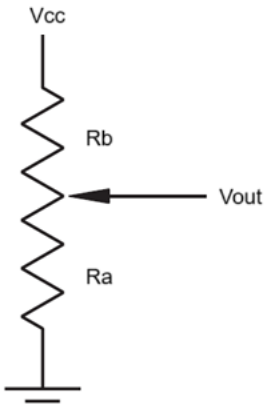
Figura 25.- *Sensores de posición, velocidad y aceleración*



Fuente: (Corona,2014)

El sensor potenciométrico transduce la posición lineal o angular de un objeto a un cambio en la resistencia de un elemento alimentado con un voltaje continuo (Corona,2014). El potenciómetro consta de tres terminales, dos extremos y uno intermedio, funcionando como dos resistencias variables en serie, como se muestra en la Figura 26. Cuando el elemento móvil se desplaza hacia arriba, la resistencia *Rb* disminuye y la resistencia *Ra* aumenta, por lo que *Vout* aumenta también.

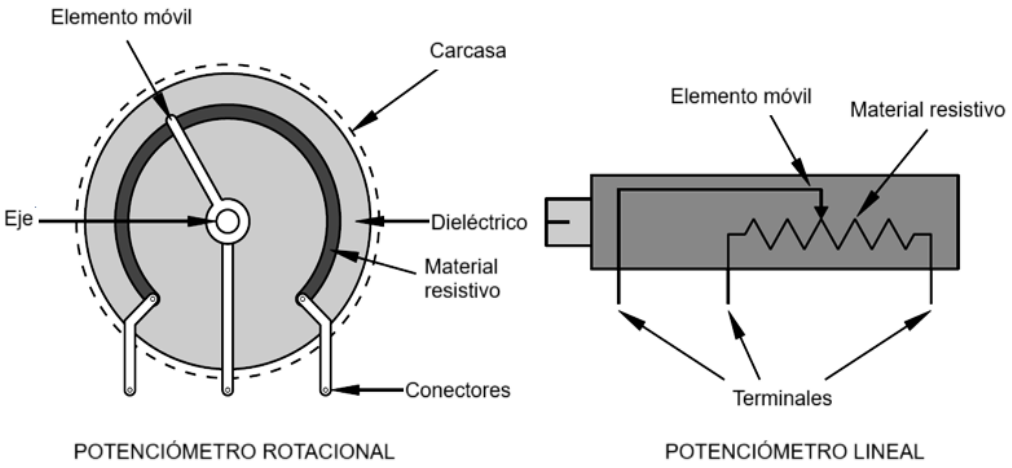
Figura 26.- *Sensor potenciométrico*



Fuente: (Corona,2014)

Existen dos tipos de potenciómetro como se muestra en la Figura 27, el de uso más común es el rotacional.

Figura 27.- *Tipos de sensores potenciométricos*



Fuente: (Corona,2014)

Para lograr la correlación entre voltaje de salida y posición de un objeto, se debe acoplar mecánicamente el terminal móvil del potenciómetro al objeto; si el terminal móvil del potenciómetro está en el extremo superior Vout será máximo (idealmente *Vcc*) y en el extremo inferior *Vout* será casi cero, la transducción responde a la ley (7):

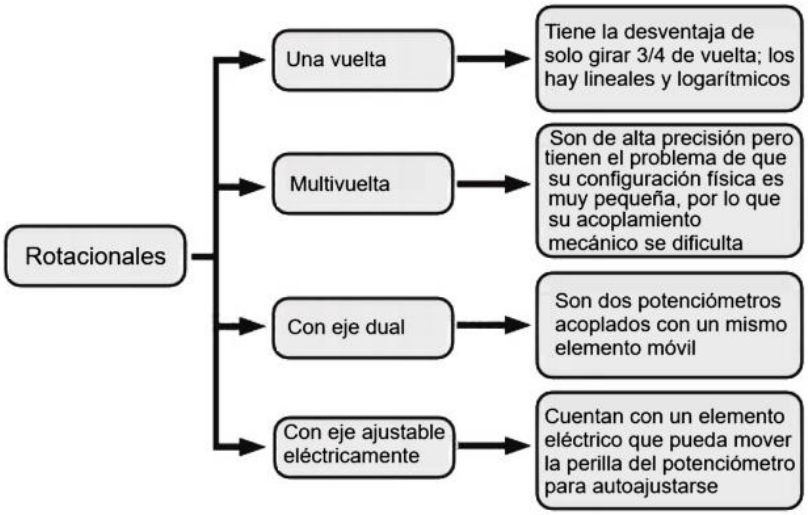
$$Vout = \frac{Ra}{Ra + Rb} Vcc \qquad (7)$$

Al seleccionar el potenciómetro como sensor de posición de un objeto, hay que verificar la linealidad en la salida de este, lo que depende del tipo de material con el que está fabricado, dentro de los más apropiados tenemos:

a) Potenciómetros de bobina: constan de una bobina envuelta sobre un material dieléctrico, la bobina está aislada eléctricamente y el elemento móvil pasa a lo largo de la bobina generando una zona en la que el deslizador y la bobina están en contacto, este sensor es de baja resolución por el mínimo intervalo de resistencia entre una vuelta y otra.
b) Películas de carbono: fabricados de una mezcla de carbono y arcilla, que es la resistencia por la cual pasa el deslizador, colocada sobre un elemento no conductor, el sustrato. Este tiene mejor resolución por un contacto contínuo y meno interferencia mecánica.
c) Elementos plásticos conductores: mezclan carbono con una película conductora, con un contacto más suave con el deslizador, la desventaja es su pobre desempeño a altas temperaturas.
d) Películas metálicas: se deposita una aleación metálica sobre un cerámico; lo que lo hace más duradero.
e) Cermet: abreviatura de "metal cerámico", compuesto de una mezcla de cerámicos y partículas metálicas, colocada sobre un sustrato cerámico y calentada hasta que su unión. De gran durabilidad y poco afectado por la temperatura, con la resistividad del material cerámico.

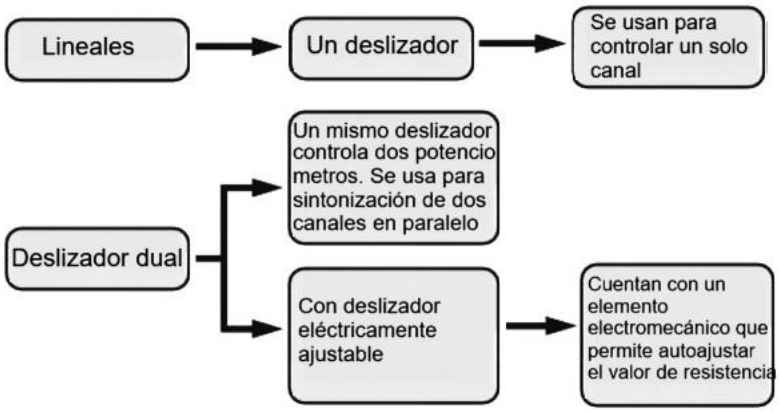
En la Figura 28 se muestra la clasificación de los sensores potenciométricos angulares y en la Figura 29 la de los lineales

Figura 28.- *Tipos de sensores potenciométricos rotacionales o angulares*



Fuente: (Corona,2014)

Figura 29.- *Tipos de sensores potenciométricos lineales*



Fuente: (Corona,2014)

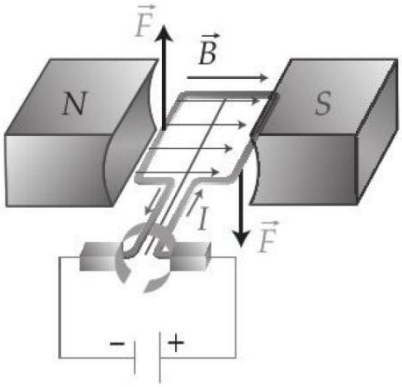
2.4. Actuadores

Son dispositivos que transforman magnitudes físicas en otro tipo de magnitudes con la finalidad de que interactúen con el entorno que los rodea, para lograr un cambio deseado en el estado de un sistema, estas variables de salida pueden ser fuerza, posición, velocidad y aceleración. Los actuadores se clasifican según su funcionamiento en eléctricos, neumáticos e hidráulicos (Corona,2014).

2.6.1 Actuadores eléctricos

Este tipo de actuadores transforman la energía eléctrica en energía mecánica, su funcionamiento se basa en el efecto que produce en una espira conductora por la que circula una corriente y que se halla dentro de un campo magnético. La espira experimenta una fuerza electromagnética que induce un desplazamiento perpendicular al campo magnético, como se muestra en la Figura 30 (Corona,2014).

Figura 30.- *Principio de funcionamiento de un actuador eléctrico*



Fuente: (Corona,2014)

Cuando se requiere una fuerza de desplazamiento mayor se utiliza un conjunto de espiras, denominado devanado del motor. Los motores se clasifican según el tipo de energía eléctrica que usan para generar el movimiento mecánico:

- Motor de corriente continua (CC)
- Motor de corriente (CA)
- Motor paso a paso

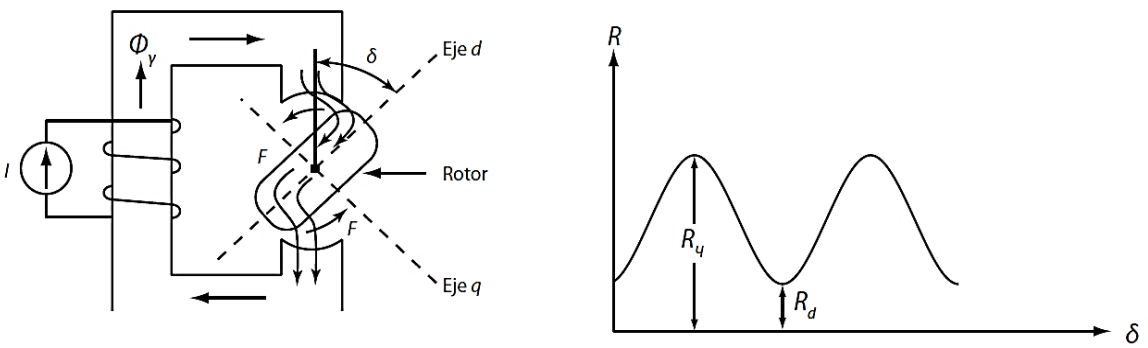
2.6.2 Motor paso a paso

Trabajan con el mismo principio físico que los motores eléctricos rotatorios de corriente continua y alterna, la conmutación se obtiene usando un circuito electrónico o microcontrolador (podría ser parte de un sistema embebido) que genera la secuencia de pulsos de corriente para alimentar los devanados internos o polos del motor, para generar desplazamientos angulares desde 1.8° hasta 90°, según la secuencia (Corona, 2014). Se tienen tres tipos de motores de paso:

a) Motor paso a paso de reluctancia variable

Caracterizado por la variación de la resistencia a la circulación del flujo magnético (reluctancia). Cuando la corriente circula por el inductor, se genera un flujo magnético, que actúa sobre el entrehierro por lo que este varía su resistencia por efecto del cambio de permeabilidad. También depende de la posición del rotor, si su eje *d* está en posición vertical (ángulo δ es 0° o 180°), la reluctancia es mínima, la variación es periódica y depende del ángulo δ., como se ve en la Figura 31.

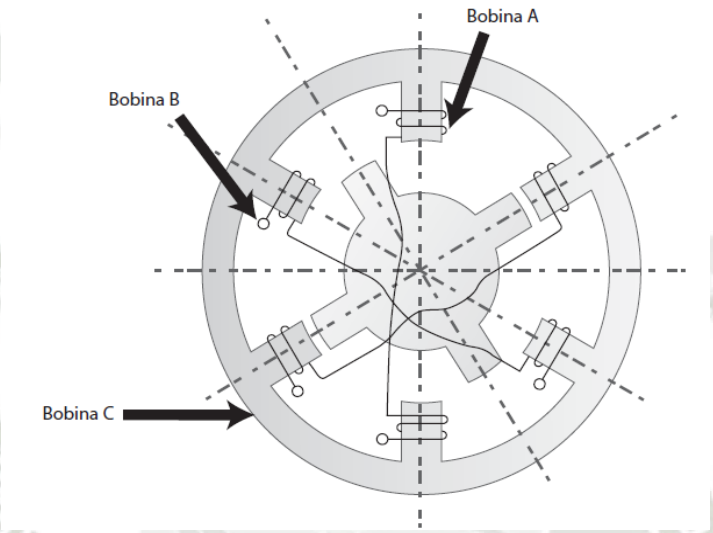
Figura 31.- *Variación de la reluctancia del motor paso a paso*



Fuente: (Corona,2014)

El estator y rotor están constituidos por un conjunto de láminas de acero, para evitar corrientes parásitas. El estator tiene seis polos donde residen las bobinas, excitadas en tiempos diferentes, creando las fases, en la Figura 32, por ejemplo, tres. El rotor tiene cuatro polos, pero sin devanados. Para hacerlo girar se requiere aplicar los voltajes de fase según la Tabla 2, donde una vuelta completa requiere cinco ciclos.

Figura 32.- *Motor paso a paso de reluctancia variable*



Fuente: (Corona,2014)

Tabla 2.- *Secuencia de fases del motor de reluctancia variable*

Ciclo	Fase A	Fase B	Fase B	Posición del Rotor (o)
1	Activa	No Activa	No Activa	0
	No Activa	Activa	No Activa	30
	No Activa	No Activa	Activa	60
2	Activa	No Activa	No Activa	90
	No Activa	Activa	No Activa	120
	No Activa	No Activa	Activa	150
3	Activa	No Activa	No Activa	180
	No Activa	Activa	No Activa	210
	No Activa	No Activa	Activa	240
4	Activa	No Activa	No Activa	270
	No Activa	Activa	No Activa	300
	No Activa	No Activa	Activa	330
5	Activa	No Activa	No Activa	360

Fuente: (Corona,2014)

El ángulo de paso θ_p será:

$$\theta_p = \frac{2\pi}{n_F\, p} \tag{8}$$

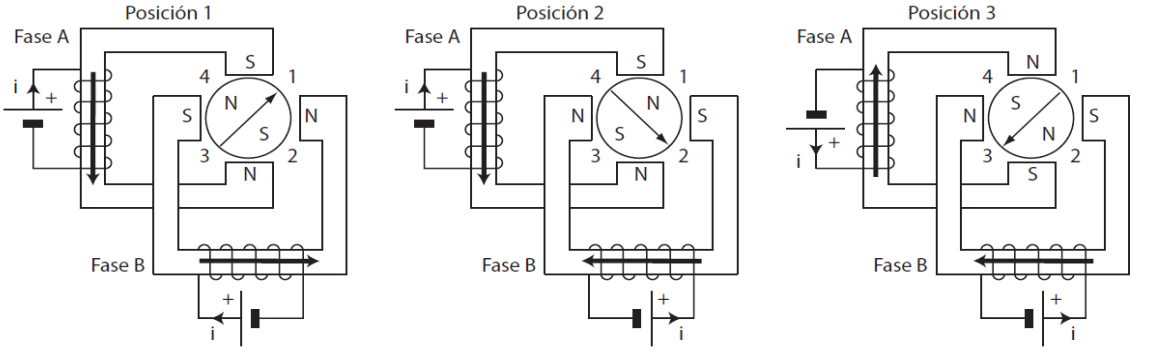
n_F es el número de fases

p es el número de dientes del rotor

b) Motor paso a paso de imán permanente

No posee devanados en el rotor, pero tiene imanes permanentes. Tiene seis o cuatro polos en el estator, y seis, cuatro o dos polos en el rotor, definiendo el ángulo de paso que sería 30°, 90° o 180° respectivamente. En la Figura 33 se muestra el funcionamiento de uno de dos polos en el estator, mostrando la posición del motor para los voltajes aplicados en ambas fases (Corona, 2014). Para hacerlo girar se requiere aplicar los voltajes de fase según la Tabla 3, donde una vuelta completa requiere cinco ciclos.

Figura 33.- *Motor paso a paso con imanes permanentes y dos polos en el estator*



a) Posición inicial. b) Posición después del primer paso. c) Posición después del segundo paso.

Fuente: (Corona,2014)

Tabla 3.- *Secuencia de fases del motor de reluctancia variable*

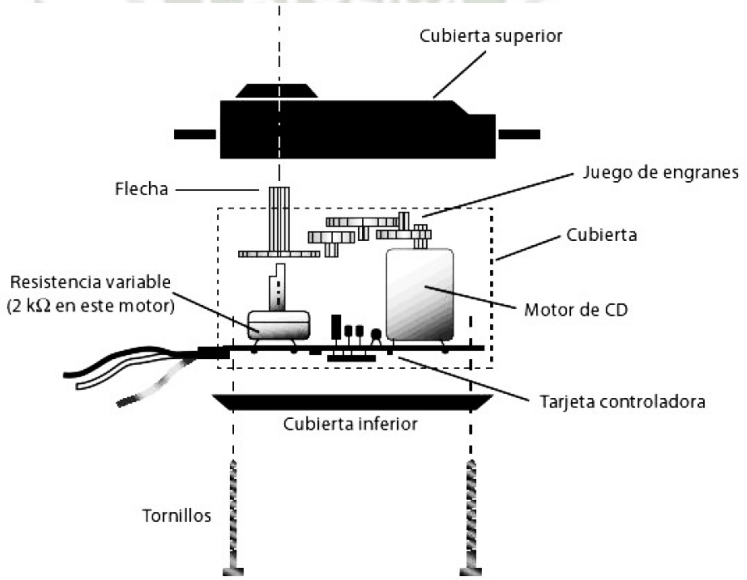
Ciclo	Fase A	Fase B	Posición del Rotor (o)
positivo	Activa	No Activa	0
	No Activa	Activa	90
negativo	Activa (negativo)	No Activa	180
	No Activa	Activa (negativo)	270
positivo	Activa	No Activa	360

Fuente: (Corona,2014)

c) Servomotor

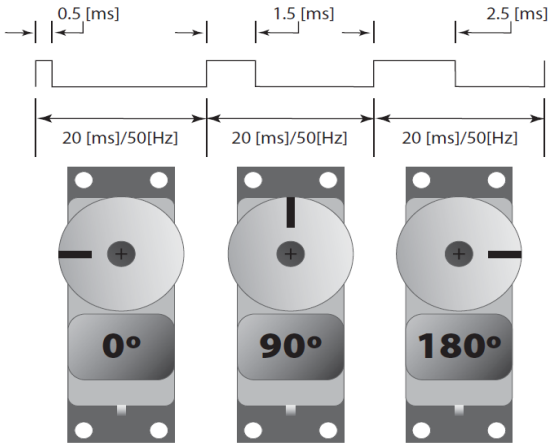
Este actuador está constituido por un motor con un reductor de velocidad y multiplicador de fuerza y cuenta con un circuito de control. El ángulo de giro del eje es casi siempre de 180º, pero puede modificarse para un giro libre de 360º. La estructura se muestra en la Figura 34. Para controlar el servomotor se aplica un pulso de duración y frecuencia específico que es usado por el circuito de control diferencial para llevarlo a una posición. Los servomotores se conectan por tres cables, dos de alimentación *V*CC (4.8 a 6 V) y Gnd, y el tercero para aplicar la secuencia de pulsos de control, como se muestra en la Figura 35. (Corona, 2014)

Figura 34.- *Estructura de un servomotor*



Fuente: (Corona,2014)

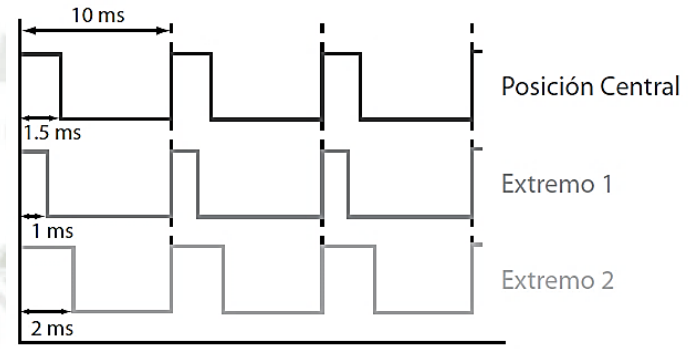
Figura 35.- *Posicionamiento del servomotor según el pulso de control*



Fuente: (Corona,2014)

Por lo explicado a la técnica de control se denomina modulación por ancho de pulso PWM (Pulse Width Modulation), usando un tren de pulsos en el que se varía el tiempo que el pulso está en el nivel alto, con un periodo constante, para establecer la posición deseada del eje, como se muestra en la Figura 36. Normalmente los sistemas embebidos como Arduino® ya disponen de hardware específico.

Figura 36.- *Posicionamiento por PWMl*



Fuente: (Corona,2014)

Los márgenes de operación del servomotor están limitados por el ancho de pulso máximo y mínimo (común entre 1 y 2 mseg.) para obtener posiciones extremas de 0º y 180º, entonces un ancho de 1.5 mseg. definiría la posición central o 90º. Es posible conseguir ángulos mayores de 180º, con pulsos fuera del rango indicado, dependerá del tope mecánico del potenciómetro.

El intervalo entre pulso y pulso no es importante, pero se suele usar valores entre 10 y 30 mseg. Finalmente, para mantener el servomotor en una posición se debe aplicar en forma continua el pulso del ancho necesario, por lo que se requiere un sistema de control en lazo cerrado para corregir la distorsión por alguna interferencia.

2.5. Lenguaje de programación

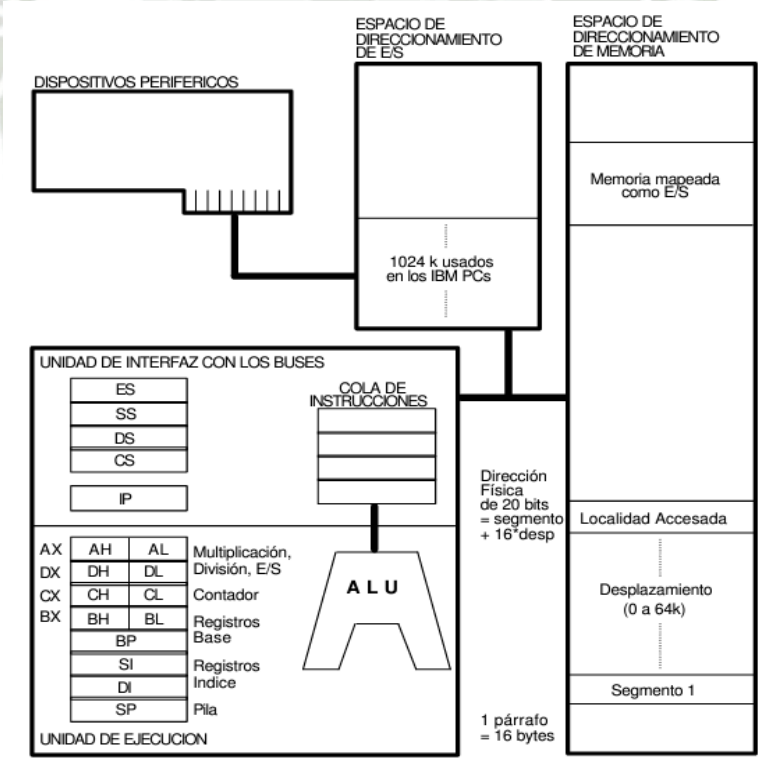
La definición de este considera que es una herramienta para desarrollar software o programas de computadora, los que definen el funcionamiento de los componentes físicos y lógicos de una computadora. Lo anterior se logra mediante la creación e implementación de algoritmos de precisión que se utilizan como una forma de comunicación humana con la computadora (UAPA, 2014). Considera un conjunto de símbolos y reglas sintácticas y semánticas.

La generación de un programa para la comunicación usuario-máquina es un proceso que incluye el análisis, diseño, implementación, prueba y depuración del algoritmo, usando un lenguaje que compila y genera código máquina (1s y 0s) ejecutable por la computadora. El elemento más importante del lenguaje de programación es el intérprete que lee las instrucciones línea por línea y genera el código máquina correspondiente (compilación y generación del ejecutable).

2.7.1 Lenguaje ensamblador

Es el lenguaje de más bajo nivel que opera un juego de instrucciones para el manejo directo de los recursos y elementos del microprocesador, cada una de las instrucciones tiene un equivalente directo en código binario (Medina,1992), así por ejemplo si se tratase de un microprocesador 8086 de la familia INTEL, cuya estructura se muestra en la Figura 37, la misma que considera registros, una pequeña memoria para almacenar instrucciones y una ALU, para procesar, implementar operaciones de entrada/salida y operaciones sobre memoria.

Figura 37.- *Microprocesador 8086*



Fuente: (Medina,1992)

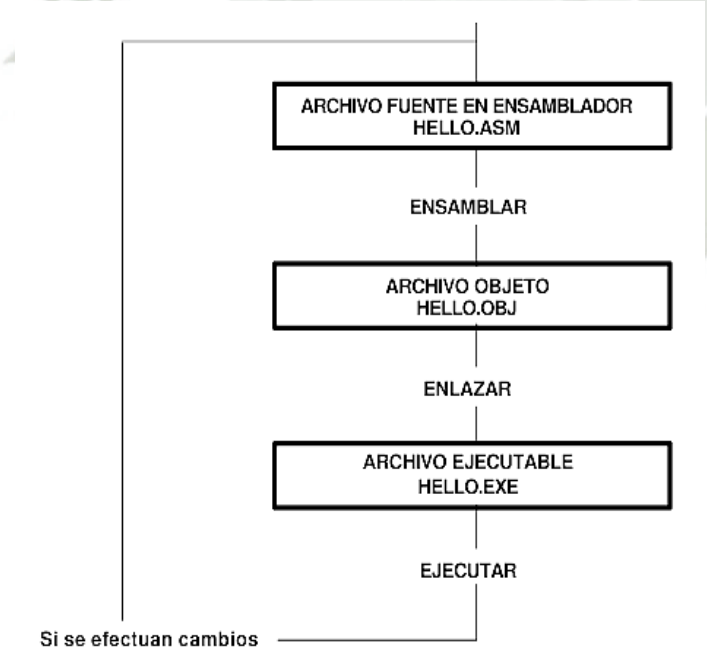
El lenguaje ensamblador considera varios tipos de instrucciones:

- De transferencia de datos

- Aritméticas
- Lógicas
- Manejo de cadenas
- Control del contador de programa
- Saltos condicionales
- Control del procesador
- Entrada y salida
- Manejo de interrupciones
- Rotación y desplazamiento

El ciclo de vida de un programa en ensamblador se muestra en la Figura 38, donde el lenguaje assembler convierte el programa fuente en una forma intermedia llamada programa objeto, y la etapa de enlace combina uno o más módulos objeto en un archivo ejecutable. Mientras que en la Figura 39 se muestra un programa ejemplo, que inicia con la declaración de variables, el código de programa y la finalización.

Figura 38.- *Ciclo de vida de un programa en assembler*



Fuente: (Medina,1992)

Figura 39.- *Estructura de un programa en assembler*

```
MODEL           Small
STACK           100h
DATASEG
Tiempo          DB ' ¿ Es de mañana o de tarde (M/T) ?$'
BuenosDias      DB 'Buenos días, Mundo',13,10,'$'
BuenasTardes    Db ´Buenas tardes, Mundo',13,10,'$,
CODESEG
Inicio:         mov ax,@data
                mov ds,ax                   ; DS ahora apunta al segmento de datos.
                mov dx,OFFSET Tiempo        ; Apuntador al mensaje Tiempo.
                mov ah,9                    ; Función de impresión de cadenas.
                int 21h
                mov ah,1                    ; Función de lectura de un carácter desde el teclado.
                int 21h
                cmp al,'m'                  ; ¿ Es de mañana ?
                jz AM                       ; si
                cmp al,'M'
                jz AM
                mov dx,OFFSET BuenasTardes ; no, Es de tarde
                jmp MostrarMensaje
AM:             mov dx,OFFSET BuenosDias
MostrarMensaje: mov ah,9                    ; Función de impresión de cadenas.
                int 21h
                mov ah,4Ch                  ;  Transferencia del control al DOS.
                int 21h
END             Inicio
```

Fuente: (Medina,1992)

2.7.2 Lenguaje C++

Desarrollado por Bjarnes Stroutstrup en los laboratorios AT&T Bell en 1995, es un lenguaje de programación orientado a objetos de nivel medio, por su fácil interfaz con el lenguaje máquina (Osorio, 2006). En la Figura 40 se muestra la estructura de un programa

Figura 40.- *Estructura de un programa en C++*

```
#include <iostream.h>                    }declaración de librerías

int main(void){                          }función main
    cout<<°hola mundo°<<endl;            }secuencia de instrucciones
    return 0;                            }valor de retorno de la
                                         función
}                                        }llaves de cierre de la
                                         función
```

Fuente: (Osorio,2006)

Se muestran las instrucciones, tipos de variables, caracteres de control y algunos de sus operadores en la Figura 41

Figura 41.- *Comandos, tipos de variables, caracteres de control y operadores en C++*

Instrucciones			
and	and_eq	asm	auto
bitand	bitor	bool	break
case	match	char	class
compl.	const	const_cast	continue
default	delete	do	Double
dynamic_cast	else	enum	explicit
export	extern	false	float
for	friend	goto	if
inline	int	long	mutable
namespace	new	not	not_eq
operator	or	or_eq	private
protected	public	register	reinterpret_cast
return	short	signed	sizeof
static	static_cast	struct	switch
template	this	throw	true
try	typedef	typeid	typename
union	unsigned	using	virtual
void	volatile	wchar_t	while
xor	xor_eq		

Caracteres de control	Significado
\n	salto de línea
\t	tabulador horizontal
\v	tabulador vertical
\a	alerta sonora
\0	carácter nulo
\b	mueve el cursor hacia atrás

Tipo	Tamaño en bytes	Intervalo
short	2	-32768 a 32767
unsigned short	2	0 a 65535
long	4	-2147483648 a 2147483647
unsigned long	4	0 a 4294967295
int	dependiendo del compilador utilizado podría ser 2 o 4	-32768 a 32767
unsigned int	2 o 4	0 a 65535
float	4	1.17549535e-38 a 3.402823466e+38 con 8 cifras decimales
double	8	2.2250738585072014e-308 a 1.7976931348623158e+308 con 16 cifras decimales
long double	10	
char	1	-128 a 127
unsigned char	1	0 a 255
bool	1	Valor lógico o booleano que puede ser trae (cierto) o false (falso).

Operador	Significado
+	adición
-	sustracción
*	multiplicación
/	división
%	resto de la división entera

Operador	Significado
<	menor que
<=	menor o igual
>	mayor que
>=	mayor o igual
= =	igual
!=	desigual

Fuente: (Osorio,2006)

C++ soporta en sus librerías operaciones de:

- Entrada/salida de datos
- Múltiples estructuras de datos
- Modificación de formato
- Estructuras de control
- Manejo de arrays y cadenas
- Punteros
- Funciones como prototipos, paso de argumentos, valores de retorno, macros, recursividad
- Entrada y salida por archivos
- Contenedores e iteradores
- Operaciones de diagnóstico
- Algoritmos generales
- Soportes de programa

2.6. MIT App inventor

MIT App Inventor es un ambiente de programación visual e intuitivo que permite elaborar aplicaciones completamente funcionales para tablets y teléfonos inteligentes, este entorno de desarrollo ágil permite un desarrollo completo de una aplicación en menos de una hora, basada en programación por bloques difiere de la programación tradicional. El proyecto MIT App Inventor busca propulsar la creación de software y nueva tecnología.

El desarrollo de nuevas aplicaciones está impulsado por un grupo de estudiantes y profesores de CSAIL (Laboratorio de Ciencias de la Computación e Inteligencia Artificial del Instituto de Tecnología de Massachusetts), liderado por Hal Abelson, ha generado todo un conjunto de equipos de desarrollo en torno a MIT App Inventor trabajando sobre una plataforma gratuita en línea, hoy con más de 25 millones de usuarios en el mundo.

Los programas de codificación basados en bloques buscan desde este entorno educativo el empoderamiento científico y creativo en torno a la inteligencia artificial. Orientado a niños y educadores, buscan conseguir un impacto social de valor inmenso en sus comunidades (MIT, sf)

2.7. Impresión 3D

Fabricación aditiva es el seudónimo técnico que abarca todas las tecnologías de impresión 3D, se trata de la elaboración de objetos tridimensionales por aportación de material en vez de sustracción. En impresión 3D, partiendo de un archivo digital (modelo 3D), se utilizan variados procesos aditivos en los que se aplican capas sucesivas de materia para establecer un objeto tangible. Existen varias formas de abordar el problema y cada tecnología tiene sus virtudes e inconvenientes.

La fabricación aditiva como herramienta es una tecnología revolucionaria ya que no tiene las limitaciones que presentan las tecnologías de fabricación tradicional, permitiendo emular cualquier cuerpo sin limitación alguna.

Una impresora 3D no admite diferencias en cuanto a las formas que debe imprimir, le cuesta el mismo esfuerzo fabricar un simple prisma de 6 caras que algo que desde un punto de vista humano se considera mucho más complejo como una parte anatómica, una pieza mecánica sumamente compleja o un simple cubo, por lo que se eliminan las limitaciones propias del uso de moldes o la fabricación usando herramientas de corte.

Una impresora 3D ofrece además la ventaja de poder producir objetos dentro de otros objetos, eliminando procesos posteriores de ensamblaje de piezas, garantizando una funcionalidad total.

Finalmente existen programas de modelado e impresión a través de servicios online, fablabs o makerspaces superando la necesidad de tener un equipo de impresión propio (Barrientos et Al, 2016).

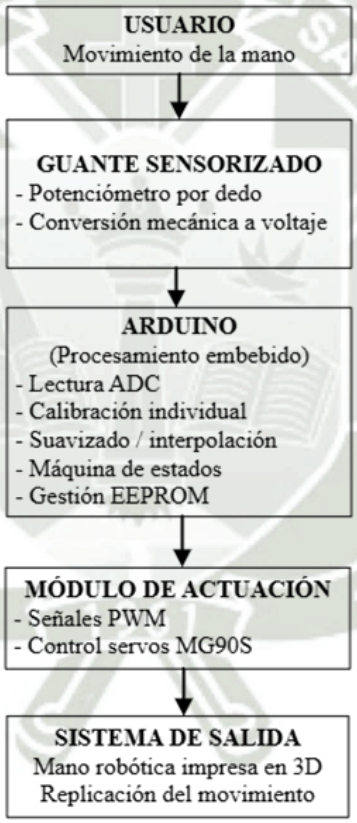
CAPITULO III

3. DESARROLLO DE INGENIERIA DEL PROYECTO

3.1.Solución propuesta

La Figura 42 presenta el diagrama conceptual del sistema propuesto. El flujo inicia con el movimiento del usuario, el cual es capturado por un guante sensorizado y procesado por un sistema embebido basado en Arduino. Finalmente, el sistema genera señales de control que permiten replicar el movimiento mediante una mano robótica impresa en 3D.

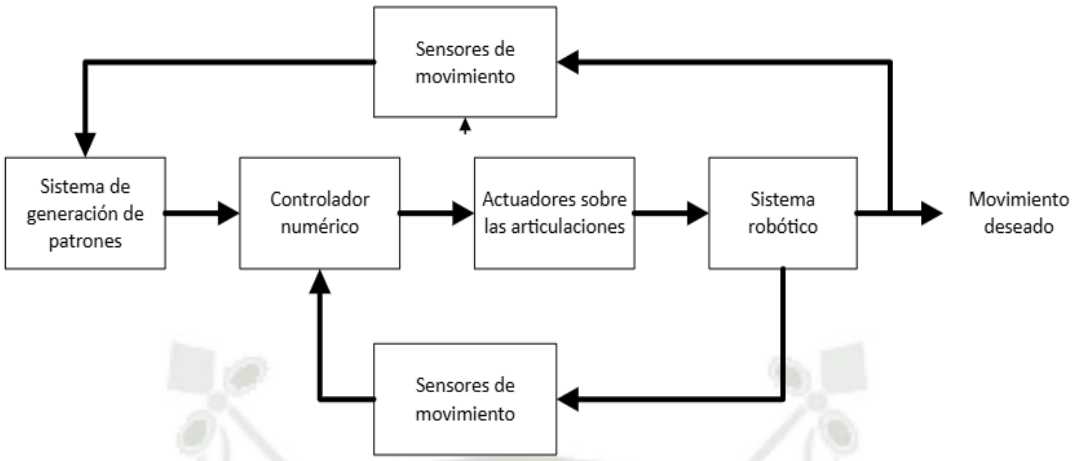
Figura 42.- *Diagrama conceptual del sistema a implementar*



Fuente: elaboración propia

En este trabajo de investigación se plantea el diseño e implementación de un servomecanismo con control numérico, cuyo diagrama de bloques se muestra en la Figura 43.

Figura 43.- *Diagrama de bloques de la mano robótica con control numérico*



Fuente: elaboración propia

Se distinguen los siguientes subsistemas:

- Sistema de generación de patrones: subsistema encargado de generar el patrón numérico correspondiente a la trayectoria deseada a partir de un proceso de aprendizaje automático.
- Controlador numérico: encargado de gobernar los actuadores para implementar el movimiento deseado en el sistema robótico, en este bloque se va a utilizar un microcontrolador o sistema embebido con múltiples entradas y salidas, sobre el que se va a programar las técnicas de control numérico diseñadas.
- Actuadores; en este bloque se consideran servomotores DC de precisión, se van a diseñar los circuitos de acondicionamiento necesarios.
- Sistema robótico: se va a diseñar e implementar un prototipo 3D con al menos cuatro grados de libertad, sobre el que se va a aplicar el control numérico multivariable. Asimismo, se va a modelar las trayectorias y movimientos en el mismo, con la precisión deseada las que van a ser utilizadas al momento de diseñar el software en el controlador.
- Sensores hacia el sistema de generación de patrones: utilizados para el aprendizaje de patrones de trayectorias
- Sensores hacia el controlador numérico: que detectan estados anómalas en el sistema robótico

3.2.Metodología de diseño

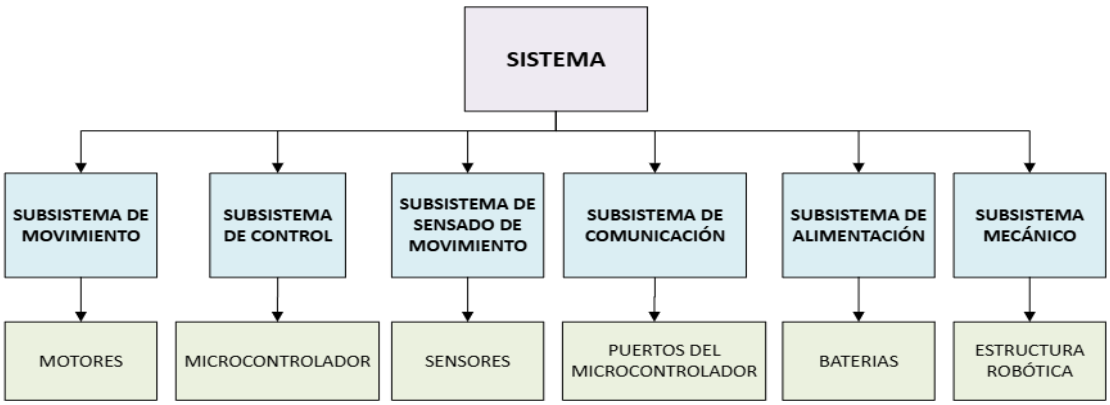
La metodología de diseño utilizada corresponde a la denominada Top Down, basada en el paradigma "divide y vencerás", debido a que diseñar una estructura robótica, requiere conocimiento e integración de diversas áreas de la ingeniería como la mecánica, la electrónica, la mecatrónica, la ingeniería de control y cuando el hardware está completo es la programación la vía de integración funcional; por lo expuesto se requiere el uso de una metodología estructurada que permite dividir el sistema general en subsistemas más pequeños hasta llegar a componentes específicos, luego en el proceso de implementación se seguirá el proceso inverso.

De manera general la metodología considera las siguientes etapas:

1. Definición de los requerimientos funcionales del sistema.
2. Descomposición del sistema general en subsistemas funcionales y la descomposición de cada subsistema en subsistemas más pequeños (si fuese necesario).
3. Selección de los componentes elementales para implementar los subsistemas del último nivel.
4. Integración de los subsistemas para conformar el sistema requerido (incorporando hardware y software)
5. Validación del cumplimiento de los requerimientos funcionales de diseño.

Aplicando este proceso al brazo robótico con control numérico planteado en esta tesis, se obtiene el diagrama de la metodología Top Down a seguir en el proceso de diseño, el que se muestra en la Figura 44

Figura 44.- *Metodología de diseño Top Down del brazo robótico con control numérico*



Fuente: elaboración propia

Se ha elegido esta metodología porque permite mejorar la planificación y documentación durante el proceso de diseño, asimismo posibilita la identificación de errores durante este proceso y permite una gestión modular del sistema; sin embargo, hay que ser muy cuidadoso en la definición de las especificaciones iniciales, ya que un error o ambigüedad podría desencadenar en un diseño erróneo y a necesidad de comenzar desde cero (Craig, 2005).

3.3.Definición de los requerimientos funcionales

El objetivo general del trabajo es implementar un brazo robótico, dotado de sensores y actuadores apropiados que le permitan al controlador numérico, generar patrones de movimiento de manera automática por repetición y luego reproducirlos de manera exacta y en número indeterminado de veces sin variaciones o ambigüedades.

El sistema entonces debe responder a los siguientes requerimientos funcionales:

1. La estructura robótica debe responder al modelamiento de una mano, con seis grados de libertad, con movimientos de flexión de cada dedo y prensión entre dos dedos, no se considera la articulación de muñeca, solo de dedos.

2. El sensado de los movimientos generan señales analógicas, por lo que deberá disponerse de conversores análogo a digitales, para enviar información al controlador programable.

3. El movimiento de los dedos será generado por un juego de motores de baja potencia, para permitir una fuente de alimentación independiente. Estos motores son entonces los actuadores que serán gobernados por el controlador.

4. El controlador, para posibilitar la portabilidad deberá ser un microcontrolador que tenga los buses estándares apropiados en número y tipo, para la comunicación con sensores y actuadores.

5. El controlador deberá tener el espacio de memoria apropiado para almacenar un mínimo de patrones de movimiento.

6. El sistema debe permitir la adquisición de patrones de movimiento, definidos como una secuencia de números, adquiridos automáticamente a partir de un proceso de entrenamiento.

7. El sistema debe tener un bajo consumo, para que la alimentación de este se realice a través de batería(s), dándole la independencia apropiada.

8. El montaje de los componentes debe permitir el mantenimiento y reemplazo de manera rápida y segura, asimismo debe respetar las recomendaciones en cuanto a disipación y potencia.

9. La estructura robótica debe ser implementada a partir de un diseño mecánico robusto, usando herramientas CAD, considerando la dinámica de la propia mano y el módulo de entrenamiento.

A partir de estas especificaciones funcionales del brazo robótico, se pueden, luego del análisis apropiado, formular las especificaciones técnicas que el sistema debe cumplir, las cuales se detallan según su naturaleza en la Tabla 4 mostrada a continuación.

Tabla 4.- *Especificaciones técnicas del brazo robótico*

Tipo	Especificaciones		
Mecánicas	Grados de libertad	6	
	Tipos de movimiento	Presión y Flexión	
	Peso	500 mgr	
	Dimensiones	10 x 30 x 40 cm	
Electrónicas	Microcontrolador	Arduino MEGA	Múltiples I/O
	Sensores	Tipo	Flexómetro
		Entrada	Desplazamiento
		Salida	Voltaje (0 a 5V)
		Acondicionador	No requiere
	Actuadores	Tipo	Servomotores
		Entrada	PWM
		Salida	Ángulo de rotación
		Alimentación	5V máximo
		Acondicionador	Driver
	Acondicionadores	Tipo	Escudo de sensores
		Entrada	0 a 5V
		Salida	0 a 5V
	Alimentación	5 a 12V / 10A	
Características Adicionales	Precisión definida por diseño CAD Fácil mantenimiento e implementación Capaz de adquirir y modelar numéricamente patrones de movimiento Capaz de almacenar diferentes patrones		

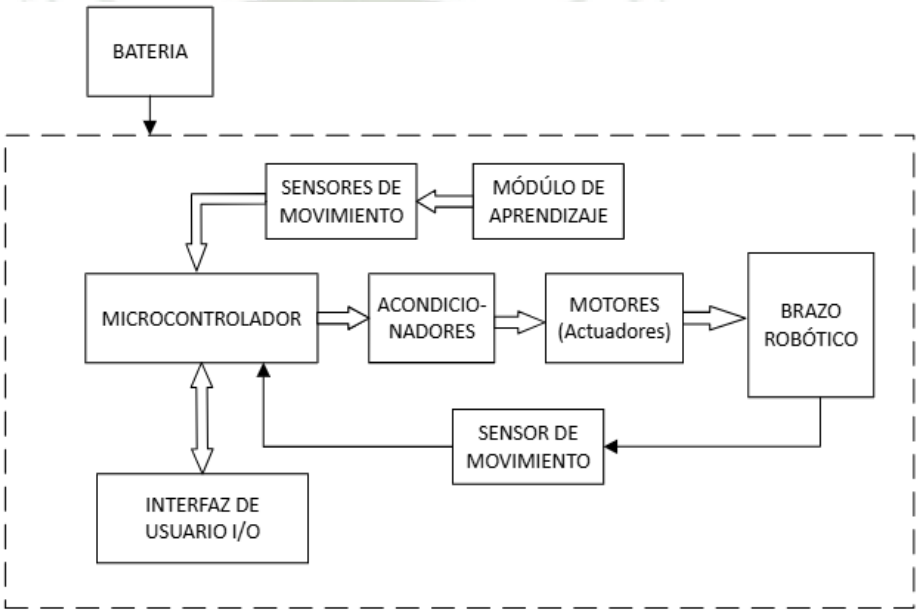
Fuente: elaboración propia

3.4.Descomposición de subsistemas funcionales

En la Figura 42 se planteó el diagrama de bloques del sistema, se hizo el análisis funcional utilizando la metodología mostrada en la Figura 44, dando origen a las especificaciones técnicas iniciales resumidas en la Tabla 4.

A partir de estas especificaciones técnicas se propone la estructura funcional mostrada en la Figura 45, que guiará el proceso de diseño que se mostrará en los siguientes puntos.

Figura 45.- *Diagrama final del brazo robótico con control numérico a ser implementado*



Fuente: elaboración propia

3.5.Microcontrolador

Para controlar el sistema se requiere un microcontrolador para implementar un sistema embebido. Se muestra en la Tabla 5 una comparación de los microcontroladores más usados, indicando las principales características, ventajas y desventajas.

Tabla 5.- *Comparación entre microcontroladores*

Característica	PIC16F877A	Arduino MEGA	Raspberry Pi
Arquitectura	Harvard RISC (8 bits)	ATmega2560 (8 bits)	ARM Cortex (64 bits)
Reloj	20 MHz	16 MHz	1.5 GHz
Alimentación	5V (2 – 5.5V)	5V (7 – 12V)	5V / 3V
Lenguaje de programación	Assembler, C	C/C++	Python, C/C++, Java
Entradas/Salidas (GPIO)	33 pines	54 pines / 16 AI	40 pines
Memoria Flash	14 KB	256 KB	Hasta 32 GB
EEPROM	256 bytes	4 KB	Según MicroSD
RAM	368 bytes	8 KB (SRAM)	2GB, 4GB, 8GB
Comunicaciones	USART, I2C, SPI	UARTs, I2C, SPI, USB	UART, I2C, SPI, Ethernet, WFi
Sistema operativo	No	No	Linux, Raspberry Pi
Ventajas	Bajo costo Bajo consumo Confiable	Fácil de programar Amplio soporte	Potente, multitarea Compatible con Linux
Desventajas	Programación compleja	Limitado en capacidad y soporte	Mayor consumo

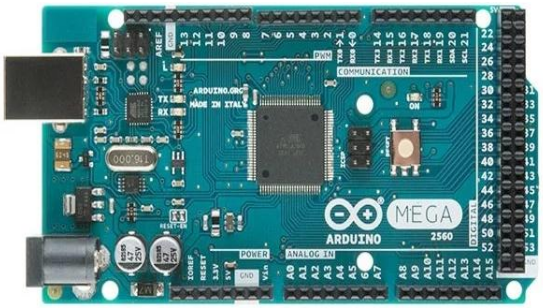
Fuente: elaboración propia

De la comparación mostrada en la Tabla 5 se seleccionó el microcontrolador Arduino Mega 2560 R3 cuyas especificaciones técnicas se presentan en la Hoja de Datos disponible en https://www.alldatasheet.com/datasheet-pdf/download/1425035/ETC/MEGA-2560.html y que se muestra en el Anexo A.

3.5.1 Microcontrolador Arduino MEGA 2560 R3

Este microcontrolador es un dispositivo desarrollado para sistemas embebidos que requieren una gran cantidad de recursos con relación a entradas/salidas, memoria e interfaces.

Figura 46.- *Microcontrolador Arduino Mega 2560*



Fuente: https://lab.bricogeek.com/

El microcontrolador ATmega2560(Atmel/Microchip), posee 54 entradas/salidas digitales (15 de ellas pueden usarse como PWM), 16 entradas analógicas, 4 UARTs (puertos seriales), un oscilador cerámico de 16 MHz, puerto USB, conector de alimentación, conector ICSP (In Circuit Serial Programming) para su programación, un botón de Reset, permite alimentación por el puerto USB, una fuente de voltaje o batería.

En cuanto a su programación esta se basa en C++ y se usa la interfaz IDE propia de la plataforma Arduino.

En la Tabla 6 se muestran las principales especificaciones técnicas proporcionadas por el fabricante, que permiten confirmar las capacidades del controlador para satisfacer los requerimientos funcionales planteados.

De la revisión de esta y por reconocimiento de las comunidades de desarrollo, podemos concluir que es un recurso que se presta a múltiples aplicaciones en robótica, ya que cuenta con múltiples módulos adicionales del propio fabricante, como módulos i/o, interfaces a PWM para el manejo de servomotores, entre otros.

Tabla 6.- *Especificaciones técnicas microcontrolador ATmega2560*

Microcontrolador	ATmega2560
Voltaje de operación	5V
Voltaje de entrada (recomendado)	7 – 12V
Voltaje de entrada (límite)	6 – 20V
Pines de E/S digitales	54 (de los cuales 15 proporcionan PWM)
Pines de entrada analógica	16
Corriente CC por pin de E/S	20 mA
Corriente CC para pin de 3,3V	50 mA
Memoria Flash	256 KB de los cuales 8 KB son utilizados por el gestor de arranque
Memoria RAM	8KB
EEPROM	4KB
Velocidad del reloj	16 MHz
LED integrado	13
Longitud	101,52 mm
Ancho	53,3 mm
Peso	37 gr

Fuente: https://store.arduino.cc/

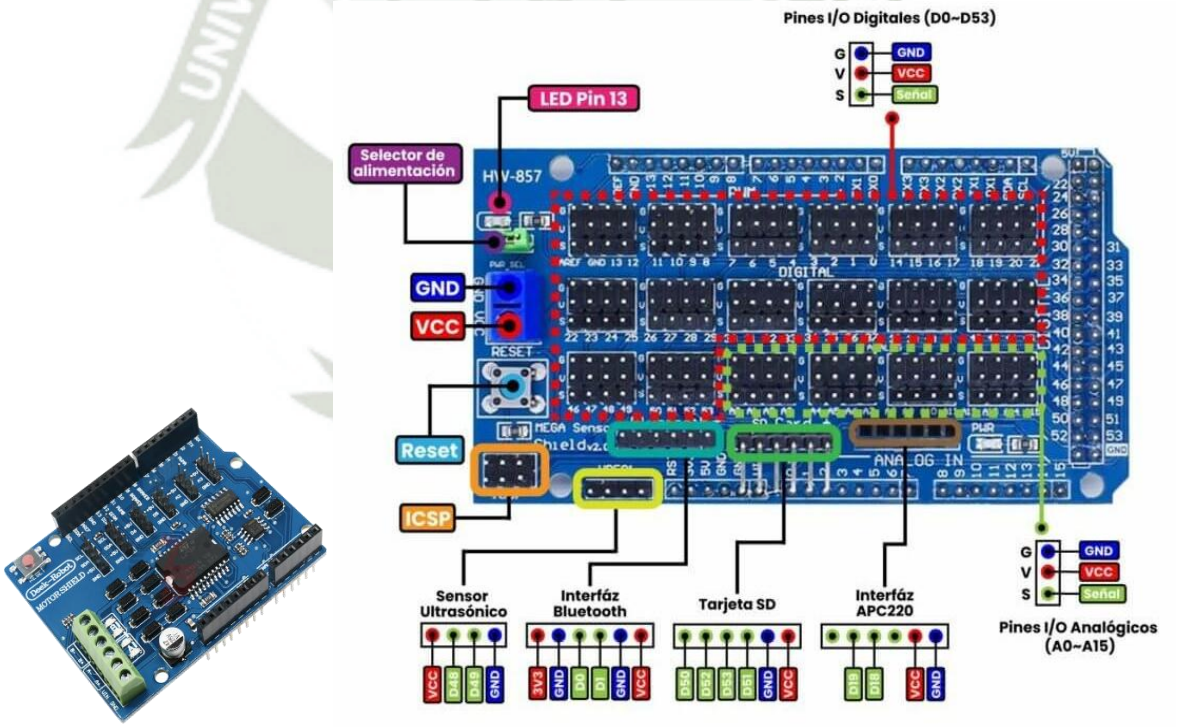
3.5.2 Escudo de sensores Deek-Robot

Dada la gran cantidad de sensores a utilizar, se ha previsto el uso del MICRO sensor shield DEEK-Robot que es una placa de expansión para ampliar el Arduino Mega con pines de sensor (IO, GND, VCC) para todas las entradas y salidas, una ventaja adicional es que la alimentación de los pines de sensor se puede conmutar entre el Arduino Mega o una fuente de alimentación externa. En resumen, se comporta como un escudo de datos.

Hay conectores para diferentes familias de semiconductores como ICSP, SDL/SCD, TTL y 3.3V. Posee un regulador de voltaje integrado con la finalidad de que solo se alimenten 5V a los pines del sensor, aunque la fuente externa pueda llegar hasta 20V, lo que garantiza la medición exacta.

En la Figura 47 se muestra la tarjeta y el PINOUT de la misma

Figura 47.- *MICRO sensor shield DEEK-Robot*



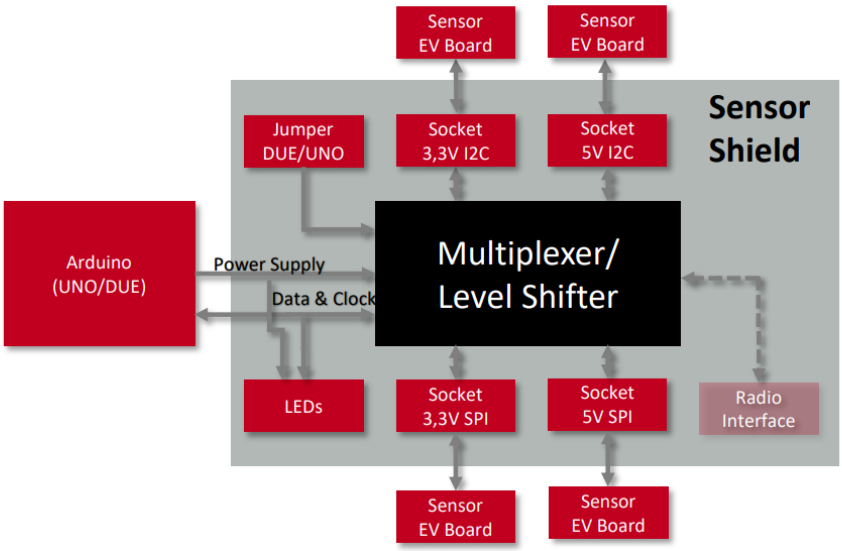
Fuente: https://www.galaxus.ch/en/s1/product/deek-robot-dk-micro-sensor-shield-v10-shield-electronics-modules-5999047
https://dcc-ex.com/reference/hardware/motorboards/deek-robot-motor-shield.html

La siguientes son las características técnicas más importantes:

- Tipo: Shield para Arduino Mega
- Voltaje de alimentación: 5V (tiene un regulador reductor de 3.3 V.)
- Led de PWR (encendido)
- leds programables
- Botón de RESET
- 14 bloques de pines PWM (Vcc, GND y Señal)
- bloques de pines Analógicos (Vcc, GND y Señal) a 2ª.
- Interfaz directa con Arduino Mega
- Interfaces: ICSP, sensor Ultrasónico URF, módulos bluetooth, módulos de tarjeta SD, PWM, módulos de radiofrecuencia APC220, led conectado a Pin 13, UART, I2C
- Salida de onda cuadrada (SQ) del RTC
- Protección contra escritura (WP)
- Detección de tarjeta (CD)

La información técnica se halla en la web del fabricante (ver enlace en la Figura 47), el diagrama de bloques se muestra en la Figura 48.

Figura 48.- *Diagrama de bloques del sensor shield V1.0*



Fuente: https://www.mouser.com/datasheet/2/445/2501000101291_SensorShield_for_Arduino_25010001012-2320397.pdf?utm_source=chatgpt.com

3.6 Sensores de movimiento

En sistemas robóticos, se requiere tener una medición precisa de la flexión con la finalidad de poder controlar el movimiento, monitorear la estructura y para la realimentación sensorial. En este caso, los sensores requeridos, deben realizar la medición sobre el movimiento de cada dedo, a partir del ángulo de flexión de cada articulación, a partir de esta edición el módulo de aprendizaje podrá generar patrones numéricos a ser enviados al brazo robótico para que este efectúe el movimiento deseado. Hay que indicar además que el número de sensores utilizados y su disposición espacial corresponde al número de articulaciones que deben ser sensadas, por ello se requiere el escudo de sensores descrito en el acápite 3.5.2.

3.6.1 Comparación y selección del sensor de flexión

Existen diversos sensores para implementar esta medición (Culha,2014)

- Sensores de flexión resistivos, los que varían su resistencia eléctrica al doblarse. Son económicos y fáciles de integrar, pero generan problemas de linealidad y durabilidad.
- Sensores piezorresistivos, en base a materiales que varían su resistencia con la deformación, sin embargo, son sensibles al ruido y la temperatura.
- Encoders rotativos, se instalan en las articulaciones para medir la rotación y calcular en base a ella la flexión, son precisos.
- Sensores de fibra óptica (FBG), detectan deformaciones por los cambios en la longitud de onda reflejada por rejillas de Bragg, ofrecen precisión, inmunidad a los campos electromagnéticos y un buen comportamiento dinámico.
- Sensores inerciales (IMU), miden aceleración y rotación, y se calcula la flexión a partir de estos datos. Compactos y versátiles, son poco precisos y requieren el procesamiento adicional.
- Sensores capacitivos, se producen cambios en la capacitancia cuando varía la distancia entre las placas. Muy sensibles a las perturbaciones ambientales.

La comparación se muestra en la Tabla 7.

Tabla 7.- 7*Comparación entre sensores de flexión*

Tipo de Sensor	Precisión	Complejidad para la medición	Costo	Aplicaciones
Flexión resistivos	Media	Baja	Bajo	Robots blandos Guantes hápticos
Piezoresistivos	Media	Media	Medio	Robots antropomórficos
Encoders rotativos	Alta	Media	Medio	Robótica industrial
Fibra óptica	Alta	Alta	Alto	Aplicaciones médicas Aeroespaciales
Inerciales	Media	Alta	Medio	Exoesqueletos Monitoreo corporal
Capacitivo	Alta	Media	Medio	Instrumentación precisa Robótica suave

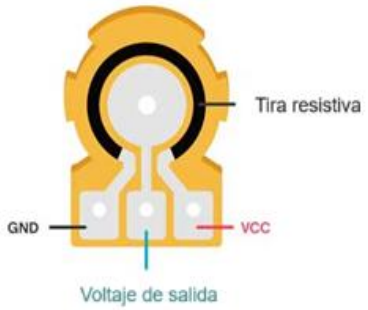
Fuente: elaboración propia

Se ha seleccionado en este proyecto un sensor de flexión resistivo, debido a su operación y diseño simple, su fácil integración en sistemas robóticos, bajo costo, compatibilidad con las entradas de los microcontroladores, tamaño compacto. Con las restricciones de cuidar las interferencias generadas por el polvo y limitar el rango de rotación en la medición a menos de 300º.

3.6.2 Sensor resistivo de flexión

Un sensor resistivo de flexión es una resistencia variable cuya impedancia varía al producirse una flexión, entonces la resistencia es directamente proporcional a la magnitud de este movimiento. En este caso se ha elegido un potenciómetro lineal, este dispositivo varía el valor de su resistencia eléctrica en función de la posición de un cursor móvil sobre una pista resistiva, como se muestra en la Figura 49.

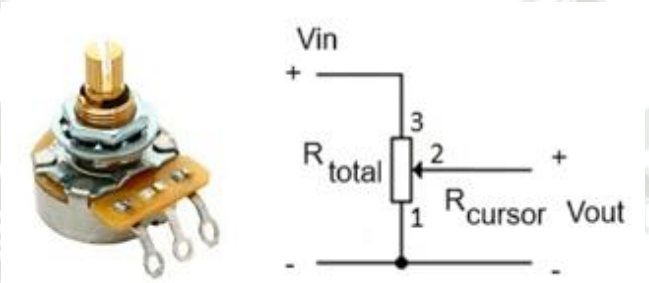
Figura 49.- *Estructura interna de un potenciómetro*



Fuente: https://www.digikey.es/es/supplier-centers/cui-devices

Los potenciómetros giratorios usan el movimiento angular generado por una perilla giratoria y un conectado a un elemento wiper. Este elemento se desliza sobre la tira resistiva generando los cambios en la resistencia. Para ser utilizado como sensor se aplica un voltaje constante entre los terminales extremos y la posición del cursor define el voltaje de salida el que será proporcional a la posición, actuando como un divisor de tensión como se ve en la Figura 50, el voltaje de salida se utilizará para determinar la posición angular o lineal del componente mecánico, en este cado dedo al que está acoplado.

Figura 50.- *Divisor de tensión de un potenciómetro*



Fuente: Elaboración propia

La relación entre el voltaje de salida $Vout$ y la posición del cursor se determina a partir de la ley de Ohm:

$$Vout = \frac{Rcursor}{Rtotal} Vin \tag{9}$$

Donde:

Vin: voltaje en los extremos del potenciómetro

$Rcursor$: resistencia entre el extremo y el cursor del potenciómetro

$Rtotal$: resistencia total del potenciómetro (entre extremos)

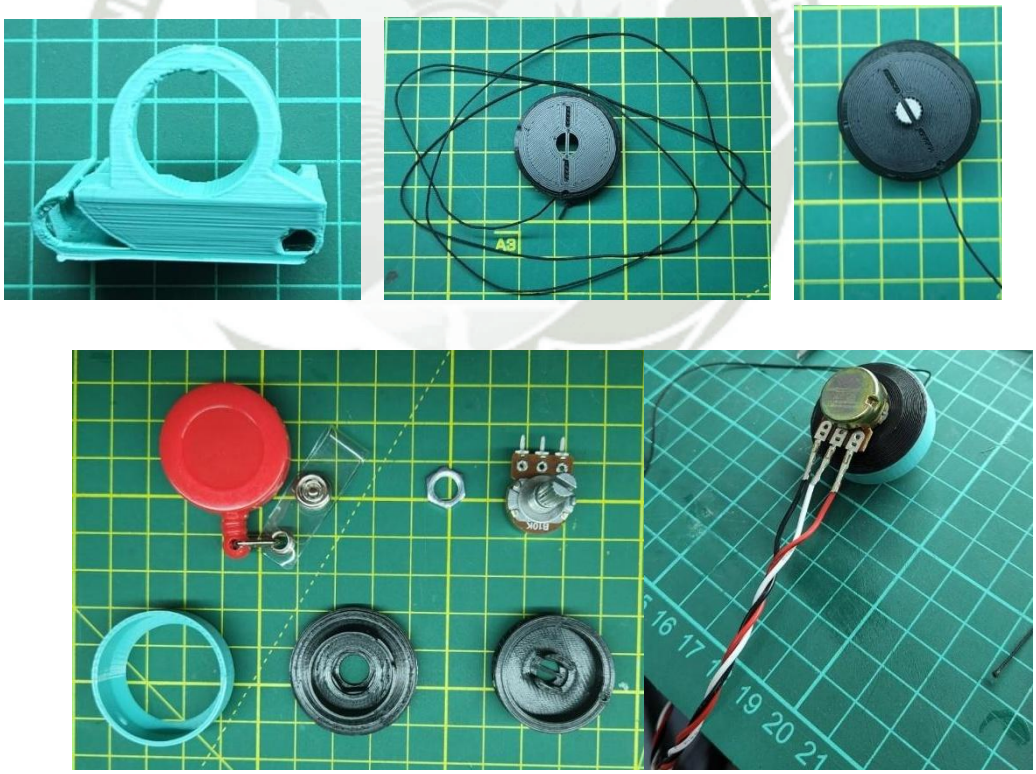
A partir de la expresión (9) se establece una relación directa y lineal entre la posición física del cursor y un valor de voltaje.

El potenciómetro lineal elegido WH148 Rotatorio 3 Pines tiene las siguientes características:

- Resistencia: 10KΩ.
- Potencia nominal máxima: 0.5 Watts
- Resolución: 0 a 5V
- Ruido de deslizamiento: < 47mV
- Coeficiente de temperatura:
- Vida mecánica: 10000 ciclos
- Ángulo máximo de linealidad: 300º
- Diámetro del eje: 6 mm.
- Longitud del eje: 10 mm.
- Diámetro base: 16 mm.

Se muestra en la Figura 51 el montaje previsto para el potenciómetro sobre un soporte ad-hoc diseñado e impreso en 3D para este proyecto, asimismo se muestra la conexión eléctrica del mismo.

Figura 51.- *Montaje del sensor resistivo de flexión*



Fuente: Elaboración propia

3.6.3 Integración al módulo de aprendizaje

En el módulo de aprendizaje, cuya descripción se hará más adelante, los potenciómetros se han integrado en las articulaciones de cada dedo, con la finalidad de medir su ángulo de flexión. Para ello se ha utilizado un hilo de nylon rígido, fijando uno de sus extremos sobre el mismo dedo y el otro extremo se ha fijado al cursor del potenciómetro, por lo que cualquier movimiento del dedo provoca una rotación determinada del potenciómetro lineal de 10KΩ., generando un cambio en su resistencia y con ello en el voltaje de salida. En la Figura 52 se muestra la integración de manera general, pudiéndose disponer de los potenciómetros en cualquier articulación.

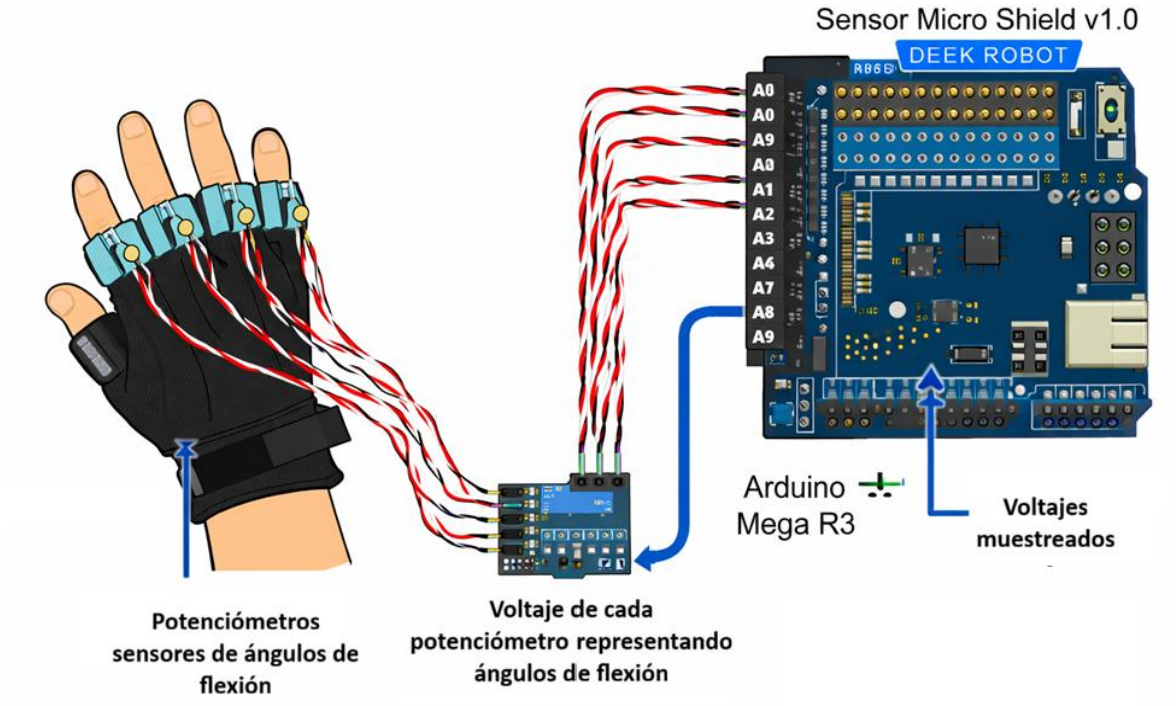
Figura 52.- *Integración del sensor resistivo de flexión en el guante*

Fuente: Elaboración propia

3.7 Guante sensorizado

El sistema, mostrado en la Figura 53, considera un guante instrumentado implementado para capturar el movimiento de la mano, ha sido diseñado para medir el ángulo de flexión de cada dedo de forma independiente utilizando para ello sensores analógicos, con la finalidad de muestrear de manera precisa el movimiento de la mano para implementar un proceso de reconocimiento de patrones que luego serán reproducidos por una mano robóticas (exoesqueleto).

Figura 53.- *Arquitectura de sensado cinemático*



Fuente: Elaboración propia

3.7.1 Interfaz humano-máquina (HMI)

La mano del entrenador debe introducirse en un guante ergonómico, el que cumple la función de interfaz mecánica entre el sistema de sensado y el movimiento natural de los dedos. La estructura del guante tiene una elasticidad y tamaño que debe garantizar:

- Alineación correcta de los sensores con las articulaciones.
- Confort durante la operación.
- Repetibilidad en las mediciones de flexión.

3.7.2 Sistema de sensado

En cada dedo se ha montado un potenciómetro rotacional WH148, fijado y tensado mediante soportes impresos en 3D a lo largo de cada dedo, que convierten el ángulo de flexión del dedo en un voltaje continuo directamente proporcional. Ello se debe a que cuando el dedo se flexiona, el eje del potenciómetro rota, lo que produce una variación proporcional de voltaje entre 0 y 5 voltios.

Es necesario indicar que cada potenciómetro es una señal analógica independiente, es decir existe un canal asociado a cada dedo en particular, con ello el sistema traduce el

movimiento biomecánico en una señal eléctrica cuantificable, que será luego digitalizada para su posterior procesamiento.

3.7.3 Sistema de adquisición y acondicionamiento de señales

Los voltajes generados por los potenciómetros son conducidos mediante cableado trenzado, considerando las posibles interferencias o ruidos que se pueden producir, debe considerarse:

- Posibles interferencias electromagnéticas.
- Ruidos generados en la propia señal analógica.
- Errores generados en el proceso de muestreo en los puertos del microcontrolador.

Esto se previene utilizando el Sensor Micro Shield v1.0 DEEK ROBOT, especialmente diseñado para aplicaciones robóticas, con la finalidad de:

1) Actuar como interfaz eléctrica entre los sensores de movimiento y el microcontrolador.

2) Distribuir a cada canal las tensiones de alimentación (5V y GND).

3) Proveer los puertos de entrada directa de las señales analógicas al microcontrolador, necesario en cuanto al número de entradas, el filtrado y acondicionamiento de estas señales analógicas.

3.8 Actuadores

3.8.1 Servos

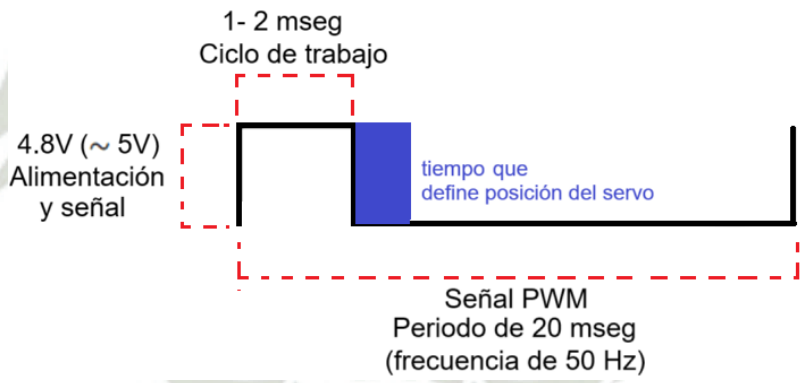
Se ha elegido el servo MG90S, que integra un servomotor, un decodificador PWM y los circuitos de interface en una sola unidad, para generar el movimiento de cada dedo en la mano robótica, debido a ser pequeño y de alta precisión, tal como su fabricante indica, está hecho para aplicaciones robóticas, al tener engranajes metálicos tiene un elevado torque. El control de movimiento es preciso y se integra con facilidad a los microcontroladores.

Figura 54.- *Servo MG90S*

Fuente: Data Sheet del MG90S

Como entrada de control se utiliza una señal PWM (Pulse Width Modulation) que se convierte en movimiento rotatorio de hasta 180 grados, dependiendo la posición del eje de la longitud del pulso de la señal PWM, como se ve en la Figura 55.

Figura 55.- *Señal PWM para el control del servo MG90S*



Fuente: Elaboración propia

El ciclo de trabajo responde a la Tabla 8:

Tabla 8.- *Control de posición del MG90S*

Acción	Ancho de pulso (mseg)
180 grados	1.00
90 grados	1.25
Posición central (0 grados)	1.50
-90 grados	1.75
-180 grados	2.00

Fuente: elaboración propia

Las características del MG90S se muestran en la Tabla 9:

Tabla 9.- *Características del MG90S*

Especificación	Valor
Rango de movimiento	0 – 180 grados
Voltaje de operación	4.8 – 6V
Par de torsión	1.8 kg/cm a 4.8V
Par máximo de torsión	2.2 kg/cm a 6V
Velocidad	0.1 s/60° a 4.8V
Peso	13.4 gramos
Engranajes metálicos	Mayor resistencia y durabilidad
Conector universal	1 – marrón (tierra) 2 – rojo (alimentación) 3 – naranja (entrada PWM)

Fuente: elaboración propia

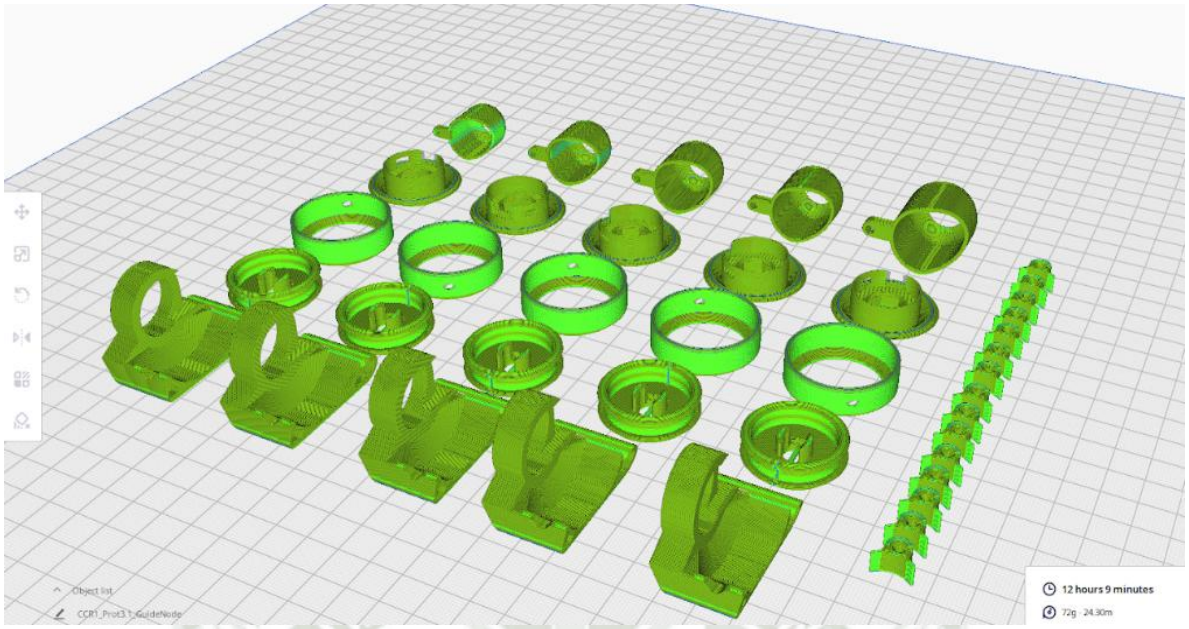
3.8. Mano robótica

La mano robótica desarrollada e implementada, en forma de exoesqueleto, ha sido diseñado utilizando herramientas de desarrollo 3D, en este caso Solid Work, software de diseño asistido por computadora (CAD) paramétrico, para modelar las piezas, crear ensamblajes y generar planos, permitiendo el diseño y fabricación de los sólidos requeridos, tomando como punto de partida múltiples diseños de manos ya existentes en la web.

Con la finalidad de reproducir los movimientos y patrones sensados y definidos por el módulo de aprendizaje, debe presentar el mismo número de grados de libertad, en este caso cinco, asociados al ángulo de flexión de cada uno de los dedos, las dimensiones de las partes mantienen la proporcionalidad de la mano humana de un adulto promedio.

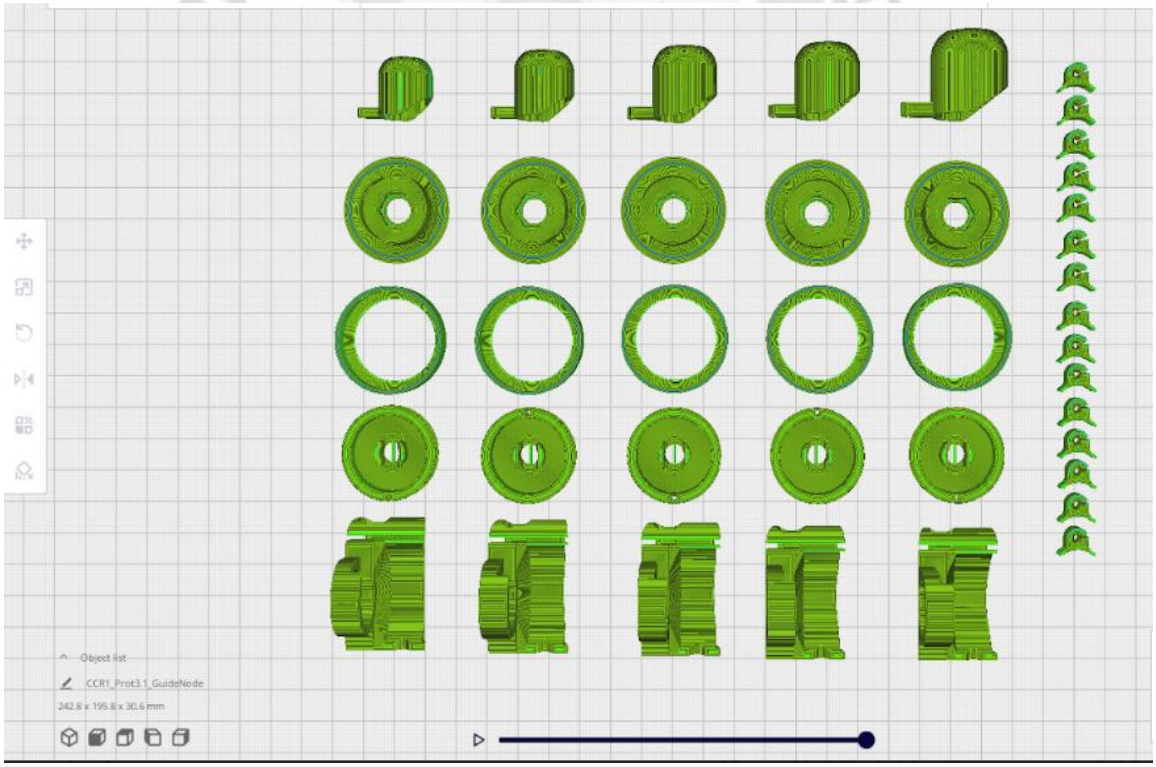
Se muestran en las Figuras 56 y 57 algunos de los planos de diseño de piezas requeridas para la construcción de la mano robótica, que serán impresas y luego ensambladas apropiadamente para reproducir el movimiento de la mano humana con la precisión deseada y requerida.

Figura 56.- *Vista en 3D de las piezas de la mano robótica*



Fuente: Elaboración propia

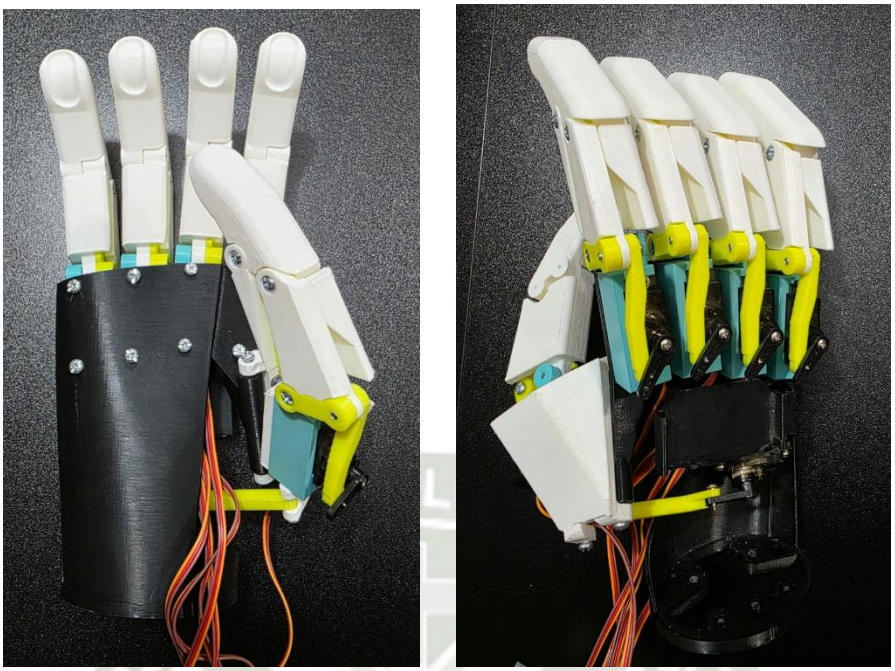
Figura 57.- *Vista superior de las piezas de la mano robótica*



Fuente: Elaboración propia

En la Figura 58 se muestra la mano robótica con el montaje completo

Figura 58.- *Exoesqueleto de la mano robótica*



Fuente: Elaboración propia

3.8.1 Integración del servomotor a la mano robótica

En la Figura 59 se muestra la integración o montaje de los servos en cada uno de los dedos de la mano robótica, de manera análoga a la disposición de los sensores de flexión en el guante.

Figura 59.- *Integración de los servos a la mano robótica*

Fuente: Elaboración propia

3.9 Unidad de procesamiento

3.9.1 Lectura de valores

El Arduino Mega R3 constituye la unidad central de adquisición de datos del sistema. Implementa las siguientes funcionalidades:

1) Muestreo periódico de los voltajes analógicos presentes en cada uno de sus puertos de entrada, mediante el convertidor A/D interno de 10 bits (2^{10}=1024 niveles).

2) Digitalización del voltaje leído, el código binario generado es proporcional al ángulo de flexión del dedo. Considerando que el Arduino Mega utiliza como referencia $V_{ref} = 5V$, siendo V_i la tensión presente en uno de los puertos (puerto o dedo i), se digitaliza al valor N_i:

$$N_i = \frac{V_i}{V_{ref}} 1023 \qquad N_i \in [0,1023] \tag{10}$$

La relación con el ángulo de flexión del dedo será:

$$\theta_i = \alpha_i V_i + b_i \tag{11}$$

Las constantes α_i y b_i pueden ser calibradas individualmente por software manteniendo la mano totalmente estirada (ángulo $\theta_i = 0$) y totalmente flexionada (ángulo $\theta_i = \theta_{i\max}$). También se tiene la posibilidad de aplicar filtrado digital si en el medio se encontrase excesiva interferencia electromagnética.

3) Se trabaja con canales analógicos simultáneos, lo que permite el sensado concurrente de todos los dedos. El conjunto de datos adquiridos conforma el vector de ángulos de flexión de todos y cada uno de los dedos, modelando el estado de la mano humana en un instante de tiempo. Este vector será utilizado para:

- Control de una mano robótica (modo repetición).
- Entrenamiento para aprendizaje automático de patrones (modo aprendizaje).

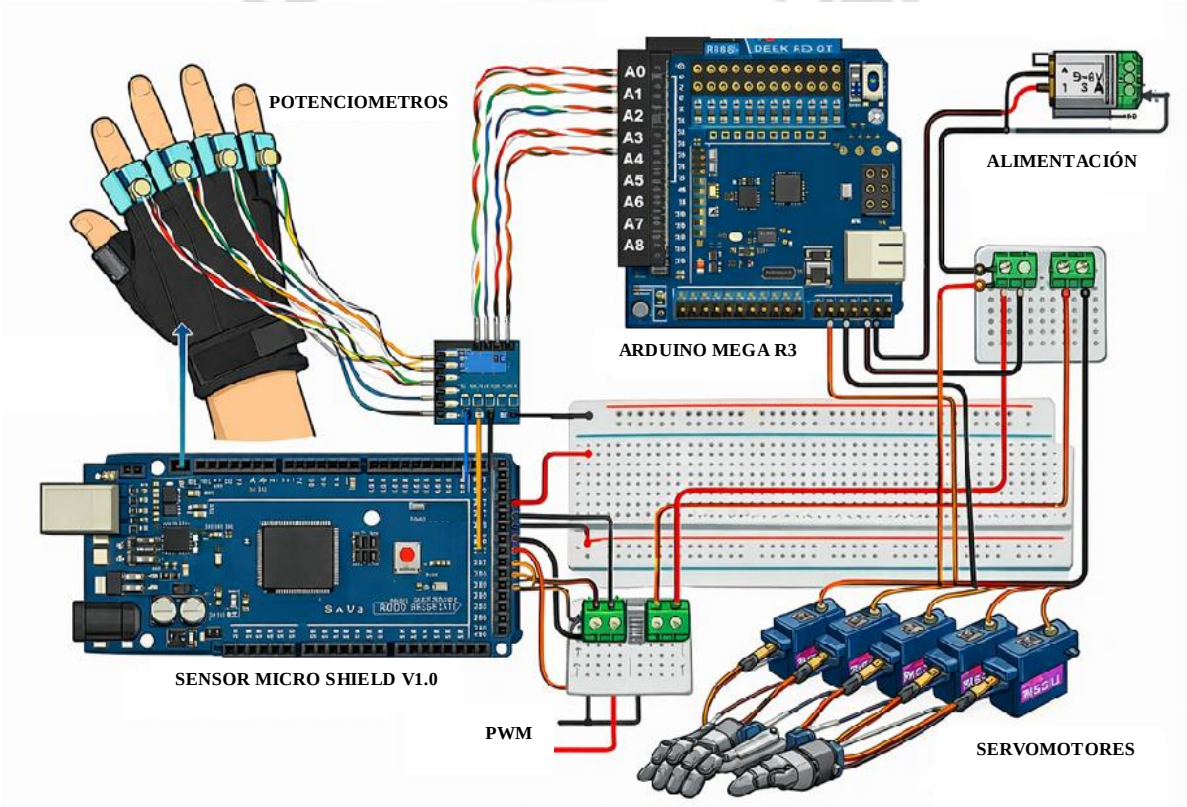
3.9.2 Reproducción de patrones

Una vez implementado la mano robótica se requiere la programación para implementar la reproducción de los movimientos deseados. Los patrones se entienden como una secuencia de seis valores que se envían en tiempo real o que han sido almacenados en un dataset y que debe ser leído posición a posición. Cada uno de estos seis valores representa el ángulo de flexión deseado de cada uno de los cinco dedos y debe ser enviado por el Arduino Mega en el valor PWM al servomotor correspondiente.

3.9. Sistema integrado

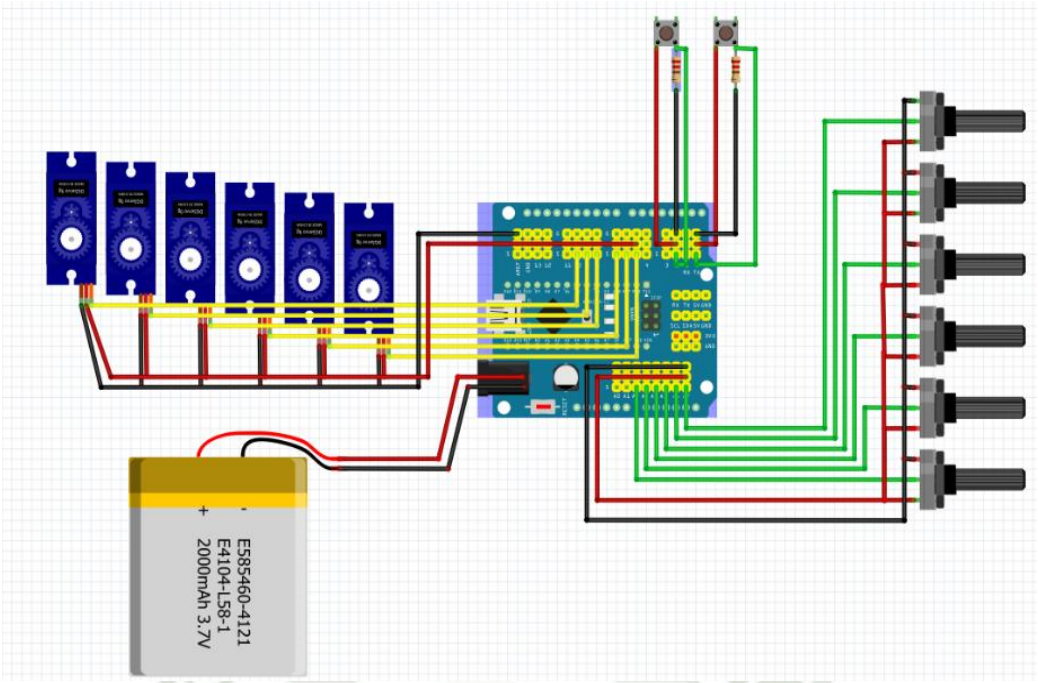
En la Figura 60 se muestra el sistema final implementado de manera física, mientras que en la Figura 61 se muestra el esquema circuital.

Figura 60.- *Sistema físico final*



Fuente: Elaboración propia

Figura 61.- *Diagrama circuital*



Fuente: Elaboración propia

3.10. Cálculo del consumo eléctrico del sistema

Para calcular el consumo del sistema, se identifican los voltajes de operación de cada uno de los componentes electrónicos que vamos a utilizar y el consumo de corriente máximo para la elección de la batería adecuada.

Tabla 10.- *Consumo de corriente del sistema*

Componente	Condición	Corriente aproximada
Servomotor MG90S	En reposo	100 – 150 mA
	En movimiento sin carga	200 – 300 mA
	Bajo carga	500 – 700 mA
	Corriente pico	Hasta 800 – 900 mA
Arduino ATmega 2560	Operación normal	70 – 90 mA
	Con periféricos activos	Hasta 100 mA
Pantalla LCD 16x2	Lógica LCD	1 – 2 mA
	Retroiluminación	15 – 25 mA
Potenciómetros	Divisores resistivos	5 mA
Botones	Lógica digital	5 mA

Fuente: elaboración propia

Estos valores son teóricos, pero servirán para validar la selección de la batería a ser utilizada para la alimentación del sistema

Para un diseño seguro se debe considerar el peor caso para el correcto dimensionamiento de la fuente. En el caso de los servomotores consideramos un escenario extremo en el que todos se encuentren bajo carga simultánea, por lo que tendríamos un consumo de 4.8 A (6 servos x 800 mA). Para el Arduino Mega consideramos el máximo consumo posible, 100 mA, al igual que para la pantalla LCD, 25 mA. Dando un consumo máximo de 4.94 A. Se recomienda agregar un 20 – 30% de margen de seguridad con lo que tenemos una corriente recomendada de mínimo 6A.

El dimensionamiento de la fuente considerando el consumo máximo garantiza la estabilidad del sistema incluso en condiciones de carga elevada, asegurando un funcionamiento confiable y seguro.

Los valores típicos de consumo de corriente del circuito, es decir en una operación normal lo vemos en la Tabla 11.

Tabla 11.- *Consumo típico de corriente del sistema*

Componente	Corriente
Servomotor MG90S	1.80 A
Arduino ATmega 2560	0.10 A
Pantalla LCD 16x2	0.025 A
Otros	0.01 A
Total	1.94 A

Fuente: elaboración propia

La alimentación para la parte de potencia (servomotores) de la de control (Arduino y periféricos) es separada para evitar caídas de tensión, reinicios inesperados del microcontrolador y daños en los reguladores internos.

3.10.1 Alimentación batería de polímero de litio (LiPo)

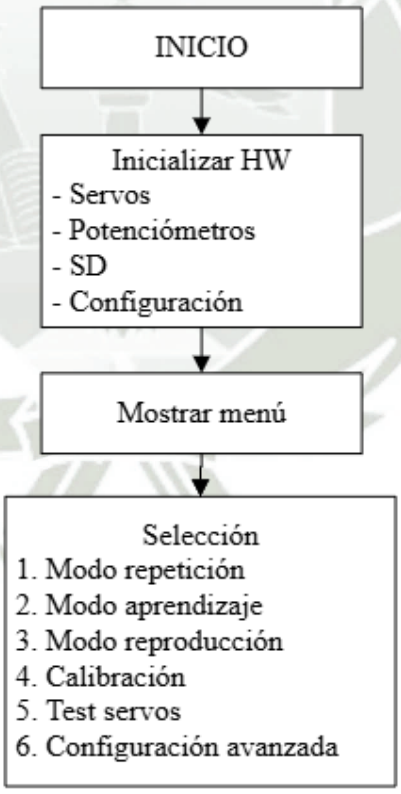
La batería seleccionada para proveer la alimentación al sistema es una LiPo (polímero de litio) recargable de 2 celdas (2S1P) de 1300mAh. Esta batería provee un voltaje máximo de 8.4V cuando está con carga al 100% y un voltaje mínimo de 7.4V, cuando la carga se reduce al 30%. Sus características son:

- Potencia: 1300mAh
- Voltaje: 7.4V a 8.4V
- Tasa de descarga: 25C
- Peso: 45 gramos
- Dimensiones: 42x31x18 mm
- Conector: JST
- Tiempo de uso por carga: 30 minutos (determinado experimentalmente)

3.11. Modos de operación del sistema

El software fue implementado como una máquina de estados finitos, donde cada modo de operación representa un estado independiente, facilitando la legibilidad, depuración y escalabilidad del sistema.

Figura 62.- *Máquina de estados del sistema*



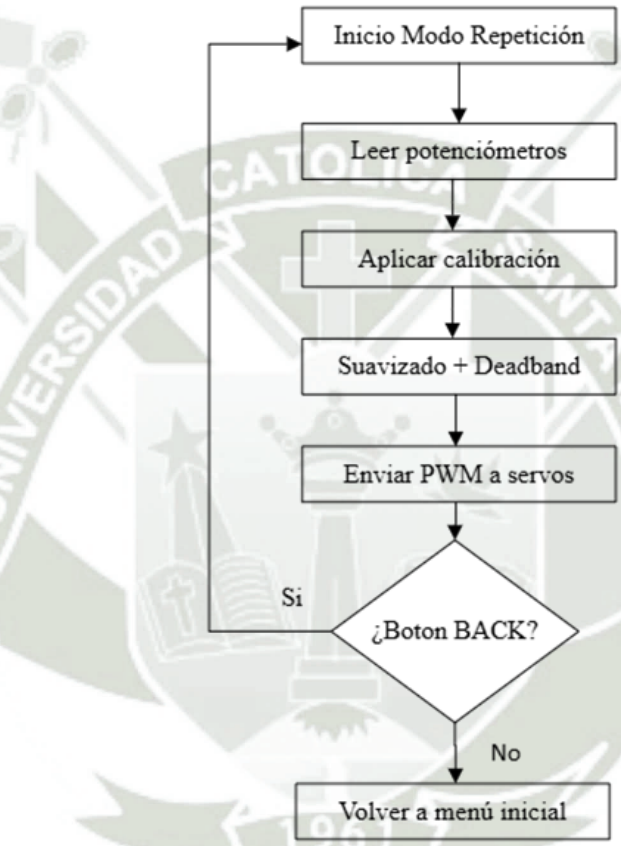
Fuente: Elaboración propia

3.11.1. Modo repetición

En este modo, la mano robótica replica en tiempo real los movimientos del usuario.

Se aplicó un algoritmo de suavizado incremental con banda muerta, reduciendo vibraciones mecánicas y ruido eléctrico.

Figura 63.- *Flujo de ejecución del modo repetición*

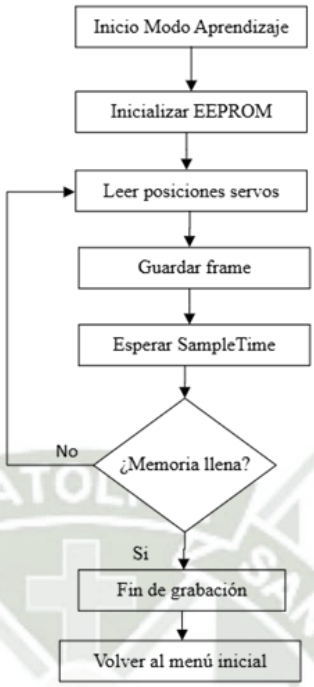


Fuente: Elaboración propia

3.11.2. Modo aprendizaje

Permite almacenar secuencias de movimiento en memoria EEPROM, registrando los ángulos de cada servo en intervalos temporales definidos por el usuario.

Figura 64.- *Flujo de ejecución del modo aprendizaje*

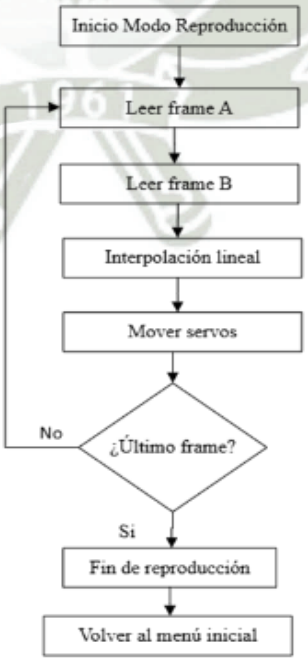


Fuente: Elaboración propia

3.11.3. Modo reproducción

Las secuencias almacenadas son reproducidas aplicando interpolación lineal entre cuadros consecutivos, generando trayectorias angulares continuas y visualmente más naturales.

Figura 65.- *Flujo de ejecución del modo reproducción*

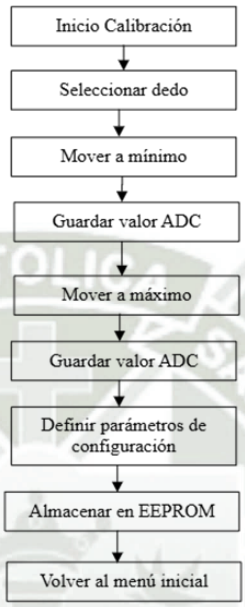


Fuente: Elaboración propia

3.11.4. Modo de calibración

Permite definir los rangos mínimos y máximos de movimiento de cada dedo, asegurando correspondencia precisa entre el movimiento humano y el robótico.

Figura 66.- *Flujo de ejecución del modo de calibración*



Fuente: Elaboración propia

3.11.5. Modo de test de servomotores

Se implementó un modo de diagnóstico que verifica el correcto funcionamiento de cada servomotor de forma individual, facilitando tareas de mantenimiento y detección de fallas.

CAPITULO IV

4. PRUEBAS Y RESULTADOS

El presente trabajo se desarrolló bajo un enfoque experimental–aplicado, orientado al diseño e implementación de un sistema mecatrónico de control embebido capaz de replicar movimientos humanos mediante una mano robótica impresa en 3D.

La metodología seguida se basa en el diseño iterativo, donde cada subsistema fue desarrollado, probado y optimizado de forma independiente antes de su integración final, garantizando estabilidad, escalabilidad y confiabilidad del sistema.

4.1 Arquitectura implementada

El sistema está compuesto por cuatro bloques funcionales principales:

a) Módulo de adquisición de datos o sensores de movimiento

b) Módulo de procesamiento embebido o microcontrolador

c) Módulo de actuación

d) Módulo de interacción hombre–máquina (HMI)

Cada bloque se comunica mediante señales eléctricas normalizadas, permitiendo una integración modular y mantenimiento sencillo.

4.1.1 Módulo de sensores de movimiento

El módulo de adquisición de datos o sensores de movimiento está a cargo de un guante instrumentado, en el que se ha incorporado en cada dedo un potenciómetro de desplazamiento angular, acoplado mecánicamente mediante un sistema de cuerdas tensoras.

La variación angular de cada dedo del usuario se traduce en una señal analógica de voltaje, la cual es muestreada por los conversores analógico–digitales (ADC) del microcontrolador.

Con el fin de adaptarse a diferentes usuarios del guante, se implementó el procedimiento de calibración independiente para cada dedo, permitiendo definir valores mínimos y máximos

de los ángulos de flexión, estos valores definidos por software son almacenados en memoria no volátil EEPROM. Finalmente se dispuso de un filtrado digital de las señales para disminuir la presencia de ruidos y perturbaciones externas.

4.1.2 Módulo de procesamiento embebido o microcontrolador

El núcleo del sistema es un microcontrolador Arduino, encargado de la lectura de las señales analógicas, el procesamiento de datos, la calibración y filtrado, el manejo de los modos de operación y el control de los actuadores sobre la mano robótica.

El software programado se concibió como una máquina de estados finitos, fundamento del control numérico donde cada modo de operación representa un estado claramente definido del sistema (posición de los dedos de la mano).

El sistema cuenta con dos modos de operación:

a) Modo reproducir

En este modo, la mano robótica replica en tiempo real los movimientos que el usuario realiza a través de la mano insertada en el guante instrumentado. En este modo el sensado es sometido a un algoritmo de filtrado digital para el suavizado del movimiento, reduciendo vibraciones mecánicas y ruido eléctrico, así como la calibración independiente de cada dedo para asegurar la exactitud.

b) Modo aprender

En este modo se almacenan secuencias de movimiento en memoria EEPROM, registrando los ángulos de cada servo en intervalos temporales definidos por la frecuencia o tiempo de muestreo y el número de muestras recogidas, ambos valores están definidos por software. Luego las secuencias almacenadas son reproducidas como trayectorias angulares continuas y visualmente naturales, debido a que aquí también se dispone del filtrado digital, la calibración individual y la validación de la no saturación al definirse rangos de flexiones permitidos dentro de un máximo y un mínimo.

El sistema de manera intrínseca también permite dos modos adicionales, el modo de calibración al definir los rangos mínimos y máximos de movimiento de cada dedo. Asimismo, el modo de

test de servomotores que permite el funcionamiento mientras que cada servomotor esté funcionando correctamente de forma individual.

4.1.3 Módulo de actuación

La mano robótica utiliza servomotores MG90S, seleccionados por el equilibrio que presentan entre precisión, torque y tamaño compacto, cada servomotor es controlado mediante señales PWM generadas por el microcontrolador, permitiendo posicionamiento angular preciso entre 0° y 180°.

4.1.4 Interfaz hombre–máquina (HMI)

Al ser Arduino un sistema programable se pueden implementar configuraciones avanzadas sobre las siguientes características, que serán almacenadas en la EEPROM para la persistencia:

- Frecuencia de muestreo.
- Número de muestras por patrón.
- Suavidad de movimiento (coeficiente del filtro digital).
- Calibración (estados extremos).
- Ángulos de flexión máximo y mínimo.

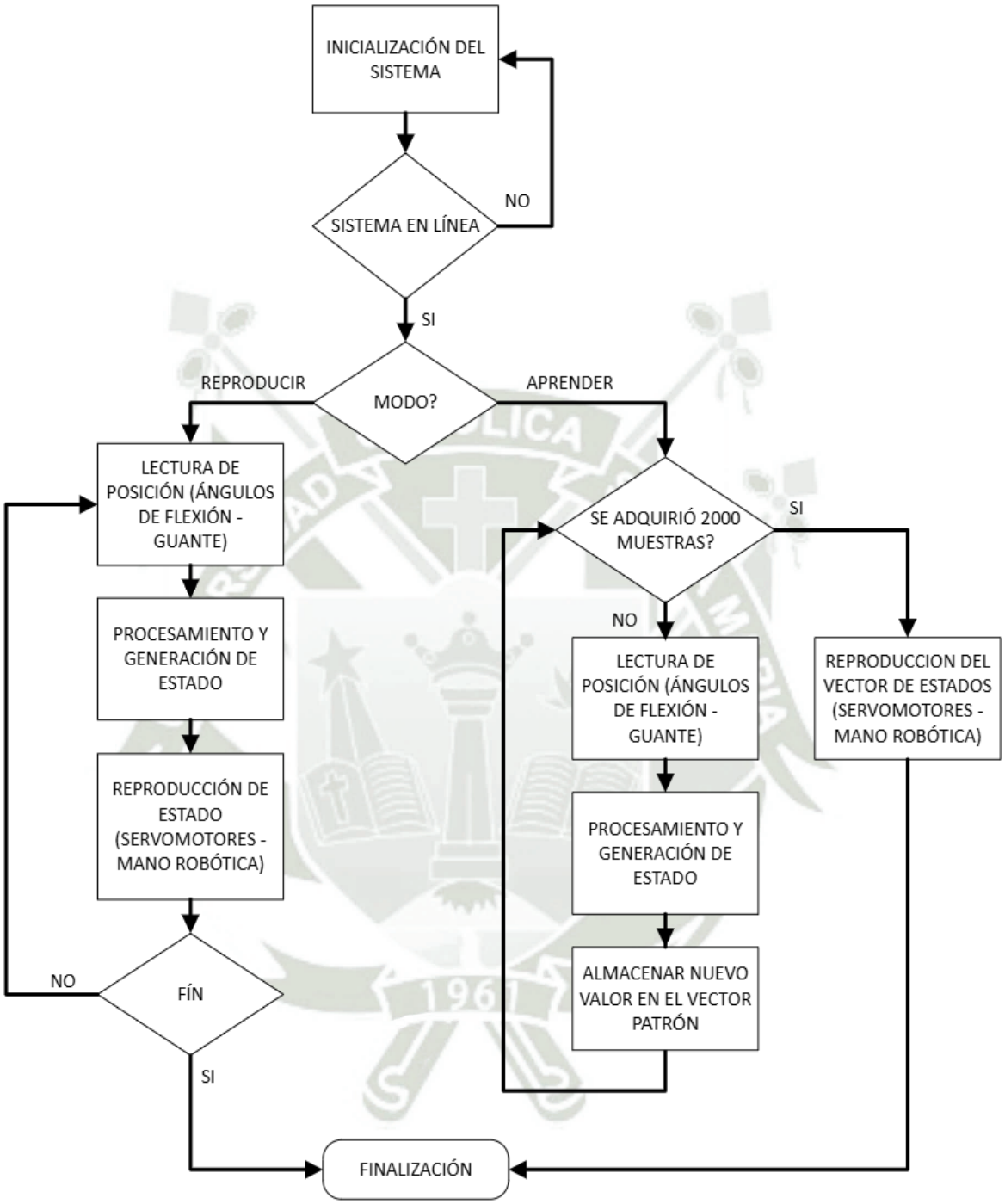
Adicionalmente el circuito permite por hardware

- Selección de modos a partir de un botón físico.
- Visualización del funcionamiento de los servomotores a partir de un juego de leds.

4.2 Funcionalidad conseguida

El sistema construido exhibe de manera detallada la funcionalidad descrita en la Figura 67.

Figura 67.- *Funcionalidad implementada*



Fuente: Elaboración propia

4.3 Validación experimental

La validación experimental se realizó mediante la ejecución repetida de movimientos predefinidos, dentro del rango propio de los cinco grados de libertad del sistema, utilizando el guante con la instrumentación implementada. Dentro de estos movimientos tenemos:

a) cierre de mano

b) apertura de mano

c) pinza

Los patrones generados fueron reproducidos por la mano robótica en los dos modos de trabajo considerados: repetición y aprendizaje. Se realizó la comparación entre el movimiento deseado y el reproducido, determinando errores menores al 5% de las flexiones angulares de los dedos (precisión del seguimiento)..

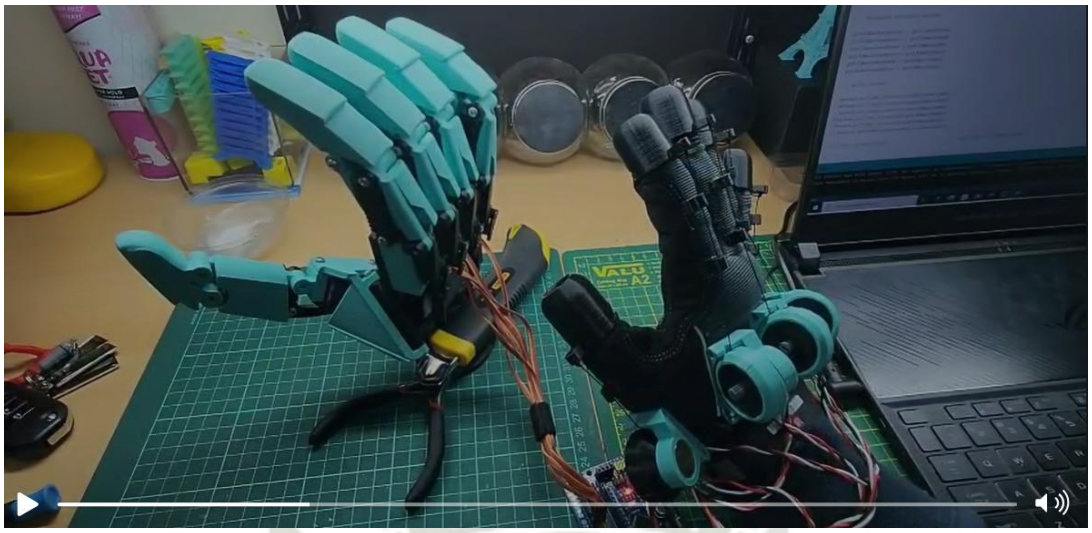
Los resultados muestran una alta repetibilidad del patrón, lo que confirma que el sistema es capaz de capturar y reproducir de manera fiable el movimiento humano.

La validación se realizó mediante pruebas funcionales y observación directa del comportamiento del sistema, evaluando:

- Precisión de seguimiento
- Suavidad del movimiento
- Repetibilidad
- Estabilidad mecánica

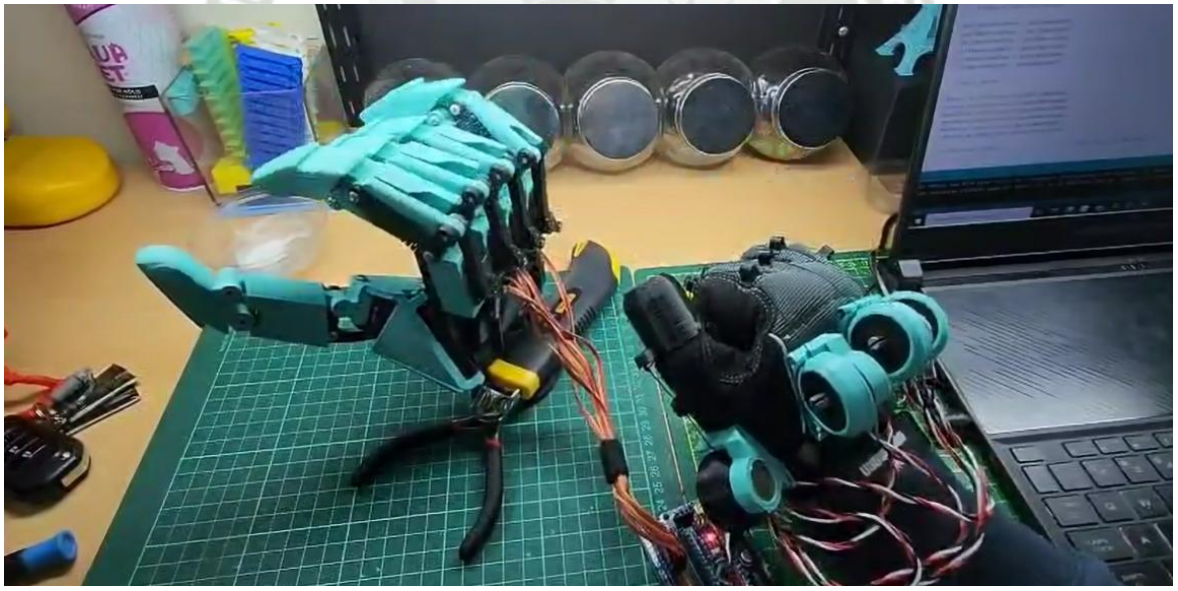
Se muestran a continuación algunas evidencias de las pruebas realizadas:

Figura 68.- *Apertura de mano*



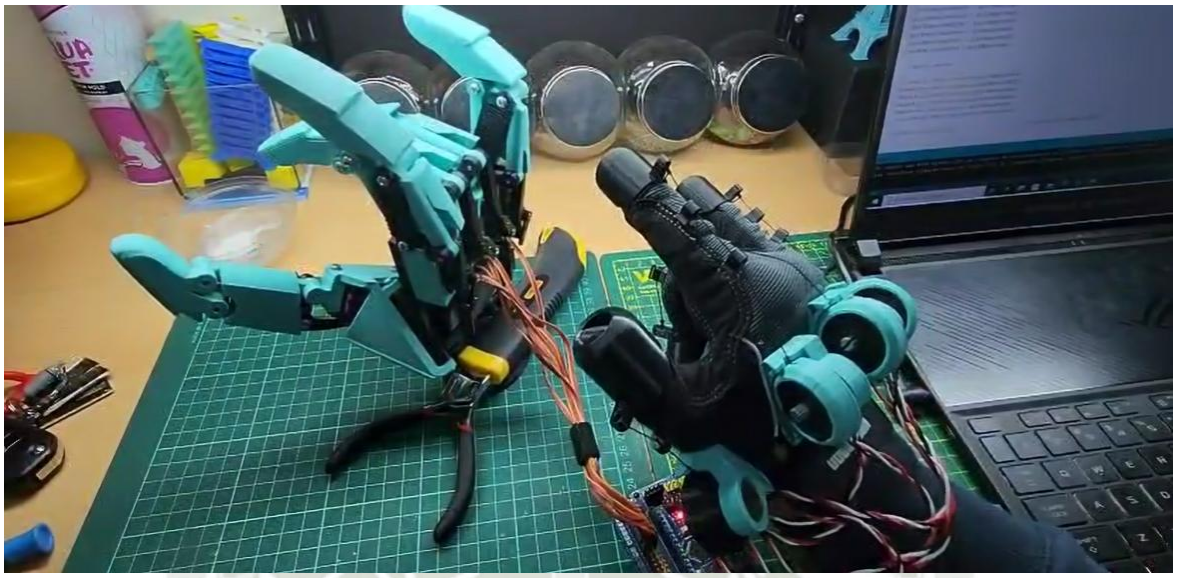
Fuente: Elaboración propia

Figura 69.- *Cierre de mano*



Fuente: Elaboración propia

Figura 70.- *Diferentes flexiones*



Fuente: Elaboración propia

Figura 71.- *Pinza*

Fuente: Elaboración propia

4.4 Resumen económico

Para la implementación del proyecto se ha incurrido en los siguientes costos de implementación referidos en Nuevos Soles (S/.)

Tabla 12.- *Costo de implementación*

No Ítem	Componente	Descripción	Cantidad	Precio Unitario	Sub-totales
1	Led	5mm	5	0.20	1.00
2	Bateria	LiPo 7.4v 1300mAh	1	50.00	50.00
3	Resistencia	Varios valores	10	0.30	3.00
4	Switch	12mm	1	2.00	2.00
5	Potenciómetros lineales	3 pines	5	4.00	20.00
6	Guante	Guante aislante y elástico	1	15.00	15.00
7	Arduino	Mega	1	30.00	30.00
8	Sensor Shield	V1.0	1	18.00	18.00
9	Servomotores	MG90S	5	20.00	100.00
10	Contactos		2	2.00	4.00
11	Impresión 3D	Piezas mano y guante	50	5.00	250.00
12	Otros	Software, material de consulta			300.00
TOTAL					775.00

Fuente: elaboración propia

El presupuesto corresponde a costos de distribuidores locales.

CONCLUSIONES

1. Se diseñó un controlador numérico para manejar un sistema robótico (mano) de cinco grados de libertad utilizando técnicas de control de estados finitos.

2. Se modelo numéricamente los movimientos propios de una mano en base a los ángulos de flexión de cada uno de los dedos, utilizando un guante sensorizado.

3. Se diseñó algoritmos requeridos para implementar la técnica de control y el manejo del sistema, que incluyeron el filtrado digital para generar un movimiento suave de la mano, eliminando ruidos en el proceso de sensado. La calibración individual de cada dedo para asegurar la exactitud requerida y la consideración de la saturación para la confiabilidad y repetibilidad del proceso, al momento de generar los patrones de movimiento.

4. El uso de un microcontrolador a partir de especificaciones funcionales bien definidas en el proceso de diseño asegura el desarrollo de un prototipo apropiado y reconfigurable.

5. Se ha diseñado y fabricado las piezas requeridas por el sistema de sensado y la mano robótica utilizando los fundamentos y herramientas propias del diseño asistido por computadora, lo que ha permitido la construcción con un acople perfecto de las diferentes piezas, lo que asegura la robustez y la suavidad del movimiento.

6. Los sensores seleccionados mantienen una relación lineal entre los ángulos de flexión de cada dedo y el valor eléctrico que ingresa al microcontrolador. Esta relación lineal requiere dos parámetros o constante para su definición, las mismas fueron determinadas por el proceso de calibración en las dos posiciones extremas consideradas para cada dedo en el guante. Este proceso asegura la exactitud en el proceso de medición.

7. Los servomotores son los actuadores por excelencia en el campo de la robótica, pero es importante que la generación PWM esté integrada en este servomotor para mejorar la modularidad del sistema.

8. El sistema final implementado responde con propiedad a los requisitos funcionales de diseño planteados en el acápite 3.3 y que son el punto de partida en el proceso de diseño en ingeniería.

9. El correcto funcionamiento del sistema es parte de la integración de software y hardware.

10. El control numérico es fundamental en sistemas que tienen un comportamiento que requiere por sus aplicaciones garantizar la exactitud, versatilidad y reproducibilidad del funcionamiento.

RECOMENDACIONES

1. Es posible el uso de técnicas de inteligencia artificial como el reconocimiento de patrones, redes neuronales y machine learning para potenciar el manejo de patrones en el contexto de aplicaciones en ambientes con mayores interferencias o necesidad de exactitud.

2. Es posible aumentar el número de grados de libertad bajo el mismo principio, disponiendo de un mayor número de sensores y actuadores en la medida en que la capacidad del microcontrolador lo permita.

3. Es posible evaluar la influencia del periodo de muestreo y el número de muestras recogidas, pero la cantidad de información requerida es directamente proporcional a la capacidad de procesamiento requerida y el almacenamiento necesario, por ende, la capacidad del microcontrolador.

4. Es posible incorporar técnicas de preprocesamiento a la data para suavizar aún más el movimiento, como la interpolación de tres puntos, entre otras.

5. Es importante este tipo de investigaciones para aplicaciones donde la interacción directa de los operarios es riesgosa o las acciones son sumamente repetitivas.

6. Es posible considerar mejores tecnológicas como la integración de sensores de fuerza, la comunicación inalámbrica, entre otras.

REFERENCIAS BIBLIOGRÁFICAS

Asea Brown Boveri. (s. f.). *Robótica ABB*. https://new.abb.com/products/robotics/

Barrientos, A., Peñín, L., Balaguer, C., & Aracil, R. (2016). *Fabricación digital: Introducción al modelado e impresión 3D* (pp. 8–9). Secretaría General Técnica.

Blog de Ingeniería. (s. f.). *Qué es la robótica y cuáles son sus principales usos*. Facultad de Ingeniería, Universidad ORT. https://fi.ort.edu.uy/blog/que-es-la-robotica-y-cuales-son-sus-usos

Craig, J. J. (2005). Introduction to robotics: Mechanics and control (3rd ed.). Pearson.

Corona, L., Abarca, G., & Mares, J. (2014). Sensores y actuadores: Aplicaciones con Arduino. Grupo Editorial Patria.

Culha, U., Nurzaman, S. G., Clemens, F., & Iida, F. (2015). Sensing skin for a robot hand using flexible optical fibers. IEEE Sensors Journal, 15(3), 1643–1655. https://doi.org/10.1109/JSEN.2014.2364277

De La Paz, C., Arteaga, C., Ruiz, D., & Méndez, M. (s. f.). Diseño y desarrollo de un prototipo de robot cuadrúpedo didáctico. Revista Aristas. http://revistaaristas.tij.uabc.mx/index.php/revista_aristas/article/view/78

Del Valle, T. (2018). Arquitectura y fabricación robotizada (Tesis de grado). Universidad Politécnica de Madrid.

FANUC. (s. f.). Robots industriales FANUC. https://www.fanuc.eu/es/es/robots

Fiestas Cobeñas, F. L., & Tesén Bardales, C. A. (2024). Desarrollo y control de una mano robótica con visión artificial para la emulación de movimientos del ser humano (Tesis de grado). Universidad de Piura. https://hdl.handle.net/11042/7104

Groover, M. P. (2014). Fundamentals of modern manufacturing: Materials, processes, and systems (5th ed.). Wiley.

Hossian, A., Merlino, H., & Alveal, E. (2020). Desarrollo e impacto del campo de la robótica en América Latina: Hacia una propuesta superadora en el contexto de la IV revolución

industrial. En Actas del II Simposio Argentino de Historia, Tecnologías e Informática y VI Simposio de Historia de la Informática en América Latina y el Caribe.

Instituto de Tecnología de Massachusetts. (s. f.). MIT App Inventor. https://appinventor.mit.edu/about-us

Kawasaki Heavy Industries. (s. f.). Kawasaki Robotics. https://robotics.kawasaki.com/en1/about/

Kawasaki Heavy Industries. (s. f.). Small-medium payload robots. https://robotics.kawasaki.com/en1/products/robots/small-medium-payloads/

KUKA. (s. f.). Sistemas de robot de KUKA. https://www.kuka.com/es-es/productos-servicios/sistemas-de-robot

Lamikiz Mentxaka, A. (s. f.). Fabricación asistida por ordenador: Control numérico. Universidad del País Vasco. https://www.ehu.eus/manufacturing/docencia/502_ca.pdf

MCI Educación. (s. f.). Interfaz del sensor flex con Arduino. https://cursos.mcielectronics.cl/2022/12/27/interfaz-del-sensor-flex-con-arduino/

Maldonado Toro, G. (2012). Diseño, construcción y programación de una máquina de control numérico aplicada al prototipado rápido de modelado por deposición fundida (Trabajo de grado). Universidad Internacional del Ecuador. http://repositorio.uide.edu.ec/handle/37000/449

Medina, R. (1992). Programación avanzada en lenguaje ensamblador. https://www.academia.edu/download/35605518/Programacion_en_ensamblador_avanzada.pdf

Miranda Colorado, R. (2016). Cinemática y dinámica de robots manipuladores. Alfaomega.

Osorio Rojas, A. (2006). C++ manual teórico-práctico. http://slent.iespana.es/programacion/index.html

Paccha Medina, R. X., & Triviño Sánchez, J. V. (2023). Prototipo de robot cartesiano utilizando control numérico computarizado para la manipulación de botellas PET en el proceso de paletizado (Tesis de licenciatura). Universidad Politécnica Salesiana.

Parra Farfán, M. (2025). Diseño de un sistema de control embebido para prótesis transradial funcional (Tesis de grado). Pontificia Universidad Católica del Perú. http://hdl.handle.net/20.500.12404/29788

Patiño Carrillo, J., & Chitiva Muñoz, J. (2022). Diseño e implementación de un mecanismo basado en el control numérico por computadora para la automatización de una máquina duplicadora de llaves (Tesis de bachiller). Universidad del Pacífico.

Reátegui Pinazo, J. E. (2024). Diseño e implementación de un sistema embebido portátil para la adquisición y procesamiento de señales electromiográficas del antebrazo (Tesis de grado). Pontificia Universidad Católica del Perú. http://hdl.handle.net/20.500.12404/27800

Saldaña Suárez, R. J. (2024). Estudio para el desarrollo de un prototipo de prótesis robótica de mano controlada por señales musculares para manejar vehículos automáticos (Tesis de grado). Pontificia Universidad Católica del Perú. https://tesis.pucp.edu.pe/items/cb9dacdd-a2f4-4443-936e-5483f0170d57

Santana Camilo, J., Blanco Ortega, A., Antúnez Leyva, E., Magadán Salazar, A., & Gómez Becerra, F. (2017). Control numérico en una máquina para rehabilitación de tobillos. Pistas Educativas, 39(125), 592–610. http://www.itc.mx/ojs/index.php/pistas/article/view/898

Santiago, F. (2021). El microcontrolador ATMega328P de Microchip: Programación en ensamblador, lenguaje C y un enlace con Arduino. Universidad Tecnológica de la Mixteca. http://repositorio.utm.mx:8080/jspui/handle/123456789/388

Tecnología Mecánica. (s. f.). Control numérico. http://www.sitenordeste.com/mecanica/control_numerico.htm

Universidad Nacional Autónoma de México. (2014). Lenguajes de programación. https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/1023/mod_resource/content/1/contenido/index.html

Úbeda Sequeira, L. E., Jiménez López, Y. E., & Méndez Talavera, A. A. (2011). Máquina de control numérico computarizado CNC de cuatro ejes con comunicación USB para la

automatización de los procesos de grabado en madera, plástico y materiales no convencionales. Universidad Nacional de Ingeniería.

Universal Robots. (s. f.). Universal Robots. https://www.universal-robots.com/es/

Yaskawa Europe GmbH. (s. f.). Yaskawa Europe GmbH. https://www.yaskawa.eu.com/about-yaskawa/

ANEXO A

MICROCONTROLADOR ARDUINO MEGA R3 DATA SHEET

Arduino® MEGA 2560 Rev3

Product Reference Manual
SKU: A000067

Description

Arduino® Mega 2560 is an exemplary development board dedicated for building extensive applications as compared to other maker boards by Arduino. The board accommodates the ATmega2560 microcontroller, which operates at a frequency of 16 MHz. The board contains 54 digital input/output pins, 16 analog inputs, 4 UARTs (hardware serial ports), a USB connection, a power jack, an ICSP header, and a reset button.

Target Areas

3D Printing, Robotics, Maker

Features

- **ATmega2560 Processor**
  - Up to 16 MIPS Throughput at 16MHz
  - 256k bytes (of which 8k is used for the bootloader)
  - 4k bytes EEPROM
  - 8k bytes Internal SRAM
  - 32 × 8 General Purpose Working Registers
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Four Programmable Serial USART
  - Controller/Peripheral SPI Serial Interface
- **ATmega16U2**
  - Up to 16 MIPS Throughput at 16 MHz
  - 16k bytes ISP Flash Memory
  - 512 bytes EEPROM
  - 512 bytes SRAM
  - USART with SPI master only mode and hardware flow control (RTS/CTS)
  - Master/Slave SPI Serial Interface
- **Sleep Modes**
  - Idle
  - ADC Noise Reduction
  - Power-save
  - Power-down
  - Standby
  - Extended Standby
- **Power**
  - USB Connection
  - External AC/DC Adapter
- **I/O**
  - 54 Digital
  - 16 Analog
  - 15 PWM Output

Contents

1 The Board

Arduino® Mega 2560 is a successor board of Arduino Mega, it is dedicated to applications and projects that require large number of input output pins and the use cases which need high processing power. The Arduino® Mega 2560 comes with a much larger set of IOs when we compare it with traditional Uno board considering the form factor of both the boards.

1.1 Application Examples

- **Robotics**: Featuring the high processing capacitity, the Arduino Mega 2560 can handle the extensive robotic applications. It is compatible with the motor controller shield that enables it to control multiple motors at an instance, thus making it perfect of robotic applications. The large number of I/O pins can accommodate many robotic sensors as well.
- **3D Printing**: Algorithms play a significant role in implementation of 3D printers. Arduino Mega 2560 has the power to process these complex algorithms required for 3D printing. Additionally, the slight changes to the code is easily possible with the Arduino IDE and thus 3D printing programs can be customized according to user requirements.
- **WI-FI**: Integrating wireless functionality enhances the utility of the applications. Arduino Mega 2560 is compatible with WiFi shields hence allowing the wireless features for the applications in 3D printing and Robotics.

1.2 Accessories

1.3 Related Products

- Arduino® Uno Rev 3
- Arduino® Nano
- Arduino® DUE without headers

2 Ratings

2.1 Recommended Operating Conditions

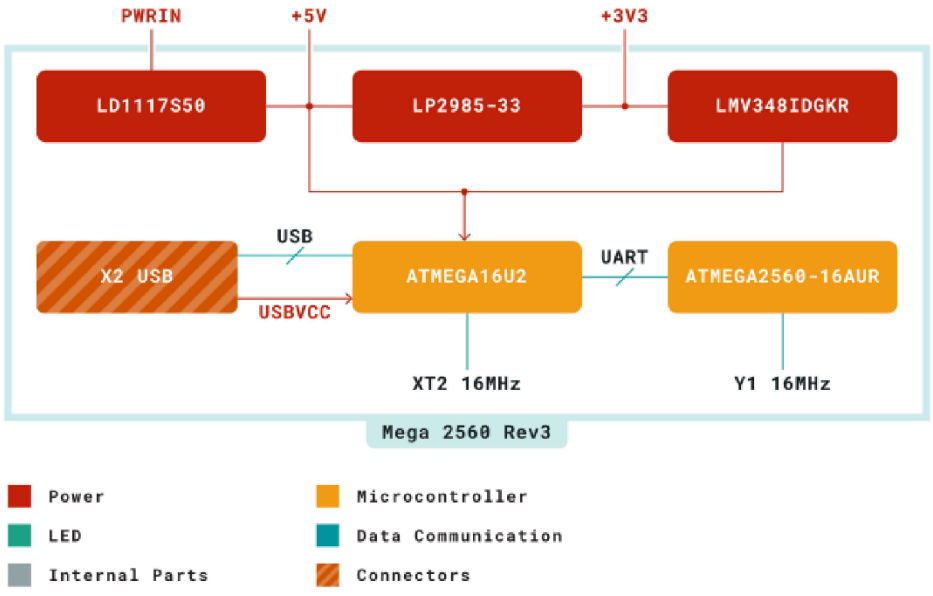
Symbol	Description	Min	Max
TOP	Operating temperature:	-40 °C	85 °C

2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
PWRIN	Input supply from power jack		TBC		mW
USB VCC	Input supply from USB		TBC		mW
VIN	Input from VIN pad		TBC		mW

3 Functional Overview

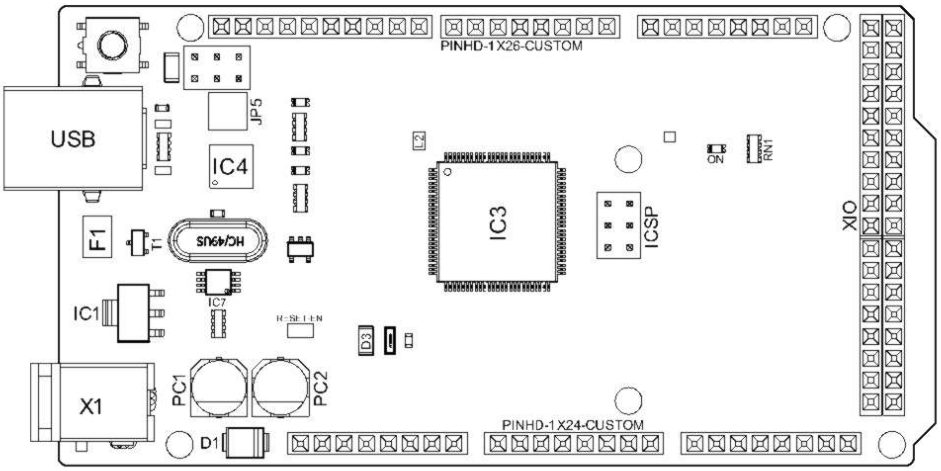
3.1 Block Diagram



Arduino MEGA Block Diagram

3.2 Board Topology

Front View



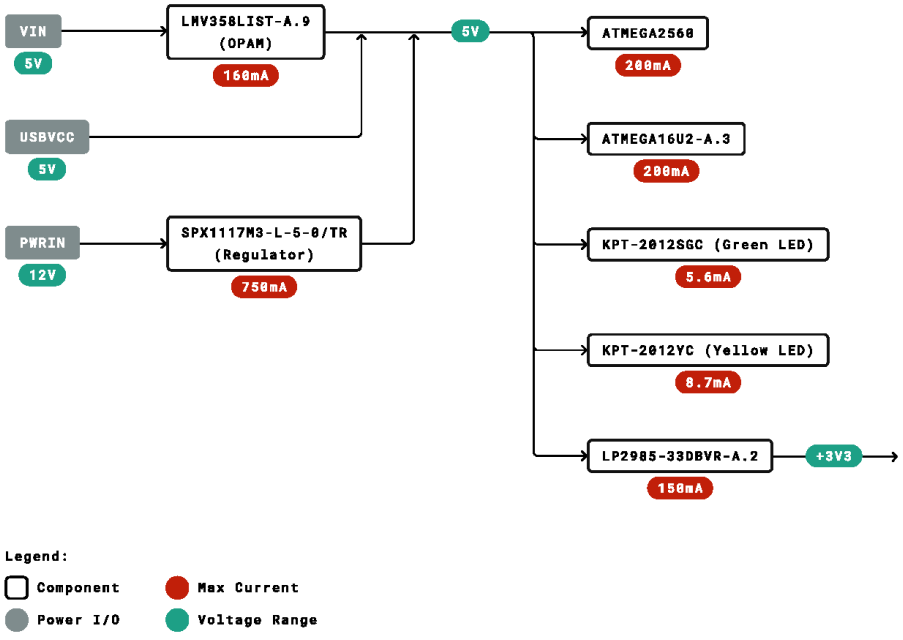
Arduino MEGA Top View

Ref.	Description	Ref.	Description
USB	USB B Connector	F1	Chip Capacitor
IC1	5V Linear Regulator	X1	Power Jack Connector
JP5	Plated Holes	IC4	ATmega16U2 chip
PC1	Electrolytic Alumninum Capacitor	PC2	Electrolytic Alumninum Capacitor
D1	General Purpose Rectifier	D3	General Purpose Diode
L2	Fixed Inductor	IC3	ATmega2560 chip
ICSP	Connector Header	ON	Green LED
RN1	Resistor Array	XIO	Connector

3.3 Processor

Primary processor of Arduino Mega 2560 Rev3 board is ATmega2560 chip which operates at a frequency of 16 MHz. It accommodates a large number of input and output lines which gives the provision of interfacing many external devices. At the same time the operations and processing is not slowed due to its significantly larger RAM than the other processors. The board also features a USB serial processor ATmega16U2 which acts an interface between the USB input signals and the main processor. This increases the flexibility of interfacing and connecting peripherals to the Arduino Mega 2560 Rev 3 board.

3.4 Power Tree



Power Tree

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino® MEGA 2560 while offline you need to install the Arduino® Desktop IDE **[1]** To connect the Arduino® MEGA 2560 to your computer, you'll need a Type-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino® boards, including this one, work out-of-the-box on the Arduino® Web Editor **[2]**, by just installing a simple plugin.

The Arduino® Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

4.3 Sample Sketches

Sample sketches for the Arduino® MEGA 2560 can be found either in the "Examples" menu in the Arduino® IDE

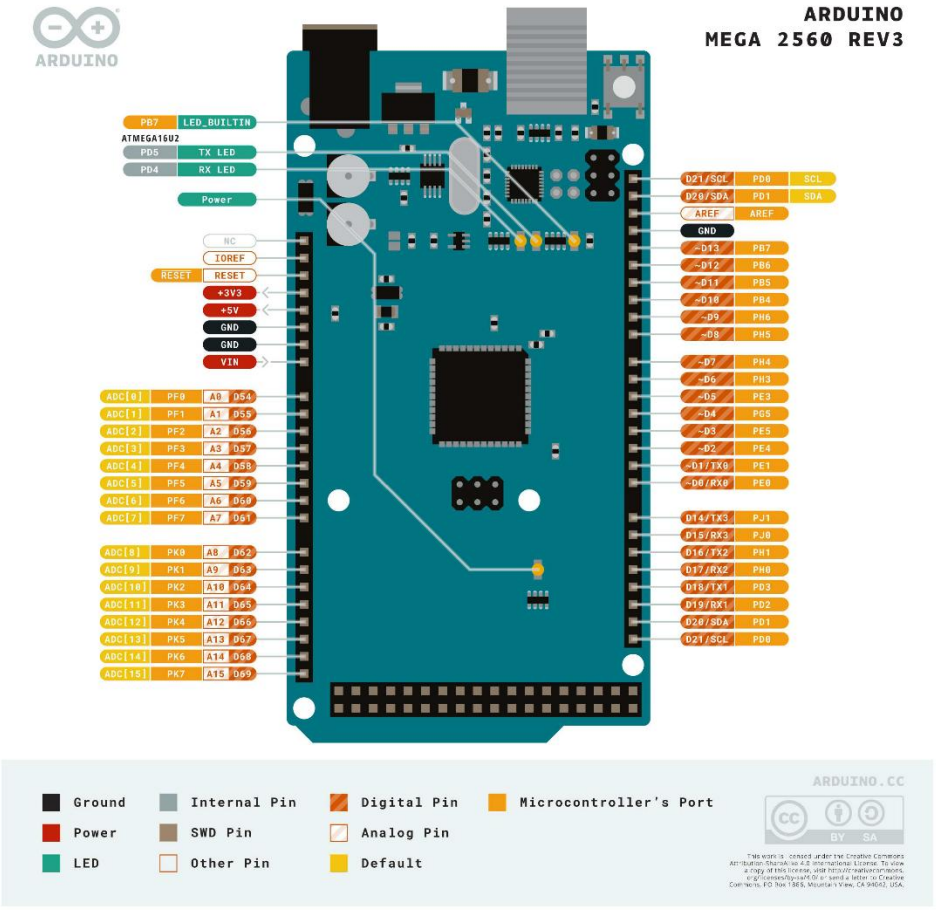
4.4 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino® Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more.

4.5 Board Recovery

All Arduino boards have a built-in bootloader which allows flashing the board via USB. In case a sketch locks up the processor and the board is not reachable anymore via USB it is possible to enter bootloader mode by double-tapping the reset button right after power up.

5 Connector Pinouts



Arduino Mega Pinout

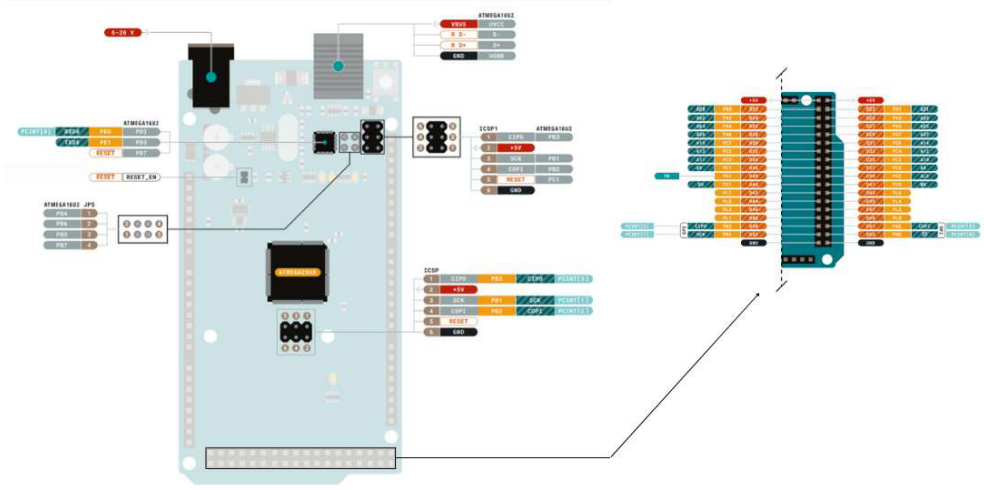
5.1 Analog

Pin	Function	Type	Description
1	NC	NC	Not Connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog	Analog input 0 /GPIO
10	A1	Analog	Analog input 1 /GPIO
11	A2	Analog	Analog input 2 /GPIO
12	A3	Analog	Analog input 3 /GPIO
13	A4	Analog	Analog input 4 /GPIO
14	A5	Analog	Analog input 5 /GPIO
15	A6	Analog	Analog input 6 /GPIO
16	A7	Analog	Analog input 7 /GPIO
17	A8	Analog	Analog input 8 /GPIO
18	A9	Analog	Analog input 9 /GPIO
19	A10	Analog	Analog input 10 /GPIO
20	A11	Analog	Analog input 11 /GPIO
21	A12	Analog	Analog input 12 /GPIO
22	A13	Analog	Analog input 13 /GPIO
23	A14	Analog	Analog input 14 /GPIO
24	A15	Analog	Analog input 15 /GPIO

5.2 Digital

Pin	Function	Type	Description
1	D21/SCL	Digital Input/I2C	Digital input 21/I2C Dataline
2	D20/SDA	Digital Input/I2C	Digital input 20/I2C Dataline
3	AREF	Digital	Analog Reference Voltage
4	GND	Power	Ground
5	D13	Digital/GPIO	Digital input 13/GPIO
6	D12	Digital/GPIO	Digital input 12/GPIO
7	D11	Digital/GPIO	Digital input 11/GPIO
8	D10	Digital/GPIO	Digital input 10/GPIO
9	D9	Digital/GPIO	Digital input 9/GPIO
10	D8	Digital/GPIO	Digital input 8/GPIO
11	D7	Digital/GPIO	Digital input 7/GPIO
12	D6	Digital/GPIO	Digital input 6/GPIO
13	D5	Digital/GPIO	Digital input 5/GPIO
14	D4	Digital/GPIO	Digital input 4/GPIO

Pin	Function	Type	Description
15	D3	Digital/GPIO	Digital input 3/GPIO
16	D2	Digital/GPIO	Digital input 2/GPIO
17	D1/TX0	Digital/GPIO	Digital input 1 /GPIO
18	D0/Tx1	Digital/GPIO	Digital input 0 /GPIO
19	D14	Digital/GPIO	Digital input 14 /GPIO
20	D15	Digital/GPIO	Digital input 15 /GPIO
21	D16	Digital/GPIO	Digital input 16 /GPIO
22	D17	Digital/GPIO	Digital input 17 /GPIO
23	D18	Digital/GPIO	Digital input 18 /GPIO
24	D19	Digital/GPIO	Digital input 19 /GPIO
25	D20	Digital/GPIO	Digital input 20 /GPIO
26	D21	Digital/GPIO	Digital input 21 /GPIO



Arduino Mega Pinout

5.3 ATMEGA16U2 JP5

Pin	Function	Type	Description
1	PB4	Internal	Serial Wire Debug
2	PB6	Internal	Serial Wire Debug
3	PB5	Internal	Serial Wire Debug
4	PB7	Internal	Serial Wire Debug

5.4 ATMEGA16U2 ICSP1

Pin	Function	Type	Description
1	CIPO	Internal	Controller In Peripheral Out
2	+5V	Internal	Power Supply of 5V
3	SCK	Internal	Serial Clock
4	COPI	Internal	Controller Out Peripheral In
5	RESET	Internal	Reset
6	GND	Internal	Ground

5.5 Digital Pins D22 - D53 LHS

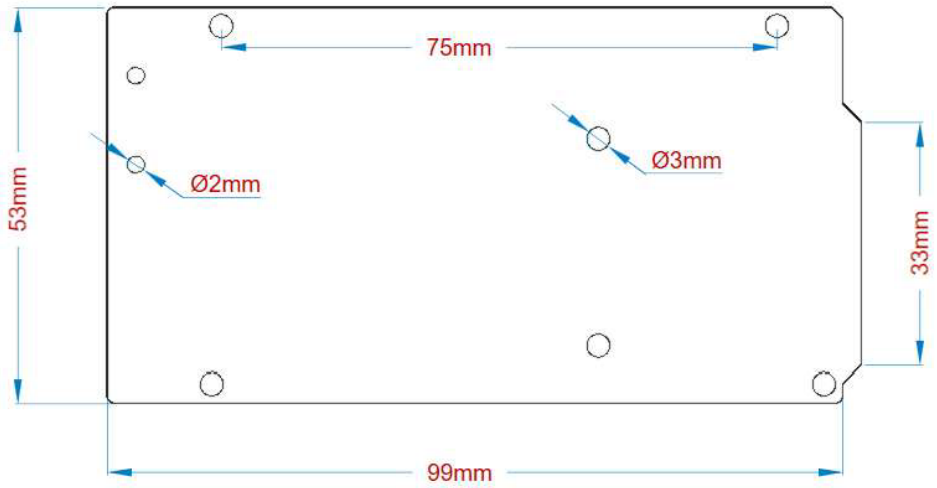
Pin	Function	Type	Description
1	+5V	Power	Power Supply of 5V
2	D22	Digital	Digital input 22/GPIO
3	D24	Digital	Digital input 24/GPIO
4	D26	Digital	Digital input 26/GPIO
5	D28	Digital	Digital input 28/GPIO
6	D30	Digital	Digital input 30/GPIO
7	D32	Digital	Digital input 32/GPIO
8	D34	Digital	Digital input 34/GPIO
9	D36	Digital	Digital input 36/GPIO
10	D38	Digital	Digital input 38/GPIO
11	D40	Digital	Digital input 40/GPIO
12	D42	Digital	Digital input 42/GPIO
13	D44	Digital	Digital input 44/GPIO
14	D46	Digital	Digital input 46/GPIO
15	D48	Digital	Digital input 48/GPIO
16	D50	Digital	Digital input 50/GPIO
17	D52	Digital	Digital input 52/GPIO
18	GND	Power	Ground

5.6 Digital Pins D22 - D53 RHS

Pin	Function	Type	Description
1	+5V	Power	Power Supply of 5V
2	D23	Digital	Digital input 23/GPIO
3	D25	Digital	Digital input 25/GPIO
4	D27	Digital	Digital input 27/GPIO
5	D29	Digital	Digital input 29/GPIO
6	D31	Digital	Digital input 31/GPIO
7	D33	Digital	Digital input 33/GPIO
8	D35	Digital	Digital input 35/GPIO
9	D37	Digital	Digital input 37/GPIO
10	D39	Digital	Digital input 39/GPIO
11	D41	Digital	Digital input 41/GPIO
12	D43	Digital	Digital input 43/GPIO
13	D45	Digital	Digital input 45/GPIO
14	D47	Digital	Digital input 47/GPIO
15	D49	Digital	Digital input 49/GPIO
16	D51	Digital	Digital input 51/GPIO
17	D53	Digital	Digital input 53/GPIO
18	GND	Power	Ground

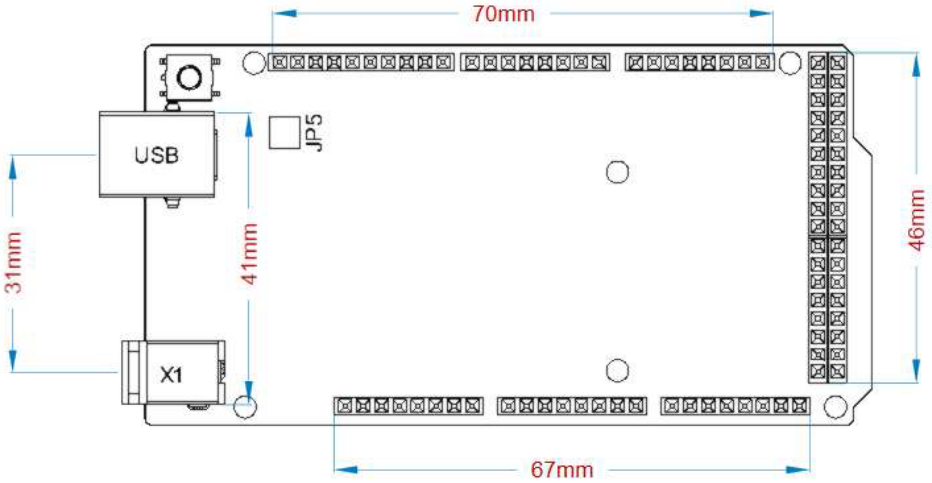
6 Mechanical Information

6.1 Board Outline



Arduino Mega Outline

6.2 Board Mount Holes



Arduino Mega Mount Holes

Certifications

7 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

8 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum Limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions : No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

9 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

10 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for licence-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 85°C and shouldn't be lower than -40°C.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 201453/EU. This product is allowed to be used in all EU member states.

11 Company Information

Company name	Arduino S.r.l.
Company Address	Arduino SRL, Via Andrea Appiani 25, 20900 Monza MB, Italy

12 Reference Documentation

Ref	Link
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Pro Website	https://www.arduino.cc/pro
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/libraries/
Online Store	https://store.arduino.cc/

13 Revision History

Date	Revision	Changes
29/09/2020	1	First Release

ANEXO B

SENSOR SHIELD V1.0

WÜRTH ELEKTRONIK

SENSOR SHIELD

INTERFACE EXTENSION BOARD FOR ARDUINO

order code	product name
2501000101291	Sensor Shield

VERSION 1.0

NOVEMBER 18, 2020

1 Supported products

The evaluation board described in this manual is a stackable extention board for the Arduino (UNO and DUE) board. It can be used to connect the following Sensor EVAL-Boards:

Sensor	Evaluation board Order code	Sensor order code
Temperature sensor IC	2521020222591	2521020222501
Humidity sensor with integrated temperature sensor	2525020210091	2525020210001
3 axis acceleration sensor	2533203301691	2533020201601
Absolute pressure sensor	2511223013391	2511020213301
Differential pressure sensor	2513254510091 2513254510191 2513254510291 2513254510391 2513254510491	2513130810001 2513130810101 2513130810201 2513130810301 2513130810401

Table 1: Sensor shield compatibility

The following interfaces are supported by the sensors: The corresponding sensors that can be evaluated with this sensor shield are:

Sensors & Sensor Eval Board	Interface			
	3.3V I^2C	3.3V SPI	5V I^2C	5V SPI
Temperature sensor IC	x	x		
Humidity sensor with integrated temperature sensor	x	x		
3 axis acceleration sensor	x	x		
Absolute pressure sensor	x	x		
Differential pressure sensor			x	

Table 2: Supported sensor interface

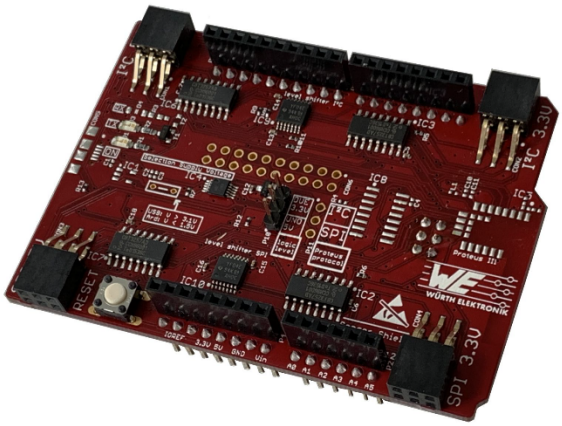


Figure 1: Sensor shield for Arduino

Order code:2501000101291 contents	Quantity
Sensor shield including needed jumpers	1
ESD safe packaging	1
Radio module with individualized firmware	0
Arduino board	0

Table 3: Contents sensor shield EV-kit

2 Overview

2.1 Block diagram

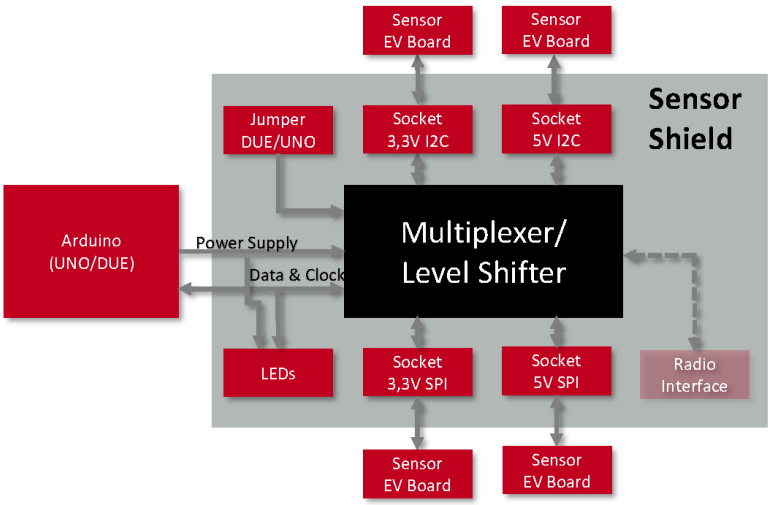


Figure 2: Block Diagramm

2.2 Functional description

The Sensor shield is an interface extension board for Arduino UNO and DUE. It offers the user the possibility to connect sensors with the two different logic levels of 3.3V or 5V to either an Arduino UNO or an Arduino DUE via SPI or I^2C communication interface.

Software libraries for the Arduino platform are also provided for quick software applications. it implements the drivers and example codes to use the Sensor shield with the Würth Elektronik eiSos GmbH & Co. KG Sensor EV-Boards listed in chapter `Supported products`.

On demand there is also a possibility to add a radio module from the Proteus / Thyone family with individualized firmware instead of hosting the sensors with the Arduino. Additional information on the firmware individualisation can be found on the website *here*.

Feel free to check our YouTube channel:
www.youtube.com/user/WuerthElektronik/videos for video tutorials, hands-ons and webinars relating to our products.

2.3 Taking into operation

Sensor shield can either be mounted on Arduino UNO or DUE. For easy orientation of the shield on the Ardiuino board, it follows Arduino board's shape. Match the diagonal bulge one

above the other as shown in Figure 12.

For the proper operation of the Sensor shield, place the jumper (P10) on the correct location based on the Arduino board variant in use (UNO or DUE). Further information can be found in the section 3.1

Connect external power supply to the Arduino board and make sure the VCC is stable and able to reliably supply the module's static and peak current consumption as specified by the Arduino UNO/DUE manual.

The next step is to connect the Arduino board along with the shield to the PC using a USB-cable. In that way a COM port can be detected. Check the device manager to acquire the COM port number of the EV board.

Plug in the sensor evaluation boad that you need to take into operation to the appropriate 6-pin connector (CON1-CON4).

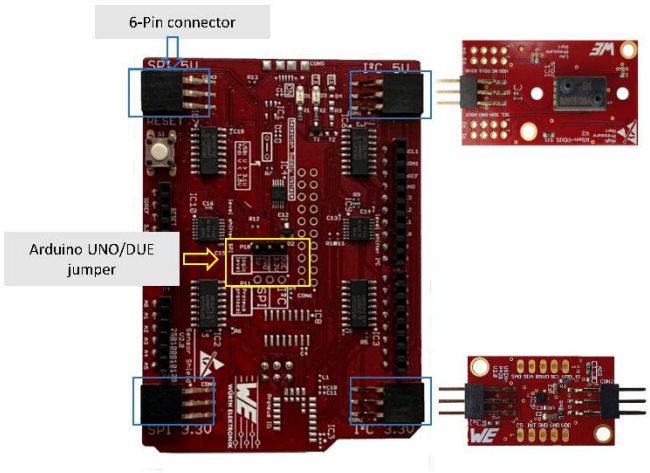


Figure 3: Sensor shield and WSEN-EVAL compatibility

Make sure to connect the sensor EVAL board to the correct connector on the Sensor shield according to the operating voltage and the communication interface of the sensor.

STOP

Connecting the sensor EVAL board to the wrong connector can cause permanent damage to the sensor.

2.4 Sensor libraries for Arduino

The Arduino libraries for I²C are delivered as a compressed zip-file. All codes related to a sensor supported by the Arduino libraries are placed under a sub directory named after the corresponding sensor. Each sensor directory contains three sub-directories, src, examples and Function_Test. The sub-directory src contain the Arduino platform dependent code. The examples folder contains sample applications that can run on the Arduino and the Function_test folder implements the functions to test the library.

The Arduino libraries are available for download at *https://www.we-online.de/katalog/de/wco/sensors/evaluation_boards_wsen*

2.5 Integration into Arduino IDE

The open-source Arduino IDE makes it easy to write code and upload it to the board. This software can be used with any Arduino board. For further information please refer to the installation guide on the Arduino official website: *https://www.arduino.cc/en/Guide*

The sensor libraries can be added in the Arduino IDE software directly as .ZIP file. Select the required sensor library from your PC.

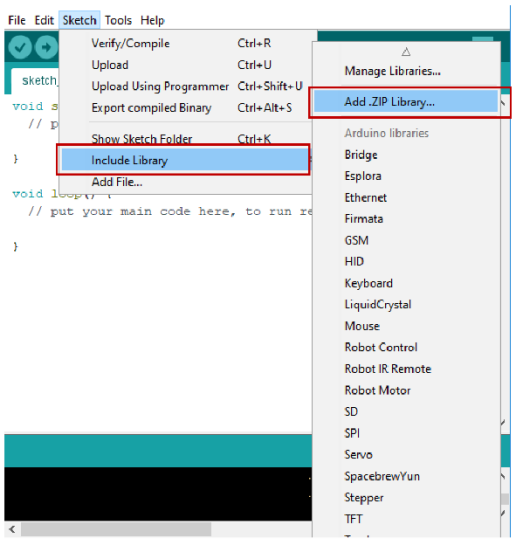


Figure 4: Add sensor libraries to the Arduino software

Successfully included libraries could be found under 'contributed libraries'.

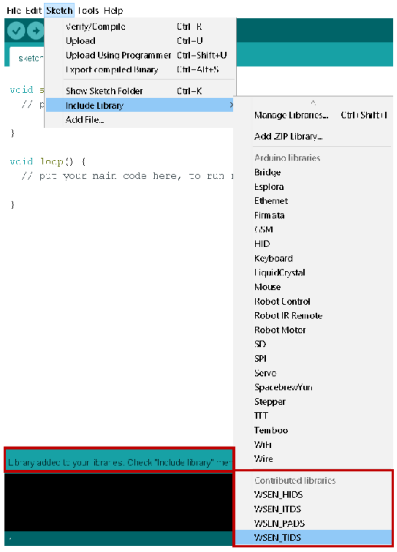


Figure 5: Sensor libraries

The sensor libraries also include some examples for quick start. They are available under File>Examples>Examples from custom libraries.

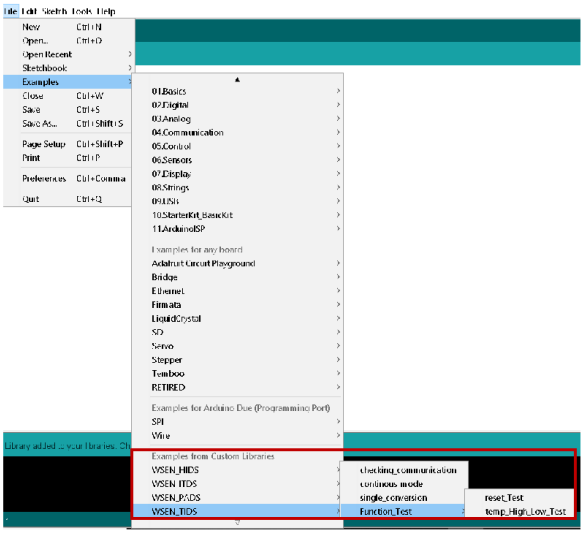


Figure 6: Examples from the sensor libraries

An example code for the temperature sensor WSEN-TIDS (Part number: 2521020222501) in continuous mode is shown in Figure 7

Temperature sensor evaluation board (part no. 2521020222591) must be connected to 3.3V I^2C pin header.

Make sure to place the Sensor shield jumper on the correct position based on the Arduino board variant in use. Further information is given in the section 3.1

File Edit Sketch Tools Help

continous_mode §

```
#include "WSEN_TIDS.h"

Sensor_TIDS sensor;

//The Output Data Rate in Hz
int ODR = 25;

void setup()
{

  Serial.begin(9600);

  // Initialize the I2C interface
  sensor.init(TIDS_ADDRESS_I2C_1);

  // Set the free run mode with given ODR
  sensor.set_continous_mode(ODR);
}

void loop()
{
  // Read and calculate the temperature
  int temperature = sensor.read_temperature();

  // Print the temperature on the serial monitor
  Serial.print(temperature);
  Serial.println(" Celsius");

  // Waiting time between measurement
  int waitMillis = 1000 / ODR;

  // Wait before continuing with the next measurement
  delay(waitMillis);
}
```

Figure 7: WSEN-TIDS example code (continuous mode)

After the Arduino is connected to the PC through the USB, a connected Arduino device should be visible on a COM port. Select the correct Arduino board, corresponding COM port and upload the code. Once the code is successfully uploaded, the output can be observed on the serial monitor.

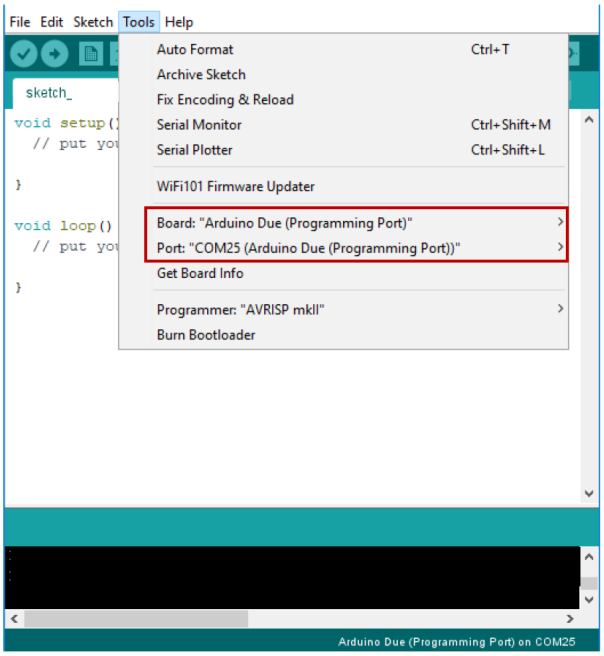


Figure 8: Select Arduino board, COM port and upload the code

3 Sensor shield

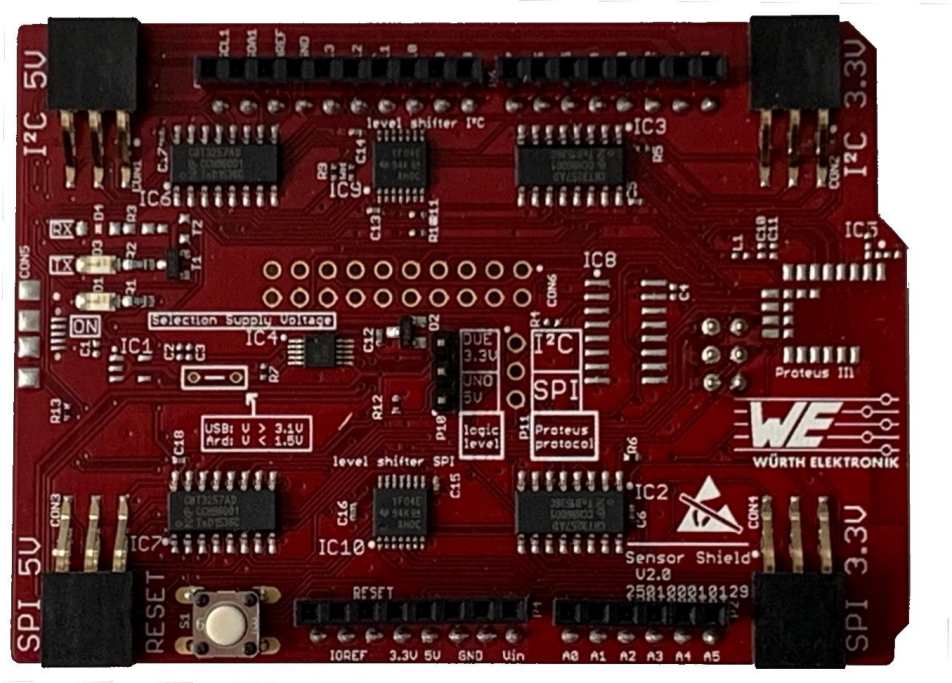


Figure 9: Evaluation board top view

The evaluation board is provided with an informative silkscreen to ease its use.

3.1 Jumpers

Only one jumper is needed to decide about connecting either an Arduino UNO, switching the multiplexer and level shifter in the accordingly function for 5V logic of the Arduino, or connecting an Arduino DUE, switching the multiplexer and level shifter accordingly for 3.3V logic on the Arduino connector. The following figure shows the jumper position to use an Arduino DUE.

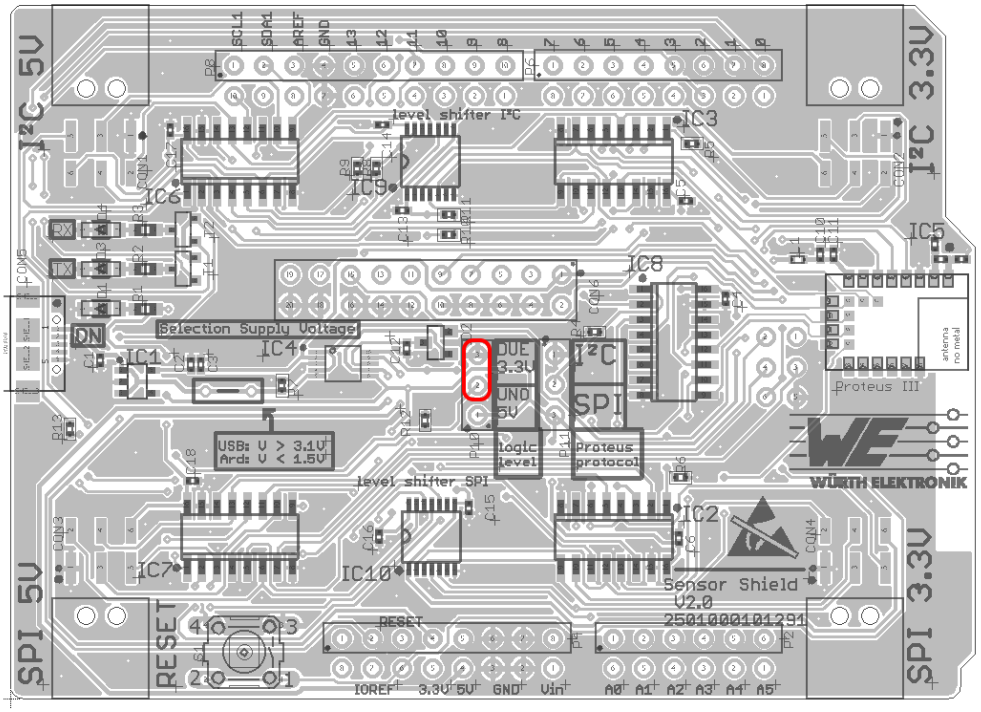


Figure 10: Jumper setting connecting Arduino DUE

3.2 Connectors and pin headers

There are several connectors on the top and bottom side of the sensor shield. They serve to connect the Arduino board, to stack other Sensor shields and to connect sensors via 2.54mm headers or to connect directly the Würth Elektronik eiSos GmbH & Co. KG Sensor EV-Boards.

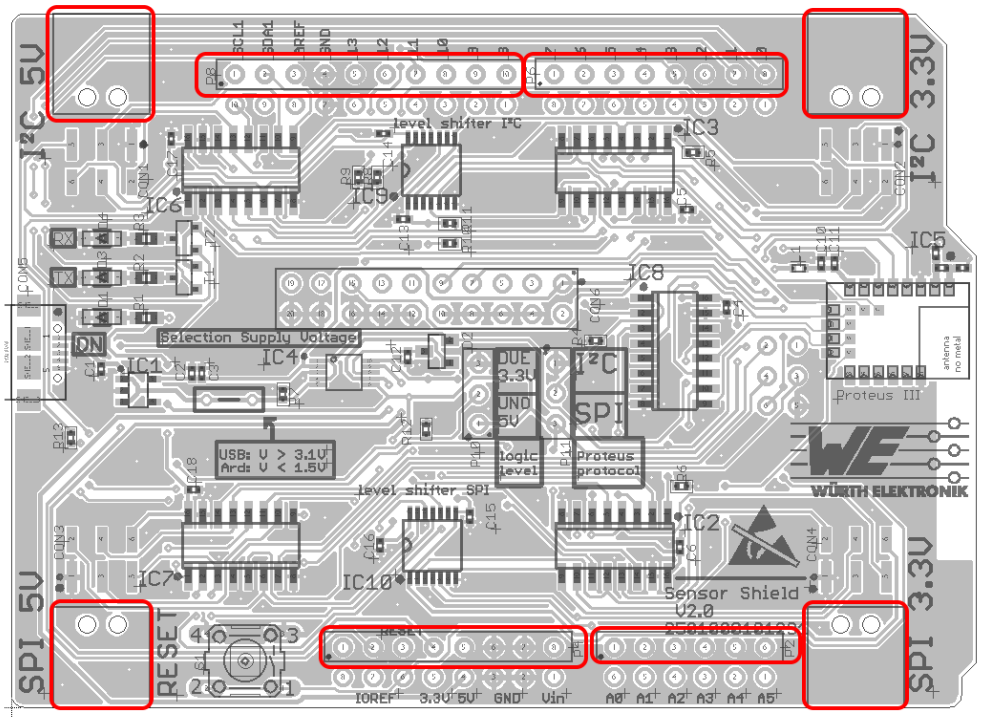


Figure 11: Connectors and pin headers top side

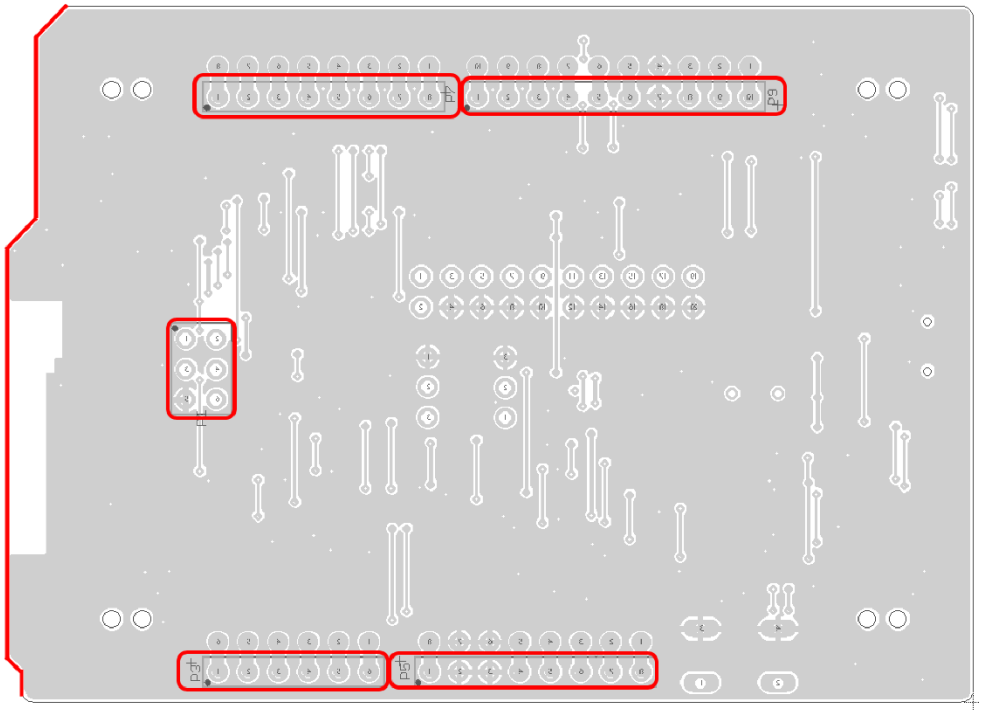


Figure 12: Connectors and pin headers bottom side

header / connector	Description
P2, P4, P6, P8	Socket to stack up another Sensor shield
CON1	I²C 5V connector
CON2	I²C 3.3V connector
CON3	SPI 5V connector
CON4	SPI 3.3V connector
P3, P5, P7, P9	header on bottom side to connect the Arduino board
P1	socket on bottom side to connect the Arduino board

Table 4: Connectors and pin headers

For easy orientation of the shield on the Ardiuino board, it follows Arduino board's shape. Match the diagonal bulge one above the other.

3.2.1 CON1 and CON2 I²C connectors

The pinning of connector CON1 and CON2 fits directly to the sensor evaluation boards mentioned in chapter `Supported products`. The connector on the shield is a 2.54mm pitch socket to have the possibility to connect any other I²C device with the most commenly used headers.

Pin	Description
1	GND
2	SCL
3	SDA
4	INT1
5	INT0
6	VDD

Table 5: Pinning I²C connector CON1 and CON2

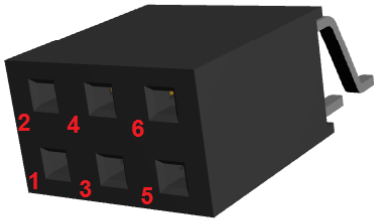


Figure 13: Pinning I²C connector CON1 and CON2

3.2.2 CON3 and CON4 SPI connectors

The pinning of connector CON3 and CON4 fits directly to the sensor evaluation boards mentioned in chapter `Supported products`. The connector on the shield is a 2.54mm pitch socket to have the possibility to connect any other SPI device with the most commenly used headers.

Pin	Description
1	GND
2	SCLK
3	MOSI
4	CS
5	MISO
6	VDD

Table 6: Pinning SPI connector CON3 and CON4

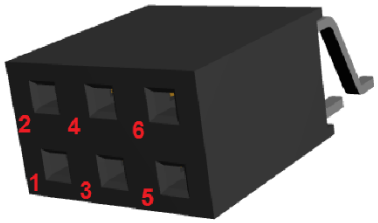


Figure 14: Pinning SPI connector CON3 and CON4

3.3 Buttons

3.3.1 Reset button

The reset button pulls the reset line low to trigger a reset. It is connected to the Arduino header to trigger a reset of the Arduino processor.

3.4 Function blocks

3.4.1 Power supply

The Sensor Shield is powered from the Arduino board via the socket P5.

Powering the Sensor shield via an USB cable through on board USB connector (not mounted) will switch off the I^2C/SPI signal lines between the sensor connectors (CON1-CON4) and the Arduino.

3.4.2 Level shifter

Two level shifters are used to change logic level between 5V logic and 3.3V logic. One IC is used for the SPI lines, the other for the I^2C lines.

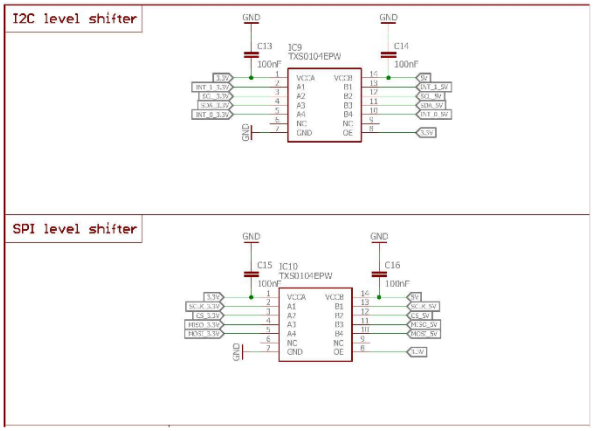


Figure 15: Level shifter

3.4.3 Multiplexer

The multiplexers serve the purpose of splitting the 3.3V logic level signals from Arduino DUE towards connector CON2 and CON4 and through the level shifter towards CON1 and CON3. Similarly in case of Arduino UNO, the multiplexer split the 5V logic level signals from the Arduino UNO directly toward CON1 and CON3 and through level shifter towards CON2 and CON4.

By setting the jumper based on the Arduino variant (UNO or DUE) on the three pin header P10 (as already described in section 3.1), the logic level on P1 - P9 can be selected.

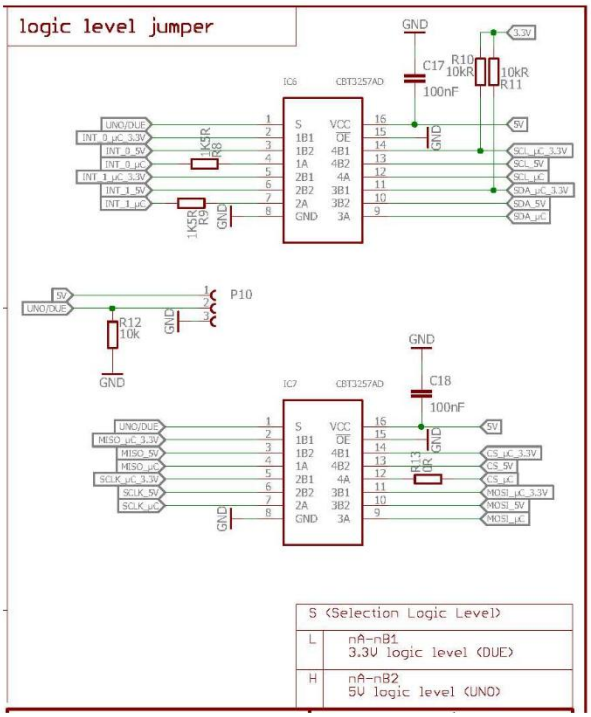


Figure 16: Multiplexer

3.4.4 Radio interface with firmware individualization on demand

There is a dedicated place holder for a radio module of the type Proteus or Thyone-I on the Sensor shield. With an individualized Firmware this radio module could independently collect sensor data and send them via radio protocol to a remote station. For this feature no Arduino board is used and the Sensor shield operates stand alone.

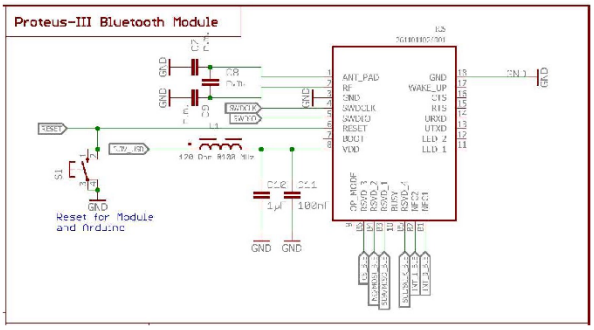


Figure 17: Radio module

The modules Proteus-I, Proteus-II, Proteus-III and Thyone-I standard firmware does not support this independent solution.

Please contact your local field sales engineer (FSE) or wireless-sales@we-online.com for further information.

3.4.4.1 Switching SPI and I²C interface to radio module

When connecting an USB cable directly to the Sensor shield, 5V from the USB is used to switch on the IC2 and IC3 multiplexers to connect the interface from the sensor connectors to the radio module. Without USB connection, pull down resistors switch off the multiplexers so that the sensor connectors are connected to the Arduino.

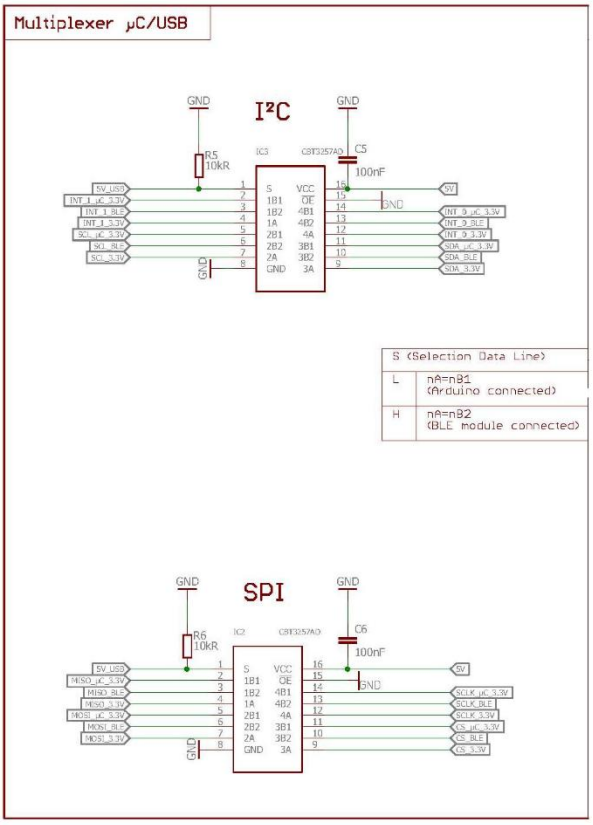


Figure 18: Radio module connection

3.4.4.2 Programming interface

The evaluation board provides a 2 x 10 pin connector to connect directly a JTAG or SWD flash adapter used for development. Please take care of the correct mounting of the flash adapter (Pin 1 is marked as such). Depending on your flash tool an additional adapter may be required.
The recommended flash adapter is one from the "Segger J-Link" family.

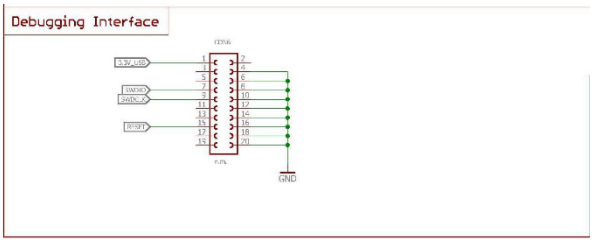


Figure 19: Debugging interface of the optional radio module