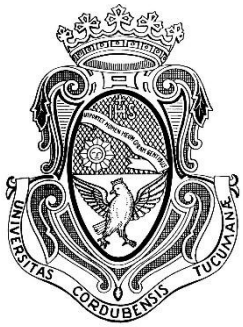



UNIVERSIDAD NACIONAL DE CÓRDOBA
FACULTAD DE CIENCIAS EXACTAS, FÍSICAS Y NATURALES
CARRERA INGENIERÍA ELECTRÓNICA

PROYECTO INTEGRADOR PARA LA OBTENCIÓN DEL TÍTULO DE GRADO INGENIERO ELECTRÓNICO

# “CONTROL DE UN ROBOT ANIMATRÓNICO TIPO REPTIL VOLADOR”

Alumnos
Moreno, Agustín Ezequiel y Frenguelli, Gino

Director
Ing. Esp. Pailos, N Hugo

Córdoba, República Argentina

Marzo 2019

## *UNIVERSIDAD NACIONAL DE CÓRDOBA*

## *Facultad de Ciencias Exactas, Físicas y Naturales*

## *Escuela de Ingeniería Electrónica*

El Tribunal Evaluador reunido en este acto y luego de haber aprobado la Solicitud de Aprobación de Tema y efectuado las distintas instancias de correcciones del Informe del Proyecto Integrador para la obtención del Título de Grado “Ingeniero Electrónico” y cumpliendo con el Reglamento correspondiente, declaran el Informe Final de los estudiantes **Moreno Agustín Ezequiel** y **Frenguelli Gino** como “aceptado sin correcciones” y la defensa oral Aprobada. Por lo tanto, luego de haber tenido en cuenta los aspectos de evaluación que indica el Reglamento, el Proyecto Integrador se considera Aprobado.

Se firma el Acta de Examen correspondiente y se distribuyen los ejemplares impresos.

NOTA:

Firma y aclaración del Tribunal Evaluador

Fecha:

## *UNIVERSIDAD NACIONAL DE CÓRDOBA*

## *Facultad de Ciencias Exactas, Físicas y Naturales*

## *Escuela de Ingeniería Electrónica*

Quien suscribe el Profesor Pailos Hugo Nicolas en su carácter de Director del Proyecto Integrador de los Estudiantes Moreno Agustín Ezequiel y Frenguelli Gino, denominado Control de un robot animatrónico tipo reptil volador, considera que el desarrollo del trabajo se ha completado según lo especificado en la Solicitud de Aprobación de Tema y se encuentra en condiciones de tramitar su defensa.

A los efectos de quién corresponda, en fecha 08/03/2019

Firma y aclaración del Director

## Dedicatoria

*De parte de Agustín…*

*A mi abuela Rosa, mis padres Patricia y Ángel y a mi hermana María Laura los pilares de mi familia, por su apoyo incondicional y por la confianza depositada en mí.*

*También a Noelia, mi novia quien me alentó y acompañó en los momentos mas difíciles de la carrera.*

*De parte de Gino…*

*A mis padres Magdalena y Walter, hermanos Serena y Walter, que han sido un pilar fundamental para mi formacion brindando confianza, consejos y recursos para lograrlo.*

*Por último, a mi esposa Coni e hija Paulina, por estar siempre dando su amor, cariño y comprensión.*

## Agradecimientos

*A toda nuestra familia por la confianza entregada durante todo este tiempo de estudio y sobre todo por haber sido de apoyo en este camino hecho.*

*A nuestro director Ing. Esp. Hugo Pailos por su entusiasmo en querer compartir su experiencia sobre los diseños robóticos y sobre todo por el incentivo a terminar este ciclo.*

*A Samuel Gonzalez, nuestro Paleoartista que colaboro con el diseño del animatrónico.*

*Y a todas aquellas personas que nos ayudaron directa o indirectamente a terminar el proyecto.*

## Resumen

En el presente trabajo se rediseñó y reconstruyó un robot tipo reptil volador extinto que representa a una especie Tapejara, ahora renombrado por los científicos con el nombre de “Tupandactylus” (Tapejara Imperator fue un reptil volador que vivió en Brasil hace 108 millones de años) usando tecnología completamente nueva. Este animatrónico tiene movimientos de cabeza (arriba y abajo) pico (abre y cierra) y de alas (aleteo) con naturalidad y es controlado de dos modos: un modo “rutina” que tiene una serie de movimientos cada un cierto tiempo, y un modo “controlado” en el cual se controla mediante un “dispositivo interactivo” la posición de las alas.

## Área Temática

Las Áreas Temáticas son: Computación e Informática, Comunicaciones, Control, Digitales.

## Asignaturas

Asignaturas: Robótica y Animatrónica, Sistemas de Control I y Electrónica Digital III.

## Palabras Claves

control, animatrónico, robot, reptil volador, extinto

## Abstract

In the present work, an extinct flying reptile-type robot that represents a Tapejara species, now renamed by scientists with the name of "Tupandactylus" (Tapejara Imperator was a flying reptile that lived in Brazil 108 million years ago) was redesigned and reconstructed using completely new technology. This animatronic has realistic movements of head (up and down), peak (opening and closing) and wings (flapping) and is controlled in two ways: a "routine" mode that has a series of movements every certain periods of time, and a "controlled mode" which controls the position of the wings by means of an "interactive device".

## Thematic area

The Thematic Areas are: Computing and Computing, Communications, Control, Digital.

## Subjects

Subjects: Robotics and Animatronics, Control Systems I and Digital Electronics III.

## Key Words

control, animatronic, robot, flying reptile, extinct.

## Resumo

No presente trabalho, um extinto robô do tipo réptil voador que representa uma espécie de Tapejara, agora renomeado por cientistas com o nome de "Tupandactylus" (Tapejara Imperator era um réptil voador que viveu no Brasil há 108 milhões de anos) foi redesenhado e reconstruído. usando tecnologia completamente nova. Este animatrônico tem movimentos de cabeça (para cima e para baixo) pico (abrir e fechar) e asas (batendo) naturalmente e é controlado de duas maneiras: um modo de "rotina" que tem uma série de movimentos a cada certo tempo, e um modo "controlado" em que se controla por meio de um "dispositivo interativo" a posição das asas.


## Área Temática

As Áreas Temáticas são: Computação e Informática, Comunicações, Controle, Digital.

## Assuntos

Assuntos: Robótica e Animatronics, Sistemas de Controle I e Eletrônica Digital III.


## Palavras Chaves

controle, animatronic, robô, réptil voador, extinto

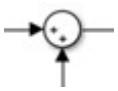
# Índice

# Lista de tablas

# Lista de figuras

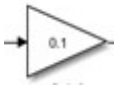
# Lista de Símbolos y Convenciones

| | |
|---|---|
| **IMU** | Unidad de medida inercia |
| **I2C** | Circuito inter integrado |
| **SCL** | System clock |
| **SDA** | System Data |
| **ISM** | Frecuencia de radio de la banda industrial, científico y medico |
| **GND** | Tierra (Groud) |
| **BT** | Bluetooth |
| **DSP** | Digital Signal Processor |
| **PWM** | Modulación por ancho de pulso |
| **Vref** | Tensión de referencia |
| **V** | Tensión |
| **I** | Corriente |
| **R** | Resistencia |
| **C** | Capacitor |
| **L** | Inductor |
| **E** | Fuente |
| **Vpp** | Tensión pico a pico |
| $\boldsymbol{\eta}$ | Rendimiento o eficiencia |
| $\boldsymbol{P_{OUT}}$ | Potencia salida |
| $\boldsymbol{P_{IN}}$ | Potencia entrada |
| $\boldsymbol{V_{IN}}$ | Tensión entrada |
| $\boldsymbol{I_{IN}}$ | Corriente entrada |
| $\boldsymbol{V_{OUT}}$ | Tensión salida |
| $\boldsymbol{I_{IN}}$ | Corriente salida |
| $\boldsymbol{I_Q}$ | Corriente del regulador |
| $\boldsymbol{\delta}$ | Dutty cycle (ciclo de trabajo) |
| $\boldsymbol{T_{on}}$ | Tiempo encendido |
| $\boldsymbol{T}$ | Periodo |
| $\boldsymbol{f_{osc}}$ | Frecuencia de oscilación |
| $\mathbf{V_L}$ | Tensión inductor |
| $\mathbf{I_L}$ | Corriente inductor |
| $\mathbf{V_C}$ | Tensión Capacitor |
| $\mathbf{I_C}$ | Corriente Capacitor |

| | |
|---|---|
| $I_D$ | Corriente Diodo |
| $V_{GS}$ | Tensión gate-source |
| $I_S$ | Corriente source |
| $V_S$ | Tensión source |
| $R_L$ | Resistencia de carga |
| $f_x$ | Frecuencia de conmutación |
| $\propto$ | Constante de ambos filtros |
| $H_a$ | Función de Transferencia del acelerómetro |
| $H_g$ | Función de Transferencia del giroscopio |
| $\theta a$ | Ángulo obtenido por el acelerómetro |
| $\widehat{\theta}$ | Ángulo estimado |
| $\theta g$ | Ángulo obtenido por el giroscopio |
| **µC** | Microcontrolador |
| **ADC** | Conversor analógico digital |
| **DCA** | Conversor digital analógico |
| **SRAM** | Static Random Access Memory |
| **EEPROM** | Electrically Erasable Programmable Read-Only Memory |
| **ROM** | *Read only Memory* |
| **UART** | Universal Asynchronous Receiver Transmitter |
| **PROM** | Programmable read only memory |
| $I_{ind,dc}$ | Corriente media en el inductor |
| $V_{sat}$ | Tensión colector emisor del transistor |
| **Hy** | Henrios |
| **F** | Faradios |
| $R_c$ | Resistencia de compensación |
| $C_c$ | Capacitor de compensación |
| **SMD** | Dispositivo de montaje superficial (Surface Mounted Device) |
| $C_{out}$ | Capacitor de salida |
| $r$ | Resolución del servomotor |
| **M** | Momento |
| **3D** | Tres dimensiones |
|  | Bloque sumador |
|  | Osciloscopio |
|   | Integrador en tiempo discreto |

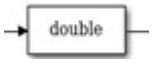

Constante


Guarda datos




Ganancia


Función trigonométrica arcotangente




Conversión de tipo de datos

Multiplexor


Toma de muestra previa

# Capítulo 1. Introducción

Siempre que se rediseña un objeto, un circuito o, como en este caso un robot, se tiende a diseñarlo todo de nuevo alegando que es más fácil “partir de cero”; lo cierto es que en muchos casos esto es verdad; sin embargo, en el caso de “*Tapejara*”, hoy renombrado “*Tupandactilus*”, se pueden recuperar varias partes y sobre todo varios conceptos ya estudiados en su versión original.

La morfología, por ejemplo, no ha cambiado y esto simplifica mucho el diseño actual, se respetó; todos los planos originales se mantienen intactos y fueron muy útiles; si bien los servos de la cabeza y pico estaban un poco deteriorados la posición de estos fue respetada. Sin lugar a duda el aporte más importante en este trabajo final se trata de la forma del censado de posición final de las alas dada por los servomotores de alto torque cuya posición es censada por el mismo servomotor, de la captura de movimiento del animatrónico y de la posibilidad generar un plan de vuelo cuya versión original no posee.

Cabe aclarar que las mejoras con respecto a las fuentes de alimentación eran evidentes por el tipo de electrónica y servomotores usados. El conjunto de mejoras se puede apreciar en un funcionamiento más suave y preciso de todo el sistema y de la previsibilidad de movimientos e interactividad con el usuario.

| | VERSIÓN ORIGINAL | VERSIÓN ACTUAL |
|---|---|---|
| **FUENTE DE ALIMENTACIÓN** | Analógica única | Switching |
| **ACTUADORES ALAS** | Motorreductores | Servomotores de alto torque |
| **SENSORES** | Tres optoacopladores por ala, solo tres posiciones a censar | Señal procesada de la de Unidades De Medición Inercial (IMU´s), censado continua resolución aprox. 1°. |
| **CAPTURA DE MOVIMIENTO** | No tiene | Por medio de (IMU´s) conectadas a los brazos del titiritero. con reproducción en tiempo real. |
| **PLAN DE VUELO** | No tiene | Grabado en memoria de un procesador |
| **COMUNICACIÓN** | Alámbrico (Cables) | Alámbrico/Inalámbrico |
| **PROCESADOR** | PIC 16F84 | ATmega AVR |
| **INTERACTIVIDAD** | Control de velocidad de las alas por medio de potenciómetros. | Control de posición y velocidad por medio de un titiritero, mediante interfaz corporal. |

***Tabla 1.1.-*** *Cuadro comparativo de mejoras versión original Vs Versión actual. Fuente: Propia.*

## 1.1 Antecedentes

En el año 2004 El Doctor Alexander Kellner invita al Ing. Hugo Pailos a participar del “II Congreso Latinoamericano de Paleontología de Vertebrados” en Río de Janeiro. Luego de varias comunicaciones con el Dr. Kellner deciden que se llevará al congreso un animatrónico representando a la especie “Tapejara imperator” a realizarse en el laboratorio de Animatrónica y Control Dinámico de la Carrera de Ingeniería Electrónica. Dicho trabajo de investigación tuvo como director al Ing. Pailos y como asesor principal al paleontólogo Kellner.

El Ingeniero Pailos convoca a las siguientes personas a comienzo del año 2005 para el desarrollo y la construcción del que sería el primer robot representando una especie de pterosaurio de Brasil.

Asesoramiento:

- Lic. Teresa Sanchez (Paleontología) †
- Dr. Alexander Kellner (Paleontología)

Integrantes:

- Rubén Juárez Valieri (Paleontología)
- Lic. Augusto Haro (Paleontología)
- Pamela Horbacovsky (Electrónica y logística)
- Javier Quadri (Mecánica y electrónica)
- Mariano Jean Charles (Hardware, Electrónica y mecánica)
- Luciano Orellano (Mecánica y logística)
- Santiago Druetta (Biología y arte)
- Mariano Rodriguez Isleño (Animación Computada y Diseño 3D)

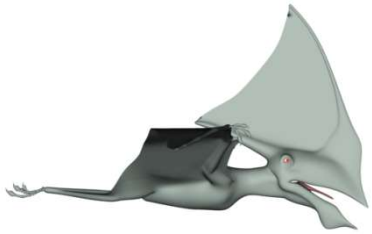


***Figura 1.1.-*** *Modelo en 3D del reptil volador Tupandactylus Imperator. Fuente: Mariano Rodríguez Isleño.*

El prototipo funciona bien, pero es muy básico en sus movimientos igualmente es llevado en el año 2009 Nuevamente a Brasil a las jornadas “Dinos en Rio”, donde por un error el robot es dejado en el museo de Rio de Janeiro donde se realizó el evento.

En el año 2016 el robot es repatriado al laboratorio de Animatrónica y Control Dinámico esperando ser rediseñado para darle mejores movimientos.

## 1.2 Motivación para la elección del Tema

Si bien el animatrónico funciona de acuerdo con lo esperado en los dos eventos expresados en el punto anterior, solo puede mover sus alas en una secuencia fija consistente en 6 (seis) aleteos y movimientos de cabeza y pico mediante botones.

La motivación principal es rediseñar al animatrónico para que sus movimientos sean más naturales por un lado y luego que el sistema nuevo permita hacer “motion capture” desde sensores gobernados por un “titiritero” que puede ser el público en general.

Otra motivación es que se puede usar casi todo lo referente a la parte artística, que se encuentra bien conservada y que da al robot ese aspecto del animal en vida.

## 1.3 Relevancia del Trabajo

La intención del Laboratorio de Animatrónica y Control Dinámico es seguir desarrollando estos dispositivos como lo viene haciendo desde hace años con el fin de ser presentados en eventos científicos, parques temáticos y museos.

Un punto importante es que los zoológicos están alcanzados en todo el mundo, incluida Argentina, por leyes que los obligan a no tener más animales encerrados. Esto implica una refuncionalización de los zoológicos en poco tiempo. En este punto el Laboratorio ve una posible solución en el reemplazo de animales vivos por animatrónico con las tremendas ventajas que este reemplazo tiene.

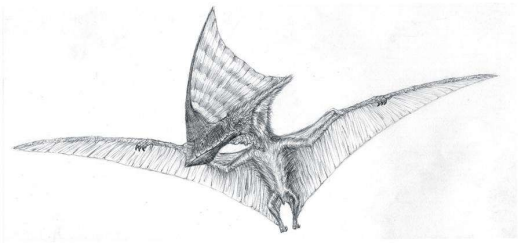


***Figura 1.2.-*** *Representación de tapejara imperatos. Fuente: Santiago Druetta.*

***Figura 1.3.-*** *Foto de Tapejara Imperator en la presentación del Aula Magna de la Facultad de Ciencias Exactas Físicas y Naturales UNC. Fuente: Propia.*

***Figura 1.4.-*** *Foto de cabeza real de Tapejara Imperator. Fuente: Alexander Kellner*

## 1.4 Formulación del problema

La especie representada en forma de robot era llamada originalmente “Tapejara Imperator” que luego cambió su nombre por “Tupandactilus Imperator”. Este reptil volador, ya extinto, del período cretácico (hace 110 millones de años) fue encontrado en la “Chapada do Araripe” en el norte de Brasil; fue estudiado por el paleontólogo Alexander Kellner nacido en Liechtenstein y el brasileño Diogenes de Almeida Campos, ambos paleontólogos trabajaban en el Museo de Rio de janeiro recientemente incendiado casi en su totalidad.

La primera publicación de la especie fue realizada por ambos paleontólogos en el año 1989. Luego los paleontólogos David M. Unwin y David Martill reclasificaron la especie llamándola como la conocemos hoy, “Tupandactilus Imperator”. En la *figura 2.1* se puede ver la rama (clado) en la que está ubicada la especie en cuestión.

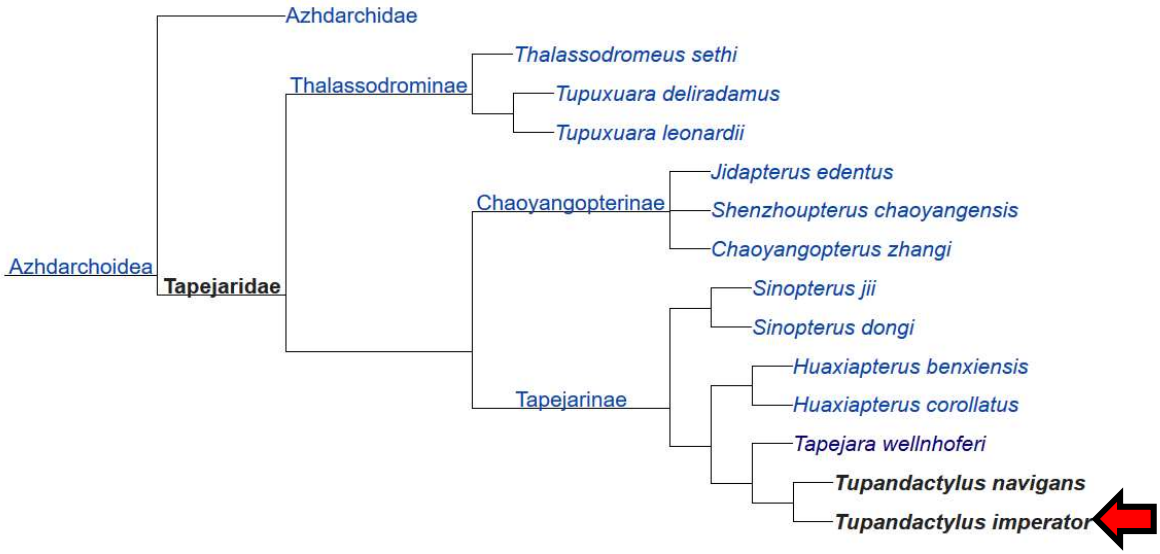



***Figura 1.5.-*** *Imagen familia del reptil volador Tupandactylus.*

## 1.5 Objetivos

### 1.5.1 Objetivos Principales

Se pretende rediseñar reconstruir y controlar un robot tipo reptil volador que representa a una especie Tapejara, ahora renombrado por los científicos con el nombre de “Tupandactylus” (Tapejara Imperator fue un reptil volador que vivió en Brasil hace 108 millones de años) usando tecnología completamente nueva. Como objetivo y criterio de diseño se busca que este animatrónico pueda tener movimientos de cabeza (arriba y abajo) pico (abre y cierra) y de alas (aleteo) con determinada naturalidad y ser controlado de diferentes modos, ya que la finalidad es la exposición en parques y museos temáticos. El robot contará con un modo “rutina” en el cual repetirá una serie de movimientos cada un cierto tiempo, y un modo “controlado” en el cual una persona del público controlará mediante un “dispositivo interactivo” la posición de las alas. Cabe acotar que todo el mecanismo permitirá soportar la morfología de la especie citada en el presente trabajo o en un futuro próximo, siempre cuidando el objetivo del actual proyecto que es el control del animatrónico en sus diversas formas.

### 1.5.2 Objetivos Específicos

- Estabilizar el sistema de lazo cerrado en una sola ala, de modo que el ala tome una serie de ángulos estables provistos por un procesador.
- Probar el sistema en el que los ángulos de entrada son provistos por medio de un dispositivo que controla el usuario (IMU) y chequear que los ángulos de salida del ala sean respetados.
- Montar todo el sistema (alas y cabeza) en un bastidor cuya morfología sea la de la especie en estudio.
- Comprobar el funcionamiento de todo es sistema de control.

## 1.6 Metodología utilizada para lograr los objetivos propuestos

La señal de gobierno de cada ala viene generada por una persona que mueve un sistema de captura de movimiento consistente en unas barras que están solidarias con sendos acelerómetros/giroscopios a los brazos de dicha persona (titiritero) cuyos movimientos se desean imitar; esta señal es proporcional al ángulo de cada ala del animatrónico y debe ser previamente filtrada y procesada por un microcontrolador para ser enviada a los servomotores que mueven las alas del animatrónico.

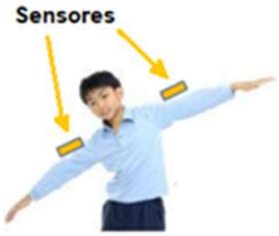



***Figura 1.6.-*** *Movimiento de cada ala generada por una persona. Fuente: Propia*

Como el proyecto está pensado para la exposición en parques y museos, se requieren dos modos de funcionamiento, uno “rutina” y uno “interactivo”.

Cabe aclarar que el animatrónico estará disponible para ser presentado en esta Facultad y a disposición para futuras exposiciones de la Universidad Nacional de Córdoba.

Modo rutina: consta de una rutina precargada con una serie de movimientos de alas, cabeza y pico que se repite cada determinado tiempo. Estas variables son definidas por el programador y deberán ser seleccionadas para cada exposición.

Modo interactivo: Teniendo en cuenta que este animatrónico será exhibido en algún parque o museo, su principal modo de control deberá estar relacionado con las últimas reglas de la divulgación científica en este tipo de demostraciones, teniendo en cuenta esta premisa, se pretende brindar al usuario un dispositivo que permita reproducir los movimientos de dicho usuario en el robot.

# Parte I

# Marco Teórico

# 2. Descripción

En el siguiente capítulo se encuentran todos los contenidos teóricos que fueron necesarios manejar para la realización del proyecto. Se describen software, sensores utilizados, fenómenos físicos y tecnologías utilizadas.

## 2.1 Entorno de programación

A lo largo de estos últimos años, la informática fue avanzado tan rápidamente que creó una variedad de programas según la tarea que se desea realizar, cumpliendo las necesidades del usuario. Estos programas incorporan numerosas herramientas, utilidades y aplicaciones ya desarrolladas que ayudan y facilitan al desarrollo de este.

### 2.1.1 Matlab

MATLAB es un software matemático que está integrado con un lenguaje de programación. Sus prestaciones permiten la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario y la comunicación con programas en otros lenguajes y con otros dispositivos. El paquete MATLAB dispone de herramientas adicionales, a saber, Simulink que es muy utilizado para realizar simulaciones.

### 2.1.2 Simulink

SIMULINK es una herramienta de MATLAB que permite simular el comportamiento de los sistemas dinámicos. Puede simular sistemas lineales y no lineales, modelos en tiempo continuo y tiempo discreto y sistemas híbridos de todos los anteriores. Es un entorno gráfico y se construye arrastrando los diferentes bloques que lo constituyen. Los modelos SIMULINK se guardan en ficheros con extensión model (.mdl).

### 2.1.3 Arduino IDE

Es un entorno de desarrollo integrado de Arduino. Es una aplicación multiplataforma tanto para Windows , macOS o Linux que está escrita en el lenguaje de programación Java . Se utiliza para escribir y cargar programas en la placa Arduino.

El IDE de Arduino suministra bibliotecas de software, que proporciona muchos procedimientos comunes de entrada y salida. El código escrito por el usuario solo requiere dos funciones básicas, para iniciar el boceto y el bucle del programa principal, que se compilan y vinculan con un apéndice de programa. El IDE de Arduino emplea el programa para convertir el código ejecutable en un archivo de texto en codificación hexadecimal que se carga en la placa Arduino mediante un programa de carga en el firmware de la placa.

## 2.2 Tipos de Sensores

Los sensores de posición angular de un eje son componentes fundamentales en la tecnología de control. Utilizando un acoplamiento directo o algún tipo de acoplamiento mecánico que realice la adaptación, un codificador de posición angular se puede utilizar para monitorear cualquier desplazamiento.

Dependiendo de la técnica utilizada, se pueden clasificar de la siguiente manera:

- Sensores de resistencia variable o potenciómetros
- Sensores ópticos
- Sensores inerciales (IMU)

### 2.2.1 Potenciómetros

El elemento sensor es simplemente una resistencia sobre una pista conductora. Dicha resistencia es proporcional a la posición del cuerpo cuyo desplazamiento se va a medir. Se une el objeto en cuestión a un extremo de la pista y se aplica una tensión constante. En función del voltaje de salida obtenido se calcula la distancia recorrida por el objeto.

La pista conductora puede ser lineal o angular dependiendo de los requisitos. Éstas pueden estar hechas de carbono, alambre resistivo o material piezoeléctrico.

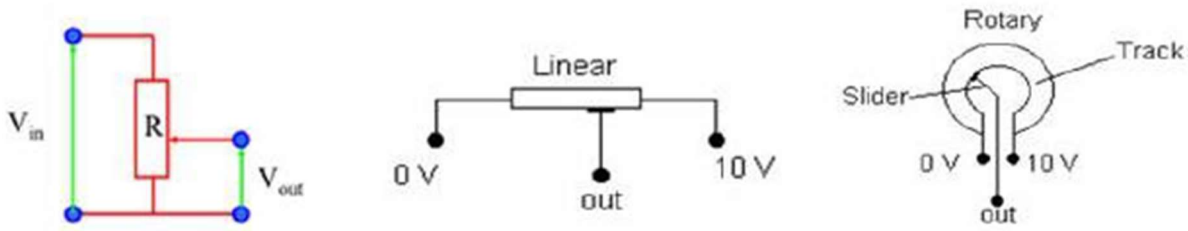



***Figura 2.1.-*** *Tipos de potenciómetros. Fuente: Tesis Vara Rodríguez, David [1]*

### 2.2.2 Sensor Óptico

Los sensores ópticos tienen dos posibles mecanismos:

En el primero, la luz se transmite desde un extremo hasta otro y se monitoriza el cambio en una de sus características: intensidad, longitud de onda, polarización, etc. Se utilizan en codificadores ópticos para proporcionar información acerca de la posición de los actuadores.

Los codificadores ópticos consisten en un disco de vidrio o plástico que gira entre una fuente de luz (LED) y un receptor (fotodetector). El disco está codificado con luz

alterna y sectores oscuros de manera que los pulsos son generados a medida que el disco gira. Basándose en el recuento de los impulsos, y la velocidad del disco, la posición angular se calcula. Para identificar la dirección del movimiento, se utilizan dos fotodetectores. Los codificadores ópticos absolutos tienen un código único que puede ser detectado para cada posición angular.

***Figura 2.2.-*** *Sensor Óptico tipo disco. Fuente: Tesis Vara Rodríguez, David [1]*

En el segundo tipo, la luz transmitida se refleja en el objeto y ésta es la que se monitoriza.

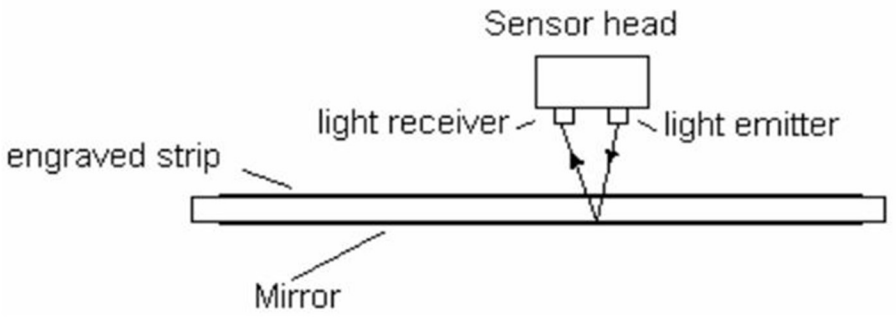



***Figura 2.3.-*** *Sensor Óptico tipo reflector. Fuente: Tesis Vara Rodríguez, David [1]*

### 2.2.3 IMU (Unidad de medida inercial)

La unidad de medición inercial (sus siglas IMU) es un dispositivo electrónico cuyo objetivo es obtener mediciones de velocidad, rotación y fuerzas gravitacionales, usando una combinación de acelerómetros, giróscopos y a veces magnetómetros.

Principio Físico

Acelerómetro: el acelerómetro mide la aceleración angular. Consiste en una placa montada sobre una superficie, y una masa unida a un resorte que se mueve a lo largo de una dirección fija, de modo que cuando se aplica una aceleración con una cierta dirección la masa se moverá y la capacitancia entre la placa y la masa cambiará; este cambio se procesa para obtener un valor particular de aceleración.

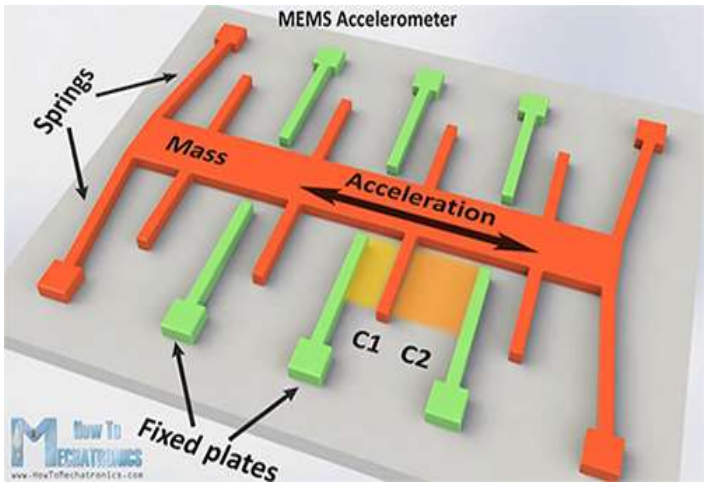



***Figura 2.4.-*** *Funcionamiento del acelerómetro. Fuente: How to mechatronics [2]*

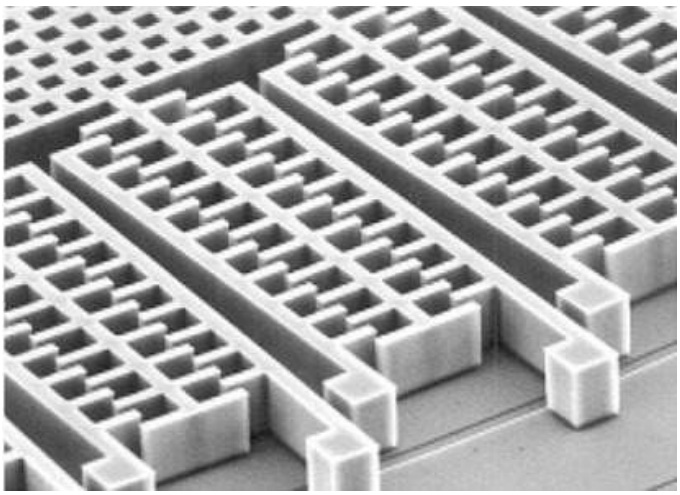


***Figura 2.5.-*** *La microestructura del acelerómetro. Fuente: digikey [3]*

Giroscopio: el giroscopio mide velocidad angular. Cuando la masa se está moviendo en una cierta dirección y con una velocidad particular, al aplicar una velocidad angular como se muestra con la flecha verde de la *figura 2.6*, se producirá una fuerza (flecha roja y azul) que causará el desplazamiento perpendicular de la masa. Al igual que el acelerómetro, este desplazamiento cambia la capacitancia y se procesa para obtener una velocidad angular particular.

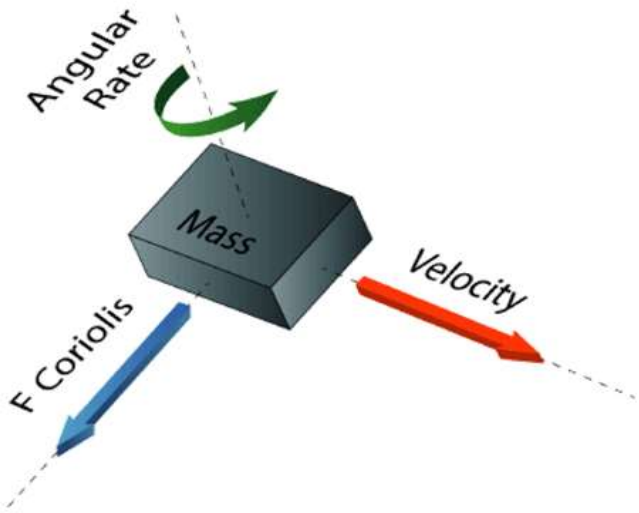



***Figura 2.6.-*** *Funcionamiento del giroscopio. Fuente: How to mechatronics [2]*

La microestructura de un giroscopio consiste en una masa que está en constante movimiento, u oscilando, y cuando se le aplica una velocidad angular externa la parte flexible de la masa se mueve provocando un desplazamiento perpendicular, tal como se ilustra en la *figura 2.7*.

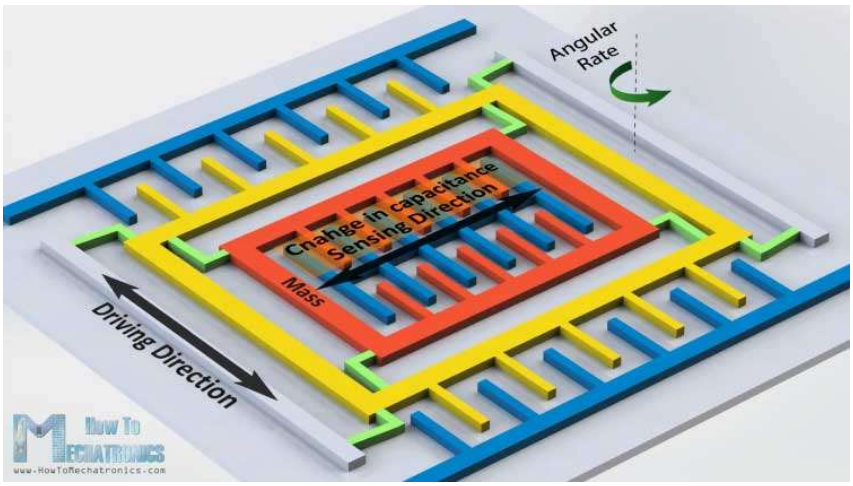



***Figura 2.7.-*** *La microestructura del giroscopio. Fuente: How to mechatronics [2]*

## 2.3 Comunicación I2C

I2C significa Circuito Inter Integrado y es un protocolo de comunicación serial desarrollado por Phillips. Se creó para comunicar varios chips al mismo tiempo dentro de los televisores.

Descripción de las señales

- SCL (System Clock) es la línea del clock utilizada para sincronizar el sistema.
- SDA (System Data) es la línea por la que se mueven los datos entre los dispositivos.
- GND (Ground).

Las líneas SDA y SCL son del tipo drenador abierto debiendo conectar resistores "pull-up", lo que define una estructura de bus que permite conectar en paralelo múltiples entradas y salidas.

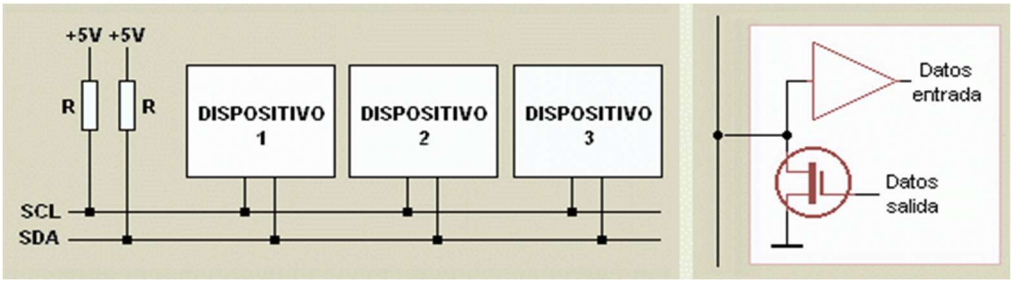



***Figura 2.8.-*** *Funcionamiento protocolo I2C. Fuente: Robots argentina [4]*

### 2.3.1 Modo de comunicación

Existiendo varios dispositivos conectados sobre el bus, es lógico que para establecer una comunicación a través de él se deba respetar un protocolo. Existen dispositivos “maestros**”** y dispositivos “esclavos**”**. Sólo los dispositivos maestros pueden iniciar una comunicación.

La condición inicial, de bus libre, se da cuando ambas señales están en estado lógico alto. El primer byte que se transmite después de la condición “start” contiene siete bits que componen la dirección del dispositivo al que se desea comunicar, y un octavo bit que corresponde a la operación que se quiere realizar con él (lectura o escritura).

Si el dispositivo cuya dirección corresponde a la que se indica en los siete bits (A0-A6) está presente en el bus, éste contesta con un bit en bajo, ubicado inmediatamente luego del octavo bit que ha enviado el dispositivo maestro. Este bit de reconocimiento (ACK) en bajo le indica al dispositivo maestro que el esclavo reconoce

la solicitud y está en condiciones de comunicarse. Aquí la comunicación se establece y comienza el intercambio de información entre los dispositivos.

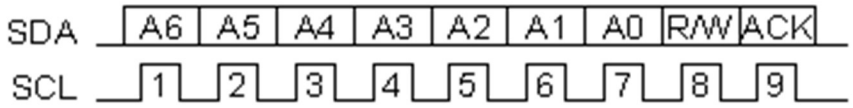



***Figura 2.9.-*** *7 Bits protocolo I2C. Fuente: Robots argentina [4]*

Si el bit de lectura/escritura (R/W) fue puesto en esta comunicación a nivel lógico bajo (escritura), el dispositivo maestro envía datos al dispositivo esclavo. Esto se mantiene mientras continúe recibiendo señales de reconocimiento, y el contacto concluye cuando se hayan transmitido todos los datos.

En el caso contrario, cuando el bit de lectura/escritura estaba a nivel lógico alto (lectura), el dispositivo maestro genera pulsos de reloj para que el dispositivo esclavo pueda enviar los datos. Luego de cada byte recibido el dispositivo maestro (quien está recibiendo los datos) genera un pulso de reconocimiento.

El dispositivo maestro puede dejar libre el bus generando una condición de parada (o detención; stop en inglés).

Si se desea seguir transmitiendo, el dispositivo maestro puede generar otra condición de inicio en lugar de una condición de parada. Esta nueva condición de inicio se denomina "inicio reiterado" y se puede emplear para direccionar un dispositivo esclavo diferente o para alterar el estado del bit de lectura/escritura.

| | |
|---|---|
| Número de vías o cables | 2 |
| Velocidad máxima | Modo estándar (Sm) = 100kbps |
| | Modo rápido (Fm) = 400kbps |
| | Modo High Speed (Fm+) = 3.4Mbps |
| | Modo Ultra Fast (Hs-mode) = 5Mbps |
| Síncrono o Asíncrono | Síncrono |
| Paralelo o Serial | Serial |
| Número máximo de Maestros | Ilimitado |
| Número máximo de Esclavos | 1008 |

***Tabla 2.1.-*** *Características del protocolo I2C. Fuente: T-Bem [5]*

### 2.3.2 Composición de un mensaje enviado


***Figura 2.10.-*** *Composición del mensaje enviado en protocolo I2C. Fuente: T-Bem [5]*

Condición de Inicio – Start: La vía SDA cambia de un nivel de tensión alto a un nivel de tensión Bajo, antes de que el canal SCL cambie de Alto a nivel Bajo.

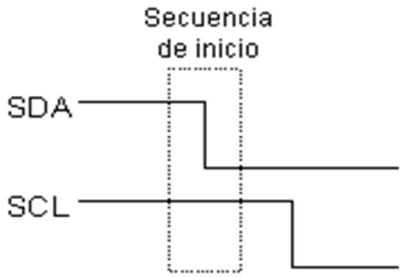



***Figura 2.11.-*** *Condición de inicio del protocolo I2C. Fuente: Robots argentina [4]*

Condición de Paro – Stop: La vía SDA ahora cambia de un nivel de voltaje Bajo a Alto, después de que la vía SCL cambia de Bajo a Alto.

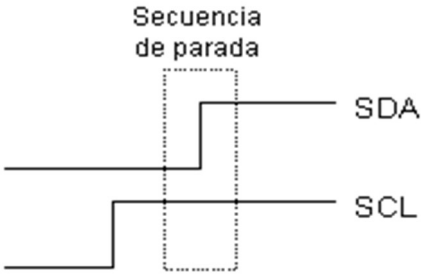



***Figura 2.12.-*** *Condición de paro del protocolo I2C. Fuente: Robots argentina [4]*

Trama de Dirección – Addres Frame: Es una secuencia única que va de los 7 a los 10 bits. Esta sección (Frame) se envía a cada Esclavo, y va a identificar al Esclavo con el que el Maestro se quiere comunicar.

Bit para Lectura/Escritura A – Read/Write Bit A: Es un bit de información enviado a los Esclavos. Por medio de este bit el Maestro indica si le va a enviar información al Esclavo (Nivel Bajo de voltaje = Escritura), o si el Maestro quiere solicitarle información al Esclavo (Nivel Alto de Voltaje = Lectura).

Bit ACK/NACK: Después de cada sección (Frame) de información enviada en un mensaje, podemos notar que lleva un bit acknowledge/no-acknowledge (reconocido/no-reconocido). Esto ayuda a identificar si la información fue enviada correctamente. En seguida de que se envía un Frame, si este fue recibido con éxito, se retorna un bit ACK al remitente. Si la información no fue recibida con éxito, se retorna un bit NACK.

## 2.4 Filtro Digital

Los filtros complementarios son filtros de mucho interés en sistemas de navegación inercial. Algunas aplicaciones son estimación de la velocidad vertical (la combinación de las lecturas de aceleración vertical y velocidad barométrica vertical) o IMUs y sistemas de visión.

Un filtro complementario es en sí un filtro de Kalman de estado estacionario para una cierta clase de problemas de filtrado, este no tiene en cuenta ninguna descripción estadística del ruido que corrompe las señales y es obtenido solamente por un análisis en frecuencia. El filtro complementario puede ser tratado de manera muy simple matemáticamente y es de fácil implementación debido a su bajo consumo computacional.

La idea básica del filtro complementario es combinar la salida del acelerómetro y del giroscopio para obtener una buena estimación del ángulo, compensando el error acumulado del giroscopio con la dinámica lenta del acelerómetro.

El filtro complementario propuesto es el que se encuentra en la *figura 2.13,* donde $\theta a$ es el ángulo medido por el acelerómetro cuya señal está interferida por ruido de frecuencias altas que provienen de vibraciones, $\theta g$ es el ángulo medido por el giroscopio afectado por el error acumulado, y $\hat{\theta}$ es el ángulo estimado.

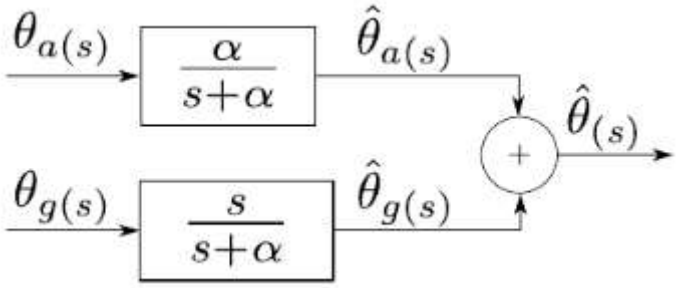



***Figura 2.13.-*** *Filtro Complementario. Fuente: CASE [17]*

Las funciones de transferencia del filtro deben ser elegidas de acuerdo con la ecuación (2.1)

$$H_a(s)\,G(s) + H_g(s)(1 - G(s)) = 1 \qquad (2.1)$$

En donde $H_a$y $H_g$ representan las Funciones de Transferencia del acelerómetro y el giroscopio.

Suponiendo que las funciones de transferencia de los sensores son igual a uno, esto es:

$$H_a(s) = H_g(s) = 1 \tag{2.2}$$

La función de transferencia elegida para $G(s)$ es un pasa bajos de primer orden, lo cual hace que la estimación de baja frecuencia dependa del acelerómetro.

$$G(s) = \frac{\propto}{s+\propto} \tag{2.3}$$

En tanto que la función de transferencia elegida para $1 - G(s)$ es un filtro pasa altos de un polo:

$$1 - G(s) = 1 - \frac{s}{s+\propto} \tag{2.4}$$

Este filtro nos permite que las componentes de alta frecuencia estén incorporadas por las mediciones del giroscopio.

Como se puede ver en la *figura 2.14*, si ambas mediciones son ideales, la función de transferencia final del filtro resulta:

$$G(s) + \big(1 - G(s)\big) = 1 \tag{2.5}$$

Dicho de otra manera, se puede ver en la *figura 2.14*, donde la suma de ambos filtros (pasa bajo y pasa alto) siendo los mismos ideales, se obtiene el ángulo proveniente de las componentes $\theta a$ y $\theta g$.

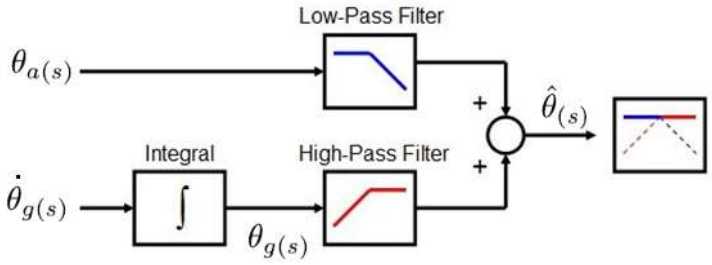



***Figura 2.14.-*** *Filtro Complementario compuesto por Filtro Pasa Bajo y Pasa Alto. Fuente: ASKIX [18]*

Para la implementación de filtros digitales en microcontroladores es necesario discretizar la función de transferencia del filtro, para este caso se hará con un retentor de orden cero a la entrada. De esta manera se obtiene una expresión compacta del filtro.

Para ello, si definimos

$$G_1(s) = G(s)$$

$$G_2(s) = 1 - G(s).$$

$$G_1(z) = \frac{\left(1-e^{-\frac{T}{\tau}}\right)z^{-1}}{1-e^{-\frac{T}{\tau}}z^{-1}} \quad (2.6)$$

$$G_2(z) = \frac{1-z^{-1}}{1-e^{-\frac{T}{\tau}}z^{-1}} \quad (2.7)$$

Donde $\propto = \frac{1}{\tau}$ y $\tau$ representan la constante de ambos filtros.

El algoritmo del filtro en el microcontrolador surge del paso a ecuaciones diferenciales de las funciones de transferencia de cada filtro.

$$\hat{\theta}[k] = \hat{\theta}_a[k] + \hat{\theta}_g\,[k] \quad (2.8)$$

$$\hat{\theta}_a[k] = e^{-\frac{T}{\tau}}\hat{\theta}_a\,[k-1] + \left(1 - e^{-\frac{T}{\tau}}\right)\theta_a[k-1] \quad (2.9)$$

$$\hat{\theta}_g[k] = e^{-\frac{T}{\tau}}\,\hat{\theta}_g\,[k-1] + \theta_g[k] - \theta_g[k-1] \quad (2.10)$$

$$\omega[k]\,T = \theta_g[k] - \theta_g[k-1] \quad (2.11)$$

$$\hat{\theta}_g[k] = e^{-\frac{T}{\tau}}\,\hat{\theta}_g\,[k-1] + \omega[k]\,T \quad (2.12)$$

## 2.5 Bluetooth

Existe una forma de evitar cierto cableado en toda comunicación entre uno o más dispositivos. La comunicación a través de bluetooth sin duda es una de las opciones más utilizadas habitualmente a la hora de comunicarse entre distintos módulos brindando comodidad y seguridad en la comunicación. De esta manera, los usuarios se evitan un dolor de cabeza cuando se trata de conectividad a través de cableados tediosos y que requieren la portabilidad de este.

### 2.5.1 Estándar de Comunicación

La tecnología Bluetooth define un estándar de comunicaciones inalámbricas de corto alcance mediante señales de radiofrecuencia que permite la transmisión de voz y datos, buscando eliminar el cableado de conexiones entre dispositivos electrónicos, manteniendo altos niveles de seguridad. En la actualidad existen varias tecnologías de

corto alcance en la *Tabla 2.2* se muestra la comparación entre las características principales de la tecnología bluetooth sobre las demás.

| Tecnología | Frecuencia | Tasa tx | Alcance | Seguridad | Aplicación |
|---|---|---|---|---|---|
| ANT+ | 2.4GHz | 1Mbps | <10m | - | Monitoreo:salud y deportes |
| Bluetooth | 2.4GHz | 1-54Mbps | 10-100m | Alta con encriptación | Aplicaciones de audio, transferencia datos. |
| Infrared | 800-1000um | 9600bps | 1m | - | Mando a distancia |
| NFC | 13.56MHz | 106-848kbps | <30cm | Alta | Accesos controlados |
| RFID | 125kHz, 13.56MHz, 902 a 928MHz | 424kbps | <1m | Vulnerable | Rastreo, inventarios |
| 6LoWPAN | 2.4GHz | - | <10cm | Alta | Monitoreo y control via internet |
| UWB | 3.1 a 10.6GHz | 53Mbps - 480Mbps | <10m | Alta | Transferencia de video |
| Wi-Fi | 2.4GHz | 54-300Mbps | <100m | Alta con encriptación | Redes locales, acceso a internet |
| Z-Wave | 908.42MHz | 9600bps - 40kbps | <30m | - | Monitoreo y control del hogar |

***Tabla 2.2.-*** *Comparación entre distintos tipos de tecnología de corto alcance.*

Cada una de estas tecnologías fueron creadas con el propósito de cubrir las deficiencias de otras, alguna se desarrolló para aplicaciones específicas, sin embargo, otras son flexibles y genéricas. Es verdad que algunas características se pueden comparar de forma cuantitativa, se debe tomar en cuenta el campo de aplicación y elegir de acuerdo con ésta, cuál de ellas se adapta mejor.

Entre las ventajas de la tecnología Bluetooth sobre las demás tecnologías inalámbricas de corto alcance, se destacan: la confiabilidad en la transmisión de datos, la baja interferencia, alto alcance de transmisión a baja potencia, bajo costo, intercambio de voz y datos, entre otras.

### 2.5.2 Topología

Cuando un dispositivo Bluetooth se encuentra dentro del radio de cobertura de otro, puede establecer un enlace entre ellos. Hasta ocho unidades pueden comunicarse entre ellas y formar una Piconet. La unión de varias Piconets se denomina Scatternet o Red Dispersa.

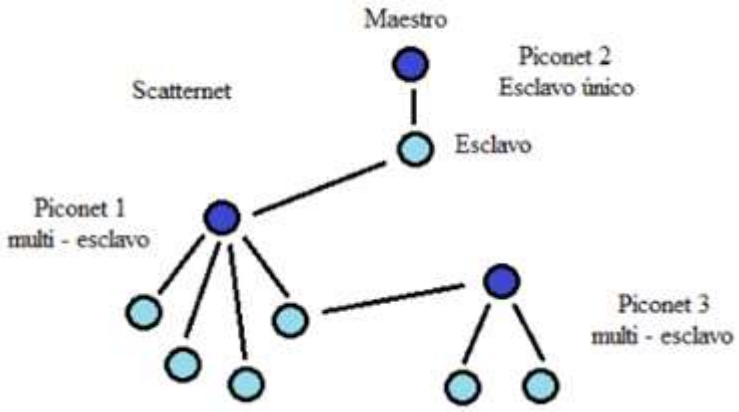



***Figura 2.15.-*** *Enlace de Scatternet entre distintos Piconet. Fuente: Moreno, Alberto "Estandar Bluethoot" [9]*

Los dispositivos que están dentro de una piconet pueden ser: maestros o esclavos. En una piconet solo puede existir un maestro, quien normalmente inicia la conexión, el resto de las unidades son esclavos. Cualquier dispositivo puede hacer las veces de maestro o esclavo.

El maestro es el dispositivo de una piconet cuyo reloj y patrón de saltos se utilizan para sincronizar a los demás integrantes.

### 2.5.3 Especificaciones técnicas

Bluetooth trabaja en una frecuencia de radio situada en el rango de 2.4 a 2.48GHz de la banda ISM (Industrial, Scientific and Medical - Banda libre sin licencia industrial científica y médica) disponible a nivel mundial, lo que significa una compatibilidad universal entre dispositivos BT.

Los dispositivos BT se clasifican por clases de acuerdo con su potencia de transmisión, sin perder la compatibilidad entre ellos cualquiera sea la clase.

| Clase | Potencia máxima permitida(mW, dBm) | Potencia mínima (mW,dBm) | Alcance (m) |
|---|---|---|---|
| Clase 1 | 100, 20 | 1, 0 | 100 |
| Clase 2 | 2,5, 4 | 0.25, -6 | 10 |
| Clase 3 | 1, 0 | N/A | 1 |

***Tabla 2.3.-*** *Diferencias entre la clase según la potencia. Fuente: Barriaga, Walter y Zuñiga, Fabian [10].*

## 2.6 Modulo Bluetooth HC-05

El hardware que compone el dispositivo Bluetooth está compuesto por dos partes:

- un dispositivo de radio, encargado de modular y transmitir la señal.
- un controlador digital, compuesto por una CPU, un procesador de señales digitales (DSP – Digital Signal Processor) llamado Link Controller (o controlador de Enlace) y de las interfaces con el dispositivo anfitrión.

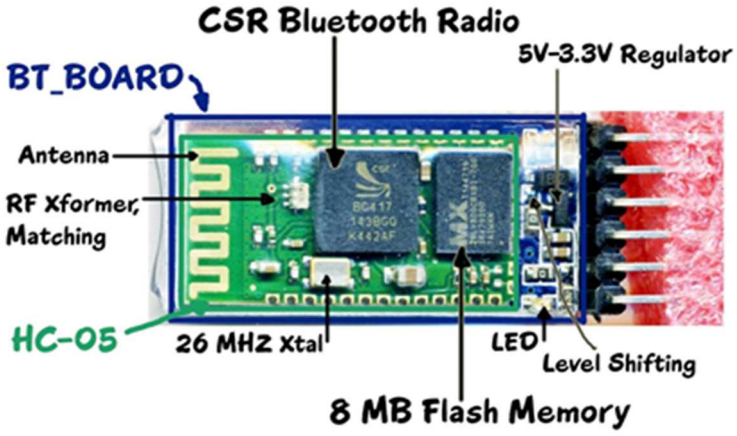



***Figura 2.16.-*** *Partes del módulo bluethoot HC-05. Fuente: Altronics [13]*

El Link Controller se encarga de las funciones de transferencia tanto asíncrona como síncrona, la codificación de audio y el cifrado de datos.

Los dispositivos Bluetooth pueden actuar como Maestros o como Esclavos. La diferencia es que un esclavo solo puede conectarse a un maestro y a nadie más, en cambio un maestro, puede conectarse a varios esclavos o permitir que ellos se conecten (hasta un máximo de 7 esclavos) y recibir y solicitar información de todos ellos, arbitrando las transferencias de información.

Cada uno de los dispositivos que se identifican vía bluetooth presentan una dirección única de 48 bits y además un nombre de dispositivo que sirve para identificarlo.

Así un módulo bluetooth puede ser maestro o esclavo y dispone de una dirección única, y también incluye una contraseña de conexión o número de identificación para obtener acceso al mismo. Cuando se vinculan dos dispositivos bluethoot, se inicia un proceso en el que ellos se identifican por nombre y dirección interna y se solicitan la contraseña para autorizar la conexión.

### 2.6.1 Comunicación entre la PC y el módulo de forma indirecta a través de un Arduino

El esquema principal para la configuración del módulo se basa en la realización de un programa de tal manera que pueda comunicarse vía comunicación serie, desde el módulo con la placa y la placa con la PC, como se muestra en el siguiente esquema.


***Figura 2.17.-*** *Comunicación bluethoot – monitor serial. Fuente: Propia*

## 2.7 Servomotor

El servo está conformado por un motor de corriente continua, una caja reductora y un circuito de control, tiene la capacidad de ubicarse en cualquier posición dentro de un rango de operación (generalmente de 0° a 180°) y de mantenerse estable en dicha posición.

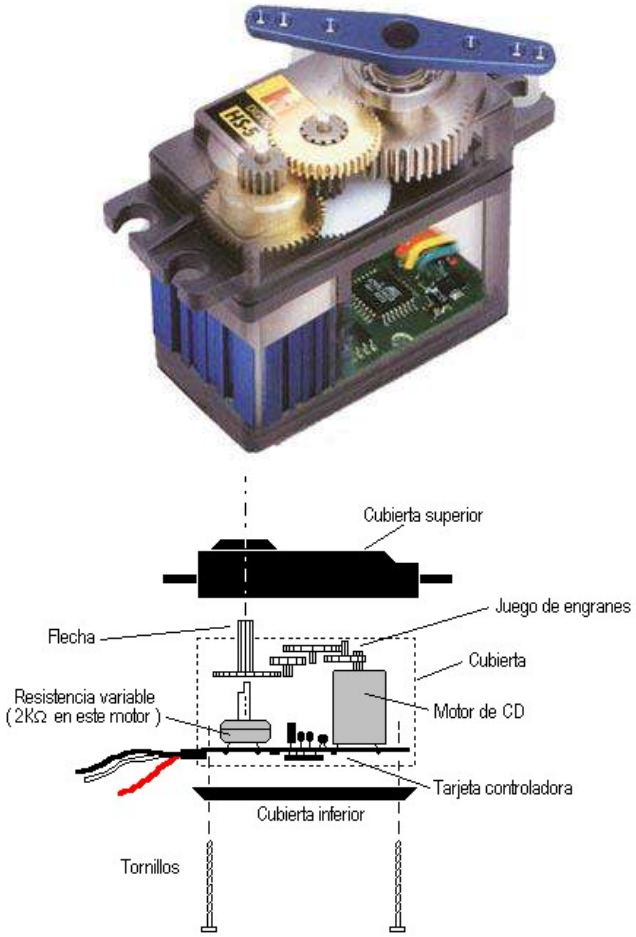



***Figura 2.18.-*** *Composición de un servomotor. Fuente: Monografía [14]*

El dispositivo utiliza un circuito de control para realizar la ubicación del motor en un punto, consiste en un controlador proporcional por ancho del pulso de la señal de entrada, es decir, una señal con pulsos más anchos (mayor duración) ubicará al motor en un ángulo mayor, y viceversa.

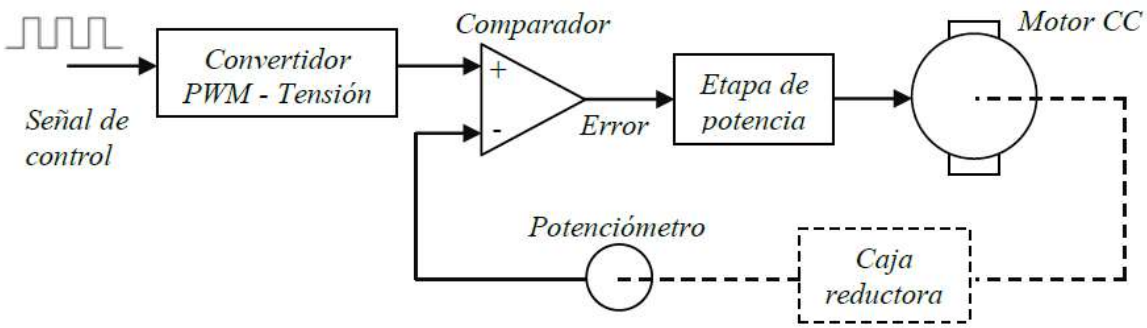



***Figura 2.19.-*** *Partes de un servomotor. Fuente: Propia*

La señal de control PWM se convierte en un valor analógico de tensión, mediante un convertidor PWM-Tensión. El valor de la posición del motor se obtiene usando un potenciómetro de realimentación acoplado solidariamente en el eje final de la caja reductora de modo que, cuando el motor gire, el potenciómetro también lo hará, variando el voltaje que se introduce al comparador, donde éste calcula el valor del error de posición, que es la diferencia entre la referencia y la posición en que se encuentra el motor medida con el potenciómetro. Una vez que se ha obtenido el error de posición, éste se amplifica con una ganancia, y posteriormente se aplica a los terminales del motor. Si el servo se encuentra en la posición deseada, el error será cero, y no habrá movimiento.

Normalmente los valores de tiempo de la señal en alto están entre 0,5 y 2,5 ms, que posicionan al motor en ambos extremos de giro (0° y 180°, respectivamente). El tiempo en el que la salida se mantiene en alto para ubicar la posición del servo, se determina mediante una relación completamente lineal; para un tiempo en alto de 1,5 ms se obtiene la posición central, y otros valores de duración del pulso dejarían al motor en la posición proporcional a dicha duración.

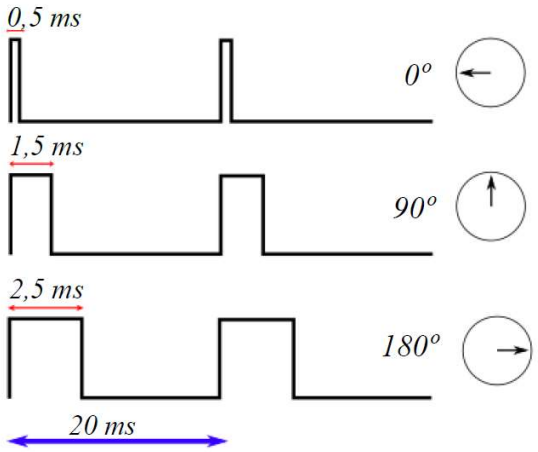



***Figura 2.20.-*** *Ángulos según el tiempo del ancho de pulso del pwm. Fuente: Wikipedia [15]*

La duración del pulso en alto para conseguir un ángulo α deseada, estará dada por $t = 0.5 + \frac{\alpha}{90}$ donde t está dado en milisegundos y en α en grados. Sin embargo, debe tenerse en cuenta que ningún valor debe estar fuera del rango de operación del dispositivo (2ms) ya que el servo tiene una limitación física impuesta por el potenciómetro del control de posición.

Para bloquear el servomotor en una posición, es necesario enviarle continuamente la señal con la posición deseada. De esta forma, el sistema de control seguirá operando, y el servo conservará su posición y se resistirá a fuerzas externas que intenten modificarlo. Si los pulsos no son enviados, el servomotor quedará liberado y cualquier fuerza externa podrá modificar su posición.

## 2.8 Puente H

Un Puente en H es un circuito electrónico que se usa para permitir a un motor eléctrico DC girar en ambos sentidos, o sea en avance y retroceso. Son ampliamente usados en robótica. Los puentes H están disponibles como circuitos integrados, pero también pueden construirse a partir de componentes discretos.

El término "puente H" proviene de la típica representación gráfica del circuito. Un puente H se construye con cuatro interruptores (generalmente mediante transistores). Para entender su funcionamiento, se observa en la *figura 2.21* la disposición de los cuatro interruptores S1, S2, S3 y S4.

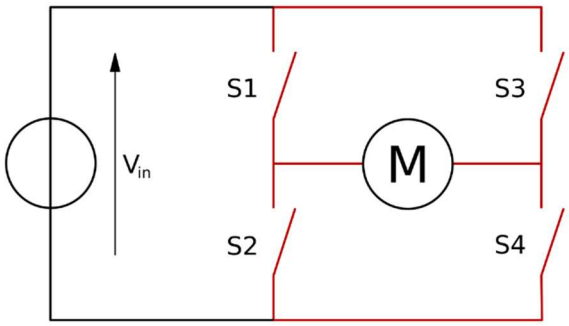



***Figura 2.21.-*** *Puente H. Fuente: Wikipedia [19]*

Cuando los interruptores S1 y S4 están cerrados y los S2 y S3 abiertos, *figura 2.22*, se aplica una tensión positiva en el motor, haciéndolo girar en un sentido. Abriendo los interruptores S1 y S4 y cerrando S2 y S3, el voltaje se invierte, permitiendo el giro en sentido inverso del motor, *figura 2.23*.

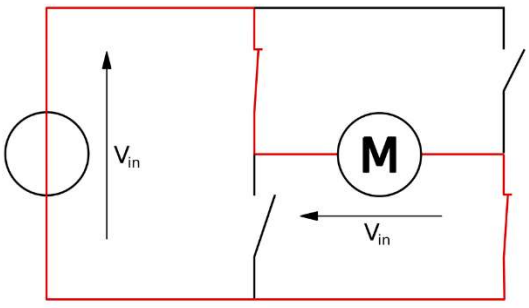



***Figura 2.22.-****Transistores S1 y S4 cerrados, el motor gira en un sentido. Fuente: Wikipedia [19]*

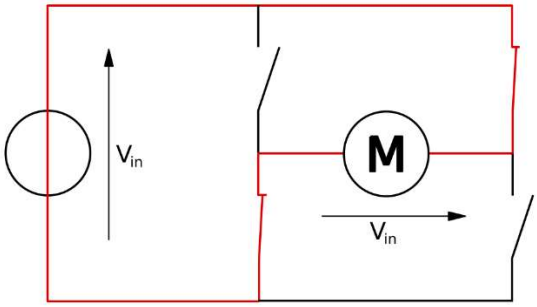



***Figura 2.23.-*** *Transistores S2 y S3 cerrados, el motor gira en el sentido inverso. Fuente: Wikipedia [19]*

Con la nomenclatura que estamos usando, los interruptores S1 y S2 nunca podrán estar cerrados al mismo tiempo, porque esto cortocircuitaría la fuente de tensión. Lo mismo sucede con S3 y S4.

Si bien un puente H se utiliza para mover un motor, en un sentido o en otro, también se lo utiliza para frenar dicho movimiento y esta frenada puede ser brusca (al hacer un corto en los bornes del motor) o frenar bajo su propia inercia (cuando desconectamos el motor de la fuente que lo alimenta).

| S1 | S2 | S3 | S4 | Resultado |
|---|---|---|---|---|
| 1 | 0 | 0 | 1 | El motor gira en *avance* |
| 0 | 1 | 1 | 0 | El motor gira en *retroceso* |
| 0 | 0 | 0 | 0 | El motor se detiene bajo su inercia |
| 1 | 0 | 1 | 0 | El motor frena (*fast-stop*) |
| 0 | 1 | 0 | 1 | El motor frena (*fast-stop*) |
| 1 | 1 | 0 | 0 | Corto circuito |
| 0 | 0 | 1 | 1 | Corto circuito |
| 1 | 1 | 1 | 1 | Corto circuito |

***Tabla 2.4.-*** *Combinaciones de los interruptores en un puente H. Fuente: Propia*

## 2.9 Conversor DC/DC

Cuando se trabaja con dispositivos electrónicos en un proyecto, es común que los mismo que traten con distintos rangos de alimentación (pudiendo ser 12v, 24v, 5v, 3.3v, etc). Una manera de cumplir con los requisitos de alimentación de cada dispositivo en cuestión es a través de conversores DC/DC que trabajan con una alimentación de entrada única pudiendo elegir una tensión mayor deseada.

### 2.9.1 Reguladores lineales

Los reguladores de voltaje lineales entregan una tensión de salida estable desde una fuente de alimentación más o menos estable. Esto indica que el regulador lineal es

capaz de filtrar de manera efectiva variaciones en la entrada, pero como limitación se obtiene un bajo tiempo de respuesta debido a la realimentación interna del amplificador de error.

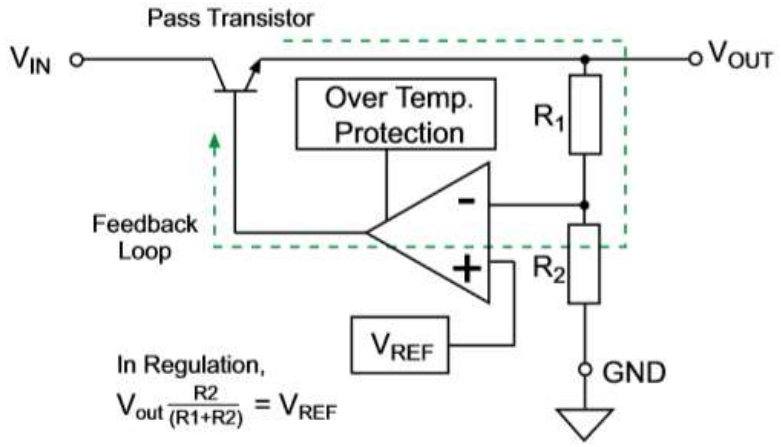



***Figura 2.24.-*** *Regulador de tensión lineal básico. Fuente: Ebook Recom [16]*

La mayoría de los reguladores lineales tienen un control de lazo cerrado como muestra la *figura 2.24* que muestra un tipo de regulación. El transistor de paso es el elemento regulador, y un divisor resistivo es calculado para obtener la tensión de salida.

Si la tensión a la salida se eleva por una reducción de la carga o por un incremento en la tensión de entrada, el valor a la entrada no inversora del amplificador de error será mayor que *Vref* y la salida del amplificador es menor reduciendo la excitación del transistor de paso y disminuyendo la tensión de salida. Contrariamente si la carga disminuye o hay una baja de la tensión de entrada, la entrada inversora será menor que *Vref* y la salida del amplificador aumentará la excitación del transistor de paso aumentando la tensión a la salida del regulador. Un factor muy importante para tener en cuenta es la estabilidad de *Vref* ante variaciones de temperatura para obtener una tensión de salida estable y precisa. Con un buen diseño de PCB se pueden obtener resultados de un ripple de 50uVpp a la salida.

### 2.9.2 Eficiencia de los reguladores lineales

La eficiencia de los reguladores lineales está definida por la relación entre la potencia de salida sobre la potencia de entrada como se muestra en la formula (2.13).

$$\eta = \frac{P_{OUT}}{P_{IN}} \tag{2.13}$$

Donde:

$$P_{OUT} = V_{OUT}\, I_{OUT} \tag{2.14}$$

$$P_{IN} = V_{IN}\, I_{IN} \tag{2.15}$$

$$I_{IN} = I_{OUT} + I_Q \tag{2.16}$$

Donde $I_Q$ es la corriente del regulador lineal en la condición inactivo, es decir sin carga. Si se reemplaza a la ecuación del rendimiento se tiene:

$$\eta = \frac{V_{OUT}\, I_{OUT}}{V_{IN}\,(I_{OUT} + I_Q)} \tag{2.17}$$

De esta ecuación se desprende que la eficiencia es variable y depende tanto de la tensión de entrada como de la carga conectada.

### 2.9.3 Otras propiedades de los reguladores lineales

Los reguladores lineales tienen una serie de ventajas considerables, pero también algunas desventajas que requieren un cuidado especial a la hora de utilizarlos.

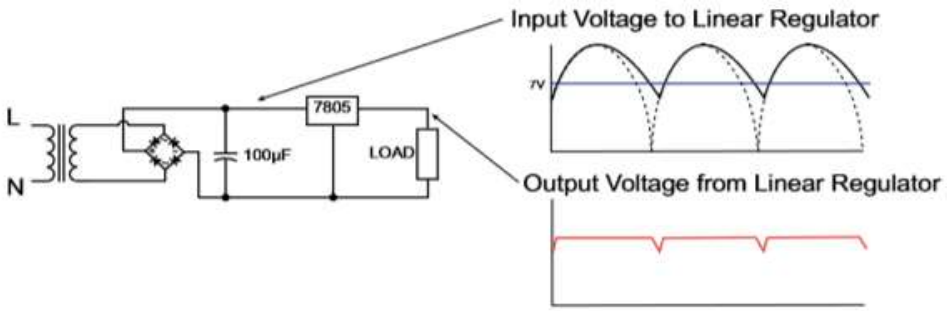



***Figura 2.25.-*** *Recorte de tensión en el regulador de tensión. Fuente Ebook Recom [16]*

Como fue mencionado antes, si la diferencia entre la entrada y la salida está por debajo del mínimo requerido (normalmente 2v), el lazo de control no puede funcionar correctamente. Un problema frecuente se da cuando un rectificador de alterna posee un ripple muy elevado debido al valor pequeño del capacitor de filtrado en la entrada. Si la tensión de entrada cae por debajo del mínimo necesario, la salida del regulador mostrará caídas periódicas al doble de la frecuencia de la red. Estas caídas de tensión a la salida no se pueden medir con un multímetro que mide valor medio, y al mismo tiempo pueden afectar el funcionamiento de distintos integrados.

### 2.9.4 Reguladores switching

En contraste con los reguladores lineales, que disipan el exceso de potencia en forma de calor para limitar la tensión de salida, los reguladores switching aprovechan la propiedad de almacenamiento de energía de los inductores y capacitores para transmitir paquetes de energía discretos. La energía se almacena en el campo magnético (en los inductores) y en el campo eléctrico (en los capacitores) y el controlador de conmutación se encarga de que sólo se transfiera la energía necesaria para la carga haciendo que la

eficiencia del sistema sea muy elevada. A continuación, en la *figura 2.26* se muestra un esquema básico de un regulador de conmutación.

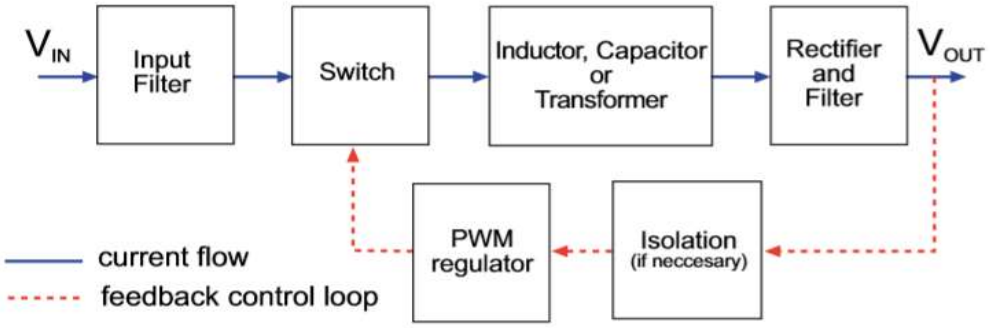



***Figura 2.26.-*** *Recorte de tensión en el regulador de tensión. Fuente: Recom [21]*

Para transferir la energía de la entrada a la salida de manera controlada se necesita de una técnica más compleja que la del regulador lineal. El tipo más utilizado es el PWM (Modulación por Ancho de Pulso), donde la cantidad de energía entregada es modulada por un tiempo variable dentro de un período fijo. La relación entre el tiempo de encendido sobre el período T se llama Duty Cycle (ciclo de trabajo, $\delta$).

$$\delta = \frac{T_{on}}{T} \tag{2.18}$$

Donde:

$$T = \frac{1}{f_{osc}} \tag{2.19}$$

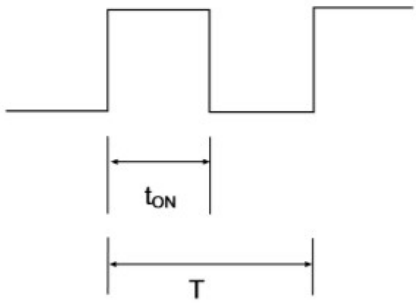



***Figura 2.27.-*** *Duración de pulso y frecuencia en un ciclo de trabajo. Fuente: Recom [21]*

Para muchos reguladores conmutados, la tensión de salida es proporcional al ciclo de trabajo. El lazo de control utiliza el ciclo de trabajo para controlar el elemento de conmutación, esto resulta una ventaja ya que las pérdidas en este tipo de reguladores se dan en los tiempos de cambio de estado en el dispositivo de conmutación, a diferencia de los reguladores lineales donde las pérdidas se dan de manera continua.

### 2.9.5 Topologías de reguladores conmutados

El término topología se refiere a las diferentes combinaciones de elementos de conmutación y almacenamiento de energía que son necesarios para la transmisión, control y regulación de un voltaje o corriente de salida desde una fuente de voltaje de entrada.

Las muchas topologías diferentes para los reguladores de conmutación se pueden dividir en dos grupos principales:

- Convertidores no aislados, en los que la fuente de entrada y la carga de salida comparten una ruta de corriente común durante la operación
- Convertidores aislados, en los que la energía se transfiere a través de componentes magnéticos (transformadores) acoplados mutuamente, en donde el acoplamiento entre el suministro y la carga se logra únicamente a través de un campo electromagnético, lo que permite el aislamiento galvánico entre la entrada y la salida.

#### 2.9.5.1 Convertidores no aislados

La selección de los convertidores se basa en considerar variables como el costo, rendimiento, y características de control que generalmente están determinadas por los requisitos de la aplicación. Ninguna topología es mejor o peor que otra, sino que dependiendo de la necesidad se aprovechan algunas ventajas de algunos tipos frente a otros.

Para los convertidores no aislados existen cinco topologías básicas:

- Buck or step-down converter (reductor).
- Boost or step-up converter (elevador)
- Buck-boost or step-up-down converter (elevador-reductor).
- Two-stage Inverting Buck-boost (Ćuk elevador-reductor con salida invertida)
- Two stage non-inverting Buck-boost (conversor Sepic, conversor ZETA)

##### Boost converter

En este convertidor la energía que produce la fuente primaria es conducida por el elemento de conmutación para ser almacenada en la bobina. Este almacenamiento de energía se efectúa durante el período de conducción sin que exista transferencia de energía a la carga. Cuando el conmutador se abre, la tensión que se genera en la bobina se suma a la tensión de la fuente obteniéndose una tensión de salida mayor a la tensión de entrada y con la misma polaridad.

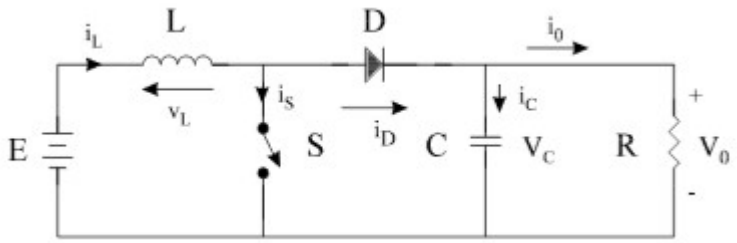



***Figura 2.28.-*** *Esquema básico del funcionamiento de un Boost Converter. Fuente: Recom [21]*

El esquema básico de este convertidor es el de la *figura 2.29 y 2.30,* en la que se refleja sus dos posibles estados. En el primer estado, (en la *figura 2.29* para 0 < t < Ton), el conmutador se encuentra cerrado, y solo existe circulación de corriente a través de la bobina, ya que el diodo se encuentra polarizado inversamente. A lo largo de este intervalo se produce un almacenamiento de energía en la bobina. Por otro lado, cuando el interruptor se abre *figura 2.30,* se produce una inversión de la polaridad en la bobina, que por su naturaleza física impide variaciones bruscas de corriente. En esta condición la bobina actúa como generador y se suma a la tensión de la fuente. Se agrega además un capacitor de filtro para mantener la tensión y la corriente en la carga durante el tiempo que la bobina no entrega energía.

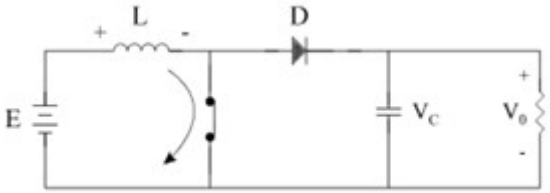



***Figura 2.29.-*** *Interruptor cerrado en el instante 0 < t < Ton. Fuente: Recom [21]*

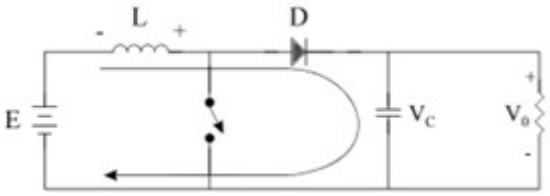



***Figura 2.30.-*** *Interruptor abierto en el instante Ton < t < T. Fuente: Recom [21]*

## Cálculo de componentes y análisis de ecuaciones

A continuación, se realiza un análisis de este modelo considerando que los componentes son ideales, evitando pérdidas en los mismos. Adicionalmente se los considera independiente de la frecuencia y lineales para los elementos pasivos, mientras que los elementos de conmutación como el transistor y el diodo son considerados ideales sin efectos inductivos o capacitivos para no tener en cuenta las pérdidas y así simplificar el análisis.

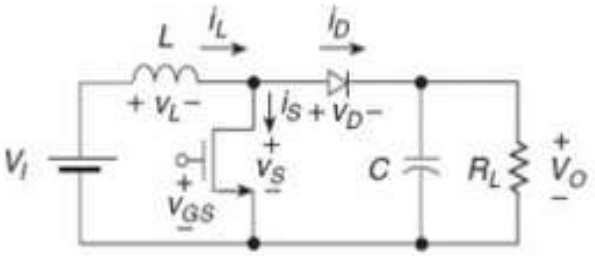



***Figura 2.31.-*** *Convertidor Boost con transistor Mosfet. Fuente: Recom [21]*

Se divide la operación del circuito en base al ciclo útil de trabajo.

- En el ciclo 0 < t < DT

Se comienza el análisis cuando el transistor entra en estado de encendido. El circuito respectivo puede apreciarse en la *figura 2.32,* reemplazando los elementos de conmutación por interruptores ideales.

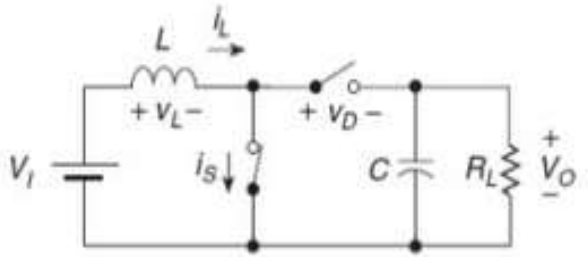



***Figura 2.32.-*** *Interruptor abierto en diodo y cerrado en el transistor. Fuente: Recom [21]*

Se ve que el diodo está polarizado en sentido inverso debido a que la tensión sobre el mismo es igual a -$V_o$ por lo cual no entra en conducción. La tensión existente sobre el transistor y la corriente del diodo son iguales a cero.

El voltaje a través del inductor es igual a $V_L = V_I = L(\frac{di_L}{dt})$. Se ve que la corriente suministrada al transistor es igual a la corriente que circula por el inductor y a la corriente de entrada, por lo que integrando la expresión anterior tenemos que la corriente sobre los elementos es igual a:

$$i_I = i_s = i_L = \frac{1}{L}\int_0^{DT} V_i\, dt \; + i_L(0) \qquad (2.20)$$

Donde $i_L(0)$ es el valor de la corriente en el inductor en el inicio del ciclo de encendido. De la expresión anterior el valor pico de corriente sobre el inductor en el intervalo es:

$$i_{Lpico}(DT) = \frac{V_i\, DT}{L} + i_L(0) \qquad (2.21)$$

La variación de corriente en el inductor o el valor pico a pico de la corriente se calcula por medio de la expresión:

$$\Delta i_L = i_L(DT) - i_L(0) = \frac{V_i\, DT}{L} = \frac{V_i\, D}{L\, f_x} \qquad (2.22)$$

Donde $f_x$ es la frecuencia de conmutación del transistor. Finalmente, la corriente sobre el condensador de salida es igual a $i_c = \frac{-V_0}{R}$.

- En el ciclo DT < t < T

Para este ciclo de trabajo el circuito equivalente se muestra en la *figura 2.33*, donde el transistor se encuentra en estado de apagado y el diodo ha comenzado a conducir debido a que el inductor se comporta como una fuente de corriente y porque ha cambiado su polaridad, haciendo que el terminal del ánodo sea más positivo con respecto al cátodo:

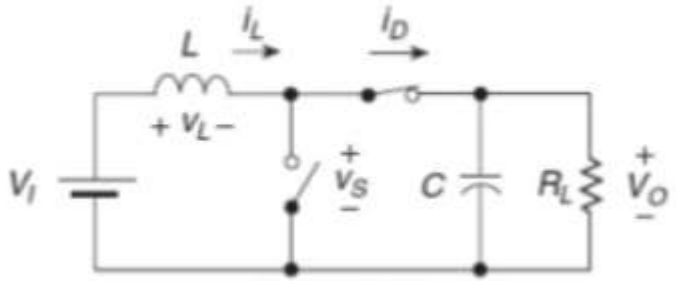



***Figura 2.33.-*** *Interruptor cerrado en diodo y abierto en el transistor. Fuente: Recom [21]*

Durante este tiempo, la corriente en el transistor y la tensión del diodo son iguales a cero. El valor de tensión sobre el inductor es igual a $V_L = V_I - V_0 = L\frac{di_L}{dt} < 0$. Lo que indica que la tensión de salida es mayor que la de entrada. En este caso la corriente que circula por el diodo es igual a la corriente del inductor y la misma entregada a la carga, por lo que integrando la expresión anterior se encuentra que la corriente en estos elementos es:

$$i_I = i_D = i_L = \frac{1}{L}\int_{DT}^{T} V_I - V_0\, dt \; + i_L(DT) = \frac{V_I - V_0(1-D)T}{L} + i_L(DT) \qquad (2.23)$$

$$i_I = i_D = i_L = \frac{V_I - V_0(1-D)}{L\, f_x} + i_L(DT) \qquad (2.24)$$

Donde $i_L(DT)$ es la corriente inicial del inductor en el valor de t=DT. El valor de la corriente pico en el inductor es igual a:

$$\Delta i_L = i_L(DT) - i_L(T) = \frac{V_I - V_0(1-D)}{L\, f_x} \qquad (2.25)$$

El voltaje a través del transistor es igual a la tensión de salida es decir $V_S = V_0$. La corriente en el condensador es igual a $i_c = i_L - i_R$. Finalmente nos disponemos a

calcular la función de transferencia del circuito, para lo cual usaremos el concepto de que la variación de corriente sobre el inductor en un ciclo de operación debe ser cero, es decir:

$$\frac{V_i\, D}{L\, f_x} + \frac{V_I - V_0(1-D)}{L\, f_x} = 0 \rightarrow \frac{V_0}{V_I} = \frac{1}{1-D} \tag{2.26}$$

De acuerdo con lo anterior, este convertidor produce una tensión de salida mayor o igual a la entrada. Sin embargo, se observa que cuando el ciclo de trabajo se acerca al valor de uno, la salida tiende a alcanzar un valor infinito. En la práctica esto no es posible ya que, al trabajar con elementos reales, se generan pérdidas que evitan que se obtenga una tensión tan elevada. La corriente en el inductor, diodo y capacitor se pueden apreciar en la *figura 2.34 a), 2.34 b) y 2.34 c).*

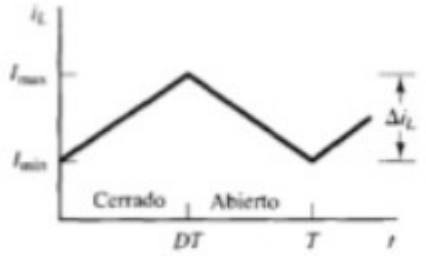



***Figura 2.34 a).-*** *Gráfico corriente en el inductor. Fuente: Recom [21]*

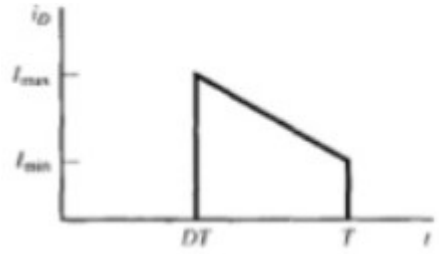



***Figura 2.34 b).-*** *Gráfico corriente en el diodo. Fuente: Recom [21]*

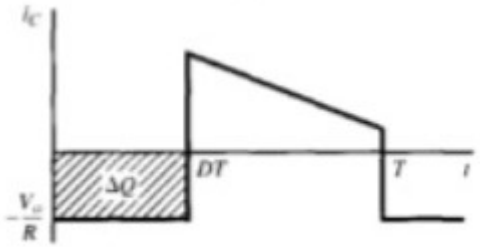



***Figura 2.34 c).-*** *Gráfico corriente en el capacitor. Fuente: Recom [21]*

Podemos calcular el valor medio de la corriente en el inductor, para lo cual supondremos que la potencia entregada a la carga es igual que la potencia suministrada por la fuente, es decir $P_I = P_0$en el caso ideal. La potencia de salida es igual a:

$$P_0 = \frac{{V_0}^2}{R} \tag{2.27}$$

Mientras que la potencia de entrada es $P_I = V_I\, I_I\ = V_I\, I_L$, recordando que la corriente sobre el inductor es la misma corriente que entrega la fuente. Igualando las potencias se obtiene:

$$\frac{{V_0}^2}{R} = V_I\, I_L \rightarrow I_L = \frac{{V_0}^2}{R\, V_I} = \frac{(\frac{V_I}{1-D})^2}{R\, V_I} = \frac{V_I}{R(1-D)^2} \tag{2.28}$$

Y los valores máximos y mínimos en el inductor son:

$$I_{LMAX} = I_L + \frac{\Delta i_L}{2} \tag{2.29}$$

$$I_{LMIN} = I_L - \frac{\Delta i_L}{2} \tag{2.30}$$

Para asegurar que la corriente del inductor opere en modo continuo, el mínimo valor de la corriente debe ser igual a cero. Tomando la expresión anterior para la mínima corriente del inductor e igualando a cero, se calcula el mínimo valor del inductor para operar en modo continuo:

$$I_L - \frac{\Delta i_L}{2} = 0 \rightarrow \frac{V_I}{R(1-D)^2} = \frac{V_i\, D}{2\, L_{MIN}\ f_x} \rightarrow L_{MIN} = \frac{RD(1-D)^2}{2\, f_x} \tag{2.31}$$

De la *figura 2.34 c)*, se calcula el valor del condensador. En la práctica el rizado que posee el condensador debe ser una variación o valor pico a pico AC de la tensión de salida muy pequeño, por lo cual se consideraría que la tensión obtenida es casi constante. Como vemos, la forma de onda de la corriente del condensador posee un área rectangular, lo cual facilita los cálculos del condensador, ya que la variación de carga almacenada o entregada en un periodo de acuerdo con el área de la gráfica es igual a:

$$|\Delta Q| = \frac{V_0\, DT}{R} = C\, \Delta V_0 \tag{2.32}$$

Por lo que el valor del condensador con un determinado rizo y una tensión de salida a una frecuencia de conmutación $f_x$ es:

$$C = \frac{V_0\, D}{f_X\, R\, \Delta V_0} \tag{2.33}$$

Desventajas:

Este convertidor posee una baja capacidad para evitar fallas y transitorios peligrosos. Si existe una sobre tensión a la entrada del convertidor que supere la tensión

de salida, puede ocasionar que el diodo conduzca durante muchos ciclos debido a los saltos del ciclo, haciendo que circule grandes picos de corriente a través del diodo llevándolo a su destrucción. Un problema similar puede ocurrir cuando el convertidor trabaja por primera vez. Para solucionar esto, la mejor manera es colocar un diodo que sirva de protección, ubicando su ánodo en la terminal positiva en la fuente y su cátodo en la salida del filtro capacitivo.

Adicionalmente, la corriente pico a pico a través del diodo y del capacitor son un poco altas.

Ventajas:

La forma de onda de la corriente de entrada es continua no pulsante. Adicional a eso, el transistor MOSFET puede ser disparado de manera sencilla debido a que la compuerta de éste esta referenciada a la tierra del circuito. Para ciclos útiles menores a uno, la eficiencia de un circuito real se puede aproximar al modelo ideal, pero se debe evitar que el ciclo útil este cercano a la unidad porque la eficiencia es considerada muy pobre.

Algo que hay que considerar, es que los diodos en polarización directa poseen una tensión de compensación o de offset que puede llegar a ser un poco alta e influir en la eficiencia, por lo que en ocasiones es necesario reemplazarlo por un transistor MOSFET que no posea este tipo de offset, y el cual tenga una resistencia de encendido para que las perdidas por conducción sean bajas y aumentando la eficiencia. Sin embargo, es necesario considerar el efecto del diodo del cuerpo del transistor porque puede originar que el convertidor sea bidireccional. Adicionalmente, uno de los transistores no está referenciado a tierra, complicando su disparo. El circuito es mostrado en la *figura 2.35*.

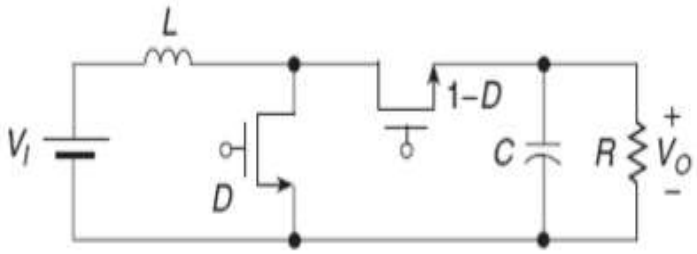



***Figura 2.35.-*** *Reemplazo de diodo por un transistor Mosfet. Fuente: Recom [21]*

## 2.10 Control, el análisis de los procesadores

Un microcontrolador (abreviado µC, UC o MCU) es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Características:

- Velocidad del reloj u oscilador
- Tamaño de palabra
- Memoria: SRAM, Flash, EEPROM, ROM, etc.
- I/O Digitales
- Entradas Analógicas
- Salidas analógicas (PWM)
- DAC (Digital to Analog Converter)
- ADC (Analog to Digital Converter)
- Buses
- UART
- Otras comunicaciones.

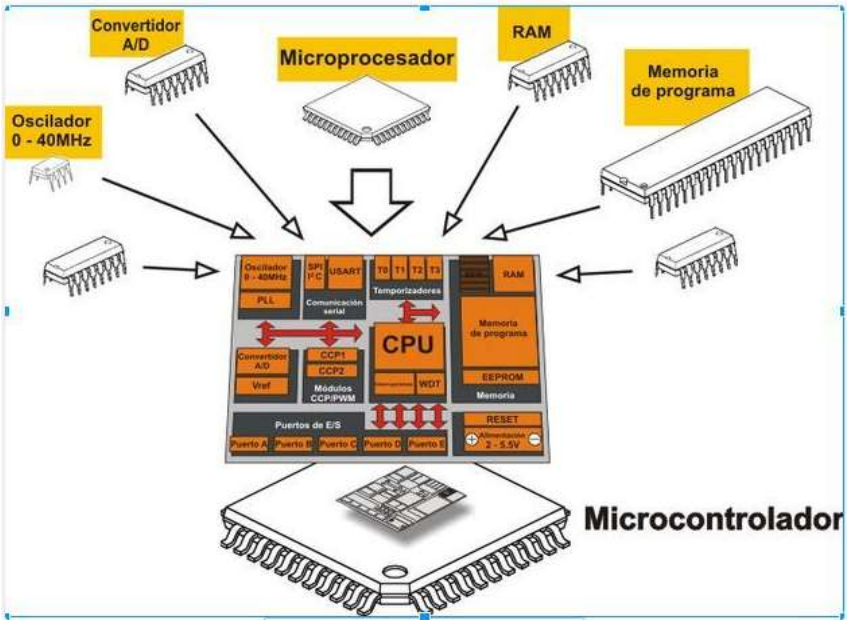



***Figura 2.36.-*** *Composición de un microcontrolador. Fuente: Aprendiendo Arduino [22]*

### 2.10.1 Arquitectura Von Neuman

La arquitectura Von Neumann utiliza el mismo dispositivo de almacenamiento tanto para las instrucciones como para los datos, siendo la que se utiliza en un ordenador personal porque permite ahorrar una buena cantidad de líneas de Entradas/Salidas (E/S), que son bastante costosas, sobre todo para aquellos sistemas donde el procesador se monta en algún tipo de zócalo alojado en una placa madre. También esta organización les ahorra a los diseñadores de placas madre una buena cantidad de problemas y reduce el costo de este tipo de sistemas.

En un ordenador personal, cuando se carga un programa en memoria, a éste se le asigna un espacio de direcciones de la memoria que se divide en segmentos, de los cuales típicamente tendremos los siguientes: código (programa), datos y pila. Es por ello por lo que podemos hablar de la memoria como un todo, aunque existan distintos dispositivos físicos en el sistema (disco duro, memoria RAM, memoria flash, unidad de disco óptico, etc).

En el caso de los microcontroladores, existen dos tipos de memoria bien definidas: memoria de datos (típicamente algún tipo de SRAM) y memoria de programas (ROM, PROM, EEPROM, flash u de otro tipo no volátil). En este caso la organización es distinta a las del ordenador personal, porque hay circuitos distintos para cada memoria y normalmente no se utilizan los registros de segmentos, sino que la memoria está segregada y el acceso a cada tipo de memoria depende de las instrucciones del procesador.

A pesar de que en los sistemas integrados con arquitectura Von Neumann la memoria esté segregada, y existan diferencias con respecto a la definición tradicional de esta arquitectura; los buses para acceder a ambos tipos de memoria son los mismos, del procesador solamente salen el bus de datos, el de direcciones, y el de control. Como conclusión, la arquitectura no ha sido alterada, porque la forma en que se conecta la memoria al procesador sigue el mismo principio definido en la arquitectura básica.

Algunas familias de microcontroladores como la Intel 8051 y la Z80 implementan este tipo de arquitectura, fundamentalmente porque era la utilizada cuando aparecieron los primeros microcontroladores.

### 2.10.2 Arquitectura Hardvard

La otra variante es la arquitectura Harvard, y por excelencia la utilizada en supercomputadoras, en los microcontroladores, y sistemas integrados en general. En este caso, además de la memoria, el procesador tiene los buses segregados, de modo que cada tipo de memoria tiene un bus de datos, uno de direcciones y uno de control.

La ventaja fundamental de esta arquitectura es que permite adecuar el tamaño de los buses a las características de cada tipo de memoria; además, el procesador puede acceder a cada una de ellas de forma simultánea, lo que se traduce en un aumento significativo de la velocidad de procesamiento. Típicamente los sistemas con esta arquitectura pueden ser dos veces más rápidos que sistemas similares con arquitectura Von Neumann.

La desventaja está en que consume muchas líneas de E/S del procesador; por lo que en sistemas donde el procesador está ubicado en su propio encapsulado, solo se utiliza en supercomputadoras. Sin embargo, en los microcontroladores y otros sistemas integrados, donde usualmente la memoria de datos y programas comparten el mismo encapsulado que el procesador, este inconveniente deja de ser un problema serio y es por ello por lo que encontramos la arquitectura Harvard en la mayoría de los microcontroladores.

Por eso es importante recordar que un microcontrolador se puede configurar de diferentes maneras, siempre y cuando se respete el tamaño de memoria que este requiera para su correcto funcionamiento.

Estructura genérica de un microcontrolador:

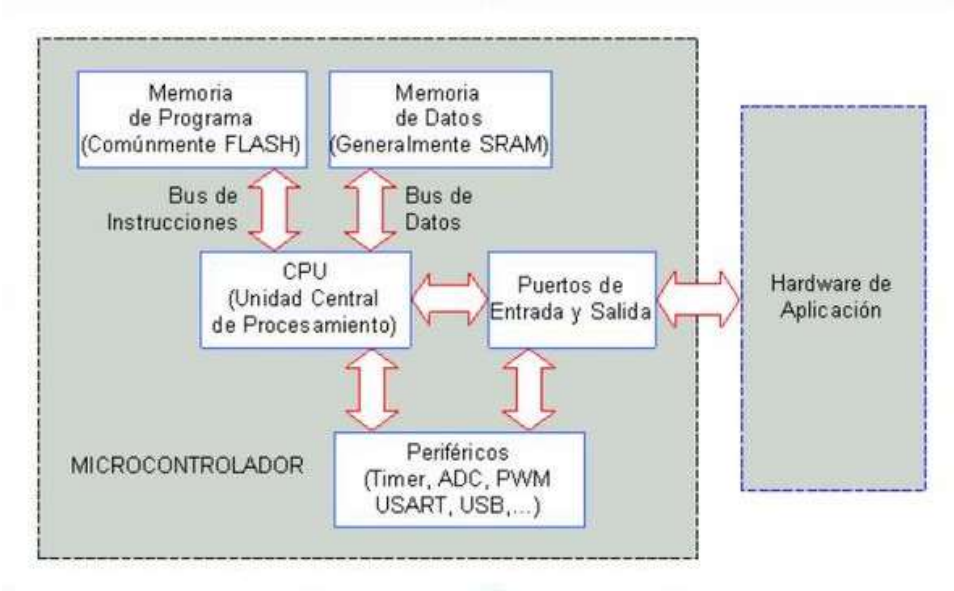



***Figura 2.37.-*** *Estructura de un microcontrolador. Fuente: Aprendiendo Arduino [22]*

- CPU (unidad central de proceso):

Podemos decir que la CPU, siglas en inglés de unidad central de proceso, es el núcleo del microcontrolador. Se encarga de ejecutar las instrucciones almacenadas en la memoria. Es lo que habitualmente llamamos procesador o microprocesador, término que a menudo se confunde con el de microcontrolador. En esta línea cabe aclarar que, tal y como estamos viendo, ambos términos no son lo mismo: el microprocesador es una parte de un microcontrolador y sin él no sería útil; un microcontrolador, en cambio, es un sistema completo que puede llevar a cabo de forma autónoma una labor.

- Memoria:

Entendemos por memoria los diferentes componentes del microcontrolador que se emplean para almacenar información durante un periodo determinado de tiempo. La información que necesitaremos durante la ejecución del programa será, por un lado, el propio código, y por otro, los diferentes datos que usemos durante la ejecución de este. Hablaremos por tanto de memoria de programa y de memoria de datos, respectivamente.

La diferente naturaleza de la información que hay que almacenar hace necesario el uso de diferentes tipos memorias. Sin hacer especial énfasis en este apartado, sí habrá que tener en cuenta una clasificación básica, que distingue entre memoria volátil y no volátil. La primera es aquella que pierde la información que almacena al

desconectarla de la alimentación; la segunda, como resulta obvio, no. Por lo tanto, se hace evidente que al menos la memoria de programa deberá ser no volátil: no sería práctico que el programa grabado en el microcontrolador se borre cada vez que se apague el dispositivo. Con respecto a la memoria de datos, diremos por el momento según la situación puede interesarnos una u otra.

- Unidades de entrada/salida:

Las unidades de entrada/salida son los sistemas que emplea el microcontrolador para comunicarse con el exterior. Con ciertas limitaciones en cuanto a valores de tensión y corriente, que pueden ser utilizados tanto como para leer valores de sensores (entradas) o para realizar algún cambio de estado en determinado actuador fuera del microcontrolador (salidas). Así, los dispositivos de entrada nos permitirán introducir información en el microcontrolador y los de salida nos servirán para que éste la saque al exterior.

- Periféricos:

Son todos esos subsistemas que están intercomunicados dentro del microcontrolador y cumplen una tarea específica, como el ADC, los Timers, interfaz i2c, controlador PWM, etc. Las prestaciones de estos periféricos están diseñadas por cada fabricante, de acuerdo con la necesidad se selecciona uno u otro microcontrolador.

- Microcontroladores AVR

Los AVR son una familia de microcontroladores RISC (computador con conjunto de instrucciones reducidas) del fabricante estadounidense Atmel, compañía adquirida por Microchip Technology en 2016. La arquitectura de los AVR fue concebida por dos estudiantes en el Norwegian Institute of Technology, y posteriormente refinada y desarrollada en Atmel Norway, la empresa subsidiaria de Atmel, fundada por los dos arquitectos del chip. Cuenta con bastantes aficionados debido a su diseño simple y la facilidad de programación.

RISC es una filosofía de diseño de CPU para computadora que está a favor de conjuntos de instrucciones pequeñas y simples que toman menor tiempo para ejecutarse aumentando así la performance del microcontrolador.

# Parte II

# Marco Metodológico

# 3. Diseño

En el siguiente capítulo se describen los módulos utilizados con sus respectivas pruebas de funcionamiento, el diseño de la fuente y los ensayos de los actuadores que mueven las alas.

## 3.1 El Procesador

En el área de la robótica, uno de los elementos indispensables para el funcionamiento y el accionar de ciertos dispositivos, es la computadora. La misma, quien coordina los tiempos de procesamiento de señales y acciona una respuesta determinada, es precargada por un programador según las necesidades y tareas a realizar por medio del entorno de programación (*ver apartado 2.2 del marco teórico*).

### 3.1.1 Arduino

Dentro del entorno de programación de la robótica con Arduino, es muy frecuente utilizar ciertos microcontroladores que son de mucha fiabilidad a buen costo económico, conteniendo una buena cantidad de pines de entradas/salidas tanto analógicas como digitales.

Existen razones válidas para utilizar Arduino, aquí solo se nombrará algunas, las más relevantes.

- **Es Open Source:** Esto quiere decir que se puede acceder a toda parte del circuito y del código de las librerías, puedes modificarlas y no necesitas ninguna licencia para utilizarlo.
- **Fácil de programar:** Arduino cuenta con muchas librerías que nos resuelven muchos problemas, gran porcentaje del código más complejo que necesitamos podría ser como para establecer una comunicación serie o inicializar un sensor. Otra gran ventaja es que no se necesita ninguna tarjeta de programación como sucede en la mayoría de las placas de desarrollo, Arduino cuenta con un software conocido como bootloader que viene cargado en el microprocesador que se auto programa a sí mismo y nos evita la necesidad de contar con una tarjeta programadora para programar el microcontrolador.
- **Amplia variedad de placas:** Arduino no es una única placa, es toda una familia de placas, cada una de ellas cuenta con diferencias y similitudes, según el proyecto que se realice, se utilizara una u otra.
- **Costo:** Podemos conseguir nuestra placa Arduino por menos de $500 pesos, es un precio muy económico comparado con otras placas que intentan cumplir los mismos requisitos. Obviamente existen diferentes modelos y alternativas, el costo varía, pero no demasiado.
- **Diversas aplicaciones:** se puede construir muchas aplicaciones a bajo costo y con poco tiempo de desarrollo, solamente hay que tener imaginación.
- **Proyectos profesionales:** Existen algunas tarjetas de Arduino que cuenta con el microcontrolador en un zócalo desmontable, se puede utilizar la placa solo para programarlo y luego retirar el micro de la placa y utilizarlo sin el board.

Dentro de esta selección, una de las posibilidades a utilizar en el proyecto, es la Arduino Nano (gama media). También, como otra posible opción, es la Arduino Pro Mini, que cumple con los estándares de comunicación que se necesita y a un menor costo y lógicamente, con menor cantidad de pines digitales y analógicos, ya que no se utilizarían tantos para el proyecto.

#### 3.1.1.1 Arduino Mega 2560

Arduino Mega es una tarjeta de desarrollo open-source construida con un microcontrolador modelo Atmega2560 que posee pines de entradas y salidas (E/S), analógicas y digitales. Esta tarjeta es programada en un entorno de desarrollo que implementa el lenguaje Processing/Wiring. Arduino puede utilizarse en el desarrollo de objetos interactivos autónomos o puede comunicarse a un PC a través del puerto serial (conversión con USB).

El Arduino Mega tiene 54 pines de entradas/salidas digitales (14 de las cuales pueden ser utilizadas como salidas PWM), 16 entradas análogas, 4 UARTs (puertos serial por hardware), cristal oscilador de 16MHz, conexión USB, Jack de alimentación, conector ICSP y botón de reset. Arduino Mega incorpora todo lo necesario para que el microcontrolador trabaje simplemente conéctalo a la PC por medio de un cable USB o con una fuente de alimentación externa (9V a 12V).

Características:

- Microcontrolador ATmega2560.
- Voltaje de entrada de 7-12V.
- 54 pines digitales de Entrada/Salida (14 de ellos son salidas PWM).
- 16 entradas análogas.
- 256k de memoria flash.
- Velocidad del reloj de 16Mhz.

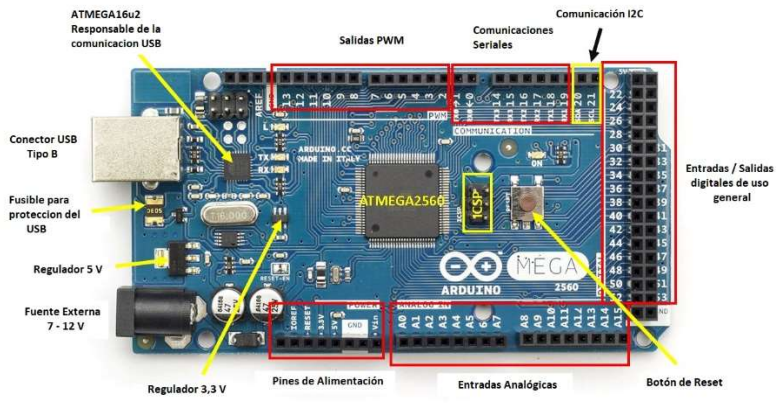



***Figura 3.1.-*** *Partes de una Arduino Mega 2560. Fuente: Arduino [1]*

#### 3.1.1.2 Arduino Nano

Es mucho más pequeño que el Arduino Mega. Está basado en el microcontrolador ATmega328. Tiene una entrada mini-USB a través de la cual se puede subir el código fuente para la ejecución de los comandos.

Viene con 14 puertos digitales de entrada/salida, 8 puertos analógicos, una memoria de 16 KB, 1 KB de SRAM y 512 bytes de EPROM. Su ClockSpeed es 16 MHz. Funciona con un voltaje que puede estar en el rango de 7 a 12 voltios. Entrega una corriente de 40 mA.

Aparte de algunas desventajas como un número menos de puertos de entrada/salida o un menor espacio en la memoria, es prácticamente idéntico al Arduino Mega. Se carga el código desde Arduino IDE, utilizando el mismo proceso que para con el Arduino Mega. Los códigos son perfectamente compatibles de una placa a otra.

Algunos pines tienen funciones especializadas:

- Comunicación Serie: 0 (Rx) y 1 (Tx). Se utiliza para recibir (Rx) y transmitir (Tx) datos serie TTL. Estos pines se conectan a los pines correspondientes del chip serial FTDI USB a TTL.
- Interrupciones externas: 2 y 3. Estos pines se pueden configurar para activar una interrupción en un valor bajo, un flanco ascendente o descendente, o un cambio en el valor.
- PWM: Pines 3, 5, 6, 9, 10 y 11. Proporcione una salida PWM de 8 bits.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estos pines son compatibles con la comunicación SPI, que, aunque provista por el hardware subyacente, no se incluye actualmente en el lenguaje Arduino.
- LED: 13. Hay un LED incorporado conectado al pin digital 13. Cuando el pin tiene un valor ALTO, el LED está encendido, cuando el pin está BAJO, está apagado.
- El Nano tiene 8 entradas analógicas, cada una de las cuales proporciona 10 bits de resolución (es decir, 1024 valores diferentes). Por defecto, miden desde el suelo hasta 5 voltios, aunque es posible cambiar el extremo superior de su rango utilizando la función de referencia analógica (). Los pines analógicos 6 y 7 no pueden usarse como pines digitales. Además, algunos pines tienen funcionalidad especializada:
- I2C: 4 (SDA) y 5 (SCL). Admita la comunicación I2C (TWI) utilizando la biblioteca de Wire (documentación en el sitio web de Wiring).
- Hay un par de otros pines en el tablero:
- AREF. Tensión de referencia para las entradas analógicas. Utilizado con analogReference ().
- Reiniciar. Lleve esta línea BAJA para reiniciar el microcontrolador. Normalmente se usa para agregar un botón de reinicio a los escudos que bloquean el que está en el tablero.

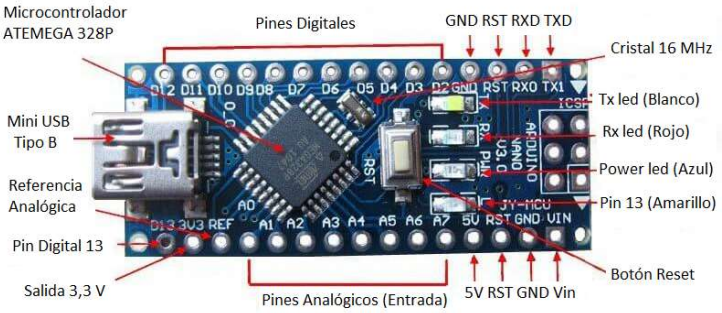



***Figura 3.2.-*** *Partes de una Arduino Nano. Fuente: Arduino [1]*

#### 3.1.1.3 Arduino Pro Mini

El Arduino Pro Mini es una tarjeta de desarrollo basada en el ATmega328. Cuenta con 14 pines de entradas/salidas digitales (de las cuales 6 se puede usar como salidas PWM), 6 entradas analógicas, un botón de reinicio, y agujeros para colocar conectores. Se puede conectar un conector de 6 pines para emplear un cable FTDI o una tarjeta FTDI para suministrar voltaje USB y establecer comunicación con el circuito.

Características

- Procesador ATmega328 de 16MHz
- Conexión USB fuera de la tarjeta
- Permite reset automático
- Regulador de 5V
- Salida máxima de 150 mA
- Protección contra corrientes excedentes
- Pesa menos de 2 gramos
- Entrada DC 5V hasta 12V
- Puertos analógicos: 8
- Puertos digitales I/O: 14
- Salidas PWM: 6

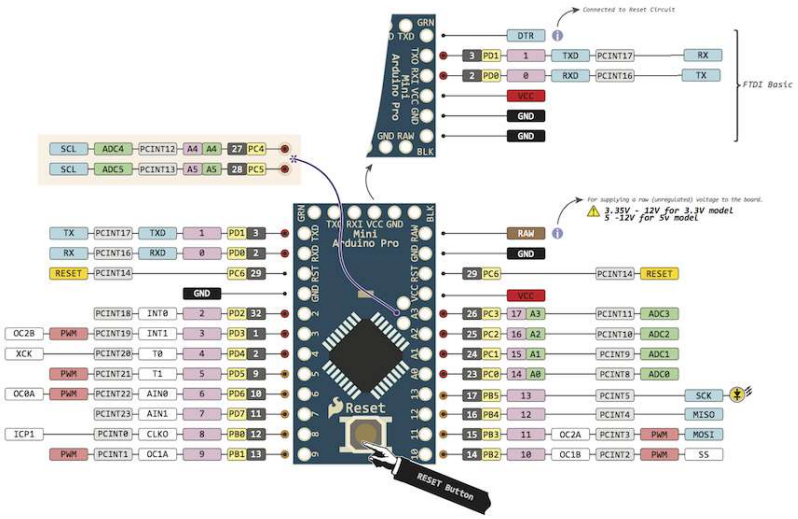



***Figura 3.3.-*** *Pines de una Arduino Pro Mini. Fuente: Arduino [1]*

Comparación entre las posibles placas a utilizar

Con la idea de poder encontrar la mejor opción para el proyecto, se hace una comparación de los distintos tipos de Arduinos más comunes y utilizados, con el objetivo de simplificar la información más importante en un cuadro, que muestra las características principales de cada placa.

| Name | Processor | Operating/Input Voltage | CPU Speed | Analog In/Out | Digital IO/PWM | EEPROM [kB] | SRAM [kB] | Flash [kB] | USB | UART |
|---|---|---|---|---|---|---|---|---|---|---|
| Mega 2560 | ATmega2560 | 5 V / 7-12 V | 16 MHz | 16/0 | 54/15 | 4 | 8 | 256 | Regular | 4 |
| Nano | ATmega168<br>ATmega328P | 5 V / 7-9 V | 16 MHz | 8/0 | 14/6 | 0.512<br>1 | 1<br>2 | 16<br>32 | Mini | 1 |
| Pro Mini | ATmega328P | 3.3 V / 3.35-12 V<br>5 V / 5-12 V | 8 MHz<br>16 MHz | 6/0 | 14/6 | 1 | 2 | 32 | - | 1 |

***Tabla 3.1.-*** *Comparación entre las distintas Arduinos utilizadas. Fuente: Arduino [23]*

## 3.2 Diseño de la fuente

A la hora de diseñar la fuente se optó por un convertidor DC DC del tipo Step Up como se explica en el marco teórico (*ver punto 2.10*), ya que para la implementación del proyecto se cuenta con una fuente de 5V, y los actuadores que mueven las alas requieren de 7V para su funcionamiento.

Luego de analizar varios integrados que pueden cumplir esta tarea, se seleccionó el LM2577 fabricado por Texas Instrument, ya que cumple con las necesidades de tensión y corriente y además es bastante accesible en el mercado local.

Los parámetros de diseño son los siguientes:

| | |
|---|---|
| Vin= 5V | Vout = 7V |
| Iin(max) = 3A | Iout(max) = 1.5A |

***Tabla 3.2.-*** *Parámetros de funcionamiento. Fuente: Propia*

En la siguiente tabla se observan los valores de trabajo máximo que son soportados por el LM2577.

**Operating Ratings**

| | | |
|---|---|---|
| Supply Voltage | | $3.5V \le V_{IN} \le 40V$ |
| Output Switch Voltage | | $0V \le V_{SWITCH} \le 60V$ |
| Output Switch Current | | $I_{SWITCH} \le 3.0A$ |
| Junction Temperature Range | LM1577 | $-55°C \le T_J \le +150°C$ |
| | LM2577 | $-40°C \le T_J \le +125°C$ |

***Tabla 3.3.-*** *Rangos de operación del LM2577. Fuente: Datasheet Texas Instruments [3].*

Comparando la *tabla 3.2* y la *tabla 3.3,* se puede ver que las necesidades de diseño son superadas por las capacidades del integrado.

### 3.2.1 Descripción del integrado

Este integrado requiere un número mínimo de componentes externos para su funcionamiento, es relativamente económico y muy simple de utilizar. Incluye dentro del chip un transistor NPN de 3A, que está asociado a un circuito de protección limitador de corriente, otro que limita por exceso de temperatura y uno que bloquea por baja tensión.

### 3.2.2 Funcionamiento y diseño del circuito

De la hoja de datos del fabricante, se utilizó el esquema propuesto para una fuente de tensión variable, ya que de esta forma se puede llegar a un ajuste fino de la tensión requerida.

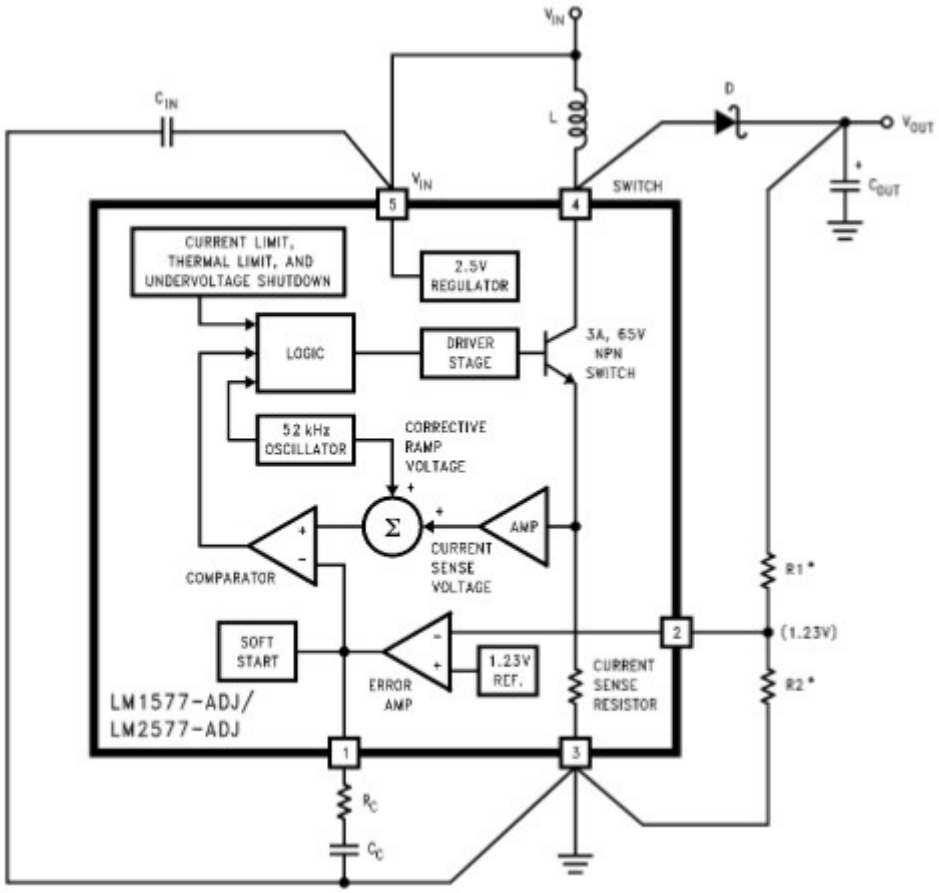



***Figura 3.4.-*** *Esquema – Composición interna del LM2577.Fuente: Datasheet Texas Instruments [3].*

Como se puede ver, es la misma configuración del boost converter visto en el marco teórico *Figura 2.24.*

Como se observa en la *figura 3.4*, el transistor interno es el que hace las veces de interruptor permitiendo que la energía almacenada en la bobina sea liberada hacia la carga. Este transistor está manejado por un controlador, que envía señal al transistor siempre y cuando ninguna de las protecciones internas estén activadas, y trabaja a 52KHz y utiliza una modulación por ancho de pulso (PWM).

El LM2577 posee un lazo de realimentación que se forma por un lado de una muestra de la tensión de salida llamada feedback (pin 2 del chip) que llega a una entrada del comparador. Y, por otro lado, desde el oscilador de 52KHz y la muestra de corriente, se genera una onda diente de sierra que llega a la otra entrada del comparador, y la salida de este da como resultado un ancho de pulso que es el entregado al transistor para que realice la conmutación. De esta manera, cuando la tensión de salida baja por

el aumento de una carga, el valor a la entrada FB (pin 2) también disminuye y luego la salida del comparador entrega un PWM con un ancho de mayor duración haciendo que la $V_{out}$ a la salida aumente; el análisis es contrario cuando la carga disminuye y se obtiene una disminución de la tensión a la salida.

### 3.2.3 Selección de componentes

Para la selección de componentes externos del integrado, se utilizan las fórmulas desarrolladas en el marco teórico, junto con algunos métodos prácticos basados en curvas del fabricante.

#### 3.2.3.1 Selección del inductor

La selección del inductor se basa en tres elementos:

- $D_{max}$ (es el valor máximo de dutty cycle al que trabajará el integrado)

$$D_{(max)} = \frac{V_{out} + V_f - V_{in\,(min)}}{V_{out} + V_f - V_{sat}} \tag{3.1}$$

Donde $V_f$ es la caída de tensión directa en el diodo y $V_{sat}$ es la tensión colector-emisor del transistor interno, para el LM2577 es de 0.6V según el fabricante. Reemplazando valores $V_f = 0.5v$ para un schottky y $V_{sat} = 0.6v$ se obtiene un $D_{(max)} = 0.362$.

Como el $D_{(max)} \leq 0.8$ se asegura que trabaja en modo continuo.

- $E \times T$ es el producto de tensión por tiempo de carga del inductor y se obtiene

$$E \times T = \frac{D_{(max)}\left(V_{in\,(min)} - V_{sat}\right) \times 10^6}{F_x} \qquad \text{[V.µSeg]} \tag{3.2}$$

Sabiendo que $F_x$ es la frecuencia del oscilador interno igual a 52KHz da un resultado $E \times T = 30.65$ V.µSeg

- $I_{ind,dc}$ es la corriente media en el inductor en condición de plena carga, se obtiene con la siguiente ecuación que está afectada por un porcentaje de ripple que para este caso es de 5%, y recordando que la máxima corriente es de 1.5A.

$$I_{ind,dc} = \frac{1.05 \times I_{load}}{1 - D_{max}} = 2.468\ A \tag{3.3}$$

Finalmente, se selecciona del Figura 3.4 utilizando los valores obtenidos $E \times T$ y de $I_{ind,dc}$ dando como resultado un inductor de 47uHy.

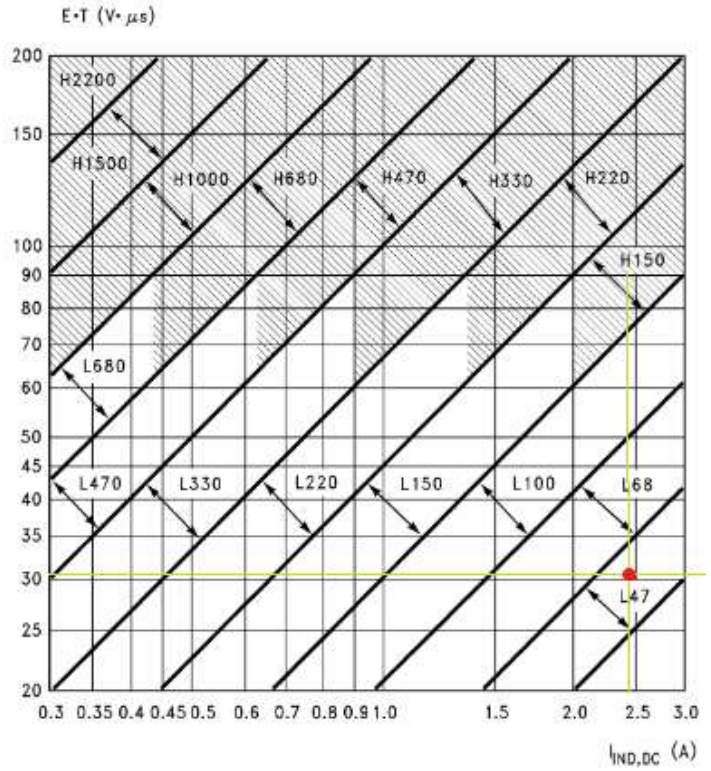



***Figura 3.5.-*** *Selección del inductor utilizando el grafico del fabricante. Fuente: Datasheet Texas Instruments [3].*

### 3.2.3.2 Selección de la red de compensación y el capacitor de salida

El $R_C$ y el $C_C$forman una red de compensación polo-cero que estabilizan al regulador, estos valores dependen de la ganancia del regulador, de la máxima corriente, del inductor y del capacitor de salida.

- Primero se calcula el máximo valor de RC

$$R_C \leq \frac{750 \times I_{load(max)} \times {V_{out}}^2}{{V_{in(min)}}^2} = 2.2\ K\Omega \tag{3.4}$$

- Luego se calcula el mínimo valor de $C_{out}$ utilizando las siguientes dos ecuaciones

$$C_{out} \geq \frac{0.19 \times L \times R_C \times I_{load(max)}}{V_{in(min)} \times V_{out}} \tag{3.5}$$

$$C_{out} \geq \frac{V_{in(min)} \times R_C \times \left(V_{in(min)} + \left(3.74 \times 10^5 \times L\right)\right)}{487{,}800 \times {V_{out}}^3} = 38.27\ \mu F \tag{3.6}$$

Se optó por un capacitor de aluminio electrolítico de aluminio SMD 100 $\mu F$ x 63V.

- Finalmente se calcula el mínimo valor de CC

$$C_C \geq \frac{58.5 \times {V_{out}}^2 \times C_{out}}{{R_C}^2 \times V_{in(min)}} = 220\ nF \tag{3.7}$$

Se optó por un capacitor cerámico SMD 0603 de 330 $nF$ x 50V.

#### 3.2.3.3 Valor de la Tensión de Salida

El valor de la tensión de salida viene dado por un divisor resistivo y se calcula de la forma:

$$V_{out} = 1.23v \times \left(1 + \frac{R_1}{R_2}\right) \tag{3.8}$$

Sabiendo la tensión de salida requerida, solo resta seleccionar $R_1$ y $R_2$, para ello se fija un valor y se calcula el otro.

$$\frac{R_1}{R_2} = \frac{V_{out}}{1.23v} - 1 \tag{3.9}$$

$$R_1 = \left(\frac{V_{out}}{1.23v} - 1\right) \times R_2 \tag{3.10}$$

Eligiendo $\mathrm{R}_2 = 2.2\mathrm{K}\Omega$ el resultado es $\mathrm{R}_1 = 10.32\mathrm{K}\Omega$. Para llegar a un valor preciso se coloca un potenciómetro multivuelta.

#### 3.2.3.4 Selección del diodo

El diodo del regulador tipo boost debe ser capaz de resistir tanto la tensión inversa, que para este caso es la tensión de salida, como también la corriente pico del LM2577. Este diodo debe tener un valor muy pequeño de caída de tensión en modo directo ($V_f$). La barrera de tensión en los diodos schottky favorecen la operación de los reguladores conmutados y para este caso se utiliza el 1N5820.

| PRIMARY CHARACTERISTICS | |
|---|---|
| $I_{F(AV)}$ | 3.0 A |
| $V_{RRM}$ | 20 V, 30 V, 40 V |
| $I_{FSM}$ | 80 A |
| $V_F$ | 0.475 V, 0.500 V, 0.525 V |
| $T_J$ max. | 125 °C |
| Package | DO-201AD |
| Diode variations | Single |

***Tabla 3.4.-*** *Selección del diodo schottky 1N5820 con datos del fabricante. Fuente: Datasheet [4].*

El 1N5820 es un diodo schottky de bajo costo y muy popular en el mercado local, posee bajas pérdidas en conducción y es capaz de trabajar en altas frecuencias con buen rendimiento. Para más detalles ver el Anexo III *apartado 7.3 "hoja de datos diodo Schottky 1N5820".*

### 3.2.3.5 Diseño del circuito y PCB

Una vez que se cuenta con los elementos seleccionados se procede a realizar el pcb para luego soldar los componentes y testear el funcionamiento. El diseño se realizó en Altium Designer 2018 y fue fabricado con la fresadora del laboratorio de computación (modelo LPFK E33). El tiempo de fabricación fue de una hora logrando un muy buen resultado en términos de calidad. Se compraron los componentes y el montaje se realizó con una estación de soldado (YAXUN 881D).

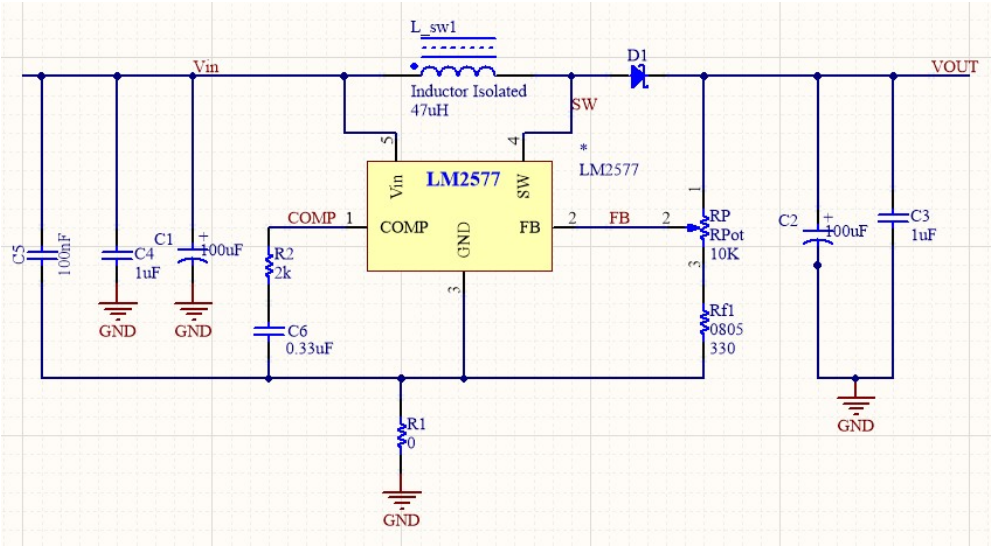



***Figura 3.6.-*** *Esquemático de la fuente diseñada en Altium. Fuente: Propia.*

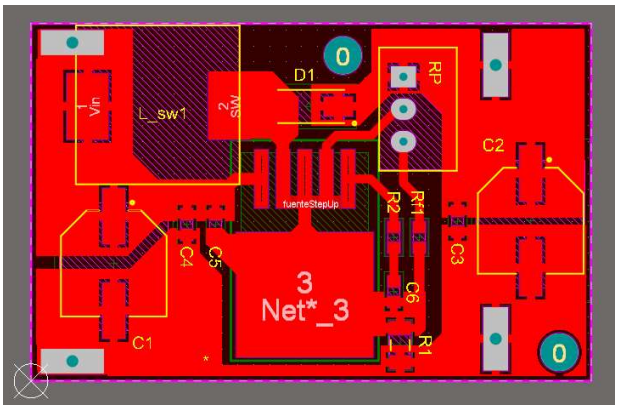



***Figura 3.7.-*** *Layout de la fuente diseñada en Altium. Fuente: Propia.*

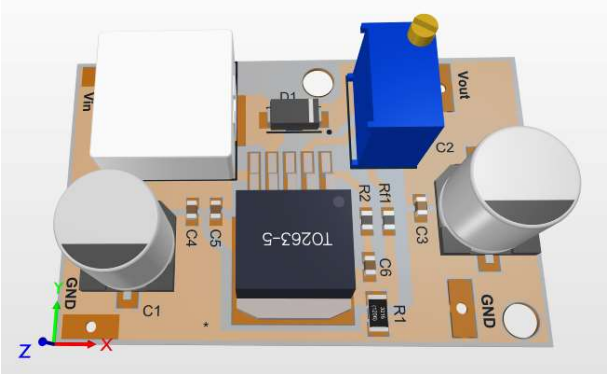



***Figura 3.8.-*** *Vista 3D de la fuente diseñada en Altium. Fuente: Propia.*

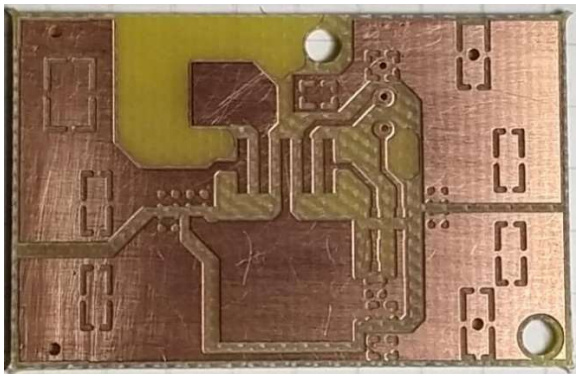


***Figura 3.9 a)-*** *Foto de placa impresa de la fuente diseñada. Fuente: Propia.*

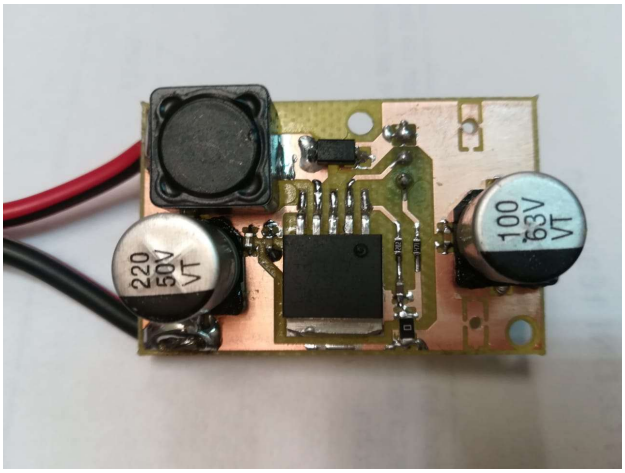



***Figura 3.9 b)-*** *Foto final de la fuente diseñada. Fuente: Propia.*

## 3.3 Actuadores

### 3.3.1 Consideraciones sobre el modelo

Una vez comprendido el funcionamiento del servomotor que fue explicado en el marco teórico (*ver apartado 2.8 "Servomotor"*) podemos analizar las consideraciones necesarias para el control de las alas. Observamos que todo el rango de movimiento, es decir, los 180º se encuentran contenidos en un total de 2ms por lo que podemos expresar la siguiente relación:

$$r = \frac{2000\mu\text{Seg}}{180°} = 11.1\mu\text{Seg/grados} \tag{3.11}$$

Siendo $r$ la resolución del servo.

Por lo que dentro de su rango de operación, el brazo del servo producirá una rotación de 1º grado cada 11,11 µs o fracción, es decir, 0,1º en 1,11 µs .

Es importante notar que los valores antes mencionados pueden variar levemente dependiendo del fabricante del servomotor, ya que estamos suponiendo que el rango de operación es de 2ms (0,5ms min – 2,5 ms max). Al momento de realizar la implementación del sistema, estos valores deberán ser ajustados dependiendo el servomotor utilizado según las especificaciones del fabricante.

### 3.3.2 Importancia de la resolución del servo

Todos los datos que provee el fabricante del servo son importantes, pero uno en particular es prioritario en nuestra aplicación y es la resolución con la que puede trabajar el servomotor, es decir, cuantos pasos intermedios puede generar en su rango de operación, cuántos grados o décimas de grado rota con cada paso. Se destaca este parámetro sobre los demás dado que fundamentalmente de él dependerá la precisión que podamos obtener en el posicionamiento del ala.

### 3.3.3 Cálculos y selección de los servomotores

Para llevar a cabo esta selección, se realizó el estudio de los momentos y las fuerzas que actúan sobre el ala en las diferentes posturas que puede adoptar.

A la hora de determinar el momento máximo, se observa que éste se obtiene para un ángulo de 0° con respecto a la horizontal. Y se determina mediante la expresión:

$$M = R \times D \times Cos\ \theta \tag{3.12}$$

Sabemos que el ala pesa 110gr y mide 60cm por lo que el resultado obtenido es

$$M = 0.11\ Kg \times 60\ cm \times 1 = 6.6\ Kg.cm \tag{3.13}$$

Aplicando un factor de seguridad de 1.75 da como resultado

$$M \times 1.75 = 6.6\ Kgcm \times 1.6 = 10.56 kg.cm \qquad (3.14)$$

Por ello seleccionamos un servo de 12Kg x cm pensando en el agregado de los gramos extra que aporta la parte artística.

### 3.3.4 Estudio de momentos máximo y comportamientos de servos bajo carga

Ante la falta de documentación técnica disponible para los servomotores adquiridos, se realizó un ensayo del comportamiento de estos. Para su ensayo se sujetó a un centímetro de distancia del eje del servo una balanza digital vertical y un recipiente como se ve en la *figura 3.11*.

***Figura 3.10.-*** *Ensayo de Servomotor, frente a una carga nula. Fuente: Propia*

Se excitó al servo con un PWM tal que lo alineó de forma paralela a la línea del piso, y luego se cargó el recipiente de forma progresiva y se registraron los distintos valores de corriente cada 500 gramos de carga hasta que la reducción interna de los engranajes fue vencida.


***Figura 3.11.-*** *Ensayo de Servomotor, sujeto a una carga. Fuente: Propia*

La prueba realizada dio los siguientes resultados:

| Carga [gr] | Intensidad [A] |
|---|---|
| 2790 | 0 |
| 3000 | 0.50 |
| 3500 | 0.52 |
| 4000 | 0.53 |
| 5000 | 0.55 |
| 6000 | 0.64 |
| 7000 | 1.04 |

***Tabla 3.5.-*** *Datos del ensayo realizado al Servomotor. Fuente: Propia.*

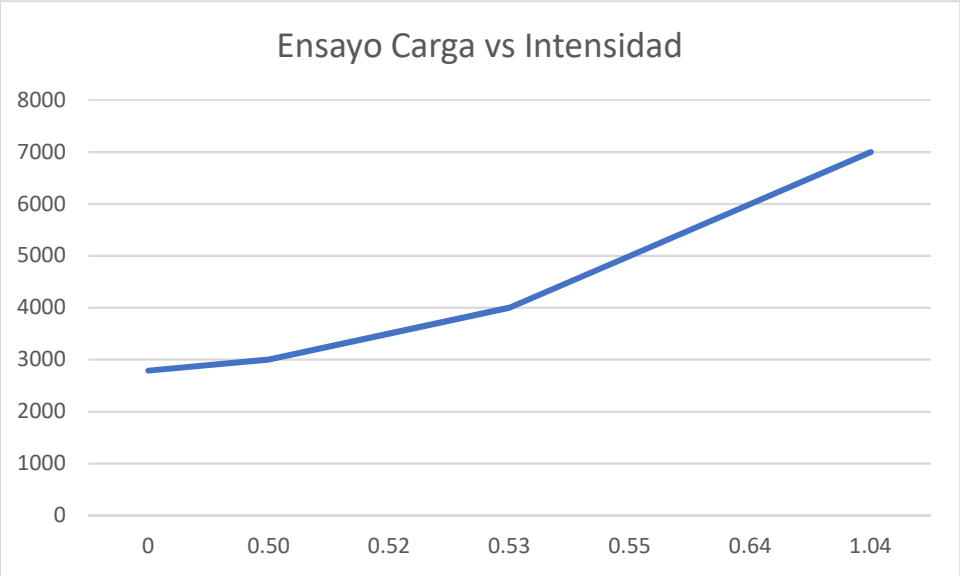



***Figura 3.12.-*** *Ensayo de a una carga vs Intensidad de corriente. Fuente: Propia.*

## 3.4 Generación del movimiento

Una de las maneras para estudiar un dispositivo y poder llegar a un producto final, es ir desmenuzando el proyecto en varios "mini proyectos". Es decir, hacer distintas pruebas e ir progresando cada vez más hasta llegar al modelo final. En este camino, nos encontramos con conclusiones favorables y no favorables, hasta incluso se encontraron soluciones a problemas que no eran los esperados, o hallar reemplazos para aquellos casos que no cumplen con lo esperado. A continuación, solamente se detalla uno de los experimentos que se realizó y los demás ensayos restantes que complementan al estudio, se encuentran en el Anexo I.

### Conexiones Sensor – Microcontrolador

Conectando la IMU MPU6050 (cuya explicación de funcionamiento se encuentra en marco teórico *apartado 2.3.3 "IMU Unidad de Medida Inercial"*) al microcontrolador se logró establecer la comunicación por medio del protocolo I2C, y se recibió datos del sensor IMU MPU6050, y al ser procesados por la placa desarrolladora, realizó ciertos cálculos obteniendo el Angulo de manera indirecta.

La conexión entre el microcontrolador Arduino Mega 2560 y nuestro sensor IMU MPU6050 se realiza de la siguiente forma:

| IMU MPU6050 | ARDUINO MEGA 2560 |
|---|---|
| VCC | 5 V |
| GND | GND |
| SDA | Pin 20 |
| SCL | Pin 21 |

***Tabla 3.6.-*** *Pines de conexión sensor IMU MPU6050 y Arduino Mega. Fuente: propia*

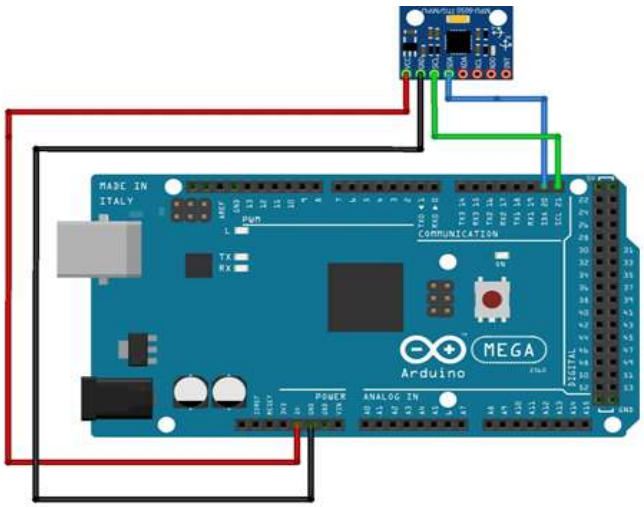



***Figura 3.13.-*** *Conexión sensor IMU MPU6050 y Arduino Mega. Fuente: Propia.*

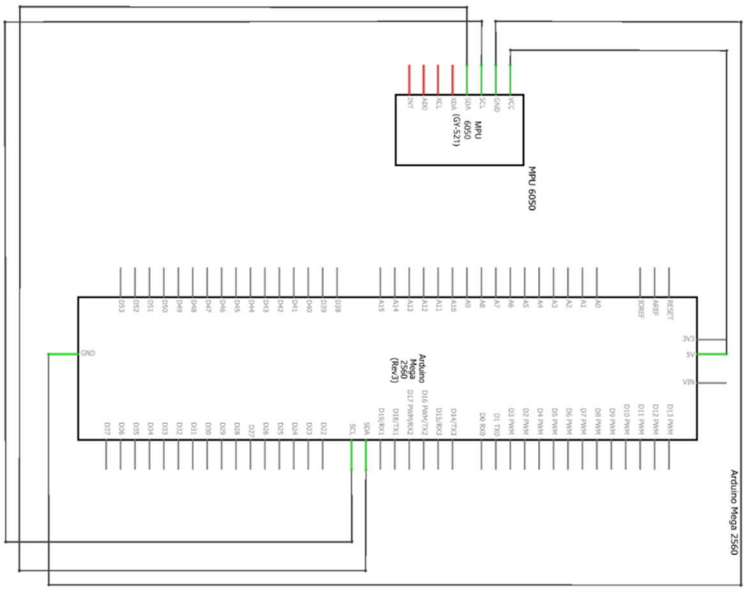



***Figura 3.14.-*** *Esquema eléctrico sensor IMU MPU6050 y Arduino Mega. Fuente: Propia.*

La comunicación entre nuestro dispositivo y la PC mediante conexión cableada se realiza vía puerto serie. A través de este tipo de comunicación podemos enviar y recibir datos desde nuestro Arduino Mega.

El cable con el que programamos el microcontrolador desde una PC es también cable de comunicación por puerto serie. Para establecer la comunicación, lo primero es abrir ese puerto serial en la aplicación de Arduino.

Para ello utilizamos la función: `Serial.begin(9600);`

El número que va entre paréntesis es la velocidad de transmisión. Todos los dispositivos que van a comunicarse deben tener la misma velocidad para poder entenderse.

Se incluye la librería Wire.h, necesaria para la interacción vía protocolo I2C (explicado en el marco teórico *apartado 2.4*) y se define la dirección I2C de la IMU que se especifica en la documentación oficial (0x68).

Existen datos de conversión que son especificados en la documentación del sensor, y eso por ello que se debió dividir los valores que dé el sensor entre estas constantes para obtener un valor coherente.

La IMU da los valores en enteros de 16 bits y como Arduino los guarda en enteros de menor bits, hubo que declarar las variables que almacenan los enteros provenientes de la IMU como un tipo de enteros especiales.

Finalmente se tuvo tres arrays provenientes de los cálculos, que guardan el ángulo X del Acelerómetro, el Giroscopio y el resultado del Filtro respectivamente.

Con estos valores se leen las aceleraciones de los distintos ejes que son recibidos del sensor.

```
AcX=Wire.read() <<8|Wire.read(); //Cada valor ocupa 2 registros
AcY=Wire.read() <<8|Wire.read();
AcZ=Wire.read() <<8|Wire.read();
```

A partir de los valores del acelerómetro, se calculan los ángulos Y, X respectivamente, con la fórmula de la tangente.

```
Acc [0] = atan((AcY/A_R) /sqrt(pow((AcX/A_R),2)
+ pow((AcZ/A_R),2))) *RAD_TO_DEG
```

Se piden los datos del giroscopio que servirán para calcular el Angulo del giroscopio Y, X respectivamente.

```
GyX=Wire.read() <<8|Wire.read();
GyY=Wire.read() <<8|Wire.read();

//Calculo del angulo del Giroscopio
Gy [0] = GyX/G_R;
Gy [1] = GyY/G_R;
```

Luego, se aplicar el Filtro Complementario (explicado en el marco teórico *apartado 2.5 "Filtro Digital"*) según la definición:

```
Angle [0] = 0.98 *(Angle [0]+Gy [0]*0.010) + 0.02*Acc [0];
```

El código completo se encuentra en el Anexo I *apartado 5.2.1 "Conexión IMU MPU6050*".

## 3.5 Comunicación Inalámbrica

Existen maneras de obtener una comunicación sin la utilización de medios físicos tales como los cables, etc. Aquí se probó un módulo cuyo medio de funcionamiento es a través del espectro radioeléctrico, siendo un medio inalámbrico que no manifiesta molestia alguna en el espacio. La conexión necesaria para la realización de la comunicación es a través de la configuración de un módulo que se comunica vía serie.

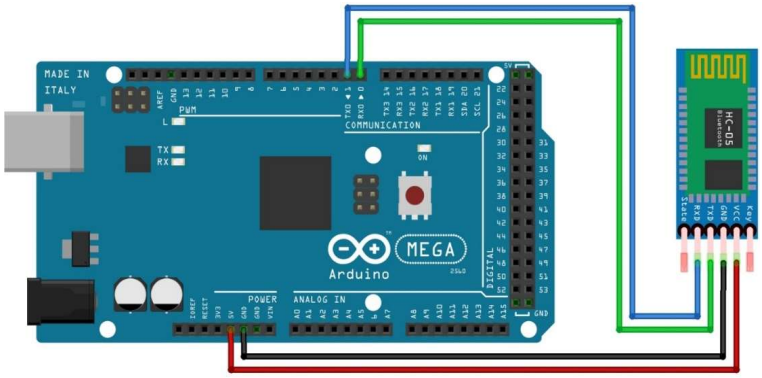



***Figura 3.15.-*** *Conexión Arduino Mega 2560 y Modulo Bluethoot HC05. Fuente: Propia*

Para ello, fue necesario hacer un configurar cada módulo HC05 para que pueda vincularse con el Arduino. Para que estos módulos funcionen como se desea, fue necesario desarrollar un programa que pudieran configurar ciertos parámetros indispensables para el funcionamiento y su vinculación entre los mismos (explicado en el Anexo I *apartado 5.2.5.1 "configuración modulo bluetooth*").

Una vez cargado el mismo, se procedió a configurar los parámetros indispensables que diferenciarán entre un maestro y un esclavo y su vinculación entre ellas. Se aprecia la configuración en Anexo I *apartado 5.2.5.2* "*Configuración Maestro-Esclavo por comandos AT*".

# Descripción del Modelo Experimental

Este capítulo trata sobre el proyecto en cuestión, el modelo final y la mejora que se prevé a futuro. Además, hay dos experimentos indispensables; uno fue útil para entender el funcionamiento del sensor y el otro para garantizar la fiabilidad del dispositivo y del código.

## 4.1 Modelado del sensor IMU MPU6050 en Simulink

Se pretende estudiar la IMU MPU6050 en el entorno de Simulink, siendo éste una de las herramientas más completas para la simulación en tiempo real. Se lo puso en ensayo bajo distintas condiciones y llegar a encontrar como es su diagrama en bloques completo, y obtener información sobre la construcción de este elemento.

Se comenzó instalando la librería de la IMU MPU6050 que contiene un acelerómetro y giroscopio en su interior. Como se mencionó, el tipo de datos que entrega este dispositivo no es exactamente el ángulo, dado que la medición es de manera indirecta. Esto se debe que se agregó ciertos componentes que ayudaron a dar valores de ángulos en grados.

Existen dos formas de obtener el Angulo desde este sensor, por medio de la aceleración angular y la velocidad angular. Se experimentó de ambas maneras y se concluyó cuál de las dos opciones fue la mejor.

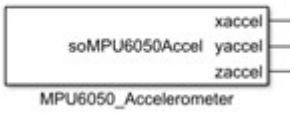



***Figura 4.1.-*** *Componente Acelerómetro de la IMU. Fuente: Propia*

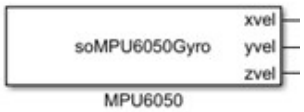



***Figura 4.2.-*** *Componente Giróscopo de la IMU. Fuente: Propia*

Se buscó en Simulink Library Browser la librería “Rensselaer Arduino Support Package” y allí se encontraron ambos elementos del mpu6050.

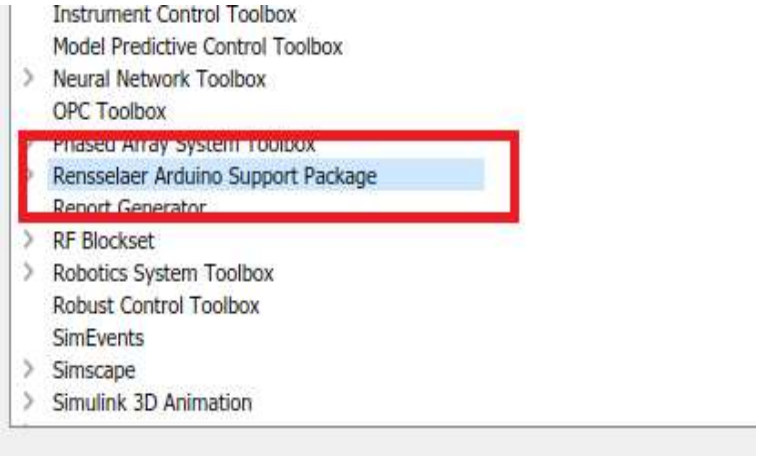



***Figura 4.3.-*** *Bloque Giróscopo y Scope sobre eje Z. Fuente: Propia*

Este proyecto sólo se utilizará a lo largo de un solo eje, asique según sea la posición que se encuentre la IMU, será motivo de estudio, siendo nuestro caso el eje Z. Agregando un bloque Scope en el eje Z, se graficó la misma, y además para darle fin a las salidas de los ejes X e Y, se agrega un bloque terminator, evitando cargar más datos al Arduino.

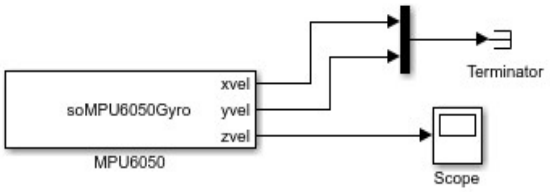



***Figura 4.4.-*** *Bloque Giróscopo y Scope sobre eje Z. Fuente: Propia*

Se corrió el programa y se movió la IMU de un lado a otro, obteniendo la gráfica de salida sobre el eje Z.

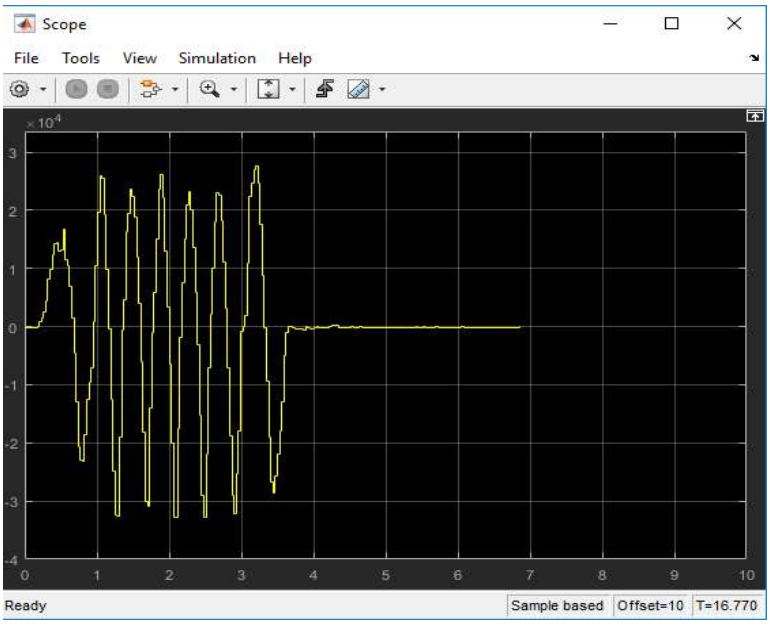



***Figura 4.5.-*** *Scope - Velocidad angular sobre eje Z. Fuente: Propia*

Como se mencionó, la señal observada representa la velocidad angular, o sea, la velocidad en la que se mueve el objeto (MPU6050) siendo la velocidad en el eje “Z”, y que no es el ángulo.

En una primera instancia, se giró en un sentido y luego hacia el otro, obteniendo lectura. Luego, se dejó el sensor en estado de reposo con una inclinación determinada, y se observó que no dio lectura, permaneciendo constante (*ver figura 4.5*).

El motivo, lo que se está midiendo es la velocidad angular y no el Angulo en sí, es por eso lo que al dejar de moverlo no genera velocidad alguna y permanece en reposo. Para obtener el ángulo, fue de necesidad de agregar algunos complementos para transformar la velocidad angular a Angulo.

### 4.1.1 El ángulo desde la velocidad angular

Se agregó un bloque integrador a la salida del bloque giróscopo para integrarla (en tiempo discreto, ya que todas nuestras mediciones son discretas) y así se conoció el Angulo sobre el eje Z.

A tales fines de simplificar y ordenar los bloques, se realizó un subloque denominado “MPU6050 zvel” que en su interior contiene el Giroscopio con el bloque terminator, y se observa a continuación.

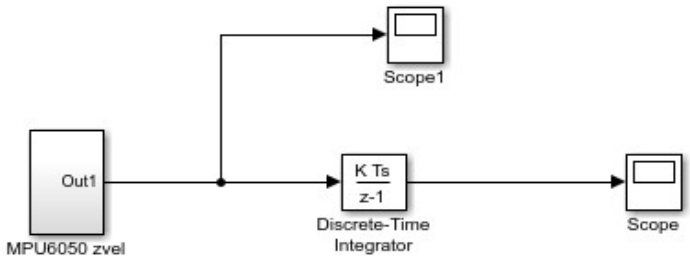



***Figura 4.6.-*** *Diagrama en Bloque - Giroscopio Eje Z e Integrador. Fuente: Propia*

Se comparó las gráficas de la velocidad angular y el ángulo obtenido y se encontró datos que servirán para el estudio.

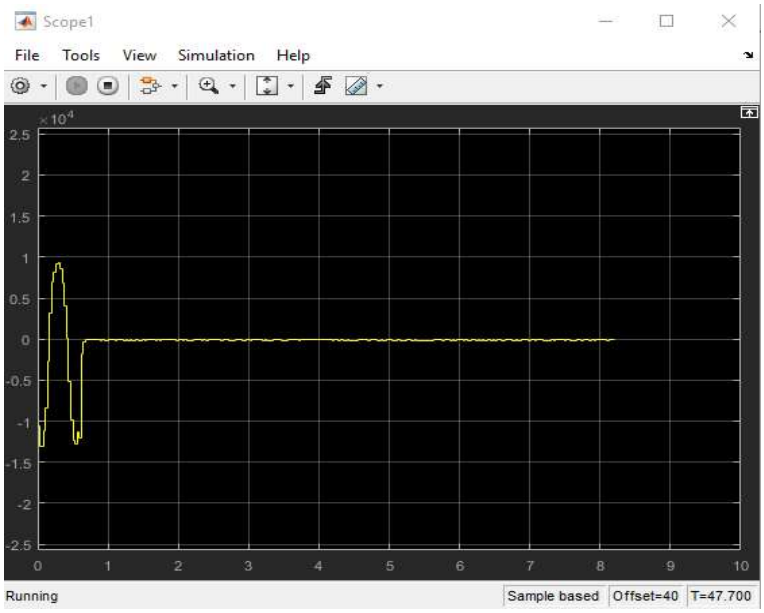



***Figura 4.7.-*** *Grafico de Velocidad Angular. Fuente: Propia*

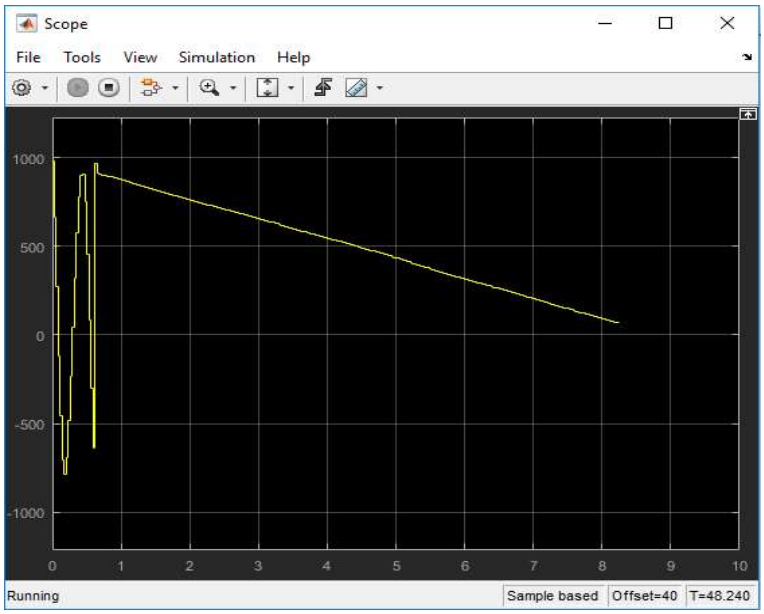



***Figura 4.8.-*** *Grafico de la posición Angular. Fuente: Propia*

Como bien se ve, en la f*igura 4.8. se* genera una rampa descendente cuando la IMU está en posición de reposo (mismo instante donde la *figura 4.7.* muestra que no hay velocidad).

Buscando fundamentación y explicación este fenómeno obtenido en f*igura 4.8.*, se realizó un zoom a la *figura 4.7*.

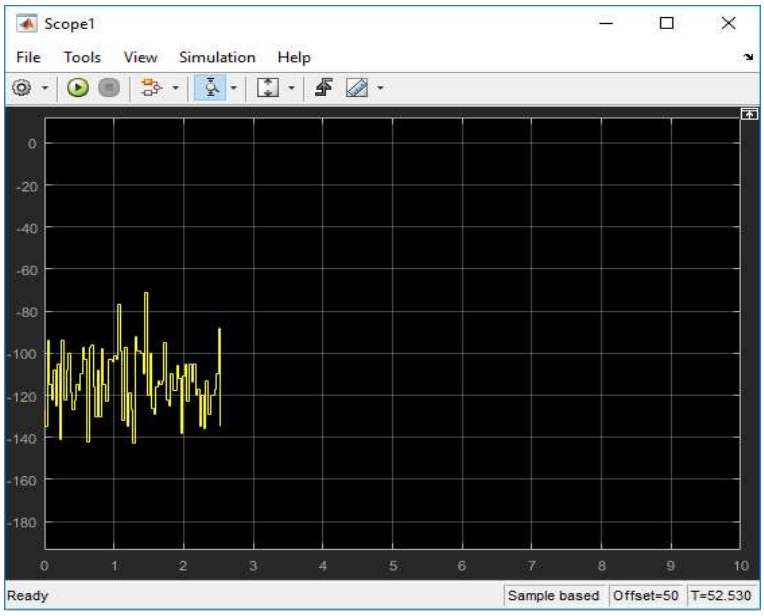



***Figura 4.9.-*** *Zoom al grafico de la posición Angular. Fuente: Propia*

Se observó que existe un offset determinado y de valor negativo, y al integrar este valor da el hecho inesperado. La única forma para lograr que la integración sea

nula, es llevar el offset lo más cercano a cero. Dicho esto, se agregó un bloque sumador con un valor de constante determinada, que contrarreste ese offset.

Un aproximado y a ojo, es sumar un valor de 110 (ya que el offset es de -110), pero es una forma de hacerlo sin datos estadísticos. Otra manera más exacta, como se trabaja en tiempo discreto, se puede tomar datos en instantes de tiempo y guardarlos en Matlab, para luego hacer un promedio y así obtener una estimación mucho más certera.

En una primera instancia, se procedió a sumarle una constante de 110 y se vio como es su respuesta, y se comparó la graficas nuevamente.

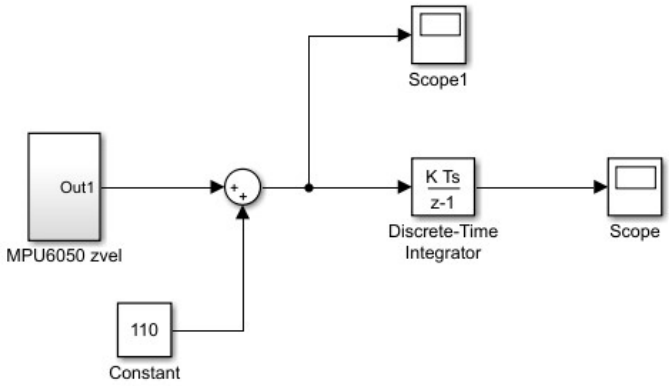



***Figura 4.10.-*** *Agregado constante y sumador al diagrama en bloque. Fuente: Propia*

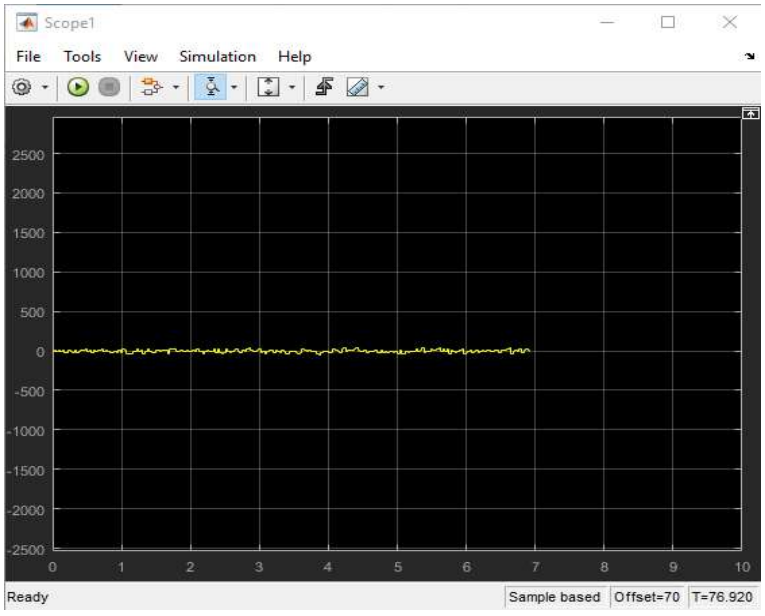



***Figura 4.11.-*** *Velocidad Angular con el agregado de constante de 110 y sumador. Fuente: Propia*

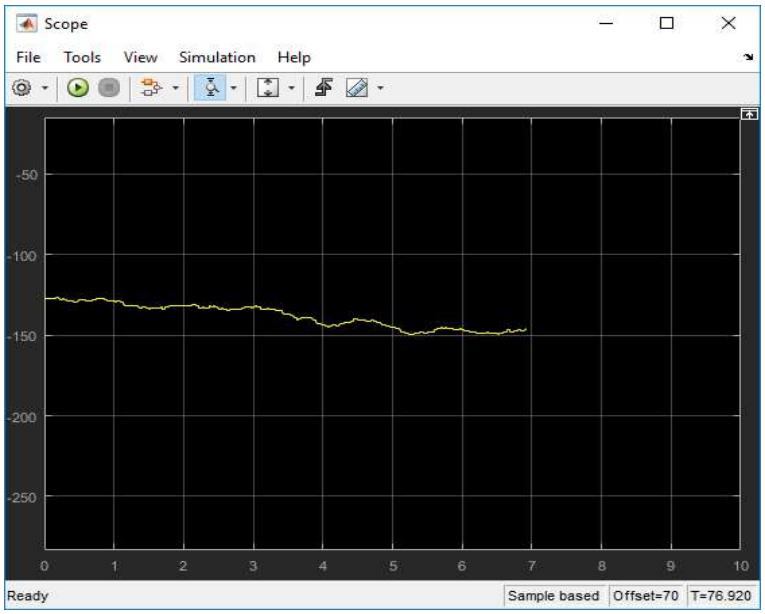



***Figura 4.12.-*** *Zoom Posición Angular con el agregado de constante y sumador. Fuente: Propia*

Si bien se observó que el offset pudo haber desaparecido y que el valor de la velocidad angular pareciera ser cero, se pudo ver que no lo es. Observando la gráfica de la posición angular, queda demostrado que sigue descendiendo la misma, no mejorando el sistema.

Esto llevó a concluir, por más leve que sea el offset, traerá problemas para nuestro sistema dando una lectura incorrecta, y es de necesidad realización una tarea más fina para eliminar el offset que causa este inconveniente.

Con la intención de solucionar el problema anterior, se conectó un bloque “simout” a la salida del bloque “MPU6050 zvel” que nos dio lectura de la velocidad angular en el tiempo. Luego de unos segundos y obtenidos varias lecturas, se realizó un promedio de estos, logrando conocer el valor apropiado por medio de la estadística.

El bloque “Simout”, generó una tabla con valores de lectura que luego se exportó al Matlab, para que realice los cálculos necesarios con el fin de obtener el offset de incógnita.

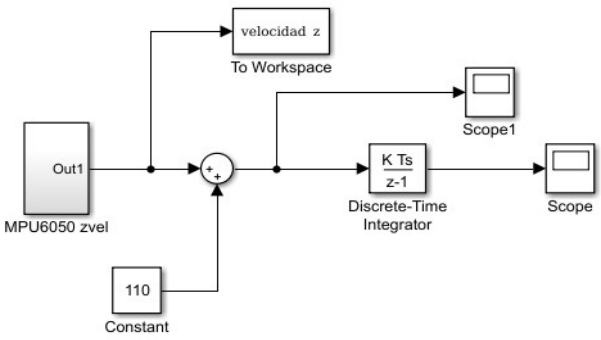



***Figura 4.13.-*** *Envío de datos al Workspace de Matlab. Fuente: Propia*

Se simuló por un tiempo de sesenta segundos con el sensor IMU MPU6050 en estado de reposo, y se evitó perturbaciones que modifiquen la realidad, y se capturó la mayor cantidad de lectura de datos provenientes del offset.

Dentro de la ventana del Workspace en Matlab, se generó un vector que contiene la lectura del giroscopio, cuyo nombre es el que nosotros colocamos anteriormente.

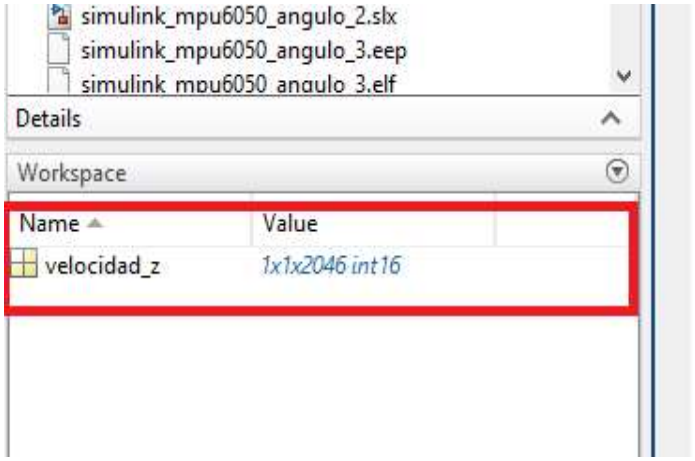



***Figura 4.14.-*** *Recepción de datos del Simulink. Fuente: Propia*

Se calculó el promedio de los datos, colocando "*mean*" y el nombre del vector que para nuestro caso es "*velocidad_z*".

```
>> mean(velocidad_z)

ans =

 -111.4140

>>
```

***Figura 4.15.-*** *Promedio de los valores tomados. Fuente: Propia*

El valor que se obtuvo es de -111.41. Si bien fue muy cercano a la aproximación que se definió con anterioridad, existe una diferencia entre ambas, y por más pequeña que sea, al integrar la misma no daba cero y se seguía graficando la misma rampa descendiente.

Se colocó el valor de la constante calculada y se verificó nuevamente la salida que sea deseada.

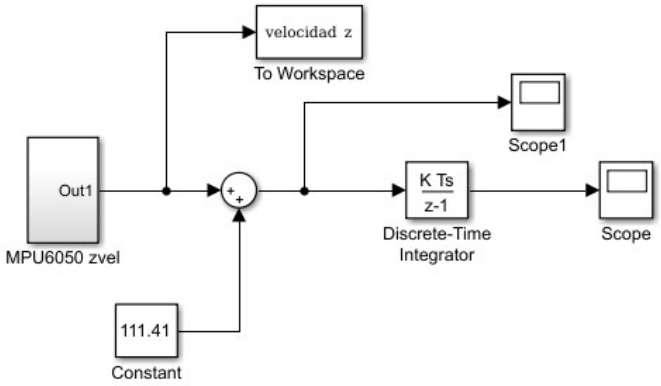



***Figura 4.16.-*** *Cambio del valor de la constante a 111.41. Fuente: Propia*

Se volvió a simular y se observó los nuevos datos obtenidos en el Scope1 de la posición angular.

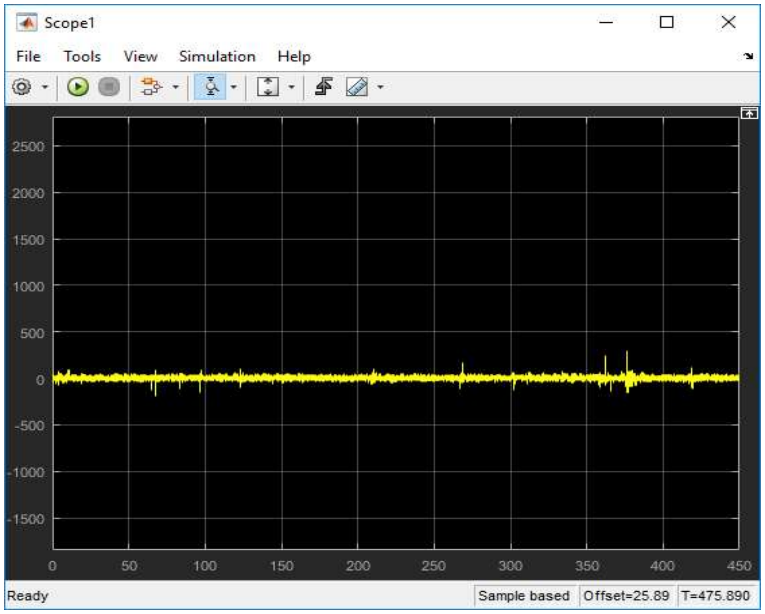



***Figura 4.17.-*** *Velocidad angular con una constante de 111.41. Fuente: Propia*

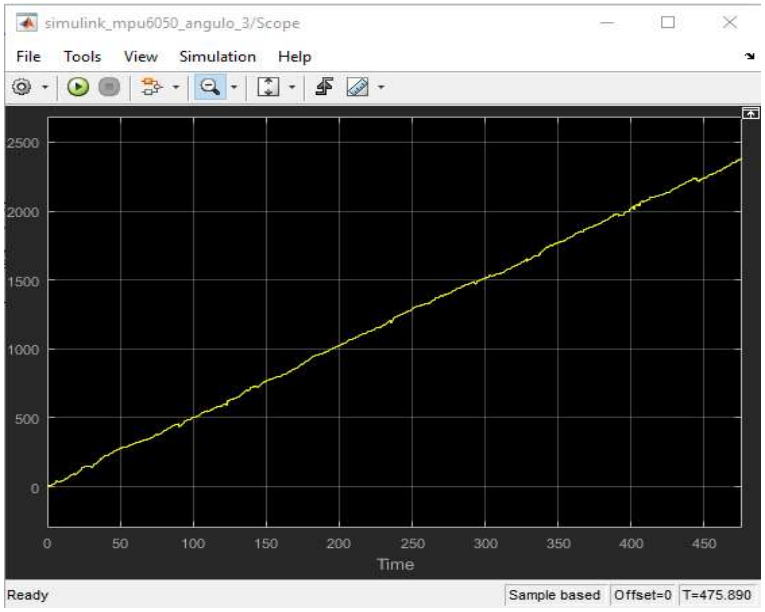



***Figura 4.18.-*** *Posición angular con una constante de 111.41. Fuente: Propia*

El resultado, es una rampa ascendente. Esto se debió que se colocó un valor que está por encima del verdadero. Como solución al problema, se dejó simular por un periodo prolongado de tiempo, y se volvió a calcular el promedio para obtener un resultado más certero.

Para esta nueva simulación, se dejó simular por 475 segundos y se volvió a calcular el promedio con la función de Matlab, y arrojo el siguiente valor.

```
>> mean(velocidad_z)

ans =

 -106.3934

>>
```

***Figura 4.19.-*** *Promedio de los valores tomados. Fuente: Propia*

Una vez obtenido el nuevo valor del promedio, se cambió el valor de la constante de nuestro sistema y se volvió a simular.

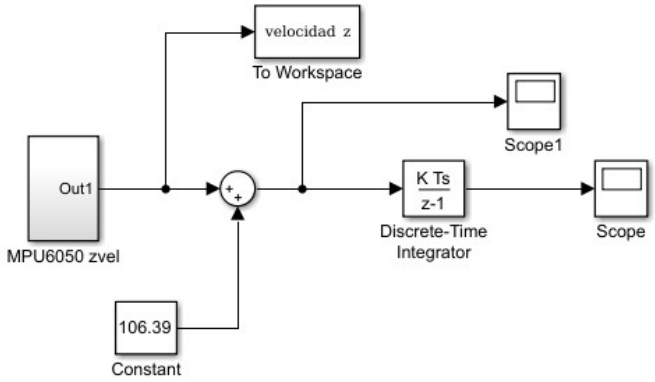



***Figura 4.20.-*** *Cambio del valor de la constante a 106.39. Fuente: Propia*

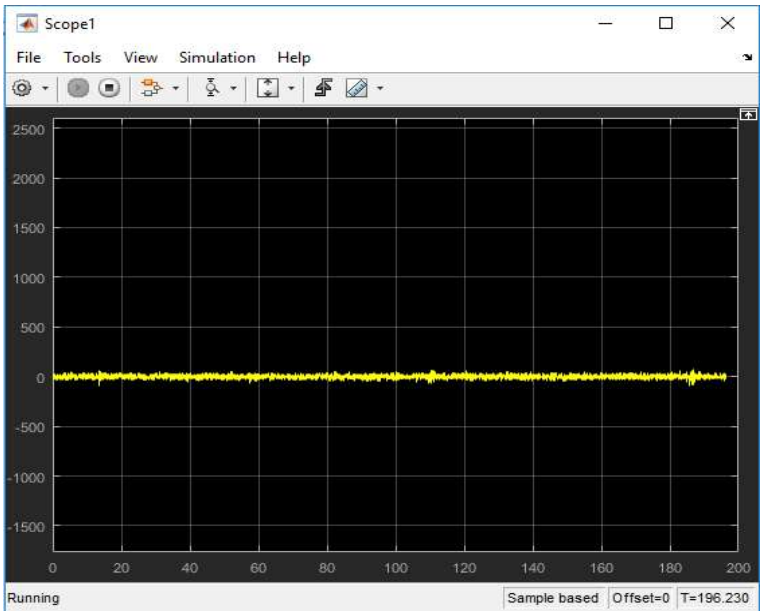



***Figura 4.21.-*** *Velocidad angular con una constante de 106.39. Fuente: Propia*

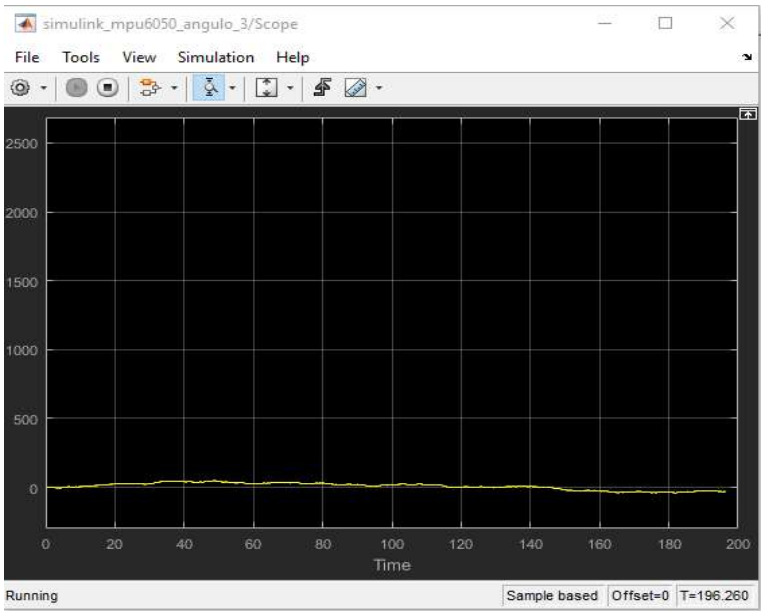



***Figura 4.22.-*** *Posición angular con una constante de 106.39. Fuente: Propia*

De la nuestra última *figura 4.22,* se observó que la posición angular ahora si se encuentra en un estado estacionario (desapareció la rampa), y aunque por más que exista un leve ondulamiento, podemos decir que se encuentra promedio en cero. La manera de ver si funciona correctamente, se movió el giroscopio para que de lectura y de esta forma se afirmó que el valor de la constante es el adecuado.

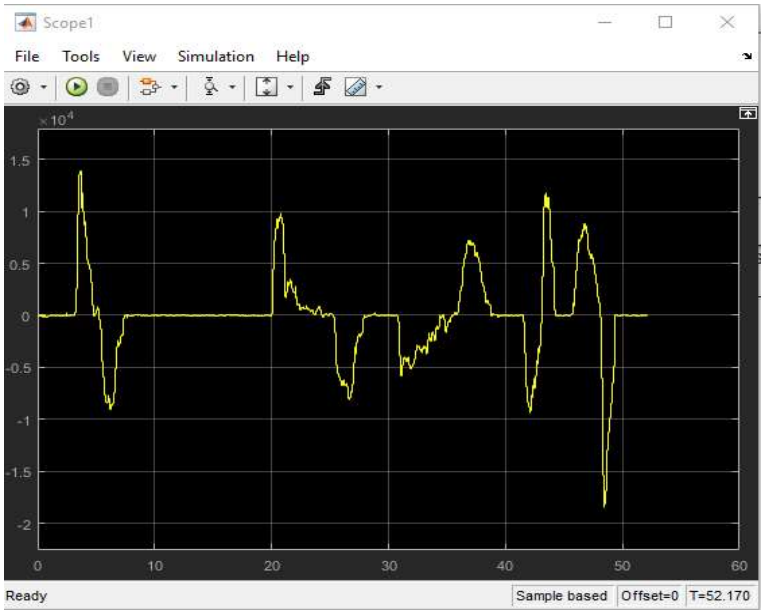



***Figura 4.23.-*** *Velocidad angular en movimiento con la constante de 106.39. Fuente: Propia*

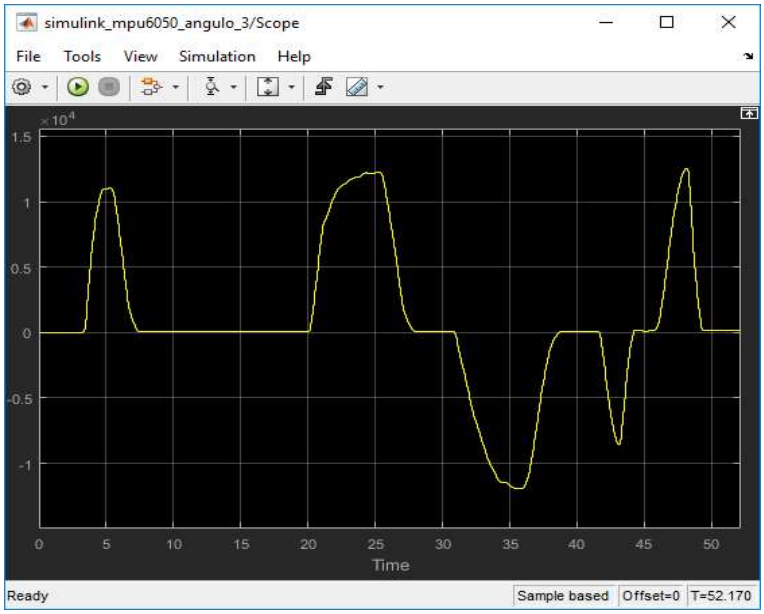



***Figura 4.24.-*** *Posición angular en movimiento con la constante de 106.39. Fuente: Propia*

La última *figura 4.24* se demostró que dio lectura de un valor de ángulo y que cuando el sensor está en reposo se mantiene en cero. Hasta ahora, solo se ocupó de estudiar el sensor IMU MPU6050 y su configuración en Simulink, ahora como ya se obtuvo lectura correcta del mismo, se procedió a escalear o a llevarlo a escala en grados (como se observa en esta última *figura 4.24,* no contiene la escala correspondiente).

Para realizar un escalado se necesitó conocer los límites máximos y mínimos que nos dio la gráfica cuando el sensor IMU MPU6050 se encuentra físicamente en 90° y -90°.

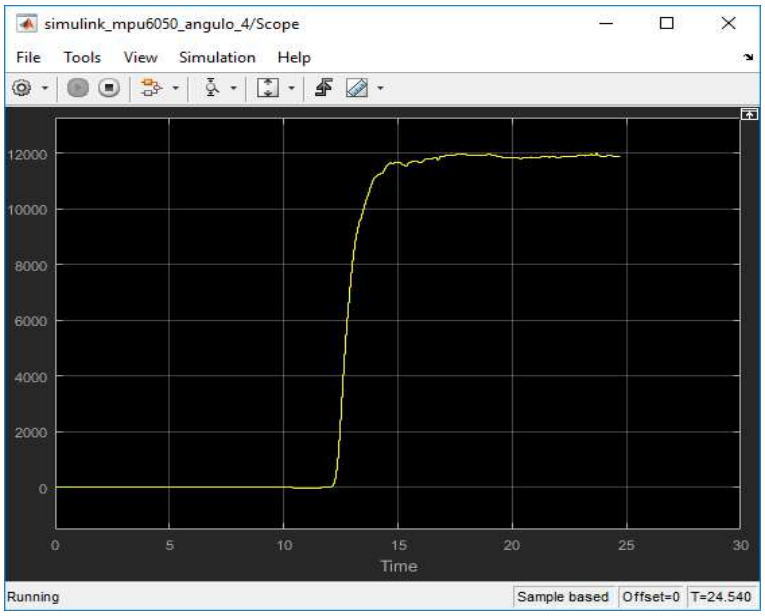



***Figura 4.25.-*** *Posición angular cuyo sensor se encuentra a 90°. Fuente: Propia*

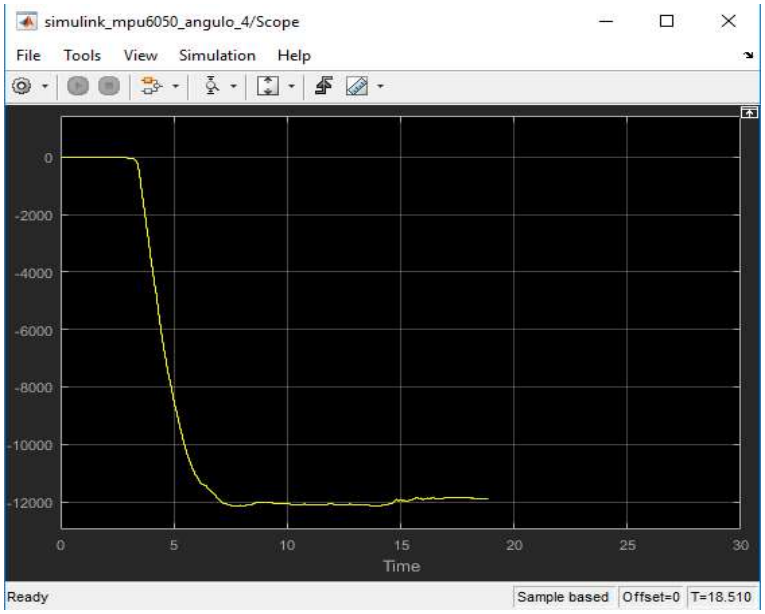



***Figura 4.26.-*** *Posición angular cuyo sensor se encuentra a -90°. Fuente: Propia*

Se observó que el valor máximo, cuando el sensor esta físicamente inclinado hacia los 90°, tiende a ser un valor aproximado de 12000. De manera contraria, cuando el sensor está a -90° la lectura muestra un valor de -12000.

Ya conocido estos límites, se procedió a convertir para que se den lecturas de 90° y no 12000, y viceversa. Para ello, se colocó un bloque de ganancia donde la misma fue de 90/12000, que será la conversión adecuada para nuestra lectura correspondiente.

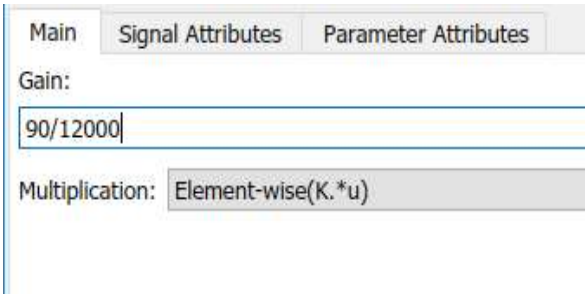



***Figura 4.27.-*** *Valor de la ganancia para escalear el valor del ángulo. Fuente: Propia*

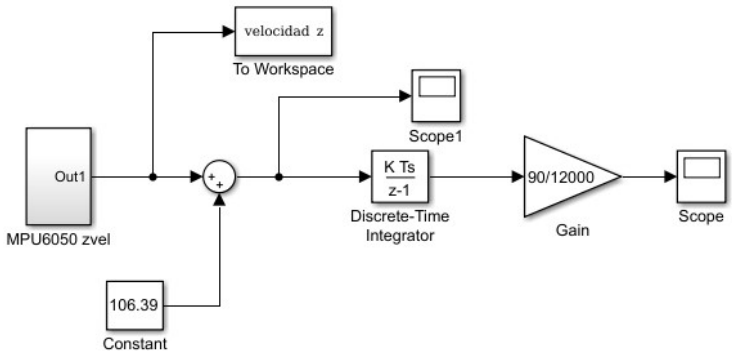



***Figura 4.28.-*** *Agregado del bloque Gain para escalear el valor del ángulo. Fuente: Propia*

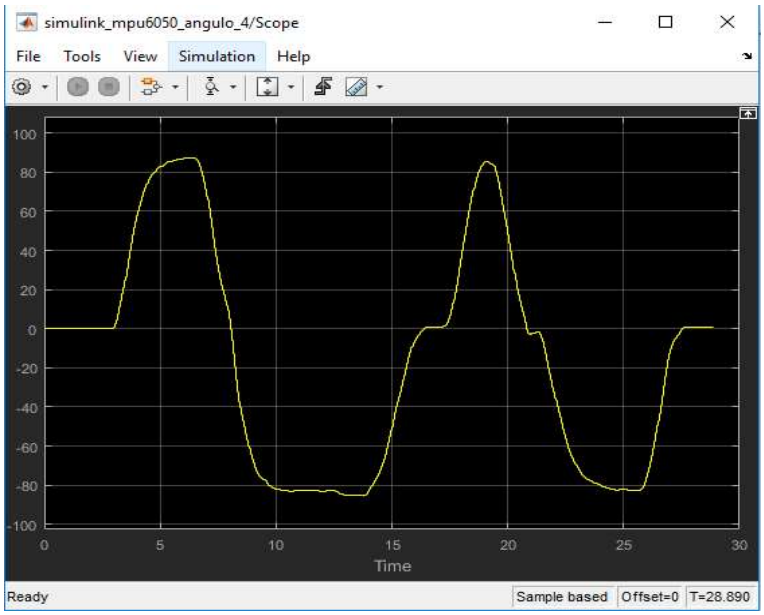


***Figura 4.29.-*** *Posición angular expresados en grados. Fuente: Propia*

Se simuló nuevamente corroborando los nuevos datos que entregó el sensor IMU MPU6050 expresados en grados cuyos límites máximos y mínimos son 90° y -90°.

### 4.1.2 El ángulo desde la aceleración angular

Otra de las maneras para obtener el ángulo es a través de la aceleración angular [*Véase Referencia Marco teórico*]. Partiendo de la fórmula, para representar la misma se agregó un bloque de la componente de aceleración de la IMU MPU6050 y sus anexos.

$$\Phi = \tan^{-1}(\frac{ay}{az})$$

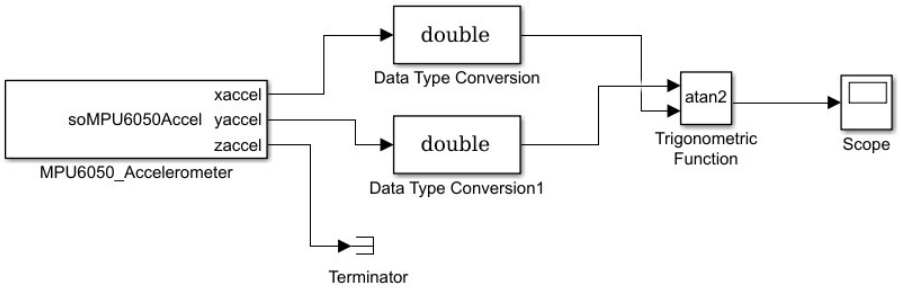



***Figura 4.30.-*** *Obtención del ángulo partiendo de la función. Fuente: Propia*

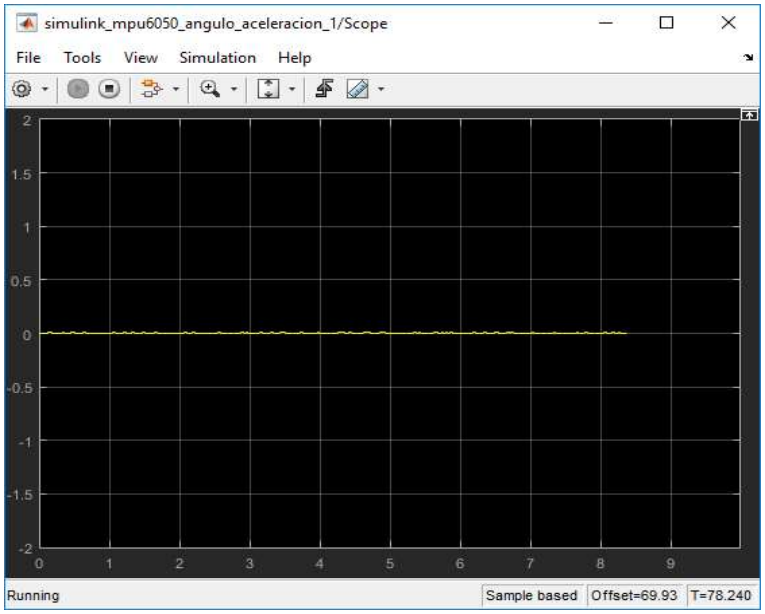



***Figura 4.31.-*** *Posición angular de la componente aceleración. Fuente: Propia*

Se giró en un sentido y luego en otro obteniendo valores cercanos a PI/2 (*Figura. 4.32)* y -PI/2 (*Figura 4.33)* respectivamente.

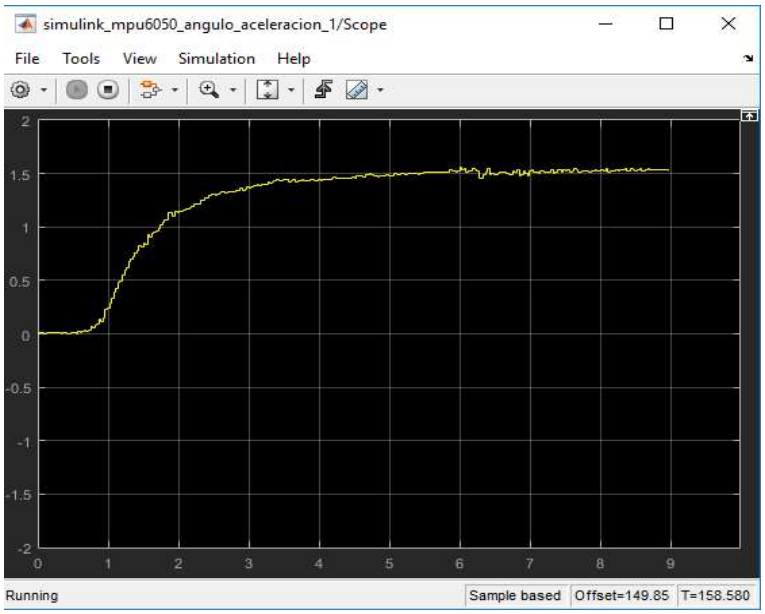



***Figura 4.32.-*** *Posición angular en PI/2. Fuente: Propia*

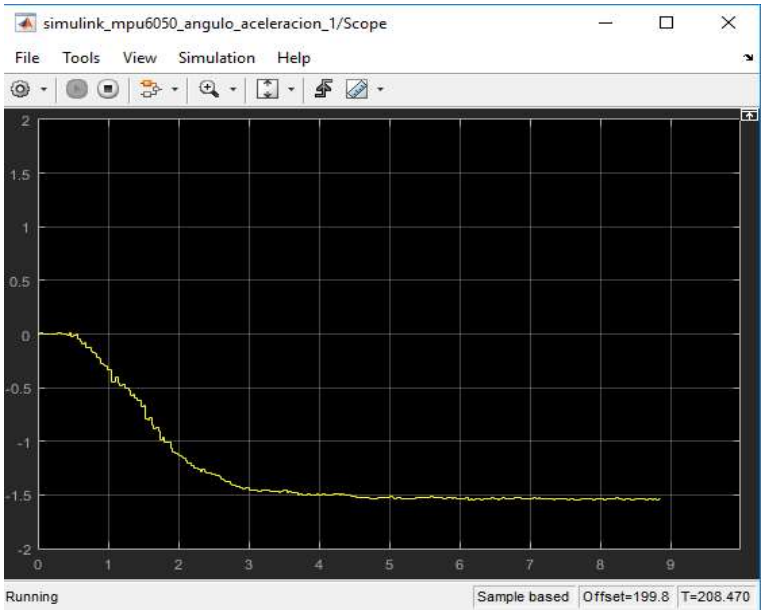



***Figura 4.33.-*** *Posición angular en -PI/2. Fuente: Propia*

El inconveniente que ocasionó de medir la posición angular mediante la aceleración es si se llegase a presentar alguna vibración en la tarjeta (producto de algún tipo de movimiento o vibración) será visible el error, generando otra aceleración aparte a la de la gravedad de la tierra, produciendo malas lecturas como se ve en la *figura 4.34*.

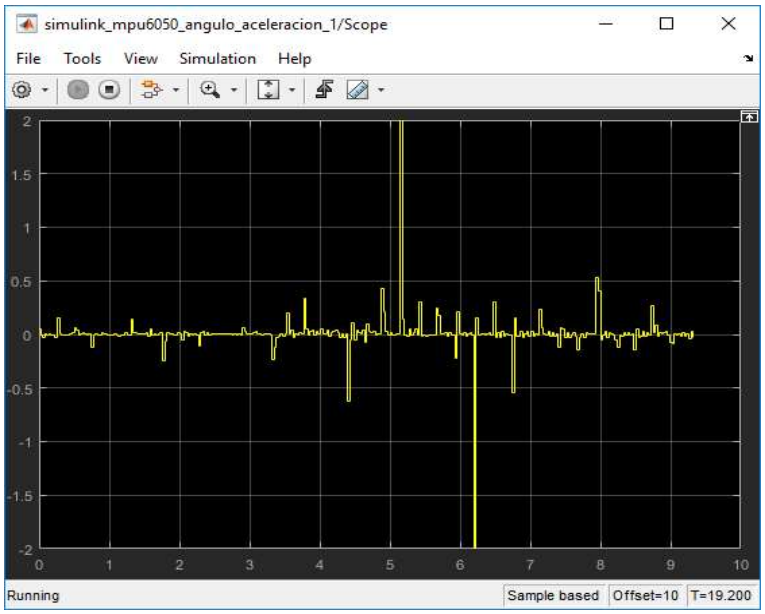



***Figura 4.34.-*** *Sensor IMU MPU6050 en estado estacionario junto a vibraciones. Fuente: Propia*

Como solución al problema anterior, se diseñó un filtro que evitó las imperfecciones vistas en la *figura 4.34*. El filtro más adecuado se llama filtro complementario y constó de tomar la posición angular desde la velocidad angular calculada en el punto 2.2.1 y la posición angular medida desde la aceleración como en el caso de la 2.2.2.

### 4.1.3 Principio de funcionamiento del Filtro complementario.

En ocasiones anteriores, se explicó el funcionamiento del filtro complementario [*Véase Referencia Marco teórico*], aquí se desarrolló el mismo, partiendo de medir la velocidad angular y la aceleración angular y se combinó ambas.

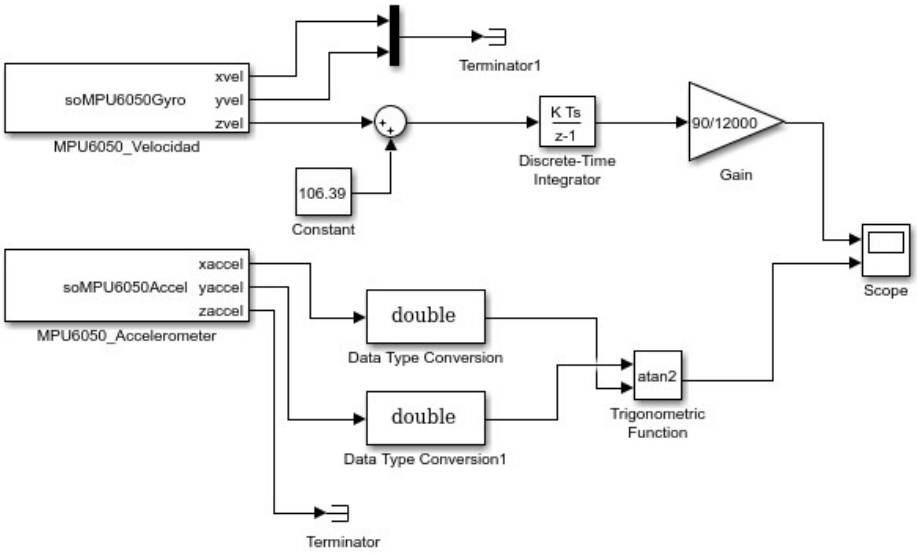



***Figura 4.35.-*** *Filtro complementario combinación del bloque acelerómetro y giróscopo. Fuente: Propia*

En la *figura 4.35* se observó que se tiene el ángulo a la salida, y es producto del cálculo de la posición angular con respecto a la velocidad y a la aceleración respectivamente, agregando un Scope para ver la salida y se comparó ambas configuraciones.

En la *figura 4.36* se notó diferencia entre ambos modos, siendo el de color amarillo a través de la velocidad y en azul por la aceleración.

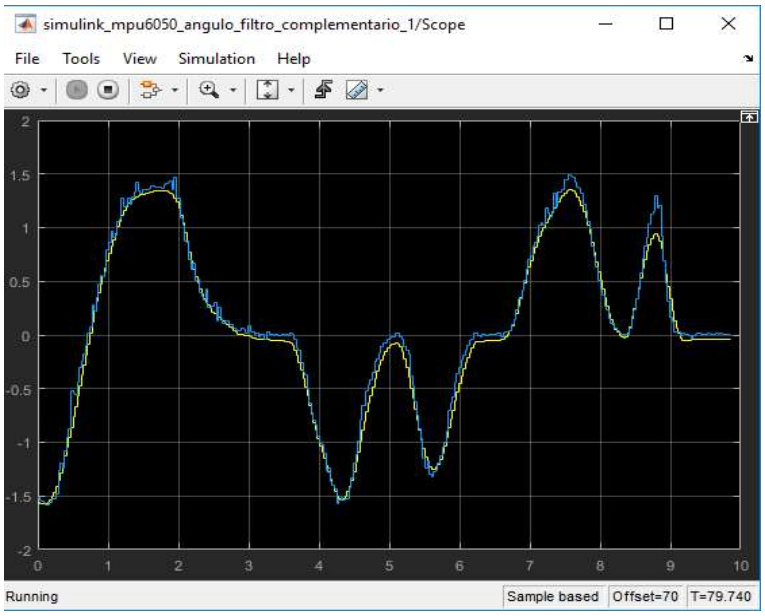



***Figura 4.36.-*** *Salida del filtro complementario en movimiento, color azul es bloque acelerómetro y el amarillo el bloque giróscopo. Fuente: Propia*

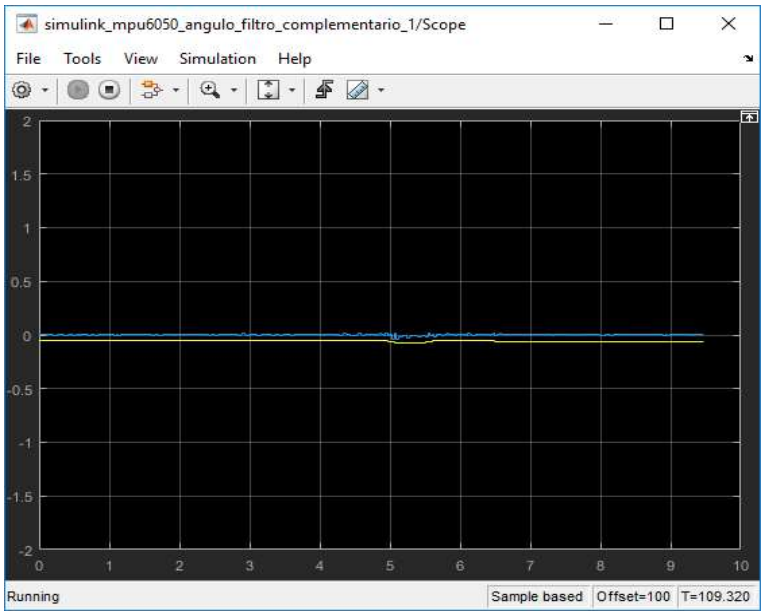



***Figura 4.37.-*** *Salida del filtro complementario en reposo, color azul es bloque acelerómetro y el amarillo el bloque giróscopo. Fuente: Propia*

En la *Figura 4.37,* se observó que el ángulo obtenido a través del cálculo por medio de la velocidad angular, se tiene menos ruido o vibraciones, que la que se utilizó por la aceleración. De manera contraria, en la *Figura 4.38*, se vio que el ángulo por medio de la velocidad angular tendió a dar un error o desfasaje, proveniente de la integral que acumula dicho error.

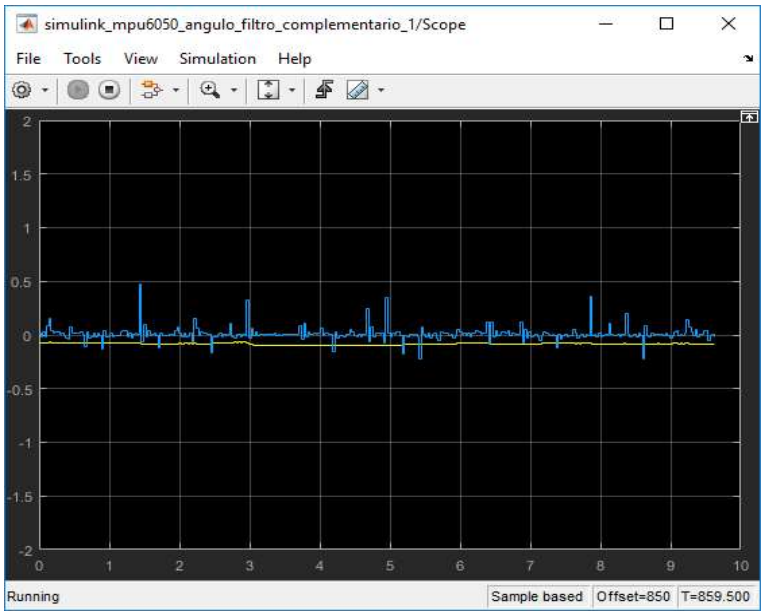



***Figura 4.38.-*** *Zoom a la salida del filtro complementario en reposo, color azul es bloque acelerómetro y el amarillo el bloque giróscopo. Fuente: Propia*

Si bien existen ventajas y desventajas sobre el uso de uno u otro, el sensor IMU MPU6050 si es perturbado, se vio que es mucho más sensible del punto de vista de la aceleración. De manera contraria, el bloque giróscopo es menos sensible, pero se desvía del cero a lo largo del tiempo.

### 4.1.4 El Filtro Complementario en si

En esta parte, se tuvo en cuenta las medidas que se obtuvieron por la velocidad angular y la aceleración angular, y se introdujo ciertos elementos que ayudaron a mejorar el filtro. En la *figura 4.39* se puede ver el diagrama completo del filtro complementario.

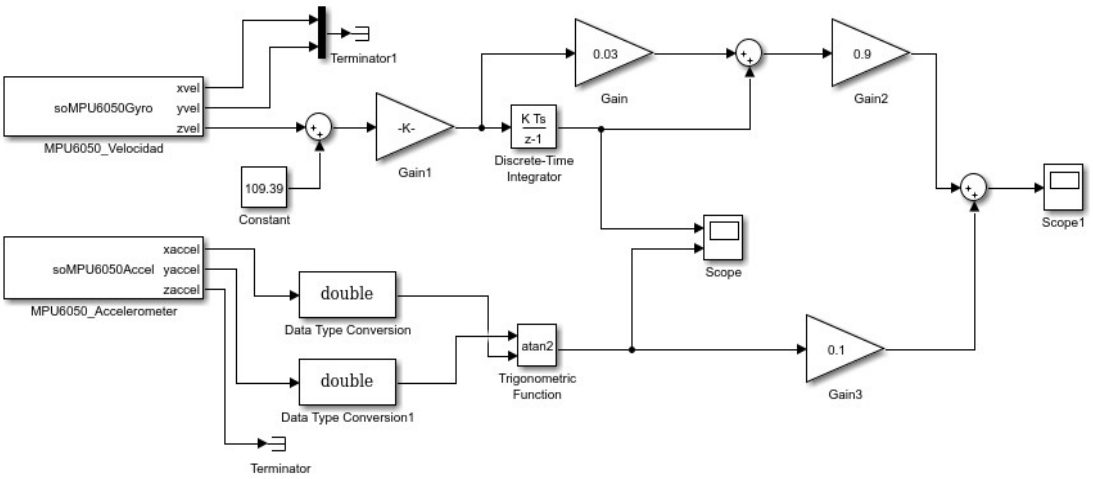



***Figura 4.39.-*** *Mejoras al filtro complementario. Fuente: Propia*

Luego de simular, se procedió a ver la respuesta que tuvo este filtro, aplicando inclinaciones del sensor en un sentido y luego en sentido opuesto. Se llevó hasta 90° (o pi/2) y luego se lo dejo en reposo para ver si existe el desfasaje que anteriormente se mencionó.

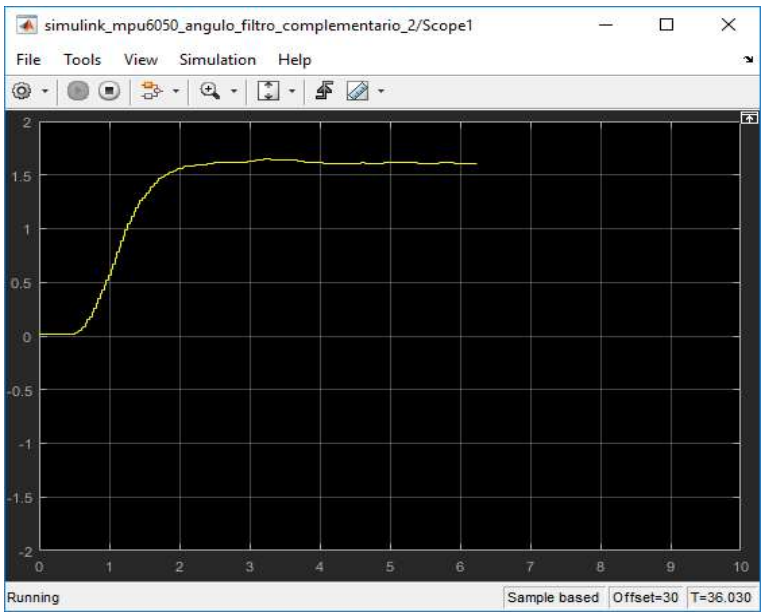



***Figura 4.40.-*** *Salida del filtro complementario mejorado con la posición del sensor a 90°. Fuente: Propia*

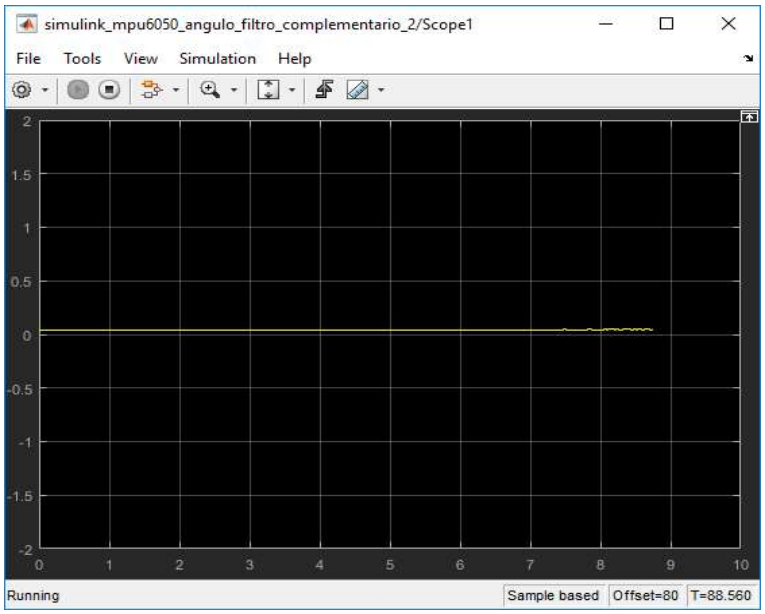



***Figura 4.41.-*** *Salida del filtro complementario mejorado con desfasaje. Fuente: Propia*

Al retornar el giroscopio, se vio que existe un pequeño desfasaje. Nuevamente, se giró hacia los -90° (o -pi/2), se observó que también llegó sin problemas al ángulo deseado. Luego se probó realizando diversos movimientos al sensor, tanto en sentido ascendente y descendente, y se corroboró que si existe el error o desfasaje.

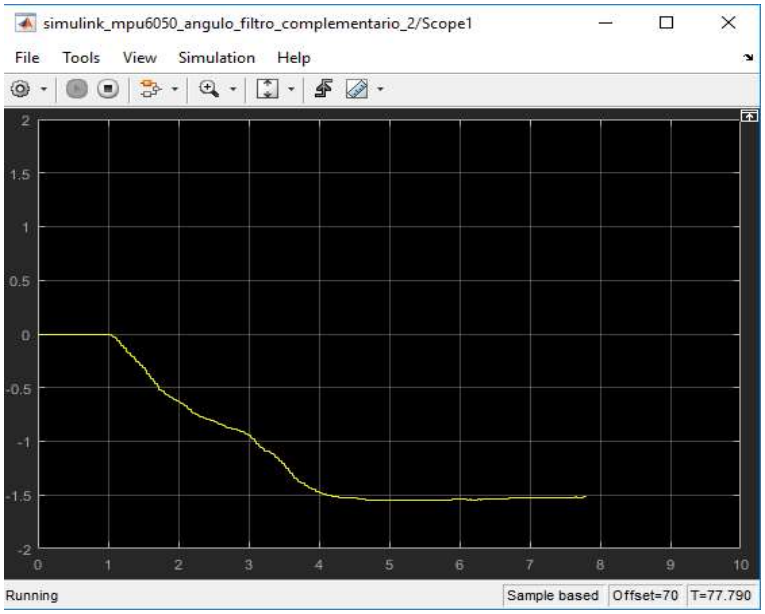



***Figura 4.42.-*** *Salida del filtro complementario mejorado con la posición del sensor a -90°. Fuente: Propia*

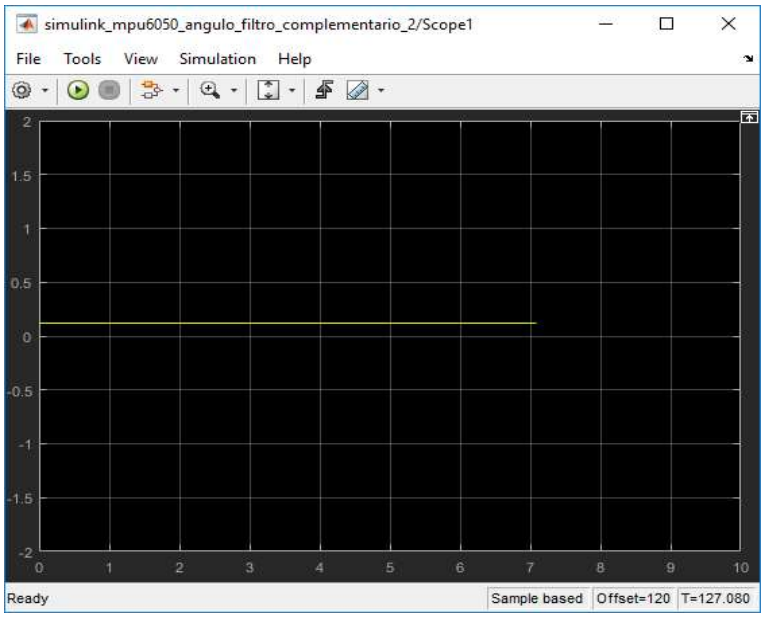



***Figura 4.43.-*** *Salida del filtro complementario mejorado con desfasaje. Fuente: Propia*

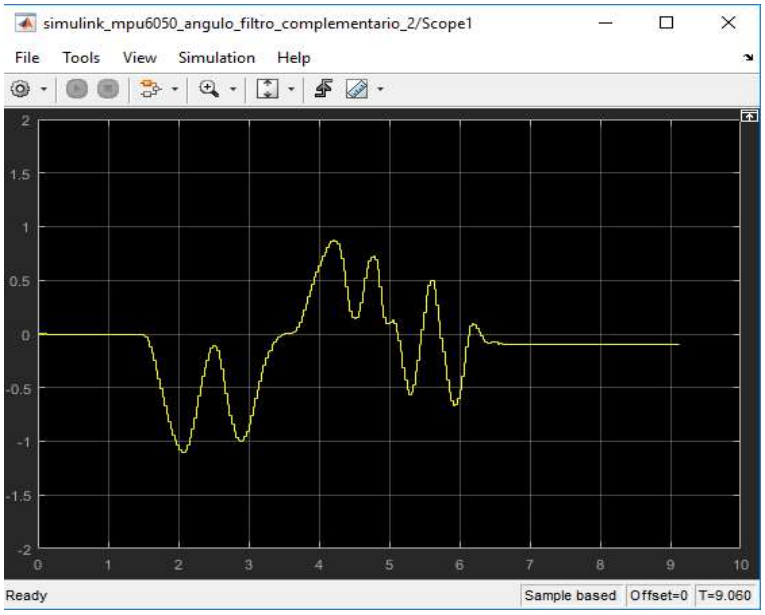



***Figura 4.44.-*** *Salida del filtro complementario mejorado con diversos movimientos. Fuente: Propia*

Evidentemente, se generó el mismo error o desfasaje cuando dejamos en reposo el sensor. A continuación, se mejoró este filtro y se modificó las ganancias, que disminuyó el error, provocado por la medición del ángulo mediante la velocidad angular.

### 4.1.5 Sintonización al Filtro Complementario

Se sintonizó o calibró el filtro, cambiando las constantes de las ganancias de tal manera se pudo eliminó en gran parte el offset mencionado anteriormente. Se cambió de 0.9 a 0.7 la ganancia del ángulo proveniente de la velocidad angular, y de 0.1 a 0.3 de la ganancia del ángulo de la aceleración angular. Siempre la suma de ambas ganancias tiene que dar igual a 1.

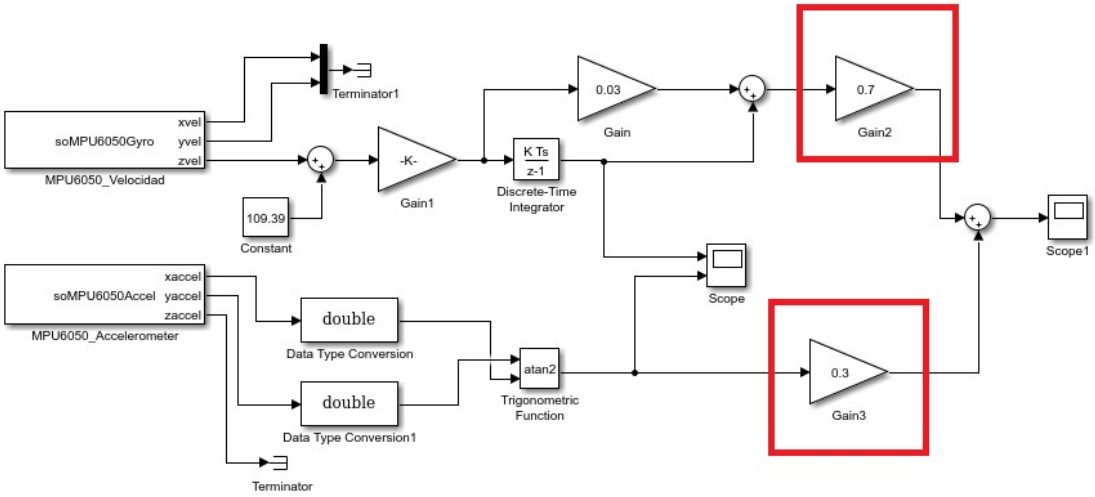



***Figura 4.45.-*** *Sintonización del filtro complementario. Fuente: Propia*

Nuevamente, se procedió a probar esta nueva modificación, girando el sensor a los 90° y luego a -90° como se muestra en la *figura 4.46* y *figura 4.47* respectivamente.

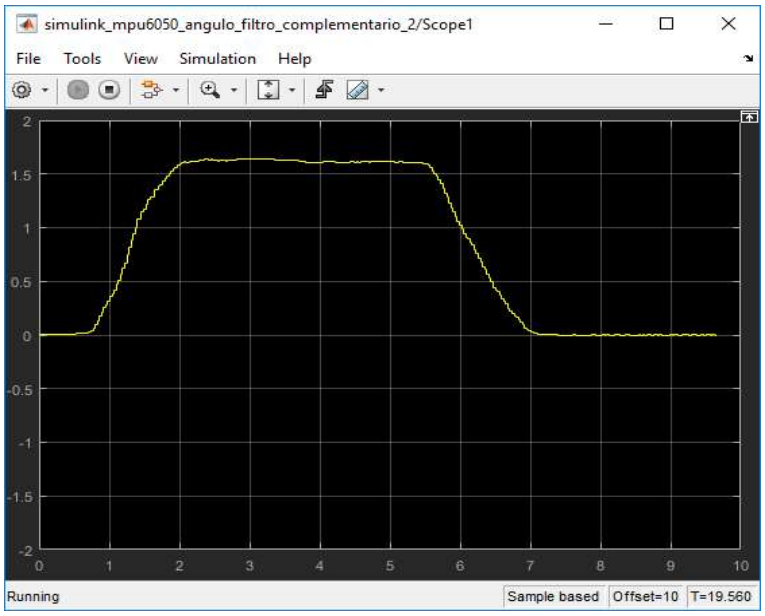



***Figura 4.46.-*** *Sintonización al filtro complementario, girando a 90°. Fuente: Propia*

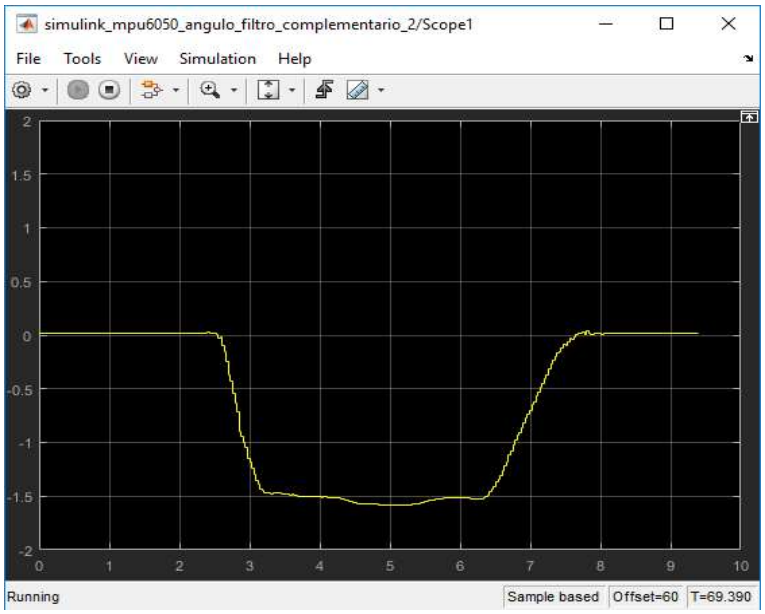



***Figura 4.47.-*** *Sintonización al filtro complementario, girando a -90°. Fuente: Propia*

A simple vista, se observó un mínimo desfasaje siendo menor que las anteriores. El motivo que se llegó a disminuir es porque se le dio más ponderación a la posición angular obtenida por la aceleración angular, y esto disminuyó ese error que era provocado por le velocidad angular. Como se pretende eliminar por completo el offset, se mejoró el filtro complementario, y se cambió algunas partes del esquema.

### 4.1.6 Mejoras al Filtro Complementario

Como una mejora el filtro, se realimentó la salida del ángulo que se desea, pero no es la misma como tal, sino que se tomó la medición anterior a la misma y se colocó un delay, que ayudó a calibrar el sistema eliminando por completo el offset.

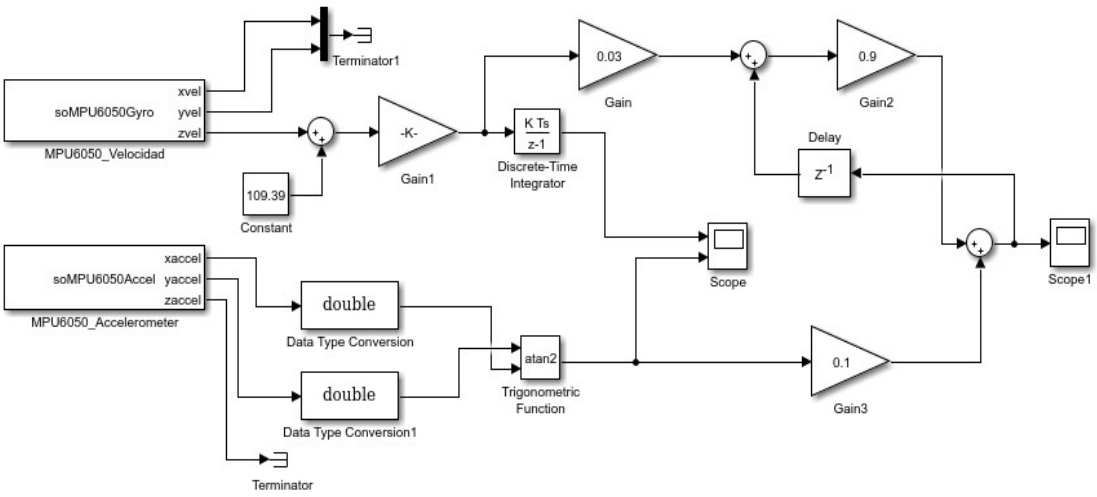



***Figura 4.48.-*** *Mejoras al filtro complementario. Fuente: Propia*

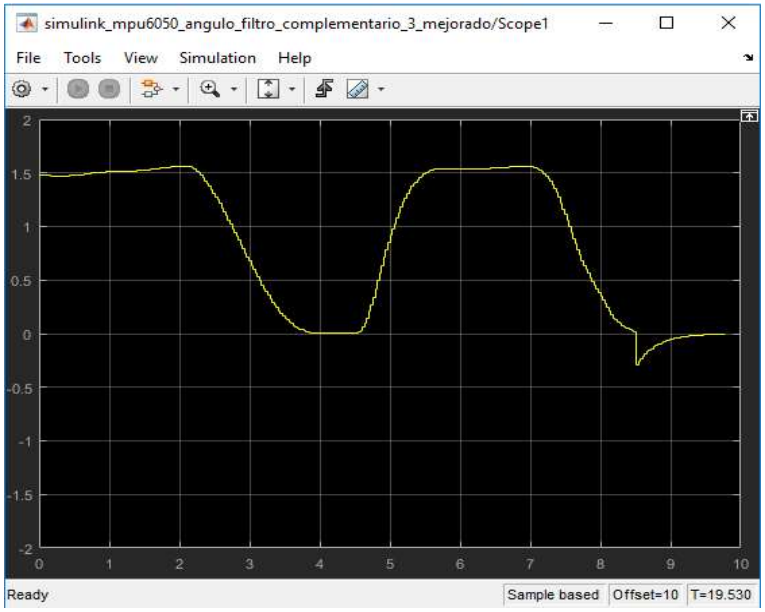



***Figura 4.49.-*** *Movimientos aleatorios del sensor IMU MPU6050. Fuente: Propia*

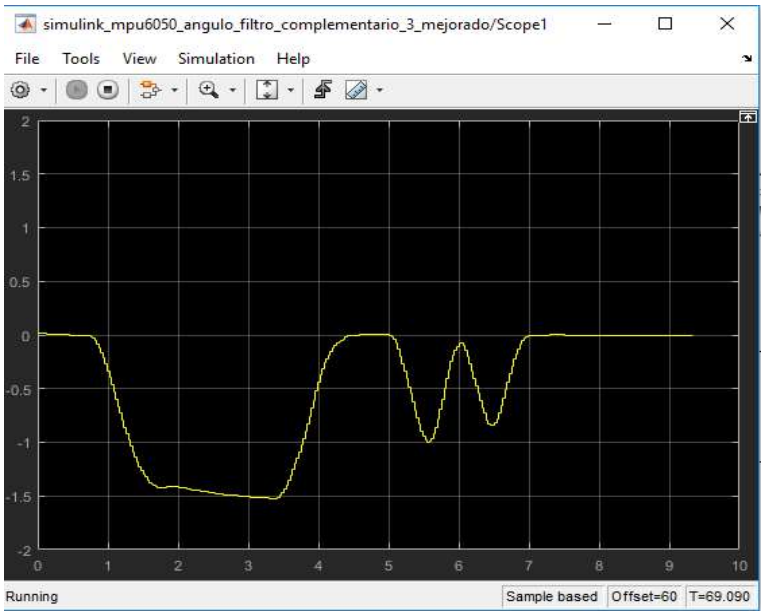



***Figura 4.50.-*** *Movimientos aleatorios del sensor IMU MPU6050. Fuente: Propia*

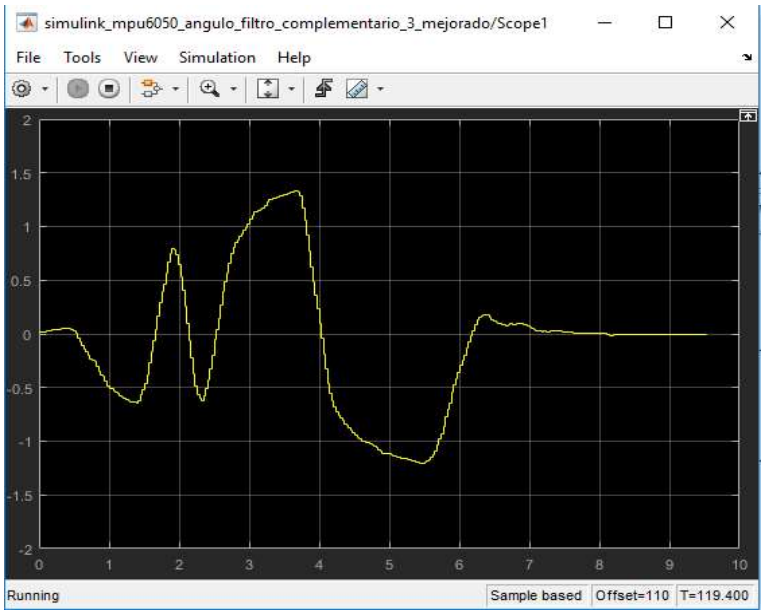



***Figura 4.51.-*** *Movimientos aleatorios del sensor IMU MPU6050. Fuente: Propia*
*Propia*

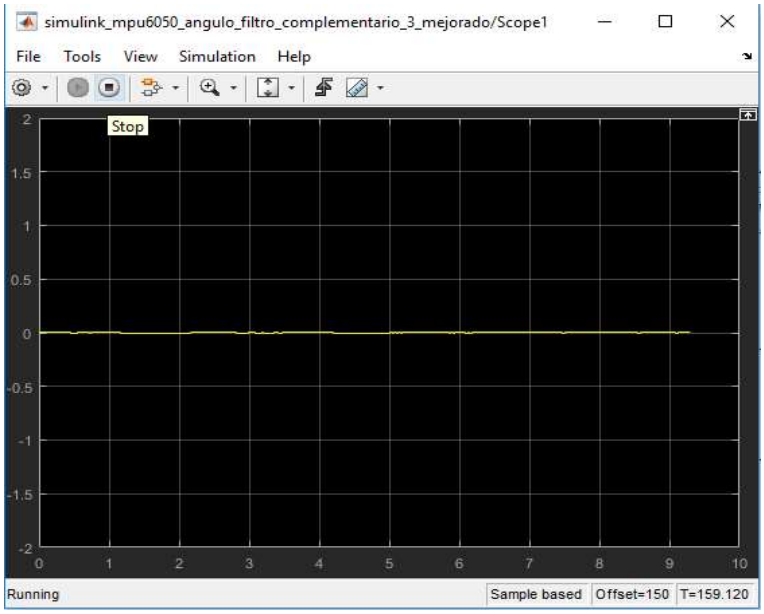



***Figura 4.52.-*** *Sensor IMU MPU6050 en reposo y con offset eliminado. Fuente: Propia*

Finalmente, el error del offset fue eliminado por completo, a partir de aquí, se tuvo el giroscopio con el filtro adecuado y mejorado, y se utilizó en las próximas pruebas.

## 4.2 Prueba de repetitividad de una IMU con servo

A continuación, se describe el ensayo realizado del conjunto compuesto por el MPU6050 y el servo seleccionado como actuador para mover el ala.

Con el fin de garantizar la confiabilidad del conjunto se ensayó una prueba de repetitividad donde la MPU selecciona un ángulo como “SetPoint” movido a mano y el servo se mueve hasta alcanzarlo. Para llevar a cabo la prueba se fabricó una maqueta que contiene una escala angular para la MPU y otra para el servo como se muestra en la *figura 4.53*.

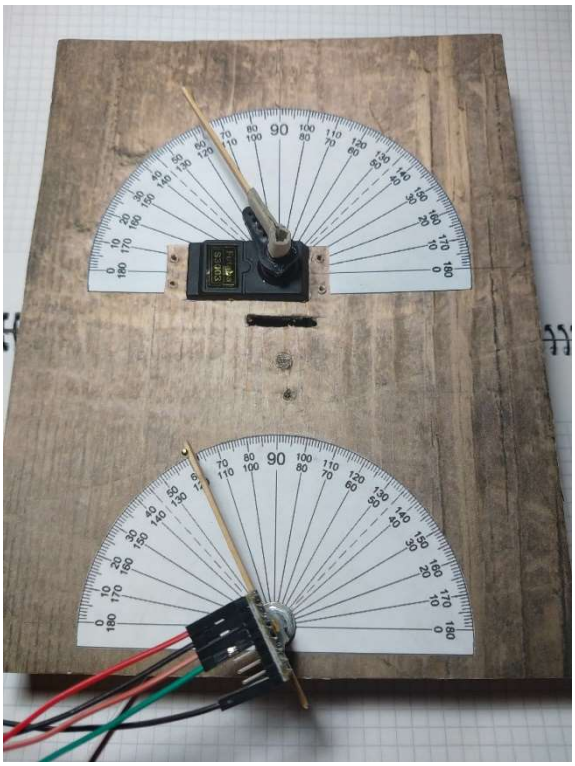


***Figura 4.53.-*** *Maqueta de prueba de repetitividad de un servo utilizando una IMU a un ángulo de 80°. Fuente: Propia.*

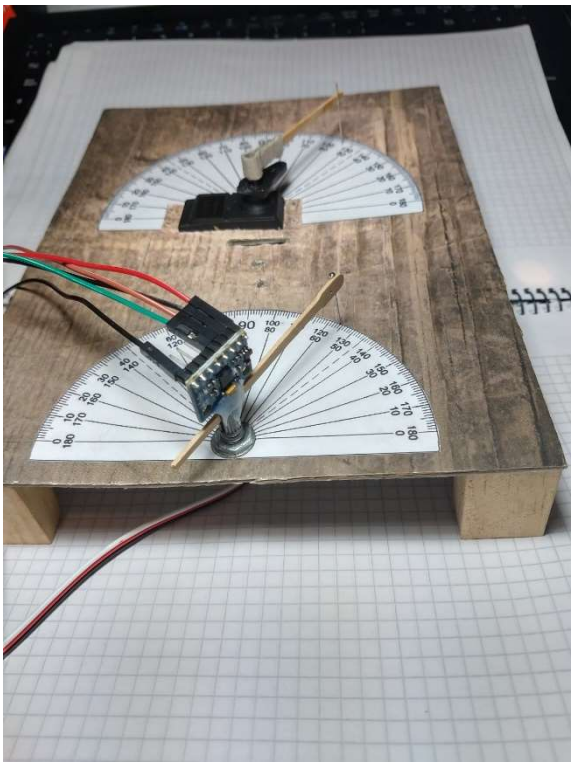


***Figura 4.54.-*** *Maqueta de prueba de repetitividad de un servo utilizando una IMU a un ángulo de 120°. Fuente: Propia*

Se realizaron diez pruebas de confiabilidad partiendo desde los 0° hasta los 110° y se obtuvieron los siguientes resultados.

| Set Point | Actuador |
|---|---|
| 110 | 112 |
| 110 | 112 |
| 110 | 113 |
| 110 | 108 |
| 110 | 112 |
| 110 | 107 |
| 110 | 107 |
| 110 | 109 |
| 110 | 111 |
| 110 | 108 |

***Tabla 4.1.-*** *Prueba de repetitividad y resultados para un ángulo de 110°. Fuente: Propia.*

.

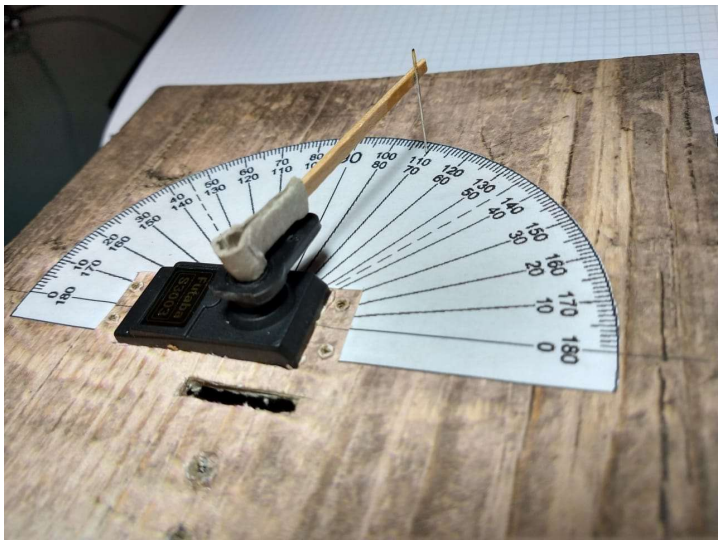


***Figura 4.55.-*** *Prueba de repetitividad de un servomotor con una IMU llevando de 0° a 110°. Fuente: Propia.*

Luego se realizaron diez pruebas más partiendo desde 0° hasta los 60° y estos fueron los resultados:

| Set Point | Actuador |
|---|---|
| 60 | 58 |
| 60 | 57 |
| 60 | 58 |
| 60 | 61 |
| 60 | 64 |
| 60 | 59 |
| 60 | 59 |
| 60 | 62 |
| 60 | 61 |
| 60 | 61 |

***Tabla 4.2.-*** *Prueba de repetitividad y resultados para un ángulo de 60°. Fuente: Propia.*

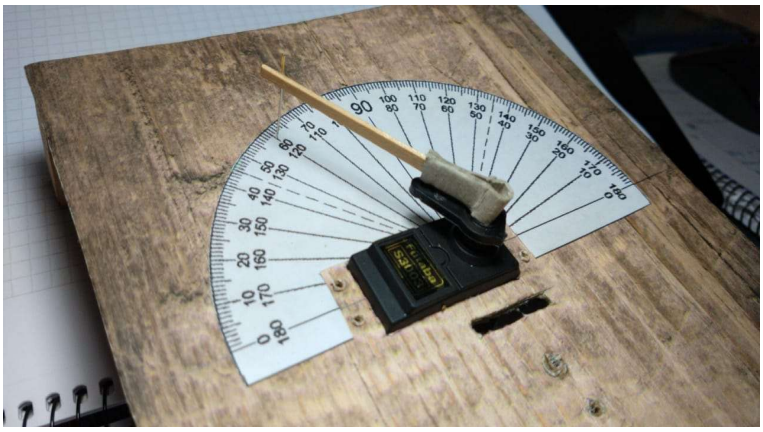


***Figura 4.56.-*** *Prueba de repetitividad de un servomotor con una IMU llevando de 0° a 60°. Fuente: Propia.*

Interpretando estos resultados podemos concluir que la precisión obtenida es más que suficiente ya que trata de un robot animatrónico y que el programa realizado en el microcontrolador funciona correctamente.

## 4.3 Ensayo del conversor DC/DC

La finalidad de este ensayo es conocer el rendimiento del conversor que se diseñó. Se midió la tensión y corriente en la entrada y salida (ver *figura 4.57*), y se colocó una carga resistiva de 100 Ω de 2,5 W.

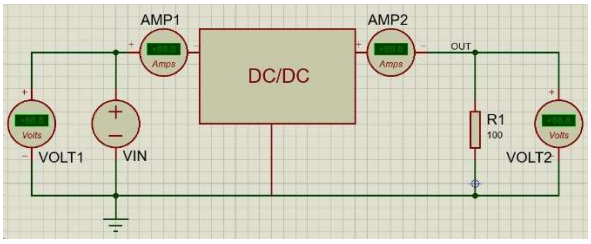



***Figura 4.57.-*** *Esquemático de mediciones del conversor. Fuente: Propia.*

Luego se hizo un barrido de tensión en la salida manteniendo fija la tensión de entrada de 5V y se tomaron las lecturas de las corrientes de entrada y de salida para calcular las potencias y así su rendimiento. Los datos se muestran en la siguiente tabla (*tabla 4.3*).

| Vin[V] | Iin[mA] | Vout[V] | Iout [mA] | RL[OHM] | Pin[W] | Pout[W] | Rend[%] |
|---|---|---|---|---|---|---|---|
| 5 | 96 | 6 | 59,6 | 100,3 | 0,48 | 0,359 | 74,776 |
| 5 | 132 | 7 | 70,3 | 100,3 | 0,66 | 0,489 | 74,020 |
| 5 | 175 | 8 | 79,6 | 100,3 | 0,875 | 0,638 | 72,924 |
| 5 | 202 | 9 | 90,2 | 100,3 | 1,01 | 0,808 | 79,958 |
| 5 | 255 | 10 | 100,5 | 100,3 | 1,275 | 0,997 | 78,197 |
| 5 | 311 | 11 | 111,2 | 100,3 | 1,555 | 1,206 | 77,581 |
| 5 | 370 | 12 | 120,6 | 100,3 | 1,85 | 1,436 | 77,605 |
| 5 | 440 | 13 | 130,6 | 100,3 | 2,2 | 1,685 | 76,588 |
| 5 | 515 | 14 | 141,1 | 100,3 | 2,575 | 1,954 | 75,889 |
| 5 | 597 | 15 | 150,2 | 100,3 | 2,985 | 2,243 | 75,151 |

***Tabla 4.3.-*** *Lecturas de tensión y corriente del ensayo del conversor. Fuente: Propia.*

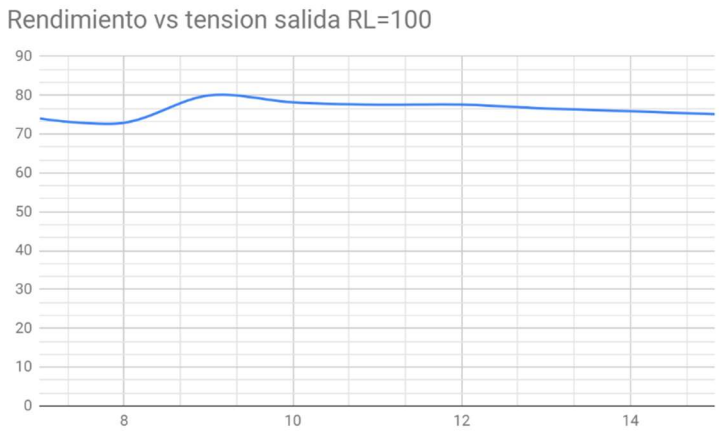



***Figura 4.58.-*** *Grafico del rendimiento del conversor. Fuente: Propia.*

Para comprobar las características eléctricas del rendimiento mencionadas en la hoja de datos, se utilizó una carga de 8.2Ω de 7W para una tensión de trabajo de 7V (que alimentaran los servos), entregando una corriente de salida de 840mA y una potencia de 5,88W, obteniendo un rendimiento del 79,8%, muy próximo a lo que dice el fabricante como se muestra en la *figura 4.59*.

| Symbol | Parameter | Conditions | Typical | LM1577-ADJ Limit(1)(2) | LM2577-ADJ Limit(3) | Units (Limits) |
|---|---|---|---|---|---|---|
| **SYSTEM PARAMETERS** Circuit of Figure 31 (4) | | | | | | |
| $V_{OUT}$ | Output Voltage | $V_{IN}$ = 5V to 10V<br>$I_{LOAD}$ = 100 mA to 800 mA(1) | 12.0 | | | V |
| | | | | 11.60/11.40 | 11.60/11.40 | V(min) |
| | | | | 12.40/12.60 | 12.40/12.60 | V(max) |
| $\Delta V_{OUT}/\Delta V_{IN}$ | Line Regulation | $V_{IN}$ = 3.5V to 10V<br>$I_{LOAD}$ = 300 mA | 20 | | | mV |
| | | | | 50/100 | 50/100 | mV(max) |
| $\Delta V_{OUT}/\Delta I_{LOAD}$ | Load Regulation | $V_{IN}$ = 5V<br>$I_{LOAD}$ = 100 mA to 800 mA | 20 | | | mV |
| | | | | 50/100 | 50/100 | mV(max) |
| η | Efficiency | $V_{IN}$ = 5V, $I_{LOAD}$ = 800 mA | 80 | | | % |

***Figura 4.59.-*** *Rendimiento según fabricante. Fuente: Propia.*

## 4.4 Modelo Final Cableado

A continuación, se presenta el modelo final del animatrónico, controlado por la placa de control que lee el valor de las IMUs y lo envía directamente con cables a la placa principal y de allí a los servos.

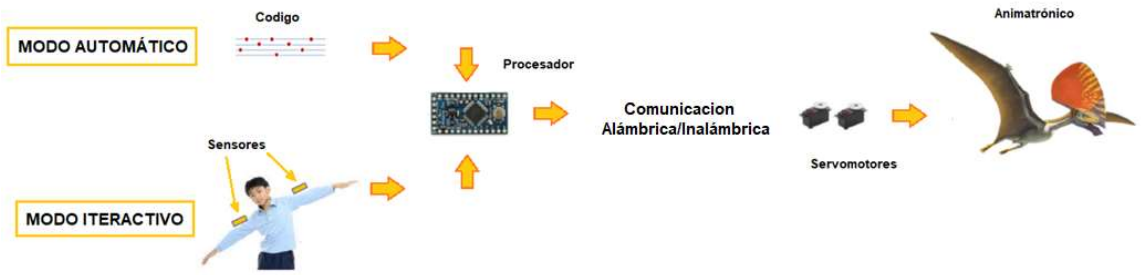



***Figura 4.60.-*** *Esquema modelo final del proyecto. Fuente: Propia.*

### 4.4.1 Esquema de conexión

Antes de construir los PCBs, fue necesario un esquema funcional que consiste en la interconexión de módulos para desarrollar y validar tanto el código como los pines asignados del microcontrolador. En las siguientes dos imágenes se muestran las conexiones de todos los elementos (ver *figura 4.61 y 4.62*).

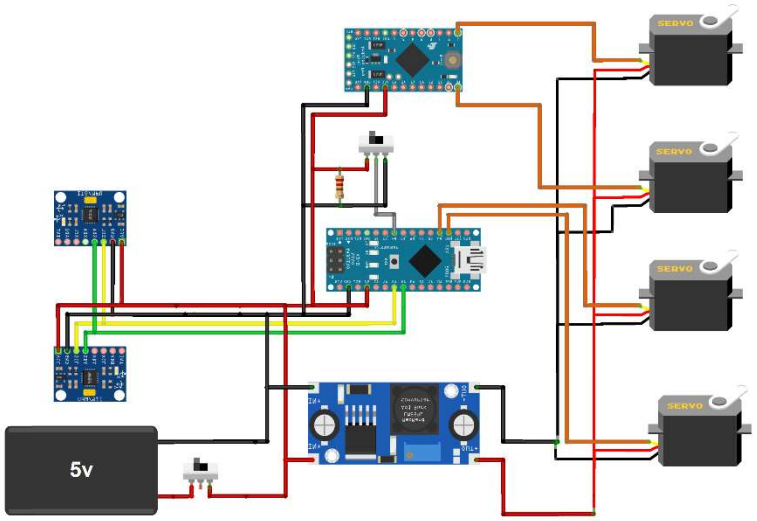



***Figura 4.61.-*** *Esquema de conexión. Fuente: Propia.*

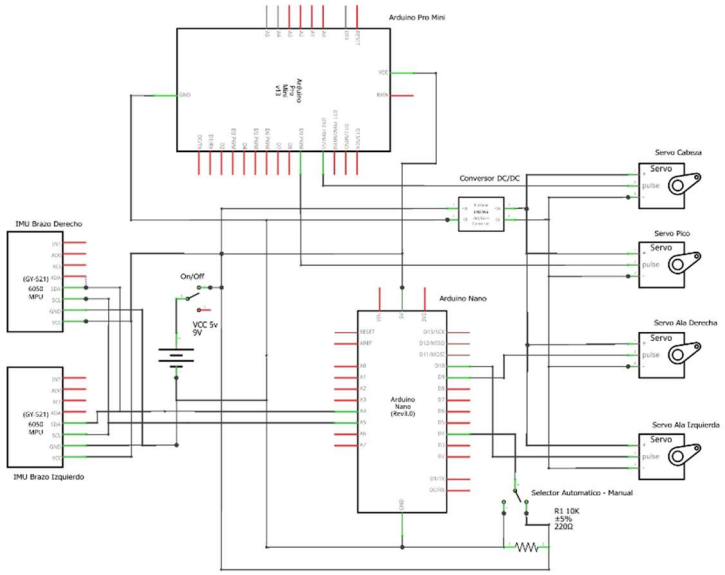



***Figura 4.62.-*** *Esquemático modelo final del proyecto. Fuente: Propia.*

El diseño, está equipado con dos placas desarrolladoras que poseen un microcontrolador que maneja al animatrónico. Una de estas placas, es la encargada de mover las alas, de procesar la información que proviene de las IMUs y de seleccionar el modo de operación de trabajo: modo automático o modo interactivo. Por otro lado, la segunda placa de desarrollo tiene la finalidad de controlar los servomotores de la cabeza y el pico por medio de señales PWM.

La alimentación utilizada para los microcontroladores y las IMUs son provista por una fuente externa switching de +5V. En cambio, los servomotores trabajan con una alimentación +7V mediante un conversor DC/DC diseñado especialmente para este caso (ver 3.2 "*diseño de la fuente"* del Marco Metodológico).

Los servos utilizados son los MG 996R de marca TowerPro de alto torque. Sus bornes de conexión de tres pines permiten conectar los +7V, GND y la señal de control del PWM. Se utilizan cuatro servos; dos para las alas, uno para la cabeza y el ultimo para el pico.

| Placa1 | IMU_1 MPU6050 | IMU_2 MPU6050 | Servo_1 | Servo_2 |
|---|---|---|---|---|
| 5 v | Vcc | Vcc | - | - |
| GND | GND | GND | - | - |
| Pin A4 | SDA | SDA | - | - |
| Pin A5 | SCL | SCL | - | - |
| Pin 9 | - | - | Naranja | - |
| Pin 10 | - | - | - | Naranja |

***Tabla 4.4.-*** *Conexiones de Placa1. Fuente: Propia.*

| Placa2 | Servo_3 | Servo_4 |
|---|---|---|
| Pin 9 | Naranja | - |
| Pin 10 | - | Naranja |

***Tabla 4.5.-*** *Conexiones de Placa2. Fuente: Propia.*

Tal y como se muestra en la *figura 4.61 y 4.62*, los servos de las alas se conectan a los pines D9 y D10 de la Placa1 (ala derecha y ala izquierda respectivamente), y en los pines D9 y D10 de la Placa2 se conectan los servos del cabeza y pico.

Siguiendo la misma *figura 4.61*, para la conexión de los pines de comunicación SDA y SCL de las IMUs MPU6050 al Arduino Nano es el pin A4 y A5 de la misma. Además, un led testigo en el pin D13 señala en el modo de operación de trabajo del animatrónico (modo automático o interactivo).

### 4.4.2 Desarrollo PCBs

Los diseños de las placas fueron realizados a partir de las dimensiones predefinidas por la estructura torácica del animatrónico, teniendo en cuenta las cavidades y soportes para la ubicación de los componentes, y perforaciones de fijación.

#### 4.4.2.1 Placa Principal

Se diseñó una placa principal que se alojará dentro del cuerpo del animatrónico, e integrará todo el hardware interconectando todos los dispositivos utilizados (tarjetas de desarrollo, pines conexión de servos, etc) que sirven para su funcionamiento. Esta placa se diseñó con el software Altium Designer 2018, y permitió exportar un archivo 3D que sirvió para presentarla en el diseño de la estructura y ver que coincidieran las perforaciones y que no se toquen los componentes con la estructura, ya que se contaba con un espacio bastante reducido por las dimensiones del animatrónico.

En el diseño del Layout se tuvo en cuenta la densidad de corriente para el cálculo del ancho de pista, como así también el cuidado del ruteo de señales de comunicación como por ejemplo el par diferencial del puerto serial. Además, se creó un plano de masa para generar un blindaje y disminuir el ruido eléctrico evitando así interferencias de señales. La placa es de doble faz permitiendo introducir los componentes sobre ambas caras de la placa optimizando el espacio (*ver figura 4.63 a y 4.63 b*).

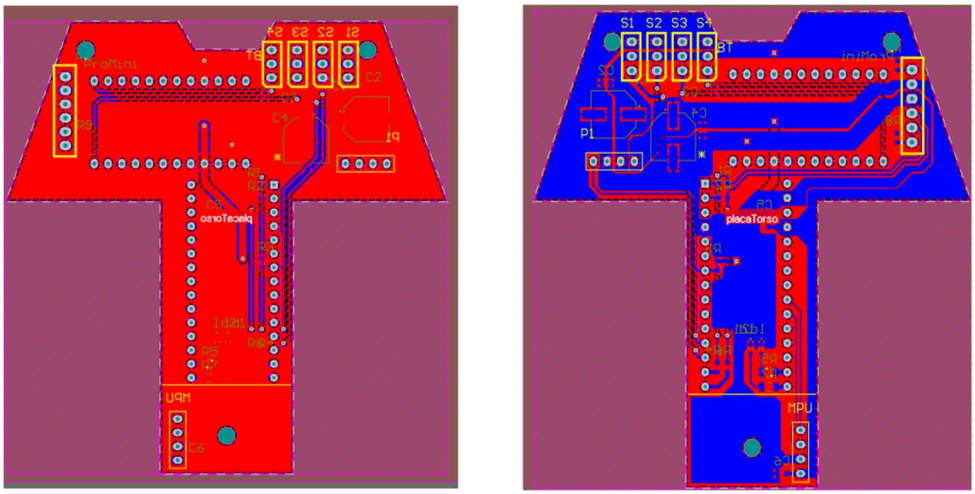



***Figura 4.63.-*** *Imagen layout de la placa principal. (a) Izquierda: Top Layer (Capa superior). (B)Derecha: Bottom Layer (Capa inferior). Fuente: Propia.*

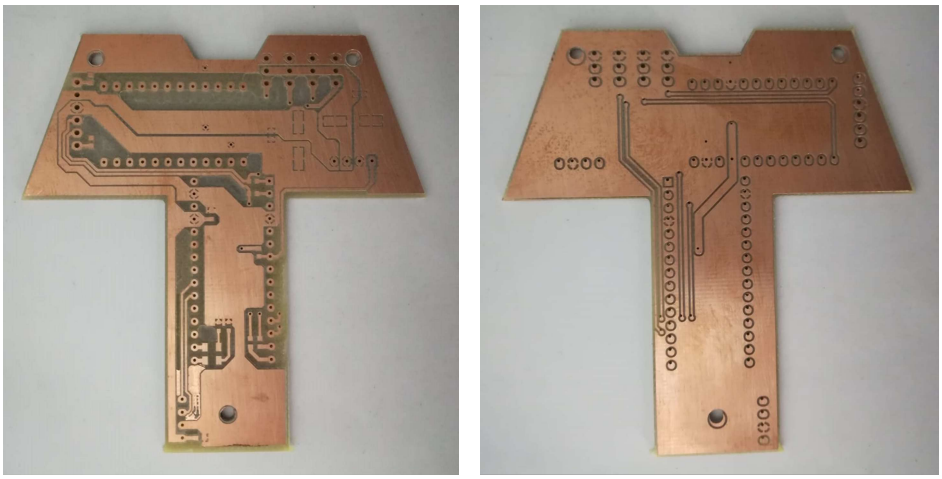


***Figura 4.64.-*** *Vista de la placa principal impresa. (a) Izquierda: Top Layer (Capa superior). (B)Derecha: Bottom Layer (Capa inferior). Fuente: Propia.*

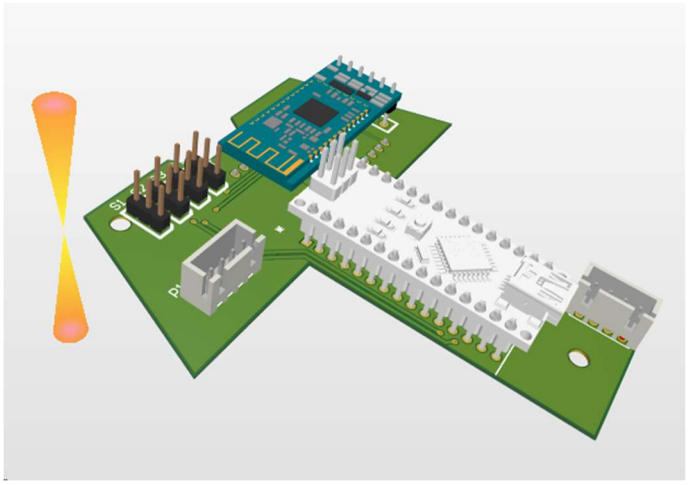


***Figura 4.65.a)-*** *Vista 3D de la placa principal Top Layer (Vista superior). Fuente: Propia.*

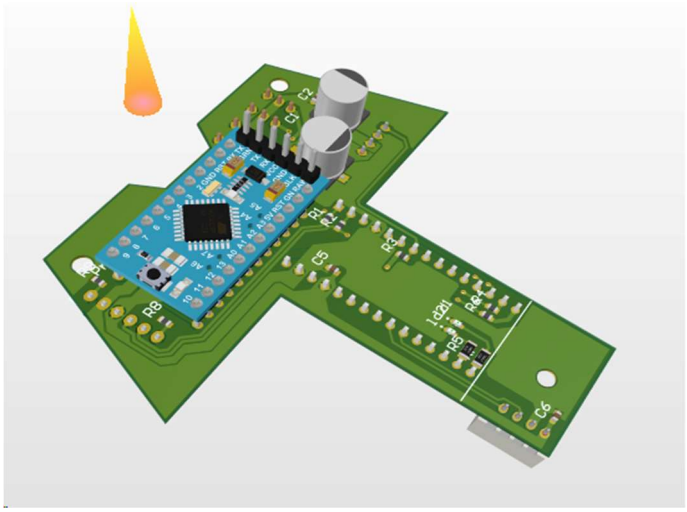


***Figura 4.65.b)-*** *Vista 3D de la placa principal Bottom Layer (Vista inferior). Fuente: Propia.*

***Figura 4.66. –*** *Imagen armado placa principal con microscopio. Fuente: Propia.*

#### 4.4.2.2 Placa de Control

Además del diseño de la placa principal, también se diseñó una placa de control (*ver figura 4.67, 4.68 a) y 4.68 b*) que tiene por finalidad la interconexión de los módulos IMUs MPU6050, y mediante un cableado externo se vinculan con la placa principal. Por otro lado, cuenta con todo el hardware necesario para enviar los datos de las IMUs vía inalámbrica por medio de los módulos bluethoot HC-05 teniendo ésta doble función. (*ver apartado 4.5 Modelo Final Inalámbrico*).

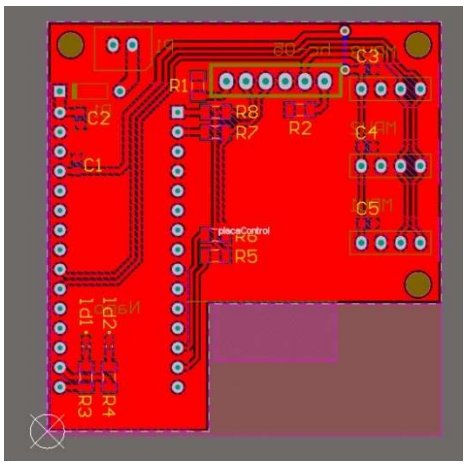



***Figura 4.67. –*** *Imagen Layout placa de control. Fuente: Propia.*

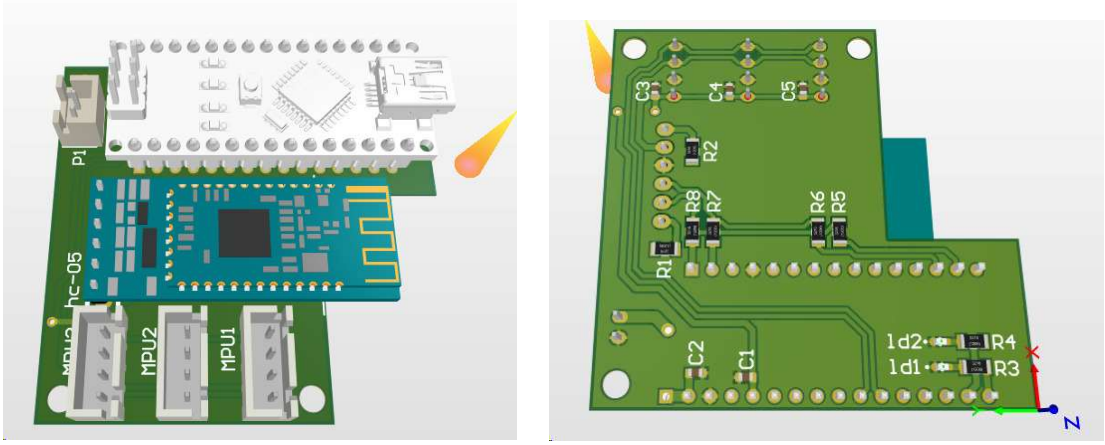



***Figura 4.68. –*** *Vista 3D placa de control (a)Izquierda: Top Layer (Vista superior) y (b) Bottom Layer (Vista inferior). Fuente: Propia.*

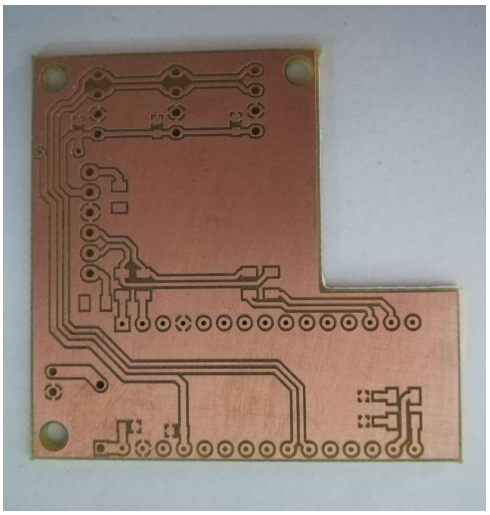


***Figura 4.69. –*** *Foto Placa de control impresa. Fuente: Propia.*

### 4.4.3 Diseño del cuerpo

El cuerpo del animatrónico, tal y como se observa en la *figura 4.70*, se realizó con una impresora 3D y previamente fue diseñado por un “Paleoartista” especialista en modelos mecánicos con un software denominado SolidWorks. Las dimensiones de reptil representado está en escala **5:1** y se tuvo que tener en cuenta la posición y alojamiento de los servomotores, de tal manera que el movimiento de las alas sea libre de rozamientos.

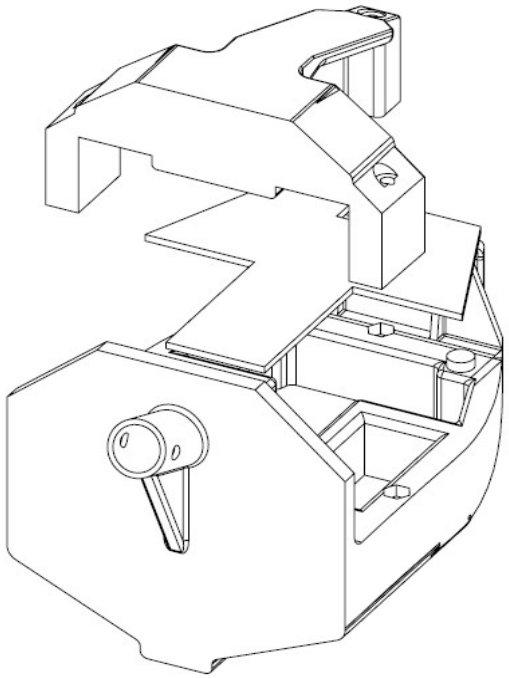


***Figura 4.70.-*** *Vista 3D del cuerpo del animatrónico realizado en SolidWorks. Fuente: Propia.*

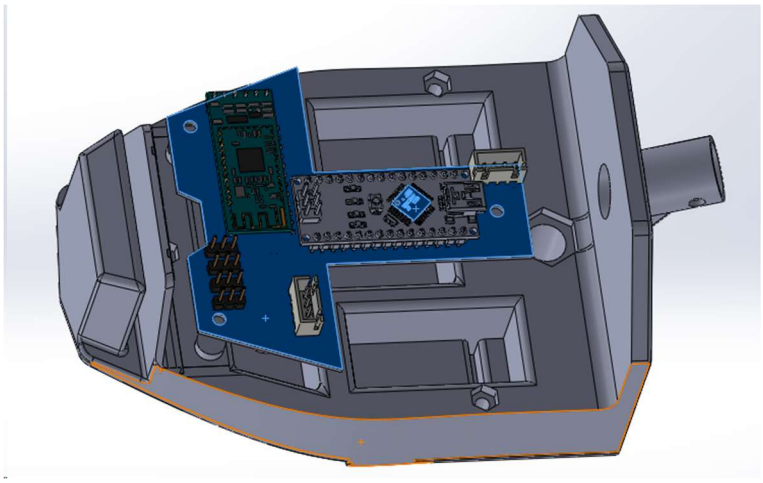


***Figura 4.71 a).-*** *Vista 3D superior placa principal y cuerpo del animatrónico en SolidWorks. Fuente: Propia.*

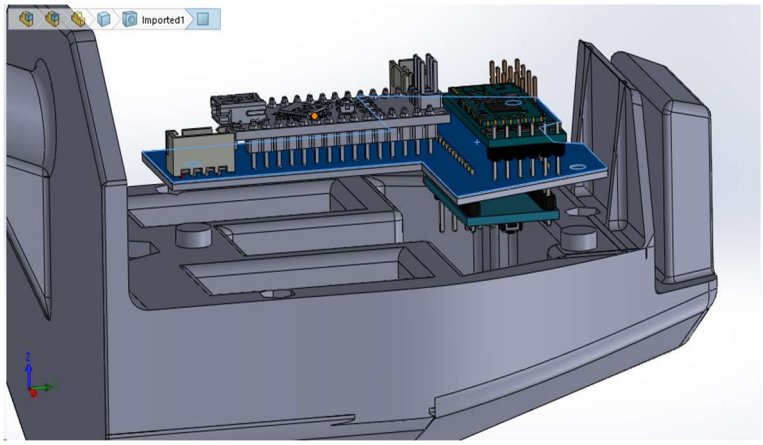



***Figura 4.71 b).-*** *Vista 3D lateral placa principal y cuerpo del animatrónico en SolidWorks. Fuente: Propia.*

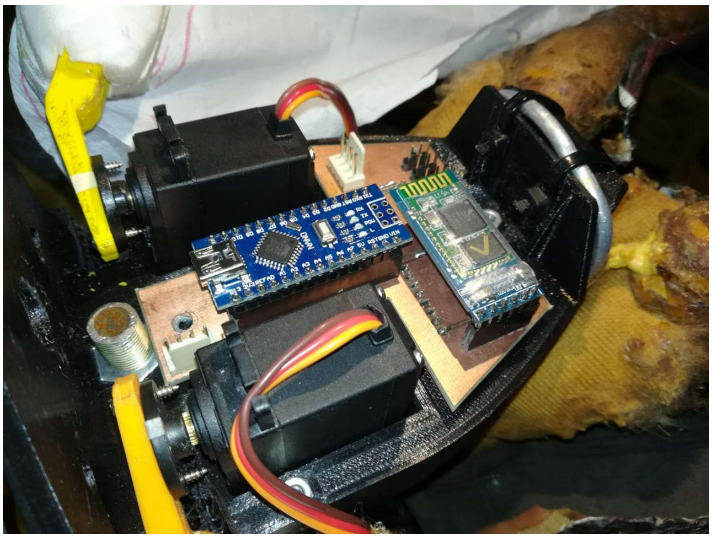


***Figura 4.72.-*** *Foto placa principal dentro del tórax del animatrónico. Fuente: Propia.*

### 4.4.4 Soporte del Ala

No solamente el cuerpo del animatrónico fue la única pieza en ser impresa en 3D, sino también los soportes de las alas. Se tuvo que construir un suplemento especial capaz de sostener el ala y acoplarlo al servomotor (*ver figura 4.73*). De esta manera, resulto importante un diseño para que el mismo sea genérico, y sea sujetado por cualquiera de los accesorios incluidos (brazo dos puntas, acople circular, acople en cruz, etc) de los servomotores. Para ello, el soporte debió contener varios orificios distanciados unos de otros y así se pudiera coincidir en al menos en dos, para ser atornillado

.

***Figura 4.73.-*** *Soporte de ala atornillado a un acople circular de un servomotor. Fuente: Propia.*

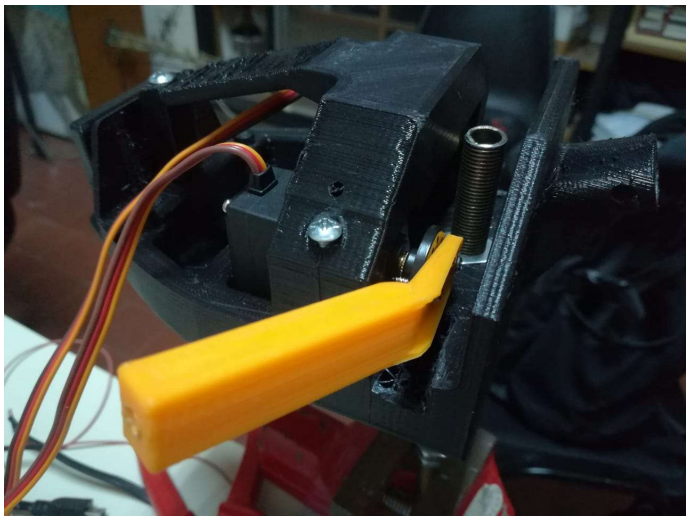


***Figura 4.74-*** *Soporte sujeto en el servomotor dentro del cuerpo del animatrónico. Fuente: Propia.*

### 4.4.5 Construcción de Alas

Las alas, están construidas con caño hueco de 4mm de diámetro y 1mm de espesor en aluminio, simulando ser parte del hueso, y un agregado de tela supone ser los músculos parte de hueso y la piel como se ve en la *figura 4.75.a*). Luego fue pintada con colores cuidadosamente seleccionados por el artista para cuidar la imagen del reptil (*figura 4.75 b*)

***Figura 4.75 a)-*** *Imagen construcción del Ala. Fuente: Propia.*

***Figura 4.75 b)-*** *Imagen del Ala final pintada. Fuente: Propia.*

### 4.4.6 Programación

El código final completo, se encuentra en Anexo I *apartado 5.1 "Código final"*

## 4.5 Modelo final inalámbrico.

Aquí se buscó desarrollar la transmisión de los datos en forma inalámbrica pensando en la muestra del animatrónico en su lugar de exposición, ya sea un parque o un museo, y mejorando la experiencia tanto del usuario como del público, ya que evita el uso de un cableado molesto a la hora de exponer.

Esta comunicación es vía bluetooth utilizando un módulo HC-05 (*ver punto 2.7 del Marco Teórico* sobre más información del dispositivo).

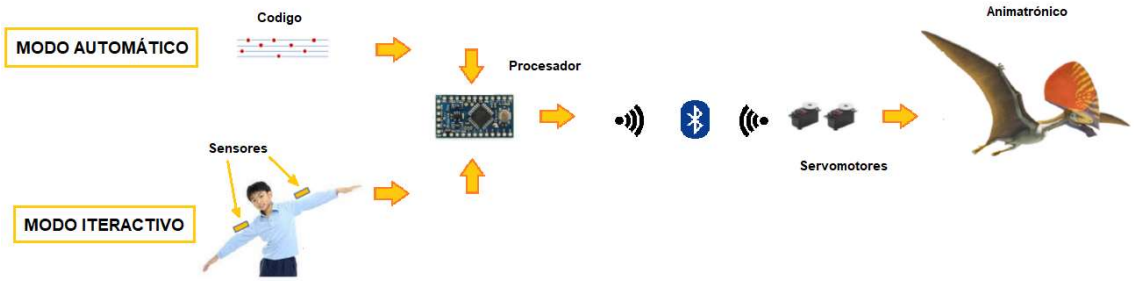



***Figura 4.76.-*** *Esquema modelo final inalámbrico. Fuente: Propia.*

Para mayor simplicidad, al inicio, se dividió el proceso en el manejo y control de un ala para luego replicarlo en la otra. Si bien la idea era realizar un control inalámbrico completo, al no llegar a cumplir con los tiempos dispuestos se optó por la versión cableada que ya se encontraba en una versión final estable. No obstante, a la hora de diseñar los PCBs se incluyó todo el hardware necesario para establecer el manejo inalámbrico, faltando únicamente, implementar el código que fue desarrollado en la transmisión inalámbrica de un ala, y extenderlo para la otra.

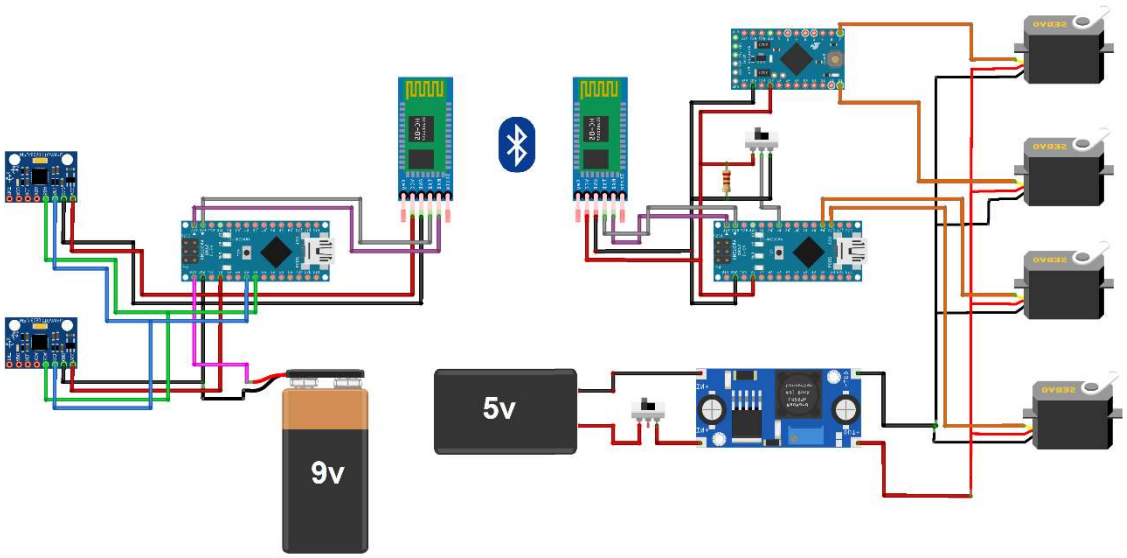



***Figura 4.77.-*** *Diagrama electrónico del modelo final inalámbrico. Fuente: Propia.*

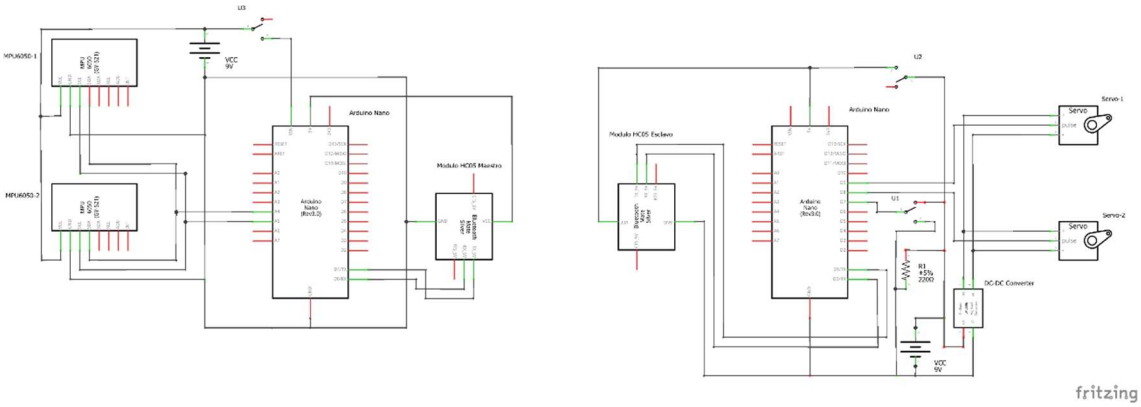



***Figura 4.78.-*** *Esquema electrónico del modelo final inalámbrico. Fuente: Propia*

## Montaje final

***Figura 4.79.-*** *Vista frontal animatrónico terminado. Fuente: Propia*

***Figura 4.80.-*** *Vista en perspectiva animatrónico terminado. Fuente: Propia*

***Figura 4.81.-*** *Vista lateral animatrónico terminado. Fuente: Propia*

## Resultados

El rediseño con el uso de nueva tecnología en el sentido del control mediante una persona (titiritero) en vez de un movimiento automático y repetitivo se cumplió correctamente.

Los modos de control mediante captura de movimiento y el de rutina pregrabada están implementados en su totalidad en el sistema propuesto.

La implementación con nueva tecnología respetó también la morfología del animal como se esperaba.

La estabilización del sistema se logró con un servomotor y no con un conjunto motorreductor/IMU debido a imposibilidad de modelar el motorreductor con varias técnicas propuesta y desarrolladas.

Las pruebas de repetitividad satisfacen el comportamiento del sistema.

El montaje del nuevo sistema a un bastidor diseñado ad hoc permite un fácil acceso al sensado y mantenimiento del sistema respetando las dimensiones originales.

## Conclusiones

El sistema funciona de acuerdo con los objetivos propuestos.

Se logró solucionar el problema de un sistema repetitivo para pasar a un sistema de captura de movimiento interactivo con nueva tecnología como objetivo principal del proyecto

Se logró diseñar una placa electrónica de modo que cupiese en el volumen torácico pequeño del animal

Si bien se ocupó mucho tiempo de estudio y desarrollo en la implementación de un sistema motorreductor/IMU realimentado no llegando a resultados favorables, se repasaron todos los tópicos de control realimentado, se hicieron modelos matemáticos en matlab y simulik con pruebas físicas de laboratorios aprendiendo nuevas estrategias de control, pero sin poder estabilizar el sistema.

El proyecto es una plataforma conceptual, física y electrónica para poder realizar futuros estudios y desarrollos que pueden mejorar el sistema sin necesidad de rediseñar completamente el hardware.

Se puede mejorar el firmware agregando un nuevo modo de trabajo que permita capturar los movimientos del titiritero grabándolos para su posterior reproducción.

Sólo insertando un módulo de conexión inalámbrica y modificando el firmware de la placa principal y la de control se tiene un sistema de captura de movimiento sin cables.

Se pueden realizar nuevos filtros de software para estudiar los resultados en la eficiencia y velocidad de la captura de movimiento.

## Bibliografía

(1) Barrientos, A, Peñin, F, Balaguer, C y Aracil, R. (2007). “Fundamentos de Robótica”. Madrid, España. Editado por S.A. MCGRAW-HILL / INTERAMERICANA DE ESPAÑA. ISBN 9788448156367.

(2) Milella, A, Cicirelli, G. (2010). “Mechatronic Systems Simulation Modeling and Control”. Editado por InTech. ISBN 978-953-307-041-4.

(3) Nourbakhsh, S, Illah, R. (2004). “Introduction to Autonomous Mobile Robots”. Editado por Cambridge (Mas.) ; London : MIT Press. ISBN 0262015358 / 9780262015356.

(4) Reyes, F. (2012). “Matlab Aplicado a Robótica y Mecatrónica”. México DF. Editado por Alfaomega Grupo Editor S.A. ISBN 9788426718365.

(5) Reyes, F. (2011). “Robótica, Control de Robots Manipuladores”. México DF. Editado por Alfaomega Grupo Editor S.A. ISBN 978-607-707-190-7.

(6) Ollero Baturone, A. (2001). “Robótica: Manipuladores y Robots Móviles. Barcelona, España. Editado por Marcombo S.A. ISBN 84-267-1313-0.

(7) Craig, J. (2006). “Robótica”. Naucalpan de Juárez, Edo. de México. Editado por Pearson Educación. ISBN 970-26-0772-8.

(8) Ogata, K.(2010). “Ingeniería de Control Moderna 5ta Edición”. Madrid, España. Editado por Pearson Educación S.A. ISBN 9788483226605.

(9) Ogata, K. (2005). “Sistemas de Control en Tiempo Discreto. 2da Edición”. México. Editado por Pearson- Prentice Hall. ISBN 968-880-539-4.

(10) DC/DC Book of Knowledge. Practical tips for the User. Steve Roberts, M.Sc. B.Sc. Technical Director, RECOM

## Referencias


[1] https://uvadoc.uva.es/bitstream/10324/12884/1/TFG-P-161.pdf

[2]https://howtomechatronics.com/how-it-works/electrical-engineering/mems-accelerometer-gyrocope-magnetometer-arduino/

[3] https://www.digikey.com/en/articles/techzone/2012/jul/a-designers-guide-to-mems-sensors

[4] http://robots-argentina.com.ar/Comunicacion_busI2C.htm

[5] https://teslabem.com/learn/fundamentos-del-protocolo-i2c-aprende/

[6] https://hetpro-store.com/multiplexor-tca9548a-i2c/

[7] https://dspace.ups.edu.ec/bitstream/123456789/4785/1/UPS-CT002640.pdf

[8] https://unefa.edu.ve/

[9]http://www.seguridadmobile.com/bluetooth/especificacion-bluetooth/estandar-bluetooth/index.html

[10] "*Tecnologías inalámbricas de corto alcanza Zibee y Bluetooth*". Tesis para la obtención del título de Ingeniero Electrónico. Universidad Politécnica Salesiana, Cuenca, Ecuador. Noviembre 2006.

[11] "*Wireless Personal Área Network (WPAN) & Home Networking*" Tesos profesional para la obtención del título de Ingeniera Electrónica. Universidad de las Américas de Puebla, México 2003.

[12] "*Análisis de la Tecnología Bluetooth en la formación de redes WPAN empleando dispositivos móviles*". Proyecto de grado para la obtención del título en ingeniería. Escuela Politécnica del Ejercito, Sangolquí, Ecuador. Abril 2011.

[13] https://altronics.cl/mod-bluetooth-hc05

[14] https://www.monografias.com/trabajos60/servo-motores/servo-motores.shtml

[15] https://es.wikipedia.org/wiki/Servomotor_de_modelismo

[16] Regulado de tensión lineal básico

[17] Filtro Complementario para estimación de actitud aplicado al controlador embebido de un cuatrirrotor. Congreso Argentino de Sistemas Embebidos. Buenos Aires. Marzo 2011.
https://www.researchgate.net/publication/275033545_Filtro_complementario_para_estimacion_de_actitud_aplicado_al_controlador_embebido_de_un_cuatrirrotor

[18] http://www.askix.com/pcb-cuadrotor-sin-cepillo_16.html

[19] https://es.wikipedia.org/wiki/Puente_H_(electr%C3%B3nica)

[20] Reservado para regulador de tensión

[21]https://www.recom-power.com/emea/downloads/bok/dc-dc-book-of-knowledge.html

[22] https://aprendiendoarduino.wordpress.com/category/atmel/

[23] http://arduino.com

# Anexo I

En este anexo contiene todo sobre el modelo final cableado, códigos, esquemático, listado de materiales. Además, contiene las experiencias realizadas con los sensores, servomotores, módulos inalámbricos, etc que fueron indispensables para lograr modelo deseado.

## 5.1 Sobre el modelo final

### 5.1.1 Código completo.

```
#include <Wire.h>
#include <Servo.h>
Servo alaDer;
Servo alaIzq;

int DerMax = 40;
int DerMin = 110;
int IzqMax = 140;
int IzqMin = 70;
int IniDer = 110;
int IniIzq = 70;
int AuxDer = 110;
int AuxIzq = 70;

//Direccion I2C de la IMU
#define MPU 0x68
#define MPU2 0x69
//Ratios de conversion
#define A_R 16384.0
#define G_R 131.0

//Conversion de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779

//MPU-6050 da los valores en enteros de 16 bits
//Valores sin refinar
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;
int16_t AcX2, AcY2, AcZ2, GyX2, GyY2, GyZ2;

//Angulos
float Acc[2];
float Gy[2];
float Angle[2];
```

```
float Angulo = 0;

float Acc2[2];
float Gy2[2];
float Angle2[2];
float Angulo2 = 0;

void setup() {
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);

  Wire.begin();
  Wire.beginTransmission(MPU2);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  pinMode(4, INPUT);
  pinMode(13, OUTPUT);

  digitalWrite(13, LOW);
  alaDer.attach(9);
  alaIzq.attach(10);
  Serial.begin(115200);

  alaIzq.write(IniIzq);
  alaDer.write(IniDer);
}
void alasArriba()
{
  for (int i = 1 ; i < 80 ; i = i + 1)
  {
    AuxDer = AuxDer - 1 ; //llega a 30
    AuxIzq = AuxIzq + 1; //llega a 150
    alaIzq.write(AuxIzq);
    alaDer.write(AuxDer);
    delay(13);
    Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(50);
}

void alasAbajo()
{
  for (int i = 80 ; i > 1 ; i = i - 1)
  {
```

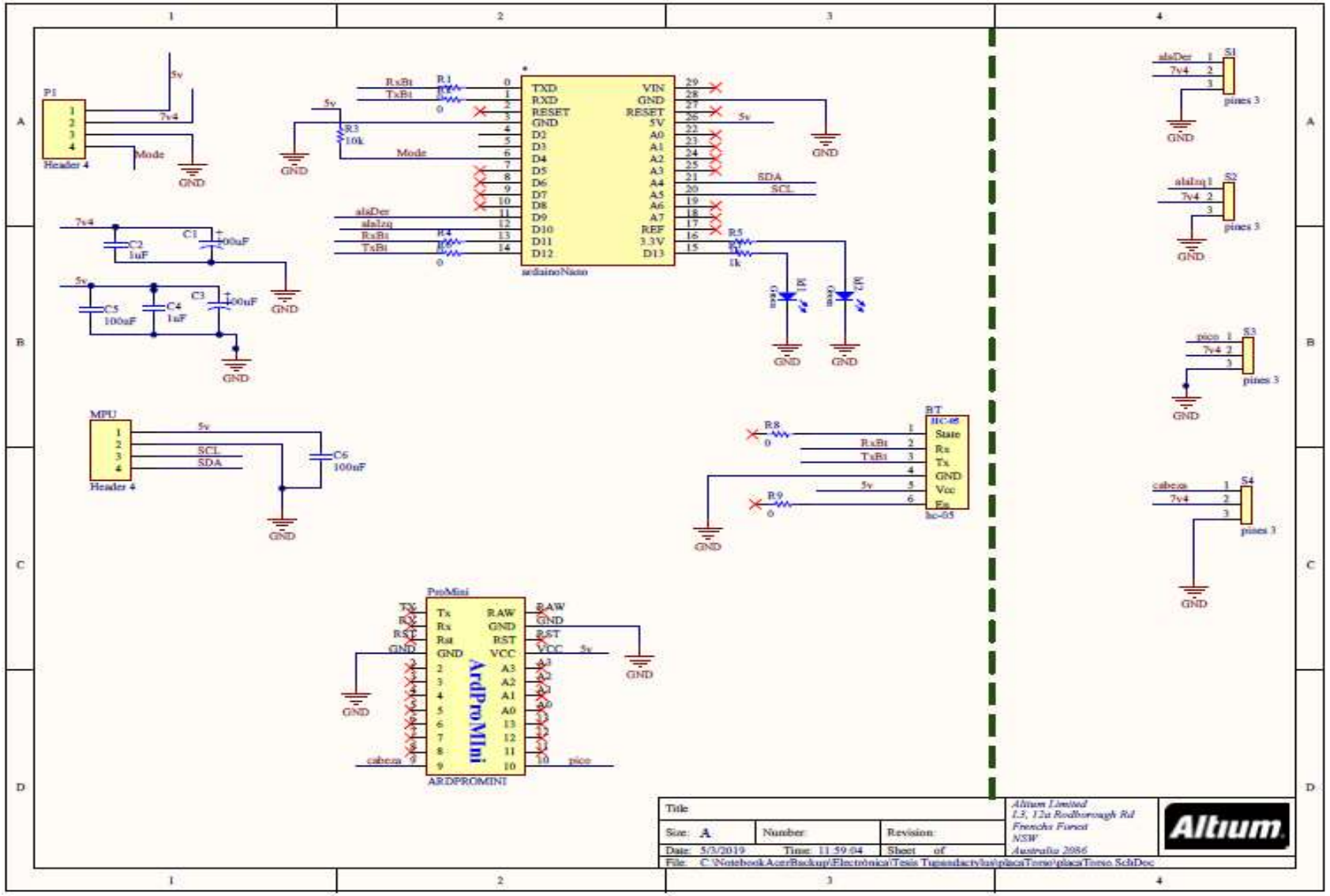
```
   AuxDer = AuxDer + 1 ; //llega a 110
   AuxIzq = AuxIzq - 1; //llega a 70
   alaIzq.write(AuxIzq);
   alaDer.write(AuxDer);
   delay(9);
   Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(10);
}

void alasMediaArriba()
{
  for (int i = 1 ; i < 40 ; i = i + 1)
  {
   AuxDer = AuxDer + 1 ;
   AuxIzq = AuxIzq - 1;
   alaIzq.write(AuxIzq);
   alaDer.write(AuxDer);
   delay(8);
   Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(50);

  for (int i = 40 ; i > 1 ; i = i - 1)
  {
   AuxDer = AuxDer - 1 ;
   AuxIzq = AuxIzq + 1;
   alaIzq.write(AuxIzq);
   alaDer.write(AuxDer);
   delay(8);
   Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(50);

}

void alasMediaAbajo()
{
  for (int i = 1 ; i < 40 ; i = i + 1)
  {
   AuxDer = AuxDer - 1 ;
   AuxIzq = AuxIzq + 1;
   alaIzq.write(AuxIzq);
   alaDer.write(AuxDer);
   delay(8);
   Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(50);
```

```
  for (int i = 40 ; i > 1 ; i = i - 1)
  {
    AuxDer = AuxDer + 1 ;
    AuxIzq = AuxIzq - 1;
    alaIzq.write(AuxIzq);
    alaDer.write(AuxDer);
    delay(8);
    Serial.print(AuxIzq); Serial.print("   "); Serial.println(AuxDer);
  }
  delay(50);
}


void loop() {
  if (digitalRead(4))
  {
    digitalWrite(13, HIGH);

    delay(90);
    alasArriba();
    delay(90);
    alasAbajo();
    delay(50);
    alasArriba();
    delay(90);
    alasMediaArriba();
    delay(100);
    alasAbajo();
    delay(50);
    alasArriba();
    delay(90);
    alasAbajo();
    delay(50);
    alasMediaAbajo();
    delay(90);
    alasMediaAbajo();
    delay(50);
    alasArriba();
    delay(90);
    alasMediaArriba();
    delay(90);
    alasMediaArriba();
    delay(90);
    alasAbajo();
    delay(50);
  }
```

```
 else
 {
   digitalWrite(13, LOW);
   //Leer los valores del Acelerometro de la IMU1
   Wire.beginTransmission(MPU);
   Wire.write(0x3B); //Pedir el registro 0x3B
   Wire.endTransmission(false);
   Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6 registros

   AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
   AcY = Wire.read() << 8 | Wire.read();
   AcZ = Wire.read() << 8 | Wire.read();

   //Leer los valores del Acelerometro de la IMU2
   Wire.beginTransmission(MPU2);
   Wire.write(0x3B); //Pedir el registro 0x3B
   Wire.endTransmission(false);
   Wire.requestFrom(MPU2, 6, true); //A partir del 0x3B, se piden 6 registros

   AcX2 = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
   AcY2 = Wire.read() << 8 | Wire.read();
   AcZ2 = Wire.read() << 8 | Wire.read();

   Acc[1] = atan(-1 * (AcX / A_R) / sqrt(pow((AcY / A_R), 2) + pow((AcZ / A_R), 2))) *
RAD_TO_DEG;
   Acc[0] = atan((AcY / A_R) / sqrt(pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

   Acc2[1] = atan(-1 * (AcX2 / A_R) / sqrt(pow((AcY2 / A_R), 2) + pow((AcZ2 / A_R), 2))) *
RAD_TO_DEG;
   Acc2[0] = atan((AcY2 / A_R) / sqrt(pow((AcX2 / A_R), 2) + pow((AcZ2 / A_R), 2))) *
RAD_TO_DEG;

   //Leer los valores del Giroscopio
   Wire.beginTransmission(MPU);
   Wire.write(0x43);
   Wire.endTransmission(false);
   Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerometro, solo se piden 4 registros
   GyX = Wire.read() << 8 | Wire.read();

   Wire.beginTransmission(MPU2);
   Wire.write(0x43);
   Wire.endTransmission(false);
   Wire.requestFrom(MPU2, 4, true); //A diferencia del Acelerometro, solo se piden 4 registros
   GyX2 = Wire.read() << 8 | Wire.read();

   //Calculo del angulo del Giroscopio
   Gy[0] = GyX / G_R;
   Gy2[0] = GyX2 / G_R;
```

```
  //Aplicar el Filtro Complementario
  Angle[0] = 0.93 * (Angle[0] + Gy[0] * (25 / 1000.0)) + 0.07 * Acc[0];
  Angle2[0] = 0.93 * (Angle2[0] + Gy2[0] * (25 / 1000.0)) + 0.07 * Acc2[0];

  //Mostrar los valores por consola
  Angulo = map(Angle[0], -85, 85, 0, 180);
  Angulo2 = map(Angle2[0], -85, 85, 0, 180);

  delay(1);
  alaDer.write(Angulo);
  delay(1);
  alaIzq.write(Angulo2);
  delay(8); //Nuestra dt sera, pues, 0.010, que es el intervalo de tiempo en cada medida*/
 }
}
```

5.1.2 Esquemático.



### 5.1.3 Listado de materiales.

| Comment | Description | Designator | Footprint | LibRef | Quantity |
|---|---|---|---|---|---|
| arduinoNano | | | Arduino nano | arduinoNano | 1 |
| hc-05 | | BT | hc-05 | hc-05 | 1 |
| 100uF | Polarised capacitor | C1, C3 | CAPAE830X1020N | | 2 |
| 1uF | | C2, C4 | CAP 0603 | CAP 1206 | 2 |
| 100nF | | C5, C6 | CAP 0603 | CAP 1206 | 2 |
| Green | LED Green 1C 2A | ld1, ld2 | LED_OSRAM_LGL29K | | 2 |
| Header 4 | Header, 4-Pin | MPU, P1 | b4b-xh | Header 4 | 2 |
| ARDPROMINI | | ProMini | ArdProMini | ARDPROMINI | 1 |
| 0 | | R1, R2, R4, R6, R8, R9 | RES 0603 | RES 1206 | 6 |
| 10k | | R3 | RES 0603 | RES 1206 | 1 |
| 1k | | R5, R7 | res 1206 | RES 1206 | 2 |
| pines 3 | | S1, S2, S3, S4 | 3pines | pines 3 | 4 |

***Tabla 5.1.-*** *Listado completo de materiales utilizados para el modelo final. Fuente: Propia*

## 5.2 Sobre pruebas realizadas

### 5.2.1 Configuración IMU MPU6050

En este programa se desarrolló la configuración del Giroscopio-Acelerómetro y se muestra el funcionamiento con gráficos y datos en el puerto serial.

Conexión

| **Arduino Mega 2560** | **IMU MPU6050** |
|---|---|
| 5 v | Vcc |
| GND | GND |
| Pin 20 | SDA |
| Pin 21 | SCL |

***Tabla 5.2.-*** *Pines de conexión del módulo IMU MPU6050 a la Arduino. Fuente: Propia*

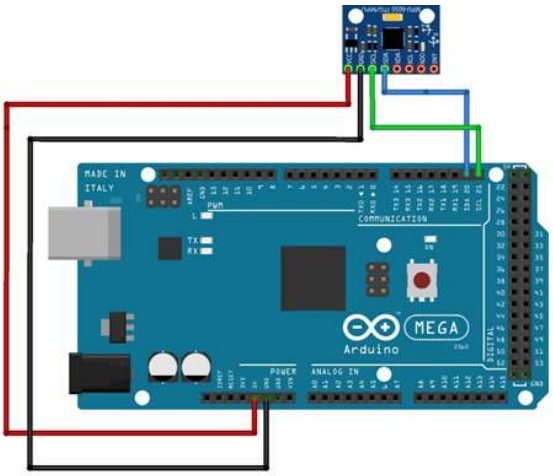



***Figura 5.1.-*** *Conexión del módulo IMU MPU6050 a la Arduino. Fuente: Propia*

Esquema electrónico

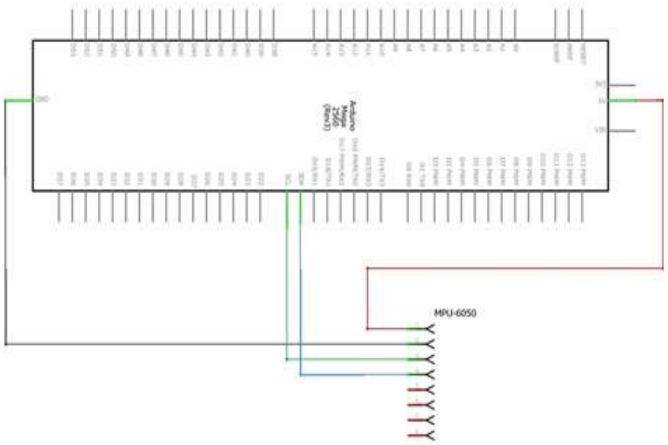



***Figura 5.2.-*** *Diagrama electrónico de la conexión del módulo IMU MPU6050 a la Arduino. Fuente: Propia*

El programa

```
#include <Wire.h>

//Dirección I2C de la IMU
#define MPU 0x68

//Datos de conversión
#define A_R 16384.0
#define G_R 131.0
```

```
//Conversión de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779

//MPU-6050 da los valores en enteros de 16 bits
//Valores sin refinar
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;

//Ángulos
float Acc [1];
float Gy [1];
float Angle [1];

void setup ()
{
Wire.begin();
Wire.beginTransmission(MPU);
Wire.write(0x6B);
Wire.write(0);
Wire.endTransmission(true);
Serial.begin(9600);
}

void loop ()
{
  //Leer los valores del Acelerómetro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,6, true); //A partir del 0x3B, se piden 6       registros
  AcX=Wire.read() <<8|Wire.read(); //Cada valor ocupa 2 registros
  AcY=Wire.read() <<8|Wire.read();
  AcZ=Wire.read() <<8|Wire.read();

   //A partir de los valores del acelerómetro, se calculan los ángulos Y, X
   //respectivamente, con la fórmula de la tangente.
  Acc [0] = atan((AcY/A_R) /sqrt(pow((AcX/A_R),2) + pow((AcZ/A_R),2))) *RAD_TO_DEG;


//Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
```

```
  Wire.requestFrom(MPU,4, true); //A diferencia del Acelerómetro, solo se piden 4 registros
  GyX=Wire.read() <<8|Wire.read();
  GyY=Wire.read() <<8|Wire.read();

  //Calculo del angulo del Giroscopio
  Gy [0] = GyX/G_R;
  Gy [1] = GyY/G_R;

  //Aplicar el Filtro Complementario
  Angle [0] = 0.98 *(Angle [0]+Gy [0]*0.010) + 0.02*Acc [0];

  //Mostrar los valores por consola
  Serial.print("Angle X: "); Serial.print(Angle [0]); Serial.print("\n");
Serial.print("\n----------\n");

  delay (10); //Nuestra dt será, pues, 0.010, que es el intervalo de tiempo en cada medida
}
```

El filtro tarda un tiempo a adaptarse a los cambios, lo cual es normal al filtrar las frecuencias.

Graficas

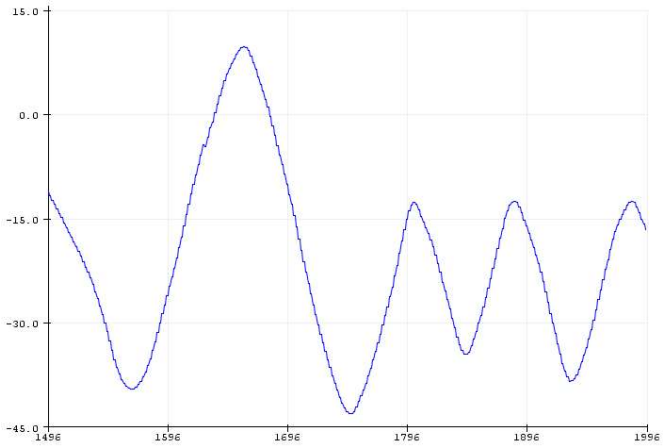



***Figura 5.3.-*** *Grafica de salida de datos del módulo IMU MPU6050. Fuente: Propia*

### 5.2.2 Conexión de dos IMU´s MPU6050 a la Arduino en un mismo bus de datos.

Programa para enviar datos de dos IMUs simultáneamente. Como las mismas están conectadas en un mismo bus de datos, se utiliza dos direcciones físicas distintas (dato que da el fabricante) y a su vez, es la cantidad máxima permitida que una placa de desarrollo puede leer.

Conexión

| Placa | IMU_1 MPU6050 | IMU_2 MPU6050 |
|---|---|---|
| 5 v | Vcc | Vcc |
| GND | GND | GND |
| Pin 20 | SDA | SDA |
| Pin 21 | SCL | SCL |

***Tabla 5.3.-*** *Pines de conexión de los dos módulos IMU MPU6050 a la Arduino en un mismo bus de datos a la Arduino. Fuente: Propia*

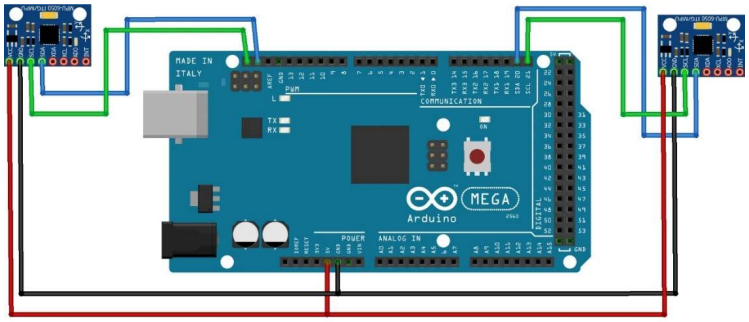



***Figura 5.4.-*** *Conexión dos IMU´s en un mismo bus de datos a la Arduino. Fuente: Propia*

Esquema Electrónico

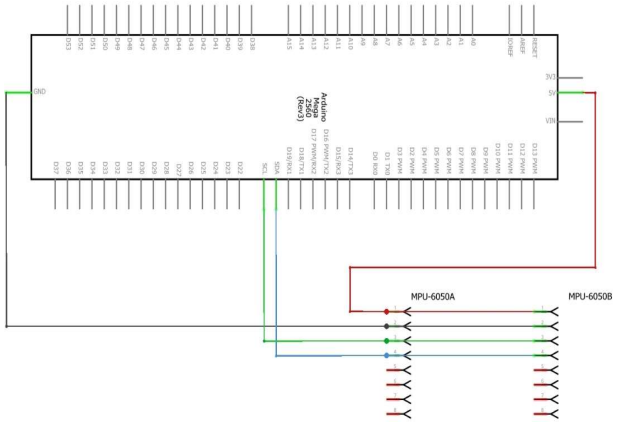



***Figura 5.5.-*** *Diagrama electrónico de dos IMU´s en un mismo bus de datos a la Arduino. Fuente: Propia*

Programa

```
#include<Wire.h>

  const int MPU_addr=0x69;  // I2C address of the MPU-6050
  const int MPU_addr2=0x68;  // I2C address of the MPU-6050
  int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
  int16_t AcX2,AcY2,AcZ2,Tmp2,GyX2,GyY2,GyZ2;
  void setup(){
   Wire.begin();
   Wire.beginTransmission(MPU_addr);
   Wire.write(0x6B);  // PWR_MGMT_1 register
   Wire.write(0);     // set to zero (wakes up the MPU-6050)
   Wire.endTransmission(true);
   Serial.begin(9600);
   Wire.begin();
   Wire.beginTransmission(MPU_addr2);
   Wire.write(0x6B);  // PWR_MGMT_1 register
   Wire.write(0);     // set to zero (wakes up the MPU-6050)
   Wire.endTransmission(true);

  }
  void loop(){
   Wire.beginTransmission(MPU_addr);
   Wire.write(0x3B);  // starting with register 0x3B (ACCEL_XOUT_H)
   Wire.endTransmission(false);
   Wire.requestFrom(MPU_addr,14,true);  // request a total of 14 registers
   AcX=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
   AcY=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
   AcZ=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
   Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
   GyX=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
   GyY=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
   GyZ=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
   Serial.print("AcX = "); Serial.print(AcX);
   Serial.print(" | AcY = "); Serial.print(AcY);
   Serial.print(" | AcZ = "); Serial.print(AcZ);
   Serial.print(" | Tmp = "); Serial.print(Tmp/340.00+36.53);  //equation for temperature in degrees
C from datasheet
   Serial.print(" | GyX = "); Serial.print(GyX);
   Serial.print(" | GyY = "); Serial.print(GyY);
   Serial.print(" | GyZ = "); Serial.println(GyZ);

   Wire.beginTransmission(MPU_addr2);
   Wire.write(0x3B);  // starting with register 0x3B (ACCEL_XOUT_H)
   Wire.endTransmission(false);
   Wire.requestFrom(MPU_addr2,14,true);  // request a total of 14 registers
   AcX2=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
```

```
    AcY2=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
    AcZ2=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
    Tmp2=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
    GyX2=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
    GyY2=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
    GyZ2=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
    Serial.print("AcX2 = "); Serial.print(AcX2);
    Serial.print(" | AcY2 = "); Serial.print(AcY2);
    Serial.print(" | AcZ2 = "); Serial.print(AcZ2);
    Serial.print(" | Tmp2 = "); Serial.print(Tmp2/340.00+36.53);   //equation for temperature in
degrees C from datasheet
    Serial.print(" | GyX2 = "); Serial.print(GyX2);
    Serial.print(" | GyY2 = "); Serial.print(GyY2);
    Serial.print(" | GyZ2 = "); Serial.println(GyZ2);
    Serial.println("_____________________________________________________________");

    delay(1500);
  }
```

### 5.2.3 Conexión de dos IMU´s MPU6050 a la Arduino utilizando un multiplexor I2C.

Programa para utilizar varios dispositivos que se comunican a través de I2C y los mismos se conectan simultáneamente con la tarjeta de desarrollo. Como el proyecto requirió de la utilización de cuatro IMUs (dada la imposibilidad de conectar más de dos, dado que el mismo puede ser asignado mediante dos direcciones físicas) entonces la solución es utilizar un multiplexor capaz de comunicar varias IMUs.

Conexión

| Arduino Mega 2560 | Multiplexor TCA 9548 |
|---|---|
| 5 v | Vcc |
| GND | GND |
| Pin 20 | SDA |
| Pin 21 | SCL |

***Tabla 5.4 a).-*** *Pines de conexión del módulo multiplexor i2c TCA9548 a la Arduino en un mismo bus de datos a la Arduino. Fuente: Propia*

| Multiplexor TCA 9548 | IMU_1 MPU6050 | IMU_2 MPU6050 |
|---|---|---|
| Vcc | Vcc | Vcc |
| GND | GND | GND |
| SDA | SD0 | SD1 |
| SCL | SC0 | SC1 |

***Tabla 5.4 b).-*** *Pines de conexión de los dos módulos IMU MPU6050 al multiplexor i2c TCA9548. Fuente: Propia*

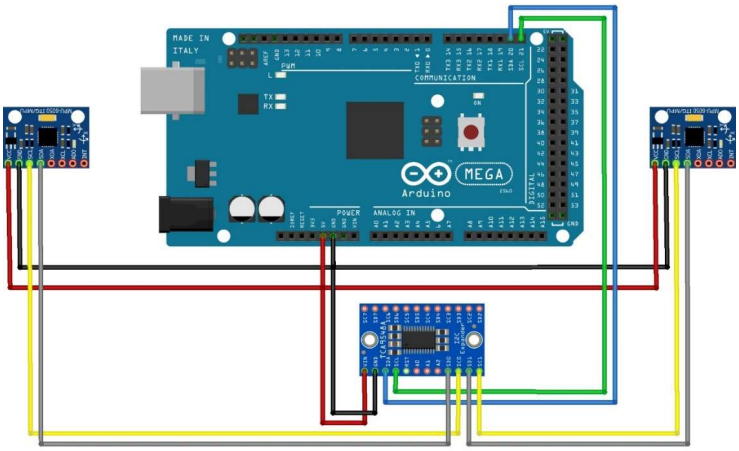



***Figura 5.6.-*** *Conexión de dos IMU´s a través de un multiplexor i2c a la Arduino. Fuente: Propia*

Esquema Electrónico

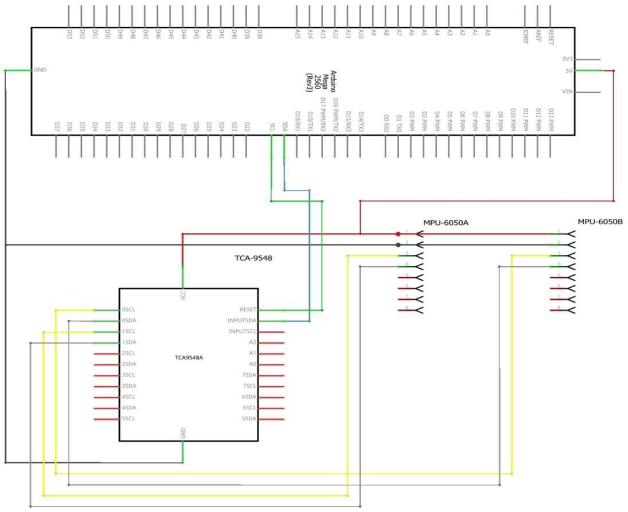



***Figura 5.7.-*** *Diagrama electrónico de conexión de dos IMU´s a través de un multiplexor de i2c a la Arduino. Fuente: Propia*

Programa

```
#include <TimerOne.h>
#include <Wire.h>

//Direccion del multiplexor de I2C
#define TCAADDR 0x70
//Direccion I2C de la IMU
#define MPU 0x68
//Ratios de conversion
#define A_R 16384.0
#define G_R 131.0
//Conversion de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779
//MPU-6050 da los valores en enteros de 16 bits
//Valores sin refinar
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;
//Angulos
float Acc_0[2], Acc_1[2];
float Gy_0[2], Gy_1[2];
float Angle_0[2], Angle_1[2];


//Flags de control
bool fLeer = false;

//Auxiliares
int selector = 0;
int muestra = 0;
int tin, tfin, t;

//Funcion que usa para el inicio del multiplexor I2C
void tcaselect(uint8_t i)
{
  if (i > 7) return;
  Wire.beginTransmission(TCAADDR);
  Wire.write(1 << i);
  Wire.endTransmission();
}

void setup()
```

```
{
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(115200);
  Timer1.initialize(5000);//Timer1.initialize(5000) 5mseg
  Timer1.attachInterrupt(intPorTimer); // Activa la interrupcion y la asocia a intPorTimer
}
void loop()
{
  if (fLeer) leer(); //Si el flag "leer" está en alto voy a la función leer
}

void intPorTimer(void)
{

  Serial.print("  "); Serial.print(Angle_1[0]); Serial.print("  "); Serial.print(Angle_0[0]); Serial.print("\n");
  fLeer = true;
}

void leer(void)
{
  muestra++;

  if (selector == 0)
  {
    tcaselect(0);
    selector = 1;
    IMU_0();
  }
  else
  {
    tcaselect(1);
    selector = 0;
    IMU_1();
  }

  fLeer = false;

}
```

```
void IMU_0(void)
{
  tin;
  //Leer los valores del Acelerometro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6 registros
  AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
  AcY = Wire.read() << 8 | Wire.read();
  AcZ = Wire.read() << 8 | Wire.read();

  //A partir de los valores del acelerometro, se calculan los angulos Y, X
  //respectivamente, con la formula de la tangente.
  Acc_0[0] = atan((AcY / A_R) / sqrt(pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

  //Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerometro, solo se piden 4 registros
  GyX = Wire.read() << 8 | Wire.read();

  //Calculo del angulo del Giroscopio
  Gy_0[0] = GyX / G_R;

  //Aplicar el Filtro Complementario
  tfin = micros();
  t = tfin - tin;
  Angle_0[0] = 0.98 * (Angle_0[0] + Gy_0[0] * t / 1000000) + 0.02 * Acc_0[0];

}

void IMU_1(void)
{

  //Leer los valores del Acelerometro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6 registros
```

```
AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
AcY = Wire.read() << 8 | Wire.read();
AcZ = Wire.read() << 8 | Wire.read();

//A partir de los valores del acelerometro, se calculan los angulos Y, X
//respectivamente, con la formula de la tangente.
Acc_1[0] = atan((AcY / A_R) / sqrt(pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

//Leer los valores del Giroscopio
Wire.beginTransmission(MPU);
Wire.write(0x43);
Wire.endTransmission(false);
Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerometro, solo se piden 4 registros
GyX = Wire.read() << 8 | Wire.read();

//Calculo del angulo del Giroscopio
Gy_1[0] = GyX / G_R;

//Aplicar el Filtro Complementario
Angle_1[0] = 0.98 * (Angle_1[0] + Gy_1[0] * t / 1000000) + 0.02 * Acc_1[0];
tin = micros();

}
```

Gráficos de la salida

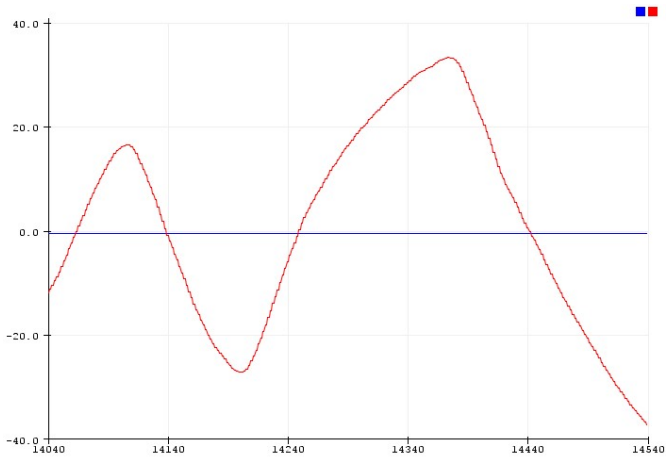



***Figura 5.8.-*** *Grafico de salida de datos de una IMU y la otra en reposo. Fuente: Propia*

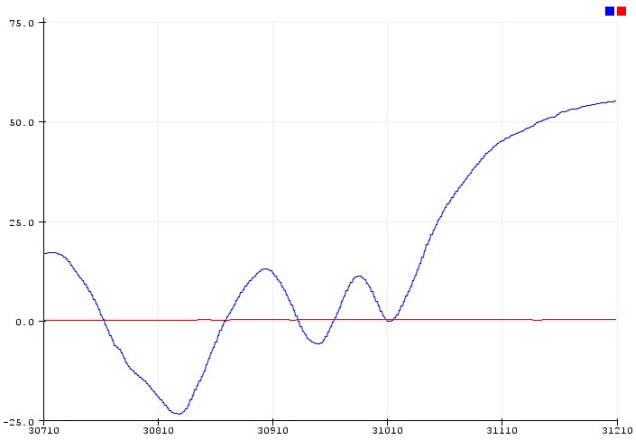



***Figura 5.9.-*** *Grafico de salida de datos de la otra IMU y la otra en reposo. Fuente: Propia*

### 5.2.4 Servomotor que copia movimiento de una IMU.

Programa el cual el servomotor copia los movimientos generados por la IMU.
Conexión

| **Arduino Mega 2560** | **IMU MPU6050** | **Servomotor** |
|---|---|---|
| 5 v | Vcc | Vcc (Rojo) |
| GND | GND | GND (Negro) |
| Pin 20 | SDA | NC |
| Pin 21 | SCL | NC |
| Pin 9 | NC | Pulse (Amarillo) |

***Tabla 5.5.-*** *Conexión de dos IMU´s a través de un multiplexor i2c a la Arduino. Fuente: Propia*

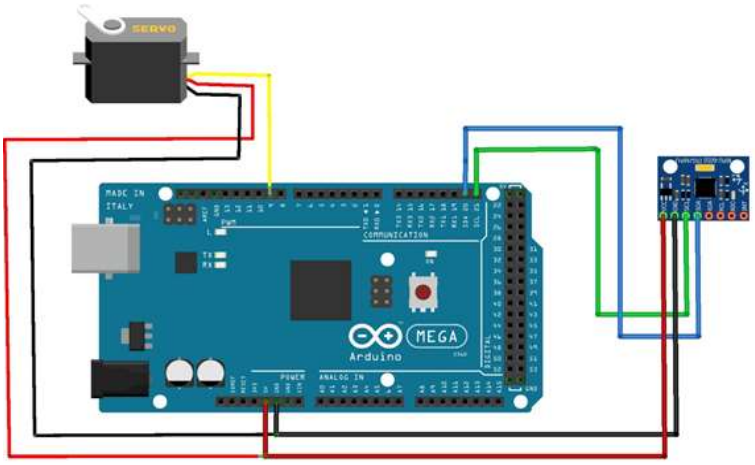



***Figura 5.10.-*** *Conexión de dos IMU´s a través de un multiplexor i2c a la Arduino. Fuente: Propia*

Esquema Electrónico

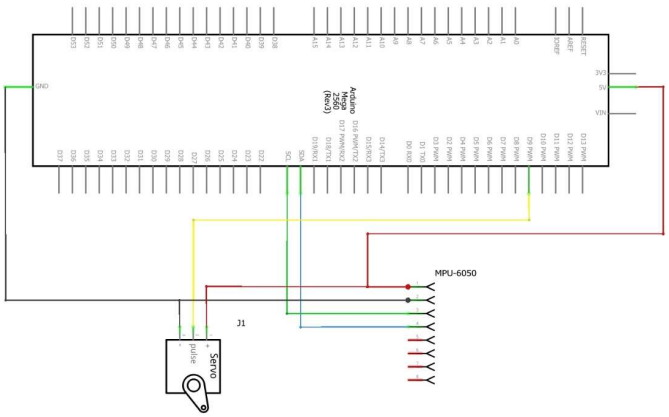



***Figura 5.11.-*** *Conexión de dos IMU´s a través de un multiplexor i2c a la Arduino. Fuente: Propia*

Programa

```
#include <Wire.h>

//Dirección I2C de la IMU
#define MPU 0x68

//Datos de conversión
#define A_R 16384.0
#define G_R 131.0

//Conversión de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779
```

```
//MPU-6050 da los valores en enteros de 16 bits
//Valores sin refinar
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;

//Ángulos
float Acc [1];
float Gy [1];
float Angle [1];

//Auxiliares Servo
Servo myservo;
int Angle_anterior;

void setup ()
{
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600);
  myservo.attach(9);
}

void loop ()
{
  //Leer los valores del Acelerómetro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6       registros
  AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
  AcY = Wire.read() << 8 | Wire.read();
  AcZ = Wire.read() << 8 | Wire.read();

  //A partir de los valores del acelerómetro, se calculan los ángulos Y, X
  //respectivamente, con la fórmula de la tangente.
  Acc [0] = atan ((AcY / A_R) / sqrt (pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

  //Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
```

```
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerómetro, solo se piden 4 registros
  GyX = Wire.read() << 8 | Wire.read();
  GyY = Wire.read() << 8 | Wire.read();

  //Calculo del angulo del Giroscopio
  Gy [0] = GyX / G_R;

  //Aplicar el Filtro Complementario
  Angle [0] = 0.98 * (Angle [0] + Gy [0] * 0.010) + 0.02 * Acc [0];

  //Mostrar los valores por consola
  Serial.print("Angle X: "); Serial.print(Angle [0]); Serial.print("\n");

  delay (10); //Nuestra dt será, pues, 0.010, que es el intervalo de tiempo en cada medida

    if (Angle [0] != Angle_anterior)
    {
       myservo. write(val);
       Angle_anterior = Angle [0];
    }
           delay (10);
}
```

### 5.2.5 Comunicación Inalámbrica

#### 5.2.5.1 Configuración modulo bluetooth HC-05

El siguiente código, sirve para activar el *modo configuración* del módulo bluetooth HC-05 y luego configurarlos mediante los comandos AT a través de su puerto serie.

```
#include <SoftwareSerial.h>   //incluimos la librería

SoftwareSerial BT(10,11); //creamos un objeto y todas sus funciones asociadas para establecer conexión entre el módulo y Arduino. Pines (RX, TX).
void setup()
{
 Serial.begin(9600);   //inicializamos la comunicación serie a 9600 entre el Arduino y la PC.
 Serial.println("Conectado...");    //imprime en el monitor serie que se conectó.
 BT.begin(38400);    //inicializamos la comunicación serie de 34800 entre el módulo BT y Arduino.
}
void loop()  //acá se van a recibir las respuestas entre el módulo bluethoot y el Arduino.
{
 if(BT.available())    //lee el módulo bluethoot y envía dato a Arduino
 Serial.write(BT.read());   //muestra en el monitor serie

 if(Serial.available())
 BT.write(Serial.read());     //lee Arduino y envía dato al módulo bluethoot
}
```

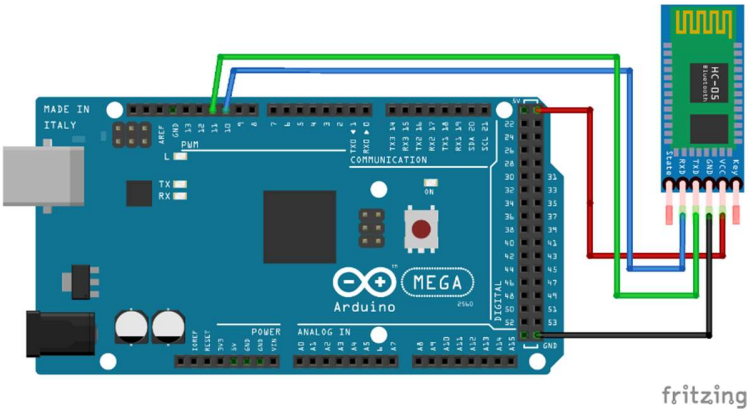



***Figura 5.12.-*** *Conexión del módulo HC05 para configuración. Fuente: Propia*

#### 5.2.5.2 Configuración Maestro-Esclavo por comandos AT

Para entrar al modo de configuración AT, antes de alimentar o encender el módulo es necesario presionar su botón, mantener presionado y alimentar el módulo, después que

enciende recién podemos soltar el botón. Si el LED Parpadea lentamente es porque ya está en Modo configuración AT.

| Comando | Descripción | Respuesta si hay comunicación |
|---|---|---|
| AT | Comando de prueba | OK |
| AT+ROLE? | Comando para verificar el rol de el modulo | +ROLE:0(0 para esclavo y 1 para maestro) |
| AT+ROLE=0 | Comando para poner el modulo en modo Esclavo | OK |
| AT+ROLE=1 | Comando para poner el modulo en modo Maestro | OK |
| AT+VERSION? | Comando para obtener la version de el Firmware | +VERSION:hc01.comV2.1 |
| AT+BAUD="numero" | Comando para configurar el baurate al que trabajará el bluetooth | OK |
| AT+NAME? | Comando para saber el nombre que lleva el módulo HC-05 | +NAME:HC-05 |
| AT+NAME="nombre" | Comando para cambiar el nombre que llevara el módulo HC-05 | OK |
| AT+PSWD? | Comando para saber la contraseña actual del módulo bluetooth | +PSWD:1234 |
| AT+PSWD="numero de 4 digitos" | Comando para cambiar la contraseña del módulo bluetooth | OK |

***Tabla 5.6.-*** *Tabla de comandos AT con descripción y respuesta. Fuente: Propia*

Ahora se explicarán algunos comandos que serán los más utilizados para nuestra configuración (tanto esclavo como maestro) siendo los siguientes:

- Para testear la comunicación

Comprobamos si nuestro bluetooth responde a los comandos AT

**Enviar:** AT
**Recibe:** OK

Si recibimos como respuesta un OK entonces podemos continuar, sino verificar las conexiones o los pasos anteriores.

- Cambio de nombre del modulo

Se puede cambiar el nombre con el siguiente comando AT

**Enviar:** AT+NAME=<Nombre>

- Agregar una contraseña para vinculación

Por defecto viene con una contraseña de “1234”, para cambiarlo hay que enviar el siguiente comando AT

**Enviar**: AT+PSWD=<Pin>

Se puede saber cuál es la contraseña actual de nuestro modulo, enviando el siguiente comando: **AT+ PSWD?**

- Cambiar la velocidad de comunicación

La velocidad por defecto es de 9600 baudios, con Stop bit =0 (1 bit de parada), y sin Paridad, para cambiar estos parámetros, se hace uso del siguiente comando AT:

**Enviar**: AT+UART=<Baud> ,< StopBit>,< Parity>

Donde :

< Baud > es la velocidad, los valores pueden ser: 4800, 9600, 19200, 38400, 57600, 115200, 23400, 460800, 921600 o 1382400.

< StopBit> es el Bit de parada, puede ser 0 o 1, para 1 bit o 2 bits de parada respectivamente. Para aplicaciones comunes se trabaja con 1 bit por lo que este parámetro normalmente se lo deja en 0.

< Parity> Es la paridad, puede ser 0 (Sin Paridad), 1 (Paridad impar) o 2 (Paridad par). Para aplicaciones comunes no se usa paridad, por lo que dejamos este parámetro en 0.

Se puede saber cuál es la configuración actual, para eso hay que enviar el siguiente comando: **AT+UART?**

Configuración para esclavo

**Enviar:** AT
**Recibe:** OK
- Establecer el Role como Esclavo

**Enviar:** AT+ROLE=0
**Respuesta:** OK

- Configurar el Nombre del modulo

**Enviar:** AT+NAME=ESCLAVO
**Respuesta:** OK
- Establecer el Pin de vinculación

**Enviar**: AT+PSWD=1010
**Respuesta**: OK
- Configura la Velocidad

**Enviar**: AT+UART=38400,0,0
**Respuesta**: OK
- Verificar los parámetros cambiados

**Enviar:**
AT+ROLE?
AT+PSWD?
AT+UART?
**Respuesta:**
+ROLE:0
OK
+PSWD:1010
OK
+UART:38400,0,0
OK
- Reiniciar el modulo
**Enviar**: AT+RESET
**Respuesta**: OK

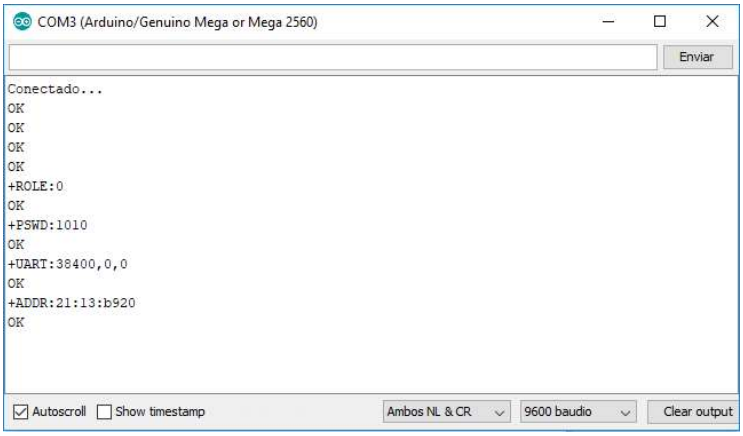



***Figura 5.13.-*** *Configuración del módulo bluetooth HC-05 como Esclavo mediante comandos AT. Fuente: Propia.*

Configuración para maestro

**Enviar:** AT+ROLE=1
**Respuesta:** OK
- Configurar el Nombre del modulo

**Enviar:** AT+NAME=MAESTRO
**Respuesta:** OK
- Establecer el Pin de vinculación

**Enviar**: AT+PSWD=1010
**Respuesta**: OK
- Configura la Velocidad

**Enviar**: AT+UART=38400,0,0
**Respuesta**: OK
- Configurar el modo de conexión

**Enviar:** AT+CMODE=0
**Respuesta:** OK
- Especificar la dirección del dispositivo a conectarse
**Enviar**: AT+BIND=21,13,b920
**Respuesta**: OK
- Verificar los parámetros cambiados
**Enviar:**
AT+ROLE?
AT+PSWD?
AT+UART?
AT+CMODE?
AT+BIND?
**Respuesta:**
+ROLE:1
OK
+PSWD:1010
OK
+UART:38400,0,0
OK
+CMOD:0
OK
+BIND: 21:13:b920

OK
- Reiniciar el modulo

**Enviar**: AT+RESET

**Respuesta**: OK

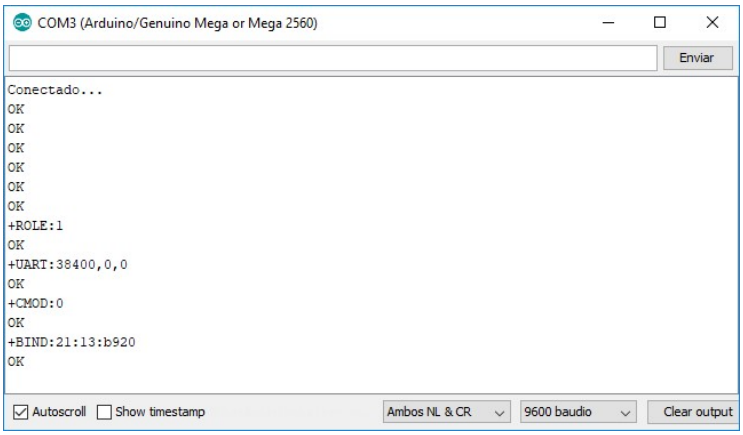



***Figura 5.14.-*** *Configuración del módulo bluetooth HC-05 como Maestro mediante comandos AT. Fuente: Propia.*

### 5.2.6 Transferencia de Señal Analógica de un potenciómetro entre dos dispositivos Arduinos a través de Bluethoot.

Programa que envía datos analógicos provenientes de un potenciómetro. La placa desarrolladora lo procesa, mapea y luego lo grafica al puerto serie. La comunicación es vía inalámbrica.

Conexiones

| Arduino Mega 2560 | Potenciómetro | Modulo HC-05 |
|---|---|---|
| 5 v | Vcc | Vcc |
| GND | GND | GND |
| Pin A0 | Referencia | - |
| Pin D18 | - | RXD |
| Pin D19 | - | TXD |

***Tabla 5.7.-*** *Pines de conexión del módulo bluetooth HC05 y potenciómetro a la Arduino del maestro. Fuente: Propia*

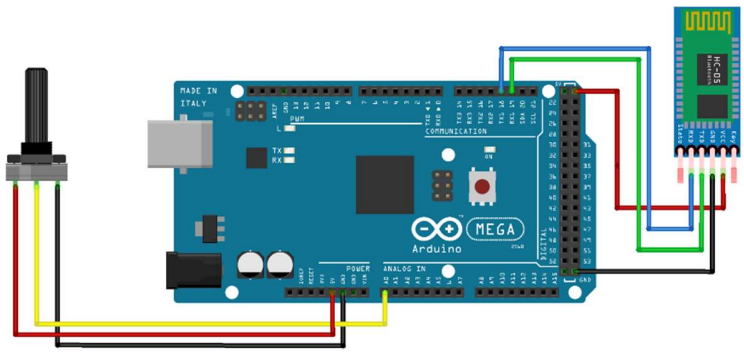


***Figura 5.15.-*** *Conexión del módulo HC05 y potenciómetro al maestro. Fuente: Propia*

| Arduino Mega 2560 | Modulo HC-05 |
|---|---|
| 5 v | Vcc |
| GND | GND |
| Pin D18 | RXD |
| Pin D19 | TXD |

***Tabla 5.8.-*** *Pines de conexión del módulo bluetooth HC05 a la Arduino del esclavo. Fuente: Propia*

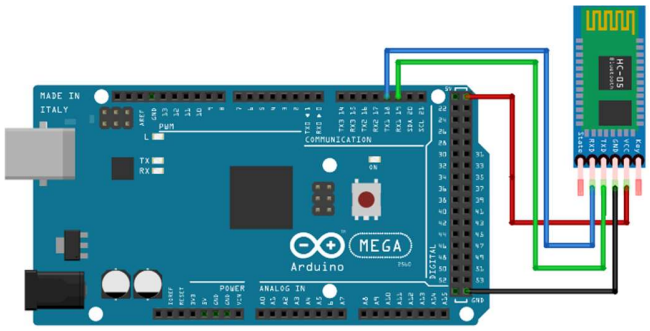


***Figura 5.16.-*** *Conexión del módulo HC05 del esclavo. Fuente: Propia*

Esquema Electrónico

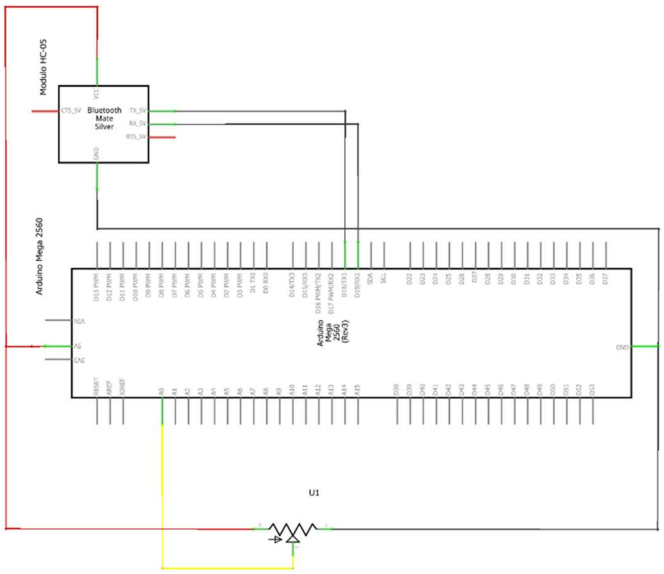



***Figura 5.17.-*** *Esquemático conexión del módulo bluetooth HC05 y potenciómetro del maestro. Fuente: Propia*

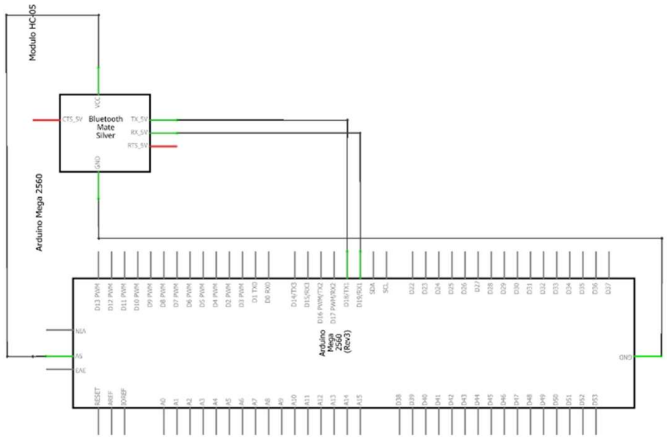



***Figura 5.18.-*** *Esquemático conexión del módulo bluetooth HC05 del esclavo. Fuente: Propia*

Programa

Configuración para modulo bluethoot maestro

```
int potenciometro; //Variable para guardar el valor analogico
int posicion = 0; //variable de posicion en grados

void setup() {
```

```
  Serial.begin(9600); //Iniciacion del puerto serial para la comunicación pc-arduino
  Serial1.begin(38400); //Iniciacion del puerto serial para el Bluetooth
}

void loop() {
  //Lectura del pin analogico y envio de datos
  Serial1.print("*"); //Se envia un primer dato para activar el receptor
  potenciometro = analogRead(0); // Se lee el valor del potenciometro con el ADC del arduino
  posicion = map(potenciometro, 0, 1023, 15, 180); //ajuste de valores maximos y minimos de
angulo
  Serial1.print(String(posicion)); // Se envia el valor deposicion
  Serial.println(posicion); //Imprime en el monitor serie
  delay(100); //Retardo para evitar conflictos
```

Configuración para modulo bluethoot esclavo

```
int posicion = 0; // Variable para la señal analogica
char BluetoothData; //Variable para la informacion que transportara

void setup() {
  Serial.begin(9600); //Iniciacion del puerto serial para la pc-arduino
  Serial1.begin(38400); //Iniciacion del puerto serial para el Bluetooth
  delay(400);
 }

void loop() {
  //Espera de dato entrante del Bluetooth
    while(Serial1.available()>0){
    BluetoothData=Serial1.read();
    posicion=Serial1.parseInt(); //Se recupera la siguente cantidad entera, despues del *
   Serial.println(posicion); //Se escribe la señal analogica recibida
      }
}
```

Resultados

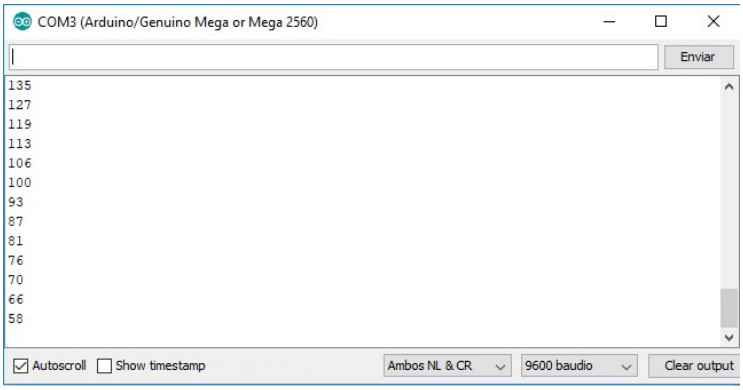



***Figura 5.19.-*** *Comunicación desde el Módulo Maestro. Fuente: Propia.*

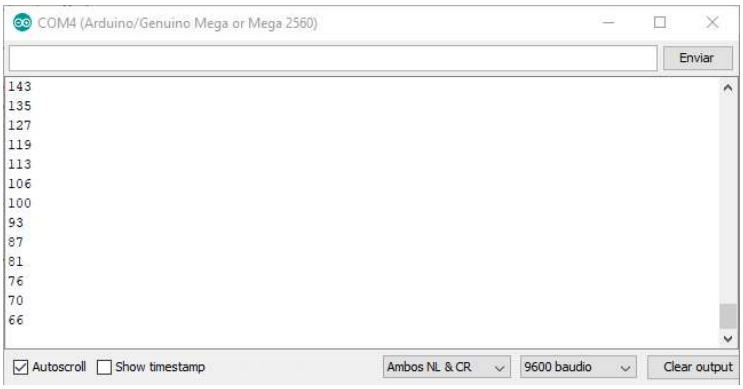



***Figura 5.20.-*** *Recepción desde el Módulo Esclavo. Fuente: Propia.*

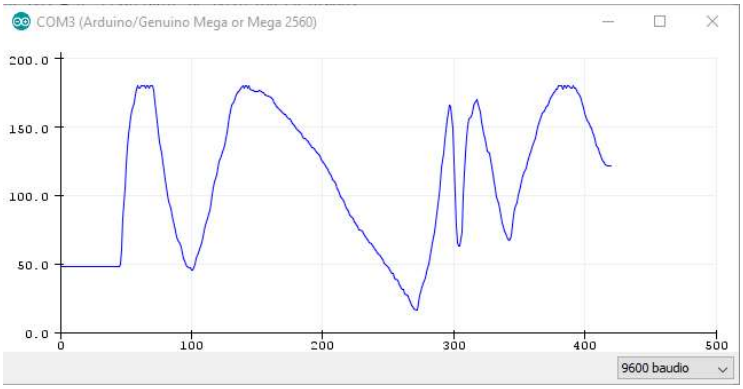



***Figura 5.21.-*** *Comunicación desde el Módulo Maestro. Fuente: Propia*

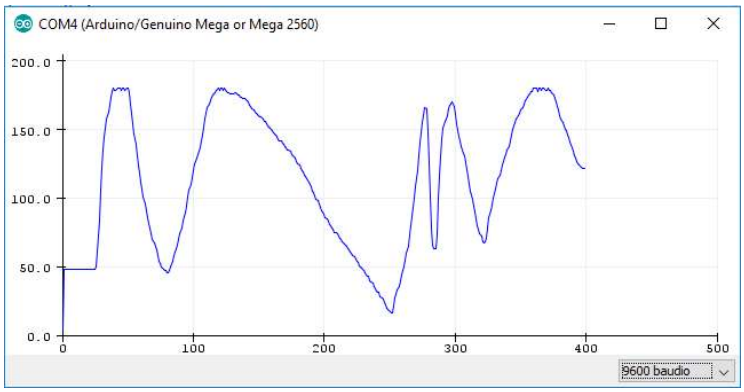



***Figura 5.22.-*** *Recepción desde el Módulo Esclavo. Fuente: Propia.*

### 5.2.7 Movimiento de un servomotor con una IMU a través de Bluethoot.

Programa que tiene por finalidad, copiar el movimiento de una IMU proveniente del módulo maestro y luego mover el servomotor según el grado de giro de la IMU. Esta comunicación es vía inalámbrica.

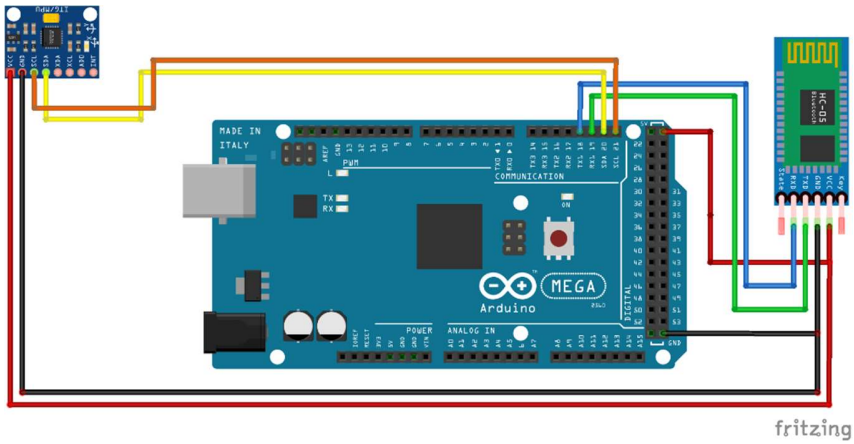



***Figura 5.23.-*** *Conexión del módulo bluetooth HC05 y la IMU MPU6050 del maestro. Fuente: Propia*

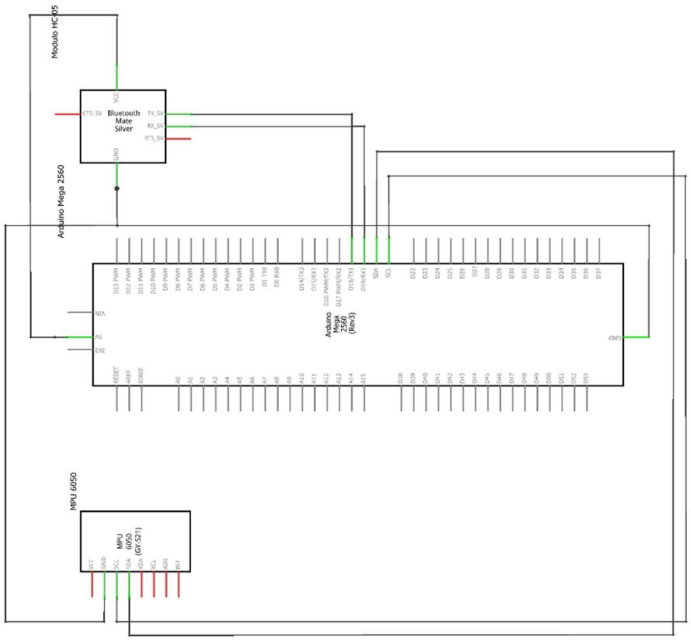



***Figura 5.24.-*** *Esquemático conexión del módulo bluetooth HC05 y la IMU MPU6050 del maestro. Fuente: Propia*

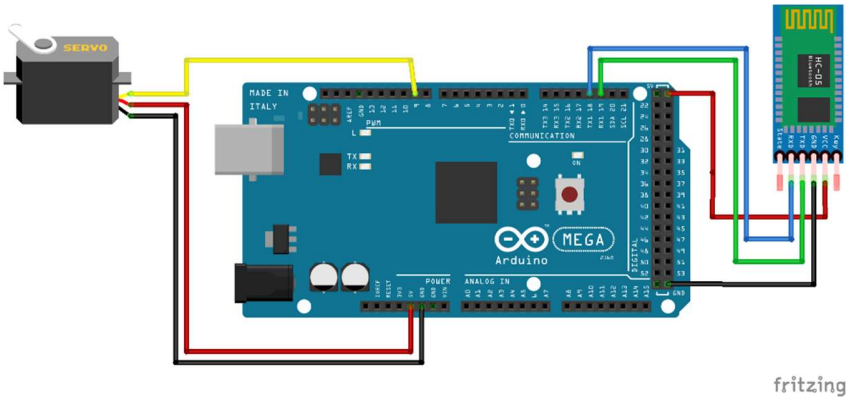



***Figura 5.25.-*** *Conexión del módulo bluetooth HC05 y el servomotor del esclavo. Fuente: Propia*

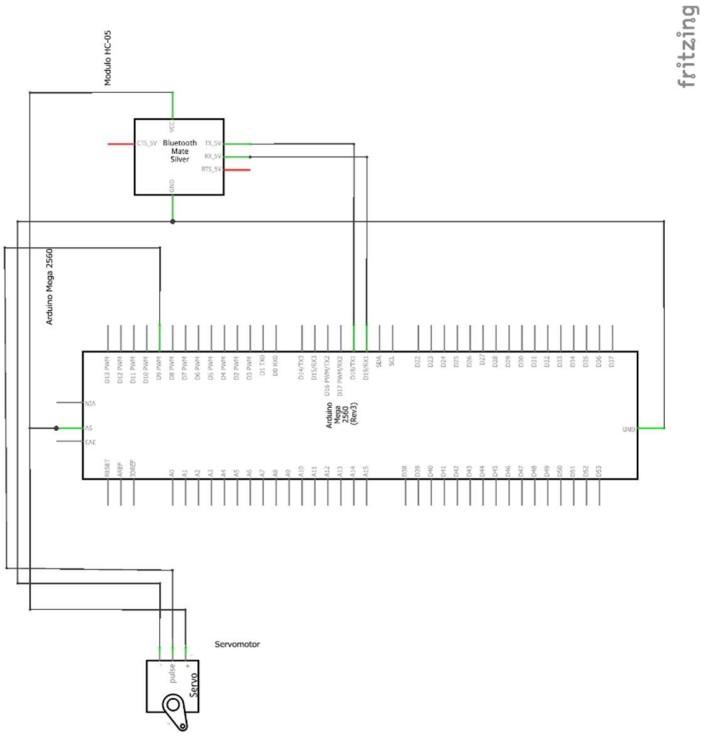



***Figura 5.26.-*** *Esquemático conexión del módulo bluetooth HC05 y el servomotor del esclavo.*
*Fuente: Propia*

## Programa

### Configuración para el maestro

```
#include <Wire.h>

//Dirección I2C de la IMU
#define MPU 0x68

//Datos de conversión
#define A_R 16384.0
#define G_R 131.0

//Conversión de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779

//MPU-6050 da los valores en enteros de 16 bits
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ; //Variables para los distintos usos

//Ángulos
float Acc [1]; //Array que guarda el angulo en X del acelerometro
```

```
float Gy [1]; //Array que guarda el angulo en X del giroscopio
float Angle [1]; //Array que guarda el angulo en X resultante del filtro
float Angle_anterior; //Variable que se utiliza para guardar el angulo antes de obtener el nuevo angulo
float Angulo_map; //Variable que se utiliza para mapear


int potenciometro; //Variable para guardar el valor analogico
int posicion = 0; //variable de posicion en grados

void setup()
{
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600); //Iniciacion del puerto serial
  Serial1.begin(38400); //Iniciacion del puerto serial para el Bluetooth
}

void loop()
{
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6 registros
  AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
  AcY = Wire.read() << 8 | Wire.read();
  AcZ = Wire.read() << 8 | Wire.read();

  Angle_anterior = Acc [0];
  Acc [0] = atan((AcY / A_R) / sqrt(pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

  //Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerómetro, solo se piden 4 registros

  GyX = Wire.read() << 8 | Wire.read();
  Gy [0] = GyX / G_R;
```

```
  Angle [0] = 0.9 * (Angle [0] + Gy [0] * 0.010) + 0.1 * Acc [0]; //Aplicar el Filtro Complementario

  //Mostrar los valores por consola
  /*Serial.print("Angle X: "); Serial.println(Acc [0]); /*Serial.print("\n");*/
  Serial.print("Angle X: "); Serial.println(Angle [0]);/* Serial.print("\n");*/

  delay (20); //Nuestra dt será, pues, 0.02, que es el intervalo de tiempo en cada medida

  Angulo_map = map (Angle [0], -120, 120, 0, 180); //Mapea los angulos que van de -90° a 90° y los
convierte de 0° a 180°
  Serial1.print(String(Angulo_map));

  delay(20);  //Retardo para evitar conflictos
}
```

## Configuración para el esclavo

```
//Se agraga la libreria para el uso de servos
#include <Servo.h>

Servo myservo; // Se crea el objeto para el control del servo

float posicion = 0; // Variable para el control de la posicion del servo
char BluetoothData; //Variable para la recepcion de datos del puerto bluethoot

void setup()
{

  myservo.attach(9);  // Se  define un pin para el objeto
  Serial.begin(115200); //Iniciacion del puerto serial
  Serial1.begin(38400); //Iniciacion del puerto serial para el Bluetooth
  delay(400); //Espera de 0.4 segundos

}

void loop()
{
  while (Serial1.available() > 0)  //Espera de dato entrante del Bluetooth
  {
    BluetoothData = Serial1.read(); //Lee el dato y lo guarda en la variable
    posicion = Serial1.parseInt(); //Se recupera la siguente cantidad entera
```

```
   Serial.print("Angle X: "); Serial.print(posicion); Serial.print("\n"); //Grafica los grados y envia por el
monitor serie
   myservo.write(posicion); //Mueve el servo segun
  }
}
```

# Anexo II

El presente anexo se compone de la tecnología que quedó descartada con respecto a los actuadores de las alas. Se pretendió hacer un control de los motorreductores que anteriormente poseía el animatrónico, su estudio y desarrollo de control. A pesar del tiempo y el esfuerzo invertidos en esta parte del proyecto, se tuvo que descartar esta opción por no llegar al modelo matemático necesario, cambiando la tecnología del actuador por un servomotor desarrollado en el marco teórico.

## 6.1 Principio de funcionamiento de un motor de cd

La acción de la fuerza ejercida en un conductor que porta corriente eléctrica dentro de un campo magnético externo puede deducirse mediante la regla de la mano izquierda de la acción motor (ver *Figura 6.1*), esta regla establece lo siguiente: si se extienden los dedos pulgar, índice y medio de la mano izquierda, formando ángulos rectos entre sí y el dedo índice apunta en la dirección del flujo magnético F de los polos de norte a sur y el dedo medio apunta en la dirección del flujo de corriente convencional *i* en el conductor, entonces el dedo pulgar apuntará en la dirección de la fuerza que se ejerce sobre el conductor.

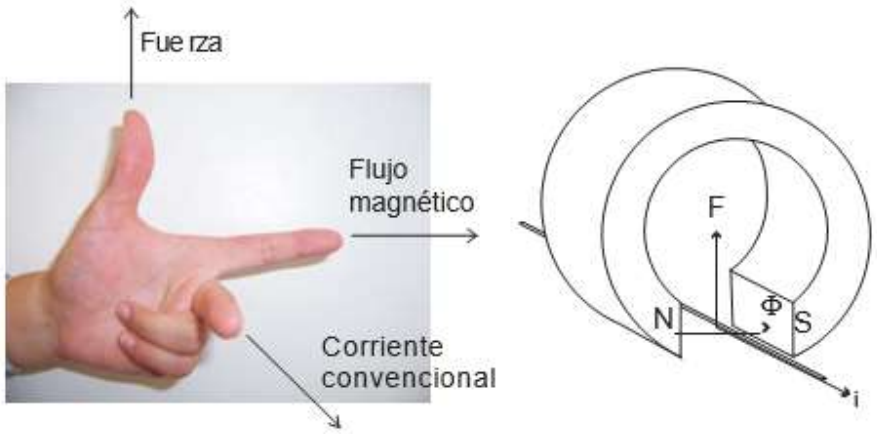



***Figura 6.1.-*** *Regla de la mano izquierda de la acción motor. Fuente: Tesis de grado [1].*

La regla anterior también tiene su equivalente siguiendo el flujo de corriente de electrones, a la cual se le conoce como la regla motor de la mano derecha, formando ángulos rectos entre sí y el dedo índice apunta en la dirección del flujo magnético Φ de los polos de norte a sur y el dedo medio apunta en la dirección del flujo de corriente de electrones $i_e$ en el conductor, entonces, el dedo pulgar apuntara en la dirección de la fuerza que se ejerce sobre el conductor (ver *Figura 6.2*). Cualquiera de las dos reglas es

válido siempre y cuando aplique teniendo en cuenta la dirección del flujo de corriente eléctrica o de electrones.

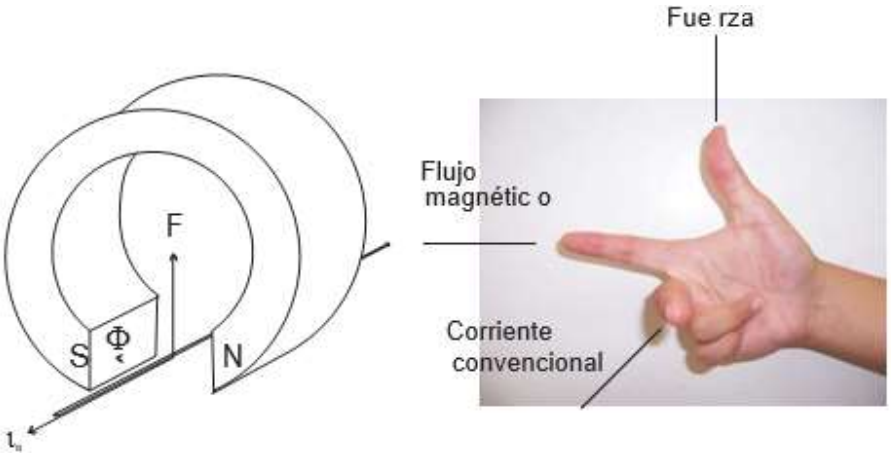



***Figura 6.2.-*** *Regla de la mano derecha. Fuente: Tesis de grado [1].*

Si se toma un conductor y se dobla en forma de espira, como se muestra en la *Figura 6.3*, con un eje imaginario sobre el cual la espira es libre de rotar, se coloca dentro de un campo magnético externo, y se hace pasar una corriente eléctrica a través de la espira en dirección del flujo, el par de fuerzas magnéticas paralelas, de igual magnitud y de dirección opuesta, crearán un momento de torsión haciendo girar la espira en dirección de las manecillas del reloj. Estas fuerzas y el momento de torsión son el principio de funcionamiento de un motor de corriente directa.

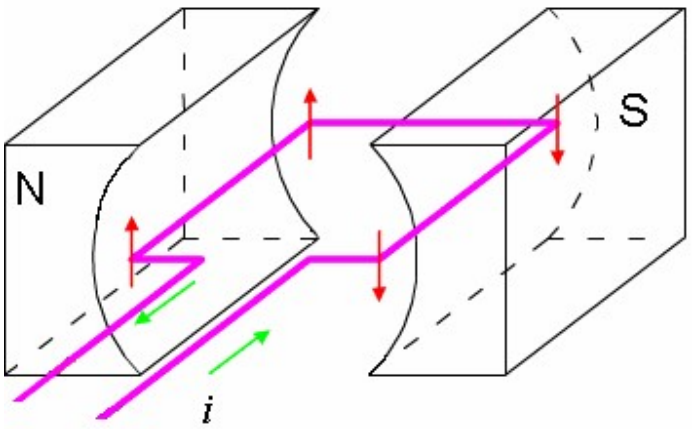



***Figura 6.3.-*** *Espira dentro de un campo magnético. Fuente: Tesis de grado [1].*

En la *Figura 6.4* se muestra el motor básico de corriente continua, el conmutador está formado por un anillo conductor dividido en dos segmentos (delgas), cada segmento está conectado a cada extremo de la espira conductora y aislados eléctricamente uno del otro. A medida que la espira gira cada escobilla toca alternativamente ambos segmentos

del conmutador. En consecuencia, las conexiones eléctricas se invierten a la mitad de cada revolución en el instante en que la espira es perpendicular al campo magnético. De esta manera, el momento de torsión que actúa sobre la espira lo hace siempre en la misma dirección y esta gira continuamente en el mismo sentido. Generalmente la armadura es el elemento del motor de cd que se encuentra girando y los imanes son estacionarios; la parte del motor que gira se denomina rotor y la parte estacionaria se denomina estator.

Existe otro fenómeno que se observa al mover un conductor dentro de un campo magnético. Faraday demostró que el desplazamiento mecánico de un conductor dentro de un campo magnético produce una corriente eléctrica. Faraday lo denominó voltaje inducido ya que se genera sin que exista contacto entre el imán y el conductor, el voltaje inducido también es conocido como fuerza electromotriz inducida. La ley de Faraday se anuncia como sigue:

“el voltaje inducido en un conductor es proporcional a la rapidez de cambio de las líneas de fuerza magnética que atraviesan el conductor”.

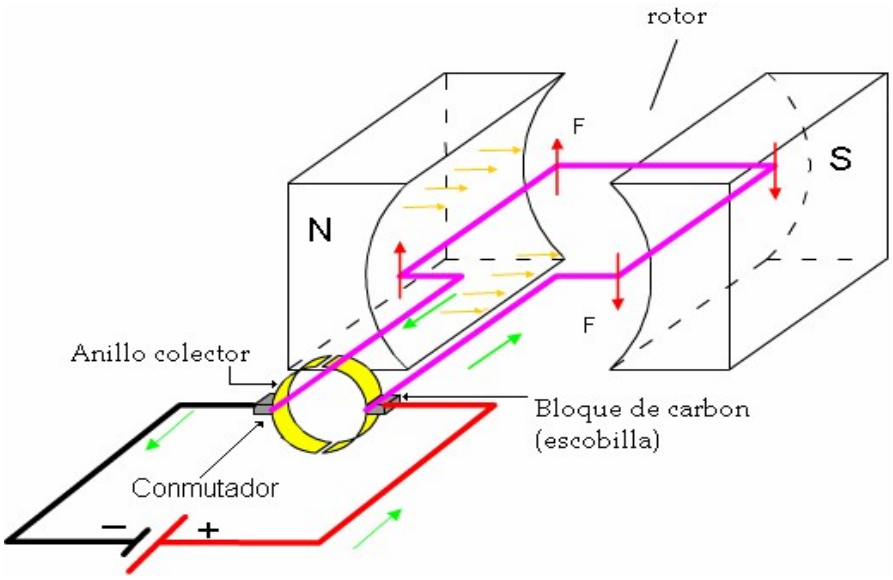



***Figura 6.4.-*** *Motor básico de corriente directa. Fuente: Tesis de grado [1].*

En la *Figura 6.5 (a)* se muestra como el movimiento de un conductor dentro de un campo magnético induce un voltaje. John Ambrose Fleming creo una regla que relaciona la ley de Faraday de la forma siguiente: si se extienden los dedos pulgar, índice y medio de la mano derecha, formando ángulos rectos entre sí y el dedo índice apunta en la dirección del flujo magnético Φ de los polos de norte a sur y el dedo pulgar apunta en la dirección de movimiento del conductor, entonces, el dedo medio apuntará hacia la terminal positiva del voltaje inducido, que también es la dirección del flujo de corriente convencional *i* inducida en el conductor, ver *figura 6.5 (b).*

De lo anterior, se hace evidente que un motor de cd no solamente puede operar como motor, sino que también puede operar como generador de cd, a los modos de operación anteriores se les conoce como modo motor y modo generador respectivamente.

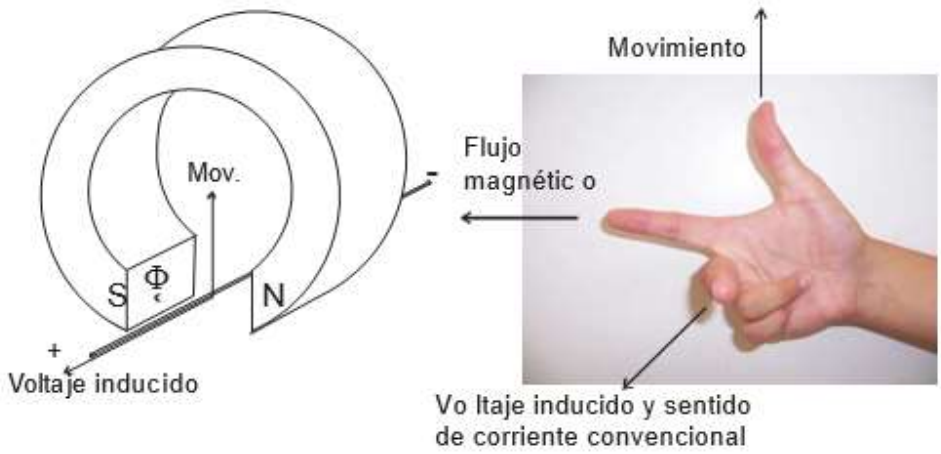



***Figura 6.5.-*** *Regla de Fleming. (a) voltaje inducido, (b) regla de la mano derecha. Fuente: Tesis de grado [1].*

Cuando la armadura de un motor gira a consecuencia del par que se produce por la acción motor, actúa como generador al mismo tiempo. Como existe un campo magnético producido por los imanes a fin de que se produzca la acción motora, ese mismo campo genera un voltaje en los conductores de la armadura. De la ley de Faraday, la relación entre la regla de Fleming de la mano derecha de la acción generador y la regla de la mano izquierda de la acción motor, se aprecia que el voltaje inducido se opone a la corriente eléctrica producida por el voltaje aplicado que causa la acción motora. A este voltaje inducido que se opone al voltaje aplicado se le conoce como fuerza contraelectromotriz.

Cuando una fuente eléctrica es conectada al motor, esta gira su eje, por lo que un motor convierte la energía eléctrica en energía mecánica.

Una aplicación de los motores de cc es aquella situación en que se necesitan amplias variaciones de velocidad. Hasta hace poco tiempo, los motores de cc eran insuperables en aplicaciones de control de velocidad.

Se sabe que una máquina de cc puede funcionar como generador y como motor, se conoce que cuando funciona como generador debe estar acoplado a un motor primario para su funcionamiento, en cambio, cuando funciona como motor de cc se le aplica una tensión entre escobillas, en las cuales se crearan unas fuerzas mecánicas, con lo que conseguirá un movimiento en los conductores del devanado. Este funcionamiento hace que la máquina de cc funcione como motor.

El movimiento que produce el inducido genera una fuerza contraelectromotriz, que se opone a la tensión aplicada a las escobillas, ésta es llamada fuerza contraelectromotriz.

El sentido de la fuerza contraelectromotriz, según la ley de Lenz, es opuesto a la tensión de la red, por ser ella la que motiva el giro del inducido.

El inducido es la parte de la máquina de cc que transforma la energía eléctrica en mecánica y viceversa. Tanto si funciona como generador o como motor, produce fuerza contraelectromotriz y desarrolla a la vez fuerzas mecánicas.

Cuando la máquina funciona como generador, produce una fuerza contraelectromotriz, de signo positivo y reacciona con un par negativo, llamado par resistente; se llama resistente por oponerse al par positivo dado por la máquina que mueve al generador.

Si una máquina funciona como motor, producirá un par motor positivo a la vez que ofrece una reacción, llamada fuerza contraelectromotriz (f.c.e.m) por ser de signo opuesto a la tensión de la red.

## 6.2 Motorreductores

### 6.2.1 Funcionamiento

Los reductores y motorreductores mecánicos de velocidad se pueden contar entre los inventos más antiguos de la humanidad y aún en estos tiempos del siglo XXI se siguen utilizando prácticamente en cada máquina que tengamos a la vista, desde el más pequeño reductor o motorreductor capaz de cambiar y combinar velocidades de giro en un reloj de pulsera, cambiar velocidades en un automóvil, hasta enormes motorreductores capaces de dar tracción en buques de carga, molinos de cemento, grandes máquinas cavadoras de túneles o bien en molinos de caña para la fabricación de azúcar. Un motorreductor tiene un motor acoplado directamente, el reductor no tiene un motor acoplado directamente. La sencillez del principio de funcionamiento y su grado de utilidad en una gran variedad de aplicaciones es lo que ha construido la trascendencia de este invento al través de los siglos.

A continuación, se dan los principios básicos de un reductor o motorreductor de velocidad: Supongamos que la rueda “A” (ver *Figura 6.6*) tiene un diámetro de X cm y la rueda “B” es el triple de la “A”, o sea 3X cm, entonces recordando la fórmula para calcular el perímetro:

$$Perimetro\ [cm] = \pi\ x\ d[cm] \tag{6.1}$$

El perímetro de la rueda A y B respectivamente será:

$$Perimetro_A = \pi\ x\ X = X\pi\ cm \tag{6.2}$$

$$Perimetro_B = \pi\, x\, (3X) = 3X\pi\ cm \quad (6.3)$$

El perímetro es la longitud total de la envolvente de la rueda.

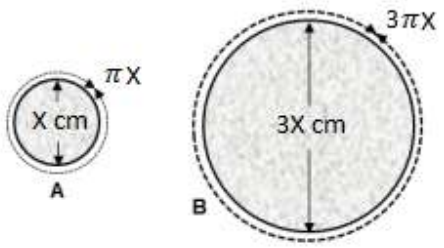



***Figura 6.6.-*** *Diámetros ruedas A y B. Fuente: Potencia electromecánica [2].*

### 6.2.2 Relación de reducción

En la *figura 6.7*, cuando gira la rueda "A" hará que a su vez gire la rueda "B" pero sucederá que por cada tres vueltas que dé "A", la rueda "B" solamente dará una vuelta, esto es, el diámetro de "B" dividido por el diámetro de "A" (3x/x = 3). Este número 3 será la relación de reducción de este reductor o motorreductor elemental y se indica como 3:1.

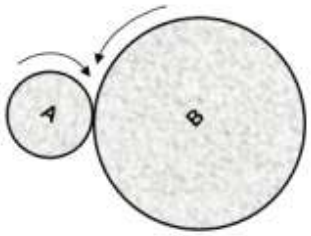



***Figura 6.7.-*** *La rueda A gira en un sentido y hace girar a la rueda B en el sentido opuesto. Fuente: Potencia electromecánica [2].*

Con esta simple combinación se ha logrado disminuir la velocidad de rotación de la rueda "B" a la tercera parte de la velocidad de la rueda "A". Si a la combinación de ruedas antes descrito encadenamos otras ruedas adicionales entonces cada vez lograremos una velocidad cada vez menor hasta donde sea necesario para la aplicación y puede ser por ejemplo 6:1, 30:1, 100:1 o aún mayor para lograr velocidades muy pequeñas que se pudieran necesitar y que, por ejemplo, la rueda "A" tuviera que girar cientos de veces para que la última rueda girara una sola vez.

En este caso tendremos un motorreductor de varios trenes de reducción, entendiendo como un tren de reducción a un par de ruedas. Con seis ruedas tendríamos tres trenes de engranes. Con este sistema de reducción no solamente disminuimos la velocidad de "B" a un giro más lento que es útil para la mayoría de las aplicaciones, sino que al mismo tiempo estaremos aumentado el "par" o "torque" en la última rueda del motorreductor que generalmente se conoce como la rueda de salida a la que va ensamblada la "flecha de salida" del reductor o motorreductor.

### 6.2.3 Par o torque en un motorreductor

El torque o par es una fuerza de giro. Si se observa la *figura 6.8*, es la fuerza de giro de salida del motorreductor y es también la fuerza de giro de un motor.


***Figura 6.8.-*** *Torque o Par en un motor CC. Fuente: Potencia electromecánica [2].*

Este torque o par a lo largo de un tiempo de determinado, aplicado o ejecutado se convierte en una potencia. Un motor eléctrico tiene una determinada potencia (expresadas en HP) y tiene una cierta velocidad de operación a la cual gira la salida, medida en revoluciones por minuto (RPM). Estas dos características (velocidad y potencia) llevan aparejado un cierto torque o par que puede liberar el motor. Es precisamente lo que permitirá que se pueda o no girar una determinada carga, cuanto más alto el par más grande será la carga que podamos girar. El que tan rápido se pueda hacerlo dependerá de la potencia del motorreductor. Las dos características están interrelacionadas y dependen una de la otra. Esta combinación de potencia, par y velocidad en un motor o motorreductor está regida por la siguiente fórmula:

$$\tau \left[\frac{kg}{m}\right] = \frac{Potencia\ [HP]\ x\ 716}{Velocidad\ [rpm]} \qquad (6.4)$$

Como se observa en la fórmula, para una potencia dada, cuanto más baja sea la velocidad de giro del motorreductor, más alto será el par, aunque la potencia siga siendo la misma. Inversamente: Cuanto más alta sea la velocidad del reductor o motorreductor, será más bajo el par aun cuando la potencia sea la misma.

### 6.2.4 Engranaje planetario

El reductor planetario se compone de tres elementos: un engranaje central denominado planeta o sol, uno o más engranajes conocidos como satélites, que están situados a su alrededor, y una corona central o interna (ver *figura 6.9*). El eje de conexión o entrada está conectado a su vez con el engranaje principal o central. Éste es el encargado de transmitir el movimiento rotacional a los engranajes satélite. Los engranajes satélites a su vez rotan sobre la corona central del reductor, así transmiten el movimiento al eje de salida del reductor.

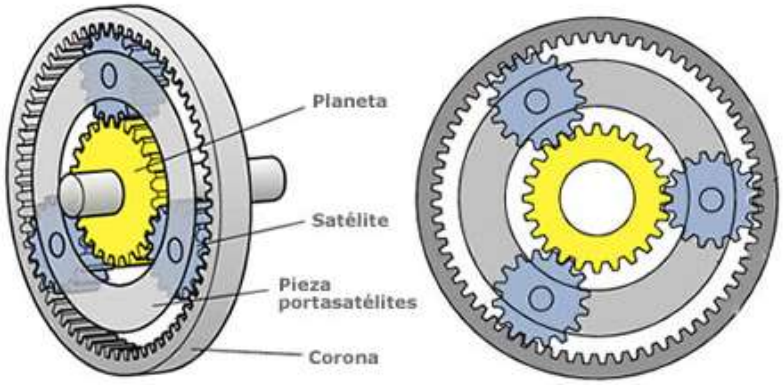



***Figura 6.9.-*** *Partes de un sistema planetario. Fuente: Mecánicos. Tren de engranajes planetarios. [3].*

Por otro lado, los reductores planetarios de precisión pueden ser de una o varias etapas de reducción. Los engranajes helicoidales aumentan la superficie de contacto en comparación con el engranaje recto. De esta forma consiguen un juego más reducido y menor ruido. Su funcionamiento está gobernado por cinco estados que proporcionan la clave para entender los diferentes flujos de potencia de engranajes. Para que se den los distintos estados de potencia, hacen falta cinco estados diferentes que creen las condiciones necesarias. Estos cinco estados son: el estado neutro, el estado de reducción también llamado estado de marcha, el estado de supermarcha, el estado de transmisión directa y el estado de inversa.

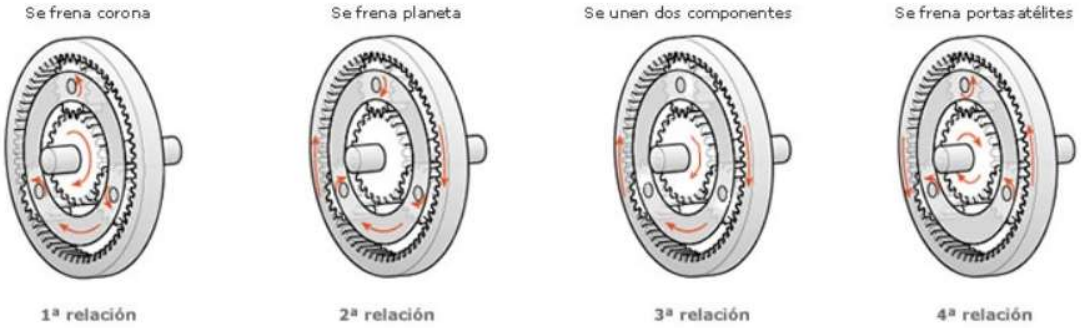



***Figura 6.10.-*** *Funcionamiento del engranaje epicicloidal. Fuente: Blog Seas. [4].*

- Estado Neutro: Ninguno de los elementos del planetario está bloqueado (Punto muerto). El piñón actúa como miembro de entrada conductor, y los satélites rotan libremente sobres sus ejes pues la corona también puede girar libremente.

- Estado de Reducción (1° relación): Pongamos un elemento de reacción fijo como ser la corona y que la salida sea el portasatélite el cual transmitirá el movimiento a las ruedas.

- Estado de Supermarcha (2° relación): cuando tenemos un elemento de reacción fijo y el portasatélite es la entrada, en este caso tenemos una multiplicación del giro, produciendo un efecto contrario a la 1° relación (estado de reducción de marcha), reduciendo el par y aumentando la velocidad.

- Estado de transmisión directa (3° relación): Obtenemos este estado bloqueando entre sí dos miembros cualesquiera del tren de engranajes planetarios. Conducir dos miembros al mismo tiempo con relación a la velocidad y en la misma dirección produce el mismo efecto.

- Estado de Inversa (4° relación): Este estado lo obtenemos reteniendo el portasatélite para que no rote, entonces la corona y el piñón tendrán sentido de giro distintos, sea que la entrada fuere por el piñón y la salida por la corona o viceversa.

| Reacción | Corona | Planeta | Portasatélite | Desmultiplicación |
|---|---|---|---|---|
| **1** | Fija | Salida de la Fuerza | Impulsión | Grande |
| **2** | Salida de la Fuerza | Fija | Impulsión | Menor |
| **3** | Fija | Fija | Salida de la Fuerza | Sin desmultiplicación |
| **4** | Impulsión | Salida de la Fuerza | Fija | Inversión de giro |

***Tabla 6.1.-*** *Resumen de las distintas etapas en un sistema planetaria. Fuente: Propia*

## 6.3 Controladores clásicos para el motor de cd

El control automático es parte importante e integral en la industria. Con los avances en la teoría y en la práctica del control automático se ha logrado obtener un mejor desempeño de los sistemas. La manera en la cual el controlador automático produce la señal de control se denomina acción de control.

La variable controlada es la cantidad o condición que se mide y controla, por ejemplo: posición, velocidad y/o par de un motor de cd. Comúnmente la variable controlada es la salida del sistema también denominada señal de salida.

La variable manipulada es la cantidad o condición que el controlador modifica para afectar el valor de la variable controlada. También se denomina señal de control.

La planta es cualquier objeto físico que se desea controlar, tal como, un dispositivo mecánico, eléctrico, biológico, térmico, electromecánico, etc.

A una señal que tiende a afectar adversamente el valor de salida de un sistema se le conoce como perturbación. Si ésta se genera dentro del sistema se le conoce como perturbación interna, si se produce fuera del sistema se le denomina perturbación externa y es una entrada.

### 6.3.1 Sistemas de control de lazo abierto

Un sistema de control en lazo abierto es aquel en donde la salida no afecta la acción de control. En un sistema de control en lazo abierto no se realimenta la salida como se muestra en la *figura 6.11*.

Los sistemas en los cuales la salida no tiene efecto sobre la acción de control se denominan sistemas de control en lazo abierto. En otras palabras, en un sistema de control en lazo abierto no se mide la salida ni se realimenta para compararla con la entrada.

En cualquier sistema de control en lazo abierto, la salida no se compara con la entrada de referencia. Así, a cada entrada de referencia le corresponde una condición de operación fija. En la práctica el control en lazo abierto sólo se usa si se reconoce la relación entre la entrada y la salida y si no hay perturbaciones internas y externas.

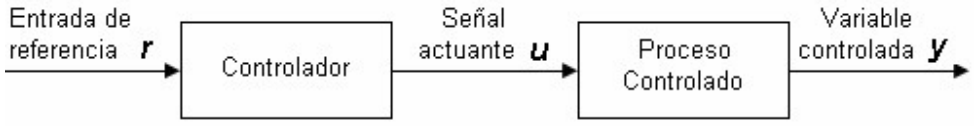



***Figura 6.11.-*** *Elementos en un sistema de control de lazo abierto. Fuente: Propia.*

### 6.3.2 Sistemas de control de lazo cerrado

El control realimentado es una estrategia que en presencia de perturbaciones trata de reducir la diferencia entre la salida deseada de un sistema y alguna entrada de referencia.

Los sistemas de control realimentados o retroalimentados se denominan también sistemas de control en lazo cerrado. El control en lazo cerrado implica el uso de una acción de control retroalimentada para reducir el error en un Sistema.

En la *figura 6.12* se muestra el diagrama en bloque de un control de lazo cerrado, también denominado sistema con realimentación negativa por el signo negativo que tiene el bloque de suma. El bloque $G_p(s)$ representa la función de transferencia (sus siglas FT) de la plata, $G_c(s)$ representan la FT de la acción de control y $H(s)$ la FT del sensor. Los elementos restantes son señales, donde $R(s)$ es la señal de entrada de referencia, $E(s)$ la señal de error, $U(s)$ la señal de control, $C(s)$ la señal de salida y $B(s)$ la señal de realimentación. A todas estas señales y bloques se les ha aplicado la transformada de Laplace para que sea más fácil resolver estas operaciones algebraicas dentro del plano complejo.

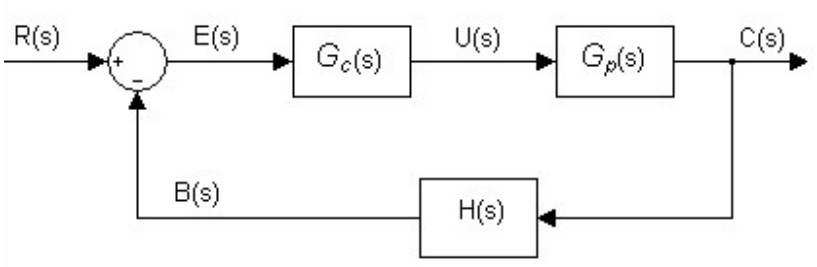



***Figura 6.12.-*** *Sistema de control de lazo cerrado. Fuente: Propia*

En le figura 6.13 se muestra la forma general de un control de lazo cerrado, donde el bloque controlador $G_c(s)$ y el bloque de la planta $G_p(s)$ se simplifican en el bloque $G(s)$ conocido como función de transferencia directa.

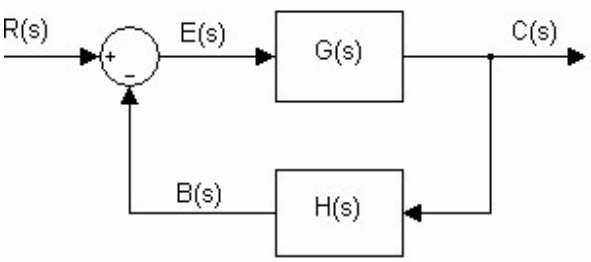



***Figura 6.13.-*** *Sistema general de un control de lazo cerrado. Fuente: Propia*

La señal de entrada $G(s)$ y la señal de error $E(s)$quedan determinadas por:

$$C(s) = G(s)E(s) \tag{6.5}$$

$$E(s) = R(s) - B(s) = R(s) - C(s)\,H(s) \tag{6.6}$$

Sustituyendo (6.5) en (6.6) y mediante operaciones algebraicas se obtiene la ecuación general de una función de transferencia de lazo cerrado

$$\frac{C(s)}{R(s)} = \frac{G(S)}{1+G(s)H(s)} \tag{6.7}$$

En la ecuación (6.3) se observa la presencia del término $G(s)H(s)$ denominado función de transferencia de lazo abierto, que es la función formada por el producto de la función de transferencia directa $G(s)$ y la función de transferencia de la realimentación $H(s)$. El termino completo de la ecuación (6.7) al ser igualado a acero forma una relación que es la utilizada para proporcionar los polos de la función de lazo cerrado, denominada ecuación característica (6.8).

$$1 + G(s)H(s) = 0 \tag{6.8}$$

En general, $G(s)$ y $H(s)$ se obtienen como cocientes de polinomios, así la función de transferencia en lazo cerrado obtenida en (6.3) se escribe como:

$$\frac{C(s)}{R(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \cdots + b_{m-1}s + b_m}{a s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n} \quad (m \le n) \tag{6.9}$$

De donde $b_0, b_1, \ldots, b_m$ y $a_0, a_1, \ldots, a_n$ son los coeficientes de los polinomios, $m$ y $n$ son las potencias más altas de $s$ de los polinomios del numerados y denominador respectivamente. Si la potencia más alta de $s$ de los polinomios del denominador de la función de transferencia es igual a $n$, el sistema se denomina sistema de $n$-ésimo orden.

Otra forma de expresar la ecuación (6.9) mediante las raíces de los polinomios del numerador y del denominador, una vez factorizados los polinomios de la ecuación (6.9) se escribe como:

$$\frac{C(s)}{R(s)} = \frac{K(s+z_1)(s+z_2)\ldots(s+z_m)}{(s+p_1)(s+p_2)\ldots(s+p_n)} \quad (m \le n) \tag{6.10}$$

De donde $z_1, z_2, \ldots, z_m$ y $p_1, p_2, \ldots, p_n$ son las cantidades reales o complejas que proporcionan los ceros y polos respectivamente, de la función de transferencia.

### 6.3.3 Estabilidad, respuesta transitoria y error en estado estacionario

El diseño de un sistema de control debe ser capaz de predecir su comportamiento dinámico a partir del conocimiento de sus componentes. La característica más importante del comportamiento dinámico de un sistema de control es la estabilidad absoluta, es decir, si un sistema es estable o inestable.

Un sistema de control está en equilibrio si en ausencia de cualquier perturbación o entrada, la salida permanece en el mismo estado. Un sistema de control es estable si la salida termina por regresar a su estado de equilibrio cuando el sistema está sujeto a una condición inicial. Un sistema de control es críticamente estable si las oscilaciones de la salida continúan por siempre. Es inestable si la salida diverge sin límite a partir de su estado de equilibrio cuando el sistema está sujeto a una condición inicial.

Entre los comportamientos importantes de un sistema que deben recibir una cuidadosa consideración, aparte de la estabilidad absoluta, están la respuesta transitoria y el error en estado estacionario $e_{ss}$. Cuando un sistema ha alcanzado un estado de equilibrio se dice que se encuentra en estado estacionario.

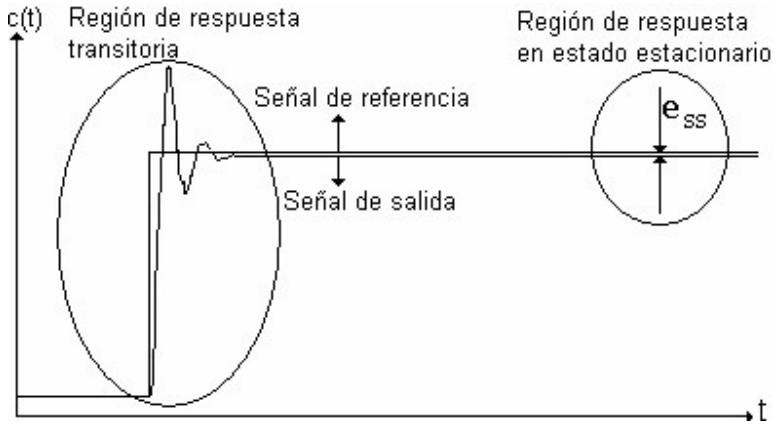



***Figura 6.14.-*** *Regiones de interés en un sistema. Fuente: UNEFA [8]*

En la *figura 6.14*, se muestran las dos regiones temporales de interés para especificar el comportamiento de un sistema. La primera región se llama respuesta transitoria y la segunda región es la de estado estacionario.

Cuando un sistema de control está sujeto a una entrada, la salida no cambia inmediatamente, si no que exhibe una respuesta transitoria antes de alcanzar un estado estacionario, siempre y cuando, el sistema sea estable.

En la práctica, la respuesta transitoria de un sistema de control presenta oscilaciones amortiguadas antes de alcanzar un estado estacionario. Si la salida de un

sistema no coincide exactamente con la referencia, se dice que el sistema tiene un error en estado estacionario $e_{ss}$. Este error indica la precisión del sistema.

Dentro de los sistemas de control estables existen cuatro tipos de clasificaciones bien definidas:

- El sistema amortiguado, *figura 6.15 (a).*
- El sistema críticamente amortiguado, *figura 6.15 (b).*
- El sistema sobre amortiguado *figura 6.15 (c).*
- El sistema críticamente estable, *figura 6.15 (d).*

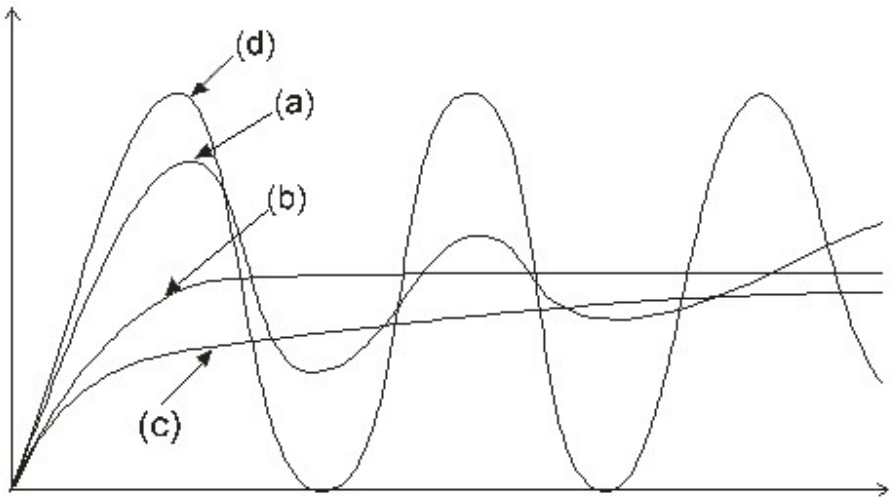



***Figura 6.15.-*** *Clasificación de un sistema de control estable. Fuente: Propia*

#### 6.3.3.1 Respuesta transitoria

Comúnmente los sistemas no responden instantáneamente ante una entrada y exhiben respuestas transitorias sujetos a estas entradas o perturbaciones. Frecuentemente, las características de desempeño de un sistema de control se especifican en términos de la respuesta transitoria para una entrada escalón unitario, dado que ésta es

fácil de generar y suficientemente drástica.

La respuesta transitoria de un sistema para una entrada escalón unitario depende de las condiciones iniciales. Por conveniencia, al comparar respuestas transitorias de varios sistemas, es común que las condiciones iniciales del sistema sean cero. De este modo, las características de respuesta se comparan con facilidad.

Al especificar las características de respuesta transitoria de un sistema de control para una entrada escalón unitario, es común especificar las siguientes definiciones:

- Tiempo de retardo $t_d$, es el tiempo requerido para que la respuesta alcance por primera vez la mitad del valor final.
- Tiempo de levantamiento $t_r$, es el tiempo requerido para que la respuesta pase del 10% al 90%, del 5% al 95% o del 0% al 100% de su valor final. Para sistemas subamortiguados de segundo orden, comúnmente se usa el tiempo de levantamiento de 0% a 100%. Para sistemas sobre amortiguados, suele usarse el tiempo de levantamiento de 10% a 90%.
- Tiempo pico $t_p$, es el tiempo requerido para que la respuesta alcance el primer pico de sobre paso.
- Sobre paso máximo $M_p$, es el valor pico máximo de la curva de respuesta, medido a partir de la unidad. Si el valor final en estado estacionario es diferente de la unidad, es común usar el porcentaje de sobre paso máximo, que se define mediante:

$$Porcentaje\ de\ sobrepaso\ maximo = \frac{c(t_p)-c(\infty)}{c(\infty)}\ x\ 100\% \qquad (6.11)$$

- Tiempo de asentamiento $t_s$, es el tiempo que se requiere para que la curva de la respuesta alcance un rango dentro del valor final, de tamaño especificado por el porcentaje absoluto del valor final y la señal permanezca dentro de este rango, por lo general ± 2% a ± 5%. El tiempo de asentamiento se relaciona con la mayor constante de tiempo del sistema de control. Los objetivos particulares de diseño de un sistema de control determinan que criterio de error en porcentaje utilizar.

Estas especificaciones se resumen en la *tabla 6.2* y se aprecian en forma gráfica en la *figura 6.16.*

Las especificaciones en el dominio del tiempo que se han proporcionado son importantes, ya que casi todos los sistemas de control son sistemas en el dominio del tiempo; es decir, deben presentar respuestas de tiempo aceptables. Esto significa que el sistema de control debe modificarse hasta que la respuesta transitoria sea satisfactoria.

| | |
|---|---|
| Tiempo de retardo | $t_d$ |
| Tiempo de levantamiento | $t_r$ |
| Tiempo pico | $t_p$ |
| Sobre paso máximo | $M_p$ |
| Tiempo de asentamiento | $t_s$ |

***Tabla 6.2.-*** *Especificaciones. Fuente: Propia*

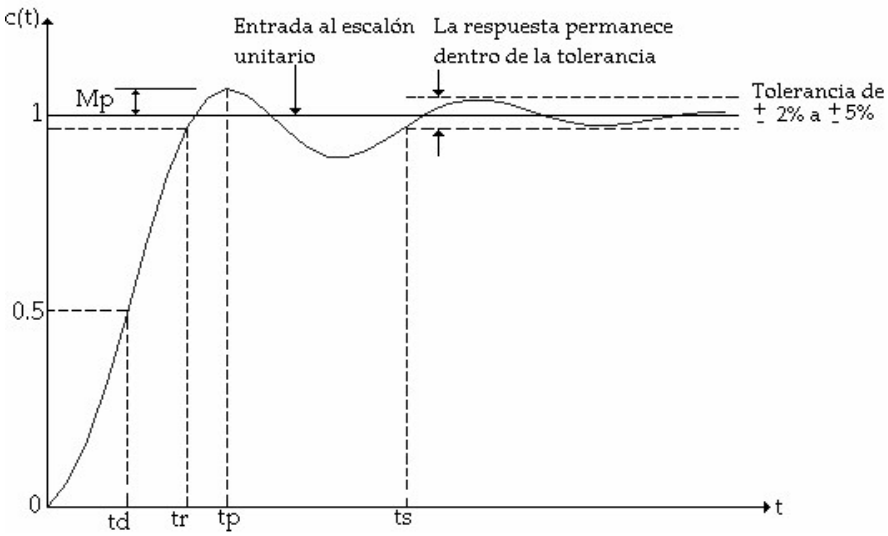



***Figura 6.16.-*** *Detalles a la curva de un sistema de control ante una entrada escalón. Fuente: UNEFA [8]*

### 6.3.3.2 Error en estado estacionario

Una característica importante de los sistemas de control es su comportamiento en estado estacionario. Por error en estado estacionario se debe entender que es la diferencia entre el valor al cual debió llegar la respuesta y el valor que realmente alcanzó después de transcurrido un tiempo en la referencia.

En la práctica, cualquier sistema de control presenta un error en estado estacionario a ciertos tipos de entrada. Un sistema puede no tener un error en estado estacionario a una entrada escalón, pero el mismo sistema puede exhibir un error en estado estacionario diferente de cero a una entrada rampa. El que un sistema determinado exhiba un error en estado estacionario a un tipo específico de entrada, depende del tipo de función de transferencia en lazo abierto del sistema.

Anteriormente se había dicho que para el sistema mostrado en la *figura 6.13*, la función de transferencia en lazo cerrado es (6.3).

$$\frac{C(s)}{R(s)} = \frac{G(S)}{1+G(s)H(s)} \tag{6.12}$$

Y la función de transferencia que define a la señal de error entre la señal de entrada es:

$$\frac{E(s)}{R(s)} = \frac{1}{1+G(s)H(s)} \tag{6.13}$$

Que al despejar el error se obtiene la siguiente ecuación:

$$E(s) = \frac{R(s)}{1+G(s)H(s)} \tag{6.14}$$

#### 6.3.3.3 Tipos de señal de entrada

Aunque exista una gran variedad de formas que puede adoptar la señal de entrada $r(t)$ a un sistema, éstas se pueden clasificar de acuerdo con tres tipos básicos

- Entrada escalón unitario $r(t) = u(t)$
- Entrada rampa unitaria $r(t) = t$
- Entrada parábola unitaria $r(t) = t^2$

En la *tabla 6.3* se resumen los tipos básicos de entradas, sus funciones en el dominio del tiempo y su transformada de Laplace, así como sus formas de ondas correspondientes.

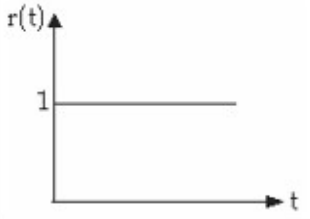
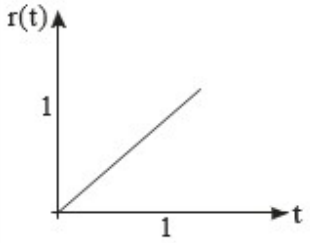
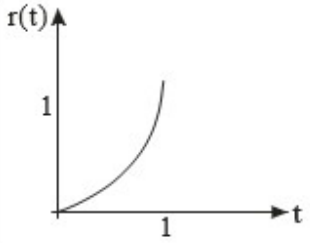
| Entrada | Función en el tiempo | Transformada de Laplace | Graficas en el tiempo |
|---|---|---|---|
| Escalón unitario | $r(t) = u(t)$ | $R(s) = \frac{1}{s}$ |   |
| Rampa unitaria | $r(t) = t$ | $R(s) = \frac{1}{s^2}$ |   |
| Parábola unitaria | $r(t) = t^2$ | $R(s) = \frac{1}{s^3}$ |   |

***Tabla 6.3.-*** *Características de los distintos tipos de señales de entrada. Fuente: Propia*

#### 6.3.3.4 Clasificación de los sistemas de control

Los sistemas de control se clasifican de acuerdo con su capacidad de seguir entradas escalón, rampa, parábola, etc. Para poder clasificar los tipos de sistemas es necesario expresar la función de transferencia en lazo abierto en la forma:

$$G(s)H(s) = \frac{K}{s^N}\left(\frac{b_0 s^m + b_1 s^{m-1} + \cdots + b_{m-1}s + b_m}{a s^j + a_1 s^{j-1} + \cdots + a_{j-1}s + a_j}\right) \quad (m < j) \qquad (6.15)$$

El sistema contiene el término $s^N$ en el denominador, representa un polo de multiplicidad *N* en el origen. El esquema de clasificación se basa en la cantidad de integraciones indicadas por $s^N$ en la función de transferencia en lazo abierto. Un sistema se denomina del tipo *0* si *N*=0, tipo *1* si *N*=1, tipo *2* si *N*=2, ..., y así respectivamente. Esta clasificación es diferente de la que se basa en el orden del Sistema. Conforme *N* es mayor, mejora la exactitud; sin embargo, al aumentar *N* también se agrava el problema de estabilidad. En la práctica, es raro encontrar sistemas de tipo 3 o superior, por lo general resulta difícil diseñar sistemas estables con dos o más integradores.

El sistema tipo 0 no tiene polos en el origen en tanto que los sistemas tipo 1 y 2 tienen uno y dos polos en el origen respectivamente. Por tanto, como ya se mencionó, el tipo de sistema está asociado al número de polos que estén en el origen definidos por la función de transferencia en lazo abierto.

### 6.3.4 Acciones de control

Existen varias acciones de control, entre las cuales se encuentran

- Acción de Control Proporcional (P).
- Acción de Control Proporcional Derivativa (PD).
- Acción de Control Proporcional Integral (PI).
- Acción de Control Proporcional Integral Derivativa (PID).

#### 6.3.4.1 Acción de Control Proporcional

La acción de control proporcional (P) siempre está presente en los sistemas de lazo cerrado lineales. El efecto que tiene la acción proporcional sobre el comportamiento de los sistemas es incrementar la exactitud de estos, provocando también en la mayoría de los casos un incremento en la velocidad de respuesta, resultando un incremento en el sobrepaso, causando oscilaciones en la salida. La relación entre la salida del controlador $u(t)$ y la señal del error $e(t)$, para este tipo de acción proporcional es:

$$u(t) = k_p\, e(t) \qquad (6.16)$$

Es decir, que la acción de control es proporcional a la señal de error; la función de transferencia del controlador proporcional es una constante:

$$\frac{U(s)}{E(s)} = K_p \tag{6.17}$$

$$G_c(s) = K_p \tag{6.18}$$

Donde $K_p$ es denominada ganancia proporcional. Es importante resaltar la diferencia que existe entre los términos $k_p$ y $K_p$, donde el primer termino indica el coeficiente estático de error de posición y el segundo la ganancia proporcional.

El control proporcional es una de las acciones de control más fáciles de implementar, sin embargo, si la señal de error es muy pequeña, también lo será la acción de control. Entonces la acción proporcional podría ser insuficiente para minimizar o eliminar el error en estado estacionario en la salida del sistema. Otro inconveniente del control proporcional es que si la ganancia proporcional es muy grande el sistema funcionará inadecuadamente debido al ruido. En la *figura 6.17* se muestra un diagrama a bloques simplificado del controlador proporcional.

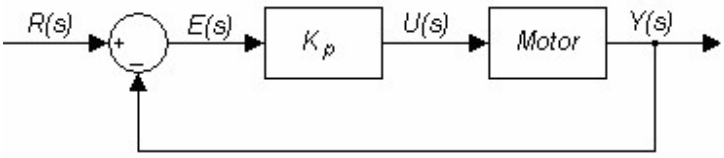



***Figura 6.17.-*** *Diagrama en bloque de un control proporcional. Fuente: Propia*

Dentro de las principales características de este tipo de controlador se tienen las siguientes.

- Hacer más rápida la respuesta del sistema.
- Incrementa la ganancia del controlador.
- Al incrementar la ganancia puede provocar que el sistema sea más oscilatorio.

#### 6.3.4.2 Acción de Control Proporcional Derivativa

La acción de control derivativa (D) siempre debe ir acompañada de otras acciones de control y no puede ser aplicada sola. La definición del control derivativo indica que la señal de control $u(t)$ es proporcional a las variaciones de la señal de error $e(t)$. Una expresión que indica el comportamiento de esta acción en el dominio del tiempo es

$$u(t) = K_D \frac{d_e(t)}{dt} \tag{6.19}$$

Donde $K_D$ es una constante ajustable conocida como ganancia o constante de acción derivativa. Por lo tanto, la acción de control derivativa siempre va a acompañada por la acción proporcional o proporcional integra.

Al combinarse el efecto de la acción de control proporcional con la derivativa, se dice que se tiene un control proporcional derivativo (PD) cuyo comportamiento se define mediante la ecuación siguiente:

$$u(t) = K_p\, e(t) + K_D \frac{d_e(t)}{dt} \qquad (6.20)$$

La función de transferencia del control proporcional derivativa es la siguiente:

$$\frac{U(s)}{E(s)} = K_p + K_D s \qquad (6.21)$$

$$G_c(s) = K_p + K_D s \qquad (6.22)$$

En la *figura 6.18*, se muestra un diagrama en bloque simplificado del controlador proporcional derivativo.

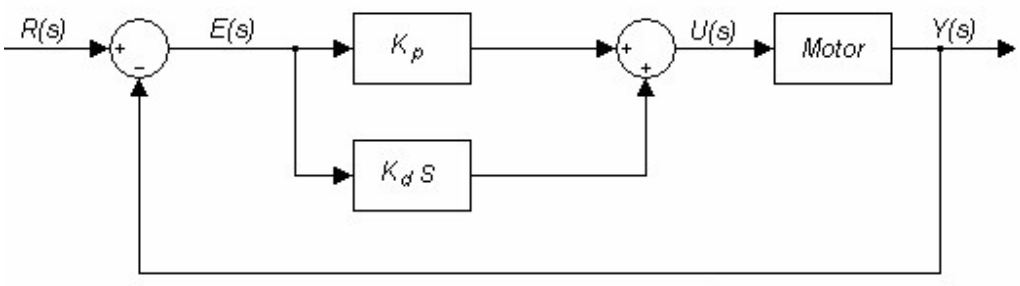



***Figura 6.18.-*** *Diagrama en bloque de un control proporcional derivativo. Fuente: Propia*

El control derivativo no afecta en forma directa el error en estado estacionario, pero sí adiciona amortiguamiento al sistema, por lo que se puede dar un valor más grande a la constante proporcional, lo cual provoca una mejora en la exactitud en estado estacionario.

Dentro de las principales características de este tipo de controlador se tienen las siguientes:

- Aumenta el coeficiente de amortiguamiento $K_D$ lo que reduce las oscilaciones del sistema en estado transitorio.
- El control $K_D$ reduce el tiempo de levantamiento.
- No aumenta el orden del sistema en lazo cerrado.

#### 6.3.4.3 Acción de Control Proporcional Integrativo

En un controlador con acción de control integral $(i)$, el valor de la salida del controlador $u(t)$ varía proporcionalmente a la señal de error $e(t)$ conforme a la siguiente relación

$$\frac{du(t)}{dt} = K_i\, e(t) \tag{6.23}$$

Resolviendo para $u(t)$ se llega al porqué se le denomina acción integral, es decir, la acción de control es la integral de la señal de error en el tiempo como se muestra a continuación:

$$u(t) = K_i \int_0^t e(t)dt \tag{6.24}$$

Donde $K_i$ es una constante ajustable, conocida como ganancia o constante de acción integral. Al combinar el efecto de la acción de control proporcional con la integral se dice que se tiene un control proporcional integral (PI) cuyo comportamiento se define con la siguiente ecuación.

$$u(t) = K_p\, e(t) + K_i \int_0^t e(t)dt \tag{6.25}$$

La función de transferencia del control proporcional integral es la siguiente:

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} \tag{6.26}$$

$$G_c(s) = K_p + \frac{K_i}{s} \tag{6.27}$$

En la *figura 6.19* se muestra un diagrama a bloques simplificado del controlador proporcional integral.

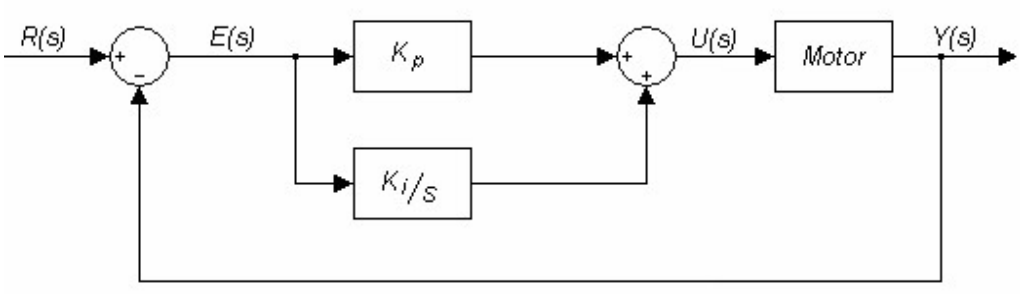



***Figura 6.19.-*** *Diagrama en bloque de un control proporcional Integral. Fuente: Propia*

Dentro de las principales características de este tipo de controlador se tienen las siguientes:

- Aumenta el coeficiente de amortiguamiento.
- Incrementa el tiempo de levantamiento.
- Mejora el error en estado estable

#### 6.3.4.4 Acción de Control Proporcional Integral Derivativo

La combinación de una acción de control proporcional, con una acción de control Integral y una acción de control Derivativa se denomina acción de control Proporcional Integral Derivativa (PID). Esta acción combinada tiene las ventajas de cada una de las tres acciones de control individuales. La ecuación del controlador con esta acción combinada es:

$$u(t) = K_P\, e(t) + K_I \int_0^t e(t)dt + K_D \frac{d_e(t)}{dt} \qquad (6.28)$$

La función de transferencia del control proporcional integral derivativa es la siguiente

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_D s \qquad (6.29)$$

$$G_c(s) = K_p + \frac{K_i}{s} + K_D s \qquad (6.30)$$

En la *figura 6.20* se muestra un diagrama a bloques simplificado del controlador proporcional integral derivativo.

Esta acción combinada tiene la característica de cada una de las tres acciones de control individuales que se mencionó anteriormente.

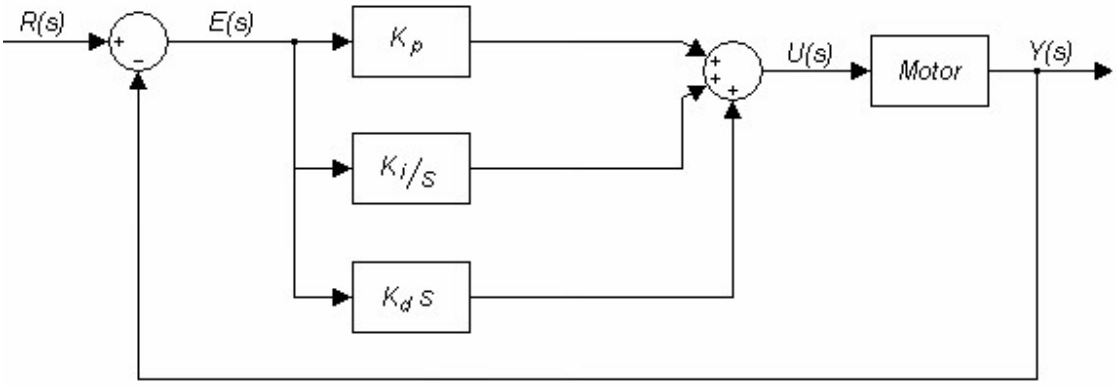



***Figura 6.20.-*** *Diagrama en bloque de un control Proporcional Integral Derivativo. Fuente: Propia*

## 6.4 Modelado de motor CC

### 6.4.1 Modelo matemático de un motor DC

Los motores DC suelen tener un comportamiento lineal en un rango predefinido de tensiones de entrada. Sin embargo, a bajas velocidades angulares la fricción produce no linealidades, además de que el motor sufre perturbaciones a su entrada y ruido a su salida, normalmente debidas, las primeras al efecto de la carga y la segunda al ruido en la medida de los sensores. También sufre el motor variaciones no lineales cuando se utiliza de forma continua, debido a efectos térmicos produciendo variaciones en la resistencia de la bobina. A tensiones elevadas puede haber una desmagnetización del hierro, lo que supone variaciones en las constantes del motor. Por último, la etapa de potencia, es decir, la electrónica que acondiciona la señal de baja potencia de un microcontrolador a la potencia eléctrica del motor suele añadir ruido a su salida. Si se utiliza como etapa de potencia un puente en H para la generación de una señal PWM puede ocurrir que la tensión de referencia no sea suficientemente estable, debido al conversor AC/DC. Como consecuencia la tensión de entrada al motor no sería la deseada, sino que estaría afectada por un ruido difícil de modelar.

Tener un modelo matemático de un sistema tiene muchas ventajas. Algunas de estas son:

- Permite estimar la respuesta del sistema a diferentes entradas.
- Estudiar el desempeño del sistema en lazo cerrado variando los parámetros del controlador.
- Conocer las condiciones de operación del motor para el desempeño deseado: cuanto voltaje, corriente o torque necesita el motor para mover una carga o seguir un perfil de movimiento. Así, se puede dimensionar adecuadamente el motor, la circuitería de potencia, la fuente de alimentación.
- También, si se quiere agregar un sensor de velocidad o de solo de posición, entonces nos vemos bajo la necesidad de diseñar, a parte del controlador, un algoritmo para estimar la velocidad a partir de una señal de posición.

Estas son algunas de las ventajas del diseño basado en el modelo. Es un paradigma bastante utilizado en la industria que permite reducir los tiempos de entrega de productos y los costos, ya que permite la iteración rápida de cambios de diseño. Muchas veces no funciona, entonces se hace modificaciones y se simula de nuevo. Así podemos estar hasta encontrarnos con un modelo más complejo y preciso del sistema real. Una vez encontrado el modelo, se construye el prototipo.

Para modelar un motor DC, partimos de su circuito eléctrico. Como sabemos que es una maquina capaz de convertir energía eléctrica a trabajo mecánico, entonces, se espera encontrar en el modelo matemático aparezcan ecuaciones de tensiones y corrientes acopladas a ecuaciones de fuerzas y velocidades.

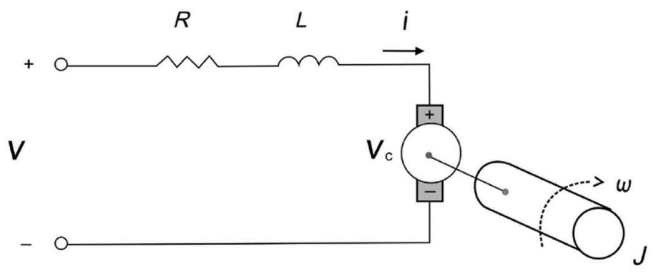



***Figura 6.21.-*** *Modelo de un motor de corriente directa.*

Suponemos que se puede definir, que la parte eléctrica del motor se puede modelar como se muestra a continuación:

$$L\frac{di}{dt} + Ri + V_c = V_{in} \tag{6.31}$$

Donde $i$ es la corriente que consume el motor, $R$ y $L$ son la resistencia e inductancia del embobinado del motor, $V_{in}$ es el voltaje de entrada y $V_c$ es el voltaje de fuera contra electromotriz.

Si la corriente que fluye por el motor, a través de inducción genera un torque sobre el eje del motor, entonces podemos escribir ecuaciones de movimiento de la siguiente manera:

$$J\frac{d\omega}{dt} + k_f\omega = \tau_m + \tau_l \tag{6.32}$$

En esta ecuación $\omega$ es la velocidad del motor, $J$ y $K_f$ son el momento de inercia y el coeficiente de fricción viscosa, $\tau_m$ es el torque que genera el motor y $\tau_l$ es la carga que afecta al motor, lo que se pretende mover.

Como el torque producido por el motor, depende de la corriente que consume el mismo.

$$\tau_m = k_T i \tag{6.33}$$

Donde $k_T$ es conocida como constante de torque. Y la fuerza contra electromotriz, depende de la velocidad de giro del eje del motor.

$$V_c = k_C\omega \tag{6.34}$$

De donde $k_C$ es la constante de fuerza contra electromotriz. Si se sustituye estas ecuaciones que relacionan variables eléctricas y mecánicas en las ecuaciones diferenciales, estamos combinando ambos sistemas: mecánico y eléctrico en un solo. El modelo quedaría de la siguiente manera.

$$L\frac{di}{dt} + Ri + k_C\omega = V_{in} \tag{6.35}$$

$$J\frac{d\omega}{dt} + k_f\omega = k_T i + \tau_l \tag{6.36}$$

Una vez obtenida la ecuación, existen incógnitas a resolver, por ejemplo, cómo saber cuál es el valor de $R$, $L$, $k_f$, $k_C$, $J$ del motor.

Si bien, la hoja de datos del fabricante contiene estos valores, este método no suele resultar muy favorable, ya que los valores divergen bastante del valor real del motor. Otra opción es usar herramientas de identificación de parámetros a través de métodos experimentales, pero al fin de cuentas, suele ser muy engorroso y tampoco suele arroja los valores reales del motor, ya que hay un conjunto infinito de soluciones.

Pero hay un método más sencillo y que no se necesita conocer los valores del motor, trasladando al dominio de S por medio de la transformada de Laplace.

Si aplicamos la transformada al sistema para encontrar la función de transferencia entre voltaje y salida, llegamos al siguiente sistema.

$$\frac{\Omega(s)}{V(s)} = \frac{k_T}{(Ls+\ )\left(Js+k_f\right)+k_T k_C} \tag{6.37}$$

Debido a la construcción de los motores de corriente continua, la inductancia $L$ es mucho menor que la resistencia $R$, entonces el termino $L/R$ se puede despreciar. De esta manera, si despreciamos este factor el sistema se convierte, de uno segundo orden, a uno de primer orden. Entonces, la función de transferencia toma la siguiente forma.

$$\frac{\Omega(s)}{V(s)} = \frac{\frac{k_T}{R}}{Js+\left(k_f+\frac{k_T k_C}{R}\right)} \tag{6.38}$$

El sistema tiene forma de filtro pasabajos, y se puede representar de la forma:

$$\frac{\Omega(s)}{V(s)} = \frac{K}{\tau s+1} \tag{6.39}$$

Donde:

$$K = \frac{\omega_{max}}{V_{in}} \tag{6.40}$$

De esta manera, nos olvidamos de $L, R, J.$ y esas cosas que son complicadas de medir directamente. Sólo queda determinar la denominada ganancia $K$ y la constante de

tiempo del sistema $\tau$. Estas se obtienen fácilmente a partir de la respuesta del motor a una entrada constante de voltaje.

### 6.4.2 Retardo puro

Como consecuencia de la inercia de los elementos de almacenamiento de energía que hay en los sistemas, suelen aparecer retardos netos de tiempo entre la señal de salida respecto a la excitación de entrada. Hay muchos modelos de plantas que usan retardos puros en su función de transferencia. Así, por ejemplo, en la propuesta de Ziegler-Nichols para sistemas que, ante una entrada en escalón, su salida se parezca a un sistema sobreamortiguado, el modelo del sistema se configura con un retardo en la transmisión más un polo de primer orden.

$$G_p(s) \approx e^{-sT_d} \frac{K}{\tau s+} \tag{6.41}$$

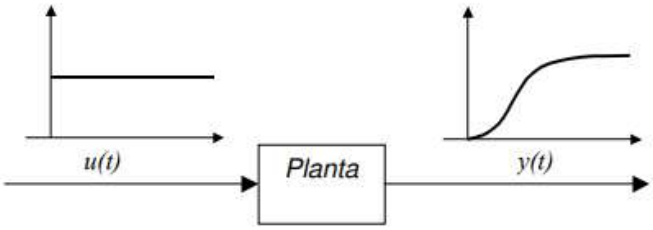



***Figura 6.22.-*** *Respuesta de una planta frente a una entrada unitaria. UNEFA [8]*

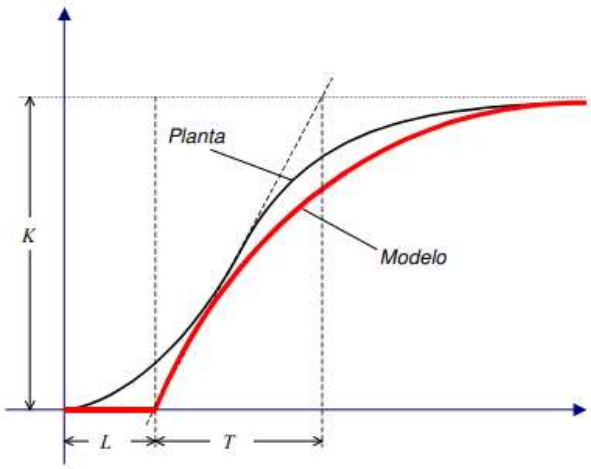



***Figura 6.23.-*** *Diferencia entre la planta y el modelo. UNEFA [8]*

El término $e^{-sT_d}$, por el teorema de traslación temporal de las transformadas de Laplace, corresponde con un retardo puro de tiempo, siendo $T_d$ el tiempo de ese retraso. Este concepto es empleado cuando se quiere expresar analíticamente un desfase de tiempo en la propagación de la señal al pasar por el sistema. Esta expresión es no lineal y hace que la relación causa efecto no se pueda formalizar en una función de transferencia de polinomios de coeficientes constantes. Por tanto, no es del tipo linealmente invariable en el tiempo. Sin embargo, existen varias aproximaciones de la función exponencial en términos lineales e invariante con el tiempo. Para valores pequeños de tiempo de retardo se puede aproximar a un polo de primer orden.

$$e^{-sT_d} \cong \frac{1}{1+T_d s} \tag{6.42}$$

Aunque es más correcto mediante la aproximación de Henri Padé.

$$e^{-sT_d} \simeq \frac{1-T_d\frac{s}{2}}{1+T_d\frac{s}{2}} \tag{6.43}$$

La ecuación final para modelar queda de la siguiente manera:

$$G_p(s) = \frac{1-T_d\frac{s}{2}}{1+T_d\frac{s}{2}} \; x \; \frac{K}{\tau s+1} \tag{6.44}$$

Para una ganancia del sistema $K$ igual a:

$$K = \frac{Señal\ de\ salida}{Señal\ de\ entrada} \tag{6.45}$$

## 6.5 Multiplexación

### 6.5.1 Funcionamiento

Para entender más fácilmente, de la *figura 6.26*, un multiplexor es un circuito combinacional digital que selecciona una de entre varias entradas ($I_x$) de datos y lleva su valor lógico a la única salida del circuito. La selección de los datos se realiza mediante una o varias entradas de control. La cantidad de líneas de control que debe de tener el multiplexor depende del número de canales de entrada. En este caso, se utiliza la siguiente fórmula:

Número de canales de entrada = $2^n$.

De donde n es el número de líneas de selección.

La codificación binaria resultante de las entradas ($S_x$) indica el índice de la entrada que pasa a la salida.

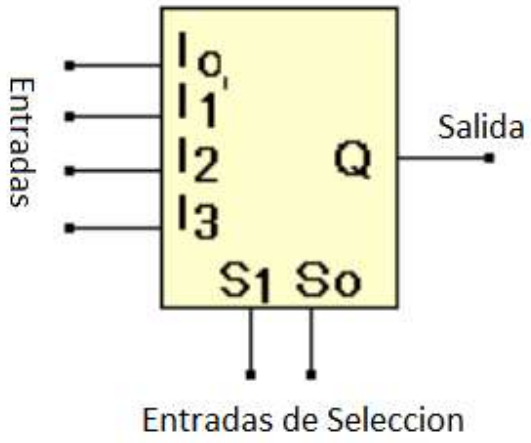



***Figura 6.24.-*** *Multiplexor de cuatro entradas. Fuente: Blog Electrónica unicrom. [5].*

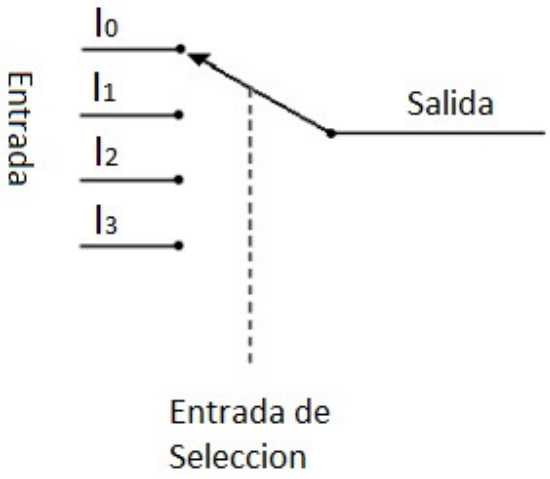



***Figura 6.25.-*** *Representación tipo switching de un multiplexor de cuatro entradas. Fuente: Propia.*

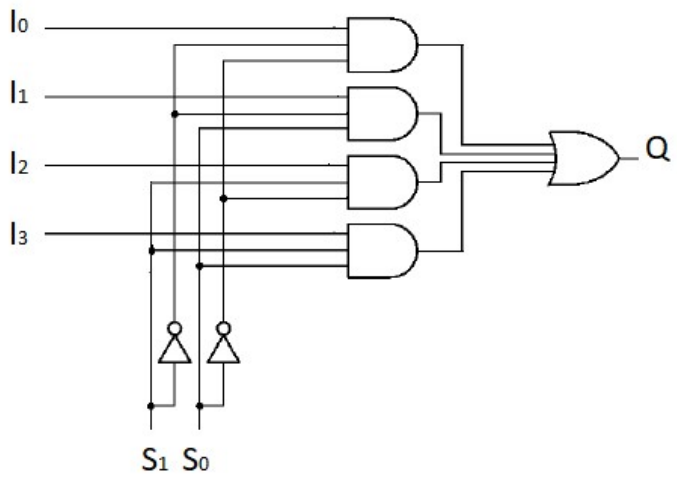



***Figura 6.26.-*** *Representación electrónica de un multiplexor de cuatro entradas con compuertas. Fuente: Propia.*

La tecnología utilizada para su diseño es TTL, de alta integración, y la potencia que disipan suele ser de unos 150 mW. El tiempo de retardo típico es de unos 25 nanosegundos. Normalmente, estos circuitos suelen darnos dos tipos de salida: un nivel alto y el nivel bajo.

### 6.5.2 Multiplexor I2C

Cuando se desea trabajar con dos o más dispositivos cuya comunicación es a través del protocolo de I2C, y a su vez estos poseen una dirección fija que no se puede cambiar debido a su construcción, existe este módulo que permite trabajar con ambos al mismo tiempo a través de sus pines SDA / SCL evitando el conflicto de datos entre estos dispositivos.

El multiplexor en sí está en la dirección I2C 0x70, aunque puede ajustarse de 0x70 a 0x77 y se escribe un solo byte con el número de salida multiplexado deseado para ese puerto, y luego cualquier paquete I2C futuro será enviado a ese puerto. Se podría tener hasta ocho de estos multiplexores en cada una de las direcciones 0x70-0x77 para controlar sesenta y cuatro de la misma parte dirigida por I2C.

La forma de obtener hasta ocho dispositivos I2C de la misma dirección conectados a un microcontrolador, es a través del controlador de acceso y éste transfiere los comandos al conjunto seleccionado de pines I2C con su comando. Solo se puede seleccionar un canal SCx / SDx a la vez, determinado por el contenido del registro de control programable (ver *figura 6.27 y figura 6.28*).

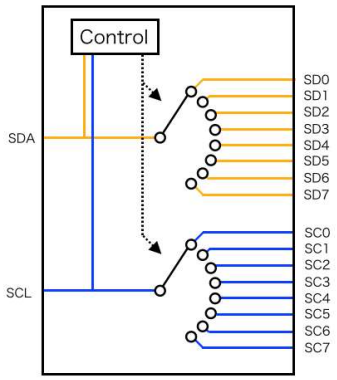



***Figura 6.27.-*** *Representación de multiplexor de I2C. Fuente: mbed [6].*

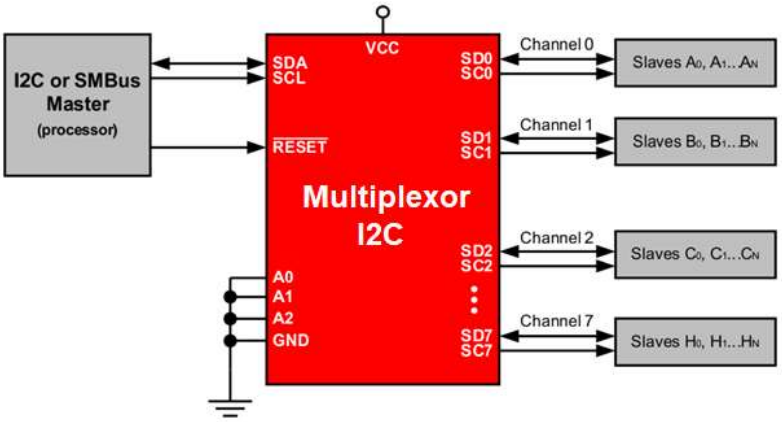



***Figura 6.28.-*** *Aplicación típica de multiplexor de I2C. Fuente: Datasheet Texas Instruments [7].*

Aplicaciones:

- Servidores.
- Routers.
- Instalaciones fabriles.
- Dispositivos esclavos con conflicto de dirección.
- Entre otros

## 6.6 Modelo Experimental Anexo II

En un principio, lo que se buscó en este proyecto fue realizar un control angular sobre un Moto-reductor. Éste contenía el ala acoplada en el eje, y de esta manera se podía controlar el movimiento del ala del animatrónico mediante el giro del moto-reductor. Para lograr el control de este, es necesario contar con un modelo matemático que describa el comportamiento del moto-reductor. En este anexo se describen algunos intentos llevados a cabo para obtener el modelo.

### 6.6.1 Descripción general

Siempre se buscó modelar todo el conjunto para poder aplicar los controladores estudiados, por lo que el sistema que se analizó está formado por el Sensor 1 (IMU MPU6050), Sensor 2 (IMU MPU6050), un microcontrolador, un puente H como driver y por último el moto-reductor IGNIS M6.

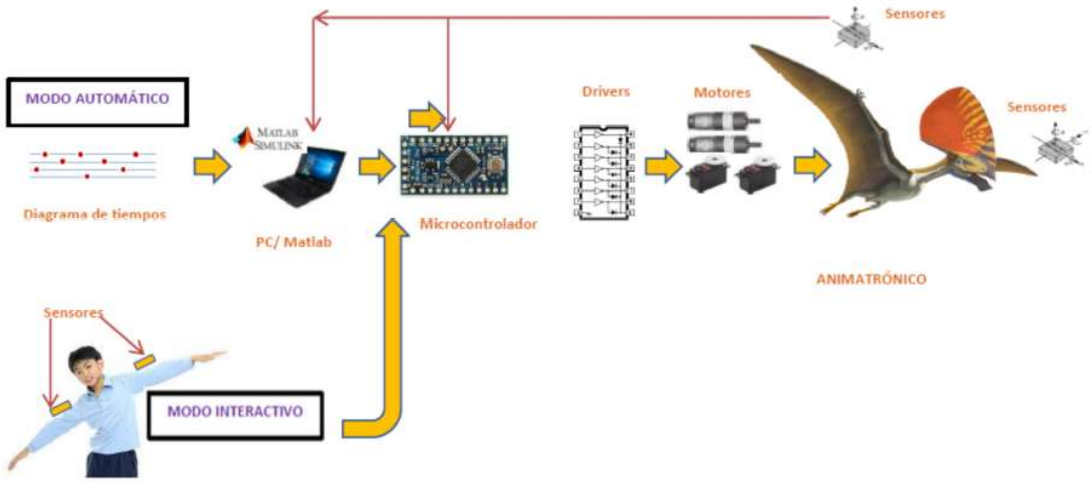



***Figura 6.29.-*** *Diagrama en bloque del sistema para modelar el motorreductor. Fuente: Propia.*

Una vez formado todo el sistema se diseñaron varias piezas para imprimir en 3D, algunos para soportar a los motores y eliminar cualquier vibración, y otros para acoplar los sensores al eje del motorreductor. En las siguientes imágenes se muestran los diseños.

### 6.6.2 Modelados del motorreductor

La primera forma de modelado que se llevó a cabo consistió en energizar al motor con una tensión conocida y analizar la respuesta mediante los oscilogramas. Lugo por una serie de aproximaciones se llegó a una función de transferencia de primer orden. A la hora de validar el modelo en Matlab los resultados no fueron los esperados.

La segunda forma fue utilizando una herramienta de Matlab que modela el contenido de una “caja negra” partiendo de los datos de entrada y salida. Para este caso la entrada fue la variación del escalón y la salida el ángulo del ala.

La tercera forma fue colocar el sistema completo en un ángulo inicial y aplicarle un escalón conocido para ver la variación y con esto aproximar a una función de segundo orden con retraso. Al igual que en el primer modelo los resultados no fueron validados con MatLab.

#### 6.6.2.1 Modelado ante una respuesta al escalón

Para la primera experiencia, se energizó al motorreductor a una tensión continua de +12v (tensión de trabajo) y se analizó en un oscilograma la respuesta de este. Analizando la *figura 6.30* se observa que se trata de un sistema de primer orden ya que no posee sobre picos ni oscilaciones. Esto tiene sentido ya que el IGNIS cuenta con una gran reducción de engranaje (ver Anexo II *punto 6.2.4 "engranaje tipo planetario"*).

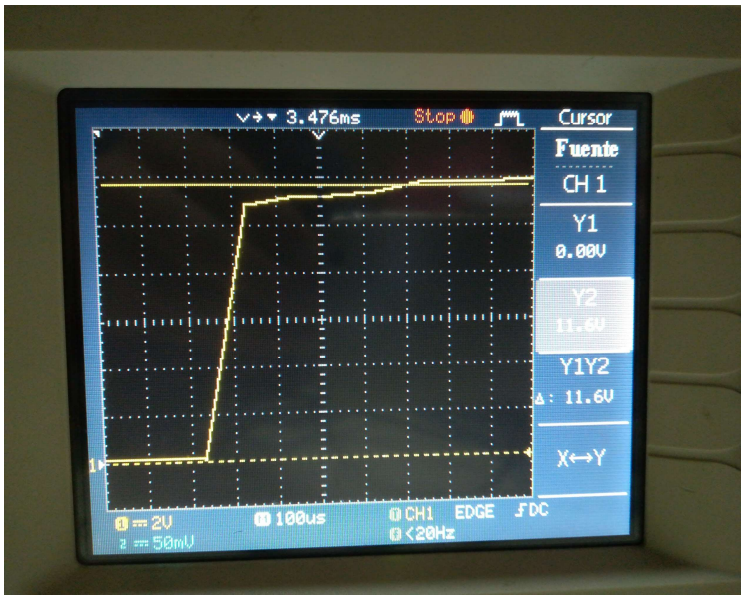



***Figura 6.30.-*** *Respuesta al escalón de tensión de un IGNIS. Fuente: Propia.*

De acuerdo con la característica de este motor, presenta un leve retardo debido al conjunto de engranajes y por este motivo a la hora de generar el modelo matemático, nos basamos en la ecuación (6.41) (sistema de primer orden con retardo).

Viendo la *figura 6.31* y *6.32*, se obtienen los siguientes datos sin tener en cuenta el retardo o delay (*ver tabla 6.4*).

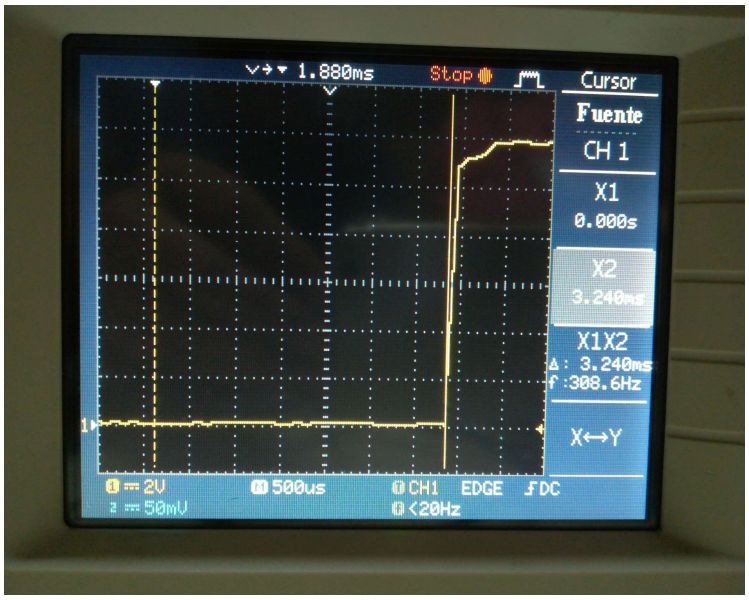



***Figura 6.31.-*** *Tiempo delay de arranque del IGNIS. Fuente: Propia.*

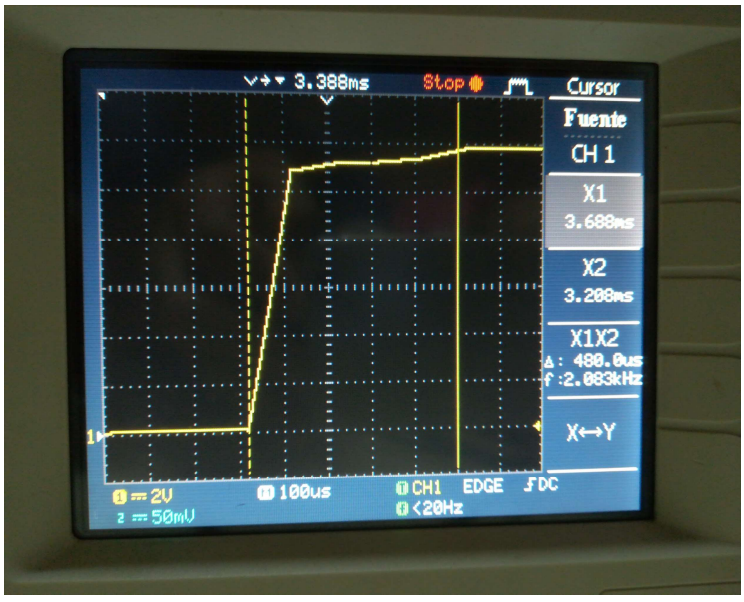



***Figura 6.32.-*** *Tiempo de establecimiento del IGNIS. Fuente: Propia.*

| | |
|---|---|
| Tensión Alimentación | 12,3 V |
| Tiempo Establecimiento (95%) | 480 µSeg |
| Tensión Establecimiento | 11,6 V |
| Tensión Tau (63%) | 7,44 V |
| Tiempo delay | 3,24 mSeg |

***Tabla 6.4.-*** *Respuesta al escalón de tensión de un IGNIS. Fuente: Propia.*

De la *figura 6.32* ingresando con el valor de tensión de Tau (7.44 V) corta la curva a un tiempo de 160 µSeg, y este es el valor del tiempo Tau. Utilizando la ecuación (6.44) y reemplazando obtenemos la función de transferencia, y reemplazando la ecuación (6.45) de la ganancia $K$ se tiene:

$$K = \frac{11.6\ V}{12.3\ V} = 0{,}943$$

$$G_p(s) = \frac{1 - 0.00324\frac{s}{2}}{1 + 0.00324\frac{s}{2}}\ x\ \frac{0{,}943}{1 + 0.00016\ s}$$

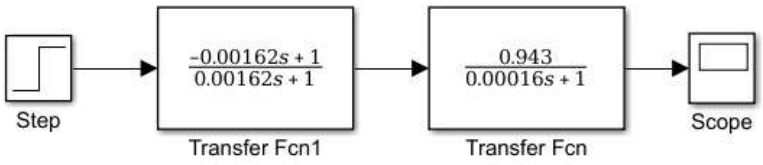



***Figura 6.33.-*** *Respuesta al escalón de tensión de un IGNIS. Fuente: Propia.*

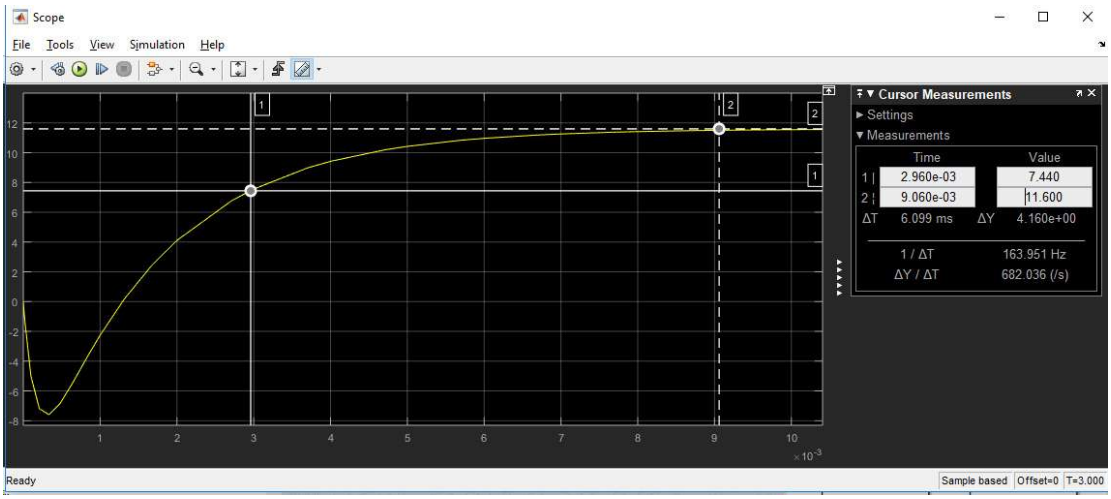



***Figura 6.34.-*** *Respuesta al escalón de tensión de un IGNIS. Fuente: Propia.*

#### 6.6.2.2 Modelado a través de una función de MatLab

MatLab cuenta dentro de su Toolbox con una herramienta llamada “Ident”, renombrada hace poco como “System Identification”. Esta función permite construir modelos matemáticos de sistemas dinámicos partiendo de un conjunto de datos entrada-salida ya sea que estén en el dominio del tiempo o de la frecuencia, estén en tiempo continuo o tiempo discreto, para obtener funciones de transferencia. A continuación, se detalla el proceso realizado y los resultados obtenidos con esta herramienta.

Con el sistema montado sobre un soporte mecánico, se procedió a conectar la MPU en el eje del motor (ver *figura 6.35*) de modo que se tuviera una lectura cada 5mSeg del ángulo que formaba el ala respecto a la horizontal.

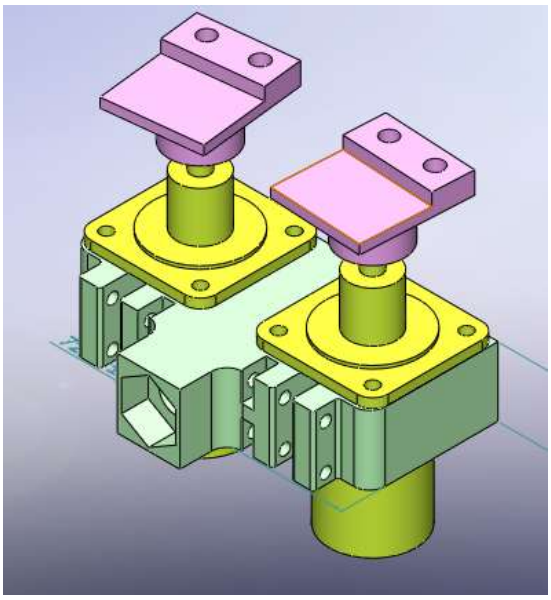


***Figura 6.35.-*** *Respuesta al escalón de tensión de un IGNIS. Fuente: Propia.*

El driver que maneja al moto-reductor fue excitado con un PWM conocido, de modo que posicione al ala a un ángulo determinado (para este caso -15°). Realizando lecturas de la IMU cada 5mSeg, se aplicó un escalón en el PWM que llevo al ala hasta 30°. Se registraron más de 1500 datos en una tabla y luego se graficaron (*figura 6.36*).

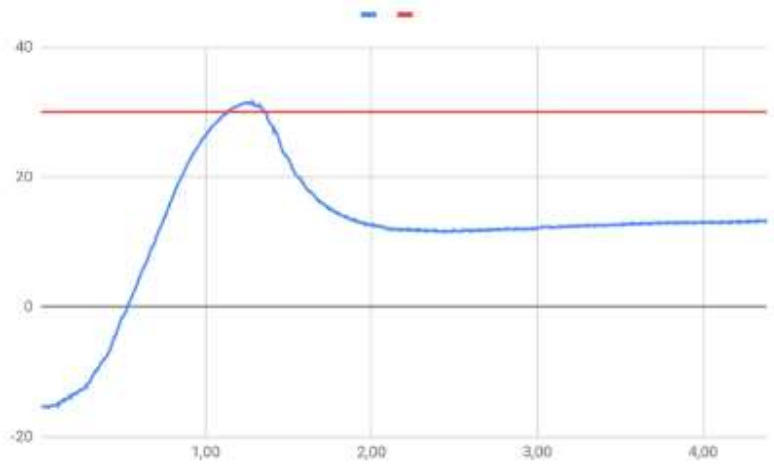



***Figura 6.36.-*** *Diagrama en bloque del sistema para modelar el motorreductor. Fuente: Propia.*

En color rojo se ve el escalón desde el tiempo cero cuando fue aplicado, y en color azul la respuesta del sistema medido. Estos datos se cargaron en dos vectores para procesarlos con la función System Identification de MatLab.

Para utilizar esta función, los datos fueron cargados en dos vectores (“Entada” y “Salida”) que se ingresaron como se ve en la siguiente imagen:

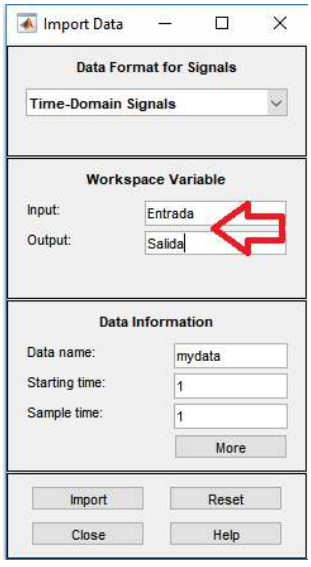



***Figura 6.37.-*** *Agregado de vectores “Entrada” y “Salida”. Fuente: Propia.*

Una vez que los datos fueron cargados, se pudo estimar la función de transferencia con “my data” que es el conjunto introducido anteriormente.

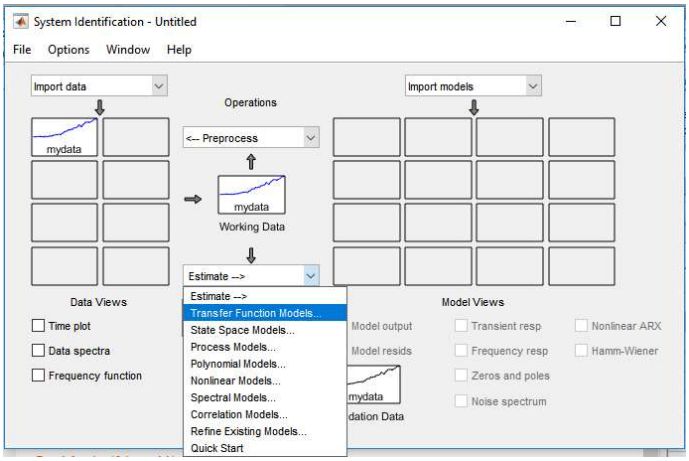



***Figura 6.38.-*** *Estimado de la función de transferencia por medio del System Identification. Fuente: Propia.*

Luego, de que el programa realizó varias iteraciones como se muestra en la *figura 6.39*, obtuvo un modelo con un determinado porcentaje de confianza y se pudo exportar la función de transferencia calculada.

```
Transfer Function Identification
Estimation data: Time domain data mydata
Data has 1 outputs, 1 inputs and 878 samples.
Number of poles: 2, Number of zeros: 0, I/O delay: 0.05
Initialization Method: "iv"
```

Estimation Progress

| Iteration | Cost | Norm of step | First-order optimality | Improvement (%) Expected | Achieved | Bisections |
|---|---|---|---|---|---|---|
| 0 | 16.603 | - | 1.33e+07 | 4.4 | - | - |
| 1 | 6.50614 | 1.19e+05 | 8.34e+06 | 4.4 | 60.8 | 0 |
| 2 | 5.41134 | 6.14e+04 | 3.94e+06 | 2 | 16.8 | 0 |
| 3 | 5.01774 | 2.61e+04 | 1.51e+06 | 0.931 | 7.27 | 0 |
| 4 | 4.89975 | 1.6e+04 | 9e+05 | 0.306 | 2.35 | 0 |
| 5 | 4.86018 | 9.36e+03 | 5.04e+05 | 0.107 | 0.808 | 0 |
| 6 | 4.84748 | 5.27e+03 | 2.79e+05 | 0.0347 | 0.261 | 0 |
| 7 | 4.8435 | 2.97e+03 | 1.52e+05 | 0.0109 | 0.0821 | 0 |
| 8 | 4.84228 | 1.65e+03 | 8.33e+04 | 0.00336 | 0.0252 | 0 |
| 9 | 4.84191 | 910 | 4.55e+04 | 0.00102 | 0.00766 | 0 |

```
Estimating parameter covariance...
done.
```

Result

```
Termination condition: Near (local) minimum, (norm(g) < tol).
Number of iterations: 9, Number of function evaluations: 19

Status: Estimated using TFEST
Fit to estimation data: 78.82%, FPE: 4.89737
```

***Figura 6.39.-*** *Parámetros de estimación del System Identification del MatLab. Fuente: Propia.*

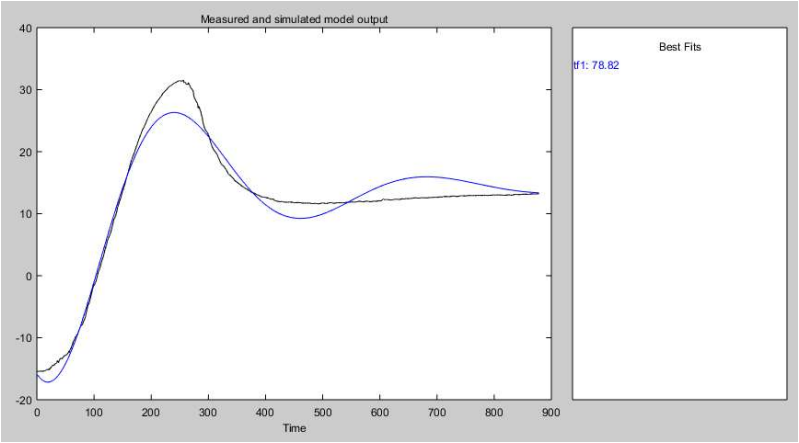



***Figura 6.40.-*** *Función de transferencia estimada. Fuente: Propia.*

En la *figura 6.40* se observa en color negro se ven datos obtenido a través de la IMU en el ensayo, y en color azul se muestra la respuesta obtenida por MatLab.

Para analizar la función de transferencia estimada, se generó en Simulink un modelo similar cuya excitación es igual a los ángulos de la entrada y se comparó con los datos obtenidos en el correspondiente Scope.

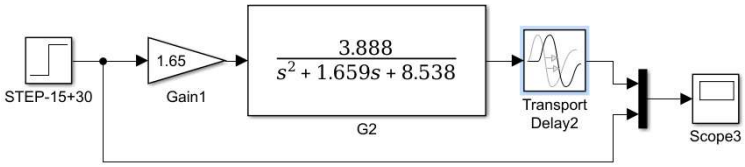



***Figura 6.41.-*** *Diagrama en bloque en Simulink para comparar la función de transferencia estimada. Fuente: Propia.*

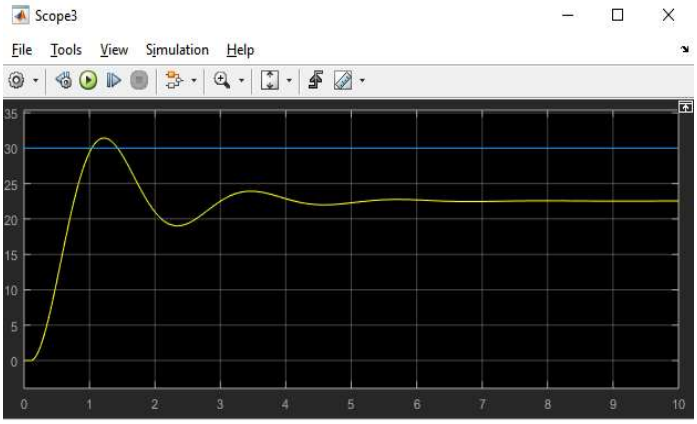



***Figura 6.42.-*** *Scope de la salida del diagrama en bloque de la función de transferencia estimada. Fuente: Propia.*

La imagen del Scope muestra que si bien la función tiene un comportamiento similar al que se midió en el sistema, los valores máximos y los tiempos no se corresponden con el grafico de la *figura 6.36*.

Luego de recopilar varias muestras con el sensor angular en el motor y plantear varias condiciones a la hora de medir, ninguna de las funciones de transferencia que se estimaron arrojó un resultado considerable y por esto no se pudo utilizar el modelado matemático que brinda MatLab.

#### 6.5.2.3 Modelado del sistema de segundo orden con retraso

Para realizar este proceso se ubicó al ala en un ángulo de -15° con respecto a la horizontal y se le aplicó un PWM al IGNIS tal que lo lleve a 30°, como en el apartado anterior, dando así el equivalente a una excitación del tipo escalón necesaria para plantear el modelo.

Para este caso se utilizó el modelo de una función de segundo orden con retraso y los cálculos se muestran a continuación:

$$G(s) = \frac{K\,{W_n}^2}{S^2 + 2\,\xi\,W_n\,S + {W_n}^2}\,e^{-Ts} \qquad (7.1)$$

$$\delta = \frac{\Delta y_{max} - \Delta y_{(\infty)}}{\Delta y_{(\infty)}} = e^{\frac{-\xi\pi}{\sqrt{1-\xi}}} \qquad (7.2)$$

$$t_p = \frac{t_{osc}}{2} \qquad (7.3)$$

Dónde:

$W_n =$ Frecuencia natural

$K =$ Ganancia del sistema

$\xi =$ Factor de amortiguamiento

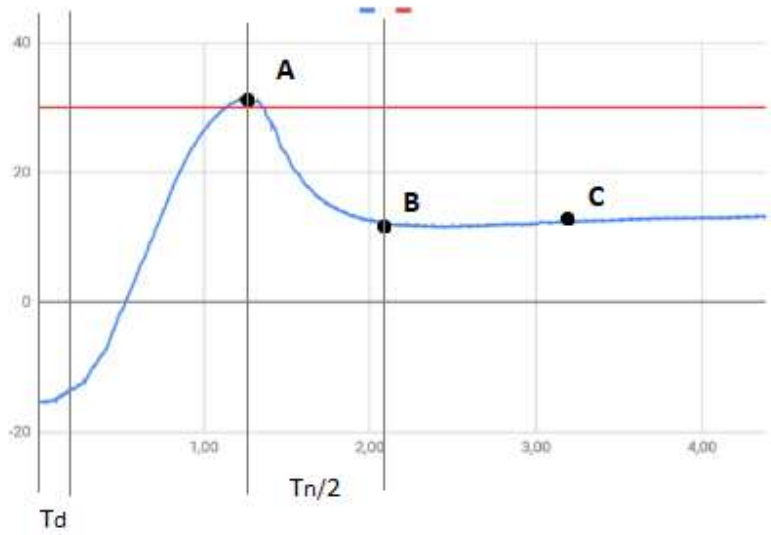



***Figura 6.43.-*** *Ensayo de la respuesta al escalón. Fuente: Propia.*

$$T_d = 0{,}05\ seg = 50\ mseg$$

$$A = (1{,}275\ seg; 31{,}58°)$$

$$B = (2{,}4\ seg; 11{,}58°)$$

$$C = (1{,}5\ seg; 13{,}3°)$$

$$\frac{T_n}{2} = T_B - T_A = 1{,}125\ seg$$

$$T_n = 2 \times 1{,}125\ seg = 2{,}25\ seg$$

$$W_n = \frac{2\pi}{T_n} = 2{,}7925268\ \left[\frac{rad}{seg}\right]$$

$$\delta = \frac{\Delta y_{max} - \Delta y_{(\infty)}}{\Delta y_{(\infty)}} = \frac{31{,}58° - 13{,}3°}{13{,}3°} = 1{,}3744$$

Y sabiendo que:

$$\delta = e^{\frac{-\xi}{\sqrt{1-\xi}}}$$

Podemos despejar a $\xi$:

$$\xi = 0{,}176676$$

También conocemos a la ganancia $K$:

$$K = \frac{y_{(\infty)}}{U} = \frac{13{,}3°}{30° - (-15°)} = 0{,}2955$$

Reemplazamos en la ecuación (7.1) obteniendo:

$$G(s) = \frac{2{,}3}{S^2 + 0{,}987S + 7{,}7982}\, e^{-0{,}05s}$$

Una vez que se llegó a la función de transferencia de manera práctica, se la modeló en Simulink para comparar la respuesta partiendo del mismo escalón (-15° a 30°).

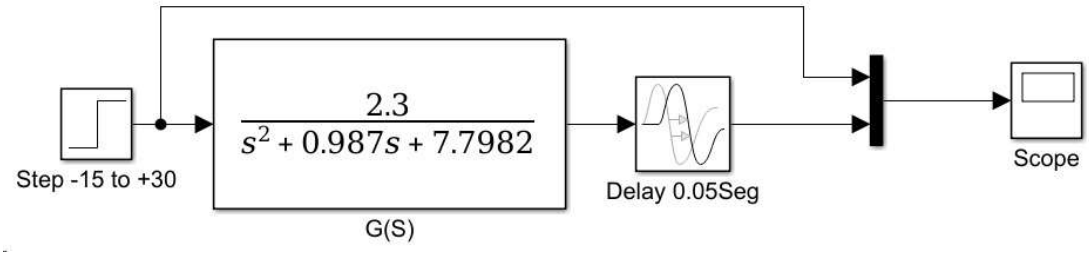



***Figura 6.44.-*** *Diagrama en bloque en Simulink para validar la función de transferencia calculada. Fuente: Propia.*

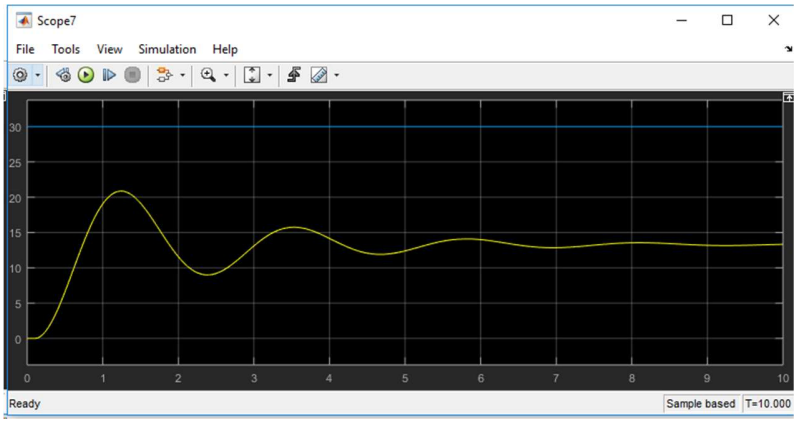



***Figura 6.45.-*** *Scope de la salida del diagrama en bloque de la función de transferencia calculada. Fuente: Propia.*

Como se observa en la *figura 6.45*, se puede confirmar que el modelo calculado presenta una respuesta distinta a la obtenida en la medicion del sistema que se realizo en el ensayo.

Luego de intentar sin éxito calcular la función de transferencia que representa al motorreductor, y con distintas técnicas con MatLab o con métodos gráficos, se consultó con profesores especialistas en el tema, que aportaron sus conocimientos y experiencias, y ayudaron a determinar que el sensor seleccionado no es el adecuado para realizar este tipo de tareas. Es decir, que la IMU (unidad de medida inercial) MPU6050 acoplada al eje del motor no es la técnica más efectiva, ya que es muy sensible a vibraciones que provocan lecturas erróneas y también hay una complejidad mecánica de alinear perfectamente los planos del sensor con el plano transversal al eje del motor, de modo que el ángulo que mide el sensor tiene una pequeña diferencia respecto al ángulo del IGNIS.

### 6.5.3 Pruebas de movimiento realizadas a un motorreductor

A continuación, se muestra una serie de códigos realizados para estudiar el movimiento del motorreductor junto con el giroscopio. De a poco, se fue progresando según el grado de complejidad y según la necesidad, aunque cabe aclarar que los mismos fueron programación de manera progresiva hasta llegar a un movimiento controlado del motorreductor. Dada esta imposibilidad de llegar a nuestro objetivo, queda constancia de los códigos realizados y de las conclusiones obtenida habiendo elegido este camino.

#### 6.5.3.1 Movimiento de un motorreductor

```
int IN3 = 5;    // Input3 conectada al pin 5
int IN4 = 4;    // Input4 conectada al pin 4
int ENB = 3;    // ENB conectada al pin 3 de Arduino
void setup()
```

```
{
 pinMode (ENB, OUTPUT);
 pinMode (IN3, OUTPUT);
 pinMode (IN4, OUTPUT);
}
void loop()
{
  //Preparamos la salida para que el motor gire en un sentido
  digitalWrite (IN3, HIGH);
  digitalWrite (IN4, LOW);
  // Aplicamos PWM al pin ENB, haciendo girar el motor, cada 2 seg aumenta la velocidad
  analogWrite(ENB,255);
  delay(175);

  digitalWrite (IN3, LOW);
  digitalWrite (IN4, HIGH);

  analogWrite(ENB,55);
  delay(1100);


  // Apagamos el motor y esperamos 5 seg
  analogWrite(ENB,0);
  delay(50);
}
```

#### 6.5.3.2 Movimiento en un sentido y en otro de un motorreductor según el movimiento de la IMU sin funciones

```
#include <Wire.h>
#include <I2Cdev.h>
#include <MPU6050.h>
#define TCAADDR 0x70
MPU6050 mpu;

int PWM = 0;//valor del pwm
int IN3 = 4;// IN2 VERDADERO. salida del driver
int IN4 = 5;// IN1 VERDADERO. salida del driver
int ENB = 3;// ENA VERDADERO. habilita el driver

int16_t ax, ay, az;
int16_t gx, gy, gz;

float EMA_ALPHA = 0.35;
int EMA_LP0 = 0;
int EMA_LP1 = 0;
```

```
int i = 1;
float dato = 0;
float dato_2 = 0;

float getMeasure()
{
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
  dato = map(ay, -17000, 17000, 0, 179);
  return dato;
}

void tcaselect(uint8_t i) {
if (i > 7) return;
Wire.beginTransmission(TCAADDR);
Wire.write(1 << i);
Wire.endTransmission();
}

void setup()
{

    Wire.begin();
   Serial.begin(115200);
   Serial.println("Initialize MPU");
   mpu.initialize();
   Serial.println(mpu.testConnection() ? "Connected" : "Connection failed");
}

void loop()
{

  tcaselect(0);
    float filteredLP0 = EMALowPassFilter0(getMeasure());
    delay(10);
  tcaselect(1);
    float filteredLP1 = EMALowPassFilter1(getMeasure());

if (filteredLP0 < filteredLP1)
   {
      Serial.println("debe subir");

      PWM=150;
      digitalWrite (IN3, HIGH);
      digitalWrite (IN4, LOW);
      analogWrite(ENB,PWM);

      if((0 < (filteredLP1 - filteredLP0)) && ((filteredLP1 - filteredLP0) < 6))
      {
```

```
        while((0 < (filteredLP1 - filteredLP0)) && ((filteredLP1 - filteredLP0) < 6))
        {
        delay(10);
        Serial.println("subio");

        PWM=0;
        analogWrite(ENB,PWM);

        tcaselect(0);
        float filteredLP0 = EMALowPassFilter0(getMeasure());
        delay(10);
        tcaselect(1);
        float filteredLP1 = EMALowPassFilter1(getMeasure());
        delay(10);

           if((7 < (filteredLP1 - filteredLP0)) || ((filteredLP0 - filteredLP1) > 7))
           {
             break;
           }

        }

      }

   }

else if(filteredLP0 > filteredLP1)
      {
       Serial.println("debe bajar");

       PWM=100;
       digitalWrite (IN3, LOW);
       digitalWrite (IN4, HIGH);
       analogWrite(ENB,PWM);

         if((0 < (filteredLP0 - filteredLP1)) && ((filteredLP0 - filteredLP1) < 6))
         {
            while((0 < (filteredLP0 - filteredLP1)) && ((filteredLP0 - filteredLP1) < 6))
            {
              delay(10);
                Serial.println("bajo");

                PWM=0;
                analogWrite(ENB,PWM);

                tcaselect(0);
                float filteredLP0 = EMALowPassFilter0(getMeasure());
                delay(10);
```

```
            tcaselect(1);
            float filteredLP1 = EMALowPassFilter1(getMeasure());
            delay(10);

             if((7 < (filteredLP0 - filteredLP1)) || ((filteredLP1 - filteredLP0) > 7))
             {
                break;
             }
          }
       }
    }
}



float EMALowPassFilter0(float dato_2)
{
  EMA_LP0 = EMA_ALPHA * dato_2 + (1 - EMA_ALPHA) * EMA_LP0;
  return EMA_LP0;
}

float EMALowPassFilter1(float dato_2)
{
  EMA_LP1 = EMA_ALPHA * dato_2 + (1 - EMA_ALPHA) * EMA_LP1;
  return EMA_LP1;
}
```

6.5.3.3 Movimiento en un sentido y en otro de un motorreductor según el movimiento de la IMU con funciones.

```
#include <Wire.h>
#include <I2Cdev.h>
#include <MPU6050.h>
#define TCAADDR 0x70
MPU6050 mpu;
int16_t ax, ay, az;
int16_t gx, gy, gz;

float EMA_ALPHA = 0.35;
int EMA_LP0 = 0;
int EMA_LP1 = 0;
int i = 1;
float dato_0 = 0;
float dato_1 = 0;


float leer_0 (void);
float leer_1 (void);

void tcaselect(uint8_t i)
{
   if (i > 7)
   return;
   Wire.beginTransmission(TCAADDR);
   Wire.write(1 << i);
   Wire.endTransmission();
}

void setup()
{

   Wire.begin();
   Serial.begin(38400);
   Serial.println("Initialize MPU");
   mpu.initialize();
   Serial.println(mpu.testConnection() ? "Connected" : "Connection failed");

}

void loop()
{

  leer_0();
delay(10);
```

```
  leer_1();
delay(10);
}

float leer_0(void)
{
   tcaselect(0);
   mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
   dato_0 = map(ay, -17000, 17000, 0, 179);
   EMA_LP0 = EMA_ALPHA * dato_0 + (1 - EMA_ALPHA) * EMA_LP0;
   Serial.print("   ");
   Serial.print(dato_0);
   Serial.print(",");
   Serial.print(EMA_LP0);

}

float leer_1(void)
{
   tcaselect(1);
   mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
   dato_1 = map(ay, -17000, 17000, 0, 179);
   EMA_LP1 = EMA_ALPHA * dato_1 + (1 - EMA_ALPHA) * EMA_LP1;
   Serial.print("      |      ");
   Serial.print(dato_1);
   Serial.print(",");
   Serial.println(EMA_LP1);

}
```

#### 6.5.3.4 Movimiento automático de un motorreductor

```
#include <Wire.h>
#include <I2Cdev.h>
#include <MPU6050.h>
MPU6050 mpu;
int16_t ax, ay, az;
int16_t gx, gy, gz;

float EMA_ALPHA = 0.6;// constante del filtro alpha
int EMA_LP = 0; // valor filtrado
float dato;// valor del mapeo
int PWM = 80;//valor del pwm
int IN3 = 5;// salida del driver
int IN4 = 4;// salida del driver
int ENB = 3;// habilita el driver
```

```
float leer (void);// Esta funcion lee el estado del sensor IMU, pasado por el filtro complementario
void subir (void);//Esta funcion acciona el motor de modo que el bazo del robot suba
void bajar (void);// Esta funcion acciona el motor de modo que el brazo del robot baje
void parar (void);// Esta funcion detiene el motor de modo que el brazo del robot baje
void iniciar (float);// Esta funcion coloca al ala en la posicion inicial

bool parada=true;  //Es la variable que detiene todo el programa cuando esta en false
float angulo;     //Es la variable que contiene el angulo proporcionado por el sensor
float ang_inicial=45;  //Es el angulo donde se posiciona al inicio para comenzar la rutina

//variables para el control del motor DC



void setup()
{
   Wire.begin();
   Serial.begin(9600);
   Serial.println("Initialize MPU");
   mpu.initialize();
   Serial.println(mpu.testConnection() ? "Connected" : "Connection failed");
}

void loop() {
  iniciar(leer());
  delay(3000); //tiempo para asegurar la posicion inicial del ala.
  while (parada)
  {
    angulo=leer(); //tomo una lectura del ángulo
//si el angulo supera loa 120 grados debe subir

   parar(); //
   delay(250);
   while (angulo<120)
   {
    bajar();
    angulo=leer();

   }

//Si el angulo alcanza los 30 grados debe bajar
   while (angulo>30)
   {
    subir();
    angulo=leer();
   }
  }
  }
```

```
//En esta funcion se lee el sensor y se filtra el valor para asegurar un dato confiable
 float leer (void)
{
   mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
   dato = map(ay, -17000, 17000, 0, 179);   //mapeo del valor de la IMU  entre 0 y 180
   EMA_LP = EMA_ALPHA * dato + (1 - EMA_ALPHA) * EMA_LP;    // valor filtrado del mapeo
   Serial.print(dato);
   Serial.print("  ,   ");
   Serial.println(EMA_LP);
   delay (1);       //para dar tiempo al serial print
   return(EMA_LP);
}

//Esta funcion se encarga de llevar al ala a la posicion inicial,
//Mide el angulo del ala y la deja en el angulo ingresado ang_inicial
void iniciar(float valor)
{
 bool inicio = false; //variable de control para salir de la funcion
 //codigo de la funcion
 while(!inicio) // queda encerrado en el while hasta que inicio=true
   {

   valor=leer();
   Serial.print("iniciando");
   Serial.print("   ");
   Serial.println(valor);

  if (valor<ang_inicial)
  {
   //Si el valor es menor al angulo gira en un sentido
  // Serial.print("Girando_Horario");
   PWM=80;
   digitalWrite (IN3, LOW);
   digitalWrite (IN4, HIGH);
   analogWrite(ENB,PWM);
  }
  if (valor>ang_inicial)
  {
   //Sino gira en el otro sentido
  // Serial.print("Girando_Antihorario");
   PWM=255;
   digitalWrite (IN3, HIGH);
   digitalWrite (IN4, LOW);
   analogWrite(ENB,PWM);
  }
  if (valor >(ang_inicial-8)&& valor<(ang_inicial+2))
   {
```

```
      inicio=true;
      analogWrite(ENB,0);
    }
  }
}

void subir(void)
{
  Serial.print(leer());
  Serial.println("subir   ");

    PWM=200;
    digitalWrite (IN3, HIGH);
    digitalWrite (IN4, LOW);
    analogWrite(ENB,PWM);
}

void bajar(void)
{
  Serial.print(leer());
  Serial.println("bajar   ");
    PWM=100;
    digitalWrite (IN3, LOW);
    digitalWrite (IN4, HIGH);
    analogWrite(ENB,PWM);
}
void parar (void)
{
   PWM=0;
    analogWrite(ENB,PWM);

}
```

#### 6.5.3.5 Movimiento de un motorreductor controlado proporcionalmente.

```
#include <TimerOne.h>
#include <Wire.h>

//Direccion I2C de la IMU
#define MPU 0x68
//Ratios de conversion
#define A_R 16384.0
#define G_R 131.0
//Conversion de radianes a grados 180/PI
#define RAD_A_DEG = 57.295779
//MPU-6050 da los valores en enteros de 16 bits
//Valores sin refinar
```

```
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;
//Angulos
float Acc[2];
float Gy[2];
float Angle[2];


//Flags de control
bool fLeer = false;
bool fSubir = false;
bool fBajar = false;
bool fPID = false;

//Variables para el PID
float sp = -15;   //SetPoint Angulo deseado
//float ei = 0;   //Error actual
float eiAnterior = 0;
double kp = 12.5, ki = 2 , kd = 0;   //0.5 1.3 0.0
double ei, termI, termD;

//Config Driver
int IN3 = 3;// salida del driver
int IN4 = 4;// salida del driver
int ENB = 6;// habilita el driver
int PWM; // Valor del pwm
int PWMmax = 255;
int PWMmin = 70;

//Auxiliares
int muestra = 0;
int tin, tfin, t;

void setup()
{
  pinMode(2, INPUT_PULLUP); // Declaro entrada D2 con pullUp para interrupcion externa.
  pinMode(13, OUTPUT);
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(115200);
  attachInterrupt( 0, ServicioBoton, FALLING); // Confiuro como entrada D2 para que interrumpa
  Timer1.initialize(5000);//Timer1.initialize(5000) 5mseg
  Timer1.attachInterrupt(intPorTimer); // Activa la interrupcion y la asocia a intPorTimer
}
void loop()
{
```

```
  if (fLeer) leer(); //Si el flag "leer" está en alto voy a la función leer
  if (fSubir) subir();//Si el flag "subir" está en alto voy a la función subir
  if (fBajar) bajar();//Si el flag "bajar" está en alto voy a la función bajar
  if (fPID) actualizarPID();//Si el flag "PID" está en alto voy a la funcion actualizarPID
}

void intPorTimer(void)
{
  fPID = true;
  fLeer = true;
}
void ServicioBoton (void)
{
  sp = 30;
  Serial.println("IIIIIIIAAAAAA");
  detachInterrupt(0);
}

void leer(void) {
  muestra++;
  tin;
  //Leer los valores del Acelerometro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //A partir del 0x3B, se piden 6 registros
  AcX = Wire.read() << 8 | Wire.read(); //Cada valor ocupa 2 registros
  AcY = Wire.read() << 8 | Wire.read();
  AcZ = Wire.read() << 8 | Wire.read();

  //A partir de los valores del acelerometro, se calculan los angulos Y, X
  //respectivamente, con la formula de la tangente.
  // Acc[1] = atan(-1 * (AcX / A_R) / sqrt(pow((AcY / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;
  Acc[0] = atan((AcY / A_R) / sqrt(pow((AcX / A_R), 2) + pow((AcZ / A_R), 2))) * RAD_TO_DEG;

  //Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 4, true); //A diferencia del Acelerometro, solo se piden 4 registros
  GyX = Wire.read() << 8 | Wire.read();
  // GyY = Wire.read() << 8 | Wire.read();

  //Calculo del angulo del Giroscopio
  Gy[0] = GyX / G_R;
  //Gy[1] = GyY / G_R;

  //Aplicar el Filtro Complementario
```

```
 tfin = micros();
 t = tfin - tin;
 Angle[0] = 0.98 * (Angle[0] + Gy[0] * t / 1000000) + 0.02 * Acc[0];
 // Angle[1] = 0.98 * (Angle[1] + Gy[1] * 0.010) + 0.02 * Acc[1];
 tin = micros();


 //Mostrar los valores por consola
 // Serial.print("Angle X: ");
 // Serial.print(Angle[0]); Serial.print("\n");
 //  Serial.print("Angle Y: "); Serial.print(Angle[1]); Serial.print("\n----------\n");
 fLeer = false;
}
void subir(void)
{
 fSubir = false;
 // Serial.println("subir");
 digitalWrite (IN3, HIGH);
 digitalWrite (IN4, LOW);
 analogWrite (ENB, PWM);
}

void bajar(void)
{
 fBajar = false;
 // Serial.println("bajar");
 digitalWrite (IN3, LOW);
 digitalWrite (IN4, HIGH);
 analogWrite (ENB, PWM);
}
void actualizarPID(void)
{

 ei = sp - Angle[0]; //Guardo el error de posición
 termI = termI + ei;
 if (termI >= 0 )
 {
  termI = constrain (termI, 5, 60);
 }
 else
 {
  termI = -constrain (termI, 5, 30);
 }
 termD = (ei - eiAnterior) / 0.005;
 eiAnterior = ei;
 PWM = ei * kp + termI * ki + termD * kd;
 //Calculo el pwm de forma lineal
 //PWM = map(abs(ei), 0 , 45, 65, PWMmax);
```

```
 PWM = constrain(PWM, PWMmin, PWMmax);

 ////////////////////////////////////////////////
 //Actualizo los flags de acuerdo al error
 if (ei < 0) ///   Si ei menor que cero tengo que subir y desactivo el fBajar por las dudas
 {
  fBajar = false;
  fSubir = true;
 }
 if (ei > 0)///   Si ei mayou que cero tengo que bajar y desactivo el fSubir por las dudas
 {
  fSubir = false;
  fBajar = true;
 }
 if (ei == 0) // queda quieto y desactivo los flags
 { fSubir = false; fBajar = false; analogWrite(ENB, 0); Serial.println("LLEGOOO");
 }
 ////////////////////////////////////////////////////
 Serial.print(ei);  Serial.print("      ");    Serial.print  (sp);  Serial.print("     ");  Serial.print(Angle[0]);
/*Serial.print("  "); Serial.print(PWM);*/ Serial.print("\n");
 fPID = false;
}
```

# Anexo III

En este capítulo, se exponen las hojas de datos de elementos que se utilizaron en el Proyecto.

## 7.1 Dimensiones cuerpo torácico del animatrónico

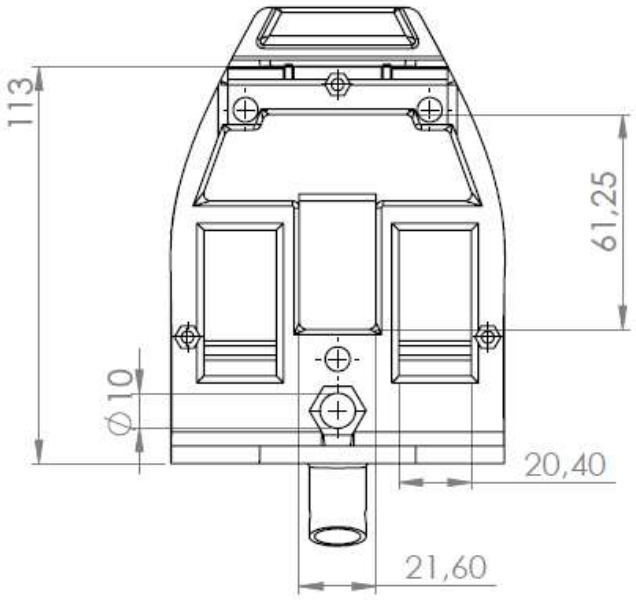



***Figura 7.1.-*** *Dimensiones y vista superior tórax animatrónico. Fuente: Propia.*

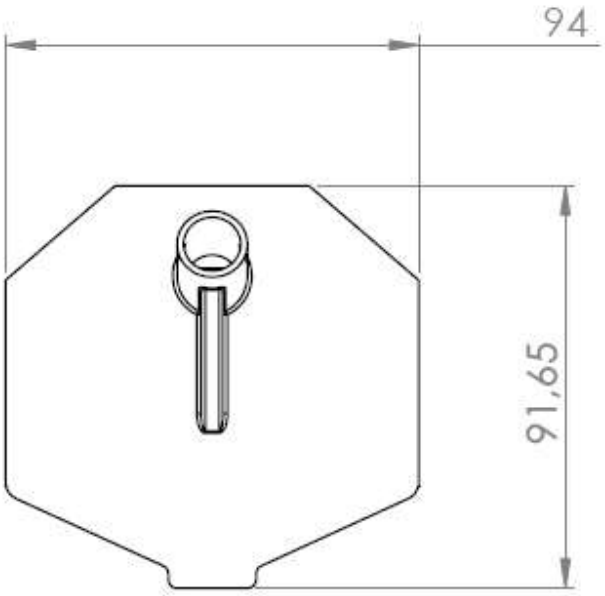



***Figura 7.2.-*** *Dimensiones y vista frontal del tórax animatrónico. Fuente: Propia.*

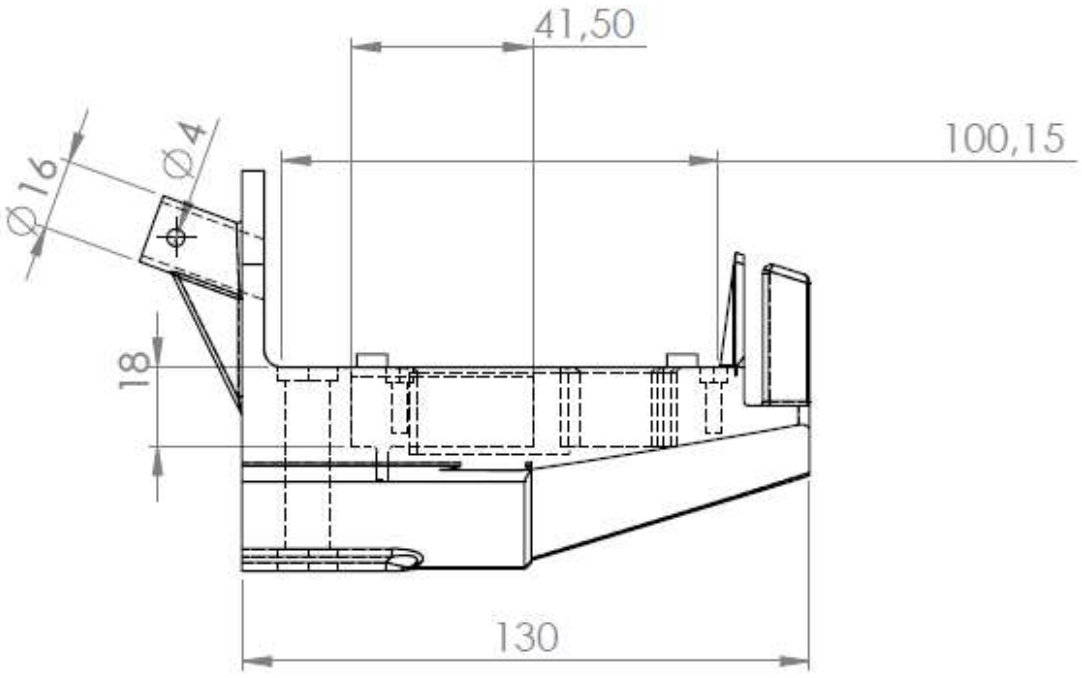



***Figura 7.3.-*** *Dimensiones y vista lateral de tórax animatrónico. Fuente: Propia.*

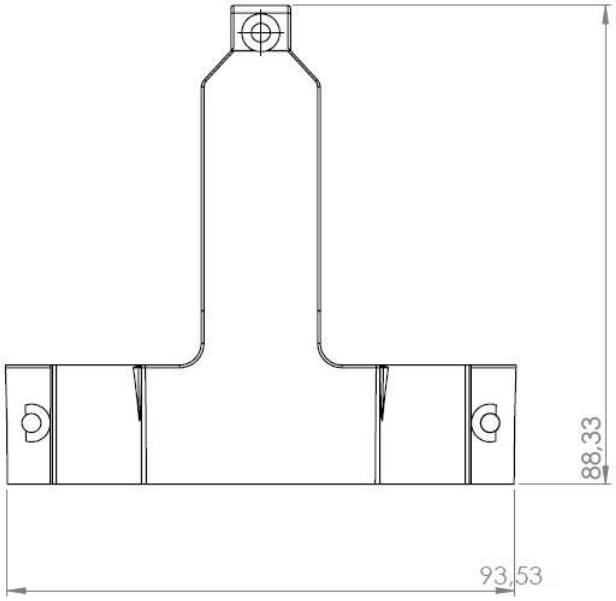



***Figura 7.4.-*** *Dimensiones y vista superior del complemento torácico. Fuente: Propia.*

## 7.2 Hoja de datos regulador de tensión LM2577

http://www.ti.com/lit/ds/symlink/lm2577.pdf

## 7.3 Hoja de datos diodo Schottky 1N5820

https://www.vishay.com/docs/88526/1n5820.pdf

## 7.4 Hoja de datos chip ATMega 328P

https://www.sparkfun.com/datasheets/Components/SMD/ATMega328.pdf

## 7.5 Hoja de datos IMU MPU6050

https://store.invensense.com/datasheets/invensense/MPU-6050_DataSheet_V3%204.pdf

## 7.6 Hoja de datos Servomotor MG996R

https://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf

## 7.7 Hoja de datos Modulo bluetooth HC-05

https://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/CSR-BC417-datasheet.pdf

# Anexo del Proyecto Integrador

*Facultad de Ciencias Exactas, Físicas y Naturales*
**AREA INGENIERÍA**
**ESCUELA DE ELECTRONICA**
**C.C. 755 - Correo Central - 5000 - CÓRDOBA**
**Tel. Directo (0351) 33-4147 int 110**
**Conmutador: 433-4141 y 33-4152 - Interno 10**

Sr. Director de la Escuela de Ingeniería Electrónica
Ing.: RODRIGO BRUNI

Me dirijo a Ud. a fin de solicitar la **aprobación del tema del *Proyecto Integrador*** (***PI***) que propongo a continuación:

**TEMA**

**NOMBRE DEL PROYECTO**: Control de un Robot Animatrónico tipo reptil volador

**DESCRIPCION**: Ver Anexo I

**DESARROLLO DE PROTOTIPO**: Se desarrollaráun prototipo.

**AREA TEMATICA DEL PI**: Robótica y Animatrónica.

**ASIGNATURAS**: Robótica y Animatrónica, Sistemas de Control I y Electrónica Digital III,

**Director de PI**

**Nombre**: Ing. Hugo Pailos
**Cargo**: Profesor Adjunto Robótica y Animatrónica
**Dirección Personal o Laboral**: Av. Velez Sarsfield 1611 – C.P. X5000EAS – FCEFyN - UNC
**TE**: 4334147 int. 114
**Email:** hpailos@hotmail.com
**Firma del Director / Co Director**: .....

**Datos del Estudiante**

**Nombre y Apellido:** Moreno Agustín Ezequiel
**Matrícula**: 36480216
**Materias que faltan aprobar**: 1
**Dirección**: San Lorenzo 213, Nueva Córdoba
**Localidad**: Córdoba Capital **Provincia**: Córdoba
e-**mail**: agumoreno12@gmail.com
**Teléfono:** 3571-570172

**Firma**:.....

**Datos del Estudiante**

**Nombre y Apellido:** Frenguelli Gino
**Matrícula**: 34501846
**Materias que faltan aprobar**: 2
**Dirección**: Av Velez Sarfield 1358, Guemes.
**Localidad**: Córdoba Capital **Provincia**: Córdoba
e-**mail**: gino_f34 @hotmail.com
**Teléfono:** 3571-552314

**Firma**:.....

**Objetivo General:** Se pretende rediseñar reconstruir y controlar un robot tipo reptil volador que representa a una especie Tapejara, ahora renombrado por los científicos con el nombre de “Tupandactylus” (Tapejara Imperator fue un reptil volador que vivió en Brasil hace 108 millones de años ) usando tecnología completamente nueva. Como objetivo y criterio de diseño se busca que este animatrónico pueda tener movimientos de cabeza (arriba y abajo) pico (abre y cierra) y de alas (aleteo) con determinada naturalidad y ser controlado de diferentes modos, ya que la finalidad es la exposición en parques y museos temáticos. El robot contará con un modo “rutina” en el cual repetirá una serie de movimientos cada un cierto tiempo, y un modo “controlado” en el cual una persona del público controlará mediante un “dispositivo interactivo” la posición de las alas. Cabe acotar que todo el mecanismo permitirá soportar la morfología de la especie citada en el presente trabajo o en un futuro próximo, siempre cuidando el objetivo del actual proyecto que es el control del animatrónico en sus diversas formas.

**Objetivos Específicos**

- Estabilizar el sistema de lazo cerrado en una sola ala, de modo que el ala tome una serie de ángulos estables provistos por un procesador.
- Probar el sistema en el que los ángulos de entrada, son provistos por medio de un dispositivo que controla el usuario (IMU) y chequear que los ángulos de salida del ala sean respetados.
- Montar todo el sistema (alas y cabeza) en un bastidor cuya morfología sea la de la especie en estudio.
- Comprobar el funcionamiento de todo es sistema de control descripto en el ANEXO I

**Antecedentes de Proyectos similares**: En este laboratorio ya se desarrollaron otros animatrónicos entre ellos uno similar que realizaron Luis María Tizeira y Guillermo rojas que funcionan mediante un sistema de captura de movimiento, (kinect) pero como tiene una cámara y sensores ópticos el usuario que oficia de titiritero, tiene que estar aislado en un entorno muy preciso y por delante y detrás de él no pude haber ninguna persona puesto que el sistema se confunde y da posiciones erróneas. El sistema que se propone es poner sobre el cuerpo del usuario sensores, de modo que en el campo de vista del usuario puede haber gente sin que el sistema deje de funcionar, se podría pensar esto como un desmejora del sistema, pero en realidad el hecho de que quien maneja el animarónico esté aislado, como en el sistema anterior, representa un problema de diseño y de uso bastante difícil de cumplir, es por eso que proponemos esta mejora, pudiendo el titiritero estar rodeado de personas que es la situación más común en el uso de este tipo de animatrónicos.

**Duración y Fases de las tareas previstas**: En el ANEXO II, se presenta un diagrama a este efecto.

**Metodología**

**Lugar previsto de realización:** Lugar previsto de realización: Laboratorio de Animatrónica y Control Dinámico del Departamento de Electrónica de la FCEFyN – UNC

**Requerimiento de Instrumental y equipos**: Se cuenta con computadoras, con conexión a internet, y software necesario para simulación y diseño de los circuitos, fuentes de alimentación, osciloscopio, generador de señales, multímetros, todos los componentes necesarios para la realización del sistema (procesadores, motoreductores, sensores, módulos inalámbricos, herrajes, cables etc.)

**Inversión estimativa prevista por el alumno:** $ 20.000 realizada por los estudiantes a presentar el trabajo, juntamente con el director del laboratorio mencionado.

**Apoyo Económico externo a la Facultad:** La Facultad brinda el espacio del Laboratorio como lugar de trabajo.

**Referencias Bibliográficas o de Software**:

K.S. Fu, R.C. González, C.S.G. Lee. ROBOTICA: Control, Detección, visión e inteligencia. Editorial McGraw-Hill, 1990. 599 p.

Domingo, Juán. Robótica, Apuntes para la Asignatura.. Febrero, 2001. 177 p.

Kelly, Rafael. Santibáñez, Víctor. Automática Robótica. Control del movimiento de Robots Manipuladores. Editorial Prentice may, 2003.

Barrientos, Antonio. Peñin, Luis Felipe. Balagues, Carlos. Aracil, Rafael. Fundamentos de Robótica. Editorial McGraw-Hill, 97.

Timothy J. Ross. Fuzzy Logic Whit Engineering Applications. Editorial McGraw-Hill, 95.

Eugenio Martín Cuenca, José Ma. Angulo Usategui, Ignacio Angulo Martínez. Https://es.mathworks.com/help/matlab/

Peter Wellnhofer, John Sibbick. Atlas Ilustrado de los Pterosaurios , 1994 Editorial Susaeta

Alexander W A Kellner. Pterossauros os señores do céu do Brasil 2006. Editorial Vieira & Lent.

Steve Parker, Alice Roberts. Evolución , toda la historia 2016 , Editorial Sophi Blackman, Rebecca Gee, Frank Ritter, Dorothy Stannard.

**Recibido Cátedra PI**

.....

Firma

Córdoba, / / .

## ANEXO I

## DESCRIPCIÓN Y DESARROLLO DEL PROYECTO

Tapejara Imperator fue un reptil volador que vivió en Brasil hace 108 millones de años en la“Formación Fantana” y que fue reconstruido en forma de robot en el año 2005 con el fin de presentarlo en congresos y charlas de divulgación científica, sin embargo en su último viaje (2009) , Tapejara quedó varado en Brasil por razones de fuerza mayor y estuvo más de 8 años sin poder mostrarse. En un viaje realizado por el director del laboratorio de Animatrónica y Control Dinámico y un miembro del mismo laboratorio, fue recuperado y traído de regreso a Córdoba con el objetivo de recuperarlo y repararlo.

Para realizar el control del animatrónico se utilizará un acelerómetro y un giroscopio (por ala) que se ubicará solidario al eje del motoreductor de modo que brinde una señal proporcional al ángulo de cada ala. Esta señal deberá ser procesada e interpretada (mediante algún filtro) por el microcontrolador, el cual se encarga de accionar a los correspondientes drivers de los motores.

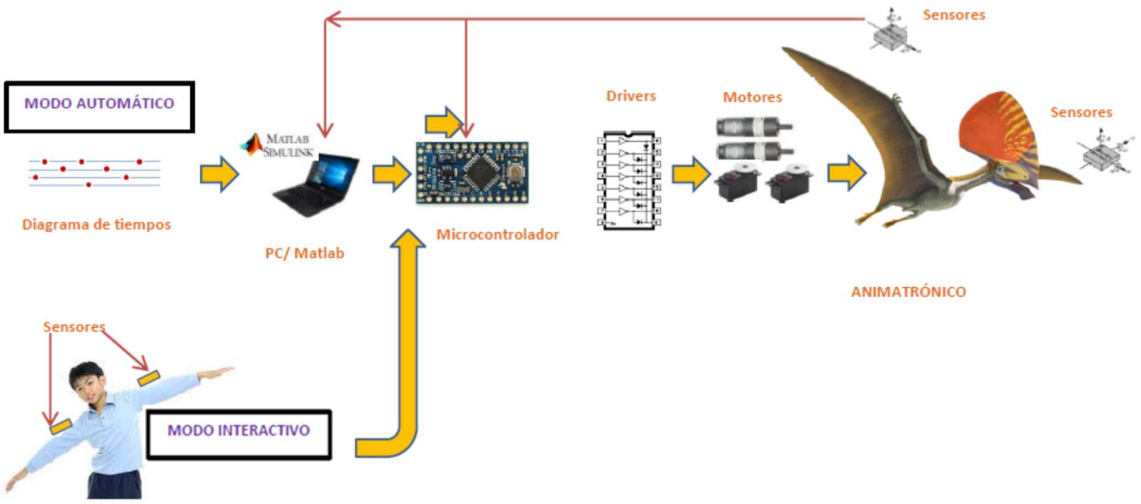
Como el proyecto está pensado para la exposición en parques y museos, se requieren dos modos de funcionamiento, uno “rutina” y uno “interactivo”.

Cabe aclarar que el animatrónico estará disponible para ser presentado en esta Facultad y a disposición para futuras exposiciones de la Universidad Nacional de Córdoba.

Modo rutina: consta de una rutina precargada con una serie de movimientos de alas, cabeza y pico que se repite cada determinado tiempo. Estas variables son definidas por el programador y deberán ser seleccionadas para cada exposición.

Modo interactivo: Teniendo en cuenta que este animatrónico será exhibido en algún parque o museo, su principal modo de control deberá estar relacionado con las últimas reglas de la divulgación científica en este tipo de demostraciones, teniendo en cuenta esta premisa, se pretende brindar al usuario un dispositivo que permita reproducir los movimientos de dicho usuario en el robot.

Si bien no se sabe excatamente a priori el esquema exacto que van a tener los dos modos de funcionamiento principales se muestran seguidamente dos esquemas probables.




## MODO AUTOMÁTICO

**Diagrama de tiempos:**
Es el diagrama que da el paleontólogo o experto de los movimientos que tienen que tener alas y cabeza para una secuencia verosímil.
**PC/ Matlab:**
Se codifican los movimientos con los tiempos requeridos en Matlab y se procesan cada uno de los movimientos.
**Microcontrolador:**
Será el dispositivo que permita el procesado de las señales tanto hacia los actuadores (motores) como la que viene de los sensores.
**Drivers :**
Serán los circuitos que permitan activar los dos tipos de motores que a priori creemos utilizar.
**Motores**
Serán del tipo motorreductores los que están adosados a las alas y los que muevan la cabeza y el pico serán servomotores.
**ANIMATRÓNICO**
El animatrónico es el robot a controlar que como se expresó va a ser un pterosaurio.
**Sensores:**
Los sensores estarán colocados en las alas para realimentar el ángulo requerido y serán acelerómetros tipo IMU 5060.

## MODO INTERACTIVO

En este modo, el usuario podrá controlar los movimientos del animatrónico con su cuerpo generando un diagrama de tiempos especial para ser procesado hasta el robot.

**ANEXO II**

Cabe aclarar que la escritura del trabajo final se irá realizando a medida que avance el proyecto pero al final se asignará un tiempo especial para la presentación.

| | Primer mes | | | | Segundo mes | | | | Tercer mes | | | | Cuarto mes | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1º semana | 2º semana | 3º semana | 4º semana | 5º semana | 6º semana | 7º semana | 8º semana | 9º semana | 10º semana | 11º semana | 12º semana | 13º semana | 14º semana | 15º semana | 16º semana |
| Selección y prueba de módulos sensores y actuadores. | | | | | | | | | | | | | | | | |
| Prueba en conjunto y accionamiento por software | | | | | | | | | | | | | | | | |
| Programación y diseño de los modos de operación y Pruebas mecánicas | | | | | | | | | | | | | | | | |
| Puesta a punto y calibrado y escritura del trabajo | | | | | | | | | | | | | | | | |