



Escuela Técnica Superior de
Ingeniería Informática

Trabajo Fin de Grado

Medidor de impedancia usando microcontrolador STM

Realizado por

Emilio Silvestre Mérida - Salvador Meléndez Muñoz

Dirigido por

Pablo Pérez García

Para la obtención del título de

Grado de Ingeniería Informática - Ingeniería de Computadores

Departamento de Tecnología Electrónica - Universidad de Sevilla

Convocatoria Julio – Cursos 2023/2024

1.	INTRODUCCIÓN DEL PROYECTO	2
1.1.	Descripción del proyecto	2
1.2.	Estado del arte	3
1.2.1.	Fundamentos teóricos.....	3
1.2.2.	Medición de la Impedancia	5
1.3.	Objetivos del proyecto	11
2.	PLANIFICACIÓN DEL PROYECTO	13
2.1.	Análisis de requisitos.....	13
2.2.	Elección de componentes y herramientas.....	15
2.2.1.	Microcontrolador	15
2.2.2.	Resistencias	16
2.2.3.	Software	16
2.2.4.	Osciloscopio	17
2.2.5.	Otros componentes considerados	18
2.3.	Análisis de costes.....	20
2.4.	Planificación de hitos.....	21
2.4.1.	Descripción y estructura de los hitos	21
2.4.2.	Planificación temporal de los hitos.....	25
2.4.3.	Distribución de trabajo	34
2.4.4.	Matriz de Riesgo	36
3.	DESARROLLO DEL PROYECTO	39
3.1.	Fundamentos del Sistema Microcontrolador	39
3.2.	Diseño del sistema	47
3.2.1.	Generador de señal	47
3.2.2.	Lector de señal.....	50
3.3.	Desarrollo del sistema	52
3.3.1.	F1: Estudio previo y elección del proyecto.....	52
3.3.2.	F2: Obtención de conocimientos.....	53
3.3.3.	F3: Generación de documentos.....	55
3.3.4.	F4: Generación de señal.....	55
3.3.5.	F5: Lectura de la señal.....	80
3.3.6.	F6: Pruebas y PCB.	90
3.4.	Seguimiento de la planificación	96
3.5.	Plan de pruebas	100
4.	CIERRE DEL DOCUMENTO	101
4.1.	Conclusiones	101
4.2.	Bibliografía.....	103
4.3.	Galería de figuras	104

1. INTRODUCCIÓN DEL PROYECTO

1.1. Descripción del proyecto

En este proyecto trata sobre el diseño y desarrollo de un dispositivo medidor de impedancia haciendo uso de un microcontrolador. El dispositivo realizará una medición sobre una muestra a la que aplicará corriente proveniente de una fuente generadora de señal para medir su impedancia.

El diseño del circuito está basado en la tesis doctoral del tutor Pablo Pérez García[1].

El dispositivo consta de dos partes:

- **Generación de la señal:** Esta parte es la encargada de generar una señal senoidal a partir de una amplitud y frecuencia previamente definida. Esta señal se aplicará a una muestra mediante dos sondas.
- **Lectura de resultado:** Esta parte realizará una medición en seis puntos e interpretará los resultados.

Durante la ejecución del proyecto se intenta implementar la mayoría de los componentes del circuito usando los componentes internos del microcontrolador.

1.2. Estado del arte

1.2.1. Fundamentos teóricos

CONCEPTO DE IMPEDANCIA

Se denomina **impedancia** (Z) a la medida de la oposición de un circuito cuando se ve afectado por la aplicación de una tensión. Se trata de un elemento importante en lo que a circuitos electrónicos y circuitos eléctricos y viene definida por la relación que existe entre el voltaje y la intensidad en **circuitos de corriente alterna**.

$$Z = \frac{V}{I} [\Omega]$$

En estos circuitos alimentados por una corriente alterna tanto la fase de la corriente, como su amplitud, pueden verse alteradas con respecto al flujo (corriente sinusoidal). Es por esto que, a diferencia de un circuito alimentado por corriente continua donde el flujo es limitado, la impedancia va más allá de la resistencia y se le suma lo que se denomina como **reactancia**. De modo que, matemáticamente:

$$Z = R + j X [\Omega]$$

La **impedancia** (Z) viene representada como un número complejo resultado de una suma vectorial donde:

- R hace referencia a la resistencia real del circuito, parte real.
- X hace referencia a la reactancia, parte imaginaria (variación en el flujo de corriente debido a elementos capacitivos e inductivos).

CONCEPTO DE FASOR

Puesto que hablamos de una corriente sinusoidal, las ondas senoidales son el tipo de onda más común que encontramos en la corriente alterna. Estas magnitudes sinusoidales son representadas mediante **fasores**, elementos que se identifican con fase y la magnitud de la onda.

Dada esta fase, se cuenta con una tensión cuya magnitud no es constante, sino que varía en el tiempo. De modo que:

$$v(t) = V_{max} \cdot \text{sen}(\omega t)$$

Donde:

- $v(t)$ hace referencia a la tensión de la fuente que varía en el tiempo.
- V_{max} hace referencia al valor máximo de la tensión (amplitud de la onda).
- ωt hace referencia al argumento de la tensión, indicando la posición de la onda.

Por lo tanto, dado que la impedancia viene representada como un número complejo que tiene como variable su reactancia, haremos uso de los fasores para representarla. Tal y como se mencionó anteriormente, los fasores se componen de dos partes fundamentales representadas en la fórmula anterior:

- Magnitud: Hace referencia a la amplitud de la onda senoidal.
- Fase: Hace referencia a la posición de la onda.

Se tratan pues de números complejos que tienen como variable a su fase, motivo por el cual son usados para representar la impedancia.

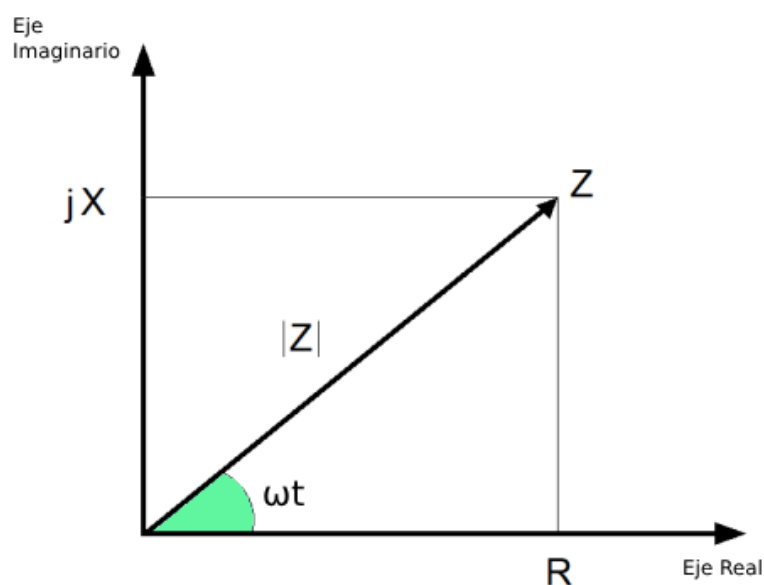


Figura 1. Impedancia en plano fasorial

1.2.2. Medición de la Impedancia

El hecho de medir la impedancia en un circuito resulta algo muy útil e importante a la hora de analizar y diseñar el circuito. Esta medida nos va a aportar información sobre el comportamiento del circuito ante las diferentes frecuencias a las que éste va a operar. Esta es una de las características de la impedancia, su reactancia (X) va a hacer que varíe al depender ésta de la frecuencia a la que opera el circuito.

De modo que es de gran importancia tener un rango de frecuencias que cubra el rango de operación del circuito, observando así posibles fallos, optimizaciones, rendimiento óptimo, etc, del mismo.

La impedancia, al ser un valor “imaginario”, hace que se requiera de instrumentos específicos para medirla, así como el uso de diferentes técnicas de medición en base a la aplicación que va a tener el circuito o el rango de frecuencias al que va a operar.

GENERADOR DE SEÑALES Y OSCILOSCOPIO

Se procede generando una señal conocida para testear. En función de la aplicación del circuito o de lo que se requiera testear, esta función podría ser senoidal, cuadrada u otra forma que sea relevante para el análisis. También se establece la frecuencia necesaria para realizar la medición correspondiente.

Mediante la adecuada conexión de los cables, se produce a inyectar esta señal en el componente sobre el que se va a realizar la prueba. Es importante tener en cuenta algunos factores, como aislamiento eléctrico, a la de realizar la prueba para evitar interferencias o posibles daños al circuito.

Al aplicar la señal prueba, mediante el uso del osciloscopio, se puede observar la forma de onda resultante, siendo posible visualizar su fase, amplitud y forma para el posterior análisis y ajustes en el circuito en base a los resultados.

MEDICIÓN ICR O PUENTE DE IMPEDANCIA

Esta técnica permite la medición de impedancias partiendo de un amplio rango de frecuencias y valores. Se trata de un método que determina con gran precisión la impedancia del componente que se pone bajo prueba.

Se utiliza un circuito tipo puente del tipo LCR (Inductancia, Capacitancia y Resistencia) que consta de elementos con impedancias conocidas (generalmente elementos resistivos y capacitivos), así como impedancias desconocidas (componente que se pondrá bajo prueba). Mediante el ajuste de los valores de los componentes que sí se conocen, la relación entre las impedancias proporciona una medición con gran precisión del componente a prueba.

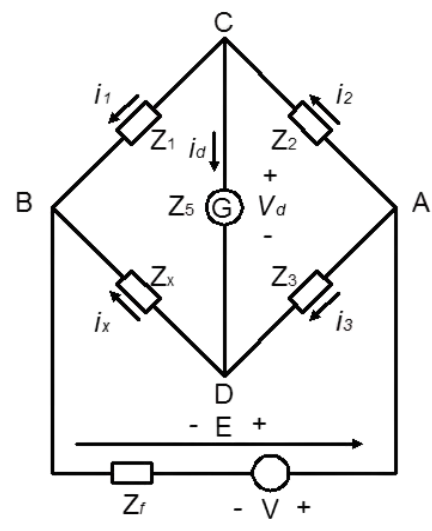


Figura 2. Puente LCR [2]

Medición en 2 puntos

Este método considera medir resistencia de puesta a tierra, pues esta es principalmente resistiva en la mayoría de los casos. En el caso de la medición en dos puntos, la resistencia de puesta a tierra del elemento que se quiere conocer, es medida en serie con una resistencia de tierra auxiliar. Esta resistencia de tierra auxiliar resulta insignificante en comparación con la del elemento que se quiere conocer, por lo que la resistencia resultante de la medición (suma de ambas resistencias) se atribuiría a la resistencia de tierra que se requiere conocer.

Se trata de un método muy simplificado que, aunque aporta la resistencia de tierra que se requiere conocer, puede generar algunos errores. Por ejemplo, cuando la resistencia de tierra auxiliar es muy parecida a la resistencia de tierra que se requiere conocer.

En nuestro caso, por sencillez será el método que emplearemos, pero consideramos importante tener en cuenta otros métodos que podrían ofrecer mejores resultados.

Medición en 3 puntos

Se trata de un método similar a la medición en dos puntos, con la diferencia de que son necesarias 2 resistencias de tierra auxiliares (R_2 , R_3) además de la resistencia de puesta a tierra del elemento que se quiere conocer (R_1). Empleando el método de los dos puntos, se mide la resistencia entre cada par de resistencias obteniendo R_{12} , R_{23} y R_{13} . De esta forma se obtiene que:

$$R_1 = \frac{R_{12} - R_{23} + R_{13}}{2}$$

Es importante tener en cuenta que puede haber errores en las mediciones de los pares de resistencias si las resistencias auxiliares son mucho más altas que la del elemento que se quiere conocer. Esto puede afectar a la precisión de los resultados finales y, además, este método puede producir valores inexactos si no se mantiene una distancia adecuada entre resistencias.

Medición en 4 puntos

Es un método de medición en el que se van a usar 4 puntos de contacto, a diferencia de los métodos anteriores. En este caso, 2 de estos puntos serán utilizados para aplicar una corriente de prueba al sistema. De esta forma, se usan los otros 2 puntos para medir la caída de tensión obteniéndose así una medida precisa.

Mediante la Ley de Ohm será posible realizar la medida de la impedancia, pues como se definió anteriormente:

$$Z = \frac{V}{I} [\Omega]$$

Donde V es la caída de tensión medida, I es la corriente de prueba aplicada y Z la impedancia del sistema.

MFIA Impedance Analyzer

Es un medidor conocido hoy día para medir impedancias. Se trata de un analizador de impedancia digital y medidor de LCR de precisión que establece el nuevo estándar para las mediciones de impedancia en el rango de frecuencia de 1 mHz a 500 kHz. El MFIA tiene una precisión básica del 0.05% y opera en un rango de medición que abarca desde 1 mΩ hasta 1 TΩ. También se caracteriza por una alta repetibilidad en las mediciones y un pequeño cambio con la temperatura.



Figura 3. MFIA Impedance Analyzer [3]

Tiene varias aplicaciones, como pueden ser:

- **Ingeniería eléctrica:** sensores, supercondensadores, caracterización de semiconductores, DLTS (Deep Level Transient Spectroscopy), tecnología de pantallas, resistencias ultra altas, dieléctricos de alta Q.

- **Investigación de materiales:** dieléctricos de polímeros, cerámicas y compuestos, materiales solares, caracterización de películas delgadas y nanoestructuras.
- **Bioimpedancia:** análisis de impedancia de tejidos, crecimiento celular, investigación alimentaria.

Entre 1 mHz y 500 kHz, y entre 1 Ω y 1 M Ω (con limitaciones hacia frecuencias más altas) se especifica una precisión del 0.05%, aunque el rango de medición se extiende aún más con una precisión especificada reducida del 0.1% y 1% para cubrir un rango de medición de 10 m Ω a 1 G Ω . Incluso fuera de este rango, son posibles mediciones repetibles pero la precisión podría caer por debajo del 1%.

Medir impedancias altas a bajas frecuencias puede ser particularmente desafiante cuando los valores tienen que obtenerse cerca de la frecuencia de línea. Un blindaje de muestra adecuado junto con un filtro sinc y la posibilidad de operación con batería proporcionarán los resultados más precisos.

Impedimed SFB7

Se trata de un dispositivo de espectroscopia de bioimpedancia (BIS) de un solo canal y tetrapolar que escanea 256 frecuencias entre 4 kHz y 1000 kHz. El dispositivo utiliza el modelado de Cole con la teoría de mezcla de Hanai para determinar el agua total del cuerpo (TBW), el fluido extracelular (ECF) y el fluido intracelular (ICF) a partir de los datos de impedancia. La masa libre de grasa (FFM) y la masa grasa (FM) se calculan luego en el dispositivo. Se puede realizar un análisis adicional de datos en el software de soporte. Por lo tanto, no se requieren ecuaciones de predicción específicas para la población (algoritmos) para el análisis de datos.



Figura 4. Impedimed SFB7 [4]

Keysight E4980A Precision LCR Meter

Se trata de un medidor LCR ampliamente utilizado en diversas aplicaciones. Su funcionalidad principal se centra en medir inductancia (L), capacitancia (C) y resistencia (R) con una precisión y resolución con gran precisión. Esta versatilidad lo convierte en una potente herramienta cuando se requiere una caracterización precisa de sus componentes.



Figura 5. Keysight E4980A Precision LCR Meter [5]

Posee un amplio rango de frecuencia de operación (desde 20 Hz hasta 2 MHz), con una resolución con gran precisión en cualquier rango. Esta versatilidad en la frecuencia permite realizar mediciones precisas en una amplia gama de dispositivos y circuitos.

Cuenta con una precisión básica del 0.05%, la cual se complementa con una gran repetibilidad en las mediciones, tanto en impedancias bajas como altas, asegurando mediciones confiables y precisas en una variedad de escenarios.

1.3. Objetivos del proyecto

El objetivo principal del proyecto es desarrollar un **dispositivo capaz de medir la impedancia** en una muestra haciendo uso de un microcontrolador. Dicho dispositivo debe tener unas características concretas:

- La lógica del dispositivo, así como todos los componentes que sean posibles han de implementarse usando el microcontrolador.
- El dispositivo debe generar una señal senoidal y realizar la medida de la transmisión de dicha señal tanto a la entrada como a la salida del circuito.
- La señal generada debe ser configurable en términos de amplitud y frecuencia.

Además del objetivo principal cabe destacar los conocimientos y las distintas áreas que se van a trabajar durante la ejecución del proyecto:

- **Microcontroladores:** se trabajará tanto la lógica del microcontrolador como el funcionamiento y configuración de sus componentes internos como DAC, ADC, timers, etc.
- **DMA:** se configurará y usará “Direct Memory Access” en todos los componentes que sea posible para aumentar la velocidad de trabajo del dispositivo.
- **Tierra virtual:** el diseño a implementar contiene amplificadores operacionales que requieren una tierra virtual lo cual quiere decir, que la referencia de dichos componentes no estará conectada a la tierra del dispositivo sino a un potencial de referencia constante que habrá que implementar.
- **Impedancia:** durante el proyecto se aprenderá a trabajar con la impedancia y a interpretar los resultados obtenidos de las lecturas tomadas en las muestras.

- **Ondas:** la lógica del programa implementado en el microcontrolador debe permitir modificar la amplitud y la frecuencia de onda, por lo tanto, se debe conocer cómo trabajar con las ondas senoidales y ajustar dichos parámetros.
- **Soldadura:** durante el proyecto se elaborará un prototipo que requerirá de soldaduras para poder conectar todos los componentes que son externos al microcontrolador.
- **PCB:** se realizará una PCB para implementar el diseño final usando Kicad.
- **Corriente Alterna y Corriente Continua:** para generar la señal que se aplicará a la muestra habrá que transformar la corriente continua que usa el microcontrolador en corriente alterna usando un DAC. Una vez hecha la lectura habrá que transformar la corriente alterna obtenida en corriente continua usando un ADC para poder interpretar los resultados.

El alcance del proyecto incluye el estudio del estado del arte, el diseño, la implementación y el testeo de un prototipo sobre placas perforadas y por último el diseño y desarrollo de un dispositivo de reducidas dimensiones que permita realizar medidas de impedancia.

2. PLANIFICACIÓN DEL PROYECTO

2.1. Análisis de requisitos

- **R1- Microcontrolador:** El diseño del sistema se debe implementar usando el microcontrolador como elemento principal e implementar de manera interna la mayor cantidad de componentes posibles.
 - **R1.1- DAC:** El diseño necesita que el microcontrolador cuente con al menos 1 DAC para poder generar la señal inicial(**R2**).
 - **R1.2- ADC:** El diseño necesita que el microcontrolador cuente con al menos 2 ADCs, uno para leer la señal antes de aplicarla a la muestra (**R5**) y otro para realizar las lecturas en la muestra (**R7**).
 - **R1.3- OPAM:** El diseño necesita que el microcontrolador cuente con al menos 4 OPAMPs:
 - 1 amplificador Operacional para inyectar la señal del DAC en modo follower.
 - 2 amplificadores Operacionales en configuración de Bipolarity Current Source Sink [1].
 - 1 amplificador Operacional para establecer un nivel de referencia o Tierra Virtual (**R6**).
 - **R1.4- Timers:** El diseño necesita que el microcontrolador cuente con al menos 2 timers, 1 que dispare la generación de señal y 1 que dispare la lectura.
- **R2- Señal senoidal:** El sistema debe generar una señal senoidal usando el dispositivo microcontrolador.
- **R3- Modificación de señal:** El sistema debe poder modificar la señal senoidal tanto en amplitud como en frecuencia para poder tomar varias lecturas con distintas señales.
- **R4- DMA:** El sistema debe usar Direct Memory Access para agilizar el funcionamiento y tener un mayor rendimiento.

- **R5- Medición de señal en circuito:** La señal debe de medirse antes de ser aplicada a la muestra para poder obtener el valor inicial de la misma, realizando una medición en 2 puntos.
- **R6- Tierra Virtual:** El circuito tiene que contar con una tierra virtual para que el sistema lo use como referencia.
- **R7- Interpretación de los resultados:** El sistema debe poder adquirir las medidas a partir de los ADCs y devolver las medidas a través del puerto UART (Virtual COM).
- **R8- PCB:** Se debe desarrollar una PCB que contenga los elementos del sistema intentando que sea lo más compacta posible no superando un círculo de 3cm de diámetro.

2.2. Elección de componentes y herramientas

2.2.1. Microcontrolador

Para solventar el requisito **R1** y **R4** el microcontrolador del sistema debe tener suficientes componentes internos y tener la posibilidad de usar DMA. Además, se debe tener en cuenta que algunos microcontroladores tienen componentes suficientes, pero no son compatibles a la hora de configurarlos de manera conjunta.

El microcontrolador elegido en nuestro caso es el **STM32G474RE** que tiene un Cortex-M4, hemos elegido este microcontrolador porque es capaz de implementar la mayoría de componentes de manera interna, concretamente todos menos las resistencias. Además, implementa un sistema DMA que permite aligerar el micro de las tareas básicas con los periféricos. Por último, consideramos que el fabricante STMicroelectronics es adecuado ya que sus tecnologías son ampliamente estudiadas a lo largo de la carrera.

Componentes del **STM32G474RE**:

- 5 ADCs
- 7 DACs
- 6 OPAMPs
- 17 Timers:
 - 2 temporizadores de 32 bits y 2 temporizadores de 16 bits con hasta cuatro entradas IC/OC/PWM o contador de pulsos y entrada de codificador cuadratura (incremental)
 - 3 temporizadores de control de motor avanzados de 16 bits y 8 canales, con hasta 8 canales PWM, generación de tiempo muerto y parada de emergencia
 - 1 temporizador de 16 bits con 2 IC/OC, un OCN/PWM, generación de tiempo muerto y parada de emergencia

- 2 temporizadores de 16 bits con IC/OC/OCN/PWM, generación de tiempo muerto y parada de emergencia
- 2 temporizadores de vigilancia (independientes, ventana)
- 1 temporizador SysTick: contador descendente de 24 bits
- 2 temporizadores básicos de 16 bits
- 1 temporizador de bajo consumo de energía
- 16 canales de DMA

Otros microcontroladores que fueron considerados fueron el STM32H743 y STM32L486, pero debido a la falta de OPAMPs y la disponibilidad de un STM32G474RE en el departamento hicieron que fueran descartados.

2.2.2. Resistencias

Durante el desarrollo del circuito del prototipo se necesitan una serie de resistencias que no pueden ser implementadas usando el microcontrolador, para ello se emplean resistencias SMD debido a su disponibilidad en el laboratorio y a que su tamaño reducido y modo de unirse al circuito facilitan la fabricación del prototipo.

2.2.3. Software

A la hora de desarrollar el proyecto hemos usado dos programas:

- **STM32Cube IDE:** es una plataforma de desarrollo para microcontroladores de STM32, hemos decidido usarla debido a que se estudia su funcionamiento y configuración durante el grado, y debido a su compatibilidad con el microcontrolador que hemos seleccionado.
- **WaveForms:** es un instrumento virtual de Digilent que se usa para medir y probar señales de los dispositivos. En nuestro caso lo usaremos como osciloscopio para medir las señales generadas a lo largo del proyecto y como generador de señal para poder probar la lectura de los ADCs.

2.2.4. Osciloscopio

Se hará uso de un osciloscopio para poder medir las señales obtenidas del sistema, este osciloscopio se usará de manera complementaria al software WaveForms. Para poder usar dicho software empleamos el dispositivo Analog Discovery 2 que nos permite usar sus funcionalidades. Lo usamos con dos configuraciones distintas:

- Con sondas:

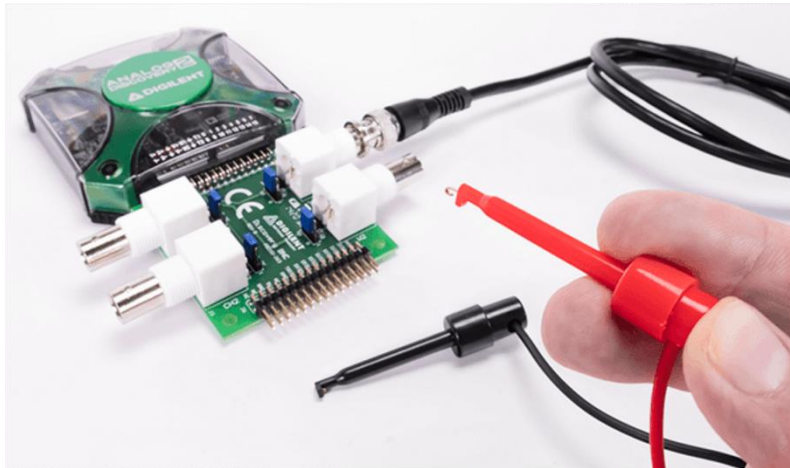


Figura 6. Analog Discovery 2 con conector BNC [6]

- Con cableado directo:

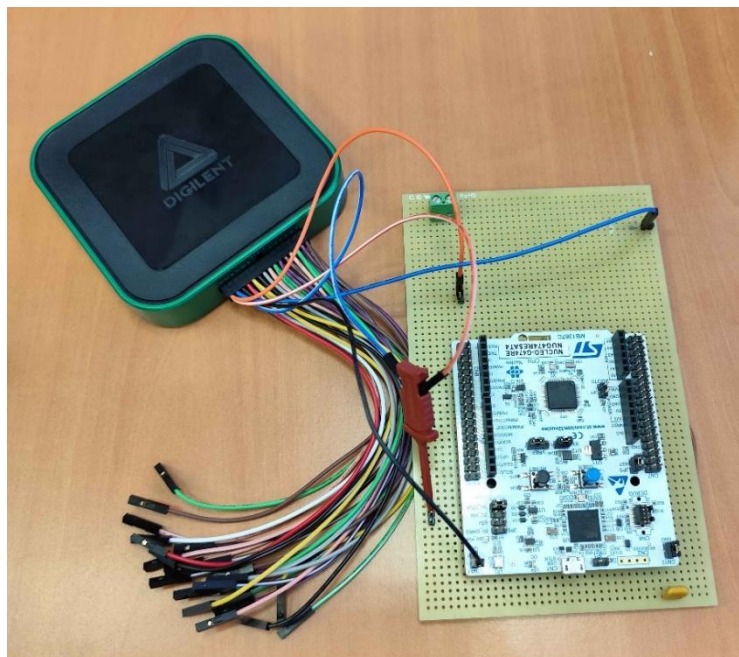


Figura 7. Analog Discovery 2 con conector a pines

2.2.5. Otros componentes considerados

Durante la planificación del proyecto hemos considerado el uso de otros dispositivos que finalmente han sido descartados o sustituidos:

- **MUX/DEMUX:** se consideró usar un demultiplexor analógico para hacer un selector de caminos con diferentes resistencias fijas a la salida del generador de señal para en combinación con la fuente de corriente [1] controlar los niveles de corriente a inyectar en la muestra a medir.

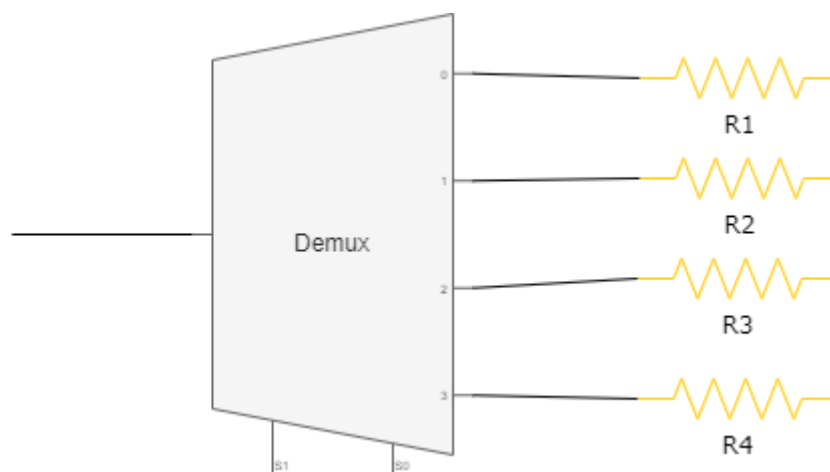


Figura 8. Esquema camino de resistencias con Demux

Entre los demultiplexores que encontramos están:

- **ADV3221/ADV3222:** son multiplexores analógicos 4:1 de alta velocidad que cuenta con líneas de control con latches que permiten la conmutación síncrona.
- **DG408:** destaca por su alta precisión en las señales conmutadas. Cuenta con interruptores SPST y es adecuado para aplicaciones que requieren una alta fidelidad en la conmutación de señales analógicas.
- **MAX4617:** Destaca en su aplicación para señales de alta calidad. Es de alto rendimiento con interruptores SPST y ofrece una resistencia de baja conducción.

- **ADG609:** Destaca por su bajo consumo de energía. Es una opción eficiente en términos de energía para aplicaciones portátiles. Cuenta con interruptores SPST.

Todos ellos son multiplexores que pueden funcionar como demultiplexores, tras analizar las posibilidades y ver que muchos de ellos se polarizan a $\pm 5V$, algo que es demasiado para el microcontrolador, por lo que se descartó finalmente esta idea.

- **OPAMPs externos:** en un inicio antes de elegir el microcontrolador se valoró la posibilidad de implementar OPAMPs de manera externa, algunas de las opciones fueron:
 - **LM324M:** es un circuito integrado de Texas instruments que contiene 4 OPAMPs con una corriente de polarización mínima de $\pm 1.5V$
 - **LM3900n:** es un circuito integrado de Texas instruments que contiene 4 OPAMPs con una corriente de polarización mínima de $\pm 2.2V$

Finalmente, al escoger un microcontrolador y comprobar que era compatible con el resto de los componentes, se descartó esta opción.

- **Amplificador de instrumentación:** En una etapa previa, se consideró la utilización de un amplificador de instrumentación adicional para realizar medidas de impedancia muy grande. Debido a la capacidad de amplificación de estos dispositivos se intuye que la medida resultante debe ser más precisa en el caso de impedancias muy grandes que no puedan medirse con la circuitería propuesta. Los componentes específicos fueron valorados:
 - AMP01: amplificador instrumental con una ganancia mínima de 0,1.
 - AD625: amplificador instrumental con una ganancia mínima de 1.

Esta idea se mantiene como trabajo futuro para el proyecto.

2.3. Análisis de costes

- Componentes:

Componente	Precio
1 Microcontrolador STM32G474RE	30.41€
4 Resistencias	0.02€ (aprox.)
Licencia Software	Gratis (Open Source)
1 Osciloscopio (Opcional)	192.21€
1 Analog Discovery 2	359.60€
TOTAL	582.24€

- Recursos humanos:

Personal	Horas	Precio/Hora	Precio
Emilio Silvestre Mérida	300	9€/h	2700
Salvador Meléndez Muñoz	300	9€/h	2700
TOTAL	600	9€/h	5400

2.4. Planificación de hitos

Durante la realización del proyecto y para facilitar la organización del mismo se realizan una serie de fases en las que podemos dividir el proyecto que terminan con la consecución de un hito asociado, aunque antes de alcanzar el hito se realizan una serie de tareas necesarias para ello. Dicha organización se detalla a continuación usando la siguiente nomenclatura:

- **FX:** hace referencia a la fase del proyecto X.
- **HX:** hace referencia al hito de la fase X.
- **TX.Y:** hace referencia a la tarea Y de la fase X.

La tarea final de una fase siempre acaba dando lugar a la consecución del hito.

2.4.1. Descripción y estructura de los hitos

F1: Estudio previo y elección del proyecto.

Durante esta fase el objetivo es encontrar un punto de partida para el proyecto de forma que podamos empezar a trabajar en adquirir los conocimientos necesarios para la acometida del mismo.

H1- Proyecto seleccionado: definición del ámbito del proyecto que se va a realizar, así como de un alcance temporal que pueda servir como punto de partida para la realización del mismo.

T1.1- Estudio previo: para poder barajar posibilidades sobre qué tipo de proyectos se pueden abordar, se siguen las recomendaciones del tutor en cuanto a qué campos pueden tener contenido suficiente para la realización del proyecto y se realiza un estudio superficial de dichos campos para saber qué conocimientos se ponen en práctica.

T1.2- Elección del proyecto: se descartan proyectos y seleccionamos el proyecto que nos parece más interesante.

T1.3- Definición del proyecto: se valora el alcance total del proyecto de manera temporal y se realiza una planificación temporal superficial, dando lugar al **H1**.

F2: Obtención de conocimientos.

Durante esta fase el objetivo es obtener la base de conocimientos necesaria para poder empezar a desarrollar el proyecto.

H2- Conocimientos necesarios: obtención de los conocimientos necesarios para poder ejecutar la siguiente fase del proyecto y tener un mayor conocimiento general del campo que estamos trabajando.

T2.1- Reuniones con el tutor: reuniones previas que servirán como base para saber en qué temas debemos centrar nuestros esfuerzos ya que serán de utilidad a la hora de desarrollar el proyecto.

T2.2- Refuerzo: refuerzo de los conocimientos previamente adquiridos en otras asignaturas durante la realización del grado, ya que contienen mucha información condensada que puede resultar de utilidad.

T2.3- Ampliación: ampliación de los conocimientos a través de libros y medios de información disponibles con el objetivo de adquirir nuevos enfoques de la tecnología que vamos a emplear.

T2.4- Comprensión del dispositivo: obtener los conocimientos necesarios para trabajar con el dispositivo y conocer sus componentes, capacidades y funcionamiento.

T2.5- Repaso: puesta en común junto con el tutor de todos los conocimientos adquiridos en las tareas previas y validación por parte del tutor de que se posee al menos una base para poder iniciar el desarrollo del proyecto, completando así el **H2**.

F3: Generación de documentos.

Esta fase se realiza de forma paralela durante todo el proyecto tras la finalización de las dos fases anteriores y el objetivo es crear un documento final entregable que sirva como memoria del proyecto.

Ya que se indica que es en este punto donde se comienza a redactar este documento, se interpreta que tanto la **F1** como la **F2** están completas y por lo tanto se conocen las fases restantes del proyecto.

H3- Memoria entregable: documento que recoge todo lo necesario sobre el proyecto para ser presentado.

T3.1- Recolección de datos: se recaban datos de la realización y el desarrollo del proyecto.

T3.2- Redacción: se redacta la memoria ordenando y clasificando los datos obtenidos en la fase anterior, la repetición de tanto esta tarea como la anterior durante todo el proyecto dará lugar a la consecución del **H3**.

F4: Fase de generación de señal.

En esta fase se trabaja todo lo correspondiente a generar la señal que se aplicará a la muestra para cumplir con los requisitos **R2**, **R4** y **R6**.

H4- Prototipo de generador: será un prototipo diseñado en una placa de pruebas que será capaz de generar una señal senoidal modificable por código y debe de poder medirse la señal en varios puntos del circuito para comprobar que todo funciona correctamente.

T4.1- Configurar un DAC disparado por timer: se configura un DAC usando una LUT ("Look Up Table") fija con un Opamp que sirva de buffer y probar que se puede realizar la generación de forma correcta.

T4.2- Añadir DMA: modificar el DAC anterior para que use DMA ("Direct Memory Access") y cumplir con el requisito **R4**.

T4.3- Configuración Opamps del sistema: necesitaremos añadir dos Opamps más para el sistema de generación de la señal.

T4.4- Soldar y tierra virtual: soldar los componentes necesarios para poder probar la generación de señal en el circuito del generador de señal e implementar una tierra virtual (mediante un cuarto Opamp) para usarlo como referencia de voltaje y cumplir el requisito **R6**.

T4.5- Puntos de medición: soldar varios pines para poder medir la salida del Opamp conectado al DAC, la tierra virtual y la salida del circuito para poder comprobar que todo funciona adecuadamente.

T4.6- Lógica generadora: se implementan las funciones necesarias para cumplir con el requisito **R2** y conseguir que la señal generada por el DAC sea modificable.

T4.7- Testeo: probar el circuito para comprobar que todo funciona correctamente, con esto ya estaría completo el prototipo generador y por tanto se daría por conseguido el hito **H4**.

F5: Fase de lectura de señal.

En esta fase se implementa la parte del circuito encargada de recoger la lectura en la muestra e interpretarla para cumplir con los requisitos **R5** y **R7**.

H5- Toma de lecturas: adaptar el prototipo anterior para que sea capaz de tomar lecturas tanto del sistema antes de introducir la señal al circuito generador, es decir, al salir del Opamp conectado al DAC y en la salida del sistema. La lectura obtenida debe ser transmitida a través del micro-USB usando la UART del microcontrolador.

T5.1- Configurar ADCs: configurar 2 ADCs para que puedan tomar las lecturas oportunas y almacenar los datos.

T5.2- Conexiones y testeo: realizar las conexiones de los ADCs, uno a la salida del Opamp del DAC y otro a la salida del sistema, para realizar la medición en 2 puntos.

T5.3- UART: implementar una transmisión usando la UART del microcontrolador para poder transmitir los datos almacenados a través del puerto micro-USB para poder cumplir el **H5**.

F6: Fase de pruebas y PCB.

En esta fase se realizará una PCB para cumplir con el requisito **R8** y se realizarán una serie de pruebas en el sistema con el fin de obtener más información sobre su funcionamiento.

H6- PCB y resultados: el hito de esta fase consiste en los ficheros gerbers generados a partir del desarrollo de una PCB usando la herramienta KiCad y el informe de las pruebas realizadas al sistema.

T6.1- PCB: Elaboración de PCB usando KiCad y generación de los ficheros gerbers.

T6.2- Pruebas: realización de las pruebas que el tutor considere interesantes para la realización del proyecto.

T6.3- Informe: elaboración de un informe que recoja toda la información obtenida en la tarea anterior y otras pruebas del sistema, una vez realizado se completa el plan de pruebas y se da por conseguido el **H6**.

2.4.2. Planificación temporal de los hitos

A la hora de planificar los hitos se tiene en cuenta la duración estimada de cada tarea que se ha determinado teniendo en cuenta que el proyecto comenzó el 12 de septiembre y tiene que estar terminado como muy tarde el 26 de junio.

Tras asignar las 600 horas disponibles para la realización de este proyecto nos queda la siguiente tabla:

Fase	Duración estimada	Duración máxima
F1 - Estudio previo	Septiembre	Septiembre-Octubre
F2 - Adquisición de conocimientos	Septiembre-Octubre	Septiembre- Noviembre
F3 - Documentación	Octubre-Abril	Octubre-Junio
F4 - Generación de señal	Octubre-Diciembre	Octubre-Abril
F5 - Lectura de señal	Enero-Marzo	Enero-Junio
F6 - Pruebas y PCB	Abril	Junio

Tras la asignación de horas a cada una de las fases se tiene en cuenta la relación existente entre las fases para desarrollar el siguiente diagrama de PERT donde todas las fases son críticas y su retraso retrasaría el proyecto:

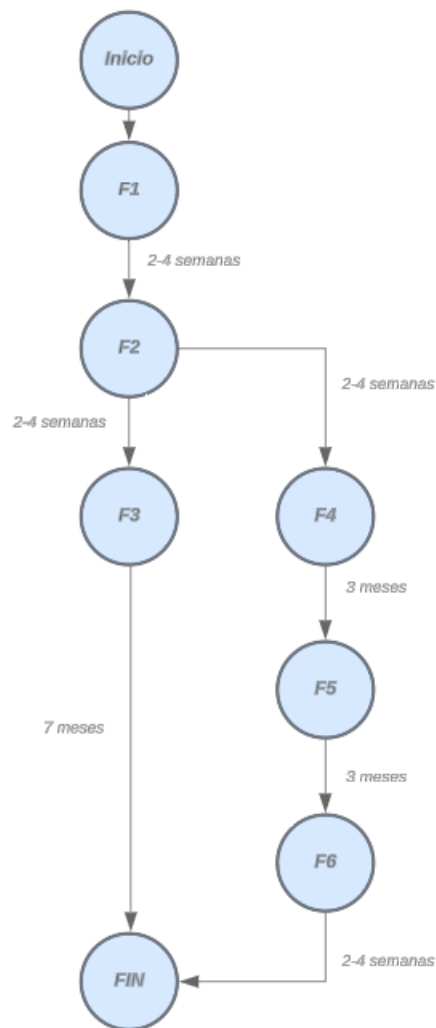


Figura 9. Diagrama PERT del proyecto

Con toda la información proporcionada en la distribución de horas y el grafo del camino crítico la planificación temporal de las tareas se muestra en los siguientes diagramas de Gantt:

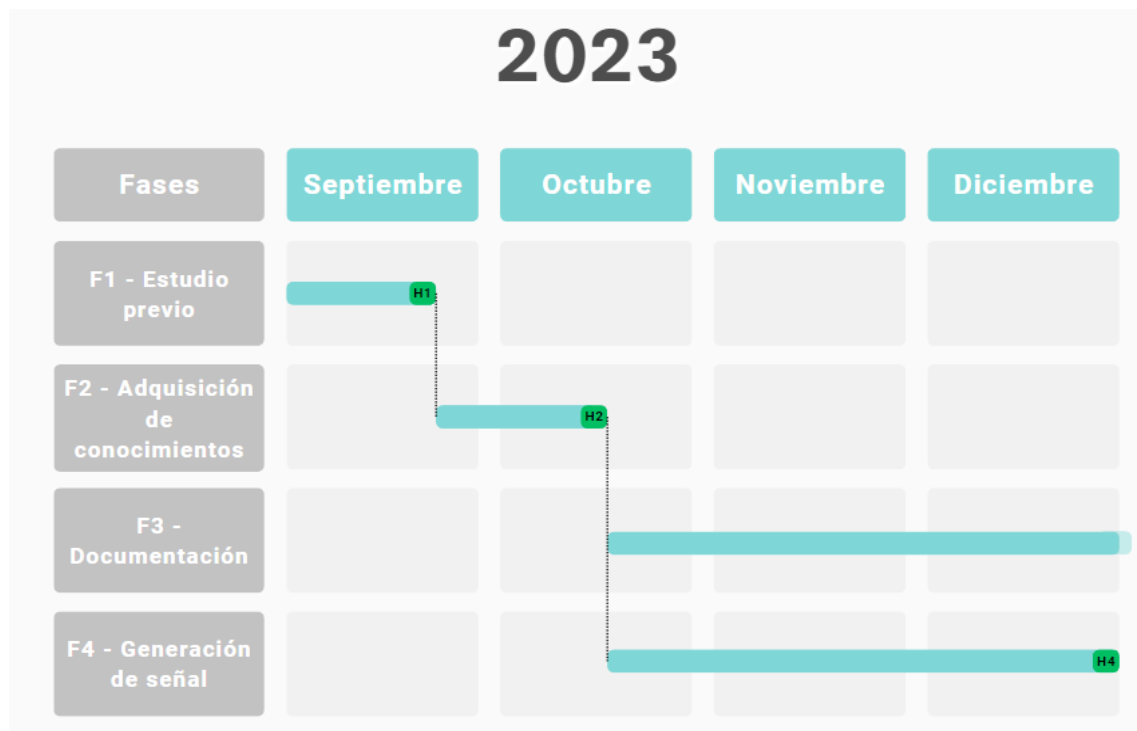


Figura 10. Diagrama de Gantt Planificación 2023

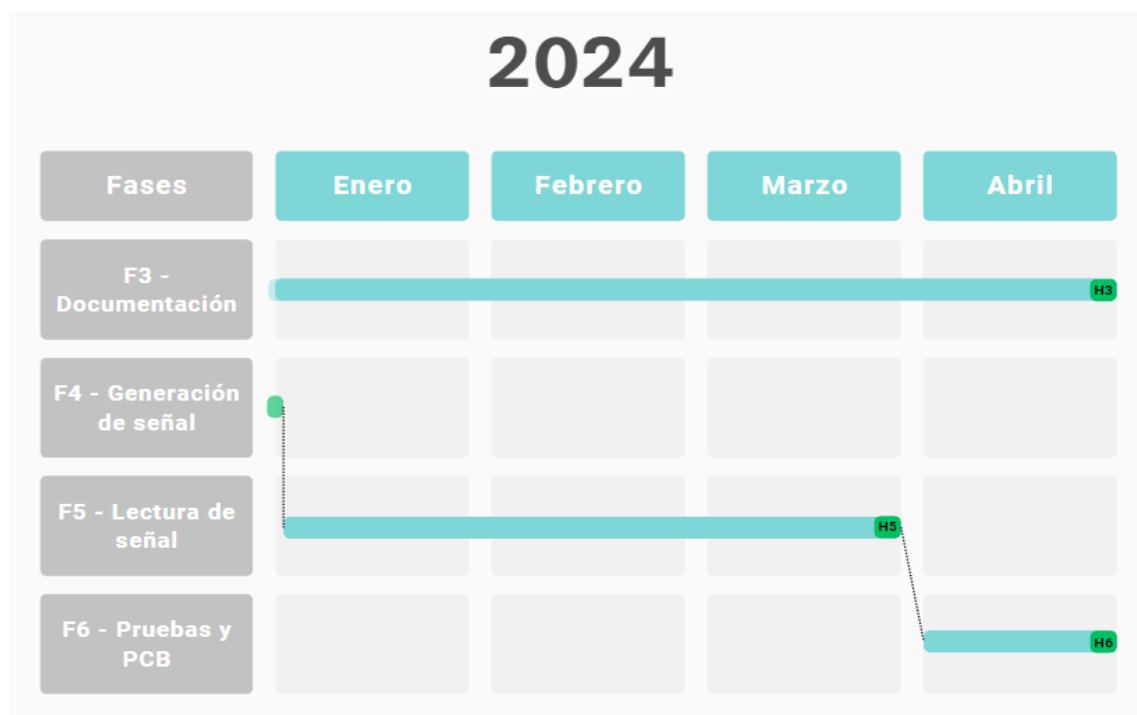


Figura 11. Diagrama de Gantt Planificación 2024

Y la distribución en cada fase sería la siguiente:



Figura 12. Diagrama Gantt Fase 1

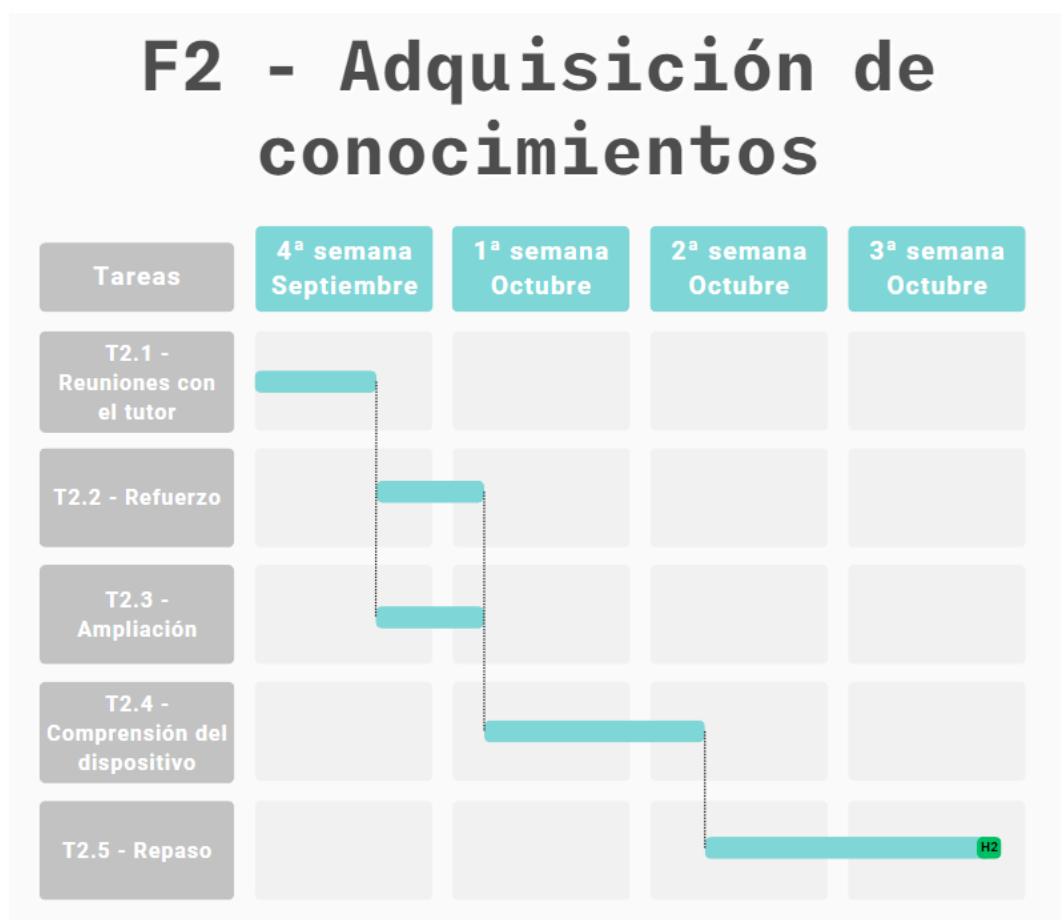


Figura 13. Diagrama Gantt Fase 2

F3 - Documentación

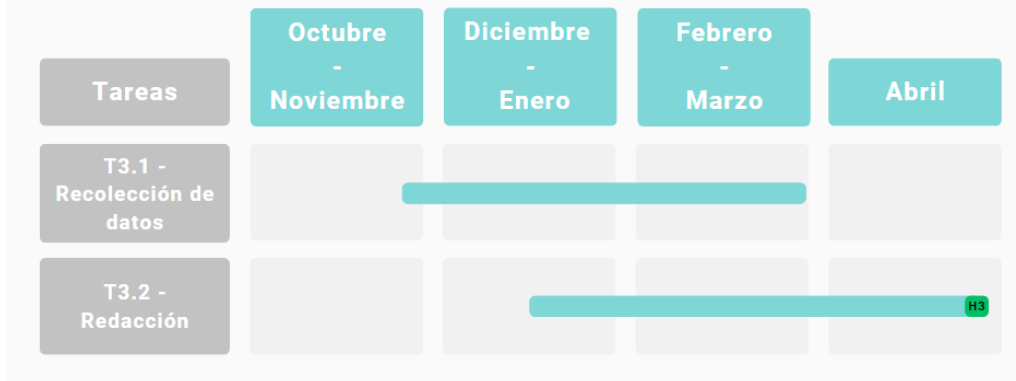


Figura 14. Diagrama Gantt Fase 3

F4 - Generación de señal

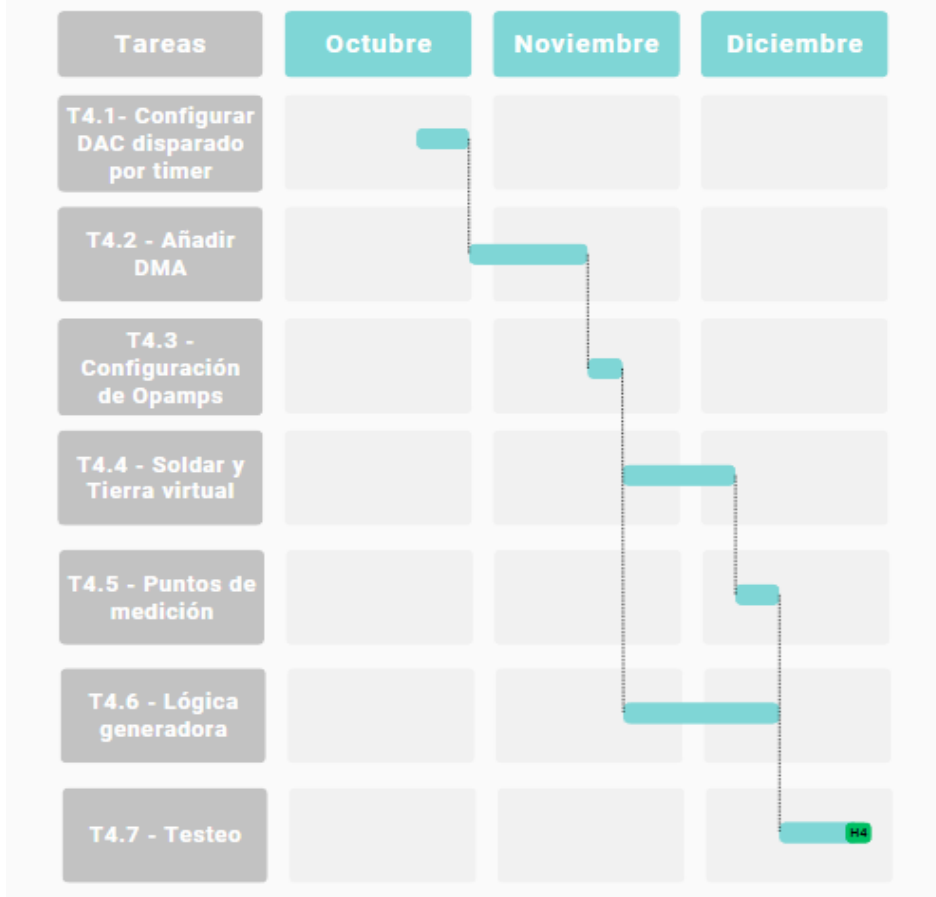


Figura 15. Diagrama Gantt Fase 4

F5 - Lectura de señal

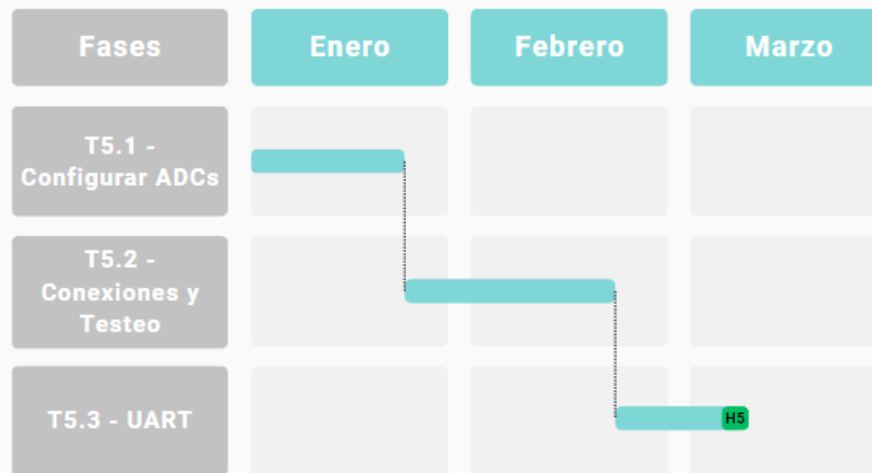


Figura 16. Diagrama Gantt Fase 5

F6 - Pruebas y PCB



Figura 17. Diagrama Gantt Fase 6

Tabla de tiempos estimados para cada fase con la asignación de horas:

Fase	Tiempo estimado	Distribución de horas
F1 - Estudio previo	2-4 semanas	40h (20h x 2 personas)
F2 - Adquisición de conocimientos	2-4 semanas	80h (40h x 2 personas)
F3 - Documentación	7 meses	100h (50h x 2 personas)
F4 - Generación de señal	3 meses	200h (100h x 2 personas)
F5 - Lectura de señal	3 meses	100h (50h x 2 personas)
F6 - Pruebas y PCB	2-4 semanas	90h (40h x 2 personas)

Tabla de tiempos más detallada de la distribución de horas por tarea

Tarea	Distribución de horas
F1 - Estudio previo	
T1.1 - Estudio previo	26h (13h x 2 personas)
T1.2 - Elección del proyecto	4h (2h x 2 personas)
T1.3 - Definición del proyecto	10h (5h x 2 personas)
F2 - Adquisición de conocimientos	
T2.1 - Reuniones con el tutor	8h (4h x 2 personas)
T2.2 - Refuerzo	10h (10h x 1 persona)
T2.3 - Ampliación	10h (10h x 1 persona)
T2.4 - Comprensión del dispositivo	30h (15h x 2 personas)
T2.5 - Repaso	22h (11h x 2 personas)

F3 - Documentación	
T3.1 - Recolección de datos	30h (15h x 2 personas)
T3.2 - Redacción	70h (35h x 2 personas)
F4 - Generación de señal	
T4.1 - Configurar DAC disparado por timer	20h (10h x 2 personas)
T4.2 - Añadir DMA	36h (18h x 2 personas)
T4.3 - Configuración de Opamps	10h (5h x 2 personas)
T4.4 - Soldar y Tierra virtual	50h (25h x 2 personas)
T4.5 - Puntos de medición	4h (2h x 2 personas)
T4.6 - Lógica generadora	50h (25h x 2 personas)
T4.7 - Testeo	30h (15h x 2 personas)
F5 - Lectura de señal	
T5.1 - Configurar ADCs	30h (15h x 2 personas)
T5.2 - Conexiones y testeo	50h (25h x 2 personas)
T5.3 - UART	20h (10h x 2 personas)
F6 - Pruebas y PCB	
T6.1 - PCB	30h (15h x 2 personas)
T6.2 - Pruebas	40h (25h x 2 personas)
T6.3 - Informe	20h (10h x 2 personas)

2.4.3. Distribución de trabajo

En la siguiente matriz se pretende resumir la asignación del trabajo de los dos participantes del proyecto, de aquí en adelante nos referiremos a Salvador Meléndez Muñoz como “Salvador” y a Emilio Silvestre Mérida como “Emilio”.

Tarea	Distribución del trabajo
F1 - Estudio previo	
T1.1	Ambos nos encargamos de buscar información sobre posibles trabajos.
T1.2	Ambos participamos activamente en la decisión sobre qué proyecto abordar, con la asesoría del profesor.
T1.3	Ambos propusimos los límites y el alcance del proyecto.
F2 - Adquisición de conocimientos	
T2.1	Ambos participamos en las reuniones de orientación del profesor.
T2.2	Salvador fue el encargado de buscar información en asignaturas previas.
T2.3	Emilio fue el encargado de ampliar la información buscando en otros medios.
T2.4	Ambos participamos en el estudio del dispositivo para conocer sus características.
T2.5	Ambos trabajamos en la puesta en común de los conocimientos.
F3 - Documentación	
T3.1	Ambos guardamos los datos obtenidos por cada uno durante la realización del proyecto.
T3.2	Ambos pusimos en común los datos de la tarea anterior.

	A la hora de redactar Salvador fue el encargado de redactar los apartados con mayor fundamento teórico y Emilio de redactar aquellos puntos que están relacionados con la organización del proyecto y sus fases. A excepción del apartado del desarrollo del sistema de esta memoria donde ambos trabajamos conjuntamente.
F4 - Generación de señal	
T4.1	Emilio fue el encargado de implementar la configuración y probar el DAC. Salvador fue el encargado de crear la LUT y supervisar las pruebas.
T4.2	Emilio fue el encargado de probar el DMA sin éxito y ambos trabajamos en la solución de errores para acabar implementado DMA.
T4.3	Salvador fue el encargado de configurar los Opamps y ambos realizamos las pruebas pertinentes.
T4.4	Salvador fue el encargado de soldar la mayoría de componentes y Emilio fue el encargado de soldar la tierra virtual específicamente en la placa de prototipado.
T4.5	Ambos participamos en la soldadura de los puntos de medición.
T4.6	Emilio fue el encargado de implementar las funciones para la generación de señal y ambos participamos en el arreglo y testeo de las funciones.
T4.7	Ambos participamos en el testeo del sistema generador.
F5 - Lectura de señal	
T5.1	Salvador fue el encargado de configurar los ADCs y Emilio fue el encargado de inicializarlos. Ambos buscamos información y ejemplos sobre los ADCs
T5.2	Salvador fue el encargado de realizar las conexiones con la supervisión de Emilio. Ambos participamos en el testeo de la implementación

T5.3	Ambos realizamos la implementación de la UART bajo la supervisión del tutor.
F6 - Pruebas y PCB	
T6.1	Salvador fue el encargado de realizar el esquemático y el logo de la PCB. Emilio fue el encargado de seleccionar las huellas y el enrutado de las pistas con el apoyo del tutor.
T6.2	Ambos realizamos la selección de las resistencias a probar y participamos en la realización de las pruebas.
T6.3	Ambos añadimos el material correspondiente a los resultados de las pruebas de los que disponíamos.

2.4.4. Matriz de Riesgo

A continuación, se muestra una tabla que detalla todos aquellos acontecimientos que pueden retrasar la ejecución del proyecto, dichos acontecimientos se clasifican según su posibilidad de producirse a lo largo de todo el proyecto en:

- Poco probable
- Probable
- Muy probable

También se clasifican según su impacto en la ejecución del proyecto en:

- Leve
- Medio
- Grave
- Crítico

Acontecimiento	Probabilidad	Impacto	Mitigación
Baja por enfermedad puntual de algún miembro	Poco probable	Leve	Posponer las tareas asignadas
Imposibilidad de trabajar por motivos personales	Poco probable	Leve	Posponer las tareas asignadas
Errores desconocidos o comportamientos inesperados del sistema que impidan continuar	Probable	Medio	Acudir a una reunión con el tutor
Malfuncionamiento de un componente	Poco probable	Medio	Adquirir otro componente
Falta de tiempo para realizar alguna tarea	Probable	Grave	Replanificar el proyecto
Cancelación de reunión por motivos del profesor	Poco probable	Leve	Posponer la reunión
Desconocimientos sobre el funcionamiento de alguna parte del sistema o implementación	Muy probable	Medio	Acudir a una reunión con el tutor

Comienzo de prácticas en empresa	Probable	Crítico	Replanificar el proyecto
Solapamiento de trabajo con asignaturas pendientes	Probable	Medio	Posponer las tareas
Fallo en la planificación inicial	Poco probable	Grave	Replanificar el proyecto

Esta tabla se tendrá en cuenta para resolver los problemas que vayan surgiendo a lo largo del proyecto, poniendo especial atención en aquellos que sean críticos o que sean muy probables.

3. DESARROLLO DEL PROYECTO

3.1. Fundamentos del Sistema Microcontrolador

Un **convertidor analógico-digital (ADC)** es un dispositivo capaz de convertir una entrada analógica en su correspondiente señal digital mediante una codificación binaria, generalmente.

Funciona a través de un primer muestreo de la señal en intervalos periódicos para generar una serie de puntos a los que posteriormente se les dará un valor digital y se codificarán en binario.

Existen varios tipos de ADCs, aunque el conversor de aproximaciones sucesivas es el que más se suele usar para obtener una mejor flexibilidad, precisión y eficiencia en el diseño. En este tipo de ADCs, se parte de un voltaje de referencia que será comparado con la señal de entrada mediante un comparador estableciendo así una aproximación del rango de voltaje seleccionado. En cada comparación, esta aproximación se irá guardando en el Registro de Aproximaciones Sucesivas, SAR. La idea es ir comparando si la señal es mayor o menor que el voltaje de referencia y, en función de eso, ir codificando la señal sucesivamente.

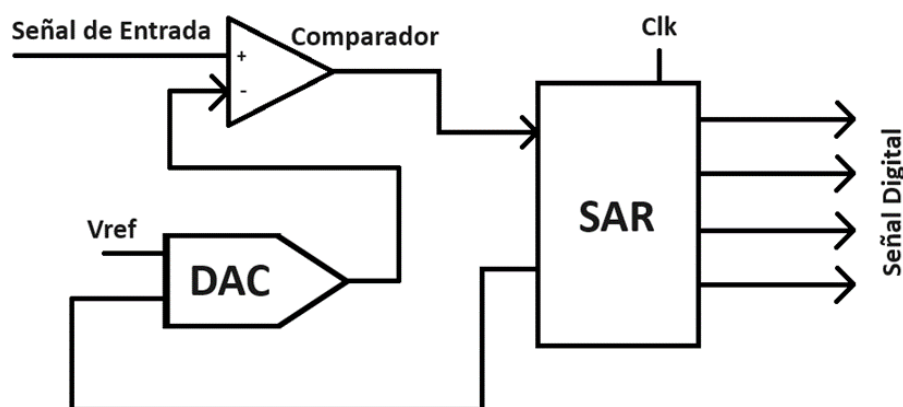


Figura 18. Estructura interna ADC

Un **convertidor digital-analógico (DAC)** es un dispositivo electrónico capaz de convertir un valor binario en una señal analógica de voltaje. En este tipo de conversiones, se parte de una secuencia de bits que será convertida a una correspondiente señal analógica.

Los DAC se utilizan en multitud de aplicaciones, algunas de las más habituales son: Sistemas de audio digital, comunicaciones, control de motores, señales de vídeo, control de procesos, etc.

Hay diversos tipos de DAC, por lo que los conceptos más importantes a tener en cuenta sobre los DAC son:

- **Resolución:** Depende totalmente del número de bits de la señal de partida. A mayor cantidad de bits, mayor exactitud en la resolución.
- **Precisión:** Para poder determinar la fiabilidad de un DAC, los fabricantes proponen los siguientes métodos:
 - Error de escala: Conociendo el valor teórico de salida del DAC, este error mide la desviación máxima de la salida del DAC con respecto a su valor teórico. Normalmente se suele indicar en porcentaje.
 - Error de linealidad: Se trata de la mayor desviación que puede tomar la salida del DAC frente a incrementos de voltaje.
- **Tiempo de respuesta:** Hace referencia al periodo necesario para que la señal resultante del DAC pase a escala completa cuando la señal de entrada pasa de ser todo 0's a todo 1's.
- **Voltaje de balance:** Cuando todas las entradas de bits son 0, al voltaje de salida del DAC se le denomina voltaje de balance. En este escenario, ese voltaje debería ser 0, pero existen pequeñas imperfecciones en el DAC que dan lugar a la aparición de un voltaje de salida muy pequeño incluso cuando la entrada del DAC es todo 0's. Este error se conoce como Error de Balance y se expresa como un error de escala. De modo que este voltaje hace referencia a un porcentaje del rango total de voltaje que es capaz de producir un DAC.

El **Acceso Directo a Memoria (DMA)** se trata de un método que permite recibir y enviar datos directamente en memoria sin tener que pasar por la CPU. El uso de DMA mejora la eficiencia del sistema haciendo que la transferencia de datos entre la memoria y los periféricos sea independiente de la CPU pudiendo ésta realizar otras tareas. Aunque sí es necesario el uso del bus del sistema.

Cada vez que se realice una transferencia por DMA se tienen 3 etapas:

1. **Inicialización de la transferencia**

En esta primera etapa la CPU lleva a cabo la configuración de controlador de DMA, **DMAC**, estableciendo ciertas señales de control AR (dirección de memoria) y WC (nº palabras a transferir), así como ciertos parámetros como el nº canal u otros elementos de control.

Una vez realizada la configuración, la CPU vuelve a sus tareas y se despreocupa de la transferencia.

2. **Transferencia**

Una vez configurado el DMAC, comienza la transferencia DMA en la que la CPU continúa haciendo otras tareas y el DMAC controla los accesos a memoria y acceso a los dispositivos.

El dispositivo enviará una petición de DMA al DMAC y este activará la línea de petición asumiendo el control del bus del sistema.

Una vez transferida una palabra, se irán actualizando los registros de las señales de control AR y WC, de tal forma que cuando WC = 0, la transferencia habrá terminado y el DMAC enviará a la CPU una interrupción. Además, liberará el bus del sistema y le devolverá el control a la CPU.

3. Finalización de la transferencia

La CPU atenderá la interrupción de la finalización de la transferencia.

Un **amplificador operacional (OPAMP)** es un dispositivo electrónico cuya función se fundamenta en la amplificación de señales. Está formado por varios transistores y resistencias conectadas entre sí y consta de dos entradas, una salida y sus terminales de alimentación.

En cuanto a sus entradas:

- **Entrada Inversora:** Denominada inversora si la señal de entrada es mayor, la salida del OPAMP se verá disminuida.
- **Entrada No Inversora:** Denominada no inversora si la señal de entrada es mayor, la salida del OPAMP también tenderá a ser mayor.

La configuración de estas entradas, definirán el comportamiento del circuito.

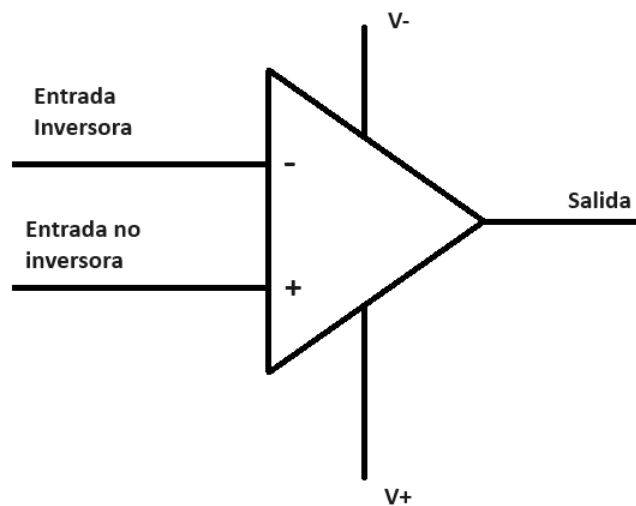


Figura 19. OPAMP

OPAMP en Lazo Abierto (Standalone)

Este tipo de configuración de los OPAMPs parte de que la ganancia se ajusta en base a la diferencia del valor de las entradas. Esta ganancia suele estar ajustada a un valor muy grande y se corresponde con la salida máxima obtenible por el OPAMP. De modo que una pequeña variación entre la tensión de entrada dará como resultado una tensión de salida cercana a la tensión de alimentación.

Este tipo de configuración se suele utilizar comúnmente en comparadores. Si la entrada inversora se conecta directamente a la tierra o a través de una resistencia y salida será la de la máxima tensión positiva de alimentación; si V_{in} es negativo, la salida será el valor negativo de alimentación. El amplificador operacional funciona como comparador porque no hay realimentación desde la salida a la entrada.

OPAMP en Lazo Cerrado (Follower)

Al tratarse de un lazo cerrado, de alguna forma se está realimentando el circuito. En esta configuración se mantiene una ganancia constante siendo muy común la realimentación en la entrada negativa del OPAMP. Al obtener una ganancia más pequeña la salida, esto permite que sea más precisa y controlable. Además, la realimentación negativa afecta positivamente dando lugar a una mayor estabilidad del sistema.

Amplificador de Instrumentación

Un **amplificador de instrumentación** es un dispositivo creado a partir de amplificadores operacionales cuya función es de amplificación diferencial. Se puede establecer una ganancia con muy alta precisión, permitiendo así ajustar una ganancia del circuito sin tener que modificar demasiados valores.

Este dispositivo cuenta con impedancias de entradas muy altas, pues se conectan a las entradas no inversoras de los OPAMPs que lo conforman. Además de una ganancia ajustable mediante una única resistencia.

Al tratarse de una amplificación diferencial, la operación que realiza este amplificador es la resta de las señales de su entrada, siendo esta la amplificada mediante un factor de ganancia para eliminar o reducir el ruido existente entre ambas señales. La capacidad de rechazar el ruido de ambas entradas es una

característica importante de los amplificadores instrumentales. Esto se logra mediante métodos de diseño que cancelan o reducen el efecto del ruido que afecta a ambas entradas. Se produce entonces un Rechazo de Modo Común.

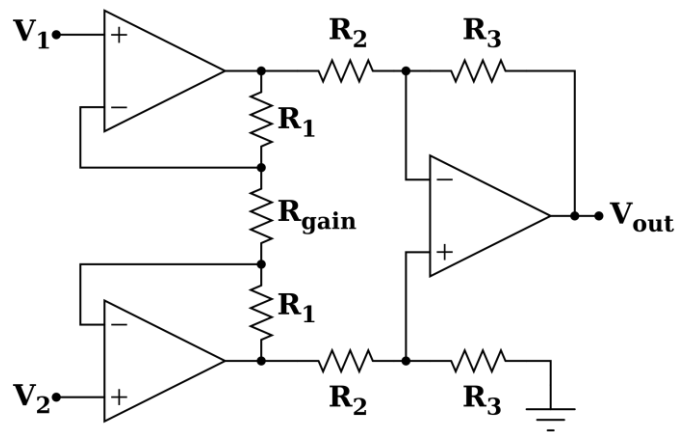


Figura 20. Estructura interna Amplificador Operacional

Tierra Virtual

Una **tierra virtual** es un nodo referencia que se encuentra dentro de un circuito amplificador que se mantiene constantemente a un potencial de referencia sin estar conectado directamente a la tierra física del circuito.

Es muy usada en un contexto de amplificadores cuando se quiere obtener la amplificación de la diferencia entre dos señales de entrada al circuito. Por ejemplo, partiendo de un OPAMP configurado como amplificador en modo diferencial, se aplica un potencial de referencia en una de las entradas creando así una retroalimentación negativa. Al no estar conectada una de las entradas directamente a la tierra física del circuito, todo esto hará que la señal de salida resultante de la diferencia se amplifique y el OPAMP funcione independientemente de la tierra física.

Además, conlleva a la ventaja de obtener una mejor precisión en las mediciones o en las señales amplificadas resultantes.

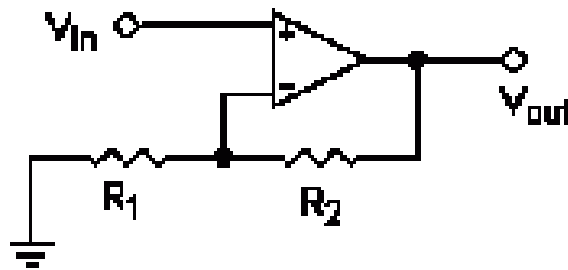


Figura 21. Estructura interna Tierra Virtual

Modos Standalone y Follower

Los modos **Standalone** y **Follower** son modos de configuración aplicados a los OPAMPs.

- **Modo Standalone:** El OPAMP actuaría como un amplificador de ganancia. En este caso no hay realimentación directa y lo que se consigue es una alta impedancia de entrada y una baja impedancia en la salida.
- **Modo Follower:** El OPAMP actuaría como un buffer de voltaje. En una de sus entradas se conectaría la señal de entrada mientras que la otra se vería afectada por el lazo de realimentación entre la salida del OPAMP y esa entrada.

Modos Single-Ended y Differential

Los modos **Single-Ended** y **Differential** son modos de configuración aplicados a los DACs y ADCs.

- **Modo Single-Ended:** Es un método menos costoso y más simple, pues solo es necesario un canal debido a que únicamente una señal es muestreada y convertida a su correspondiente digital o analógica. Puede ocasionar ruido y no mantener la integridad de la señal en entornos donde se requiera mucha precisión.
- **Modo Differential:** Es una opción más costosa al utilizar dos canales para las entradas y salidas. En este caso no solo se muestrea y se convierte una única señal, por lo que se mejora la situación de ruido y se puede obtener una mejor precisión y rendimiento del sistema.

3.2. Diseño del sistema

En este apartado se detalla la estructura y el uso que se le da a cada componente del sistema. Como se menciona en la introducción, el sistema consta de dos partes principalmente:

- Generador de señal
- Adquisición de señal

3.2.1. Generador de señal

La estructura general del generador de señal es la siguiente:

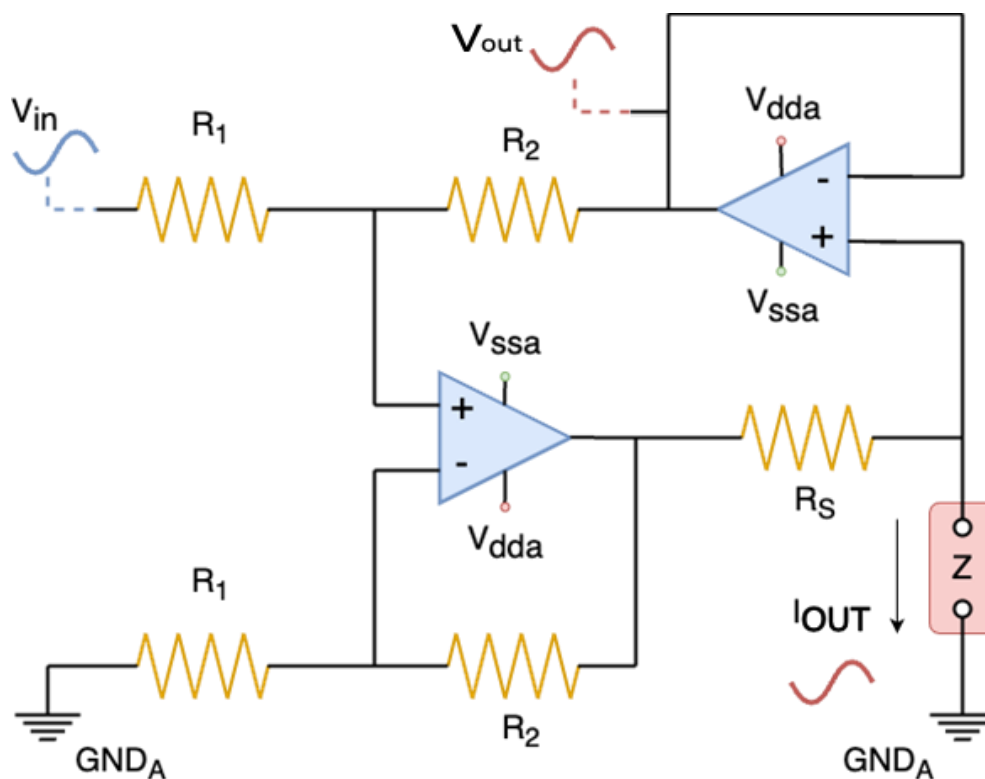


Figura 22. Estructura del Sistema Generador de Señal [1]

El funcionamiento de este circuito implementa la siguiente fórmula:

$$I_{out} = \frac{R_1}{R_2} * \frac{V_{in}}{R_s}$$

Donde V_{in} es la señal de salida del DAC inyectada a través de un Opamp en modo follower y I_{out} es la magnitud de la corriente de salida del circuito generador. Este circuito se comporta como una fuente de corriente controlada por voltaje.

En este circuito es importante que las resistencias estén emparejadas en valor.

La zona donde se va a medir la impedancia (Z) viene representada por un cuadro rojo, esta es la zona donde se implementará la parte de lectura del sistema.

Además, destacar que la tierra que estamos usando (GND_A) no es la tierra del sistema, si no una tierra virtual implementada de la siguiente manera:

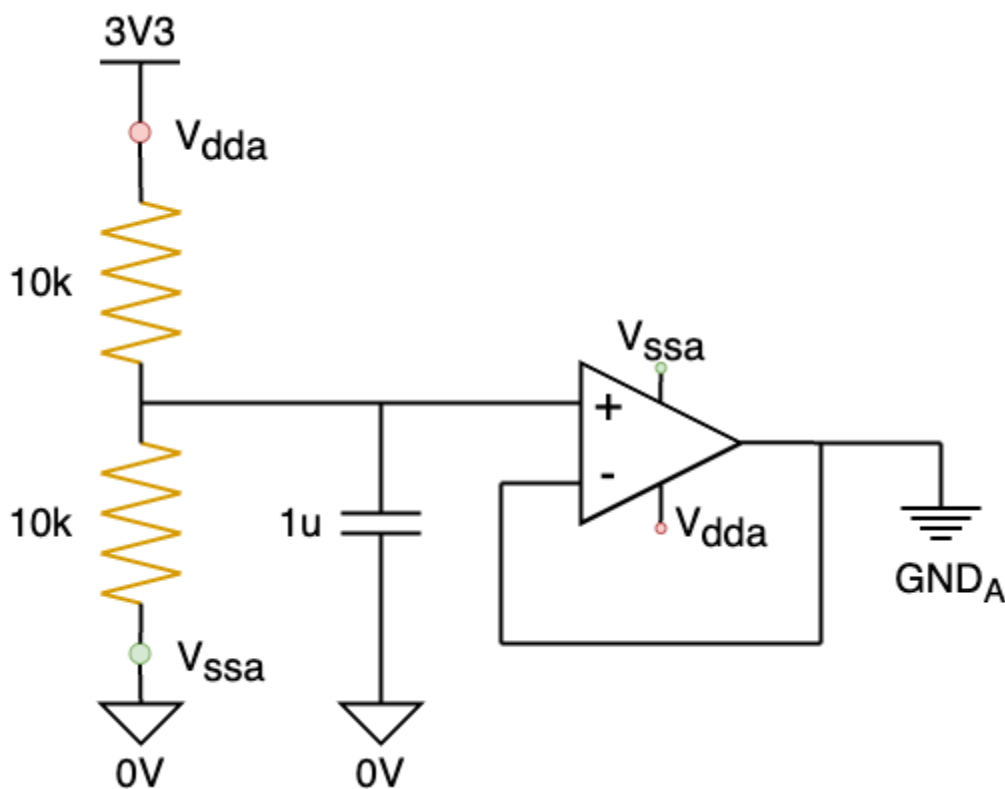


Figura 23. Estructura interna Tierra Virtual

Esta tierra virtual está alimentada al igual que los Opamps del sistema por la alimentación del microcontrolador e implementa un divisor de tensión para generar una referencia a medio voltaje ($V_{cc}/2$) y un condensador para estabilizar el nivel seguido de un Opamp en configuración seguidora para actuar como buffer al tener una alta impedancia de entrada. Este sistema generará una tierra que usarán los componentes del sistema como referencia.

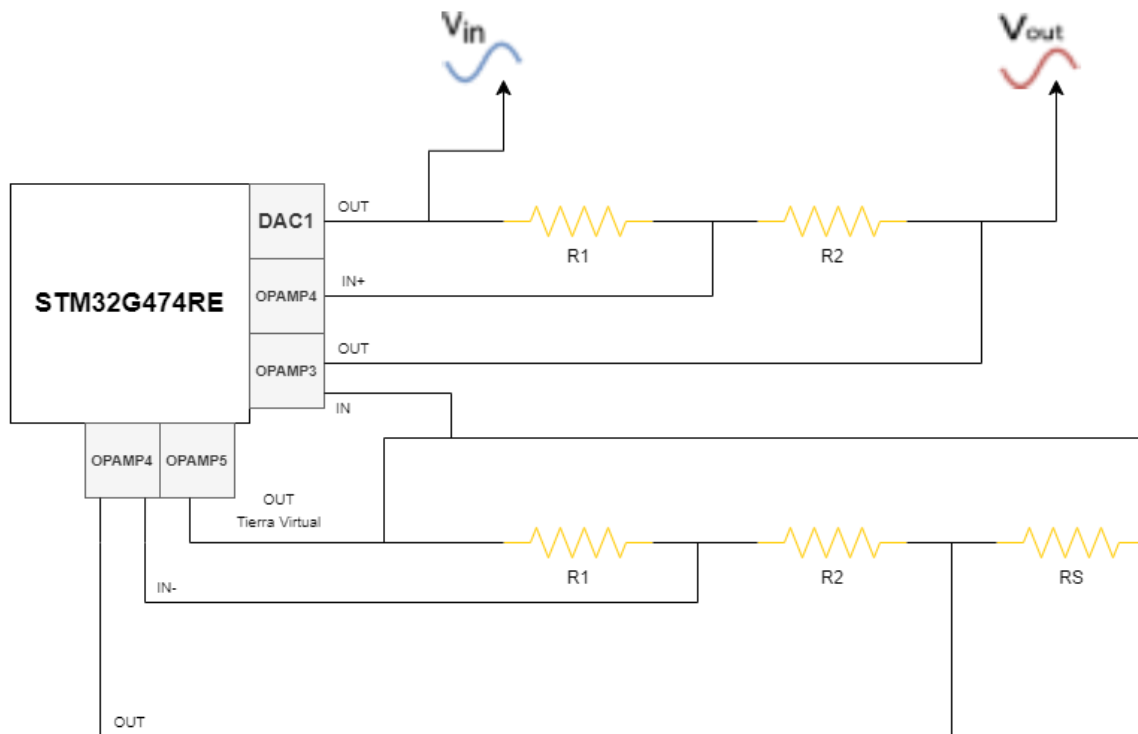


Figura 24. Esquema de la implementación del generador de señal

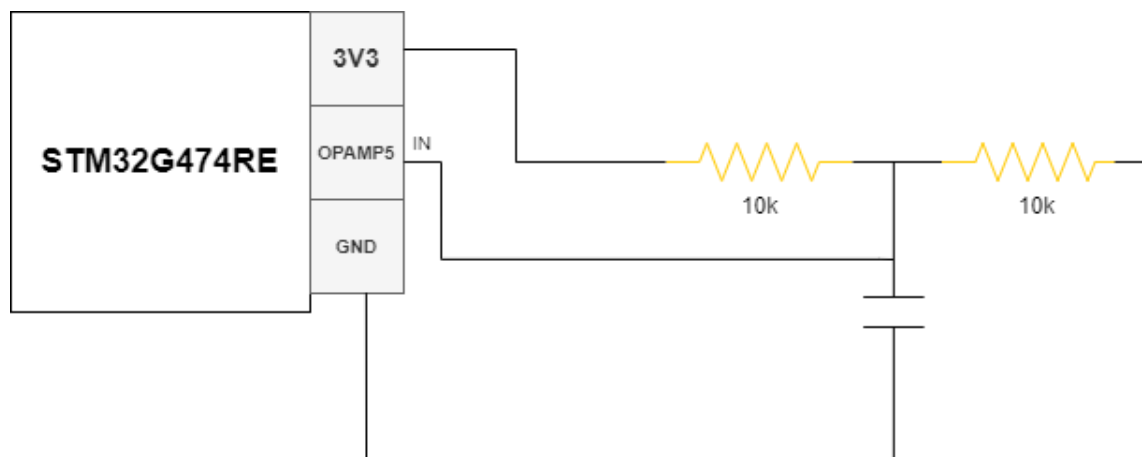


Figura 25. Esquema de la implementación de la tierra virtual

3.2.2. Lector de señal

La parte del lector de señal consiste en dos ADCs en single-ended conectados tanto a V_{in} como a V_{out} que se encargarán de medir la señal de entrada al sistema generador y la señal una vez se haya visto afectada por la muestra.

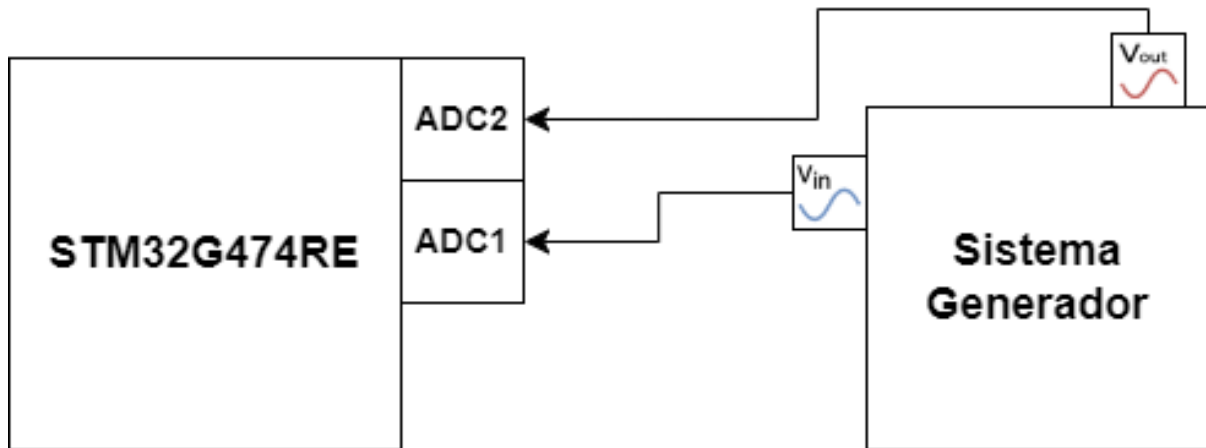


Figura 26. Esquema de la estructura del lector de señal

Esto es necesario para el cálculo de la impedancia y se transmitirán ambos buffers de los ADCs por la UART del dispositivo. Para conocer la impedancia se requerirá de un cálculo adicional no contemplado en este proyecto.

La impedancia puede ser calculada a través de la ley de Ohm:

$$Z = \frac{V_{out}}{I_{out}}$$

Para realizar este cálculo necesitaríamos trabajar con fasores, podemos obtener V_{out} del buffer de uno de los ADC, mientras que podemos usar el generador de señal para conocer I_{out} ya que atiende a la siguiente fórmula:

$$I_{out} = \frac{R1}{R2} * \frac{V_{in}}{R_s}$$

Donde conocemos el valor de R_s y V_{in} es obtenida del buffer del otro ADC.
Siendo el valor del fasor de la impedancia:

$$Z = \frac{V_{out} \cdot R_2 \cdot R_s}{R_1 \cdot V_{in}}$$

3.3. Desarrollo del sistema

En este apartado trataremos todo lo relacionado con la implementación del sistema y el procedimiento que hemos seguido en cada una de las tareas del proyecto, así como los problemas que hemos encontrado y solucionado.

3.3.1. F1: Estudio previo y elección del proyecto.

El hito correspondiente a esta fase es **H1- Proyecto seleccionado**.

T1.1- Estudio previo

Para esta tarea decidimos hacer una investigación previa de posibles proyectos que requieran de montaje hardware, entre todos ellos destacar la implementación de un sensor lidar con el objetivo de usarlo como escáner de entorno [7] o realizar un dispositivo capaz de medir parámetros del agua útiles en la acuariofilia.

Tras consultar con el tutor en una reunión orientativa, decidimos descartar la idea del lidar tras las recomendaciones de seguridad por parte del tutor y decidimos ampliar la investigación buscando información sobre la medición de parámetros del agua usando la impedancia.

Tras una segunda investigación encontramos algo de información sobre el uso de la impedancia en la medición de parámetros [8] pero solo mencionaba la medición del ph y no de otros componentes, decidimos buscar más contenido relacionado con la impedancia y encontramos que otro posible proyecto podría ser la implementación de una mano biónica con EMG (Electromiografía) e impedancia [9].

T1.2- Elección del proyecto

Tras una reunión de puesta en común con el tutor se llegó a la conclusión de que queríamos trabajar con la impedancia, pero no teníamos claro una aplicación final, por lo que se decidió implementar un dispositivo genérico capaz de medir impedancia sin un uso concreto.

T1.3- Definición del proyecto

Tras investigar el estado del arte de los dispositivos capaces de medir la impedancia decidimos definir el alcance del proyecto basado en las recomendaciones del tutor.

El proyecto iba a implementar un circuito previamente usado por el profesor, pero usando un microcontrolador relativamente nuevo (2021) del que no se había probado a realizar esta implementación, además se decidió implementar la mayor cantidad de componentes posibles usando el microcontrolador para abaratar costes y poder trabajar en profundidad con el microcontrolador.

El microcontrolador elegido fue el STM32G474RE debido a los conocimientos previos que teníamos de trabajar con el fabricante STMicroelectronics y a la disposición de este mismo en el departamento, no sin antes realizar una búsqueda comparativa con otros microcontroladores que tuviesen los componentes para implementar los requisitos ya mencionados.

Se acordó realizar durante el proyecto una implementación de una PCB y mantuvo la posibilidad de ampliar este proyecto y darle una aplicación final al medidor de impedancia.

3.3.2. F2: Obtención de conocimientos.

El hito correspondiente a esta fase es **H2- Conocimientos necesarios.**

T2.1- Reuniones con el tutor

Tras un par de reuniones con el tutor definimos una serie de conceptos base que era necesario tener claros, entre ellos estaban el funcionamiento de la familia del microcontrolador que vamos a usar (STM32G4), en concreto el funcionamiento y configuración de los ADCs, DAC's, Timers, DMA y los modos de los OPAMPs, con respecto a los componentes externos un multiplexor analógico (2:4) que se contempló para poder realizar varios caminos con distintas resistencias y un amplificador instrumental, con respecto a conocimiento teóricos los fasores, las señales senoidales, la medición en 2 y 4 puntos y la implementación y funcionamiento de la tierra virtual. También

decidimos consultar distintos métodos de soldadura de componentes de cara a diseñar un prototipo del sistema para la fase de pruebas.

T2.2- Refuerzo

Esta tarea se realizó en paralelo con la **T2.3** y se encargó en su totalidad Salvador. Se buscó en la documentación de varias asignaturas de años anterior entre ellas Sistemas Empotrados en Tiempo Real 1 y 2, Arquitectura de Computadores y Laboratorio de Desarrollo Hardware.

T2.3- Ampliación

Esta tarea se realizó en paralelo con la **T2.2** y se encargó en su totalidad Emilio. Debido a no poder encontrar toda la información de los temas necesario para la realización del proyecto en la documentación de asignaturas previas, decidimos ampliar las fuentes de conocimiento a través de diferentes medios web, principalmente la wiki y el foro oficial de STMicroelectronics, sitios de venta de componentes con sus datasheet correspondientes y algunos medios adicionales para buscar conceptos teóricos.

T2.4- Comprensión del dispositivo

En esta tarea nos centramos exclusivamente en comprender el microcontrolador con el que íbamos a trabajar, a partir del datasheet obtuvimos información de los componentes que tenía y que nos interesaban a la hora de realizar el proyecto entre ello destacar los siguientes timers ya mencionados anteriormente.

En nuestro caso, decidimos usar los timers de 32-bit por su capacidad de configuración y debido a que se recomiendan para usarlos junto a los DACs y ADCs.

También, revisamos ejemplos incluidos en el propio IDE que nos ayudaron a comprender como tendríamos que configurar los componentes e inicializarlos.

T2.5- Repaso

Pusimos en común todos los conocimientos adquiridos en las tareas previas con el tutor, analizamos los puntos clave y el tutor dio el visto bueno para comenzar con la implementación del sistema.

3.3.3. F3: Generación de documentos.

El hito correspondiente a esta fase es **H3- Memoria entregable**.

T3.1- Recolección de datos

De cara a la realización de la memoria, se acordó guardar los datos necesarios que estén directamente relacionados con el proyecto como el código del microcontrolador o el diseño de la PCB en un repositorio de GitHub compartido con el tutor [10] para facilitar su revisión y todos los datos no relacionados directamente con el proyecto como documentaciones adicionales y datasheets de componentes en un servidor de Discord.

T3.2- Redacción

La tarea de redacción corresponde a la elaboración de este mismo documento, se comenzó cuando se tuvo suficientes datos e información para poder aportar algo al documento, este documento recoge el desarrollo completo del proyecto “Medidor de impedancia” junto con la documentación teórica necesaria para comprender el proyecto.

3.3.4. F4: Generación de señal.

El hito correspondiente a esta fase es **H4- Prototipo de generador**

T4.1- Configurar un DAC disparado por timer

Para empezar, creamos un proyecto vacío en el STM32CubeIDE eligiendo nuestra placa de desarrollo STM32G474RET6 y pasamos directamente a configurar el DAC.

Para configurar el DAC encargado de generar la señal lo primero es definir que valores va a recorrer en la salida, a esta tabla de valores se le conoce como LUT (“Look Up Table”).

Para generar la LUT aplicamos un script de python que genera una secuencia senoidal entre -1 y 1.

```
import math
i = 0
n = 32
step = 2*math.pi/n
lut = []
while(i < n):
    lut.append(math.sin(i*step))
    i = i+1
print(lut)
```

Figura 27. Script de python para generar la LUT

Pero necesitamos adaptar la función para poder usarla en el microcontrolador, el DAC solo funciona con valores positivos, por lo tanto, la secuencia deberá tener un offset predefinido del mismo valor que la amplitud de la función que estamos buscando. Además, habrá que calcular el valor real de “i” en la función y no el valor normalizado entre -1 y 1, para ello, multiplicamos el valor porcentual por el valor máximo que es:

$$\frac{AMP}{\sin\left(\frac{(2 \cdot \pi)}{4}\right)}$$

La función quedaría de la siguiente manera:

```
import math
i = 0
n = 32
amp = 2048
step = 2*math.pi/n
lut = []
while(i < n):
    lut.append(math.sin(i*step)*amp/math.sin(2*math.pi/4)+amp)
    i = i+1
print(lut)
```

Figura 28. Script de python para generar LUT adaptada al DAC

Esta función genera una LUT que va desde 0 a 4095 formando una secuencia senoidal, el valor de amplitud máxima tolerada es la mitad de 12-bits (2048) ya que 0xFFFF (4095) es valor máximo del DAC de 12-bits del microcontrolador STM32G4.

El siguiente paso es activar y configurar los parámetros del DAC, para ello necesitamos cambiar el fichero .ioc del proyecto. En "Analog->DAC3" activamos la primera salida y ponemos como Trigger el Timer 3 que es el timer que usaremos para disparar las interrupciones que modificarán la salida del DAC3. Hemos decido usar el DAC3 ya que por recomendación del tutor añadimos un OPAMP a la salida del DAC que hará de buffer y el microcontrolador nos ofrece la posibilidad de realizar esta conexión de manera interna usando el DAC3 o el DAC4, ya que el DAC1 y DAC2 tienen conexiones externas.

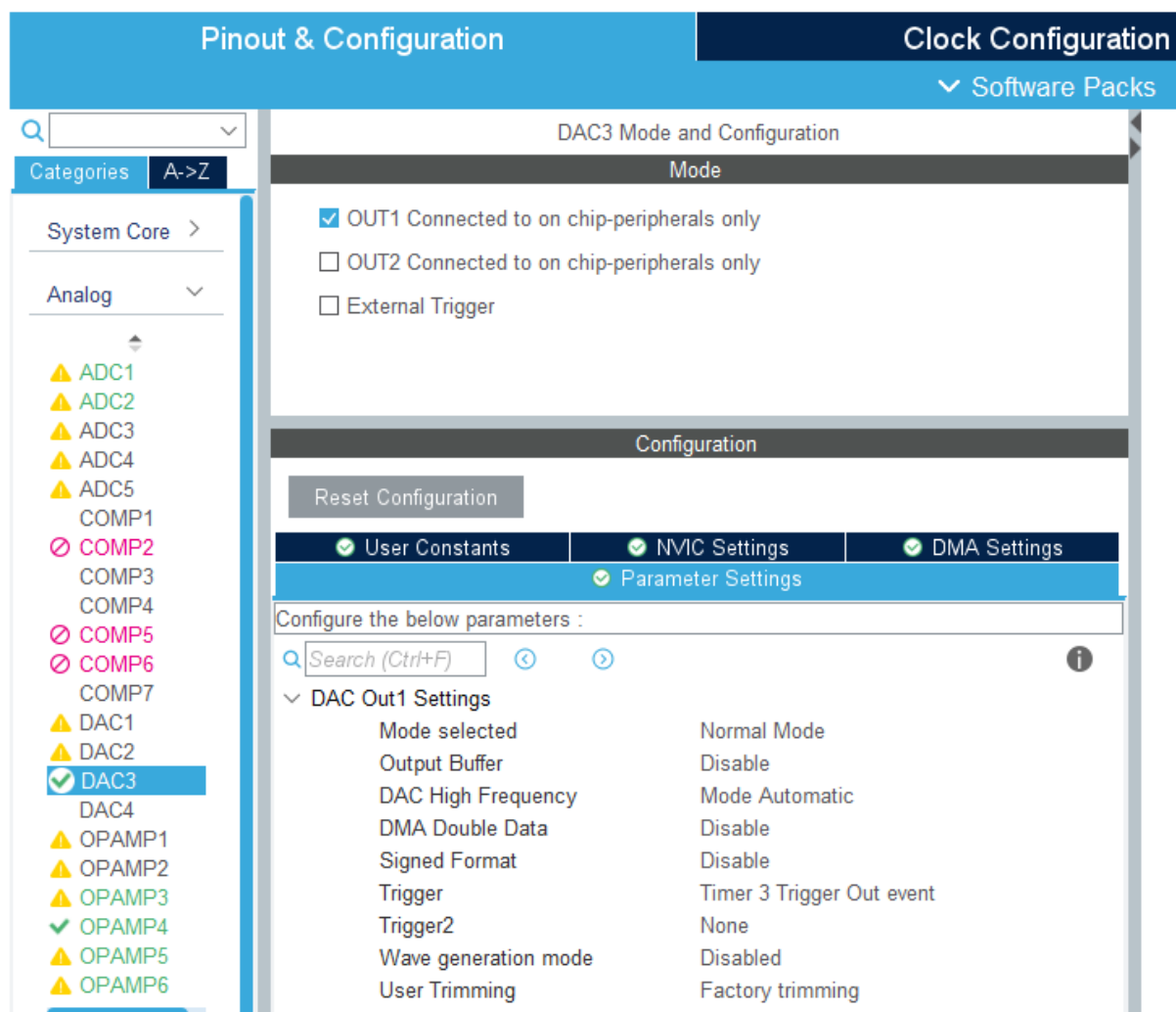


Figura 29. Configuración Parameter Settings DAC

Para configurar el Timer encargado del cambio de valores del sistema generador hemos decidido usar el Timer 3, es un timer que se encuentra recomendado para usar junto con el DAC3 debido a que ofrece una mayor facilidad de configuración para disparar las interrupciones.

Lo hemos configurado en el fichero .ioc en “Timers->TIM3” de la siguiente manera:

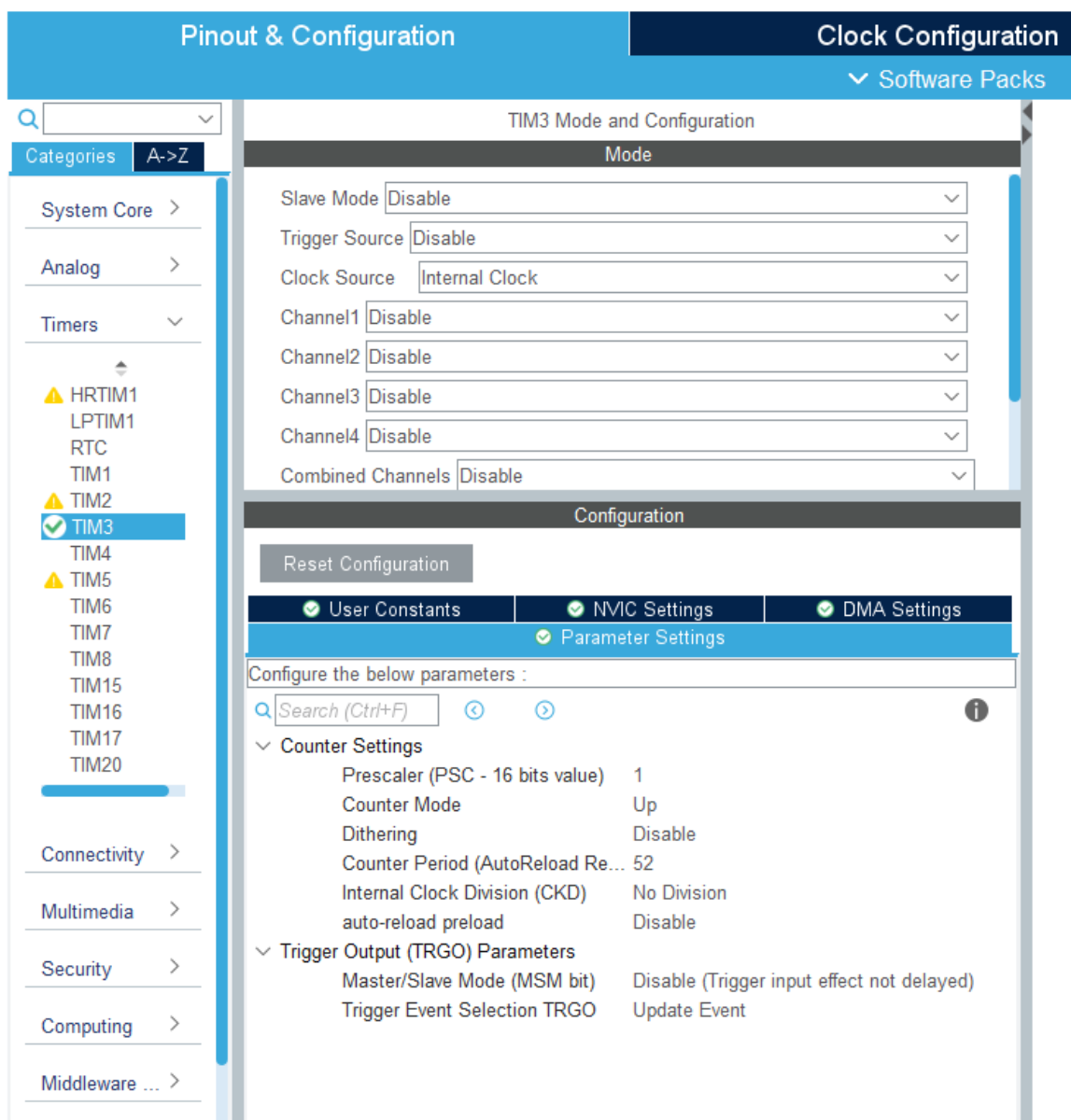


Figura 30. Configuración Parameter Settings Timer 3

Además, de activar las interrupciones globales en NVIC Settings. Mencionar que los valores de “Prescaler” y “Counter Period” son valores genéricos que modificaremos por código más adelante.

Quedaría por configurar el OPAMP encargado de la salida generadora externa. En este caso el OPAMP asociado al DAC3 es el OPAMP6:

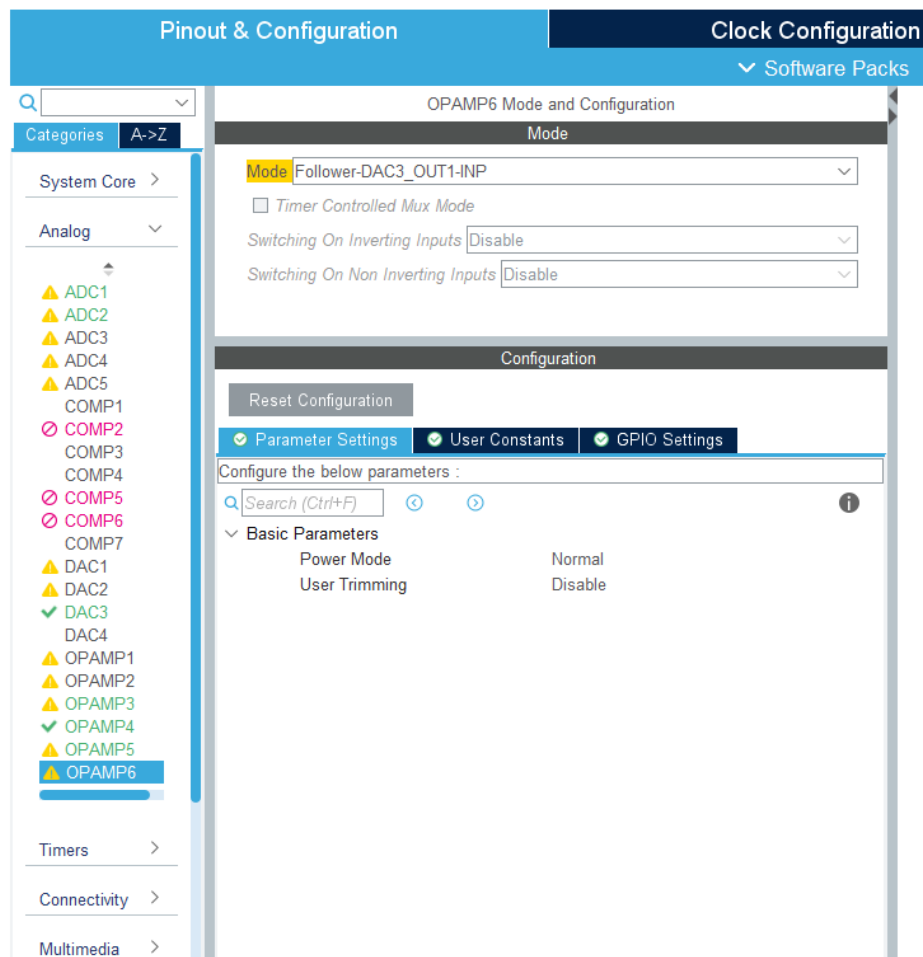


Figura 31. Configuración Parameter Settings OPAMP 6

Por último, usamos la librería HAL con las funciones “Start” para inicializar los componentes y la función “setValue” en la rutina de interrupción del timer para recorrer la LUT y cambiar el valor de salida del DAC.

Para estar seguros de que todo se encuentra configurado correctamente medimos a la salida del OPAMP6 que corresponde al pin PB11 y obtenemos el siguiente resultado en el osciloscopio:

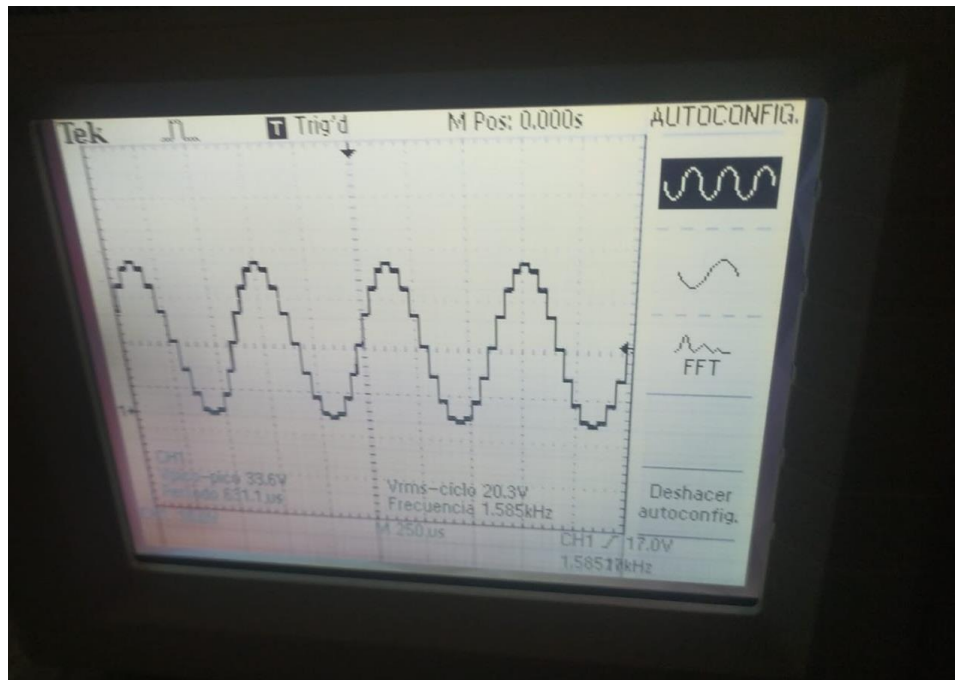


Figura 32. Señal senoidal resultante en osciloscopio

T4.2- Añadir DMA

Al principio para implementar el DMA configuramos en el .ioc el apartado DMA del DAC para poder usarlo, lo configuramos en modo circular, en tamaño de datos 16 bits (Word) y en prioridad Alta.

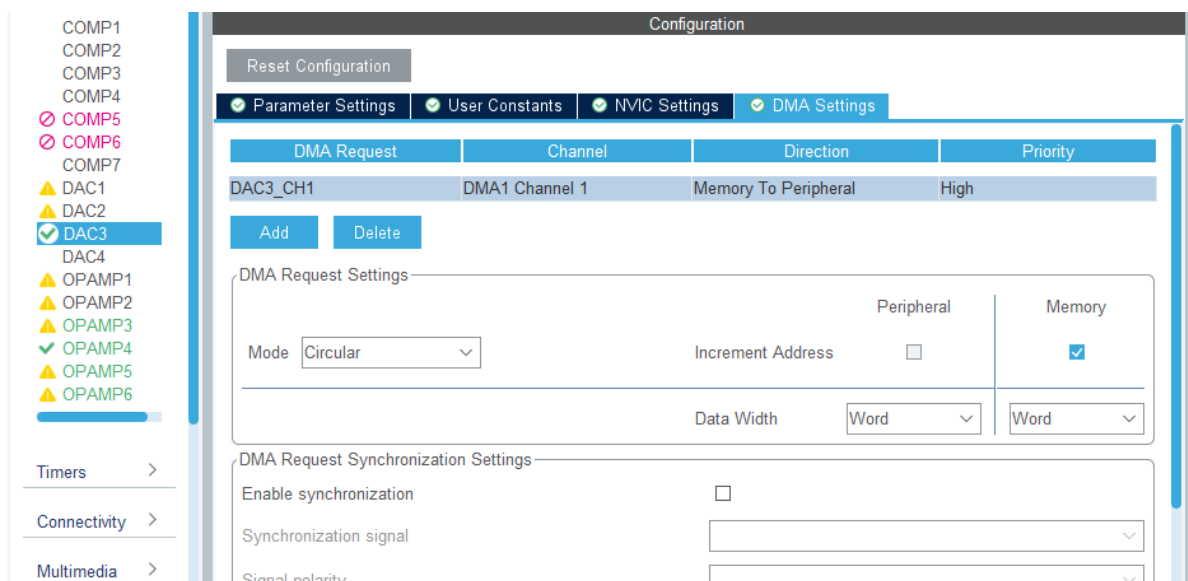


Figura 33. Configuración DMA Settings DAC

Una vez configurado y generado el código, tuvimos problemas a la hora de implementarlo, ya que seguimos nuestra experiencia tratando con rutinas de interrupción e implementamos el código de la función “setValue” del DAC en la rutina de interrupción del DMA, esto no funcionó y por recomendación del tutor descubrimos que la mejor manera de abordar este problema era trabajar con los ejemplos que viene incluidos en el IDE STM32CubeIDE.

En el caso de la implementación del DMA el ejemplo es el siguiente:

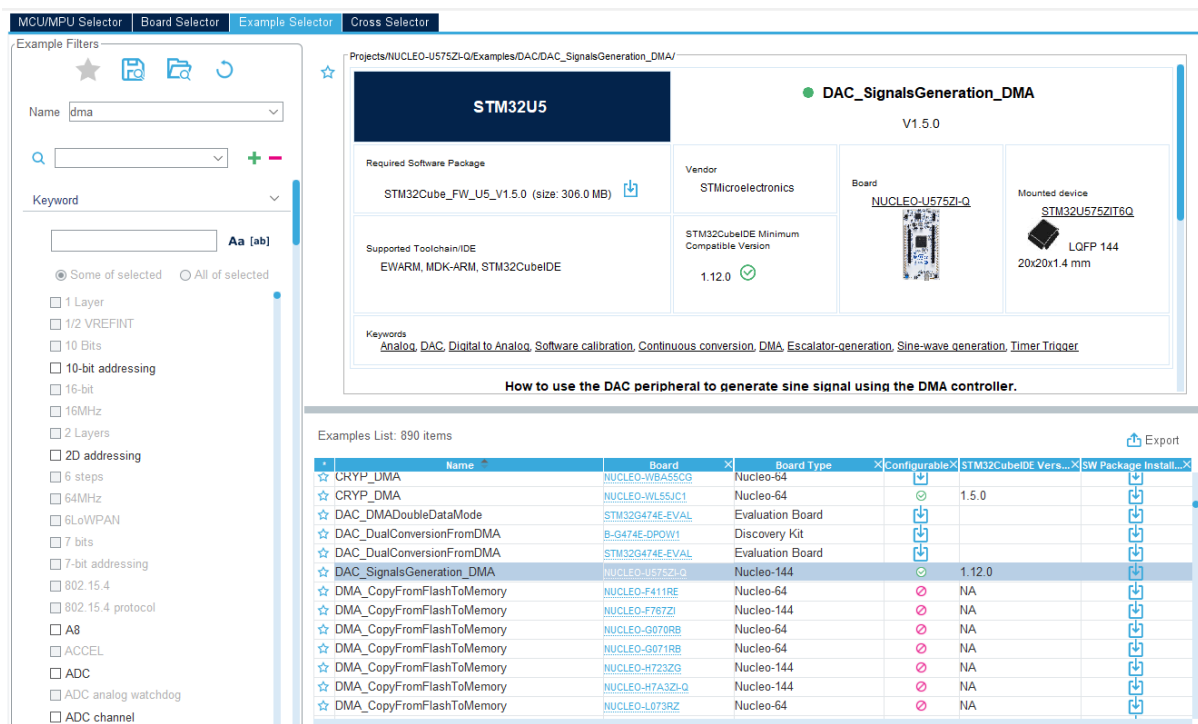


Figura 34. Proyecto ejemplo de DAC con DMA usado como modelo

A partir de este ejemplo pudimos confirmar que nuestra configuración era la correcta y crear nuestra propia implementación del DAC con DMA usando la siguiente inicialización disponible en el código fuente final:

```
//DAC
if(HAL_DACEx_DualSetValue(&hdac3, DAC_ALIGN_12B_R, 0, 0) != HAL_OK) {
    Error_Handler();
}
if(HAL_DACEx_DualStart_DMA(&hdac3, DAC_CHANNEL_1, (uint32_t*)lut, lut_size, DAC_ALIGN_12B_R) != HAL_OK) {
    Error_Handler();
}

HAL_Delay(10); // Wait 10ms so that the signal is stable.
```

Esta implementación se encuentra en el fichero main, después de las inicializaciones generadas automáticamente y fuera del bucle while(1), la primera función se encarga de poner los registros del DAC a 0, la segunda función recorrerá la LUT donde se encuentran los valores generados en python que han sido introducidos manualmente y almacenados en la variable “lut” usando DMA. También se define una variable global “lut_size” con valor 32 actualmente.

T4.3- Configuración Opamps del sistema

El siguiente paso para la implementación del sistema generador una vez tenemos lista la parte de la entrada (Vin) sería implementar el resto de los componentes del sistema. Primero, nos centramos en aquellos que dependen del microcontrolador, es decir, los Opamps. Para ello, necesitaremos 2 Opamps para el circuito del sistema, usaremos los Opamps 3 y 4.

Usaremos estos Opamps concretamente debido a un problema de incompatibilidad con otros componentes ya configurados no mencionado por el IDE que descubrimos al intentar implementar los Opamps 1 y 2 que no funcionaron correctamente y encontramos un mensaje detallando el error en el foro oficial de STMicroelectronics por otros usuarios.

La configuración del Opamp 3 será en modo Follower de la siguiente manera:

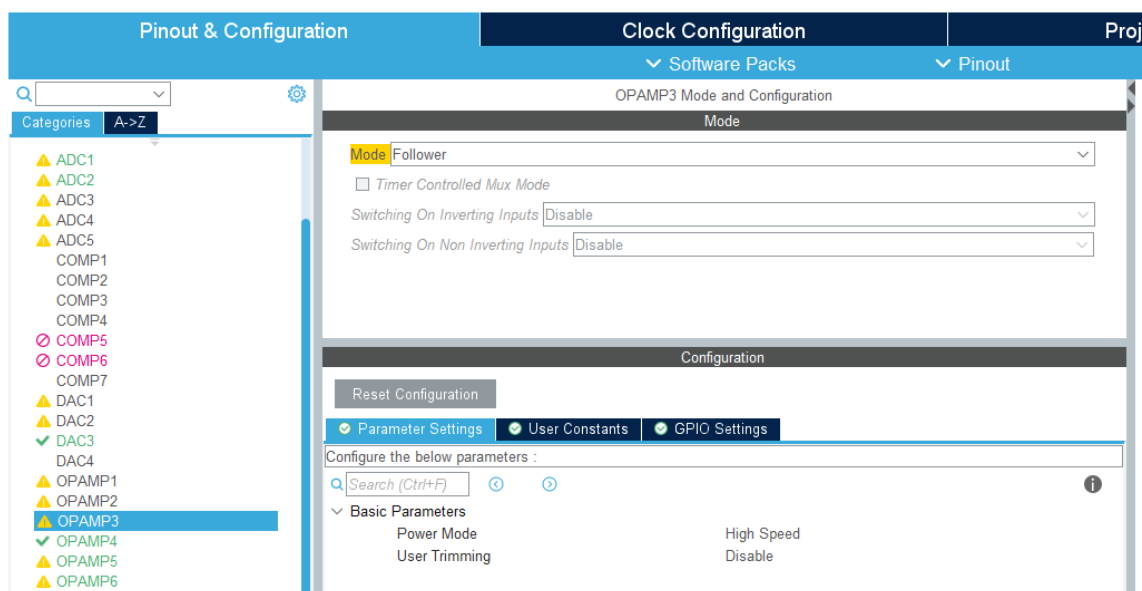


Figura 35. Configuración Parameter Settings OPAMP 3

Y la del Opamp4 en modo Standalone:

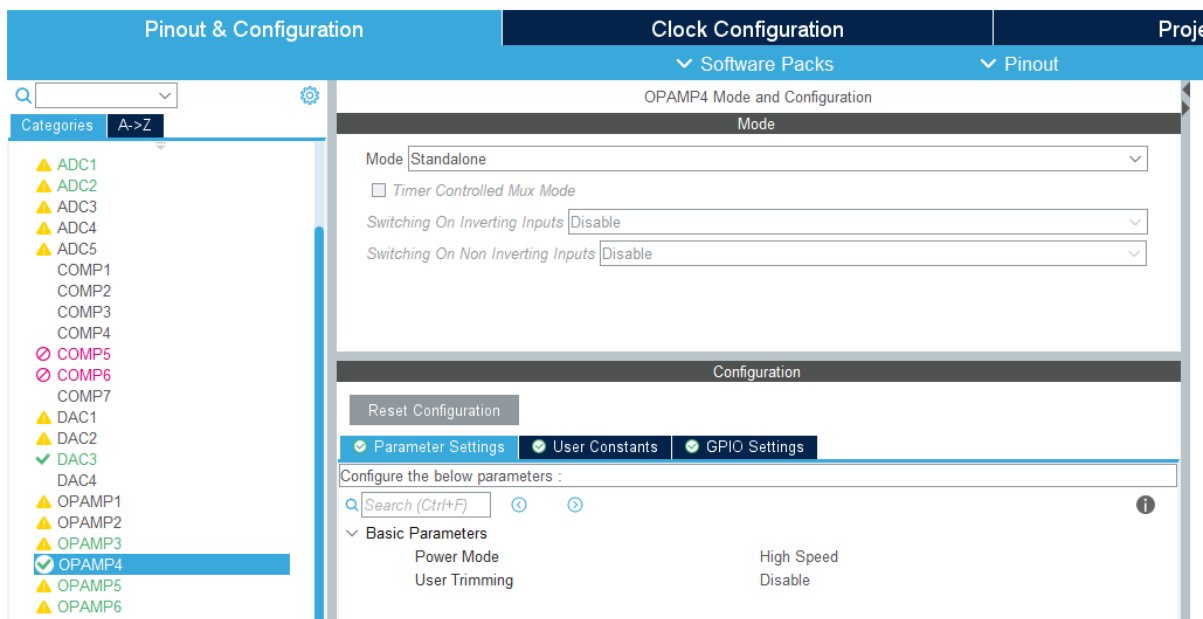


Figura 36. Configuración Parameter Settings OPAMP 4

Tras generar código lo que falta es añadir las inicializaciones en el código del fichero main.c junto a las del DAC de la tarea anterior:

```
//OPAMS
if(HAL_OK != HAL_OPAMP_Start(&hopamp6)) { Error_Handler(); }
if(HAL_OK != HAL_OPAMP_Start(&hopamp4)) { Error_Handler(); }
if(HAL_OK != HAL_OPAMP_Start(&hopamp3)) { Error_Handler(); }
```

T4.4- Soldar y tierra virtual

Para esta tarea necesitaremos un Opamp adicional para crear la tierra virtual que usará el sistema como referencia. Este Opamp es necesario debido a que el opamp estabiliza el nivel de tensión para poderlo utilizar en el resto del circuito. La utilización directa del divisor de tensión genera un problema al alterar el circuito que estamos queriendo implementar. Se utiliza el seguidor para aislar el divisor de tensión del resto del circuito. En una primera implementación no tuvimos en cuenta este detalle y tuvimos que volver a realizar la implementación correctamente.

Este Opamp también tuvo que configurarse e inicializar como el Opamp 3 en modo Follower, usamos el último Opamp disponible, el Opamp 5.

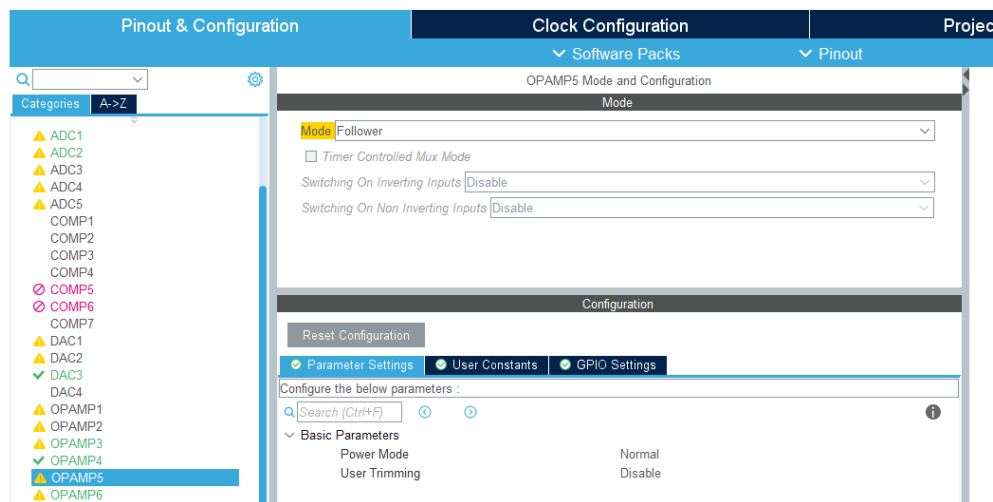


Figura 37. Configuración Parameter Settings OPAMP 5

Con el código de inicialización:

```
if(HAL_OK != HAL_OPAMP_Start(&hopamp5)) { Error_Handler(); }
```

Con todos los componentes del microcontrolador ya configurados el siguiente paso era implementar un prototipo del sistema en una placa perforada. Para la implementación física de este sistema en concreto las resistencias señaladas del mismo color tienen que tener el mismo valor o el más cercano posible con una tolerancia máxima de 3 Ohmios:

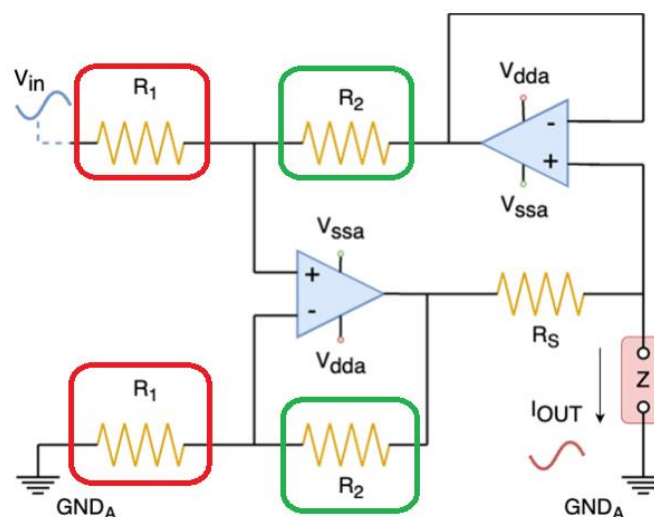


Figura 38. Resistencias emparejadas en el Sistema Generador de Señal

Para ello tuvimos que medir una a una el valor de varias resistencias SMD para encontrar 2 pares entre las resistencias R1 y R2 de 22KΩ.

Para la zona de la muestra, añadimos un conector con la intención de probar distintas resistencias en la fase de testeo para comprobar el funcionamiento del sistema.

Una vez teníamos los componentes listos, usamos una lista con los pines del microcontrolador que usa para las entradas y salidas de los componentes internos que habíamos obtenido de consultar el .ioc y comparar con el datasheet disponible en la web de STMicroelectronics [11] como referencia:

- PA7 CN5 4 INP OPAM1 STANDALONE BLANCO
- PA2 CN9 2 OUT OPAM1 STANDALONE NEGRO
- PA3 CN9 1 INM OPAM1 STANDALONE MARRON
- PB0 CN8 4 IN OPAM3 FOLLOWER BLANCO
- PB1 CN10 24 OUT OPAM3 FOLLOWER NEGRO
- PB11 CN10 18 OUT OPAM6 FUENTE VERDE
- PB13 CN10 30 INP OPAM4 STANDALONE BLANCO
- PB12 CN10 16 OUT OPAM4 STANDALONE NEGRO
- PB10 CN9 7 INM OPAM4 STANDALONE MARRON
- PC3 CN7 37 IN OPAM5 FOLLOWER TIERRA VIRTUAL
- PA8 CN10 23 OUT OPAM5 FOLLOWER TIERRA VIRTUAL

Esta lista nos sirvió de guía para soldar el siguiente prototipo en el laboratorio:

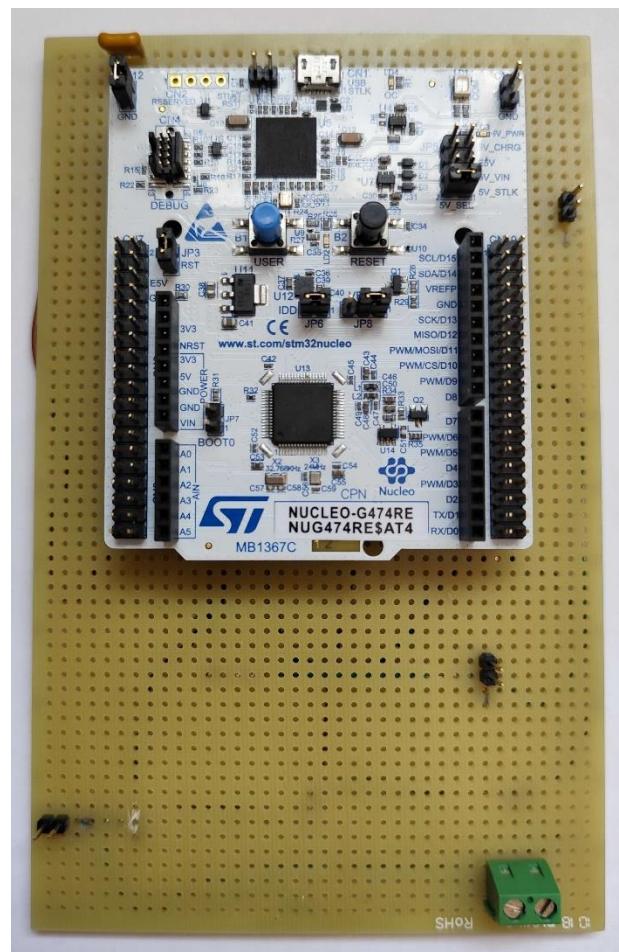


Figura 39. Parte delantera del prototipo del sistema

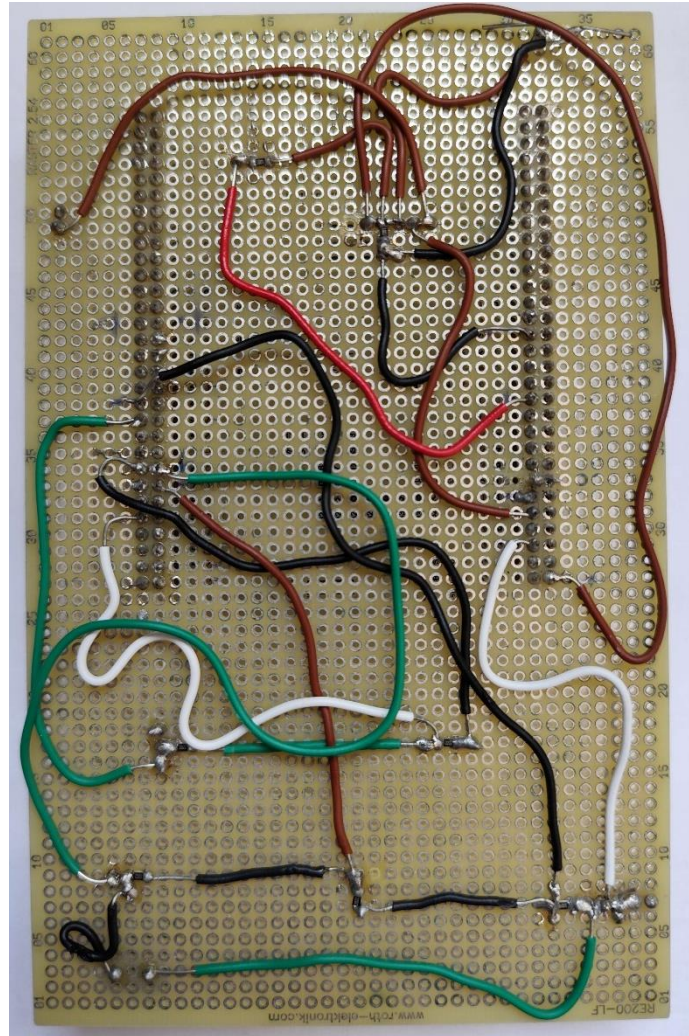


Figura 40. Parte trasera del prototipo del sistema

T4.5- Puntos de medición

Una vez ya teníamos el prototipo soldado con la tierra virtual, necesitamos soldar una serie de puntos de medición para realizar el testeo y las comprobaciones de que el sistema funcionaba correctamente.

Los puntos son:

- Salida del Opamp6 para comprobar la señal que viene del DAC.
- Salida del sistema, que se encuentra en la salida del Opamp3 y la conexión con la resistencia R2 superior.
- A la salida del divisor de tensión de la tierra virtual.

En cada uno de estos puntos se soldaron una tira de dos pines para mayor estabilidad, aunque solo uno de ellos aporta la medición.

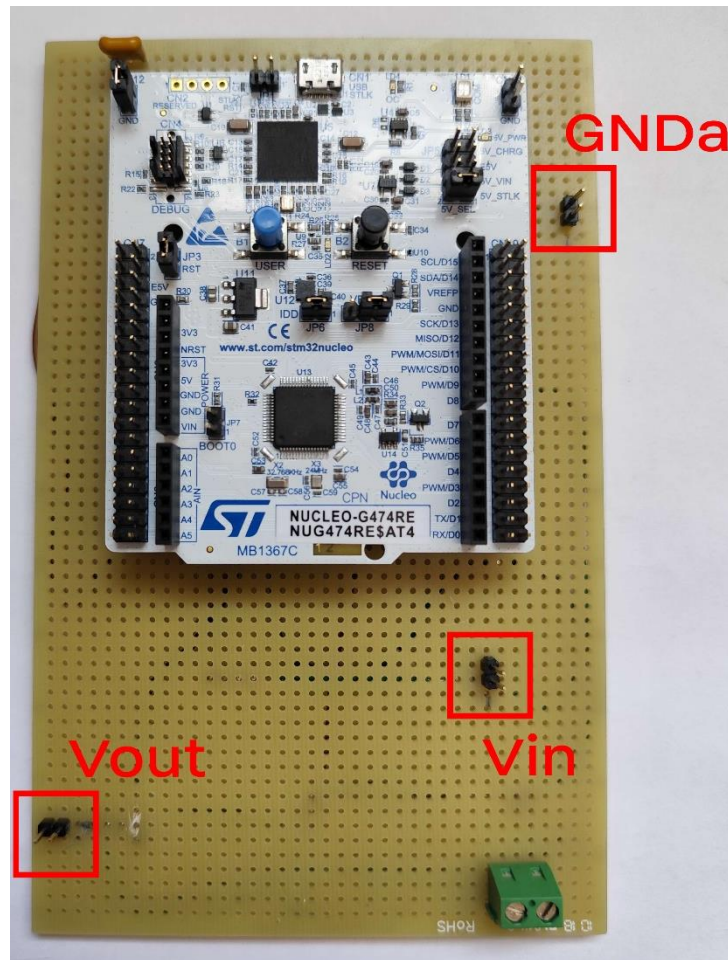


Figura 41. Puntos de medición en el prototipo

T4.6- Lógica generadora

En esta tarea el objetivo es implementar dos funciones capaces de controlar la amplitud y la frecuencia de la señal generada. Para ello hemos creado dos funciones, una llamada `CalculaLut()` y otra llamada `AjustaTimer()`.

La función CalculaLut() es una adaptación del código generador de la LUT del script en python, se usa para modificar la amplitud y la precisión de la generación de señal:

```
void calculaLut(){
    float step = 2*M_PI/TAM_LUT;
    uint32_t i = 0;
    while(i < TAM_LUT){
        //sin(TAM_LUT/4*step) = 0xFFFF
        uint32_t v = (uint32_t)((sin(i * step)*amp/sin(2*M_PI/4))+amp);
        lut[i] = v;
        i++;
    }
}
```

Para su uso se definen varias variables globales:

```
#define TAM_LUT 32
uint32_t amp = 2048;
uint32_t lut[TAM_LUT] = {0};
uint32_t lut_size = sizeof(lut)/sizeof(lut[0]);
```

- TAM_LUT: Indica el tamaño de la lista, a mayor tamaño mayor precisión.
- amp: es el valor de la amplitud, cuyo valor máximo es 2048 debido a que el registro del DAC es de 12bits.
- lut: es la LUT como tal, al definirla se le asignan 0s en todas las posiciones, que luego se modifican en calculaLut()
- lut_size: es equivalente a TAM_LUT, se calcula teniendo en cuenta el tamaño en bytes de la lut al completo dividiendo el tamaño en bytes del primer registro, en este caso al ser un número sin signo de 32 bits son 4 bytes.

La otra función que hemos implementado es `AjustaTimer()`, esta función se encarga de ajustar el TIM3 encargado de disparar las interrupciones encargadas de modificar el valor del DAC para poder controlar la frecuencia de la señal:

```
void ajustaTimers(){
    //TIM3 prescaler y periodo
    //170Mhz clock
    //100mhz-500khz
    //100mhz, 1hz, 10hz, 50hz, 100hz, 200hz, 500hz, 700hz, 1khz, 10khz, 50khz, 100khz, 200khz, 500khz
    //f_deseada = f_timer/((ARR+1)x(PSC+1))
    //tim3 ARR 16 bits - PSC 16 bits
    switch(frequency){
        case 11://0.1 100mHz
            htim3.Init.Prescaler = 2999;

            htim3.Init.Period = 17740;
            break;
        case 1://1
            htim3.Init.Prescaler = 299;

            htim3.Init.Period = 17740;
            break;
        case 10://10
            htim3.Init.Prescaler = 29;

            htim3.Init.Period = 17740;
            break;
        case 50://50
            htim3.Init.Prescaler = 5;

            htim3.Init.Period = 17740;
            break;
        case 100://100
            htim3.Init.Prescaler = 2;

            htim3.Init.Period = 17740;
            break;
    }
```

```

    case 200://200
        htim3.Init.Prescaler = 2;

        htim3.Init.Period = 8870;
        break;
    case 500: //500
        htim3.Init.Prescaler = 4;

        htim3.Init.Period = 2129;
        break;
    case 700://700
        htim3.Init.Prescaler = 3;

        htim3.Init.Period = 60819;
        break;
    case 1000://1000
        htim3.Init.Prescaler = 2;

        htim3.Init.Period = 1774;
        break;
    case 10000://10000
        htim3.Init.Prescaler = 2;

        htim3.Init.Period = 177;
        break;
    case 50000://50000
        htim3.Init.Prescaler = 0;

        htim3.Init.Period = 105;
        break;
    case 100000://100000
        htim3.Init.Prescaler = 0;

        htim3.Init.Period = 52;
        break;
    case 200000://200000
        htim3.Init.Prescaler = 0;

        htim3.Init.Period = 25;
        break;
    case 500000://500000
        htim3.Init.Prescaler = 0;

        htim3.Init.Period = 4;
        break;
    default:
        htim3.Init.Prescaler = 0;

        htim3.Init.Period = 4;
        break;
}
}

```

Para poder cambiar la configuración de un timer es necesario desinicializarlo primero, cambiar la configuración y luego volver a inicializarlo, este proceso se detallará al final de la tarea.

Para poder comprender que valores de la configuración de un timer afectan a la frecuencia de disparo de la interrupción es necesario conocer el funcionamiento de dos atributos del timer, en este caso del TIM3, esto son el valor del

Preescalador (PSC) y el valor del periodo de Auto Recarga (ARR). El valor del preescalador es el valor que divide la señal de reloj del sistema usado por el timer y el valor del periodo de AutoRecarga es el valor máximo que alcanza el contador del timer antes de disparar la interrupción. Ambos actúan a la hora de aumentar o reducir la frecuencia de la interrupción, mientras más bajo sean sus valores, mayor será la frecuencia.

Para configurar correctamente la frecuencia, se atiende a la siguiente fórmula teniendo en cuenta que el reloj que hemos configurado para el sistema es de 170MHz:

$$TIM(Hz) = \frac{Clock}{(PSC + 1) \cdot (ARR + 1)}$$

Podríamos modificar ambos valores a la vez, pero por norma general se intenta minimizar el valor de uno de ellos, en nuestro caso el preescalador (PSC).

Esto funcionaría en la teoría, pero debido a nuestra experimentación la frecuencia del sistema no se ajusta con exactitud. Tras consultarlo con el tutor llegamos a la conclusión que podría deberse a que el sistema realiza algún tipo de operación sobre la señal y que el reloj del sistema no alcanza realmente los 170MHz. Por lo tanto, decidimos hacer este ajuste de manera experimental tomando como referencia el valor de la formula, por ejemplo, en el caso de 100Hz que fue el primero que decidimos ajustar el valor de la fórmula sería:

$$100 = \frac{\text{Original } 170000000}{(0 + 1) \cdot (1699999 + 1)}$$

Aquí encontramos el primer problema y es que el tanto registro del periodo de AutoRecarga (ARR) como el del preescalador (PSC) solo tienen 16 bits por lo que su valor máximo sería 65535, por lo que para ajustarlo debemos aumentar el valor PSC hasta que el valor de ARR sea menor que 65535.

$$100 = \frac{\text{Ajustado} \quad 170000000}{(99+1) \cdot (16999+1)}$$

Pero al probar la configuración con estos valores, los resultados no eran los esperados, por lo tanto, como hemos mencionado decidimos ajustarlos directamente en base al resultado obtenido aplicando multiplicaciones y divisiones, teniendo el siguiente resultado:

$$100 = \frac{\text{Real} \quad 170000000}{(2 + 1) \cdot (17740 + 1)}$$

Como podemos apreciar hay una diferencia de un factor de multiplicación de alrededor de 33 entre el valor real y el ajustado, esto implicaría tener que cambiar los ajustes del reloj para poder hacer coincidir en la medida de lo posible ambos valores, esto sería proceso demasiado tedioso que puede ser sustituido por ajustar los valores de forma experimental y de esta forma confirmar su valor real. En el caso de querer cambiar los ajustes tendríamos que cambiar los multiplicadores y divisores de la siguiente configuración, proceso que hemos descartado.

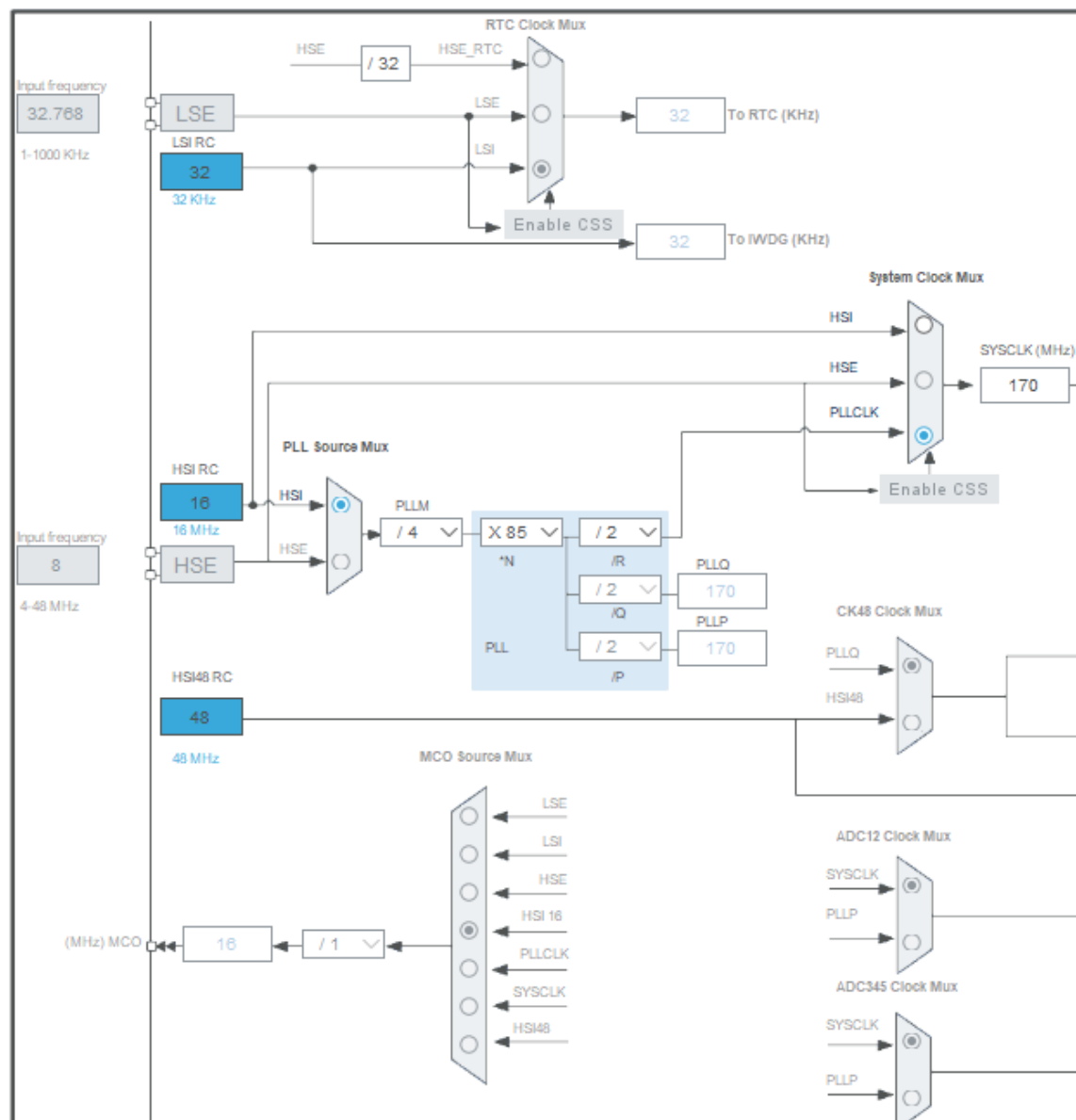


Figura 42. Configuración del reloj

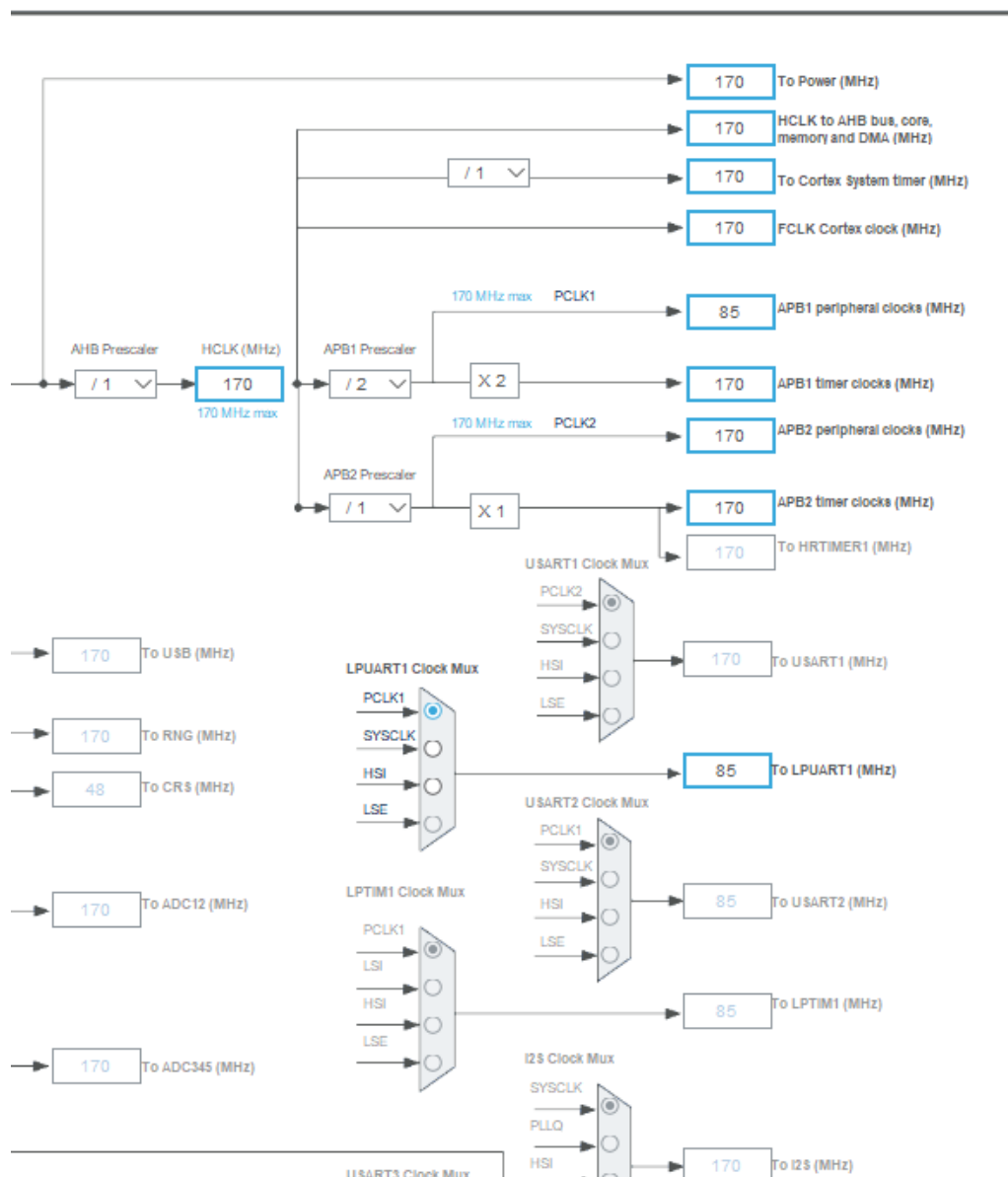


Figura 43. Configuración del reloj

A partir de aquí ya podríamos conseguir todas las configuraciones posibles, por ejemplo, si queremos 200Hz hay que duplicar el valor de la frecuencia por lo que el valor obtenido al multiplicar $(PSC+1) * (ARR+1)$ debe ser la mitad, en nuestro caso reducimos el ARR a la mitad, pasando de 17740+1 a 8870+1.

De esta forma obtuvimos todas las frecuencias en la función y usamos una variable global para decidir con una estructura switch-case que frecuencia vamos a usar.

Como comentamos anteriormente, para usar esta función es necesario “desinicializar” el TIM3 para poder modificar su configuración, lo implementamos con el siguiente código en el fichero main.c antes del bucle while(1):

```
HAL_TIM_Base_DeInit(&htim3);

ajustaTimers();
calculaLut();

//Timers
if (HAL_TIM_Base_Init(&htim3) != HAL_OK){Error_Handler();}

HAL_TIM_Base_Start(&htim3);
```

Primero realizamos la “desinicialización”, luego ajustamos el TIM3 y terminamos volviendo a inicializarlo y lanzando su ejecución.

T4.7- Testeo

Para comprobar el funcionamiento del generador de señal cambiamos el valor de la variable “frequency” para comprobar todos los casos de la estructura switch-case y medimos sus valores usando el dispositivo DiscoveryV2 y el software WaveForms en el punto de medición colocado a la salida del Opamp6 en el punto de Vin del sistema.

Los resultados fueron los siguientes:

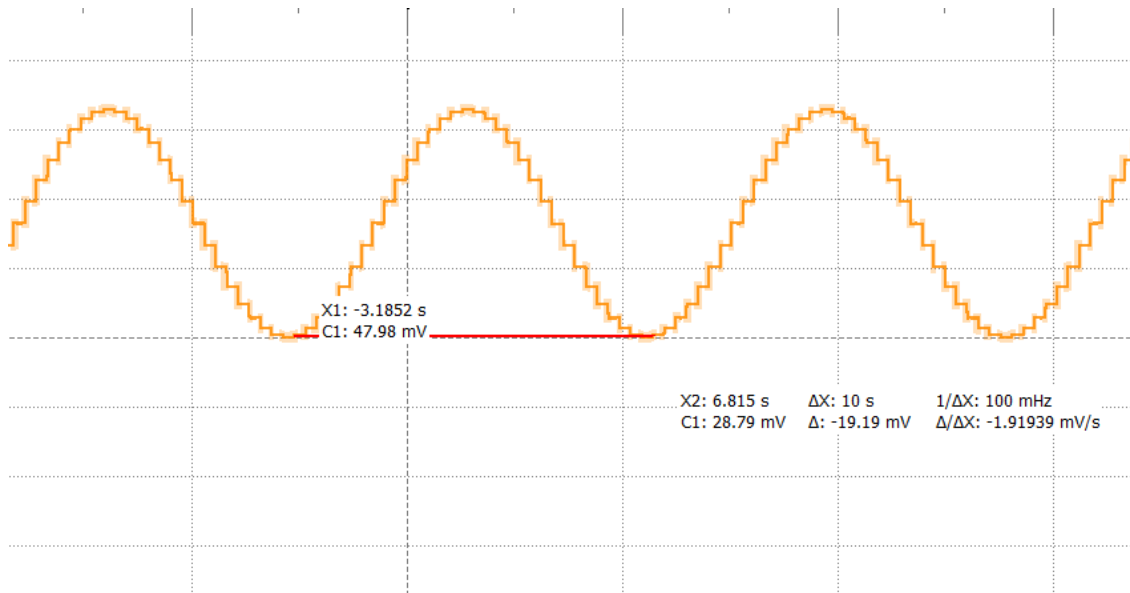


Figura 44. Resultado a una frecuencia de 100 mHz

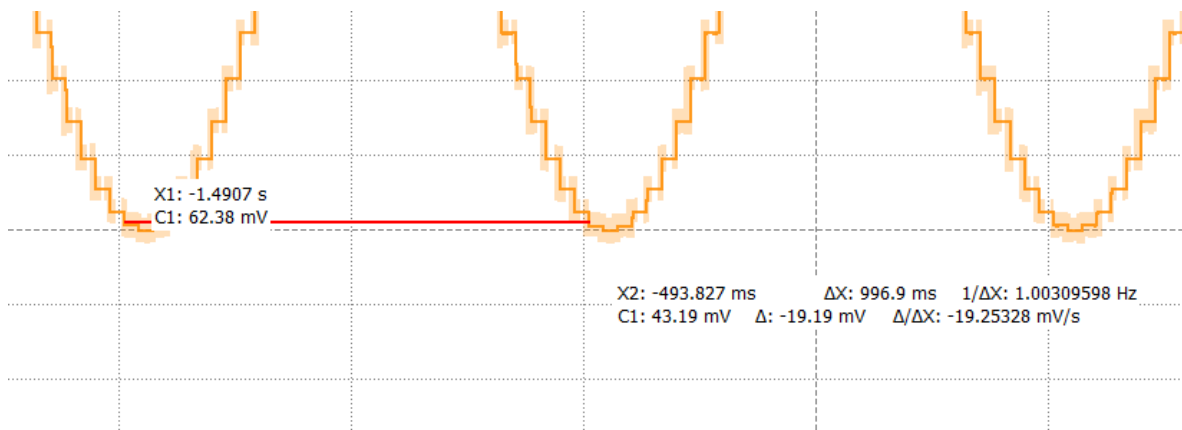


Figura 45. Resultado a una frecuencia de 1 Hz

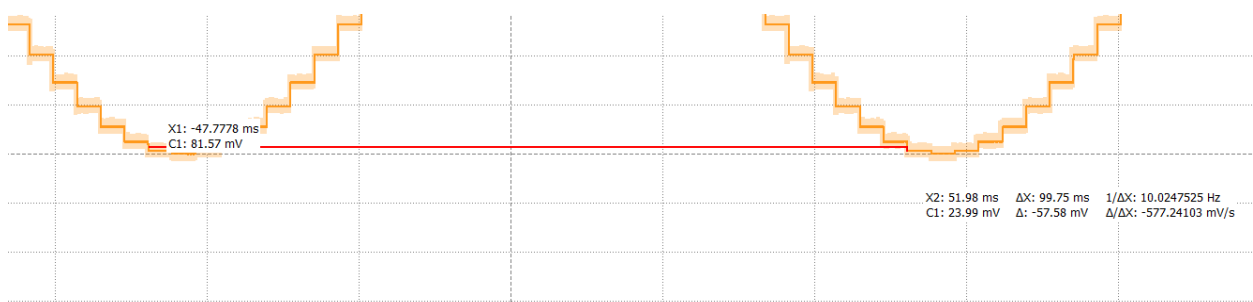


Figura 46. Resultado a una frecuencia de 10 Hz

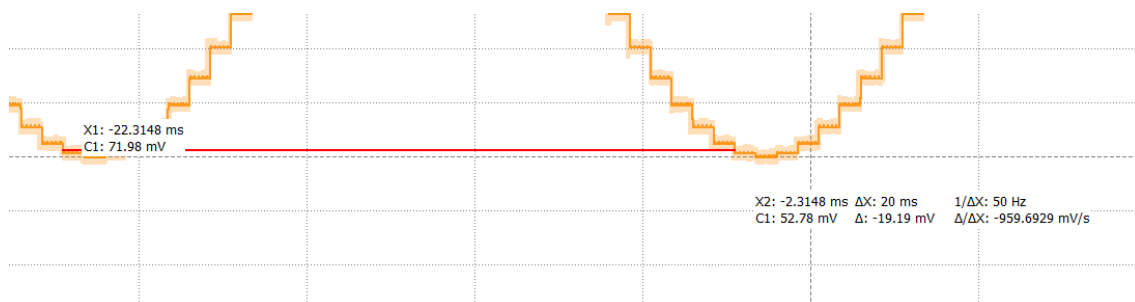


Figura 47. Resultado a una frecuencia de 50 Hz

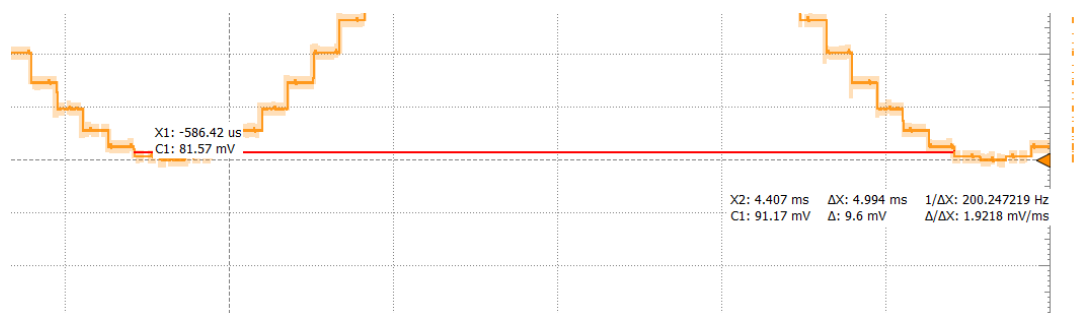


Figura 48. Resultado a una frecuencia de 200 Hz

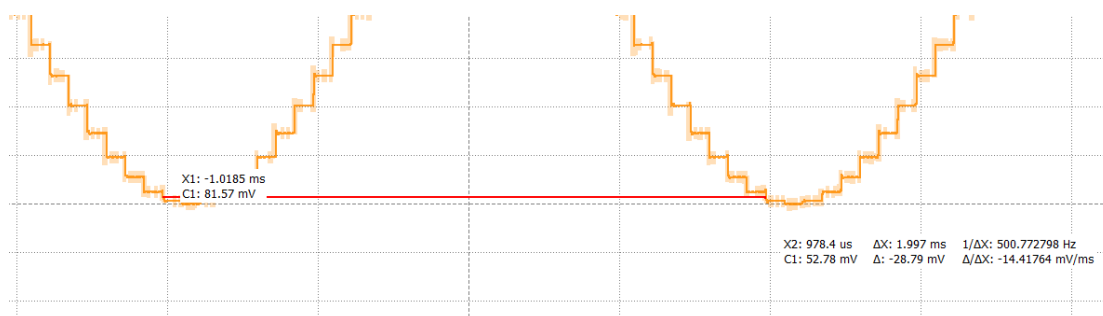


Figura 49. Resultado a una frecuencia de 500 Hz

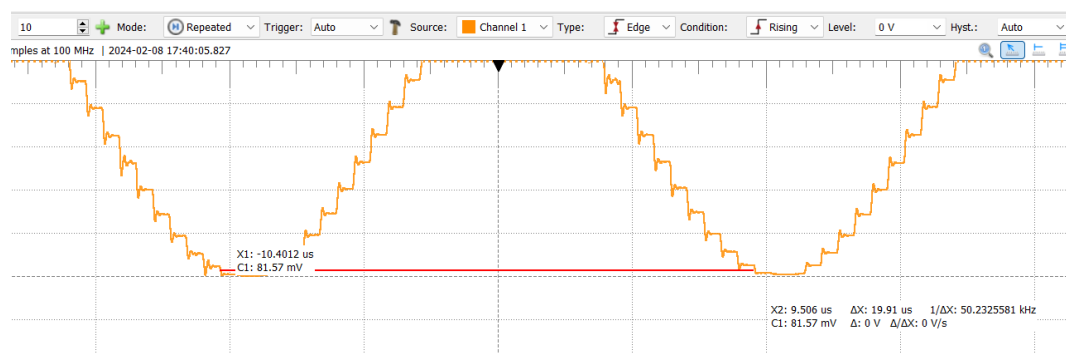


Figura 50. Resultado a una frecuencia de 50 kHz

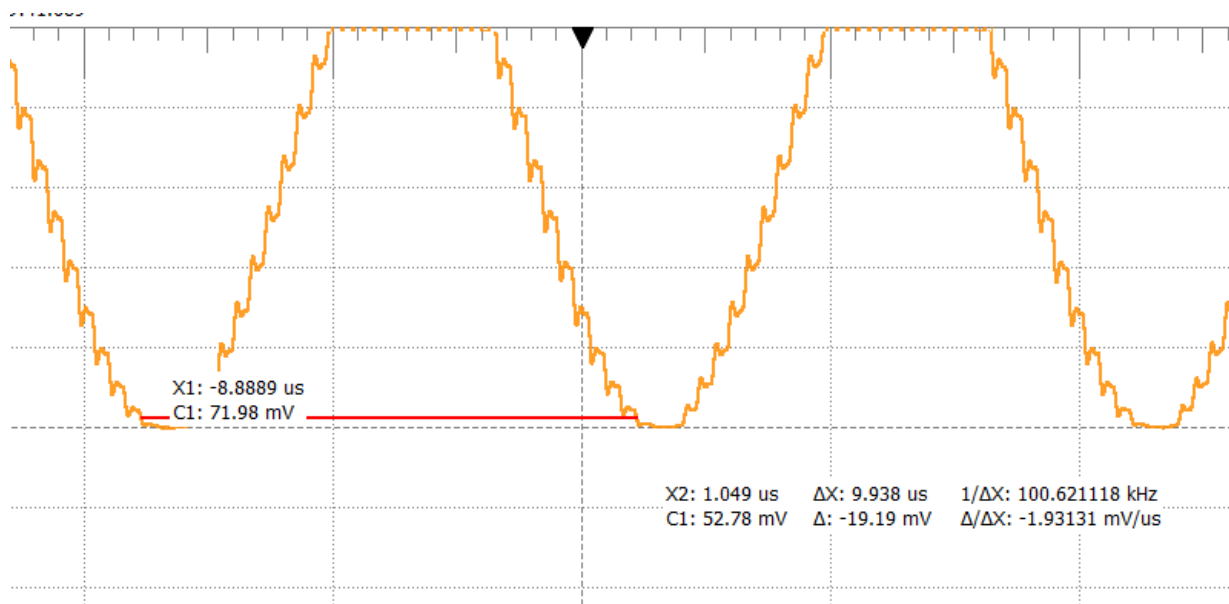


Figura 51. Resultado a una frecuencia de 100 kHz

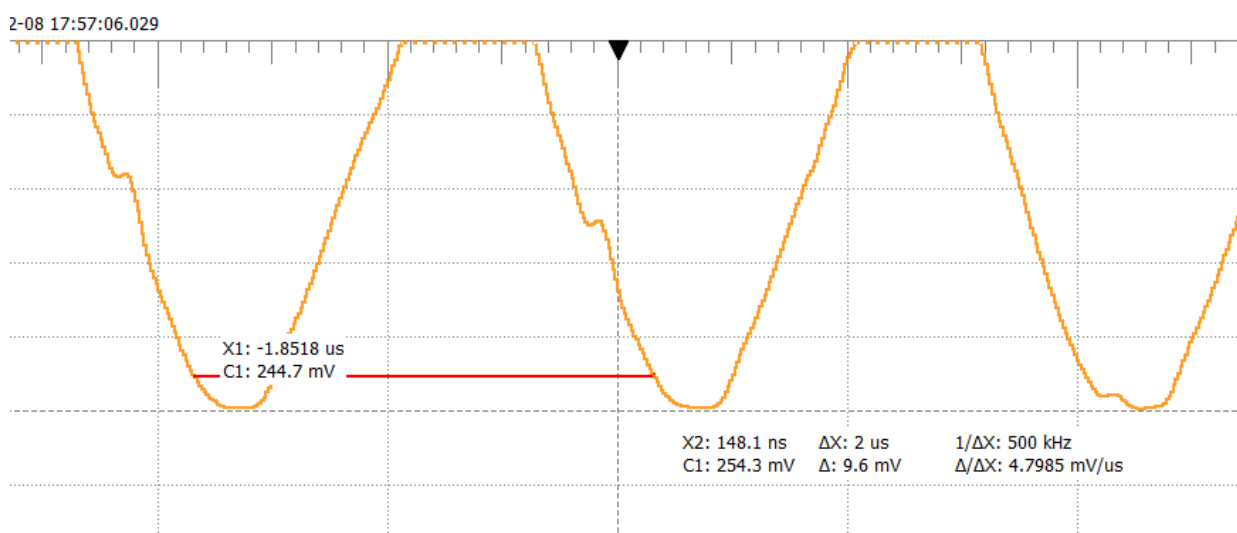


Figura 52. Resultado a una frecuencia de 500 kHz

3.3.5. F5: Lectura de la señal.

El hito correspondiente a esta fase es **H5-Toma de lecturas**.

T5.1- Configurar ADCs

Para esta fase lo primero es configurar en el fichero .ioc los dos ADCs encargados de leer la señal, uno a la salida del Opamp6 (V_{in}) y otro entre el R2 superior y Opamp3 (V_{out}), esto es necesario para aplicar la fórmula del sistema y conocer la impedancia.

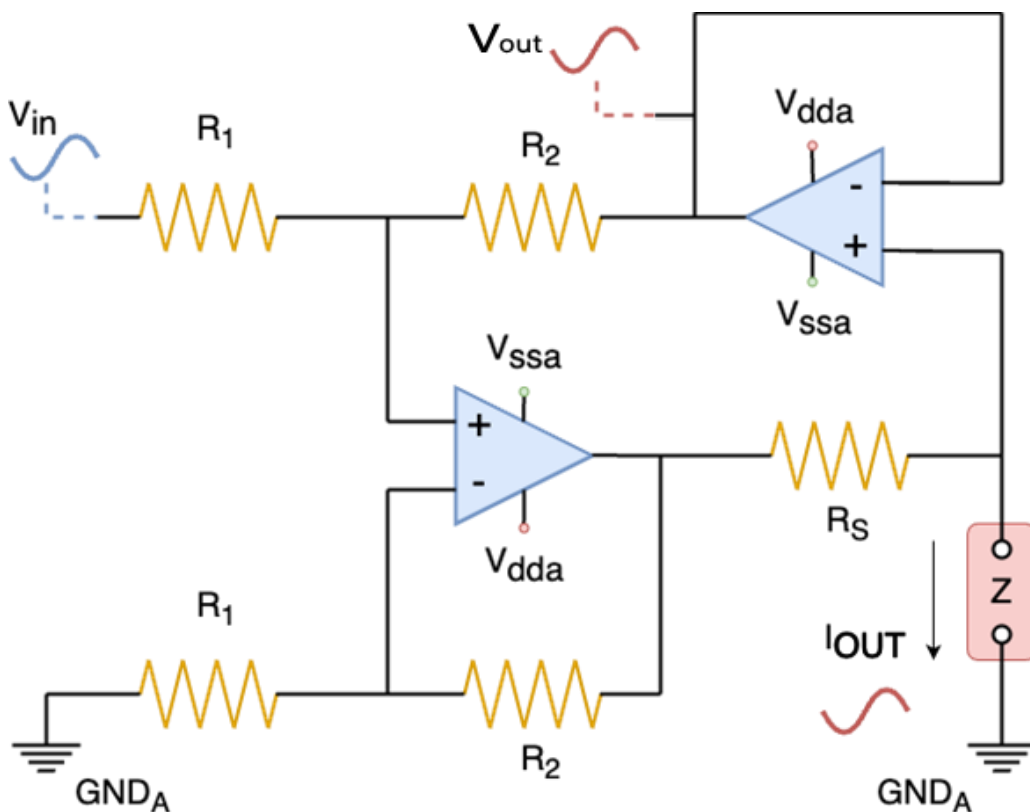


Figura 53. Estructura del Sistema Generador de Señal

Ambos ADCs estarán configurados en single-ended y la configuración es la siguiente en ambos:

- ADC1(Vin)

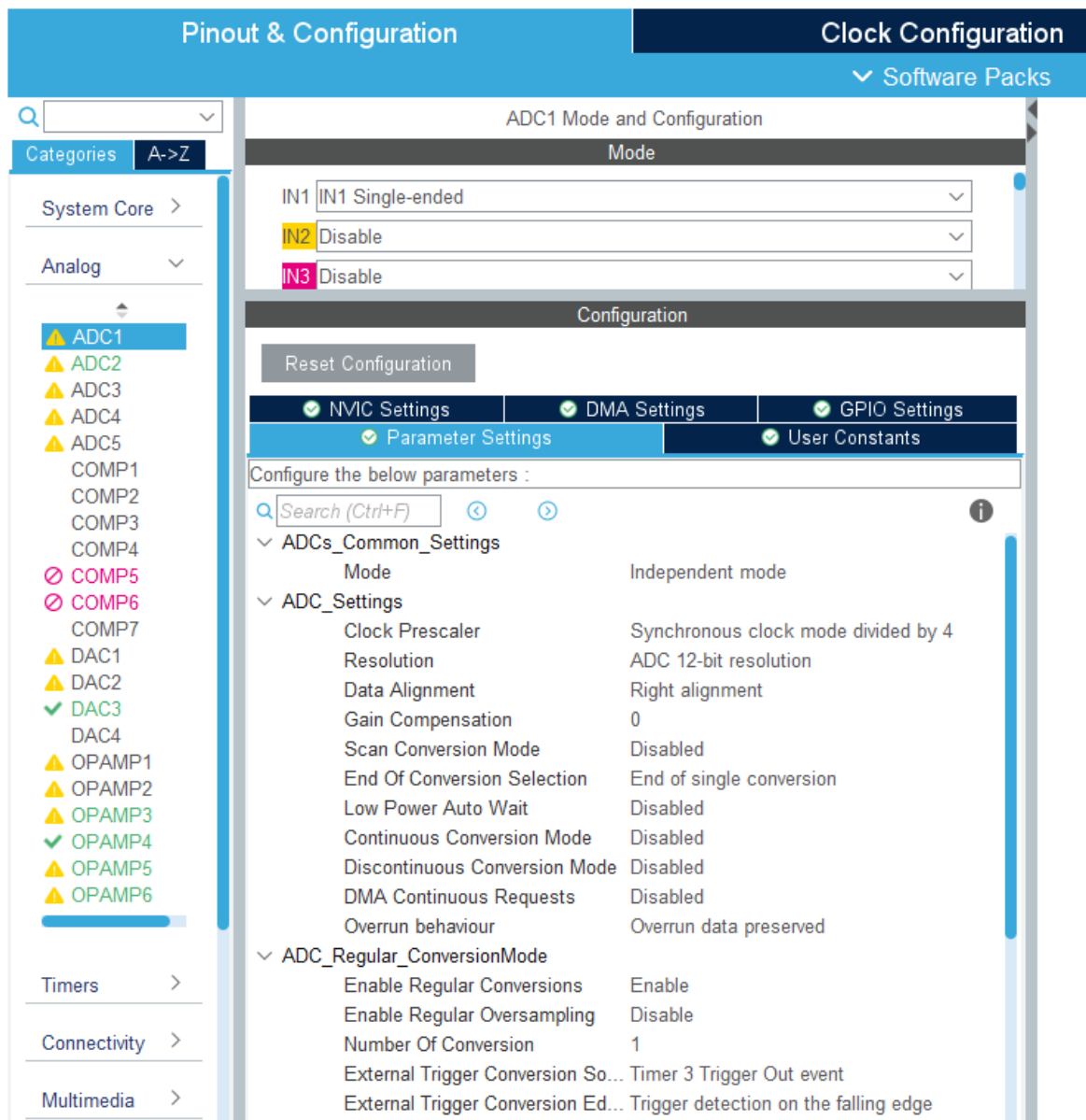


Figura 54. Configuración Parameter Settings ADC 1

- ADC2 (Vout)

Pinout & Configuration

Clock Configuration

Software Packs

Categories

A-Z

System Core >

Analog >

ADC1

ADC2

ADC3

ADC4

ADC5

COMP1

COMP2

COMP3

COMP4

COMP5

COMP6

COMP7

DAC1

DAC2

DAC3

DAC4

OPAMP1

OPAMP2

OPAMP3

OPAMP4

OPAMP5

OPAMP6

Timers >

Connectivity >

Multimedia >

ADC2 Mode and Configuration

Mode

IN2 Disable

IN3 IN3 Single-ended

IN4 Disable

Configuration

Reset Configuration

Parameter Settings

User Constants

NVIC Settings

DMA Settings

GPIO Settings

Configure the below parameters :

Search (Ctrl+F)

ADCs_Common_Settings

Mode

Independent mode

ADC_Settings

Clock Prescaler

Synchronous clock mode divided by 4

Resolution

ADC 12-bit resolution

Data Alignment

Right alignment

Gain Compensation

0

Scan Conversion Mode

Disabled

End Of Conversion Selection

End of single conversion

Low Power Auto Wait

Disabled

Continuous Conversion Mode

Disabled

Discontinuous Conversion Mode

Disabled

DMA Continuous Requests

Disabled

Overrun behaviour

Overrun data preserved

ADC_Regular_ConversionMode

Enable Regular Conversions

Enable

Enable Regular Oversampling

Disable

Number Of Conversion

1

External Trigger Conversion Source

Timer 3 Trigger Out event

External Trigger Conversion Edge

Trigger detection on the falling edge

Rank

1

Figura 56. Configuración Parameter Settings ADC 2

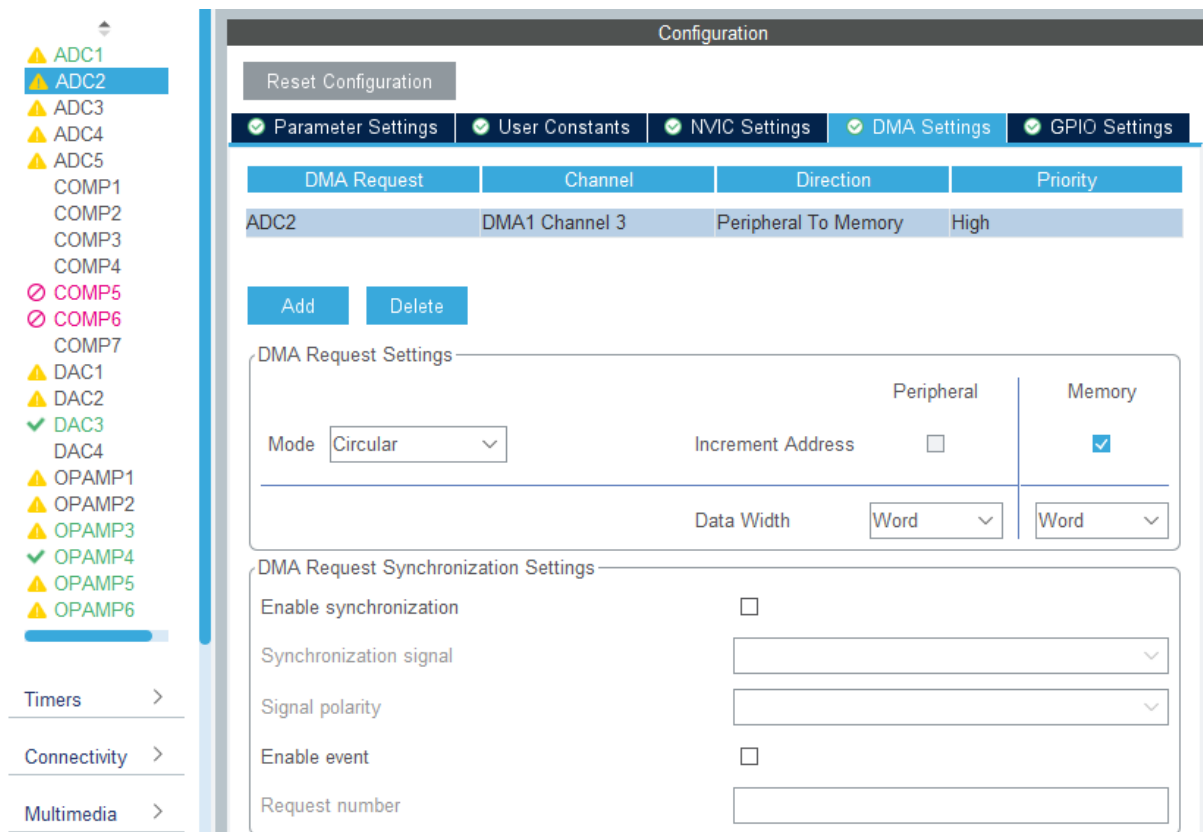


Figura 57. Configuración DMA Settings ADC 2

Ambos están controlados por el TIM3 y toman la lectura en flanco de bajada usando DMA para guardar los datos. Según habíamos configurado el DAC este era controlado también por el TIM3 pero en el flanco de subida, aunque es una característica que no se puede configurar desde el fichero .ioc. El tiempo transcurrido desde el flanco de subida y el de bajada es suficiente para actualizar el nuevo valor y que pueda ser leído por los ADCs.

Ahora pasamos a la implementación en código, para implementar un ADC es necesario definir un buffer donde guardar los datos leídos, en nuestro caso será de 10 veces el tamaño de la LUT. La inicialización sería la siguiente:

```
#define TAM_BUFFER TAM_LUT*10
uint32_t adc1_buffer[TAM_BUFFER] = {0};
uint32_t adc2_buffer[TAM_BUFFER] = {0};
```

```
//ADC
if (HAL_ADC_Start_DMA(&hadc1, (uint32_t *)adc1_buffer,TAM_BUFFER) != HAL_OK) {
    Error_Handler();
}
if (HAL_ADC_Start_DMA(&hadc2, (uint32_t *)adc2_buffer,TAM_BUFFER) != HAL_OK) {
    Error_Handler();
}
```

Esto irá guardando en el buffer los valores leídos a través del pin PA0 para el ADC1 (Vin) y PA6 para el ADC2 (VOut).

T5.2- Conexiones y testeo

Para las conexiones en este caso decidimos no soldarlas al prototipo ya que en la siguiente fase del proyecto se realizará una PCB con las conexiones implementadas. Optamos por unir directamente los pines del microcontrolador con cables temporales:

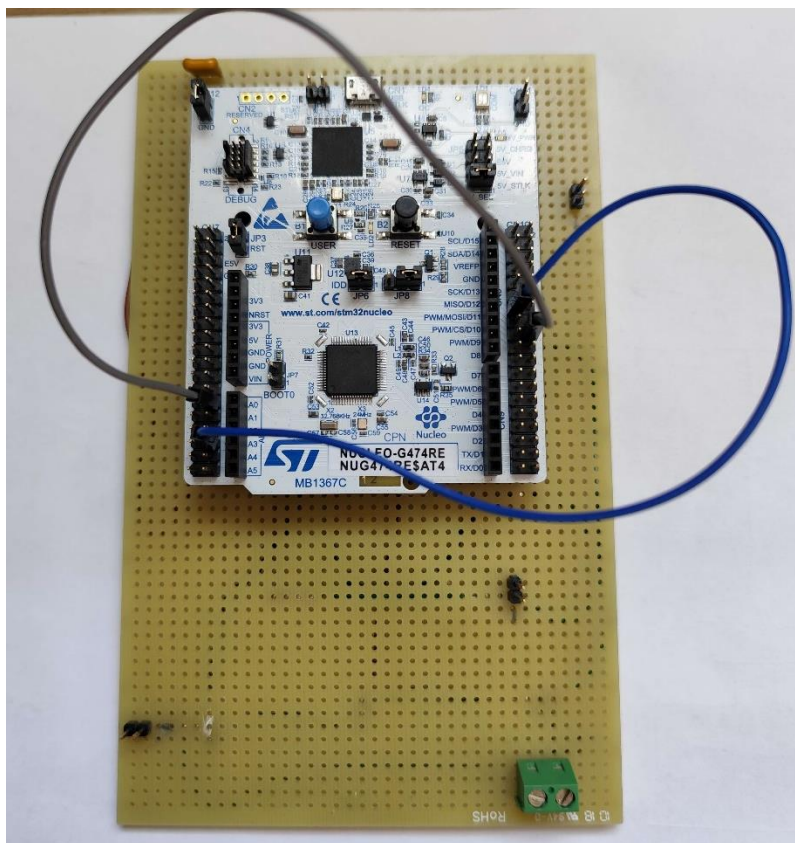


Figura 58. Conexiones de los ADCs en el prototipo

Una vez realizado en cableado, para probar que el sistema funciona correctamente decidimos depurar el código en el microcontrolador y observar el contenido del buffer del ADC1 ya que es el que podemos conocer directamente su valor que debe ser similar a los valores de la LUT del DAC repetidos 10 veces.

Expression	Type	Value
(x)= adc1_buffer[24]	uint32_t	531
(x)= adc1_buffer[25]	uint32_t	86
(x)= adc1_buffer[26]	uint32_t	0
(x)= adc1_buffer[27]	uint32_t	67
(x)= adc1_buffer[28]	uint32_t	529
(x)= adc1_buffer[29]	uint32_t	1188
(x)= adc1_buffer[30]	uint32_t	1971
(x)= adc1_buffer[31]	uint32_t	2746
(x)= adc1_buffer[32]	uint32_t	3419
(x)= adc1_buffer[33]	uint32_t	3863
(x)= adc1_buffer[34]	uint32_t	4015
(x)= adc1_buffer[35]	uint32_t	3867
(x)= adc1_buffer[36]	uint32_t	3425
(x)= adc1_buffer[37]	uint32_t	2761
(x)= adc1_buffer[38]	uint32_t	1976
(x)= adc1_buffer[39]	uint32_t	1197
(x)= adc1_buffer[40]	uint32_t	531
(x)= adc1_buffer[41]	uint32_t	85
(x)= adc1_buffer[42]	uint32_t	0
(x)= adc1_buffer[43]	uint32_t	69
(x)= adc1_buffer[44]	uint32_t	529
(x)= adc1_buffer[45]	uint32_t	1187
(x)= adc1_buffer[46]	uint32_t	1970
(x)= adc1_buffer[47]	uint32_t	2743

Figura 59. Contenido del buffer del ADC 1

T5.3- UART

Para la implementación de la conexión UART lo primero es configurarla en el fichero .ioc, definimos el modo, la velocidad y el tamaño de los datos, en nuestro caso modo Asíncrono, un “baud rate” de 115200bits/s y un tamaño de 8 bits por dato transmitido.

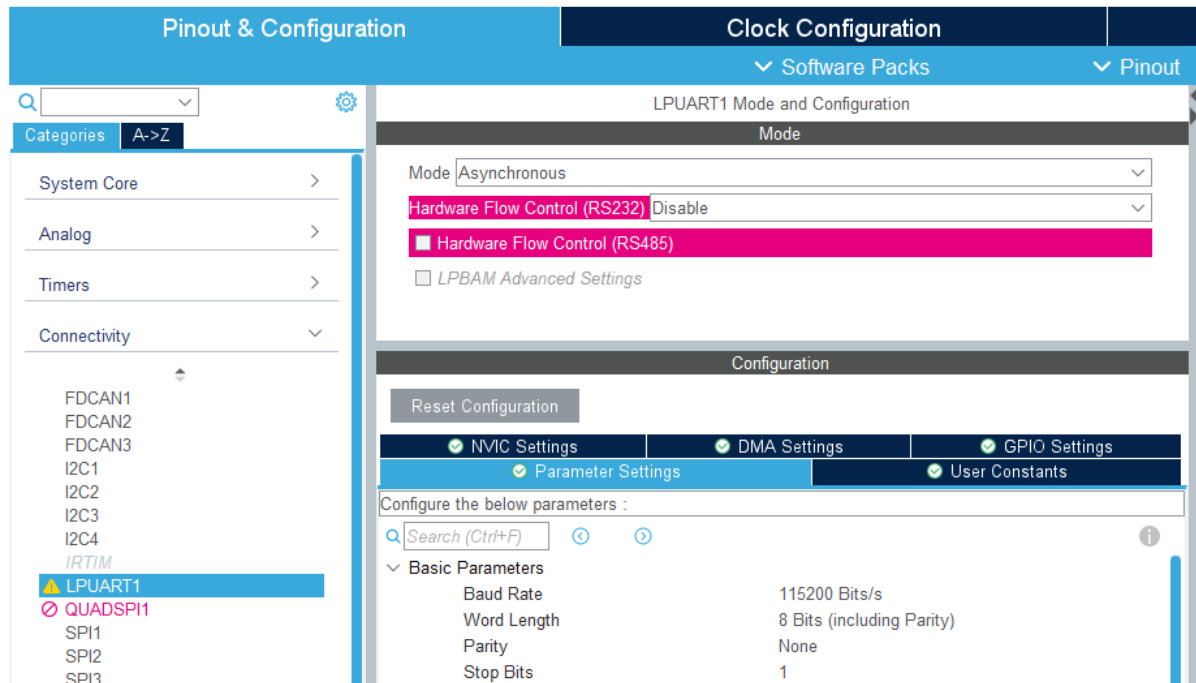


Figura 60. Configuración Parameter Settings UART

Lo siguiente es la implementación en código para el cual el procedimiento general es implementar la función printf(), ya que el microcontrolador trabaja en C y las cadenas de texto se transmiten byte a byte (en ascii) y se interpretan en el host como corresponda. La función printf utiliza PUTCHAR como base para imprimir caracteres. Al escribir la siguiente función, indicamos que los caracteres deben enviarse por el puerto UART:

```
PUTCHAR_PROTOTYPE{
    HAL_UART_Transmit(&hlpuart1, (uint8_t *) &ch, 1, 0xFFFF);
    return ch;
}
```


Y la llamada a la función, dependiendo del microcontrolador:

```
#ifndef __GNUC__
#define PUTCHAR_PROTOTYPE int __io_putchar(int ch)
#else
#define PUTCHAR_PROTOTYPE int fputc(int ch, FILE *f)
#endif
```

Ahora ya podemos usar el método printf() para enviar datos por la UART. En nuestro caso queremos transmitir toda la información de los buffers en cada iteración del bucle while(1) y añadir un parpadeo al led de la placa para asegurarnos de que se encuentra en funcionamiento:

```
while (1)
{
    HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
    HAL_Delay(1000);

    printf("<M1>");
    for(int j=0; j<TAM_BUFFER; j++){
        printf("%d,",adc1_buffer[j]);
    }
    printf("</M1>\n\r");

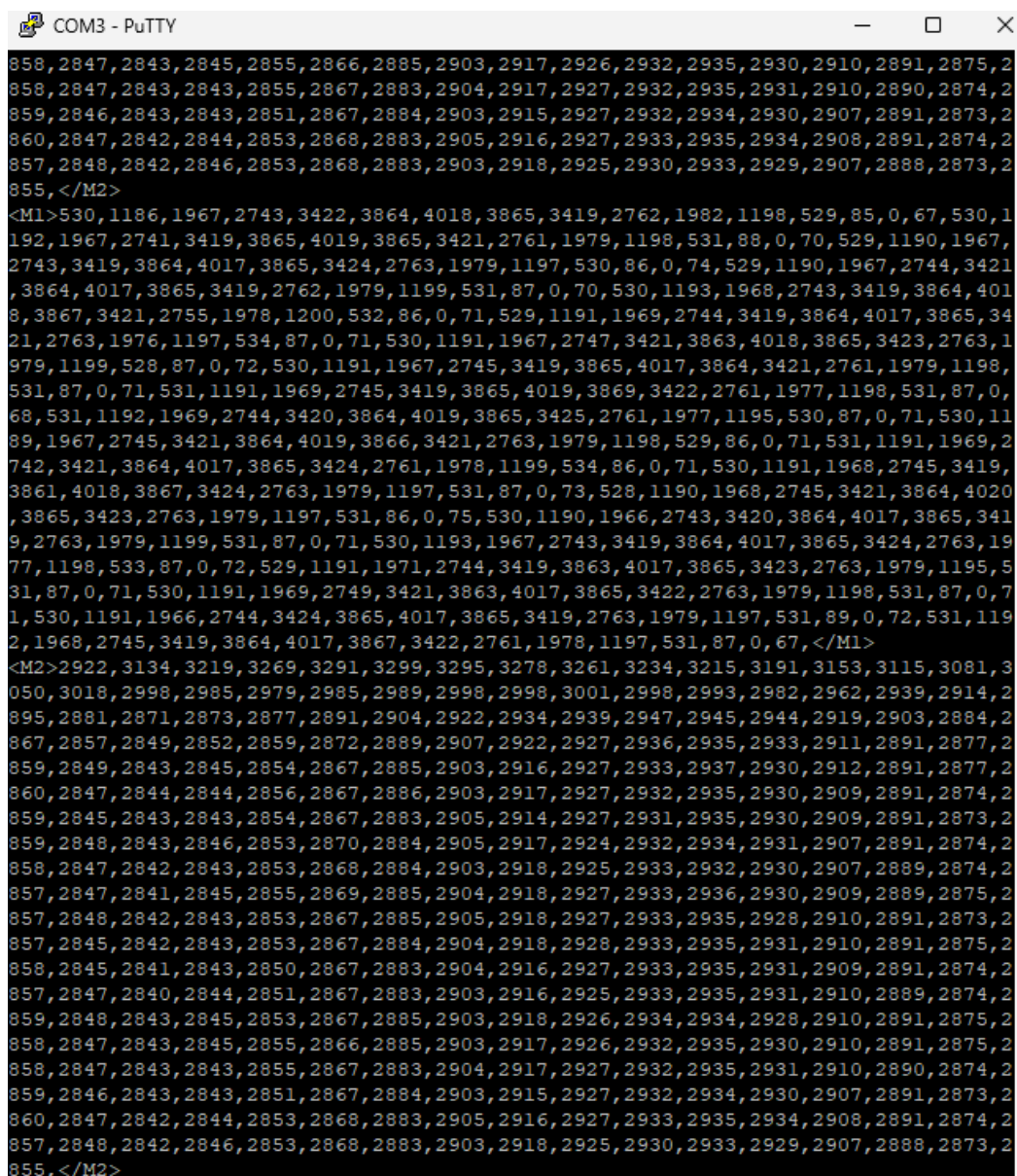
    printf("<M2>");
    for(int j=0; j<TAM_BUFFER; j++){
        printf("%d,",adc2_buffer[j]);
    }
    printf("</M2>\n\r");

    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
}
```

Para poder saber de qué ADC se trata añadimos un par de etiquetas M1 para ADC1 y M2 para ADC2.

Y el resultado visualizado en el software PuTTY es el siguiente:



```
COM3 - PuTTY
858,2847,2843,2845,2855,2866,2885,2903,2917,2926,2932,2935,2930,2910,2891,2875,2
858,2847,2843,2843,2855,2867,2883,2904,2917,2927,2932,2935,2931,2910,2890,2874,2
859,2846,2843,2843,2851,2867,2884,2903,2915,2927,2932,2934,2930,2907,2891,2873,2
860,2847,2842,2844,2853,2868,2883,2905,2916,2927,2933,2935,2934,2908,2891,2874,2
857,2848,2842,2846,2853,2868,2883,2903,2918,2925,2930,2933,2929,2907,2888,2873,2
855,</M2>
<M1>530,1186,1967,2743,3422,3864,4018,3865,3419,2762,1982,1198,529,85,0,67,530,1
192,1967,2741,3419,3865,4019,3865,3421,2761,1979,1198,531,88,0,70,529,1190,1967,
2743,3419,3864,4017,3865,3424,2763,1979,1197,530,86,0,74,529,1190,1967,2744,3421
,3864,4017,3865,3419,2762,1979,1199,531,87,0,70,530,1193,1968,2743,3419,3864,401
8,3867,3421,2755,1978,1200,532,86,0,71,529,1191,1969,2744,3419,3864,4017,3865,34
21,2763,1976,1197,534,87,0,71,530,1191,1967,2747,3421,3863,4018,3865,3423,2763,1
979,1199,528,87,0,72,530,1191,1967,2745,3419,3865,4017,3864,3421,2761,1979,1198,
531,87,0,71,531,1191,1969,2745,3419,3865,4019,3869,3422,2761,1977,1198,531,87,0,
68,531,1192,1969,2744,3420,3864,4019,3865,3425,2761,1977,1195,530,87,0,71,530,11
89,1967,2745,3421,3864,4019,3866,3421,2763,1979,1198,529,86,0,71,531,1191,1969,2
742,3421,3864,4017,3865,3424,2761,1978,1199,534,86,0,71,530,1191,1968,2745,3419,
3861,4018,3867,3424,2763,1979,1197,531,87,0,73,528,1190,1968,2745,3421,3864,4020
,3865,3423,2763,1979,1197,531,86,0,75,530,1190,1966,2743,3420,3864,4017,3865,341
9,2763,1979,1199,531,87,0,71,530,1193,1967,2743,3419,3864,4017,3865,3424,2763,19
77,1198,533,87,0,72,529,1191,1971,2744,3419,3863,4017,3865,3423,2763,1979,1195,5
31,87,0,71,530,1191,1969,2749,3421,3863,4017,3865,3422,2763,1979,1198,531,87,0,7
1,530,1191,1966,2744,3424,3865,4017,3865,3419,2763,1979,1197,531,89,0,72,531,119
2,1968,2745,3419,3864,4017,3867,3422,2761,1978,1197,531,87,0,67,</M1>
<M2>2922,3134,3219,3269,3291,3299,3295,3278,3261,3234,3215,3191,3153,3115,3081,3
050,3018,2998,2985,2979,2985,2989,2998,2998,3001,2998,2993,2982,2962,2939,2914,2
895,2881,2871,2873,2877,2891,2904,2922,2934,2939,2947,2945,2944,2919,2903,2884,2
867,2857,2849,2852,2859,2872,2889,2907,2922,2927,2936,2935,2933,2911,2891,2877,2
859,2849,2843,2845,2854,2867,2885,2903,2916,2927,2933,2937,2930,2912,2891,2877,2
860,2847,2844,2844,2856,2867,2886,2903,2917,2927,2932,2935,2930,2909,2891,2874,2
859,2845,2843,2843,2854,2867,2883,2905,2914,2927,2931,2935,2930,2909,2891,2873,2
859,2848,2843,2846,2853,2870,2884,2905,2917,2924,2932,2934,2931,2907,2891,2874,2
858,2847,2842,2843,2853,2868,2884,2903,2918,2925,2933,2932,2930,2907,2889,2874,2
857,2847,2841,2845,2855,2869,2885,2904,2918,2927,2933,2936,2930,2909,2889,2875,2
857,2848,2842,2843,2853,2867,2885,2905,2918,2927,2933,2935,2928,2910,2891,2873,2
857,2845,2842,2843,2853,2867,2884,2904,2918,2928,2933,2935,2931,2910,2891,2875,2
858,2845,2841,2843,2850,2867,2883,2904,2916,2927,2933,2935,2931,2909,2891,2874,2
857,2847,2840,2844,2851,2867,2883,2903,2916,2925,2933,2935,2931,2910,2889,2874,2
859,2848,2843,2845,2853,2867,2885,2903,2918,2926,2934,2934,2928,2910,2891,2875,2
858,2847,2843,2845,2855,2866,2885,2903,2917,2926,2932,2935,2930,2910,2891,2875,2
858,2847,2843,2843,2855,2867,2883,2904,2917,2927,2932,2935,2931,2910,2890,2874,2
859,2846,2843,2843,2851,2867,2884,2903,2915,2927,2932,2934,2930,2907,2891,2873,2
860,2847,2842,2844,2853,2868,2883,2905,2916,2927,2933,2935,2934,2908,2891,2874,2
857,2848,2842,2846,2853,2868,2883,2903,2918,2925,2930,2933,2929,2907,2888,2873,2
855,</M2>
```

Figura 61. Contenido de los buffers de los ADCs a través del puerto serie

3.3.6. F6: Pruebas y PCB.

El hito correspondiente a esta fase es **H6-PCB y resultados**.

T6.1- PCB

Para realizar la PCB primero tuvimos que terminar las fases anteriores **F4 y F5** para poder tener un prototipo probado y asegurarnos de que el dispositivo no tiene ningún tipo de fallo debido a incompatibilidades, sobre todo en la asignación de pines.

Para esta tarea hemos usado el software KiCad, para trabajar en KiCad primero hay que definir un esquemático que refleje todas las conexiones que habrá en la PCB. Para ello tenemos que fijarnos en cómo queda la asignación de pines en nuestro fichero .ioc tal y como lo hicimos a la hora de montar el prototipo.

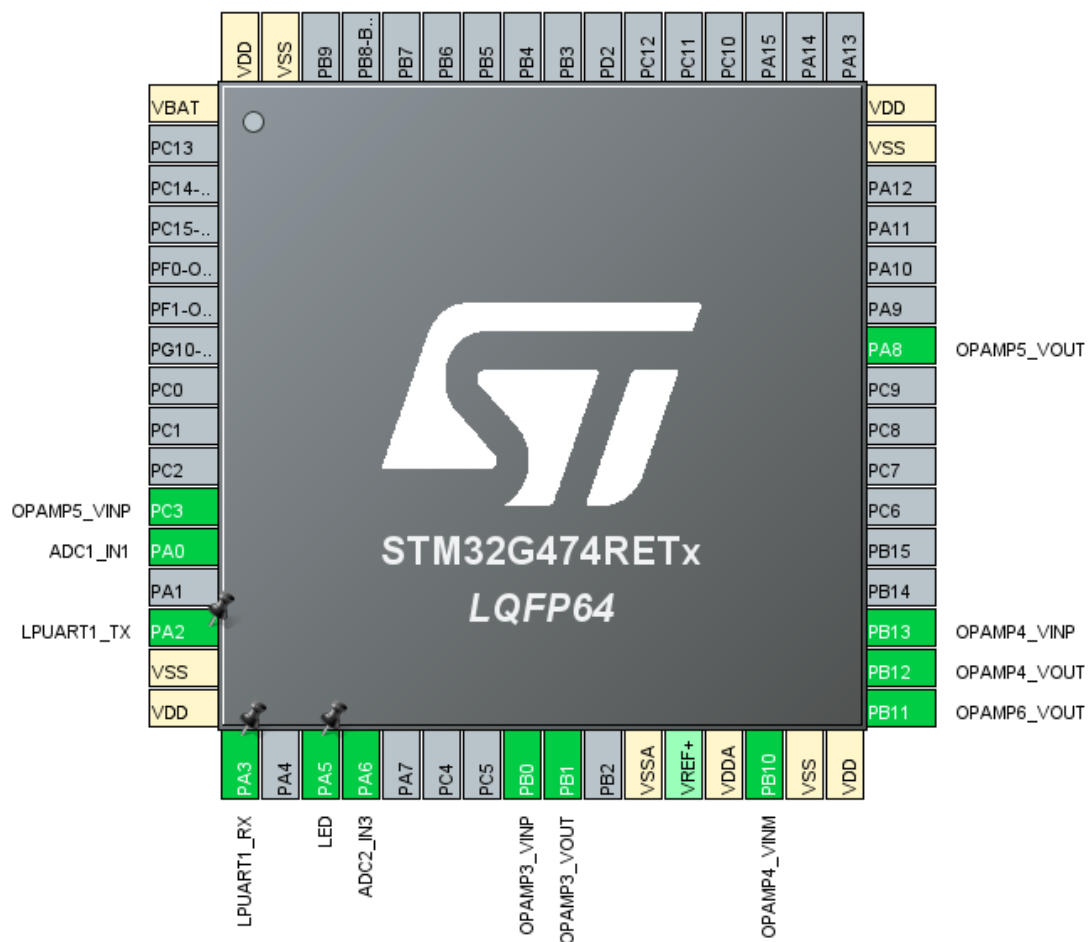


Figura 62. Asignación de los pines del sistema

Una vez tenemos claro que pines se conectan entre sí atendiendo al diseño del sistema. Conectamos los componentes al esquema del microcontrolador proporcionado por el tutor en KiCad.

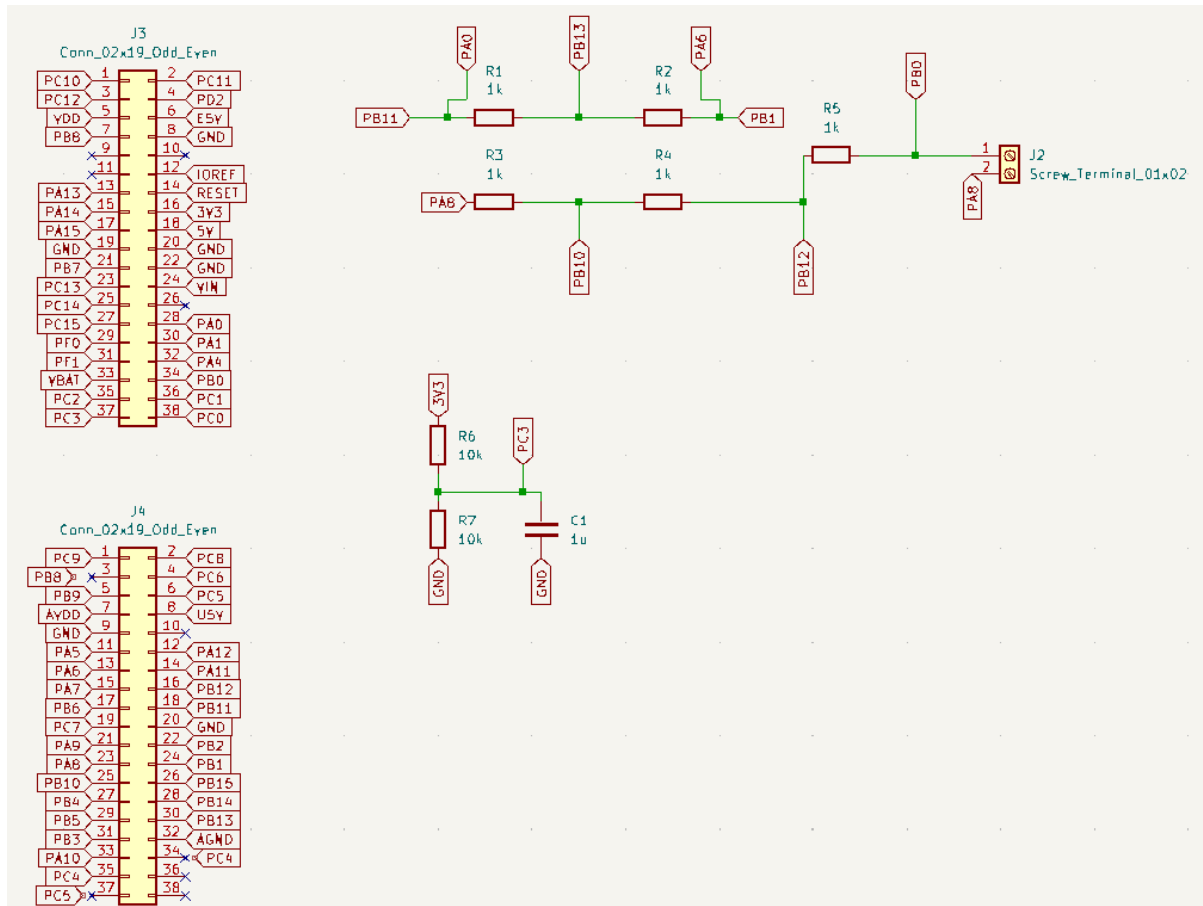


Figura 63. Esquemático de la PCB

Como se muestra en la imagen hemos conectado el esquema usando las etiquetas para las conexiones con los conectores del microcontrolador para una mayor facilidad y claridad. Las conexiones superiores corresponden a la implementación del sistema generador, mientras que las inferiores corresponden a la tierra virtual. Además, para facilitar las conexiones de las pistas hemos desconectado aquellas etiquetas de pines que se encuentran duplicadas.

Una vez realizado el esquema y comprobadas las reglas eléctricas, asignamos las huellas que vamos en el diseño de la PCB final para cada uno de los componentes.

Símbolo: Asignación de huellas		
1	C1 -	1u : Capacitor_SMD:C_1206_3216Metric
2	J2 -	Screw_Terminal_01x02 : TerminalBlock_Phoenix:TerminalBlock_Phoenix_MKDS-1,5-2-5.08_1x02_P5.08mm_Horizontal
3	J3 -	Conn_02x19_Odd_Even : Connector_PinHeader_2.54mm:PinHeader_2x19_P2.54mm_Vertical
4	J4 -	Conn_02x19_Odd_Even : Connector_PinHeader_2.54mm:PinHeader_2x19_P2.54mm_Vertical
5	R1 -	1k : Resistor_SMD:R_1206_3216Metric
6	R2 -	1k : Resistor_SMD:R_1206_3216Metric
7	R3 -	1k : Resistor_SMD:R_1206_3216Metric
8	R4 -	1k : Resistor_SMD:R_1206_3216Metric
9	R5 -	1k : Resistor_SMD:R_1206_3216Metric
10	R6 -	10k : Resistor_SMD:R_1206_3216Metric
11	R7 -	10k : Resistor_SMD:R_1206_3216Metric

Figura 64. Asignación de huellas de la PCB

El siguiente paso sería actualizar la placa desde el esquema y realizar las conexiones, el diseño de la forma, los agujeros de taladro y la distribución de los conectores del microcontrolador ha sido proporcionado por el tutor, nosotros nos encargamos de añadir y conectar el resto de componentes. Además, se han definido dos zonas de llenado donde la cara frontal de la PCB corresponde a tierra y la cara posterior corresponde a la alimentación (3V3).

El diseño final es el siguiente:

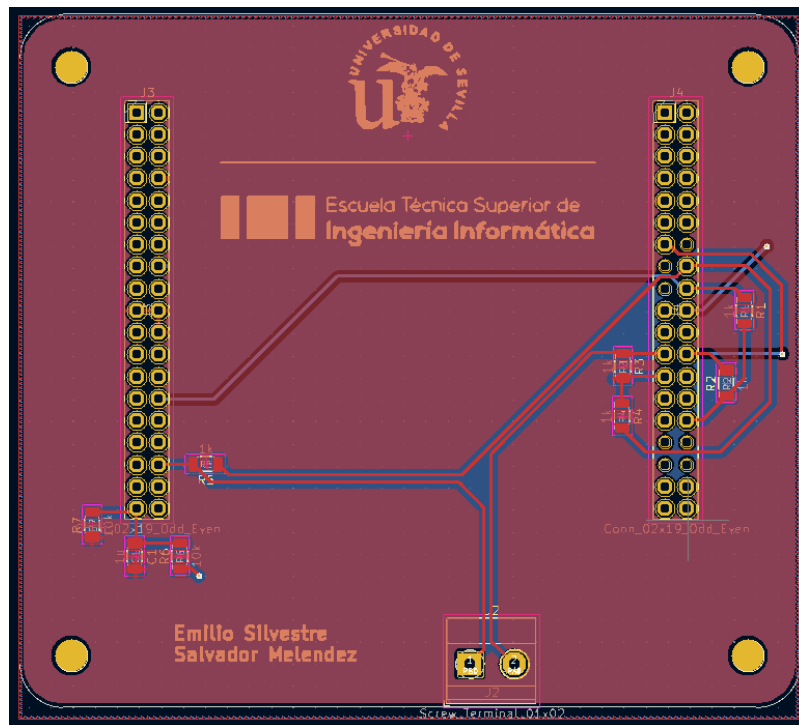


Figura 65. Diseño de la PCB

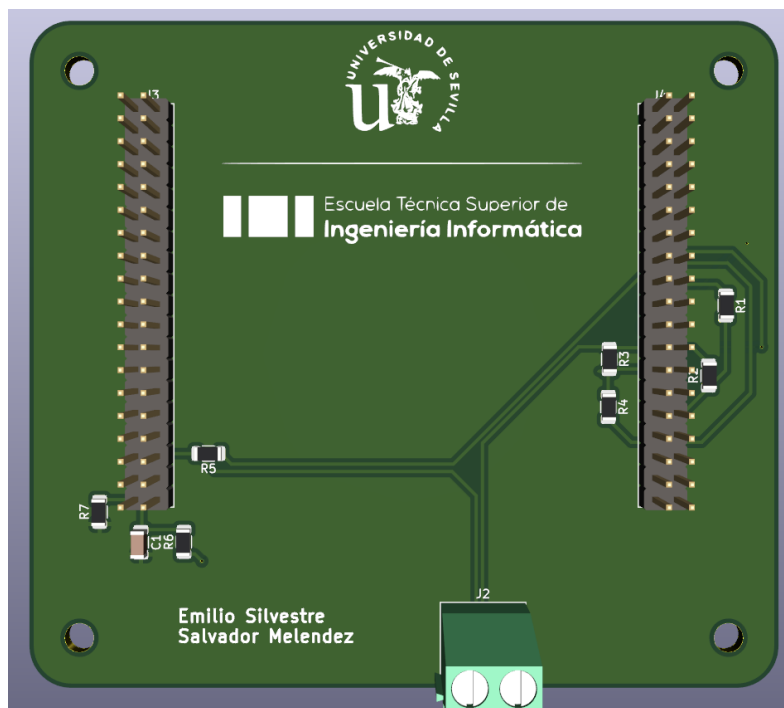


Figura 66. Diseño 3D de la PCB

T6.2- Pruebas

Con el prototipo finalizado realizamos una serie de pruebas adicionales para comprobar que la señal se ve afectado ante el paso de la corriente por una resistencia colocada en el conector externo, en estas pruebas decidimos visualizar el resultado usando el software waveforms y el osciloscopio Analog Discovery 2 para comprobar de una manera más visual el resultado sin usar la UART.

Para estas pruebas usamos 3 resistencias de $39\text{k}\Omega$, $10\text{k}\Omega$ y 330Ω cada una y las introducimos en el conector con el osciloscopio conectado tanto a la entrada (señal naranja) como a la salida (señal azul).

Los resultados son los siguientes:

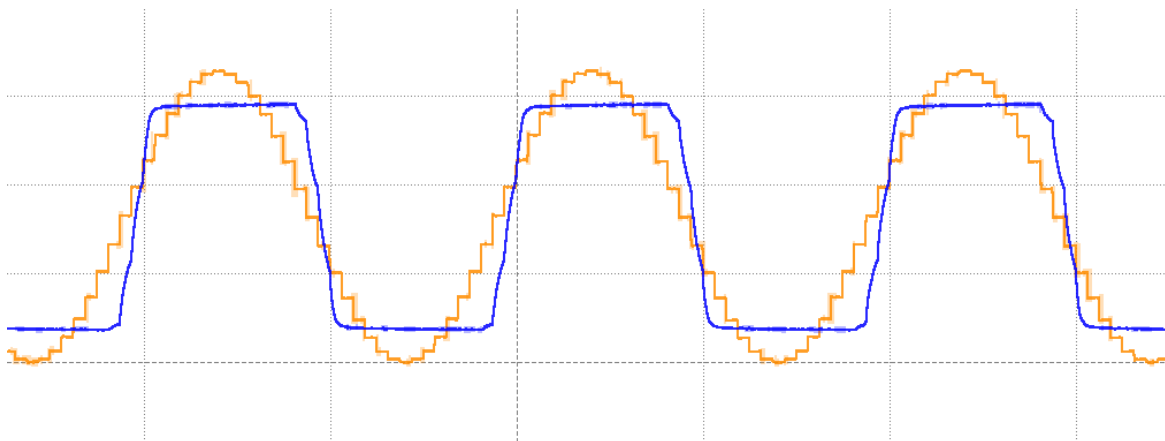


Figura 67. Resultado con una resistencia de 39k ohmios

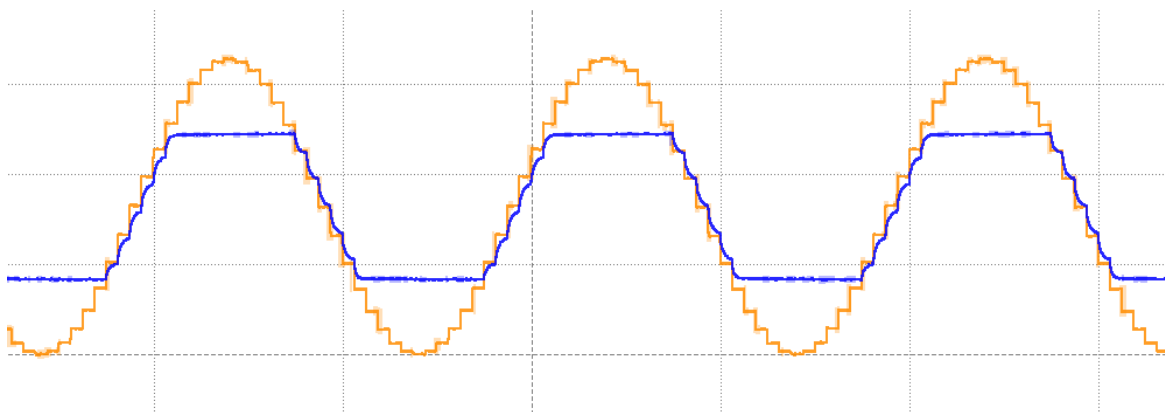


Figura 68. Resultado con una resistencia de 10k ohmios

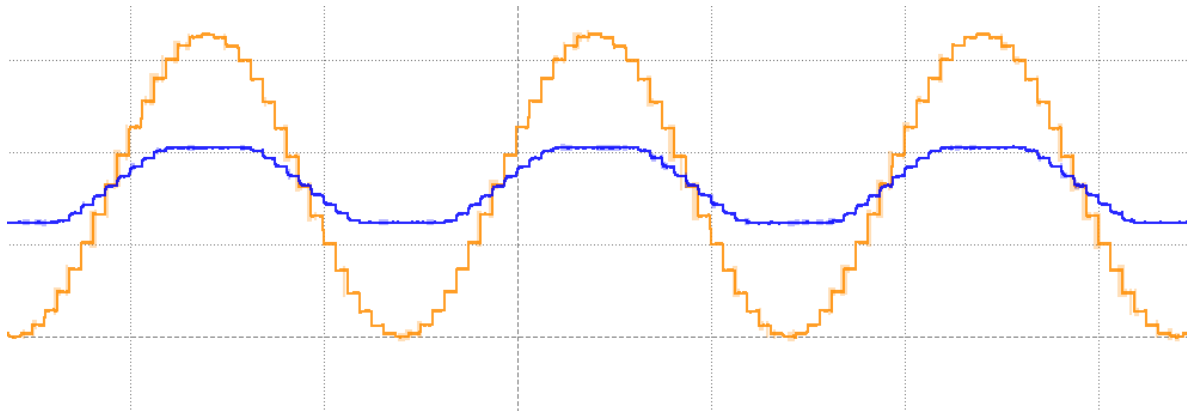


Figura 69. Resultado con una resistencia de 330 ohmios

T6.3- Informe

Decidimos recopilar todas las pruebas realizadas durante la ejecución del proyecto y elaborar un plan de pruebas donde se explica brevemente la prueba que se ha realizado, en qué fase y tarea se ha realizado y cuál ha sido el resultado.

3.4. Seguimiento de la planificación

Durante la ejecución proyecto se produjeron algunos inconvenientes recogidos dentro del análisis de riesgos que retrasaron y dificultaron la ejecución del proyecto. Hemos hecho un análisis de dichos riesgos:

Acontecimiento	Impacto	Producido	Comentarios
Baja por enfermedad puntual de algún miembro	Leve	Si	No ha requerido replanificación
Imposibilidad de trabajar por motivos personales	Leve	Si	No ha requerido replanificación
Errores desconocidos o comportamientos inesperados del sistema que impidan continuar	Medio	Si	Han retrasado el proyecto no más de 1 semana
Malfuncionamiento de un componente	Medio	No	Sin comentarios
Falta de tiempo para realizar alguna tarea	Grave	No	Sin comentarios
Cancelación de reunión por motivos del profesor	Leve	Si	No ha requerido replanificación

Desconocimientos sobre el funcionamiento de alguna parte del sistema o implementación	Medio	Si	No ha requerido replanificación
Comienzo de prácticas en empresa	Crítico	Si	Han retrasado el proyecto casi 2 meses
Solapamiento de trabajo con asignaturas pendientes	Medio	Si	Han retrasado el proyecto no más de 2 semana
Fallo en la planificación inicial	Grave	Si	Ha sido necesario replanificar el proyecto debido a dos de los riesgos

Destacar que el comienzo de las prácticas en empresa ha tenido un gran impacto en el proyecto ya que hemos decidido retrasar el proyecto para tener un periodo de adaptación más descargado, teniendo que retrasar casi 2 meses el proyecto en febrero y realizar varias sesiones dedicadas a retomar el ritmo de la ejecución del proyecto y realizar su replanificación.

La planificación temporal final del proyecto queda de la siguiente manera:

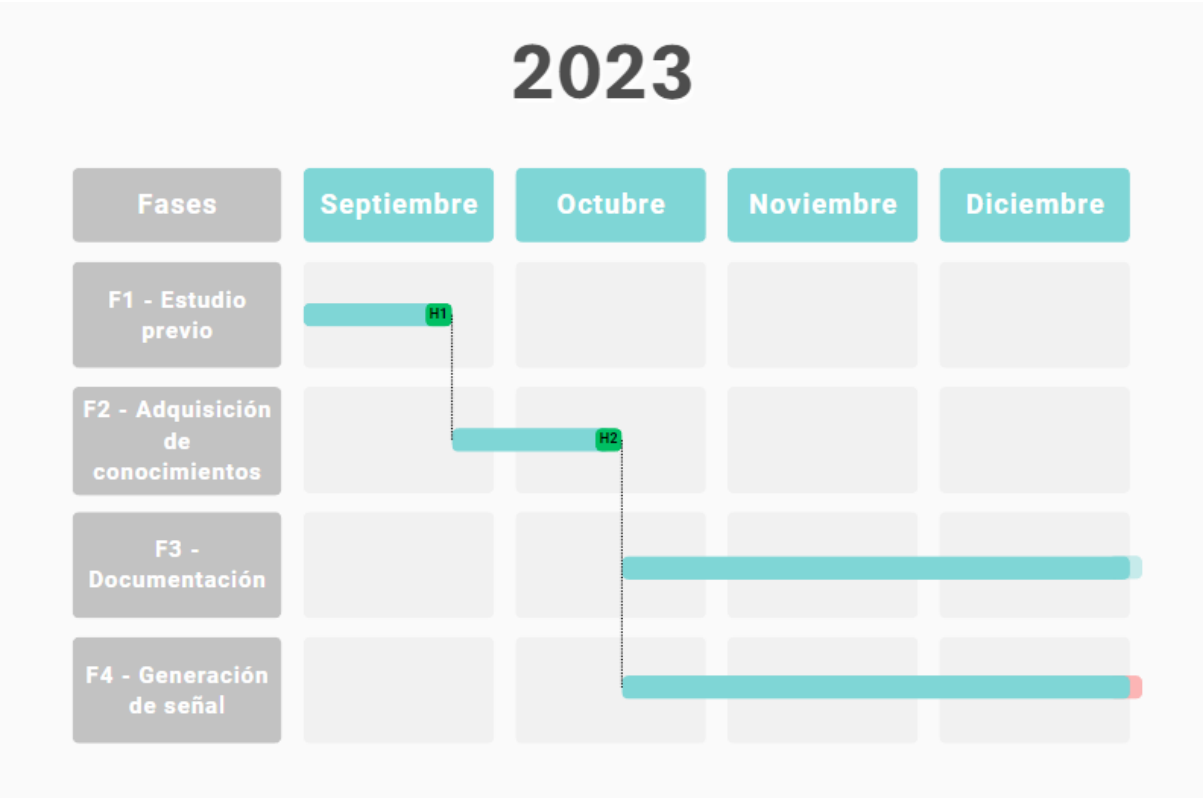


Figura 70. Diagrama de Gantt Replanificación 2023

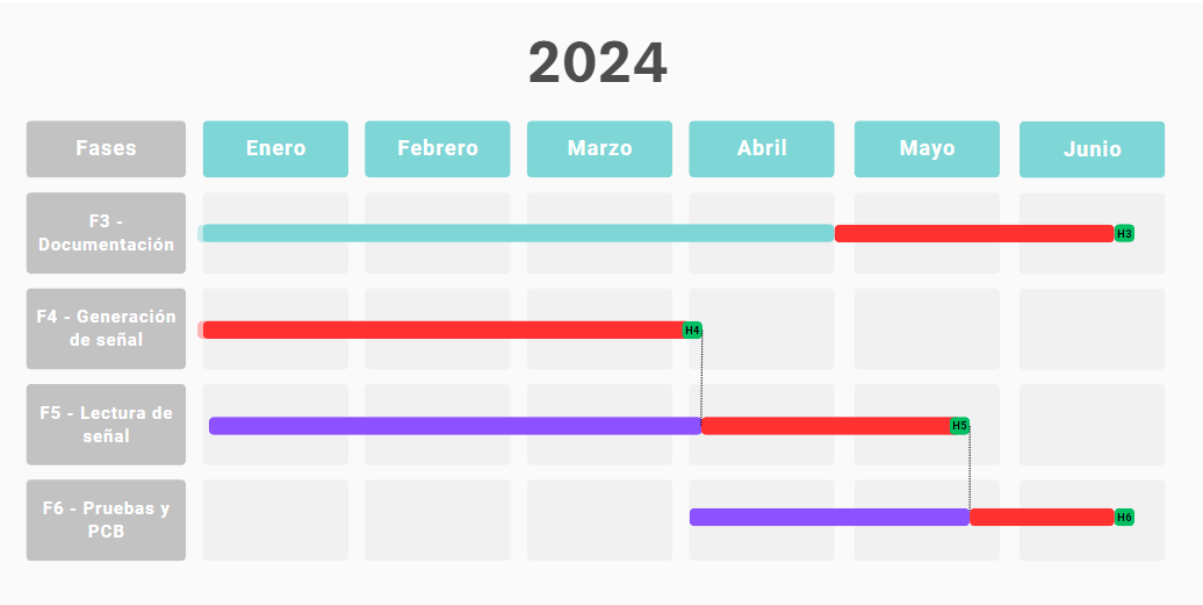


Figura 71. Diagrama de Gantt Replanificación 2024

Para interpretar los diagramas de Gantt de la planificación es necesario conocer que el **color azul** representa la ejecución de una tarea en el tiempo asignado, el **color rojo** representa un retraso en la ejecución de la tarea y el **color morado** representa un retraso en el comienzo de la tarea debido al retraso en una tarea anterior de la cual es dependiente.

Como se puede observar después del periodo de navidades el proyecto sufrió un grave retraso debido al comienzo de prácticas en empresa y a lo largo de los meses de abril y mayo también sufrió varios retrasos a causa de tareas de asignaturas pendientes.

3.5. Plan de pruebas

A continuación, exponemos un resumen de las pruebas realizadas a lo largo del proyecto:

Fase (Tarea)	Descripción	Resultado	Éxito
F4 (T4.1)	Comprobar generación de señal del DAC desde el osciloscopio	Imagen donde se observa una señal senoidal con la amplitud máxima	Si
F4 (T4.6)	Comprobar funcionamiento y ajuste correcto de la función ajustaTimer()	Conjunto de imágenes de WaveForms donde se aprecia la frecuencia de cada medición	Si
F5 (T5.2)	Comprobar la lectura de señal mirando el buffer de los ADCs al depurar el sistema	Imagen de los buffers que muestra un registro cuyos valores ascienden y descienden en forma de onda senoidal	Si
F5 (T5.3)	Comprobar la transmisión de los datos de los ADCs a través de la UART	Imagen del software PuTTY que muestra el contenido de la UART	Si
F6 (T6.2)	Probar distintas resistencias y medir que la señal de entrada se ve afectada de forma distinta por cada una de ellas	Imagen que muestra la señal de entrada y de salida para los distintos valores de resistencias	Si

4. CIERRE DEL DOCUMENTO

4.1. Conclusiones

Tras realización del proyecto nos gustaría valorar nuestras impresiones en varios aspectos:

Enfoque del proyecto

En primer lugar, nos gustaría recalcar que el enfoque del proyecto se ha centrado en aprender a implementar un sistema basado en un microcontrolador. Hemos priorizado la obtención y aprendizaje de nuevos conocimientos y técnicas por encima del rendimiento que aporta la realización del proyecto entre dos personas, queriendo participar ambos en todos los aspectos del proyecto.

Conocimientos obtenidos

A lo largo del desarrollo del proyecto, hemos podido trabajar de práctica muchos de los conocimientos relacionados con microcontroladores que hemos obtenido a lo largo del grado. Entre ellos, queremos destacar la implementación de DMA, el trabajo con componentes internos del microcontrolador, los aspectos de la investigación sobre un microcontrolador en concreto, la soldadura de componentes, el uso de herramientas alternativas a las tradicionales como el Analog Discovery 2 que ha servido como osciloscopio y aspectos teóricos relacionados con las señales senoidales.

Resultados del proyecto

Consideramos que el resultado del proyecto ha sido un éxito, tanto en la parte de aprendizaje como en el objetivo de la implementación del mismo, ya que el dispositivo es capaz de obtener mediciones a las que con un simple cálculo adicional en un script externo, como ya se ha detallado en uno de los puntos de esta memoria, se podría obtener la medición de la impedancia.

Recomendaciones en proyectos similares

Si tuviésemos que volver a realizar el proyecto tendríamos en cuenta varios aspectos que nos han tanto aportado como complicado el desarrollo, nos gustaría mencionar los siguientes:

- División del proyecto en fases: la división del proyecto en fases nos ha ayudado a organizar mejor el proyecto y separar en bloques con una definición clara de los objetivos, siendo así más fácil saber en qué punto del proyecto estábamos y replanificar el mismo en caso necesario.
- Planificación: la planificación ha sido clave en el proyecto, ya que al principio debido a la inexperiencia no teníamos claro la duración concreta de cada una de las tareas ni los posibles riesgos que se podían dar, pero debido a planificar la finalización del proyecto con bastante antelación hemos podido afrontarlo en su totalidad.
- Contacto con el tutor y el resto del equipo: el contacto para la resolución de dudas con el tutor ha sido una herramienta recurrente para desbloquear situaciones en determinadas fases del proyecto, que debido a la falta de conocimiento, hubiesen causado una ralentización del mismo. A su vez el poder contactar de manera remota entre miembros del equipo ha facilitado la ejecución del proyecto, ya que hemos podido utilizar el tiempo de los traslados hacia un entorno físico en la realización del proyecto y, además, gracias a los mecanismos de control de versiones hemos podido realizar cambios de manera segura en el proyecto.
- Ejemplos: revisar ejemplos en algunos casos es mejor que revisar directamente la documentación oficial, aunque es importante conocer el funcionamiento de cada componente, a la hora de implementarlos hay muchas maneras que escapan a nuestro conocimiento y que son ajenas al proyecto, pero debido a que son configuraciones posibles, quedan recogidas en la documentación oficial haciendo el documento más denso. De manera que en algunos casos resulta más fácil y rápido consultar ejemplos que muestren una implementación de los componentes que necesitamos.

4.2. Bibliografía

- [1] P. Pérez García y A. Yúfera García Dra Gloria Huertas Sánchez, «Circuit design for biomedical laboratories based on impedance measurement», 2019.
- [2] «Circuito+puente_2.bmp (361×460)». Disponible en: https://3.bp.blogspot.com/-YGzp33_N09I/UwsXAtVFCI/AAAAAAAAAR8/e6HYphLUe54/s1600/Circuito%2Bpuente_2.bmp
- [3] «MFIA 500 kHz / 5 MHz Impedance Analyzer | Zurich Instruments». Disponible en: <https://www.zhinst.com/europe/en/products/mfia-impedance-analyzer#introduction>
- [4] «SFB7 BIS Research Device | ImpediMed». Disponible en: <https://www.impedimed.com/products/research-devices/sfb7/>
- [5] «E4980A Precision LCR Meter, 20 Hz to 2 MHz | Keysight». Disponible en: <https://www.keysight.com/us/en/product/E4980A/precision-lcr-meter-20-hz-2-mhz.html>
- [6] «BNC to Minigrabber Cable - Digilent». Disponible en: <https://digilent.com/shop/bnc-to-minigrabber-cable/>
- [7] «GNSS-LiDAR: Drone 3D Mapping - YouTube». Disponible en: https://www.youtube.com/watch?v=cbczfgH1x0s&ab_channel=taroz1461
- [8] L. Folkertsma, L. Gehrenkemper, J. Eijkel, K. Gerritsen, y M. Odijk, «Reference-Electrode Free pH Sensing Using Impedance Spectroscopy», *Proceedings 2018, Vol. 2, Page 742*, vol. 2, n.º 13, p. 742, nov. 2018, doi: 10.3390/PROCEEDINGS2130742.
- [9] T. Tsuji, K. Shima, N. bu, y O. Fukuda, «Biomimetic Impedance Control of an EMG-Based Robotic Hand», 2010. doi: 10.5772/9184.
- [10] «emisilmer/TFGImpedancia». Disponible en: <https://github.com/emisilmer/TFGImpedancia>
- [11] «Features • STM32G4 microcontroller (Arm® Cortex®-M4 at 170 MHz) in LQFP64 package featuring:-128 KBytes of Flash memory and 32 Kbytes of SRAM for STM32G431RBT6-512 KBytes of Flash memory and 96 Kbytes of SRAM for STM32G491RET6-512 KBytes of Flash memory and 128 Kbytes of SRAM for STM32G474RET6 • Fully compatible with STM32G473RET6 (512 Kbytes of Flash memory and 128 Kbytes of SRAM)», Disponible en: www.st.com.

4.3. Galería de figuras

Figura 1. Impedancia en plano fasorial.....	4
Figura 2. Puente LCR [2]	6
Figura 3. MFIA Impedance Analyzer [3]	8
Figura 4. Impedimed SFB7 [4]	10
Figura 5. Keysight E4980A Precision LCR Meter [5]	10
Figura 6. Analog Discovery 2 con conector BNC [6]	17
Figura 7. Analog Discovery 2 con conector a pines	17
Figura 8. Esquema camino de resistencias con Demux.....	18
Figura 9. Diagrama PERT del proyecto.....	27
Figura 10. Diagrama de Gantt Planificación 2023	28
Figura 11. Diagrama de Gantt Planificación 2024	28
Figura 12. Diagrama Gantt Fase 1.....	29
Figura 13. Diagrama Gantt Fase 2.....	29
Figura 14. Diagrama Gantt Fase 3.....	30
Figura 15. Diagrama Gantt Fase 4.....	30
Figura 16. Diagrama Gantt Fase 5.....	31
Figura 17. Diagrama Gantt Fase 6.....	31
Figura 18. Estructura interna ADC	39
Figura 19. OPAMP	42
Figura 20. Estructura interna Amplificador Operacional	44
Figura 21. Estructura interna Tierra Virtual.....	45
Figura 22. Estructura del Sistema Generador de Señal [1]	47
Figura 23. Estructura interna Tierra Virtual.....	48
Figura 24. Esquema de la implementación del generador de señal.....	49
Figura 25. Esquema de la implementación de la tierra virtual	49
Figura 26. Esquema de la estructura del lector se señal	50
Figura 27. Script de python para generar la LUT	56
Figura 28. Script de python para generar LUT adaptada al DAC.....	56
Figura 29. Configuración Parameter Settings DAC	57
Figura 30. Configuración Parameter Settings Timer 3	58
Figura 31. Configuración Parameter Settings OPAMP 6.....	59
Figura 32. Señal senoidal resultante en osciloscopio.....	60
Figura 33. Configuración DMA Settings DAC	60

Figura 34. Proyecto ejemplo de DAC con DMA usado como modelo.....	61
Figura 35. Configuración Parameter Settings OPAMP 3.....	62
Figura 36. Configuración Parameter Settings OPAMP 4.....	63
Figura 37. Configuración Parameter Settings OPAMP 5.....	64
Figura 38. Resistencias emparejadas en el Sistema Generador de Señal	64
Figura 39. Parte delantera del prototipo del sistema	66
Figura 40. Parte trasera del prototipo del sistema	67
Figura 41. Puntos de medición en el prototipo.....	68
Figura 42. Configuración del reloj	74
Figura 43. Configuración del reloj	75
Figura 44. Resultado a una frecuencia de 100 mHz.....	77
Figura 45. Resultado a una frecuencia de 1 Hz	77
Figura 46. Resultado a una frecuencia de 10 Hz	77
Figura 47. Resultado a una frecuencia de 50 Hz	78
Figura 48. Resultado a una frecuencia de 200 Hz.....	78
Figura 49. Resultado a una frecuencia de 500 Hz.....	78
Figura 50. Resultado a una frecuencia de 50 kHz.....	78
Figura 51. Resultado a una frecuencia de 100 kHz	79
Figura 52. Resultado a una frecuencia de 500 kHz	79
Figura 53. Estructura del Sistema Generador de Señal	80
Figura 54. Configuración Parameter Settings ADC 1.....	81
Figura 55. Configuración DMA Settings ADC 1.....	82
Figura 56. Configuración Parameter Settings ADC 2.....	83
Figura 57. Configuración DMA Settings ADC 2.....	84
Figura 58. Conexiones de los ADCs en el prototipo	85
Figura 59. Contenido del buffer del ADC 1	86
Figura 60. Configuración Parameter Settings UART	87
Figura 61. Contenido de los buffers de los ADCs a través del puerto serie	89
Figura 62. Asignación de los pines del sistema	90
Figura 63. Esquemático de la PCB.....	91
Figura 64. Asignación de huellas de la PCB	92
Figura 65. Diseño de la PCB	93
Figura 66. Diseño 3D de la PCB.....	93
Figura 67. Resultado con una resistencia de 39k ohmios.....	94
Figura 68. Resultado con una resistencia de 10k ohmios.....	94

Figura 69. Resultado con una resistencia de 330 ohmios.....	95
Figura 70. Diagrama de Gantt Replanificación 2023.....	98
Figura 71. Diagrama de Gantt Replanificación 2024.....	98