Universidad de la República
Facultad de Ingeniería

Cancelación de Artefactos en Neuroestimuladores

Memoria de proyecto presentada a la Facultad de Ingeniería de la Universidad de la República por

Julián Evia, Rodrigo García, Josefina Schmitd

en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniero Electricista.

Tutor

Germán Fierro ..... Universidad de la República
Fernando Silveira ..... Universidad de la República

Tribunal

Santiago Martínez ..... Universidad de la República
Juan Pablo Oliver ..... Universidad de la República
Álvaro Gómez ..... Universidad de la República

Montevideo
viernes 24 mayo, 2024

Cancelación de Artefactos en Neuroestimuladores, Julián Evia, Rodrigo García, Josefina Schmitd.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 214 páginas.
Compilada el viernes 24 mayo, 2024.
http://iie.fing.edu.uy/

Agradecimientos

Queremos aprovechar esta oportunidad, en la cual finalizamos una etapa importante en nuestras vidas, para agradecerle a todos aquellos que colaboraron directa o indirectamente para nuestra formación. Desde distintos ámbitos, el soporte recibido fue fundamental para nosotros.

En cuanto al rubro académico, nos gustaría dar gracias a nuestros tutores Fernando Silveira y Germán Fierro por su orientación, apoyo y seguimiento día a día. También mencionar a Facultad de Ingeniería de la Universidad de la República, nuestra casa de estudios, por los conocimientos y oportunidades otorgados. En particular al Instituto de Ingeniería Eléctrica y al Grupo de Microelectrónica, por proporcionarnos los recursos y el espacio adecuados para llevar a cabo este proyecto.

A Focus, queremos agradecerle por su apoyo logístico, brindando sus instalaciones y herramientas.

A Marcelo Baru por su buena disposición a discusiones técnicas, que fueron de gran utilidad.

En especial queremos expresar nuestra gratitud a nuestra familia, amigos y parejas, aquellos quienes nos acompañaron desde el inicio y a lo largo de la carrera, brindando el aliento y apoyo necesario. Su compañía y paciencia en los buenos y malos momentos fueron los cimientos que nos sostuvieron para llegar a donde estamos hoy.

Esta página ha sido intencionalmente dejada en blanco.

Resumen

Durante las últimas décadas, los avances en el desarrollo de los *Dispositivos Médicos Implantables Activos* cobraron gran relevancia en el ámbito de la Medicina y la Ingeniería conjuntamente. En lo que confiere al Estado del Arte del rubro, actualmente las innovaciones apuntan al diseño de dispositivos en *Lazo Cerrado* en los neuroestimuladores. Este tipo de dispositivos introducen una nueva problemática característica. En la zona de estimulación se realizan medidas de la respuesta del tejido, con el fin de obtener realimentación sobre la terapia aplicada. La señal de estimulación filtrada por el cuerpo humano provoca la superposición de lo que se denominará *Stimulus Artifact*, con la señal neural de interés llamada *Evoked Compound Action Potential*. La primera resulta ser de órdenes de magnitud mayor a la segunda y no son separables en frecuencia, lo cual dificulta la adquisición.

El sistema CANE tiene como objetivo dar solución a dicho problema a partir de la *Cancelación de Artefactos*. Para esto implementa, basado en resultados de investigación recientes, una etapa de Hardware y Firmware para la eliminación del *Stimulus Artifact*, seguida de una etapa de Software donde se remueven los residuos remanentes. En la primera etapa, se diseñó una PCB para funcionar como shield sobre el kit de desarrollo con el MCU escogido, siendo este la plataforma NUCLEO-L552ZE-Q para el MCU STM32L552ZE. Por otro lado, la etapa de post procesamiento se implementó utilizando herramientas de *Python* en combinación con el instrumento *Analog Discovery 2*.

Como resultados, se recuperó un *Evoked Compound Action Potential* exitosamente teniendo una relación de amplitud de 56 dB con respecto al artefacto de estimulación.

Siglas

ABE	Analog Back End.
AD2	Analog Discovery 2.
ADC	Analog to Digital Converter.
AFE	Analog Front End.
AIMDs	Active Implantable Medical Devices.
API	Application Programing Interface.
CANE	Cancelación de Artefactos en Neuroestimuladores.
CMRR	Common Mode Rejection Ratio.
DAC	Digital to Analog Converter.
DBS	Deep Brain Stimulation.
DMA	Direct Memory Access.
DNL	Differential Non Linearity.
ECAP	Evoked Compound Action Potential.
FD	Fase de Despolarización.
FR	Fase de Repolarización.
FW	Firmware.
GPIO	General Port Input/Output.
GUI	Graphic User Interface.
HPP	Hiperpolarización Post Potencial.
HW	Hardware.
I2C	Inter-Integrated Circuit.
IC	Integrated Circuit.
ICMR	Input Common Mode Range.
INA	Instrumentation Amplifier.
INL	Integral Non Linearity.

Siglas

LDO	Low Dropout Regulator.
LSB	Least Significant Bit.
MCU	Micro-Controller Unit.
OSW	Output Swing.
PA	Potencial de Acción.
PCB	Printed Circuit Board.
PDN	Power Delivery Network.
PRA	Período Refractario Absoluto.
PRR	Período Refractario Relativo.
PSD	Power Spectral Density.
PSRR	Power Supply Rejection Ratio.
RAM	Random Access Memory.
SA	Stimulus Artifact.
SNR	Signal to Noise Ratio.
SPI	Serial Peripheral Interface.
SR	Slew Rate.
SW	Software.
SYS	System.
THD	Total Harmonic Distortion.
UI	User Interface.

Tabla de contenidos

Capítulo 1

Introducción y objetivos

1.1. Motivación

Los Dispositivos Médicos Implantables Activos cumplen un rol importante en la medicina, ayudando a mejorar la calidad de vida de pacientes cuyas dolencias no pudieron ser tratadas a través de los métodos habituales. Ampliar estudios en dicha área resulta, por lo tanto, de gran necesidad para las personas y a su vez supone retos interesantes desde el punto de vista ingenieril.

1.2. Neuroestimuladores de Lazo Cerrado

En primer lugar, se debe aclarar a qué refiere el término *Neuroestimulación*. El mismo se utiliza para describir la acción de estimular determinadas estructuras pertenecientes principalmente al Sistema Nervioso Central, compuesto por el cerebro y la médula espinal, con la finalidad de provocar una respuesta que resulte en un efecto terapéutico o ayude al diagnóstico del paciente. Por lo tanto, se denomina *Neuroestimulador* a los dispositivos electrónicos que utilizan impulsos eléctricos para lograr dicha estimulación.

Se recurre a estos dispositivos cuando las implicaciones de una enfermedad no pueden ser tratadas mediante los tratamientos típicos, es decir, farmacológicamente, o a través de fisioterapias o incluso cirugías. Es relevante resaltar que generalmente el tratamiento que otorga un Neuroestimulador no cura la enfermedad, sino que aligera o incluso puede llegar a eliminar los síntomas asociados a la misma. Los Neuroestimuladores se emplean en general para tratar dolores crónicos, para lo cual se estimula comúnmente la médula espinal; y reducir síntomas de enfermedades como por ejemplo Parkinson y Epilepsia, donde se utiliza la técnica llamada Deep Brain Stimulation DBS [1].

Son de particular interés para la tesis los Neuroestimuladores de lazo cerrado. Estos dispositivos se distinguen por implementar una interfaz bidireccional con el

[1]Procedimiento mediante el cual se estimula un área específica del cerebro a través de electrodos de un neuroestimulador que se introducen en el mismo.

tejido. En otras palabras, significa que, además de estimular, poseen la capacidad de adquirir señales eléctricas provenientes del tejido, lo cual se puede traducir en información.

A partir de la recolección de datos, los mismos pueden ser procesados para evaluar los resultados de la estimulación y en base a ello el dispositivo puede automáticamente tomar la decisión de ajustar la misma en tiempo real, en caso de ser necesario. La inclusión de la realimentación supone un avance en la terapia ofrecida al paciente, logrando en algunos casos una terapia más efectiva así como también la reducción de consumo. [2]

1.3. Actividad Neuronal y Estímulo

Es necesario ahondar en los fundamentos biológicos de cómo se propagan las señales a través del tejido, con el fin de comprender por qué resulta útil la estimulación desde el exterior.

Naturalmente, en todas las células existe el denominado *Potencial de Membrana* (E_m). Este término refiere a la diferencia de potencial presente en la membrana plasmática de la célula. En particular, las células nerviosas (como también las musculares) son *excitables*, lo que significa que tienen la capacidad de modificar dicho potencial de forma rápida y transitoria. Además, bajo determinadas condiciones este fenómeno puede ser propagado en forma de onda hacia células vecinas, generando así la transmisión de mensajes hacia distintas partes del cuerpo.

En las neuronas, la transmisión ocurre a través de los *axones*[3] luego de la generación de un *Potencial de Acción* (se verá a continuación qué significa este fenómeno). El potencial de membrana se da por la existencia de una desigualdad en la distribución de los iones dentro y fuera de la célula, y también por la permeabilidad de la membrana con respecto a iones de diferente especie. El interior de la célula tiene una carga más negativa de iones que el exterior de la misma, por lo que, tomando como referencia al exterior de la célula, el potencial de membrana es negativo y típicamente ronda entre los $-60mV$ a $-70mV$. Cuando el potencial de membrana se vuelve más positivo, por la acción de un pulso eléctrico, se dice que la célula se *despolariza*. Si la despolarización alcanza un valor umbral, el fenómeno se acelera y se alcanza un valor de despolarización máximo donde el potencial de membrana resulta positivo. Además, esto provoca que el efecto se propague hacia las células vecinas, las cuales también se despolarizarán. Se define entonces al Potencial de Acción PA como la generación de una despolarización mayor al valor umbral que resulta en la transmisión de la perturbación en forma de onda a través del axón. Las perturbaciones consecutivas en las siguientes células no perderán amplitud y no hay atenuación.

A continuación, en la Fig. 1.1, se observa la forma de onda característica de un PA.

[2] Referencias: [1], [2], [3], [4]

[3] *Prolongación de las neuronas especializadas en conducir el impulso nervioso desde el cuerpo celular o soma hacia otra célula.* [5]

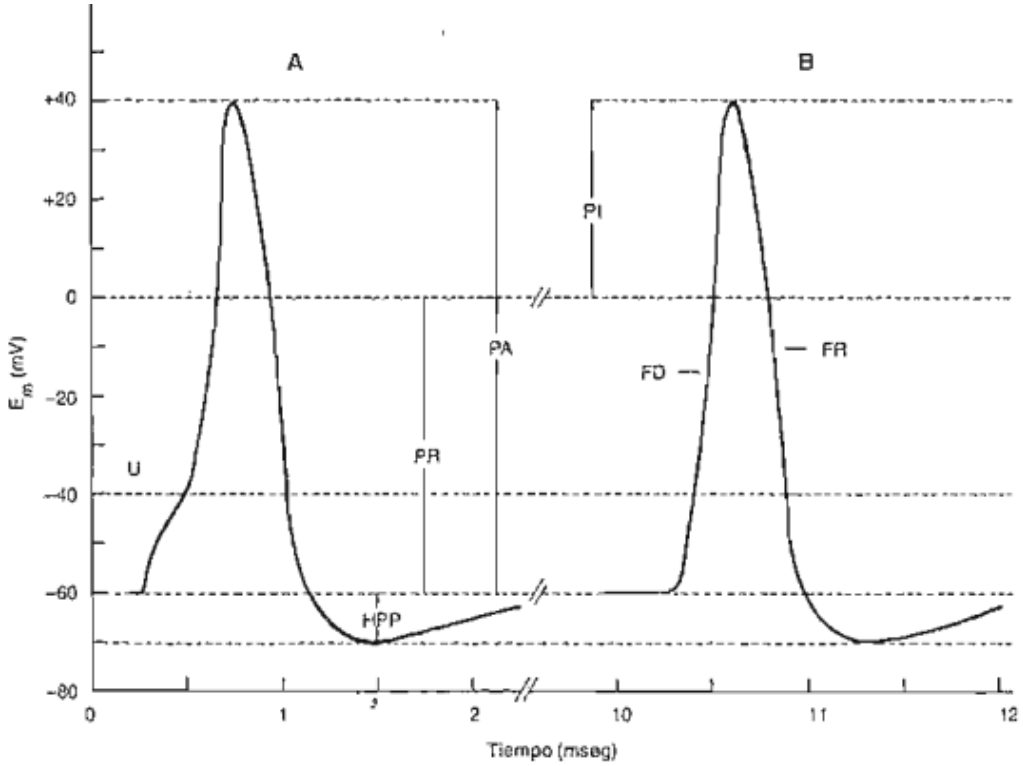


Figura 1.1: Forma de onda de un Potencial de Acción PA. (A) Corresponde a un PA medido próximo al sitio de estimulación. (B) Corresponde a un PA medido lejos de la zona de estimulación. Es la propagación de la perturbación original, la cual mantiene las mismas características. Imagen tomada de [6] Sección I, Pág 38. A continuación se explican las siglas que aparecen en ambas figuras, las cuales describen las distintas etapas características de un PA, en orden cronológico de aparición. Fase de Despolarización FD: el potencial de membrana se vuelve cada vez más positivo hasta llegar a 0 mV. PI (Potencial Invertido): El valor de potencial de membrana pasa a ser positivo. Fase de Repolarización FR: El valor de potencial de membrana se vuelve más negativo nuevamente. Hiperpolarización Post Potencial HPP: El potencial de membrana pasa su valor de reposo y se vuelve más negativo. PR : Valor del potencial de membrana en reposo.

Cuando el PA es provocado por la estimulación eléctrica del tejido desde el exterior, el mismo recibe el nombre en inglés de Evoked Compound Action Potential ECAP. En la Fig. 1.1.A se muestra un ECAP en la zona de estimulación. Se distingue al inicio la despolarización lenta producida por el estímulo y luego la despolarización de la célula que sobrepasa el umbral y da origen al ECAP.

Por otro lado, en la Fig. 1.1.B solo se muestra el ECAP que se propagó hasta una zona lejana a la zona de estímulo, donde la forma y la amplitud mantienen las características de la perturbación inicial. Es importante resaltar que, en caso de que la despolarización de la célula inicial no alcance el valor umbral, no ocurrirá un ECAP, la perturbación solo ocurre localmente; no hay propagación eléctrica. Para que un estímulo logre generar un ECAP, debe cumplir con ciertas características de intensidad y de duración. Generalmente, se espera que cuanto más débiles sean los

pulsos en cuanto a intensidad, mayor debe ser la duración del estímulo para lograr la generación del ECAP. Entonces, dado un tiempo, se encuentra una mínima intensidad con la cual debe ser entregado el estímulo para generar un ECAP. Se llamará a este valor de intensidad el *Umbral de Captura*.

Con la generación de un ECAP, aparece el Período Refractario Absoluto PRA que dura hasta la finalización de FR. Durante este período, ningún estímulo puede lograr la aparición de otro ECAP. A partir de aquí en adelante continúa el Período Refractario Relativo PRR, donde estímulos de mayor intensidad que el umbral pueden lograr la generación de un ECAP. En definitiva, el PRA impone el límite en la frecuencia máxima de estímulo.

1.4. Problema a Resolver

Como se mencionó previamente en la Secc. 1.2, en los neuroestimuladores de lazo cerrado es necesario cumplir con las funciones de estímulo y medida de señales provenientes del tejido simultáneamente. Dicha propiedad adhiere una complejidad extra al dispositivo cuando se desea realizar ambas acciones en la zona de estimulación, pues la información a medir (ECAP) se encuentra contaminada por la señal de estímulo. A esta contaminación se la denominará *Artefacto*, o su traducción al inglés Stimulus Artifact SA. El problema resulta, entonces, en encontrar una solución que permita obtener la lectura correcta de los ECAP a partir de la superposición de los mismos con los artefactos.

Otra dificultad a considerar en esta situación es que, la amplitud de la señal SA es de órdenes de magnitud mayores a la de la señal ECAP, por lo que podría causar la saturación de los dispositivos de medida.

A la remoción del artefacto es a lo que se llamará como *Cancelación de Artefacto*.

1.5. Objetivo General

El objetivo de este proyecto es generar una plataforma que requiera desarrollo Hardware (HW), Firmware (FW) y Software (SW), que permita realizar una prueba de concepto de un Sistema de Cancelación de Artefactos en Neuroestimuladores (CANE) basada en [2] y [7]. En estos trabajos, correspondientes a una tesis de doctorado, se implementa un mecanismo innovador de cancelación de artefactos en señales neurales que encuentra una solución al problema mencionado anteriormente. Este proyecto se inspira en dicha tesis para diseñar y desarrollar una plataforma de cancelación de artefactos.

1.6. Objetivos Específicos

A continuación, se listan los objetivos específicos a cumplir asociados a cada etapa de la solución:

- En una primera instancia se buscará la verificación de la efectividad de la solución propuesta mediante una plataforma de simulación.
- Como parte del proyecto, se desarrollará una plataforma de HW que realice la medición de las señales e implemente físicamente la solución propuesta para la reducción de los artefactos.
- Desarrollar FW que cumpla con las funciones requeridas según la solución propuesta.
- Implementar SW con el fin de tener una etapa de post procesamiento de la señal obtenida a la salida de la plataforma de HW. Esto facilitará la visualización y el análisis de los datos adquiridos, así como la evaluación de la efectividad de los métodos de cancelación de artefactos en todas las etapas.
- Investigar señales representativas del problema en bases de datos de casos de aplicación. Utilizar un generador de onda arbitraria para verificar luego el correcto funcionamiento del sistema aplicado a dichas señales.

1.7. Alcance del Proyecto

Dentro del alcance del proyecto se encuentra el cumplimiento de los objetivos especificados en la Secc. 1.6. A modo de agregado, quedando por fuera del alcance y en caso de ser posible, se evaluará llevar a cabo pruebas en suero fisiológico o un medio similar. Esto permitiría caracterizar el comportamiento del sistema desarrollado en condiciones ambientales más cercanas a las correspondientes en la aplicación final. Dado que el proyecto supone una prueba de concepto, quedan por fuera del alcance las consideraciones de diseño propias de Active Implantable Medical Devices (AIMDs por su sigla en inglés), como por ejemplo, la reducción de consumo, análisis de riesgos y aplicación de medidas de seguridad.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 2

Solución Propuesta

2.1. Introducción

Previamente a entrar en detalle sobre la estructura de la solución, es importante introducir de forma genérica las hipótesis bajo las cuales surje la misma. A su vez, es necesario resaltar los principales elementos de su composición.

En definitiva, la solución se encuentra constituida por dos etapas principales (ver Fig. 2.1). La primera de ellas es una etapa que incorpora tanto Hardware como Firmware, con la cual se espera recuperar casi en su totalidad la señal de interés. Para mayor eficiencia y exactitud en su recuperación, se agrega una segunda etapa que adhiere procesamiento en Software. De esta forma, los residuos que quedan presentes en la salida de la etapa de Hardware y Firmware se reducen o eliminan para revelar aún mejor la señal de ECAP obtenida.

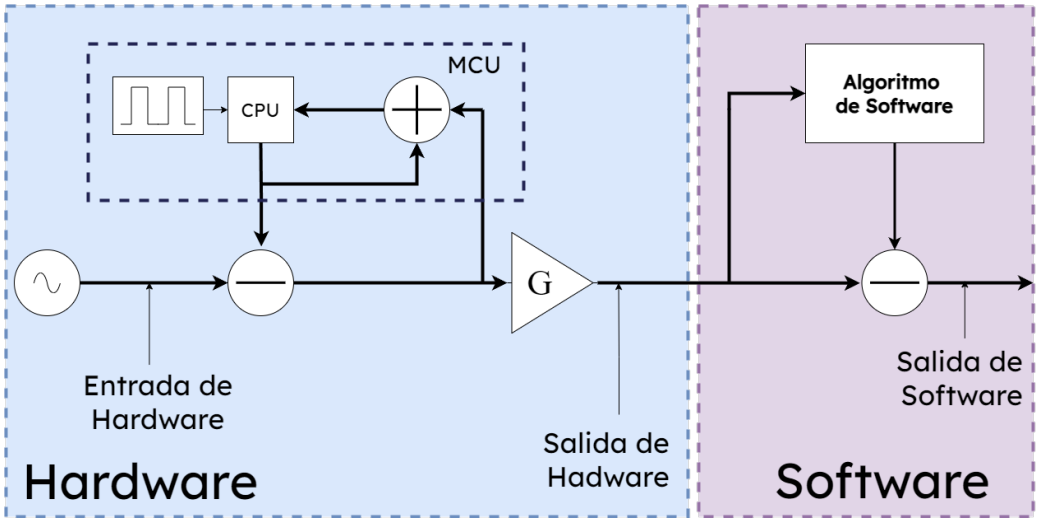


Figura 2.1: Diagrama de bloques del sistema completo.

Resulta relevante resaltar, también, qué procesos se llevan acabo en cada una de estas etapas para cumplir con las funciones estipuladas en el párrafo anterior.

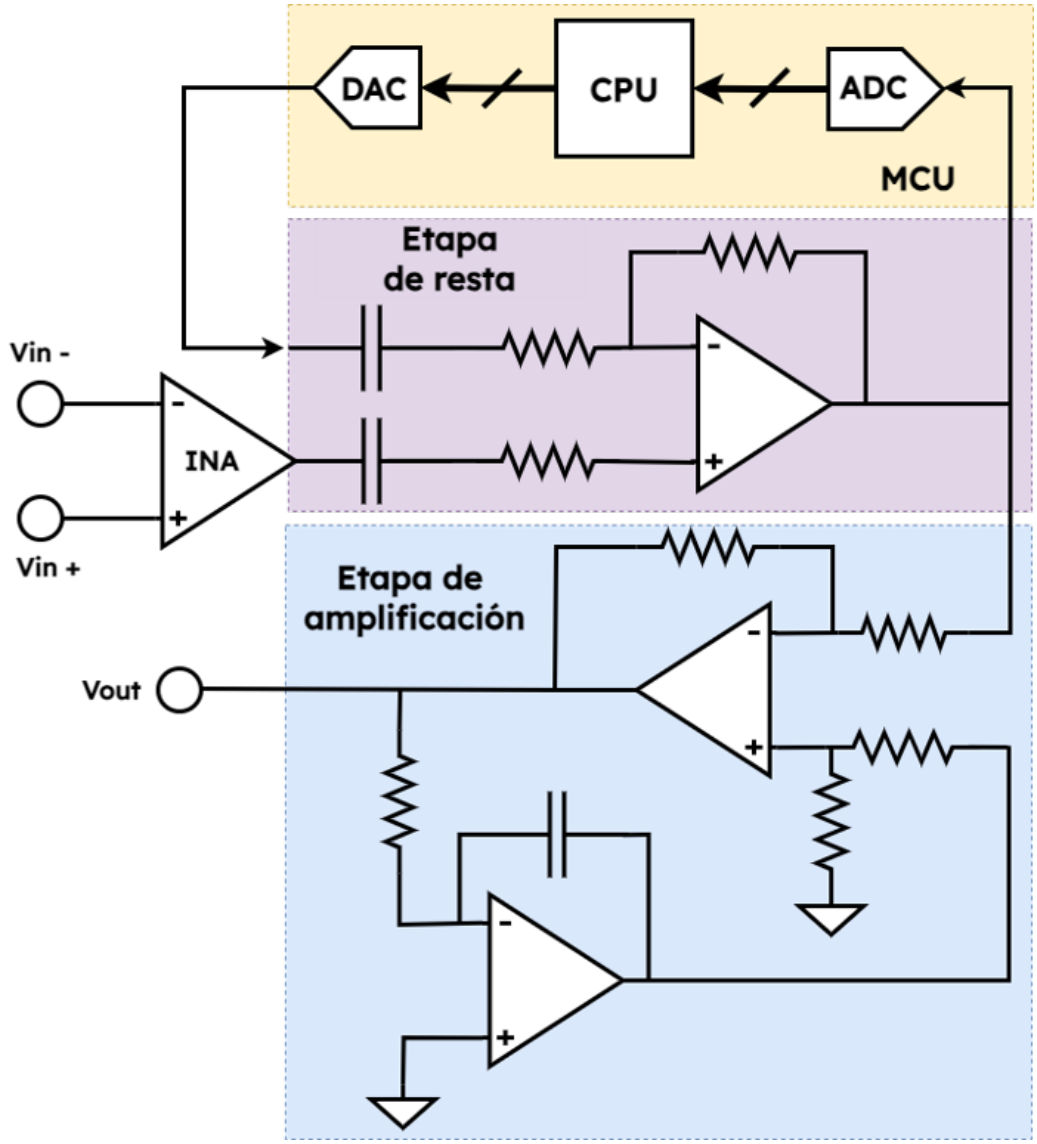


Figura 2.2: Estructura de la etapa de Hardware y Firmware.

La idea es generar un *Template* o *Plantilla* independientemente en cada etapa, el/la cual luego será restado/a a la señal de entrada de su respectiva etapa. El Template se define como la representación digital de las muestras obtenidas en cada una de las etapas, se verá ahora qué se muestrea en cada caso. En la etapa de Hardware y Firmware, las muestras son aquellas que se obtienen del muestreo que realiza el ADC mostrado en la Fig. 2.2, donde se visualiza en detalle la estructura de esta etapa. Dicho Template será almacenado por el MCU, y más adelante reproducido por el mismo a través del DAC. Ambos conversores deben ser estrictamente diseñados para prevenir la captura de una señal de amplitud equivalente a la máxima del rango admisible por la ECAP. En otras palabras, su apreciación debe ser mayor a dicha amplitud. Es así que, mientras las señales de ECAP y SA

entran en el sistema a la primera etapa, solo la SA es muestreada y almacenada en el Template. Como resultado, únicamente la SA termina siendo restada a la entrada, y la ECAP es entonces recuperada a menos de ruido y residuos remanentes de la cancelación. Esta funcionalidad, es lograda a partir de la combinación de la primera y la segunda etapa de amplificación de la etapa de Hardware y Firmware, en conjunto con procesamiento desde un MCU. Una vez que se efectuó la cancelación del artefacto, la tercera etapa de amplificación se utiliza para un mejor reconocimiento y manipulación de la señal obtenida que contiene a la ECAP. Debe ser notado que, en esta tercera amplificación es donde se da el mayor riesgo de saturación de los circuitos analógicos durante funcionamiento en régimen del sistema. En cuanto a la etapa de Software, el Template es el resultado de un muestreo mucho más rápido que el anterior, y en este caso, de la señal proveniente de la etapa de Hardware y Firmware.

Por otro lado, la interacción entre las dos etapas principales se basa en las hipótesis de que no todas las señales de estimulación causan una ECAP. Se tomará como hipótesis, que es la amplitud de la señal SA la propiedad de la cual depende este fenómeno. En particular, si ambas señales, tanto ECAP como SA, entrarán al sistema durante todo el proceso de cancelación de punta a punta, la ECAP recuperada y amplificada a la salida de la primera etapa sería muestreada por el Software y consecuentemente restada también. Para evitar este problema, primero se ingresará al sistema solo con una señal SA tal que su amplitud teóricamente no cause un ECAP pero esté cerca de hacerlo. Inmediatamente, el Template de Hardware y Firmware debe comenzar a ser adquirido. Una vez que el Template se obtuvo y se restó a la entrada logrando una convergecia estable a la salida, se indica al Software que comience a muestrear dicha salida estable para generar su Template. A continuación, una vez que el Software termina de generar su Template, el mismo deja de muestrear y a la entrada de la etapa de Hardware y Firmware se ingresa una SA que genere ECAP. La forma de hacer esto es, escalar la señal de SA hasta el mínimo de amplitud de la misma que logre generar un ECAP. Al mismo tiempo, el Template de Hardware y Firmware también es escalado. ES importante aclarar que, el factor de escala es relativamente pequeño, puesto que se asume que el comportamiento de las señales permanece lineal en un pequeño rango. Adicionalmente, la ECAP es introducida a la entrada en combinación con la nueva SA. Esto permite que el Software reste el Template que generó previamente (el cual no contiene a la ECAP) a la nueva salida de la etapa de Hardware y Firmware, recuperando finalmente la señal de interés a su salida.

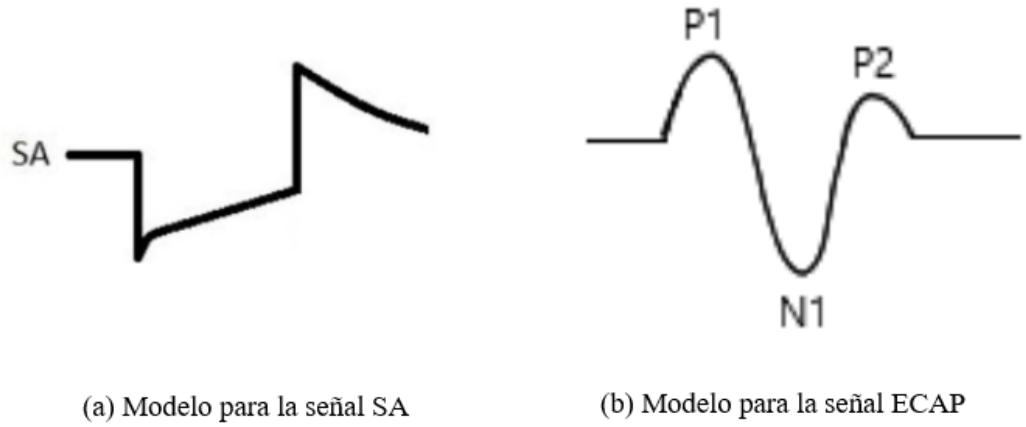
2.2. Modelos a Utilizar para las Señales de Interés

Las señales con las cuales se trabajará deberán seguir el comportamiento determinado por los modelos estipulados por un tercero. En este caso, los modelos fueron otorgados al proyecto por un grupo de investigación internacional que trabaja en colaboración con el Grupo de Microelectrónica de la Facultad de Ingeniería de la Universidad de la República.

En los siguientes párrafos se describe la información que fue recibida. El proceso

de construcción de las señales a partir de la interpretación de los datos se detalla más adelante en este documento, en la sección Secc. 3.3.

Previo a definir las características más específicas de cada señal, se enseña en la Fig. 2.3 los modelos de ambas señales para determinar su morfología.



(a) Modelo para la señal SA (b) Modelo para la señal ECAP

Figura 2.3: Modelo a utilizar para las señales de interés.

2.2.1. Stimulus Artifact

Para emular la señal de SA, se deberá considerar una señal compuesta de dos fases, una negativa y otra positiva, como muestra la Fig2.3.A. La fase negativa, debe tener un comportamiento lineal seguido de un comportamiento exponencial. La fase positiva debe ser modelada como una caída netamente exponencial. Además, debe cumplir con las características listadas a continuación.

- Amplitud $V_{pp} = 70 \dots 240\ mV$
- Ancho de banda $BW_{SA} = 16\ kHz$
- Frecuencia de estimulación, $f_{SA} = 900\ Hz$
- Duración de fase negativa 300 μs
- Constante de tiempo características (modelo para ambas fases), $\tau_{SA} = 250\ \mu s$
- Pendiente de la fase negativa $1/800\ V_{pp}/\mu s$

2.2.2. Evoked Compound Action Potential

En cuanto a la señal de ECAP, se debe considerar una señal sinusoidal de $1{,}2kHz$ atenuada temporalmente utilizando funciones exponenciales, con el objetivo de recrear la morfología del ECAP. En particular, se deben recrear las ondas P1, N1, y P2 como se muestra en la Fig.2.3.B. Las características específicas de esta señal son las siguientes.

- Amplitud Vpp ECAP $= 2 \dots 150uV$
- Ventana de ocurrencia del frente de ondas del ECAP (P1, N1, P2)
  - La onda N1 debe ocurrir según una distribución normal $\mathcal{N}(\mu, \sigma^2)$ luego del comienzo de la fase negativa del SA con:
    - $\mu = 470\ \mu s$, $\sigma = 62\ \mu s$
  - Ancho de banda entre 300 Hz y 5 kHz

2.2.3. Modelo Modo Común

La adquisición de señales neurales se realiza por medio de la medida de la diferencia de potencial entre dos electrodos uno positivo y uno negativo, E_+ y E_- respectivamente. El artefacto así como la señal neural, corresponden a la componente diferencial, $V_D = E_+ - E_-$, de esta medida. Sin embargo, la estimulación también genera variaciones en el modo común, es decir $V_{CM} = \frac{E_+ + E_-}{2}$, el cual hay que tener en cuenta en la etapa de adquisición para garantizar la correcta adquisición de la SA y ECAP sin ocasionar la deformación de estas señales.

Por ende, además de la descripción de las señales de interés, fue proporcionado un modelo del modo común ocasionado por la misma estimulación que generaría la SA. El modelo consistió en un circuito cuya implementación en LTSpice [8] se puede ver en la Fig. 2.4. Donde:

- Las fuentes I_1 e I_2 generan una corriente I_{stim} que varia entre 0,2 mA y 2,5 mA durante la fase negativa de la SA definida en la Secc. 2.2.1. Esta corriente, carga los condensadores C_1 y C_2 durante dicha fase.
- Durante las fase positiva de la SA, las fuentes de corriente se apagan y se cierran los *switchs* S_1 y S_2 por medio de la señal de control V_{CTRL}, que previamente se encontraban abiertos. Durante esta fase, un voltaje DC constante, V_{MID} variable entre ± 2 V, se aplica sobre los electrodos E_- y E_+.

Por lo tanto, en el modelo se establece una componente DC del modo común dada por V_{MID}, así como una componente en señal cuya amplitud varía en función de la corriente I_{stim}. En la Fig. 2.5 se pueden observar los resultados de simular este circuito durante tres periodos de estimulación, 3,3 ms, en régimen, con la máxima corriente de estimulación y un $V_{MID} = 2$ V.

En el modo común, se obtiene una señal con una excursión pico a pico de 1,47 Vpp, desde 2 V hasta 3,47 V.

También es de interés tener en cuenta el caso en que las fuentes de corriente I_{stim} se encuentren en sentido contrario entre tierra y los electrodos. Con esta consideración, y con V_{MID} en -2 V, se obtiene una morfología análoga como se puede ver en la Fig. 2.6 con 1,47 Vpp, desde -2 V hasta -3,47 V.

A partir de estas simulaciones, se determinó que el modo común a la entrada del sistema puede contener una componente DC variable entre ± 2 V y variaciones

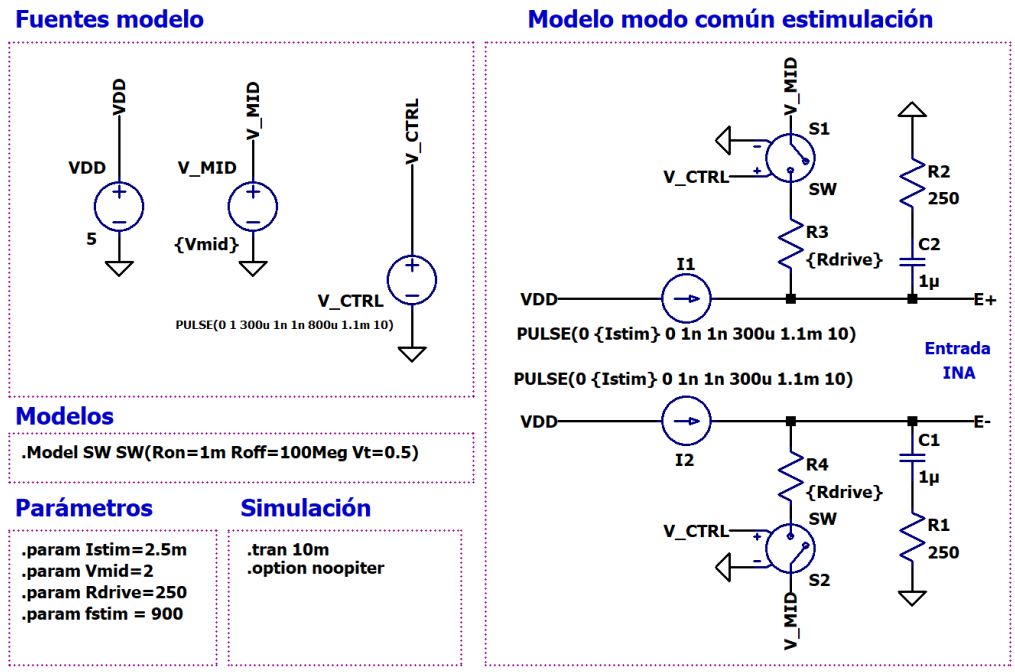


Figura 2.4: Modelo del modo común de la estimulación implementado en LTSpice a partir de las especificación proporcionadas por un tercero.

en señal de excursiones máximas de 1,47 Vpp, variando en total desde −3,47 V hasta 3,47 V.

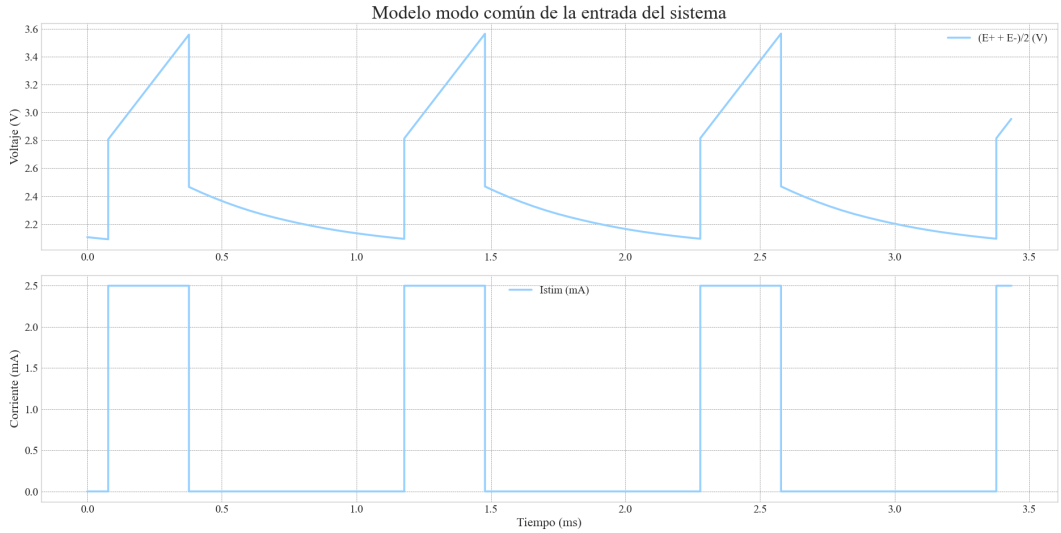


Figura 2.5: Resultados de la simulación realizada con el circuito de la Fig. 2.5. para una corriente de estímulo de 2,5 mA y un voltaje DC configurado en 2 V

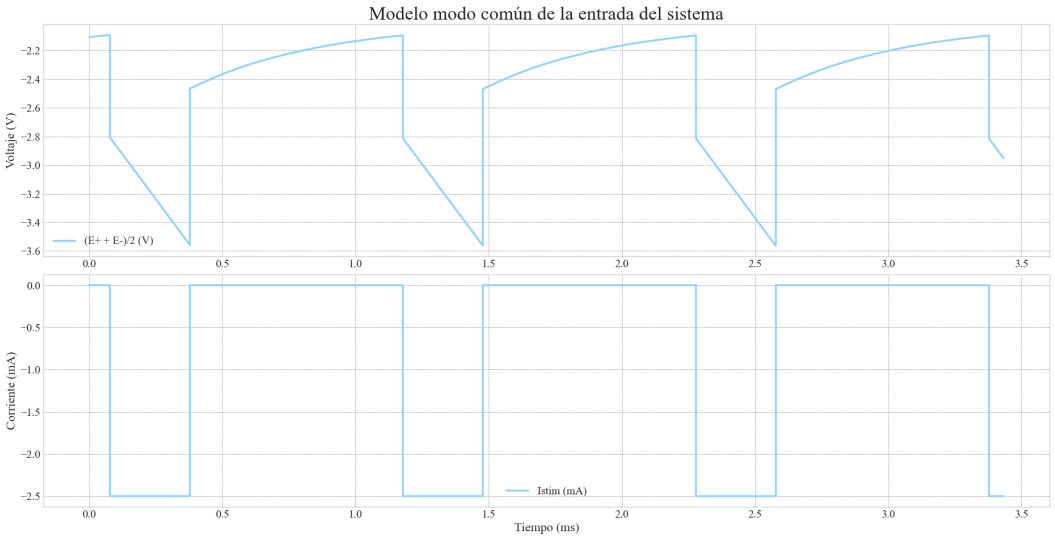


Figura 2.6: Resultados de la simulación realizada con el circuito de la Fig. 2.5. para una corriente de estímulo de $2{,}5\,\mathrm{mA}$ en sentido contrario y un voltaje DC configurado en $-2\,\mathrm{V}$

2.3. Arquitectura del Sistema

Recordando la Secc. 2.1, la Fig.2.1 muestra el diagrama funcional básico de la solución propuesta inspirada en el artículo [2]. El sistema cuenta cuenta con dos grandes módulos, el HW+FW y SW. El módulo de HW consta de amplificadores y filtros analógicos, microcontrolador (MCU) con FW específico para el algoritmo de cancelación de artefactos. Este microcontrolador cuenta con un convertidor analógico a digital (ADC) y un convertidor digital a analógico (DAC) donde el conjunto constituye el dispositivo de cancelación de artefactos.

En la salida del módulo de HW se adquieren datos y se los transmite al módulo de SW donde se procesan nuevamente, para reducir aún mas el artefacto residual.

Ambos diseños de módulos aprovechan el hecho de que los artefactos de estimulación son periódicos y relativamente similares en forma y amplitud, ya que la estimulación neural se realiza comúnmente mediante trenes periódicos de pulsos con amplitudes de pulso, anchos y retrasos entre pulsos constantes.

2.3.1. Módulo de Hardware + Firmware

La implementación de alto nivel de este módulo se vió en la Fig.2.2.

A continuación, se explicarán los cuatro bloques importantes que se pueden identificar, aunque el detalle se discutirá en el Cap.4.

- **Amplificador de Instrumentación (INA):** El objetivo del mismo es obtener el artefacto y la señal neural con bajo ruido y que ser capaz de manejar las componentes en modo común de las mismas.

- **Amplificador en configuración diferencial:** En esta etapa se produce la resta del artefacto.
- **Microcontrolador:** Se encarga de obtener mediciones utilizando un ADC y reproducir el resultado del algoritmo iterativo por un DAC.
- **Etapa de amplificación:** Se utiliza para amplificar la señal neural, en esta etapa es donde se corre el riesgo de la saturación del amplificador.

El diseño de HW reduce la amplitud del artefacto con el objetivo de no saturar el amplificador a la salida del módulo, sin alterar las señales neurales. Esto se logra a través de un bucle iterativo. El mismo consiste en almacenar el artefacto como una Plantilla o Template inicial (ver Secc. 2.1). Luego, se actualizará iterativamente la plantilla basada en las diferencias medidas, hasta que el artefacto entrante y la plantilla almacenada converjan a un valor tal que el error obtenido no sature el amplificador en la salida. Finalmente la plantilla final se resta a todos los artefactos siguientes, y el resultado se amplifica.

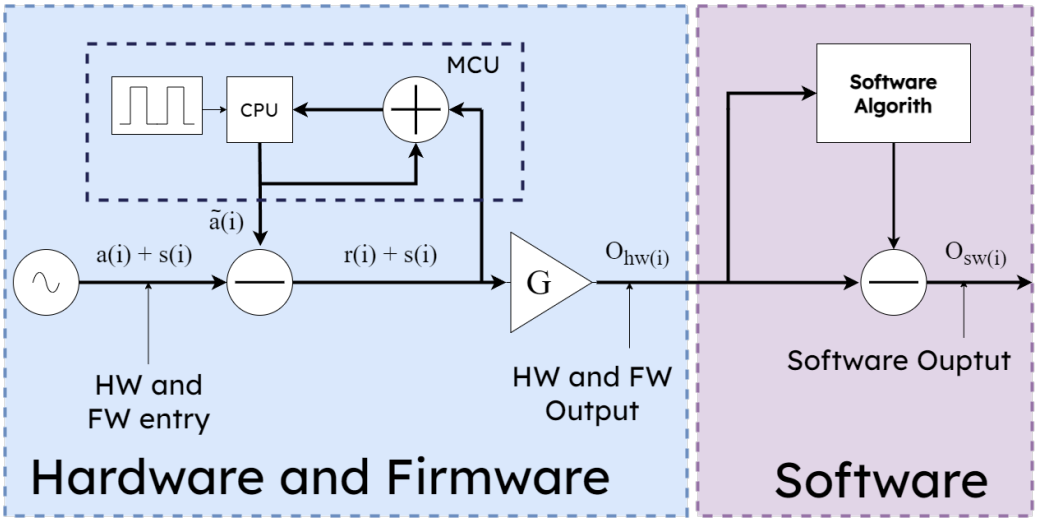


Figura 2.7: Arquitectura general del sistema completo con análisis matemático.

La salida del módulo de HW se puede ver en la Fig.2.7. En la instancia de estimulación *i-ésima* después de la convergencia ($i \geq C$) puede describirse como:

$$o_{hw}(i) = (a(i) + s(i) - \tilde{a}(C)) \times G = (\Delta a(i) + s(i)) \times G \quad , i > C \tag{2.1}$$

Donde:

- $s(i)$: es la señal neural subyacente al artefacto.
- $a(i)$: representa una única señal de artefacto.
- $\tilde{a}(C)$: es la plantilla del artefacto luego de la convergencia al ser reproducida por el DAC.

- G: es la amplificación a la salida del módulo de HW.

Por lo tanto $\Delta a(i) = a(i) - \tilde{a}(C)$, es el residuo $r(i)$ entre el artefacto entrante y la plantilla obtenida de la iteración. El objetivo es disminuir el residuo de forma de obtener a la salida la señal neural amplificada $s(i) \times G$.

El artefacto residual, $r(i)$ puede ser muy grande comparado con la señal neural de interés, sin embargo no habrá pérdida de información siempre y cuando la salida de HW sea menor que el voltaje de saturación $V_{hw_{sat}}$ del amplificador.
Asumiendo que $r(i) >> s(i)$, se requiere que:

$$r(i) \times G < V_{hw_{sat}} \tag{2.2}$$

Por lo tanto, existe un valor máximo que puede tomar el residuo del artefacto, esto impone una mínima resolución de ADC y DAC, se entrará más en detalle sobre esta discusión en el Cap.4.

Otra consideración a tener en cuenta sobre el ADC es que su apreciación (un LSB), debe ser mayor que la amplitud máxima de la señal neural, de esta manera se asegura que el ADC solo muestreé el artefacto. Por ende, durante el algoritmo, del microcontrolador solo cancelará el artefacto contaminante.

2.3.2. Módulo de Software

Este módulo elimina el residuo de la salida de HW mediante la generación y posterior resta de una plantilla de SW. Esta plantilla se obtiene mediante el promedio de 100 períodos de estimulación.

Para que funcione correctamente las estimulaciones con las cuales se adquiere la plantilla deben ser *estimulaciones sub-umbrales*, es decir de amplitud menor al umbral de captura, esto es para para que las mismas no generen una ECAP (ver Secc. 1.3). Sino, el ECAP formaría parte de la plantilla y al realizar la resta también se cancelaría la señal de interés. Luego se realiza tanto en FW como en SW un escalado de la plantilla del artefacto. El factor a utilizar debe ser por lo menos la mínima relación entre la amplitud de la estimulación sub-umbral y el umbral de captura que provoque la generación de la respuesta neural deseada. Este método funciona siempre y cuando la relación entre estimulación sub-umbral y de captura no sea demasiado grande, de manera que se conserve el comportamiento lineal del residuo.

Una vez adquirida la plantilla de SW se procede a realizar la resta con la salida de HW dando como resultado la señal neural buscada.

2.4. Requerimientos del Sistema

Se busca lograr la *Cancelación de Artefactos* (ver Secc. 1.4) para recuperar las señales de interés (ECAP) que ocurren en la zona del estímulo.

La solución debe utilizar la señal resultante de la medición en la zona de estímulo, entendida como el artefacto y la señal neural superpuestas, para procesarla

mediante un sistema que logre la supresión del artefacto en tiempo real. De esta forma, se debe poder adquirir/reproducir ininterrumpidamente las respuestas neurales.

Es importante definir las características generales con las que debe cumplir el sistema mencionado previamente. Se trabajará en reconstruir la solución propuesta en [2], la cual logra satisfacer todos los requerimientos mediante una topología que combina circuitos en HW y una aplicación de FW. Se considerará al sistema solución entonces como la combinación de una plataforma de HW y FW, con una etapa de post procesamiento mediante la implementación de SW. HW+FW conforman una primer etapa de cancelación de artefactos, mientras que el SW, además manejar tareas como adquisición y visualización de datos, busca eliminar componentes de artefactos residuales de la primera etapa de cancelación.

Siguiendo con las especificaciones en términos generales, el sistema a diseñar debe cumplir con dos requisitos principales para que cumpla las funciones deseadas:

- El rango de voltajes de la entrada debe ser lo suficientemente alto como para manejar correctamente y suprimir artefactos de estimulación grandes. Esto requiere de una eliminación parcial de artefactos a la entrada, previo a la amplificación de la señal neural para así evitar la saturación de los circuitos analógicos.

- El sistema debe presentar una relación señal a ruido (SNR) suficientemente alta para poder distinguir las señales neurales correctamente. Este ruido incluye tanto el ruido aleatorio generado por componentes electrónicos, así como el resultante del artefacto cancelado. Por lo tanto, el ruido debe ser controlado cuidadosamente ya que la señal neural es varios ordenes de magnitud más pequeña que el artefacto superpuesto.

2.4.1. Requerimientos Detallados

A continuación, en las tablas Tab. 2.1, Tab. 2.2, Tab. 2.3 y Tab. 2.4 se listan los requerimientos detallados del sistema a diseñar. Dichos requerimientos se dividen según las áreas a abordar en el desarrollo de la solución.

En específico, se diferenciaron cuatro áreas determinantes para completar las especificaciones necesarias: el sistema en general (SYS), donde se listan requerimientos relevantes para todas las áreas; el Hardware; el Firmware; y por último el Software.

Id.	Módulo	Descripción
Sistema		
SYS-001	SYS	El sistema deberá ser capaz de trabajar con señales en continua diferenciales entre $\pm 100\ mV$.
SYS-002	SYS	El sistema deberá ser capaz de trabajar con componentes en modo común en su entrada de $V_p\ < \pm 3{,}47\ V$.
SYS-003	SYS	Se apunta a reducir en 50 dB artefactos cuya amplitud máxima tenga una relación de 90 dB con respecto a la amplitud mínima de la señal de interés (ECAP). 80 dB serán aceptables. En caso de no llegar a la meta, realizar una caracterización de la relación máxima de interés y artefacto reducido.
SYS-004	HW	El sistema deberá ser capaz de detectar ECAPs con una amplitud de 2 a 150 uV, con una onda N1 que ocurrirá luego del comienzo de la fase negativa del SA según una distribución normal $\mathcal{N}(\mu,\, \sigma^2)$ con $\mu = 470\ \mu s$ y $\sigma = 62\ \mu s$.
SYS-005	HW	El sistema deberá ser capaz de reducir artefactos de estimulación con una fase negativa lineal y otra positiva exponencial, con una amplitud de pico de 70 a 240 mV y ancho de banda de 16 kHz. La frecuencia de estimulación es de $900 \pm 5\,\%\ Hz$. Tendrá una duración de la fase negativa de $300 \pm 10\ \mu s$, una constante de tiempo asociada de $250 \pm 10\ \mu s$ y una pendiente de la fase negativa de $1\ V_{pp}$ / $800\ \mu s \pm 0.1\ V_{pp}$ / $800\ \mu s$.

Tabla 2.1: Requerimientos detallados para el Sistema General

Id.	Módulo	Descripción
Hardware		
HW-001	PDN	El sistema deberá tener dos alimentaciones, una para los circuitos digitales y otra para los analógicos.
HW-002	PDN	El circuito analógico deberá ser alimentado de forma tal que sea capaz de trabajar con señales en modo diferencial de al menos 250 mV.
HW-003	MCU	El MCU elegido deberá ser capaz de ser programado por JTAG.
HW-004	MCU	El MCU elegido deberá tener al menos un periférico UART.
HW-005	Muestreo	El DAC y el ADC deberán ser elegidos tal que con el voltaje de referencia especificado más su precisión se obtenga que el LSB sea mayor al máximo valor de la señal de interés (ECAP), es decir 150 uV.
HW-006	Muestreo	El ADC y DAC deberán ser elegidos tal que puedan operar a una frecuencia de muestreo mayor o igual a 100 kHz. Notar que esto refiere a que puedan ser configurados por el MCU (DAC) y leídos (ADC) a esta velocidad.
HW-007	UI	Se deberán tener LEDs que se enciendan indicando que las fuentes de alimentación están encendidas.
HW-008	UI	Se deberá contar con un botón de reset para el MCU.
HW-009	Analog	Se deberá utilizar un amplificador de instrumentación para la primera etapa (sensado). La amplificación deberá tener además un CMRR mayor a 60 dB en el ancho de banda de interés (300 Hz a 5 kHz)
HW-010	Analog	La segunda etapa de amplificación definirá un filtro pasabanda con ganancia en banda pasante de 1 V/V $\pm$ 5 %. Su f_{-3dB} inferior deberá ser menor a 300 Hz y su f_{-3dB} superior mayor a los 5 kHz.
HW-011	Analog	La tercera etapa de amplificación debe tener una respuesta en frecuencia pasa bajos con una ganancia de 60 $\pm$ 10 % dB en banda pasante y f_{-3dB} mayor a los 5 kHz. Esta etapa se admite que sea dividida en varias etapas para llegar a la ganancia deseada.

HW-012	Analog	La salida de la etapa de amplificación no debe tener voltaje de offset mayor a 50 mV.
HW-013	Analog	El filtro de segundo orden a la salida del módulo de HW/FW deberá tener una frecuencia de corte mayor a la frecuencia máxima de interés (5 kHz).

Tabla 2.2: Requerimientos detallados para Hardware

Id.	Módulo	Descripción
Firmware		
FW-001	Comunicación Muestreo	Se deberá contar con un módulo dentro del firmware para el manejo del DAC y el ADC a una velocidad no menor a 100 kHz.
FW-002	Interrupciones	El FW recibirá una señal que indica que se ejecutó un estímulo cuyo artefacto se desea eliminar.
FW-003	HW	El FW deberá ser capaz de guardar una forma de onda arbitraria en memoria, ya sea RAM o FLASH.

Tabla 2.3: Requerimientos detallados para Firmware

Id.	Módulo	Descripción
Software		
SW-001	Muestreo	El SW diseñado deberá ser capaz de manejar un sistema de adquisición de datos de salida al PC (tarjeta adquisidora o AD2).
SW-002	UI	El SW deberá ser capaz de manejar la comunicación hacia el sistema de FW.
SW-003	UI	Debe contar con un GUI.
SW-004	Procesamiento	El SW deberá recibir las muestras del sistema de adquisición y procesarlas tal de obtener una salida con una correlación mayor a 0.7 con la señal de la respuesta al estímulo de referencia.

Tabla 2.4: Requerimientos detallados para Software

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Simulación.

3.1. Motivación

Debido a la complejidad del sistema introducido en el Cap.2, el primer paso realizado fue implementar un entorno de simulación que permitiera entender el funcionamiento del mismo. Además, permitió contar con una herramienta clave para el trabajo siguiente, pues la capacidad de obtener una representación abstracta del sistema donde se pueden rápidamente realizar cambios y pruebas es de suma utilidad para el desarrollo del sistema completo.

3.2. Entorno de Simulación

El primer paso fue la selección de una herramienta de simulación adecuada, la misma debe ser capaz de simular tanto sistemas reales en tiempo continuo así como sistemas de tiempo discreto. También es de utilidad que la herramienta sea capaz de imitar el funcionamiento de un microcontrolador. En un principio se consideraron diversas herramientas, como puede ser Proteus [9]. La ventaja de este software es que permite la interacción de sistemas reales con microcontroladores, a los cuales se les puede escribir directamente el código en C que se le desea ejecutar. Sin embargo se optó por Simulink [10] de MathWorks, que se encuentra en el paquete de herramientas de Matlab.

Se eligió Simulink debido a que cuenta con una intuitiva interfaz gráfica que permitió desarrollar rápidamente el modelo en cuestión y cuenta con extensa documentación en foros y en la página oficial donde se puede encontrar casos de uso similares y soluciones a problemas que surgen en desarrollo. En cuanto a la implementación del microcontrolador, cuenta con bloques específicos donde se le puede agregar una sección de código de Matlab, con lo que cumplía con los requerimientos buscados.

3.3. Generación de Señales

Para el diseño de las señales en Simulink, se trabajó bajo las hipótesis estipuladas por los modelos vistos en la Secc,2.2. En las secciones contiguas Secc.3.3.1 y Secc. 3.3.2 se detalla el proceso de creación de las mismas.

Las señales creadas en la simulación serán utilizadas luego para el desarrollo y testeo de las siguientes etapas del proyecto.

3.3.1. Evoked Compound Action Potential

Para cumplir con las especificaciones de la señal de ECAP se optó modular una función sinusoidal a partir de exponenciales de distintas constantes. Esta metodología logra transformar la función base en una onda que respeta la morfología indicada en Fig.2.3.b. Se utilizó por lo tanto un seno de 1,2 kHz como función base, donde luego a cada cresta y valle se lo multiplicó por una función exponencial. Finalmente, se pasó la señal generada a través de un filtro analógico pasabanda de primer orden, para respetar el ancho de banda de las especificaciones.

En la Secc. 2.2.2 se expresó que la aparición de la señal de ECAP sigue el comportamiento de una distribución normal. Para el caso de la simulación, no se incorporó esta característica del modelo. En su lugar, se ubicó temporalmente la señal en donde se da la media del modelo normal.

Como resultado, se obtuvo la señal de la Fig.3.1. Dicha señal es la que se logra luego del filtrado. Para ver en detalle la implementación en Simulink dirigirse al Anexo A.

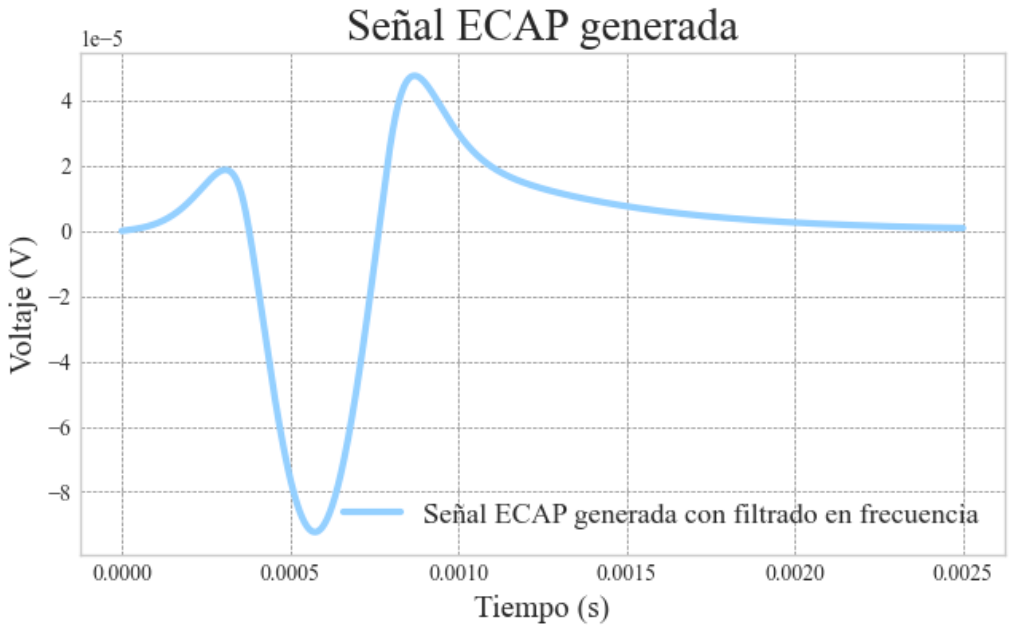


Figura 3.1: Representación de un período de la señal ECAP con filtrado.

3.3.2. Stimulus Artifact (SA)

Para la construcción de esta señal se modelaron por separado las dos fases que la componen, su fase negativa y su fase positiva. Recordando la especificación del artefacto de estimulación de la Secc.2.2, la fase negativa se compone por una componente lineal seguida de una componente exponencial, mientras que la fase positiva es puramente exponencial.

A continuación, se presenta el resultado final obtenido para la construcción de la señal SA después de la salida del filtro pasabajos (Fig. 3.2). Nuevamente el detalle de implementación en Simulink se encuentra en el Anexo A.

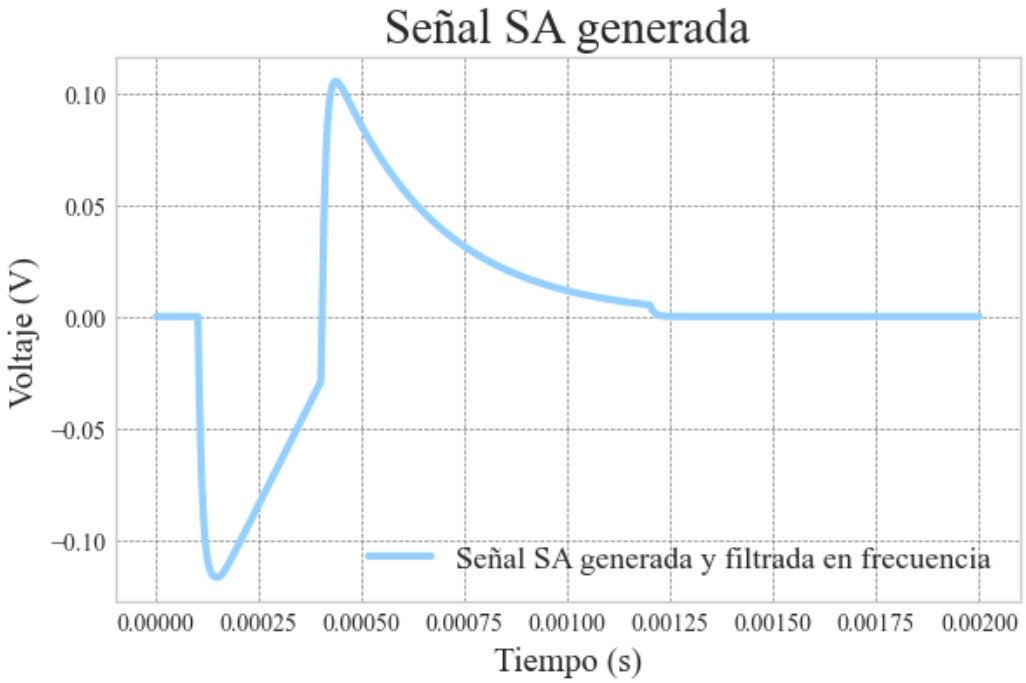


Figura 3.2: Representación de un período de la señal SA luego del filtrado.

3.4. Módulo de Hardware y Firmware

Como fue descrito anteriormente en la Secc.2.3.1 se contará con una serie de amplificadores en distintas configuraciones en cascada. También se contará con un circuito de muestreo digital dirigido por un microcontrolador tal que permita realizar una plantilla del artefacto a restar. Por último esta plantilla será reproducida por un conversor digital a analógico, de manera que no sature el amplificador de salida de esta etapa.

3.4.1. Etapa de Amplificación 1

El propósito de esta primera etapa es la detección de la señal de interés (ECAP) superpuesta con el artefacto producido por la estimulación. Debido a la amplitud mínima apuntada a detectar de 2 uV, es importante que la primera etapa cuente con el mínimo ruido intrínseco posible y alto CMRR. Además, es de interés poder realizar una preamplificación de la señal detectada de ser necesario.

Un amplificador de instrumentación de bajo ruido es ideal para esta aplicación y se modelará en *Simulink* como un bloque de suma ideal donde se puede restar a la SA el SA, así como un ruido equivalente a la entrada. El ruido se modeló como sumar un ruido blanco en la entrada filtrado en la banda de interés de nuestro circuito, es decir 16 kHz (ancho de banda de la señal SA). Para esto se utilizó el bloque *White Noise* de Matlab donde se tuvo que especificar la potencia del la PSD del mismo así como su tiempo de muestreo. Para definir potencia de la PSD se tomó un enfoque empírico con tal de conseguir una máxima amplitud del ruido a la entrada igual a la mínima amplitud posible del ECAP, correspondiente a 2 uV. Se diseñaron las entradas de las señales ECAP y del ruido de tal manera que puedan ser fácilmente desconectadas de la entrada con un switch.

Por último, se realiza una amplificación que puede ser ajustada previa a la simulación para modelar la pre-amplificación variable. Esta configuración se puede ver en la Fig.3.3, donde se muestra un diagrama de bloques representativo de la implementación en Simulink.

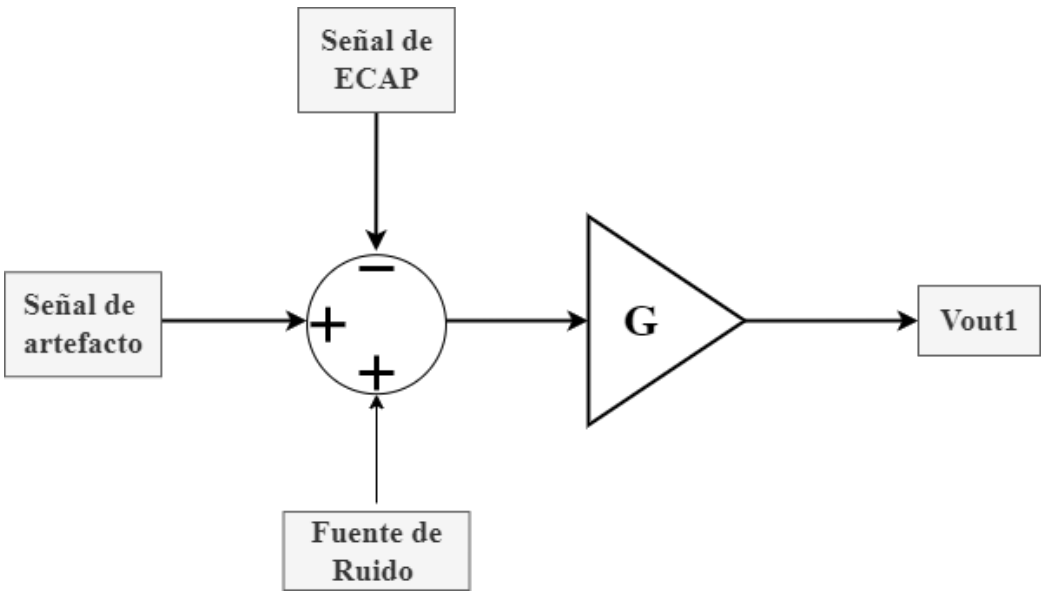


Figura 3.3: Diagrama de bloques de la implementación de la etapa de amplificación 1 en Simulink.

3.4.2. Etapa de Amplificación 2

Luego de la preamplificación de la señal, se realiza la resta de la plantilla generada por el microcontrolador en la segunda etapa de amplificación que cuenta con ganancia unitaria. El circuito de la implementación en hardware se puede ver en la Fig 3.4 donde se tiene que C_c cumple la función de condensador de desacople y el amplificador realiza la resta de las señales V_{DAC} y V_1. Realizando los cálculos (para ver el detalle de los mismo ver D.2)para la transferencia de la dos señales a la salida se obtiene que:

$$V_{o2} = V_{BIAS} + \frac{-R_{FB}C_c s}{1 + R_{FB}C_c s}(V_1 - V_{DAC}) \tag{3.1}$$

Que corresponde a un filtro pasa-altos para V_{DAC} y V_1 con polo en:

$$p_1 = \frac{1}{2\pi R_{fb}C_c} \tag{3.2}$$

Ya que V_{DAC} y V_1 cuentan con la misma transferencia a la salida del restador pero con signo contrario, es muy fácil modelar esta etapa en Simulink con un bloque restador y un bloque de transferencia como muestra la figura 3.5. Notar que no fue de interés modelar el nivel DC que se establece a la salida con V_{BIAS}. En este capítulo se denomina restador al bloque que realiza la resta matemática entre las dos señales entrantes a la etapa dos, mientras que etapa de amplificación se referirá al bloque que aplica la transferencia del amplificador diferencial a la salida del restador.

Es importante la aclaración que en el resto del proyecto, como todo está implementado con un solo amplificador diferencial, se utilizarán los términos etapa de amplificación 2, amplificador diferencial o restador como sinónimos ya que todas estas funciones son realizadas por una única etapa. Este no será el caso de la simulación donde contamos con un bloque particular para realizar la resta y otro para aplicar la transferencia.

En la salida de la etapa dos se realiza el muestro por el ADC comandado por el microcontrolador para la obtención de la plantilla del artefacto.

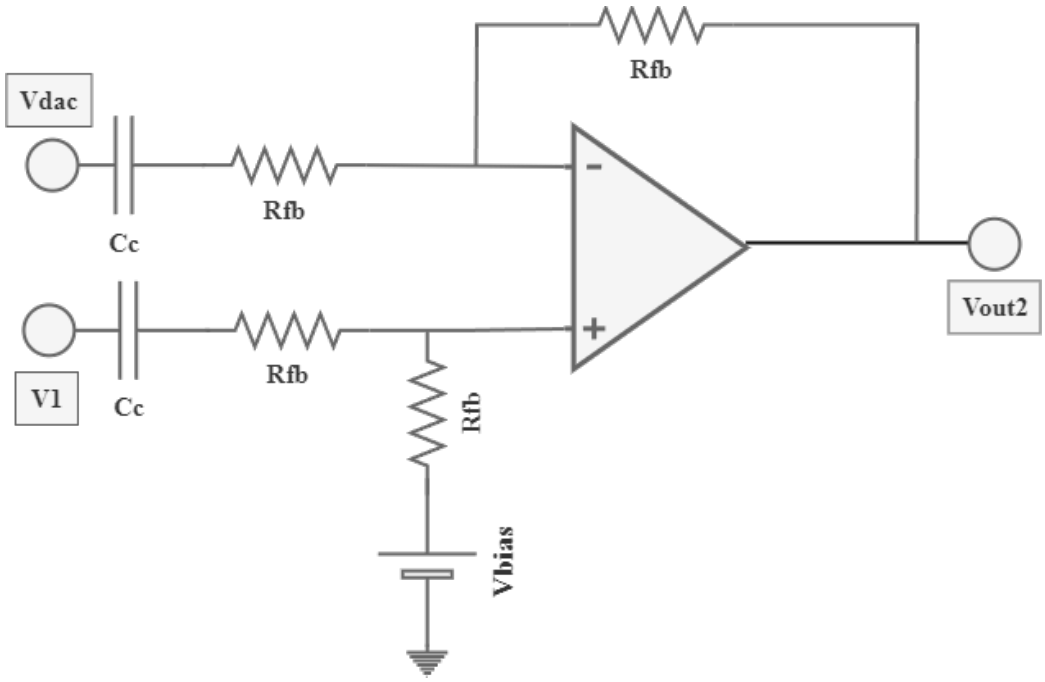


Figura 3.4: Circuito de la etapa de amplificación dos.

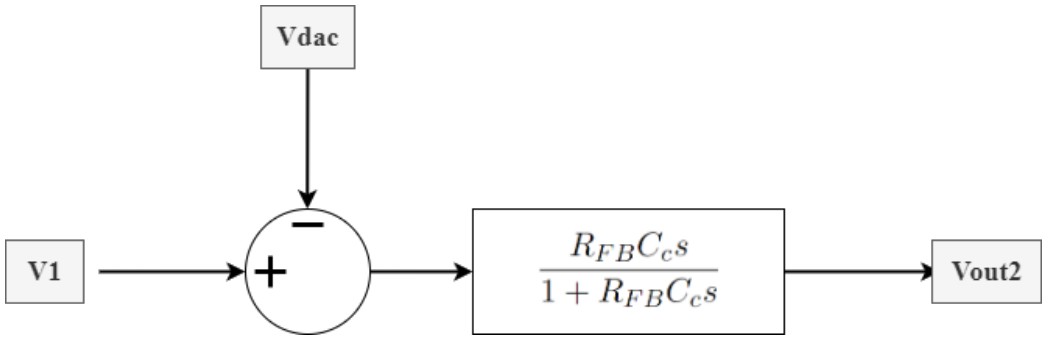


Figura 3.5: Diagrama de bloques de la implementación de la etapa dos en Simulink.

3.4.3. Etapa de Amplificación 3

Por último, la tercera etapa será la encargada de la amplificación de la señal de interés. La misma está formada por un amplificador operacional en su configuración de amplificador inversor de ganancia $\frac{-R_1}{R_2}$. Se apunta a llegar a producir un amplificación de $60dB$. Notar que en este paso es donde puede saturar la salida debido a la amplificación de un residuo de artefacto muy grande. Es por eso que se realiza la resta de la plantilla en la etapa anterior, apuntando a reducir la amplitud del artefacto tal que este no genere la saturación de la tercera etapa.

Para modelar esta etapa se utilizó un bloque de ganancia constante y luego uno que modele la saturación para $\pm 5\ V$. Se puede ver el diagrama implementado en Simulink en la figura 3.6

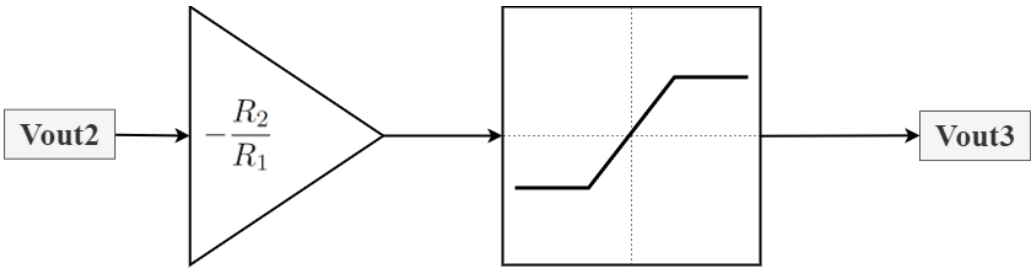


Figura 3.6: Diagrama de bloques de la implementación de la etapa de amplificación tres en Simulink.

3.4.4. Lazo de Control Cerrado en Simulink

Para continuar con la implementación de los módulos se debe profundizar en cómo maneja un lazo cerrado *Simulink*. Cuando en *Simulink* se implementa un sistema en donde se utiliza el valor de una variable o bloque para calcular el valor de salida de la misma, se obtiene lo que se denomina un "algebraic loop". En estos casos, se está creando una dependencia circular entre los valores de salida y entrada en el mismo tiempo de muestreo. Este es por ejemplo el caso de un modelo de variables de estado, donde la derivada de los estados depende de una función de su propio valor.

Para estos casos, se utiliza un algoritmo no lineal en cada paso de tiempo para resolver el *algebraic loop* donde se realizan varias iteraciones hasta llegar a una solución si esta existe.

En el sistema a implementar, se debe establecer un lazo de realimentación negativa, como se puede ver en la Fig3.5. El problema en este caso en particular es que el lazo de control actúa estimulación a estimulación pero durante cada una de estas su comportamiento es constante. Es decir, no se puede establecer un *algebraic loop* al conectar la salida de la etapa de amplificación 2 a la entrada de la misma, generando así un esquema de realimentación clásico.

Para solucionar esto, se optó por simular el sistema de la siguiente forma: en cada ejecución, se realiza la cancelación del artefacto por medio del DAC de manera independiente al muestreo y actualización de la plantilla del que se encarga el ADC. De esta manera, la muestra en cierto tiempo del DAC es independiente de la que esta muestreando el ADC y el algoritmo de *Simulink* no detecta un *algebraic loop*. Los datos muestreados por el ADC, que resultan de la resta del artefacto con la plantilla producido por el DAC son utilizados para actualizar la plantilla de ser necesario y lo guarda en memoria para la siguiente ejecución de la simulación, la cual representa la siguiente estimulación.

De esta manera, se logra cerrar el lazo de ejecución a ejecución. En la primera ejecución, no se cuenta con una plantilla guardada en memoria y por ende esta consistirá en generar la plantilla inicial. Esto trae consigo que si se cambia algún parámetro del sistema, como puede ser la velocidad de muestreo del ADC, se tenga que desechar la plantilla actual y empezar todo el proceso nuevamente.

3.4.5. Microcontrolador y Muestreo de Datos

Para modelar el sistema embebido que forma parte del módulo HW + FW se tuvo que implementar tres bloques: un conversor analógico digital (ADC), que permita realizar el muestro de la salida de la etapa de amplificación dos; la lógica que realice el procesamiento de la plantilla realizado dentro del MCU; y por último un conversor digital analógico (DAC), que genere la plantilla y lo reproduzca para la resta con el artefacto.

A esto se le suma la necesidad de obtener una realimentación que satisfaga las observaciones vistas en la Secc.3.4.4. Para esto se realizó la configuración que muestra la Fig.3.7. Dónde lo más importante a destacar es que se utilizó el bloque *Matlab function* para modelar el procesamiento realizado por el microcontrolador. El algoritmo llevado a cabo será descrito con más detalle en la Secc.3.4.5.

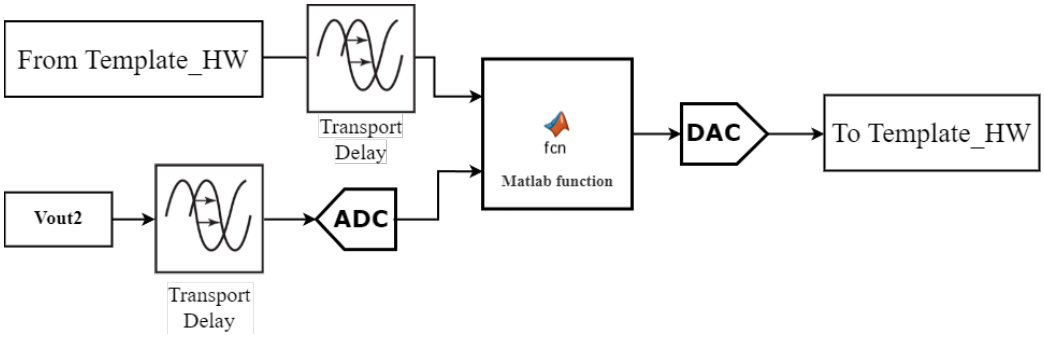


Figura 3.7: Diagrama de bloques de la implementación del ADC + MCU + DAC en Simulink. Como entrada se tiene la salida de la segunda etapa de amplificación compuesta por el restador (out_amp2) y el la plantilla de la iteración anterior (out.TemplateHW) y su salida es la señal de nombre TemplateHW que corresponderá a la plantilla a restar.

ADC

El ADC utilizado fue obtenido de la librería de bloques de Simulink que corresponde al modelo de un flash ADC [11] donde se puede configurar la precisión (cantidad de bits), fondo de escala y velocidad de muestreo, entre otros. Además cuenta con la capacidad de modelar no idealidades como el error de offset y en ganancia. Para adaptarlo a una solución real se le agregó un retardo en el tiempo a la entrada del ADC debido a que tanto sea utilizando un ADC externo o interno a un microcontrolador, se tiene un tiempo de retardo entre indicarle al periférico que realice una medición y que la termine. Por ejemplo, para el ADC interno al STM32L55, hay un retardo de 12.5 ciclos de reloj del periférico entre el inicio de una medición y que esta sea completada. Por ende, fue necesario modelar este fenómeno en la simulación para obtener cotas superiores para el retardo tal que no afecte significativamente el desempeño del sistema.

La entrada del bloque ADC se conecta directamente a la salida de la etapa dos y su salida a *Matlab function* para que realice el procesamiento de la señal. Luego, la salida del mismo se guarda dentro del entorno de trabajo como una variable, para ser reproducida en la siguiente iteración.

DAC

Simulink cuenta con pocas opciones para implementar conversores digitales a analógicos y en la versión 2020 con la que se trabajó, el único bloque donde se implementa un DAC generó algunos problemas en el entorno a los intervalos iniciales de tiempo en los cuales sus valores no están definidos. Por ende, se optó por reproducir directamente la señal guardada por el ADC, utilizando el bloque *From Workspace* que permite realizar interpolación de los datos si fuera necesario y un multiplicador para ajustar los valores utilizados por el ADC a valores de voltaje.

Algoritmo implementado

Es importante mencionar que el algoritmo se basa en un tipo de variable particular que se puede asignar en Matlab llamado *persistant*. Para entender bien como funciona este nuevo tipo, es importante mencionar que una función de Matlab incluida en Simulink, es ejecutada en cada paso de tiempo tomado por el *Solver* de Simulink. Las variables dentro de la función nacen y mueren en el contexto de la misma, sin acarrear su valor a iteraciones posteriores. El tipo *persistant* permite que en instantes de muestreo diferentes, o sea ejecuciones distintas de la función, una variable conserve su valor.

Debido a que el ADC cuenta con una frecuencia de muestreo fija mayor a los pasos temporales que Simulink toma para resolver la simulación, la función dentro del bloque *Matlab Function* observa, en casi todo los tiempos, una salida no válida del ADC. Se utiliza el tipo de dato *persistant* para guardar el último dato válido del ADC entre ejecución y ejecución de la función y obtener a la salida de la misma, escalones con valores constantes en todos los pasos temporales que realiza Simulink.

El funcionamiento general del algoritmo dentro del bloque *Matlab Function* se puede ver en el siguiente pseudocódigo.

Algorithm 1: Pseudocódigo del algoritmo implementado.

```
Data: template_in, adc_in, ready
Result: out_alg
if ready == 1 then
    last_adc_count ← adc_in
    if abs(adc_in) ≥ umbral then
        out_alg ← template_in + adc_in;      /* Si es mayor al umbral
         sumo el error.  */
    else
        out_alg ← template_in;           /* Si es menor al umbral lo
         mantengo.  */
    end
else
    out_alg ← adc_in;  /* Cuando el adc no está listo copio última
     muestra.  */
end
```

3.4.6. Resultados de la Simulación de las Etapas de Hardware y Firmware

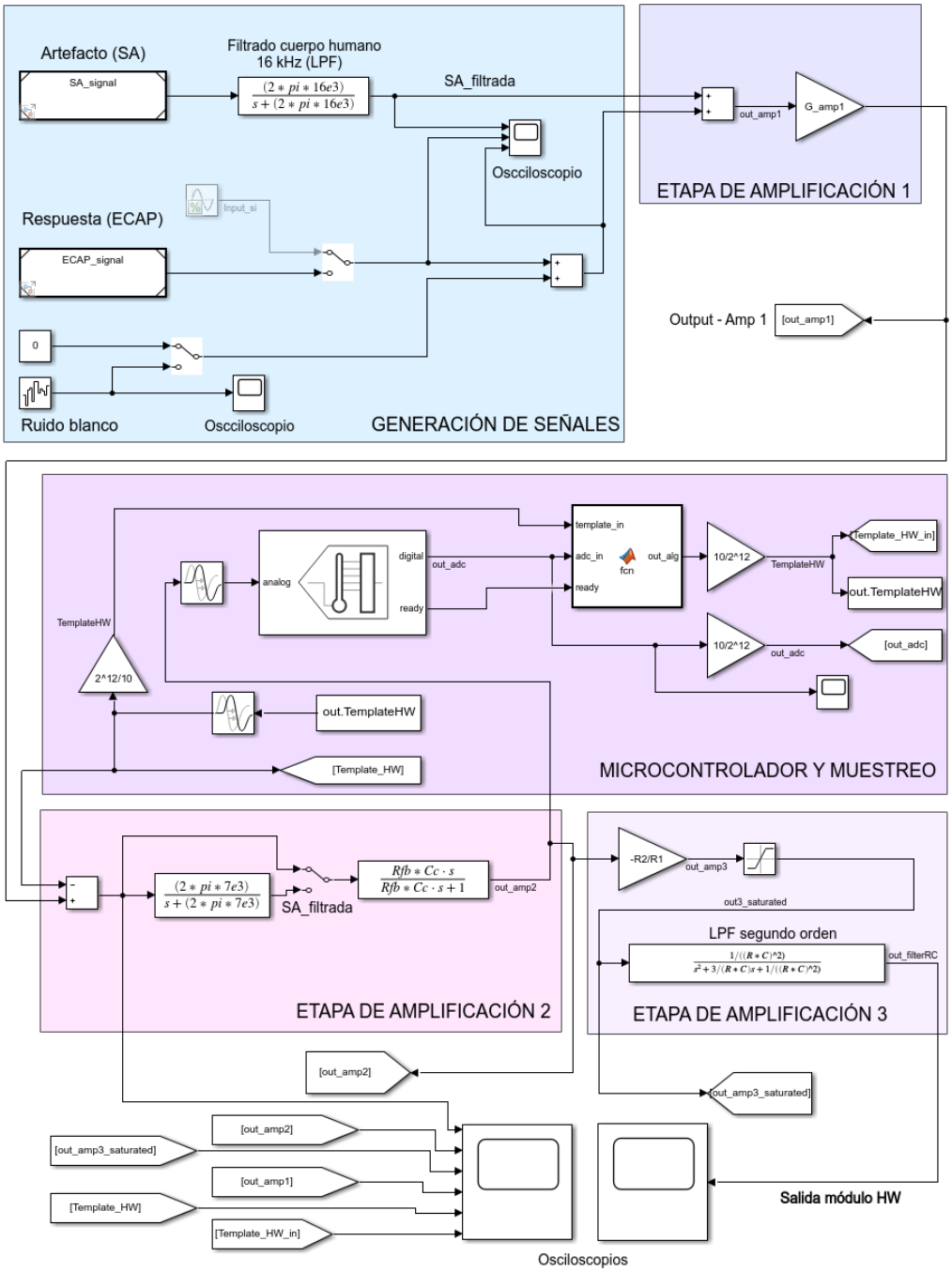


Figura 3.8: Simulación en Simulink de las etapas de HW + FW.

En la Fig. 3.8 se presenta una captura de la simulación implementada en Simulink de los bloques de HW y FW en su totalidad. Las etapas descritas a lo largo de la Secc. 3.4 se encuentran diferenciadas en la figura mediante bloques de colores.

Puesto que en la simulación no hay introducción de ruido por los componentes, se decidió añadir ruido en paralelo a la entrada de la señal de ECAP con el fin de asemejar el sistema aún más a la realidad. Esto se puede ver en la Fig. 3.8, dentro del bloque de *Generación de señales* se colocó una instancia *Band-Limited White Noise*, como se mencionó anteriormente. Esta instancia genera números normalmente distribuidos de forma aleatoria. El ruido blanco además tiene un tiempo de correlación 0, una densidad espectral de potencia (PSD) plana y una energía total infinita. Los parámetros elegibles para el ruido corresponden a la amplitud de la potencia, el tiempo de muestreo y el número semilla a partir del cual parte la aleatoriedad. Se recuerda la elección de que la máxima amplitud del ruido a la entrada sea igual a la mínima señal de ECAP posible que es de 2 uV.

A partir de esta implementación se lograron probar distintas características del sistema y verificar condiciones que deben de ser cumplidas para que efectivamente la solución funcione. A continuación se detallan las pruebas realizadas y sus respectivos resultados.

3.4.7. Verificación del Comportamiento Esperado

A continuación se mostrará un análisis del comportamiento obtenido a partir de la implementación total. Se realizaron varias iteraciones para comprobar que efectivamente el comportamiento era el esperado.

Como configuración de parámetros inicial se utilizaron los valores de la Tab. 3.1. Para comenzar, se fijó la frecuencia de muestreo del ADC en 200 kHz, considerando que la señal de SA tiene componentes de hasta 16 kHz.

Parámetro	Valor
Valores de Componentes Discretos	
Rfb	100 $k\Omega$
R	47 kHz
C	1 nF
R2	100 kHz
R1	100 Ω
Parámetros de Muestreo y Saturación	
Frecuencia ADC	200 kHz
Voltaje de Saturación	3.3 V
Ganancia Etapa 1	1 V/V
Delay ADC	1.5 μs
Delay DAC	1.5 μs

Tabla 3.1: Valores de los componentes y parámetros utilizados para la simulación del sistema en esta instancia.

En la Fig. 3.9 se pueden observar varias etapas del circuito en Simulink luego de una primera iteración corrida de la simulación, utilizando únicamente la señal SA como entrada. La razón por la que solo se utiliza la SA en los primeros pasos es porque se va a adquirir la plantilla de HW con el objetivo de luego utilizar la salida de la etapa para la adquisición de la plantilla de SW. Si se incorporase la señal de ECAP en la salida, el post procesamiento eliminaría también esta señal. Este mismo proceso será realizado por nuestro sistema final CANE donde se agregará el paso adicional del escalado de las plantilla. En cuanto a la amplitud a utilizar para la señal SA, se escogió su amplitud máxima (Ver especificaciones en Secc. 2.2). Se verá a continuación el funcionamiento de la adquisición de la plantilla de HW que se utilizará para la cancelación del artefacto y evitar la saturación de la tercera etapa de amplificación. Más adelante en la Secc. 3.5.2 se verá como se obtiene la plantilla de SW para la cancelación de residuo y así obtener a la salida del sistema la señal neural subyacente.

En la primera iteración, se adquiere por primera vez información para ser guardada como plantilla de HW. Esto se verifica en la Fig. 3.9.d, donde se muestra como la plantilla de HW aún no tiene datos almacenados. Tiene sentido entonces ver la salida del restador (Fig. 3.9.a) igual a la señal de salida de la etapa de amplificación 1 (Fig. 3.9.b), ya que en la realimentación no hay datos aún. Es coherente por lo tanto observar la misma forma de onda a la salida de la etapa de amplificación 2 (Fig. 3.9.c) tras aplicarle el bloque que modela la transferencia de la etapa a la salida del restador. Ahora, esto provoca la saturación total a la salida de la etapa de amplificación 3 (Fig. 3.9.e) para ambas fases de la señal SA, puesto que no se redujo de ninguna forma la amplitud de la señal original y los circuitos analógicos se ven comprometidos. Finalmente, en la Fig. 3.9.f se comprueba la entrada de información actual que recibirá la plantilla de HW al correr el siguiente paso de la iteración. Se distingue efectivamente que la información a ser almacenada reconstruye la señal SA.

Se verá ahora lo que ocurre en el sistema al correr por segunda vez la simulación (ver Fig. 3.10). En esta instancia, la plantilla de HW contiene la reconstrucción de la señal de entrada SA. Esto quiere decir que ahora a la salida del restador se debe observar la resta de la salida de la etapa de amplificación uno y la plantilla de HW. Viendo por lo tanto la Fig. 3.10.a se verifica el cumplimiento de dicha afirmación. Es importante notar que los dos picos que aparecen en la imagen anterior (Fig. 3.10) se dan debido a que el muestreo se ve limitado en las zonas de transiciones más rápidas de la señal. Estas zonas corresponden a los flancos en las transiciones rápidas entre las fases positiva y negativa del modelo de la señal SA. Otra diferencia es que la salida de la etapa de amplificación 3 no se encuentra totalmente saturada en este caso (ver Fig. 3.10.e). Aún así la saturación está presente también en las zonas del cambio de fase, lo cual es coherente con lo visto para el restador y la salida de la etapa de amplificación 2.

Los resultados obtenidos son válidos para la configuración actual de la Tab.3.1. Dado que en este caso se pudo verificar que el sistema cumple con el comportamiento esperado, el siguiente paso a tomar es estudiar cuál es la mejor configuración de parámetros para la aplicación.

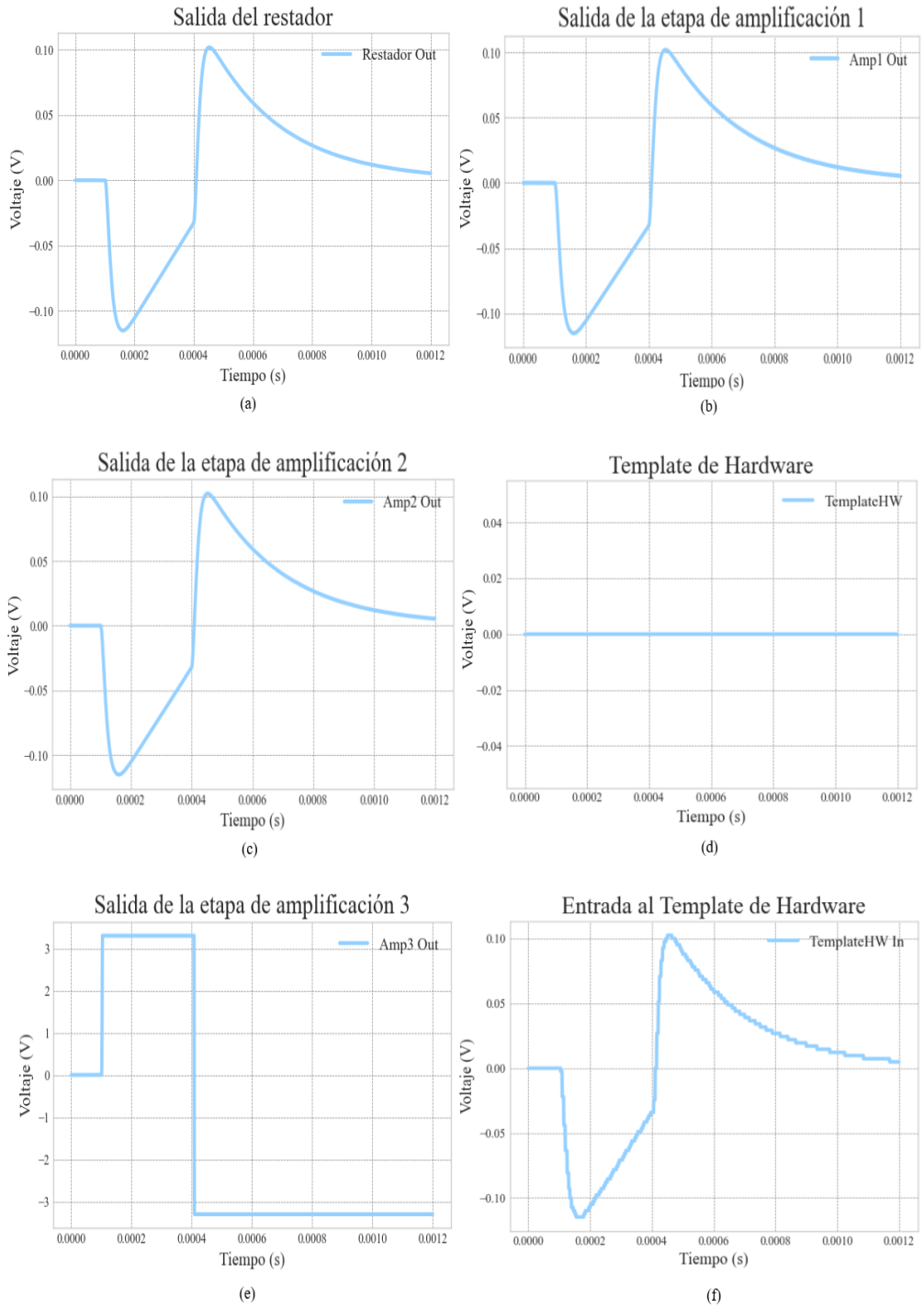


Figura 3.9: Gráficos obtenidos a la salida de distintas etapas luego de la primera iteración de la simulación. La subfigura a) es el resultado de la resta entre la plantilla de HW y la salida de la etapa de amplificación 1. La subfigura b) consiste la señal de la figura a) pasada por el filtro de la etapa de amplificación 2.

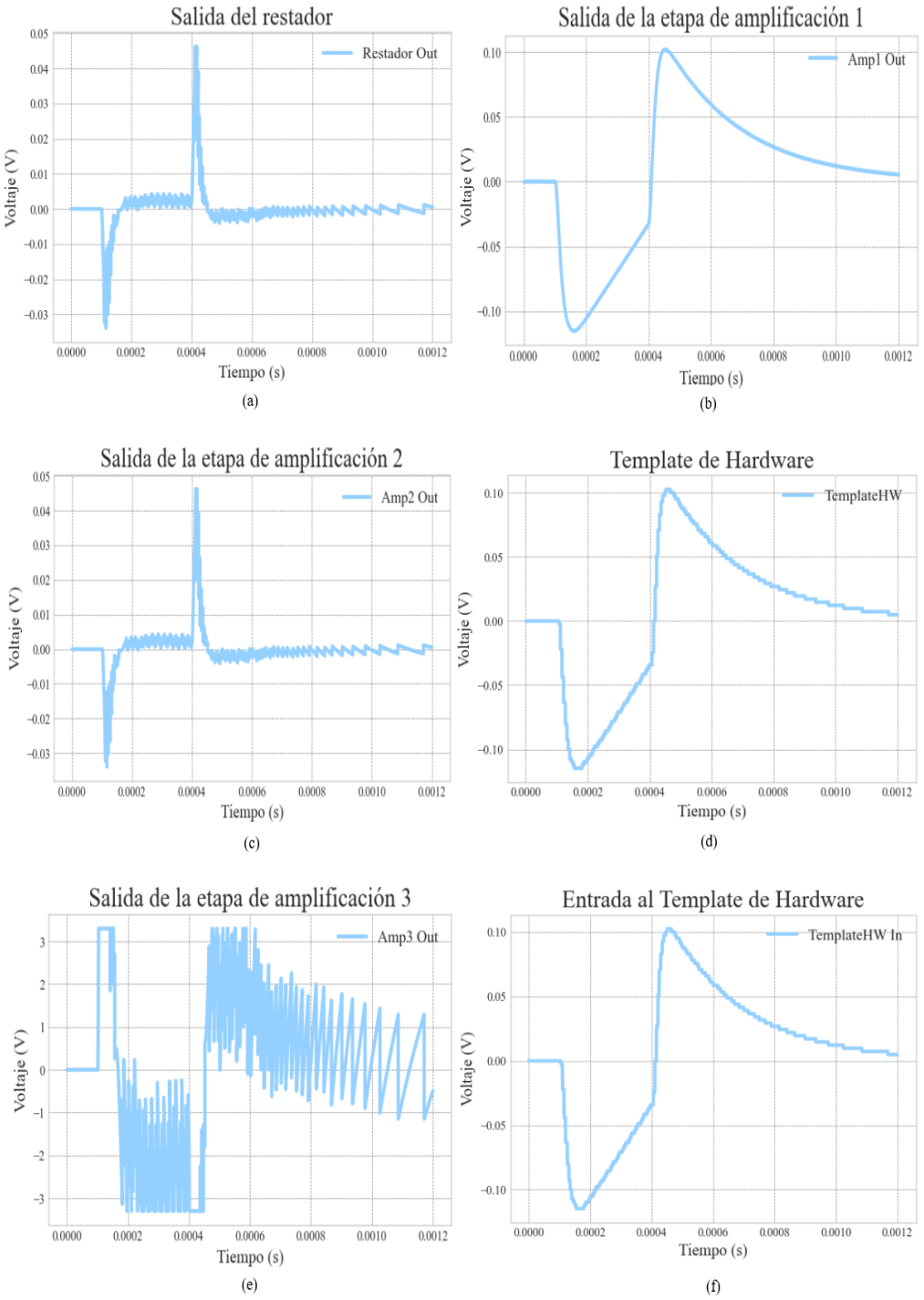


Figura 3.10: Gráficos obtenidos a la salida de distintas etapas luego de la segunda iteración de la simulación.

Máximo retardo posible en ADC y DAC

Para esta prueba se mantuvo la configuración de la Tab. 3.1, y se procedió a variar los retardos asociados a los conversores. Nuevamente se trabajó con la amplitud máxima de la señal SA.

Al modificar los retardos temporales del ADC y el DAC, se observó que tienen fuerte relación con la saturación a la salida de la etapa de amplificación tres. En particular, se determinó que a mayor retardo, el amplificador tres permanece saturado por más cantidad de tiempo.

Un caso extremo de ejemplo para esta situación es cuando el retardo conjunto entre ambos conversores es de 10 μs. Las repercusiones de esta configuración se observan en la Fig. 3.11. El cambio más notorio es la saturación durante más tiempo de la etapa de amplificación tres (Fig. 3.11.e). Además de esto, los picos a la salida del restador resultan de mayor amplitud (Fig. 3.11.a), lo cual explica la mayor probabilidad de saturación vista antes.

Es importante recordar que en la configuración actual de la simulación (ver Tab. 3.1), la frecuencia de muestreo del ADC se encuentra fija en 200 kHz. Por lo tanto, es coherente que utilizar un retardo de 10 μs tenga estos efectos, ya que es un valor mayor al tiempo de muestreo, en particular el doble más grande.

Realizando pruebas para distintos valores, se concluyó que el máximo retardo aceptable para evitar saturaciones que incapaciten el sistema es de $5\mu s$. En dicho límite, el tiempo de retardo introducido es exactamente igual al tiempo de muestreo. En la Fig. 3.12 se enseña la salida de la etapa de amplificación tres para este valor de retardo. Observar que es similar a lo obtenido para el caso de la Fig. 3.10.e.

Notar que los datos relevados en esta prueba son relativos a la frecuencia de muestreo actual de 200 kHz. Aún así muestra como afectan las limitaciones del sistema. Para la aplicación final de FW el retardo máximo posible va a depender entonces de la frecuencia a la que se fije el ADC a utilizar.

Se seguirán estudiando más parámetros más adelante, teniendo en consideración la siguiente etapa de simulación de SW. En particular, es de especial interés definir cuales es la mínima frecuencia de interés a usar para poder adecuadamente realizar la cancelación del artefacto de estimulación.

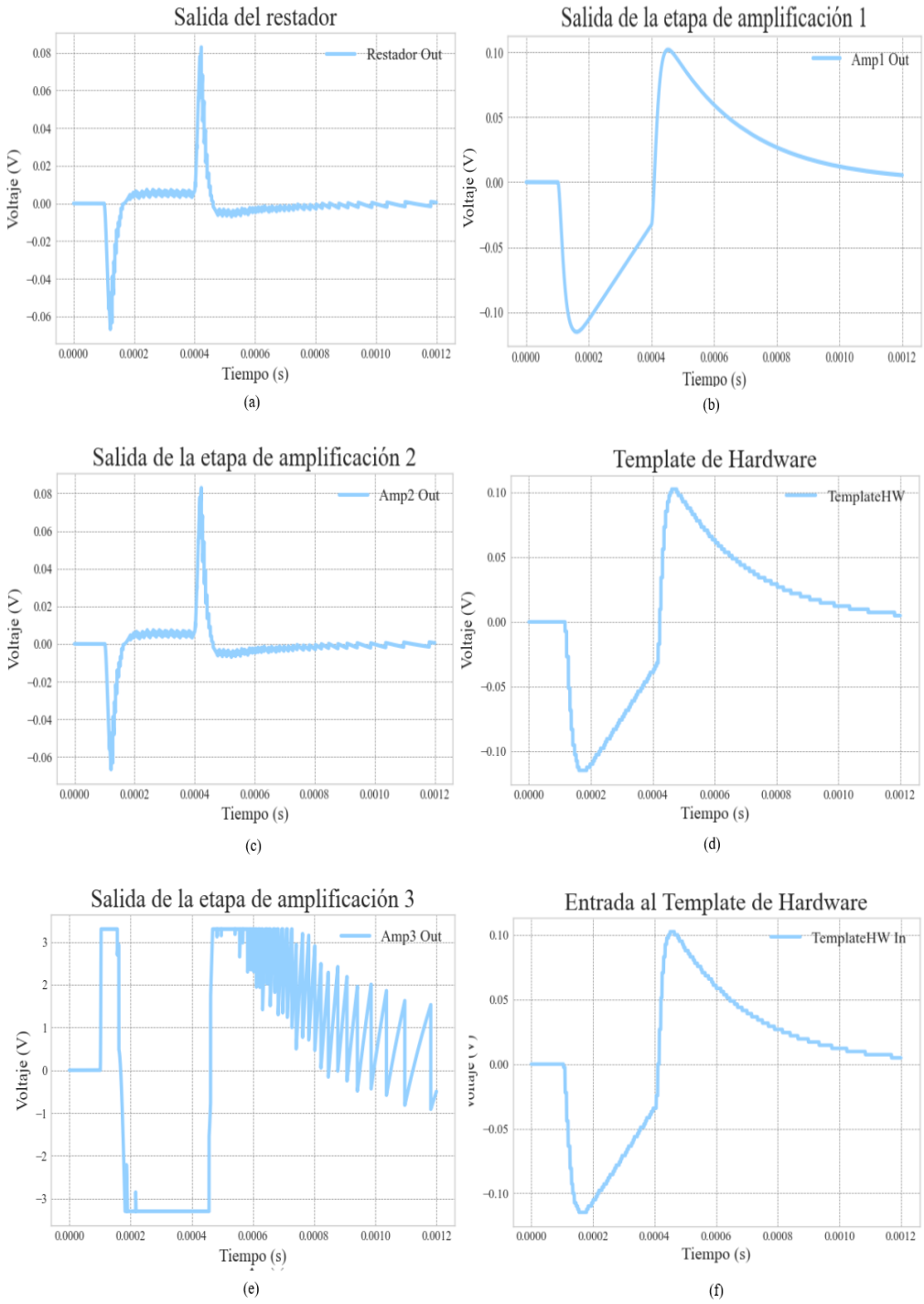


Figura 3.11: Gráficos obtenidos a la salida de distintas etapas cuando el retardo introducido por los conversores es de $10\mu s$.

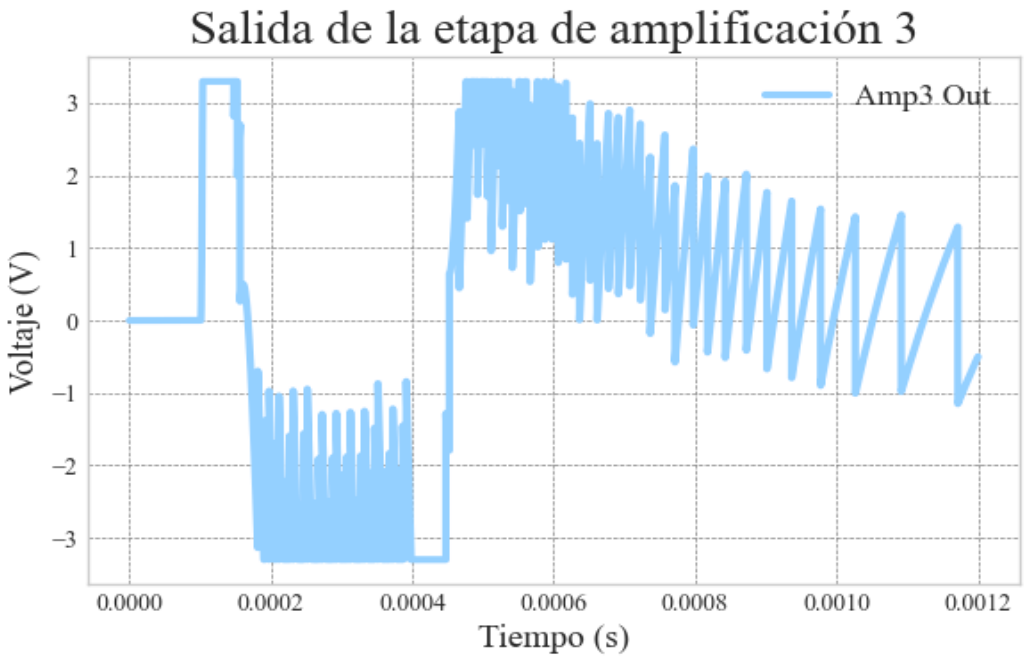


Figura 3.12: Gráficos obtenidos a la salida de distintas etapas cuando el retardo introducido por los conversores es de 10 μs.

Mínima frecuencia de muestreo del ADC para evitar la saturación

Para evaluar esta propiedad, se fijó el retardo total de los conversores en 3 μs para asegurar que no hay saturación debido al mismo. Además, la amplitud de la señal de SA se fijó en su valor máximo posible y el nivel de saturación de voltaje se fijó en $\pm 5\ V$. Luego, se procedió a analizar el comportamiento del sistema para distintos valores de frecuencia de muestreo. En la Fig. 3.13 se muestran los resultados a los cuales se llegó. Se considerará como criterio de no saturación que solo se de la misma durante los primero 75 µs luego de las transiciones rápidas entre fases de la SA. Se puede tener como ejemplo de éxito la Fig. 3.10.

Observando los resultados obtenidos en términos generales, en la Fig. 3.13 se observa como el aumento de la frecuencia de muestreo del ADC favorece la disminución de la saturación a la salida de la etapa de amplificación tres. En particular, la frecuencia donde comienza a desaparecer sutilmente la saturación en las zonas exponenciales es para la frecuencia de 130 kHz (Fig. 3.13.b). De 130 kHz en adelante se deja de observar saturación en las zonas de interés. Aún así, la fiabilidad del muestreo comienza a partir de la frecuencia de 200 kHz (Fig. 3.13.d). De este valor en frecuencia en adelante, se comienza a distinguir de mejor forma que la forma de onda corresponde a aquella de la salida de la etapa de amplificación dos (ver Fig. 3.10.c como ejemplo) amplificada. Resulta relevante destacar que la saturación que se da por los picos de gran velocidad de la SA siempre prevalece. Esto ocurre debido a que se debería contar con una velocidad de muestreo mucho mayor para que el error entre muestra y muestra reproducida por el DAC y la SA no genere la saturación de la etapa tres.

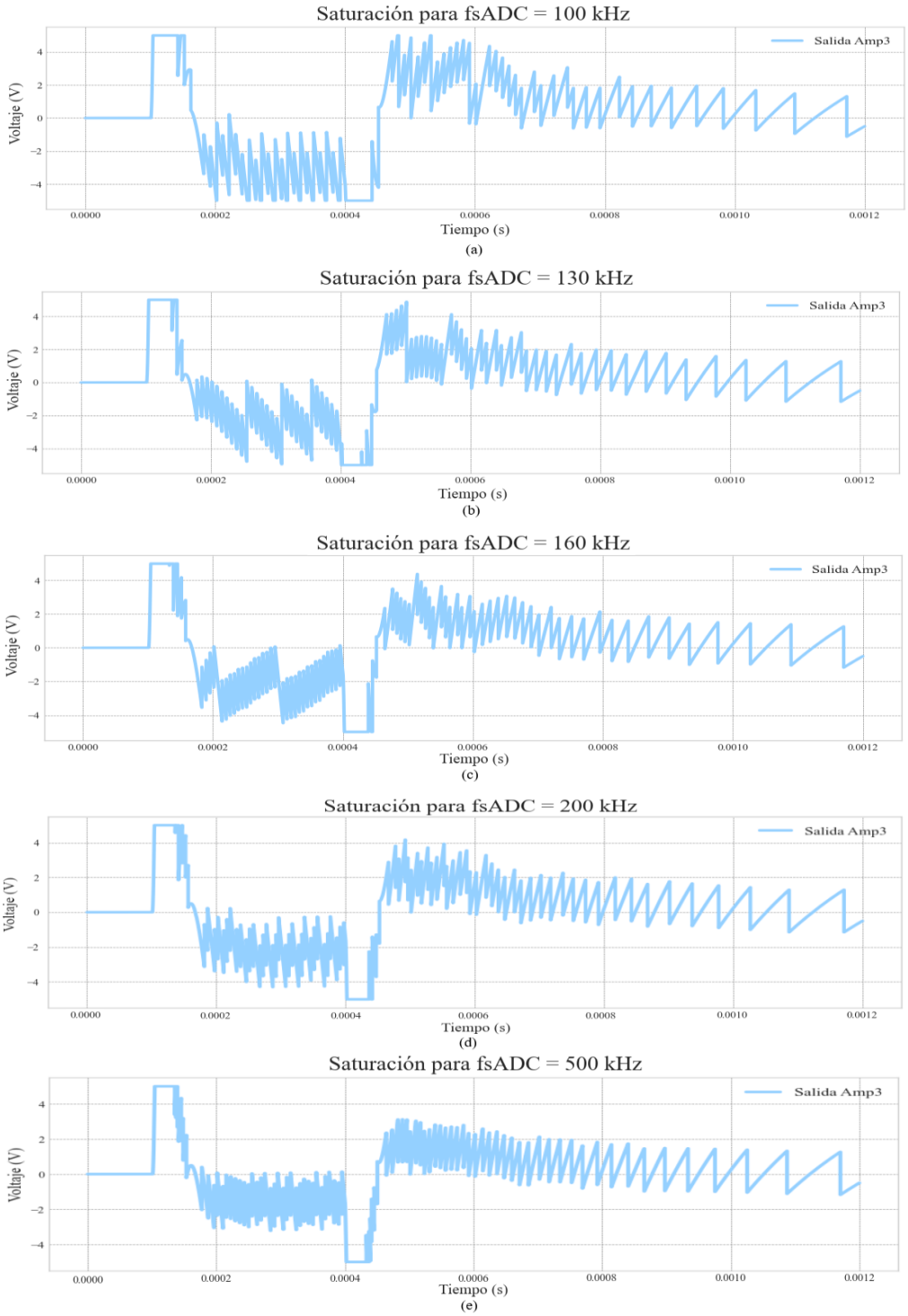


Figura 3.13: Análisis de la influencia de la frecuencia de muestreo del ADC en la saturación de la etapa de amplificación 3.

3.5. Módulo Software

Luego del módulo HW + FW se encuentra el módulo de software que se encarga de muestrear a mayor velocidad la señal de salida amplificada y cancelar el residuo del artefacto para obtener la señal neural de interés (ECAP). Esta última etapa cuenta con dos grandes bloques, un adquisidor de alta velocidad como puede ser una tarjeta adquisidora de datos y un post procesamiento de la señal adquirida para eliminar el artefacto residual. El funcionamiento del post procesamiento aplicado en la simulación es sencillo, se adquiere una plantilla de la salida de HW + FW, siendo la entrada del sistema completo únicamente la señal de artefacto. Luego se le agrega la señal de la respuesta al estímulo y se resta la plantilla previamente adquirida.

En la Fig. 3.14 se observa cómo se implementó lo descrito anteriormente en la plataforma Simulink.

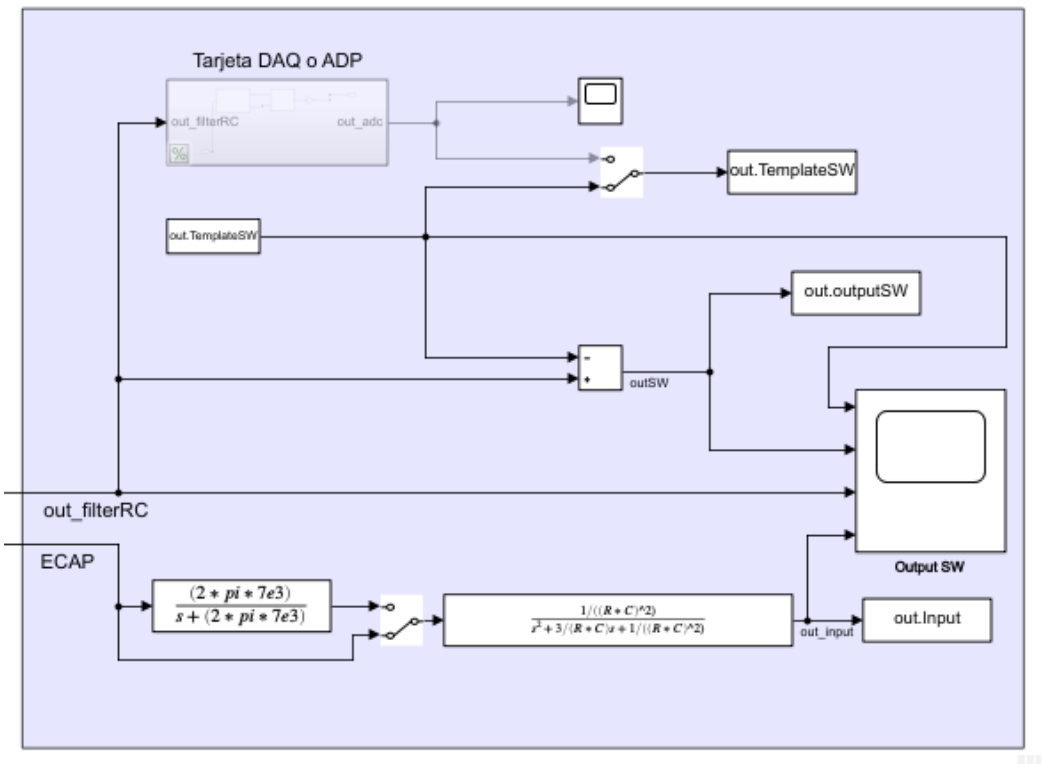


Figura 3.14: Implementación en Simulink del módulo de software.

Con esto en cuenta el módulo de software cuenta con tres bloques:

- Tarjeta de adquisición: se utiliza para muestrear la salida de la etapa HW + FW.
- Guardado y reproducción de la plantilla de SW.
- Restador de la entrada con la plantilla de SW.

Dada la simplicidad de los últimos dos bloques, se entrará en detalle únicamente para describir la construcción del bloque de la Tarjeta de Adquisición.

3.5.1. Tarjeta DAQ

En definitiva, la Tarjeta de Adquisición se implementa a partir de tres bloques simples de Simulink, como se distingue en la Fig. 3.15. En primer lugar, se utiliza una instancia de FlashADC para implementar el muestreo. La frecuencia del ADC se fija tal que la velocidad de muestreo sea mucho mayor que la implementada en la etapa de HW + FW, estando ahora en el orden de los MHz. Seguido de este bloque se encuentra una instancia para incorporar una función personalizada. La función copia las conversiones provenientes del ADC, en base a si el mismo se encuentra habilitado o no, indicado a través de la salida *ready* del ADC. Si el resultado aún no está disponible, *ready* estará en 0 y la función devolverá el último valor que le haya sido entregado por el ADC. En cambio, si *ready* está en 1, devuelve la nueva conversión. Finalmente se implementa una instancia de DAC de misma forma a la descrita en la Secc. 3.4.5.

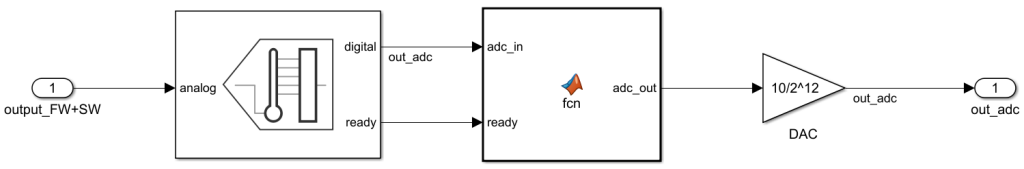


Figura 3.15: Implementación de la tarjeta de adquisición.

3.5.2. Adquisición de la Plantilla de Software

Una vez que se generó una plantilla de HW a partir de la entrada de la señal SA, se utiliza el bloque de la Tarjeta DAQ para muestrear la salida de la etapa de HW y almacenar el resultado en la plantilla de SW. Luego de que se forma la plantilla de SW, se agrega a la entrada de la etapa de HW la incidencia de la señal de ECAP. Lo que hará la etapa de post procesamiento ahora es restar la salida de la etapa de HW y la plantilla de SW generada.

Este procedimiento se realiza manualmente, controlando los switches mostrados en la Fig. 3.14.

En la Fig. 3.16 se muestran los resultados de la implementación de esta etapa luego recuperada la señal de ECAP a la salida del post procesamiento. En la Fig. 3.16.A se visualiza la señal de ECAP recuperada, mientras que en Fig. 3.16.B se encuentra la señal de ECAP vista a la entrada, habiendo pasado por los filtros correspondientes (la cual se busca recuperar).

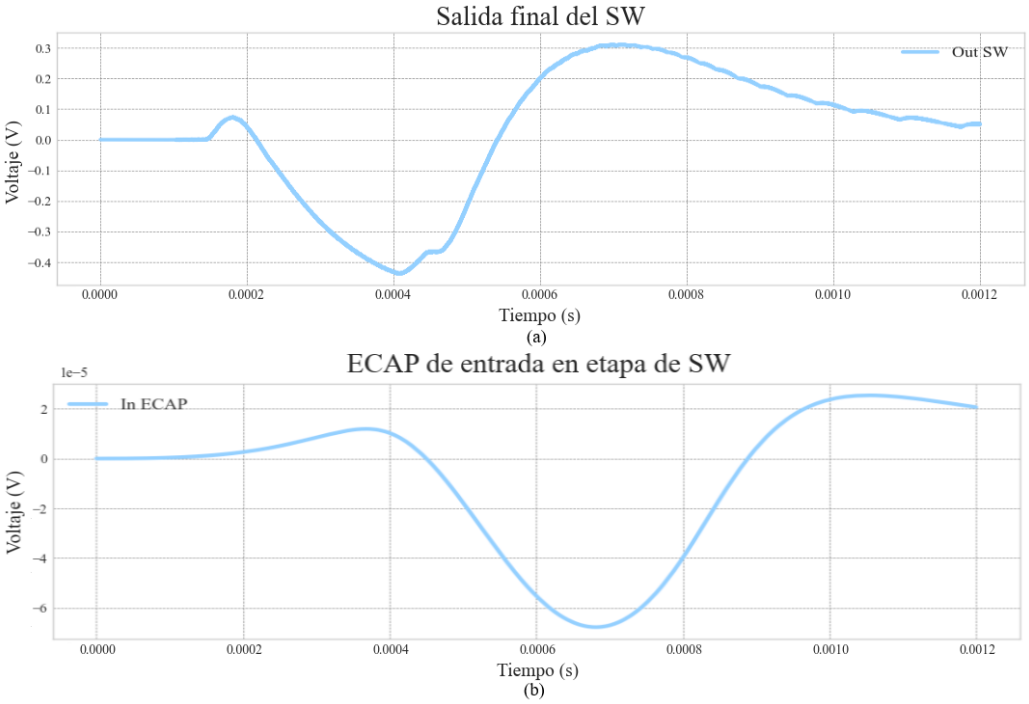


Figura 3.16: Resultado de la etapa de SW al recuperar la señal de ECAP.

3.5.3. Correlación a la Salida de SW entre Señal de ECAP Entrante y Señal de ECAP obtenida

Un parámetro relevante utilizado para la validación del sistema diseñado como solución es la correlación entre la señal ECAP de entrada y la señal ECAP obtenida a la salida del post procesamiento. Para realizar esto, se llama a la función de MatLab *corrcoef*, donde las entradas serán las variables almacenadas *out.Input* y *out.outputSW*. La conexión para estas variables se ven en la imagen que contiene la implementación del módulo de SW (ver Fig. 3.14). En la figura se visualiza que a la señal de ECAP de entrada se le efectúa un filtrado, este corresponde al mismo por el cual pasa la señal al entrar al sistema de HW. Resulta necesario el agregado del mismo, ya que la señal que recuperará el post procesamiento es aquella que se encuentre a la salida de la etapa de HW. De esta forma la comparación entre señales es coherente.

La primera prueba a realizar para estudiar la correlación es cómo varía la misma en función de las características del ruido blanco a la entrada. Para esto, se dejará la frecuencia de muestreo del ADC de HW fija en 300 kHz y se procederá a variar la densidad espectral de potencia del ruido blanco a la entrada. Se escogió esta frecuencia en base a las pruebas realizadas en la Secc. 3.4.7, sin utilizar una frecuencia mayor para asegurar que sea aplicable luego en el Firmware. El rango de valores estudiados y los resultados obtenidos de correlación se observan en la Tab. 3.2. La amplitud de las señales SA y ECAP corresponden a las máximas dentro del rango de cada una.

PSD	Amplitud máxima observada de ruido	Correlación
5e-19	2,79 μV	0.9975
5e-18	8,82 μV	0.9972
1e-16	39,4 μV	0.9965
1e-12	3,94 mV	0.6134
1e-10	39,4 mV	0.5995

Tabla 3.2: Resultados obtenidos de correlación entre la señal ECAP de entrada y la señal de ECAP obtenida a la salida del post procesamiento simulado.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Desarrollo de Hardware

En esta sección se va a describir el HW desarrollado así como todo componente externo que fue comprado. Como ya fue comentado en el Cap.2 , el sistema debe ser capaz de adquirir y procesar señales neurales, producir una plantilla, restarla a la señal de entrada y poder amplificar el residuo resultante. Para lograr esto, se pueden identificar componentes claves del sistema CANE:

- Adquisición de las señales: el HW deberá contar con una etapa de adquisición de la señal compuesta por el artefacto más la señal neural subyacente.
- Conversor Analógico Digital (ADC): Será el encargado de muestrear el artefacto para adquirir la plantilla.
- Microcontrolador (MCU): cumplirá la tareas de control del ADC y DAC, procesamiento de la plantilla, así como de sincronización y control con la estimulación.
- Conversor Digital Analógico (DAC) : Permitirá la reproducción de la plantilla para la cancelación del artefacto para obtener un residuo por debajo del voltaje de saturación.
- Etapa de substracción: permitirá la resta de la plantilla producida por el DAC y la señal adquirida para obtener el residuo.
- Etapa de amplificación: deberá amplificar la señal obtenida como resultante de la etapa de substracción que contiene la señal neural de interés. Debido a las pequeñas amplitudes de las mismas, la amplificación es clave para su correcta detección.

Todos estos componentes pueden ser identificados en la Fig. 2.2 vista previamente en el Capítulo 2. Cada uno de ellos, deberá cumplir con requerimientos mínimos que aseguren el correcto funcionamiento del sistema y la aplicación del algoritmo de *Cancelación de Artefactos*.

También será necesario encontrar una plataforma capaz de realizar la adquisición de la señal resultante de la cancelación inicial para poder implementar una siguiente etapa de post-procesamiento.

4.1. Diagrama de Bloques del Sistema

La solución implementada para este proyecto se puede ver en la figura 4.1 como diagrama de bloques. Se puede apreciar que el sistema se compone por tres grandes componentes. El primero, consiste en la placa de desarrollo, la NUCLEO-L552ZE-Q, del MCU seleccionado, el STM32-L552ZE, que fue adquirido en base a los requerimientos extraídos en el Cap.3.

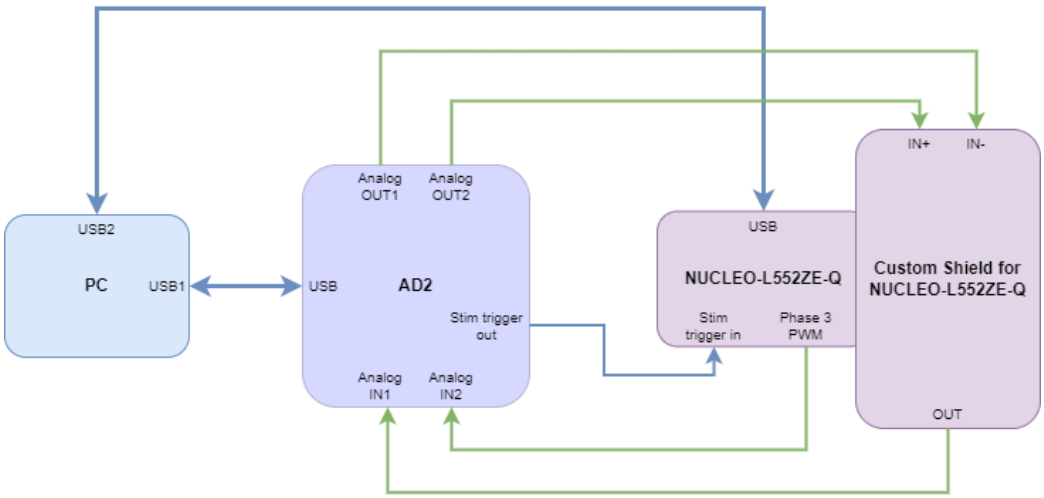


Figura 4.1: Diagrama de bloques del HW utilizado en el sistema CANE.

Luego, se cuenta con un circuito impreso (PCB), desarrollado como parte del proyecto, que incluye toda la electrónica específica que permite la adquisición de todas las señales, la cancelación del artefacto y amplificación de la señal de interés. También incorpora todas las interfaces necesarias para establecer la comunicación con la placa de desarrollo y el MCU, y la tarjeta de adquisición de datos para llevar a cabo el post-procesamiento.

El PCB fue diseñado como una placa de adaptación o *shield*, compatible con el kit de desarrollo del MCU que permite una fácil integración. Además, esta modalidad permite reducir de manera considerable el trabajo de desarrollo de HW pues no es necesario incluir el MCU en el PCB, simplificando su diseño así como disminuyendo el riesgo asociado a errores que dificulten o no permitan el uso de la PCB. Debido a las características del diseño de HW, cometer un error luego de la manufactura y ensamblaje de la PCB, que imposibilite su uso, consiste en un riesgo demasiado grande debido a las características de tiempo y recursos finitos del proyecto.

Se hará uso del Analog Discovery 2 (AD2) como instrumento multipropósito tanto para la adquisición, como para la generación y coordinación de todas las señales de interés, siendo controlado desde el post-procesamiento para llevar a cabo todas estas tareas. La elección del AD2 fue impulsada por la familiaridad de los integrantes del proyecto con la herramienta así como la versatilidad del instrumento al proveer una API en Python que permite una fácil integración de la

herramienta con los requerimientos del post procesamiento.

4.1.1. Elección del microcontrolador

Como fue mencionado antes, el MCU elegido es el STM32-L552ZE, el cual fue seleccionado tras realizar una detallada búsqueda en proveedores como Mouser [12] o Digikey [13], donde una serie de factores se tuvieron en cuenta. Se partió de la base de que el mismo debía de cumplir con las características del MCU utilizado por [2], el ATSAM3X8E-AU [14] de Atmel, un ARM-32 de frecuencia máxima 84 MHz con ADC y DAC internos así como múltiples periféricos de comunicación como SPI, I2C y UART. Como se mencionó anteriormente, también se tuvieron en cuenta los resultados de la simulación con respecto a frecuencia de muestreo mínimo y velocidad de procesamiento de datos.

Con esto en cuenta se realizó la búsqueda teniendo las siguientes consideraciones:

- Procesador de 32-bits.
- Máxima frecuencia reloj igual o superior a 84MHz.
- ADC y DACs internos que permitan una velocidad de muestreo mayor a $200kHz$.
- Al menos dos periféricos independientes tanto de los protocolos SPI como I2C que permitan el uso de un ADC o DAC externos con una velocidad de muestreo mayor a $200kHz$.
- Periféricos DMA[1] que puedan ser usados tanto con el ADC o DAC así como los periféricos de comunicación mencionados anteriormente.
- Stock de los respectivos kits de desarrollo en los proveedores mencionados anteriormente.

Con todas estas consideraciones, el STM32-L552ZE, fue el MCU con kit de desarrollo en stock que cumpliera con todas estas especificaciones. En particular cuenta con:

- Máxima frecuencia de reloj de $110MHz$.
- 2 ADCs de 12-bit con frecuencia máxima de 5 Msps, y hasta 16-bits con over-sampling.
- 2 DACs de 12-bits.

[1]Direct Memory Acces

4.2. CANE STM32-L552ZE Shield

En esta sección se entrará en profundidad en el diseño de la placa de adaptación para el kit NUCLEO-L552ZE-Q. Se verá cuales son los requisitos con los que debe contar el *shield* para poder cumplir la función propuesta.

4.2.1. Requerimientos

No se entrará en profundidad en los requerimientos detallados, sin embargo es importante mencionar ciertos requerimientos generales que se propusieron como meta a la hora del diseño del HW:

- Debido a la pequeña amplitud de las ECAP que se apunta a medir, 2 a 150 μV, se consideró la posibilidad de tener que amplificar en la etapa de entrada de nuestro sistema. Esto mejoraría la SNR de la ECAP contra el ruido aleatorio introducido por la electrónica, lo cual es clave, al estar intentando detectar señales tan chicas. Amplificar la señal de entrada previo a la cancelación del artefacto afecta directamente la amplitud máxima del artefacto admisible. Se tomó la desición de contar con un mecanismo de ganancia variable que pueda amplificar la señal de entrada previa a la canelación en un factor de al menos $\times 10\ V/V$.

- Se puso como meta que el *shield* provea al MCU la capacidad de poder autosincronizarse con la señal detectada a partir de la llegada de la misma en la entrada, sin la necesidad de que un elemento externo indique el comienzo de la estimulación.

- Además, se decidió que el *shield* incorporará ADCs y DACs externos que permitan mayor rango de excursión, con la posibilidad de obtener y generar valores bipolares, y con un mejor desempeño en cuanto a las no idealidades, que los convertidores integrados en el MCU. Esta decisión fue tomada como una posible mitigación ante el riesgo de que usar los convertidores internos del MCU no se pudiera aplicar la cancelación del artefacto de estimulación.

- Por otra parte, para mejorar la experiencia de uso, se decidió que el *shield* sea capaz de generar todas las tensiones requeridas por sus componentes, a partir de la alimentación de 5V proveniente del USB del NUCLEO-L552ZE-Q.

- Por último, al ser el *shield* parte de un prototipo, se planteó tomar en cuenta todas las consideraciones necesarias para facilitar la depuración y testeo del sistema. Como pueden ser la capacidad de desconectar circuitos por medio de resistencias de $0\,\Omega$ o jumpers así como poder alimentar el circuito tanto directamente de la placa NUCLEO o desde fuentes externas.

4.2.2. Diagrama de Bloques

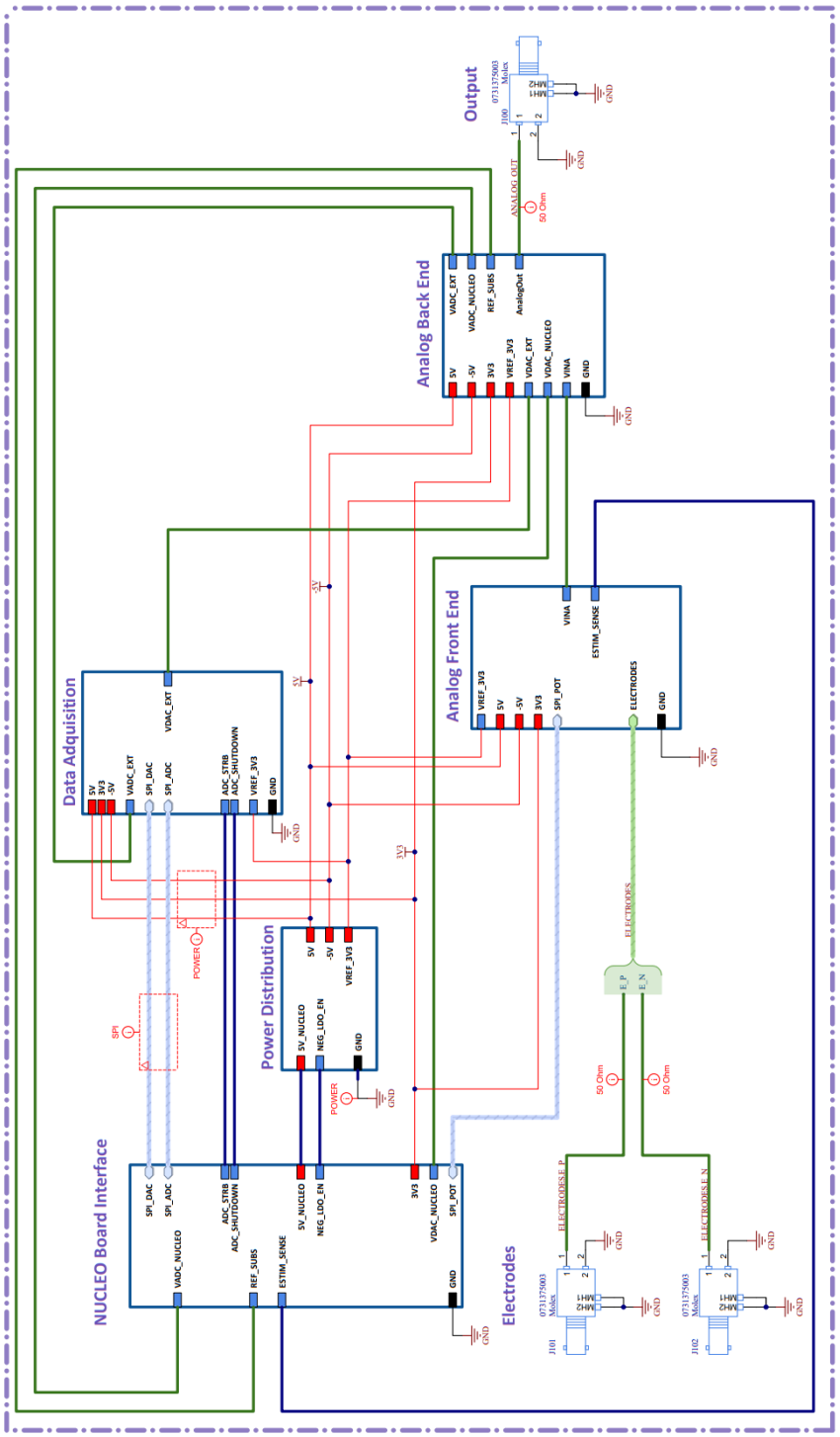


Figura 4.2: Diagrama de bloques del *shield* desarrollado para la NUCLEO-L552ZE-Q

En la Fig.4.2 se puede observar un diagrama de bloques de la electrónica del *shield*, en donde las señales relacionadas a la alimentación se encuentran en rojo, GND en negro, las señales analógicas en verde y las digitales en azul. Se pueden identificar cinco grandes bloques con funciones definidas.

NUCLEO Board Interface Se encuentra compuesto por todas las conexiones necesarias para poder conectar adecuadamente el *shield* con la placa de desarrollo del MCU. Además, incluye todas las protecciones necesarias, como pueden ser de sobre voltaje, para evitar cualquier daño a los pines del microcontrolador.

Power Distribution Es el bloque encargado de la generación y configuración de todas las alimentaciones del sistema. Esto incluye la generación de una fuente de $-5\,\mathrm{V}$ así como de una referencia de precisión de $3{,}3\,\mathrm{V}$.

Analog Front End El Analog Front End (AFE) consiste en la adquisición inicial de la señal proveniente del cuerpo humano, aunque es importante aclarar, que fue diseñado para sensar este tipo de señales, pero todas las pruebas fueron realizadas con generadores arbitrarios de ondas. El AFE permite aislar la señal en modo diferencial, proveniente de los electrodos (con conectores BNC), de la señal en modo común y su continua. Además, permite una amplificación inicial de la señal a partir de un mecanismo de selección de ganancia variable, así como un circuito de sincronización para la correcta coordinación del MCU con el inicio de la estimulación. La selección de la ganancia variable es controlada por el MCU por medio de un bus SPI bajo el nombre SPI_POT en el diagrama, y la sincronización de la estimulación se realiza por medio de la señal de salida ESTIM_SENSE. La salida analógica, V_{INA} de este bloque, contiene el artefacto y la señal neural subyacente.

Analog Back End El Analog Back End (ABE) recibe la señal adquirida por el AFE e implementa la etapa de substracción con la plantilla adquirida por el MCU, que puede ser producida tanto por el DAC interno del mismo o por el DAC externo situado en el bloque Adquisición de Datos. Es en el ABE en donde se realiza la amplificación del residuo proveniente de la resta por un factor G, lo suficientemente grande para asegurar la detección de la ECAP. Además, implementa varios filtros a lo largo de la cadena y finalmente la salida se encuentra en un conector de salida de tipo BNC.

Adquisición de datos Este último bloque incluye los ADC y DAC externos y toda la circuitería analógica necesaria para su correcto funcionamiento. Como se puede ver en la Fig.4.2, tanto el ADC y DAC se comunican por un bus SPI independiente y señales de control digitales.

En las siguientes secciones se expandirá sobre el diseño e implementación de cada bloque descrito anteriormente.

4.2.3. Analog Front End

El AFE esta compuesto por tres bloques, estos son:

- Amplificador de instrumentación INA.
- Potenciómetro digital.
- Schmitt Trigger.

Amplificador de instrumentación

La entrada de este circuito consiste en una señal diferencial, con una componente DC y señal en modo común. Por lo tanto, es necesario contar con una primera etapa de alto rechazo de modo común, con desacople de continua a la entrada y bajo ruido. Recordar que el sistema cancela los artefactos previo a la etapa de alta ganancia, y por ende, todo ruido introducido será también amplificado. Además, la etapa debe contar con el suficiente rango de entrada para las señales más grandes esperadas a la entrada. Esta consiste en una SA de amplitud máxima 140 mV en modo diferencial superpuesta con una señal en modo común que puede variar entre ±3,47 V como fue discutido en la Secc. 2.2.3. También, se debe contar con una excursión a la salida capaz de generar un SA amplificada por un factor ×10 de la misma, es decir 1,4 V.

Para esto se utilizó un Amplificador de Instrumentación integrado (INA según sus siglas en inglés), el AD8221ARMZ-R7 [15] de Analog Devices [16] en la configuración mostrada en la Fig. 4.3.

Se utilizan los filtros pasa-altos compuestos por los componentes C_{304}, C_{305} y R_{300} y R_{303} para el desacople de continua a la entrada que definen un filtro con polo en 35 Hz. Como el objetivo de este filtro discreto es filtrar la continua a la entrada, la frecuencia de corte fue elegida con el criterio de estar una década por debajo de la menor frecuencia de la banda de interés del ECAP, es decir 300 Hz.

Este circuito configura la continua a la entrada del INA, en modo diferencial y común en 0 V. En particular, es de sumo interés filtrar la componente DC en modo común a la entrada, caracterizada en ±2 V y así poder reducir la excursión máxima de la señal en modo común a las entradas del INA. Sin la componente en DC, esta queda determinada en ±1,47 V como se vio en la Secc. 2.2.3.

El AD8221 fue seleccionado tras una extensa búsqueda donde sus diferentes características fueron comparadas, entre las cuales, las más importantes son:

- Ruido equivalente en voltage de entrada a 1 kHz de $8\,\mathrm{nV}/\sqrt{(}\mathrm{Hz})$.
- Ruido aditivo en voltaje a la salida a 1 kHz de $75\,\mathrm{nV}/\sqrt{(}\mathrm{Hz})$.
- Ruido equivalente en corriente de entrada a 1 kHz de $50\,\mathrm{fA}/\sqrt{\mathrm{Hz}}$.
- Frecuencia de corte de −3 dB de 825 kHz cuando tiene ganancia configurada de $G = 1$.
- CMRR de 90 dB .

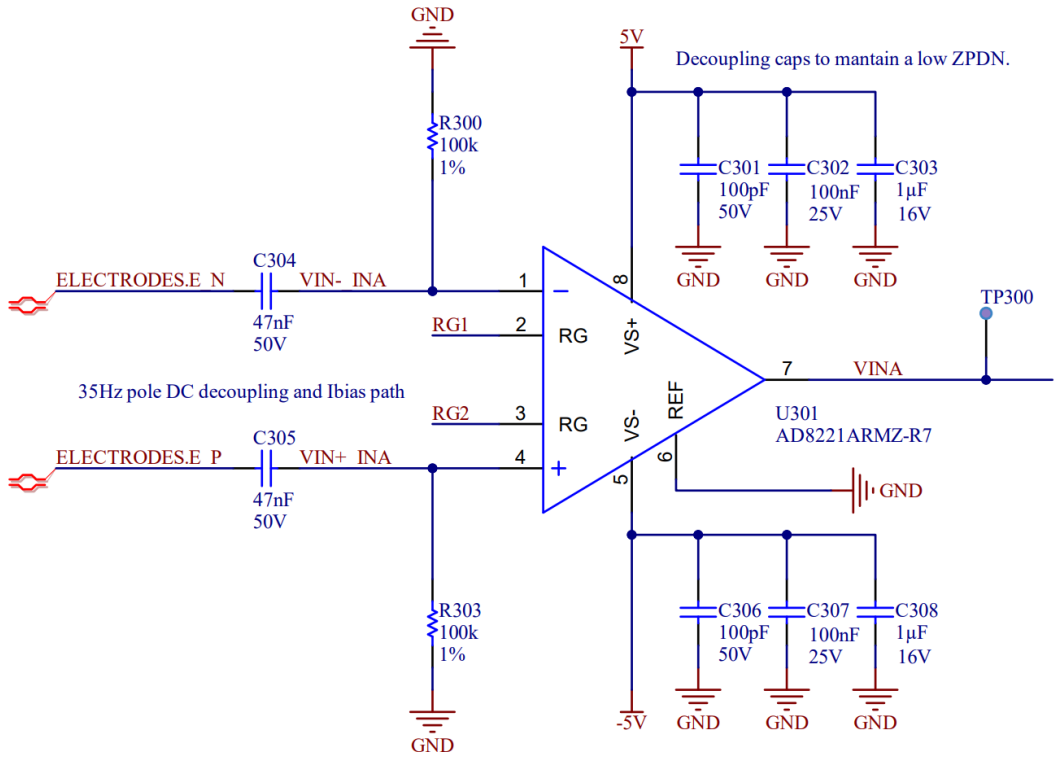


Figura 4.3: Circuito implementado para la primera etapa de amplificación basada en un INA integrado de tres amplificadores

- Offset en voltaje de 60 µV.
- Un ICMR y OSW como muestra la Fig.4.4. Notar que el ICMR y OSW no son independientes. Cuanto mayor sea el OSW necesario, menor es el ICMR y viceversa. Para esta aplicación en particular, se espera como máximo una señal a la salida del INA de ±1,4 V, lo cual corresponde a un ICMR aproximadamente entre −3 V y 4 V. Mayor que la variación esperada en modo común a la entrada de ±1,47 V.

Una comparación al detalle con las otras opciones exploradas puede observarse en el Anexo. D.1.1.

En la Fig. 4.3 se pueden observar diversos condensadores de desacople en la fuentes de ±5 V para filtrar ruido de diferentes frecuencias. A pesar de esto, hay que tener en cuenta que el AD8221 cuenta con un bajo PSRR, donde el PSRR negativo es el peor, llegando a 20 dB a 100 kHz. Diferentes consideraciones en el diseño de la fuente de −5 V fueron tomadas para minimizar el efecto del PSRR como se describirá más adelante en la sección 4.2.6. La ganancia del AD2881 esta determinada a partir de la resistencia colocada entre las terminales R_G (pines 2 y 3) y la siguiente ecuación:

$$G = \frac{49{,}4\,\text{k}\Omega}{R_G} + 1 \tag{4.1}$$

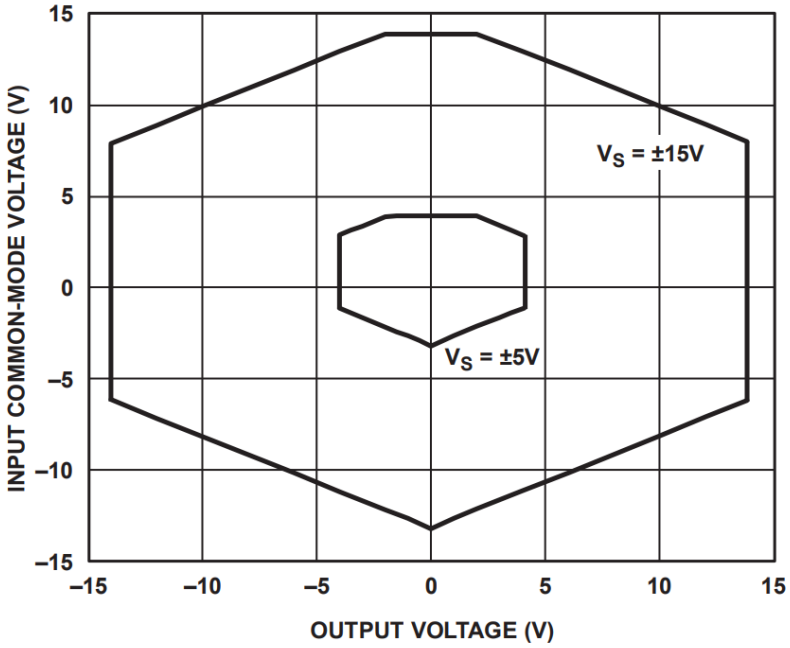


Figura 4.4: ICMR y OSW del AD2881 [*Extraído de la datasheet del AD2881*]

Cuando no se encuentra ninguna resistencia entre los pines mencionados, la ganancia del INA pasa a ser 1 V/V.

Para asegurar la correcta operación del circuito, se simuló el mismo junto con el modelo del modo común a la entrada como se puede ver en la Fig. 4.5, bajo las condiciones más restrictivas de funcionamiento:

- $V_{MID} = 2\,\text{V}$.
- $I_{stim} = 2{,}5\,\text{mA}$.
- Señal en modo diferencial a la entrada un seno de amplitud 140 mV y frecuencia en banda pasante de 2 kHz.
- Ganancia del amplificador diferencial en 10 V/V.

Los resultados de la simulación se pueden ver en la Fig. 4.6, donde el filtro a la entrada corta la DC del modo común (señal V_CM pre filtro). Luego, no se genera ninguna distorsión de la onda de la salida de la amplitud esperada de 1,4 V (señal V_OUT INA), a pesar de encontrar una componente en el modo común a la entrada del amplificador (señal V_CM_INA) de 1,47 Vpp.

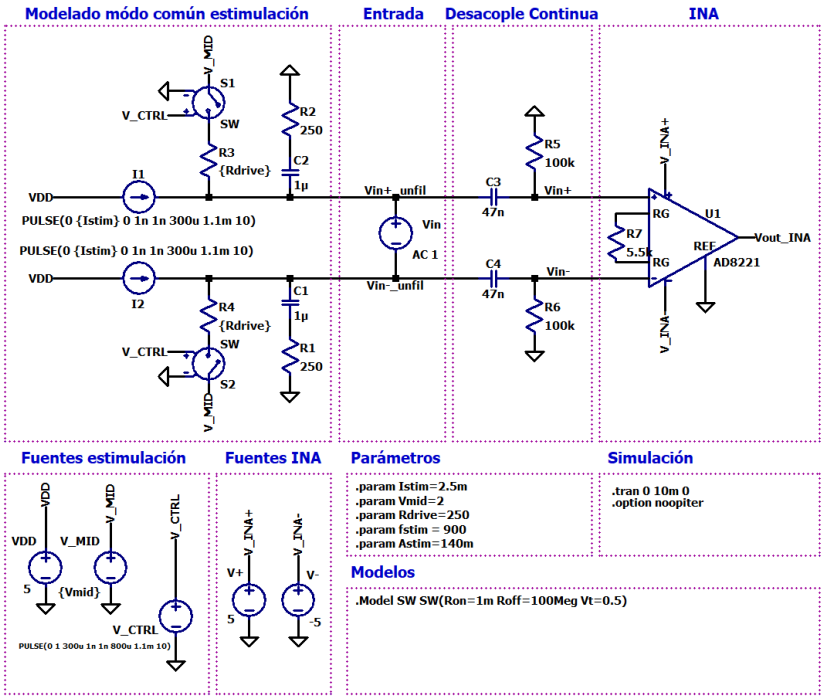


Figura 4.5: Circuito utilizado para la simulación del modo común a la entrada del INA.

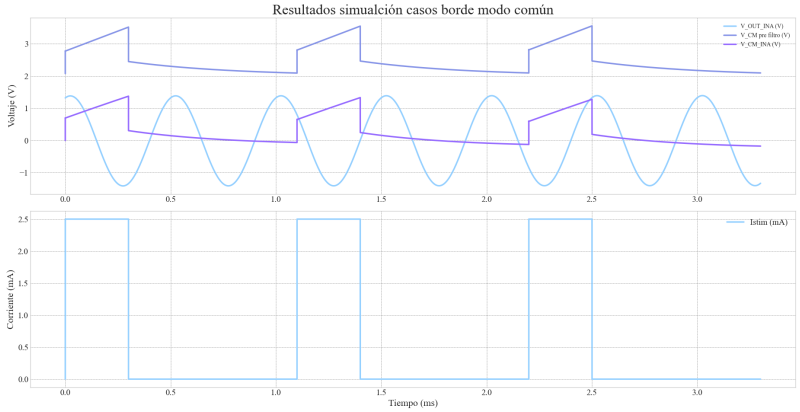


Figura 4.6: Resultados de la simulación realizada para evaluar el funcionamiento de la primera etapa de adquisición del CANE shield bajo las peores condiciones de excursión de entrada y de salida.

Potenciómetro Digital

La resistencia R_G del INA fue implementada utilizando un potenciómetro digital de 128 bits, el MCP4131-503E/SN [17] de Microchip [18], como muestra la Fig. 4.7. Este integrado implementa una red de resistencias de valor máximo 50 kΩ que puede ser seleccionada en pasos de a 390 Ω, donde la posición de las terminales W

puede configurarse en cualquiera de las 128 posiciones. Además, cualquiera de las tres terminales A, B y W pueden desconectarse de la red interna de resistencias.

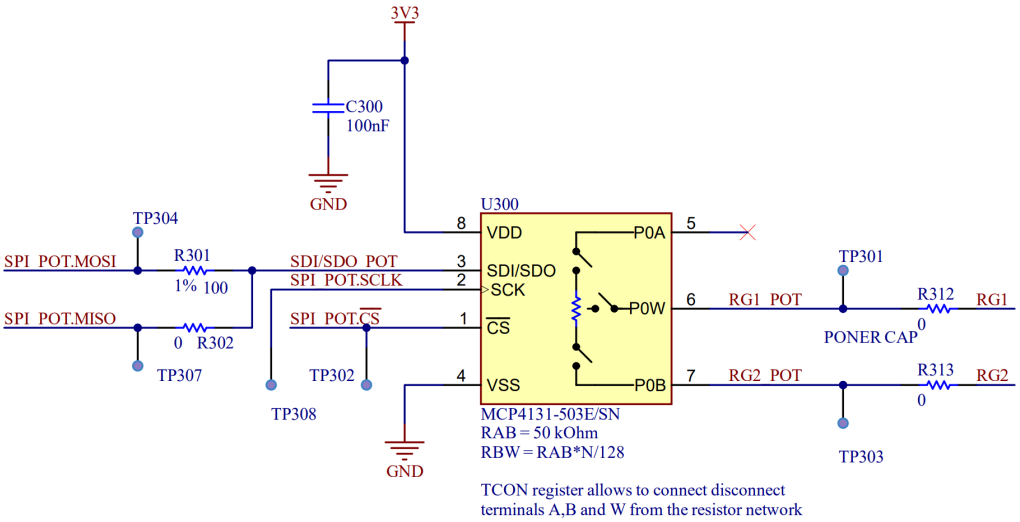


Figura 4.7: Implementación de la resistencia de ganancia del INA AD8221 por medio de un potenciómetro digital

Se conectaron las terminales W y B del potenciómetro como la resistencia R_G de tal manera que al ser variada por comandos SPI provenientes del MCU, se pueda cambiar la ganancia del INA. La ecuación de la resistencia entre las terminales $P0W$ y $P0B$ a partir del valor[2] guardado en el registro *Volatile Wiper 0* N en la dirección $00h$ del potenciómetro es:

$$R_{BW} = \frac{50k\Omega \times N}{128} \tag{4.2}$$

Durante las pruebas de validación del HW, se encontró un problema de diseño con respecto al mecanismo implementado para obtener una ganancia variable en el INA. El potenciómetro digital, al trabajar entre GND y $V_{DD} = 3{,}3\,\text{V}$, solo permite que el voltaje de todos sus pines excursionen entre estos dos puntos. Esto incluye a las terminales $P0W$ y $P0B$ que se encuentran conectadas al INA que pueden llegar a variar por debajo de 0 V o por arriba de 3,3 V dependiendo de la señal de entrada. Fue debido a esto que, al validar el funcionamiento del mecanismo de ganancia variable, se observó que la resistencia vista entre $P0W$ y $P0B$ configurada por FW no se comportaba como era esperado. En la Secc.4.4 se explorará en más detalle este problema y las posibles soluciones del mismo.

Schmitt Trigger

Durante las pruebas iniciales al inicio del proyecto, se observó que la señal SA cuenta con flancos de subida suficientemente rápidos como para ser usados para el

[2]Para el MCP4131-503, N solo puede variar entre 0 y 127, es decir es una palabra digital de 7 bit.

inicio de la *Cancelación del Artefacto*.

Fue debido a esto, que se decidió implementar un Schmitt-Trigger que genere una señal digital que se coordine con los flancos provenientes del artefacto en la entrada. El circuito correspondiente se puede ver en la Fig. 4.8

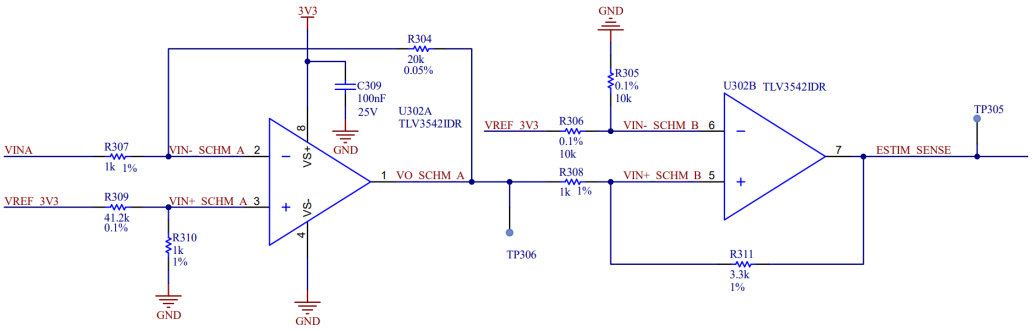


Figura 4.8: Circuito implementado para el Schmitt-Trigger y detección de la estimulación.

Primero se cuenta con una etapa de amplificación de ganancia de 20 V/V implementada por un amplificador en configuración inversora cuya entrada es la salida del INA. La señal $V_{O\_SCHM\_A}$ se encuentra determinada por la siguiente ecuación:

$$V_{O\_SCHM\_A} = \frac{-R_{304}}{R_{307}} V_{INA} + 3{,}3V \times \frac{R_{310}}{R_{310} + R_{309}} \frac{R_{307} + R_{304}}{R_{304}}. \tag{4.3}$$

Que para los valores elegidos de resistencia, se obtiene:

$$V_{O\_SCHM\_A} = 20\,\mathrm{V/V} \times V_{INA} + 1{,}65\,\mathrm{V} \tag{4.4}$$

Luego del preamplificador, se encuentra el Schmitt-Trigger que fue diseñado con una tensión de referencia de 1,65 V generada a partir del divisor resistivo R_{305}, R_{306} y la referencia de precisión $VREF\_3V3$. Dado la tensión de salida (0 o 3,3 V) y la de referencia, el umbral de conmutación está dado por:

$$V_{O\_SCHM\_A} = 1{,}65\,\mathrm{V} \times \frac{R_{311} + R_{308}}{R_{311}} - V_{OUT} \frac{R_{R311}}{R_{308}}. \tag{4.5}$$

Para los valores de resistencia elegidos, estos quedan determinados en $V_{trigger-} = 1{,}15\,\mathrm{V}$ y $V_{trigger+} = 2{,}15\,\mathrm{V}$. Las resistencias fueron seleccionadas a partir de simulaciones realizadas en LTSpice para minimizar el tiempo de levantamiento. En el peor caso de una SA a la entrada de amplitud mínima (70 mV), se obtuvo un tiempo de levantamiento de aproximadamente 0,1 µs.

Para implementar tanto el inversor como el Schmitt-Trigger, se utilizó el amplificador TLV3542 [19] de Texas Intruments [20] que fue elegido bajo el criterio de tener alto Slew Rate (SR) y ancho de banda con tal de minimizar el tiempo de levantamiento. La salida del Scmitt-Trigger, $ESTIM\_SENSE$, se conecta directamente a un pin del MCU.

4.2.4. Analog Back End

El Analog Back End cuenta con los siguientes tres circuitos:

- Segunda etapa de amplificación donde se da la *Cancelación del Artefacto*.
- Tercera etapa de amplificación que se la conocerá como la etapa de ganancia de 60 dB.
- Filtro de segundo orden pasivo.

Segunda Etapa de Amplificación

La segunda etapa de amplificación se puede observar en la Fig. 4.9 donde se utiliza un amplificador diferencial para restar a la salida las dos señales V_INA (proveniente del AFE) y V_DAC. Es en esta última donde se reproduce la plantilla y puede ser seleccionada como la salida de DAC del MCU o del DAC externo a partir de la conexión de las resistencias de 0 Ω, R_{403} y R_{405} como muestra la Fig. 4.10.

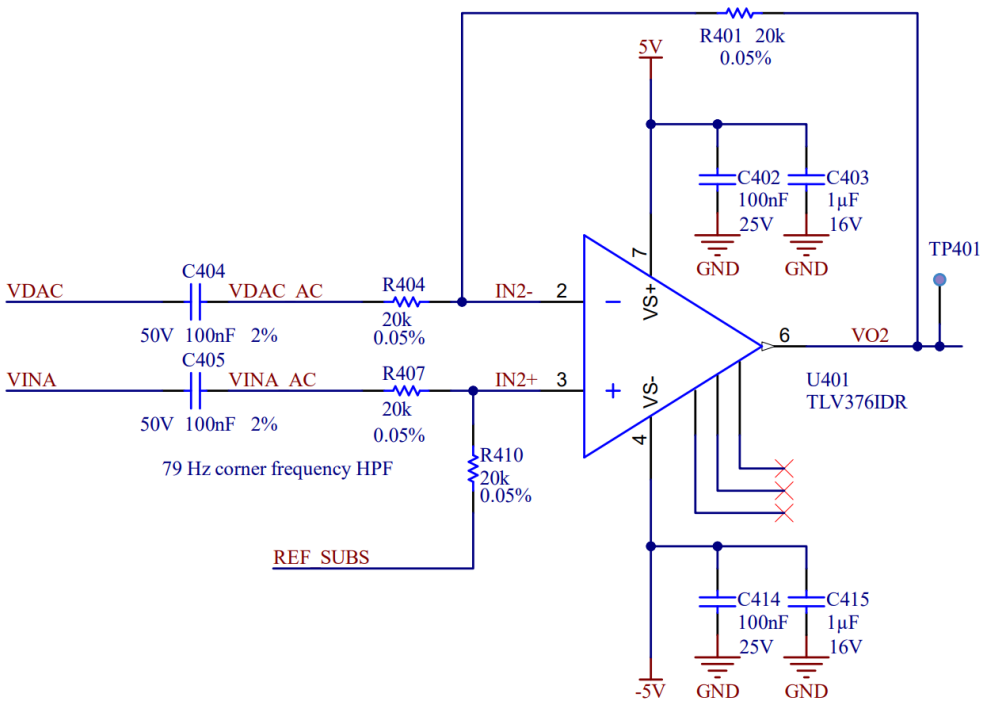


Figura 4.9: Segunda etapa de amplificación para la realización de la cancelación de artefactos

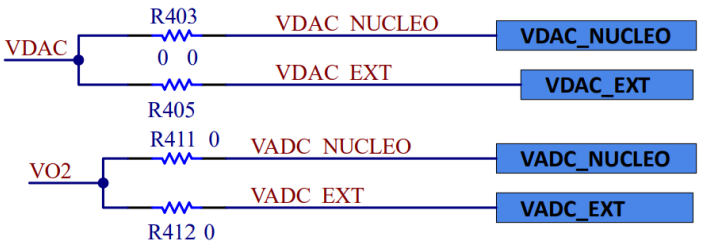


Figura 4.10: Selección de la señal de DAC a utilizar a partir de la conexión de las resistencias de 0 Ω

La salida del restador queda determinada por la siguiente ecuación[3]:

$$V_{O2} = V_{REF\_SUB} + \frac{R_{FB}C_c s}{1 + R_{FB}C_c s}(V_{INA} - V_{DAC}) \tag{4.6}$$

Donde R_{FB} es el valor de las resistencias 401, 404, 407 y 410, C_c el valor de los condensadores C_{404} y C_{405}, y V_{REF_SUB} el voltaje de referencia para determinar la continua de la salida. V_{REF_SUB} se encuentra generado por el circuito de la Fig. 4.11, donde se puede configurar para ser que su valor sea 1,65 V o 0 V dependiendo de si el jumper J_{400} se encuentra o no conectado, la resistencia R_{417} cumple la misma función. Además, desconectando el jumper J_{400} se puede configurar externamente el voltaje de referencia. Se utiliza un amplificador como buffer para asegurar una resistencia de salida casi nula para no afectar el apareo de las resistencias del restador que determinan el CMRR del mismo.

El polo pasa-altos en la transferencia del restador es $f_p = \frac{1}{2\pi R_{FB}C_c}$, que para los valores de resistencia y capacidad elegidos queda determinado en 79 Hz. Este valor fue elegido con el criterio de encontrarse por debajo de la frecuencia inferior de la banda de interés de la ECAP (300 Hz) pero no lo suficientemente bajo para obtener efectos transitorios lentos a las entrada del amplificador.

En el caso ideal de funcionamiento del sistema, en donde el artefacto en V_{INA} y la plantilla reproducida por el DAC en V_{DAC} son iguales, se tiene un amplificador diferencial con entrada diferencial nula y entrada en modo común definida por $V_{DAC} = V_{INA}$, que en este caso hipotético son iguales. En un amplificador diferencial ideal, la ganancia en modo común es nula, pero esto deja de suceder para resistencias y condensadores con tolerancias finitas.

Por lo tanto, para asegurar la correcta resta entre la plantilla y el artefacto, buscamos una ganancia en modo común lo más cercana a cero posible. Esto es equivalente a un CMRR del amplificador diferencial lo más alto posible, para lo cual, es clave el uso de resistencias y condensadores de precisión de tal manera que la transferencia vista para la señales V_{INA} y V_{DAC} sean la misma y la resta se efectúe lo mejor posible.

Las resistencias 401, 404, 407 y 410 fueron elegidas tales que, bajo el peor caso de variación de las mismas, el CMRR del amplificador diferencial sea mayor o igual a 60 dB. Este valor asegura que, ante una plantilla ideal, la cancelación del

[3]El desarrollo completo se puede observar en el anexo D.2.

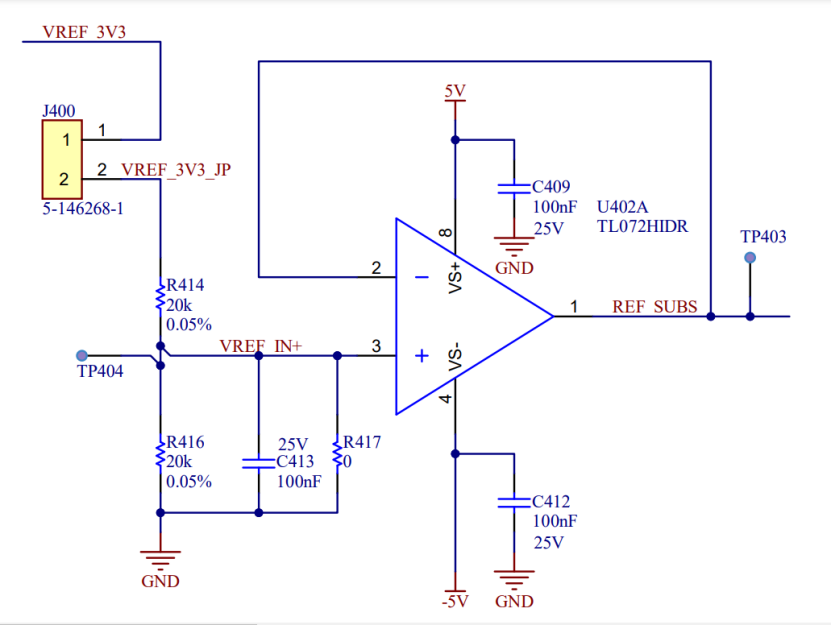


Figura 4.11: Generación de la referencia de tensión para la configuración de continua de la salida del restador.

artefacto resulte en un residuo mil veces menor que la amplitud de la SA inicial. Este factor asegura que, ante el artefacto de mayor amplitud esperado a la entrada, de 1,4 V, se obtenga ante una plantilla ideal, un residuo de 1,4 mV de amplitud a la salida. Correspondiente a 3,6 mV por debajo del límite de saturación de ±5 mV de la etapa de amplificación de 60 dB.

El máximo CMRR debido al desapareo de las resistencias en un amplificador diferencial viene dado por la siguiente ecuación[4]:

$$CMRR = \frac{G + 1 + t^2(1 - G)}{4t} \tag{4.7}$$

Donde G es la ganancia del amplificador diferencial que en nuestro caso es 1. Por ende, el $CMRR = \frac{1}{2t}$ y entonces, para asegurar un CMRR de 60 dB, se utilizaron resistencias de tolerancia de 0,05 %.

Además de las resistencias, la selección de la tolerancia de los condensadores C_{404} y C_{405} también afecta el CMRR de la etapa, pues cambia la frecuencia de polo pasa-altos de la transferencia para las señales V_{INA} y V_{DAC}. Debido a la dificultad de encontrar condensadores de precisiones mayores al 1 %, se terminó usando condensadores de 2 %. Por medio de una simulación en LTSpice, donde se evaluaron todos los casos de borde de máxima variación para los componentes discretos del circuito, se relevó el CMRR a lo largo de la frecuencia, obteniéndose el peor caso como se puede ver en la Fig.4.12. El valor mínimo corresponde a 37 dB dentro del ancho de banda de la SA de 16 kHz.

[4]El desarrollo completo se puede observar en el anexo D.2.

Dicho peor caso se da para cuando el condensador C_{404} cuenta con su valor máximo posible de 102 nF, C_{405} su valor mínimo de 98 nF, mientras que las resistencias R_{401} y R_{404} toman el valor de 19,99 kΩ y R_{407} y R_{410} el valor de 20,01 kΩ. La disminución del CMRR del valor diseñado de 60 dB a 37 dB se debe al desapareo de los condensadores que cuentan con la mayor variación posible de su valor nominal.

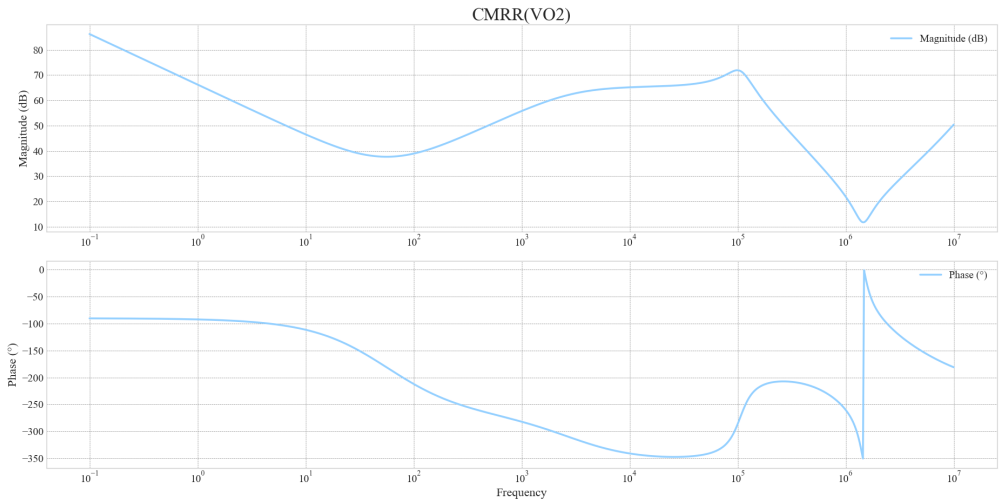


Figura 4.12: Peor resultado obtenido para el ensayo de casos de borde para la ganancia en modo común CMRR de la etapa de amplificación dos.

Por último, el amplificador seleccionado es el TLV376IDR [21] de Texas Instruments que fue elegido para ser de bajo ruido, rail-to-rail, y de rápida respuesta en frecuencia [5]. Esta última característica se debe a que la señal proveniente del DAC, al estar compuesta por escalones de incrementos de voltaje, necesita que el amplificador permita un tiempo de levantamiento pequeño con tal de reducir el error entre la plantilla y el artefacto lo más posible. Es por eso que el TLV376IDR fue seleccionado al tener un ruido en voltaje equivalente a la entrada de $8\,\mathrm{nV}/\sqrt{\mathrm{Hz}}$, $f_t = 5{,}5\,\mathrm{MHz}$ y un settling time de 2 µs. Debido a la configuración del amplificador y su f_t, se establece un filtro pasa-bajos de frecuencia de corte de 2,75 MHz.

La salida V_{O2} del restador se encuentra conectado a la entrada del ADC del MCU o del ADC externo para la adquisición de la plantilla. También se encuentra conectada a la entrada de la tercera y última etapa de amplificación como se puede ver en la Fig.4.13. Cuando se utiliza el ADC del MCU, la DC de la salida del restador debe ser 1,65 V, mientras que cuando se utiliza el ADC externo se puede utilizar 0 V.

[5] Una comparación al detalle contra las otras opciones consideradas puede observarse en el Anexo. D.2

Etapa de amplificación 60 dB

La tercera etapa de amplificación consiste en una de alta ganancia construida a partir del amplificador OPA197IDGK [22] de Texas Instruments en configuración inversora con realimentación de continua activa a la salida a partir del TL072HIDR [23] actuando como un integrador. Una discusión al detalle de la elección del OPA197DGK puede encontrarse en el Anexo.D.3.

Entre los dos amplificadores usados, se establece un filtro pasa-banda que fue diseñado de manera teórica y verificado con simulación para que cumpliera con una ganancia de 60 dB en banda pasante, frecuencia de corte pasa-altos de $f_{p1} = 500\,\text{Hz}$ y frecuencia de corte pasa-bajos de $f_{p2} = 9{,}4\,\text{kHz}$ [6]. Estas frecuencias, junto con las del filtro pasa bajo discreto que se describirá más adelante, fueron diseñadas para obtener una banda pasante correspondiente a la banda de la señal de interés, el ECAP, entre 300 Hz y 5 kHz.

En la Fig. 4.13 se puede observar el circuito implementado mientras que en la Fig.4.14 se tiene el Bode del filtro pasa-banda impuesto por la etapa de amplificación tres donde se tiene que $f_{p1} = 470\,\text{Hz}$ y $f_{p2} = 10{,}7\,\text{kHz}$.

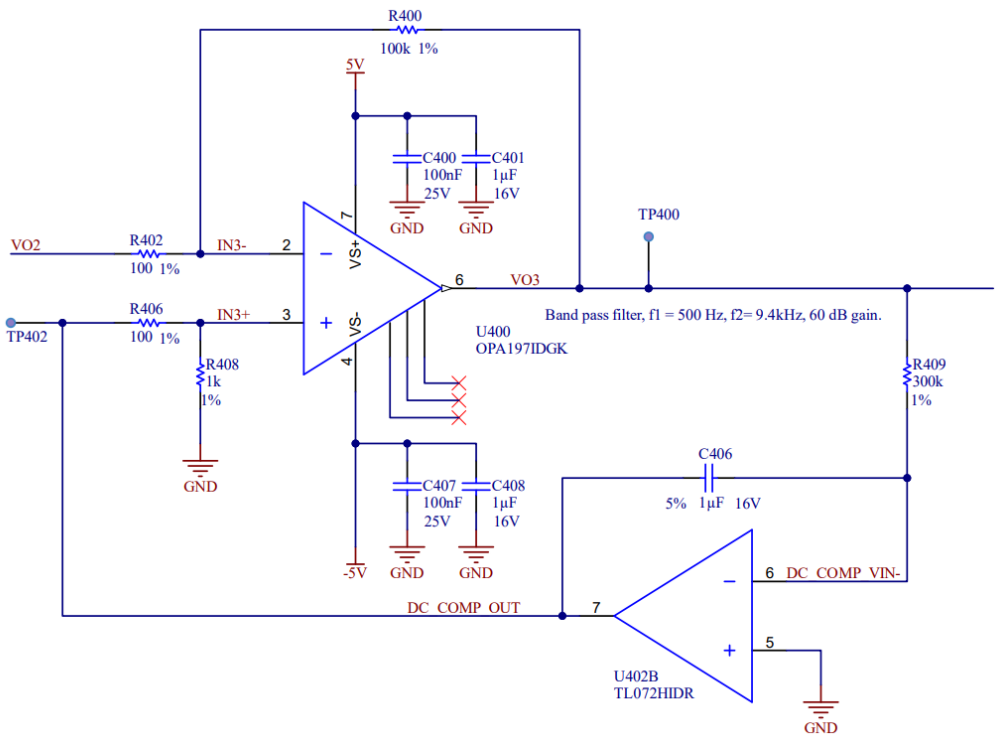


Figura 4.13: Tercera etapa de amplificación compuesta por una etapa de ganancia de 60 dB con realimentación de continua

[6] El diseño completo del filtro se puede ver en el anexo D.3.

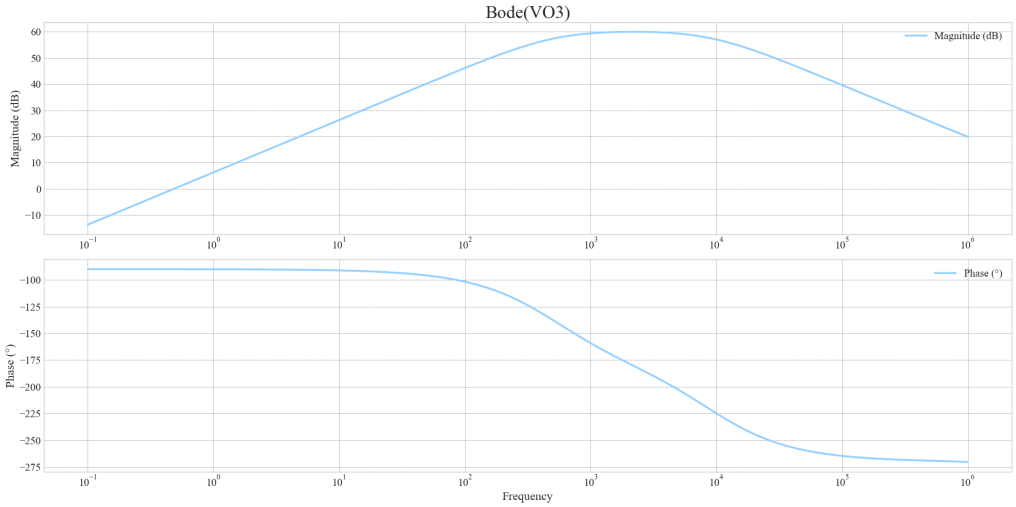


Figura 4.14: Bode simulado para la etapa de amplificación de 60 dB.

Filtro pasa banda discreto

La salida de la tercera etapa de amplificación pasa por un filtro pasa-banda de segundo orden como muestra la Fig. 4.15, de frecuencias de corte 250 Hz y 5,3 kHz correspondiente a la banda del ECAP de 300 Hz y 5 kHz.

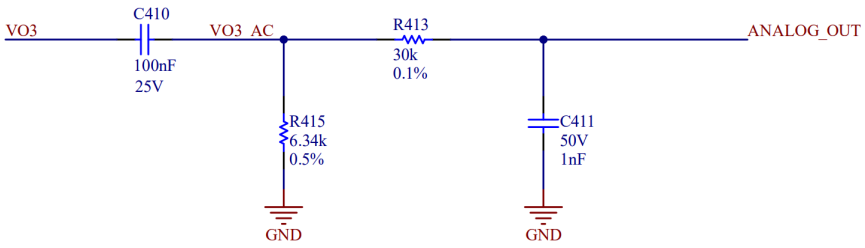


Figura 4.15: Filtro pasa-banda de salida del HW

Con este filtro y la tercera etapa de amplificación se tiene con un filtro pasa-banda con las siguientes características extraídas de la simulación de LTSpice:

- Ganancia en banda pasante de 59,3 dB.
- Frecuencia de corte inferior $f_{p1} = 508$ Hz.
- Frecuencia de corte superior $f_{p2} = 5{,}13$ kHz.

4.2.5. Adquisición de Datos

En este bloque se encuentra el ADC y DAC externo seleccionados con los circuitos de acondicionamiento necesarios. En la Fig. 4.16 se muestra el DAC

AD5443 [24] de Analog Devices de 12-bits, alto ancho de banda y con interfaz SPI que puede llegar a velocidades de reloj de 50 MHz al igual que el MCU elegido.

El DAC utiliza una referencia de precisión de 3,3 V, y como muestra el diagrama de la Fig. 4.17, implementa internamente una red de resistencias $R-2R$ inversora estándar, donde cada R tienen valor nominal de 10 kΩ y se exponen tres terminales de ella con las cuales se pueden establecer diferentes configuraciones a la salida.

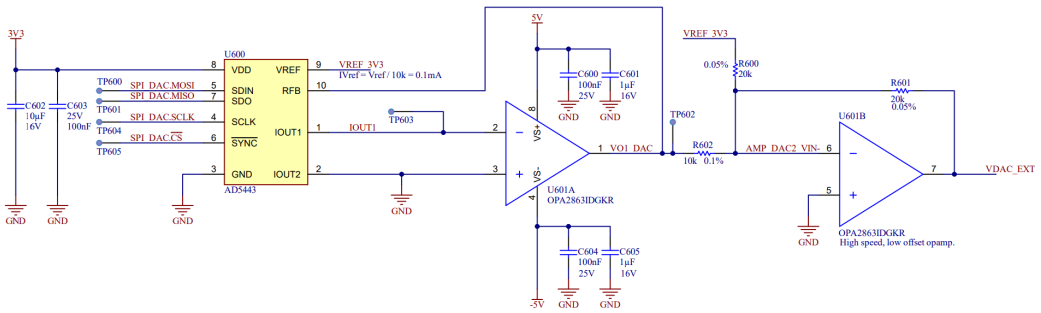


Figura 4.16: DAC externo utilizado con el circuito de acondicionamiento necesario para generar un salida bipolar

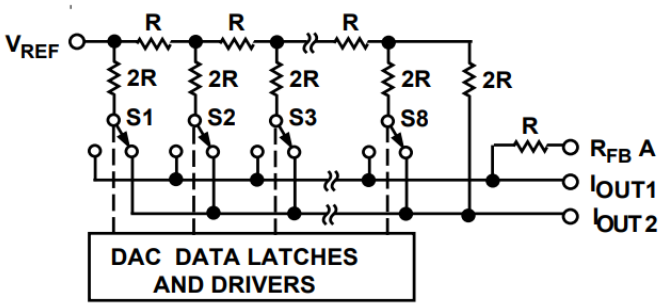


Figura 4.17: Red de resistencias internas $R-2R$ inversora implementada por el DAC AD5443 donde el valor de salida configurado por medio de SPI activa o desactiva cada una de los *switches* S_1 a S_{12}. [*Imagen extraída de la datasheet del AD5443*]

Utilizando dos amplificadores operacionales OPA2863 [25],de Texas Intruments, se establece una configuración doble inversora que permite obtener una salida bipolar entre ±3,3 V. El primer amplificador, el U_{601A} como se muestra en la Fig.4.16, utiliza la red de resistencia del DAC para establecer un inversor, cuya ganancia depende de la palabra digital configurada en el DAC[7], y con entrada el valor de la referencia. De esta manera, la salida V_{O1_DAC} esta dada por:

$$V_{O1\_DAC} = -V_{REV\_3V3} \times \frac{D}{2^n} \tag{4.8}$$

[7]La palabra digital cambia el estado de las llaves internas, generando un diferencia en las proporciones de las resistencias conectadas al amplificador

Con D el valor de la palabra digital y $n = 12$. Luego, el siguiente amplificador operacional, el U_{601B} , presenta un nuevo inversor de ganancia fija 2 V/V, configurada a partir de resistencias de precisión para evitar un error de ganancia grande. Además, se configura la continua de la salida del DAC en 0 V. De esta manera:

$$V_{DAC\_EXT} = V_{REV\_3V3} \times \frac{D}{2^{n-1}} - V_{REV\_3V3} \tag{4.9}$$

Con esta configuración se pierde un bit de precisión pero se obtiene una salida bipolar del rango mencionado previamente. El operacional OPA2863 fue seleccionado para que sean de alta velocidad para evitar un tiempo de levantamiento lento de la señal producida por el DAC a la salida y de bajo offset para asegurar que la DC de la salida esta lo más cercana a cero posible.

Por otra parte, en la figura 4.18 se puede observar el circuito para el ADC externo basado en el MAX1247 [26], de Analog Devices, que cuenta con la misma referencia de 3,3 V que el DAC. El ADC puede funcionar tanto en modo monopolar o bipolar dependiendo de la configuración realizada por software. Cuando se encuentra en modo bipolar el rango de entrada es ±1,65 V.

La INL y DNL del ADC son de como máximo 1 LSB y por ende el uso del mismo debería estar acompañado de descartar el bit menos significativo pues no es confiable.

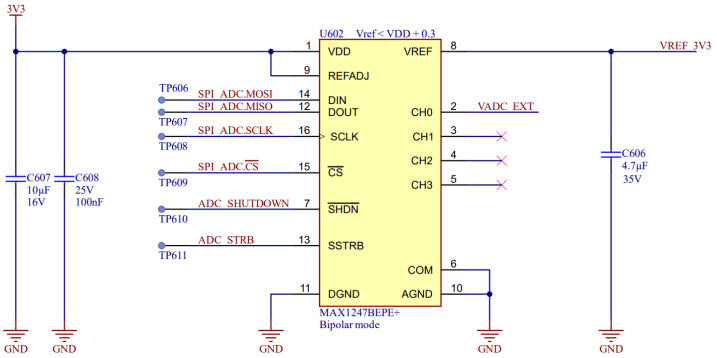


Figura 4.18: Circuito implementado para el ADC externo MAX1247

4.2.6. Power Distribution

La generación o configuración de todas las fuentes de alimentación se da en este bloque, entre las cuales se tienen las fuentes de 5 V, −5 V y la referencia de precisión de 3,3 V que se ha usado a lo largo de los bloques descritos anteriormente. Esta se implementa usando el NCP51460 [27] de ON Semiconductors [28] como muestra la Fig. 4.19. La referencia puede proveer 20 mA máximos de corriente, variando menos de 10 mV para el caso de máxima carga.

La fuente de 5 V es provista directamente desde la placa NUCLEO-L552ZE-Q a la cual el *shield* está conectado y que, a su vez, provienen del cable USB con el cual se conecta a la PC. A pesar de esto, el *shield* cuenta con la capacidad de

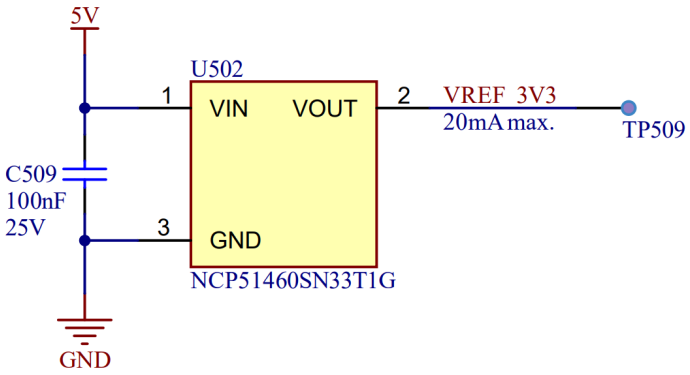


Figura 4.19: Circuito implementado para la referencia de 3,3 V usando el NCP51460

utilizar fuentes de alimentación externas por medio del uso de jumpers y el soldado de resistencias de 0 Ω como muestra la Fig.4.20.

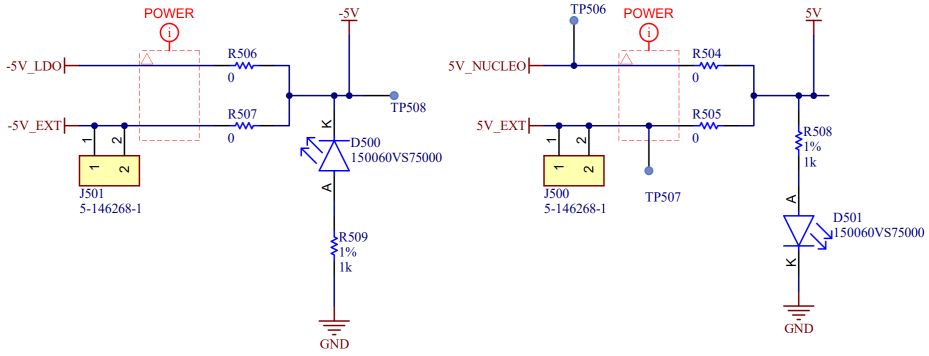


Figura 4.20: Selección de la fuentes de alimentación de ±5 V. Cuando se desea usar una fuente o la otra, la resistencia de 0 Ω correspondiente debe ser soldada mientras que la opuesta debe ser desoldada.

Para la fuente de −5 V se diseñó una configuración basada en un convertidor DC/DC más un LDO negativo para poder generar a partir de los 5 V una fuente de bajo ruido y alta estabilidad de −5 V. El uso de tanto un DC/DC y un LDO se debe al poco rechazo a la variación de las fuentes con el que cuentan algunos de los amplificadores utilizados, en particular el INA AD8221 como fue descrito en la Secc.4.2.3. Esto es especialmente perjudicial debido a que cualquier variación de alta frecuencia que se vea transmitida a la salida, representa una señal que no puede ser cancelada por el algoritmo y puede llevar a la saturación en la etapa de amplificación.

El DC/DC LT3462 [29] de Analog Devices genera con alta eficiencia una fuente de −6 V a partir de 5 V. Luego, con un LDO negativo de alto PSRR, el TPS72301 [30], se obtiene la fuente de −5 V, donde el ruido generado por las conmutaciones del DC/DC son filtradas por el PSRR del LDO. Lo antes descrito se puede observar en la Fig.4.21 y la Fig.4.22.

Una discusión de la selección de cada uno de estos integrados así como el diseño de los componentes discretos asociados puede encontrarse en el Anexo.D.5.1 y el Anexo.D.5.2.

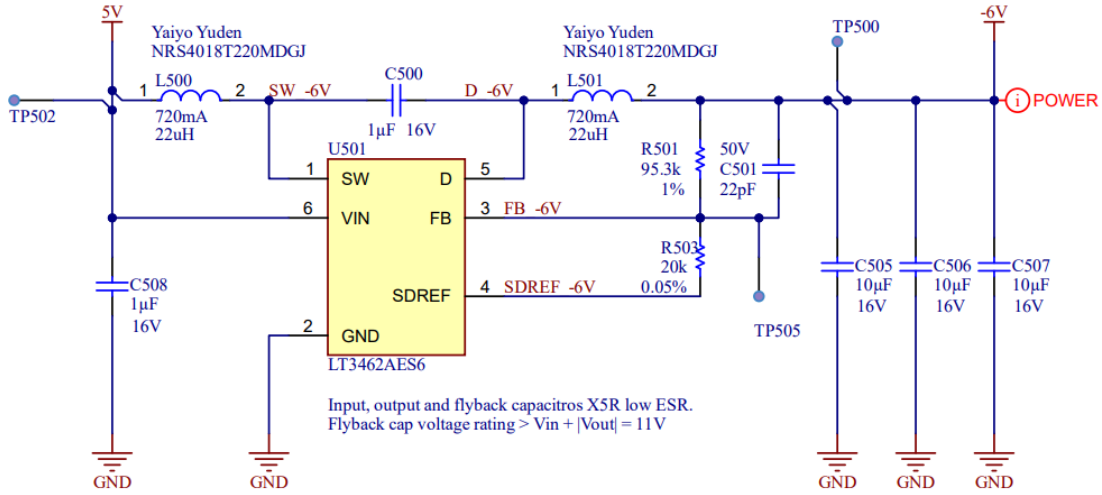


Figura 4.21: Circuito implementado para la generación de $-6\,\mathrm{V}$ a partir de la fuente de $5\,\mathrm{V}$ a partir de un DC/DC en configuración SEPIC

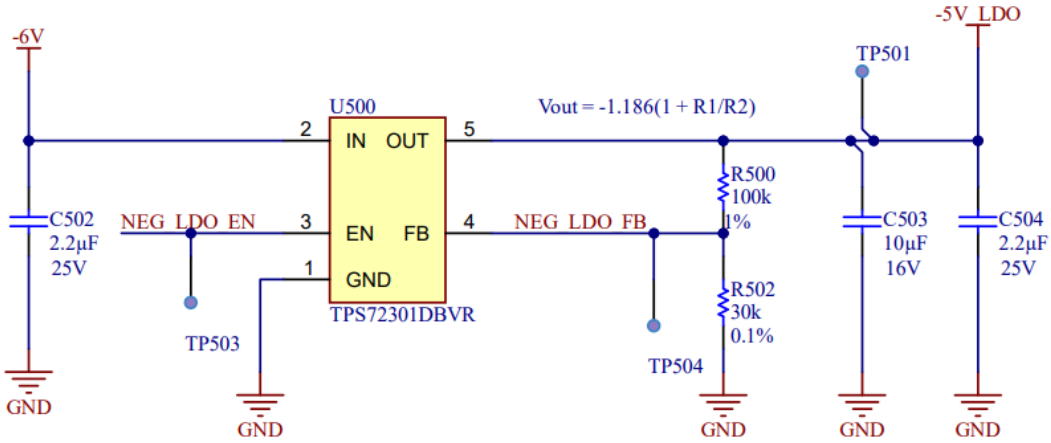


Figura 4.22: Circuito implementado para la generación de $-5\,\mathrm{V}$ a partir de la fuente de $-6\,\mathrm{V}$ utilizando un LDO negativo

El DC/DC es un Cuk converter de frecuencia fija de conmutación de $2{,}7\,\mathrm{MHz}$

cuyos componentes discretos fueron diseñados siguiendo la *Nota de Aplicación* [31]. En el diseño se tuvieron en cuenta diferentes recomendaciones realizadas por el fabricante del IC. Como por ejemplo asegurar que la corriente de pico de entrada cuando el DC/DC es por primera vez encendido no supere el límite máximo de 1,5 A. Este límite esta dado por el condensador de flyback y los inductores, el valor máximo de esta corriente calculado es 0,7 A.

El diseño realizado fue verificado por simulaciones de LTSpice donde se le exigió al conjunto DC/DC más LDO corrientes similares a las esperadas durante la cancelación de artefactos, del orden de decenas de mA y se obtuvieron variaciones en la salida del LDO de 300 µV en donde no se observaba ruido significante proveniente de la conmutación del DC/DC.

4.2.7. Interfaces

En este bloque se pueden encontrar todos los conectores compatibles con la placa de desarrollo NUCLEO-L552ZE-Q. En total, 17 pines del MCU fueron utilizados para diferentes propósitos, tres buses SPI independientes fueron usados para controlar el potenciómetro digital, ADC y DAC externos y diferentes pines utilizados como GPIOs para el control o detección de diferentes señales digitales. Se destacan los pines de MCU utilizados para estos últimos que se pueden observar en la Tab.4.1.

Señal	Función	Pin MCU	Periférico STM32L5
VADC_NUCLEO	Entrada ADC interno del MCU para la adquisición de la plantilla.	PB1	Canal 16 ADC 1
REF_SUB	Entrada ADC interno del MCU para la calibración inicial de continua del algoritmo.	PA5	Canal 10 ADC 1
VDAC_NUCLEO	Salida DAC interno del MCU para la reproducción de la plantilla.	PA4	Canal 1 DAC 1
ESTIM_SENSE	GPIO configurado como INPUT para la detección del comienzo de la estimulación por medio de la salida del Schmitt Trigger.	PB3	GPIO puerto B pin 3
NEG_LDO_EN	GPIO configurado como OUTPUT. Señal de control para la habilitación del LDO negativo para la generación de la fuente de −5 V	PC8	GPIO puerto C pin 8

Tabla 4.1: Pin analógicos del MCU y sus funciones utilizados en la placa de adaptación diseñada.

4.3. Análisis de Ruido

Debido a la baja amplitud de la señales neurales que se desean detectar, es sumamente importante tener en cuenta el ruido introducido por lo componentes discretos y activos a lo largo de la cadena de amplificación. Tal que, luego de la última etapa de amplificación, el ruido esté una década por debajo de la amplitud mínima del ECAP que se desea detectar.

Se realizó un análisis del ruido introducido por los componentes que se encuentran en el AFE y el ABE, es decir, el INA, los amplificadores de las etapas de substracción y de ganancia 60 dB, así como todo componentes discretos.

Para realizar los cálculos, se consideró que el ancho de banda final es entre la frecuencias de 300 Hz y 5 kHz, determinada a partir de los filtros implementados a lo largo de la cadena, temperatura ambiente de 25 °C y que el ruido introducido por los operacionales se considera ruido blanco de densidad espectral igual al valor nominal dado en la hojas de datos de los amplificadores a la frecuencia de 1 kHz. Esta última consideración se debe a que el ruido $1/f$ característico de los amplificadores decrece rápidamente hasta ser mucho menor que el ruido blanco base en las frecuencias de interés, como puede observarse en la Fig.4.23 para el INA AD8221 utilizado.

Con estas consideraciones se calculó etapa por etapa la densidad espectral del ruido a la salida, obteniéndose para la etapa final S_{nVOut} =7,69 $\times$ 10^{-12} V^2/Hz , lo cual es equivalente a una ruido RMS de V_{nVOut} = 239 μV_{rms}. Con esto en cuenta, una ECAP de amplitud mínima de 2 µV, tras ser amplificada alcanzaría una amplitud de 2 mV, por lo cual, el ruido RMS se encuentra una década por abajo de la amplitud mínima de salida de las señales de interés.

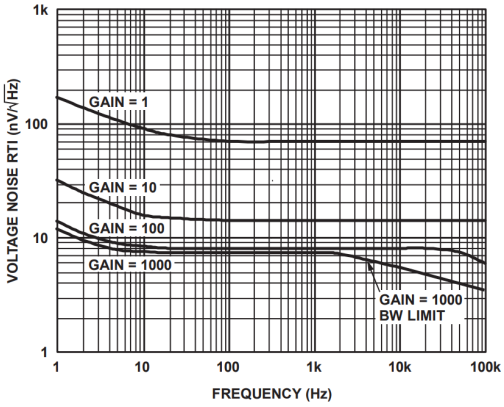


Figura 4.23: Ruido en voltaje referido contra la entrada del INA AD8221. [*Imagen extraída de la datasheet del AD8221*]

4.3.1. Diseño de la PCB

Todo el trabajo de desarrollo de los esquemáticos como de la PCB del CANE STM32-L552ZE Shield se realizó en el programa Altium Designer [32]. El shield

consiste en una PCB de 6 capas cuyo modelo 3D puede observarse en la Fig. 4.24.

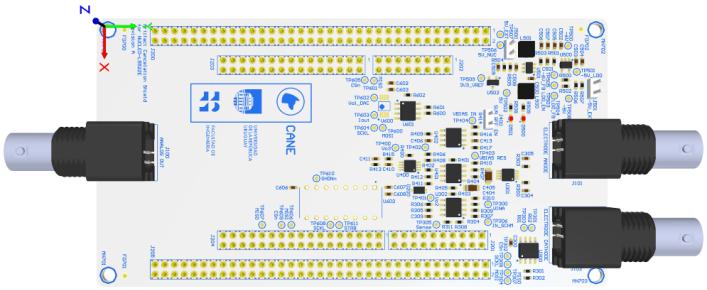


Figura 4.24: Modelo 3D de la PCB CANE STM32-L552ZE Shield diseñada .

La forma de la PCB fue definida a partir del la forma del kit de desarrollo del STM32-L552ZE, teniendo las mismas dimensiones que este último. El stackup usado se puede ver en la Fig 4.25 que corresponde a un stackup estándar de 6 capas, 1.6mm de espesor con vias through hole. La elección de utilizar esta topología surgió de priorizar la integridad de la señales analógicas presentes en el circuito. Utilizar un stackup de 6 capas permite desacoplar completamente el ruteo de las señales de interés con las señales digitales o la distribución de la alimentación que pueden introducir ruido a partir de acoples generados por las mismas.

Además, usar 6 capas permite que las señales analógicas de *Top Layer* cuenten con un plano de referencia a tierra independiente que el de las señales digitales y de la alimentación. Esto se puede ver en la Fig. 4.25 , donde la capa *L2 - GND* cumple el rol antes dicho mientras que *L5 - GND* es la referencia para las capa *L4 - Power* donde se distribuye toda la alimentación y *Bottom Layer* donde se realizó el ruteo de las señales digitales como puede ser las de SPI.

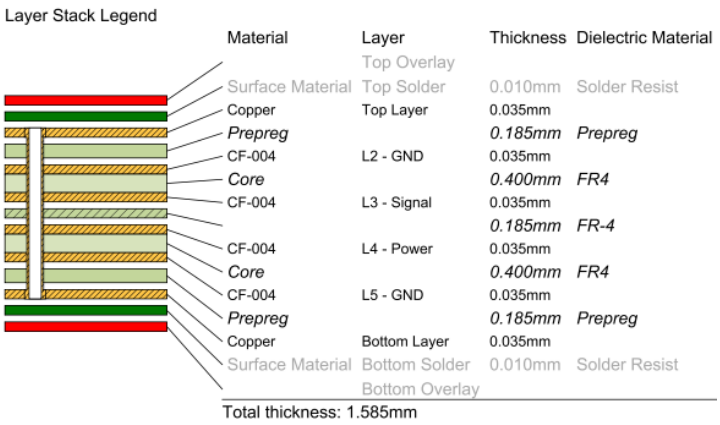


Figura 4.25: Stakcup utilizado para la PCB desarrollada

La desventaja de usar 6 capas proviene del costo más elevado de manufactura de este tipo de PCB. Para mitigar esto, se tuvieron consideraciones de diseño que disminuyeran estos costos como el uso de componentes sin empaquetados *quad flat*

no-lead (QFN) o *ball grid array* (BGA) y anchos mínimos para las pistas y vías que cumplieran con los requisitos de la tecnología de manufactura más económica. Además, se consiguió un descuento de primera compra en la manufactura de las PCB con el fabricante elegido que es AllPCB [33], dejando el costo de fabricación al mismo precio de una PCB de cuatro capas, el cual era el stackup mínimo considerado como necesario para la aplicación.

El ensamblado de las placas fue realizado por los integrantes del grupo a partir de los componentes comprados en Mouser y se ensamblaron tres placan en total. Todos los circuitos fueron soldados excepto los correspondientes al bloque de adquisición de datos, es decir, el DAC y ADC externos.

4.4. Verificación y Mediciones

En esta sección se expondrán las mediciones realizadas para caracterizar el CANE STM32-L552ZE Shield. En particular, son de interés dos series de mediciones, primero la transferencia de cada uno de los componentes de la cadena de amplificación y filtrado de nuestro sistema, así como la transferencia total. Segundo, se busca caracterizar el ruido presente en el sistema.

4.4.1. Corroboración del Funcionamiento

Tras realizarse el ensamblado del shield, la primera tarea a realizarse fue corroborar el funcionamiento de todos los circuitos diseñados. Este paso no consiste en una caracterización completa sino busca identificar cualquier falla grave en el diseño .

Se utilizó el AD2 controlado externamente para alimentar el shield, generar todas las señales digitales de control o entradas analógicas. Se corroboró la operación correcta del bloque *Power delivery* con la generación de las fuentes de $-6\,\mathrm{V}$, $-5\,\mathrm{V}$ y la referencia de precisión de $3{,}3\,\mathrm{V}$.

También se corroboró el funcionamiento del amplificador de instrumentación, Schmitt Trigger, el restador y la etapa de amplificación de $60\,\mathrm{dB}$. Al realizar pruebas sobre el funcionamiento del mecanismo de ganancia variable, se encontró una falla en el mismo que impedía el funcionamiento correcto del conjunto INA + potenciómetro digital. El error se detallará en profundidad en las secciones a continuación. Es importante notar que el INA por separado, sin estar conectado al potenciómetro digital por sus terminales 2 y 3 como se puede ver en la Fig. 4.3, operaba correctamente.

En resumen, luego del ensamblado del shield, se corroboró el funcionamiento esperado de todos los circuitos diseñados, encontrándose solamente una falla al intentar modificar la ganancia del INA a partir del potenciómetro digital.

4.4.2. Caracterización del Hardware Desarrollado

En esta sección se recorrerán todos los circuitos diseñadnos, caracterizando su funcionamiento a partir de los parámetros que son de interés para nuestra

aplicación.

Power Delivery

Como ya se discutió en las secciones de diseño, la estabilidad de las fuentes puede afectar directamente el desempeño de nuestro sistema al contar el mismo con amplificadores de bajo PSRR. Por ende es de interés caracterizar cual es la variación de las mismas cuando el sistema se encuentra en funcionamiento.

Ruido de las fuentes La variación de las fuente se midió bajo las siguientes condiciones:

- Alimentación de 5 V proveniente de USB.
- LDO negativo habilitado.
- Salida del LDO alimentando toda la electrónica que compone al AFE y al ABE.
- Entrada del INA con una onda SA generada a partir del AD2.
- MCU reproduciendo a partir de su DAC una plantilla de la SA entrante al INA.

En este contexto, se relevaron el valor promedio, voltaje V_{rms} para la salida del DC/DC, LDO negativo, la referencia de precisión de 3,3 V y la alimentación de 5 V. Los resultados que fueron obtenidos usando el AD2 se pueden ver en la Tab. 4.2. Es importante mencionar que previo a estas mediciones se relevó el ruido ambiente al medir tierra donde se obtuvo una variación de 0,321 µVrms.

Fuente	Promedio (V)	V_{rms} (mV_{rms})	V_{min} (V)	V_{max} (V)
-6 V	-6.05 ± 0.01	1.4 ± 0.1	-6.07	-6.04
-5 V	-4.97 ± 0.010	0.8± 0.1	4.96	4.98
REF 3,3 V	3.27 ± 0.01	0.4 ± 0.1	3.28	3.26
5 V	4.86 ± 0.010	5.0 ± 0.100	4.9	4.8

Tabla 4.2: Estabilidad de las fuentes generadas en el shield

La fuente de 5 V cuenta con mucho ruido al ser la fuente principal del sistema, así como provenir de un cable USB. Se puede observar como el LDO realiza su trabajo al reducir el nivel de ruido proveniente de la fuente de -6 V al pasar de tener 1,4mV_{rms} a 0,84mV_{rms}. Por último, la referencia de 3,3 V es la fuente de menor ruido.

Analog Front End

A continuacion se presentará la verificacion de funcionamiento del AFE.

INA: Las características más importante de la etapa de detección, compuesta por el filtro pasa bajos y el INA, consisten en su transferencia, y su rango de excursión de entrada y salida. Esto permite determinar cual es el conjunto completo de ondas posibles a ser manejadas por el AFE sin ser distorsionadas perdiendo información de la señal neural incidente.

La transferencia del INA se puede observar en la Fig. 4.26 relevada con el AD2 utilizando una entrada diferencial colocada en los conectores BNC. En banda pasante, se obtiene una ganancia de 0 dB y se obtienen como frecuencia de corte pasa altos de $35{,}3 \pm 0{,}1 Hz$ y pasa bajos de $964 \pm 1 KHz$. Ambos valores son acorde a lo esperado, según el diseño de la Secc.4.2.3.

Durante esta medición, el potenciómetro digital se encontraba desconectado.

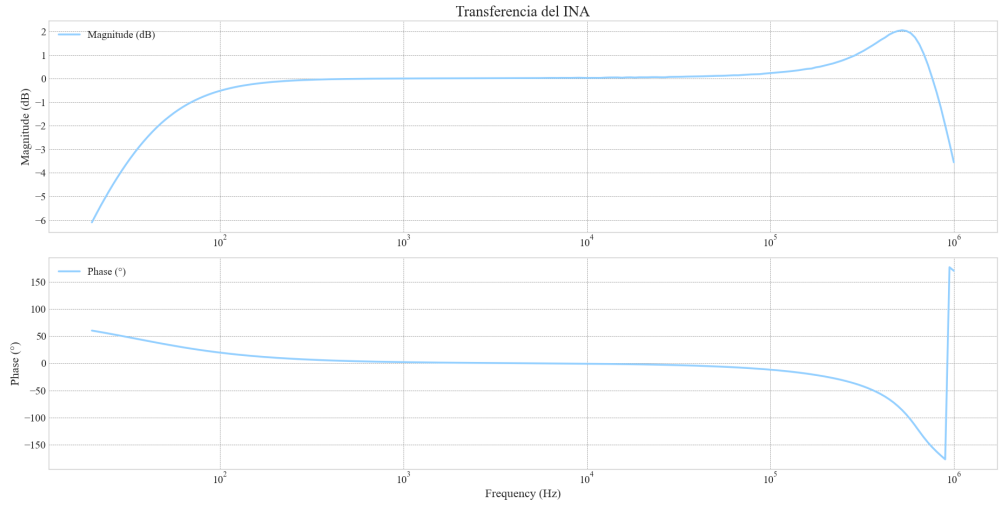


Figura 4.26: Transferencia del INA relevada.

Luego, el ICMR y el OSW del conjunto filtro discreto + INA fueron relevados. Para obtener estos valores se utilizo la distorsión armónica total (THD) de una sinusoide a la salida del INA. En el caso del OSW, se generó una sinusoide en modo diferencial a la entrada, aumentando su amplitud hasta hasta obtener una distorsión del 1 % a la salida. Para dicho valor, se definen los rangos máximos de excursión. La THD fue relevada a partir del espectrómetro con el que cuenta el AD2.

En el caso del ICMR, la medida fue relevada de manera análoga. En este caso, a cada una de las entradas del INA se genera una sinusoide de igual frecuencia y fase pero distinta amplitud. En la entrada positiva del INA se genera la onda $\alpha . \sin \omega t$ mientras que en la entrada negativa, la onda $\beta . \sin \omega t$. Con $\alpha \neq \beta$ y ω en la banda pasante. De esta manera, el modo diferencial y común del INA quedan definidos como:

$$V_D = (\alpha - \beta)\sin\omega t \tag{4.10}$$

$$V_{CM} = \frac{(\alpha + \beta)\sin\omega t}{2} \tag{4.11}$$

Se pueden variar α y β tal que la amplitud del modo diferencial se mantenga constante mientras que la amplitud del modo común se vea modificada. El ICMR se relevó aumentando la amplitud del modo común hasta observar una THD del 1 % de la sinusoide a la salida.

Los resultados se pueden ver en la Tab. 4.3.

Medida	Límite inferior (V)	Limite superior (V)
ICMR	-3.32 ± 0.01	-3.33 ± 0.01
OSW	-2.05 ± 0.01	2.05 ± 0.01

Tabla 4.3: Valores relevados para el ICMR y OSW del INA + filtro analógico a la entrada.

Potenciómetro digital Para validar el correcto funcionamiento del potenciómetro digital, se desarrollo un FW de prueba en el cual se variaba la resistencia vista entre las terminales $P0W$ y $P0B$ del integrado a partir de la comunicación SPI. Las pruebas se realizaron con el potenciómetro desconectado del INA. En la Tab. 4.4 se pueden observar los valores medidos de resistencia comparados con el valor esperado dado por la configuración en el potenciómetro digital a partir de la ecuación (4.2) y el valor cargado en el registro $00h$ del potenciometro.

Ganancia conf. (V/V)	Posición Wiper	R_{BW} conf. ($k\Omega$)	R_{BW} ($k\Omega$)
1	$P0W$ y $P0B$ desconectados	∞	Fuera de rango
2	126	49.14	47.0 ± 0.1
5	31	12.11	11.64 ± 0.01
10	14	5.46	5.37 ± 0.01

Tabla 4.4: Valores relevados de la resistencia entre las terminales $P0W$ y $P0B$ del potenciómetro digital MCP4131-503E/SN para diferentes valores configurados.

Teniendo en cuenta el error de cuantización del potenciómetro digital de 390 Ω, la tolerancia de la resistencia R_{AB} de 20 %, la *Integral Non-linearity* y *Differential Non-linearity* de 0,25 %LSB cada una y una resistencia máxima del *wiper* de 300 Ω, las diferencias entre los valores medidos y los configurados están dentro de lo esperado.

Mecanismo de ganancia variable Si bien el funcionamiento del potenciometro digital anteriormente relevado era el correcto, al conectar las terminales 2 y 3 del INA con las $P0W$ y $P0B$ del potenciómetro respectivamente, el funcionamiento del amplificador se vio afectado. Por ejemplo, para la configuración de ganancia unitaria [8], se obtuvo ante una entrada sinusoidal diferencial de amplitud 100 mV la salida del INA observada en la Fig. 4.27. El INA satura, al amplificar la onda entrante cuando en cambio debería tener ganancia de 1 V/V.

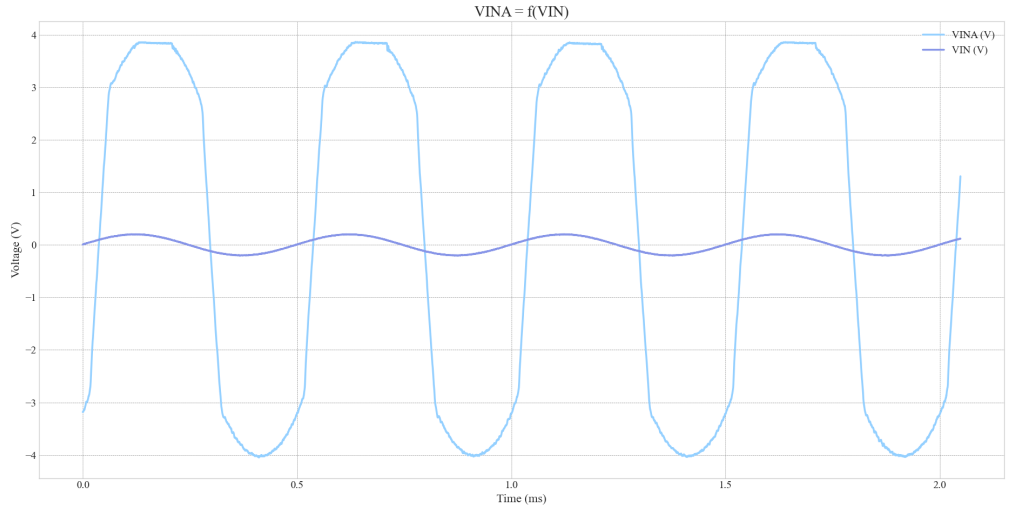


Figura 4.27: Salida INA con ganancia configurada a partir del potenciómetro digital en 1 V/V y entrada diferencial de amplitud 100 mV

Tras analizar detenidamente el circuito y las hojas de datos de los componentes, se encontró el error de diseño discutido previamente en las Secc.4.2.3.

Una posible solución de este problema, consiste en colocar la continua a la entrada del INA en un voltaje positivo distinto de 0 a partir de un divisor resistivo entre 3,3 V y GND. Debido a que estamos trabajando con señales chicas del orden de los cientos de mV, se puede realizar este cambio sin afectar el funcionamiento del INA. La desventaja se encuentra en que reduciría el ICMR del amplificador al encontrarse la entrada en modo común del mismo más cercana a su límite superior.

Una posible implementación de este cambio se puede observar en la Fig. 4.28 donde se colocó una nueva resistencia de 100 kΩ conectada a 3,3 V en las entradas del INA. Con este cambio, vistos los resultados de ICMR medidos en la Tab. 4.3 , queda definida la amplitud máxima del modo común del artefacto de estimulación de entrada en aproximadamente 1,67 V. Valor que se encuentra por arriba del esperado dado el modelo discutido en la Secc. 2.2.3 de como máximo 1,47 Vpp.

[8] El potenciómetro se encuentra configurado con las terminales $P0W$ y $P0B$ desconectadas de la red interna de resistencias

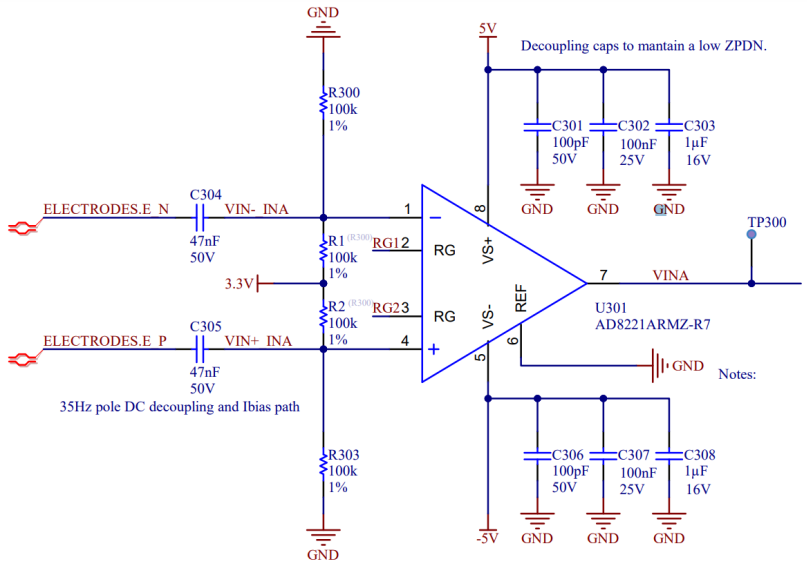


Figura 4.28: Posible solución al error de diseño encontrado en el mecanismo de ganancia variable.

Schmitt - Trigger Para el caso del Schmitt - Trigger se relevó la transferencia del pre-amplificador y los tiempos de levantamiento correspondientes a los artefactos SA de menor y mayor amplitud, 70 mV y 140 mV respectivamente. Notar que el tiempo de levantamiento corresponde solo al tiempo del propio Schmitt Trigger y por ende, no se tuvieron en cuenta el delay introducido por el INA. Para relevar estos datos, se generó una onda cuadrada de la amplitud correspondiente en el nodo V_{INA} de la Fig. 4.8 y se midieron los tiempos entre el inicio de dicha onda y el punto en el cual la salida del Schmitt Trigger llega al 90 % de su valor final. Los resultados se pueden ver en la Fig. 4.29 y la Tab. 4.5 .

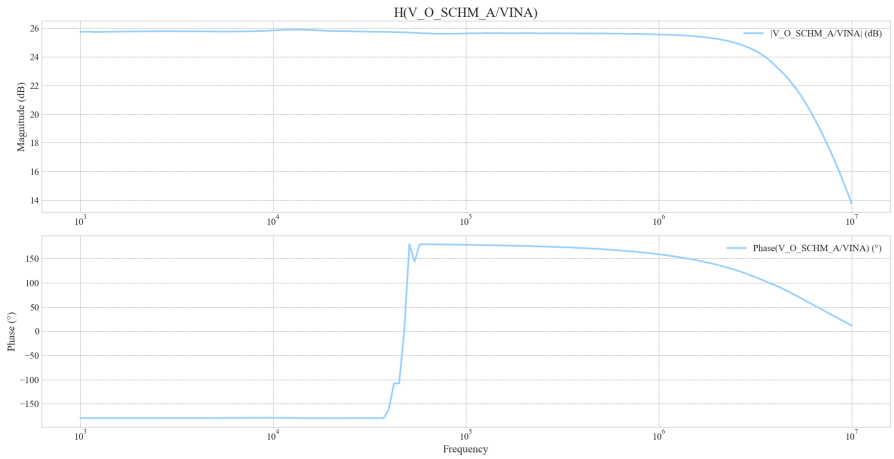


Figura 4.29: Transferencia del pre-amplificador antes del Schmitt Trigger.

Amplitud SA (mV)	Rise Time pos. (ns)	Rise Time neg (ns)
Mediciones HW		
70	101.3 ± 0.1	128.5 ± 0.1
120	87.7 ± 0.1	93.5 ± 0.1
Simulación		
70	85.8	116.6
120	66.1	68.4

Tabla 4.5: Caracterización del tiempo de levantamiento para las señales SA a la entrada del sistema de menor y mayor amplitud. También se pueden apreciar los valores obtenidos de la simulación para los mismos casos.

A partir de estos valores se concluye que se cuenta con un máximo tiempo de levantamiento de 128,5 ns que afecta al desempeño del sistema como un delay intrínseco en la sincronización entre la generación de la plantilla y el artefacto de estimulación. 128,5 ns se encuentra una década por debajo del máximo delay admisible que fue determinado a partir de la simulación (ver Cap.3) en 5 µs.

También es importante notar que este delay solo estará presente en los casos en que la señal digital generada por el Schmitt-Trigger sea usada para la sincronización.

Analog Back End

Segunda etapa de amplificación Para caracterizar el restador se realizaron tres bodes, dos para la transferencia desde las entradas hacia la salida y un último para el CMRR del amplificador en configuración diferencial. Las gráficas se pueden observar en la Fig.4.30 y la Fig.4.31 respectivamente.

El resultado mostrado en la primera gráfica es el esperado. Contamos con una diferencia de fase de 180° y una ganancia muy parecida, sobretodo en el ancho de banda de la SA de 16 kHz. En la Tab. 4.6 se pueden apreciar la ganancia y frecuencias de corte para ambas transferencias. Hay un error de 2 Hz entre las frecuencias de corte inferior de las dos transferencias, indicio del desapareo entre los condensadores de desacople de continua. En cambio, la diferencia de ganancias es prácticamente despreciable.

Transferencia	Frec. corte inferior (Hz)	Frec. corte superior (MHz)	Ganancia (mdB)
V_{O2}/V_{INA}	78.3 ± 0.1	1.75 ± 0.01	86 ± 1
V_{O2}/V_{DAC}	80.3 ± 0.1	2.18 ± 0.01	-23 ± 1

Tabla 4.6: Características del los Bodes relevados del restador y sus dos entradas.

A partir de la última gráfica podemos obtener el CMRR máximo en el ancho de banda de la SA (16 kHz), que corresponde a $46{,}9 \pm 0{,}1dB$. Esto resulta en que, ante dos entradas iguales en V_{INA} y del V_{DAC}, la cancelación será de como

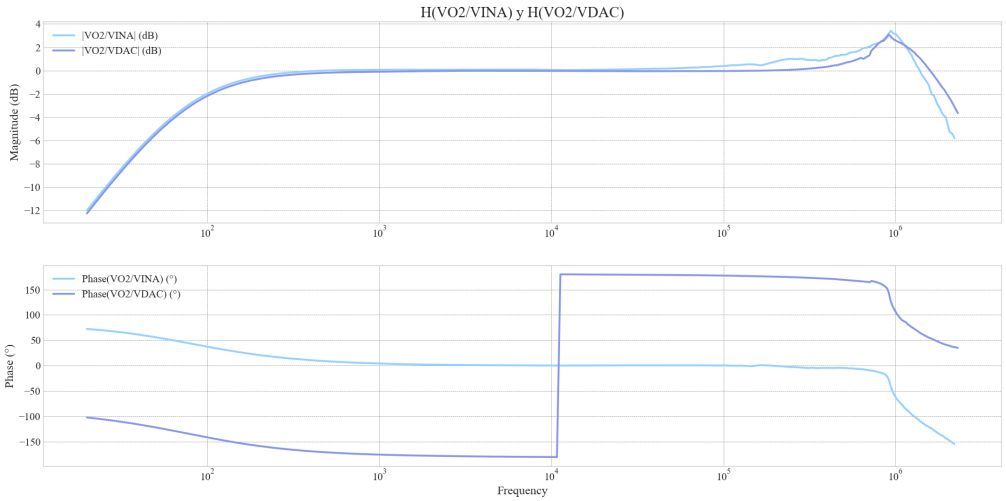


Figura 4.30: Transferencia de la salida del restador a partir de las dos entradas, V_{INA} y V_{DAC}.

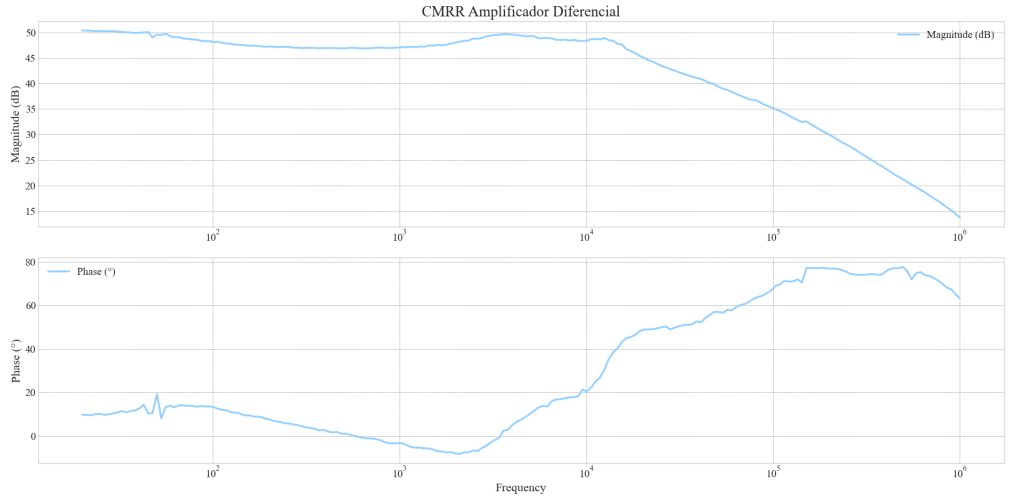


Figura 4.31: CMRR Relevado para el amplificador en configuración diferencial de la segunda etapa.

máximo 220 veces más chica que la original. Esto limita la amplitud máxima de la SA que se puede cancelar idealmente utilizando el restador implementado.

Con una ganancia de 60 dB en la tercera etapa, sabiendo que el amplificador *OPA197* es rail-to-rail y se encuentra alimentado desde ±5 V, el residuo de la resta debe ser menor a 5 mV para no saturar el amplificador. Con esto y el factor de cancelación ideal de −46,9 dB, podemos obtener la amplitud máxima ideal de una onda a la salida del INA que se pueda cancelar, correspondiente a $V_{INA,max} = 5\,\text{mV} \times 220 = 1100\,\text{mV}$.

Este límite es ideal y debe ser tratado como una cota superior del desempeño del sistema de HW+FW para la cancelación de artefactos de estimulación.

Etapa de amplificación de 60dB En la última etapa de amplificación la característica de interés es la ganancia en banda pasante. Para obtenerla se relevó el Bode de la transferencia de la tercera etapa, que se puede ver en la Fig.4.32 superpuesto con el resultado de la simulación. De la misma manera se muestran los resultados del Bode del filtro discreto en la Fig.4.33. Por último en la Tab.4.7 se puede observar la ganancia máxima en banda pasante y frecuencia de corte en los dos casos incluidos los valores obtenidos de la simulación.

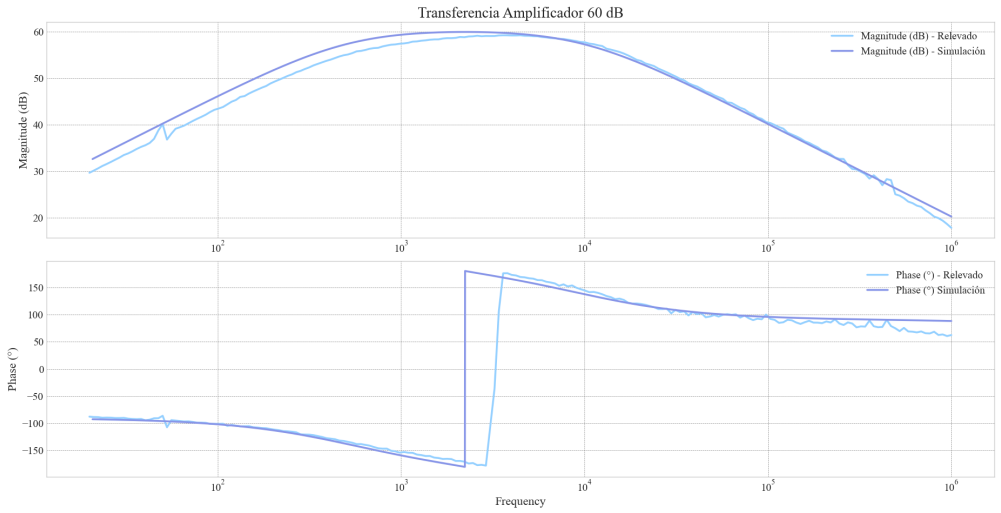


Figura 4.32: Comparación entre la transferencia relevada para el amplificador de 60 dB y la transferencia simulada

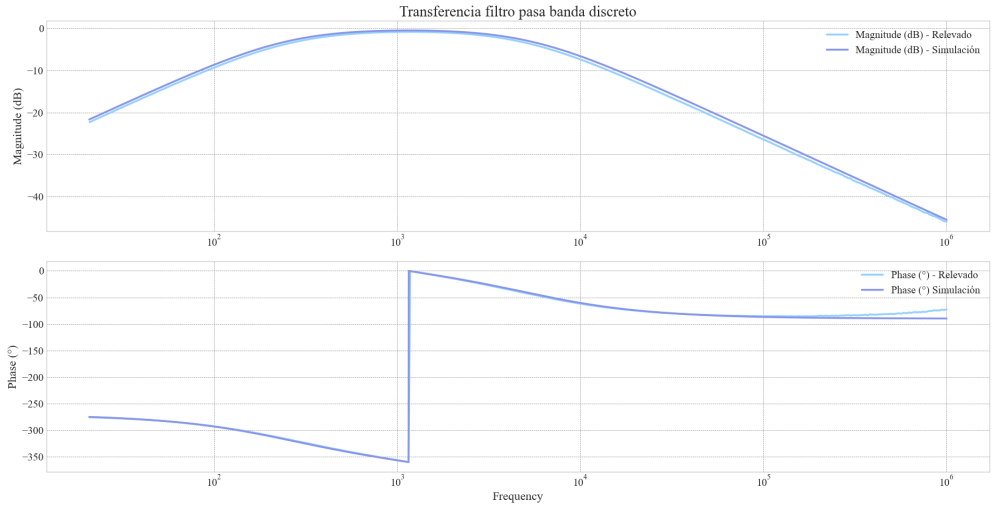


Figura 4.33: Transferencia relevada y simulada para el filtro pasa banda discreto diseñado.

Bode	Frec. corte inferior (Hz)	Frec. corte superior (kHz)	Ganancia (dB)
AMP 3 Sim.	462.0	10.8	59.9
AMP3	660.5 ± 0.1	13.6 ± 0.1	59.2 ± 0.1
BPF Sim.	230.5	5.8	0
BPF	240.9 ± 0.1	5.50 ± 0.01	-0.82

Tabla 4.7: Características del los Bodes relevados para la tercera etapa de amplificación y el filtro pasa banda digital.

4.4.3. Sistema Completo

Se relevó el Bode del sistema completo desde la entrada diferencial del INA hasta la salida. Los resultados se pueden ver en la Fig. 4.34 donde la ganancia en banda pasante es 58,9 dB con un ancho de banda de $4{,}58 \pm 0{,}01kHz$ entre $561{,}2 \pm 0{,}1Hz$ y $5{,}45 \pm 0{,}01kHz$.

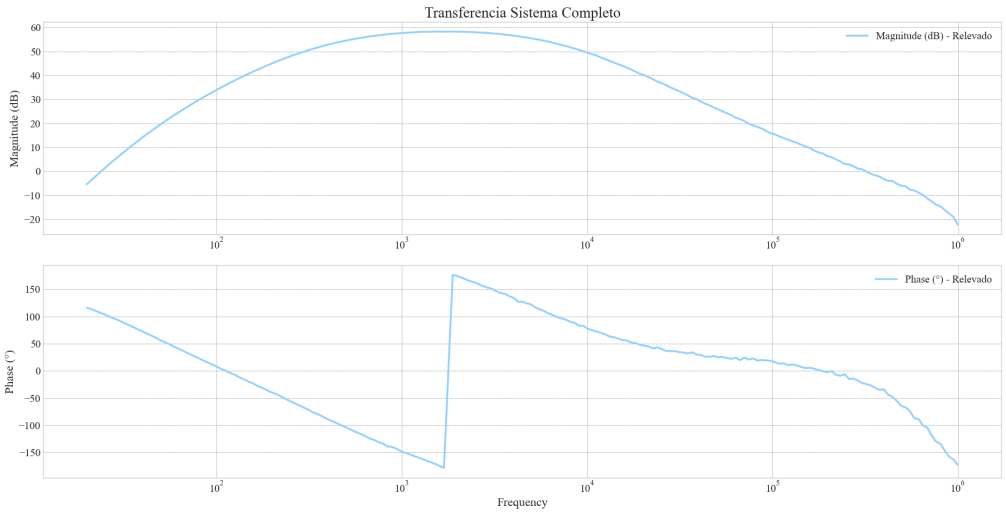


Figura 4.34: Transferencia relevada del sistema completo con entrada diferencial en los conectores de los BNC.

Por último, se relevó el ruido RMS durante un tiempo de 10 s a la salida de las tres etapas de amplificación con las entradas del INA, la salida del DAC a tierra y las fuentes prendidas. Los datos relevados se pueden ver en la Tab. 4.8

Señal	Promedio (mV)	Vrms (mV_{rms})
V_{INA}	$-1{,}0 \pm 0{,}1$	$0{,}6 \pm 0{,}1$
V_{O2}	1640 ± 1	$1{,}3 \pm 0{,}1$
V_{O3}	4 ± 1	126 ± 1
V_{OUT}	32 ± 1	$60 \pm 0{,}1$

Tabla 4.8: Caracterización del ruido obtenido a la salida de las tres etapas de amplificación.

Los valores obtenidos coinciden con lo esperado. En el caso del INA, al tener ganancia unitaria y un offset intrínseco de 165 µV es esperable un valor promedio a la salida con entradas nulas tan bajo. Para la segunda etapa, al estar la DC de la salida diseñada para ubicarse a la mitad del valor de la referencia de 3,3 V, cuyo valor medido es de 3,27 V según la Tab. 4.2. Por ende, se esperaría en V_{O2} un valor de 1,635 V, muy similar al valor relevado.

Para finalizar, como V_{O3} y la salida, cuentan con alta ganancia es esperable ver un aumento del ruido, donde se observa una variación de 100 mVrms. Esto pone un límite al ECAP mínimo que se puede llegar a detectar, pues luego de la amplificación se debe alcanzar una amplitud de la señal que sea distinguible del ruido base a la salida.

Hasta ahora, no se ha definido el criterio de decisión para la correcta adquisición de una ECAP a la salida del sistema. Parte del trabajo realizado en el Cap.7 de resultados consistirá en determinar dicho criterio. Por ahora, tener una variación de 100 mVrms a la salida imposibilita la recuperación de señales neurales de igual amplitud, lo cual equivale a un ECAP a la entrada del sistema de amplitud 100 µV. Por ende, el relevamiento del ruido base a la salida del sistema establece una cota inferior para la amplitud mínima del ECAP a detectar, correspondiente a 100 µV.

4.4.4. Resumen

A partir de los resultados relevados de esta sección se pudieron identificar dos cotas de funcionamiento para las amplitudes de las señales de interés con las cuales se va a trabajar. A partir del la medición del CMRR de la segunda etapa de amplificación se determinó una cota superior en 1100 mV para la amplitud de los artefactos de estimulación con los que se puede trabajar. Mientras que el ruido observado a la salida del sistema determina una cota inferior para la amplitud mínima del ECAP con los cuales se puede trabajar en 100 µV.

Capítulo 5

Desarrollo de Firmware

5.1. Descripción General

El objetivo de este capítulo es analizar la implementación de la solución propuesta desde el punto de vista del software embebido. Previamente es importante remarcar cuales son las tareas que debe manejar el microcontrolador para que el sistema funcione correctamente:

- Atención de eventos desencadenados por la estimulación.
- Cancelación activa de la señal de artefacto mediante un DAC.
- Adquisición de la señal de error mediante un ADC y de una plantilla inicial.
- Procesamiento de datos de error para actualizar los datos del DAC.

Considerando que la frecuencia de estimulación es de $900Hz$, se deben realizar estas tareas de manera rápida y mantener una estructura eficiente que permita una correcta interacción entre cada parte del sistema. En la Secc. 5.2 se entrará en detalle sobre como se implementó la arquitectura de firmware.

A grandes rasgos la implementación de firmware consiste en el muestreo de la salida del restador y mediante el uso de controladores PI se busca calcular una plantilla. Esta plantilla es la que será reproducida por el DAC y restada con el artefacto para su cancelación. La arquitectura del firmware se hizo como una máquina de estados la cual recibirá eventos. Estos eventos son situaciones las cuales requieren procesamiento del microcontrolador y que generarán una reacción en la máquina de estados. Por último el firmware embebido provee la interfaz para la coordinación y comunicación entre la etapa de software y la etapa de HW+FW.

Para información más detallada sobre la implementación de FW dirigirse al doxygen del mismo.

5.1.1. Algoritmo de Cancelación de Artefactos

Una de las tareas más importantes del firmware es la correspondiente al cálculo y actualización de la plantilla a reproducir mediante el DAC. Lo que se desea es

tener un sistema en lazo cerrado que sea capaz de cancelar el artefacto en tiempo real.

El lazo de control que se forma se puede ver en la Fig.5.1, en donde la entrada de hardware representa el artefacto de la SA que se desea cancelar. El microcontrolador actúa como una realimentación negativa y unitaria, es decir este será el encargado de implementar el controlador correspondiente, que como se puede ver se utiliza un control del tipo PI (proporcional-integral). El propósito de este controlador será actualizar la plantilla hasta que está se a lo más parecida al artefacto de estimulación. Cuando esto se alcanza, tras efectuarse la resta, se obtiene una salida constante en la segunda etapa.

El ADC y el DAC del microcontrolador son la interfaz para interactuar con la planta, y el resultado del lazo de control es lo que en el proyecto se llamó *residuo del artefacto*, como se mencionó en capítulos anteriores, este residuo debe ser lo suficientemente chico para no saturar el último bloque del diagrama que es un amplificador de $60dB$.

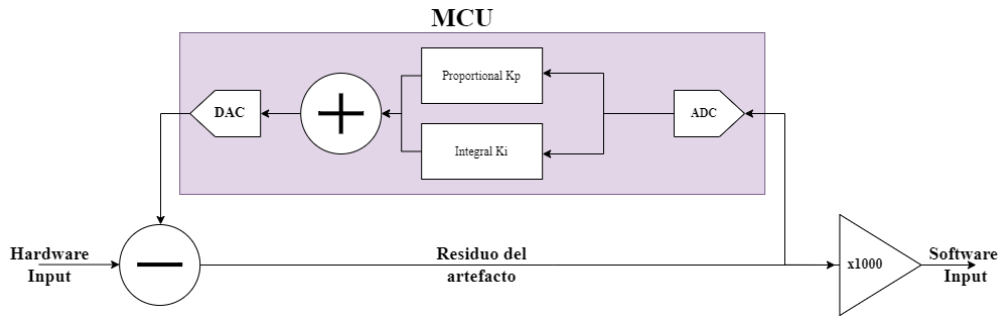


Figura 5.1: Diagrama de bloques del algoritmo de cancelación de artefactos.

El rol del microcontrolador en el lazo de control es muy importante, ya que para lograr el objetivo de cancelar en tiempo real señales que varían rápidamente como lo es la SA, un controlador convencional no es suficiente. La arquitectura propuesta se basa en que la señal a cancelar es periódica en el tiempo y ocurre cada vez que hay una estimulación, lo que permite guardar una plantilla de la señal que se debe reproducir para cancelar el artefacto. Al reproducir esta plantilla se obtendrá una señal de error resultante de la resta, y con esta nueva información se puede iterar sobre la plantilla inicial para lograr el objetivo planteado. De esta manera se distinguen dos fases del firmware en su conjunto, una donde se está iterando la plantilla para cancelar lo mejor posible el artefacto y otra donde se tiene una plantilla final que se reproduce cada vez que llega una nueva estimulación. En esta última fase se debe corroborar que se esté cancelando satisfactoriamente el artefacto, de lo contrario se volverá a la fase anterior para actualizar la plantilla.

Por esta característica del sistema donde se tienen diferentes fases, el firmware se implementó como una máquina de estados donde se distinguen tres fases, las dos mencionadas anteriormente, y una fase inicial donde se realizan acciones previas para la correcta inicialización de todos los componentes. En la Fig.5.2 se detallan las fases antes mencionadas.

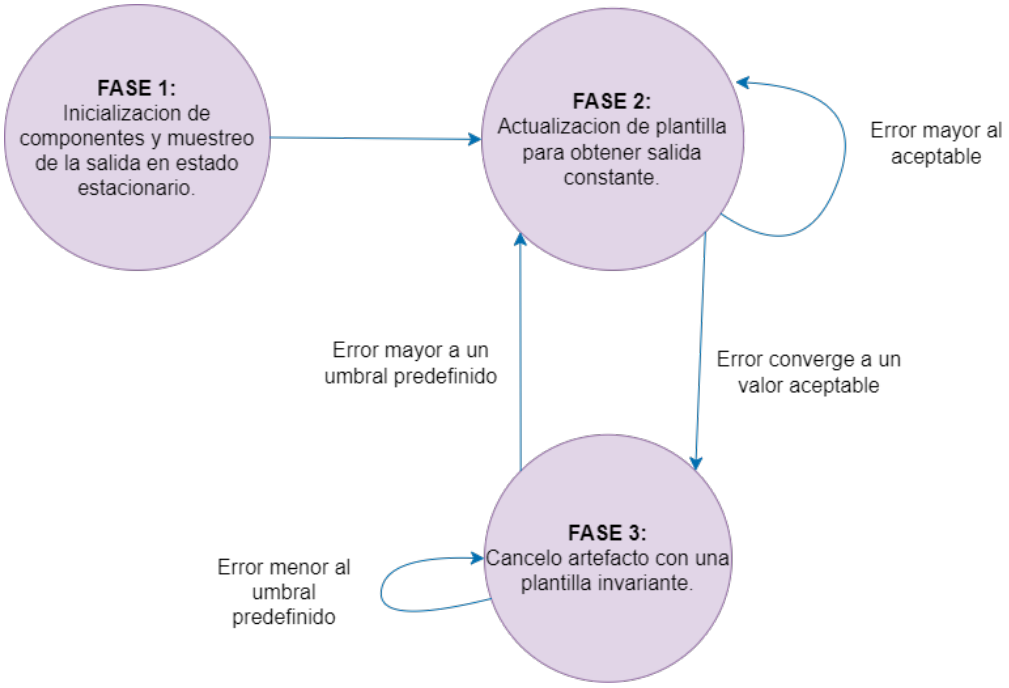


Figura 5.2: Diagrama de estados.

Implementación del controlador.

En la sección anterior se introdujo que dentro del lazo de control hay un controlador PI, también se menciona que el propósito del controlador es iterar la plantilla guardada en el microcontrolador utilizando la señal de error, durante la fase dos del sistema .

Para lograr esto se desarrolló un módulo de firmware donde se implementa un controlador PID que trabaja únicamente con números enteros. Esto se hizo para evitar el uso de números de punto flotante que resultan más costosos desde un punto de vista computacional.

El controlador PID es uno de los controladores más utilizados en una amplia variedad de aplicaciones debido a su versatilidad. Además es relativamente fácil de implementar con respecto a otros controladores más complejos y también cuenta con una base teórica bien establecida por lo que hay ejemplos y soluciones a problemas comunes fácilmente accesibles.

Es importante mencionar que en el proyecto no se utilizó el término derivativo de un PID por lo que podría considerarse un controlador PI. Existen varias razones para evitar el uso de este término, pero las más importantes son que el cálculo del término derivativo es más costoso computacionalmente en comparación con los otros dos y que amplifica el ruido en la señal de error, lo que puede provocar respuestas inestables o no deseadas.

Debido a que el objetivo es generar una plantilla de varios valores, esta implementación se diferencia de un controlador PID convencional, donde se tiene una sola entrada y se actualiza el valor del controlador cada vez que se toma una nueva

muestra. En este caso se toman M muestras, siendo M el producto entre el tiempo de cancelación activa y la frecuencia del ADC y del DAC, durante el proyecto M será 900. Por lo tanto, se establecen M instancias de controladores PID, y quedan definidos M lazos de control en tiempo discreto, con un tiempo de muestreo de $T_s = \frac{2}{f_{estim}}$. Este tiempo se debe a que se pierde un período de estimulación mientras se procesan los datos. Una de cada dos estimulaciones se convierte así en una nueva iteración en los controladores, donde cada controlador recibe la señal de error correspondiente a su posición en la plantilla y el objetivo final es llevar cada muestra al mismo valor que la entrada, cancelando completamente el artefacto.

Todo el desarrollo del controlador se hizo en un módulo particular llamado *PID.c*, adaptado a partir de la implementación presentada en [34]. En esta librería se define una estructura llamada *PIDController*, para interactuar con la mismo se define la función *PIDController_Update* que recibe como parámetros un puntero al controlador a actualizar, la medida en la iteración N y el set point al que se desea llegar, este punto es aquel al cual el controlador intentará llevar la salida, realmente este valor es arbitrario ya que basta con que la señal resultante del lazo de control sea constante para que el amplificador de salida no sature, ya que este último presenta una etapa de desacople de continua. Por último la función devuelve como resultado la salida del controlador correspondiente a la iteración $N + 1$.

5.2. Arquitectura de Firmware.

Por la naturaleza de tiempo real del problema se optó por una arquitectura donde se pudieran implementar interrupciones de manera sencilla. Por otra parte, debido a las diferentes acciones que se deben desarrollar que fueron explicadas al inicio de este capítulo, se decidió implementar el módulo principal como una máquina de estados, ya que permite generar respuestas a diferentes eventos de manera sencilla y escalable.

Con estas consideraciones el núcleo de la implementación de este proyecto es una arquitectura de encolado de eventos (ver Secc.5.2.3) con una máquina de estados subyacente (ver Secc.5.2.4). Este enfoque dual permite construir un sistema versátil y adaptable, capaz de realizar transiciones entre diferentes estados en función de eventos o condiciones.

5.2.1. Módulos de Firmware.

Al momento de escribir Software es importante segmentar las funcionalidad y modularizar el código ya que se logra secciones más pequeñas y manejables, lo que facilita su comprensión, depuración y reutilización.

Como base para describir la modularización implementada, se utilizará la Fig.5.3, que muestra el diagrama de los módulos de firmware.

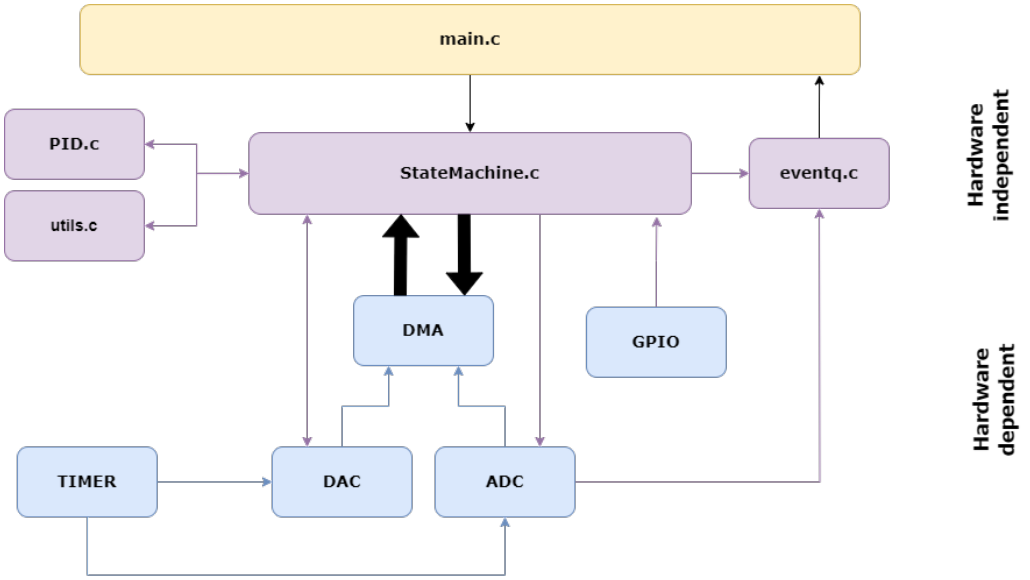


Figura 5.3: Diagrama de módulos de firmware.

Descripción de los módulos.

- **ADC.h**: este módulo maneja la funcionalidad de conversión de datos analógico a digital (ADC). Proporciona una interfaz para leer señales analógicas y convertirlas en valores digitales.
- **DAC.h**: gestiona las operaciones de conversión digital a analógico (DAC). Facilita la conversión de señales digitales en voltajes analógicos correspondientes para salida.
- **DMA.h**: se ocupa de las operaciones de acceso directo a memoria (DMA). Permite transferencias de datos eficientes entre periféricos y memoria sin intervención de la CPU, lo que mejora el rendimiento general del sistema, permitiendo paralelizar tareas relacionadas al manejo de periféricos. Este periférico es de especial importancia para el proyecto ya que permite que el ADC y el DAC funcionen de manera autónoma y simultáneamente sin ocupar al CPU.
- **GPIO.h**: se encarga de las operaciones de entrada/salida (GPIO). Su función mas importante es la de la detección del comienzo de estimulación y la interrupción correspondiente.
- **TIM.h**: gestiona las operaciones de temporizadores. Proporciona una interfaz para configurar y utilizar temporizadores en el microcontrolador principalmente para que funcionen como condición de disparo del ADC y el DAC.
- **PID.h**: mencionado anteriormente, implementa un controlador PID con datos del tipo entero.

- **UTILS.h**: incluye funciones de utilidad que cumplen diversos propósitos, entre los más importantes se encuentran la inicialización de periféricos secundarios y funciones de cálculo de error.
- **STATE_MACHINE.h**: responsable de implementar la máquina de estados, definiendo varios estados y transiciones dentro del firmware.
- **EVENTQ.h**: define la cola y los eventos para la implementación del encolado de los mismos.

5.2.2. Sobre el ADC y el DAC.

El correcto funcionamiento de periféricos encargados de la conversión y reproducción de datos analógicos es una parte vital del proyecto. Se debe tener en cuenta el funcionamiento de cada uno junto a sus no idealidades relacionadas. Principalmente lo que puede considerarse un obstáculo para la correcta convergencia del algoritmo, son los tiempos de demora que se introducen cada vez que se comienza una nueva conversión.

En la Fig.5.4 se puede ver una señal de artefacto y los momentos en los que los periféricos deberían estar encendidos.

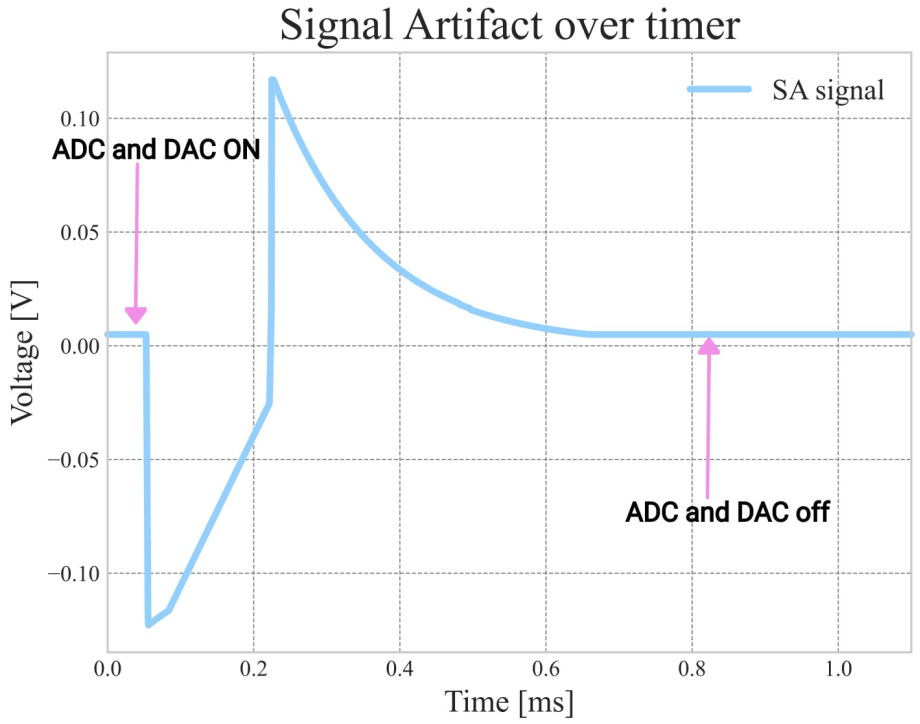


Figura 5.4: Señal de artefacto en el tiempo.

En la figura se puede notar que en los últimos 200 ms la señal de artefacto

no es muestreada ni el DAC está encendido. Esto se hizo para aprovechar que la señal en su última etapa es prácticamente constante, por lo tanto colocando la última muestra reproducida en el DAC se obtendrá el resultado deseado y el procesamiento de datos se puede comenzar antes.

Orden de los periféricos.

Considerando que ambos periféricos introducen un retardo se debe evaluar cuál es el orden en que se encenderán. En la Fig.5.4 se observa que la mayor variación de la señal se encuentra en el inicio de la estimulación, por esta razón se decidió encender el DAC antes que el ADC, de esta manera se comienza la cancelación un tiempo de retardo antes. Si los tiempos de retardo son los suficientemente grandes se podría llegar a perder el sincronismo en el tiempo entre la reproducción de la plantilla y la adquisición de la señal de error, esto causaría que el algoritmo no llegue nunca a converger, sin embargo manejando los periféricos correctamente se consigue minimizar el retardo al punto que no sean perjudiciales.

Manejo de los periféricos.

Existen diversos métodos para iniciar una conversión de datos ya sea con el ADC o con el DAC, en este proyecto el objetivo es minimizar el retardo que introducen cuando se encienden.

Durante el mismo se trabajó con el microcontrolador *STM32l552ZE* [35], y para controlar tanto los periféricos como algunas funcionalidades básicas se utilizó la capa de abstracción de hardware (por su siglas en inglés HAL) proporcionada por el fabricante [36]. Los periféricos de interés son el ADC, el DAC y los temporizadores. Además, este microcontrolador permite que algunos periféricos interactúen de manera directa con la memoria a través de otro periférico llamado DMA (Direct Memory Access). De esta manera se logran paralelizar tareas ya que no se necesita interacción del CPU para colocar o quitar los datos de la memoria. En el caso de este proyecto se utilizó para que el ADC tome muestras cada vez que expira un temporizador y las coloque en un espacio de memoria. El caso del DAC es análogo.

Para adquirir o reproducir muestras es necesario que el periférico correspondiente:

1. Este correctamente inicializado.
2. Se haya iniciado una conversión.
3. Se cumpla la condición de disparo.

La inicialización se realiza una única vez al inicio del programa principal. Luego diremos que el periférico está *preparado* cuando se haya iniciado una conversión y este a la espera de la condición de disparo. Por lo tanto, surgen dos métodos para controlar la adquisición y reproducción de muestras, una consiste en iniciar la conversión cuando se desee utilizando la biblioteca HAL, con un único temporizador que se mantiene encendido en todo momento y es el mismo para ambos periféricos . El otro mantiene los periféricos preparados en todo momento y cuando se desee

iniciar o detener la adquisición/reproducción se hace con el encendido o apagado de los temporizadores, en este caso serán dos para poder controlar independientemente cada periférico.

Buscando minimizar el retardo en la Tab.5.1 se relevó este tiempo entre los dos métodos:

- Método 1 (M1): Mantener un temporizador único para controlar los dos periféricos y al momento de ser necesario iniciar la preparación,

- Método 2 (M2): Mantener los periféricos preparados en todo momento, y encender y apagar dos temporizadores, cuando se desea controlar cada uno.

	ADC	**DAC**
Retardo utilizando M1 (μs)	8.5	4
Retardo utilizando M2 (μs)	< 1	< 1

Tabla 5.1: Tiempo de retardo de los periféricos utilizando varios métodos.

De la tabla se observa que iniciando la adquisición/reproducción utilizando los temporizadores minimiza el tiempo de retardo, por lo tanto se decidió por utilizar el M2 para el manejo de los periféricos.

Para finalizar la temática de los periféricos cabe destacar que los mismos quedaron configurados con los temporizadores, y la configuración de los mismos se hace mediante el macro *TIMER_TRIGGER_US*, en la cual se indica el valor en microsegundos a configurar los temporizadores tantos del DAC y ADC. En su estado por defecto es igual a 1 obteniendo una velocidad de muestreo y reproducción de la plantilla de 1 MHz.

Secuencia de funcionamiento.

En la Fig.5.5 se puede ver un diagrama de secuencia donde se muestra el comportamiento de los periféricos y el procesador cuando se detecta una señal artefacto.

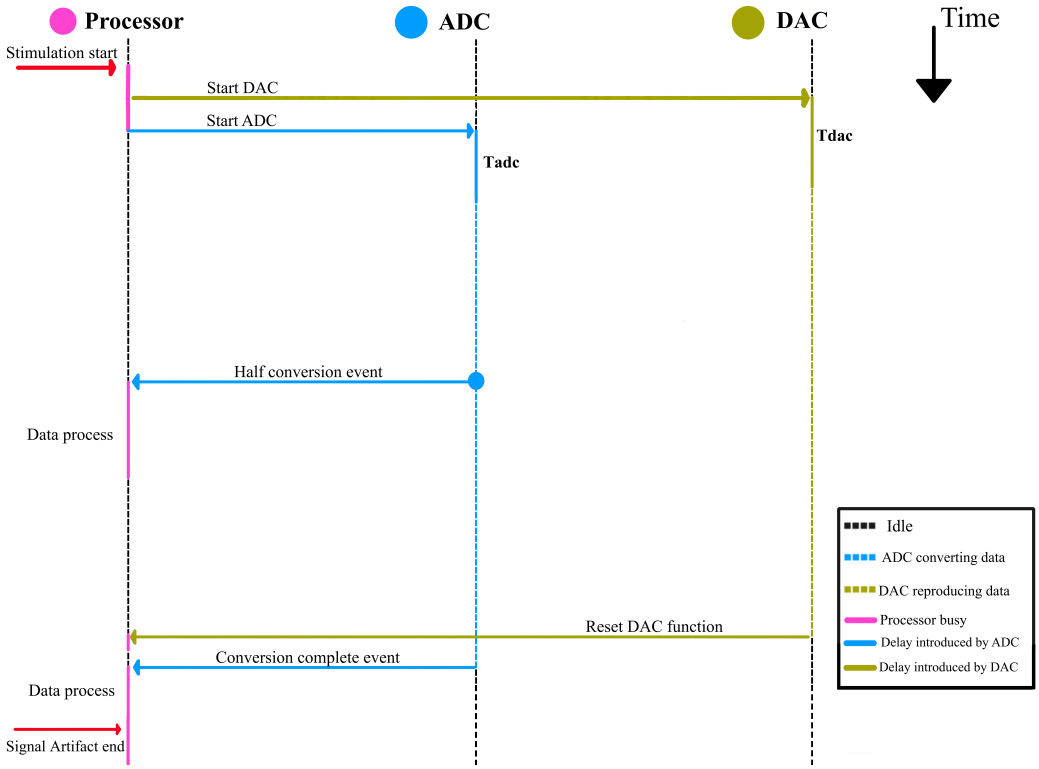


Figura 5.5: Diagrama de secuencia de los periféricos y el microcontrolador durante adquisición de muestras.

Como se mencionó anteriormente, para controlar el ADC y el DAC se mantienen preparados para la conversión en todo momento y luego cuando se quiere iniciar se enciende el temporizador correspondiente. En la figura se puede notar que cuando llega la segunda señal de estimulación aún se están procesando datos, por lo tanto uno de cada dos períodos de estimulación no es adquirido ni cancelado ya que se utiliza ese tiempo para el cálculo y actualización de la plantilla.

Por lo tanto al momento de detectar la estimulación, se ejecutará una rutina de servicio a la interrupción (ISR). Esta interrupción es la encargada de encender los temporizadores, la misma se deshabilita durante el procesamiento de datos.

Es importante remarcar que esta secuencia sucede únicamente en la fase dos del algoritmo, que es donde se realiza el calculo e iteración de la plantilla hasta lograr la convergencia.

5.2.3. Sobre Encolado de Eventos

En este proyecto se llamó “eventos” a una situación particular que requiere del procesamiento del microcontrolador, como por ejemplo la detección de estimulación o el fin de conversión del ADC.

La arquitectura esta basada en encolar estos eventos a una cola especifica para luego procesarlos utilizando la máquina de estados. Funciona de manera análoga a

una arquitectura de encolado de funciones, pero presenta la ventaja de que se puede responder de forma diferente a un mismo evento según en que estado de la máquina se encuentre el sistema. El módulo encargado de la implementación del encolado es llamado *eventq.h*, esta basado en la arquitectura propuesta en [37]. En este archivo se definen los eventos que son capaces de insertarse en cualquier parte del código. Estos eventos se envían a una cola del tipo FIFO (first input, first output), este tipo de colas permite seguir un hilo temporal donde los eventos se organizan y ejecutan de forma estructurada. Luego, el programa principal se mantiene en un bucle infinito donde se consume un elemento de la cola y se procesa el mismo con la máquina de estados.

De esta forma siempre se ejecuta la misma función, correspondiente al procesamiento de la máquina de estados, y se pasa como parámetro a esta función el último evento tomado de la cola antes mencionada.

Esta implementación presenta una importante limitación; ya que el procesamiento de un evento se hace de manera bloqueante, se debe esperar a que se termine de procesar el evento actual, para que comience el siguiente. Es por esto que se tomó la decisión de iniciar los temporizadores del DAC/ADC en la ISR ya que esta se ejecutará siempre, sin importar si se esta ejecutando o no un evento.

Para la implementación de los eventos se definió un nuevo tipo de datos, en particular es una enumeración de todos los eventos posibles, estos son:

- ENTRY_STATE: se encola siempre que se cambia de estado, se procesa en el nuevo estado al que se llegó.
- EXIT_STATE: se ejecuta siempre antes de cambiar de estado.
- STIMULATION_DETECTED: se da cuando se detecta una estimulación en uno de los pines del microcontrolador.
- GET_ERROR:se utiliza para obtener una muestra representativa del error.
- ADC_HALF_CONVERSION_COMP: evento para indicar que la conversión del ADC completo la mtiad de la cola del DMA.
- ADC_CONVERSION_COMP: evento para indicar que la conversión del ADC completo la cola del DMA.
- EVAL_TEMPLATE: se da para evaluar el desempeño de la plantilla actual.
- START_ALGORITHIM: evento para indicar el comienzo del cálculo del algoritmo para actualizar la plantilla del DAC.

5.2.4. Implementación de la Máquina de Estados

Como se mostró previamente en la Fig.5.2 para la implementación se utilizaron tres estados, que se llamaron fases, todo el desarrollo se realizó en el módulo llamado *state_machine.h*. En este archivo se definen nuevos tipos de datos para el desarrollo de la máquina de estados.

En primer lugar se define una enumeración llamada *SM_status*. Esta se utiliza para indicar el resultado del procesamiento de un evento en un estado. Es decir cada vez que se procesa un evento se puede obtener uno de los siguientes resultados:

- TRANS_STATUS: indica que se dio una transición de estados.
- HANDLED_STATUS: indica que el evento se manejó correctamente.
- IGNORED_STATUS: indica que el evento fue ignorado.

El segundo tipo de datos que se define es el *State_Handler*, esta declaración define un tipo de puntero a función. Este puede apuntar a una función que toma un argumento de tipo *Event* constante y devuelve un valor de tipo *SM_Status* según como se procese el mismo.

Por lo tanto cada uno de los estados de la máquina se define como una función a la cual se le pasa un evento como parámetro. La variable que lleva el estado actual de la máquina es un puntero a función que apunta al estado actual.

De esta forma, la máquina de estados puede reaccionar de forma diferente ante un mismo evento dependiendo del estado en el que se encuentre.

Descripción de cada fase.

A continuación se resume el funcionamiento de cada fase y se detallan los eventos implementados para cada una. Para evitar repetición se omiten los eventos de ENTRY_STATE y EXIT_STATE que se presentan en todas las fases.

- **FASE 1**: Esta fase comienza por encender los periféricos de ADC y DAC así como los temporizadores asociados a los mismos. Luego inicia una conversión en el ADC con la salida en estado estacionario para obtener el valor inicial al cual se deseará llegar con los controladores PID. Una vez se obtuvo un valor razonable, es decir un valor entre los rangos esperados que en el caso de la salida sin señal debería ser alrededor de 2048, luego se hace la transición a la fase dos. En la Fig.5.6 se puede ver un diagrama de flujo de la fase uno.

  Los eventos implementados en esta fase son:

  - ADC_HALF_CONVERSION_COMP: para ser más eficiente en el tiempo se comienza a cargar la primera mitad de los datos del DMA a una variable para trabajar dentro del modulo.
  - ADC_CONVERSION_COMP: se carga la segunda mitad de los datos del DMA, se procesan los mismos. Si son razonables se inicia la transición a fase dos.

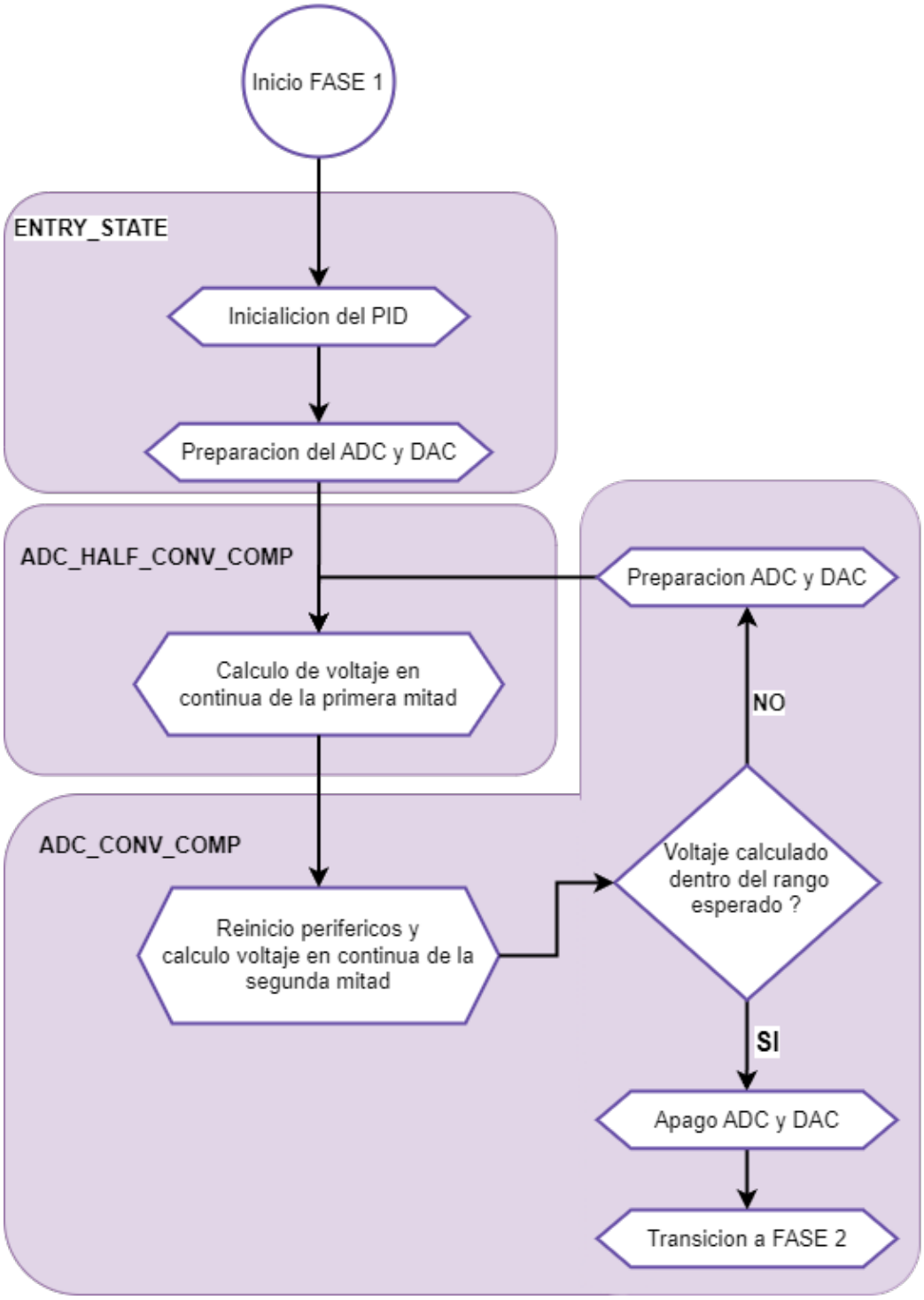


Figura 5.6: Diagrama de flujo de la fase uno de firmware.

- **FASE 2**: En esta fase es donde se actualizarán todos los controladores hasta alcanzar una plantilla que asegure una salida relativamente constante. Se realizan conversiones del ADC y se procesan los datos, actualizando en cada iteración la plantilla de DAC. La Fig.5.7 muestra el diagrama de flujo de esta fase.

  Los eventos implementados en esta fase son:

  - ADC_HALF_CONVERSION_COMP: nuevamente se comienza a cargar la primera mitad de los datos del DMA en variables internas al módulo.
  - ADC_CONVERSION_COMP: al entrar se deshabilitan las interrupciones del pin de estimulación ya que durante el cálculo de la siguiente plantilla no se desea que la anterior se siga reproduciendo. Luego se carga la segunda mitad de los datos del DMA. Por último se encola un evento llamado START_ALGORITHIM.
  - START_ALGORITHIM: al procesar este evento en fase dos se calcula el error obtenido con la plantilla actual, si es suficientemente bajo se hace la transición a la fase tres, en cambio sino llego a la condición de convergencia se actualizan todos los controladores. En cualquiera de los casos se habilitan nuevamente las interrupciones de estimulación.

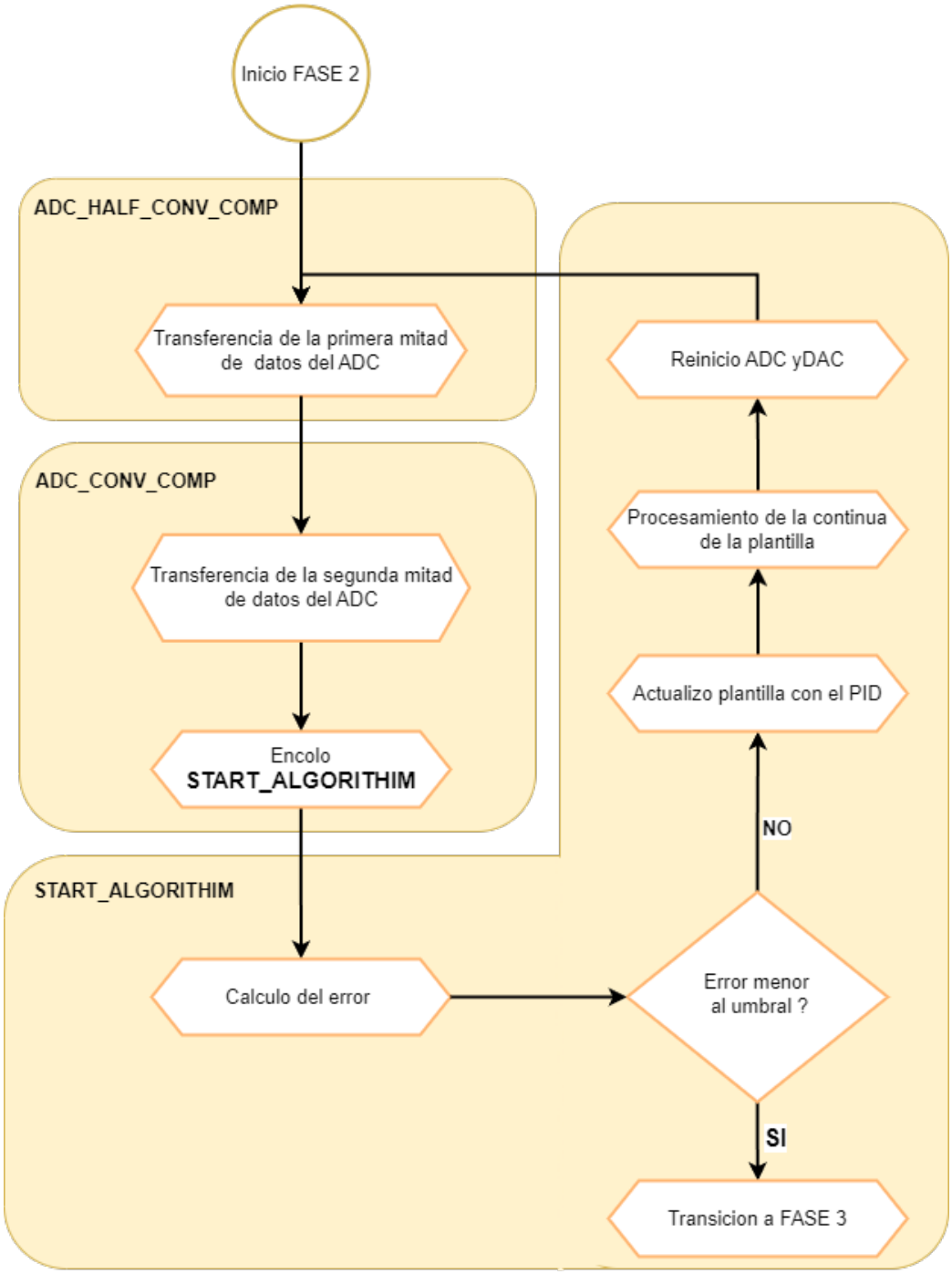


Figura 5.7: Diagrama de flujo de la fase dos de firmware.

- **FASE 3**: en esta fase se desea que al llegar una estimulación, el DAC reproduzca la plantilla generada en la fase dos. No se desea que se omita ninguna instancia de estimulación por lo que el DAC debe estar encendido siempre que se de la señal. En cambio el trabajo del ADC consiste en iniciar una conversión cada cierto tiempo para evaluar si la plantilla esta obteniendo resultados aceptables. En la Fig.5.8 se puede ver el diagrama de flujo de esta fase.

  Para lograr este comportamiento se implementan los siguientes eventos:

  - SAMPLE_ADC: Este evento es encolado por la interrupción del pin de estimulación una vez cada 100 interrupciones. Dentro del evento se enciende el ADC para iniciar una conversión.

  - SCALE_TEMPLATE: este evento es desencadenado por un pin digital controlado por el módulo de software, se utiliza para indicar que se debe escalar linealmente la plantilla obtenida. El valor por el cual se escala debe ser el mismo tanto en software como en firmware y se debe indicar antes de la compilación del código, su valor por defecto es 1,5.

  - ADC_CONVERSION_COMP: Se utiliza para detener el periférico del ADC. Luego se debe tomar un valor base del error, esto es para poder hacer comparaciones con el error al momento de converger, para esto se encola el evento llamado GET_ERROR, si el valor base ya fue adquirido entonces se encola el evento EVAL_TEMPLATE.

  - GET_ERROR: utiliza los datos del ADC para tomar una valor significativo del error de la plantilla luego de la convergencia, este valor se le llamará *error base*. Para esto realiza el promedio del error en N iteraciones, por lo tanto se debe encender el ADC las veces necesarias. Una vez obtenido el promedio se señaliza que el error base ya fue adquirido.

  - EVAL_TEMPLATE: al igual que el evento anterior realiza un promedio del error cometido en N iteraciones, una vez adquirido este promedio se calcula el *error normalizado* siendo este igual al error cometido sobre el error base, esto multiplicado por 100 para obtener el porcentaje. Luego se compara el resultado con un umbral llamado *ACEPTABLE_ERROR*, si es mayor al umbral se vuelve a fase dos, para iterar la plantilla y disminuir el error nuevamente.

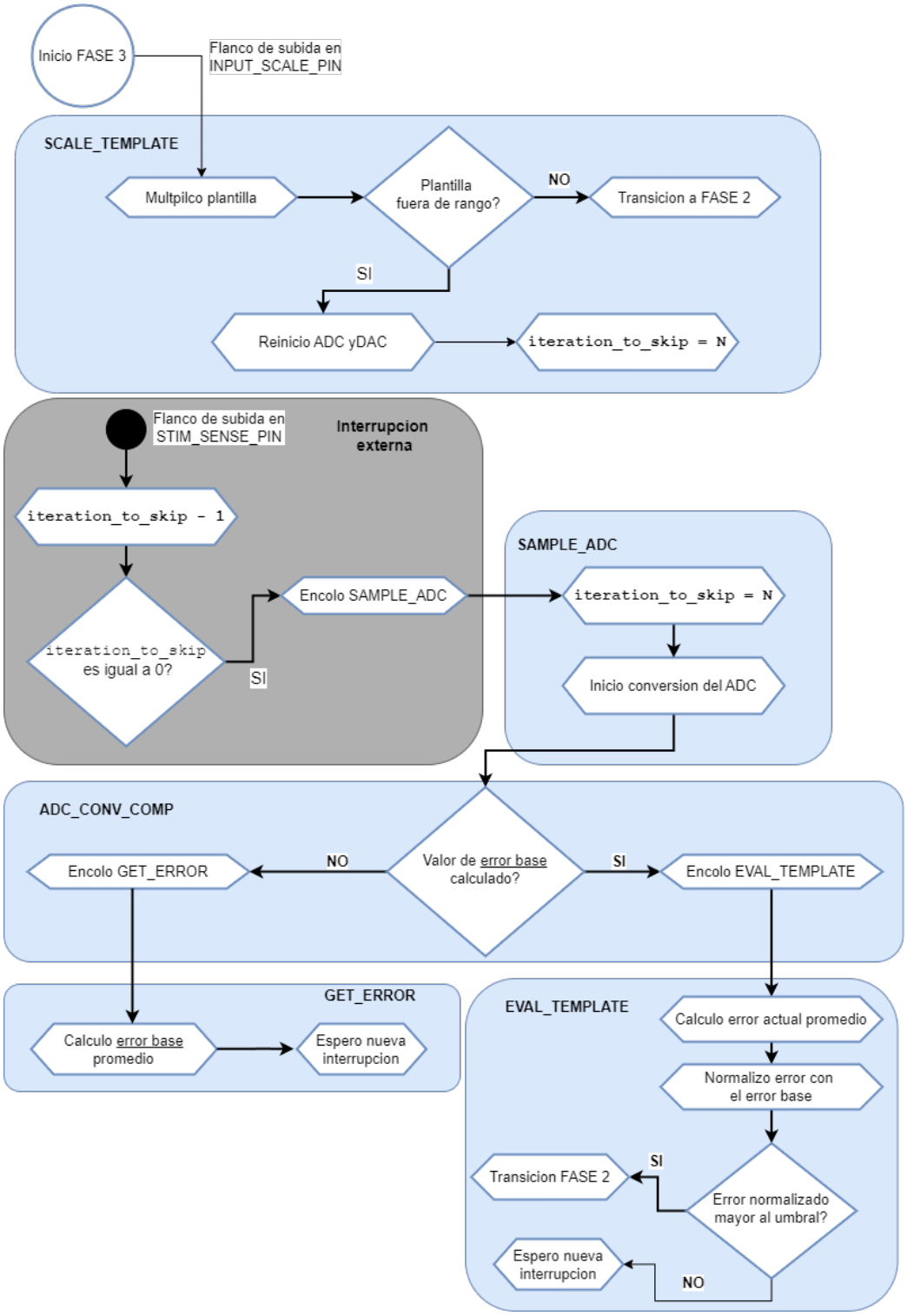


Figura 5.8: Diagrama de flujo de la tres de firmware.

Métodos de cálculo de error.

Durante este capítulo se mencionó el error como forma de controlar el flujo de la maquina de estados. Por lo tanto surge la necesidad de definir cómo se realizará este cálculo.

El objetivo del algoritmo es obtener en la salida una señal constante, por lo tanto el cálculo de error debe ser capaz de distinguir y cuantificar este atributo. Con esto en cuenta se desarrollaron 4 métodos de error. Estos se puede encontrar en el modulo de firmware *utils.h*.

- ERROR_COUNT: el objetivo de este método es cuantificar la cantidad de muestras del ADC que se desvían $5mV$ del valor deseado. El problema con este método es que la continua del residuo no esta completamente definida. Esto causa que solo converja cuando la continua coincida con el valor deseado. En este caso no es deseable ya que no se busca llegar a un valor especifico sino que la señal sea constante.

- ERROR_MEAN: al utilizar este método se hace una sumatoria de la diferencia entre el valor deseado y el de cada muestra, obtenido este valor se divide entre la cantidad de muestras. Este método presenta el mismo problema que el anterior ya que depende de la diferencia entre un valor especifico.

- ERROR_DC: este método surge para solucionar el problema de los dos anteriores. Al inicio se intenta calcular el valor DC de la señal de residuo calculando el promedio de todas las muestras. Luego se cuentan cuantas muestras se alejan más de un valor predefinido del valor DC obtenido. El problema en este caso surge de la aproximación del valor de la continua utilizando el promedio, por la naturaleza del problema, se encontraran espurios de artefacto que no serán posibles cancelar, estos espurios causan una variación considerable en el cálculo de la continua, ocasionando que no sea útil para controlar los cambios de estados.

- ERROR_STDDEV: por último se exploró la alternativa de usar la desviación estándar como figura de mérito de que tan constante es la señal en consideración. Debido al costo computacional de los cálculos con números del tipo punto flotante, se utilizó en cambio la varianza para evitar tener que hacer la operación de raíz cuadrada. Este método permite controlar el flujo de la maquina de estados ya que se logra distinguir a partir de resultados empíricos los umbrales para cambiar de fase.

5.3. Lazo de Control de Continua

Debido a la arquitectura de hardware (ver Cap.4) tanto la señal proveniente del DAC como a la de artefacto se le desacopla la continua y la resta queda centrada en un valor de referencia.

Esto puede llegar a ser un problema para el algoritmo ya que es necesario tener control sobre la continua de la plantilla, porque de lo contrario podría quedar sobre

los limites de funcionamiento del DAC. Esto se debe a que el controlador integra el residuo, si hay una pequeña variación en la continua del residuo y el valor que se desea alcanzar, ocasionará la saturación del DAC ya que tiene un rango de funcionamiento limitado. Por lo tanto, es necesario que el valor de continua de la plantilla resultante sea tal, que todos los valores queden en el rango de funcionamiento de los periféricos.

Para resolver este problema se agregó una sección de código durante el procesamiento del evento de *START_ALGORITHIM*. Este agregado se encarga de revisar la plantilla, en el caso de encontrar en la misma alguno de los valores límites del DAC; disminuye o aumenta el valor deseado al cual llegar con los controladores PID según corresponda.

En la Fig.5.9 se muestra el diagrama de flujo del lazo de control de continua.

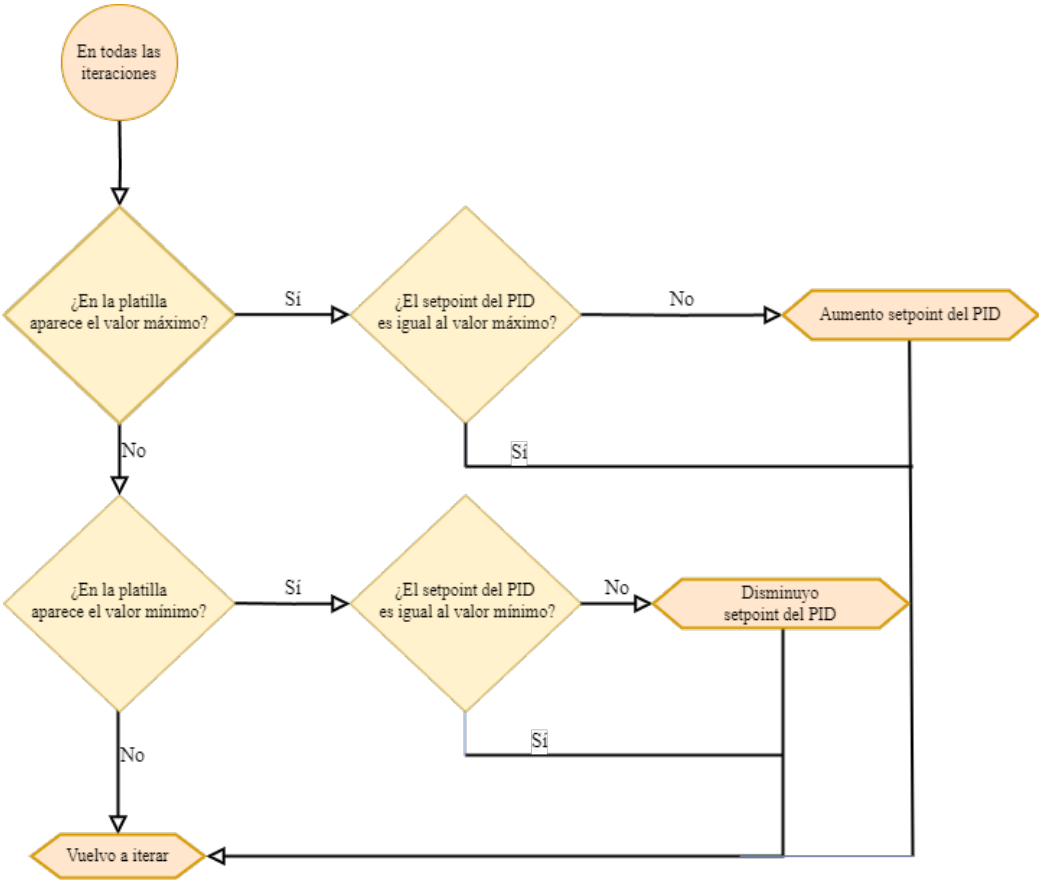


Figura 5.9: Diagrama de flujo del control de continua.

Con esta corrección en el valor deseado se logra llegar a un valor de continua con el cual el DAC es capaz de cancelar todo el artefacto.

5.4. Parámetros Configurables

Nombre	Valor por defecto	Ubicación
COMPILE_ALGORITHM	1	*state_machine.c*
VBIAS_IDEAL	2046	*state_machine.c*
ERROR_THRESHOLD	5	*state_machine.c*
ACEPTABLE_ERROR	120	*state_machine.c*
CYCLE_AVERAGE	64	*state_machine.c*
ITERATION_TO_SKIP	100	*state_machine.c*
ERROR_METHOD	STDDEV	*utils.h*
ESTIM_PERIOD_US	1100	*main.h*
BUFFER_TIME_US	100	*main.h*
TIMER_TRIGGER_US	1	*main.h*
K_p	64	pid.h
P_ACTIVE	1	pid.h
K_d	0	pid.h
Ki_q	1	pid.h
Ki_d	1	pid.h

Tabla 5.2: Parámetros configurables de FW y sus valores por defecto.

A continuación se describe el significado de cada uno de los parámetros configurables introducidos en la Tab. 5.2.

- **COMPILE_ALGORITHM:** Indica si se utilizará el PID en el lazo de control.
- **VBIAS_IDEAL:** Valor en muestras del ADC del voltaje que se debe medir con el sistema en estado estacionario.
- **ERROR_THRESHOLD:** Umbral del error para converger a fase tres.
- **ACEPTABLE_ERROR:** Umbral del error normalizado para ir de fase tres a fase dos.
- **CYCLE_AVERAGE:** Cantidad de ciclos para realizar el promedio del error en fase tres.
- **ITERATION_TO_SKIP:** Número de iteraciones para mantener el ADC apagado en fase tres. Cuando llega a cero se enciende el mismo.
- **ERROR_METHOD:** Método del cálculo de error a utilizar. Se puede setear con los siguientes valores:
  - ERROR_MEAN (0) - Se calcula el valor promedio utilizando los datos del arreglo donde se almacena el error de las muestras.

  - ERROR_COUNT (1) - Se calcula la diferencia entre cada valor del arreglo y el error deseado. Se aumenta el contador de errores cuando la diferencia supera un valor umbral definido.
  - ERROR_DC (2) - Se calcula la diferencia de cada valor del arreglo de errores con el valor DC del arreglo. Nuevamente se compara la diferencia contra un umbral.
  - ERROR_STDDEV (3) - calcula la varianza del arreglo de errores. La varianza se calcula como la elevación al cuadrado de la desviación estándar.

- **ESTIM_PERIOD_US:** Período de estimulación configurado. Valor en microsegundos.
- **BUFFER_TIME_US:** Tiempo del período de estimulación en el cual no se está cancelando activamente el artefacto. Valor en microsegundos.
- **TIMER_TRIGGER_US:** Tiempo en microsegundos para la configuración de los temporizadores
- K_p**:** Ganancia proporcional del controlador PID.
- **P_ACTIVE:** Indica si se encuentra activa la ganancia proporcional del controlador PID
- K_d**:** Ganancia derivativa del control PID.
- Ki_q**:** Ganancia Integral (Numerador).
- Ki_d**:** Ganancia Integral (Denominador).

5.5. Resultados

En esta sección se analizarán los resultados obtenidos de la implementación del HW + FW en la cancelación activa de artefactos, esto quiere decir que durante esta sección el firmware estará siempre en fase tres luego de la convergencia del algoritmo.

5.5.1. Precisión del ADC

Antes de comenzar con los resultados, se analizará un problema que resultó en un desafío para el correcto funcionamiento del sistema que no se tuvo en cuenta previamente. En la Fig.5.10 se muestra un histograma de la desviación de la muestra Nº 200 del ADC, para relevar estos valores se trabajó con el algoritmo en fase tres, es decir siempre se reproducía la misma plantilla por el DAC. Esta figura muestra que el ADC comete errores considerables para una misma entrada, para este caso la desviación estándar es de $84,4$ LSBs, que haciendo la conversión son $68mV$. Por este problema es que en fase tres se realiza un promedio del error

cometido con el objetivo de no volver a fase dos por las desviaciones aleatorias del ADC. También este problema supone un obstáculo para el controlador PID, que puede llegar a demorar más de lo debido y vuelve imperativa la condición de convergencia, ya que no se podría estar actualizando constantemente la plantilla porque el controlador trataría de corregir las desviaciones intrínsecas al ADC haciendo cambios innecesarios o incluso perjudiciales para la correcta cancelación de los artefactos.

Es importante mencionar que si bien el histograma muestra únicamente la muestra número 200 este comportamiento ocurre para todas las muestras, sin embargo la diferencia entre una muestras y la siguiente se mantiene constante, que en definitiva es lo más importante para el cálculo de la plantilla. Esta variación en la conversión del ADC se puede interpretar como una variación en la continua de la señal, sin embargo esto es un defecto del periférico ya que midiendo la señal con un osciloscopio se comprobó que la misma no presenta estas variaciones.

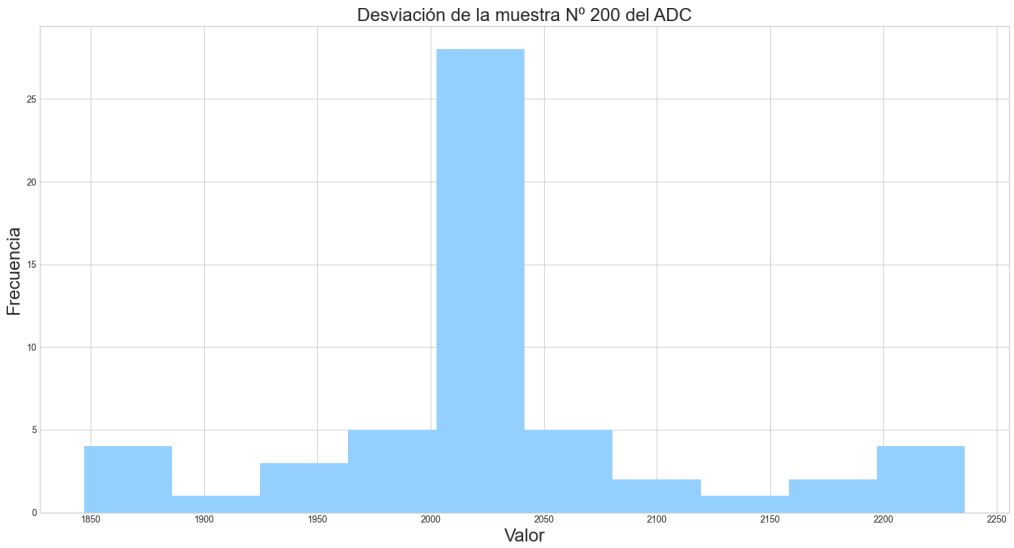


Figura 5.10: Estadística de la desviación de la muestra 200 del ADC.

5.5.2. Cancelación de Artefactos

Para comenzar el análisis de resultados se muestra la Fig.5.11, en la misma se presentan dos señales. La señal celeste es la plantilla siendo reproducida por el DAC, mientras que en violeta se muestra la entrada de una señal de SA al sistema de 200 $mVpp$, en particular esta entrada se relevó midiendo la salida del amplificador de instrumentación (INA). Para esta prueba no se agregó la señal de ECAP.

Por último es importante mencionar que a la señal de entrada se le agregó un offset de 2,8 V para hacer más fácil la visualización en el gráfico.

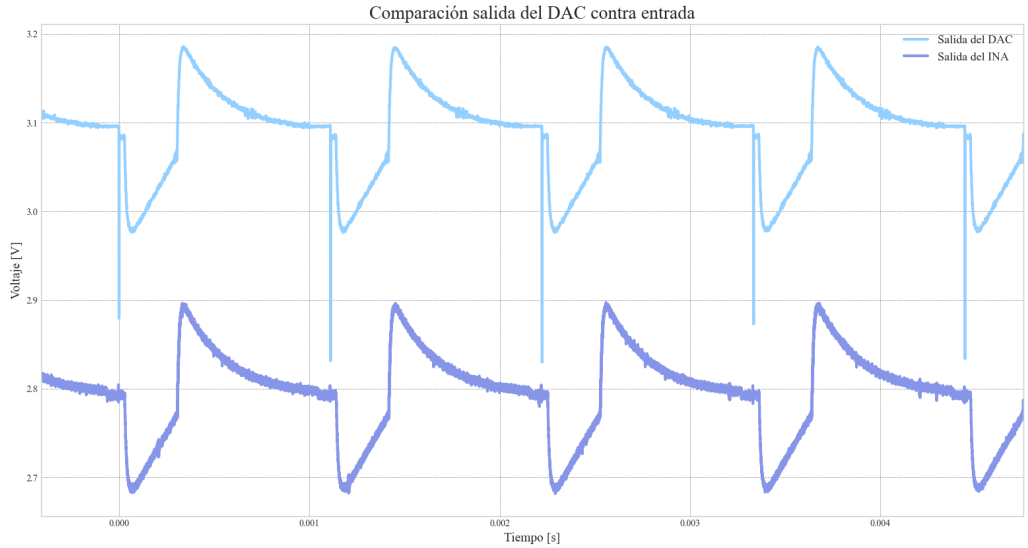


Figura 5.11: Comparación entrada al sistema y plantilla reproducida por el DAC.

De esta figura se pueden sacar varias conclusiones. La primera es que el algoritmo logra replicar la señal de entrada de manera satisfactoria teniendo algunos problemas sobretodo al inicio de la señal. Eso es esperable según lo previsto en el Cap.3.

Otro resultado importante que se desprende de esta figura, es que si bien el lazo de control de continua funciona puede llegar a ser un problema, ya que para este caso la señal esta centrada en 3,1 V, sobre el límite superior del DAC. Si la señal fuera más grande generaría conflictos y debería volver a fase dos para generar una nueva plantilla con menor valor de continua.

Para continuar el análisis se hará uso de las Fig5.13 y Fig.5.12. En estas figuras se graficaron dos señales, una correspondiente a la salida del sistema completo (en violeta) y la otra, (en celeste) correspondiente a la salida previo al filtro discreto a la salida (ver Secc.4.2.4, Fig.4.15). Esta última señal si bien es más ruidosa y no es la final se agregó a la comparación porque es de ayuda para obtener información precisa sobre la saturación del amplificador de salida. En las figuras antes mencionadas se muestran dos casos del residuo del artefacto amplificado, luego de la cancelación por parte del firmware. La entrada para ambos casos es una señal de SA de 200 $mVpp$.

El caso de la Fig.5.12 es el resultado usual de la convergencia a la fase tres, donde se observa que al inicio de la señal se obtiene el mayor error cometido. Analizando la salida previa al filtro, el amplificador a la salida satura por un breve período de tiempo, esta conclusión se llega al recordar que el rango dinámico del amplificador es de $\pm 5V$.

Sin embargo de la Fig.5.13 se deducen dos aspectos importantes, el primero es el hecho de que el residuo que resulta del algoritmo no siempre es el mismo, a pesar de que en ambos casos se cumple el criterio de convergencia. El segundo es que en esta última figura se observa como el amplificador no llega a saturar en ningún momento, siendo este el mejor caso para la detección de la señal de ECAP.

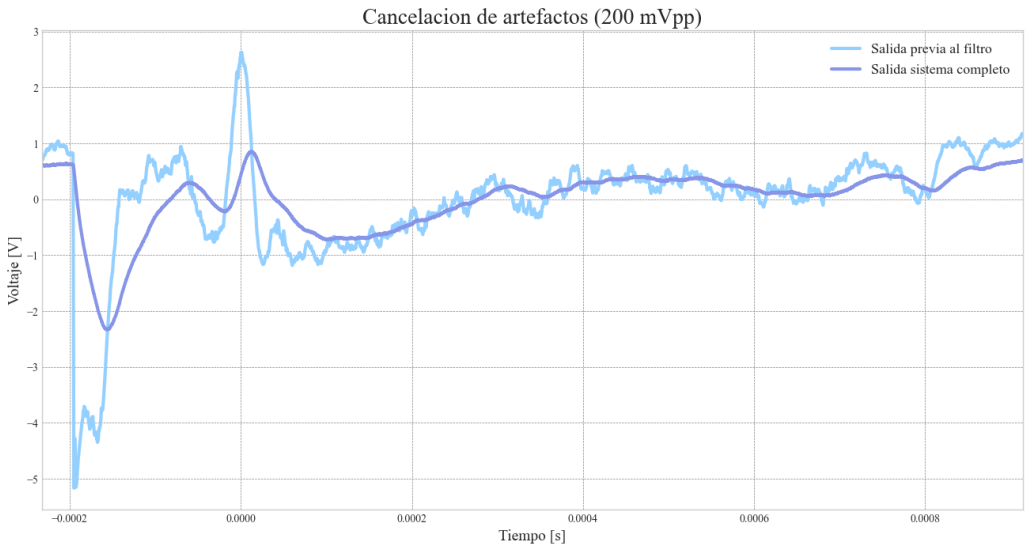


Figura 5.12: Salida del sistema con cancelación de artefactos, SA de amplitud 20mVpp sin ECAP a la entrada, caso típico.

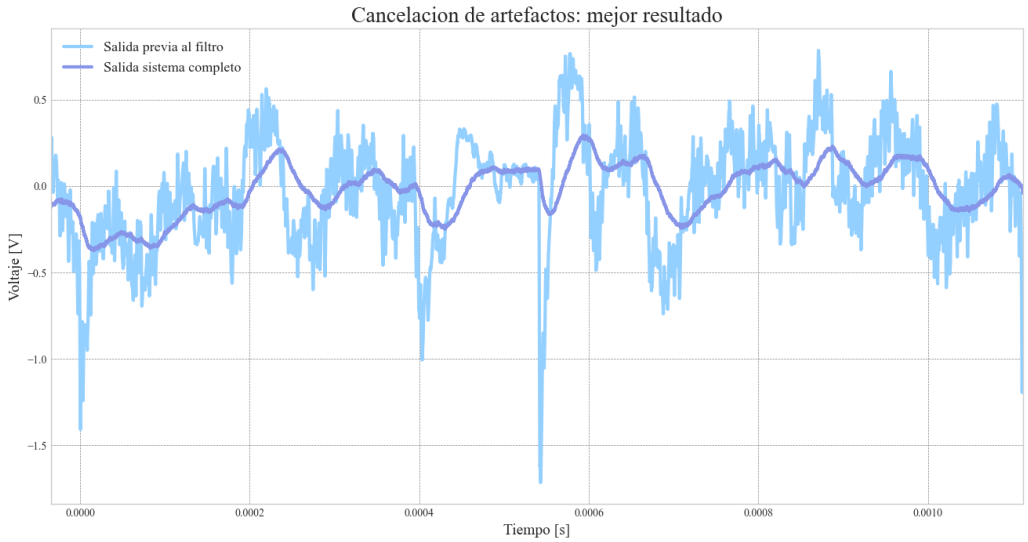


Figura 5.13: Salida del sistema con cancelación de artefactos, SA de amplitud 20mVpp sin ECAP a la entrada, mejor caso.

Para terminar con el análisis se incluyó en las pruebas la señal de ECAP para ver cómo es su efecto en las señales previamente estudiadas durante la fase tres del algoritmo. Todas las figuras a continuación fueron extraídas para el caso de la convergencia vista en la Fig.5.12, en todos los casos se estaba cancelando un artefacto de $200mVpp$ y se hizo variar la amplitud de la señal de ECAP para observar los límites de funcionamiento.

Para el caso de la Fig.5.14 se utilizó una ECAP de $4mVpp$ a la entrada, donde se observa claramente su efecto en la señal a la salida en los $400\mu s$ luego del arranque de la estimulación. Luego, para la Fig.5.15 se bajó la amplitud a $2mVpp$, se puede notar que es más difícil identificar el efecto de la señal original y se confunde con el residuo del artefacto. Por último, para la Fig.5.16 se redujo nuevamente la amplitud a $1mVpp$ donde ya no se distingue a simple vista la diferencia con el residuo sin ninguna ECAP, por lo tanto es necesario deshacerse del residuo para poder detectar señales de menor amplitud.

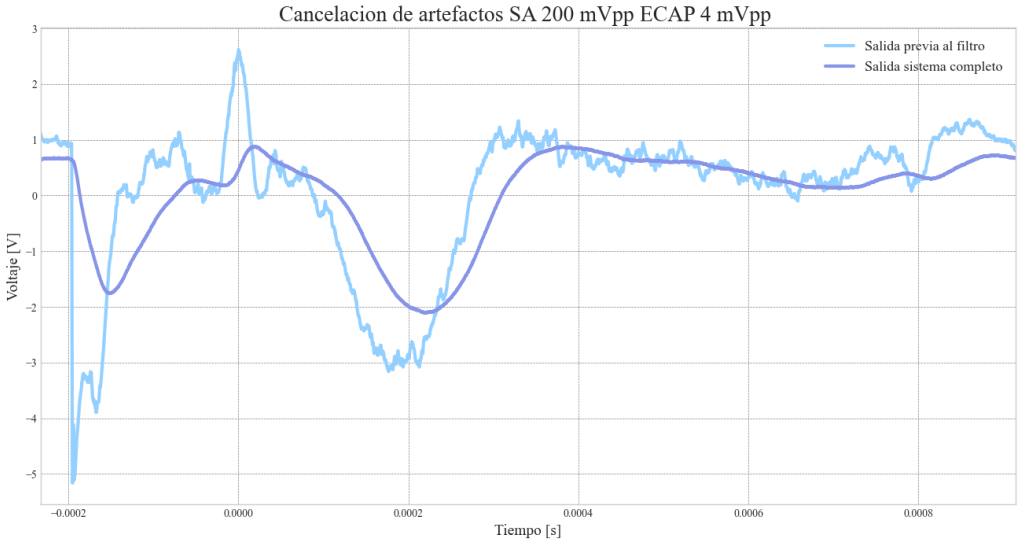


Figura 5.14: Salida del sistema con SA $200mVpp$ y ECAP $4mVpp$.

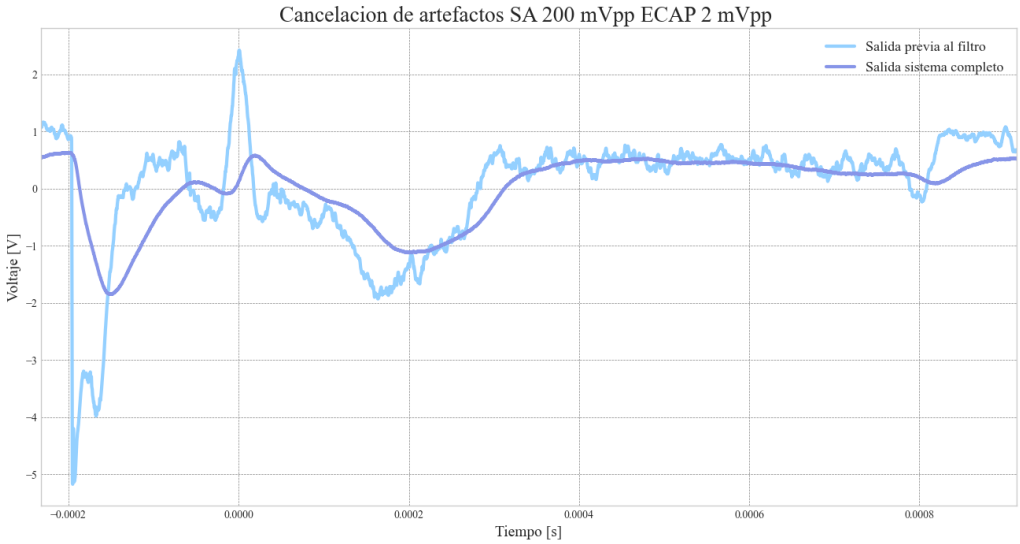


Figura 5.15: Salida del sistema con SA $200mVpp$ y ECAP $2mVpp$.

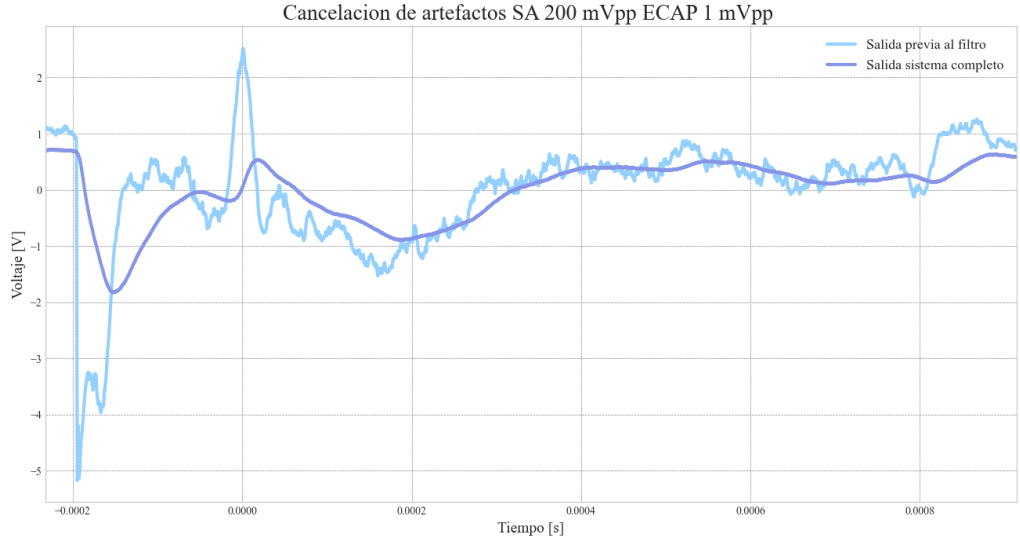


Figura 5.16: Salida del sistema con SA $200mVpp$ y ECAP $1mVpp$.

5.5.3. Detección de Cambios de Estimulación

Como se mencionó durante este capítulo, el FW debe ser capaz de percibir cuando la plantilla que está reproduciendo durante la fase tres deja de producir resultados aceptables. Por lo tanto se relevó la variación necesaria en la amplitud de la señal de SA para que el FW vuelva a fase dos. Para esto se utilizaron amplitudes de SA cercanos a los rangos especificados y para facilitar la comparación de los residuos no se incluyó ECAP en estas pruebas. Se comenzó por una SA de 60 $mVpp$, los resultados con esta entrada se pueden ver en la Fig.5.17. La señal violeta

representa la salida del sistema previo a que el mismo detecte que ha cambiado la estimulación y vuelva a fase dos. El resultado es bueno ya que con una variación del 5,7 % el FW es capaz de percibir el aumento del error en el residuo y volver a iterar sobre el mismo, incluso no se perdería información de la señal de estimulo ya que la saturación se da al inicio de la estimulación por un breve período de tiempo.

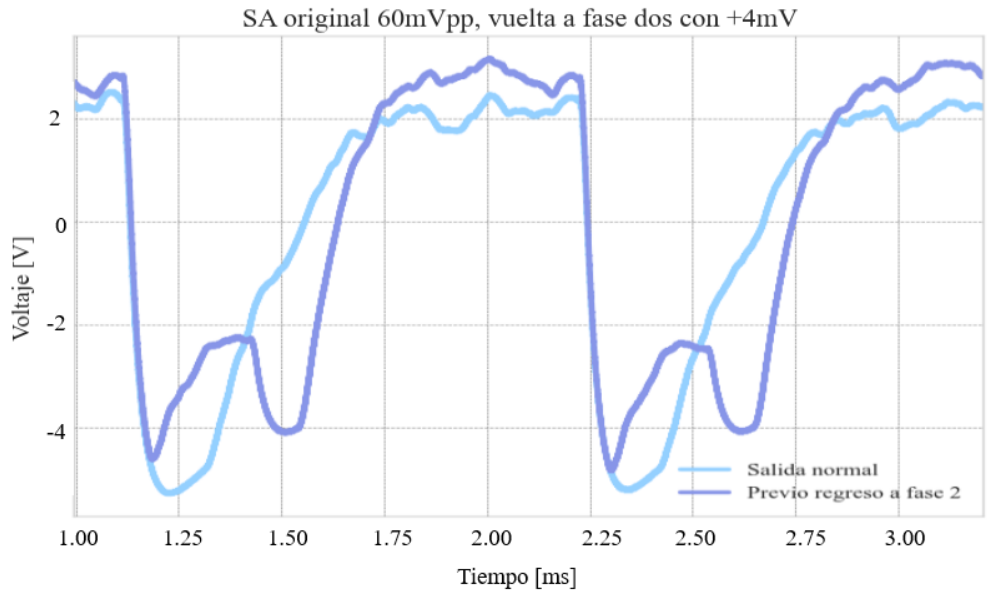


Figura 5.17: Cancelación con SA de $60mVpp$ y regreso a fase dos con SA de $64mVpp$.

Para la siguiente prueba se utilizó una SA original de $150mVpp$, en la Fig.5.18 se observan los resultados de la misma. Nuevamente en celeste se encuentra la salida luego de la convergencia, mientras que para este caso se obtuvo el regreso a la fase dos al llevar la SA original a $170mVpp$, representando esto un cambio de 13,3 %. Esta diferencia se debe a que el error base, siendo este el valor del error al momento de la convergencia (ver Secc.5.2.4), para este caso es mayor y por lo tanto se requiere mayor desviación para percibir el cambio en el residuo del artefacto. Esto es un defecto del sistema ya que la saturación en este momento es significativa y esta presente en el momento donde se espera la señal de ECAP, por lo que para este caso se perdería información de la misma.

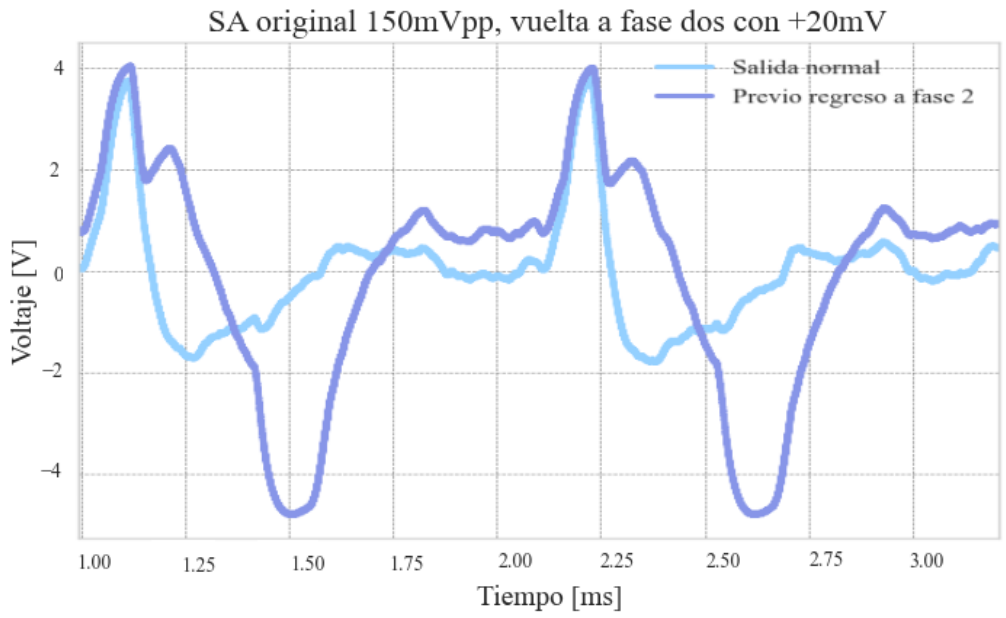


Figura 5.18: Cancelación con SA de $150mVpp$ y regreso a fase dos con SA de $170mVpp$.

Siguiendo esta tendencia para una señal original de $300mVpp$ el cambio necesario para que ocurra el regreso a fase dos es de $40mVpp$. En este caso la variación en porcentaje es igual al anterior, sin embargo en la Fig. 5.19 se puede ver que previo a volver a fase dos la salida estaba muy distorsionada, haciendo imposible la detección de la señal neural.

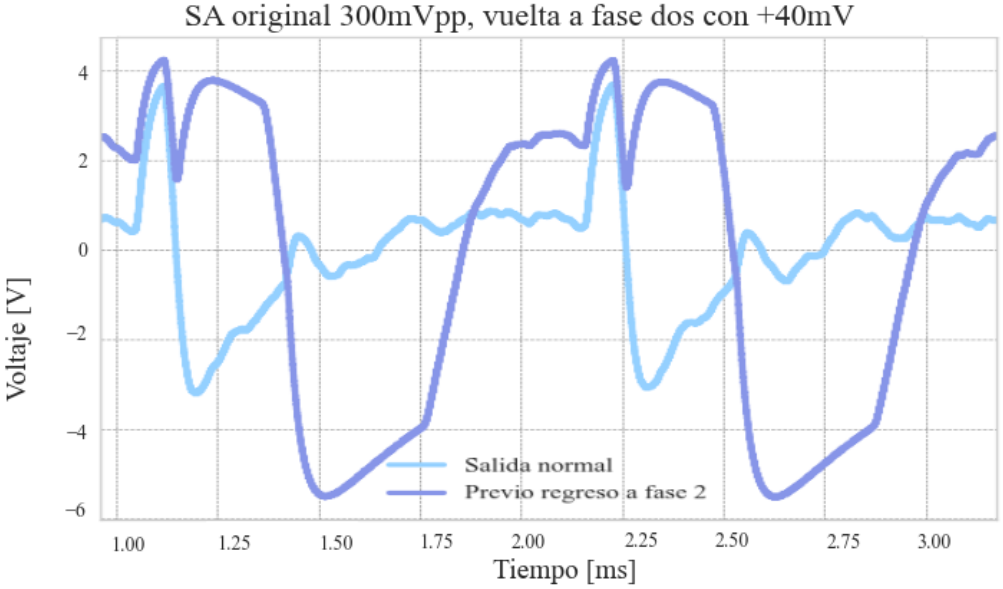


Figura 5.19: Cancelación con SA de $300mVpp$ y regreso a fase dos con SA de $340mVpp$..

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 6

Desarrollo de Software

6.1. Introducción

El subsistema de software realiza el post procesamiento de la señal obtenida a la salida del HW, es decir, la suma entre residuo amplificado resultante de la cancelación inicial del artefacto de estimulación y la señal neural subyacente. Tras el post procesamiento, se desea obtener esta última señal, que puede llegar a ser varios ordenes de magnitud menor que la primera.

Para lograr esto, es necesario realizar una segunda cancelación, donde la señal a cancelar es el residuo. La cancelación se dará de manera análoga a como se hizo en HW + FW. Se obtendrá una plantilla del residuo, para luego restarla en software a la señal de entrada y así obtener la señal neural.

Sin embargo, a diferencia del HW+FW, el SW cuenta con un obstáculo adicional. Debido a que el ECAP es una señal que se encuentra correlacionada con la SA, al ser ambas el producto de una estimulación en el tejido, si se genera una plantilla sobre una salida del HW que contenga la ECAP, esta va a pasar a formar parte de la plantilla del SW a pesar de que varios periodos sean muestreados. Por ende, al realizar la resta posterior, el ECAP también se verá cancelado. Notar que esto no sucede en el HW+FW ya que el ADC utilizado no cuenta con suficiente resolución para muestrear el ECAP. En cambio, a la entrada de SW se tiene el ECAP amplificado por lo que no funciona esta técnica, y de hecho se debe ser capaz de muestrearlo para su posterior procesamiento. Este procedimiento fue discutió previamente en la simulación en la Secc.3.4.7.

Para solucionar esto, se tomó ventaja del umbral de captura del tejido. En el caso de una aplicación real, al estimularse con una amplitud por debajo del umbral de captura (ver Secc 1.3), se generará un artefacto de estimulación sin la respuesta neural correspondiente. Con la salida del HW+FW resultante de cancelar este artefacto sub-captura, se generará la plantilla del SW.

Se imitará este comportamiento a partir de variar la amplitud de la señal SA generada a la entrada del sistema, con el objetivo de generar una adecuada plantilla de SW. Luego, se aumentará por un factor fijo de antemano, pre-acordado entre el SW y el FW, la amplitud de SA, definiendo la nueva amplitud como sobre el

umbral de captura, y se escalará también la plantilla del SW por el mismo factor. El valor por defecto es de $1,5$. Tras la nueva convergencia del HW+FW, se tendrá a la salida del mismo el residuo más el ECAP, las cuales el SW adquirirá, le substraerá su plantilla para obtener a la salida la señal neural subyacente.

En definitiva, el método utilizado para la confección de la plantilla de software, explota el comportamiento de la estimulación umbral para la generación del ECAP del tejido. En conjunto, se asume una linealidad global del sistema al escalar la plantilla de software por un factor adecuado, de forma que siga siendo capaz de cancelar el residuo de la SA a la entrada.

Para realizar esta tarea desde una computadora se plantean diferentes desafíos. El primero, consiste en la capacidad de adquirir la salida del HW con la suficiente precisión y velocidad deseadas. Cuanto mayor sea la velocidad de adquisición y la precisión, el remanente generado a partir de la segunda cancelación será menor, afectando directamente cuál es la señal neural de menor amplitud capaz de ser adquirida con el sistema en su conjunto. Esto requiere un instrumento de medida, como puede ser una tarjeta adquisidora de datos, que sea capaz de ser controlado desde una computadora y la integración del mismo se ajuste a las restricciones del proyecto en cuanto a tiempo y recursos.

El segundo desafío es la capacidad de procesamiento del sistema. Se deberán relevar datos a la mayor velocidad posible en períodos de 1,1 ms, lo cual significa que el sistema diseñado deberá ser capaz de coordinar la adquisición, procesar y mostrar los datos de manera continua bajo estrictas condiciones de tiempo.

Además, es necesario que la implementación del subsistema del post procesamiento permita la rápida configuración y modificación de sus variables para poder realizar diferentes pruebas y/o modificaciones de manera cómoda y continua. Esto permite realizar un fácil relevo del desempeño del sistema en su conjunto. Por último, al ser el sistema una prueba de concepto en la cual todas las señales son generadas de manera artificial con diferentes instrumentos, también se diseñó el sistema bajo el requerimiento de que contara con una plataforma centralizada que controle todos los instrumentos involucrados. Esto incluye cualquier generador de onda, fuentes o generador de señales digitales utilizado.

De esta manera, se diseñó un sistema de post procesamiento como una aplicación en Python basada en la división de las tareas en procesos, capaz de efectuar en paralelo la adquisición de datos, control de instrumentos, post procesamiento, generación de gráficos con los resultados y el control de una interfaz gráfica que le da al usuario la capacidad de configurar los parámetros del sistema. En la Fig. 6.1 se puede ver un resumido esquema de esta aplicación y de su interacción con los demás componentes del sistema CANE.

Para la adquisición y generación de señales tanto analógicas como digitales se utilizaron los instrumentos de *Digilent* [38] como pueden ser el *Analog Discovery 2* o *Analog Discovery Pro* [39]. Se decidió utilizar estas plataformas debido a varias razones, como pueden ser la familiaridad de los integrantes del proyecto, así como su fácil acceso, pero sobretodo, debido a las extensas y completas *Application Programming Interfaces* (APIs), tanto en python como en C++, provistas por el fabricante para el control de los instrumentos.

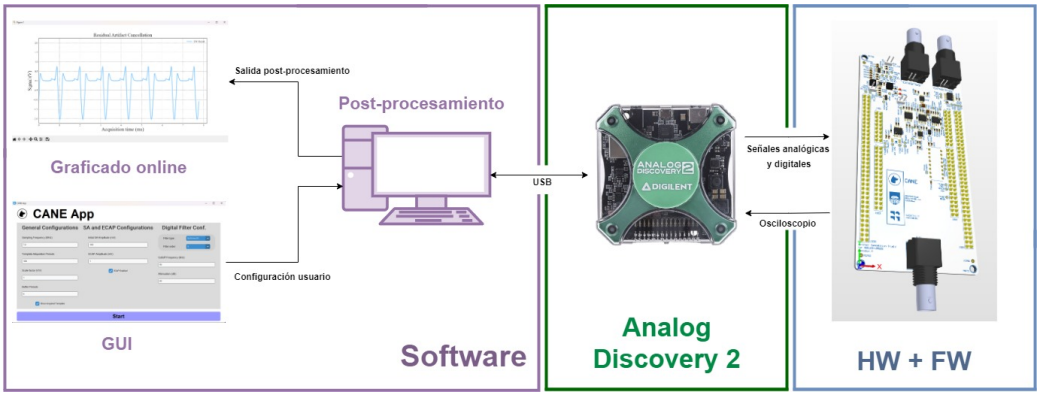


Figura 6.1: Diagrama de bloques de alto nivel del sistema de post procesamiento diseñado y su interacción con el resto de los componentes.

Se utilizó Python como lenguaje de desarrollo también debido a la familiaridad de los integrantes con el mismo, y por la extensa documentación encontrada en internet y los desafíos que implica su utilización en comparación a las alternativas existentes como pueden ser C++. A pesar de esto, se reconocen ciertas limitaciones de Python para nuestra aplicación, como puede ser su baja eficiencia y lentitud a la hora de llevar a cabo tareas de procesamiento que involucran grandes cantidades de datos. Estas desventajas se tuvieron en cuenta en el desarrollo de la aplicación y se mitigaron a partir de diferentes elecciones de diseño.

6.1.1. Funcionamiento

En esta sección describiremos el caso de uso general de nuestro sistema actuando en completo y todos los pasos que se deben seguir para obtener la señal neural a la salida. Un diagrama de flujo del funcionamiento se puede observar en la Fig. 6.2.

El primer paso consiste en configurar el *Analog Discovery 2* para generar una onda SA sub-umbral a la entrada del sistema. Luego, se espera a la convergencia del HW+FW. Esto será logrado a partir de la configuración del trigger del osciloscopio del AD2 con el flanco de subida de una señal digital sincronizada con la cancelación del artefacto, generada solamente cuando el FW se encuentra en fase 3, ver Secc5.2.4.

Se utilizarán los primeros X periodos adquiridos del residuo de la SA, donde X se encuentra definido por el usuario y por defecto es 100, para la generación de la plantilla de SW. Luego, se realizarán dos tareas, primero se le notificará al HW+FW que se obtuvo y se escaló la plantilla del SW. Esto se dará por medio de uno de los canales digitales del AD2 conectados a un pin del MCU. Desencadenando el escalado de la propia plantilla del FW, ver Secc5.2.4. Como segunda tarea, se aumentará la amplitud de la señal SA generada por el mismo factor y además se generará el ECAP con una amplitud determinada por el usuario.

Por último, el SW esperará a la nueva convergencia del HW+FW, y a cada periodo adquirido se le restará la plantilla escalada y los resultados serán mostrados

en pantalla como una gráfica en vivo.

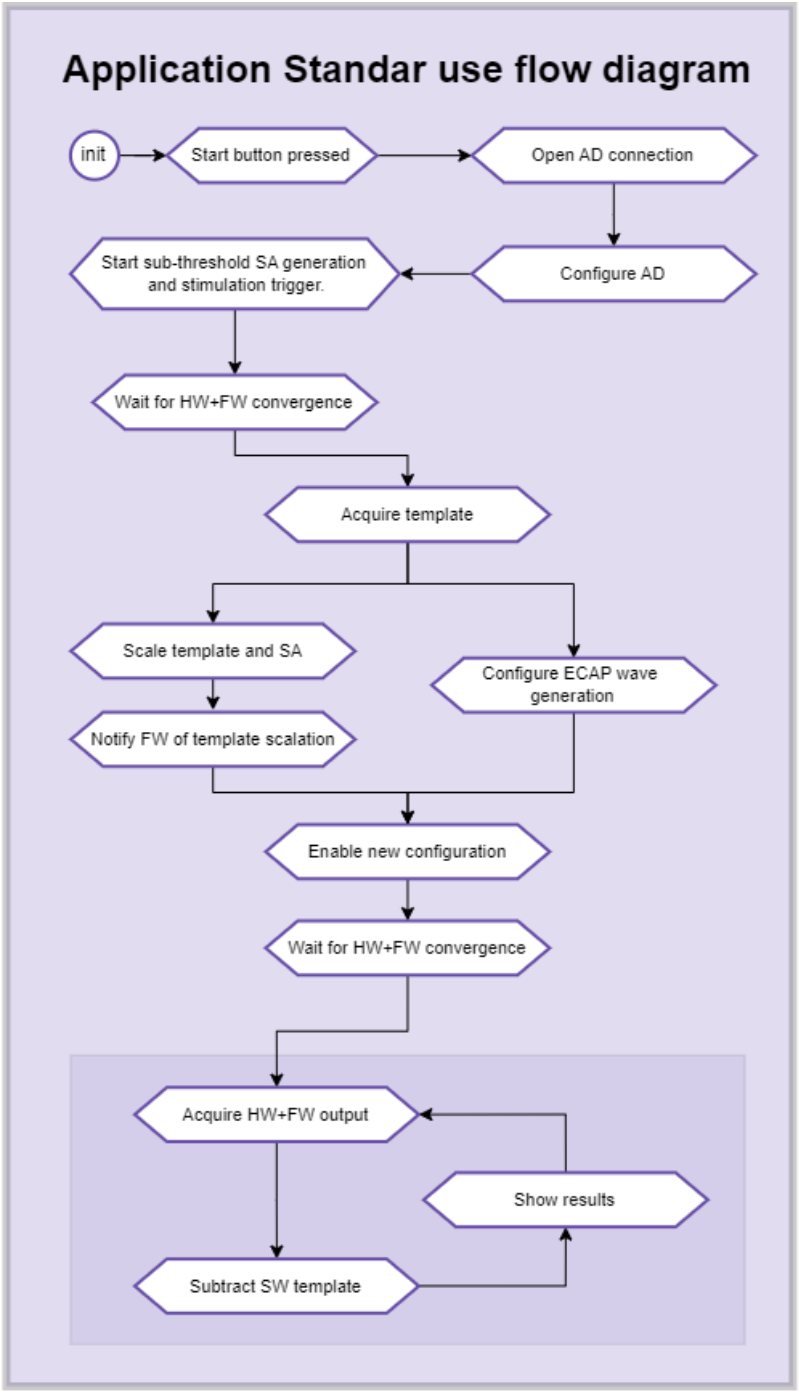


Figura 6.2: Diagrama de flujo de un caso estándar de uso de la aplicación para la obtención de la señal neural de interes a partir de la salida del HW+FW.

6.2. Diseño de la Arquitectura

Se pueden identificar cuatro tareas claves que la aplicación debe llevar a cabo:

- **Adquisición de datos**: consiste en la interacción con el dispositivo de *Digilent* para la obtención de la salida del módulo de HW + FW. No solo se debe configurar correctamente los canales del osciloscopio para el muestreo y disparo de la señal, sino también de todos los los generadores de ondas digitales y analógicas de las cuales depende el HW y FW para su correcto funcionamiento.

- **Post procesamiento**: es la tarea encargada de implementar la cancelación del residuo por medio de la creación de la plantilla, y la posterior resta de la misma a la entrada para obtener la señal neural subyacente. Además, se engloba toda otra tarea de procesamiento de la señal adquirida, como puede ser filtrado y substracción con la plantilla de SW.

- **Generación de gráficas**: se deben graficar en tiempo real los datos obtenidos del post procesamiento. Debido a la elección de usar Python, esta tarea es computacionalmente muy costosa y por ende puede llegar a ser el cuello de botella de la aplicación.

- **Interfaz de usuario (GUI)**: deberá existir una interfaz de usuario que exponga los parámetros del sistema para ser configurados.

- **Ejecución** : Para el correcto funcionamiento del sistema, se deben coordinar adecuadamente todas las tareas previamente mencionadas para sincronizar cualquier cambio o comando proveniente de la GUI hacia la ejecución del resto de las tareas.

Las primeras tres tareas descritas deben estar constantemente procesando datos, primero se adquieren los mismos, se los post-procesa y luego se grafica el resultado. Para mejorar la eficiencia del sistema, se implementó una arquitectura basada en la paralelización de las tareas antes mencionadas en procesos independientes. Esto es posible por medio de la librería *multiprocessing* [40] de Pyhton que le permite al programador usar diferentes procesadores en una computadora, realizando un mejor uso de los recursos del sistema, y así paralelizar la ejecución del programa. Se decidió utilizar *multiprocessing* por esta última característica, ya que otras librerías como *threading* [41] permiten la paralelización de tareas al generar diferentes hilos de ejecución pero sin crear diferentes procesos a nivel del procesador de la computadora.

Esta decisión generó nuevos desafíos a resolver, como la comunicación entre procesos, que permita el intercambio de grandes cantidades de datos y la comunicación y sincronización de los diferentes procesos. Es por ello, que se implementó la arquitectura mostrada en la Fig. 6.3.

En la Fig. 6.3 se puede apreciar un total de cinco procesos independientes que cumplen cada una de las tareas antes mencionadas. Estos son:

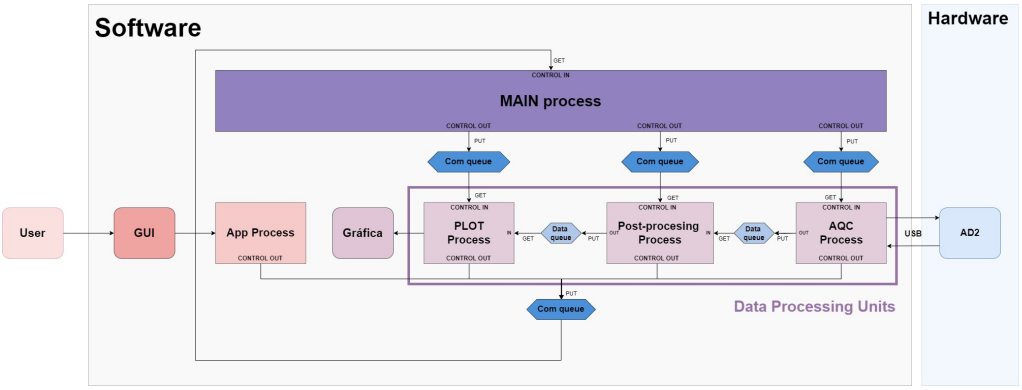


Figura 6.3: Diagrama de bloques de la arquitectura implementada para el post procesamiento. Se detallan todos procesos y su comunicación por medio de colas tanto de datos como de mensajes de control.

- ***Acquisition (ACQ) process***: se encarga de la tarea de adquisición de datos y comunicación por medio de USB con el AD2
- ***Post-processing process*** : realiza la tarea de post procesamiento y obtención de la plantilla en el software.
- ***Plotting (PLOT) process***: realiza la tarea de generación de gráficas de la señal obtenida.
- **App Process**: es el proceso encargado de generar y administrar la GUI así como de la comunicación de todo cambio en los parámetros configurados por el usuario.
- ***Main process*** : cumple el rol de orquestador del sistema al centralizar la comunicación de control entre procesos.

La comunicación entre procesos se implementó por medio de colas de diferentes largos. Encontramos dos tipos de colas, las de comunicación y las de datos.

Communication (com) queue. Son las colas utilizadas para la comunicación entre procesos a partir de mensajes de control predefinidos que generan en el receptor una respuesta predefinida. Todos los procesos cuentan con al menos una cola de comunicación. Además, el ***Main Process*** actúa como orquestador del sistema debido a que cualquier cola de comunicación fluye desde o hacia dicho proceso. De esta manera, un cambio en la GUI, que signifique la modificación de un parámetro en un instrumento del AD2, debe primero pasar por el ***Main Process*** para ser procesado y así poder coordinar con el resto de los procesos todos los cambios necesarios para hacer válida la nueva comunicación.

El sistema de comunicación entre procesos fue diseñado de esta manera por diferentes razones. La primera consiste en que tener un proceso centralizado para la comunicación reduce el número de colas necesarias. Si este no fuera el caso, cada proceso debería tener una cola de comunicación con cualquier otro, lo cual

significaría en nuestro caso doce colas en vez de cuatro. Además, en muchos casos, un cambio de configuración o acción que debe tomar el sistema ante cualquier otro estímulo pueden requerir una serie de pasos en un orden necesario. Tener un orquestador permite asegurar la correcta coordinación de todos los procesos para llevar a cabo las diferentes tareas.

Todas las colas de comunicación son del estilo FIFO, tiene un largo de 10 mensajes y son creadas a partir de la librería de Python *Queues* que se encuentra incluida en *multiprocessing*.

Data queues. Las colas de datos se utilizan exclusivamente entre los procesos englobados como ***Data Processing Units (DPU)*** y su único cometido es facilitar el pasaje de los datos entre cada uno de las tres DPUs. En este caso, no hay un tipo de dato predefinido de antemano y cuentan con largo de 1000 elementos.

Se pueden identificar tres tipos de procesos a partir de las colas utilizadas. Primero se cuenta con las ***Data Processing Units*** que cuentan todas con dos colas de datos y dos colas de comunicación, dos de entrada y dos de salida, el detalle de estas unidades se verá en la Secc. 6.4.3. En este caso, se cuenta con tres DPUs, con las cuales se implementa la cadena de procesamiento; desde los datos obtenidos a partir de AD2, hasta la generación de la gráfica de la señal de salida.

Luego, se cuenta con el **App Process** que solo cuenta con una cola de comunicación de salida. Este proceso no cuenta con una cola de comunicación de entrada debido a que la implementación de la GUI a partir de la librería de Python utilizada dificulta su implementación. Por último, el ***Main Process*** solo cuenta con colas de comunicación entrantes de todos los procesos restantes y de salida hacia las tres DPUs.

6.2.1. Diagrama de Módulos

Para implementar la arquitectura propuesta en la sección anterior se dividieron las funcionalidades descritas en diferentes módulos. Un diagrama de los mismos se puede ver en la Fig. 6.4. Los módulos son clasificados en las siguientes categorías:

- ***Front-end***: denota la parte de la aplicación con la cual el usuario interactúa directamente, en este caso, consiste en la interfaz gráfica de usuario.
- ***Middle-ware***: Denota el software que se encuentra entre el front-end y el back-end. En la aplicación, el middleware será donde todos los datos son adquiridos, procesados, y los resultados generados.
- ***Back-end***: Es la parte de la aplicación que no cuenta con interacción directa con el usuario y en el caso particular de esta aplicación cumple el rol de abstracción de lo instrumentos utilizados, exponiendo sus funciones al middleware para que sean utilizados.

Dentro de cada categoría se cuenta con diferentes módulos que serán desarrollados a continuación, empezando con el ***Back-End***.

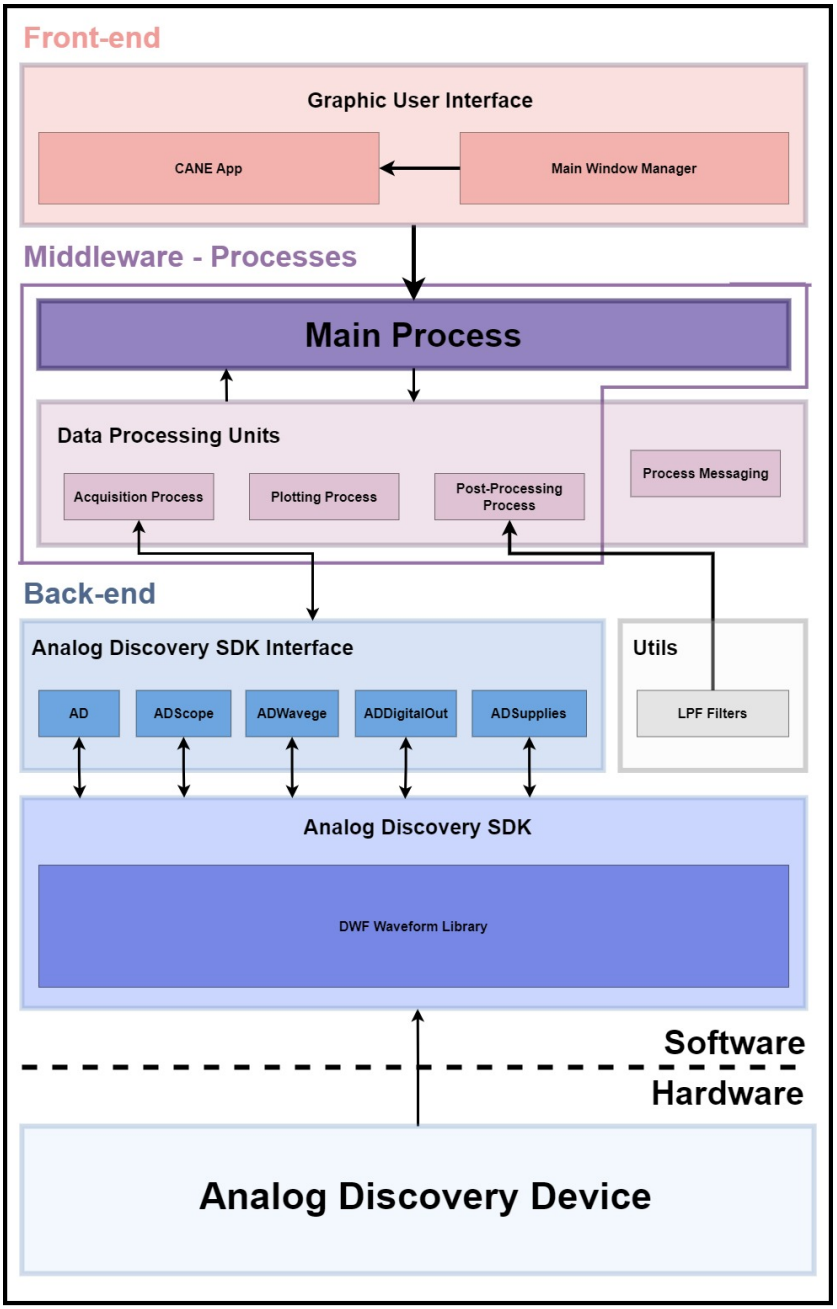


Figura 6.4: Diagrama de módulos de la aplicación de post procesamiento desarrollada.

Diagrama de relación de clases

En la Fig. 6.5 se puede apreciar un diagrama UML de relación de clases que engloba todo módulo diseñado durante el desarrollo de la aplicación de post procesamiento.

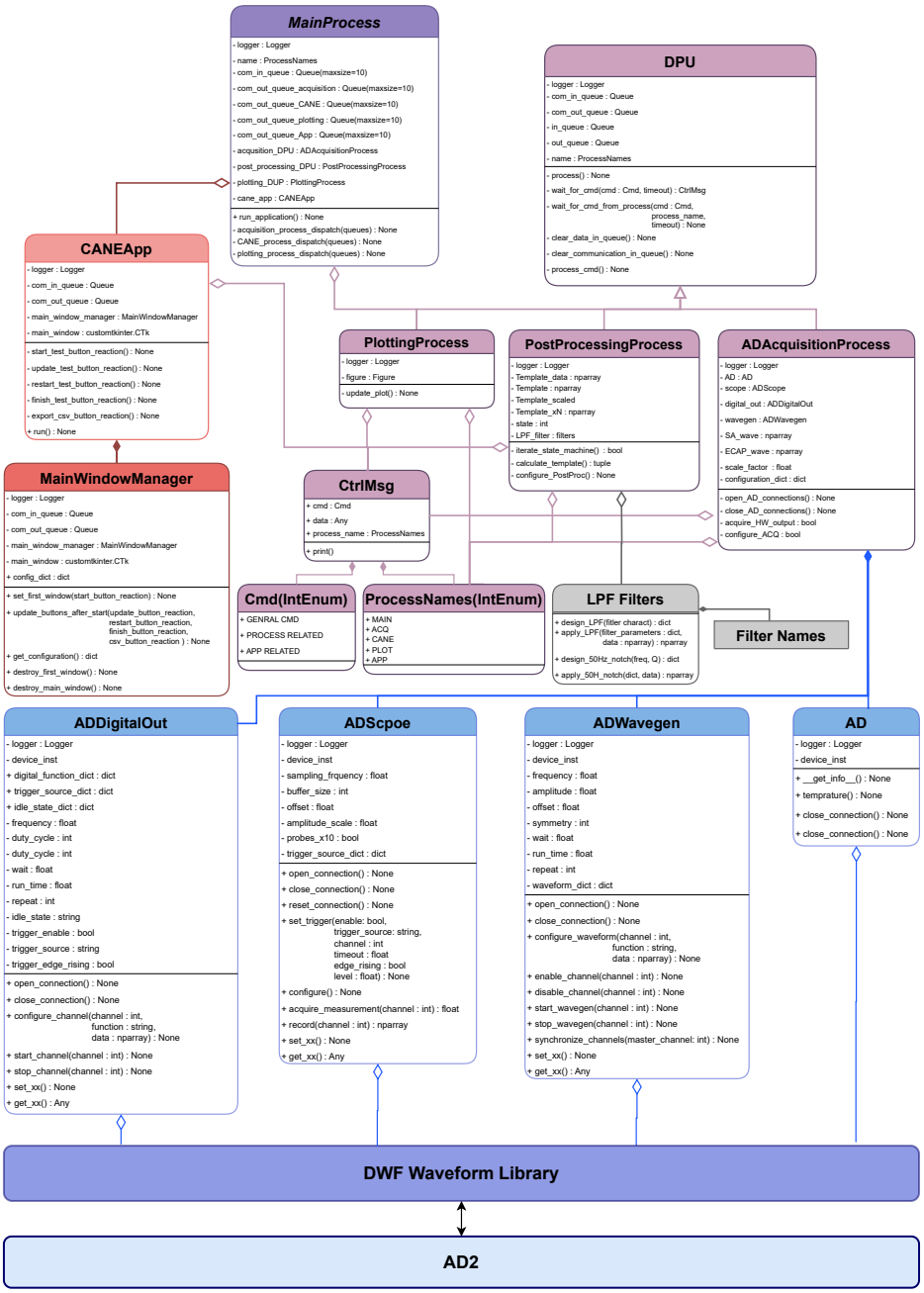


Figura 6.5: Diagrama UML de relación de clases del sistema de post procesamiento

6.3. Back-End

Interfaz Analog Discovery

Todo el sistema de cancelación de artefactos recae en el correcto uso del *Analog Discovery 2* tanto sea como generador de señales analógicas/digitales y como os-

ciloscopio para poder implementar el algoritmo de post procesamiento. Por ende, es clave asegurar un robusto uso del instrumento.

Digilent provee una librería de funciones, la ***DWF Library*** [42], que permiten configurar y acceder a cada uno de los canales de cada instrumento del Analog Discovery 2 de manera independiente. Sobre esta librería, se desarrolló una interfaz para abstraer los servicios expuestos por la API de *Digilent*. Implementar una interfaz es de suma importancia: permite la abstracción de todas las funcionalidades presentes en la API para exponer hacia la aplicación solamente los servicios que sean necesarios; asegura la integridad de la API, pues evita su modificación desde varios procesos independientes; y además, implementar una interfaz permite generar una representación virtual del instrumento en el software que facilite su uso.

La interfaz expone solamente los instrumentos del AD2 que son necesarios para nuestra aplicación. Estos son:

- Osciloscopio con todos sus canales y trigger configurable.
- Generador de ondas predefinidas y arbitrarias.
- Generador de ondas digitales.
- Fuentes de alimentación de hasta $\pm 5\,\mathrm{V}$.

En la aplicación final, todos los instrumentos mencionados fueron utilizados menos las fuentes. La interfaz de cada uno se basa en una clase de Python en un archivo independiente, como se puede ver en la Tab. 6.1, pero que mantienen ciertas similitudes en su implementación. Cada interfaz cuenta con los siguientes métodos que deben ser usados para el correcto uso del mismo:

- ***open_connection()*** : inicializa y establece la comunicación con el AD2, efectuando cualquier configuración necesaria específica del instrumento.
- ***close_connection()***: da por finalizada la conexión con el instrumento, cerrando la comunicación.

Cabe mencionar, que la interfaz diseñada parte del trabajo realizado por *Digilent* en el proyecto *WaveForms-SDK-Getting-Started-PY* [43] en Github. De este se tomó la forma de manejar el AD2 por medio de las funciones de la ***DWF Library*** pero se refactorizó el código para que las funcionalidades deseadas sean expuestas de la manera deseada. También, se agregaron nuevas features que no se encontraban antes, como pueden el sincronizado de los canales analógicos entre ellos o más opciones para el sincronizado de los canales digitales. En el anexo F.1 se puede observar con más detalles cada una de la interfaces mencionadas antes así como los parámetros expuestos para la configuración de los instrumentos por la misma. Luego, en el anexoF.2 se encuentra la configuración realizada por la aplicación de cada uno de los instrumentos.

Instrumento	Archivo	Clase
Instrumento Analog Discovery	AD.py	AD
Oscioloscopio	ADScope.py	ADScope
Generador de onda analógico	ADWavegen.py	ADWavegen
Generador de onda digital	ADDigitalOut.py	ADDigitalOut
Fuentes $\pm 5\,\mathrm{V}$	ADSupplies.py	ADSupplies

Tabla 6.1: Archivos e interfaces para cada uno de los instrumentos utilizados por la aplicación de post procesamiento

Filtros digitales

Durante el desarrollo de la aplicación del post procesamiento, se observó que la salida del sistema contaba con ruido de alta frecuencia resultante de la cancelación del residuo. Debido a esto, se implementó una serie de filtros pasa-bajos digitales que pueden ser usados durante el post procesamiento y configurados de forma dinámica.

Para ello, se creó una interfaz basada en dos funciones:

- ***design_LPF(filter_type, characteristics)*** : que a partir del filtro digital elegido y de las características deseadas del filtro, genera un diccionario con todos los parámetros que definen el filtro elegido.
- ***apply_LPF(diccionario_parametros, filter_type, data)***: utiliza el filtro seleccionado y diseñado a partir de los parámetros presentes en el diccionario para filtrar los datos también pasados como parámetros al método.

Esta arquitectura de la interfaz permite poder diseñar una sola vez el filtro y luego aplicarlo cuando sea necesario. Si se desea modificar el mismo, basta cambiar el diccionario de parámetros. Todos los filtros fueron implementados usando las librerías *numpy* [44] y *scipy* [45] de Python. Los filtros posibles son:

- Ninguno
- Moving average
- Butterworth
- Chebyshev
- FIR
- Filtro en el dominio de la frecuencia.

Además, se añadió a este módulo un filtro Notch para el filtrado del ruido introducido por la red eléctrica en los 50 Hz. El uso del mismo es análogo al de los filtros pasabajos, basándose en las siguientes dos funciones:

- ***design_50Hz_notch(sampling_frequency, Q)*** : obtiene los párametros del filtro Notch diseñado a partir del factor de calidad Q elegido así como la frecuencia de muestreo utilizada para obtener los datos sobre los cuales se aplicacará el filtro.
- ***apply_50Hz_notch(diccionario_parametros, data)***: a partir de los parámetros encontrados en el diccionario, le aplica a los datos encontrados en el buffer ***data*** el filtro Notch correspondiente.

6.4. Procesos - Middleware

El Middleware está compuesto por los procesos que llevan a cabo las grandes tareas del post procesamiento como se describió en la sección 6.2. En esta sección se explorará cómo logra cada proceso realizar su tarea y cómo se da la coordinación general de todos los procesos entre sí.

6.4.1. Mensajería de Comunicación Entre Procesos

Para organizar la comunicación entre procesos se diseñó la clase ***CtrlMsg*** como se puede ver en el diagrama UML de la Fig. 6.5 , ubicada en el archivo ***ProcessMessaging.py***. La clase define el tipo de dato a ser enviado en las colas de comunicación y cuenta con tres parámetros.

- ***cmd***: tipo de comando a enviar y puede ser solo uno de los definidos en el enumerado de enteros bajo el nombre ***Cmd***. Hay un total 12 comandos definidos:
-
  - START
  - STOP
  - CONTINUE
  - RESTART
  - ERROR
  - FINISH
  - READY
  - UPDATE
  - PROCESS_CLOSED
  - TEMP_ADQUIRED
  - EXPORT_CSV

  - START_BUTTON
  - RESTART_BUTTON
  - UPDATE_BUTTON
  - EXPORT_CSV_BUTTON

- ***process_name***: Es el nombre del proceso en el cual se origina el mensaje y puede ser cualquiera de los valores definidos en el enumerado de enteros con nombre ***ProcessNames***:

  - MAIN
  - ACQ
  - CANE
  - PLOT
  - APP

- ***data***: datos enviados con el mensaje que pueden ser de cualquier tipo.

La comunicación entre procesos solo puede pasar por medio de un menaje encolado con la estructura definida por la clase ***CtrlMsg*** .

6.4.2. Main Process

El proceso principal que se encuentra en el archivo *MainProcess.py* es el centro de nuestro programa y el orquestador del post procesamiento. El mismo se asegura que todas las tareas sean cumplidas, sincroniza los diferentes eventos y maneja cualquier error surgido en alguno de los procesos. Además, es el encargado de crear todas las colas de datos y comunicación utilizadas, así como adecuadamente inicializar todos los procesos.

En la Fig 6.6 se puede observar un diagrama de flujo de la operación del *Main Process*. Tras la inicialización y la recepción del mensaje de control nombrado *START_BUTTON* desde la interfaz gráfica, el proceso entra en un loop donde verifica constantemente si la cola de comunicación de entrada cuenta con un mensaje de control. En el caso afirmativo, lo procesa y realiza las acciones necesarias, generando en el resto de los procesos los eventos adecuados cómo pueden ser una actualización de los parámetros usando el mensaje de control *UPDATE* o terminar un proceso con el mensaje *STOP*.

6.4.3. Data Processing Units

Las ***Data Processing Units*** son procesos especiales que cuentan con, además de las colas de comunicación, dos colas de datos, una de entrada y otra de salida. De esta manera, se pueden encadenar diferentes DPUs formando una cadena de procesamiento en la cual cada bloque cumple un rol en específico. Todas las DPU comparten una estructura común compuesta por ciertos métodos y variables, por

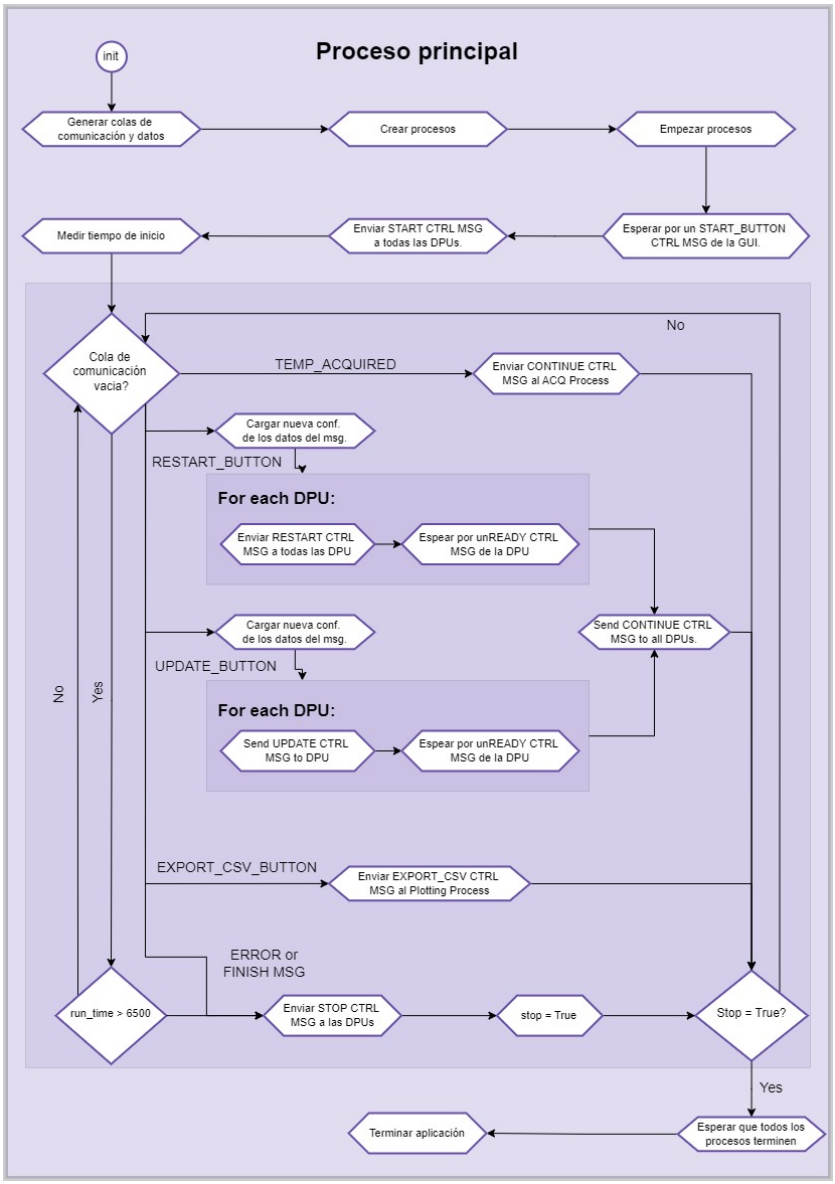


Figura 6.6: Diagrama de flujo para el proceso principal de la aplicación de post procesamiento.

lo cual se creó la clase ***DPU*** en el archivo **DPU.py**. Define las colas de datos y comunicación, así como métodos útiles para vaciar las colas de entrada o bloquear la ejecución esperando por un mensaje de control de ser necesario.

Todas las DPU descritas en la sección 6.2 heredan esta estructura general de la clase DPU y serán descritas a continuación.

6.4.4. Proceso de Adquisición de Datos

Este proceso se encarga de la configuración del AD2 y la adquisición de datos por medio de la interfaz del osciloscopio para poder encolarlos en la cola de salida

para que sean post-procesados.

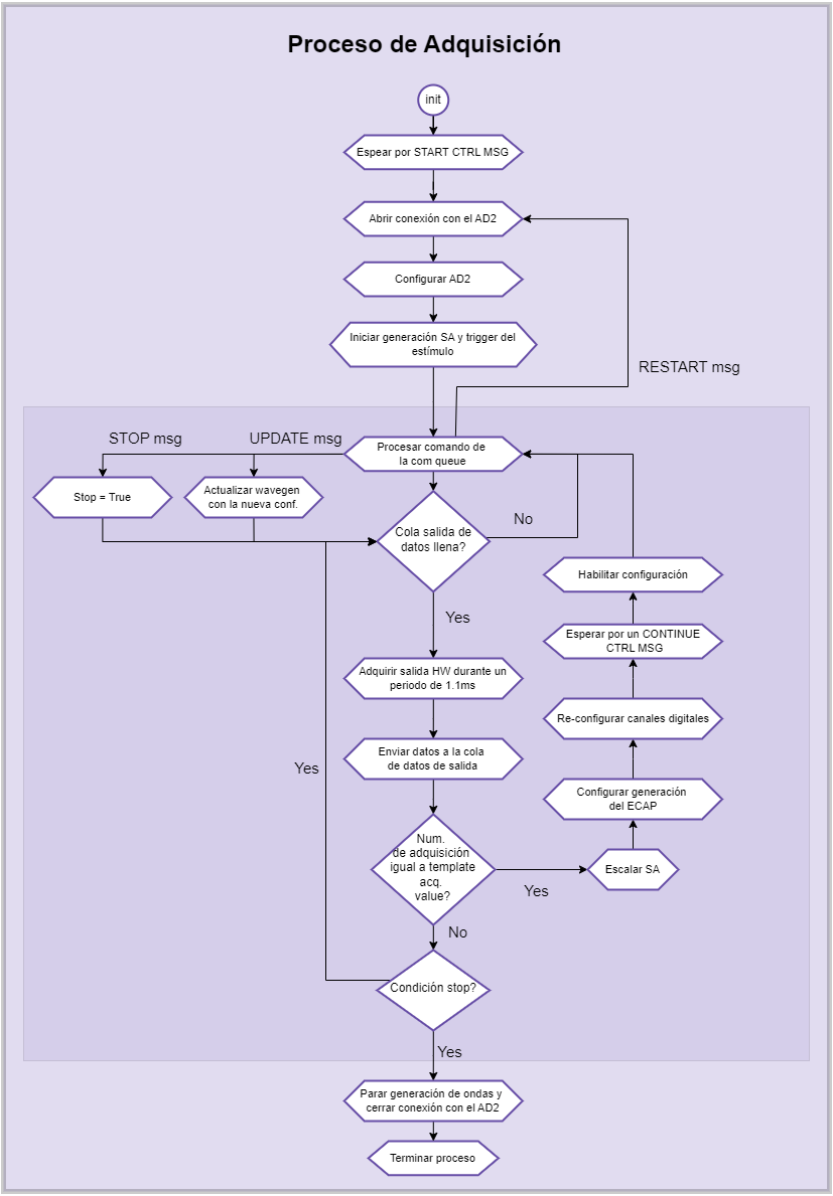


Figura 6.7: Diagrama de flujo para el proceso de adquisición de datos.

El proceso de adquisición configura los instrumentos del AD2. Para el generador de ondas analógicas, la señal SA es configurada con un periodo de 900 Hz en en canal 1 y en el canal 2 se configura la generación del ECAP [1] con el mismo período y sincronizado con el canal 1. Luego, se configura el canal digital 0 como un generador de onda cuadrada con un período de 900 Hz cuyo flanco de subida se encuentra sincronizado con el inicio de la generación de la onda SA en el canal

[1] La ECAP solo es generada luego de que el SW haya obtenido su plantilla.

analógico 1. Además, luego de adquirir la plantilla de software, se configura el canal digital 1 para dar un flanco de subida para poder notificar al FW que debe escalar su propia plantilla.

Por último, el software configura el osciloscopio 1 para muestrear una señal de amplitud máxima $\pm 5\,\mathrm{V}$. El trigger se configura en el canal dos con un nivel de $1{,}65\,\mathrm{V}$ y se encuentra conectado a un GPIO del MCU que da un flanco de subida cada vez que se empieza una cancelación de artefacto en fase 3 en el FW. La configuración completa de todos lo parámetros de los instrumentos del AD2 se puede ver en el anexo F.2.

En la Fig. 6.7 se puede observar un diagrama de flujo de este proceso. Tras la inicialización del AD2 se realizan dos tareas. Primero se verifica si hay un mensaje encolado en la cola de comunicación de entrada. Si es el caso, este es procesado. Independientemente, luego se adquiere una nueva muestra siempre y cuando la cola de salida no esté llena. Cuando se alcanza el número de períodos N configurado por el usuario, la adquisición es pausada, se reconfigura el AD2 para generar la ECAP y la señal en el canal digital 1 antes mencionada. Cuando se recibe un mensaje de control CONTINUE, estas nuevas configuraciones son habilitadas en el AD2 y se vuelve adquirir períodos ininterrumpidamente.

6.4.5. Proceso de Post Procesamiento

Este proceso se encarga de obtener la plantilla del SW y de aplicar el post procesamiento luego. Por ende encontramos dos estados muy diferentes de este proceso, previo a la adquisición de la plantilla y luego de ésta, donde se deben realizar tareas diferentes.

Para esto, una máquina de estados fue diseñada dentro del proceso. La misma cuenta con dos estados para representar este conjunto de tareas diferentes que se deben llevar a cabo. Un diagrama de la máquina de estados se puede ver en la Fig. 6.9.

Un diagrama de flujo se puede apreciar en la Fig. 6.8. Tras el inicio del proceso y al recibir un mensaje de control START por parte del proceso principal. Luego se da la configuración del post procesamiento donde se obtienen los parámetros que definen el filtro pasa bajos a ser aplicado sobre los datos y se inicia la máquina de estados. A continuación, se procesa cualquier comando de control recibido, como por ejemplo un mensaje de *restart* para volver al estado inicial, y se itera una vez la maquina de estados del post procesamiento. Dicha iteración comprende cómo procesar un elemento[2] en la cola de entrada dependiendo del estado. Las tareas en cada estado se muestran en la Fig. 6.9.

Al iniciar el proceso de post procesamiento, se comienza en el primer estado llamado ***Template Acquisition state***, en donde al recibir un elemento de la cola de entrada, se guarda en un arreglo bidimensional de tamaño $N \times M$ donde N es el número de arreglos a guardar, siendo un valor definido por el usuario. M, es largo del arreglo que depende de la velocidad de muestreo del osciloscopio.

[2]Recordar que un elemento en la cola se refiere a un arreglo de datos correspondientes a un período de estimulación

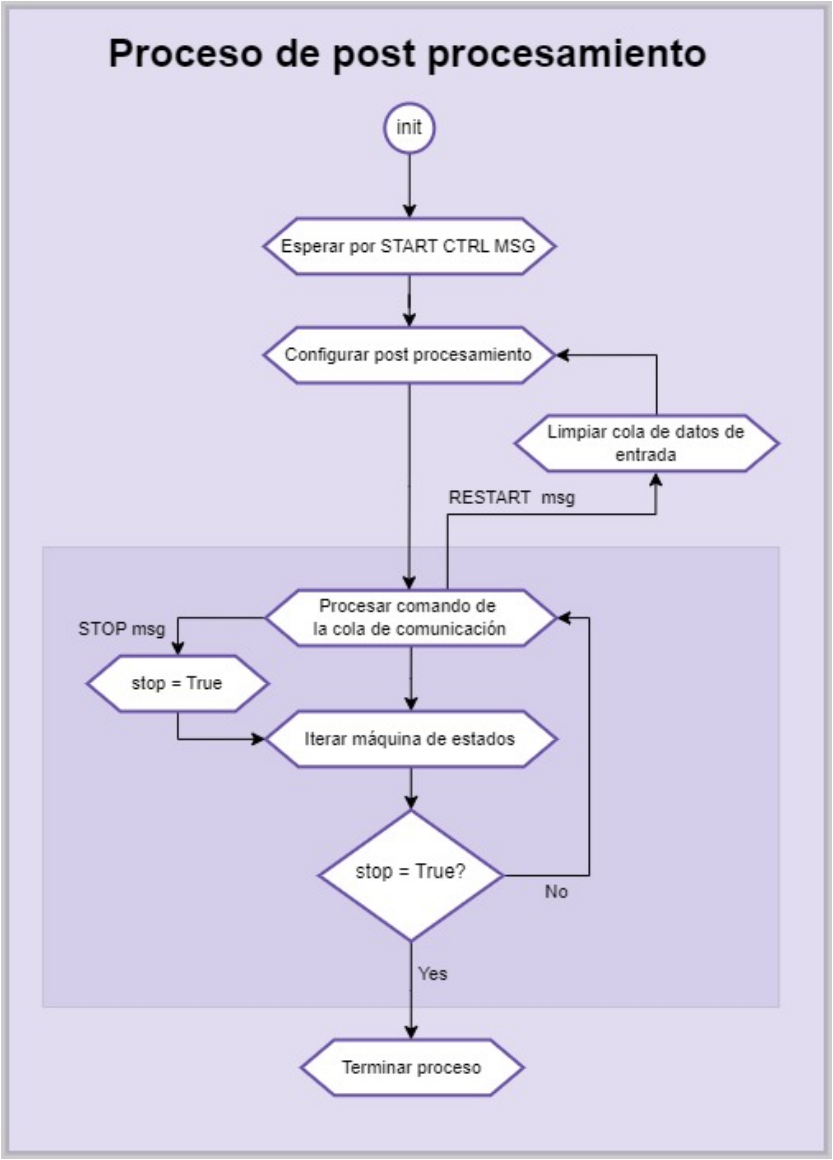


Figura 6.8: Diagrama de flujo para el proceso de post procesamiento.

Cuando se completan las N filas, se promedia los N valores de cada una de las M columnas para obtener un nuevo arreglo de largo M que será nuestra plantilla de SW. Cuando este proceso es finalizado, un mensaje de control es enviado al proceso principal y se pasa al siguiente estado, el ***Steady state***.

En ***Steady state***, cada vez que un nuevo elemento es encolado, este es obtenido y concatenado al final de un buffer. Luego de k, veces se obtiene un arreglo de largo $M \times k$, donde k es un valor definido por el usuario. Luego, se le resta la plantilla obtenida anteriormente[3], se filtran los datos con el LPF digital seleccionado por el

[3]Para realizar adecuadamente la resta se genera una nueva plantilla de largo $M \times k$ compuesta por la plantilla inicial k veces concatenada

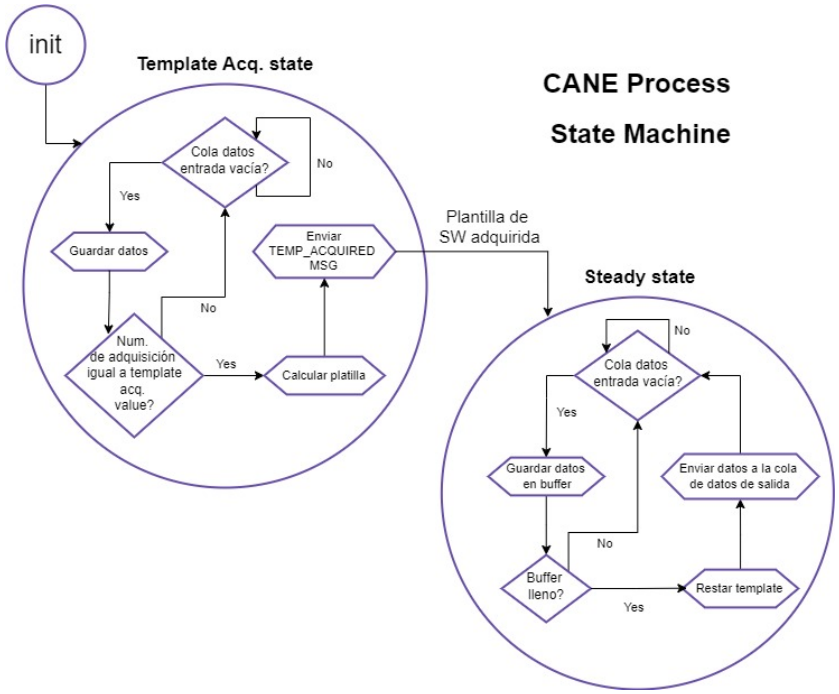


Figura 6.9: Máquina de estados incluida internamente en el proceso de post procesamiento.

usuario, y se encola el resultado de la resta, entre la señal adquirida y la de SW, en la cola de datos de salida para que el siguiente proceso los pueda graficar. La decisión de anidar k datos de entrada fue realizada con el propósito de no saturar el último proceso de la cadena.

6.4.6. Proceso de *Plotting*

La única tarea de este proceso es graficar, con la librería *matplotlib* [46] de python, todos los datos que son encolados por el proceso de post procesado. La Fig. 6.10 muestra el diagrama de flujo de este proceso que es muy similar a los anteriores. La única diferencia a resaltar es que, tras la actualización de la gráfica para mostrar los datos más recientes, se realiza un chequeo de si está fue cerrada por el usuario, lo cual, en el caso afirmativo, se abre nuevamente la gráfica para continuar graficando los datos.

Durante el diseño, se descubrió que este proceso es le cuello de botella del sistema debido a la retraso que introduce la actualización de la gráfica generada por matplotlib. Tras unos pocos segundos, la cola de datos de entrada del proceso se llena, generando que el proceso de post procesamiento no procese ninguno de sus datos encolados debido a que su cola de salida se encuentra llena. Para obtener una medida de la eficiencia del sistema, se ejecuto la aplicación durante 300 s y se midió la cantidad de periodos de estimulación que fueron graficados por el proceso. Estos períodos son de 1,1 ms, correspondientes a una frecuencia de estimulación de 900 Hz.

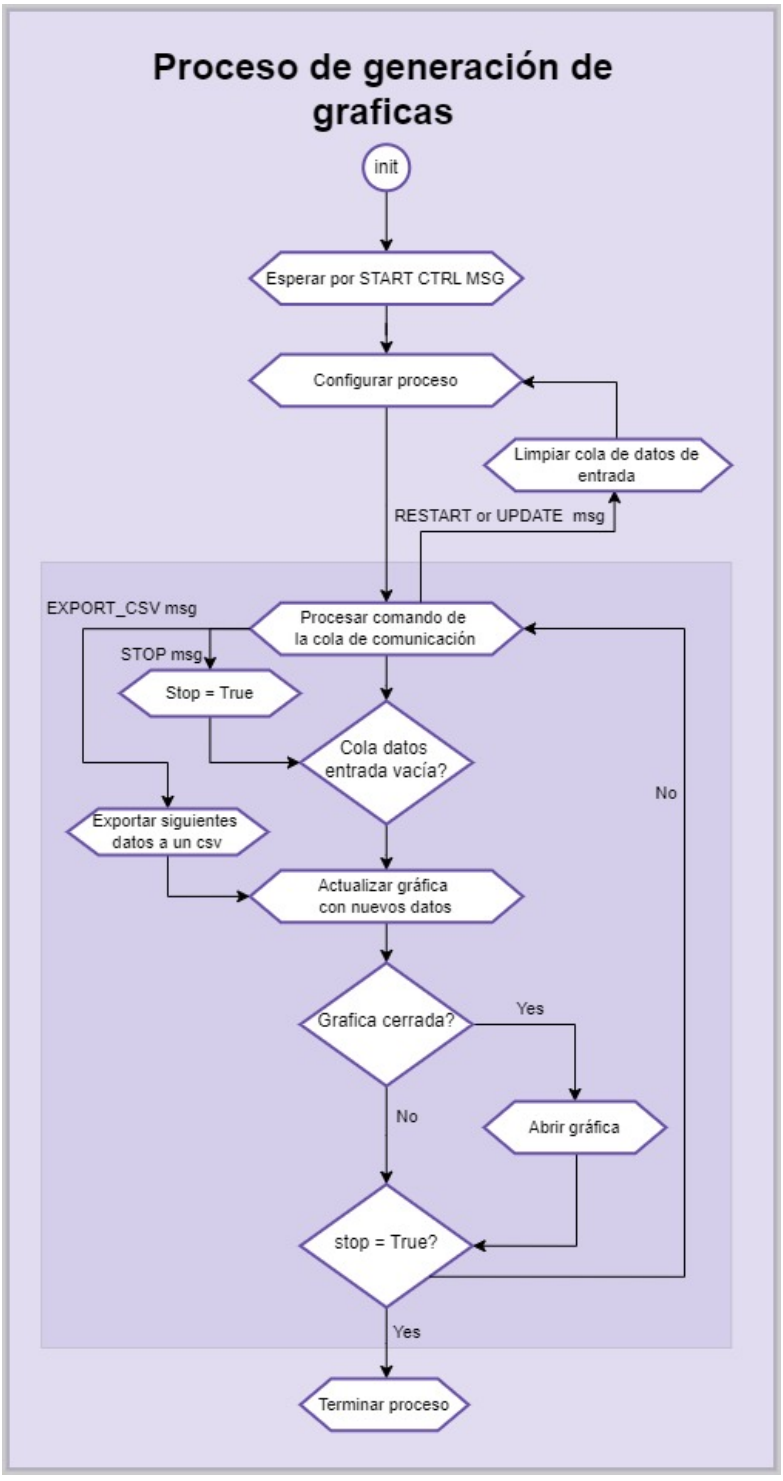


Figura 6.10: Diagrama de flujo para el proceso de generación de gráficas del resultado del post procesamiento.

El resultado fue un total de 45 555 periodos, lo cual representa solo el 16.9 %

de la cantidad total de periodos de 1,1 ms sucedidos durante 300 s. Esta medida fue tomada con un número de periodos graficados de 5, valor configurado desde la interfaz de usuario que se verá a continuación. Sin embargo cuanto mayor es este número, aumenta el desempeño del sistema. Por ejemplo, al aumentarse el número de periodos graficados a 15, para el mismo tiempo se grafican un total de 68 250 periodos, lo cual equivale al 25.3 % del número de periodos totales.

6.5. Interfaz Gráfica de Usuario - Front-End

El último proceso de nuestra aplicación es la interfaz gráfica de usuario o GUI que fue diseñada usando la librería *customtinker* [47] de python. La interfaz fue creada como una herramienta para los integrantes del proyecto, así como los tutores o integrantes del tribunal, con el propósito de facilitar la configuración y uso del sistema en su completitud.

La imagen de la Fig. 6.11 muestra una imagen de la interfaz diseñada y de los posibles parámetros a ser configurados.

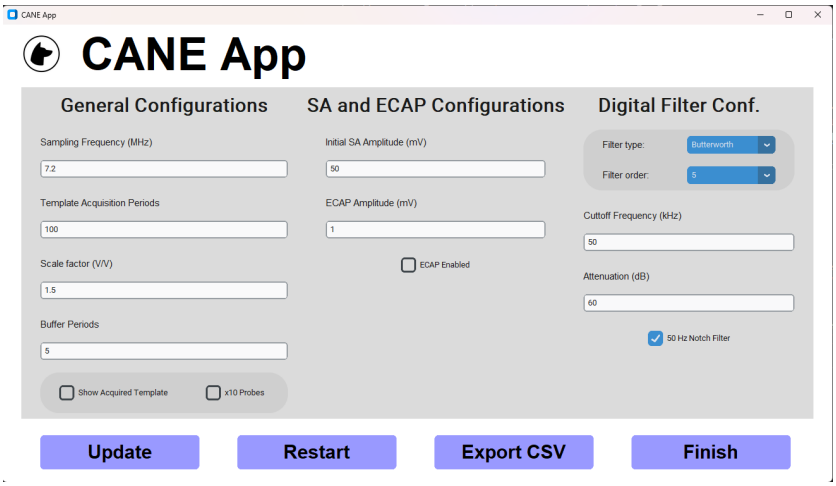


Figura 6.11: Interfaz gráfica de usuario de configuración diseñada como parta de la aplicación de post procesamiento.

En la GUI cuenta con tres categorías de parámetros. La primera consiste en parámetros generales como la frecuencia de muestreo del osciloscopio, el número de periodos de estimulación tomados para calcular la plantilla, el factor de escala utilizado para re-escalar la plantilla[4](introducido en la Secc.2.3.2) y el número de períodos de estimulación a mostrar en la gráfica de salida. Además, bajo esta categoría, se tiene la opción de que el SW muestre la plantilla cuando esta es generada, así como de habilitar la opción para las puntas de osciloscopio $\times 10$. Cuando esta opción está habilitada, todo dato adquirido por el AD2 es multiplicado por dicho factor.

[4]Este debe ser el mismo que el configurado por el FW

La segunda categoría engloba las amplitudes de las señales analógicas a utilizar que son en este caso la SA y el ECAP, así como una opción para habilitar o deshabilitar el ECAP. Por último, se cuenta con la configuración del filtro LPF aplicado durante el post procesamiento. Cuenta con un menú de selección para el tipo de filtro a ser usado, el orden, la frecuencia de corte y la atenuación del mismo. Notar que no todas las configuraciones son válidas y no todos los parámetros son utilizados por todos los filtros. Por ejemplo, el filtro *Moving Average*, no utiliza ninguno de los otros parámetros. El último parámetro corresponde a la habilitación de un filtro Notch de 50 Hz para todo acople que se produzca debido a la red eléctrica.

Luego, la interfaz también cuenta con cuatro botones. El botón de ***Update*** actualiza algunos parámetros sin reiniciar el post procesamiento, por lo cual no se actualizara la plantilla que se esta actualmente usando. Además, este botón no actualiza la velocidad de muestreo, el factor de escalado o la cantidad de periodos para la adquisición de la plantilla. El botón de ***Restart*** actualiza todos los parámetros por los valores actuales y reinicia el post procesamiento, esto puede ser útil cuando se quiere poner a prueba la vuelta a fase dos del FW (ver Cap.5). Es decir, todas las colas de datos son vaciadas, el AD2 reconfigurado y la plantilla re-calculada a partir de la nueva salida de HW como resultado de las nuevas señales generadas. Accionar este botón equivale a cerrar la aplicación y ejecutarla devuelta. El botón ***Export CSV*** provoca un evento tal que los próximos datos correspondientes a la salida del SW se exporten y guarden en un archivo CSV. Por último, el botón de ***Finish*** finaliza la aplicación, terminando todos los procesos.

Todas estas acciones son realizadas por medio del encolado de mensajes de control, detallados en la Secc.6.4.1, que cuentan con el sufijo **_BUTTON**. Estos mensajes son enviados por el proceso de la aplicación hacia el proceso principal en donde la nueva configuración de parámetros, si corresponde, viaja como parte de los datos del mensaje de control.

6.5.1. Guía de Uso de la Interfaz

El uso de la interfaz gráfica es sumamente sencillo, al iniciar el programa a partir de la ejecución del script ***PostProcessing.py***, aparece la única ventana, como muestra la Fig 6.12. La configuración por defecto ya se encuentra cargada en las diferentes burbujas de texto. El usuario, tras realizar la configuración deseada, debe presionar el botón ***Start*** para que el post procesamiento sea iniciado. Cuando esto sucede, la ventana es actualizada a la mostrada en la Fig. 6.11.

Si la opción de ***Show Acquired Template*** se encontraba seleccionada, aparecerá una gráfica como se observa en la Fig. 6.13, mostrando la plantilla de SW obtenida, y la misma tras haberse realizado el escalado configurado. Para continuar, se deberá cerrar esta gráfica y aparecerá la correspondiente a la salida del SW. Si la configuración no es la anterior (*Show Acquired Template* no seleccionado) , directamente aparecerá la gráfica con la salida del post procesamiento como muestra la Fig 6.14.

En cualquier momento luego de la aparición de esta última gráfica, se puede ac-

cionar el botón ***Update*** actualizar los parámetros, reiniciar el post procesamiento con el botón ***Restart*** o dar por finalizado la aplicación con el botón ***Finish***.

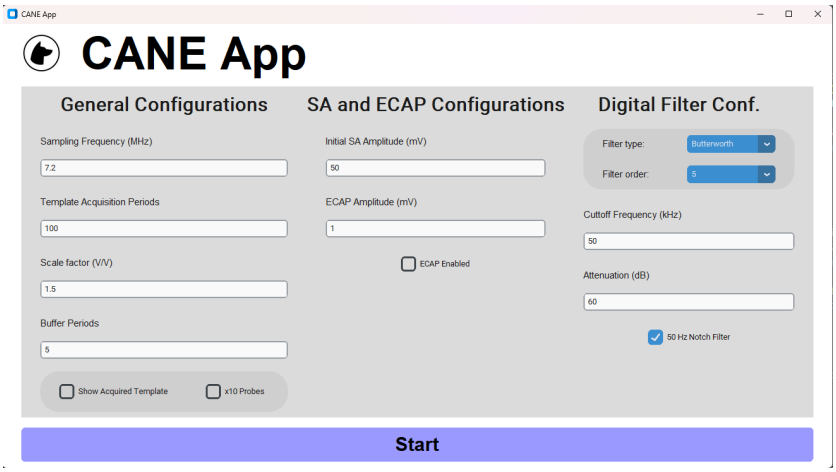


Figura 6.12: Estado inicial de la ventana de la GUI al iniciar la aplicación.

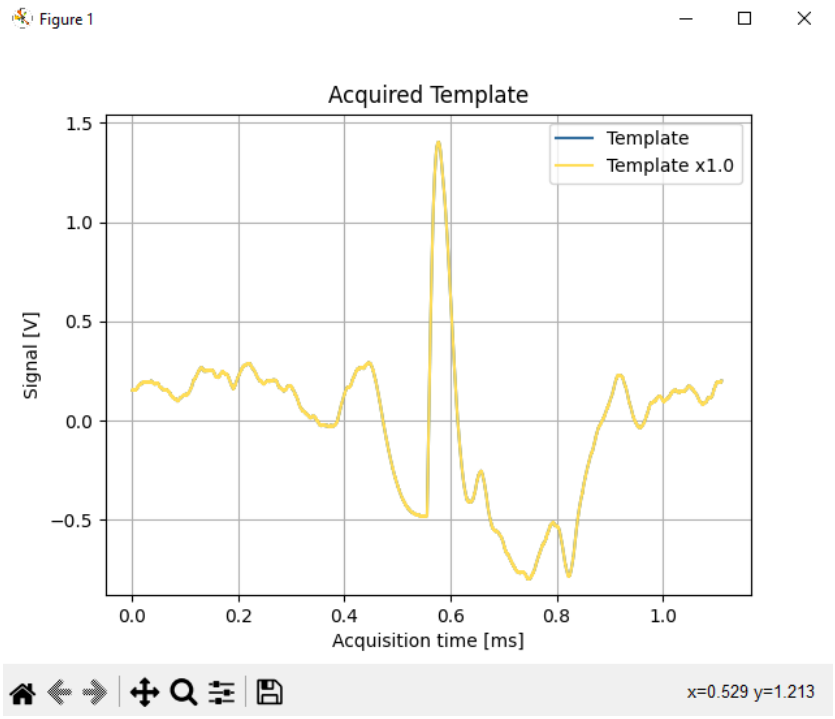


Figura 6.13: Plantilla adquirida mostrada si la configuración ***Show Acquired Template*** se encuentra activada al momento de iniciar el post procesamiento.

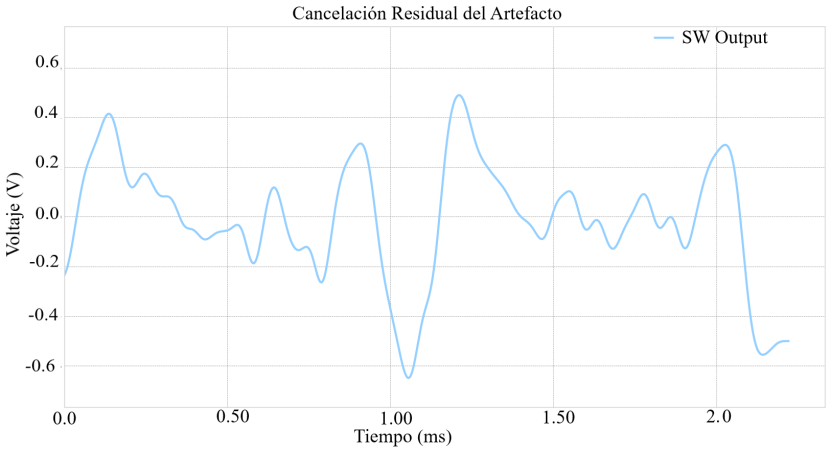


Figura 6.14: Ejemplo de la salida del post procesamiento

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 7

Resultados

En los capítulos anteriores se presentaron resultados intermedios a modo de verificar el correcto funcionamiento de cada etapa. En este capítulo, se trabajará en la evaluación de los resultados referidos al sistema general, con el fin de validar que la solución propuesta logra cumplir con sus objetivos.

7.1. Funcionamiento Punta a Punta

Luego de finalizado el desarrollo de los tres componentes del sistema, se probó el funcionamiento punta a punta para corroborar la correcta cancelación del artefacto de estimulación en el bloque de HW+FW y la del residuo en SW. En la Fig.7.1 se puede ver un diagrama de la interconexión de todos los componentes del sistema, mientras que en la Fig. 7.2 se puede apreciar una foto del mismo conectado.

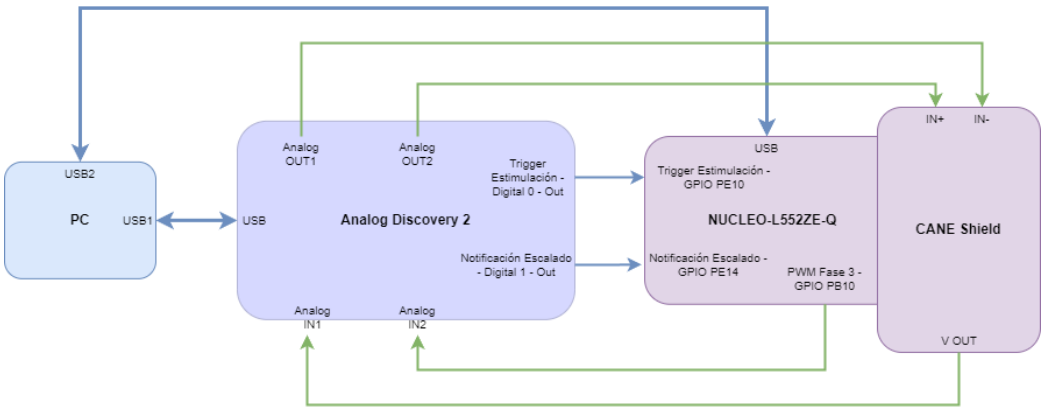


Figura 7.1: Diagrama de conexión para todos los componentes del sistema.

Se fijaron los parámetros de SW según la configuración que se puede observar en la Tab. 7.1 , equivalente a cancelar una SA de amplitud de 300 $mVpp$ luego de aplicado el escalado. Notar que, en este caso, se utilizaron ondas SA y ECAP con amplitudes fuera del rango especificado. La razón de esto consiste en que el AD2

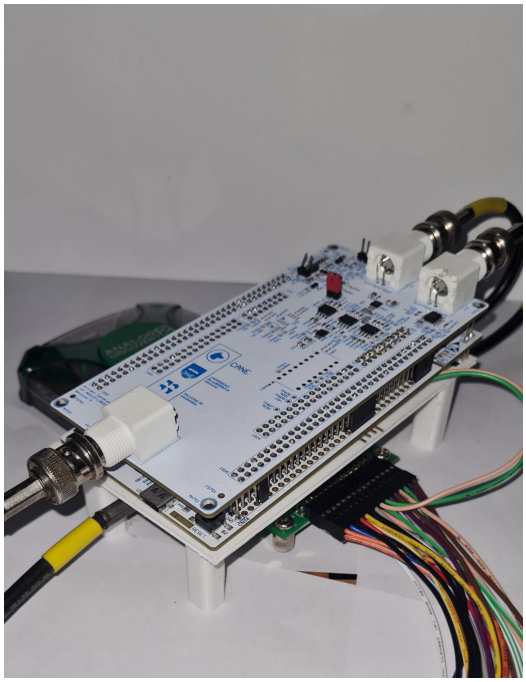


Figura 7.2: Sistema CANE ensamblado

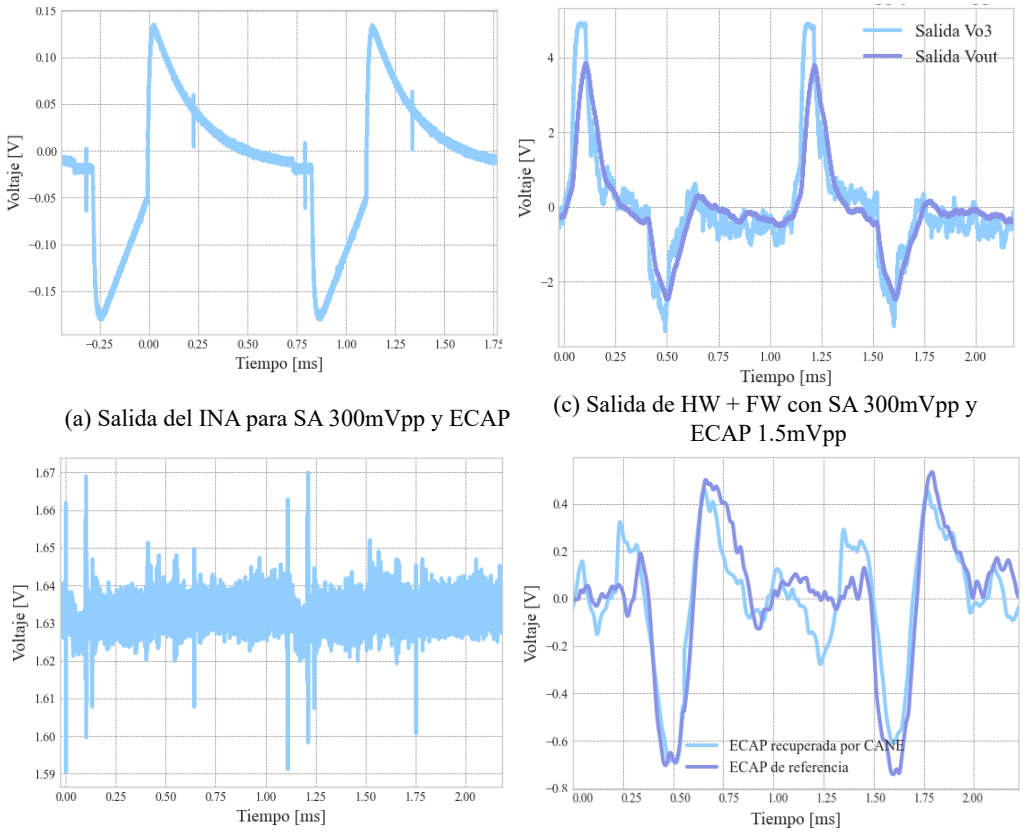
no cuenta con la precisión suficiente para generar ECAPs menores a 0,25 mV. Un divisor resistivo no fue utilizado para poder generar señales de menor amplitud debido a que el uso del mismo introduce ruido que afecta directamente el desempeño del sistema.

Parámetro	Valor
General Configurations	
Sampling Frequency	7,2 MHz
Template Acquisition Periods	100
Scale Factor	$1(V/V)$
Buffer Periods	5
SA and ECAP Configurations	
Initial SA Amplitude	100 mV
ECAP Amplitude	1 mV
Digital Filter Configuration	
Filter Type	Butterworth
Filter Order	5
Cutoff Frequency	50 kHz
Attenuation	60 dB

Tabla 7.1: Valores de parámetros utilizados para la configuración de SW. Los parámetros se encuentran nombrados al igual que aparecen en la aplicación (Fig. 6.12).

En la Fig. 7.3 se muestran las salidas relevadas en diferentes puntos del sistema. En particular, la Fig. 7.3.B se observa el resultado de la cancelación del artefacto realizado por el FW. Al igual que en el Cap.5, en la Fig. 7.3.C se visualiza la salida de la etapa de amplificación de 60 dB previo al filtro de salida, así como la salida total del HW. Se distingue una saturación del amplificador de salida al inicio de la cancelación debido a la transición rápida de la SA, resultado muy similar a lo observado en la simulaciones realizadas al inicio del proyecto.

Este efecto no imposibilita la adquisición de la señal neural como se puede ver a la salida del SW en la Fig.7.3.D, donde se encuentra también superpuesta una señal de referencia correspondiente a una ECAP de igual amplitud introducido a la entrada, sin SA.



(a) Salida del INA para SA 300mVpp y ECAP

(c) Salida de HW + FW con SA 300mVpp y ECAP 1.5mVpp

(b) Salida del restador para SA 300mVpp y ECAP 1.5mVpp

(d) Salida SW con SA 300mVpp y ECAP 1.5mVpp

Figura 7.3: Gráficos obtenidos para distintos puntos del sistema CANE completo.

Se observa la similitud entre la señal de referencia y la salida del SW, las cuales mantienen una correlación de 0,80. Resulta relevante notar que las diferencias más notorias entre estas señales provienen de ruido introducido por el sistema. El mismo se puede caracterizar observando la salida del SW sin la presencia de ECAP a la entrada. Se presenta entonces en la Fig. 7.4 como las diferencias observadas antes son propias de la respuesta del sistema, en las condiciones estipuladas.

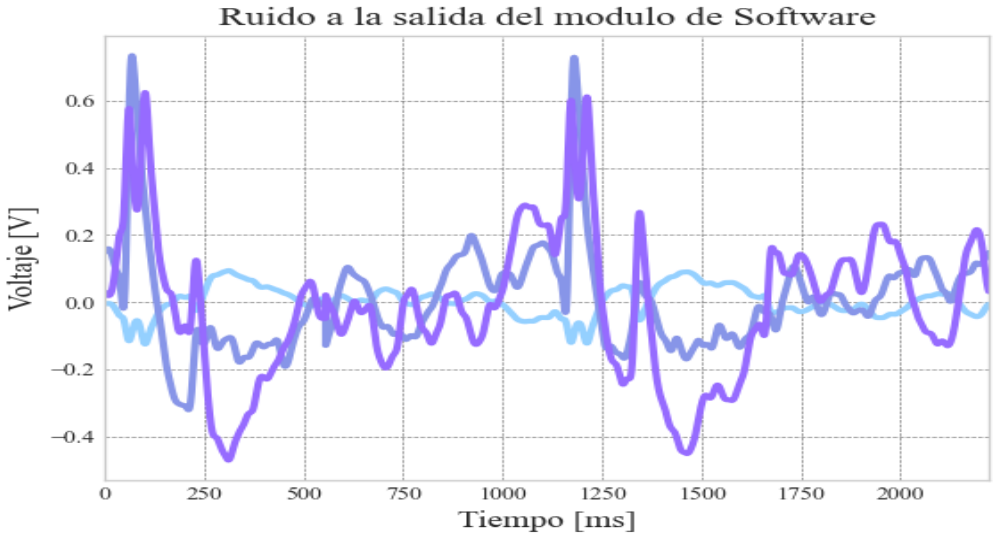


Figura 7.4: Tres iteraciones de la salida del SW sin presencia de ECAP a la entrada del sistema CANE.

7.2. Relevamiento de la Máxima Relación entre la Señal SA y la Señal de ECAP

Una vez corroborado el correcto funcionamiento del sistema completo, se relevó información sobre los limites del desempeño del mismo.

Para realizar esto, fue necesario establecer una figura de mérito con la cual definir qué se considerará como casos de éxito en la recuperación de la señal de ECAP. Con el fin de determinar la misma, al igual que en el Cap.3, se estudió el comportamiento de la correlación obtenida entre la señal de referencia y la salida del SW. Para comenzar se utilizó en la entrada una señal SA de 300 V_{pp} y distintos valores para la señal ECAP. Se escogió este valor para la SA dado que es el máximo valor posible que puede tomar dentro del rango especificado en su modelo (Secc. 2.2). A continuación, se enseña en la Tab. 7.2 los resultados obtenidos para esta prueba. Es importante resaltar que las amplitudes utilizadas para el ECAP a la entrada resultan mayores al rango estipulado en su modelo. Es así por varias razones. En primer lugar, ocurre que el AD2 no permite generar señales menores a 250 μV. Esto ya deja por fuera varios valores dentro del rango. En segundo lugar y más importante, el ruido a la salida resulta tener una amplitud (ver Tab. 4.8) que no permite recuperar correctamente valores de ECAP a la entrada en el orden de los micro voltios.

Amplitud ECAP (mV_{pp})	Correlación obtenida
1.5	0.892
0.75	0.800
0.38	0.7649
Correlación obtenida a la salida del SW sin presencia de ECAP	
0.567	

Tabla 7.2: Resultados obtenidos para la correlación entre la ECAP recuperada a la salida del sistema CANE con respecto a la señal ECAP de referencia.

La primera observación a realizar de los resultados de la Tab. 7.2 es que sin la presencia de ECAP a la entrada ya se obtiene a la salida una correlación de 0.567. Este dato es útil como punto de partida para determinar qué valor de correlación se fijará como criterio de éxito para afirmar que una ECAP fue recuperada exitosamente. Por otro lado, se observa que para la mayor amplitud de ECAP se obtuvo una correlación de 0.892. Para determinar el criterio de éxito, con mayor precisión, se verán los gráficos de como resulta la salida del SW para cada caso de la tabla. Recordar que en la Fig. 7.4 se enseña la salida de SW sin la presencia de la señal de ECAP a la entrada.

En la Fig. 7.5 se observa la salida del SW cuando la señal de ECAP a la entrada tiene una amplitud de 0,38 mV_{pp}. Para este caso, visualmente no se distingue la morfología de la ECAP. Vale la pena recordar que para esta prueba, la correlación resultó en 0.7649.

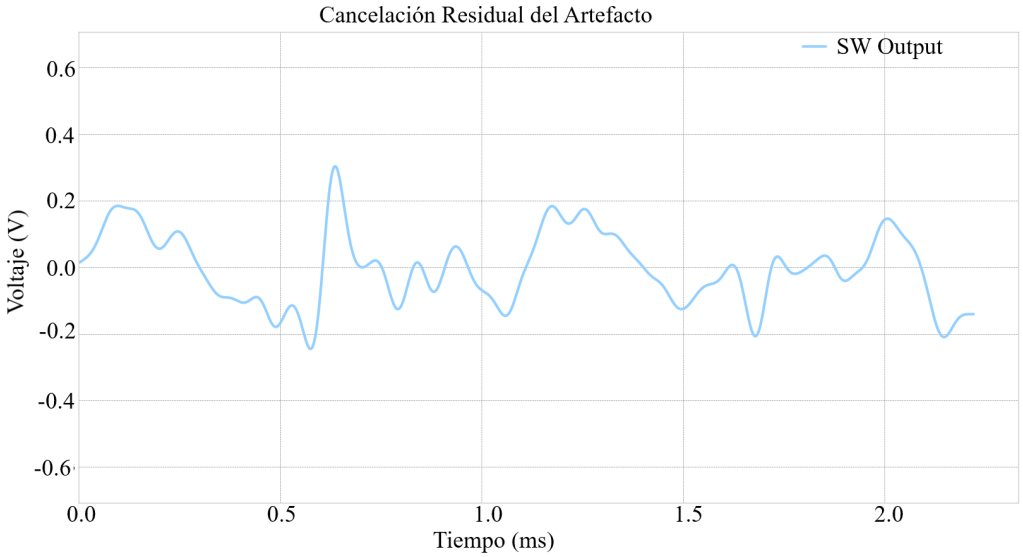


Figura 7.5: Salida del SW con ECAP de 0,38 mV_{pp} y SA de 300 mV_{pp} a la entrada del sistema CANE.

Recordar que la morfología que se debería presentar es la que se muestra en la Fig.7.6.

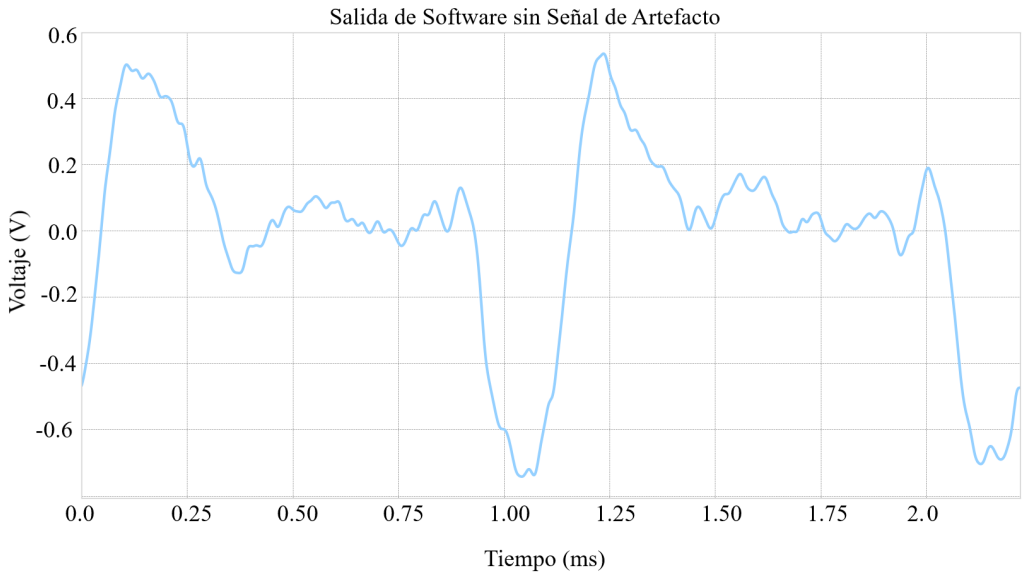


Figura 7.6: Salida del sistema para ECAP de de $1{,}5mV$, sin señal de artefacto.

Sin embargo, al aumentar la amplitud de la ECAP a 0,75 mV_{pp}, como se ve en la Fig. 7.7, comienza a aparecer la forma característica de la señal. Por lo tanto, se utilizará este caso como base para aceptar como válida la recuperación de la ECAP. Finalmente, según la Tab. 7.2 la correlación en esta situación es de 0.800. Teniendo lo anterior en cuenta, se tomará como criterio de éxito el valor de correlación correspondiente a 0.8.

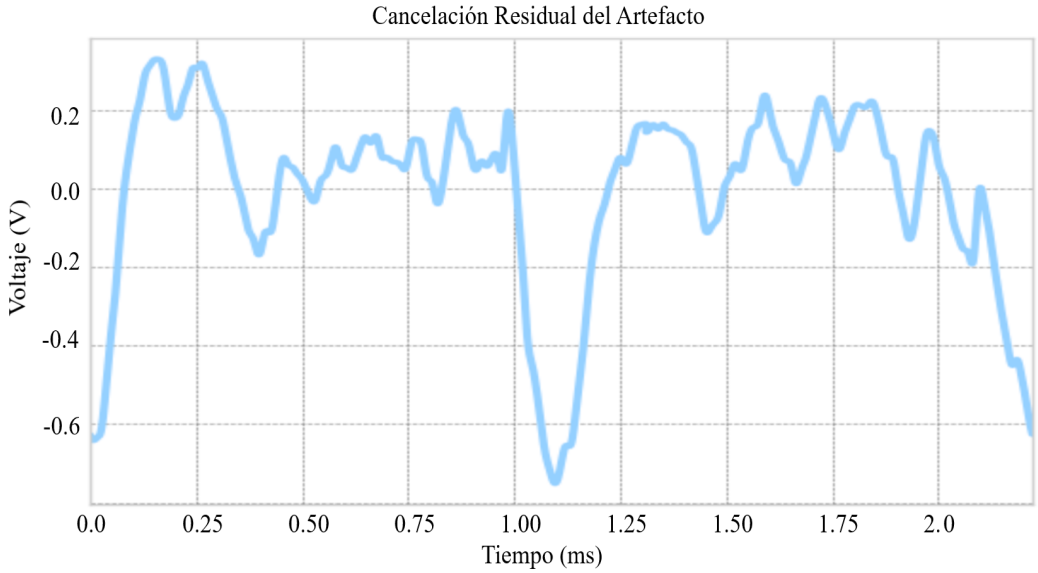


Figura 7.7: Salida del SW con ECAP de $0{,}75\ mV_{pp}$ y SA de $300\ mV_{pp}$ a la entrada del sistema CANE.

En el último caso de estudio, donde la señal de ECAP a la entrada es de

1,5 mV_{pp} se logra ver que aparece claramente la morfología completa de la señal de ECAP. Resulta ser el caso de mayor correlación resultante hasta ahora, siendo la misma de 0.892. Ver Fig. 7.8.

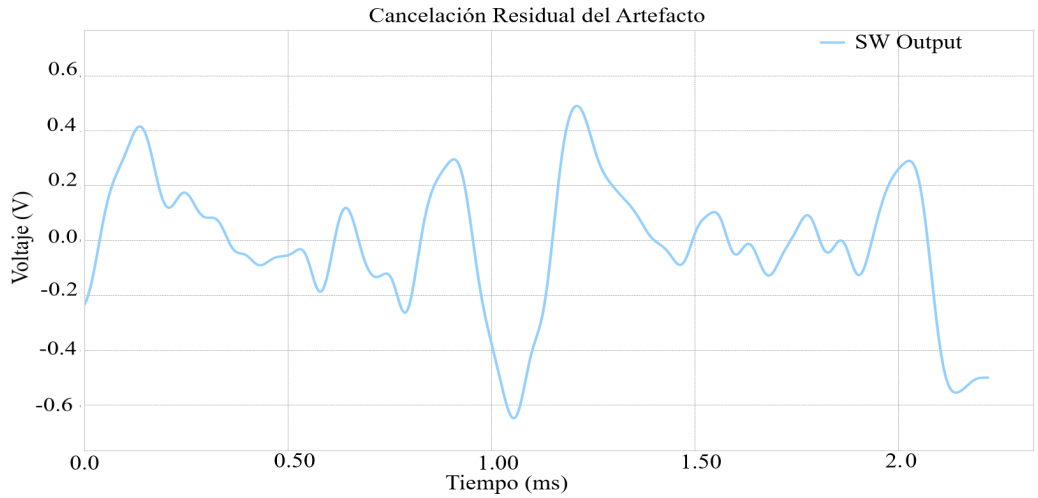


Figura 7.8: Salida del SW con ECAP de 1,5 mV_{pp} y SA de 300 mV_{pp} a la entrada del sistema CANE.

Con el criterio de éxito definido, se realizó una exploración de los parámetros del sistema para determinar cual es el máximo factor entre la amplitud de la SA y el ECAP.

En la Fig. 7.9 se muestra la salida del SW para para una SA de 405 mVpp y un ECAP de 0,75 mVpp. No se distingue la forma de onda del ECAP, teniendo una correlación de 0,709. Por otro lado, para igual amplitud de SA pero ECAP de 1,5 mVpp se obtienen los resultados de la Fig. 7.10 con una correlación de 0,769.

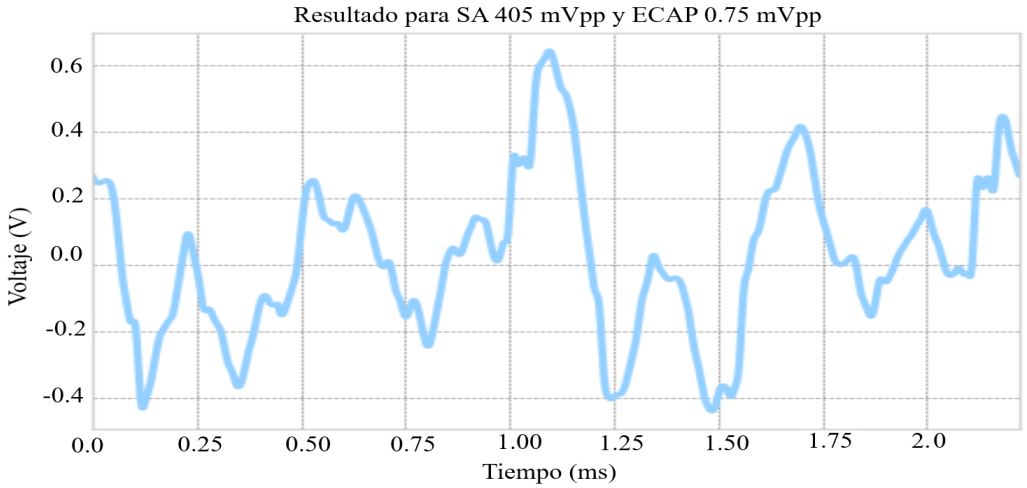


Figura 7.9: Salida del SW con ECAP de 0,75 mV_{pp} y SA de 405 mV_{pp} a la entrada del sistema CANE.

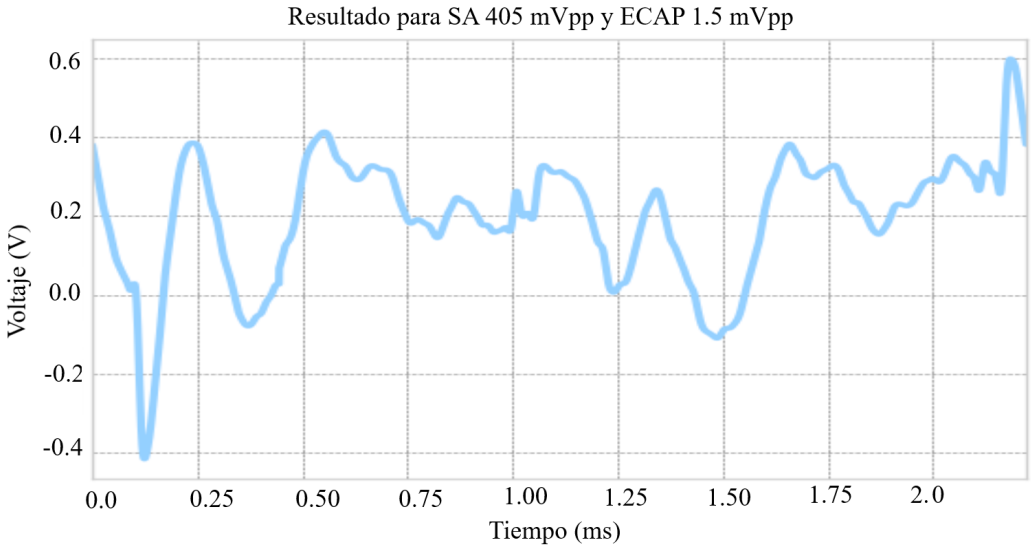


Figura 7.10: Salida del SW con ECAP de 1,5 mV_{pp} y SA de 405 mV_{pp} a la entrada del sistema CANE.

Para amplitudes más grandes, con una SA de 450 mVpp y ECAP de 1,5 mVpp se obtiene una correlación de 0,69. En la Tab. 7.3 se pueden apreciar el desempeño del sistema para todas las relaciones de amplitudes exploradas.

SA (mV)	ECAP (mV)	Relación de Amplitud entre SA y ECAP a la entrada (dB)	Correlación
240	1.5	44.1	**0.940**
240	0.75	50.1	**0.871**
240	0.38	56.0	**0.812**
300	1.5	46.0	**0.893**
300	0.75	52.0	**0.800**
300	0.38	57.9	**0.765**
350	1.5	47.4	**0.821**
350	0.75	52.9	**0.794**
350	0.38	59.3	**0.743**
405	1.5	48.6	**0.769**
405	0.75	54.6	**0.709**
405	0.38	60.6	**0.687**
450	1.5	49.5	**0.695**
450	0.75	55.6	**0.657**
450	0.38	61.5	**0.617**

Tabla 7.3: Correlaciones obtenidas para diferentes relaciones de amplitudes del artefacto y la señal neural.

Dentro de los casos que cumplen el criterio establecido, la mayor relación entre las amplitudes del artefacto y la señal neural se da para 240 mVpp y 1,5 mVpp respectivamente, lo cual equivale a una relación entre las señales de 56,0 dB. Es importante notar que en la tabla existen casos en los cuales se obtiene un peor desempeño para relación entre amplitudes menores. Por ejemplo, para el caso 405 mV y 1,5 mV se obtiene una correlación de 0,769 a pesar de tener una relación de 48,6 dB, 3,4 dB menor que el mejor caso antes mencionado.

Esto se debe a que hay cotas para la amplitud máxima de la SA que se puede cancelar, debido a varias razones como pueden ser el desempeño del ADC y DAC, el retardo de reproducción y el ruido introducido que afectan la exactitud de la plantilla reproducida. Cuanto mayor es la amplitud de la SA utilizada, la cancelación se degrada, aumentando la saturación y variación de la salida del sistema.

También se cuenta con una amplitud mínima de la ECAP que se pude usar. Este límite viene dado por un lado por los instrumentos utilizados que son incapaces de generar señales de amplitudes menores a 0,5 µV con exactitud. Mientras que también existe un límite proporcionado por el ruido observado a la salida del sistema que es consecuencia de diversos factores como la eficacia de la cancelación del artefacto, el ruido de las fuentes así como el ruido propio de los componentes usados en la cadena de amplificación del HW.

Los límites para estas dos señales genera una zona de funcionamiento para la cual se obtiene una correlación de 0,8 o mayor. Esta región de funcionamiento se puede construir a partir de los valores mostrados en la tabla, los cuales fueron utilizados para graficar la Fig. 7.11. Se observa aproximadamente la región estricta de funcionamiento del sistema CANE relevada a partir de los valores de la Tab.7.3 con correlaciones mayores a 0,8. Notar que no fueron incluidos en la región los puntos con excursiones para la ECAP mayores a 1,5 mVpp y SA menores o iguales a 350 mVpp debido a que ninguna medición fue realizada en esta zona.

Cabe destacar que todas las pruebas anteriores fueron realizadas para una ECAP con la onda N1 (ver Secc.2.2.2) que ocurre $470\mu s$ luego de la estimulación. La última prueba verifica el funcionamiento en el peor caso de la variación del desplazamiento temporal. Este caso corresponde a una onda ECAP donde la onda N1 ocurre $280\mu s$ luego de la SA, este resultado surge de recordar que la saturación del amplificador de salida de HW es más probable que sature al inicio de la SA y la onda N1 debe ocurre según una distribución normal con $\sigma = 62\mu s$. Para la prueba se modificaron las señales para que la ECAP ocurra con una desviación de -3σ con respecto a la media.

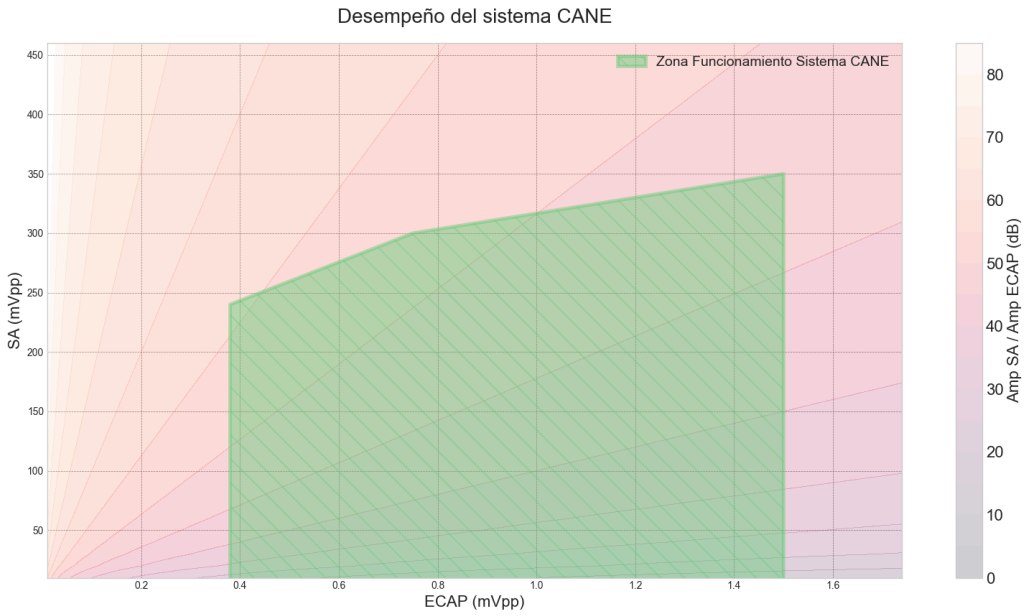


Figura 7.11: Región de funcionamiento del sistema CANE para la recuperación de una señal neural con una correlación mayor al 0,8.

En la Fig. 7.12 se puede observar la salida del SW para dicho caso con amplitudes para el ECAP y la SA de 1,5 mVpp y 300 mVpp respectivamente. Para dicha salida, se obtiene una correlación con la señal de referencia de 0,861, correspondiente a una correcta adquisición de la señal neural.

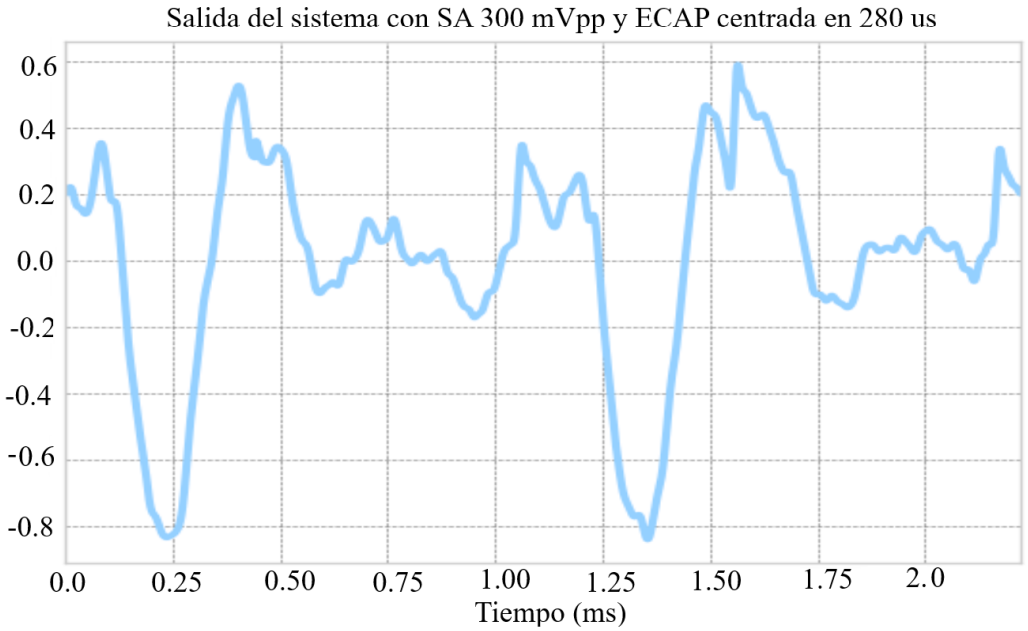


Figura 7.12: Salida del sistema para el peor caso de variación temporal.

Capítulo 8

Conclusiones

Es relevante destacar que durante el transcurso del proyecto se abarcaron distintas áreas que aportaron aprendizajes y experiencia en cuanto a organización y ejecución. Sin dudas son herramientas que a futuro serán de utilidad.

Habiendo recopilado la información del desarrollo y los resultados en cada etapa del proyecto, se concluyó que se finalizó con éxito el sistema CANE. El mismo es capaz de suprimir artefactos en tiempo real y detectar señales neurales de amplitud decenas de decibeles menores a los artefactos que acompañan al estímulo.

Comenzando el análisis por el capítulo de Simulación, se destaca que mediante la misma se logró emular un sistema real y permitió hacer un planeamiento estratégico del proyecto. Las proyecciones y observaciones que surgieron de la simulación resultaron útiles para algunas consideraciones del diseño del Hardware, el Firmware y el Software. En particular se destacan las pruebas realizadas para determinar el máximo retardo y la mínima frecuencia de muestreo aceptables para los conversores. Como resultado se obtuvo que se debía contar con 5 μs de retardo como máximo y utilizar una frecuencia de 200 kHz como mínimo.

Dentro del desarrollo del Hardware, se diseñaron distintos circuitos analógicos que trabajan en conjunto para una cancelación de artefactos exitosa. Tanto los circuitos de amplificación, los de filtrado de señal y los de alimentación aportan sustento a la solución. Todo esto se realizó en una PCB personalizada que facilita el uso del sistema completo. La decisión de diseñar la PCB para ser utilizada como shield fue correcta, de esta forma el set up para realizar las medidas y manipular el dispositivo resultó sencilla, cómoda y repetible. Aportó a esto también, el utilizar conectores de tipo BNC para las entradas y salidas del sistema.

Siguiendo con el análisis del Firmware, se implementó un código personalizado en el cual se hizo uso de librerías provistas por el HAL de ST. Es así que, utilizando estratégicamente los periféricos del microcontrolador se consiguió muestrear a alta velocidad (1 MHz) y procesar los datos para un correcto manejo del lazo de control, incluyendo diferentes métodos de detección de error y condición de convergencia para controlar la máquina de estados.

Finalmente, se desarrolló una sistema de Software basado en la paralelización de tareas. Dicha metodología otorgó la posibilidad de controlar simultáneamente el algoritmo de post procesamiento, la visualización y adquisición de datos en tiempo

real y la interacción con el usuario mediante una interfaz gráfica. A partir de esta última, el usuario tiene acceso a la configuración básica de los parámetros del sistema y a la modificación de los mismos durante el funcionamiento del sistema CANE.

Para la recopilación de resultados, y caracterización de los casos exitosos, se utilizó como figura de mérito la correlación entre la señal de ECAP reconstruida y la señal ECAP de referencia. En particular, de forma experimental se llegó a determinar como criterio de éxito que la correlación sea mayor o igual a 0.8. Con esto definido, se relevaron los límites del sistema CANE en cuanto a la relación máxima entre la señal neural y el artefacto incidente. Como resultados numéricos, la relación máxima entre la señal SA y señal de ECAP a la entrada con la cual se logró reconstruir una ECAP a la salida que cumpla el criterio de éxito es de 56 dB. Para dicha situación, se obtuvo una correlación de 0.812 entre la señal de ECAP reconstruida y la señal ECAP de referencia. Sin embargo, la mayor correlación obtenida fue de 0.940, para una relación entre SA y ECAP de entrada de 44,1 dB. También se relevó el rango de operación del sistema, en cuanto a las amplitudes de las señales de ECAP y SA donde se obtiene la señal reconstruida que cumple el criterio de éxito. Un resultado que cabe destacar, aunque quedó por fuera del criterio de éxito definido, es el de una relación de 59,3 dB entre las señales obteniendo una correlación de 0.765.

8.1. Trabajo a Futuro y Posibles Mejoras

A lo largo del desarrollo, verificación y pruebas de las diferentes áreas se identificaron posibles soluciones a algunos problemas con los cuales nos encontramos así como mejoras a ser introducidas en un sistema más completo.

8.1.1. Hardware

Para el HW se identificaron posibles mejoras. La primera consiste en la solución del mecanismo de ganancia variable para el amplificador de instrumentación. Una posible solución ya fue mencionada en la Secc. 4.4 pero esta limita el ICMR del amplificador y por ende no es ideal. Otras soluciones consisten en realizar una mejor búsqueda de potenciómetros digitales que cuenten con condiciones de funcionamiento similares a las del circuito o cambiar la topología de funcionamiento utilizando llaves discretas.

También es importante resaltar que el sistema en su totalidad, como puede ser observado en la Tab. 4.8 de la Secc. 4.4, cuenta con una importante componente de ruido a la salida del sistema que directamente limita la mínima ECAP a ser detectada. Un nuevo análisis de ruido del sistema debe ser llevado a cabo para identificar posibles causas de estas variación, como puede ser la alimentación desde USB de 5 V. Posibles soluciones pueden ser alimentar el sistema desde una batería, que permita reducir el ruido desde la fuente principal, así como generar una alimentación separada de 5 V para la circuitería analógica.

Por otro lado, una de los mayores obstáculos encontrados durante el desarrollo del FW proviene de tener que realizar un lazo separado de control de continua de la plantilla. La necesidad de esta función surge que la segunda etapa de amplificación cuenta con el filtrado de continua en las etapas de entrada. De esta manera, el lazo de control establecido por el microcontrolador no es capaz de corregir desviaciones en continua a la salida. Una posible solución se detalla en el Anexo. F.3.1.

Una mejora adicional para el HW consiste en un rediseño del filtrado analógico para conseguir un filtro de mayor orden en la banda pasante de interés. Por último, sería de suma utilidad incluir en el HW la capacidad de atenuar una señal de entrada para poder correctamente generar una señal neural de interés de la correcta amplitud. Posibles soluciones fueron discutidas y el uso de un transformador de audio fue considerado como alternativa para generar una atenuación que introduzca bajo ruido.

8.1.2. Firmware

En una primera instancia, sería interesante continuar con el desarrollo de FW para el DAC y ADC externos. Esta funcionalidad se encontraba por fuera del alcance y objetivos del proyecto. Los mismos fueron introducidos como posible solución en caso de que el rango de los conversores internos no resultara suficiente para cumplir con la relación máxima entre las señales de interés. Por lo tanto, se decidió no ahondar en el desarrollo de FW de los conversores externos con el fin de perfeccionar elementos que sí estaban incluidos en el alcance. Queda como trabajo continuar este labor para poder aumentar la relación máxima entre las señales SA y ECAP.

Otro aspecto en donde se encontró oportunidad de mejora es entre la interacción del módulo de SW y el FW, para optimizar la coordinación entre los dos subsistemas. Esto podría ser implementado utilizando un protocolo de comunicación serial como puede ser UART, por lo que se requeriría implementar un módulo para manejar el periférico y también para procesar los comandos recibidos y enviar las respuestas adecuadas.

8.1.3. Software

Para el SW las mejoras identificadas están relacionadas a la capacidad de procesamiento del subsistema. En particular, se encontró el cuello de botella presente en la capacidad de graficar la salida del SW por medio de la librería de python *matplotlib*. Un análisis en profundidad del método utilizado para mostrar los resultados debe ser llevado a cabo para optimizar el rendimiento.

En cuanto a trabajo futuro, nuevas técnicas de post procesamiento podrían ser exploradas que sean especialmente robustas ante ruido como puede ser un filtro de Kalman. Para finalizar, como se mencionó anteriormente la comunicación entre el módulo de SW y el FW podría ser mejorada, lo cual requeriría el desarrollo de un stack de comunicación serial capaz de enviar y recibir mensajes que permita coordinar la ejecución del post procesamiento con el estado actual del FW.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice A

Generación de señales en Simulink

A.1. ECAP

A continuación, en las Fig. A.1, Fig. A.2 y Fig. A.3, se presentan los bloques implementados para la formación de la ECAP.

En una primera instancia (ver Fig. A.1), se generó la función sinusoidal base para la formación del ECAP. Para ello se utilizó un bloque que genera dicha función, y se la multiplicó por una función escalón de amplitud uno para obtener como salida un período y medio de la señal sinusoidal. De esta forma se obtienen los tres picos que luego serán modulados.

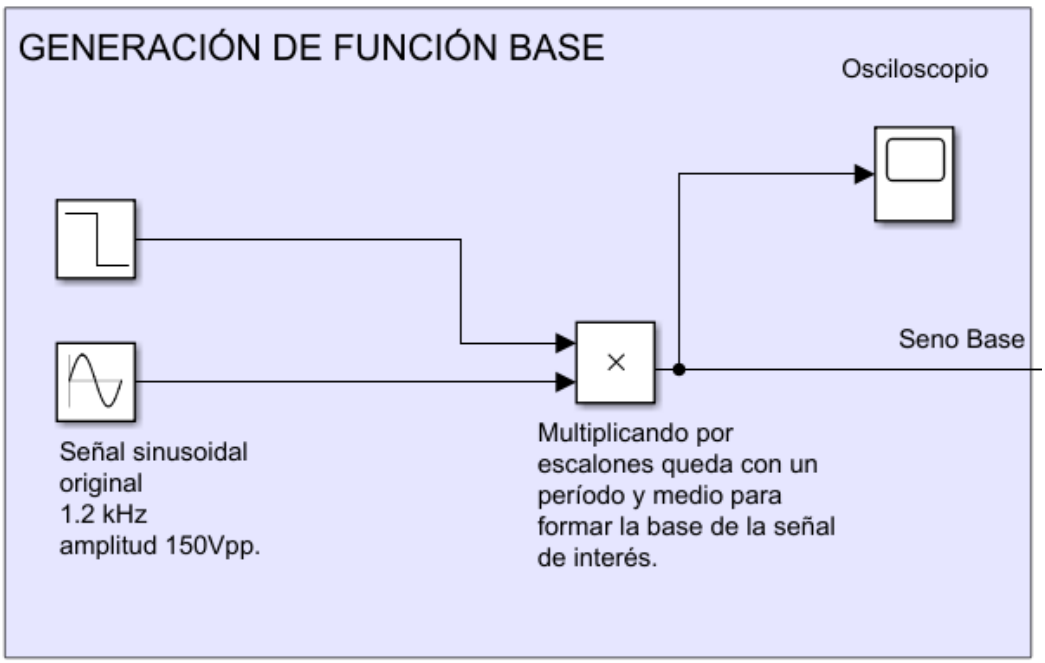


Figura A.1: Generación de la función sinusoidal base para la formación de la señal ECAP. La frecuencia del seno es de 1.2 kHz.

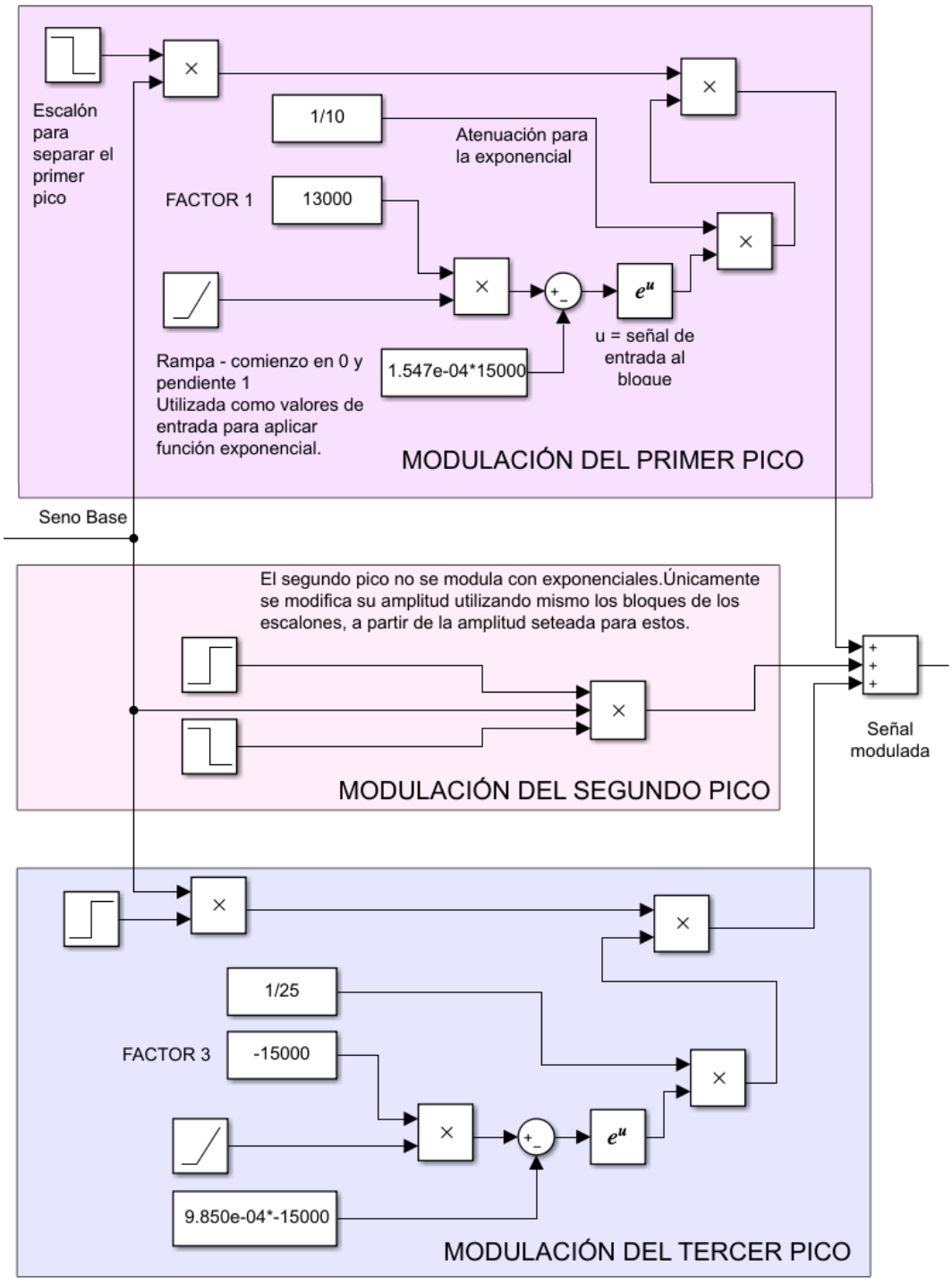


Figura A.2: Bloques implementados para la modulación individual de los picos de la función sinusoidal base para la formación de la ECAP.

Luego de la obtención de los tres picos del seno, se procedió a modelar los mismos con diferentes funciones, como se indicó anteriormente. Para lograr esto,

se trabajó en cada uno de los picos de forma independiente para dar la morfología necesaria a cada uno de ellos. Como se observa en la Fig. A.2, se utilizaron funciones escalón para separar los picos y así aplicar modulaciones distintas a cada uno de ellos. El primer y tercer pico fueron modulados a partir de funciones exponenciales, mientras que al segundo pico únicamente se le modificó la amplitud .

En el caso de la modulación exponencial, el procedimiento de modulación consiste en multiplicar el pico por una función de la forma e^x, donde el valor de x varía en cada paso temporal de la simulación. Las diferencias en los resultados de la modulación dependen de cómo es generado el x. Para el primer y tercer pico, el x por lo tanto se genera a partir de valores constantes distintos y utilizando como base una recta de pendiente 1 y comienzo en 0. De esta forma, en cada paso temporal la recta tiene un valor distinto, el cual luego es multiplicado por un factor constante que determinará la forma de la exponencial. Se juega tanto con el valor de la constante como con su signo. A la función exponencial resultante se la multiplica por una constante para atenuarla hacia una amplitud acorde, luego se procede a multiplicar por el pico. En definitiva, la señal por la cual se multiplicará al pico será de la forma que se estipula en la Ec. A.1. Donde a, b y c son parámetros diferentes para cada pico.

$$a.e^{b.x} + c \tag{A.1}$$

Los picos modulados se unifican posteriormente a través de la suma de los mismos.

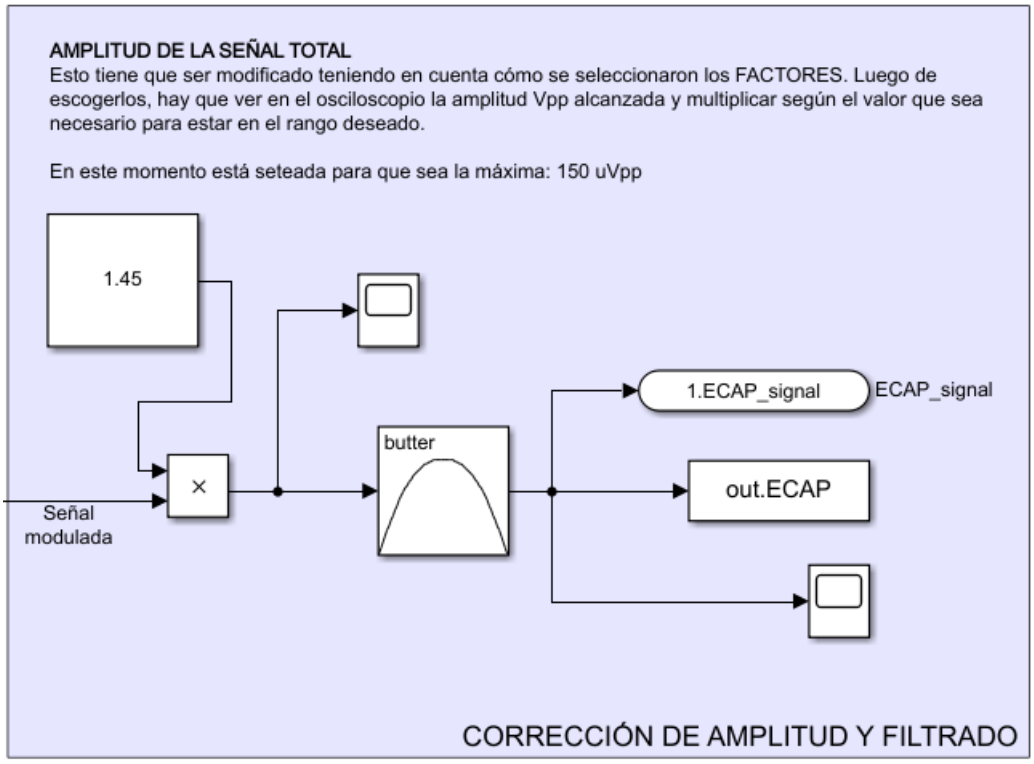


Figura A.3: Corrección de la amplitud y filtrado de la señal para la formación de la señal ECAP.

Dentro de la última instancia de bloques (ver Fig. A.3), después de que los picos fueron modulados y sumados, es necesario corregir la amplitud de la señal final para cumplir con el rango especificado para la misma. Para ello se multiplica la señal por una constante, fijada actualmente para crear la amplitud máxima característica de la señal. Además, para asegurar que la señal se encuentra dentro del ancho de banda correcto, se pasa la misma por un filtro pasabanda de primer orden en la banda de 300 kHz a 5 kHz como se indicó anteriormente.

En la Fig. A.4 se visualiza la división de bloques mediante la cual cada fase fue construida. Para facilitar su creación, la fase negativa se dividió a su vez en dos etapas. Por lo tanto, se distingue en la imagen tres bloques individuales. A diferencia de la construcción de la señal ECAP, no se parte de una función base en común, sino que cada etapa es construida independientemente.

Dentro del primer bloque se crea la parte exponencial creciente de la fase negativa. Para ello se utilizó la misma técnica de modulación exponencial implementada para la señal de ECAP. La diferencia aquí es que la constante de tiempo de la exponencial se encuentra predefinida por las especificaciones de la Secc. 2.2. Además, aparece la implementación de un nuevo bloque que introduce un retardo temporal a la señal. Dicho bloque se introduce para posicionar la etapa en su posición correspondiente dentro del el eje temporal.

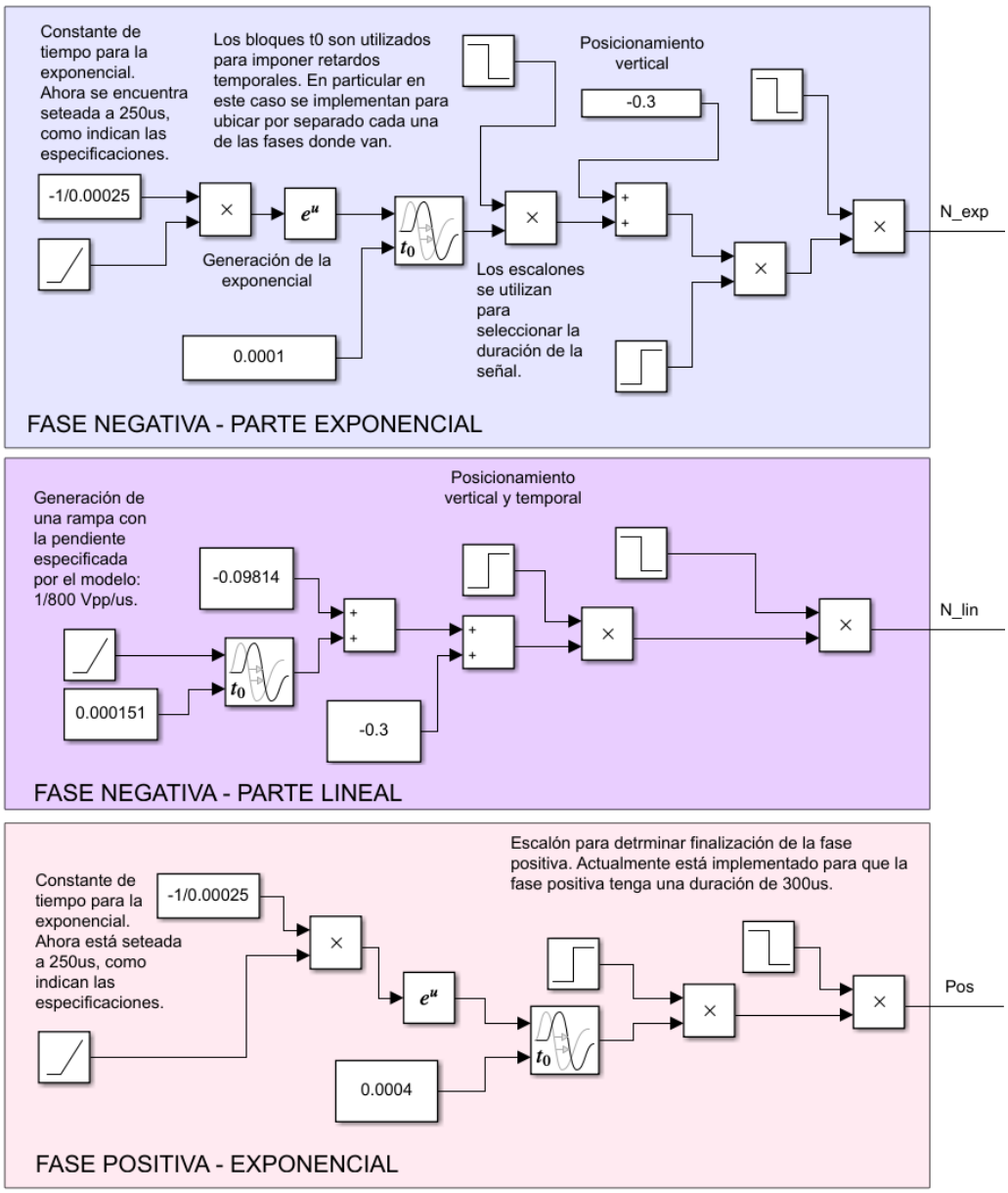


Figura A.4: Creación de las fases para la formación de la señal SA.

La segunda etapa de la fase negativa resulta en una construcción sencilla, debido a su característica lineal. Simplemente se crea una recta con la pendiente estipulada por las especificaciones y se la posiciona vertical y temporalmente.

Por último, el tercer bloque consiste en la formación de la fase positiva exponencial decreciente. Nuevamente se repite los procedimientos de modulación exponencial vistos anteriormente.

Una vez generadas las fases de la señal, como están ubicadas temporalmente en posiciones distintas, se suman para formar la señal completa. Esto se puede observar en la Fig. A.5. Adicionalmente, se corrige la amplitud y se filtra la señal con un filtro pasabajos de primer orden de 16 kHz para cumplir con las características especificadas. En particular, la amplitud fue fijada en la máxima correspondiente al rango característico de la señal SA.

La razón de utilizar filtrado de primer orden es que el mismo representa el filtrado de la señal introducido por el cuerpo humano. En definitiva, el cuerpo es el que limita el ancho de banda de la señal de estímulo a la frecuencia determinada anteriormente.

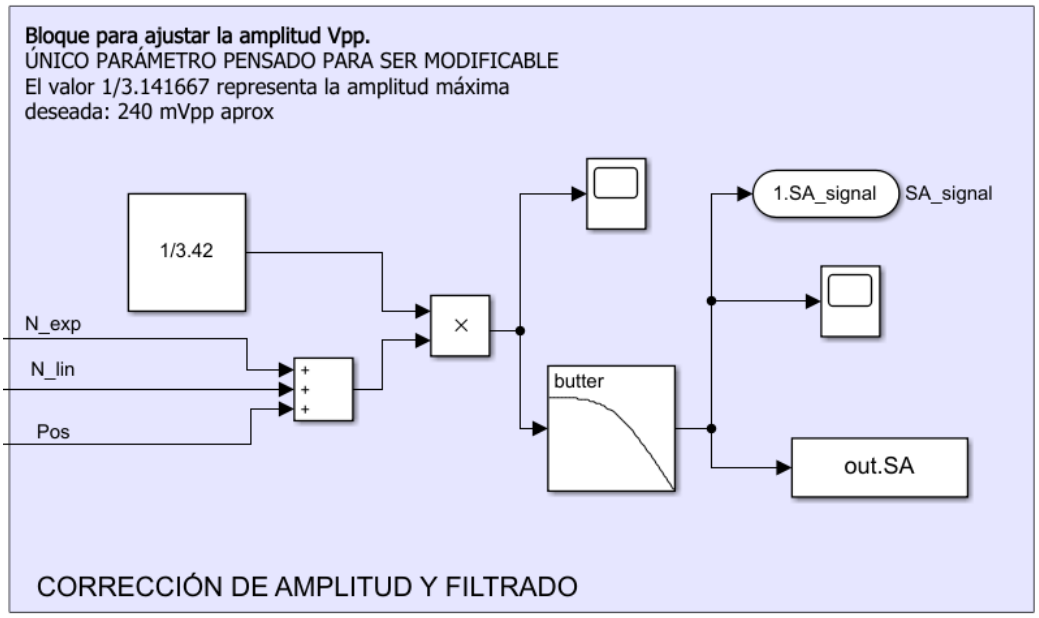


Figura A.5: Corrección de amplitud y filtrado en frecuencia para la señal SA.

Apéndice B

Utilización del Analog Disocvery como Generador de Ondas

B.1. Motivación

Una parte fundamental del proyecto son las señales a estudiar y el cómo generarlas. Para esto es necesaria una herramienta que permita la generación de ondas arbitrarias. El dispositivo Analog Discovery 2 [39] provee una solución portable y demás beneficios en cuanto a funcionalidades. En particular, el mismo es de utilidad como osciloscopio y generador de fuentes de alimentación.

B.2. Python como Herramienta de Control de Analog Discovery

Python tiene disponible la librería *pydwf* [48] para controlar dispositivos de Digilent, en específico: Electronics Explorer, Analog Discovery, Analog Discovery 2, Digital Discovery, Analog Discovery Studio y Analog Discovery Pro. La misma se basa en la librería low-level de C que provee Digilent.

De esta librería se utilizan principalmente dos clases, *DwfLibrary* y *DwfDevice*. La primera de ellas incluye métodos que no son específicos para un dispositivo. En definitiva, la instancia de esta clase otorga funcionalidades para acceder a los mismos. La instancia se pasa cómo parámetro a *pydwf.utilities.OpenDwfDevice()* y de esta forma se accede a la segunda clase mencionada.

La clase *DwfDevice* va a representar un dispositivo en particular al que ya se accedió. La misma posee varios métodos específicos a los cuales se puede acceder a través de sus atributos, cada uno asociado a funcionalidades distintas. En particular, los atributos de interés son: *AnalogIn*, para la utilización del intrumento como osciloscopio; *AnalogOut*, para la generación de señales; y por último *DigitalIO*, para el control de pines digitales.

B.3. Scripts de Python para la Generación de Ondas

Los scripts implementados cumplen fundamentalmente con dos funciones, generar las señales de interés (SA y ECAP) en los canales de waveforms y adquirirlas mediante los canales de osciloscopio del Analog Discovery.

Para la generación de señales se hizo uso del atributo *AnalogOut* de la clase *DwfDevice*. Con el mismo, se configuran y setean los dos canales disponibles para la generación de ondas. En particular, se configuraron para obtener la señal SA en el canal 1 y la ECAP en el canal 2. Las señales son creadas a partir de un archivo de texto con la información de la amplitud en cada paso. Estos archivos se crearon descargando los datos generados para las señales en la simulación de Simulink (ver sección 3.3), y se importaron en el script utilizando la librería *numpy* de python.

Una vez generadas las señales fue necesario configurar el dispositivo como osciloscopio para corroborar que se hizo de forma correcta. Para esto, en simultáneo se configuraron y setearon los canales de osciloscopio a través del atributo *AnalogIn*.

El programa se divide en una función para la generación, otra para la adquisición y el main. Desde el main se accede al dispositivo, se obtiene la información de las señales desde los archivos de texto, y se configuran parámetros generales. Las demás funciones cumplen con el propósito específico de su nombre. En el caso de la función de adquisición, se llama además periódicamente a la función de generación. Esto se debe al modelo de la señal ECAP. Como se muestra en la sección 2.2.2, el momento de aparición de la misma se modela como una distribución normal que depende del acontecimiento de la fase negativa de la señal SA. Para recrear dicho comportamiento, cada vez que se llama la función de generación de ondas se pasa como parámetro un tiempo de retardo con comportamiento según una distribución normal para que la reproducción de esta onda que cumpla con las características necesarias.

B.4. Resultados

Para la visualización de la información generada y adquirida, se realizaron gráficas a través de las funciones de la librería *matplotlib.pyplot*. A continuación, en la Fig. B.1 se observa el gráfico con la adquisición de las señales generadas.

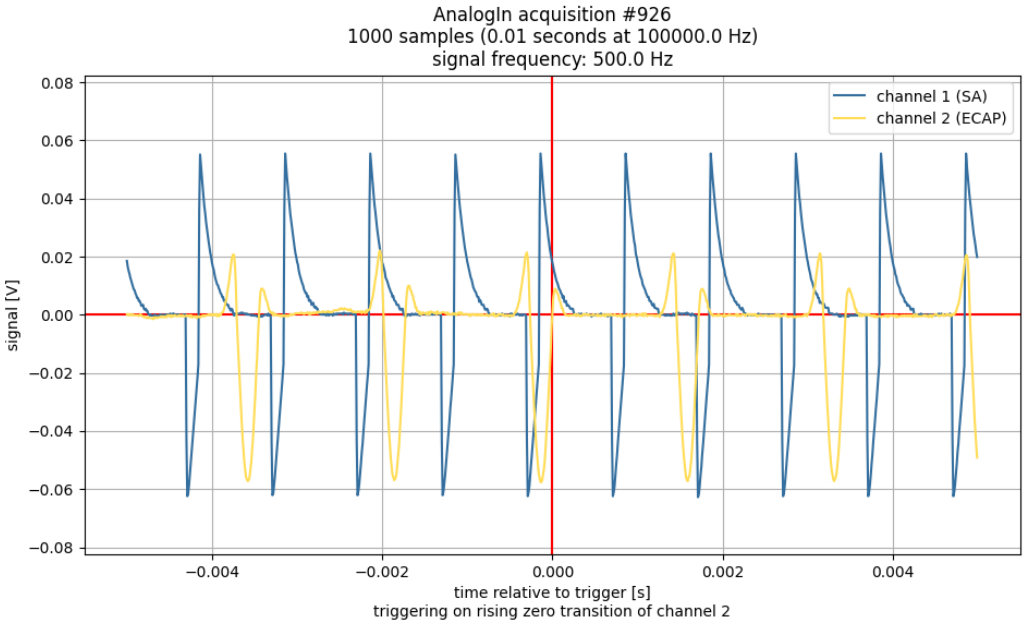


Figura B.1: Gráfico que demuestra la adquisición de las señales SA y ECAP generadas.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice C

Análisis de uso de DAC externo

Como fue expuesto en el Cap. 4, fue estudiada la posibilidad de introducir un DAC y un ADC externos a la solución. Por motivos de prioridades en cuanto a tiempo de organización del proyecto, y dado que resultaba un agregado extra, se decidió no continuar con su implementación. A continuación se especifican las pruebas y resultados obtenidos para la utilización del DAC externo. No se realizaron pruebas para el ADC.

El DAC externo escogido para cumplir con los requermimientos de la aplicación es el modelo AD5443 de 12 bits de Analog Devices [24].

C.1. Interfaz de Comunicación

La comunicación con el DAC externo se establece mediante el protocolo SPI. Los datos son enviados hacia el DAC en palabras de 16 bits, compuesta por 4 bits de control y 12 bits de datos (Ver Fig. C.1).

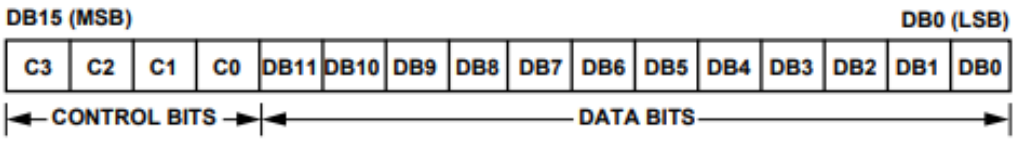


Figura C.1: Formato de la palabra de 16 bit para comunicación con DAC externo.

A partir de los bits de control se configura la funcionalidad del DAC. En Tab. C.1 se visualizan las combinaciones posibles para bits de control.

En una primera instancia fueron realizadas pruebas para verificar el establecimiento de una comunicación correcta con el DAC. Primero se realizó la configuración del mismo mediante la deshabilitación de la funcionalidad de Daisy-chain, luego Se envió una secuencia de valores de 0x000 a 0xFFF en orden creciente, utilizando el bit de control *Load and Update*. El resultado fue exitoso (Ver Fig.C.2 y Fig. C.3).

C3	C2	C1	C0	Función
0	0	0	0	No operation (Power-on default)
0	0	0	1	Load and Update (Para enviar datos)
0	0	1	0	Initiate Readback (Leer contenido)
1	0	0	1	Daisy-chain Disable
1	0	1	0	Clock data to shift register on rising edge (falling p/d)
1	0	1	1	Clear DAC output to zero scale
1	1	0	0	Clear DAC output to midscale

Tabla C.1: Funcionalidades disponibles a partir de los bits de control. Las combinaciones que no están presentes en la tabla son palabras reservadas.

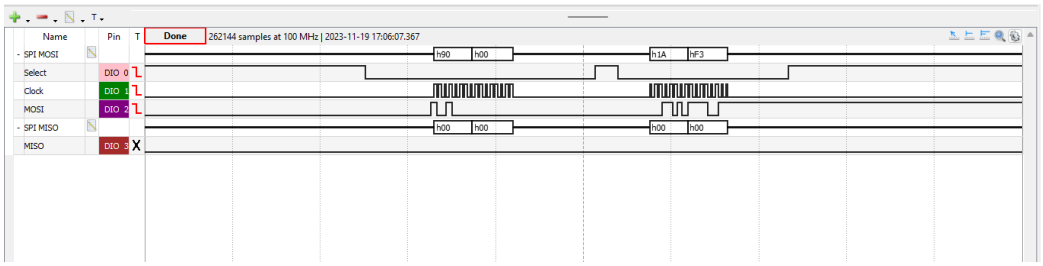


Figura C.2: Comunicación SPI con microcontrolador como maestro y DAC como esclavo. Visaualización a partir de Analizador Lógico.

En la Fig. C.2 se demuestra como se configura el dispositivo enviando la palabra de control 1001 (Daisy-chain disable) seguida de 0s como dato. Luego se utiliza la palabra 0001 (Load and Update) seguida del dato 0xAF3, para iniciar la conversión y visualizar a la salida el valor analógico. En la Fig. C.3 se muestra la salida del DAC al pasarle una secuencia de números en escalera. Cada valor es enviado junto con la palabra de control 0001 como en el caso de la Fig. C.2.

Para poder añadir la implementación del DAC externo de una forma más sencilla al FW, es de importancia mantener la utilización del DMA. Para este caso, el DMA debe ser configurado en asociación a la instancia de SPI del microcontrolador. En caso de seguir estudiando la incorporación de los conversores externos, se debería de continuar con pruebas para utilizar DMA con los mismos.

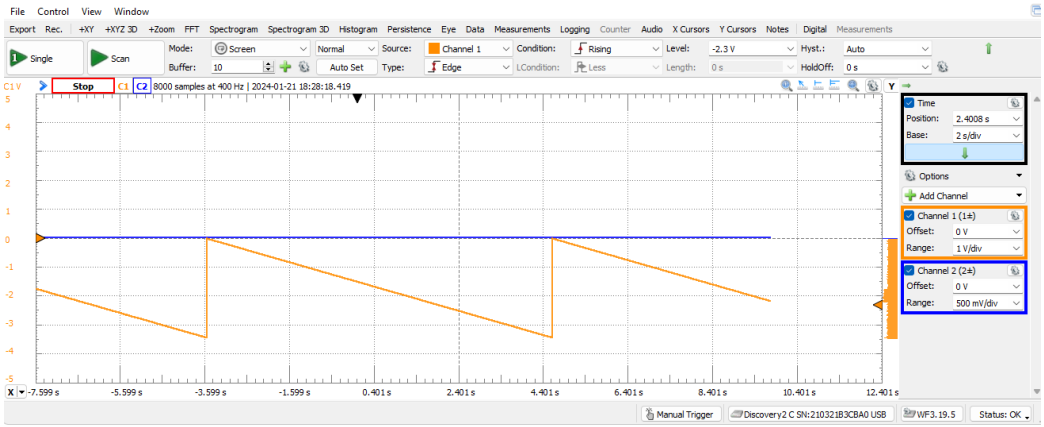


Figura C.3: Comunicación SPI con microcontrolador como maestro y DAC como esclavo. Salida del DAC.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice D

Diseño de electrónica

En este anexo se pueden encontrar los desarrollos teóricos realizados para caracterizar el comportamiento de los diferentes circuitos implementados con el fin de poder diseñar sus parámetros.

D.1. Etapa de Amplificación 1

D.1.1. Elección IC INA

Para seleccionar el INA, una extensa búsqueda fue realizada donde mas de 27 candidatos fueron analizados según sus características. De entre todos ellos, 6 fueron seleccionados en primera instancia como candidatos finales de los cuales el AD8221 de Analog Devices fue elegido.

La selección fue realizada a partir de la comparación de diferentes parámetros del los amplificadores como se puede ver en la Tab. D.1. Donde se utiliza la siguiente escala de colores:

- Verde: Mejores valores dentro de las opciones consideradas.
- Amarillo: Valores aceptables pero menor que otras opciones y/o que significan contar con cuidados en el diseño que tengan en consideración este valor.
- Rojo: Peores valores dentro de las opciones consideradas y/o que requieren tener consideraciones especiales en el diseño para contrarrestar su efecto.

Además es importante mencionar en el caso del ICMR o OSW, el criterio tomado fue el cumplimiento con las especificación de modo común de las señales de interés dadas en la Secc.2.2.3 y la máxima excursión a la salida esperada de 1,4 V como fue descrito en el Cap.4.

En el caso del INA, las características a las cuales se le deben prestar más atención son el ruido en voltaje, el CMRR y el ICMR , Luego, el GBW debe ser tal que, para una ganancia[1] de 10 V/V, el ancho de banda final debe ser mayor

[1]Todos los amplificadores considerados cuentan con una ganancia configurable

a 16 kHz. El SR y tiempo de levantamiento no son las características de mayor interés aunque cuanto mayor el primero y menor el segundo mejor. Debido al filtro DC a la salida del INA, el offset no es de gran repercusión. Todos los amplificadores comparados pueden ser alimentados desde ± 5 V.

Con estas consideraciones, el AD8221 esta dentro de los amplificadores que mejor cumplen las especificaciones del sistema teniendo un costo medio entre las otras opciones consideradas.

# de parte	AD8221	AD8295	AD8429	INA849	INA188	INA166
Precio (USD)	6.78	6.398	10.7	8.81	5.2	6.35
ICMR	OK	OK	DIFF	DIFF	OK	DIFF
OSW	OK	OK	OK	OK	OK	DIFF
R. Vin nV/$\sqrt{}$Hz	8	8	1	1	12.5	1.3
R. Vout nV/$\sqrt{}$Hz	75	75	45	45	118	Sin info
R. I (fA/$\sqrt{}$Hz)	40	40	1500	1100	440	800
GBW (MHz)	0.825	1.2	15	28	0.6	0.45
SR (V/us)	2	2	22	35	0.9	15
Set t (us)	10	10	0.75	0.4	50	3.5
CMRR (dB)	90	90	90	92	84	120
Offs Vin (uV)	60	120	150	35	225	250
Offs Ib (pA)	500	800	150000	20000	2500	1.20E+07

Tabla D.1: Comparación de las parámetros de los 6 INAs evaluados

D.2. Etapa de Amplificación 2

En la segunda etapa de amplificación del hardware mencionado, se implementó un amplificador en configuración diferencial para realizar la resta entre la plantilla producida por el MCU y la señal de entrada proveniente del INA. El circuito se puede observar en la Fig. D.1 que es análogo al mostrado en la Fig. 4.9 de la sección 4.2.4.

D.2.1. Respuesta en frecuencia

Obtengamos V_{O2} en función de V_{DAC}, V_{INA} y V_{ref} aplicando superposición. Primero, englobemos C_1 y R_1 en una impedancia Z_- y C_2 y R_2 en Z_+:

$$Z_- = Z_+ = Z = \frac{1}{C_c s} + R_{FB} = \frac{1 + R_{FB} C_c s}{C_c s} \tag{D.1}$$

Para $V_{DAC} \neq 0$ y v $V_{INA}, V_{ref} = 0$, contamos con un amplificador en configuración inversora:

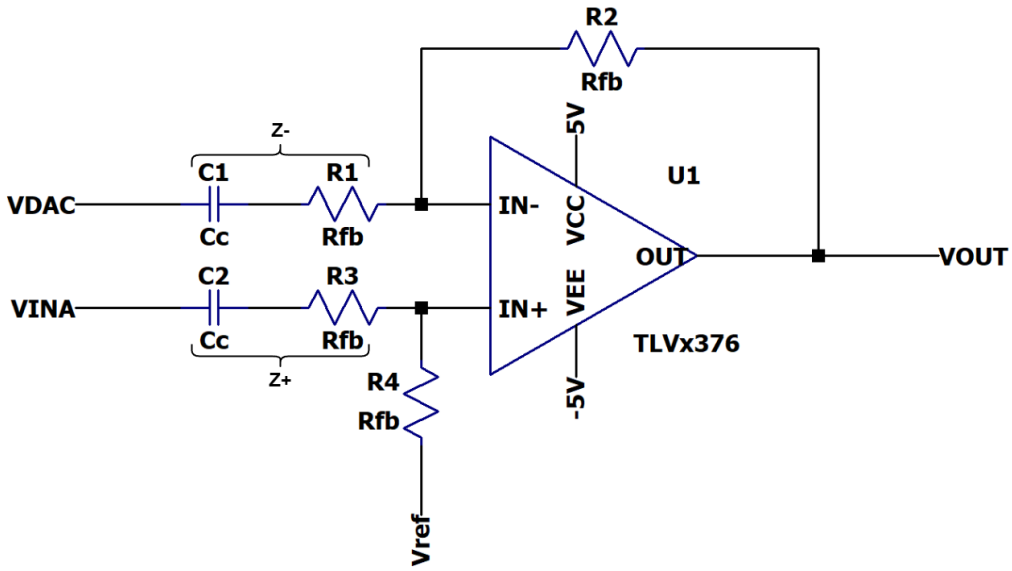


Figura D.1: Circuito correspondiente a la segunda etapa de amplificación donde se implementa una restador a partir de un amplificador en configuración diferencial.

$$V_{O2} = -\frac{R_{FB}}{Z}V_{DAC} = \frac{-R_{FB}C_c s}{1 + RC_c s}V_{DAC} \tag{D.2}$$

Para $V_{INA} \neq 0$ y v $V_{DAC}, V_{ref} = 0$, obtenemos un amplificador no inversor:

$$V_{O2} = \left(1 + \frac{R_{FB}}{Z}\right)\frac{R_{FB}}{R_{FB} + z}V_{INA} = \frac{R_{FB}}{Z}V_{INA} = \frac{R_{FB}C_c s}{1 + RC_c s}V_{INA} \tag{D.3}$$

Por último, para $V_{ref} \neq 0$ y $V_{DAC}, V_{INA} = 0$, estamos en el mismo caso que el anterior:

$$V_{O2} = \left(1 + \frac{R_{FB}}{z}\right)\frac{z}{R_{FB} + z}V_{ref} = V_{ref} \tag{D.4}$$

Sumando todas las expresiones anteriores obtenemos:

$$V_{O2} = V_{ref} + \frac{-R_{FB}C_c s}{1 + R_{FB}C_c s}(V_{INA} - V_{DAC}) \tag{D.5}$$

Entonces, contamos con un filtro pasa altos de frecuencia de corte:

$$f_1 = \frac{1}{2\pi R_{FB}C_c} \tag{D.6}$$

Luego, el amplificador impondrá una frecuencia de corte pasa bajos que depende del factor de realimentación $1 + R_4/R_1 = 2$. La frecuencia de corte será $f_2 = \frac{f_t}{1+R_4/R_1}$, donde f_t es el producto por ancho de banda del amplificador, en este caso de 5,5 MHz y por ende $f_2 = 2{,}75MHz$.

D.2.2. CMRR

En nuestra aplicación, el CMRR determinará el nivel de precisión con el cual se podrá efectuar la resta entre la señal producida por el DAC y el artefacto. A pesar de que un amplificador diferencial ideal cuenta con CMRR infinito, cuando se utilizan componentes discretos con cierta tolerancia, el CMRR queda determinado por el desapareo de dichos componentes.

Por ende, fue de interés durante el diseño de la electrónica, poder cuantificar cual es el mínimo CMRR para cierta tolerancia en los componentes usados. En el siguiente desarrollo se obtendrá el CMRR del amplificador en función de la tolerancia de las resistencias usadas. El circuito utilizado se puede ver en la Fig. D.2

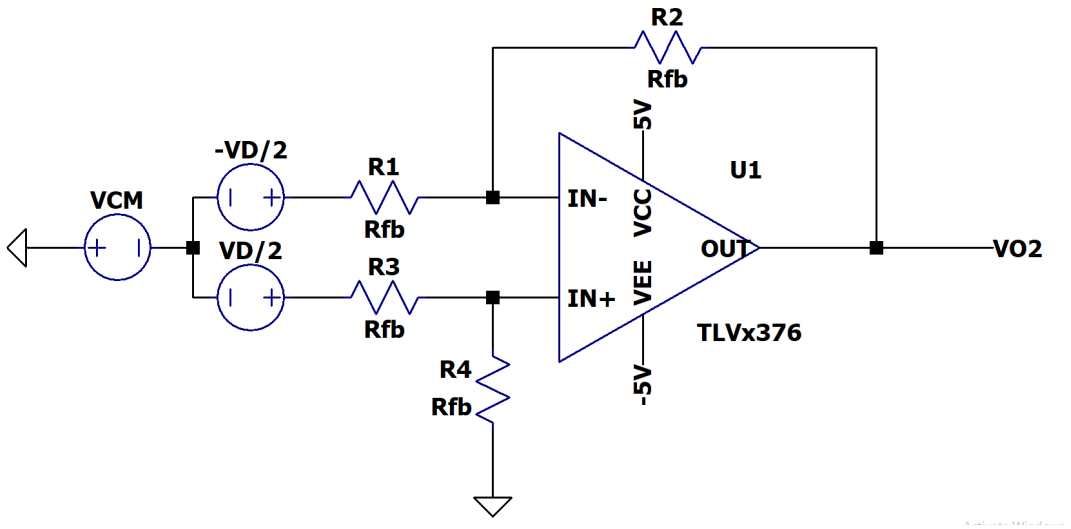


Figura D.2: Circuito utilizado para el cálculo del CMRR del amplificador diferencia..

Podemos obtener rápidamente que:

$$V_{\mathrm{OUT}} = \frac{\left(\frac{R_4}{R_3+R_4}\right)\left(V_{CM}+\frac{V_{\mathrm{D}}}{2}\right)-\left(\frac{R_2}{R_1+R_2}\right)\left(V_{CM}-\frac{V_{\mathrm{D}}}{2}\right)}{\left(\frac{R_1}{R_1+R_2}\right)} \tag{D.7}$$

La ganancia diferencial y en modo común queda:

$$\begin{aligned} A_D &= \frac{V_{\mathrm{OUT}}}{V_{\mathrm{D}}} = \frac{1}{2}\frac{\left(\frac{R_4}{R_3+R_4}\right)+\left(\frac{R_2}{R_1+R_2}\right)}{\left(\frac{R_1}{R_1+R_2}\right)} \\ A_{CM} &= \frac{V_{OUT}}{V_{CM}} = \frac{\left(\frac{R_4}{R_3+R_4}\right)-\left(\frac{R_2}{R_1+R_2}\right)}{\left(\frac{R_1}{R_1+R_2}\right)} \end{aligned} \tag{D.8}$$

Luego, el CMRR:

$$CMRR = \frac{1}{2}\frac{\left(\frac{R_4}{R_3+R_4}\right)+\left(\frac{R_2}{R_1+R_2}\right)}{\left(\frac{R_4}{R_3+R_4}\right)-\left(\frac{R_2}{R_1+R_2}\right)} = \frac{1}{2}\frac{2R_2R_4+R_1R_4+R_2R_3}{R_1R_4-R_2R_3} \tag{D.9}$$

En el caso ideal en que todas las resistencias son iguales, el CMRR es infinito. En cambio, si las resistencias tienen una tolerancia t, es decir $R_i \in [R_{iN} \times (1-t), R_{iN} \times (1+t)], \forall i \in 1,2,3,4$, obtenemos un CMRR finito, cuyo peor caso se da cuando R_1 y R_4 toman el valor máximo, mientras que R_2 y R_3 cuentan con el mínimo. Entonces:

$$\text{CMRR} = \frac{1}{2}\frac{2R_{2N}R_{4N}(1-t)(1+t)+R_{1N}R_{4N}(1+t)^2+R_{2N}R_{3N}(1-t)^2}{R_{1N}R_{4N}(1+t)^2-R_{2N}R_{3N}(1-t)^2} \tag{D.10}$$

$$\text{CMRR} = \frac{G+1+t^2(1-G)}{4t} \tag{D.11}$$

Donde G es el cociente $\frac{R_{2N}}{R_{1N}} = \frac{R_{4N}}{R_{3N}}$. En nuestro caso particular en que todas las resistencias son iguales, $G = 1$, entonces:

$$CMRR = \frac{1}{2t} \tag{D.12}$$

D.2.3. Elección Amplificador

Para implementar el amplificador diferencial se consideraron un total de 5 amplificadores como puede verse en la tabla comparativa de sus parámetros, la Tab.D.2.

Para seleccionar el amplificador se tuvieron la siguientes consideraciones en cuenta:

- El amplificador se encuentra en configuración diferencial con ganancia de 1 V/V, la frecuencia de corte pasa bajos queda definida como $GBW/2$. Luego, al ser una de las entradas del circuito V_{DAC} , la salida del DAC compuesta por incrementos de voltaje en escalones, se desea una respuesta rápida en frecuencia del restador. Esto se debe a que permite minimizar el error entre muestra y muestra entre la señal reproducida por el DAC y la de entrada (V_{INA}). Por ende, cuanto mayor sea la frecuencia de corte pasa bajos, más rápida será la respuesta del restador y se buscaron amplificadores con $GBW > 5$ MHz. Relacionado a este análisis, también se busca el mayor SR y menor tiempo de levantamiento posible.

- Se busca minimizar el ruido equivalente de entrada al ser esta una etapa previa a la amplificación de 60 dB.

- Como ya se vio, el CMRR de esta etapa limita el desempeño de la cancelación del artefacto y por ende se busca maximizar este parámetro en nuestra elección.
- Todos los amplificadores comparados pueden ser alimentados de ±5 V como de solamente 3,3 V y son rail-to-rail.

Con estas consideraciones, el TLVx376 de Texas Instruments es el amplificador que cumple todas las características deseadas con el menor costo de todas las opciones.

# de parte		**AD8605**	**TLVx376**	**AD8691**	**AD8646**	**OPx84**
Precio (USD)		2.68	0.98	1.78	2.7	4.82
Ruido Vin (nV/$\sqrt{}$Hz)		8	8	8	8	3.9
Señal	**GBW (MHz)**	10	5.5	10	24	3.25
	SR (V/us)	5	2	5	11	1.65
	Set t (us)	1	2	1	0.3	2.5
	CMRR (dB)	80	78	68	62	86
Offset Vin (uV)		300	125	2000	2500	150

Tabla D.2: Comparación de las parámetros de los 5 amplificadores considerados para la segunda etapa de amplificación.

D.3. Etapa de Amplificación 3

En esta etapa se busca realizar la amplificación del resultado de la resta efectuada en la etapa 2 por un factor de 60 dB y además realizar un filtro activo de la señal. El circuito diseñado se puede ver en la Fig. D.3.

Primero, calculemos V_F en función de V_{O3}:

$$\frac{V_{O3}}{R_5} = -V_F C_1 s \rightarrow V_F = \frac{-V_{O3}}{R_5 C_1 s} \tag{D.13}$$

Entonces:

$$V_{1+} = \frac{-R_4}{R_3}\frac{-V_{O3}}{R_5 C_1 s} \tag{D.14}$$

Ahora, tenemos que:

$$V_{O3} = \frac{A(V_{1+} - V_{1-})}{1 + \frac{sA}{\omega_t}} \tag{D.15}$$

Donde A es la ganancia del amplificador y ω_t su producto por ancho de banda. Para obtener V_{O3} nos falta V_{1-}:

$$\frac{V_{O2} - V_{1-}}{R_1} = \frac{V_{1-} - V_{O3}}{R_2} \rightarrow V_{1-} = \frac{R_2 V_{O2}}{R_1 + R_2} + \frac{R_1 V_{O3}}{R_1 + R_2} \tag{D.16}$$

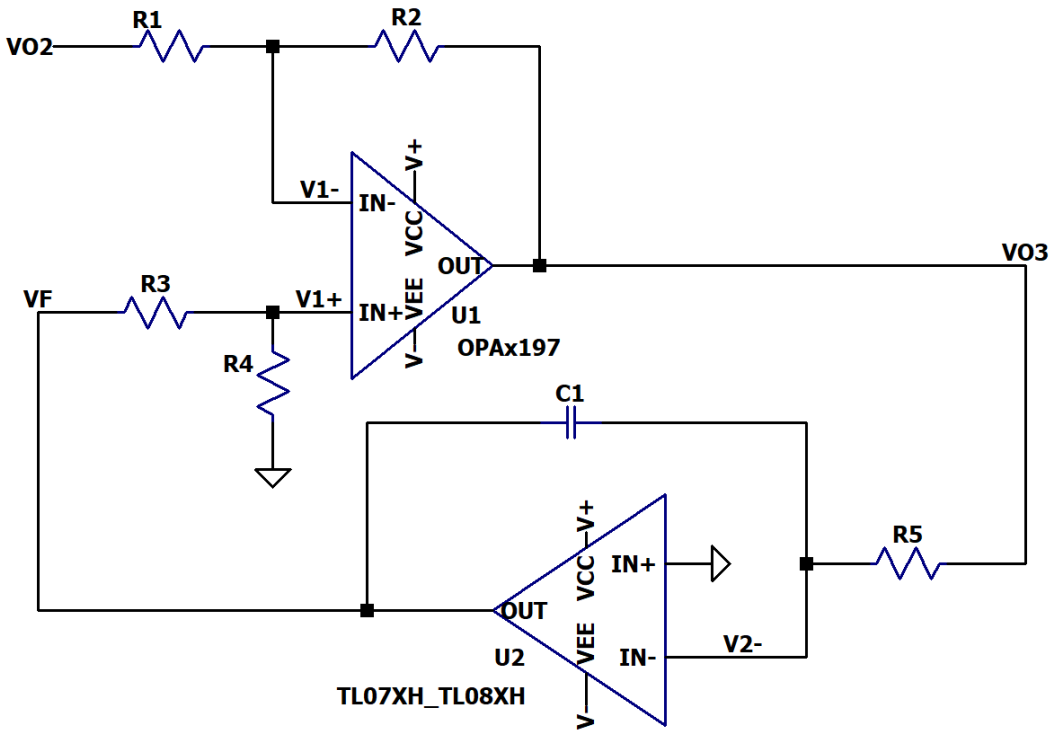


Figura D.3: Filtro pasa banda con ganancia 60dB en banda pasante implementado para la etapa 3.

Combinando (D.14), (D.15) y (D.14), obtenemos:

$$V_{O3}\frac{1+\frac{sA}{\omega_t}}{A} = \frac{-R_4}{R_3+R_4}\frac{V_{O3}}{R_5C_1s} - \left(\frac{R_2V_{O2}}{R_1+R_2} + \frac{R_1V_{O3}}{R_1+R_2}\right) \tag{D.17}$$

Operando sobre la expresión, llegamos a que:

$$V_{O3} = \frac{-R_2}{R_1}\frac{C_1s}{\frac{R_4(R_1+R_2)}{R_1R_5(R_3+R_4)} + (1+\frac{R_1+R_2}{R_1A})C_1s + \frac{R_1+R_2}{R_1\omega_t}C_1s^2} \tag{D.18}$$

Que corresponde a un filtro pasa banda de primer orden. Para obtener una amplificación de 60 dB, necesariamente $R_2 = 1000R_1$, sustituyendo esto en la ecuación anterior:

$$V_{O3} = 1000\frac{C_1s}{\frac{R_41001}{R_5(R_3+R_4)} + (1+\frac{1001}{A})C_1s + \frac{1001}{\omega_t}C_1s^2} \tag{D.19}$$

Suponiendo que $A >> 1001$, lo cual para el OPAx197 la ganancia en lazo abierto A es al menos de 100 dB, osea, cien veces mayor. Con esto y sacando de factor común el término $1001C_1/\omega_t$:

$$V_{O3} = \frac{1000}{1001}\frac{\omega_ts}{\frac{R_4\omega_t}{R_5C_1(R_3+R_4)} + \frac{\omega_t}{1001}s + s^2} \tag{D.20}$$

Sea $R_4 = kR_3$:

$$V_{O3} = \frac{1000}{1001} \frac{\omega_t s}{\frac{k\omega_t}{R_5 C_1 (k+1)} + \frac{\omega_t}{1001} s + s^2} \tag{D.21}$$

Definimos $\omega_n^2 = \frac{k\omega_t}{R_5 C_1 (k+1)}$ y $2\zeta\omega_n = \frac{\omega_t}{1001}$ entonces:

$$\omega_n = \sqrt{\frac{k\omega_t}{(k+1)}} \cdot \sqrt{\frac{1}{R_5 C_1}} \rightarrow \zeta = \frac{1}{2} \frac{\omega_t}{1001} \sqrt{\frac{(k+1) R_5 C_1}{k\omega_t}} = \frac{1}{2{,}1001} \sqrt{\frac{\omega_t R_5 C_1 (k+1)}{k}} \tag{D.22}$$

Suponemos $\zeta > 1$, que se da si y solo si:

$$\frac{2002^2}{\omega_t^2} < R_5 C_1 \frac{(k+1)}{k} \tag{D.23}$$

En este caso, los polos de la transferencia quedan en:

$$p_1 = \zeta\omega_n - \omega_n \sqrt{\zeta^2 - 1} \tag{D.24}$$

$$p_2 = \zeta\omega_n + \omega_n \sqrt{\zeta^2 - 1} \tag{D.25}$$

Ahora, obtengamos ζ y ω_n en función de los dos polos para poder tener las ecuaciones de diseño de los componentes discretos. Primero, vemos que al sumar los polos obtenemos:

$$p_1 + p_2 = 2\zeta\omega_n \rightarrow \omega_n = \frac{p_1 + p_2}{2\zeta} \tag{D.26}$$

Entonces:

$$p_1 = \frac{p_1 + p_2}{2} - \frac{p_1 + p_2}{2} \sqrt{1 - \frac{1}{\zeta^2}} \rightarrow \zeta = \left[1 - \left(\frac{p_2 - p_1}{p_1 + p_2} \right)^2 \right]^{-1/2} \tag{D.27}$$

Luego:

$$\omega_n = \frac{(p_1 + p2)}{2} \sqrt{1 - \left(\frac{p_2 - p_1}{p_1 + p_2} \right)^2} \tag{D.28}$$

Combinando las ecuaciones obtenidas en (D.22), (D.27) y (D.28):

$$\omega_t R_5 C_1 \frac{k+1}{k} = \frac{2002^2}{1 - (\frac{p_2 - p_1}{p_1 + p_2})^2} \tag{D.29}$$

$$\left(\frac{(p_1 + p2)}{2} \right)^2 \left(1 - \left(\frac{p_2 - p_1}{p_1 + p_2} \right)^2 \right) = \frac{\omega_t}{R_5 C_1} \frac{k}{k+1} \tag{D.30}$$

Realizando el cociente de las dos ecuaciones anteriores se tiene:

$$\left(\frac{(p_1+p2)}{2}\right)^2\left(1-\left(\frac{p_2-p_1}{p_1+p_2}\right)^2\right)=\frac{\omega_t^2}{\frac{2002^2}{1-\left(\frac{p_2-p_1}{p_1+p_2}\right)^2}}\rightarrow\left(\frac{(p_1+p2)}{2}\right)^2=\frac{\omega_t^2}{2002^2} \tag{D.31}$$

Lo que resulta en:

$$p_1+p_2=\frac{\omega_t}{1001} \tag{D.32}$$

Lo que, pasado a frecuencia equivale a que $f_1+f_2=f_t/1001$. Una vez elegidos los polos tal que respeten esta igualdad dado el f_t del amplificador, se eligen k y la constante R_5C_1 tal que se respeten las ecuaciones (D.30), (D.29), y que se respete la desigualdad (D.23).

Para el amplificador elegido, el OPAx197, cuenta con un $f_t = 10\,\text{MHz}$, $f_t/1001 = 9{,}9\,\text{kHz}$. En el diseño, se eligieron las frecuencias teóricas para los polos de $f_1 = 500\,\text{Hz}$ y $f_2 = 9{,}4\,\text{kHz}$. Luego, se tomó $k = 10$, para lo cual la constante R_5C_1 queda definida en $0{,}302\,\text{s}$, se tomaron $C_1 = 1\,\mu\text{F}$ y $R_5 = 300\,\text{k}\Omega$. Valores para los cuales se verifica (D.23).

D.3.1. Elección Amplificador

Para realizar la selección se tuvo consideraron más de 13 amplificadores de diferentes fabricantes de los cuales solo 10 se encuentran en la Tab. D.3. La elección del amplificador correcto se basó en:

- El amplificador debe contar con un GBW mayor a mil veces el ancho de banda de la etapa, aproximadamente 10 kHz. Es decir, $GBW > 10\,\text{MHz}$.
- Al ser una etapa de alta ganancia, el ruido equivalente a la entrada debe minimizarse con la elección.
- Bajo tiempo de levantamiento y alto SR son preferidos.
- Debido al uso de la realimentación de continua, el offset del amplificador es corregido por la propia alimentación.
- Todos los amplificadores mostrados son rail-to-rail en su salida. Esta cualidad es clave para evitar la saturación al amplificar el residuo de la cancelación del artefacto de estimulación.

Con todas estas consideraciones, el OPA197 corresponde a la mejor opción. El NE5534 puede verse como una mejor alternativa al OPA197 pero no fue seleccionado debido a que en su datasheet, la excursión de salida no se encontraba bien defina.

# de parte	(USD)	Ruido Vin (nV/$\sqrt{}$Hz)	Señal				Offset Vin (uV)
			GBW (MHz)	SR (V/us)	Sett. t (us)	CMRR (dB)	
ADA4851	2.09	12	105	375	0.055	90	600
ADA4084	4.95	3.9	15.9	3.7	4	106	200
AD8675	4.04	2.8	10	2.5	Sin info	105	240
ADA4625	11.5	3.3	16	32	0.95	74	80
LTC6240	4.47	10	12	5	0.9	80	400
ADA4805	3.44	5.2	120	190	0.035	103	125
NE5534	0.78	4	10	13	0.02	70	500
OPA197	1.77	5.5	10	20	1.8	100	100
OPA863	1.72	5.9	50	105	0.07	100	400
OPA1641	1.07	5.1	11	20	Sin info	120	1000

Tabla D.3: Comparación de las parámetros de los 10 amplificadores considerados para la tercer etapa de amplificación.

D.4. Filtro Discreto a la Salida

A continuación se enseña el cálculo de la transferencia del filtro discreto a utilizar, cuyo modelo se estipula en la Fig. D.4. Al ser un filtro de carácter pasivo, no se puede lograr una ganancia mayor que 1 V/V, sino que se puede acercar a ella desde abajo. El diseño de este filtro implica encontrar el equilibrio entre el compromiso de ancho de banda y ganancia.

En las siguientes ecuaciones se desarrolla el cálculo de su transferencia.

Planteando ley de nudos se obtienen los siguientes resultados.

$$(V_{in} - V_1)\, C_1 s = \frac{(V_{in} - V_{out})}{R_2} + \frac{V_1}{R_1} \tag{D.33}$$

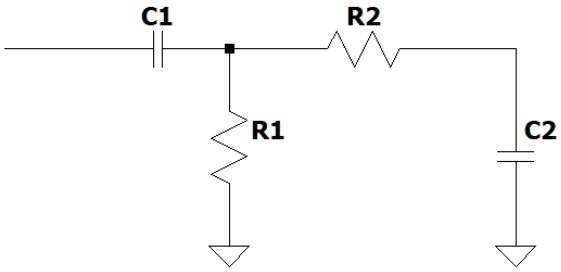


Figura D.4: Modelo del filtro discreto a implementa.r

$$V_{in} C_1 s + \frac{V_{out}}{R_2} = V_1 \left(\frac{1}{R_1} + C1_s + \frac{V_1}{R_2} \right) \tag{D.34}$$

Ahora, por divisor de tensión se obtiene:

$$V_{out} \;=\; V_1 \,\frac{1/C_2 s}{1/C_2 s \;+\; R_2} \tag{D.35}$$

Y por lo tanto, juntando los resultados de las Ec. D.34 y Ec. D.35 se tiene:

$$V_{in} C_1 s \;+\; \frac{V_{out}}{R_2} \;=\; V_{out}\left(\frac{1}{C_2 s} \;+\; R_2\right)\, C_2 s\, \left(\frac{1}{R_1} \;+\; \frac{1}{R_2} \;+\; C_1 s\right) \tag{D.36}$$

$$V_{in} C_1 s \;=\; V_{out}\left(\left(\frac{1}{C_2 s} \;+\; R_2\right)\, C_2 s\, \left(\frac{1}{R_1} \;+\; \frac{1}{R_2} \;+\; C_1 s\right) \;-\; \frac{1}{R_2}\right) \tag{D.37}$$

$$V_{in} C_1 s \;=\; V_{out}\left((1 \;+\; R_2 C_2 s)\left(\frac{1}{R_1} \;+\; \frac{1}{R_2} \;+\; C_1 s\right) \;-\; \frac{1}{R_2}\right) \tag{D.38}$$

$$V_{in} C_1 s \;=\; V_{out}\left(\frac{1}{R_1} \;+\; \frac{1}{R_2} \;-\; \frac{1}{R_2} \;+\; C_1 s \;+ R_2 C_2 s\left(\frac{1}{R_1} \;+\; \frac{1}{R_2}\right) \;+\; R_2 C_2 C_1 s^2\right) \tag{D.39}$$

$$V_{in} C_1 s \;=\; V_{out}\left(R_2 C_2 C_1 s^2 \;+\; s\left(C_1 \;+ \frac{R_2 C_2}{R_1} \;+\; \frac{R_2 C_2}{R_2}\right) \;+\; \frac{1}{R_1}\right) \tag{D.40}$$

$$\frac{V_{out}}{V_{in}} \;=\; \frac{C_1 s}{R_2 C_2 C_1 \;\left(s^2 \;+\; s\left(\frac{1}{R_2 C_2} \;+\; \frac{1}{R_1 C_1} \;+\; \frac{1}{R_2 C_1}\right) \;+\; \frac{1}{C_1 C_2 R_1 R_2}\right)} \tag{D.41}$$

Finalmente, la transferencia se puede reescribir como:

$$\frac{V_{out}}{V_{in}} \;=\; \frac{s\; 1/R_2 C_2}{\left(s^2 \;+\; s\left(\frac{1}{R_2 C_2} \;+\; \frac{1}{R_1 C_1} \;+\; \frac{1}{R_2 C_1}\right) \;+\; \frac{1}{C_1 C_2 R_1 R_2}\right)} \tag{D.42}$$

Para el diseño de los componentes se utilizarán las siguientes ecuaciones:

$$m \;=\; \sqrt{2Q} \tag{D.43}$$

Donde Q es el factor de calidad.

$$A \;=\; \frac{1 \;+\; \sqrt{1 \;-\; m^3}}{2} \tag{D.44}$$

Donde A es la ganancia que tendrá el filtro.

$$R_1 \;=\; \frac{A}{2\pi f_0 Q C_1} \tag{D.45}$$

Donde f_0 es la frecuencia central

$$R_2 = \left(\frac{m}{1 - m}\right) R_1 \tag{D.46}$$

$$C_2 = \frac{Q}{2\pi f_0 A R_2} \tag{D.47}$$

Se fijó $C_1 = 100\ nF$ y $C_1 = 1\ nF$, por ser valores ya utilizados en otros componentes, $f_0 = 2{,}25\ kHz$ y $Q = 0{,}3$ pues se busca los polos tal que $f_{p2} > 10\ f_{p1}$ y que el ancho de banda contenga la señal de ECAP. Este factor de calidad fija la ganancia en$A = 0{,}86V/V$. Con estos valores y utilizando las ecuaciones anteriores se obtiene $R_2 = 24{,}5\ k\Omega$ y $R_1 = 2{,}04\ k\Omega$.

Simulando en LTSPice y teniendo en cuenta los valores de resistencia disponibles, se optó por los valores $R_2 = 30\ k\Omega$ y $R_1 = 6{,}34\ k\Omega$, donde los polos quedan posicionados en $f_{p1} = 224\ Hz$ y $f_{p2} = 5{,}6\ kHz$. Se obtuvo una atenuación de $-484\ mdB$.

D.5. Power Management

D.5.1. Convertidor DC/DC -6V

Elección Integrado

La elección del convertidor DC/DC y de la topología a implementar fue dirigida por la capacidad de simular el circuito a partir de modelos de SPICE. Diferentes opciones fueron consideradas, incluyendo varias de Texas Instruments donde se tenía que usar el software de simulación basado en SPICE propietario de Texas, *PSpice for TI* [49]. Desafortunadamente, tras explorar PSpice, este fue descartado al no contar con muchos de los modelos para los convertidores considerados. En cambio, LTSpice de Analog Devices, cuenta con los modelos de los diferentes convertidores del mismo fabricante, entre los cuales se encuentre el LT3462.

Dentro de las opciones dadas por Analog Devices entre las cuales también se incluyen el LT3483 y el LT1617, el LT3462 contaba con el menor precio y mejor eficiencia para las corrientes demandadas esperadas por el HW. Además, debido a la banda acotada a la salida de la etapa de amplificación de 60 dB, cuanto mayor es la frecuencia de operación del DC/DC, mayor será la atenuación de cualquier componente proveniente de la conmutación del DC/DC en etapas previas del sistema. El LT3462, es la opción con mayor frecuencia de conmutación. Además, este IC auspiciado por alcanzar un ripple a la salida de aproximadamente 1 mVpp

Diseño componentes discretos DC/DC

Para dimensionar los componentes discretos se siguió la *Nota de Aplicación Power Topologies Manual* [31] donde se encuentran las ecuaciones de funcionamiento de un convertidor DC/DC en configuración cuk. Se utilizará la Fig.D.5 como referencia para las cuentas en el diseño.

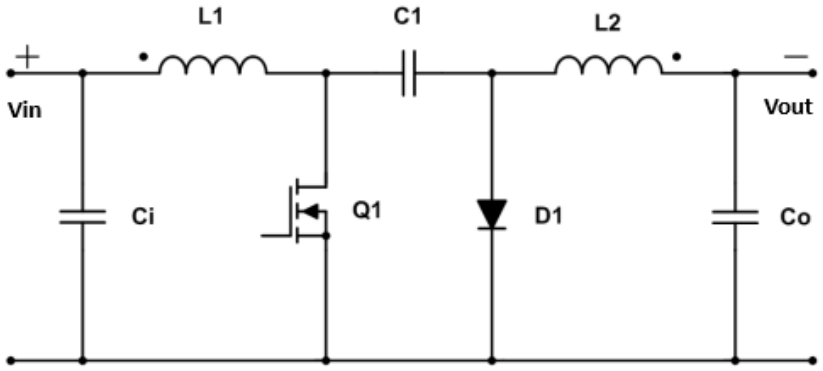


Figura D.5: Circuito de referencia para un cuk converter. [Extraido de [31]]

Antes, de empezar con el análisis, se fijarán las constantes del diseño. El integrado opera a una frecuencia de de $f_s = 2{,}7\,\text{MHz}$ y el diodo Schottky integrado con el que cuenta tiene un voltaje forward máximo $V_f = 0{,}8\,\text{V}$. Luego, $V_{in} = 5\,\text{V}$, $V_{out} = -6\,\text{V}$ y se esperan una corriente máximas a la salida de $I_{out} = 100\,\text{mA}$. Además, en la datasheet del IC se especifica que se tiene un duty cycle máximo de $D_{max} = 0{,}77$.

Primero calculemos los tiempos t_1 y t_2 cuando el transistor Q_1 esta conduciendo y cuando no respectivamente.

$$t_1 = \frac{1}{f_s}\frac{-V_{out} + v_f}{-V_{out} + V_{in} + V_f} = 213\,\text{ns} \tag{D.48}$$

$$t_2 = \frac{1}{f_s} - t_1 = 157\,\text{ns} \tag{D.49}$$

Luego, el duty cycle queda definido como:

$$D = f_s.t_1 = 0{,}576 \tag{D.50}$$

Mientras que la corriente de entrada promedio de:

$$I_{in,avg} = \frac{-(V_{out} - V_f)I_{out}}{V_{in}} = 136\,\text{mA} \tag{D.51}$$

Luego, para calcular los valores de las inductancias, se debe fijar cual es el nivel de variación o ripple en corriente en los inductores. Tras una investigación de diferentes notas de aplicación, se encontró que se recomienda diseñar para que la variación en corriente sea de un 40 % de la corriente promedio. Es decir $I_{ripple} = 0{,}4 \times I_{in,avg} = 54{,}4\,\text{mA}$. Con esto podemos calcular el valor de las inductancias a usar en:

$$L_1 = L_2 = V_{in}\frac{t_1}{I_{ripple}} = 19{,}6\ \mu H \tag{D.52}$$

A partir de este valor, se eligió usar inductores de de 22,0 μH. Para elegir el condensador de flyback C_1 se siguió la recomendación del fabricante del LT3462

en la pagina 6 de su hoja de datos en la que se especifica que este debe ser un condensador cerámico de 1 µF o más y debe estar clasificado para un voltaje nominal mayor a $V_{in} - V_{out}$. Debido a esta razón se eligió el CC0603KRX5R7BB105 de YAGEO [50] de 1 µF que cuenta con una valuación en voltaje de 16 V.

También en la página 6 en la hoja de datos del LT3462 se especifica que la corriente por el diodo integrado durante el encendido no debe superar los 1,5 A. Para caracterizar esta corriente, se proporciona la siguiente ecuación:

$$I_{inrush} = \frac{V_{in} - 0{,}6\,\mathrm{V}}{\sqrt{(\frac{L_1}{C_1} - 1)}} \exp \frac{-\pi}{2\sqrt{(\frac{L_1}{C_1} - 1)}} \tag{D.53}$$

Para los valores de componentes elegidos, queda definida en $I_{inrush} = 0{,}68\,\mathrm{A}$. Por último, para obtener el voltaje $V_{out} = -6\,\mathrm{V}$, es necesario definir las resistencias de realimentación que utiliza el IC a partir de la ecuación de la hoja 5 de la hoja de datos:

$$V_{out} = -1{,}265\,\frac{\mathrm{R}_1}{\mathrm{R}_2} \tag{D.54}$$

Se eligieron $R_1 = 95{,}3\,\mathrm{k\Omega}$ y $R_1 = 20\,\mathrm{k\Omega}$. Por último, se utilizaron condensadores de material dieléctrico X7R y X5R por recomendación del fabricante para los condensadores de entrada y de salida. En particular se usaron valores de 1 µF y 10 µF respectivamente.

D.5.2. Convertidor LDO negativo

La elección del LDO negativo se basó en la elección de usar el LT3462 como convertidor DC/DC para generar una fuente de −6 V de la cual el LDO se pueda alimentar. Dentro de las posibles opciones consideradas se priorizó:

- Modelo SPICE utilizable en LTSpice para poder simular el comportamiento de la generación de la fuente de −5 V.
- Alto PSRR.
- Corriente nominales de salida al menos dos veces la máxima esperada de 50 mA que fue determinada a partir de simulación del AFE y ABE.

Dentro de todas las opciones consideradas, entre las cuales se encuentra los IC LT1964, ADP7182 de Analgo Devices y los TPS723, TPS7A30 y TPS723xx-Q1 de Texas Instruments, el TPS723 fue elegido al contar con el mejor precio entre las opciones mencionadas y un PSRR de 35 dB. Solamente menor al del ADP7182 de −45 dB cuyo precio hacia prohibitiva esta opción.

Apéndice E

CANE STM32-L552ZE Shield PCB Ruteo

En esta sección se pueden encontrar las imágenes correspondiente a la ruteo de la PCB desarrollada en sus 6 capas. Notar la siguiente distribución de colores:

- Amarillo: 5 V
- Naranja: 3,3 V
- Fucsia: −6 V
- Violeta: −5 V
- Gris: GND

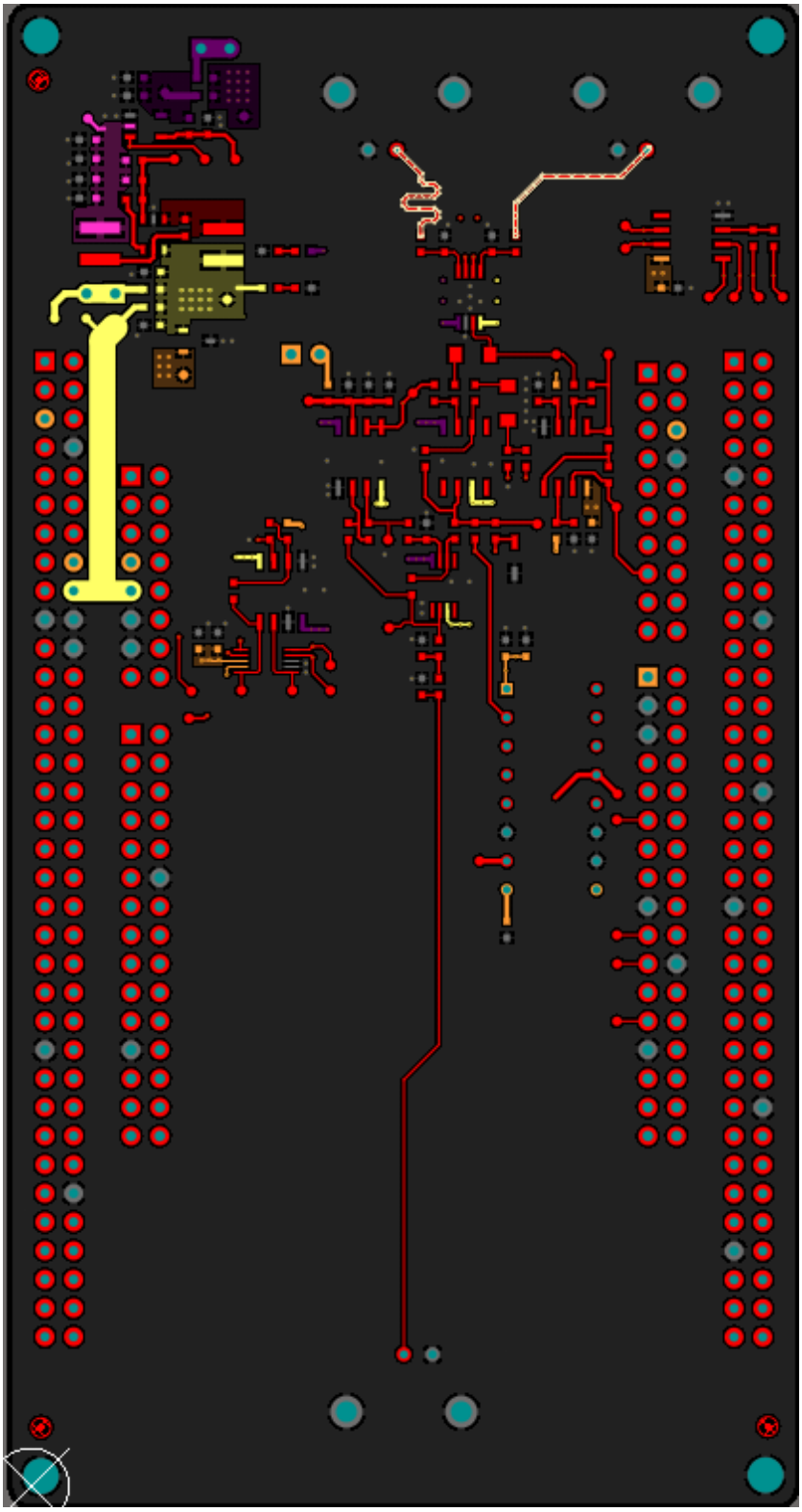


Figura E.1: Top layer del PCB CANE STM32-L552ZE Shield.

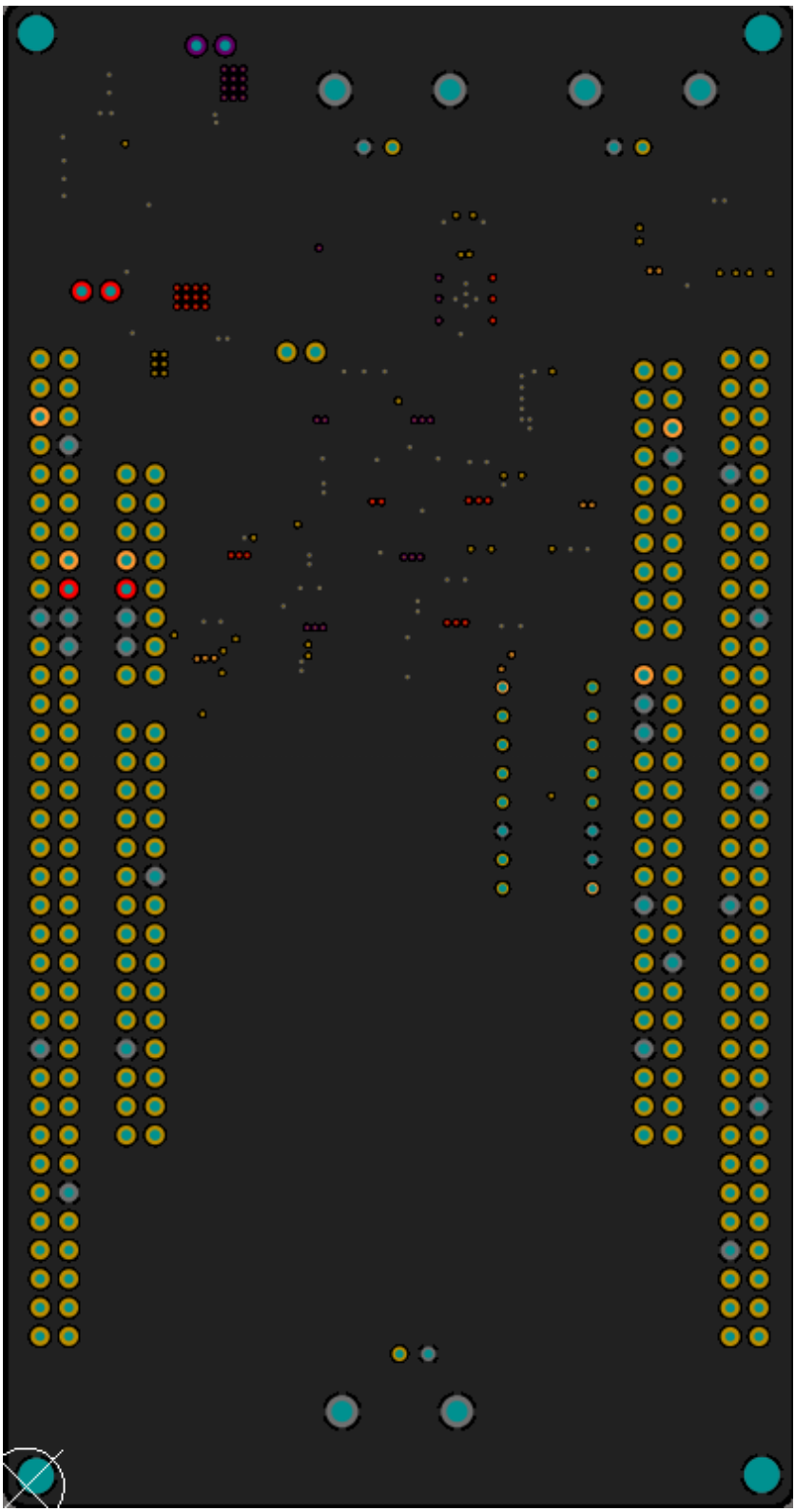


Figura E.2: Segunda capa, GND, del PCB CANE STM32-L552ZE Shield.

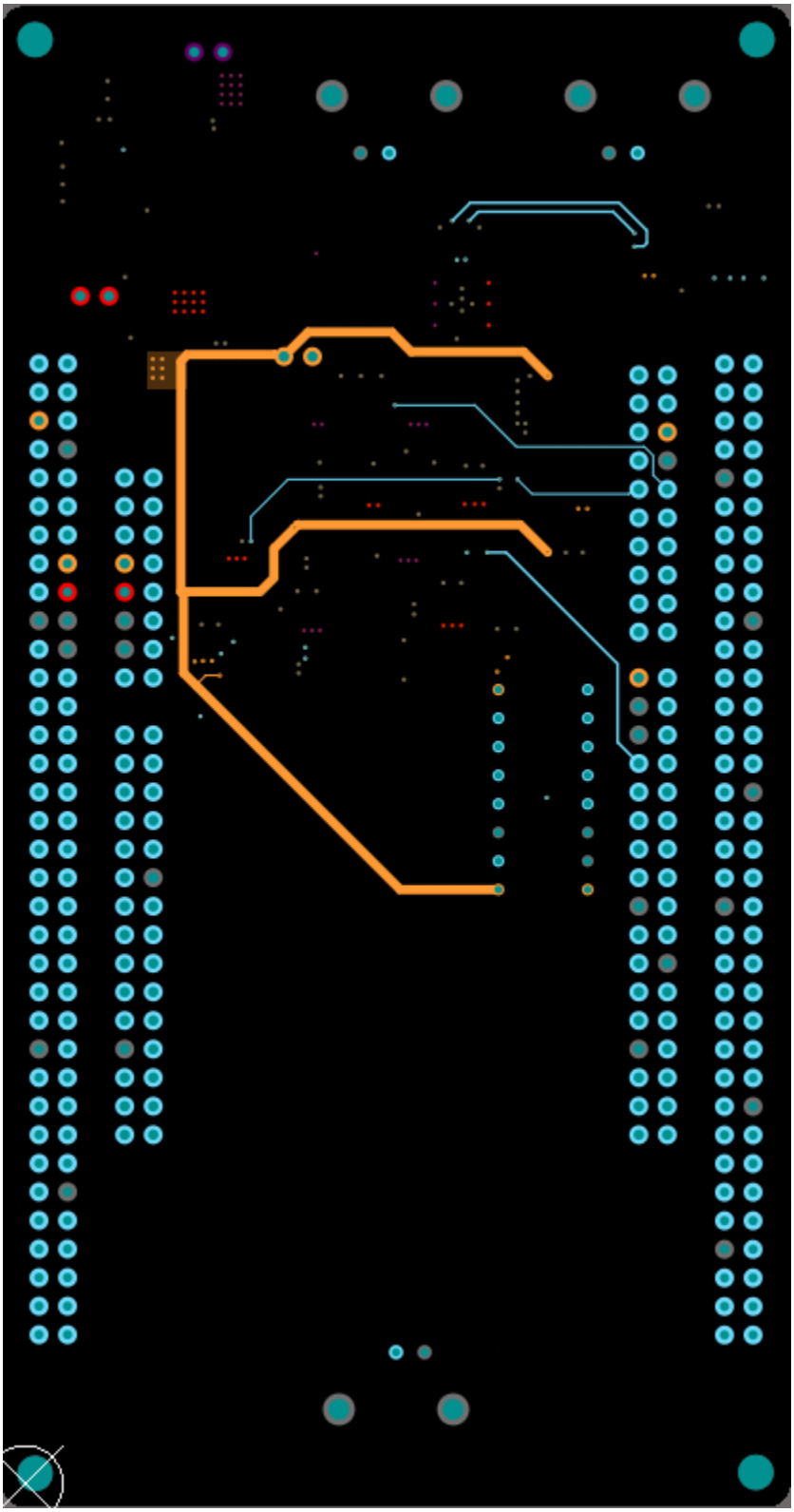


Figura E.3: Tercera capa del PCB CANE STM32-L552ZE Shield.

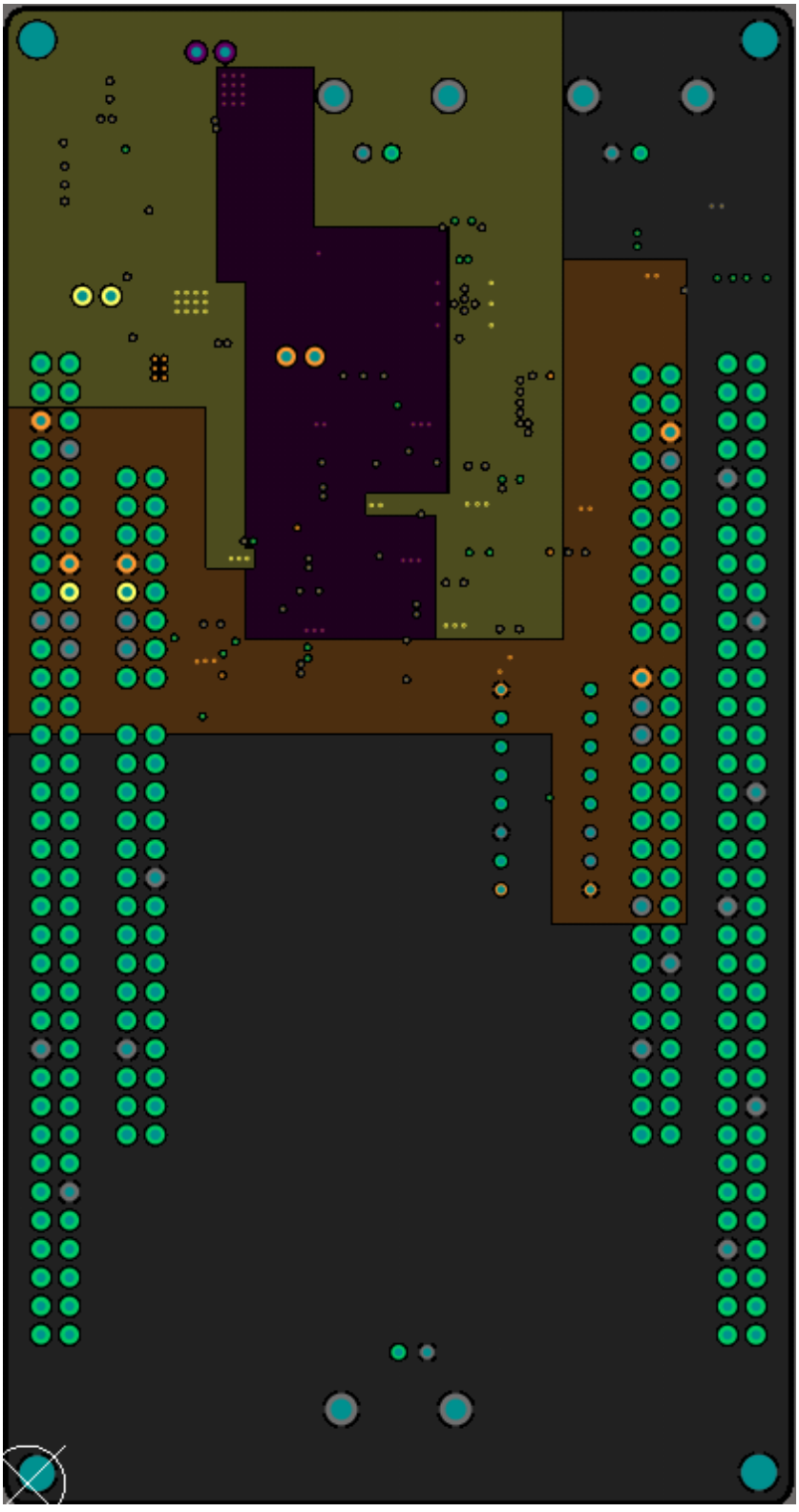


Figura E.4: Cuarta capa del PCB CANE STM32-L552ZE Shield.

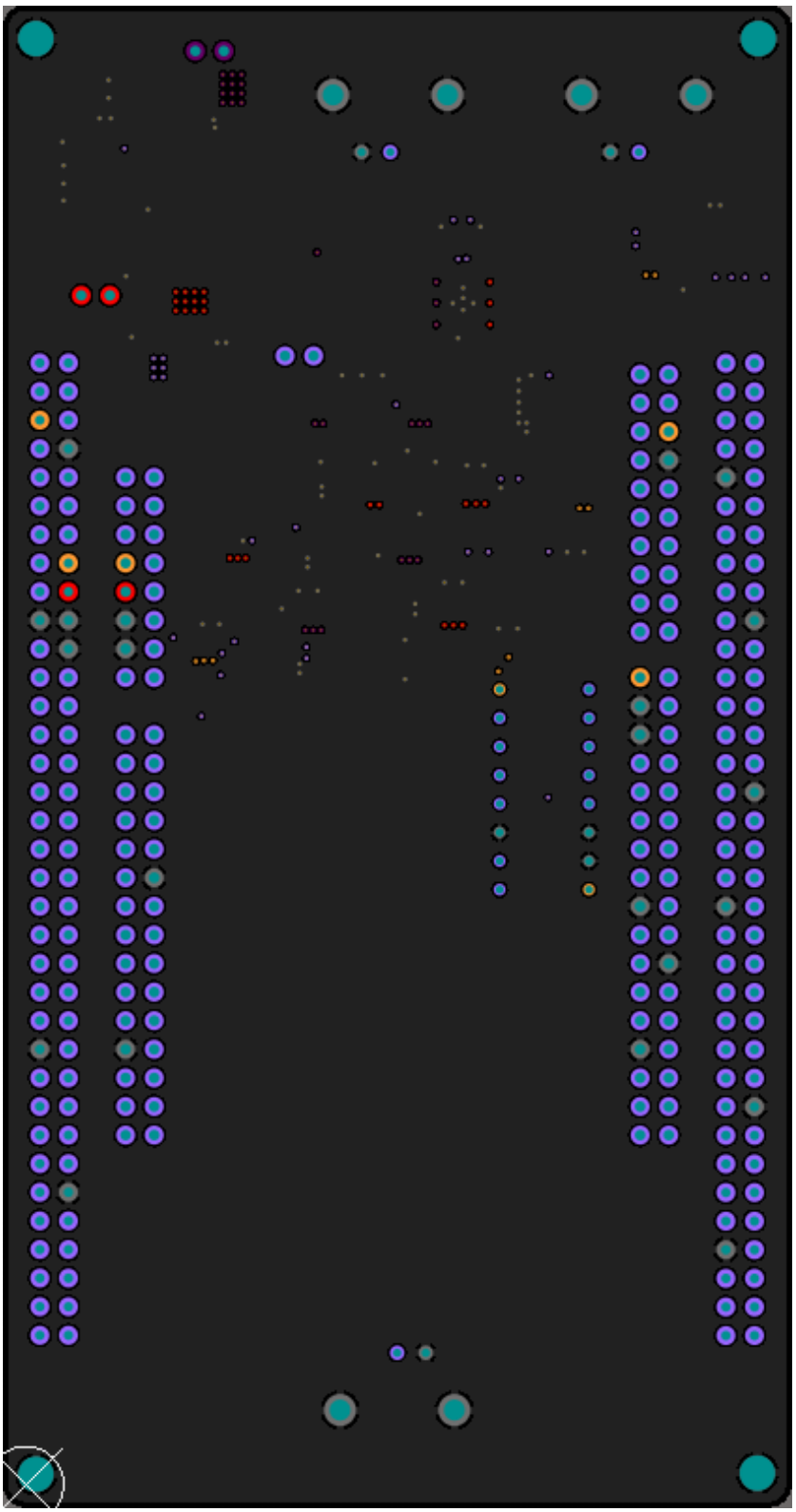


Figura E.5: Quinta capa, GND, del PCB CANE STM32-L552ZE Shield.

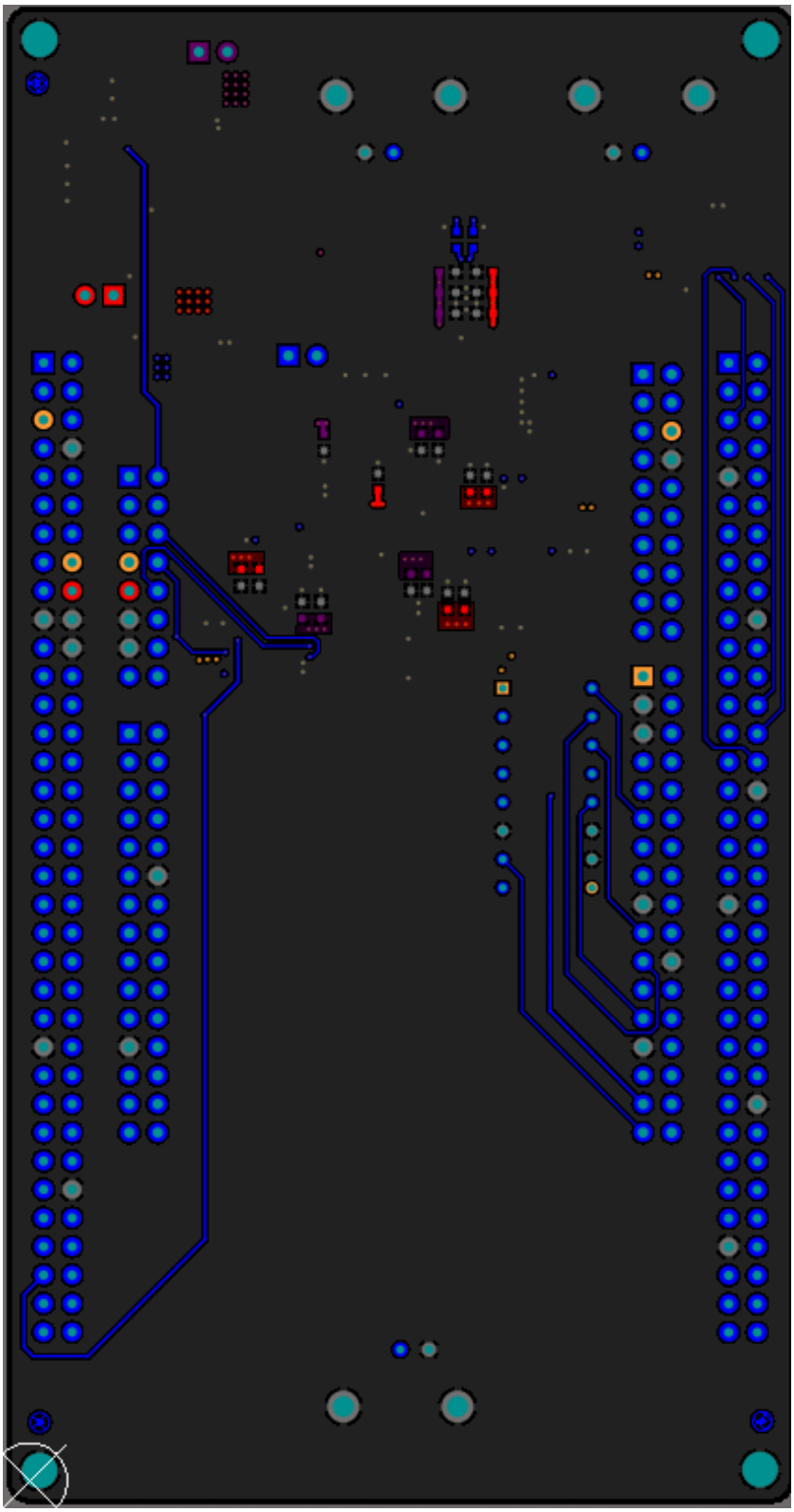


Figura E.6: Bottom Layer del PCB CANE STM32-L552ZE Shield.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice F

Interfaz diseñada AD2

En este anexo se va a encontrar todos los parámetros configurables de la interfaz diseñada para los instrumentos del AD2 así como la configuración del Analog Discovery 2 realizada por la aplicación de post-procesamiento.

F.1. Parámetros Expuestos de la Intefaz con el AD2

AD

Esta clase se encarga de iniciar adecuadamente el dispositivo Analog Dicovery, estableciendo la conexión entre el mismo y la computadora. Para el correcto uso del AD2, esta clase debe ser inicializada primero ante que cualquiera de las clases pertenecientes a los instrumentos independientes.

ADScope

Es la interfaz del osciloscopio y expone las funcionalidades básicas para la configuración de todos los parámetros de los canales y del trigger, así como la ejecución de la adquisición de alguno de los canales.

Los parámetros configurables del instrumento por medio de la interfaz se pueden ver en la Tab. F.1 mientras que el trigger puede ser configurado según los parámetros de la Tab. F.2

ADWavegen

Esta clase expone las funcionalidades del generador de onda analógicas del AD2 a partir de la configuración de los parámetros necesarios. Además, permite la sincronización de algunos, o todos los canales analógicos para la generación coordinada de las ondas necesarias.

ADDigitalOut

Interfaz análoga al generador de onda analógico pero para señales digitales. Con la diferencia de que la generación puede ser sincronizada además con el generador

Parámetro	Descripción
sampling_frequency	Velocidad de muestreo en Hz del osciloscopio. A partir de ella y del largo del buffer se define el periodo de adquisición.
buffer_size	Tamaño del buffer de adquisición. Tras ejecutar una adquisición, se obtiene un a array de largo buffer_size con las muestras adquiridas.
offset	Offset en V del osciloscopio.
amplitude_scale	Amplitud en V del osciloscopio.
probes_x10	Especifica el uso de las puntas de osciloscopio x10.

Tabla F.1: Parámetros configurables de la interfaz del osciloscopio

Parámetro	Descripción
enable	Marca si el trigger para los canales del oscilocopio estó o no activado.
trigger_source	Determina cual es la fuente del trigger que puede ser ninguno, un canal analógico, canal digital o una fuente externa.
channel	El canal correspondiente del trigger.
timeout	Timeout en segundos para el trigger de los canales analógicos.
edge_rising	Configura cual es el flanco en el cual triggerear la adquisición.
level	Valor en V del trigger.

Tabla F.2: Parámetros configurables de la interfaz del osciloscopio para el trigger

de onda analógicas, otro canal digital o incluso el osciloscopio.

F.2. Configuración del AD2 Realizada en el la Aplicación CANE

F.2.1. Generador de Ondas Analógicas

Como fue mencionado previamente, el proceso de adquisición de datos carga a partir de dos archivos de texto, las formas de onda de las señales SA e ECAP. La SA, siempre es generada en el primer canal del generador de onda mientras que la ECAP es generada en el canal 2 cuando es habilitada. En la Tab. F.5 se pueden apreciar las configuraciones del instrumento para la generación de ambas ondas.

Además, el canal analógico 2 es sincronizado con el canal 1. De esta manera, cuando se empieza a reproducir o se detiene la SA, se realizará lo mismo con la ECAP.

Parámetro	Descripción
frequency	Frecuencia de la onda generada en Hz.
amplitude	Amplitud en V de la onda generada.
symmetry	Simetría de la onda generada. Es solo válida en algunas formas de onda.
wait	Tiempo de espera entre la habilitación del generador de onda y el inicio de la generación de la onda deseada.
run_time	Tiempo durante el cual se genera la onda configurada.
repeat_count	Cantidad de veces que se repite la onda generada durante un tiempo run_time.

Tabla F.3: Parámetros configurables de la interfaz del generador de onda.

Parámetro	Descripción
frequency	Frecuencia de la onda generada en Hz.
duty_cycle	Simetría de la onda generada. Es solo válida en algunas formas de onda.
wait	Tiempo de espera entre la habilitación del generador de onda y el inicio de la generación de la onda deseada.
run_time	Tiempo durante el cual se genera la onda configurada.
repeat	Cantidad de veces que se repite la onda generada durante un tiempo run_time.
idle_state	Define el estado inicial del generador de onda previo a al comienzo de la generación de la misma.
trigger_enable	Habilita o no el trigger de la generación de la onda digital.
trigger_source	Selección de la fuente del trigger para la generación de la onda digital.
trigger_edge_rising	Selección del flanco para el trigger de la generación de la onda.
function	Define el tipo de onda a producir, que puede ser un pulso, especificada con por un array o random.

Tabla F.4: Parámetros configurables de la interfaz del generador de onda ditigal.

F.2.2. Generador de Ondas Digitales

Solamente se usan los canales digitales 0 y 1. Las configuraciones de los mismos se pueden ver en las Tab. F.6 y F.7. Notar que el canal uno se configura para dar un flanco de subida en el canal uno que nunca termina.

Parámetro	Valor
frequency	900 Hz
amplitude	La amplitud es un parámetro configurado por el usuario desde la aplicación. Una amplitud es configurada para la SA y otra para la ECAP.
symmetry	Este parámetro no importa cuando la onda generada es custom.
wait	0 s
run_time	0, equivalente a infinito.
repeat_count	0, equivalente a infinito.

Tabla F.5: Parámetros configurados del generador de onda para la generación de la SA

Parámetro	Valor
frequency	No importa
duty_cycle	100 %
wait	0
run_time	1 ms
repeat	No importa
idle_state	Np importa
trigger_enable	False
trigger_source	No importa
trigger_edge_rising	No importa
function	pulse

Tabla F.6: Parámetros configurados para el canal digital 0.

F.2.3. Osciloscopio

En la aplicación de post-procesamiento solo se utiliza el canal uno del osciloscopio para la adquisición de datos mientras que el canal 2 es usado como trigger del primero como se pueden ver en las tablas Tab. F.8 y F.9 respectivamente.

F.3. Mejoras HW

F.3.1. Control de Continua en el Restador

Para soluciona el problema de realimentación de continua, es necesario eliminar los condensadores utilizados para establecer un filtro pasa altos en el restador. Esto genera el problema de como configurar la continua de 1,65 V a la salida del mismo, que puede ser solucionado variando las entradas del restador como muestra la Fig. F.1 donde la DC de la salida la define la plantilla generada por el DAC mientras que la salida del INA siempre se encuentra centrada en 0 V. Aunque hubiera una pequeña variación de la misma, esta podría ser corregido por medio de la plantilla.

Parámetro	Valor
frequency	900 Hz
duty_cycle	50 %
wait	0
run_time	1 Ms
repeat	0, equivalente a infinitas veces.
idle_state	initial
trigger_enable	True
trigger_source	*analog out 1*, lo que equivale a ser sincronizadas con la reproducción de la señal SA en el canál analógico 1.
trigger_edge_rising	False.
function	pulse

Tabla F.7: Parámetros configurados para el canal digital 1.

Parámetro	Valor
sampling_frequency	Parámetro configurado por el usuario con un máximo valor de 7.2MHz
buffer_size	Configurado automáticametne a partir de la frecuencia de muestreo y el periodod de 1,1 ms que se desea muestrear.
offset	0 V
amplitude_scale	5 V
probes_x10	Las puntas de prueba x10 están desactivadas.

Tabla F.8: Parámetros configurados para el canal 1 del osciloscopio

Parámetro	Descripción
enable	True
trigger_source	Analog
channel	Canal 2
timeout	1000 s, para que nunca *triggeree* por timeout
edge_rising	True
level	1,65 V

Tabla F.9: Parámetros del trigger configurados por la aplicación de post procesamiento

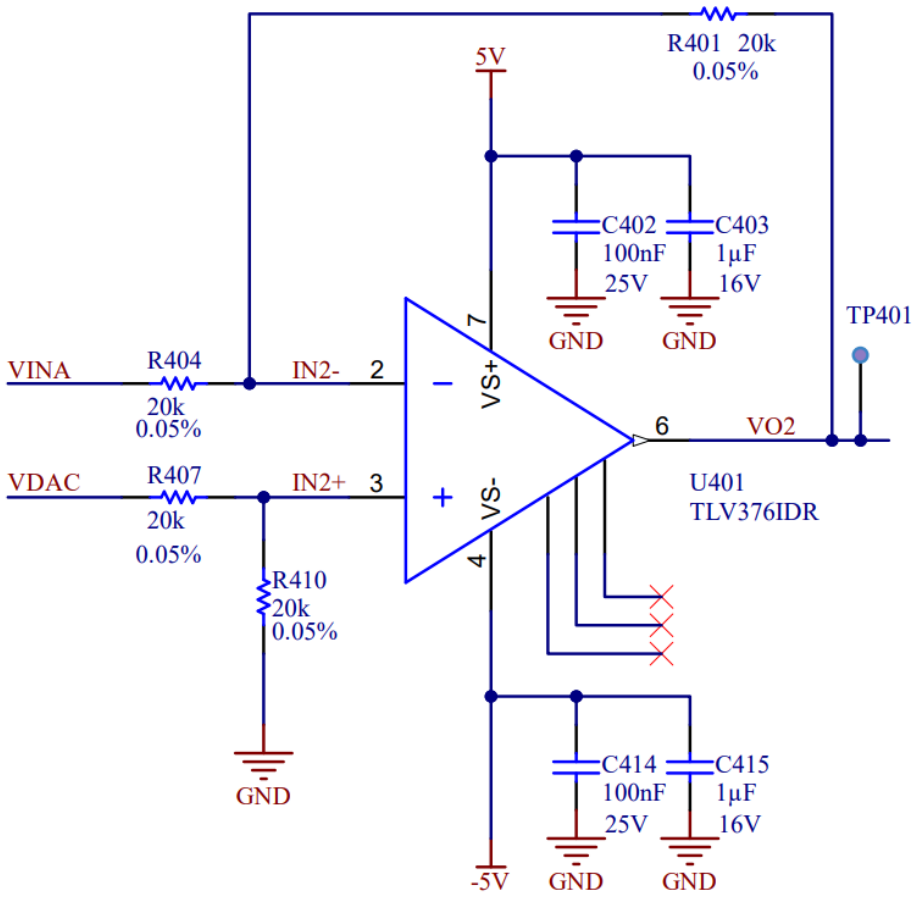


Figura F.1: Circuito propuesto para solucionar el problema de control de continua en el restador.

Referencias

[1] E. Greenwald, M. R. Masters, and N. V. Thakor, "Bidirectional neural interfaces-applications and vlsi circuit implementations," *National Library of Medicine (NIH)*, vol. Med Bio Eng Comput 54, no. 1, pp. 1–17, (2016) https://doi.org/10.1007/s11517-015-1429-x.

[2] S. Culaclii, "Design of A System for Cancelling Stimulus Artifact in Multi-Channel Neural Interfaces, PhD Thesis, UCLA," 2019. https://escholarship.org/uc/item/567316q8.

[3] Universidad de Navarra, "Diccionario médico - neuroestimulación," 2024, https://www.cun.es/diccionario-medico/terminos/neuroestimulacion: :text=f.,un

[4] Cleveland Clinic, "Deep Brain Stimulation," 2024, https://my.clevelandclinic.org/health/treatments/21088-deep-brain-stimulation.

[5] Wikipedia, "Axón," 2024, https://es.wikipedia.org/wiki/Ax%C3%B3n.

[6] H. E. Cingolani and A. B. Houssay, *Fisiología Humana, 7ma Edición*. Editorial El Ateneo, 2000.

[7] S. Culaclii, B. Kim, Y. K. Lo, L. Li, and W. Liu, "Online Artifact Cancelation in Same-Electrode Neural Stimulation and Recording Using a Combined Hardware and Software Architecture," *UCLA Electronic Theses and Dissertations*, vol. 12, no. 3, pp. 601–613, June 2018, , doi: 10.1109/TBCAS.2018.2816464.

[8] AnalogDevices, "LtSpice Software," https://www.analog.com/en/resources/design-tools-and-calculators/ltspice-simulator.html.

[9] Proteus, "Pcb Design and Simulation Made Easy, LabcenterM Electronics," July 2023,https://www.labcenter.com/.

[10] MathWorks, "Simulink para el diseño basado en modelos," 2023, https://la.mathworks.com/products/simulink.html.

[11] Wikipedia, "Flash adc," ,https://en.wikipedia.org/wiki/Flash_ADC.

[12] Mouser, "Mouser main page," https://www.mouser.com/.

[13] DigiKey, "Digikey main page," https://www.digikey.com/.

[14] Atmel - Microchip , "ATSAM3X8E Datasheet," https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf.

[15] AnalogDevices, *AD8221 - Precision Instrumentation Amplifier*, https://www.analog.com/media/en/technical-documentation/data-sheets/AD8221.pdf.

[16] "Analog devices: Mixed-signal and digital signal processing ics," https://www.analog.com/en/index.html.

[17] Microchip , *MCP4138 - 7/8-Bit Single/Dual SPI Digital POT with Volatile Memory*, https://ww1.microchip.com/ProductDocuments/DataSheets/22060b.pdf.

[18] "Microchip technology: Empowering innovation," https://www.microchip.com/".

[19] TexasInstruments, *TLV354x - 200-MHz, Rail-to-Rail I/O, CMOS Operational Amplifiers for Cost-Sensitive Systems*, https://www.ti.com/lit/ds/symlink/tlv3542.pdf?ts=1709235672348.

[20] "Texas instruments, analog, embedded processing, semiconductor company," https://www.ti.com/.

[21] *TLVx376 Low Offset and Drift, Low-Noise, Precision Operational Amplifiers for Cost-Sensitive Systems*, https://www.ti.com/lit/ds/symlink/tlv376.pdf?ts=1709235674228.

[22] *OPAx197 36-V, Precision, Rail-to-Rail Input/Output, Low Offset Voltage, Operational Amplifiers*, https://www.ti.com/lit/ds/symlink/opa197.pdf?ts=1709187108972&ref_url=https

[23] *TL07xx Low-Noise FET-Input Operational Amplifiers*, https://www.ti.com/lit/ds/symlink/tl072h.pdf?ts=1711764907330&ref_url=https

[24] *AD5443 - 8-/10-/12-Bit High Bandwidth Multiplying DACs with Serial Interface*, https://www.analog.com/media/en/technical-documentation/data-sheets/AD5426_5432_5443.pdf.

[25] *OPAx863 Low-Power, 110-MHz, Rail-to-Rail Input/Output Voltage-Feedback Op Amps*, https://www.ti.com/lit/ds/symlink/opa2863.pdf?ts=1711758666224.

[26] *MAX1247, +2.7V, Low-Power, 4-Channel, Serial 12-Bit ADCs in QSOP-16*, https://www.analog.com/media/en/technical-documentation/data-sheets/MAX1246-MAX1247.pdf.

[27] *NCP51460, Micropower Precision Voltage Reference*, https://www.onsemi.com/download/data-sheet/pdf/ncp51460-d.pdf.

[28] "Intelligent power and sensing technologies — onsemi," 2024, https://www.onsemi.com/.

[29] *LT3462, Inverting 1.2MHz/2.7MHz DC/DC Converters with Integrated Schottky* , https://www.analog.com/media/en/technical-documentation/data-sheets/lt3462.pdf.

[30] *TPS723 200-mA, Low-Noise, High-PSRR, Negative Output Low-Dropout Linear Regulators*, https://www.ti.com/lit/ds/symlink/tps723.pdf?ts=1711821762405&ref_url=https

[31] *Power Topologies Handbook*, https://www.ti.com/seclit/ug/slyu036/slyu036.pdf.

[32] "Altium designer," 2024, https://www.altium.com/altium-designer.

[33] "World's fastest pcb manufacturing - allpcb.com," 2024, https://www.allpcb.com/.

[34] Phil's Lab, "Pid controller implementation in software," 2021, https://www.youtube.com/watch?v=zOByx3Izf5U.

[35] "Stm32l552ze - Ultra-low-power with FPU Arm Cortex-M33 MCU 80 MHz with 512 Kbytes of Flash memory, USB, STM32L552ZET6TR, STm32L552ZET6, STM32l552ZET6U, STM32l552ZET6Y, STM32l552ZET6V," (2024), PDF, https://www.st.com/en/microcontrollers-microprocessors/stm32l552ze.html#overview.

[36] "Description of stm32l5 hal and low-layer drivers, STMicroelectronics," (2024), PDF, https://www.st.com/resource/en/user_manual/um2659-description-of-stm32l5-hal-and-lowlayer-drivers-stmicroelectronics.pdf.

[37] "Practical UML Statecharts in C/C++. Quantum Leaps, LLC," 2024, https://www.youtube.com/watch?v=FCymm6PBtOs&t=1045s&ab_channel=QuantumLeaps%2CLLC.

[38] "Digilent – Start Smart, Build Brilliant.," 2024, https://digilent.com/.

[39] "Analog discovery 2," 2023, https://digilent.com/reference/test-and-measurement/analog-discovery-2/start.

[40] "multiprocessing — process-based parallelism," 2024, https://docs.python.org/3/library/multiprocessing.html.

[41] "threading — thread-based parallelism," 2024, https://docs.python.org/3/library/threading.html.

[42] "Waveforms sdk reference manual," 2024, https://digilent.com/reference/software/waveforms/waveforms-sdk/reference-manual.

[43] "Waveforms-sdk-getting-started-py," 2022, https://github.com/Digilent/WaveForms-SDK-Getting-Started-PY.

[44] "Numpy documentation," 2022, https://numpy.org/doc/stable/index.html.

[45] "Scipy documentation," 2024, https://docs.scipy.org/doc/scipy/.

[46] "Matplotlib 3.8.3 documentation," 2024, https://matplotlib.org/stable/index.html.

[47] "Documentation introduction," 2024, https://customtkinter.tomschimansky.com/documentation/.

[48] "The pydwf package, Python," 2023, (Ver documentación de Python), https://pydwf.readthedocs.io/en/latest/pydwf_api/pydwf_overview.html.

[49] *PSpice for TI*, https://www.ti.com/tool/PSPICE-FOR-TI.

[50] *YAGEO Group*, https://www.google.com/search?q=YAGEO&rlz=1C1ALOY_esUY992UY992&oq=YAGEO&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIMCAEQIxgnGIAEGIoFMgwIAhAuGCcYgAQYigUyEggDEC4YQxjHARjRAxiABBiKBTIMCAQQABhDGIAEGIoFMgwIBRAAGEMYgAQYigUyDQgGEC4YxwEY0QMYgAQyBwgHEC4YgAQyBwgIEC4YgAQyBwgJEAAYgATSAQc3MzdqMGo3qAIAsAIA&sourceid=chrome&ie=UTF-8.

Índice de tablas

Índice de figuras

Esta es la última página.
Compilado el viernes 24 mayo, 2024.
http://iie.fing.edu.uy/