Universidad de la República
Facultad de Ingeniería

Cancelación de Artefactos en Neuroestimuladores Integrados

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

César Azambuya, Mateo Guerrero, Gonzalo Hernández

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

Tutores

Germán Fierro ..... Universidad de la República
Francisco Veirano ..... Universidad de la República

Tribunal

Federico Favaro ..... Universidad de la República
Nicolás Gammarano ..... Universidad de la República
Ignacio Ramírez ..... Universidad de la República

Montevideo
jueves 12 junio, 2025

Cancelación de Artefactos en Neuroestimuladores Integrados , César Azambuya, Mateo Guerrero, Gonzalo Hernández.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.2).
Contiene un total de 178 páginas.
Compilada el viernes 18 julio, 2025.
http://iie.fing.edu.uy/

Inventar es convertir la duda en hipótesis, el error en conocimiento, y el intento en posibilidad.

-Anónimo.

Esta página ha sido intencionalmente dejada en blanco.

Agradecimientos

Queremos expresar nuestro agradecimiento a todas las personas que, de una forma u otra, nos acompañaron durante este extenso proceso que implicó el desarrollo de este proyecto. Ya sea aportando ayuda técnica, soporte emocional o simplemente estando cerca, su apoyo fue fundamental para que hoy podamos presentar estos resultados.

En particular, agradecemos a nuestros tutores, cuyo acompañamiento constante y disposición para ayudarnos ante cada obstáculo fueron claves a lo largo de todo el proyecto. A nuestras familias, por brindar la contención necesaria para atravesar este desafío con determinación. A nuestras parejas, por su compañía constante y comprensión incondicional.

Agradecemos especialmente al equipo de Allegro, en particular a Conrado Rossi y Pablo Aguirre, por su asesoramiento técnico en momentos clave. También a la empresa Controles, por abrirnos sus puertas, compartir sus recursos e instalaciones y acompañarnos en el proceso de ensamblado de la PCB. Sin su colaboración, sería difícil imaginar haber alcanzado el nivel de calidad logrado en el hardware.

Un agradecimiento muy especial a Rafael Canetti, por su enorme ayuda en etapas críticas del análisis del comportamiento del sistema. Su colaboración fue decisiva para resolver problemas complejos de estabilidad.

Por último, queremos agradecer a Santiago Martínez, quien nos brindó asesoría tanto en aspectos técnicos del proyecto como en cuestiones específicas relacionadas con Dispositivos Médicos Implantables Activos.

Esta página ha sido intencionalmente dejada en blanco.

Resumen

Con la acelerada evolución de los avances tecnológicos, el desarrollo de Dispositivos Médicos Implantables Activos no ha sido la excepción. Los avances médicos relacionados con estos dispositivos implican, entre otras cosas, un aumento en la calidad de vida de los pacientes. En particular, esto es lo que buscan los denominados neuroestimuladores en lazo cerrado. Estos son una evolución de los sistemas tradicionales de estimulación neuronal, diseñados para ajustar automáticamente la terapia en función de la respuesta fisiológica del paciente. Para lograrlo presentan un objetivo común que radica en lograr capturar la respuesta neuronal del cuerpo humano, conocida como *Evoked Compound Action Potential* (ECAP). El desafío de los neuroestimuladores de lazo cerrado es que la señal ECAP se encuentra superpuesta tanto en el dominio temporal como en el frecuencial con la señal de respuesta al estímulo, conocida como *Stimulus Artifact* (SA). A su vez, la señal objetivo (ECAP) presenta una amplitud de varios órdenes de magnitud inferior al de la SA, lo que dificulta aún más su detección y análisis.

El sistema *Cancelación de Artefactos en Neuroestimuladores Integrados* (CANICs) implementa una solución a dicho problema a partir de la *Cancelación de Artefactos*. Esto implica desarrollar una serie de algoritmos iterativos con el fin de eliminar la SA.

Para ello se desarrolló un sistema analógico-digital encargado de ejecutar estos algoritmos y obtener la ECAP a partir de los mismos. El sistema digital fue diseñado a modo de obtener un módulo específico para circuitos integrados (ICs). El sistema analógico se diseñó utilizando componentes discretos, de forma tal que el diseño sea fácilmente integrable. La implementación analógica en sí misma, específica para ICs, no está contemplada en este proyecto.

A modo de validar el desempeño del sistema diseñado, se desarrolló una PCB que integra todos los elementos necesarios, y el diseño digital se sintetizó en una FPGA, generando un entorno de pruebas en conjunto con la PCB.

Las pruebas efectuadas en este entorno desarrollado demostraron que el sistema diseñado es capaz de recuperar señales ECAP con una correlación superior a 0.833 respecto a una señal ECAP de referencia, para el rango de relaciones SA/ECAP comprendidas entre 53 dB y 70 dB. El valor máximo de correlación registrado fue de 0.928, correspondiente a una relación de 53 dB.

Esta página ha sido intencionalmente dejada en blanco.

Glosario

- **AC:** Alternating Current.
- **ADC:** Analog to Digital Converter.
- **AIMD:** Active Implantable Medical Device.
- **ASIC:** Application Specific Integrated Circuit.
- **CANE:** Cancelación de Artefactos en Neuroestimuladores.
- **CANICs:** Cancelación de Artefactos en Neuroestimuladores Integrados.
- **CMOS:** Complementary Metal-Oxide-Semiconductor.
- **CMRR:** Common-Mode Rejection Ratio.
- **CSV:** Comma-Separated Values.
- **DAC:** Digital to Analog Converter.
- **DC:** Direct Current.
- **DSP:** Digital Signal Processing.
- **ECAP:** Evoked Compound Action Potential.
- **FDA:** Food and Drug Administration.
- **FF:** Flip Flop.
- **FIR:** Finite Impulse Response.
- **FPGA:** Field-Programmable Gate Array.
- **FSM:** Finite State Machine.
- **FW:** Firmware.
- **HW:** Hardware.
- **I2C:** Inter-Integrated Circuit.
- **INA:** Instrumentation Amplifier.

- **LDO:** Low Dropout Regulator.
- **LSB:** Least Significant Bit.
- **MSB:** Most Significant Bit.
- **MSD:** Mean Subtractor Device.
- **MUX:** Multiplexer.
- **PC:** Personal Computer.
- **PCB:** Printed Circuit Board.
- **PLL:** Phase-Locked Loop.
- **PSRR:** Power Supply Rejection Ratio.
- **RMS:** Root Mean Square.
- **RTL:** Register Transfer Level.
- **SA:** Stimulus Artifact.
- **SAE:** Sum of Absolute Errors.
- **SCS:** Spinal Cord Stimulation.
- **SDRAM:** Synchronous Dynamic Random Access Memory.
- **SNR:** Signal-to-Noise Ratio.
- **SoC:** System on Chip.
- **SPI:** Serial Peripheral Interface.
- **SPICE:** Simulation Program with Integrated Circuit Emphasis.
- **SW:** Software.
- **TCL:** Tool Command Language.

Tabla de contenidos

Agradecimientos III

Resumen V

Glosario VII

1. Introducción 1
- 1.1. Motivación . . . 1
  - 1.1.1. Neuroestimuladores en lazo cerrado . . . 3
- 1.2. Problema a abordar . . . 5
- 1.3. Antecedentes . . . 7
- 1.4. Objetivos . . . 7
  - 1.4.1. Objetivos especificos . . . 8
- 1.5. Alcance . . . 8

2. Diseño implementado 11
- 2.1. Introducción . . . 11
- 2.2. Especificaciones de las señales a procesar . . . 12
  - 2.2.1. Stimulus Artifact (SA) . . . 12
  - 2.2.2. Evoked Compound Action Potential (ECAP) . . . 12
  - 2.2.3. Componentes en modo común y componentes en DC . . . 14
- 2.3. Arquitectura . . . 15
  - 2.3.1. Principio de funcionamiento del sistema CANICs . . . 16
  - 2.3.2. Validación del sistema . . . 20

3. Etapa 1: Primera cancelación de la SA 23
- 3.1. Introducción . . . 23
- 3.2. Principio de funcionamiento . . . 24
- 3.3. Amplificador de Entrada . . . 25
- 3.4. Detector de Señal Entrante . . . 26
  - 3.4.1. Circuito Analógico . . . 27
  - 3.4.2. Circuito Digital . . . 29
- 3.5. Limitación en la frecuencia del sistema digital . . . 30
- 3.6. Amplificador diferencial . . . 31
- 3.7. Filtro Anti Aliasing . . . 33
- 3.8. ADC . . . 33

3.8.1. Especificaciones del ADC 33
3.8.2. Elección de la frecuencia de muestreo 34
3.8.3. Modo de operación 35
3.9. Controlador del ADC 36
3.9.1. Necesidad de Retardo para Sincronización 38
3.10. DAC 40
3.10.1. Principio de Funcionamiento 40
3.10.2. Modo de Operación 41
3.11. Controlador del DAC 42
3.12. Filtro de suavizado del DAC 43
3.13. Módulo de procesamiento de datos digitales *DSP* 45
3.13.1. Módulo *mem_handler* 46
3.13.2. Módulo *converters_handler* 48

4. Etapa 2: Cancelación de Residuo y Extracción de ECAP 49
4.1. Introducción 49
4.1.1. Principio de funcionamiento 50
4.2. Amplificador de entrada 54
4.3. Filtro Anti Aliasing 55
4.3.1. Filtro Butterworth 56
4.3.2. Arquitectura RC 57
4.4. Módulo de procesamiento de datos digitales MSD 58
4.4.1. Módulo template 60
4.4.2. Módulo template_handler 62
4.4.3. stop_average 63
4.4.4. Módulo subtractor 63
4.4.5. Módulo digital_filter 64

5. Control Principal del Sistema 67
5.1. Introducción 67
5.2. Control Flujo de Datos del ADC 69
5.3. Bloque de enventanado *window_handler* 69
5.3.1. Cálculo del Intervalo de la Ventana Temporal 70
5.3.2. Implementación 71
5.4. Máquina de Estados de Control de Etapas FSM 72
5.5. Control ADC 73
5.6. Control de error de la etapa 1 *stop_s1* 73
5.6.1. Interfaz 74
5.6.2. Funcionamiento 74
5.7. Control de error de la etapa 2 *overflow_monitor* 75
5.8. Acumulador de datos *data_counter* 76
5.9. Control de la etapa 2 *s2_control* 78

6. Simulaciones **81**
6.1. Introducción . . . 81
6.2. Testbench principal . . . 82
6.2.1. Esquemático . . . 83
6.2.2. Resultados . . . 86
6.3. Modelos de bloques analógicos . . . 88
6.3.1. ADC . . . 89
6.3.2. DAC . . . 93
6.3.3. Amplificadores operacionales . . . 93

7. Diseño de PCB CANICs **95**
7.1. Introducción . . . 95
7.2. Diagrama de Bloques del HW . . . 95
7.3. Niveles de tensión utilizados . . . 97
7.4. Estimación del consumo del sistema analógico . . . 98
7.5. FPGA utilizada . . . 99
7.6. Power Management . . . 99
7.7. Circuitos principales de la PCB . . . 105
7.7.1. Atenuador . . . 105
7.7.2. Amplificador de Instrumentación (INA) . . . 106
7.7.3. Schmitt Trigger y Rectificador . . . 106
7.7.4. Amplificador Diferencial . . . 107
7.7.5. Amplificador de entrada de la *Etapa 2* . . . 108
7.7.6. Conversor Analógico-Digital . . . 108
7.7.7. Level Shifter . . . 111
7.7.8. Conversor Digital-Analógico . . . 112
7.7.9. Filtros realizados . . . 114
7.8. Diseño general de la PCB . . . 114

8. Validación del Sistema **119**
8.1. Introducción . . . 119
8.2. Entorno de medidas utilizado . . . 119
8.3. Utilización de la FPGA . . . 121
8.4. Ganancia del circuito . . . 122
8.5. Definición de las métricas . . . 123
8.5.1. Correlación entre ECAP extraída y ECAP impuesta a la entrada . . . 123
8.5.2. Amplitud pico a pico de señal ECAP a la entrada . . . 124
8.6. Resultados y desempeño obtenido . . . 125
8.6.1. Correlación medida entre la señal ECAP extraída y la señal ECAP impuesta a la entrada . . . 127
8.6.2. Amplitud pico a pico extraída de la entrada . . . 130
8.6.3. Rechazo al modo común del circuito . . . 132

9. Conclusiones **135**
9.1. Trabajo futuro y posibles mejoras . . . 136

Tabla de contenidos

A. Sistema analógico **139**
A.1. Filtro Butterworth antialias 139
A.1.1. Atenuación en banda de rechazo (Stopband) 139
A.1.2. Respuesta en frecuencia 140
A.2. Circuito Atenuador 140

B. Sistema digital **143**
B.1. Flip-Flops utilizados para las memorias 143
B.2. Control de los Buffers 143
B.3. Funcionamiento del bloque *mem_handler* 144

Referencias **147**

Glosario **149**

Índice de tablas **150**

Índice de figuras **152**

Capítulo 1

Introducción

1.1. Motivación

En este capítulo se presentará una introducción al problema de estudio, junto con una revisión de los antecedentes y los objetivos específicos del proyecto, con su respectivo alcance.

En el ámbito de la medicina moderna, los Dispositivos Médicos Implantables Activos (AIMDs, por sus siglas en inglés) juegan un papel fundamental en el tratamiento de diversas enfermedades (por ejemplo, insuficiencia cardíaca, epilepsia, dolor crónico, entre otros) y en la mejora de la calidad de vida de los pacientes. Un AIMD se define como un producto sanitario activo destinado a ser introducido en el cuerpo humano, total o parcialmente, mediante intervención quirúrgica o médica, o a través de un orificio natural, con la finalidad de permanecer en el organismo después de la intervención [1].

Estos dispositivos no solo ofrecen soluciones terapéuticas, sino que también permiten el monitoreo continuo de la condición del paciente. Gracias a su capacidad para interactuar directamente con los tejidos biológicos, los AIMDs son cruciales en el tratamiento de afecciones crónicas y complejas. Algunos ejemplos de AIMDs son los marcapasos cardíacos, neuroestimuladores para el control de dolores crónicos, trastornos neurológicos y bombas de insulina.

Los neuroestimuladores son una clase de AIMDs diseñados para enviar impulsos eléctricos controlados a estructuras del sistema nervioso pertenecientes principalmente al Sistema Nervioso Central, formado por el cerebro y la médula espinal. Su propósito es modular la actividad neuronal y así alterar las señales nerviosas para tratar distintas afecciones, como el dolor crónico, la epilepsia, el Parkinson o trastornos neurológicos [2].

Los neuroestimuladores encargados de estimular la médula espinal son utilizados para tratar el dolor crónico, los cuales suelen estar compuestos por un generador de pulsos de corriente implantado y electrodos conectados al tejido objetivo. Pueden programarse para adaptar la terapia a las necesidades específicas de cada paciente, ofreciendo una alternativa terapéutica cuando los tratamientos farmacológicos resultan ineficaces o generan efectos adversos significativos [3]. En la figura

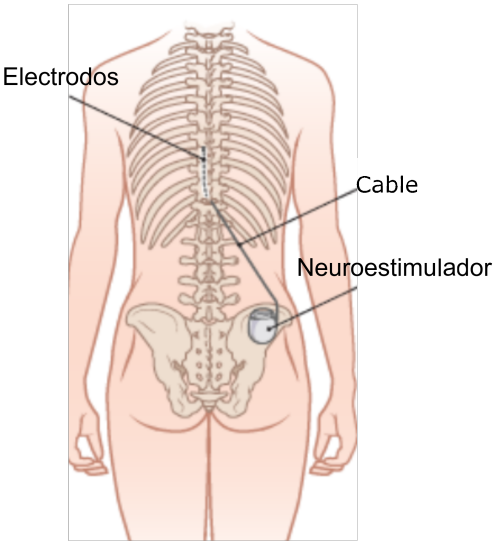

Figura 1.1: Sistema de neuroestimulación medular implantado. Se observa el neuroestimulador implantable, ubicado en la región glútea, los cables de derivación (leads) que conducen las señales eléctricas, y los electrodos posicionados adyacentes a la médula espinal, donde se aplica la estimulación. Imagen adaptada de [4].

1.1 se observa un neuroestimulador medular implantado, incluyendo el neuroestimulador propiamente dicho, los leads y los electrodos posicionados cerca de la médula espinal.

La mayoría de los neuroestimuladores comerciales emplean microprocesadores de propósito general y componentes *off-the-shelf* para gestionar la estimulación. Esto se debe a que los neuroestimuladores son tecnologías relativamente recientes en comparación con otros dispositivos médicos, como los marcapasos. Por ello, a las empresas que lideran la innovación en este campo, les resulta más conveniente optar por arquitecturas basadas en microprocesadores estándar y componentes *off-the-shelf*, ya que ofrecen mayor flexibilidad y confiabilidad durante las etapas de desarrollo y validación clínica. Sin embargo, esta arquitectura limita la miniaturización e incrementa el consumo energético [5].

En contraste, los System on Chip (SoC) ofrecen una plataforma altamente integrada que permite reducir tamaño, mejorar eficiencia energética y optimizar el procesamiento local de señales. Se han propuesto trabajos de investigación prometedores, como el presentado por Ker y Cheng [6], aunque estos aún no han sido trasladados al ámbito comercial debido a desafíos como la validación clínica, la miniaturización y la integración de múltiples funciones en un solo chip.

Este proyecto se motiva en la necesidad de abordar esta brecha mediante el desarrollo parcial del diseño de un neuroestimulador implementable sobre un SoC.

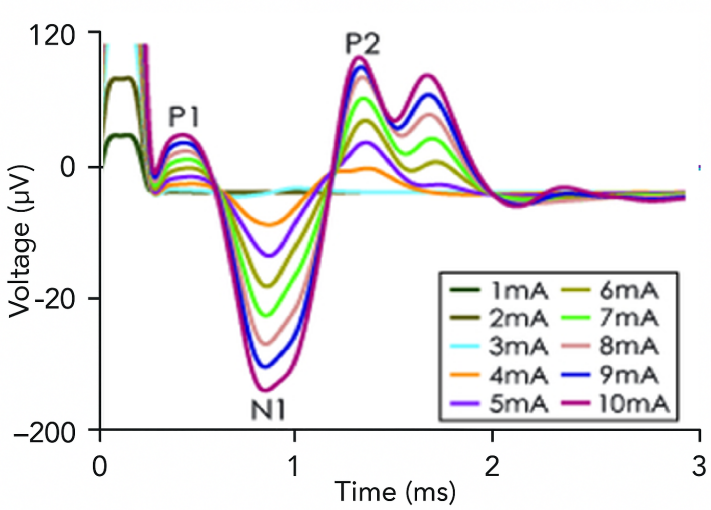

Figura 1.2: Morfología típica de la ECAP registrada en humanos durante estimulación de la médula espinal. Se observa una forma trifásica compuesta por una onda positiva inicial (P1), un pico negativo (N1) y una segunda onda positiva (P2). La amplitud del ECAP varía en función de la intensidad de corriente de estimulación, lo que permite su uso como señal de retroalimentación en sistemas de estimulación en lazo cerrado. Imagen tomada de [7].

1.1.1. Neuroestimuladores en lazo cerrado

En el contexto de los neuroestimuladores en lazo cerrado, se encuentran los neuroestimuladores para la médula espinal (SCS, por sus siglas en inglés). Estos representan una evolución significativa en el tratamiento del dolor crónico, al permitir una adaptación dinámica de la terapia en función de las respuestas fisiológicas del paciente. A diferencia de los sistemas de lazo abierto, que aplican una estimulación constante sin considerar la variabilidad individual, los dispositivos de lazo cerrado incorporan sensores, conformados por al menos dos electrodos, que monitorean señales biológicas en tiempo real, como la *Evoked Compound Action Potential* (ECAP), para luego estimular.

La ECAP es una señal de tensión generada por la activación de fibras nerviosas en respuesta a un estímulo eléctrico, actuando como una medida objetiva de la activación neuronal. Presenta una morfología trifásica: comienza con una onda positiva inicial (P1), seguida por un pico negativo marcado (N1), y culmina con una segunda onda positiva (P2), como se muestra en la figura 1.2, donde se observa la variación de la amplitud de la ECAP en función de la corriente de estimulación [7]. Esta forma se debe a la propagación y repolarización de los potenciales de acción [1] en las fibras nerviosas, y su amplitud está relacionada con la cantidad de fibras activadas.

En concreto, la amplitud de la ECAP se utiliza como señal de realimentación en tiempo real para ajustar dinámicamente la corriente de estimulación, garantizando

[1]Un potencial de acción es un impulso eléctrico breve que las neuronas usan para comunicarse entre ellas y enviar señales a otras células.

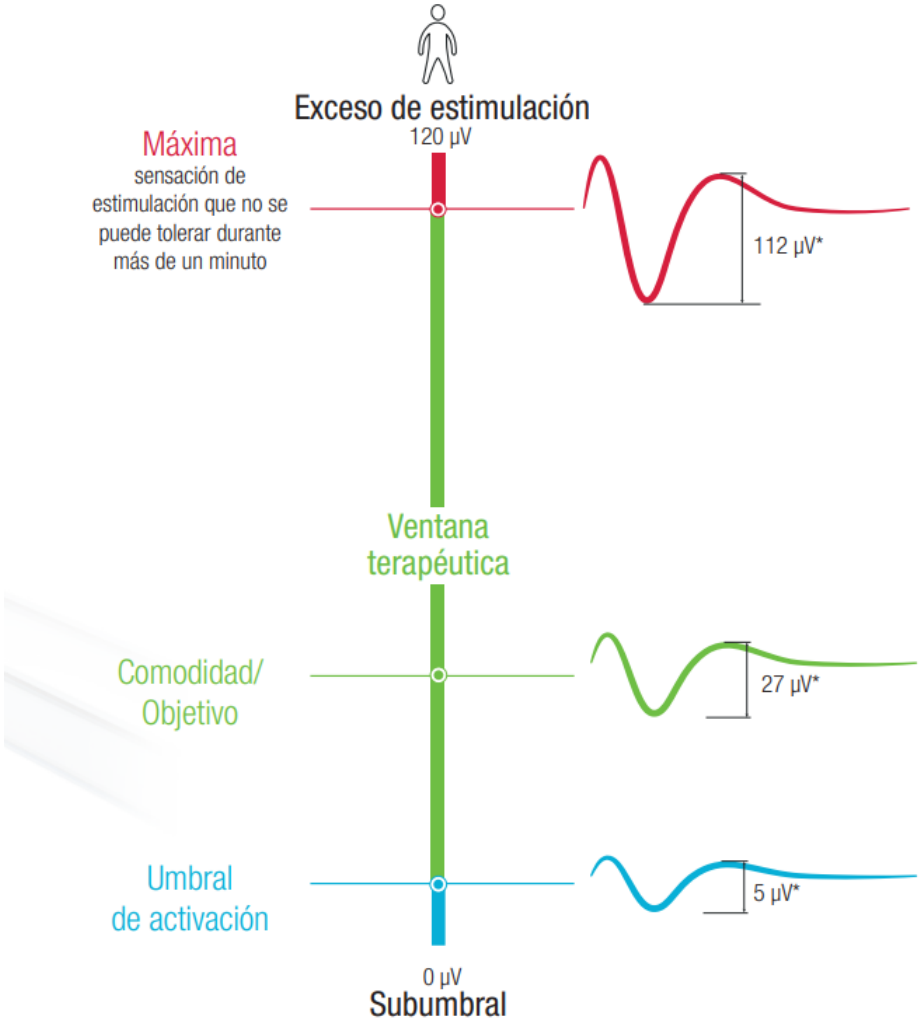

Figura 1.3: Diferentes amplitudes de la ECAP definen una ventana terapéutica, asegurando el alivio del dolor y el confort del paciente. Imagen tomada de [8]

una terapia eficaz y segura frente a variaciones fisiológicas o cambios en la posición del paciente. Dado que la amplitud de la ECAP está relacionada con la percepción del alivio del dolor y el confort del paciente, se define una ventana terapéutica que permite optimizar la corriente de estimulación. Esta ventana corresponde a un rango específico de amplitudes de la ECAP dentro del cual el paciente experimenta un alivio efectivo del dolor. Tal como se muestra en la figura 1.3, fuera de este rango (ya sea por una amplitud excesiva o insuficiente) el paciente no percibe alivio, lo que puede traducirse en una estimulación ineficaz o incómoda [8]. El valor de la ventana terapéutica varía de paciente a paciente, siendo configurada para cada caso específico.

En la figura 1.4 se muestra el comportamiento de la ECAP (ondas N1 y P2) en un sistema de lazo abierto (A) y lazo cerrado (B). Se ha superpuesto el artefacto de estimulación (SA) sobre la señal de la ECAP con fines ilustrativos, para

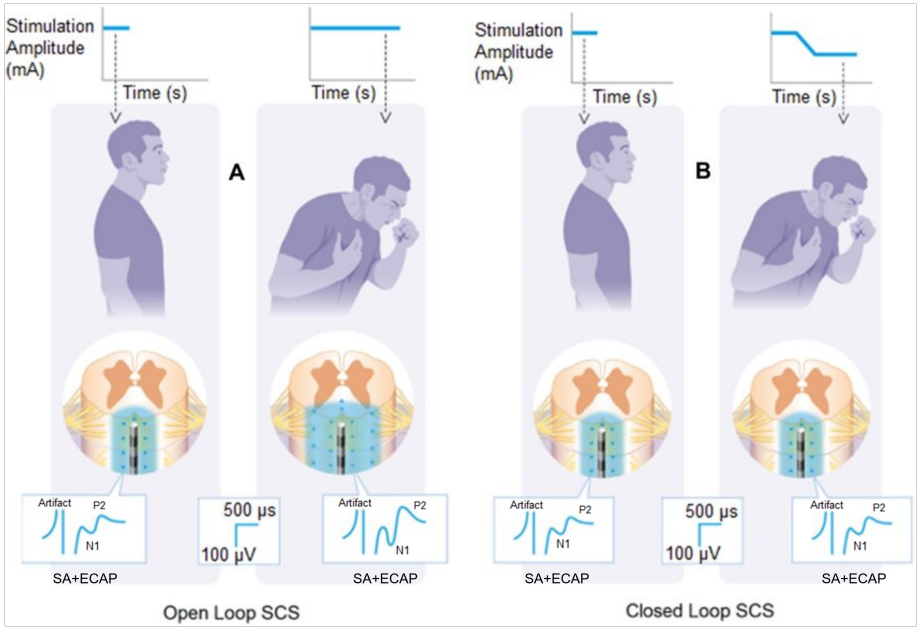

Figura 1.4: ECAP resultante (ondas N1 y P2 con P1 omitida por superposición con el artefacto, solo con fines ilustrativos) junto con el artefacto de estimulación, para el caso de lazo abierto y lazo cerrado. Cuando el paciente tose, los electrodos de estimulación medular se acercan a la médula espinal. Esto es detectado por los sensores espinales como un aumento en la amplitud de la ECAP. En la configuración de lazo abierto (A), los parámetros de estimulación son fijos; generando que la ECAP aumente en amplitud (aumento en N1 y P2) a medida que disminuye la distancia entre los electrodos y la médula. En la configuración de lazo cerrado (B), el sistema detecta el aumento de la ECAP y reduce dinámicamente la amplitud de estimulación para mantener constante la amplitud de la ECAP (amplitud de N1 y P2 constantes). Imagen tomada de [9].

mostrar cómo ambas señales coexisten. Se observa que cuando el paciente tose, se produce un cambio en la amplitud de la ECAP (aumento de las ondas P2 y N1). Sin embargo, debido a la falta de realimentación, no se ajusta la amplitud de estimulación. Esta variación en la amplitud de la ECAP, sin un ajuste en la corriente para compensarla, provoca que la terapia pierda efectividad. El comportamiento del sistema en lazo cerrado (B) permite que, ante variaciones en la amplitud de la ECAP, se ajuste dinámicamente la corriente de estimulación [9]. Por ende, este ajuste es capaz de compensar dichos cambios y mantener la amplitud de la ECAP dentro de un margen deseado (amplitud de las ondas P2 y N1 constantes), incluso si el paciente se ha movido, logrando así una mayor efectividad en la terapia a lo largo del tiempo.

1.2. Problema a abordar

En los neuroestimuladores en lazo cerrado para el tratamiento del dolor crónico, la estimulación en corriente y la medida de tensión resultante se realizan en la

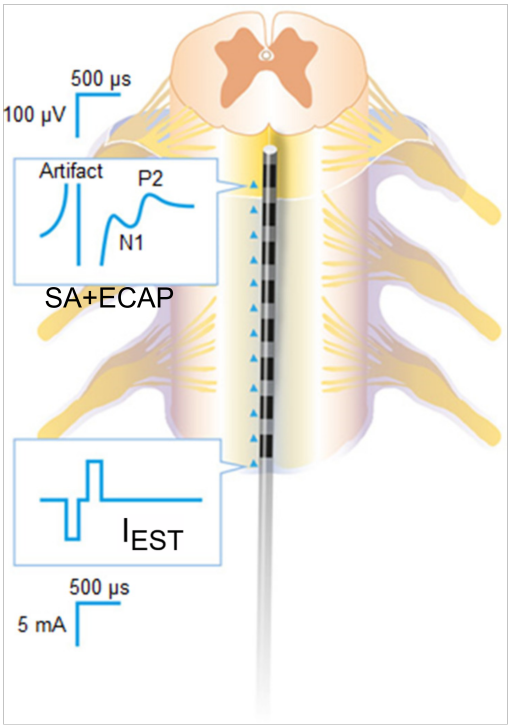

Figura 1.5: Segmento de médula espinal con estimulación en corriente (I_{EST}) aplicada a las columnas dorsales en un extremo del electrodo. En el extremo opuesto, se mide la ECAP (ondas N1 y P2 con P1 omitida por superposición con el artefacto, solo con fines ilustrativos) resultante, junto con el artefacto de estimulación asociado (SA). Imagen tomada de [9].

misma zona. Como resultado, los electrodos no solo registran la respuesta neuronal ECAP, sino también un artefacto eléctrico conocido como *Stimulus Artifact* (SA). Este artefacto se genera por la conversión del pulso de corriente de la estimulación en una señal de tensión en el tejido biológico. En la figura 1.5 se representa un segmento de médula espinal junto con los electrodos para la estimulación y la captura. Cuando se realiza la estimulación en corriente, se captura en tensión el artefacto superpuesto con la señal de interés ECAP [9].

Esto presenta la dificultad de que la SA y la ECAP no solo se superponen temporalmente, sino que también comparten componentes espectrales, lo que hace difícil su separación utilizando técnicas de filtrado convencionales. Además, la amplitud de la SA es varios órdenes de magnitud superior al de la ECAP, lo que provoca que la señal de interés quede completamente enmascarada. La combinación de esta superposición temporal y espectral junto con la diferencia en magnitudes genera una interferencia significativa, dificultando el uso de la ECAP como una señal de realimentación.

El problema a abordar consiste en el diseño de un sistema que sea capaz de obtener exclusivamente la señal ECAP a partir de la medición de la respuesta al estímulo. Para ello, se debe eliminar el artefacto SA presente en la señal capturada, que se encuentra superpuesto a la señal de interés, ECAP. A la remoción del

artefacto es a lo que se denominará *Cancelación de Artefacto*.

1.3. Antecedentes

En el marco de la tesis doctoral de Stanislav Culaclii, *Design of A System for Cancelling Stimulus Artifact in Multi-Channel Neural Interfaces*, propone un sistema que permite registrar señales neuronales inmediatamente después de un estímulo eléctrico, eliminando el artefacto de estímulo. Utiliza técnicas de Hardware (HW), Firmware (FW) y Software (SW) para cancelar estos artefactos en tiempo real. Se reporta una relación de cancelación [2] entre la SA y la ECAP de 100 dB [10].

En la Facultad de Ingeniería de la UdelaR, se tiene como antecedente el Proyecto de Fin de Carrera *Cancelación de Artefactos en Neuroestimuladores* (CANE). Basándose en el doctorado anterior, integran el desarrollo de HW, FW y SW, diseñado para implementar una prueba de concepto de un sistema de cancelación de artefactos en neuroestimuladores, permitiendo registrar señales neuronales sin interferencias generadas por el estímulo eléctrico. En su implementación, el sistema CANE logró una relación de cancelación de aproximadamente 56 dB entre la SA y la ECAP [11].

En el ámbito comercial, el sistema *Evoke* de Saluda Medical utiliza la ECAP, recuperada mediante la *Cancleación de Artefactos*, para ajustar la corriente de estimulación en tiempo real para el tratamiento del dolor crónico. Para evaluar la eficacia de la terapia se realizaron ensayos clínicos, donde se mostró una reducción del dolor de al menos el 50 % en el 77.6 % de los pacientes con dolor crónico, superando los resultados de la estimulación convencional en lazo abierto [12]. *Evoke* representa un hito clínico significativo ya que es el primer dispositivo de estimulación medular en lazo cerrado aprobado por la *Food and Drug Administration* (FDA) [13].

1.4. Objetivos

Desarrollar un sistema de *Cancelación de Artefactos en Neurestomuladores Integrados* (CANICs) como componente de un neuroestimulador en lazo cerrado (ver figura 1.6) para el tratamiento del dolor crónico, enfocado en la cancelación de artefactos para obtener la señal ECAP, basándose en [10] y [11].

El sistema debe estar compuesto por componentes de hardware y de procesamiento digital. El foco principal es el diseño del procesamiento digital a medida, diseñado para su futura implementación en circuitos integrados, para formar parte de un System on Chip (SoC). Para validar el sistema CANICs, se fabricará una PCB con los componentes de hardware requeridos y se sintetizará la arquitectura digital en una FPGA.

[2] Esto quiere decir que se logró recuperar una ECAP cuando la relación $20 \log_{10}(\text{SA}/\text{ECAP}) = 100\,\text{dB}$.

1.4.1. Objetivos especificos

1. **Diseño del núcleo digital:**
   - Diseñar el circuito digital a nivel RTL utilizando el lenguaje de descripción de hardware Verilog.
   - Realizar una verificación funcional de cada módulo digital para corroborar su funcionamiento.
   - Sintetizar el circuito en una FPGA.
2. **Modelado de bloques analógicos en Verilog-A:**
   - Modelar los amplificadores operacionales.
   - Desarrollar modelos de ADC y DAC.
3. **Simulaciones mixtas analógico-digital:**
   - Integrar los bloques analógicos modelados y digitales en un entorno común de simulación para circuitos integrados.
   - Realizar simulaciones a nivel sistema CANICs para verificar el funcionamiento de la cancelación de artefactos.
4. **Validación del sistema en entorno físico:**
   - Diseñar una PCB que integre los componentes de hardware requeridos, junto con la FPGA, permitiendo la validación del sistema completo en un entorno físico.
   - Medir el desempeño del sistema, en base a métricas comparables con los proyectos anteriores.

1.5. Alcance

El proyecto se enfoca en el desarrollo y validación del sistema CANICs como componente de un neuroestimulador implantable en lazo cerrado (ver figura 1.6). El trabajo abarca tanto el diseño digital orientado a su futura implementación en circuitos integrados como la integración con modelos analógicos para evaluación funcional.

El sistema fue validado en una primera instancia mediante simulaciones mixtas analógico-digitales, utilizando herramientas de diseño de circuitos integrados de última generación (Cadence), y en segunda instancia mediante una implementación física basada en FPGA y PCB. Se trabaja a nivel de prototipo experimental, orientado a una caracterización funcional de la cancelación de artefactos, verificable mediante pruebas con modelos de las señales biológicas involucradas. Si bien la validación se realiza en FPGA, el diseño sigue lineamientos y buenas prácticas de

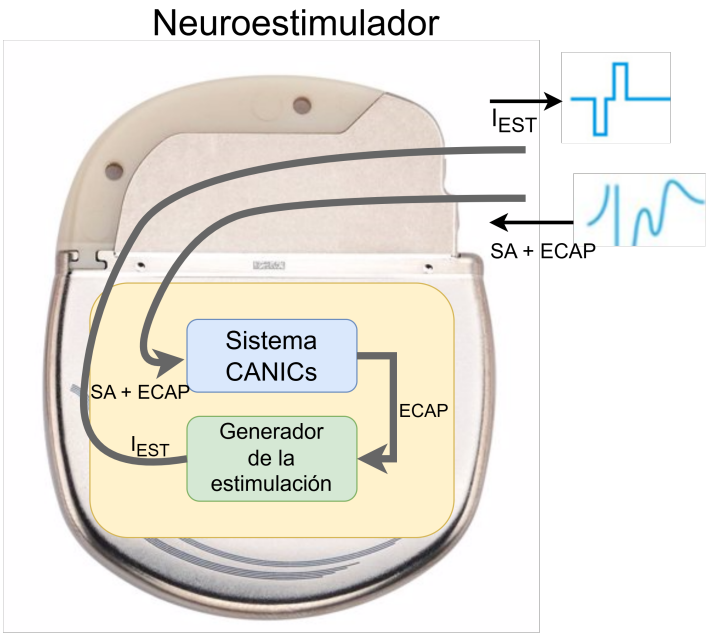

Figura 1.6: Ilustración del sistema CANICs en el contexto de sistema de neuroestimulación. Se observa que a la entrada del neuroestimulador ingresa la señal SA + ECAP, para luego ser procesadas por el sistema CANICs. Este extrae la señal ECAP, que luego sirve como realimentación para que el generador de estimulación ajuste la corriente de estimulación I_{EST} para la terapia.

diseño RTL [3] que permitan su posterior migración a una tecnología de fabricación CMOS para su implementación en un Circuito Integrado de Aplicación Específica (ASIC). Como se mencionó anteriormente, el objetivo del proyecto es el diseño digital del sistema CANICs. El diseño analógico del sistema específico para ASICs no fue contemplado en este proyecto. Se utilizaron componentes comerciales discretos, tanto en la PCB como en las simulaciones.
Queda por fuera del alcance del proyecto:

- El desarrollo completo del neuroestimulador implantable. En particular, no se aborda el diseño del *generador de la estimulación*, encargado de cerrar el lazo de control mostrado en la figura 1.6. Este módulo queda fuera del alcance del presente trabajo.
- El diseño analógico del sistema CANICs específico para ASICs.
- Pruebas del sistema *in vivo*.

[3] Las buenas prácticas incluyen el uso exclusivo de sintaxis sintetizable en Verilog, la definición exhaustiva de todas las estructuras condicionales en circuitos combinacionales (por ejemplo, incluir `else` o `default` en sentencias `if` o `case`), evitar caminos de lógicas combinacionales largos sin registrar sus entradas y salidas, mantener una separación entre lógica secuencial y combinacional, parametrizar el ancho de los buses, entre otros.

- La síntesis y el diseño Place and Route del sistema digital.

Capítulo 2

Diseño implementado

2.1. Introducción

En este capítulo se hará una introducción al diseño implementado y su estructura básica. Se explicarán las características particulares que poseen las señales que se tienen como entrada, SA y ECAP, y cómo se utilizan a lo largo del sistema para lograr la salida esperada, que son métricas de la ECAP de entrada. Además, se hará una introducción a la arquitectura del sistema CANICs, sus principales módulos, que son la *Etapa 1*, *Etapa 2*, el *Control Principal* y la interacción entre ellos.

La entrada del sistema se compone por las señales SA (figura 2.1) más la ECAP (figura 2.2). La SA es una señal periódica, debido a que la estimulación tiene esta característica. La ECAP en cambio, no es periódica, sino que surge a partir de una ventana de ocurrencia variable respecto a la SA. Este conjunto de señales ingresan superpuestas temporalmente al sistema de forma diferencial. El período se define como $T = \frac{1}{f_{\mathrm{SA}}}$, donde f_{SA} representa la frecuencia de estimulación especificada en la sección 2.2.1. La suma de las señales SA más ECAP se denominará en adelante como SENSADO, dado que corresponden a las señales sensadas por los electrodos al realizar la estimulación.

El objetivo del sistema es obtener la morfología de la ECAP y a partir de ella obtener métricas que la definen, como su amplitud pico a pico. Como se explicó en el capítulo anterior, ambas señales, SA y ECAP, además de estar superpuestas temporalmente, lo están en frecuencia; esto implica que el cambio al dominio de la frecuencia no es una solución efectiva. La solución utilizada se basa en explotar la diferencia de amplitudes entre ambas señales, la periodicidad de la estimulación, y que la ECAP tiene ventana de ocurrencia variable (Véase secciones 2.2.1 y 2.2.2). La arquitectura y sus diferentes etapas se expone en el presente capítulo.

2.2. Especificaciones de las señales a procesar

Las señales utilizadas partieron de los mismos modelos utilizados previamente por el grupo CANE [11], los cuales fueron provistos por un grupo de investigación internacional que colabora activamente con el Grupo de Microelectrónica de la Facultad de Ingeniería de la Universidad de la República. A continuación, se detalla la información recibida por parte de dicho grupo.

2.2.1. Stimulus Artifact (SA)

Para emular la señal SA, se deberá considerar una señal compuesta de dos fases, una negativa y otra positiva. 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, se deben cumplir las siguientes características:

- Amplitud: $V_{\text{pp}} = 70 \ldots 240\,\text{mV}$
- Ancho de banda: $\text{BW}_{\text{SA}} = 16\,\text{kHz}$
- Frecuencia de estimulación: $f_{\text{SA}} = 900\,\text{Hz}$
- Duración de la fase negativa: $300\,\mu\text{s}$
- Constante de tiempo para ambas fases: $\tau_{\text{SA}} = 250\,\mu\text{s}$
- Pendiente de la fase negativa: $\frac{1}{800}\ \text{V}_{\text{pp}}/\mu\text{s}$

En la figura 2.1 se observa el modelo obtenido para la señal SA. Este fue desarrollado con el objetivo de satisfacer las especificaciones propuestas anteriormente.

2.2.2. Evoked Compound Action Potential (ECAP)

En cuanto a la señal ECAP, se debe considerar una señal sinusoidal de 1,2 kHz atenuada temporalmente utilizando funciones exponenciales, con el objetivo de recrear la morfología del ECAP. Las características específicas de esta señal son las siguientes:

- Amplitud: $V_{\text{pp}} = 2 \ldots 150\,\mu\text{V}$
- 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:
    - $\mu = 470\,\mu\text{s}$
    - $\sigma = 62\,\mu\text{s}$
- Ancho de banda: entre 300 Hz y 5 kHz

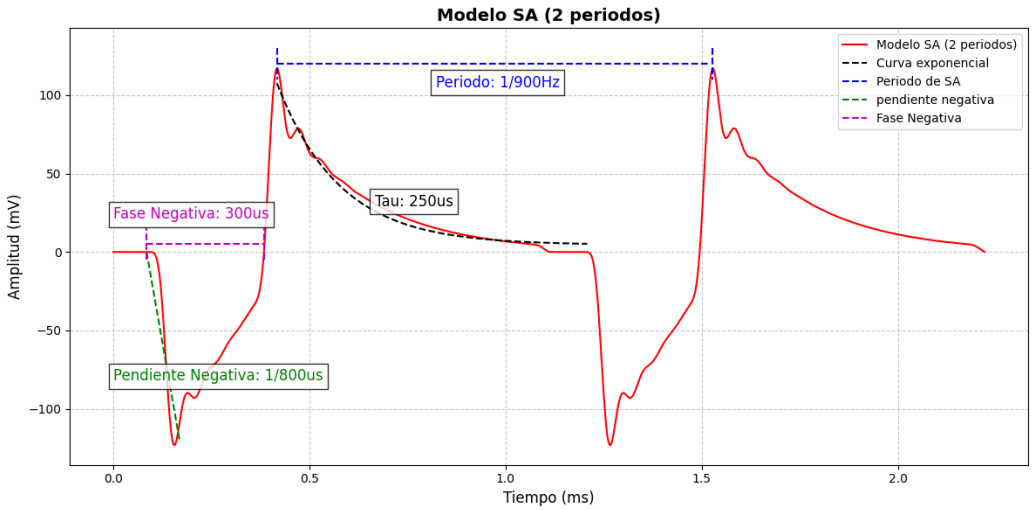

Figura 2.1: Modelo de la señal SA utilizado en el sistema.

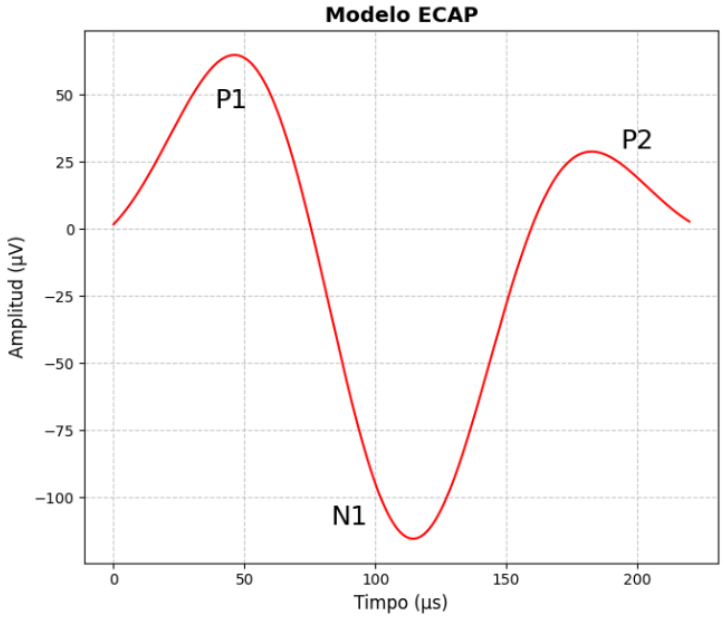

Figura 2.2: Modelo de la señal ECAP utilizado en el sistema

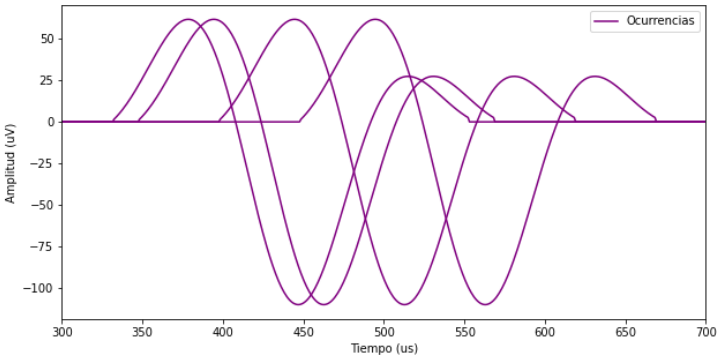

Figura 2.3: Cuatro posibles ocurrencias de la señal ECAP. El eje de tiempo se referencia respecto al comienzo de la fase negativa de la SA.

En la figura 2.2 se observa el modelo obtenido para la señal ECAP. El mismo fue generado siguiendo las especificaciones establecidas anteriormente.

La ventana de ocurrencia del frente de ondas se refiere a que el momento en que comienza la ECAP, respecto a la SA, no es fijo, sino que puede variar aleatoriamente dentro de un cierto intervalo de tiempo. A este intervalo se le llama ventana de ocurrencia. En otras palabras, aunque la ECAP siempre ocurra después del estímulo, no siempre aparece exactamente en el mismo punto en el tiempo respecto a la SA. En la figura 2.3 se muestran cuatro ejemplos posibles de cómo puede aparecer la señal ECAP en relación con la fase negativa del artefacto de estímulo (SA).

2.2.3. Componentes en modo común y componentes en DC

La adquisición de señales neurales se lleva a cabo midiendo la diferencia de potencial entre dos electrodos, uno positivo y otro negativo, denotados como E^+ y E^-, respectivamente. Tanto el artefacto (SA) como la señal neural (ECAP) corresponden a la componente diferencial de esta medición, definida como $V_D = E^+ - E^-$. También se introducen variaciones en el modo común $V_{CM} = \frac{E^+ + E^-}{2}$, que deben ser consideradas durante la etapa de adquisición, para asegurar la captura de la SA y la ECAP sin distorsiones.

El sistema debe ser capaz de rechazar señales de modo común con las siguientes características:

- $V_{\text{EMG}}^P = 2\,\text{mV}$
- $f_{\text{EMG}} = 400\,\text{Hz}$

Donde típicamente V_{EMG}^P corresponde a interferencias de origen electromiográfico (EMG), generadas por la actividad eléctrica de los músculos. En este contexto, V_{EMG}^P representa la amplitud pico de dicha señal, mientras que f_{EMG} indica una frecuencia característica [14].

Además, el sistema CANICs debe poder tolerar componentes de DC diferenciales en la entrada dentro de un rango de $\pm 100\,\text{mV}$, sin producir variaciones significativas en la salida del sistema.

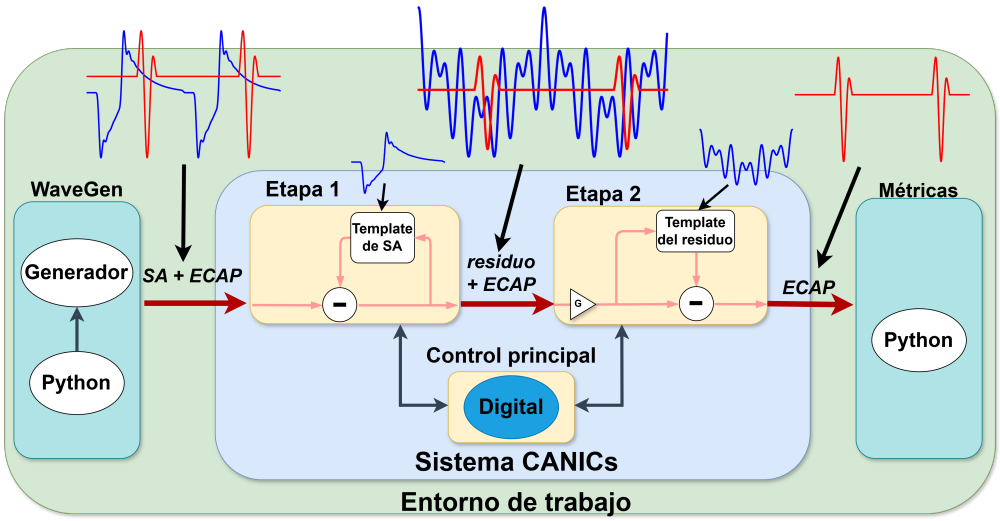

Figura 2.4: Arquitectura del sistema CANICs con dos etapas de procesamiento y un control principal del sistema. Además, se tiene un entorno de trabajo para la generación y análisis de señales.

2.3. Arquitectura

La arquitectura implementada se basa en la utilizada por el proyecto CANE [11], que también está desarrollada a partir de una tesis de doctorado [10], que presenta en detalle la implementación para abordar el problema planteado.

En la figura 2.4 se observa la arquitectura implementada. El sistema CANICs está compuesto por dos etapas, *Etapa 1* y *Etapa 2*, encargadas en conjunto de obtener la señal de interés ECAP, y por el *Control Principal* del sistema, responsable de controlar el funcionamiento de cada etapa y arbitrar el uso del HW.

Se observa que a la entrada de la *Etapa 1* ingresa la suma de las señales SA y ECAP, con una frecuencia f_{SA}. Cabe recordar que la ECAP es del orden de las centenas de μV, y la SA es del orden de las centenas de mV. En la figura se dibujaron de amplitudes similares con fines ilustrativos. La *Etapa 1* reduce lo máximo posible la presencia de la señal SA, de modo que a su salida se obtenga una señal cuyo orden de magnitud sea comparable al de la ECAP. Esta señal remanente se denominó residuo. De este modo, a la salida de la *Etapa 1* se obtiene la señal residuo sumada a la señal ECAP, de órdenes de magnitud similares, en el rango de los mV. La *Etapa 2* recibe la salida de la *Etapa 1*, residuo más ECAP, y la amplifica. Estas señales también están superpuestas temporalmente y en frecuencia. Esta etapa elimina el residuo y recupera la señal ECAP. Este proceso de eliminación del residuo y obtención de la ECAP lo hace enteramente de forma digital.

El subsistema digital *Control Principal* es el encargado de controlar y sincronizar el procesamiento en cada etapa; administra los datos entrantes a cada una de las etapas en cuanto ellas lo requieren. Como resultado, el sistema es capaz de entregar en su salida las muestras correspondientes a la señal ECAP.

Tanto la *Etapa 1* como la *Etapa 2* tienen el mismo mecanismo de funcionamiento: generar un template y restarlo a la señal que tienen como entrada. Ambas etapas reciben en su entrada la suma temporal de dos señales, una que se desea obtener y otra que se desea eliminar. Por lo tanto, el template se genera en base a la señal que se desea eliminar. Para la *Etapa 1* el template es sobre la señal SA (se desprecia la ECAP dada la diferencia de magnitud) y en la *Etapa 2* el template es sobre la señal residuo. El template requiere una determinada cantidad de iteraciones para que sea válido, lo que implica que tiene un nivel de error aceptable respecto a la señal que intenta modelar. Cuando se alcanza este nivel de error, determinado mediante simulación y medidas experimentales, se asume que el template alcanzó la convergencia. Con el template determinado, se procede a restarlo a la señal entrante, de forma de obtener la señal objetivo.

La diferencia entre ambas etapas es en cómo generan el template y efectúan la resta con su señal entrante. La *Etapa 1* posee un lazo de control, en donde el template se genera a partir de las iteraciones de restarle a la señal entrante el propio template, como muestra la figura 2.4, en donde inicialmente el template corresponde a una señal de 0 V. Por otro lado, la *Etapa 2* realiza un promedio entre períodos de su señal de entrada, y luego resta el promedio obtenido a la señal entrante, como esquematiza la figura 2.4. El principio de funcionamiento de cada etapa, junto con el *Control Principal*, se explican en la sección 2.3.1.

El entorno de trabajo, además del sistema CANICs, está compuesto por el bloque WaveGen y el bloque Métricas. El bloque WaveGen es el encargado de la generación de las señales de entrada del sistema CANICs. Para ello, toma los modelos de SA y ECAP explicados en la sección 2.2, los superpone temporalmente y genera un vector de 25 períodos en un archivo CSV. Este archivo es luego importado por un generador de señales capaz de producir salidas en tensión diferenciales.

El bloque Métricas tiene el objetivo de capturar las señales digitales provenientes del sistema CANICs. Realiza un procesamiento de las señales de salida del sistema, el cual consiste en hallar la función de correlación de la señal entrante con el modelo de ECAP visto en la sección 2.2. Las señales que verifican un nivel mínimo de correlación definido son luego caracterizadas con métricas objetivas, como la tensión pico a pico obtenida, de forma que indiquen la exactitud y dispersión de los resultados obtenidos.

2.3.1. Principio de funcionamiento del sistema CANICs

La *Etapa 1* se basa en que la componente SA de la señal SENSADO (SA más ECAP) es periódica, y que entre períodos las variaciones son mínimas [1]. Además, se utiliza que la SA es varios órdenes de magnitud mayor en amplitud que la ECAP.

La figura 2.5 se describe un esquema del lazo implementado para la *Etapa 1*. Comienza con la señal entrante SENSADO, que está compuesta por las señales $sa(i) + ecap(i)$, donde i es el i-ésimo período de la señal desde que comenzó el

[1] Las variaciones de la morfología de la señal SA provienen principalmente del cambio de la impedancia de los electrodos utilizados, y se estima que estas variaciones son lentas con respecto al período de la señal [15].

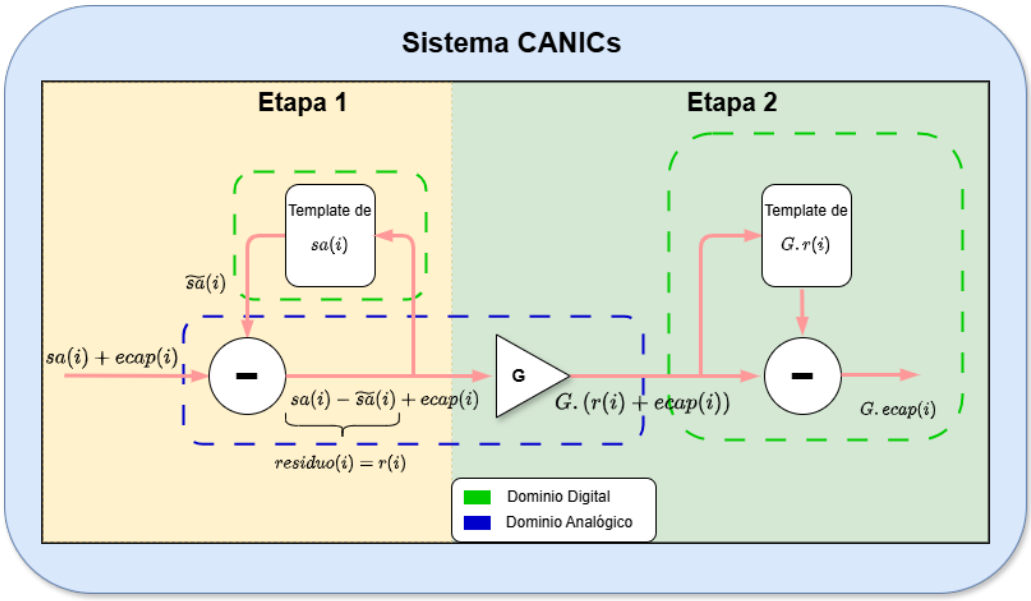

Figura 2.5: Diagrama de las etapas del sistema CANICs.

algoritmo. El funcionamiento se basa en generar un template $\widetilde{sa}(i)$ de la señal que se desea eliminar, en este caso $sa(i)$, a partir de la señal de entrada $sa(i) + ecap(i)$. En base a este template, se busca obtener la señal de interés, $ecap(i)$, restando a la señal de entrada el template, $sa(i) + ecap(i) - \widetilde{sa}(i)$ obteniendo $ecap(i)$ más un residuo. El residuo $r(i) = sa(i) - \widetilde{sa}(i)$ es el resultado de la diferencia entre la resta de la SA entrante y el template. El módulo *Template de sa(i)*, de dominio digital, se encarga de almacenar el template de la SA. Para generar el template, dado que al arranque la memoria está vacía, primero se almacena la señal entrante $sa(0)$, generando la primera iteración del template, $\widetilde{sa}(i)$ (dada la diferencia de órdenes de magnitud entre la SA y la ECAP, se asume que la ECAP es despreciable en el contexto del lazo iterativo, o de muestrearse parcialmente se considera que el aporte al template no representa un problema. Este supuesto se detalla en el capítulo 4).

La actualización del template consiste en sumar al template actual, reconstruido al dominio analógico $\widetilde{sa}(i)$, el residuo obtenido de la resta $sa(i) - \widetilde{sa}(i) = r(i)$. Básicamente la actualización del template sigue el algoritmo de la ecuación 2.1.

$$nuevo\ template = template\ actual + residuo \tag{2.1}$$

De esta forma, sumándole al template el residuo, este va adquiriendo la información faltante respecto a la $sa(i)$ original. Cuando el residuo se considera que es lo suficientemente pequeño, lo que implica que la señal $\widetilde{sa}(i)$ es semejante a la señal $sa(i)$, se dice que la *Etapa 1* logró la convergencia y se detiene este algoritmo, manteniendo el template, y por lo tanto la señal $\widetilde{sa}(i)$ con el último valor de iteración.

Una vez alcanzada la convergencia en el período $i = C$, la salida de la *Etapa 1* en un período genérico ($i > C$) puede expresarse como:

$$out_1(i) = (sa(i) + ecap(i)) - \widetilde{sa}(i) = residuo(i) + ecap(i),\ i > C \tag{2.2}$$

La *Etapa 2* se encarga de eliminar el residuo amplificado $G.r(i)$, extrayendo la ECAP mediante la generación y posterior sustracción de un template del residuo amplificado, tal como se observa en la figura 2.5. Este template se genera promediando múltiples períodos de la señal de entrada $G(r(i) + ecap(i))$. Esta etapa opera únicamente una vez que la *Etapa 1* ha alcanzado la convergencia ($i > C$) y mantiene dicha condición.

El correcto funcionamiento de esta etapa se basa en tres hipótesis: primero, que el residuo se pueda asumir periódico de período T (período de estimulación). Esto se debe a que el residuo vale $r(i) = sa(i) - \widetilde{sa}(i)$, donde $sa(i)$ de período T por hipótesis, y $\widetilde{sa}(i)$ está fijado por el template (las variaciones de la señal $sa(i)$ se asumen despreciables), por ende la diferencia será periódica. Segundo, que la ECAP tiene una ventana de ocurrencia variable. Tercero, que el residuo y la ECAP son de órdenes de magnitud similar.

Como se ilustra en la figura 2.5, la señal $r(i) + ecap(i)$ es amplificada con una ganancia G, optimizando el uso del ADC, llevándolo a la máxima excursión admisible. Una vez se tiene $G.(r(i) + ecap(i))$ se pasa al dominio digital, donde se aplica un algoritmo que promedia $ITER =$ dieciséis períodos de señal (su elección se verá en la sección 4.1.1 del capítulo 4).

Denominando $out_2(i)$ como la salida de la *Etapa 2* para el i-ésimo período de estimulación, se tiene que:

$$out_2(i) = G.out_1(i) - \frac{1}{ITER} \sum_{j=C+1}^{C+ITER} G.out_1(j),\ i > C \tag{2.3}$$

Sustituyendo $out_1(j)$, la ecuación previa se puede reescribir como

$$out_2(i) = G.out_1(i) - \frac{1}{ITER} \sum_{j=C+1}^{C+ITER} G.(r(j) + ecap(j)),\ i > C \tag{2.4}$$

Dado que la ECAP tiene ventana de ocurrencia variable, el valor de promediarla $ITER$ veces, tiende a reducir significativamente la amplitud de la misma. Este fenómeno se ilustra en la figura 2.6 donde en gris se observa dieciséis posibles ocurrencias de la ECAP, apreciándose su desviación temporal. Mientras que, en azul se ilustra el resultado del promedio de las señales. Se destaca la atenuación del promedio, respecto a la ECAP original (esto se detallará en el capítulo 4). Usando esta característica y que $r(i)$ es periódica ($r(i) = residuo,\ i > C$), se obtiene que:

$$\frac{1}{ITER} \sum_{j=C+1}^{C+ITER} G.(r(j) + ecap(j)) \approx \frac{1}{ITER} G.ITER.residuo = G.residuo \tag{2.5}$$

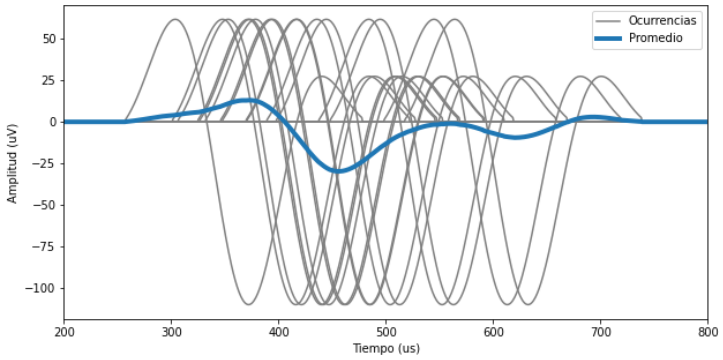

Figura 2.6: Resultado de realizar el promedio de dieciséis ECAPs, debido a la ventana de ocurrencia variable. En gris se observa las dieciséis posibles ocurrencias de las ECAPs, mientras que en azul se observa el resultado de su promedio, $remanente\_ecap$, destacándose la atenuación del promedio.

Al alcanzarse las $ITER$ operaciones, se considera que la *Etapa 2* ha convergido, manteniendo fijo el template del residuo. Por lo tanto, al sustituir este valor del promedio de la señal de entrada, obtenida en la ecuación 2.5, se obtiene que la salida de la segunda etapa es:

$$\begin{aligned} out_2(i) &\approx G.out_1(i) - G.residuo \\ &\approx G.(r(i) + ecap(i) - residuo) \\ &\approx G.ecap(i), \quad i > (C + ITER) \end{aligned} \tag{2.6}$$

Se obtiene a la salida de la *Etapa 2* la señal ECAP, salvo el factor escalar G. Si durante la generación del template de la *Etapa 1* se muestreó parcialmente la ECAP, dichas muestras pueden agruparse al $residuo$. Como este $residuo$ permanece constante una vez alcanzada la convergencia de la *Etapa 1*, el muestreo parcial de la ECAP no representa un problema, ya que su contribución se cancela durante el procesamiento de la *Etapa 2*, tal como se muestra en la ecuación 2.6.

Una explicación más en profundidad de la implementación de esta etapa y su funcionamiento se presentará en el capítulo 4.

Con el objetivo de utilizar la mínima cantidad de recursos, siguiendo los lineamientos de diseñar un sistema para bajo consumo, se decidió utilizar un único ADC, y que éste sea un recurso compartido. Tanto la *Etapa 1* como la *Etapa 2* son las que requieren los datos del ADC, por lo que es necesario administrar en cada etapa cuándo debe utilizado. Para ello, el *Control Principal*, ilustrado en la figura 2.4, habilita el funcionamiento de las etapas y arbitra el flujo de datos entre ellas y el ADC.

Al inicio, el *Control Principal* habilita y le cede el funcionamiento del ADC a la *Etapa 1* hasta que alcance la convergencia. La *Etapa 2* por su parte se encuentra en espera. Cuando la *Etapa 1* logra la convergencia de su template, el mismo detiene su ciclo iterativo y se mantiene fijo, por lo que no requiere más los datos provenientes del ADC. Con la *Etapa 1* utilizando su template en memoria, ahora el ADC muestrea la salida amplificada de la *Etapa 1* y le cede los datos a la *Etapa*

2. La *Etapa 2* se mantiene recibiendo los datos del ADC, de forma que luego de que calcula su template se obtiene la ECAP a la salida. El sistema se mantiene en este estado mientras la señal SA en la entrada del sistema mantenga su morfología constante a lo largo de los períodos.

Si por algún motivo la morfología de la SA cambia, el template de la *Etapa 1* deja de ser válido, aumentando en magnitud el residuo en la salida de la *Etapa 1*. En consecuencia, la entrada de la *Etapa 2*, que es este residuo sumado a la ECAP, luego de ser amplificado, llega al punto en que se excede el rango de tensión admisible por el ADC, produciendo una saturación y perdiendo la linealidad del sistema. En caso de ocurrir este fenómeno, se le vuelve a habilitar la *Etapa 1* y ceder los datos a ella, de forma que su template vuelva a converger, para retomar el ciclo nuevamente y habilitar la *Etapa 2* posteriormente. En el capítulo 5 se explica el módulo *Control Principal* y este ciclo de funcionamiento en detalle.

2.3.2. Validación del sistema

Con el objetivo de validar el sistema, se realizaron pruebas inicialmente en simulación. Cuando el sistema se comportó de la forma esperada, se procedió a las pruebas a nivel de HW. Primero, se realizaron simulaciones modulares y top level del sistema CANICs completo. Para ello, se utilizó el software Cadence, y fue necesario generar el entorno de simulación adecuado, generando modelos Verilog-A de los componentes analógicos utilizados. Con las simulaciones top level del sistema validadas, se procedió a fabricar una PCB, la cual contiene todo el sistema analógico necesario para poder ejecutar el sistema CANICs, y es compatible con la FPGA que contiene el circuito digital implementado. Los componentes analógicos se eligieron de forma que fueran lo más compatibles posibles con una arquitectura de diseño para circuitos integrados. Por ejemplo, se usaron ADC y DAC con comunicación en paralelo, dado que se consideró que era la arquitectura más adecuada para este tipo de diseño. Una explicación más detallada de las simulaciones se encuentra en el capítulo 6, y de la PCB en el capítulo 7.

A la hora de generar la estimulación a la entrada del sistema CANICs, fue necesario generar la señal SENSADO, es decir, generar la SA y ECAP superpuestas. Para ello, un generador de señales introduce la señal SENSADO al sistema. Las señales provistas al generador fueron creadas en base a los modelos explicados previamente en la sección 2.2, a través de un script en Python. En paralelo, se implementó otro script en Python, el cual, luego de cargarse los datos en la FPGA, realiza la captura de los datos de salida del sistema y los guarda en archivos formato CSV para poder utilizarlos posteriormente.

Finalmente, con los datos obtenidos de la FPGA, se hace un postprocesamiento para obtener las características medidas de la ECAP. Este consta de realizar promedios de la tensión pico a pico de las ECAPs obtenidas, a modo de eliminar las variaciones y obtener un resultado más preciso. Se utilizó esta característica de la ECAP porque en base a este valor se puede conocer si la terapia está siendo eficiente o no, como se explicó en la sección 1.1.1.

Además, se realiza el cálculo de la función de correlación con la ECAP de

referencia, a modo de tener una métrica de la fiabilidad de los resultados obtenidos. En el capítulo 8 se explica en detalle la generación de señales como las medidas realizadas y los resultados obtenidos.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Etapa 1: Primera cancelación de la SA

3.1. Introducción

Este capítulo profundiza el desarrollo de la *Etapa 1*, siendo este uno de los componentes principales de la arquitectura del sistema CANICs, explicada en el capítulo 2. Esta etapa se encarga de procesar la señal de entrada del sistema SENSADO y eliminar de ella la SA, con el objetivo de obtener en su salida la ECAP, junto con el residuo de la operación realizada. Cabe destacar que este residuo obtenido es del mismo orden de magnitud que el de la ECAP, por lo que a la salida de esta etapa no se espera obtener la señal ECAP, sino solamente poder extraer de la señal SENSADO la mayor parte de la señal SA, la cual es de magnitud mucho mayor. De esta forma, la señal resultante, ECAP más residuo, es procesada por la *Etapa 2* que es capaz de aislar la ECAP del residuo. Se profundizará en detalle sobre esto en el capítulo 4.

Esta etapa está conformada por los bloques detallados en la figura 3.1. Se indica solamente el flujo de interconexiones entre bloques, a modo de simplificar el diagrama. Siguiendo el flujo de procesamiento de la señal de entrada analógica **Vin**, comienza su procesamiento a través del *amplificador de entrada*, en donde se acondiciona a una señal single-ended. La salida de este amplificador es la denominada señal SENSADO. Luego, la señal SENSADO ingresa al nodo positivo del *amplificador diferencial*. La salida de este amplificador se transforma al dominio digital, que la procesa y genera el template de la SA, explicado en la sección 2.3.1. Luego, esta señal se convierte al dominio analógico e ingresa al nodo negativo del amplificador diferencial. Como se ve en la figura 3.1, la *Etapa 1* genera entonces un lazo como forma de eliminar la SA. Además, en la figura se indica de forma elemental el flujo de las señales de control, las cuales se administran desde el bloque *Control Principal*. Este bloque además controla el bloque de procesamiento de la primera etapa *DSP*. En las siguientes secciones se detallará el funcionamiento particular de cada uno de los bloques. Cabe destacar que el *Control Principal*, el *Controlador del ADC* y el *ADC* son recursos compartidos con la segunda etapa.

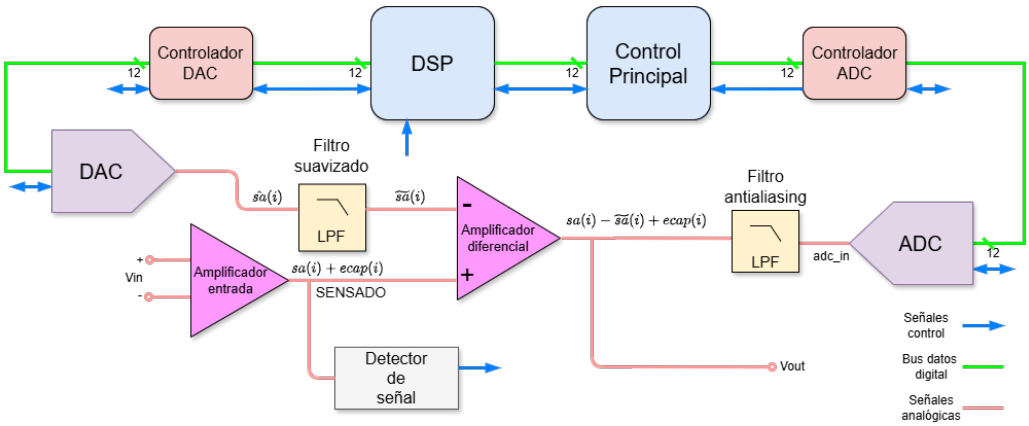

Figura 3.1: Diagrama de bloques simplificado de la etapa 1.

3.2. Principio de funcionamiento

Siguiendo el flujo de señales de la figura 3.1, comenzando por la señal diferencial de entrada **Vin**, esta es convertida a modo single-ended por el amplificador de entrada. La amplificación no es necesaria en esta implementación, dado que la señal **Vin** ya ingresa al sistema en el nivel de tensión requerido. Esto se debe a que el script que genera las señales las amplifica por software a los niveles de tensión necesarios, simulando que el *amplificador de entrada* cuenta con ganancia variable, ajustando adecuadamente la amplitud de la señal entrante, para aprovechar al máximo la excursión en el sistema CANICs. La implementación de esta ganancia variable para el *amplificador de entrada* quedó por fuera del alcance del proyecto y es parte del trabajo a futuro. Esta decisión de diseño permite simplificar la etapa de amplificación de entrada, que de otro modo requeriría una ganancia variable para adaptarse al rango dinámico. Además, se optó por esta estrategia debido a que la resolución de los generadores de señales convencionales no es suficiente para representar adecuadamente la señal ECAP en sus niveles originales (orden de cientos de μV), donde las amplitudes mínimas que pueden generar típicamente son del orden del mV.

A la salida del amplificador de entrada se encuentra la señal SENSADO, correspondiente a $sa(i) + ecap(i)$ de la figura 2.5. Esta señal ingresa a dos bloques, el *detector de señal* y al *amplificador diferencial*. El *detector de señal* se encarga de generar un pulso de 1 T_{CLK} de sincronismo, cuando detecta determinado nivel de la señal $sa(i)$. Este pulso tiene el objetivo de sincronizar la llegada de la señal SA con el funcionamiento del sistema digital. El funcionamiento detallado de este bloque se explicará en la sección 3.4. Este bloque es fundamental para el buen desempeño del sistema CANICs, dado que el sistema debe sincronizarse con alta precisión a la señal SA. El *detector de señal* junto con el requisito de que la frecuencia de reloj del sistema digital debe ser múltiplo de la frecuencia de estimulación, garantizan el buen sincronismo entre la señal de entrada y el sistema digital. Las características del sincronismo se detallan en la sección 3.5. Por otro lado, el amplificador diferen-

cial es el equivalente al módulo restador del esquema de la figura 2.5, en donde en la entrada positiva recibe $sa(i) + ecap(i)$, y en la entrada negativa recibe a $\widetilde{sa}(i)$. En su salida se encuentra el residuo $sa(i) - \widetilde{sa}(i)$ más la ECAP, el cual es la salida del sistema **Vout**.

El bloque *Template de sa(i)* de la figura 2.5 se corresponde con los bloques desde el ADC al DAC, los cuales son el diseño central de la *Etapa 1*. Luego de que la salida del amplificador diferencial sea procesada por el filtro anti aliasing, para cumplir con el Teorema de Muestreo de Nyquist, se muestrea con el ADC. Para que esto sea posible, el ADC tiene su respectivo controlador, el cual es gestionado por el bloque *Control Principal*. Este bloque controla el funcionamiento de las etapas del sistema CANICs y administra los datos del ADC para ambas etapas. Luego, el *Control Principal* le cede los datos y habilita el funcionamiento del bloque *DSP*. Este es el módulo central de la *Etapa 1*, que contiene el template y es el encargado de ejecutar el algoritmo de la ecuación 2.1. Cuando el *DSP* genera un template digital $sa(n)$, este se sincroniza con la señal de entrada $sa(i)$, a través de la señal provista por el *Detector de señal*, y genera la representación analógica del template $\widetilde{sa}(i)$, usando el DAC y su respectivo controlador, el cual gestiona el bloque *DSP*. Finalmente, la señal $\widetilde{sa}(i)$, luego de ser procesada por el filtro de suavizado, para eliminar los componentes residuales de alta frecuencia que introduce el DAC, ingresa al amplificador diferencial para cerrar el lazo de realimentación. El funcionamiento en detalle de cada bloque utilizado se desarrollará a lo largo de este capítulo.

3.3. Amplificador de Entrada

El circuito de amplificación de entrada, denominado INA, tiene como objetivo convertir la señal de entrada diferencial **Vin** a la señal single-ended SENSADO, además de eliminar las componentes de modo común que estas señales puedan contener. De aquí en adelante, se considerará como entrada del sistema a la señal SENSADO. Este circuito es prácticamente idéntico al utilizado por CANE [11] y se ilustra en la figura 3.2. Está compuesto por un amplificador de instrumentación, en el que en sus entradas posee un filtro pasa altos. El filtro cumple con el objetivo de desacoplar la continua, ubicando su polo en 35 Hz. En sus terminales de alimentación posee capacitores para reducir las variaciones y el ruido en estas entradas. La ganancia del circuito, la cual se modifica colocando una resistencia entre las terminales RG1 y RG2, se determinó que fuera unitaria. Esto se debe a motivos prácticos a la hora de generar las señales: los generadores de onda convencionales no tienen resoluciones de μV. La solución utilizada fue que el generador de señales ya implemente la señal amplificada al rango de tensiones utilizado por el sistema, y que este amplificador funcione con ganancia unitaria. Como se mencionó, el objetivo de la ganancia no es considerado en este diseño, dado que la implementación del modo de ganancia variable en el amplificador de entrada quedó por fuera del alcance del proyecto. La verdadera funcionalidad de este circuito, y el motivo por el cual se agregó, fue para mitigar el modo común, dado que, al estar presente en ambas entradas AD2_IN_N y AD2_IN_P, por el funcionamiento del INA, el

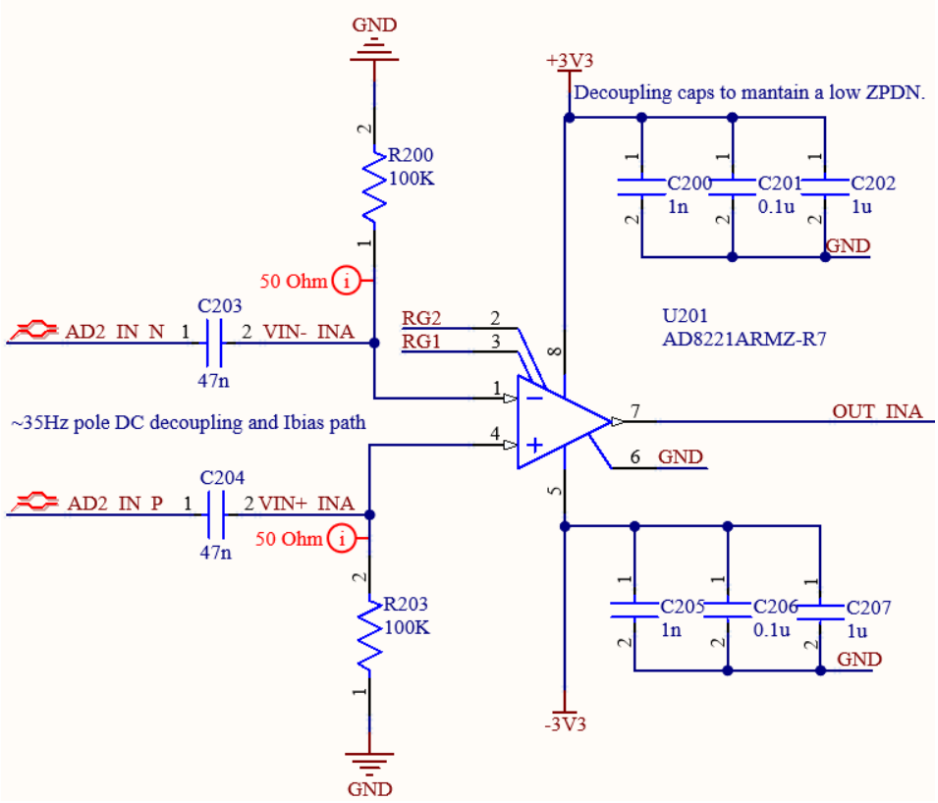

Figura 3.2: Amplificador de instrumentación de entrada.

modo común se cancela.

El amplificador utilizado es un AD8221, el cual posee 80 dB de CMRR y ancho de banda de 825 kHz [16], el cual cubre cómodamente el ancho de banda de la señal SENSADO. Además, dado que es el circuito de entrada del sistema, es extremadamente crítico que introduzca muy bajo ruido, y efectivamente cumple esta característica, dado que tiene una densidad espectral de ruido de $8\,\mathrm{nV}/\sqrt{\mathrm{Hz}}$ a 1 kHz. La elección de las tensiones de alimentación del amplificador se explica en el capítulo 7 del diseño de la PCB.

3.4. Detector de Señal Entrante

El objetivo de este bloque es sincronizar periódicamente el sistema digital con la señal de entrada SENSADO. Como se explicó en el capítulo 1 y se detalla en la figura 1.6, la señal de entrada SENSADO surge en respuesta a una señal que será creada por el generador de estímulos, a través de un pulso de corriente, por lo que el sistema neuroestimulador sabe cuándo el pulso de corriente de estímulo será generado.

Como el bloque *generador de estímulos* del neuroestimulador (figura 1.6) sabe cuándo generar el estímulo de corriente, sería posible que este bloque le provea una

señal digital periódica de sincronismo al sistema CANICs. El motivo por el cual se usa este bloque detector es que el estímulo es en corriente y la señal de entrada SENSADO es la respuesta en tensión a este estímulo. En consecuencia, los cambios de impedancia de los electrodos generan que en un mismo valor temporal dentro del ciclo t^* se obtengan niveles de tensión diferentes en los próximos ciclos, por lo que $\text{SENSADO}(t^*) \neq \text{SENSADO}(t^* + kT), k \in \mathbb{Z}$, produciendo como resultado variaciones en las medidas entre los ciclos muestreados [10]. Aunque estas variaciones se consideran mucho más lentas respecto al período de la señal SENSADO, no se consideró apropiado utilizar una señal de sincronización generada directamente por el bloque *generador de estímulos*. Esto se debe a que estas variaciones medidas entre ciclos de la señal de entrada se traducen en cambios en la morfología y amplitud del residuo, lo que degrada notoriamente el desempeño de la segunda etapa.

Dado que el sistema de procesamiento de la primera etapa es un módulo digital (*DSP*) y la señal de entrada SENSADO es analógica, el *Detector de Señal Entrante* requiere una etapa analógica que permita el procesamiento de la señal SENSADO y la acondicione para que pueda ser procesada por un bloque digital. Posteriormente, este bloque digital obtiene la señal de interés a partir del acondicionamiento analógico recibido y la entrega al *DSP*.

Como criterio para detectar la señal analógica SENSADO, se busca un nivel de tensión determinado VTHNEG, con pendiente descendente. Cuando se llega a ese valor de tensión, el bloque detector asigna un pulso **trigger**, de duración $1\,\mathrm{T_{CLK}}$, con el fin de que el sistema digital lo reciba correctamente. A continuación se explica en detalle el funcionamiento de los bloques analógico y digital que componen al *Detector de Señal Entrante*.

3.4.1. Circuito Analógico

El circuito tiene como objetivo detectar cuándo la señal SENSADO ($sa(t) + ecap(t)$) cruza el umbral de tensión VTHNEG. El valor de VTHNEG fue determinado en base a que sea un valor cercano a cero, el cual es la tensión en reposo, pero lo suficientemente alejado de cero, para evitar glitches. Además, es necesario que en ese nivel la señal tenga una pendiente significativa, para poder detectar sus variaciones; ya que si se analiza en una región plana, donde su valor permanece prácticamente constante, se obtiene un punto de sincronismo con muy poca precisión, ya que no es posible distinguir cambios significativos a lo largo del tiempo. Considerando estos factores, se fijó el valor VTHNEG en $-300\,\mathrm{mV}$, en donde la señal SA se encuentra en su pendiente de descenso (ver sección 2.2).

El circuito utilizado para lograr detectar la señal VTHNEG se presenta en la figura 3.3. Está compuesto por un amplificador, configurado como un comparador con histéresis del tipo Schmitt Trigger y a su salida posee un rectificador de media onda. La entrada POS_IN en el circuito se corresponde con la señal SENSADO de la figura 3.1.

El amplificador tiene el comportamiento en su salida AMP_OUT como lo describe la figura 3.4. También se desprenden de la figura 3.4 los valores de VTHNEG

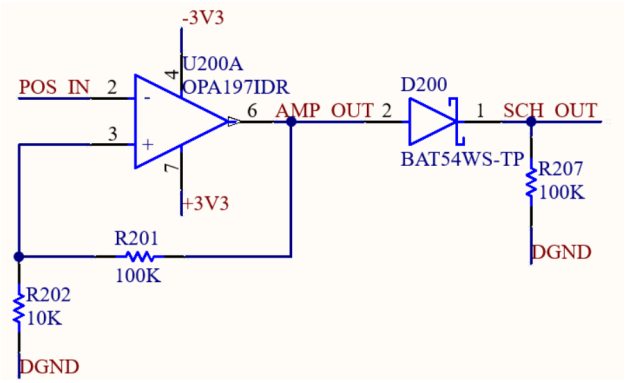

Figura 3.3: Circuito comparador en cascada con un rectificador de media onda. Su entrada POS_IN se corresponde a la señal SENSADO.

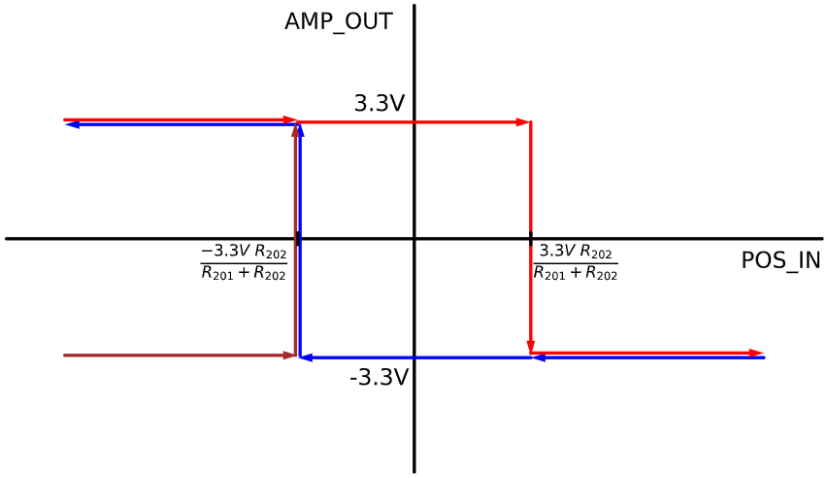

Figura 3.4: Ventana de histéresis del amplificador. Las flechas rojas y azules indican el comportamiento en régimen del circuito, mientras que las marrones son un posible caso de arranque que puede tener el circuito en caso de que, al encenderse, la salida AMP_OUT sea $-3{,}3\,\mathrm{V}$ y la señal de entrada POS_IN sea menor al valor VTHNEG.

y VTHPOS, los cuales son:

$$VTHNEG = \frac{-3{,}3V\,R_{202}}{R_{201}+R_{202}} = -300\,\mathrm{mV} \qquad VTHPOS = \frac{3{,}3V\,R_{202}}{R_{201}+R_{202}} = 300\,\mathrm{mV}, \tag{3.1}$$

en donde $R_{201} = 100\,\mathrm{K\Omega}$ y $R_{202} = 10\,\mathrm{K\Omega}$.

El propósito del rectificador es acondicionar la señal de salida del operacional, la cual excursiona entre $3{,}3\,\mathrm{V}$ y $-3{,}3\,\mathrm{V}$. Esto es para que el circuito digital, que opera entre $0\,\mathrm{V}$ y $3{,}3\,\mathrm{V}$, pueda manejarla correctamente, evitando daños a los componentes digitales del sistema.

Como se observa en la simulación del circuito implementado en la figura 3.5, la señal SCH_OUT cambia de estado entre los valores de $0\,\mathrm{V}$ y $3{,}3\,\mathrm{V}$, valiendo $0\,\mathrm{V}$ cuando el puerto de entrada POS_IN, al que se le conecta la señal SENSADO,

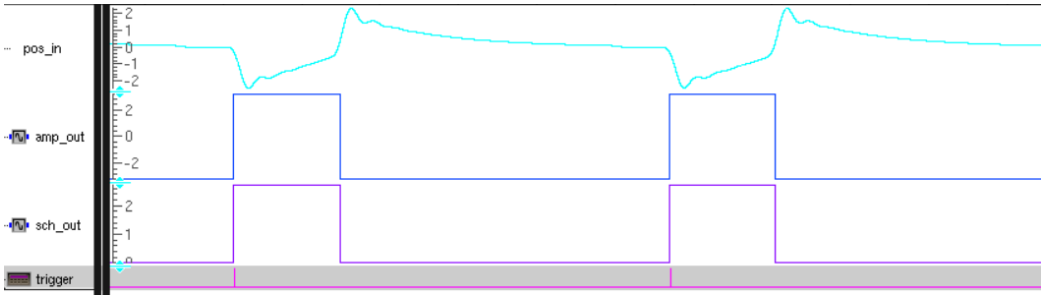

Figura 3.5: Simulación del circuito detector de señales de entrada.

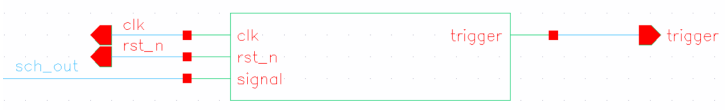

Figura 3.6: Bloque digital del circuito detector de señal.

posee valores mayores a VTHNEG; para luego tomar el valor de 3,3 V, cuando la señal SENSADO cruza el umbral VTHNEG y se mantiene debajo del umbral VTHPOS. Al superar el umbral VTHPOS, vuelve a valer 0 V, comenzando el ciclo nuevamente.

El amplificador utilizado para este circuito fue el OPA197, el cual se eligió por tener entrada y salida rail-to-rail, su ancho de banda de 10 MHz, junto con un slew rate de 20 V/µs y capacidad para manejar cargas capacitivas de hasta 1 nF permiten una respuesta rápida y precisa frente a señales de alta velocidad [17].

3.4.2. Circuito Digital

El circuito analógico descrito en la sección 3.4.1, obtiene a la salida un pulso periódico prolongado de valor 3,3 V cuando la señal atraviesa los valores VTHNEG y VTHPOS, el cual tiene una duración de varios períodos de reloj. El objetivo de la parte digital de este bloque es detectar el flanco positivo de la salida SCH_OUT, generando un pulso de duración 1 T_{CLK}. De esta forma se alerta de la llegada de una nueva señal SENSADO al sistema. El bloque digital implementado tiene la interfaz ilustrada en la figura 3.6. El pulso digital de salida es la señal trigger, la cual se genera a partir de 4 FFs, dos se utilizan como sincronizadores entre la señal SCH_OUT y el CLK (FF1 y FF2), y los otros dos se utilizan para detectar el flanco, conectándolos en cascada a la salida del sincronizador (FF3 y FF4). Dado que están en cascada, cuando se detecta el flanco, el FF3 se asigna a 1, y en el siguiente período de reloj se asigna el FF4 a 1. El período de reloj entre que ambos FFs se asignan a 1 (FF3=1 y FF4=0) es la salida trigger del bloque. Este pulso trigger se puede ver en conjunto con las otras señales del circuito en la figura 3.5. La señal trigger obtenida, es la señal de sincronismo del sistema CANICs, la cual ajusta el inicio de los bloques digitales con la señal entrante SENSADO.

3.5. Limitación en la frecuencia del sistema digital

A continuación se explica por qué es importante que el sistema CANICs adopte, como decisión de diseño, que la frecuencia de reloj del sistema digital (F_{CLK}) sea un múltiplo de la frecuencia de estimulación (f_{SA}). Para ello, se presentan dos ejemplos ilustrativos simples: primero se analiza el caso en que ambas frecuencias son independientes, y luego se estudia el caso particular en que $F_{CLK} = 4f_{SA}$.

Como se explicó en la sección anterior, el detector de señal entrante tiene como objetivo sincronizar periódicamente el sistema digital con la señal de entrada SENSADO. Dado que el sistema digital requiere un reloj para su funcionamiento, y que el procesamiento comienza con la llegada de la señal digital trigger, es fundamental diseñar la frecuencia de reloj del sistema (F_{CLK}) de manera que la señal trigger sea periódica. Esto garantiza que el algoritmo implementado en el sistema digital se ejecute de forma periódica y estable, evitando comportamientos indeseados o asincronismos que puedan degradar su desempeño.

Para asegurar la periodicidad de la señal trigger, es suficiente que la frecuencia de reloj del sistema (F_{CLK}) sea un múltiplo de la frecuencia de estimulación (f_{SA}). La figura 3.7 ilustra este comportamiento: cuando la señal SENSADO (señal azul) supera el umbral, el circuito analógico Schmitt Trigger se activa (indicado con la línea punteada), el sistema digital detecta este evento y, al tratarse de un sistema secuencial, genera el pulso de trigger en el siguiente flanco de reloj disponible. Este instante se muestra con una flecha verde en la figura.

Dado que la frecuencia de reloj del sistema es un múltiplo de la frecuencia de estimulación ($F_{CLK} = 4f_{SA}$), el desfase temporal entre el momento en que se activa el Schmitt Trigger y el instante en que el sistema digital genera la señal trigger (en el flanco de reloj siguiente) se mantiene constante a lo largo del tiempo. Esto garantiza que el intervalo entre pulsos sucesivos de trigger (indicados por la flecha verde) sea fijo, resultando en una señal periódica de $4T_{CLK}$ (este valor es ilustrativo y no representa un valor específico del sistema).

A modo complementario, en la figura 3.8 se muestra el caso en que la frecuencia de reloj del sistema no es un múltiplo de la frecuencia de estimulación. En esta situación, el desfase temporal entre la activación del Schmitt Trigger (línea punteada) y el instante en que el sistema digital genera la señal trigger (flecha verde) no es constante. Como resultado, el intervalo entre pulsos consecutivos de trigger deja de ser constante, oscilando en este ejemplo entre $3T_{CLK}$ y $4T_{CLK}$. Esta variabilidad provoca que el sistema digital inicie su procesamiento en momentos distintos respecto a la señal SA, lo cual no es deseado, ya que compromete la periodicidad del procesamiento.

Por esta razón, como decisión de diseño, se optó por limitar la frecuencia del sistema F_{CLK} a valores que sean múltiplos de la frecuencia de estimulación ($f_{SA} = 900\,\text{Hz}$). Esto se expresa matemáticamente en la ecuación 3.2, donde la constante $K \in \mathbb{N}$ se definió en $K = 5000$, lo que será explicado en la sección 3.8.2.

$$F_{CLK} = K \times f_{SA} = K \times 900, \quad K \in \mathbb{N} \tag{3.2}$$

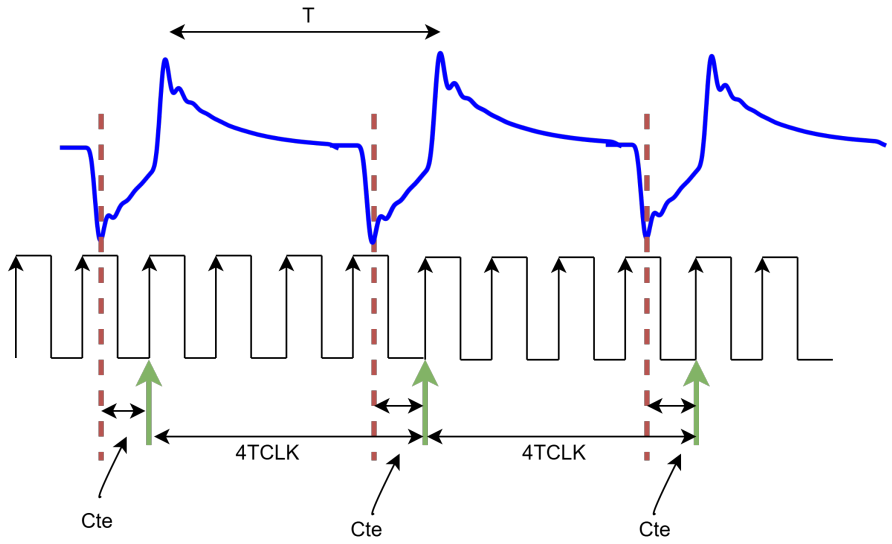

Figura 3.7: Ejemplo de sincronización periódica entre el sistema digital y la señal analógica de entrada SENSADO. Al ser la frecuencia de reloj ($\frac{1}{T_{CLK}}$) un múltiplo de la frecuencia de estimulación ($\frac{1}{T}$), el tiempo entre la activación del Schmitt Trigger (línea punteada) y la generación del pulso trigger (flecha verde) se mantiene constante, garantizando una señal de trigger periódica. El valor de $4T_{CLK}$ mostrado es ilustrativo y no representa un valor especifico del sistema.

3.6. Amplificador diferencial

A la hora de la implementación, comparando con la figura 3.1 el operador restador del diagrama está compuesto principalmente por el amplificador diferencial, el cual se implementó siguiendo el circuito de la figura 3.9. La señal SA+ECAP se corresponde a la entrada proveniente del amplificador de entrada. Su otra entrada, DAC, es la salida filtrada proveniente del DAC, que se corresponde con la señal $\widetilde{sa}(i)$. Luego, el residuo es la propia salida del amplificador diferencial ETAPA_1_OUT. El residuo ingresa al entorno digital a través del ADC, en donde se genera el modelo $\widetilde{sa}(n)$, el cual el DAC lo traduce al mundo analógico, generando la señal $\widetilde{sa}(i)$ y vuelve a ingresar al amplificador diferencial para cerrar la realimentación del sistema. Es importante destacar que la resistencia $R700D$ del divisor de tensión está conectada a una referencia de 2,5 V. Esto genera un offset de 2,5 V en la señal de salida. La razón de este diseño es que el ADC requiere que las señales analógicas estén referenciadas a 2,5 V, lo cual se explica en la sección 3.8.3. La elección de las tensiones de alimentación del amplificador se explica en el capítulo 7.

El amplificador utilizado fue el OPA182IDBVR, el cual es un amplificador operacional de precisión, de respuesta dinámica rápida y bajo nivel de ruido. Posee un offset ultra bajo (2 µV típico), una densidad de ruido de solo 5,7 nV/$\sqrt{\text{Hz}}$ y un ancho de banda de 5 MHz, más que adecuado para la aplicación. Además, tiene

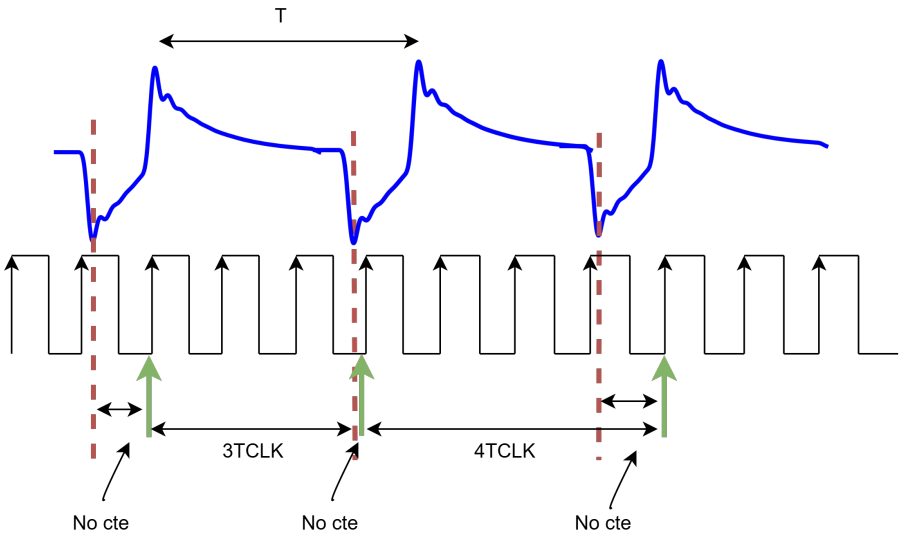

Figura 3.8: Ejemplo de sincronización NO periódica entre el sistema digital y la señal analógica de entrada SENSADO. Al no ser la frecuencia de reloj ($\frac{1}{T_{CLK}}$) un múltiplo de la frecuencia de estimulación ($\frac{1}{T}$), el tiempo entre la activación del Schmitt Trigger (línea punteada) y la generación del pulso trigger (flecha verde) no se mantiene constante, esto genera una señal de trigger que no es periódica. Los valores de $3T_{CLK}$ y $4T_{CLK}$ son ilustrativos y no representan valores específicos del sistema.

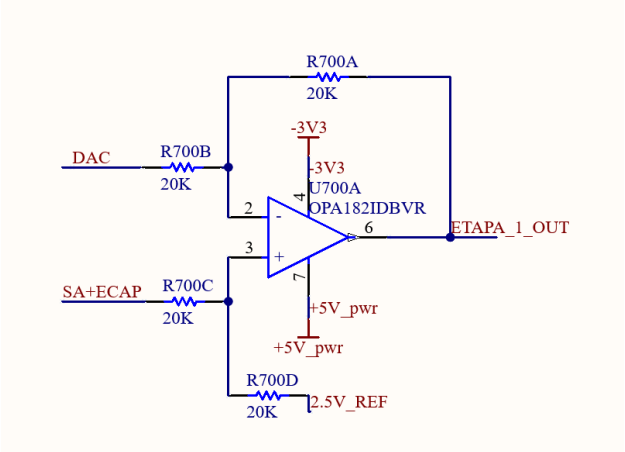

Figura 3.9: Circuito implementado para el amplificador diferencial.

un slew rate adecuado para la aplicación ($10\,\mathrm{V/\mu s}$) y capacidad de salida rail-to-rail [18].

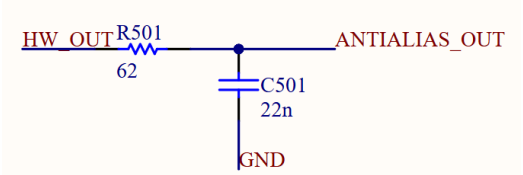

Figura 3.10: Circuito implementado del filtro anti aliasing.

3.7. Filtro Anti Aliasing

El propósito de este filtro es limitar el ancho de banda de la señal de entrada del ADC, de forma que no se produzca aliasing. La señal de entrada del ADC, como se puede ver en la figura 3.1 es la resta de la señal de entrada y el template generado por el DAC. Dado que el DAC lo que genera son pulsos escalonados, aunque el filtro de suavizado disminuye significativamente el ancho de banda, la señal resultante de esta resta puede contener componentes de frecuencia mayor a la de Nyquist, por lo que la implementación del filtro anti aliasing es crítica para reducir el ruido en el sistema.

Dada la frecuencia de muestreo del ADC, la cual es de 237 KHz (ver sección 3.8), el ancho de banda máximo que el filtro puede tener es de 118 KHz. Considerando esta restricción, se eligió utilizar un filtro de primer orden con un ancho de banda de 117 KHz. Se eligió este ancho de banda debido a que cumple con la restricción de Nyquist, además de que es un valor equilibrado en velocidad de respuesta, debido a que en esa parte del lazo es crítico no introducir retardos significativos; sino, la sincronización entre el ADC y el DAC a la hora de manejar los datos puede ser incompatible. En la sección 3.9.1 se detalla esta problemática. Considerando el ancho de banda y que es una transferencia de primer orden, se puede estimar el retardo temporal que este filtro introduce:

$$t_d \approx 3\tau = \frac{3}{2\pi fc} = \frac{3}{2\pi 117\,\mathrm{KHz}} \approx 3{,}979\,\mu\mathrm{s}. \tag{3.3}$$

La implementación del filtro fue de un circuito pasabajo RC, como lo ilustra la figura 3.10.

3.8. ADC

3.8.1. Especificaciones del ADC

El ADC es el encargado de convertir señales de tensión del dominio analógico al dominio digital. Al igual que el DAC, es un componente central de la *Etapa 1*, el cual define varios aspectos importantes en el rendimiento del sistema CANICs.

El ADC buscado tiene que poseer interfaz paralelo. Esta decisión se basó en la posibilidad de que, en futuras continuaciones de este proyecto, el sistema de terapia se integre en un SoC (System on Chip). En este escenario, la transferencia

de datos entre los dominios analógico y digital ocurriría internamente dentro del chip, lo que haría innecesaria una comunicación serial, como SPI o I2C. Además de interfaz paralelo, debe contar con 12 bits de resolución, poseer al menos dos canales de entrada analógica que soporten modo bipolar (tensión tanto positiva como negativa), uno para cada etapa y tener al menos la mínima velocidad de muestreo requerida por el sistema, que siguiendo el teorema de muestreo de Nyquist debe ser mayor a 32 Ksps. Además, idealmente debe poder operarse con frecuencias de reloj relativamente bajas (orden MHz), siguiendo las características de un ADC para un sistema de bajo consumo.

Dicho esto, el ADC elegido que cumple con los requisitos definidos anteriormente fue el MAX1266 de Maxim Integrated Products [19], siendo un ADC del tipo aproximaciones sucesivas. Sus características más relevantes a ser utilizadas son que tiene 6 canales analógicos single-ended con modo bipolar, el bus de datos se representa en complemento a 2 y posee una máxima frecuencia de muestreo de 420 Ksps. La tensión de funcionamiento son $VDD = 5\,\text{V}$, lo cual es un requisito del componente y se tuvo que adaptar la arquitectura del sistema para cumplirlo (se explicará en el capítulo 7). Para la referencia de tensión para la conversión de datos, el ADC permite utilizar una referencia interna o la posibilidad de utilizar referencia externa de hasta $VREF = VDD + 50\,\text{mV}$. La frecuencia de reloj máxima que soporta es de 7,6 MHz, dejando un margen de elección aceptable para la aplicación. También presenta la característica de tener un bus triestado de 12 bits, el cual sirve tanto para la salida de los datos muestreados, como para la entrada de datos utilizados para configurar el modo de adquisición e iniciar el muestreo. En la siguiente sección se explica el modo de funcionamiento del ADC, explicando el manejo utilizado del bus triestado y la frecuencia de funcionamiento elegida.

3.8.2. Elección de la frecuencia de muestreo

La frecuencia de muestreo del sistema (f_s) está determinada por la frecuencia de reloj del sistema (F_{CLK}) y por la forma en que el controlador del ADC gestiona la comunicación con el mismo. En particular, depende de la cadencia con la que el ADC puede entregar nuevas muestras. En este caso, el ADC tiene una latencia mínima de 19 ciclos de reloj del sistema ($19 \times T_{CLK}$) por conversión, lo que implica que puede proporcionar una nueva muestra como mínimo cada 19 ciclos de reloj (esto se detalla en la sección 3.9 del *Controlador del ADC*).

Como se explicó en la sección 3.5, se optó como decisión de diseño elegir la frecuencia del sistema como un múltiplo de la frecuencia de estimulación. Teniendo presente la ecuación 3.2, se puede escribir la frecuencia de muestreo de la siguiente forma:

$$fs = \frac{F_{CLK}}{19} = \frac{K \times 900}{19}, \quad K \in \mathbb{N} \tag{3.4}$$

Para definir el valor de la frecuencia de muestreo, basta con seleccionar un valor de $K \in \mathbb{N}$. En este caso, se optó por no elegir una frecuencia de muestreo excesivamente baja, con el objetivo de no exigir en exceso al filtro antialias y evitar

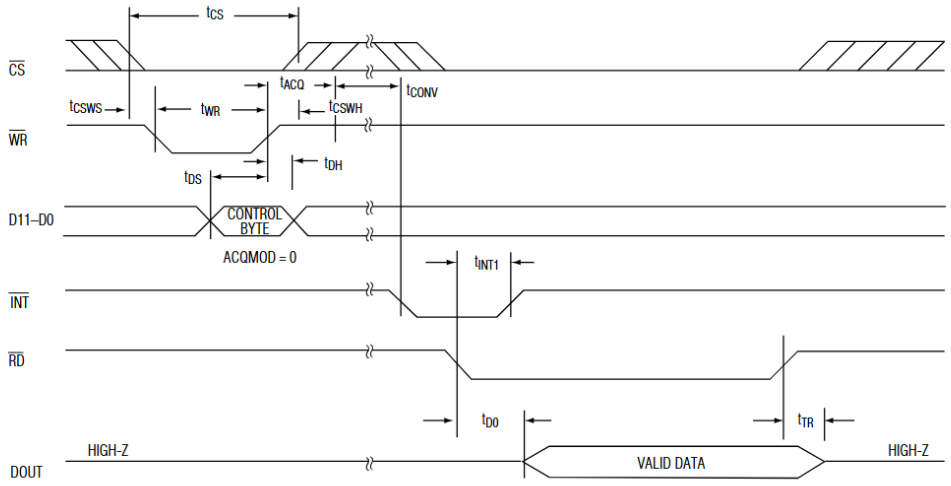

Figura 3.11: Diagrama de tiempos del ADC MAX1266 en *internal acquisition mode*. Se observa el comportamiento esperado de las señales de control $\overline{CS}$, $\overline{WR}$, $\overline{RD}$ con la señal de interrupción $\overline{INT}$, junto con el bus triestado `D11-D0`. Fuente: Hoja de datos del MAX1266 [19].

un aumento significativo del ruido en el sistema (teniendo en cuenta que se trata de un filtro de primer orden, cuya pendiente de atenuación es de apenas 20 dB por década, por lo que necesita contar con margen suficiente más allá de su frecuencia de corte para atenuar eficazmente). Por otro lado, no asignarla excesivamente alta, para no hacer uso de una memoria extensa, aunque asegurando que la frecuencia de Nyquist estuviera lo suficientemente alejada del ancho de banda de la señal de interés (aproximadamente una década del $BW_{SA} = 16\,\text{kHz}$).

En este escenario, se eligió $K = 5000$ determinando que la frecuencia del sistema sea $f_{CLK} = 4{,}5\,\text{MHz}$, donde utilizando la ecuación 3.4 se determina que la frecuencia de muestreo vale lo de la ecuación 3.5. Esta frecuencia de muestreo es un balance equilibrado entre la velocidad de muestreo y la cantidad de memoria necesaria.

$$fs = \frac{4{,}5\,\text{MHz}}{19} \approx 237\,\text{KHz} \tag{3.5}$$

3.8.3. Modo de operación

El ADC se configuró en *internal acquisition mode*, utilizando un reloj externo de 4,5 MHz y una referencia externa de valor $VREF = 5\,\text{V}$. Como entradas, se utilizaron los canales analógicos *CH0* y *CH1* en modo bipolar. Dado que se configuró en modo bipolar, el ADC requiere que $VCOM \geq \frac{VREF}{2}$, por lo tanto, se utilizó $VCOM = \frac{VREF}{2}$. Cabe destacar que, como el bus de datos es triestado, el mismo se puede configurar en tres modos diferentes: *entrada*, *salida* o *alta impedancia*.

En la figura 3.11, es de interés observar cómo es el comportamiento de las señales de control del ADC para obtener un dato muestreado. Si se quiere iniciar

una nueva adquisición, es necesario activar $\overline{WR}$, imponiendo la palabra de control correspondiente (esto es necesario para configurar el ADC en *internal acquisition mode* en modo bipolar), donde para esto, el puerto triestado $D11 - D0$ permanece como *entrada*. Al desactivar $\overline{WR}$ en el flanco de subida, el ADC inicia el muestreo de la señal analógica. Cuando finaliza, activa la señal de interrupción, $\overline{INT}$ para avisar que se tiene un dato nuevo. Durante este intervalo temporal, el puerto triestado $D11 - D0$ permanece como *alta impedancia*. La señal de interrupción permanecerá activa hasta que el maestro quiera adquirir el dato muestreado, activando la señal $\overline{RD}$, donde el puerto triestado $D11 - D0$ permanece como *salida*, imponiendo el dato muestreado.

Con la configuración descrita se tiene que:

- Rango de entrada analógico respecto a $VCOM$ desde $\frac{+VREF}{2} = 2{,}5\,\mathrm{V}$ hasta $\frac{-VREF}{2} = -2{,}5\,\mathrm{V}$. Cuando se mide respecto a tierra, el rango se extiende desde $0\,\mathrm{V}$ hasta $VREF = 5\,\mathrm{V}$.
- Frecuencia de muestreo de $fs = \frac{4{,}5\,\mathrm{MHz}}{19} \approx 237\,\mathrm{KHz}$ (ver sección 3.9).
- Resolución del ADC de $LSB = \frac{VREF}{2^{12}} = 1{,}22\,\mathrm{mV}$.

3.9. Controlador del ADC

En la figura 3.12 se aprecia la conexión entre el controlador del ADC diseñado con el ADC en sí mismo. El objetivo de este es operar las señales de control del ADC con el fin de obtener datos muestreados.

Por otro lado, se observa en punteado dos buffers triestado de la familia 74AHC9541A de Nexperia [20]. El objetivo de estos es imponer cuando sea necesario la palabra de configuración en el ADC y enviarle el dato muestreado al controlador (más información sobre el circuito diseñado se encuentra en el capítulo 7).

Cabe destacar que, fijada la frecuencia de reloj del ADC, la frecuencia de muestreo depende de cómo se gestionen las señales de control. A modo de ejemplo, cuando el ADC tenga un dato muestreado, activará su señal de interrupción $\overline{INT}$, permaneciendo activa en el tiempo hasta que el maestro le responda con la activación de $\overline{RD}$. Por ende, dependiendo de la respuesta del maestro, es la velocidad de muestreo obtenida.

En la figura 3.13 se observa el comportamiento de las señales de control diseñadas para muestrear con una cadencia fija. Cuando se habilita el controlador con la señal de enable (ENA), este activa durante un periodo de reloj la señal $\overline{WN}$ para que el ADC inicie el muestreo. Una vez que el controlador detecta que la señal $\overline{INT}$ está activa, envía un pulso en $\overline{RD}$ y guarda el dato muestreado. Consecuentemente, se repite este proceso para obtener sucesivamente datos muestreados. El controlador avisa que tiene un dato nuevo con un pulso en alto en la señal $DONE$.

Al operar las señales de esta forma y debido a los tiempos internos del ADC, se obtiene un dato nuevo cada $19\,\mathrm{T_{CLK}}$. Teniendo presente la frecuencia de reloj, se obtiene que la frecuencia de muestreo es de $fs = \frac{4{,}5\,\mathrm{MHz}}{19} = 237\,\mathrm{KHz}$.

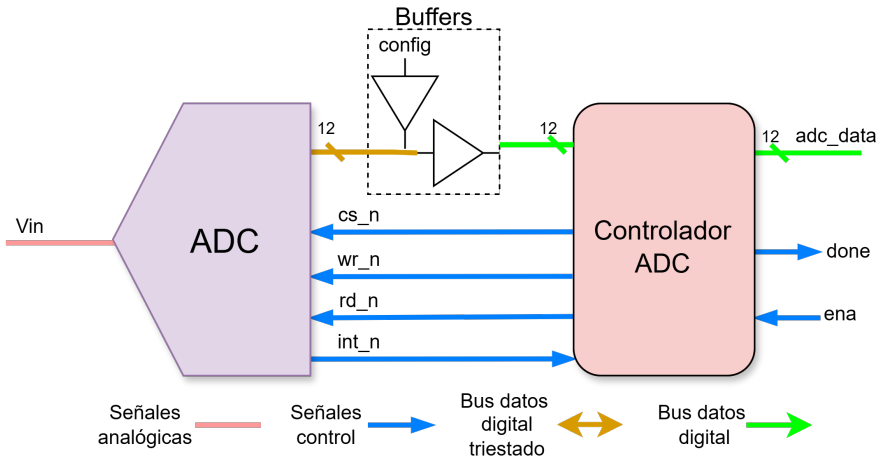

Figura 3.12: Esquema de conexión del controlador del ADC con el ADC, con sus respectivas señales. En punteado dos buffer triestado. La señal **vin** corresponde a la entrada analógica que se desea muestrear. La señal de control **cs_n** habilita el funcionamiento del ADC. Las señales **wr_n**, **int_n** y **rd_n** permiten controlar el proceso de muestreo y lectura de datos. La salida digital del ADC muestreada se representa mediante **adc_data**. Finalmente, la señal **ena** habilita el controlador, mientras que **done** indica que un nuevo dato digital está disponible.

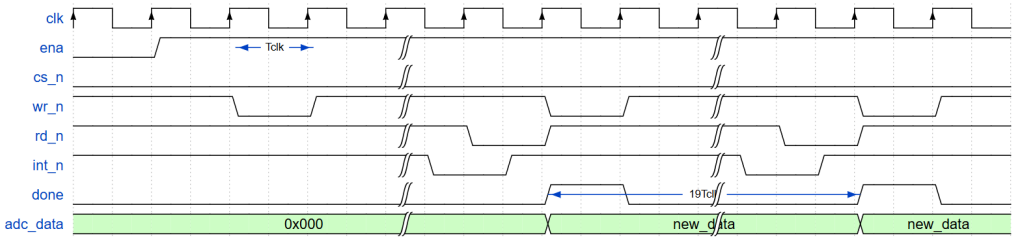

Figura 3.13: Diagrama de tiempos del comportamiento del controlador del ADC diseñado. Al habilitarse mediante la señal **ena**, se activa por un ciclo de reloj la señal **wr_n** para iniciar el muestreo en el ADC. Cuando el controlador detecta que la señal **int_n** está activa, genera un pulso en **rd_n** y almacena el dato digital muestreado. Este proceso se repite para adquirir datos de forma periódica. El controlador indica que un nuevo dato está disponible en **adc_data** mediante un pulso en alto en la señal **done**. La cadencia a la que se obtiene un nuevo dato muestreado es de $19\,\mathrm{T_{CLK}}$.

3.9.1. Necesidad de Retardo para Sincronización

Al agregar filtros en el lazo, además de cumplir con el criterio de Nyquist y disminuir el ruido, implica un compromiso de retardo temporal. Como se explica en la sección 3.7, el filtro antialias impone un retardo de aproximadamente 3,979 µs, mientras que el filtro de suavizado del DAC aporta unos 1,5 µs de retardo adicionales, como explica la sección 3.12. Esto genera un retardo total por parte de los filtros de 5,479 µs. Estos retardos repercuten directamente, tanto en el manejo de la memoria (ver la sección 3.13.1) como en la sincronización del ADC y DAC. Para mitigar este fenómeno correctamente, al momento de que el DAC genera un nuevo valor de salida, se debe esperar el tiempo de asentamiento de los filtros, para que el ADC efectivamente capture el dato generado por el nuevo valor que el DAC está proporcionando.

Estos efectos se ejemplifican en la figura 3.14. La figura muestra en un intervalo de tiempo de unos 9 µs (parte muy pequeña del período de estimulación de 1111 µs) el comportamiento de la señal de entrada SENSADO, en rosa, superpuesta con la salida del DAC sin filtrar $\hat{sa}(t)$ en verde, y la señal filtrada $\widetilde{sa}(t)$ en rojo. A su vez, se incluye la entrada del ADC, **adc_in**, en celeste. Como se observa, desde que se produce el escalón inicial en $\hat{sa}(t)$, indicado con el *Marker 1*, hasta que el DAC asigna un nuevo dato, indicado con el *Marker 2* ocurren aproximadamente 4,23 µs, el cual se corresponde con 19 períodos de reloj. Si bien esta ventana temporal es menor que el retardo total de los filtros (5,479 µs), se observa que **adc_in** está muy cerca de su valor final de asentamiento, por lo que se consideró como un criterio de diseño despreciar el remanente de señal hasta el tiempo final de asentamiento, indicado con **vs** en la figura 3.14.

Por lo tanto, dado el efecto significativo del retardo, si en el controlador del ADC no se tiene en cuenta este efecto, por más que el DAC y el resto del lazo funcionen correctamente, el ADC no va a capturar el valor esperado como consecuencia del último valor impuesto por el DAC, por lo que el lazo no funcionará correctamente. Para mitigar esta problemática, se agregó un retardo de inicio de conversión del ADC. En vez de que la conversión comience inmediatamente después de que el DAC asigne un nuevo dato, se espera el tiempo estimado de retardo que introducen los filtros para capturar los nuevos datos. Este efecto se observa a través de la señal **th_ena**, la cual es la ventana temporal donde el ADC muestra los datos. Por lo tanto, el valor capturado por el ADC se asume que es en el punto **v2**, el cual ocurre cercano a la asignación del nuevo dato del DAC, indicado con el *Marker 2*. Con este tiempo de retardo en el muestreo de datos por parte del ADC se asegura capturar correctamente los datos. En cambio, si no se esperara este tiempo de retardo, cuando se asigna un nuevo valor al DAC e inmediatamente se intenta muestrear con el ADC, lo que en realidad está muestreando el ADC es el dato previo consecuente que el DAC generó. Este fenómeno se indica con el punto **v1** de la figura. Como se observa, el valor de **adc_in** en **v1** es prácticamente idéntico al valor inicial en *Marker 1*, no el valor final esperado **v2**. Por lo tanto, el dato capturado en este punto **v1** es el equivalente al dato previo que el DAC había generado, no el actual. Por este motivo, es crítico para el correcto funcionamiento del lazo respetar el tiempo de asentamiento de los filtros, de forma de capturar los

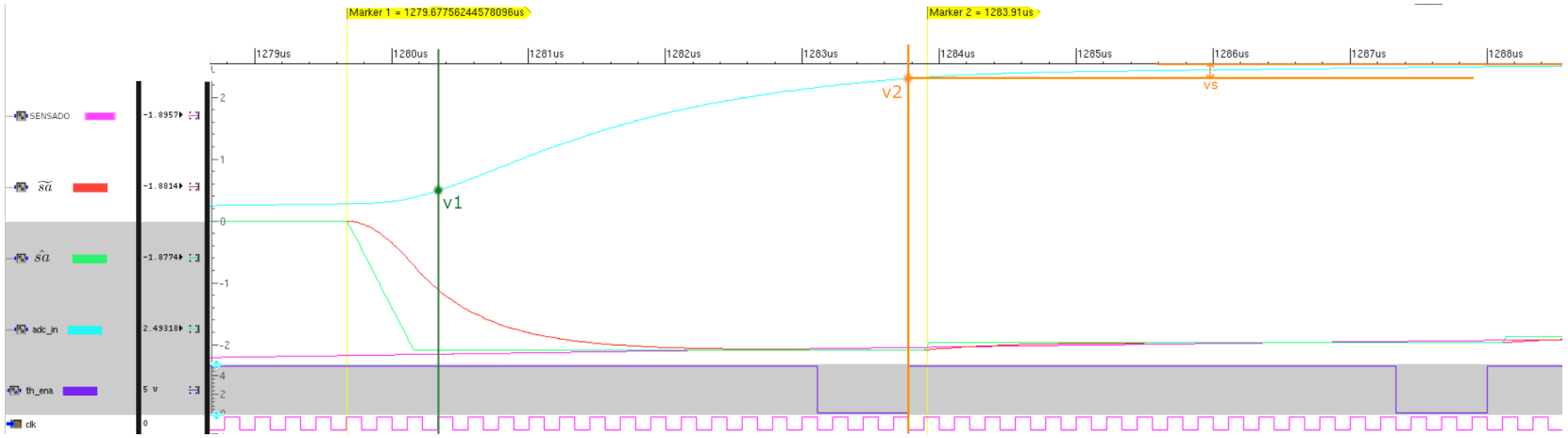

Figura 3.14: Simulación de tiempos del comportamiento del lazo de control principal, donde se observan los retardos en las señales $\widetilde{sa}$ y **adc_in**, impuestos por los filtros utilizados.

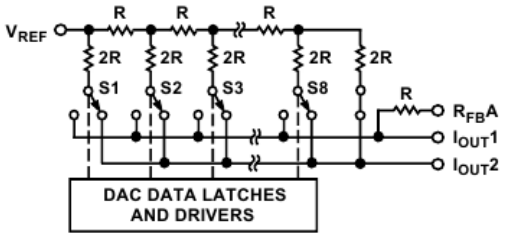

Figura 3.15: Arquitectura R-2R utilizada por el DAC. Fuente: Hoja de datos del AD5445 [21].

datos esperados a la hora de realizar el muestreo con el ADC.

3.10. DAC

El DAC es un componente central del sistema, dado que es el encargado de convertir el template digital de la *Etapa 1* al dominio analógico, de forma de poder cerrar el lazo de esta etapa. Dada la arquitectura del sistema, el DAC elegido debe contar con entrada de datos paralelo y debe poder operar con tensiones positivas y negativas, es decir, en modo bipolar. Además, debe poseer la cantidad de bits que el template contiene (12), tener la suficiente velocidad de respuesta (acorde a la cadencia de datos del ADC) y soportar los niveles de tensión utilizados en el sistema (5 V). Por estos motivos, el DAC utilizado fue un AD5445 de Analog Devices. Este DAC cuenta con las características mencionadas anteriormente, además de que su arquitectura de salida es en corriente. En las siguientes secciones se explica su funcionamiento, el modo de uso y cómo fue implementado en el sistema.

3.10.1. Principio de Funcionamiento

El principio de funcionamiento es un arreglo de resistencias R-2R con switches, siendo su salida en corriente. El arreglo se ilustra en la figura 3.15. Los switches $S_1..S_{12}$ se corresponden con un bit de entrada de datos, donde S_1 es el más significativo y S_{12} el menos significativo. Los switches varían la corriente de salida, debido a los divisores resistivos generados por la cadena R-2R, los cuales cumplen la ecuación:

$$I_{OUT} = \frac{V_{REF}}{R2^i} \qquad i = 1 \dots 12, \tag{3.6}$$

donde i es el índice del switch S_i. Dado que V_{REF} es un nodo de entrada, se puede usar independientemente de la alimentación VDD del DAC.

Como la salida del DAC es en corriente, se necesita un circuito de acondicionamiento para transformarlo a tensión. Como además se utilizó el DAC en modo bipolar, el circuito de acondicionamiento tiene que contemplar este modo de funcionamiento. Para ello se utilizó el circuito de la figura 3.16. Este circuito consta de dos amplificadores. El primero se encarga de transformar la salida en corriente

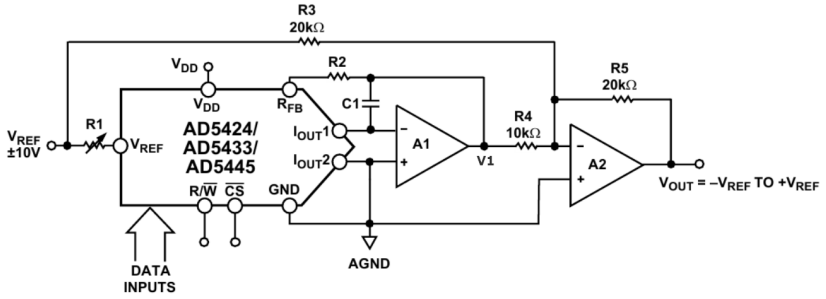

Figura 3.16: Configuración bipolar para el DAC. R1 es para ajuste de ganancia y R2 y C1 se usan en caso de usar amplificadores de alta velocidad, el cual no es el caso de aplicación implementado. Fuente: Hoja de datos del AD5445 [21].

del DAC a tensión, siguiendo la transferencia

$$V1 = \frac{-V_{REF} \times D}{2^n}, \tag{3.7}$$

donde D es la representación decimal de la palabra digital y n el número de bits (12). Este amplificador invierte la polaridad en su salida, por lo que el siguiente amplificador se encarga de volver a invertir la señal y además llevarla al rango de tensión adecuado, agregando un offset a partir de la tensión V_{REF}. El segundo amplificador tiene la transferencia

$$V_{OUT} = -2V1 - V_{REF}. \tag{3.8}$$

Uniendo ambas transferencias se obtiene que la salida total del DAC es

$$V_{OUT} = (\frac{V_{REF} \times D}{2^{n-1}}) - V_{REF}, \tag{3.9}$$

obteniendo la salida en tensión en el rango de $\pm V_{REF}$. Una particularidad del funcionamiento en modo bipolar es que el DAC pierde 1 bit de resolución, dado que el segundo amplificador posee una ganancia de 2 respecto a la entrada $V1$ [21].

3.10.2. Modo de Operación

El DAC interpreta el bus de datos de entrada como datos en código binario convencional, es decir, el valor 12'h0 es el de menor valor de tensión y el valor 12'hFFF es el valor de mayor tensión. En el caso de funcionamiento bipolar que se usa en este sistema, el valor mínimo de tensión es $-V_{REF}$ y el valor máximo de tensión es de $+V_{REF}$.

En cuanto al control para su funcionamiento, posee las señales $\overline{CS}$ y $R/\overline{W}$. Como se detalla en la figura 3.17, a la hora de escribir un nuevo dato al DAC, se deben asignar a cero ambas señales, respetando los correspondientes tiempos de setup y hold. Con respecto a estos tiempos y la relación con el CLK, con utilizar $1\,\mathrm{T_{CLK}}$ se cumplen todas las restricciones temporales necesarias, por lo que fue el mecanismo usado para respetar el timing del DAC. Debido a las características del

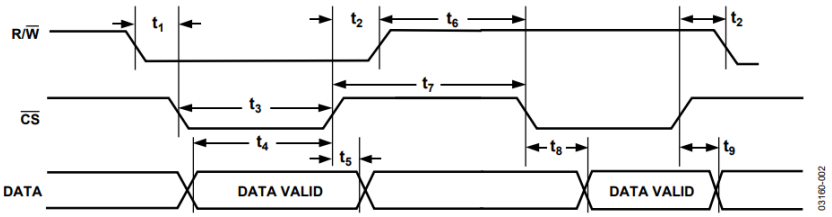

Figura 3.17: Diagrama de tiempos de señales de control del DAC AD5445. $\overline{CS}$ habilita el funcionamiento del DAC y $R/\overline{W}$ fija el funcionamiento en lectura o escritura. Fuente: Hoja de datos del AD5445 [21]

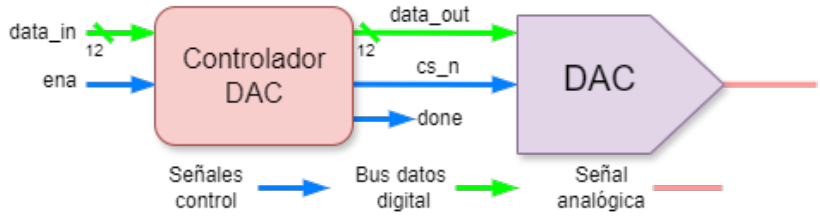

Figura 3.18: Esquema de conexión del controlador del DAC, con sus respectivas señales. **ena** es la habilitación del controlador para su funcionamiento. **done** es un pulso de sincronización con el sistema digital, **cs_n** es la señal de control que habilita el funcionamiento del DAC.

sistema, en el cual no se comparten buses de comunicación, como en el caso del ADC y por simplicidad, se decidió fijar a cero la señal $R/\overline{W}$, dejando solamente la señal $\overline{CS}$ para control del DAC, la cual es la menos restrictiva en términos de timing.

3.11. Controlador del DAC

El controlador del DAC se encarga de manejar las señales de control del conversor de forma de sincronizar el funcionamiento del DAC con el resto del sistema. El bloque posee la interfaz presentada en la figura 3.18. El bus de datos **data_in** ingresa al controlador para generar la conversión de los datos provenientes del template, los cuales se encuentran en complemento a 2, a binario convencional, lo cual es el formato que espera recibir el DAC. Para ajustar la conversión del signo, se agrega un offset en la conversión, donde los 0 V, o el 12'h000 en complemento a 2 pasan a ser la mitad de escala en binario, es decir, 12'h800. La señal **ena** es la habilitación del controlador para su funcionamiento. La señal **done** es un pulso dirigido al controlador de la memoria que contiene el template, con el objetivo de sincronizar los datos que le envía al DAC. De esta forma, cuando se envíe el nuevo dato al DAC desde la memoria, este sea el dato actualizado. Se explicará en detalle en la sección 3.13.1. Finalmente, **cs_n** es la señal de control que habilita el funcionamiento del DAC.

El comportamiento de las señales del controlador se presenta en la simulación de la figura 3.20. El sistema comienza cuando la señal **ena** sube a uno. En este punto,

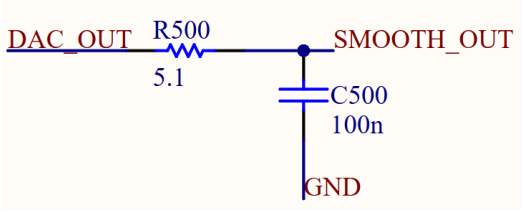

Figura 3.19: Filtro pasabajos implementado para la salida del DAC.

el sistema asume que ya tiene una dirección válida en el bus de datos, por lo que no se envía un pulso de **done** para sincronizar, resultando en que el primer período luego de que **ena** es uno, que es destinado para el pulso de **done**, no actúa ninguna señal. En el segundo período se activa la señal **cs_n**, produciendo la escritura en el DAC, como se observa en su salida **Vout**. Luego, el sistema necesita esperar que el ADC realice el muestreo del nuevo dato, lo cual demora 19 períodos de reloj. Para lidiar con este retardo temporal, el controlador del DAC tiene un contador **delay** que espera $19\,T_{CLK}$ para volver a escribir un dato en el DAC, de forma de poder recibir correctamente un nuevo dato del ADC. Por lo tanto, al cambiar la señal de enable a uno, el DAC en el segundo período asigna cero a la señal **cs_n**, para luego activar el contador y esperar los $19\,T_{CLK}$ para dar nuevamente otro dato al DAC. Finalmente, la señal **done** se activa a nivel alto un período antes que la señal **cs_n**. Como el primer **cs_n** al inicio del funcionamiento del conversor es simplemente para fijar el DAC a un valor conocido, en este caso la señal **done** no es activada. Por este motivo se espera $2\,T_{CLK}$ para iniciar el conteo. Luego del primer pulso de **cs_n**, la señal **done** siempre se activa el período anterior a la generación del pulso **cs_n**. El bus intermedio **data_converted** es la salida del bloque combinatorio que realiza la conversión del tipo de datos a binario convencional. La señal **init** es un registro auxiliar para generar la omisión del primer pulso de la señal **done**.

3.12. Filtro de suavizado del DAC

Este filtro tiene el objetivo de reducir la forma escalonada en la señal que produce el DAC a su salida, es decir, disminuir el ancho de banda de la señal a uno más similar al de la SA real, pero sin comprometer la sincronización de todo el lazo, evitando un retardo excesivo. Por este motivo, se utilizó un filtro pasabajos de arquitectura RC, análogo al filtro anti aliasing, con un ancho de banda de $\approx$300 kHz.

Se utilizó el arreglo RC de la figura 3.19. Considerando el ancho de banda y que es una transferencia de primer orden, se puede estimar el retardo temporal que este filtro introduce:

$$t_d \approx 3\tau = \frac{3}{2\pi fc} = \frac{3}{2\pi 312\,\mathrm{KHz}} \approx 1{,}53\,\mu\mathrm{s}. \tag{3.10}$$

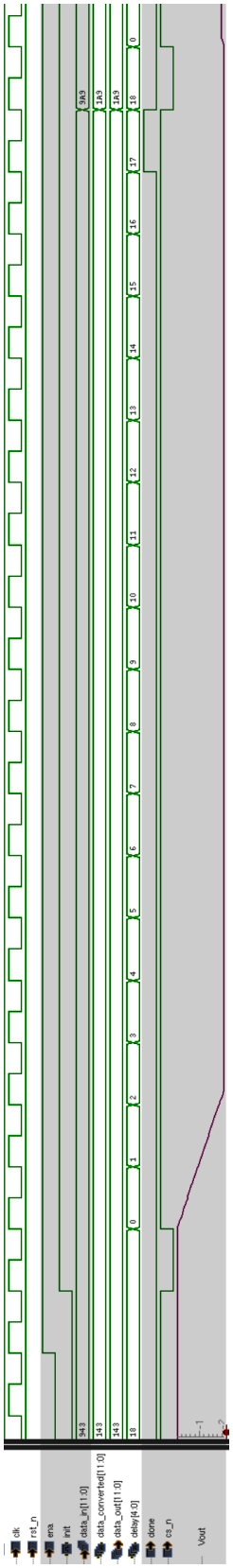

Figura 3.20: Simulación de arranque típico del controlador del DAC.

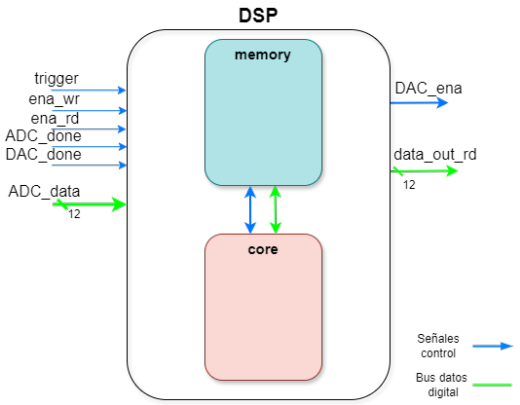

Figura 3.21: Bloque de procesamiento de datos digitales de la primer etapa.

3.13. Módulo de procesamiento de datos digitales *DSP*

Este bloque se encarga del procesamiento digital de las señales de la *Etapa 1*, así como de coordinar el funcionamiento del DAC. El bloque *DSP* se encarga principalmente de recibir los datos del ADC desde el bloque *Control Principal*, procesarlos de forma que genere el template de la señal SA, y enviar el template generado al DAC, a través de su respectivo controlador, el cual el bloque *DSP* se encarga de administrar. Cabe notar que el bloque *DSP* no tiene control sobre el ADC, sino que sólo recibe los datos del mismo. El control del ADC recae sobre la etapa de control principal, descrita en el capítulo 5.

La interfaz del bloque *DSP* con sus puertos se observa en la figura 3.21. El módulo está compuesto principalmente por dos sub bloques; una memoria, conformada por un banco de registros, en donde se guarda el template y un módulo de procesamiento denominado *core*, el cual controla la memoria y genera el template a partir de las nuevas muestras del ADC. Entre sus entradas, posee el bus de datos **ADC_data**, con los datos del ADC provistos por el bloque *Control Principal*. Como entrada de control destinada al sincronismo, recibe un pulso indicando la llegada de una nueva señal SENSADO, denominado **trigger**. Luego recibe señales de control provenientes del *Control Principal*. **ena_rd** y **ena_wr** habilitan tanto la lectura como escritura del template, dependiendo de si este alcanzó la convergencia y si el sistema está procesando la señal dentro de la ventana de probabilidad de ocurrencia de la señal SENSADO. Estas condiciones se explicarán en el capítulo 5. **ADC_done** y **DAC_done** son las señales de sincronismo que proveen los controladores del ADC y DAC. Estas se utilizan para sincronizar la memoria del template. Esta sincronización se explicará en detalle en la sección 3.13.1. Finalmente, como salidas el sistema posee la señal de control **DAC_ena**, la cual habilita el funcionamiento del controlador del DAC y el bus de datos **data_out_rd**, el cual es la salida de datos del template para el DAC.

El bloque *core* está compuesto por dos subbloques, como se describe en la figura 3.22. El módulo *mem_handler* es el núcleo central de la primera etapa. Es el encargado de procesar los datos provenientes del ADC, de controlar el bloque de

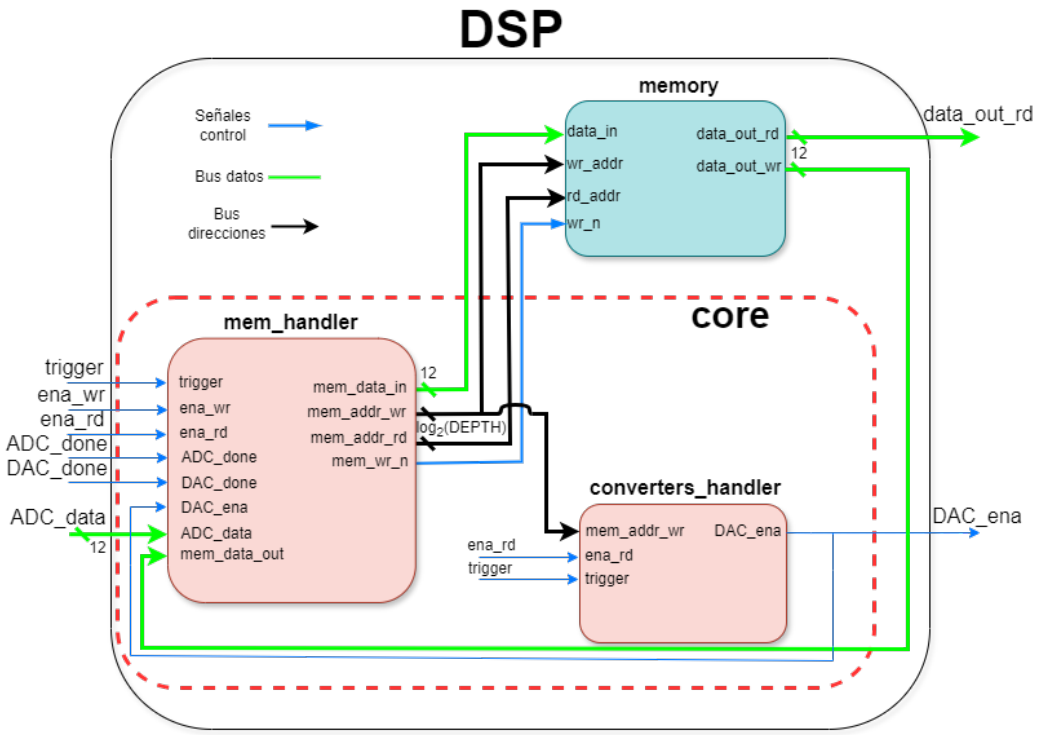

Figura 3.22: Estructura interna del bloque *DSP*.

memoria, además de implementar el algoritmo que genera el template. Finalmente, el módulo *converters_handler* se encarga del manejo del DAC, a través de la señal de control **DAC_ena**.

3.13.1. Módulo *mem_handler*

Este módulo es el núcleo central de procesamiento de datos de esta etapa, dado que se encarga de controlar la memoria y de implementar el algoritmo para generar el template.

La memoria es la encargada de almacenar el template. Inicialmente captura todos los puntos muestreados en un ciclo de la señal SENSADO. Con estos datos, en los siguientes ciclos comienza a generar el template, muestreando nuevamente la señal SENSADO y aplicando el algoritmo, que se explicará más adelante, hasta que el sistema decida que el error obtenido en la resta $template - SENSADO$ es aceptable, deteniendo la iteración y obteniendo el template final. El control del error se explicará en el capítulo 5.

Como se explicó en la sección 3.9.1, el tiempo en que el DAC coloca un nuevo dato y el tiempo en que el ADC captura la respuesta a este valor impuesto por el DAC son disjuntos, debido al retardo del filtro anti aliasing y del filtro de suavizado. Para lidiar con esta problemática, como se describe en la figura 3.22, la memoria posee dos buses de direcciones, **wr_addr** y **rd_addr**, los cuales controlan dos punteros de memoria independientes, uno de lectura y el otro de escritura. Un

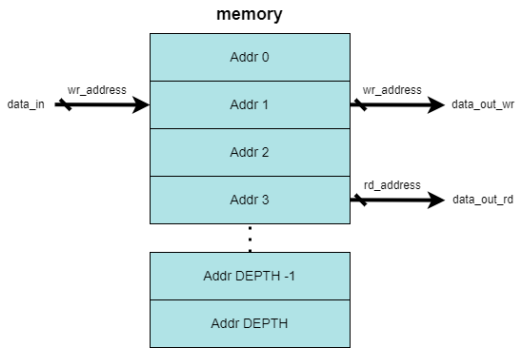

Figura 3.23: Arquitectura de la memoria implementada, compuesta por un banco de registros de FFs, con una profundidad de $DEPTH = 192$ y un ancho de palabra de $N = 12$ bits.

esquema de la memoria se observa en la figura 3.23. El uso de dos buses es necesario debido a que, como se explicó en la sección 3.9.1, a la vez que se recibe un dato proveniente del ADC (escritura), hay que proveerle un dato al DAC (lectura) para que siga realimentando el lazo de control. Para ello, en régimen, el puntero de lectura se encuentra en la dirección de memoria previa que el de escritura.

El funcionamiento básico de este bloque consta de aplicar el algoritmo del template, el cual utiliza los datos entrantes muestreados y el valor actual del template, para luego guardar estos datos procesados en la dirección correspondiente de memoria.

En el primer ciclo de la señal entrante, la memoria está vacía, de esta forma se capturan los datos entrantes intactos y se genera la primera iteración del template. En los siguientes ciclos de señal SENSADO, se generarán diferentes iteraciones del template, donde se guardan en memoria, con el objetivo de reducir el error de copia del template respecto a la señal SA. Este template continúa actualizándose iterativamente hasta lograr su convergencia (se verá en el capítulo 5), manteniéndose constante hasta que sea necesario nuevamente habilitar el lazo de control para actualizarlo y disminuir el error hasta una cota tolerable.

El funcionamiento en detalle de este bloque, donde se profundiza el uso de los buses de direcciones, explicado a través de una simulación, se encuentra en el apéndice B.3.

Algoritmo del Template

El algoritmo implementado por el template se detalla en la figura 3.24, en donde se observa que el nuevo valor a guardar en la memoria **mem_data_in**, para determinada dirección x, es generado por la suma del valor guardado en memoria, para la misma dirección x, **data_out_wr**, con el nuevo valor leído por el ADC **ADC_new_data**. Esta suma lo que produce es, al template actual de memoria, sumarle el dato muestreado por el ADC, el cual es la diferencia de la señal entrante SENSADO con el template actual. Por lo tanto, con el correr de las iteraciones, se espera que la diferencia entre la señal de entrada y el template disminuya, hasta que la diferencia sea lo más cercana posible a cero. Como medida para prevenir errores

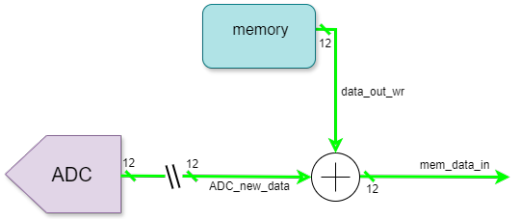

Figura 3.24: Algoritmo generador del template de la etapa 1.

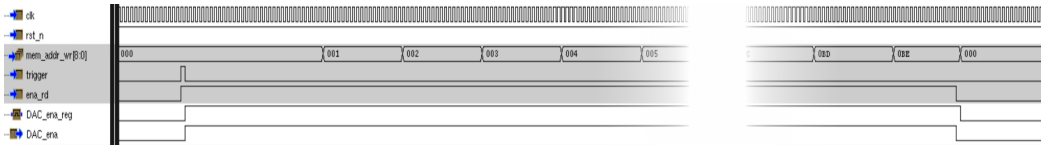

Figura 3.25: Simulación del funcionamiento típico del bloque *converters_handler*. En este caso se activa el DAC con la llegada de un pulso de **trigger**, y se desactiva cuando la señal **ena_rd** baja.

de overflow, luego de efectuar la suma **data_out_wr** + **ADC_new_data**, se chequea que, si ambos sumandos tienen el mismo signo, que es el potencial caso de overflow, el resultado de la suma no tenga signo diferente, lo que indicaría desbordamiento. En caso de que ocurra overflow, la señal homónima **overflow** se activa y se asigna como valor de suma el valor extremo, siendo el máximo si la suma es positiva o el valor mínimo si la suma es de signo negativo.

3.13.2. Módulo *converters_handler*

Este módulo se encarga principalmente de administrar el DAC, el cual sólo es usado por esta etapa. Este módulo provee al controlador del DAC la señal de enable, la cual es activa por nivel alto. La interfaz y las señales utilizadas por el bloque se observan en la figura 3.22 La señal de enable para el DAC, denominada **DAC_ena**, se eleva en el siguiente período de reloj de que se recibe el pulso de **trigger** (salida del bloque detector de señal, ver sección 3.4), mientras **ena_rd** esté activa. Luego, si ocurre otro pulso de **trigger**, la salida **DAC_ena** baja por ese período y vuelve a activarse nuevamente. Esto ocurre cíclicamente hasta que la señal **ena_rd** se inactiva, o se llene la memoria, escribiendo en el último valor de dirección disponible. Una simulación del comportamiento de este módulo se detalla en la figura 3.25.

Capítulo 4

Etapa 2: Cancelación de Residuo y Extracción de ECAP

4.1. Introducción

Este capítulo profundiza el desarrollo de la *Etapa 2*. Esta se encarga de procesar la señal de salida de la *Etapa 1* ($residuo(t) + ecap(t)$) y extraer de ella la ECAP. Para ello, se diseñaron circuitos analógicos para el acondicionamiento de la señal de entrada y un circuito digital que realiza un algoritmo para extraer la señal deseada. A continuación se detalla el diseño implementado.

Como se explicó en el capítulo 3, a la salida de la *Etapa 1* se obtiene, cada T segundos (siendo T el período de estimulación), la señal ECAP superpuesta con el residuo remanente. Además, en la *Etapa 2* el residuo tiene un orden de magnitud similar al de la ECAP, a diferencia de la SA, cuyo orden de magnitud es varias veces mayor que el de la ECAP.

En la figura 4.1 se muestra el diagrama simplificado de la *Etapa 2*, donde su entrada v_{in} es la salida v_{out} de la *Etapa 1*, ilustrada en la figura 3.1. La *Etapa 2* se divide en dos grandes grupos de procesamiento. El primero es un acondicionamiento analógico, compuesto por el amplificador de ganancia variable, para maximizar la excursión de las señales, y el filtro antialiasing, para cumplir con la frecuencia de Nyquist. La arquitectura diseñada para el amplificador corresponde a una configuración no inversora. En cuanto al filtro, se contempla la posibilidad de seleccionar entre una implementación de primer orden o de cuarto orden.

El segundo grupo corresponde al procesamiento digital. Se observa el *Controlador del ADC*, encargado de gestionar el muestreo de datos del ADC. El *Control Principal* administra el funcionamiento de las etapas del sistema CANICs y coordina los datos del ADC para ambas etapas. Estos módulos son compartidos con la *Etapa 1*. Por último, se encuentra el *Mean Subtraction Device* (MSD), siendo este bloque el encargado de implementar un algoritmo para obtener la señal ECAP digitalmente cada T segundos cuando corresponda. Para ello, este bloque genera un template de la señal residuo para luego sustraerlo de la señal entrante.

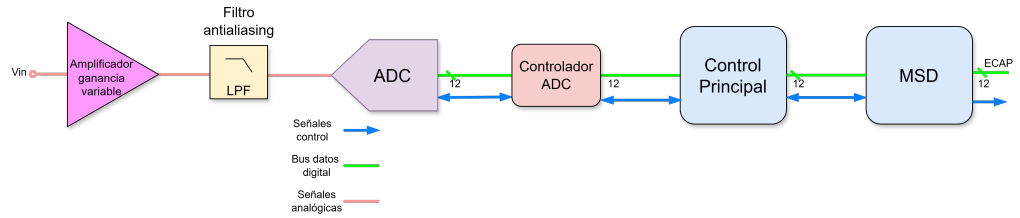

Figura 4.1: Diagrama simplificado de *Etapa 2*.

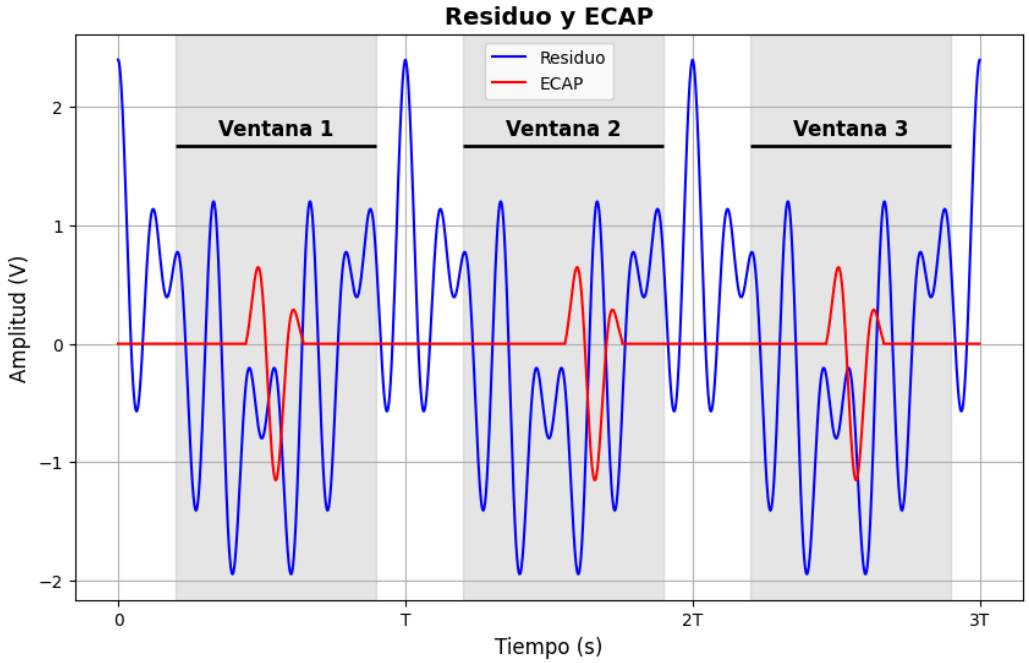

Figura 4.2: Residuo de la operación realizada por la *Etapa 1* junto con la ECAP, luego del acondicionamiento analógico, referenciadas a 2,5 V. Se observa la ventana de interés a ser procesada. La figura es ilustrativa y no representa con exactitud la forma real de las señales.

4.1.1. Principio de funcionamiento

Para profundizar en el principio de funcionamiento de la *Etapa 2*, la figura 4.2 presenta en azul el residuo generado por el error en la cancelación de la SA con su template, en rojo se muestra la señal ECAP. Es importante señalar que la figura es únicamente ilustrativa y que la morfología de las señales no refleja con exactitud su apariencia real. Se observa que el residuo posee un período T y ha sido amplificado de manera analógica para aprovechar al máximo el rango dinámico del ADC.

Recordando que la ECAP tiene una ventana de probabilidad de ocurrencia normal $\mathcal{N}(\mu = 470\,\mu\text{s}, \sigma = 62\,\mu\text{s})$, se resalta la ventana de interés a observar, ya que en esta región es más probable que se encuentre la ECAP (se profundizará respecto a esto en el capítulo 5).

Para obtener la ECAP, asumiendo que la *Etapa 1* convergió, primero se cons-

truye un template basado en el promedio de $ITER$ ventanas, expresado matemáticamente como:

$$TEMPLATE = \frac{1}{ITER} \sum_{i=0}^{ITER-1} (residuo(i) + ECAP(i)) \tag{4.1}$$

siendo $residuo(i)$ y $ECAP(i)$, el valor del residuo y la ECAP respectivamente, en la ventana i-ésima, ilustradas en la figura 4.2.

La hipótesis de periodicidad del residuo establece que $residuo(i) = residuo(i+1) = \cdots = residuo(i+n) = residuo$, para todo i y $n \in \mathbb{N}$. Esta suposición se basa en que la señal $\widetilde{sa}(i)$, generada por el DAC, se aplica de forma periódica con un período fijo T (asumiendo que la *Etapa 1* logró convergencia). Es decir, el template $\widetilde{sa}(i)$ que se utiliza como referencia es una copia fija de un ciclo típico del estímulo, construida bajo el supuesto de que las variaciones entre pulsos reales consecutivos son mínimas o despreciables. Dado que la señal $sa(i)$ entrante se repite cada T segundos, se espera que la diferencia $sa(i) - \widetilde{sa}(i) = residuo(i)$ también lo haga.

Basándose en esta hipótesis, el template se puede reescribir de la siguiente forma:

$$TEMPLATE = residuo + remanente\_ecap \tag{4.2}$$

Donde se define a $remanente\_ecap$ como:

$$remanente\_ecap = \frac{1}{ITER} \sum_{i=0}^{ITER-1} ECAP(i) \tag{4.3}$$

La hipótesis de que la ECAP tiene una ventana de ocurrencia variable (véase sección 2.2.2) implica que la misma no siempre está perfectamente alineada temporalmente en cada ventana.

Como consecuencia, al promediar la ECAP a lo largo de $ITER$ ventanas, esta se atenúa significativamente en amplitud pico a pico con respecto al ECAP original. En otras palabras, el remanente resultante de la ECAP ($remanente\_ecap$) es mucho menor en comparación con la propia ECAP, es decir, $remanente\_ecap \ll ECAP$. En la figura 4.3 se ejemplifica este fenómeno, donde en gris se observa dieciséis posibles ocurrencias de la ECAP, apreciándose su desviación temporal. Mientras que, en azul se ilustra el resultado del promedio de las señales, es decir, $remanente\_ecap$. Se destaca la atenuación del promedio, respecto a la ECAP original.

Una vez generado el template, para lograr extraer la ECAP, se realiza la resta entre las siguientes j-ésimas ventanas de $residuo(j) + ECAP(j)$ y el template. Siendo $j \geq ITER$. Es decir la $ECAP\_EXTRAIDA$ se puede expresar como:

$$ECAP\_EXTRAIDA(j) = residuo(j) + ECAP(j) - TEMPLATE \tag{4.4}$$

Sustituyendo 4.2 en 4.4, utilizando nuevamente que el residuo es periódico y que $remanente\_ecap \ll ECAP$ se obtiene que la ECAP extraída es aproximadamente la ECAP original. Es decir:

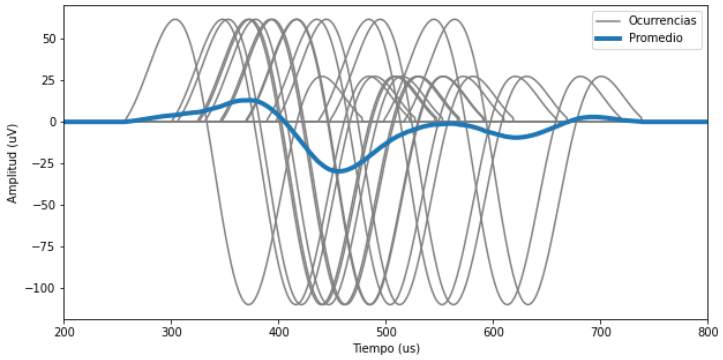

Figura 4.3: Resultado de realizar el promedio de 16 ECAPs, debido a la ventana de ocurrencia variable. En gris se observa las 16 posibles ocurrencias de las ECAPs, mientras que en azul se observa el resultado de su promedio, $remanente\_ecap$, destacándose la atenuación del promedio.

$$\begin{aligned} ECAP\_EXTRAIDA(j) &= ECAP(j) - remanente\_ecap \\ &\approx ECAP(j) \end{aligned} \tag{4.5}$$

Además, como se espera que el $residuo > ECAP$, entonces por transitiva, se concluye que $remanente\_ecap \ll residuo$, reduciendo la ecuación 4.2, obteniendo que el template del residuo es $TEMPLATE \approx residuo$.

En la figura 4.4 se observa el resultado de la ecuación 4.5. En azul se observa la ECAP original, y en naranja la ECAP extraída. Se destaca el parentesco de la ECAP extraída con la ECAP original, representada también en la figura 4.2. La diferencia en morfología respecto a ambas señales radica en el término $remanente\_ecap$ de la ecuación, aportando un sumando no nulo a la ECAP original. En esta oportunidad se utilizó ITER = dieciséis, en la sección 4.1.1 se realiza el estudio para su elección.

De haberse muestreado parcialmente la ECAP durante la generación del template de la *Etapa 1*, estas muestras se pueden agrupar como parte del residuo, pudiéndose escribir como $residuo = \widetilde{residuo} + \widetilde{ECAP}$, donde $\widetilde{residuo}$ representa el error de la cancelación de la SA propiamente dicha, mientras que $\widetilde{ECAP}$ representa haber muestreado algunos mV de la ECAP. La característica fundamental que hace que muestrear la ECAP no sea un problema es que $\widetilde{ECAP}$ es periódica de período T (debido a que está fijada por el template de la *Etapa 1*), a diferencia de la ECAP que tiene ventana de probabilidad de ocurrencia, generando que el algoritmo cancele el residuo, por ende cancelando $\widetilde{ECAP}$.

Elección de las iteraciones

La efectividad de la relación de $residuo\_ecap \ll ECAP$ depende de los parámetros probabilísticos de la normal, especificados en el capítulo 2, y de la cantidad de iteraciones ($ITER$) al realizar el template.

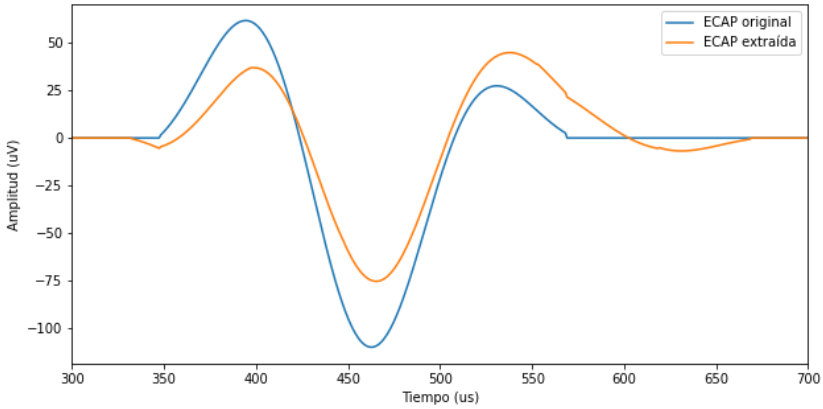

Figura 4.4: Se observa la ECAP extraída en naranja comparada con la ECAP original para un instante de tiempo arbitrario. La señal naranja es el resultado de aplicar la ecuación 4.5 al $residuo + ECAP$ de la figura 4.2. Se destaca el parentesco en morfología respecto a la original. La figura es ilustrativa y no representa con exactitud la forma real de las señales.

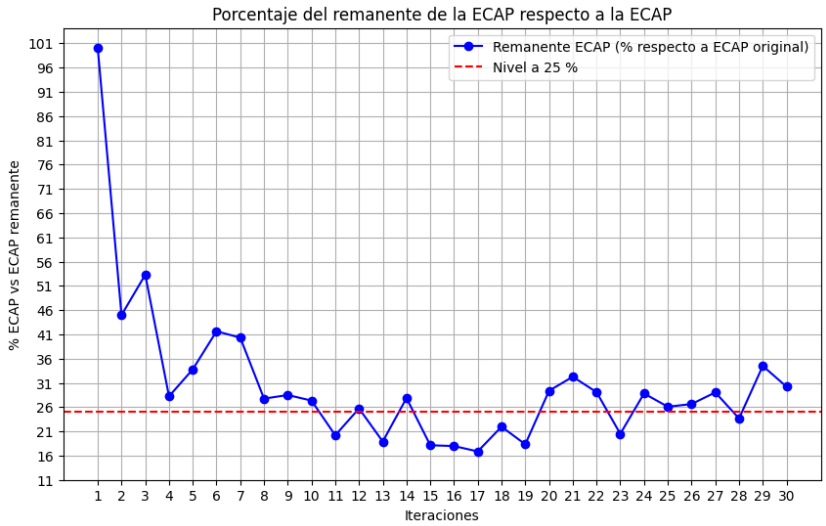

Figura 4.5: Porcentaje de la amplitud pico a pico de la ECAP versus la amplitud pico a pico del remanente de la ECAP ($100 \times \frac{remanente\_ecap_{pp}}{ECAP_{pp}}$ %) en función de las iteraciones ($ITER$). Se optó por $ITER =$dieciséis.

Para cuantificar la cantidad de iteraciones necesarias, en la figura 4.5 se muestra el porcentaje de amplitud pico a pico de la ECAP y del $remanente\_ecap$ (es decir, $100 \times \frac{remanente\_ecap_{pp}}{ECAP_{pp}}$ %) en función de la cantidad de promedios $ITER$.

Se destaca el comportamiento asintótico observado, lo que indica que no necesariamente un mayor número de iteraciones reduce de manera significativa el remanente de la señal.

Teniendo en cuenta que las ecuaciones presentadas anteriormente se implementarán en un sistema digital, se observa que si $ITER$ es una potencia de 2, la división se puede realizar simplemente mediante un desplazamiento lógico de bits. Además, a medida que aumentan las iteraciones, también aumenta el tiempo necesario para calcular el template, siendo este tiempo directamente proporcional al período de estimulación T.

Por lo tanto, se tiene un compromiso entre la cantidad de iteraciones y el tiempo de convergencia del template.

En función de la gráfica, se decidió utilizar $ITER =$ dieciséis, ya que este valor representa un punto óptimo entre la velocidad de convergencia y la capacidad de cómputo necesaria para realizar la división. Para contar con un margen de seguridad ante posibles variaciones entre realizaciones, se descartó la opción de $ITER =$ ocho, a pesar de su aparente buen desempeño.

4.2. Amplificador de entrada

Para maximizar el aprovechamiento del rango del ADC, es necesario amplificar las señales en la entrada de la *Etapa 2*. Esto se debe a que la señal $residuo+ECAP$ tiene una magnitud considerablemente menor que el valor del fondo de escala del ADC.

La arquitectura seleccionada para realizar la amplificación con ganancia variable ilustrada en la figura 4.1 es un amplificador no inversor, tal como se aprecia en la figura 4.6. Se observa que la impedancia conectada entre el nodo 2 y tierra está compuesta por la serie de un condensador y un potenciómetro.

La utilización de un potenciómetro se justifica por la incertidumbre en la amplitud exacta de las señales a procesar (ECAP y $residuo$). Si bien se dispone de estimaciones sobre el rango típico de amplitudes en el que se encuentra la ECAP (como se especificó en el capítulo 2), su valor exacto no es conocido a priori y puede variar considerablemente en función de factores fisiológicos, parámetros de estimulación y configuraciones específicas del electrodo. Lo mismo aplica al $residuo$, cuya magnitud depende de la respuesta de la *Etapa 1* en la cancelación. Fijar una ganancia estática demasiado baja podría limitar la resolución efectiva, mientras que una demasiado alta podría conducir a saturación o distorsión. El valor que toma el mismo será configurado durante la validación del sistema al utilizar la FPGA.

La señal $residuo+ECAP$ viene sumada de la *Etapa 1* a un nivel de continua que vale $\frac{VREF}{2} = 2{,}5\,\text{V}$, requerido para el correcto funcionamiento del ADC, explicado en la sección 3.8.3. Para mantener este nivel de continua en la entrada del ADC, es necesario que, durante esta etapa de amplificación, solo se amplifiquen las señales

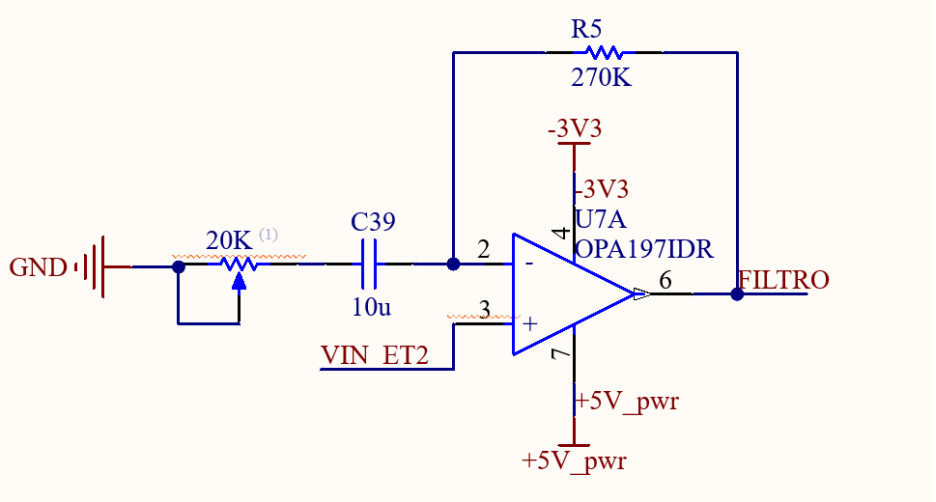

Figura 4.6: Amplificador no inversor con ganancia variable de la entrada de la *Etapa 2*.

de interés y no la componente continua.

Para seguir manteniendo este nivel de continua, se utiliza el condensador en serie con el potenciómetro, generando que, para la señal de continua, el condensador se comporte como un circuito abierto. De este modo, el amplificador opera en configuración de seguidor, copiando el nivel de continua de interés. En cambio, dentro de la banda de interés de la ECAP, el condensador se comporta como un cortocircuito, permitiendo obtener una configuración no inversora puramente resistiva con ganancia variable.

$$H(s) = \left(1 + \frac{R_5}{R_{POT}}\right)\left(\frac{\frac{1}{(R_5+R_{POT})C} + s}{\frac{1}{R_{POT}C} + s}\right) \tag{4.6}$$

La ecuación 4.6 representa la transferencia del amplificador no inversor de la figura 4.6, donde R_{POT} es el potenciómetro y C el condensador $C39$. Los valores elegidos de resistencia y condensador fueron tales que las frecuencias asociadas al polo y al cero, dadas por $f_p = \frac{1}{2\pi R_{POT}C}$ y $f_z = \frac{1}{2\pi(R_5+R_{POT})C}$ respectivamente, se encuentren al menos una década por debajo de la mínima frecuencia de la ECAP, especificada en la sección 2.2.2. Además, se espera que la ganancia en banda pasante $G = 1 + \frac{R_5}{R_{POT}}$ sea ajustable en un rango suficientemente amplio como para adaptarse a las variaciones en la amplitud de $residuo + ECAP$. En base a estos requisitos, se obtiene que la ganancia se debe encontrar en el siguiente rango de valores $G \in [14{,}5; 510]\frac{V}{V}$.

4.3. Filtro Anti Aliasing

El principal objetivo de este filtro es limitar el ancho de banda de la señal de entrada al ADC, previniendo el aliasing. Además, dado que constituye la última

etapa de filtrado analógico, es necesario eliminar la mayor cantidad posible de señal fuera del ancho de banda de la señal de interés ECAP para aumentar su SNR.

Se optó por dos arquitecturas de filtro pasabajos. Un filtro de tipo Butterworth pasabajos de cuarto orden con el objetivo de lograr una transición abrupta de 80 dB por década. Y una arquitectura de filtro RC con una transición de 20 dB por década. La selección entre una u otra topología se realiza mediante un jumper en la PCB. Por defecto, se utiliza el filtro de cuarto orden, ya que presenta una caída más pronunciada fuera de la banda de interés, lo que permite una mejor supresión del contenido espectral no deseado.

El motivo de incorporar dos filtros distintos es brindar flexibilidad durante la validación, la puesta a punto y el debug de la *Etapa 2*. La posibilidad de seleccionar entre ambos mediante un jumper en la PCB facilita la evaluación comparativa, permite adaptarse a distintas condiciones de operación y aporta una opción de respaldo en caso de fallos. A continuación se detalla el diseño de cada filtro.

4.3.1. Filtro Butterworth

De los posibles filtros de cuarto orden, se optó por esta arquitectura dado que su característica es maximizar la transferencia plana en la banda de interés [22], logrando así que no se vea afectada la amplitud de la ECAP extraída. La implementación consta de un filtro activo con arquitectura Sallen-Key como se ve en la figura 4.7.

El ADC posee una resolución de 12 bits, por lo que, para evitar aliasing, como criterio de diseño se eligió que la atenuación en la banda de rechazo[1] debe ser al menos de 75 dB. Esto genera que cualquier señal por encima de la frecuencia de Nyquist tenga una amplitud menor al ruido de cuantización del ADC, maximizando la precisión del mismo (véase apéndice A.1).

Dado que la frecuencia de Nyquist es 118 KHz con un filtro de orden 4 (-80 dB/dec) con una frecuencia de corte menor de 11 KHz (pero mayor que $BW_{ECAP} = 5$ KHz) se cumple que la atenuación a partir de la frecuencia de Nyquist alcanza la atenuación de 75 dB deseada.

Si se quiere utilizar un filtro de menor orden, la restricción se cumple si la frecuencia de corte vale 5 KHz, distorsionando la amplitud de la señal ECAP. Mientras que un orden mayor aumenta la complejidad. Por esto, se elige un filtro de cuarto orden.

Los valores se eligen tal que en el polo la ganancia genere una atenuación en la banda de interés mínima, y para que el retardo de grupo impuesto por el filtro sea cuasi constante en dicha banda. En particular, el filtro construido tiene un retardo de grupo de 8,8 μs (véase apéndice A.1) esto no es de relevancia dado que el retardo particular de la señal ECAP al ser muestreada no influye en que el sistema pueda adquirirla, ya que tiene una duración de 220 μs, mucho mayor que el retardo.

Tomando en cuenta las consideraciones de diseño, se implementa un filtro con respuesta Butterworth de cuarto orden con una frecuencia de corte de 9 KHz, im-

[1] Se define la banda de rechazo como el conjunto de frecuencias mayores a la frecuencia de Nyquist.

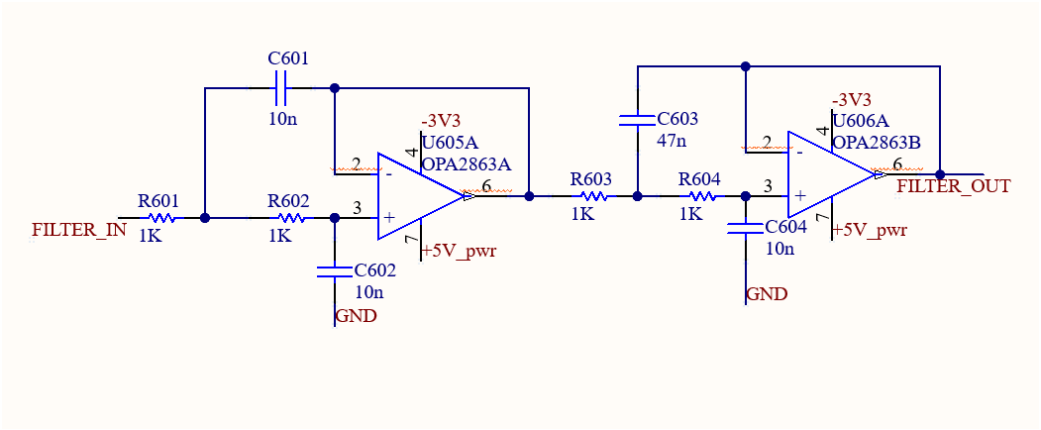

Figura 4.7: Circuito implementado del filtro anti aliasing con respuesta Butterworth.

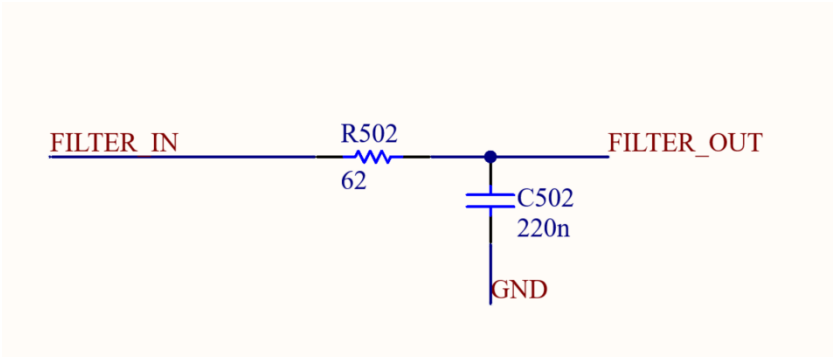

Figura 4.8: Circuito implementado del filtro anti aliasing con arquitectura RC.

plementado con una arquitectura Sallen-Key, como se observa en la figura 4.7. La atenuación medida a los 5 KHz fue de 0,96 V/V. Para el diseño se utilizaron tablas extraídas de [22] las cuales especificaban valores para el diseño previsto con una frecuencia de corte de 1 Hz. Además, se creó un script de Matlab capaz de modificar la frecuencia de corte a la deseada y proveer valores de resistencias y condensadores disponibles en la industria.

Luego de definir el diseño, el mismo se simuló utilizando la herramienta Cadence y al construirlo se le realizó un estudio en frecuencia para validar su comportamiento.

4.3.2. Arquitectura RC

En la figura 4.8 se observa el diagrama del circuito RC implementado con sus correspondientes valores. Se diseñó la frecuencia de corte levemente por encima del ancho de banda de la ECAP para considerar la caída de 3 dB de dicho filtro. Por lo tanto, se tiene que la frecuencia de corte vale $f_c \approx 12\,\text{kHz}$.

4.4. Módulo de procesamiento de datos digitales MSD

El sistema digital es el encargado de procesar periódicamente, cada T segundos (donde T se corresponde con el período de estimulación), la señal del residuo junto con la ECAP. Como se explicó anteriormente, el *Control Principal* y el *Controlador del ADC* son compartidos con la *Etapa 1*, mientras que el *MSD* es el encargado de implementar el algoritmo detallado con las ecuaciones.

En la figura 4.9 se observa una descripción en diagrama de estados del algoritmo que implementa este sistema, compuesto por tres fases.

En la primera fase, el template que se utilizará se inicializa en cero. Una vez que la *Etapa 1* finaliza el cálculo de su propio template (referenciando a que la *Etapa 1* finalizó), se pasa a la segunda fase (el pasaje entre la fase 1 y 2 es operado por el *Control Principal*, explicado en el capítulo 5).

En la segunda fase se realiza el cálculo del template del residuo (cálculo de la ecuación 4.1). Al finalizar las iteraciones del template deseadas (es decir, se promediaron las $ITER$ ventanas), se finaliza su cálculo dejándose fijo y se pasa a la tercera fase.

En la tercera fase, se utiliza el template del residuo para cancelarlo con las siguientes entradas de $residuo + ECAP$ (ecuación 4.4), obteniendo digitalmente la ECAP cada T segundos. Si el *Control Principal* (ver capítulo 5) detecta que la señal de entrada $residuo + ECAP$ está por fuera de los márgenes de excursión permitidos, reiniciará el template, lo que equivale a regresar a la primera fase, donde se espera que la *Etapa 1* converja nuevamente.

En la figura 4.10 se presenta la implementación modular del sistema digital. Las señales de control a la entrada **s2_trg**, **s2_clear** y **s2_enable** son provenientes del *Control principal*, utilizadas para habilitar el funcionamiento del MSD (**s2_enable**), inicializar vacío el template (**s2_clear**) y dar aviso del inicio de $residuo + ECAP$ (**s2_trg**). Las señales **ADC_data** y **ADC_done** también son generadas por el *Control Principal*. **ADC_data** contiene el valor muestreado por el ADC, mientras que **ADC_done** actúa como indicador de validez, notificando que el dato presente en **ADC_data** está disponible para ser procesado.

El módulo MSD se compone de *template_handler*, responsable de operar el módulo *template* para generar un template de la señal del residuo. Luego *digital_filter* implementa un filtro digital para filtrar la señal proveniente del bloque que lo precede. Se obtiene a su salida la ECAP extraída, junto con señales de control para indicar dónde se encuentra la ventana de interés. Por último, el módulo *stop_average* se encarga de detener la generación del template cuando sea necesario y dar aviso al filtro digital de que a su entrada se tiene la señal válida para ser filtrada.

En la figura 4.11 se muestra el comportamiento temporal transitorio del MSD. Esto es, desde que el bloque inicia su procesamiento hasta que converge. La señal **s2_trg** es un pulso periódico de duración $1\,\mathrm{T_{CLK}}$ que indica el inicio de la señal de entrada, haciendo que el bloque procese, cada T segundos, las señales muestreadas $(residuo(i) + ECAP(i))$. Hasta no finalizar las iteraciones, la señal **template** es nula, al igual que las salidas del sistema **ecap** , **ecap_available** y

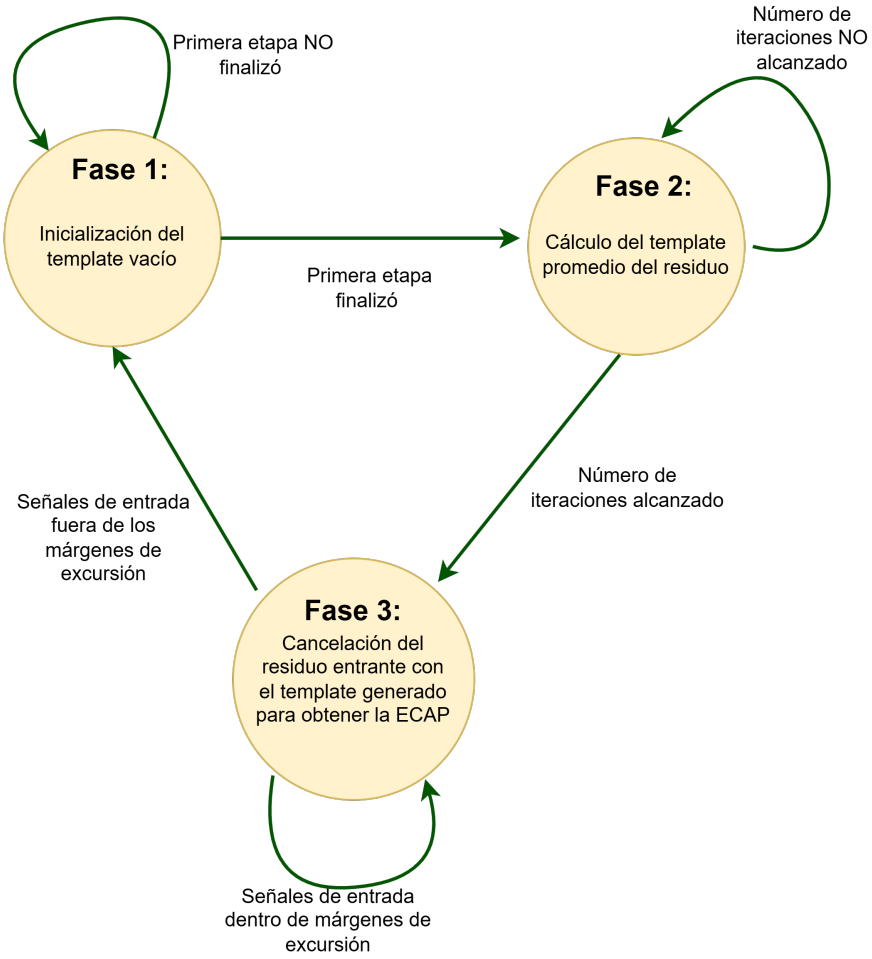

Figura 4.9: Diagrama de estados del sistema digital de la *Etapa 2*. Se distinguen tres fases principales: inicialización del template vacío, cálculo del template promedio del residuo, y cancelación del residuo entrante con el template. El *Control Principal* tiene la potestad de llevar al sistema digital a la fase 1.

ecap_sample_done.

Después de transcurrido un tiempo de $T \times ITER$, se obtiene internamente el template del residuo. A partir de este momento, la señal **ecap** comienza a proporcionar la ECAP extraída. En paralelo, se activa la señal **ecap_available**, delimitando la ventana temporal de interés, y **ecap_sample_done** genera pulsos cada 19 T_{CLK} para indicar la disponibilidad de nuevas muestras. A continuación, se describe en detalle el comportamiento temporal de estas señales.

En la figura 4.12 se muestra el comportamiento temporal de las señales de salida del MSD (que, a su vez, corresponden a las salidas del sistema CANICs) una vez alcanzado el régimen estacionario, es decir, luego de que la *Etapa 2* ha convergido y generado el template del residuo.

La señal **ecap(i)[n]** representa la muestra n-ésima de la i-ésima señal **ecap**, con

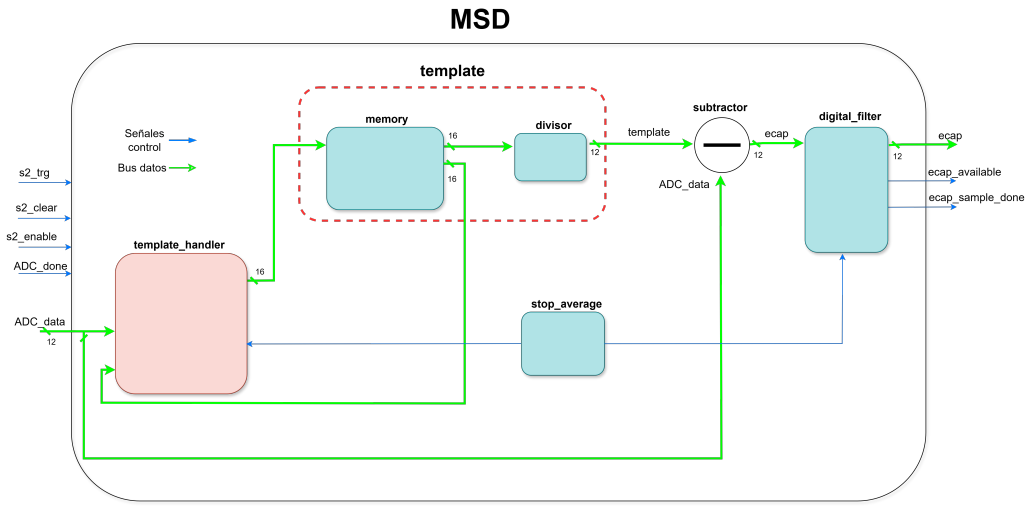

Figura 4.10: Diagrama de bloques del MSD. Las señales de entrada **s2_trg**, **s2_clear** y **s2_enable** son señales de control provenientes del *Control principal del sistema*. Las señales **ADC_data** y **ADC_done** provienen del *Control Principal*. La señal de salida **ecap** corresponde a la ECAP extraída del sistema, mientras que las señales de control **ecap_available** y **ecap_sample_done** indican cuándo la ECAP está disponible. Las señales de control de los submodulos se omitieron para simplicidad.

$i > ITER$, es decir, en régimen. Para indicar la validez temporal de la ventana de donde se encuentra la señal, se emplea **ecap_available**, activándose únicamente durante la duración de la ventana (ilustrada en la figura 4.2). A su vez, el período de **ecap_available** corresponde con el período de estimulación T.

Dado que la señal **ecap** se obtiene como resultado de la resta entre el **template** y los datos provenientes del ADC (**ecap** = **ADC_data** − **template**, según la figura 4.10), la cadencia de muestras de **ecap** queda determinada por la cadencia de datos de adquisición del ADC. En consecuencia, los datos de **ecap** están disponibles cada 19 T_{CLK}, al igual que los del ADC, tal como se detalló en la sección 3.9 del capítulo 3. Para indicar la disponibilidad de una nueva muestra de la señal **ecap**, se emplea la señal **ecap_sample_done**, la cual genera pulsos de duración T_{CLK} cada 19 T_{CLK}. Esta señal permite sincronizar el procesamiento de datos subsiguiente, asegurando que cada muestra de **ecap** sea correctamente identificada y tratada en su instante correspondiente.

4.4.1. Módulo template

Este módulo está compuesto por la memoria de la figura 4.13, siendo otra instancia de la misma memoria la *Etapa 1*, y el módulo *divisor*.

La memoria se encarga de almacenar la suma de $ITER$ ventanas temporales de la señal de entrada. Es decir, la memoria es la encargada de almacenar la siguiente suma acumulada:

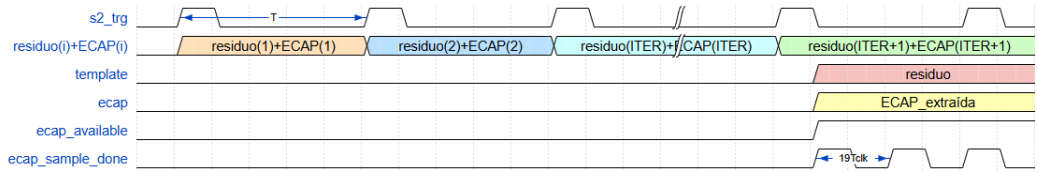

Figura 4.11: Comportamiento temporal transitorio del bloque MSD. Se observa que después de $T \times ITER$ segundos, se obtiene el $residuo$ en la señal interna **template**. Además en este instante se observa que a la salida **ecap** se tiene la $ECAP\_extraida$ junto con sus señales de control. Para simplificar, se ha omitido la señal de reloj y se asume que el bloque está habilitado por el *Control Principal*.

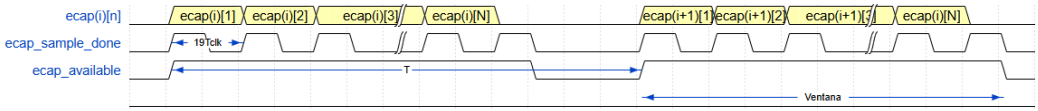

Figura 4.12: Comportamiento temporal en régimen estacionario del bloque MSD. Se observa la señal **ecap(i)[n]** representando la muestra n-ésima de la i-ésima señal *ecap*, para $i > ITER$. La señal **ecap_available** indica la validez temporal de la ventana de la señal, activándose únicamente durante su duración. La señal **ecap_sample_done** genera pulsos de duración T_{CLK} cada 19 T_{CLK} para señalar la disponibilidad de nuevas muestras, sincronizando el procesamiento posterior. Para simplificar, se omite la señal de reloj y se asume que el bloque está habilitado por el *Control Principal*.

$$\sum_{i=0}^{\mathrm{ITER}-1} (\mathrm{residuo(i)} + \mathrm{ECAP(i)}) \tag{4.7}$$

Dado que la memoria almacena la suma de $ITER = 16$ ventanas, cada una representada como una palabra de 12 bits en complemento a dos, se requiere un ancho de palabra de 16 bits para representar correctamente la suma acumulada[2], a diferencia de los 12 bits utilizados en la memoria de la *Etapa 1*.

El bloque aritmético *divisor* se encarga de realizar, muestra a muestra, la operación de división sobre la suma acumulada en memoria. Esta operación es necesaria para cumplir con el objetivo del módulo *template*, que es generar un template del $residuo$ a partir del promedio de $ITER$ repeticiones de la señal de entrada. Dado que $ITER = 16$, la división se implementa de forma eficiente utilizando lógica de desplazamiento de bits. En lugar de realizar una división aritmética explícita, el bloque *divisor* descarta los 4 ($log_2(16)$) bits menos significativos (equivalente a dividir por $2^4 = 16$) de cada muestra acumulada. Posteriormente, para mantener el formato fijo de 12 bits en la salida, el valor truncado se extiende repitiendo su bit más significativo (extensión de signo, por estar trabajando en complemento a dos) hasta completar los 12 bits requeridos. El resultado final es el valor promedio

[2]En complemento a dos, una palabra de 12 bits puede representar valores entre $-2^{11} = -2048$ y $2^{11} - 1 = 2047$. Al sumar $ITER = 16$ de estas palabras, el valor oscila entre $16 \times (-2048) = -32768$ y $16 \times 2047 = 32752$. Estos valores se encuentran dentro del rango representable por una palabra con signo de 16 bits: $[-2^{15}, 2^{15} - 1] = [-32768, 32767]$.

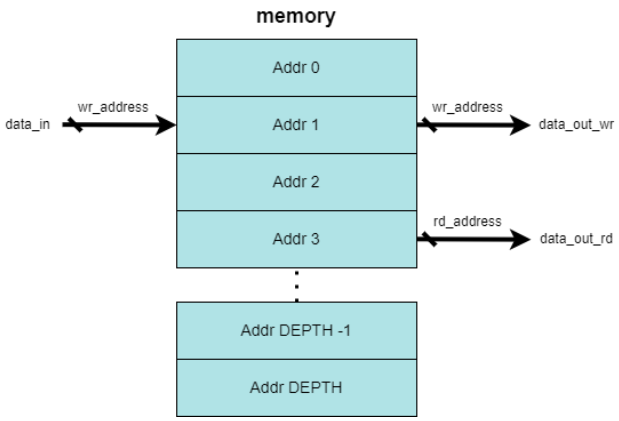

Figura 4.13: Arquitectura de la memoria implementada en la *Etapa 2*, compuesta por un banco de registros realizado FFs, con una profundidad de $DEPTH = 192$ y un ancho de palabra de $N = 16$ bits.

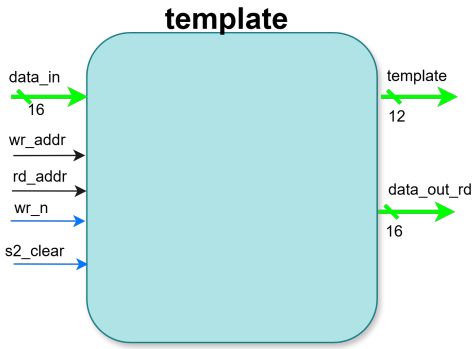

Figura 4.14: Interfaz del bloque *template*.

de cada muestra, calculado sobre $ITER = 16$ repeticiones, y constituye la salida del bloque *divisor*, es decir, el $TEMPLATE$ descrito en la ecuación 4.1.

En la figura 4.14 se muestra la interfaz de este bloque. La señal **data_in** corresponde al dato de entrada proveniente del bloque *template_handler*. La señal **s2_clear** se emplea para resetear el template, mientras que **wr_addr** y **rd_addr** son señales de direccionamiento de datos y **wr_n** señal para escribir la memoria. Por último, **template** representa el template obtenido tras $ITER$ iteraciones, y **data_out_rd** es el dato leído de la memoria interna en la dirección **rd_addr**.

4.4.2. Módulo template_handler

El funcionamiento de este módulo es análogo al de *mem_handler* descrito en la sección 3.13. Es decir, dicho módulo se encarga de recibir los datos muestreados provenientes del ADC y gestionar el módulo *template* para obtener el template deseado. Para ello, lee iterativamente los datos almacenados en la memoria y los actualiza sumando aritméticamente los nuevos datos muestreados con los previa-

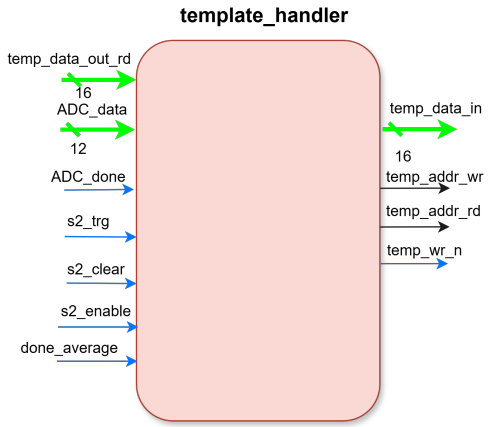

Figura 4.15: Interfaz del bloque template_handler.

Figura 4.16: Interfaz del bloque stop_average.

mente almacenados. En la figura 4.15 se muestra la interfaz de este bloque. La señal **s2_enable** habilita el funcionamiento del bloque. Por su parte **s2_trg** dispara su funcionamiento. Luego, **done_average** detiene la escritura en el template cuando corresponde. El resto de las señales se describieron en la sección previa.

4.4.3. stop_average

Este módulo se encarga de controlar la finalización del promediado. En la figura 4.15 se muestra la interfaz de este bloque. Cuando detecta que se han sumado $ITER$ ventanas (contando la aparición de la señal **s2_trg**, que indica el inicio de $residuo(i)$), activa su señal **done_average** en nivel alto para notificar a los bloques correspondientes.

4.4.4. Módulo subtractor

Bloque aritmético que realiza la resta entre la señal de entrada y el $TEMPLATE$. Es decir, es el encargado de ejecutar la ecuación 4.4, obteniendo a su salida la ECAP extraída sin filtrar (ecuación 4.5).

La figura 4.17 muestra la interfaz del bloque, el cual corresponde a un circuito combinacional que realiza la operación de resta con signo entre sus entradas, es decir, $data\_out = data\_in\_positive - data\_in\_negative$.

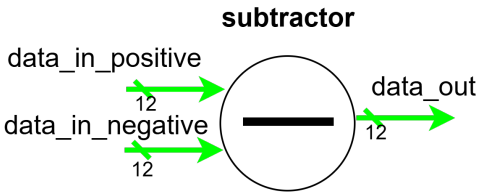

Figura 4.17: Interfaz del bloque subtractor

4.4.5. Módulo digital_filter

Con el objetivo de reducir la potencia del ruido fuera del ancho de banda de la señal ECAP, se implementó un filtro digital. Al eliminar componentes espectrales no deseadas, el espectro de la señal obtenida se asemeja más al de la ECAP original. Como consecuencia, la forma temporal resultante es más fiel a la señal original. Para el diseño del filtro, se eligió una arquitectura FIR, la cual no distorsiona la fase, basada en el método de ventanas. En particular, se utilizó la ventana de Blackman, debido a su buena atenuación de lóbulos laterales.

Al determinar el orden del filtro, se consideraron dos aspectos principales: el retardo en muestras que introduce (en filtros FIR, este retardo es aproximadamente la mitad del orden del filtro) y el área de hardware que ocupa dentro del sistema digital. En base a estos criterios, se seleccionó un filtro de orden 50 con una frecuencia de corte de $f_c = 7\,\text{kHz}$.

En la figura 4.18 se muestra la respuesta en frecuencia del filtro digital diseñado. En ella se destaca visualmente la porción del espectro que es efectivamente filtrada, en relación con el ancho total del espectro disponible, limitado por la frecuencia de Nyquist.

Este diseño introduce un retardo de grupo de 25 muestras, lo que equivale a un retardo temporal de $\tau_{\text{filter}} = 25 \times 19\,T_{\text{CLK}} \approx 0{,}1\,\text{ms}$, ya que se obtiene una muestra cada $19\,\text{T}_{\text{CLK}}$, como se mostró en la figura 4.11. Este retardo temporal es despreciable en comparación con los dieciséis períodos que le requiere la *Etapa 2* para generar su *template*, cuyo retardo es $\tau_{\text{E2}} = 16 \times T \approx 18\,\text{ms}$.

Este filtro requiere del orden de 50 unidades de memoria[3], siendo relativamente despreciable en comparación con la memoria utilizada para almacenar el template de la *Etapa 1* y *Etapa 2*, valiendo 448 unidades de memoria (véase apéndice B.1).

En la figura 4.19 se muestra el módulo *digital_filter* con su interfaz, compuesto por el filtro previamente mencionado (módulo *filter_50*), diseñado utilizando la herramienta *Filter Design* de MATLAB y luego exportado a nivel RTL. Además, incluye un controlador de las señales (módulo *signal_control*), encargado de operar las señales de control (**ecap_available** y **ecap_sample_done**) cuando a la salida del filtro se obtenga una ECAP disponible. Se omitieron de la interfaz las señales **s2_trg**, **s2_clear** y **s2_enable** para simplicidad.

En la figura 4.20 se observa el comportamiento temporal de este módulo, acti-

[3]Se define una unidad de memoria como un registro con un ancho de 12 bits.

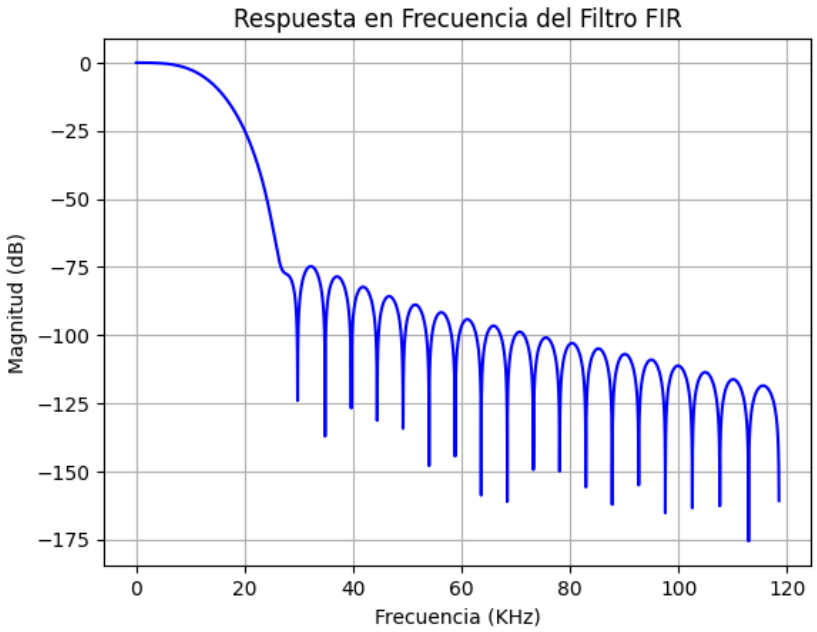

Figura 4.18: Respuesta en frecuencia del filtro pasa-bajos Blackman FIR, de orden 50 con frecuencia de corte de 7 kHz.

vándose su funcionamiento cuando el módulo *stop_ average* active su señal por nivel **done_average**. A partir de ese instante temporal, comienzan a ingresar muestras de la señal **ecap[n]** en su entrada, con una cadencia por muestra de 19 T_{CLK}, marcada por la señal **ADC_done**. Producto del retardo de grupo en muestras que introduce *filter_ 50*, se obtienen las muestras de la señal filtrada (**ecap_filtered[n]**) a la salida del sistema CANICs después de 25 muestras de la señal **ecap**.

Cuando el filtro comienza a entregar muestras, el módulo *signal_ control* se encarga de activar la señal **ecap_available** para indicar la disponibilidad de datos, y utiliza la señal **ecap_sample_done** para notificar que una muestra está disponible. El comportamiento temporal de las señales considera el retardo de grupo introducido por el filtro. La señal **ecap_available** corresponde a la señal **done_average** desplazada en el tiempo 25 muestras. Este retardo se implementó mediante un registro de desplazamiento (shift register). Por otro lado, la señal **ecap_sample_done** se genera a partir de la señal **ADC_done** y la salida del shift register antes mencionado.

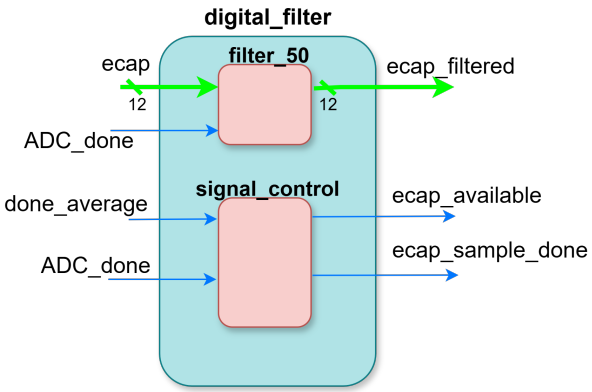

Figura 4.19: Interfaz del bloque digital_filter.

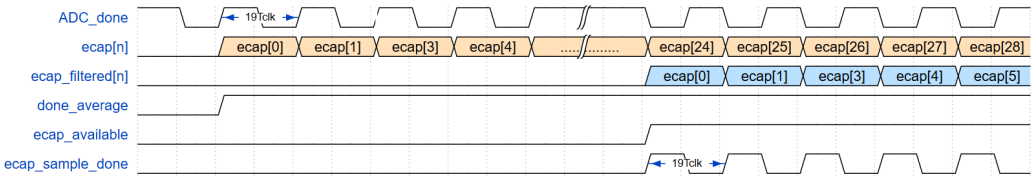

Figura 4.20: Comportamiento temporal del bloque *digital_filter*. Se ilustra el retardo de 25 muestras que introduce el *filter_50* a la señal **ecap[n]**. A su vez, se aprecia el comportamiento temporal de las señales de control **ecap_available** y **ecap_sample_done**). Se asume que el módulo está habilitado.

Capítulo 5

Control Principal del Sistema

5.1. Introducción

En este capítulo se introduce el módulo que administra el funcionamiento de las dos etapas del sistema y gestiona el funcionamiento del ADC. Este módulo está compuesto principalmente por una máquina de estados, basando su control a partir de dos monitores de error de las etapas y del estado en que se encuentren las mismas. Además, es el encargado de direccionar los datos del ADC, dado que por criterio de diseño del sistema se cuenta con sólo uno, y ambas etapas lo necesitan.

Para comprender el funcionamiento de este sistema, es importante recordar (ver sección 2.3.1) que la *Etapa 2* tiene en su entrada la señal residuo más ECAP, donde el residuo es la resta de la señal entrante del sistema, SENSADO, con el template de la *Etapa 1*. El aspecto clave es que la señal entrante de la *Etapa 2* se amplifica, de forma que el residuo y la ECAP se amplifican por igual. El problema surge si el template de la *Etapa 1* pierde eficacia y aumenta el error. Esto causa que el residuo amplificado aumente en magnitud enormemente, haciendo que la entrada de la *Etapa 2* sature, perdiendo linealidad y destruyendo totalmente el funcionamiento del sistema. Este módulo, entre otras funcionalidades, intenta evitar esta problemática.

El funcionamiento básico de este módulo es iniciar el sistema, habilitando la *Etapa 1* y brindándole el flujo de datos del ADC. La *Etapa 1* está activa hasta que logre la convergencia. Esta se logra cuando el error de la resta entre su entrada y su template está por debajo de una cota preestablecida. Luego, se cede el control a la *Etapa 2*, habilitando su funcionamiento y brindándole el flujo de datos del ADC. La *Etapa 2* también se monitorea con el objetivo de mantener su señal entrante dentro del nivel de excursión admisible. En caso de detección de valores fuera de rango, se vuelve a la *Etapa 1*, para que vuelva a calcular su template y se obtenga un error aceptable, comenzando nuevamente el ciclo de funcionamiento.

La interfaz de este módulo, sus sub bloques y el flujo principal de datos se detallan en la figura 5.1. Los sub bloques se dividieron por colores, donde el violeta se refiere al control del ADC, el rojo al control de la *Etapa 1*, el celeste al control de la *Etapa 2* y el amarillo a los de control general del sistema.

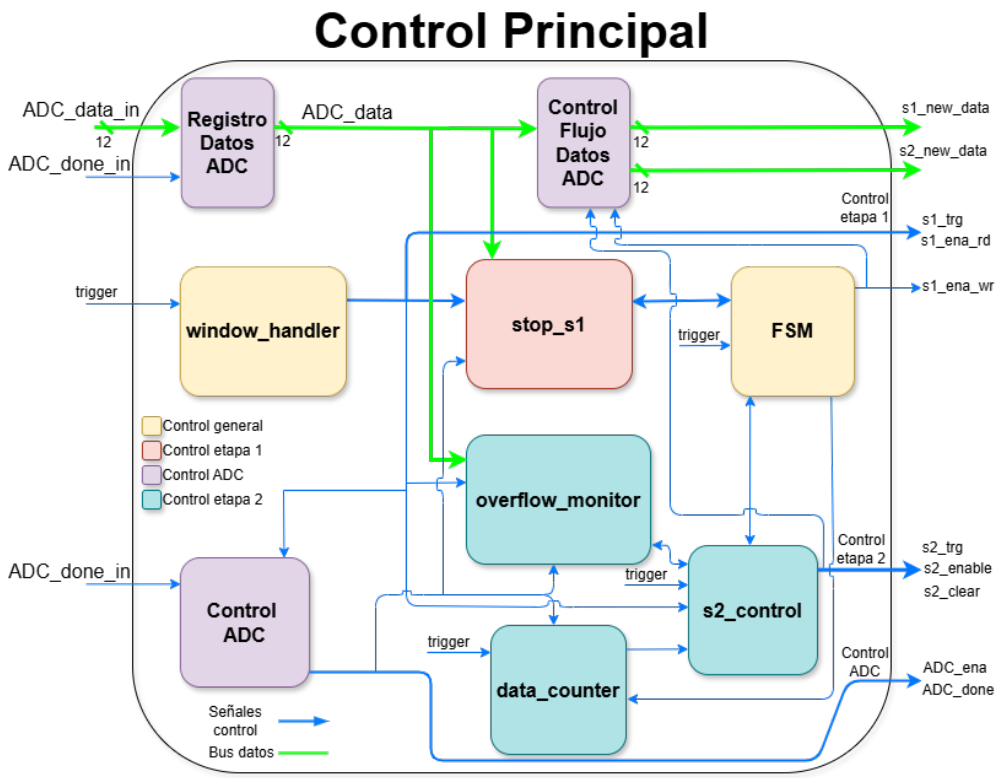

Figura 5.1: Diagrama simplificado del Control Principal. Se detallan los bloques principales que contiene y el intercambio simplificado de señales entre ellos. Los colores de los bloques indican sobre que sección del sistema actúa cada uno.

Los bloques referentes al ADC se encargan de registrar tanto los datos como la señal de control provenientes del ADC. Además, direcciona los datos registrados recibidos a la etapa correspondiente, en base a señales de control provenientes de sub bloques del módulo.

El bloque más importante es el *window_handler*. Este se encarga de establecer un intervalo del período de la señal entrante, donde se encuentra la mayor probabilidad de ocurrencia de la ECAP. Este intervalo se lo llamó enventanado. Todo el sistema CANICs se basa en este enventanado para funcionar, estando el sistema en espera fuera de este.

Cuando *window_handler* indica que se está dentro de la ventana, el bloque FSM (Finite-State Machine) habilita la *Etapa 1* a través de sus respectivas señales de control, y también pone en funcionamiento al bloque *stop_s1*. Este monitorea el estado del error del template de la *Etapa 1*. Cuando se alcanza el margen tolerable, se lo indica a la FSM, la cual entonces habilita la *Etapa 2*. Notar aquí que la *Etapa 1* sigue produciendo la señal residuo más ECAP en su salida, solo que ahora su template se mantiene fijo.

El sistema permanece entonces en la *Etapa 2* indefinidamente, mientras su entrada de datos esté dentro del rango de valores admisibles. Para controlar esto, se utiliza el bloque *overflow_monitor*, que en caso de producirse valores fuera de

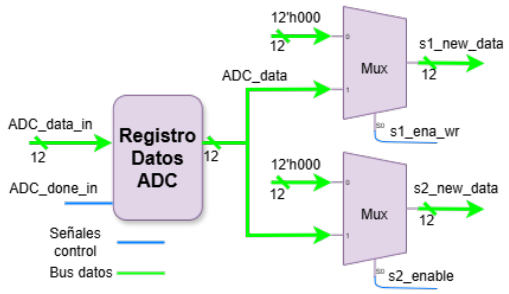

Figura 5.2: Sistema de manejo de datos del ADC. El mismo incluye un registro para guardar el nuevo dato entrante, y dos MUX para direccionar los datos a la etapa correspondiente.

rango, avisa a la FSM, la cual vuelve a habilitar la *Etapa 1* para que recalcule su template.

Los bloques *data_counter* y *s2_control* se utilizan para controlar el inicio de muestreo de datos dentro de la *Etapa 2*, con el objetivo de evitar efectos transitorios indeseados que posee el sistema. En las siguientes secciones se explica cada bloque del módulo en detalle.

5.2. Control Flujo de Datos del ADC

Los bloques *Registro Datos ADC* y *Control Flujo Datos ADC* de la figura 5.1 son los encargados de distribuir los datos provenientes del ADC a las respectivas etapas, cuando éstas lo requieran. Para ello, primero se registra el dato proveniente del ADC, para evitar depender de que el bus de datos de entrada mantenga estable la señal un tiempo excesivamente largo ($19T_{CLK}$). De esta tarea se encarga el primer bloque *Registro Datos ADC*. Es, como lo indica su nombre, un registro, en donde se guarda el dato nuevo del ADC. Por otro lado, el bloque *Control Flujo Datos ADC* se encarga de direccionar los datos del ADC a la etapa correspondiente. Su implementación es de dos MUX, los cuales poseen como señales de control **s1_ena_wr**, la cual se usa para dirigir los datos a la *Etapa 1*, y la señal **s2_enable**, la cual direcciona los datos a la *Etapa 2*. En caso de que estas señales estén inactivas, los buses de datos **s1_new_data** y **s2_new_data** van a ser fijados a cero. El bloque detallado se observa en la figura 5.2.

5.3. Bloque de enventanado *window_handler*

Este bloque es un área de control importante del sistema, dado que es el que genera el enventanado de la señal de entrada del ADC. El enventanado se refiere a capturar un intervalo temporal de la señal de entrada que contenga la ECAP. El objetivo de este enventanado es el de reducir el tiempo necesario de muestreo de la señal, por lo tanto, poder poner al sistema en espera y, como consecuencia, bajar el consumo. Por otro lado, al no estar contenida la ECAP en los tramos de señal no

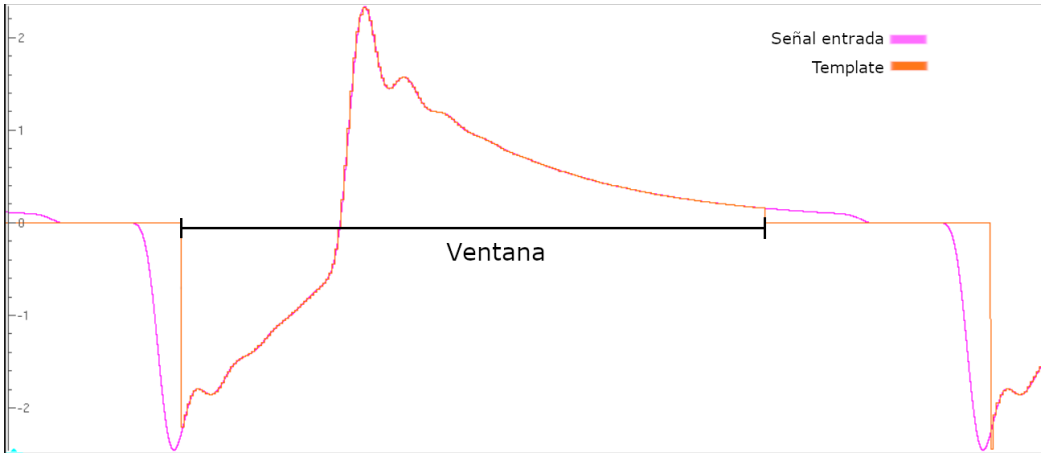

Figura 5.3: Comparación de la señal de entrada SENSADO, en rosa, con el template generado dentro de la ventana temporal, en naranja.

considerados, no tiene un efecto práctico generar un template de la SA para esos sectores, de forma que además se reduce la memoria necesaria para almacenar el template de la SA.

El enventanado se utiliza en ambas etapas y se basan en este para su funcionamiento. En consecuencia, las etapas no detectan que la señal de entrada SENSADO tenga una duración mayor que la ventana elegida para operar.

Esta ventana se eligió siguiendo las especificaciones de la sección 2, las cuales indican que la ECAP tiene una probabilidad de ventana de ocurrencia $\mathcal{N}(\mu = 470\,\mu s, \sigma = 62\,\mu s)$. Por lo tanto, por las propiedades de la distribución normal, tomando un intervalo $\mu \pm 3\sigma$, se obtiene una probabilidad del 99,73 % de ocurrencia de la ECAP, lo que se consideró aceptable para la aplicación. A modo de ejemplificar el mecanismo de este sistema, se observa en la figura 5.3 la comparativa de la señal de entrada, en rosa, con respecto al template, en naranja. El intervalo temporal donde la señal **template** es no nula se refiere a la ventana elegida. Como se ve, las secciones no contempladas por la ventana no son significativas, pero aportan una reducción de procesamiento y coste de recursos relevante para el sistema en general.

5.3.1. Cálculo del Intervalo de la Ventana Temporal

Considerando el inicio del frente de onda N1 de la ECAP, especificado en el capítulo 2, se determina que el intervalo de tiempo en el cual hay un 99,73 % de probabilidad de que comience la ECAP, se corresponde al rango $\mu \pm 3\sigma$. A partir de este rango, se obtiene el intervalo temporal, comprendido entre $[284\,\mu s, 656\,\mu s]$, medido desde el inicio de la fase negativa del SA.

Dado que este intervalo indica el posible comienzo de la ocurrencia de la ECAP, para el cálculo del intervalo de la ventana temporal, hay que añadir la duración temporal de la ECAP para poder capturarla en su totalidad. Como se desprende de la figura 2.2, la ECAP tiene una duración de $220\,\mu s$, por lo que el intervalo de

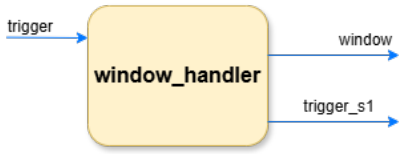

Figura 5.4: Interfaz del bloque *window_handler*. Cuenta con la entrada **trigger** y las salidas **window** y **trigger_s1**.

la ventana temporal es de [284 µs, 876 µs].

Debido al tiempo de asentamiento necesario por el template de la *Etapa 1*, se necesita esperar aproximadamente 169 µs para comenzar la ventana. En este tiempo, el template transiciona su fase de arranque y adquiere un error de copia bajo. Este fenómeno se explica en la sección 5.8. Concatenando este tiempo a la ventana se obtiene que el intervalo final de la ventana temporal es de [115 µs, 876 µs], siendo su duración 761 µs.

Dado que el período de la señal SENSADO es de 1111 µs, la ventana es aproximadamente un 68,5 % del tiempo. Usando el enventanado, se reduce considerablemente el intervalo de señal de interés, respecto al período total de señal. Fuera de la ventana, las etapas permanecen inactivas, sin muestrear ni reconstruir señales, lo que repercute en una reducción del uso de recursos del sistema. Finalmente, luego de ajustes a la hora de hacer las pruebas con la PCB, el tiempo de enventanado resultante fue de un 72,8 % del valor total del período. Este intervalo temporal equivale a utilizar 3640 períodos de reloj de los 5000 totales que posee un período la señal SENSADO, usando $f_{CLK} = 4{,}5\,\text{MHz}$. Recordando que el ADC demora $19\,\text{T}_{\text{CLK}}$ en muestrear un nuevo dato, se obtiene que por ventana se adquieren 192 muestras como máximo.

5.3.2. Implementación

Dado que lo que se desea es saber cuándo el sistema se encuentra dentro de la ventana de ocurrencia de la ECAP, el bloque *window_handler* está compuesto por un contador de períodos de reloj. Haciendo la conversión de tiempo a número de períodos ($F_{CLK} = 4{,}5\,\text{MHz}$), se pueden obtener fácilmente los valores de comienzo y fin de la ventana temporal. El bloque tiene la interfaz de la figura 5.4. La señal **trigger** se encarga de sincronizar el inicio del contador, dado que indica la llegada de una nueva señal SENSADO. Luego el bloque tiene dos salidas, **window**, la cual indica por nivel alto cuándo el sistema se encuentra dentro de la ventana y la señal **trigger_s1**, que es el pulso de trigger de la *Etapa 1*. Este trigger se acciona a la vez que la ventana comienza. Cabe destacar que la señal **window** también se utiliza como enable para la lectura de datos de la *Etapa 1* y para la generación del trigger y habilitación de la *Etapa 2*. Una simulación de estas señales, junto con la señal de entrada SENSADO en negro y el template de la *Etapa 1* en verde se observan en la figura 5.5. El intervalo entre los *Markers* indica un enventanado de la señal.

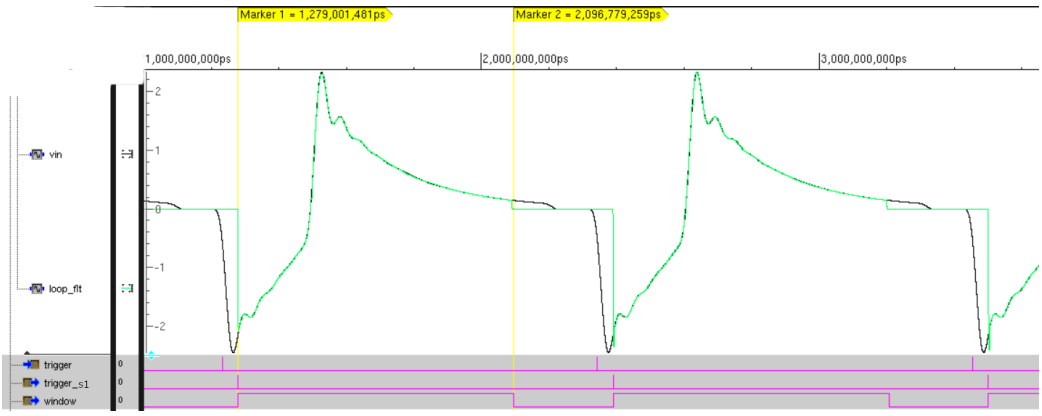

Figura 5.5: Simulación del bloque *window_handler*, mostrando el comportamiento de sus señales.

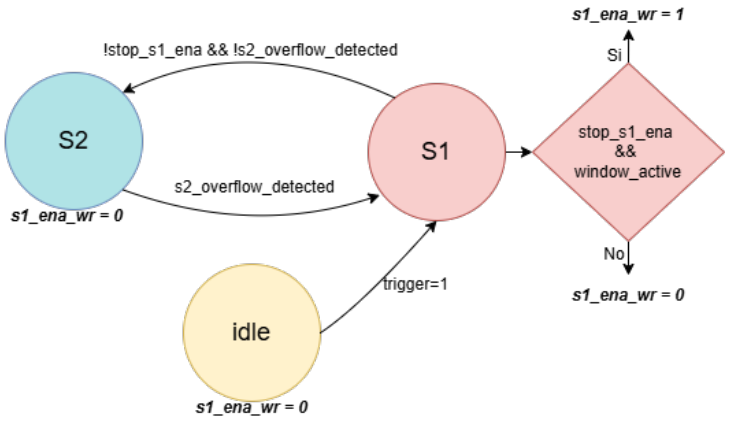

Figura 5.6: Diagrama de la máquina de estados de control de etapas, con sus señales de control. Los círculos indican los estados, mientras que el rombo es una condición lógica, en donde dependiendo de la condición **stop_s1_ena** && **window_active**, la señal **s1_ena_wr** vale uno o cero.

5.4. Máquina de Estados de Control de Etapas FSM

Esta sección se encarga de administrar el ciclo de funcionamiento de las etapas, siguiendo el diagrama de estados que se observa en la figura 5.6. La FSM tiene un estado para cada etapa de funcionamiento y un estado extra de reposo, en el cual solo se encuentra al inicio o al efectuarse un reset del sistema.

La principal señal de control que maneja la FSM es la **s1_ena_wr**, la cual activa la escritura del template de la *Etapa 1*. Esta señal se activa cuando el sistema se encuentra en el estado S1, si la ventana está activa y el error está por encima del umbral deseado. Por otro lado, los propios estados S1 y S2 sirven como señales de control para otros bloques, a través de las señales **stage_1** y **stage_2**, a modo indicativo de qué etapa está activa en el sistema. Las señales de control utilizadas por la FSM para transicionar estados son **stop_s1_ena**, salida del bloque *stop_s1*

y **s2_overflow_detected**, proveniente de *s2_control*, que indica que hubo datos muestreados fuera del rango esperado.

La FSM inicia en el estado *idle*, en el cual se mantiene hasta que llega un pulso de **trigger**, transicionando al estado *S1*. En este estado, se activa la señal **stage_1**. Dentro de la ventana, si el bloque *stop_s1* reporta que el error está por encima del umbral (**stop_s1_ena** = 1), se activa la señal **s1_ena_wr**. Esta señal se activa dentro de la ventana, mientras ocurra esta condición, no durante todo el período de señal SENSADO. En los otros estados de la FSM o bajo otras condiciones, **stop_s1_ena** = 0. Finalmente, si se llegó al umbral de error deseado en el template de la *Etapa 1* (**stop_s1_ena** = 0) y no se detectaron valores fuera del umbral en los datos de la *Etapa 2* (**s2_overflow_detected** = 0), se pasa al estado *S2*. En este estado, se activa la señal **stage_2** y la señal **s1_ena_wr** vale cero, deshabilitando la escritura del template de la *Etapa 1*. El sistema se mantiene en este estado, hasta que se detecten valores fuera de rango en la *Etapa 2* (**s2_overflow_detected** = 1), en cuyo caso se vuelve al estado *S1*.

5.5. Control ADC

Este bloque se encarga de controlar la señal **ADC_ena**, la misma habilita el funcionamiento del ADC a través del controlador del ADC, explicado en el capítulo 3. También registra la señal entrante **ADC_done_in**, a través de esa propia señal, resultando en la señal **ADC_done**. Esto es necesario para evitar depender de que un módulo externo mantenga estable la señal a la hora de utilizarla, y mantener el sincronismo respecto al delay que introduce registrar los datos provenientes del ADC. El ADC se habilita cuando alguno de los dos triggers, **s1_trg** o **s2_trg** es activado, es decir, cuando alguna de las etapas comienza a funcionar y se mantiene activo mientras la señal **window_active** esté activa, es decir, que se encuentre dentro de la ventana de funcionamiento del sistema y no se detecten valores fuera de rango mientras está funcionando la *Etapa 2*.

5.6. Control de error de la etapa 1 *stop_s1*

El objetivo de este bloque es generar una señal de control que indique cuándo detener la escritura del template de la *Etapa 1*, cuando este haya llegado a un umbral de error preestablecido, que se determinará experimentalmente. Este bloque se encarga de monitorear la diferencia del template generado en la *Etapa 1* con la señal entrante, lo que a efectos de esta sección, se considerará como el error del template. Para lograr este cometido, el bloque captura los datos provenientes del ADC mientras la *Etapa 1* está activa. El método de cálculo del error utilizado fue la suma de los valores absolutos, o SAE:

$$SAE = \sum_{i=1}^{n} \left| IN(i) - SA_{pred}(i) \right|, \quad n = \#\ muestras, \tag{5.1}$$

Figura 5.7: Interfaz del bloque *stop_s1*. Cuenta con las señales de control **trigger_s1** y **enable** para sincronizar el funcionamiento del bloque y la señal **ADC_done** para recibir los datos del ADC, a través del bus **ADC_data**. La salida del bloque es la señal **enable_template**, la cual indica por nivel alto cuando el error del ciclo está por encima del umbral preestablecido.

siendo $IN(i)$ la muestra i-ésima de la señal SENSADO y $SA_{pred}(i)$ la muestra i-ésima del template generado.

Se eligió este método respecto a otros, como por ejemplo el error cuadrático medio, porque es el más sencillo de implementar a nivel de hardware, dado que no requiere cálculos complejos, sólo se necesita sumar y eventualmente cambiar el signo de valores. Además, el objetivo de este bloque es establecer una métrica consistente para obtener el error, no determinar con precisión el error en sí mismo, por lo que para la aplicación se consideró que es el método más acertado.

Dado que $|IN(i) - SA_{pred}(i)|$ representan las muestras de la salida del amplificador diferencial, lo que tiene que hacer este bloque es simplemente tomar las muestras del ADC y efectuar la suma de los valores absolutos. Luego, esa suma de los errores individuales de cada muestra a lo largo del ciclo enventanado de la señal SENSADO, da como resultado una métrica, con la que se puede establecer experimentalmente una cota aceptable del error, en donde el template se considere válido, y dar por concluida la fase de escritura de la *Etapa 1*.

5.6.1. Interfaz

La interfaz del bloque se ilustra en la figura 5.7, en donde se observan las señales de control y datos a la entrada y su salida. La señal **trigger_s1** tiene el fin de sincronizar el comienzo del ciclo de la nueva señal de entrada SENSADO, y por lo tanto comenzar la cuenta del error desde cero a partir de ese punto. La señal **enable** indica cuándo el bloque tiene que activarse y funcionar. La señal **ADC_done** se encarga de sincronizar la llegada de un dato nuevo del ADC, proveniente del bus **ADC_data** y poder operar con él. Finalmente, la señal **enable_template** indica por nivel alto cuándo el error está por encima o es igual al umbral de error configurado, y cambia a nivel bajo cuando el error está por debajo del umbral.

5.6.2. Funcionamiento

La figura 5.8 refleja el comportamiento del bloque en un ciclo de funcionamiento, en donde **enable** indica la ventana del ciclo. Partiendo del bus **ADC_data**, se calcula el valor absoluto **abs_data** de la muestra i-ésima de la ecuación 5.1. Con este bus se realiza la suma parcial de las siguientes i muestras entrantes, generando el vector **partial_sum**. Luego, cuando $i = n$, **sae_trigger** genera un pulso cuan-

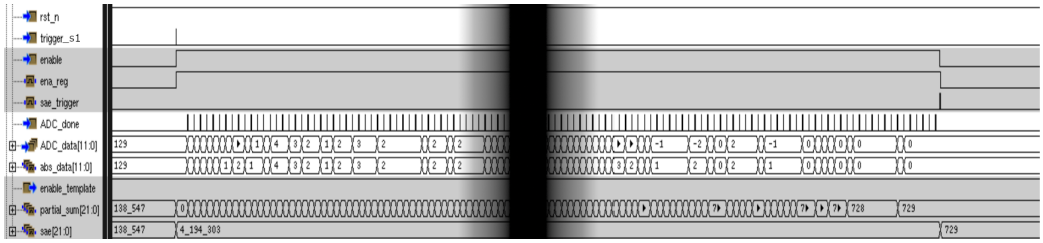

Figura 5.8: Simulación del bloque *stop_s1*. Se muestra el funcionamiento en un ciclo arbitrario de la señal SENSADO. Se omitieron valores intermedios para lograr observar tanto el ciclo de arranque como la fase de apagado.

do la señal **enable** baja, de forma de actualizar el valor obtenido del error **sae**. En ese momento, se le carga a **sae** el último valor calculado en **partial_sum**. Un detalle destacable en el diseño es que con el pulso de **trigger_s1**, al vector **sae** se le asigna su valor máximo (ver señal en figura 5.8). Esto se hace para habilitar la escritura del template de la *Etapa 1*, dado que si **sae** inicia en cero o valores por debajo del umbral, este bloque da por válido el template sin siquiera calcularlo. Al final, cuando el ciclo termina y **enable** baja, se le asigna el valor de error calculado.

5.7. Control de error de la etapa 2 *overflow_monitor*

Este bloque tiene como objetivo controlar que la señal de entrada de la segunda etapa esté dentro de la excursión esperada. La forma elegida para controlar estos márgenes fue monitorear los valores muestreados por el ADC, los cuales luego ingresan a la fase de procesamiento digital (bloque *MSD*) de la segunda etapa. Por lo tanto, el bloque compara los valores provenientes del ADC con márgenes determinados, máximo y mínimo, los cuales generan un rango de datos aceptable. Si un dato sale del rango, el sistema advierte overflow, para que se tomen las medidas pertinentes.

Los márgenes están basados en el rango de representación digital que posee el ADC. Si el ADC está reportando datos muy cercanos a su valor máximo o mínimo, es un indicador de que la señal está muy cercana a los límites de excursión admisibles por el ADC. Por lo tanto, se definió un valor umbral λ. A partir de este valor umbral y los valores extremos de escala del ADC, se definieron el margen superior $MAX\_ADC - \lambda$ y el margen inferior $MIN\_adc + \lambda$. El rango de valores entre estos márgenes define el rango admisible de datos provistos por el ADC. Se eligió experimentalmente un umbral $\lambda = 10$, de forma de maximizar la excursión del sistema. Considerando los 12 bits del ADC y que utiliza representación en complemento a 2, el rango total de datos es $[-2048, 2047]$. El rango admisible de datos, considerando $\lambda = 10$ es de $[-2038, 2037]$.

La interfaz del bloque se observa en la figura 5.9. Cuando la señal **enable** se activa, al ritmo de datos que indica la señal de control **ADC_done** el sistema monitorea el valor de los datos del ADC, provenientes del bus **ADC_data**. Si algún dato sobrepasa los márgenes preestablecidos, se activa la señal **overflow**. En

Figura 5.9: Interfaz del bloque *overflow_monitor*. El mismo cuenta con dos señales de control de entrada y un bus de datos. A su salida tiene la señal de control **overflow**.

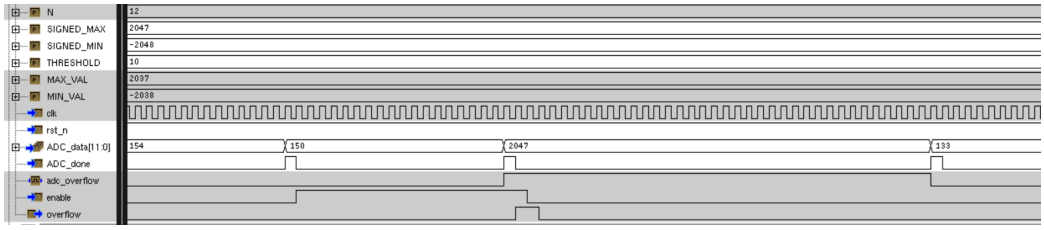

Figura 5.10: Simulación del bloque *overflow_monitor*, en donde se observa un caso de overflow por el margen superior.

ese caso, se detiene el funcionamiento de la segunda etapa, y se vuelve a activar la primera etapa de forma que actualice su template hasta que el error sea aceptable. Una simulación del sistema funcionando se observa en la figura 5.10, en donde los datos recibidos por **ADC_data** están dentro del rango aceptable, hasta que se recibe un dato fuera del margen (**ADC_data**= 2047), por lo que el sistema genera el aviso de overflow, activando la señal **overflow**.

5.8. Acumulador de datos *data_counter*

El objetivo de este contador es esperar el tiempo de asentamiento de diferentes componentes del sistema, de forma de habilitar el funcionamiento de la *Etapa 2*. Como se observa en la simulación de la figura 5.11, al inicio de la ventana, indicado con el *Marker 6*, las primeras muestras del template de la *Etapa 1* no se amoldan totalmente a la señal entrante, necesitando un período de tiempo para llegar al régimen (1). Además, la señal **OUT_AMP** que es la salida del amplificador variable de entrada de la *Etapa 2* también necesita un tiempo de asentamiento para alcanzar el régimen a su salida (2). Análogamente ocurre con el filtro Butterwoth **butter_out**, el mismo tiene un transitorio de arranque (3) y necesita un determinado tiempo de asentamiento para alcanzar el régimen en su salida. Este régimen final se observa en el *Marker 7* y se determinó mediante simulación.

Por lo tanto, este bloque *data_counter* tiene que esperar el tiempo de asentamiento de todos estos componentes para luego poner en funcionamiento la *Etapa 2*. Esta medida es necesaria para el correcto funcionamiento de la *Etapa 2*. En caso de que no se use, el error de copia en esas primeras muestras hace que sature la señal de entrada amplificada de la *Etapa 2*. Por ende, este bloque genera un sub enventanado, el cual es la ventana del bloque *window_handler* pero con el inicio

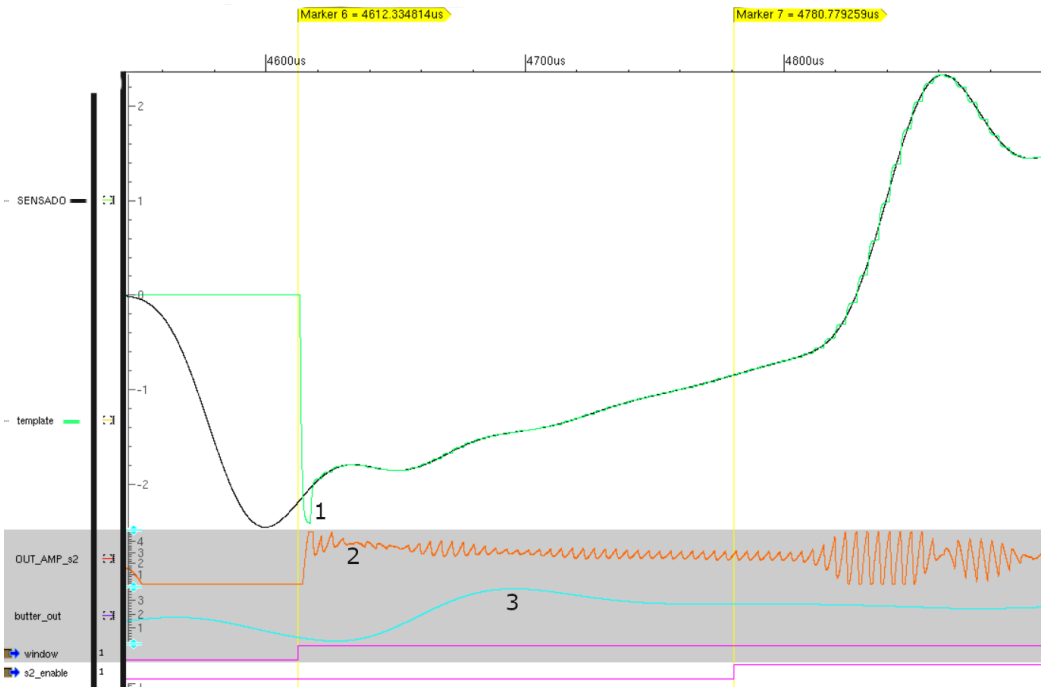

Figura 5.11: Simulación que muestra el arranque del template de la *Etapa 1* luego de su convergencia. Para que la *Etapa 2* comience a funcionar, es necesario esperar el transitorio de varios retardos. Primero que el template de la *Etapa 1* alcance el régimen (1), luego que el amplificador variable de entrada de la *Etapa 2* haga su transitorio de arranque (2), esperar el transitorio del filtro Butterworth (3), y finalmente que alcance el régimen, indicado con el *Marker 7*.

retardado la cantidad de muestras que necesita la *Etapa 1* en alcanzar el régimen, que es aproximadamente 169 µs (*Marker 7-Marker 6*). Por este motivo se anticipó el inicio del enventanado, de forma de también contemplar el tiempo de asentamiento del template de la *Etapa 1*, y el retardo temporal que introducen la etapa de amplificación y filtrado posteriores.

El agregado de este intervalo de espera fue considerado la medida más acertada, debido a que es innecesario poseer el template completo de la SA con una precisión abrumadora en los extremos de la ventana, dado que la ECAP se encuentra principalmente en el centro. Cabe recordar también que el objetivo es tener un template confiable en la ventana de ocurrencia de la ECAP, y no un template fidedigno de la SA completa.

Este bloque, como lo indica su nombre, es un acumulador de datos, no de períodos de reloj. Es decir, cuenta la cantidad de paquetes generados por el ADC. Contando la cantidad de datos provenientes del ADC se puede calcular el tiempo de delay, con la ventaja de usar un contador más pequeño comparado con contar períodos de reloj. El equivalente a los 169 µs es contar 40 datos provenientes del ADC (un dato del ADC surge cada $19\,T_{CLK} \approx 4{,}22\,\mu s$). La interfaz del bloque se presenta en la figura 5.12, tiene tres entradas de control, **enable** para habilitar el funcionamiento del bloque, **clear** para resetear la cuenta a cero y **ADC_done**,

Figura 5.12: Interfaz del bloque *data_counter*. El mismo cuenta con tres señales de control de entrada. A su salida tiene la señal de control **s2_settled**, la cual indica que el contador alcanzó el valor objetivo.

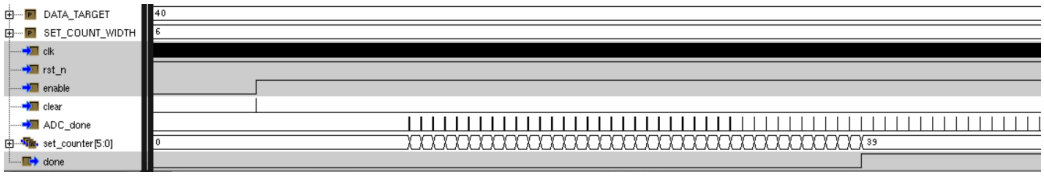

Figura 5.13: Simulación del bloque *data_counter*, en donde se observa su funcionamiento típico.

pulso que indica un nuevo paquete, el cual es la base de funcionamiento de este bloque. Cuando el contador llega al valor objetivo, la señal **s2_settled** permanece activa, hasta que se borre el contador, sea con un **clear** o un reset. A la señal **clear** se le conecta la señal de control **trigger** y a **enable** se le conecta la señal **stage_2**, proveniente de la FSM, que indica cuándo está habilitada esta etapa. Una simulación del bloque representando este funcionamiento se observa en la figura 5.13.

5.9. Control de la etapa 2 *s2_control*

Este bloque es necesario para que la *Etapa 2* inicie su funcionamiento en el instante temporal correcto, respetando el transitorio que se produce al arranque de la *Etapa 2*, explicado en la sección 5.8. Este sistema se encarga de generar las señales de control **s2_enable**, **s2_clear** y **s2_trigger**, las cuales son las principales señales de control de la *Etapa 2*. Además de estas señales, posee el FF **s2_overflow_detected**, el cual se activa cuando el sistema se encuentra en el estado de segunda etapa, **stage_2**, y **overflow** está activo. Luego, la señal vuelve a nivel bajo cuando el sistema continúa en el estado de segunda etapa y se produce un pulso de **trigger**, indicando el comienzo de un nuevo ciclo. Un esquema del comportamiento de las señales se detalla en la figura 5.14.

La señal **s2_enable** habilita el funcionamiento de la segunda etapa, se activa luego de que el bloque *data_counter* indique que concluyó el tiempo de asentamiento, con su señal de salida **s2_settled**. Además, debe estar la ventana activa, indicado por la señal **window_active** y no se debe haber producido overflow, es decir, **s2_overflow_detected** esté inactiva.

Finalmente, la señal **s2_clear** se activa cuando **s2_overflow_detected** está activada, lo cual se traduce a que si eventualmente se produce que los datos del

Figura 5.14: Esquema del comportamiento de las señales de control de la segunda etapa.

ADC excursionan fuera de los umbrales establecidos, se borra el template de la *Etapa 2*, para evitar utilizar los posibles datos corruptos ingresados.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 6

Simulaciones

6.1. Introducción

En el presente capítulo se describe el entorno de simulación utilizado para validar el funcionamiento del sistema CANICs. Se detalla el entorno de simulación diseñado, incluyendo los elementos necesarios para su implementación. Posteriormente, se muestran los resultados obtenidos a partir de las simulaciones realizadas. Finalmente, se presenta el modelado funcional de los bloques analógicos involucrados.

El primer indicio de que un diseño es confiable es que haya sido sometido a una exhaustiva verificación. Para poder lograr una verificación satisfactoria, se debe contar con el entorno adecuado para realizar las simulaciones pertinentes del sistema. Es necesario contar con modelos que representen el comportamiento funcional de los componentes analógicos discretos utilizados de forma simplificada pero representativa, permitiendo así validar el sistema CANICs en su conjunto. El objetivo de estos modelos no es reproducir cada detalle especificado por su respectiva hoja de datos, sino abstraer su funcionamiento de manera que resulten adecuados para los fines de simulación y verificación. Esto es especialmente importante en módulos como el ADC, DAC y los amplificadores operacionales, cuya interacción con el sistema digital es fundamental para el desempeño general del sistema.

Con la sección analógica modelada, es posible llevar a cabo la verificación mixed signal del sistema, enfocándose particularmente en la interacción con el diseño digital, que constituye el núcleo de este proyecto. Dado que el objetivo final de este sistema es su implementación en un circuito integrado, se optó por utilizar la plataforma de diseño Cadence, ampliamente empleada en la industria de microelectrónica. Esta herramienta permite la simulación conjunta de bloques analógicos y digitales, ofreciendo un entorno especializado para el diseño de sistemas integrados. Además, la Facultad cuenta con licencias institucionales, lo que facilita su uso durante el proceso de verificación del diseño.

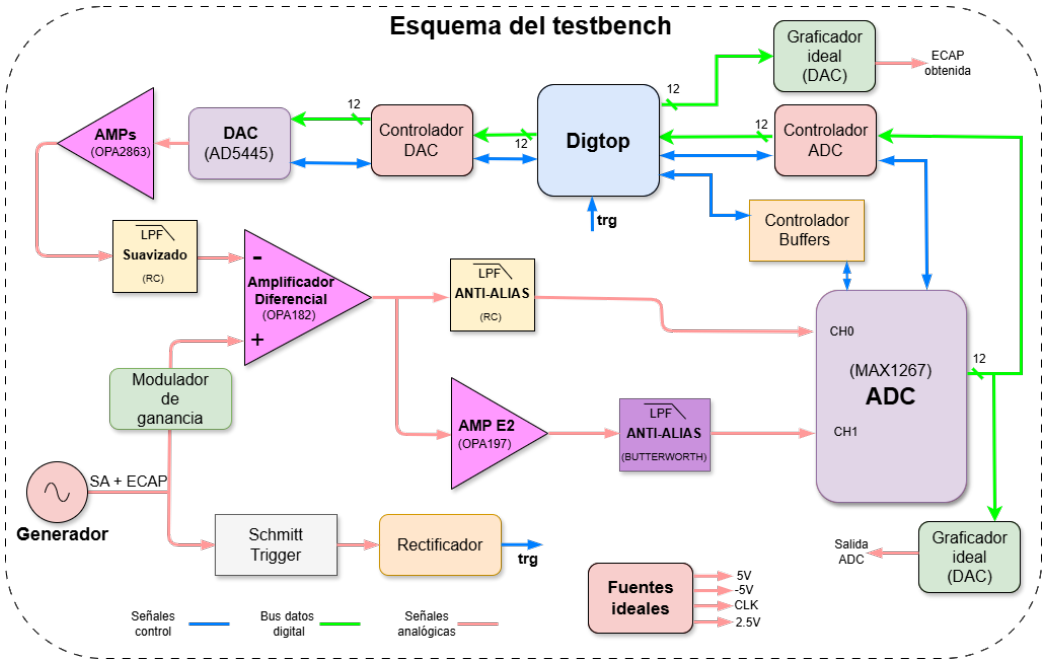

Figura 6.1: Esquema con los bloques principales que componen el testbench utilizado.

6.2. Testbench principal

En base a los modelos analógicos desarrollados, que se explicarán sobre el final de este capítulo, junto con el diseño digital elaborado, se procedió a realizar simulaciones del sistema completo. Cabe destacar que los bloques digitales se probaron individualmente con sus respectivos testbenchs, donde en capítulos previos se explican algunos diagramas de estas simulaciones. En esta sección se va a describir el procedimiento utilizado para probar el sistema CANICs en su totalidad y presentar los resultados obtenidos de estas simulaciones.

La estructura del testbench implementado se presenta en la figura 6.1. Está compuesto por un lazo con la misma estructura y componentes que el de la *Etapa 1* (figura 3.1), donde los bloques digitales se sustituyeron por el bloque denominado **Digtop**, el cual contiene todo el diseño digital del sistema CANICs, es decir, el bloque **DSP** de la *Etapa 1*, el **MSD** de la *Etapa 2* y el *Control Principal*. Además, se agregó el procesamiento analógico de la *Etapa 2*, el cual consta de su amplificador de ganancia variable y filtro anti alias, para su respectivo canal de entrada del ADC. Para el manejo del ADC se agregó el controlador de los Buffers, el cual gestiona el control de canales del ADC. Como elementos de simulación, se agregó un bloque que cambia la ganancia de la señal de entrada, de forma de hacer pruebas con el sistema luego de que haya alcanzado su convergencia. Finalmente, se generó un bloque para graficar las señales digitales obtenidas, tal como la ECAP a la salida del sistema, en base a un DAC ideal. Dado que el testbench posee una vasta cantidad de elementos, a continuación se explicará por secciones los diferentes circuitos implementados y componentes presentes.

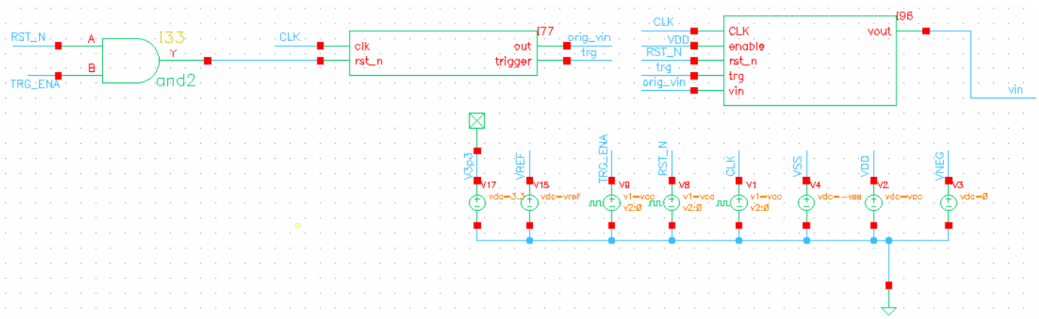

Figura 6.2: Circuito de entrada y fuentes de alimentación del testbench utilizado.

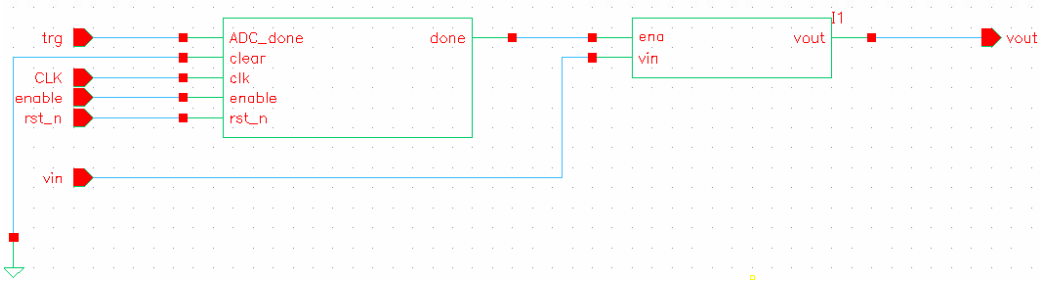

Figura 6.3: Circuito de cambio de ganancia de la señal de entrada.

6.2.1. Esquemático

La figura 6.2 muestra el circuito de entrada del sistema, junto con las fuentes de alimentación y señales del mismo. El circuito está compuesto por dos bloques principales. El I77 contiene el módulo de trigger, encargado de generar el pulso de sincronismo para el sistema digital, a partir de la señal SENSADO. Este módulo es el diseño de la sección 3.4, compuesto por los circuitos de las figuras 3.3 y 3.6. Además, el bloque I77 contiene un generador de la señal SENSADO, junto con un bloque que ajusta la ganancia de la misma al rango requerido. A su salida, el bloque entrega la señal SENSADO, denominada **orig_vin**, en el rango de tensión a utilizar, y la señal **trigger**.

El bloque I96 es un recurso de simulación diseñado para generar una variación controlada en la señal de su entrada **vin**, donde su objetivo es probar el estado del sistema CANICs en el que, estando en funcionamiento la *Etapa 2*, el sistema diverge por cambios en la entrada **orig_vin**. Está compuesto por los elementos de la figura 6.3, donde el bloque de entrada es otra instancia del bloque acumulador de datos, explicado en la sección 5.8, que en vez de contar pulsos de la señal **ADC_done**, cuenta pulsos de **trigger**. Cuando el contador llega al valor objetivo, este activa el bloque I1, generando un cambio de ganancia en la señal de entrada **vin**.

El valor objetivo se determinó que fuera superior al número mínimo de ciclos necesarios para que el sistema comience a entregar datos. Son al menos dieciséis ciclos para la *Etapa 2*, y para la *Etapa 1* se estimó experimentalmente una cota de cuatro, por lo que este umbral debe ser mayor que veinte. Finalmente, se usó

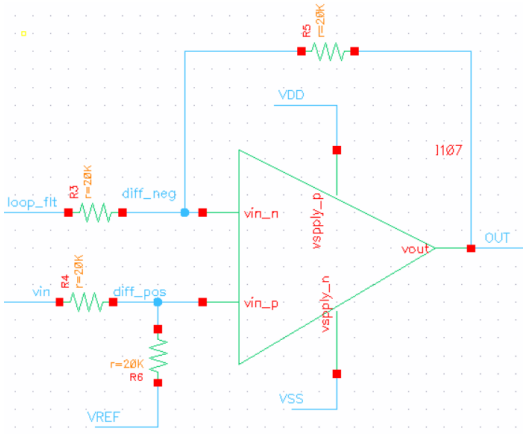

Figura 6.4: Circuito de la etapa de amplificación diferencial del testbench.

treinta, a modo de también observar algunos valores de salida en el primer ciclo de convergencia.

El circuito está diseñado para conservar este valor de ganancia hasta el final del testbench, dando posibilidad al sistema de tener la señal de entrada estable para reactivar el funcionamiento de la *Etapa 1* y volver a la convergencia. Finalmente, la salida de este bloque, **vin** de la figura 6.2, es la entrada de señal al sistema, la cual cambia de ganancia una vez.

Siguiendo el análisis del circuito y el flujo de la señal **orig_vin**, una vez que esta es procesada por el bloque I96, se obtiene en su salida la señal **vin**. Cabe notar que en las simulaciones no se consideró el amplificador de entrada INA, dado que se utilizó una señal single-ended como entrada del sistema.

Por lo tanto, la señal **vin** continúa al nodo positivo de entrada del amplificador diferencial. El circuito implementado para este amplificador se muestra en la figura 6.4. Se utilizó el modelo de operacional explicado en la sección 6.3.3, ajustando los parámetros principales, como ganancia, ancho de banda y output swing, para emular el amplificador utilizado en la realidad.

La salida **OUT** del amplificador diferencial, luego de ser procesada por el filtro anti aliasing del tipo RC, pasa a la entrada del canal cero del ADC, donde se utilizó el modelo explicado en la sección 6.3. Como se observa en la figura 6.5, la salida del ADC, además de dirigirse al controlador, se direcciona a un bloque denominado *DAC_PLOTTER*. Este bloque es un DAC modelado en Verilog-A, con el fin de poder graficar en el tiempo los valores digitales que el ADC procesa, de forma de evaluar las formas de onda resultantes. El bloque previo a la entrada del *DAC_PLOTTER* es un conversor de tipo de datos, para transformar los datos de complemento 2 a binario convencional. Finalmente, se incluye el controlador de los buffers (su funcionamiento se detalla en el apéndice B.2), responsable de gestionar los dos buffers tristado de la familia 74AHC9541A de Nexperia [20], mostrados en la figura 3.12. La incorporación de este controlador en el testbench tiene como objetivo verificar que el comportamiento temporal de las señales de control (oe_n1 y oe_n2) coincida con el diseño previsto. Para más detalles sobre el diseño de estas

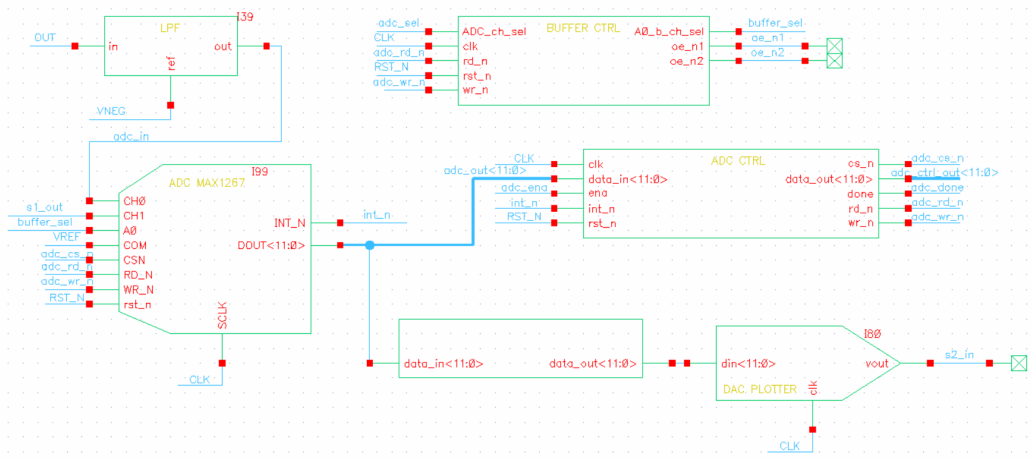

Figura 6.5: Sección correspondiente al ADC y módulos relacionados.

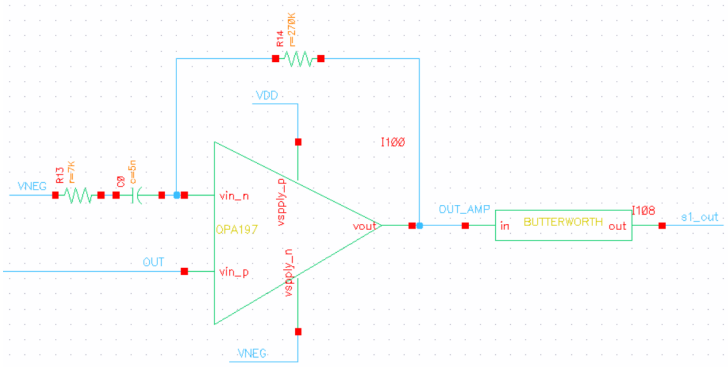

Figura 6.6: Amplificador y filtro de salida de acondicionamiento para la *Etapa 2*.

señales, consultar el apéndice B.2.

Por otro lado, la señal **OUT** eventualmente también debe dirigirse a *Etapa 2*. Para ello, se implementó el circuito de la figura 6.6, el cual implementa el amplificador de ganancia variable y el filtro de entrada de la *Etapa 2*. En este caso, en vez de utilizar el potenciómetro, se ajustó la ganancia con una resistencia. El circuito del filtro Butterworth implementado en simulación es análogo al explicado en la sección 4.3.1.

Luego de que los datos son procesados por el ADC, ingresan al bloque digital, ilustrado en la figura 6.7. Este bloque está compuesto por el bloque digital *DSP* de la *Etapa 1*, bloque digital *MSD* de la *Etapa 2* y el *Control Principal*. En la salida de datos de la *Etapa 2* también se conectó otro DAC ideal a modo de graficador de datos, para poder ver la forma de onda y las amplitudes en la salida. Cabe destacar que el bloque digital es esencialmente el mismo que se utiliza en la FPGA, por lo que algunas señales de debug para la implementación no fueron conectadas.

Por último, se encuentra el DAC y sus módulos dependientes de él. En la figura 6.8 se observa que los datos provenientes del bloque digital ingresan al controlador del DAC, para luego dirigirse al DAC en sí mismo, donde a su salida posee los

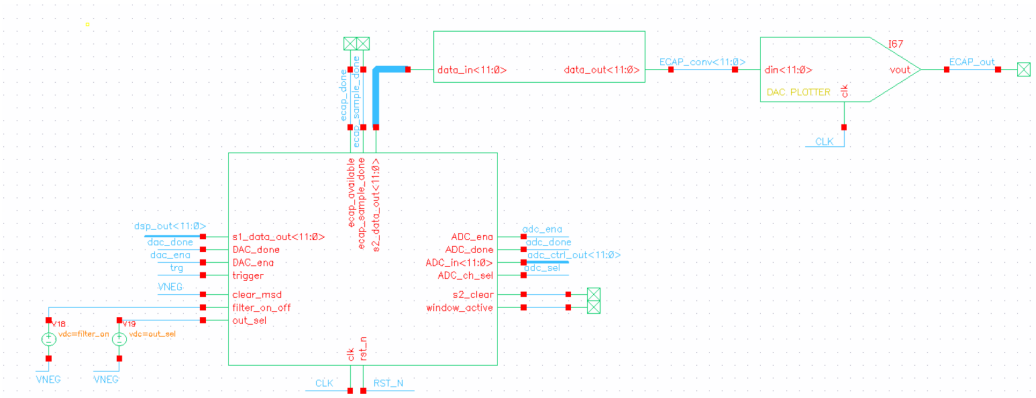

Figura 6.7: Circuito del bloque digital principal del testbench.

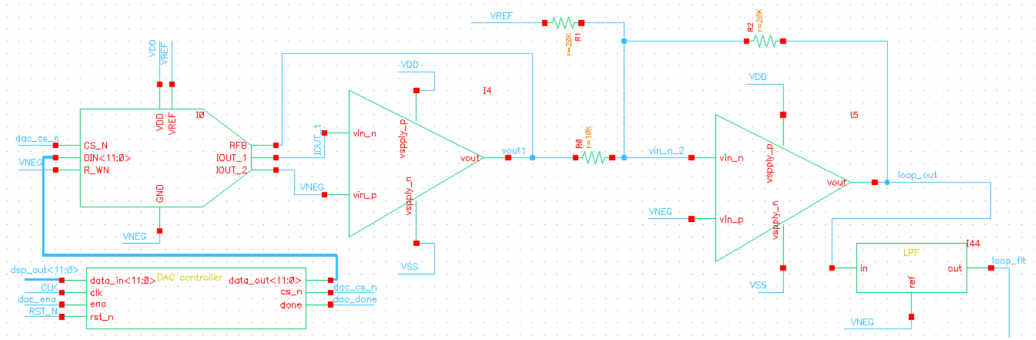

Figura 6.8: Circuito del DAC y sus módulos dependientes.

amplificadores, con sus parámetros debidamente ajustados, para convertir la salida del DAC de corriente a tensión. Finalmente, para acondicionar la señal de salida se conectó un modelo del filtro de suavizado, en el cual su salida se conecta al amplificador diferencial, cerrando el lazo de la *Etapa 1*.

6.2.2. Resultados

Con el entorno de simulación explicado anteriormente, se hicieron las pruebas globales del sistema, con el objetivo de verificar el correcto funcionamiento y tener una primera métrica del desempeño. Para ello, se realizaron simulaciones transitorias, de duración extendida (60 ms), con el objetivo de probar el sistema al arranque y en régimen. Además, como se explicó anteriormente, se verificó también el correcto funcionamiento del sistema ante eventuales cambios en la señal de entrada SENSADO.

Para ello, en la entrada se estimuló con la SA y ECAP planteadas en la sección 2.2. El bloque de entrada de la simulación las amplifica de forma que la SA tiene una amplitud de 4,8 Vpp y la ECAP tiene una amplitud de 3 mVpp. Dado que el diseño es para un dispositivo implantable, la simulación se realizó para una temperatura de 37 °C.

Figura 6.9: Funcionamiento de la *Etapa 1* en la simulación top level. Se simuló con una relación de 53 dB entre la SA y la ECAP. La salida del DAC **loop_filt** se ilustra en celeste y la entrada **vin** en rojo.

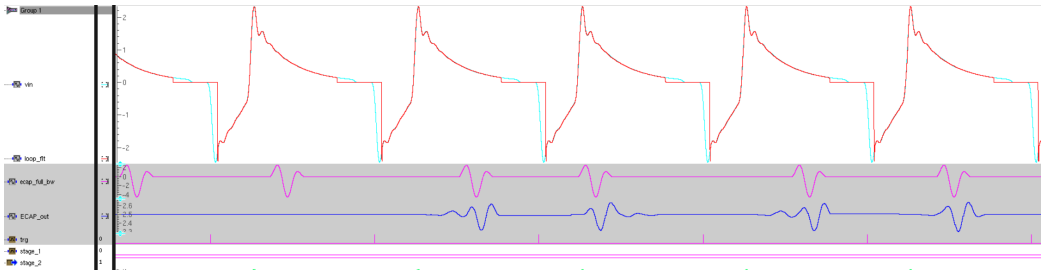

Figura 6.10: Comienzo de generación de datos del sistema. Se simuló con una relación de 53 dB entre la SA y la ECAP. La salida del DAC **loop_filt** se ilustra en rojo y la entrada **vin** en celeste.

El primer objetivo a probar fue el de verificar que la *Etapa 1* converge, y además obtener la cantidad de iteraciones que esto conlleva y con ello determinar el tiempo que demora en converger. En la figura 6.9 se observa la señal **vin**, entrada del sistema y **loop_flt**, salida filtrada del DAC, superpuestas. Se obtiene que la *Etapa 1* demora cuatro períodos de estimulación ($4T$) en converger, lo que equivale a unos 4,44 ms. Inmediatamente se activa la *Etapa 2*, indicado a través de la señal **stage_2**. Se puede observar también que en la primera iteración del template ya se asemeja mucho al valor final de señal (**loop_flt** es muy similar a **vin**).

Luego de transcurridos los dieciséis períodos de estimulación, la *Etapa 2* terminó su ciclo iterativo y comienza a desplegar resultados en su salida. La figura 6.10 refleja esta transición en la señal de salida **ECAP_out**. A modo comparativo, también se grafica la señal **ecap_full_bw**, siendo la señal ECAP de entrada de referencia.

El sistema permanece en este estado de régimen generando los valores de ECAP en la salida mientras los valores muestreados por la *Etapa 2* estén dentro de los rangos de valores aceptables, tal como se explicó en el capítulo 5. Si por algún motivo cambia la morfología de la señal de entrada SENSADO, específicamente la componente SA, la *Etapa 1* puede acumular el error de forma que la señal muestreada en la *Etapa 2* exceda los valores permitidos. Este caso es ilustrado en la figura 6.11, en donde se observa que en la salida **ECAP_out** se obtiene ECAPs normalmente, hasta que un cambio en la señal SA (ver señal **vin**) produce una variación abrupta en **ECAP_out**, logrando que la señal **overflow** se active. Con este evento, la *Etapa 2* detiene la salida de datos y el *Control Principal* queda en

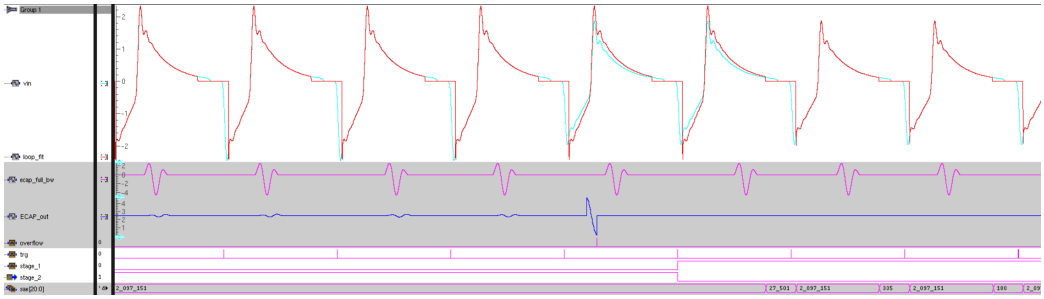

Figura 6.11: Caso en que la señal muestreada por la *Etapa 2* excursiona fuera del rango aceptable, en donde el sistema retoma el funcionamiento de la *Etapa 1*. Se simuló con una relación de 53 dB entre la SA y la ECAP.

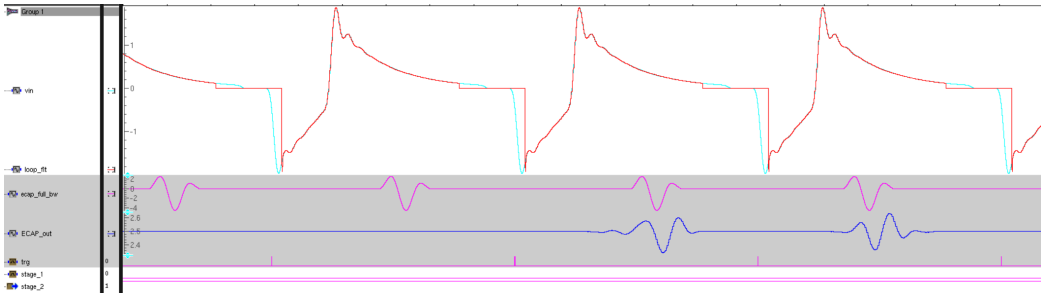

Figura 6.12: Caso de retorno a la generación de ECAPs luego de que el sistema atravesara un overflow en la *Etapa 2*. Se simuló con una relación de 53 dB entre la SA y la ECAP.

espera del comienzo del siguiente período de señal, para volver a iniciar *Etapa 1*.

Finalmente, análogo al caso de arranque, el *Control Principal* activa la *Etapa 1*, sólo que en este caso la memoria ya contiene un template previo, operando sobre este para lograr la convergencia nuevamente. Cuando se logra la convergencia, se activa la *Etapa 2*, luego de la espera de dieciséis períodos de estimulación, en donde cuando se cumplen, el sistema vuelve a generar las ECAPs a la salida, como se observa en la figura 6.12.

6.3. Modelos de bloques analógicos

Como se explicó en la sección y los capítulos anteriores, algunas de las etapas del sistema poseen o controlan bloques analógicos, siendo principalmente los conversores analógico-digitales los bloques de mayor complejidad.

Al no encontrar modelos de simulación compatibles de los componentes utilizados para el ADC y DAC, del tipo SPICE o similares, se optó por desarrollarlos siguiendo las especificaciones de las hojas de datos. Se consideró para este fin utilizar el lenguaje Verilog-A, que es un lenguaje de modelado analógico basado en Verilog, diseñado específicamente para describir el comportamiento de sistemas electrónicos analógicos y mixtos (analógico/digital), y es compatible con Cadence. Utilizar este lenguaje fue un desafío en sí mismo, dado que no se tenían anteceden-

tes por parte de ningún miembro del equipo en su utilización, así como de lenguajes similares que se destinen a modelar componentes analógicos.

Una vez realizado el modelado de los componentes, se procedió a verificar los modelos para asegurar que cumplieran con las especificaciones mínimas establecidas en las hojas de datos. Dado que el objetivo de estos modelos es validar el sistema completo, y no el componente analógico utilizado en sí mismo, se generaron modelos básicos comportamentales, que cubrieran los aspectos esenciales que permitieran generar una simulación aceptable del sistema completo.

En la *Etapa 1*, como se observa en la figura 3.1, los componentes analógicos que posee son el ADC, el DAC y amplificadores operacionales. La *Etapa 2* posee un amplificador y un filtro en la entrada; luego es esencialmente digital, y el *Control Principal* maneja también el ADC.

En resumen, son tres bloques básicos los que se necesitan modelar, ADC, DAC y amplificadores operacionales. Dado que se utilizaron amplificadores con diferentes características, se dejaron diferentes aspectos paramétricos, tales como ganancia, output swing, slew rate, entre otros. En las siguientes secciones se detalla la implementación de cada modelo analógico.

6.3.1. ADC

El objetivo es realizar un modelo lo más simple posible del ADC MAX1266 que permita validar nuestro sistema. Se busca representar únicamente aquellas características relevantes para la aplicación, como el comportamiento de las señales de control digitales, la resolución de 12 bits y la funcionalidad de tracking and hold a nivel funcional que describe el fabricante en su hoja de datos.

Dado que el MAX1266 involucra tanto señales analógicas como digitales, su implementación se divide en módulos analógicos, escritos en Verilog-A, encargados de modelar el tracking and hold de la señal analógica. Y se tiene un módulo digital en Verilog, modelando el comportamiento digital del MAX1266.

El módulo digital se encarga de modelar el funcionamiento lógico del ADC y de reproducir el comportamiento temporal especificado para su interfaz digital. En la figura 6.13 se muestran las señales digitales del dispositivo, cuyo comportamiento se busca replicar según lo descrito por el fabricante.

La interfaz obtenida se observa en la figura 6.14, esta posee los mismos pines que el MAX1266, a excepción del pin rst_n el cual se agregó intencionalmente para obtener una sincronización de los componentes internos digitales con el sistema global.

Internamente, el modelo de la figura 6.14 está compuesto por los bloques *Main control*, *selector* y *ADC T&H* , tal como se observa en la figura 6.15.

El bloque *selector* es el modelo de un multiplexor analógico, en el que dependiendo del valor de la señal de entrada vsel, se coloca a la salida vout la señal vin1 o vin2. Si vsel se corresponde a un 1 lógico, la salida será vin2; en caso contrario, será vin1. Este multiplexor modela dos de los seis canales analógicos del MAX1266, ya que solo se requieren dos: uno para la *Etapa 1* y otro para la *Etapa 2*. Por lo tanto, no es necesario representar todos los canales del ADC en el modelo.

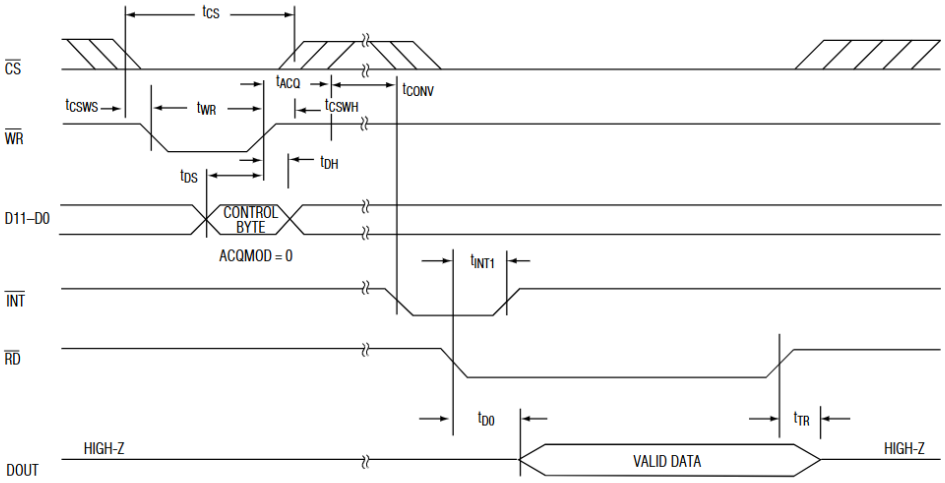

Figura 6.13: Diagrama de tiempos del ADC MAX1266 en *internal acquisition mode*. Se observa el comportamiento esperado de las señales de control $\overline{CS}$, $\overline{WR}$, $\overline{RD}$ con la señal de interrupción $\overline{INT}$, junto con el bus triestado D11–D0. Fuente: Hoja de datos del MAX1266 [19].

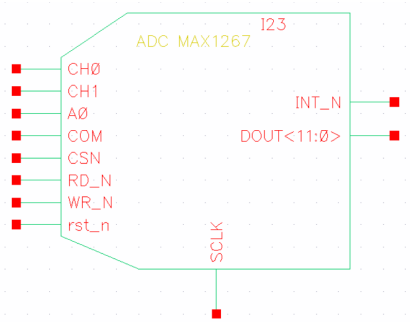

Figura 6.14: Interfaz del modelo del ADC MAX1266.

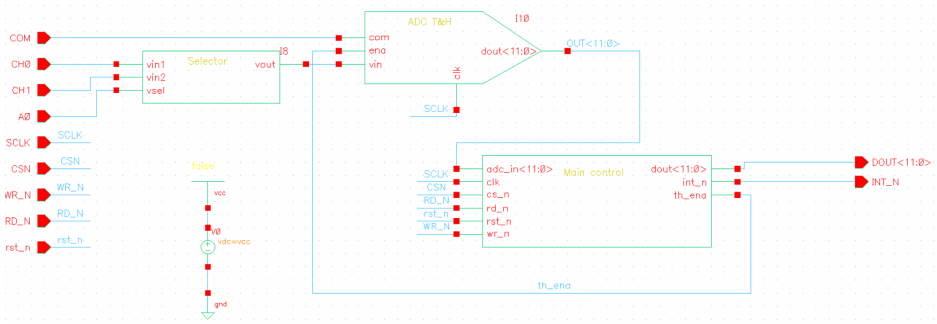

Figura 6.15: Módulos internos del modelo del ADC MAX1266. Está compuesto por los bloques *ADC T&H*, *selector* y *Main control*

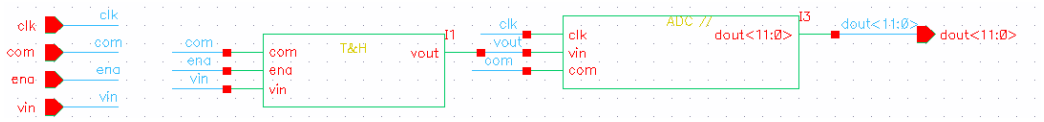

Figura 6.16: Bloque de tracking and hold de datos, junto con el conversor a palabras digitales.

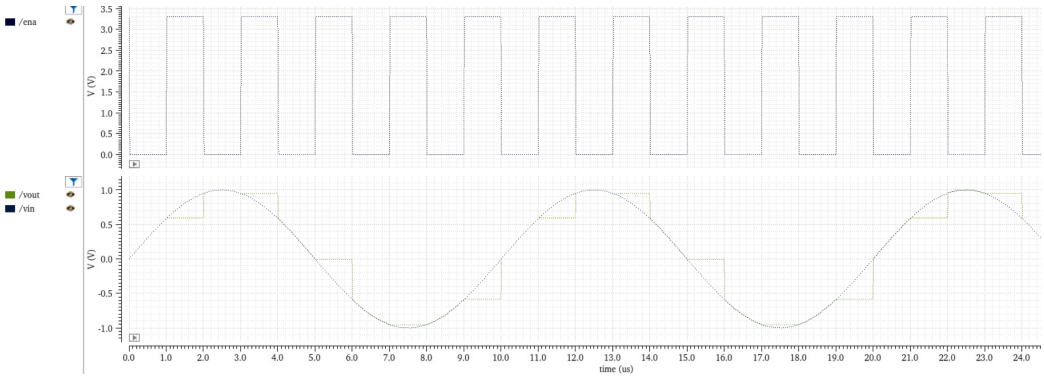

Figura 6.17: Simulación del bloque *T&H*. Cuando ena vale uno el bloque retiene el valor de entrada. Cuando ena vale cero el sigue a su entrada.

El bloque *ADC T&H* es el que emula el comportamiento del muestreo propiamente dicho del ADC. Es decir, muestrear los datos analógicos y convertirlos a valores digitales. Para simplificar el problema, se dividió este bloque en dos problemas independientes, que también se corresponde con la arquitectura en que está construido el MAX1266. En la figura 6.16 se aprecia el primer bloque *T&H*, encargado de realizar el tracking and hold de la señal analógica, y el segundo *ADC //* de procesarla y convertirla a una palabra digital de 12 bits.

En la figura 6.17 se representa el comportamiento temporal del *T&H*. Cuando la señal ena está inactiva, se observa que la salida vout sigue (tracking) a su entrada vin. Mientras que si ena está activa, el bloque mantiene (hold) el valor que tenía la señal vout en el instante temporal en que el flanco negativo de ena ocurre. Este bloque permite mantener la señal de interés estable el tiempo suficiente para que el bloque *ADC //* emule el tiempo de conversión del dato analógico a digital. El tiempo en el cual la señal ena está activa realizando tracking y haciendo hold se obtuvo de la figura 6.18 de la hoja de datos del MAX1266. Se aprecia que el tiempo en el que se está en el estado *acquisition* (tracking) es de $4T_{CLK}$, mientras que se está $12T_{CLK}$ en el estado *conversion* (hold).

El bloque *ADC //* se encarga de transformar la señal que recibe en su entrada vin a una palabra digital de 12 bits. Es un conversor ADC paralelo, sincronizado por la señal de reloj clk. El bloque tiene configurados los parámetros de forma que se comporte como el MAX1266, entre ellos los niveles de tensión de los niveles lógicos, la tensión umbral, los tiempos de rise y fall estimados, entre otros.

El bloque *Main control* se encarga de operar las señales digitales que posee el MAX1266. Esencialmente este sistema digital es una máquina de estados que tiene las entradas y salidas digitales del MAX1266, encargado de emular el com-

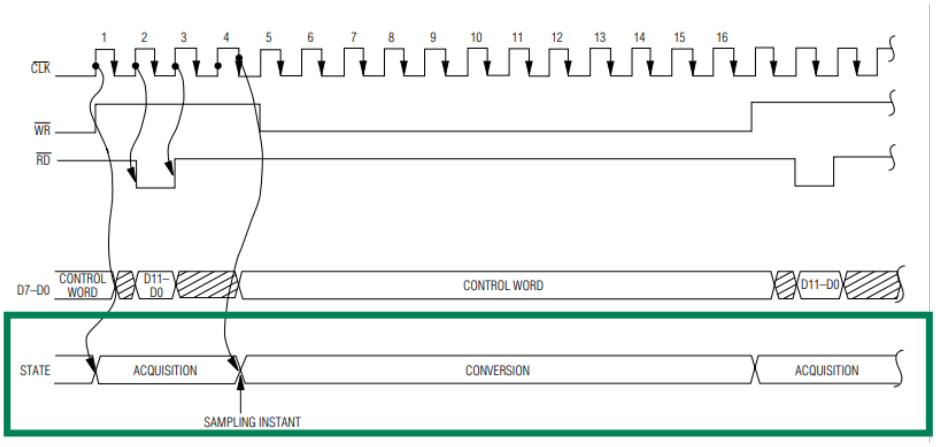

Figura 6.18: Tiempos asociados a los estados (*state*) de seguimiento (*acquisition*) y retención (*conversion*) de la señal analógica durante el muestreo: $4T_{CLK}$ para el *tracking* y $12T_{CLK}$ para la conversión. Fuente: hoja de datos del MAX1266 [19].

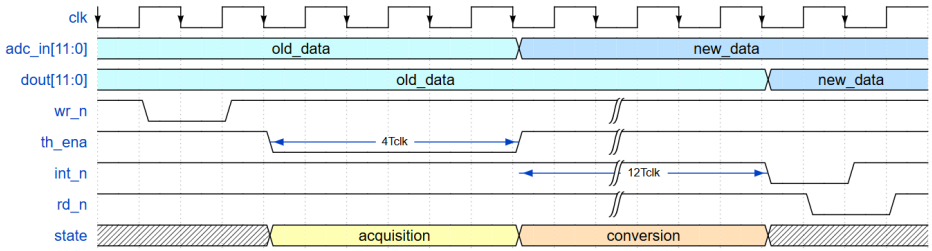

Figura 6.19: Diagrama de tiempos del comportamiento del bloque *Main control*.

portamiento temporal de las mismas, descrito en las figuras 6.13. Internamente, se encarga de operar la señal ena del bloque *T&H* de modo de cumplir con los tiempos de tracking and hold especificados por el fabricante, descritos en la figura 6.18.

En la figura 6.19 se representa el comportamiento temporal del bloque *Main control*. Se observa que cuando el host[1] quiere iniciar una adquisición de datos, genera un pulso de bajada en **wr_n**, luego el *Main control* entra en el estado de *acquistion* (tracking), activando su señal **th_ena** (siendo la misma que ena del bloque *T&H*). Transcurridos los $4T_{CLK}$, se entra al estado *conversion* (hold), donde el bloque *T&H* retiene el dato analógico, para luego ser convertido a digital por el bloque *ADC //*, que luego este le transfiere el dato nuevo al *Main control*. Los $12T_{CLK}$ restantes donde *Main control* tiene internamente el dato nuevo hasta colocarlo a su salida **dout** emulan el comportamiento real del MAX1266 al convertir el dato de analógico a digital. Al finalizar los $12T_{CLK}$, el bloque coloca a su salida **dout** el dato digital muestreado, dando aviso al bajar su señal **int_n**. Cuando el host adquiere el nuevo dato, le avisa al bloque con un pulso de **rd_n**. El

[1]Es el encargado de controlar digitalmente el funcionamiento del ADC. En este caso, corresponde al módulo denominado *Controlador del ADC*.

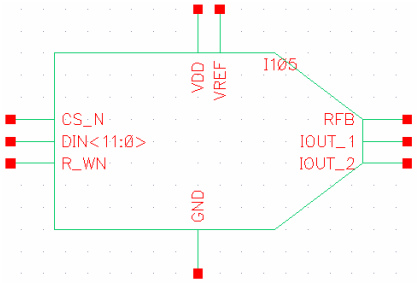

Figura 6.20: Interfaz del modelo del DAC AD5445.

comportamiento temporal por parte del host para utilizar las señales del MAX1266 fue ilustrado en la figura 3.13.

No obstante, la hoja de datos del MAX1266 [19] describe una forma de alcanzar la tasa máxima de muestreo del ADC, que consiste en escribir la palabra de control (pulso bajo de **wr_n**) antes de leer el último dato convertido. En este modelo, la lógica de control para ello no fue implementada, ya que se consideró que la frecuencia de muestreo utilizada para conversiones sucesivas (f_S =237 KHz) es suficiente para la aplicación prevista. Además, operar a la máxima velocidad complica el manejo de las señales de control, ya que requiere iniciar una nueva conversión sin haber leído aún el dato anterior.

6.3.2. DAC

El modelo que representa al DAC AD5445 es básicamente la implementación del arreglo R-2R y sus respectivas llaves, tal como se muestra en la figura 3.15. La interfaz del bloque se observa en la figura 6.20. La misma cuenta con los mismos terminales que el DAC real, por lo que el funcionamiento es directamente el mismo. Internamente, el modelo espera un pulso de CS_N para actualizar las salidas IOUT, y funciona solamente si la entrada R_WN está en nivel bajo, es decir, en modo de escritura. El modo de lectura no fue implementado, dado que no se consideró relevante para la aplicación. Al modelo se le aplicaron los valores de la datasheet de resistencia, tiempos de setup y hold, entre otros.

6.3.3. Amplificadores operacionales

Dado que se utilizaron diferentes modelos de amplificadores operacionales, los cuales varían sus características dinámicas, se decidió generar un modelo base paramétrico, dentro del cual se puedan editar estos parámetros dinámicos relevantes, tales como ganancia, slew rate, frecuencia del polo, resistencias de entrada y salida, offset e I_{bias}. Con este modelo, para cada versión de amplificador se le destina un nombre acorde y el set de parámetros pertinentes. Un modelo utilizado se presenta en la figura 6.21.

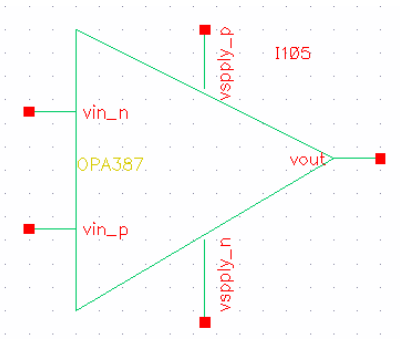

Figura 6.21: Interfaz de un modelo de amplificador operacional implementado.

Capítulo 7

Diseño de PCB CANICs

7.1. Introducción

En el presente capítulo se describen las características de la PCB desarrollada para la validación del sistema CANICs. Esta PCB integra todos los bloques analógicos del sistema mencionados en el transcurso de los capítulos anteriores. A su vez, integra el HW complementario necesario para su funcionamiento, tales como reguladores de tensión y circuitos de adaptación de niveles lógicos. Se cuenta con una interfaz diseñada para su conexión directa con la placa de desarrollo (FPGA), encargada de ejecutar el diseño digital.

Dado que el sistema constituye una prueba de concepto, la PCB fue diseñada con un enfoque modular que permite realizar pruebas por etapas y facilita el acceso a nodos intermedios para tareas de medición y verificación. A continuación, se presenta el proceso de diseño y la implementación de los distintos circuitos integrados en la placa.

7.2. Diagrama de Bloques del HW

En la figura 7.1 se puede observar un diagrama de bloques de alto nivel con los componentes de la PCB desarrollada por CANICs:

- **Atenuador**: Genera una atenuación, adaptando la señal diferencial del generador de señales al rango de tensión de entrada del sistema CANICs. Es necesario cuando la ECAP tiene menor amplitud que el LSB del DAC del generador que se esté utilizando, ya que en ese caso el equipo no puede reproducir la señal con la resolución requerida.
- **Amplificador de Instrumentación (INA)**: Dado que la entrada al sistema proviene de forma diferencial, el HW cuenta a la entrada con un amplificador de instrumentación, transformando la señal del modo diferencial a single-ended, conservando el término diferencial de la señal y eliminando su modo común. Por más detalles, ver sección 3.3.

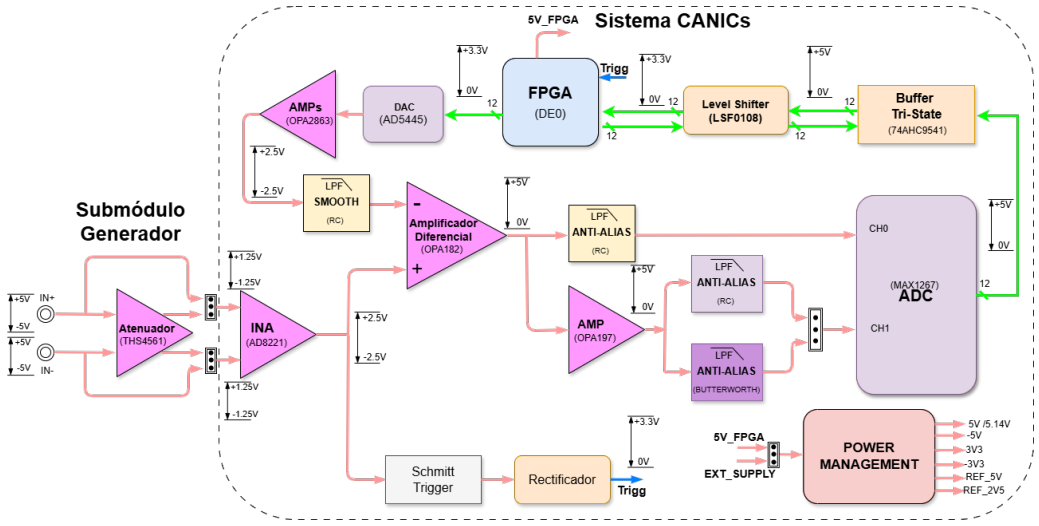

Figura 7.1: Diagrama de bloques de los componentes de la PCB junto con la FPGA.

- **Schmitt Trigger**: Genera una señal de sincronismo con la señal SENSADO para el sistema digital a partir de la señal de entrada single-ended obtenida a la salida del INA. Por más información, ver sección 3.4.
- **Rectificador**: Rectificador de media onda que adecúa la señal generada por el Schmitt Trigger a los niveles de tensión que soporta el sistema digital, con el objetivo de ser ingresada a la FPGA. Explicado en la sección 3.4.
- **Amplificador diferencial**: Es el encargado de realizar la resta entre la señal de entrada con el template generado en la *Etapa 1*. Su diseño se explicó en la sección 3.6.
- **Amplificador de ganancia variable (AMP)**: Es el encargado de amplificar la señal obtenida a la salida de la *Etapa 1* para maximizar el uso del ADC en la *Etapa 2*. Este diseño se explicó detalladamente en la sección 4.2.
- **Conversor Analógico Digital (ADC)**: Muestrea y convierte al dominio digital las señales analógicas entrantes de la *Etapa 1* y *Etapa 2*. Explicado en la sección 3.8.
- **Conversor Digital Analógico (DAC)**: Reconstruye en el dominio analógico el template digital de la *Etapa 1*. Se considera como parte del DAC a los amplificadores (AMPs) encargados de convertir la salida en corriente del DAC a tensión. Más detalles de este conversor en la sección 3.10.
- **Buffer Tri-State**: Se encarga del manejo del bus tri-state que tiene el ADC, permitiendo la habilitación de este bloque y la salida de datos hacia la FPGA. El circuito utilizado con este componente se explicará en este capítulo.
- **Level Shifter**: Debido a que existe una diferencia entre los niveles logicos del ADC y la FPGA, este bloque hace una traducción de los niveles lógicos

entre estos dos bloques, de forma bidireccional, permitiendo la comunicación entre ellos. Esta problemática y la solución desarrollada se explicará más adelante en este capítulo.

- **FPGA**: Será el bloque encargado de implementar el diseño digital realizado. En particular se opta por utilizar la development and education board (DE0) de Altera, más adelante se explica la razón de dicha elección.

- **Filtros Anti-Alias**: Los mismos efectúan un tratamiento a la señal para conservar el ancho de banda de interés, evitar el aliasing y disminuir la contribución del ruido a la hora de muestrear. Se realizaron tres filtros, explicados en la sección 3.7, 4.3.1 y 4.3.2.

- **Filtro Smooth**: Este filtro se implementó para limitar el ancho de banda de la señal generada por el DAC antes de su ingreso al amplificador diferencial. Su función principal es atenuar las componentes de alta frecuencia introducidos por la naturaleza discreta del DAC, evitando así que estos afecten la correcta cancelación de señales en la etapa diferencial. Su diseño se explicó en la sección 3.12.

- **Power Management**: Bloque transversal al diseño capaz de proveer alimentación a todos los elementos del Hardware. Se explicará en detalle en este capítulo.

7.3. Niveles de tensión utilizados

La elección de tensiones en el sistema respondió a múltiples factores. Primero, el de tener la máxima tensión posible, de forma de maximizar la SNR de la señal de entrada. Este afán tiene un límite, principalmente por la escasa variedad de integrados que cumplen las características buscadas, específicamente de los conversores ADC y DAC. Otro factor es que los niveles de tensión lógicos están estandarizados, comúnmente a 3,3 V o 5 V. Además, dado que el diseño es orientado a una implementación en un circuito integrado, se buscó también que las tensiones utilizadas sean compatibles con la tecnología de fabricación, en particular se diseñó para el proceso $XH018$ de $XFAB$. Por este motivo, se definió una tensión máxima de 5 V para las señales del sistema CANICs. Como se muestra en la figura 7.1, este rango de tensión se establece en el trayecto de la señal analógica, desde la salida del INA hasta la entrada del ADC, lo cual determina principalmente los niveles de excursión de los conversores utilizados, así como sus tensiones de alimentación.

Con esta tensión determinada, se procedió a elegir el nivel de continua de la señal SENSADO. Se optó por que fuera 0 V, dado que se consideró que sería el nivel obtenido ante la respuesta real del sistema biológico. En consecuencia, a la entrada del atenuador como en la entrada del sistema se determinó 0 V de modo común. Por lo tanto, la señal de entrada del sistema (entrada del INA) excursionará en el rango de $[-1{,}25\,\text{V}, 1{,}25\,\text{V}]$ por canal, dado que la entrada es diferencial, y la tensión

máxima admisible es 5 V. Esto se traduce en su salida a una señal de [−2,5 V, 2,5 V] single-ended, como se observa en la figura 7.1.

Con el objetivo de maximizar la excursión útil de la señal dentro del sistema, se alimentaron los amplificadores del Schmitt Trigger, del INA y los de salida del DAC con una tensión simétrica de [−3,3 V, 3,3 V]. Esta elección permite compensar el efecto del output swing típico de estos dispositivos, asegurando que puedan entregar todo el rango de tensión requerido (5 V, como se explicó anteriormente) sin sufrir recortes ni distorsión cerca de los rieles de alimentación.

Durante la implementación se identificó una limitación importante: el ADC no admite tensiones negativas en sus canales de entrada (CH0 y CH1 en 7.1). Por este motivo, fue necesario introducir un offset de continua a la señal, de manera que su excursión se mantuviera dentro del rango permitido por el convertidor, es decir, entre [0 V, 5 V]. Para generar este offset se utilizó una referencia de 2,5 V que se ingresó en el amplificador diferencial, de forma que genere este offset en su salida. En consecuencia, este amplificador debe estar alimentado al menos con [0 V, 5 V]. Como ya se contaba con la alimentación de −3,3 V, se utilizó esta como fuente de alimentación negativa de este operacional de forma de eliminar la limitante del output swing. De igual forma, se utilizó una alimentación de 5,14 V para la fuente superior, para poder maximizar la excursión en esta fase del sistema. Por lo tanto, el amplificador diferencial fue alimentado con [−3,3 V, 5,14 V]. El amplificador de la segunda etapa (AMP en la figura 7.1) también se alimentó con este rango de tensión por el mismo motivo.

Finalmente, los buffers tri-state del ADC y el filtro Butterworth también se alimentaron con la fuente de 5,14 V, al igual que el DAC y el ADC. Esta alimentación, a través de los jumpers, se puede cambiar de valor, entre 5,14 V y 5 V provenientes de la FPGA.

Como se comentó, la FPGA elegida no soporta tensiones digitales de 5 V, por ende se agregaron level shifters a modo de traducir las tensiones bidireccionalmente. En cambio, la salida digital desde la FPGA hacia el DAC no requirió adaptación de niveles lógicos, ya que los 3,3 V proporcionados por la FPGA son reconocidos por el DAC como nivel lógico alto sin inconvenientes.

7.4. Estimación del consumo del sistema analógico

Dados los componentes del sistema, se tiene que el DAC tiene un consumo típico de 0,4 µA. El ADC necesita aproximadamente 3 mA para funcionar, dada la frecuencia de reloj de 4,5 MHz. Los amplificadores requieren aproximadamente 1 mA de corriente para su funcionamiento. Dado que hay 9 amplificadores en el sistema, suman 9 mA de consumo. Considerando estos componentes principales, se tiene un consumo de unos 12 mA. Agregando el resto de la circuitería junto con el consumo de los propios reguladores y fuentes utilizados, se estimó un consumo total de aproximadamente 30 mA, incluyendo un margen de seguridad.

Por lo tanto, se diseñará la alimentación para que soporte esta cota mínima de corriente. Este cálculo es a modo estimativo, para determinar el orden de magnitud

sobre cuál elegir los reguladores y fuentes a utilizar, además de determinar si la placa de desarrollo a utilizar soporta el consumo de todo el sistema o es necesaria una fuente externa.

7.5. FPGA utilizada

La FPGA utilizada fue la Cyclone III, integrada en la Development and Educational Board (DE0) de Altera [23]. Fue la principal opción a utilizar, debido a que se utiliza en diferentes cursos de la Facultad y se posee experiencia previa trabajando con la misma, además de la gran cantidad de soporte que se tenía por parte del equipo docente en su utilización. Una vista de la placa de desarrollo junto con sus componentes principales se observa en la figura 7.2. Esta placa posee 15.408 elementos lógicos, de los cuales el diseño utiliza 13.390. Una salida de 5 V entre sus pines, conectándose directamente a su alimentación USB, por lo que cómodamente puede suministrar corrientes de 500 mA o mayores, cubriendo cómodamente las necesidades de la PCB.

Por otro lado, la DE0 cuenta con dos conectores de entrada/salida con cuarenta pines cada uno, entre los que se encuentran los pines de alimentación, entrada/salida de reloj y pines digitales de propósito general. Este número cubre la cantidad de pines requeridos por el sistema CANICs (treinta y seis).

Como característica adicional, cuenta con un PLL[1] para configurar distintas frecuencias de reloj. Cuenta con una SDRAM de 8 MB[2], posee 10 LEDs y 10 switches, que pueden ser utilizados para distintos modos de debug. Como se mencionó, el inconveniente que presenta el uso de esta FPGA es que soporta como máximo 3,3 V de tensión en sus pines digitales.

7.6. Power Management

Esta parte de la PCB es la encargada de suministrar las tensiones de alimentación a todo el sistema. Como fuente principal para todos los circuitos de alimentación, se diseñó para que la PCB fuera capaz de configurarse para recibir alimentación desde la DE0, la cual puede suministrar 5 V; o desde una fuente externa, que aporte los 5 V directamente junto con las otras tensiones requeridas. A partir de esta fuente principal de 5 V, se generan los distintos niveles de voltaje de alimentación para los elementos que componen el diseño de la PCB. Las tensiones de alimentación necesarias por los componentes del sistema son 5 V, 3,3 V, −3,3 V y −5 V.

[1] Un PLL (Phase-Locked Loop) es un circuito de control que ajusta un oscilador para igualar la fase y frecuencia de una señal de referencia. En FPGAs, se usa para generar relojes derivados, permitiendo multiplicar o dividir la frecuencia del reloj de entrada con alta precisión y estabilidad.

[2] La SDRAM presente en la placa DE0 es una memoria dinámica sincrónica, utilizada para almacenar grandes volúmenes de datos temporales. Opera sincronizada con el reloj del sistema y se accede a través del controlador SDRAM interno del FPGA.

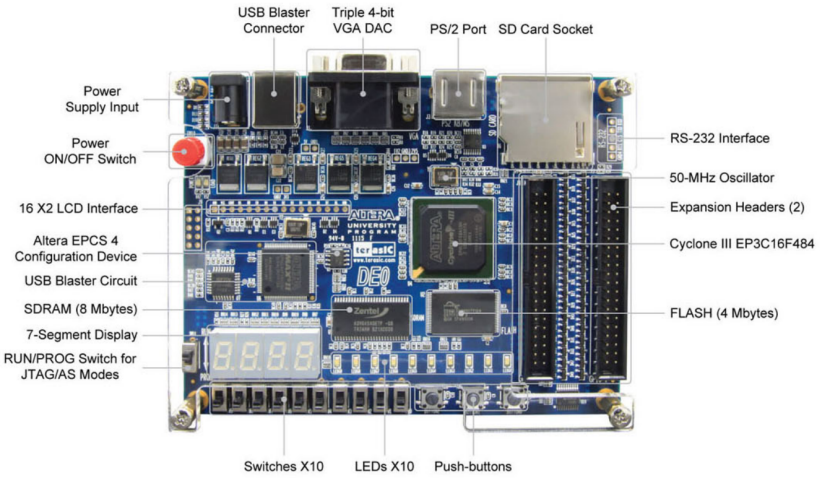

Figura 7.2: FPGA DE0 utilizada, con la identificación de sus componentes principales. Extraída de [23].

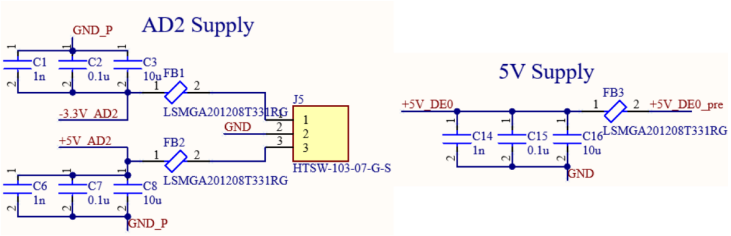

Figura 7.3: Circuito implementado para las fuentes de entrada. **J5** es un jumper de tres pines.

Para obtener estas tensiones, se usaron reguladores de voltaje lineales de baja caída (LDO). Estos generan tensiones con buena precisión y con bajo nivel de ruido intrínseco. Además, se agregaron estos componentes con el objetivo de aislar el sistema de fuentes de ruido externo, tales como el ruido de línea de las fuentes convencionales. Partiendo de la idea de generar un entorno que sea totalmente independiente de fuentes externas, se diseñó el sistema para que todas las tensiones se obtengan a partir de la tensión que provee la DE0, la cual es 5 V. También se dejó la posibilidad de suministrar estos 5 V desde una fuente externa. La tensión resultante de la elección entre estas dos alimentaciones se denominó +5V.

Inicialmente se creó un circuito con el objetivo de estabilizar y filtrar lo más posible las entradas de alimentación, tanto de la DE0 como de la fuente externa. Para ello, se utilizó el circuito de la figura 7.3. A partir de este circuito, implementado en cada fuente de entrada, se obtienen los nodos -3.3V_AD2, +5V_AD2 y +5V_DE0.

Está compuesto por una ferrita en la entrada, que suprime ruido de alta frecuencia, actuando como filtro pasabajos sin producir una atenuación de tensión significativa. Luego hay una serie de condensadores en paralelo, de diferente capacidad, con el objetivo de filtrar ruido en diferentes rangos de frecuencia y darle estabilidad a la señal en transiciones tanto en frecuencias rápidas como lentas.

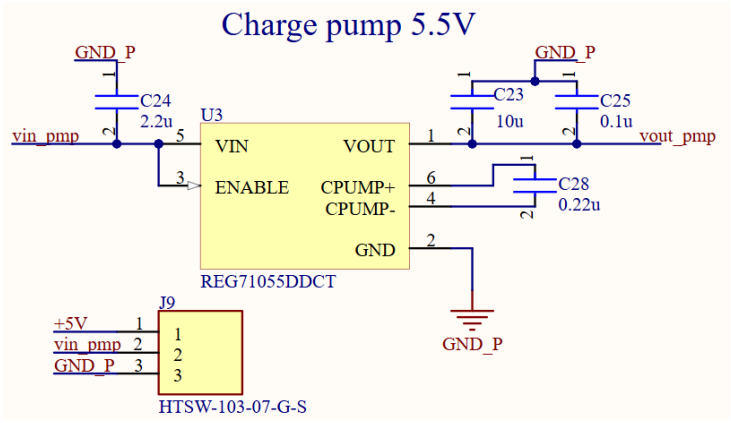

Figura 7.4: Circuito implementado para el conversor de tensión. El elemento **J9** es un jumper de 3 pines, que actúa como llave de encendido del conversor.

A la hora de generar las alimentaciones requeridas por el sistema, primero se partió de la hipótesis de que la DE0 no entregará una tensión ideal constante de 5 V. Por lo tanto, para lograr una tensión de 5 V a partir de un LDO, se necesita una alimentación mayor a 5 V. Por este motivo se eligió utilizar un elevador de tensión conmutado, del tipo Buck/Boost para obtener una tensión de 5,5 V. El conversor utilizado fue un REG71055DDCT de Texas Instruments [24]. El motivo de su elección fue su amplia tensión de entrada de 1,8 V a 5,5 V, haciendo el sistema robusto ante caídas o variaciones de tensión en la DE0. Además, puede entregar hasta 60 mA, cubriendo el consumo estimado del sistema de 30 mA (como se vio en la sección 7.4). El circuito utilizado se presenta en la figura 7.4, donde el jumper **J9** actúa como llave de encendido del conversor, o de ser necesario, como entrada externa de alimentación para este. Si no se desea usar el conversor, se puede colocar su entrada directamente a tierra.

A partir de la salida de 5,5 V del conversor conmutado, se es posible alimentar un LDO de 5 V, de forma de obtener una alimentación estable y lo más libre de interferencias posible. El LDO elegido para este caso fue el NCP160BMX280TBG de Onsemi [25], diseñado para suministrar 5,14 V. Se eligió este valor de tensión, levemente superior a los 5 V, con el objetivo de maximizar la excursión de los amplificadores, debido al output swing, principalmente del amplificador diferencial, el amplificador de entrada de la *Etapa 2* y de los amplificadores del filtro Butterworth (ver los rangos de tensiones en la sección 7.1). Este LDO soporta la tensión de alimentación de entrada de 5,5 V, tiene alta PSRR (98 dB), bajo consumo de corriente estática y bajo ruido. Además, puede suministrar corrientes de hasta 250 mA. El circuito implementado de este componente se ilustra en la figura 7.5, el cual solo requiere de condensadores en su entrada y salida con el objetivo de mantener estables esos nodos. Con este conjunto de conversor y referencia de tensión, se obtuvo una fuente de 5,14 V. En caso de no querer utilizar esta fuente, se puede alimentar el sistema directamente desde los 5 V de la DE0. Más adelante se explica la selección de fuentes de alimentación del sistema.

La tensión positiva restante a generar es la de 3,3 V. En este caso, puede utilizar-

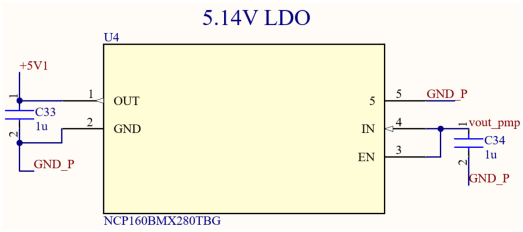

Figura 7.5: Circuito implementado para el LDO de 5,14 V.

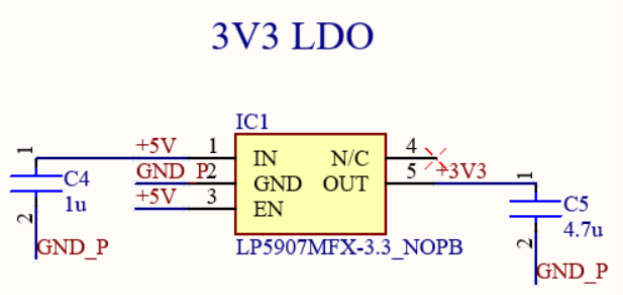

Figura 7.6: Circuito implementado para el LDO de 3,3 V.

se directamente el nodo +5V para alimentar al regulador que genere esta tensión. El LDO elegido fue un LP5907MFX-3.3 de Texas Instruments [26]. El mismo puede suministrar hasta 250 mA de corriente, posee muy bajo ruido de salida, alta PSRR (82 dB) y bajo consumo estático. El circuito implementado para este componente se describe en la figura 7.6. Los componentes que requiere son mínimos, solo necesita de condensadores para estabilizar su entrada y salida. La salida de este regulador se denominó +3.3_vref.

Con las fuentes de alimentación positivas determinadas, es necesario agregar las referencias de tensión, tanto para el DAC, que necesita una referencia de 2,5 V como para el ADC, que utiliza una referencia de 5 V.

Para implementarlas, se eligió el integrado MAX6072AAUB50 de Analog Devices [27]. Es una referencia de doble salida, suministrando 2,5 V y 5 V de forma simultánea. Esta elección permitió reducir la complejidad del diseño al evitar el uso de dos referencias independientes, además de minimizar el área ocupada en la PCB. Cada salida puede suministrar hasta 10 mA, cumpliendo el valor necesario de consumo por los conversores en sus entradas de referencia de tensión (200 µA del ADC y 2 mA para el DAC). Este integrado además es de bajo consumo estático (150 µA). El circuito implementado se observa en la figura 7.7. El integrado se alimenta desde el conversor de tensión de 5,5 V y posee dos jumpers, uno para cada referencia, que habilita su funcionamiento o las deja apagadas. Esta funcionalidad se implementó en caso de producirse fallos, o en caso de que se desee utilizar una referencia externa.

Para generar la alimentación de −3,3 V y −5 V, es necesario utilizar un conversor conmutado inversor, de forma de obtener una tensión negativa a partir de los +5V. Un inconveniente de este tipo de conversores es la conmutación inherente a

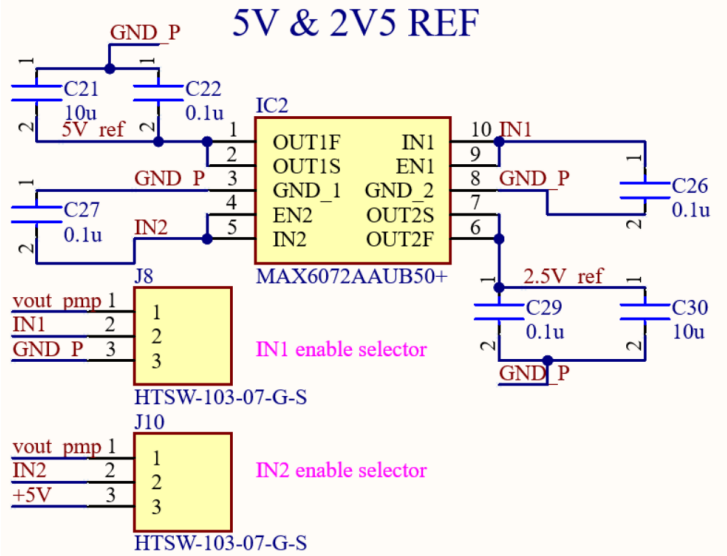

Figura 7.7: Circuito implementado para las referencias de 2,5 V y 5 V. **J8** y **J10** son jumpers de 3 pines que habilitan el funcionamiento de cada referencia individualmente.

su funcionamiento, introduciendo un nivel de ruido significativo en su salida, siendo contraproducente para la aplicación del sistema.

Una opción viable es utilizar un LDO alimentado por un convertidor conmutado, para filtrar el ruido presente en la salida de alimentación. Como se necesitan tensiones de $-3{,}3\,\text{V}$ y $-5\,\text{V}$, el convertidor debe entregar una tensión inferior a $-5\,\text{V}$, y además deben emplearse reguladores que soporten dicha tensión de entrada. Sin embargo, no se encontró un LDO que pudiera entregar $-5\,\text{V}$ cumpliendo con los requisitos de simplicidad y bajo ruido exigidos por el sistema. Por este motivo, se optó por generar los $-5\,\text{V}$ mediante un inversor conmutado, filtrando su salida, y utilizar esa misma tensión para alimentar un LDO que entregue los $-3{,}3\,\text{V}$ requeridos.

El inversor conmutado elegido fue un LTC1983ES6-5#TRPBF de Linear Technology [28]. Es un inversor del tipo charge pump. El circuito utilizado, junto con el filtro implementado, se observa en la figura 7.8. El integrado requiere solamente un capacitor de entrada (C11) para estabilizar ese nodo (vin_npmp) y otro condensador actuando como flying capacitor del charge pump (C17). A su salida se conectó un filtro pasivo tipo π (LC pasabajo), con el objetivo de atenuar el ruido de conmutación presente en la línea. Se consideró que era la mejor arquitectura a utilizar dado que no requiere componentes resistivos, lo que produciría caída de tensión en la salida. Este filtro tiene un polo $f_c = \frac{1}{2\pi\sqrt{LC}}$, donde sustituyendo con $C = 20\,\mu\text{F}$ y $L = 4{,}7\,\mu\text{H}$ se obtiene que el polo es de $16{,}4\,\text{KHz}$. Este polo se consideró aceptable, dado que la frecuencia del ripple de salida del conversor es de aproximadamente $300\,\text{KHz}$ [29]. La salida filtrada de este conversor se denominó -5V_filt en la figura 7.8.

Usando la salida filtrada del inversor conmutado, se conectó un LDO para obtener la alimentación de $-3{,}3\,\text{V}$. El regulador elegido fue un ADP7185ACPZN3.3

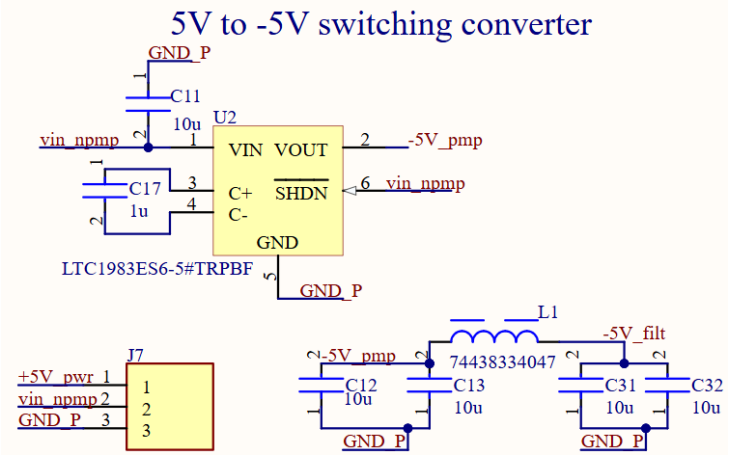

Figura 7.8: Circuito implementado del inversor conmutado junto con su filtro de salida. **J7** es un jumper de 3 pines que conecta la entrada del conversor con la alimentación o la mantiene apagada, fijando el nodo a tierra.

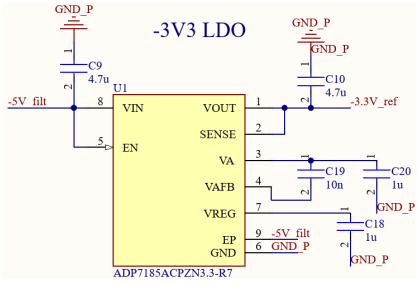

Figura 7.9: Circuito implementado del regulador lineal de $-3{,}3\,\mathrm{V}$. Este es alimentado a partir del charge pump de $-5\,\mathrm{V}$.

de Analog Devices [30]. El mismo cuenta con alta PSRR (68 dB) bajo nivel de ruido y puede entregar corrientes de hasta 500 mA. El circuito utilizado se observa en la figura 7.9, donde la salida de este regulador se denominó -3.3V_ref

Con todas las fuentes generadas, se cubren las distintas necesidades de alimentación del sistema CANICs (fuentes de $5\,\mathrm{V}$, $3{,}3\,\mathrm{V}$, $-3{,}3\,\mathrm{V}$ y $-5\,\mathrm{V}$ y las referencias de $5\,\mathrm{V}$ y $2{,}5\,\mathrm{V}$). Para prevenir posibles fallas y permitir el uso alternativo de distintas fuentes de alimentación, se incorporaron nodos de tensión redundantes en el diseño. Para elegir entre estas alimentaciones, se agregaron jumpers para cada nivel de alimentación. El circuito de estos conectores se detalla en la figura 7.10. El pin central (pin 2) corresponde a la salida de alimentación hacia el sistema, mientras que los pines extremos permiten seleccionar la fuente de alimentación en cada caso. Estas pueden provenir de los circuitos detallados previamente o de los nodos de fuente externa filtrados, como se explicó anteriormente utilizando la figura 7.3.

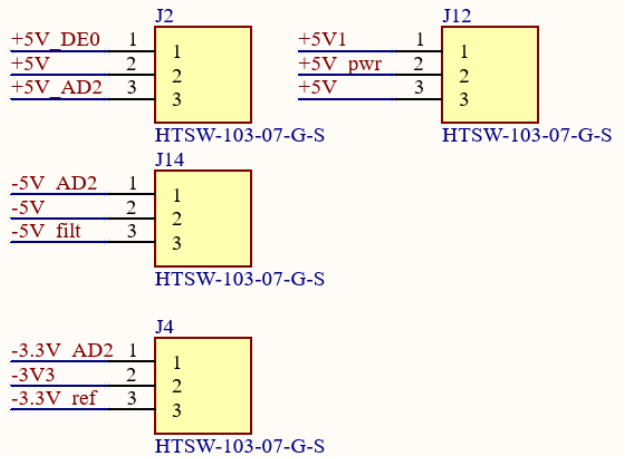

Figura 7.10: Jumpers de selección del origen de los diferentes niveles de alimentación. El pin central es la salida. Dada la composición de tres pines, se puede además utilizar el pin central como nodo de entrada para una fuente externa.

7.7. Circuitos principales de la PCB

En las siguientes secciones se explicarán en detalle los diferentes circuitos que componen la PCB, ilustrados en la figura 7.1, junto con la placa de desarrollo utilizada y la interconexión entre estos elementos, a modo de explicar el entorno de prueba utilizado.

Se presentará la descripción del hardware siguiendo el flujo de la señal SENSADO a lo largo de los distintos bloques funcionales del sistema, desde su entrada hasta su digitalización.

7.7.1. Atenuador

Este bloque se encuentra en la entrada de señal de la PCB, como se ve en la figura 7.1. El mismo surge como la solución a un problema debido al generador utilizado y no es parte del sistema CANICs. Por esa misma razón, se implementó la posibilidad de poder evitar el uso del mismo a través de jumpers en caso de contar con otra forma de imponer las señales de entrada.

Como se explicó en el capítulo 2, el setup para validar el sistema tendrá como entrada las señales impuestas desde un generador de señales. El problema que podría darse es que la ECAP tuviese una amplitud menor al LSB del DAC que posee el generador de señales para una amplitud dada de la SA.

La solución propuesta fue realizar un circuito intermedio que atenúe las señales provistas por el generador de señales. Al buscar que la solución no afecte el rendimiento del sistema CANICs, se optó por un amplificador fully differential [31]. Su implementación se observa en el circuito de la imagen 7.11, la ganancia del mismo es de $G = 0{,}23\,\mathrm{V/V}$ y tiene una respuesta pasa bajos con una frecuencia de corte $F_C = 100\,\mathrm{KHz}$. Con esta ganancia se logra maximizar el rango de amplitudes

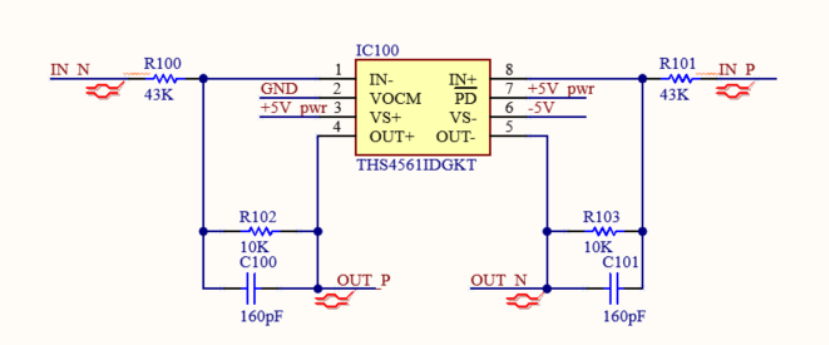

Figura 7.11: Circuito que implementa el atenuador full diferencial a la entrada.

utilizado por el generador de señales, y proveer un rango de tensiones adecuado a los siguientes componentes de la PCB, de modo que tengan la excursión esperada para que no saturen. El desarrollo más en profundidad de la implementación de este circuito se ve en el apéndice A.2.

7.7.2. Amplificador de Instrumentación (INA)

Este bloque es el primer bloque de hardware que utiliza la *Etapa 1*. Su objetivo es convertir la señal diferencial de entrada del sistema (V_{in})) a una señal single-ended (SENSADO) y suprimir las componentes en modo común presentes, como se detalló en la sección 3.3.

En la figura 7.12 se puede observar que la alimentación del amplificador es simétrica de ±3,3 V, lo que, como se explicó antes, permite que el output swing del operacional sea mayor a los 2,5 V, que es el máximo en el que va a excursionar la señal de entrada, restricción dada por las tensiones de referencia del ADC y DAC, las cuales establecen el fondo de escala.

7.7.3. Schmitt Trigger y Rectificador

Este bloque genera la señal de sincronismo **Trigg** del diagrama 7.1. Esta señal se genera a partir de un nivel de tensión de la entrada SENSADO. El pulso obtenido se rectifica y provee una señal en la excursión tolerada por el sistema digital, que luego utiliza para sincronizarse con las respuestas a los estímulos recibidos en la entrada (la señal SENSADO). El circuito que logra esto es el que se muestra en la figura 7.13, se explicó en detalle en la sección 3.4.

Los niveles de conmutación propios del circuito y las alimentaciones también fueron explicados en profundidad en la sección 3.4. Análogo al INA, este amplificador fue alimentado con ±3,3 V, de forma de maximizar el output swing de la señal de entrada.

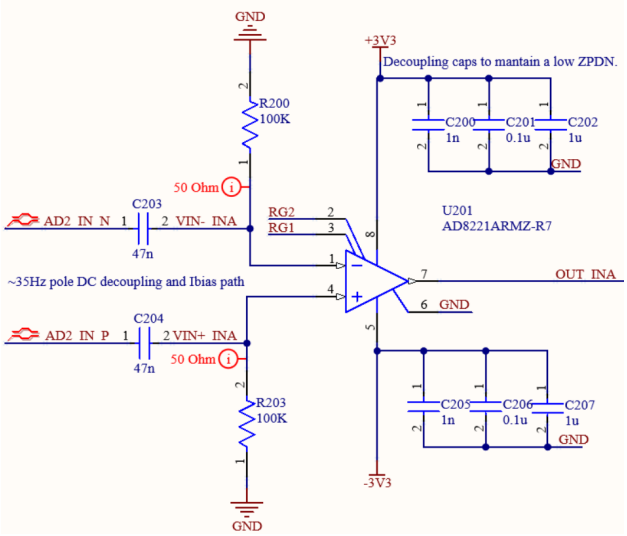

Figura 7.12: Amplificador de instrumentación de entrada.

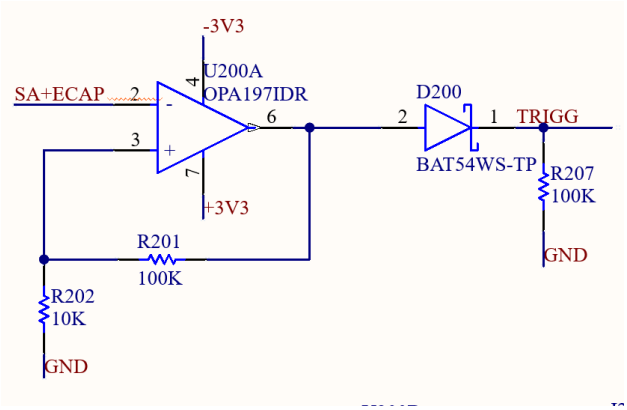

Figura 7.13: Circuito implementado del Schmitt-Trigger y el rectificador.

7.7.4. Amplificador Diferencial

El circuito de la imagen 7.14 es el que implementa la resta de la señal de entrada con el template reconstruido por el DAC de la *Etapa 1*, la explicación del HW se detalló en la sección 3.6. Algo a destacar de la implementación de este circuito es que se utilizó un array de resistencias para que las mismas tengan un error relativo menor comparado al utilizar resistencias discretas. Esto se debe a que un alto $CMRR$ es crucial para un buen desempeño del amplificador diferencial, de forma que elimine las componentes de modo común de las señales entrantes, que son en su mayoría ruido que ingresaría al sistema.

La elección para la alimentación de este amplificador se eligió con el objetivo de maximizar el output swing (OSW). De esta forma, la salida del amplificador puede excursionar entre $[5V\_pwr - OSW, -3{,}3\,\mathrm{V} + OSW]$, donde $5V\_pwr$ puede valer 5 V o 5,14 V (jumper J12 de la figura 7.10). Esto maximiza la excursión posible

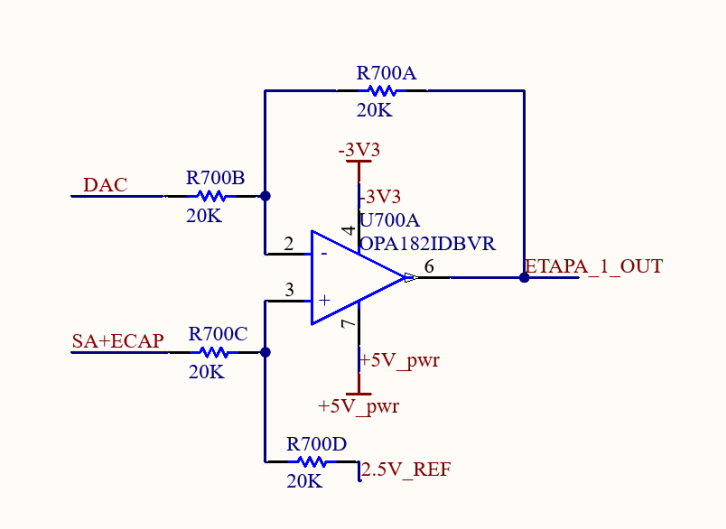

Figura 7.14: Circuito que implementa el amplificador diferencial.

de la señal de salida del amplificador, la cual debe estar entre $[5\,\mathrm{V}, 0\,\mathrm{V}]$, ya que el ADC necesita que las tensiones de entrada de sus canales se encuentren en este rango de tensión. La elección ideal es utilizar la fuente de 5,14 V, maximizando el rango superior de tensión a la salida del amplificador.

7.7.5. Amplificador de entrada de la *Etapa 2*

El amplificador representado en el diagrama 7.1 es el amplificador descrito en la *Etapa 2* en la sección 4.2. Se realizó un amplificador no inversor tal como se ve en la figura 7.15. Este bloque permite amplificar la salida de la *Etapa 1*, residuo más ECAP, con una ganancia ajustable. También conserva la componente de continua sobre el que se encuentra montada la señal proveniente de la *Etapa 1* (2,5 V). Esto resulta indispensable para el correcto funcionamiento del ADC, dado que su canal de entrada no tolera niveles de tensión negativos, tal como se mencionó anteriormente.

Análogo al caso del amplificador diferencial, como su señal de salida tiene que excursionar en el rango de $[5\,\mathrm{V}, 0\,\mathrm{V}]$, se alimentó este amplificador con las tensiones de $[5V\_pwr, -3{,}3\,\mathrm{V}]$, con el objetivo de maximizar el output swing.

7.7.6. Conversor Analógico-Digital

El ADC elegido fue el MAX1266 [19]. Posee una interfaz de datos paralelo con resolución de 12 bits, capaz de medir señales de forma bipolar y tiene más de un canal de entrada. Debido a la escasez de este tipo de conversores en el mercado, el mismo presenta ciertas características subóptimas que condicionaron el desarrollo de la PCB, entre las que se encuentran:

- Nivel de voltaje $V_{COM} = 2{,}5\,\mathrm{V}$.

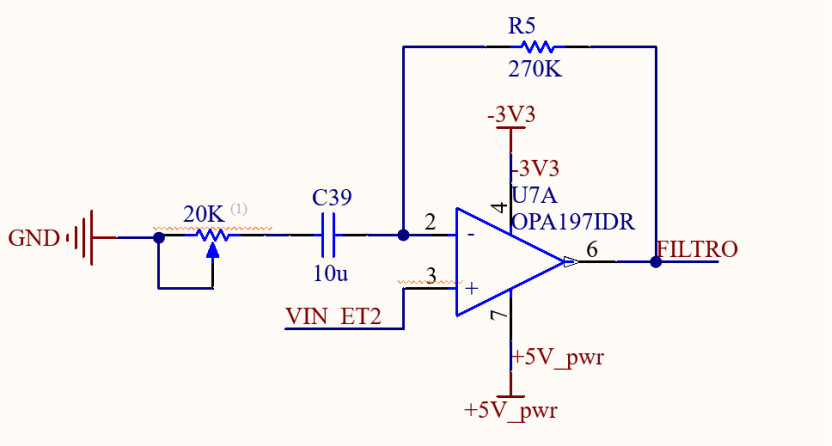

Figura 7.15: Circuito implementado del Amplificador de ganancia variable.

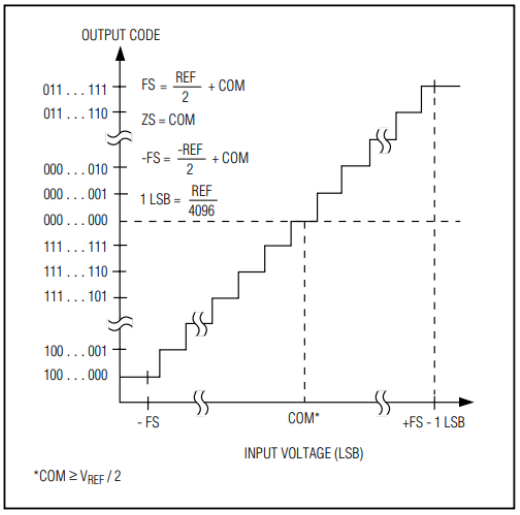

Figura 7.16: Imagen donde se detalla el valor de V_{COM}. El manual especifica que debe ser $V_{COM} \geq \frac{V_{REF}}{2}$, donde $V_{REF} = 5\,\text{V}$. Extraída de [19].

- Bus de datos bidireccional triestado.
- Nivel lógico de señales digitales $= 5\,\text{V}$.

El ADC funciona con alimentación de 5 V, en consecuencia sus niveles lógicos también tienen este nivel de tensión. La característica particular de este ADC, que aunque posee rango de representación bipolar de los datos muestreados, necesita que su referencia de tensión analógica V_{COM} esté a un voltaje de bias de al menos 2,5 V, como se especifica en la sección 7.16. Fijando esta tensión de bias en 2,5 V se maximiza el rango de representación del ADC. Los diferentes circuitos de la PCB se acondicionaron de forma adecuada para cambiar el voltaje de continua y adaptarse a esta condición, como se explicaron en las secciones previas correspondientes.

Bus de datos bidireccional

El ADC posee distintos modos de uso dependiendo del tipo de lectura, como por ejemplo, utilizar el canal uno o dos, utilizar el ADC en modo bipolar, utilizar referencia interna o externa, entre otros. Para ello, utiliza palabras de control, que lo configuran en un determinado modo de funcionamiento. Esto debe hacerse previo a cada muestreo de datos. El ADC está construido de forma tal que el bus de datos es bidireccional, permitiendo utilizar este bus para configurar el modo de funcionamiento y obtener el dato muestreado.

Para manejar la bidireccionalidad del bus de datos entre el ADC y el *Controlador del ADC* (de aquí en adelante nombrado FPGA), se utilizaron buffers triestado de la familia 74AHC9541A de Nexperia [20]. En la figura 7.17 se observa la conexión realizada entre los mismos junto con el ADC y la FPGA. El *Buffer 1* está encargado de imponer la palabra de configuración *config* (proveniente de conectarlo directamente a niveles lógicos fijos) al ADC, mientras que el *Buffer 2* está encargado de imponer a la FPGA el dato muestreado por el ADC. El comportamiento temporal generado por estos buffers se ilustra en la figura 7.18, donde **adc_bus** ilustra el comportamiento del bus de datos del ADC y **fpga_bus** ilustra el comportamiento del bus de datos de la FPGA. A modo de referencia se muestra la señal del ADC **wr_n**, utilizada para iniciar una conversión de datos, e **int_n** utilizada como señal de reconocimiento para avisar que el ADC tiene un nuevo dato **data_in**.

Cuando el *Controlador del ADC* inicia un nuevo ciclo de muestreo mediante un pulso en la señal **wr_n**, se espera que el *Buffer 1* esté habilitado, permitiendo que la palabra de configuración *config* sea impuesta sobre el **adc_bus**. Simultáneamente, el *Buffer 2* debe encontrarse en alta impedancia para asegurar que el **fpga_bus** esté eléctricamente desacoplado. Una vez que el ADC indica que el dato muestreado **data_in** está disponible, se requiere que el *Buffer 1* haya sido deshabilitado para evitar conflictos en el bus, y que el *Buffer 2* esté activo para transferir el dato hacia la FPGA. Las señales digitales involucradas para controlar a los buffers se encuentran detalladas en el apéndice B.2.

Dado que el ADC se emplea exclusivamente en un único modo de operación (esto es, *internal acquisition mode* con entradas bipolares, como se detalló en la sección 3.8.3), la palabra de configuración digital (*config*) requerida para iniciar una nueva adquisición con el ADC permanece constante en cada ciclo de muestreo. En consecuencia, dicha palabra de configuración puede fijarse por hardware simplificando la lógica de control. Esto permite que, en cada ciclo de escritura de la palabra de configuración, el ADC reciba siempre el mismo patrón de configuración. Por este motivo, las entradas del *Buffer 1* (correspondientes a la palabra de configuración) se conectaron directamente a niveles lógicos fijos (V_{DD} o GND), de manera que el patrón digital requerido para establecer el modo de operación del ADC permanezca constante.

No obstante, se reservó el bit correspondiente a la selección del canal analógico de entrada (bit A0 de *config*), permitiendo que sea controlado dinámicamente por *Control principal*. De esta manera, se puede seleccionar entre el canal 0 o canal 1 según se desee trabajar con la *Etapa 1* o la *Etapa 2* respectivamente.

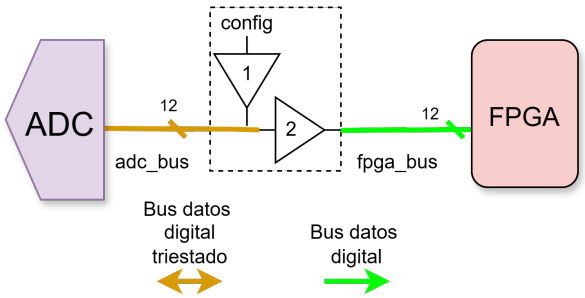

Figura 7.17: Conexión de el *Buffer 1* y *Buffer 2* representando el flujo de datos entre el ADC y la FPGA.

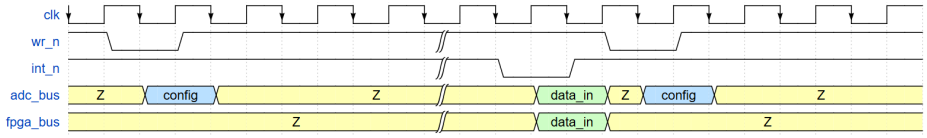

Figura 7.18: Comportamiento temporal impuesto por los buffers entre el ADC y la FPGA.

Nivel lógico de señales digitales = 5 V

El ADC opera con niveles lógicos cercanos a 5 V, mientras que la FPGA utilizada tiene un rango máximo de tolerancia en sus entradas y salidas de 3,3 V. Por esto surge la necesidad de traducir los distintos niveles lógicos de los componentes. Se resolvió utilizar level shifters, capaces de hacer traducciones de forma bidireccional de los 5 V a los 3,3 V. A continuación se explica el circuito utilizado para hacer la conversión de niveles lógicos de las señales.

7.7.7. Level Shifter

Como se explicó previamente, la salida digital del ADC opera con niveles de 5 V, superando el umbral máximo de 3,3 V tolerado por la FPGA. Por esta razón, se incorporaron circuitos de adaptación de nivel (level shifters) para asegurar la compatibilidad eléctrica entre ambos dispositivos. Estos se encargan de hacer la traducción de los niveles lógicos. Como características fundamentales, deben de poder soportar la conversión a las frecuencias de las señales, e introducir bajo retardo y jitter en las mismas.

Para ello, el integrado elegido fue el LSF0108 de Texas Instrument [32], siendo un traductor de nivel lógico bidireccional de 8 canales. Las señales que requieren este nivel de conversión son el bus de datos del ADC, que son 12 bits, las señales de control del ADC, las cuales son 3 bits, las señales de control del buffer, que son 2 bits (ver apéndice B.2), y la señal de reloj del ADC. En total es necesario convertir de nivel 18 bits. La conversión de la señal de reloj es un aspecto crítico, debido a que es la más rápida del sistema y la más sensible respecto a ruido jitter. El level shifter elegido soporta frecuencias de conversión de hasta 100 MHz y tiene tiempos

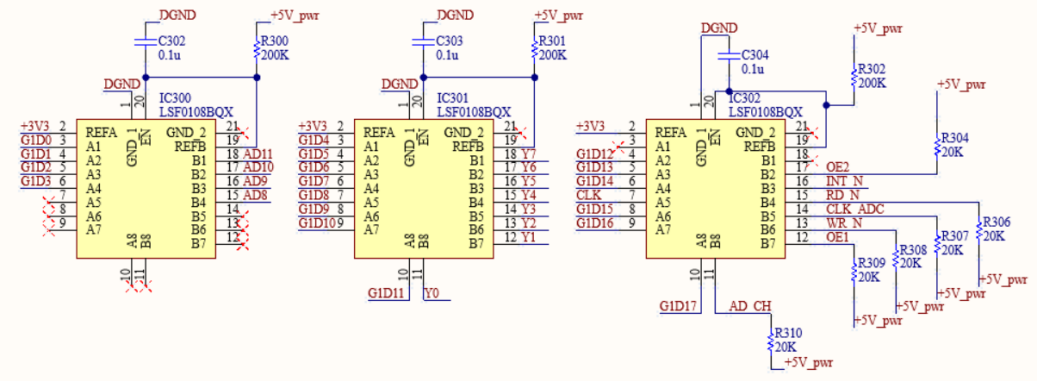

Figura 7.19: Circuito esquemático implementado en la PCB para la conexión de los level shifters entre el ADC y la FPGA.

de retardo del orden de ns, con lo cual soporta holgadamente los requerimientos de esta señal ($f_{CLK} = 4{,}5\,\text{MHz}$). El circuito implementado se observa en la figura 7.19, donde se observa que fueron necesarios 3 level shifters para cubrir la conversión de todas las señales. El integrado para que funcione correctamente sólo necesita resistencias de pull up a la referencia de mayor tensión.

Los rangos de voltaje que soporta el LSF0108 son de 0,95 V a 3,6 V en el lado A, y de 1,8 V a 5,5 V en el lado B, utilizando referencias externas, lo que permite cumplir con los niveles de tensión requeridos para la correcta comunicación entre el ADC y la FPGA. Además, opera sin necesidad de una señal de dirección para definir el sentido de transferencia, simplificando notablemente la implementación del sistema.

7.7.8. Conversor Digital-Analógico

El DAC elegido fue el AD5445 [21], consta de una arquitectura de datos paralela, resolución de 12 bits y es capaz de operar en forma bipolar. Una característica que posee es que su salida es en corriente, por lo que precisa un circuito externo para acondicionar su salida en los niveles de tensión esperados, circuito que se implementó en la PCB.

Circuito de acondicionamiento del DAC

El esquemático implementado en la PCB de este circuito se representa en la figura 7.20, siendo el circuito implementado equivalente al que se mostró en la figura 3.16 del capítulo 3, donde se explicó el funcionamiento en detalle. El elemento R403 conectado a V_{REF} es un potenciómetro, usado para ajustar la tensión de referencia de entrada 2.5V_ref. Los condensadores C401 y C402 se agregaron con el objetivo de estabilizar la alimentación, actuando además como filtro para ruido de alta frecuencia.

Como amplificadores se utilizó el integrado OPA2863 [33], el cual consta de dos amplificadores rail to rail, con un slew rate de 105 V/µs, bajo consumo y ruido.

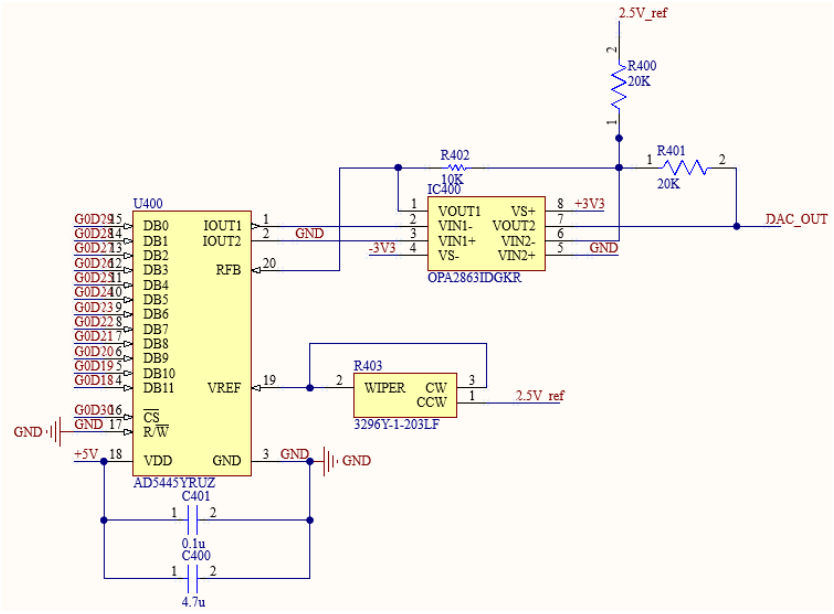

Figura 7.20: Circuito esquemático implementado en la PCB que incluye el DAC y su circuito de acondicionamiento de la señal de salida DAC_OUT, que transforma la salida del DAC en corriente (IOUT1, IOUT2), en la tensión de salida DAC_OUT.

El aspecto crítico de estos amplificadores, además del bajo ruido, es el slew rate. Como el DAC genera pulsos escalonados en su salida debido al cambio de los switches, es necesario que los amplificadores presenten una alta velocidad de respuesta dinámica, de modo que puedan seguir adecuadamente las variaciones rápidas en la señal de entrada sin introducir un retardo significativo. Los $105\,\mathrm{V/\mu s}$ equivalen a que en un período de CLK ($T_{CLK} \approx 222\,\mathrm{ns}$) los amplificadores pueden soportar un cambio de $\approx 23\,\mathrm{V}$, el cual supera ampliamente el rango de tensión utilizado en el sistema, por lo que el tiempo de respuesta, inferior a $1\,\mathrm{T_{CLK}}$, es considerado aceptable.

Otro aspecto no menor es que los amplificadores son rail to rail, por lo que no hay inconvenientes al generar valores de señal en los extremos de excursión. De cualquier forma, los operacionales están alimentados con $\pm 3{,}3\,\mathrm{V}$ y la señal de salida tiene una excursión de $\pm 2{,}5\,\mathrm{V}$, por lo que la excursión a la salida es efectiva en todo el rango disponible.

El motivo de usar como alimentación $\pm 3{,}3\,\mathrm{V}$ en estos amplificadores es para tener efectivamente el rango de excursión completo de señal, y por el lado del HW, hubiera sido necesario agregar exclusivamente para estos amplificadores reguladores de $\pm 2{,}5\,\mathrm{V}$, por lo que no se consideró práctico en términos de costos y complejidad de la PCB.

7.7.9. Filtros realizados

Los filtros realizados de suavizado y antialias RC de la *Etapa 1*, que se esquematizan en la figura 7.1 se describen en profundidad en las secciones 3.12 y 3.7 respectivamente, del capítulo 3. Los filtros antialiasing, compuestos por un RC y Butterworth, en la *Etapa 2* se describen en la sección 4.3 del capítulo 4. Los filtros pasivos no cambian los niveles de excursión de las señales.

El único filtro activo utilizado en el sistema es el Butterworth, implementado con una arquitectura Sallen-Key. Está compuesto por dos amplificadores operacionales OPA2863 [33], tal como se detalló en la sección 4.3. Análogo al caso del amplificador de ganancia variable, como su señal de salida tiene que excursionar en el rango de $[5\,\mathrm{V}, 0\,\mathrm{V}]$ (necesario para adaptar la señal al rango admitido por el ADC, que no permite niveles negativos en su entrada), se alimentaron estos amplificadores con las tensiones de $[5V\_pwr, -3{,}3\,\mathrm{V}]$, con el objetivo de maximizar el output swing.

7.8. Diseño general de la PCB

Para el diseño de la PCB, se utilizó el software de desarrollo Altium Designer, siendo una herramienta que permite realizar el diseño integral de una PCB, desde los esquemáticos, implementar la disposición de los distintos componentes con sus respectivos empaquetados y pines, junto con el ruteo entre los diferentes componentes. También permite hacer verificación, tanto de la integridad de los esquemáticos como de chequeos de ruteo vs. esquemático, siendo cruciales para evitar errores en la implementación.

A la hora de realizar la PCB, se tuvo en cuenta principalmente que el diseño fuera portable y de fácil conexión a la placa de desarrollo utilizada, con el objetivo de generar un sistema DE0 - PCB autosuficiente. Se buscó que fuera libre de cableado entre estos elementos y sin la necesidad de requerir fuentes externas. Además, dado que el sistema maneja señales de pequeña magnitud de tensión, se centró fuertemente el diseño en la reducción del ruido, optimizando el ruteo y ubicación de componentes. Finalmente, dado que la PCB es para probar el prototipo del sistema desarrollado, debe poder medirse prácticamente todos los puntos de conexión de interés, y en algunos casos poder aislar etapas entre ellas con fines de debug, es decir, diseñar una PCB pensada para test.

Para minimizar el ruido parásito del ruteo, se optó por diseñar una arquitectura de PCB de cuatro capas: dos de señal, siendo estas las caras visibles de la PCB, y dos planos internos, uno de alimentación y otro de tierra. Como se mencionó anteriormente, hay varios niveles de tensión de alimentación, por lo que este plano de alimentación debe contemplar esta división de tensiones. A su vez, pensando en el tipo de componentes utilizados, algunos de los cuales son para procesamiento de señales analógicas sensibles, otros son conversores de tensión que conmutan constantemente y otros que manejan señales digitales de alta velocidad, se buscó la forma de aislar lo más posible entre sí estas tres familias de elementos en la PCB. La primera forma de lograrlo es ubicar los componentes de una misma familia lo

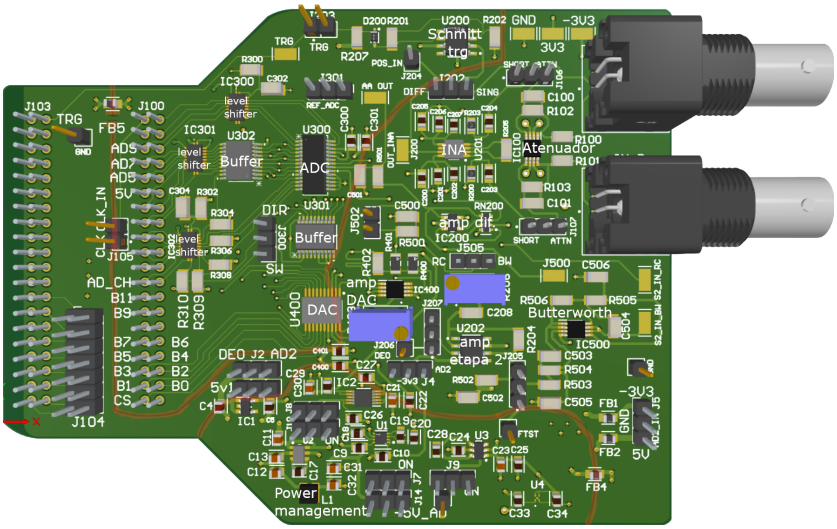

Figura 7.21: Vista superior de la PCB diseñada por CANICs con sus componentes. Se resalta en rojo la división de componentes por su tipo de funcionamiento. La inferior corresponde a la sección de conversores de tensión conmutados y fuentes de alimentación, la central a señales analógicas y la superior a señales digitales.

más cercanos posible entre sí. La figura 7.21 muestra una vista superior de la PCB junto con la separación de los componentes por familias, además de los nombres de los componentes que integran cada división.

Luego, a modo de aislar las familias entre ellas, se crearon diferentes zonas de tierra o 'planos', unidos entre ellos mediante un único punto de conexión para evitar lazos de tierra, utilizando ferritas o líneas de cobre en puntos específicos. La figura 7.22 muestra los tres planos de tierra creados. El plano conmutado está conectado a través de una ferrita al plano analógico, el cual es el que tiene la conexión de tierra global. El plano digital se conectó directo por un cable al plano analógico.

Para el plano de alimentación se siguió un razonamiento similar, generando planos de tensión siguiendo las divisiones de las familias de componentes. En este caso, sólo se aplicó un plano para la tensión de 5 V, la cual es la más utilizada por diferentes componentes. El resto de las alimentaciones se resolvieron utilizando ruteo normal entre nodos. El ruteo de las tensiones de $\pm 3{,}3$ V se definió hacerlo en capas de señal, debido a que se superponía en sobremanera con el plano de 5 V. El plano de alimentación obtenido se detalla en la figura 7.23.

Su dimensión final aproximada fue 11,5 cm $\times$ 9 cm. En la figura 7.21 se observa también la forma particular que tiene, en donde a la izquierda se angosta, con el objetivo de encastrar con el área que posee la DE0 para sus conectores de salida, como se muestra en la figura 7.24.

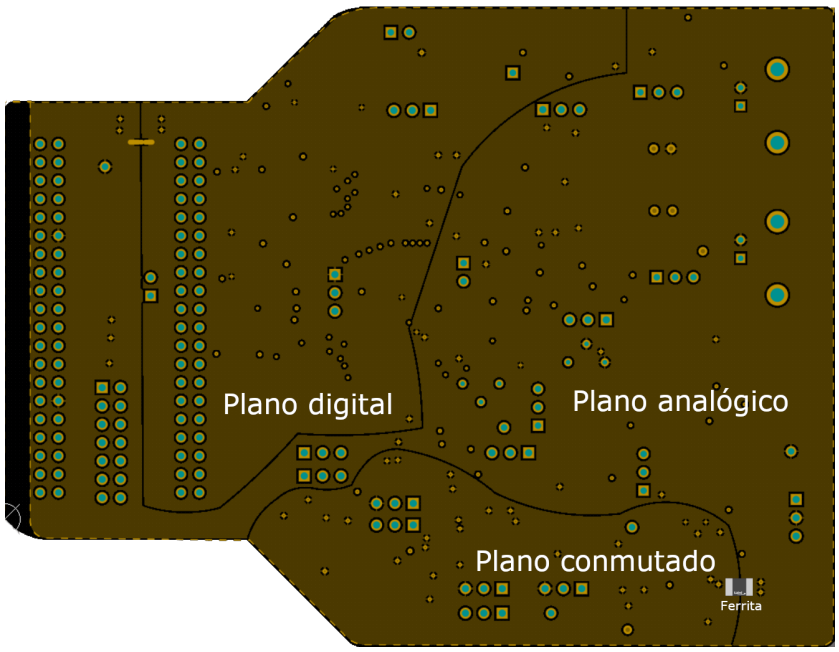

Figura 7.22: Plano de tierra de la PCB, mostrando las diferentes zonas creadas con el objetivo de evitar la interferencia entre ellas.

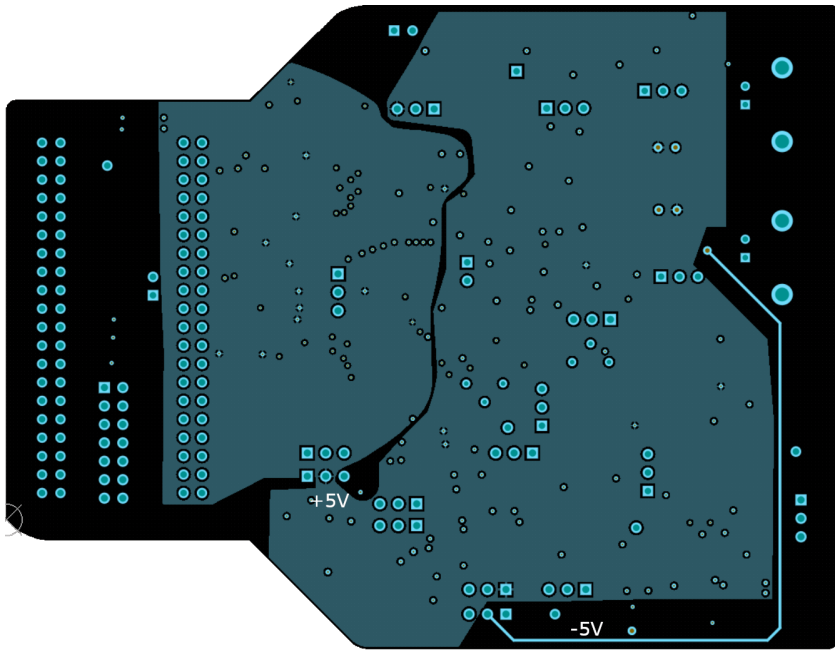

Figura 7.23: Plano de alimentación de la PCB. Se separó el plano de $5\,\mathrm{V}$ siguiendo la división por familia de componentes, a modo de evitar la interferencia y reducir el ruido.

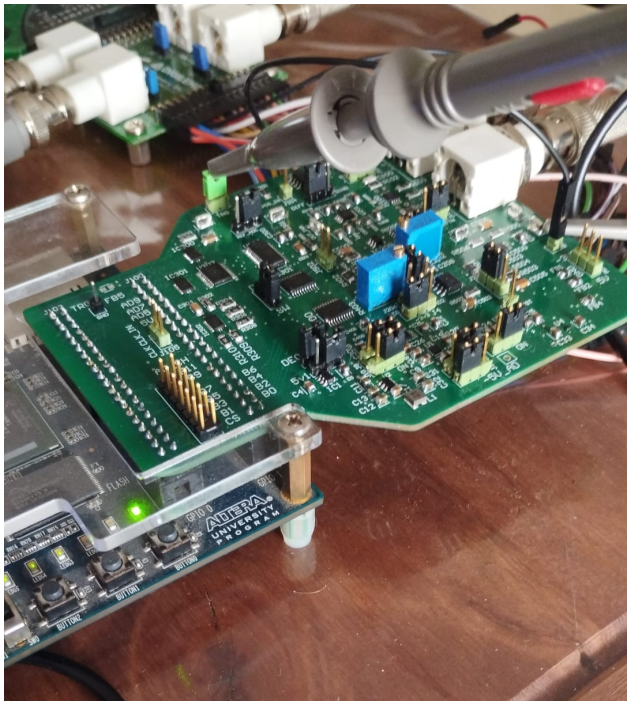

Figura 7.24: Montaje entre la PCB diseñada y la placa DE0 utilizada.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 8

Validación del Sistema

8.1. Introducción

El presente capítulo tiene como objetivo detallar el proceso de validación del sistema CANICs a nivel de hardware y verificar si el sistema cumple con las especificaciones definidas.

Previo a esta validación se establecerán métricas objetivas que permitan evaluar el rendimiento del sistema. Primero se busca definir qué señales obtenidas son de interés. Basándose en cálculos de correlación, comparándose con un modelo de señal ECAP objetivo, se determina si la señal obtenida por el sistema es efectivamente una ECAP.

Luego, utilizando las señales ECAP válidas obtenidas a la salida, se calcula la amplitud pico a pico de la ECAP a la entrada del sistema. Relevar la amplitud de la ECAP, como se expuso en el capítulo 1, constituye uno de los objetivos en el diseño de neuroestimuladores en lazo cerrado. Este parámetro permite inferir de manera indirecta el nivel de activación neural generado por el estímulo eléctrico. Su monitorización en tiempo real permite adaptar dinámicamente la corriente de estimulación frente a variaciones fisiológicas o de posición del electrodo. En este contexto, la capacidad del sistema neuroestimulador para detectar y cuantificar la amplitud de la ECAP se vuelve crítica para garantizar la eficacia y seguridad del tratamiento. Este es un parámetro clave para el control de neuroestimuladores en lazo cerrado.

Estas métricas se explicarán en detalle en la sección 8.5. Para la validación, se presentarán los resultados obtenidos a partir de las pruebas realizadas, y se analizarán dichos resultados contrastando con las métricas definidas.

8.2. Entorno de medidas utilizado

Las pruebas se realizaron con el entorno detallado en la figura 8.1. Dentro del bloque de entrada *WaveGen*, se le cargaron vectores realizados en Python al generador, conteniendo la suma de las señales SA más ECAP, especificadas en el

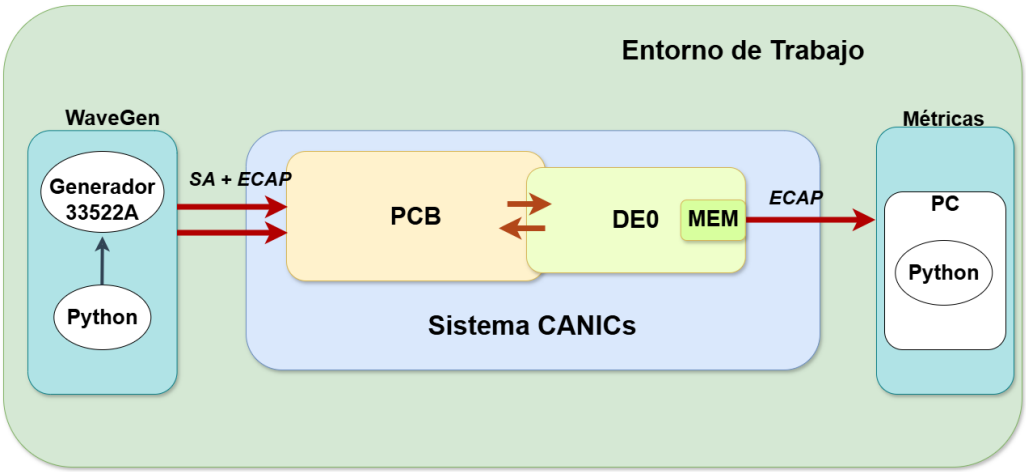

Figura 8.1: Diagrama del entorno de validación del sistema. Se distingue el módulo WaveGen, encargado de generar las señales biológicas; el sistema CANICs, representado por la interconexión entre la PCB diseñada y la placa DE0; y el módulo Métricas, responsable de calcular medidas objetivas sobre el desempeño del sistema.

capítulo 2. Se utilizó el generador de señales 33522A de Agilent [34] para imponer estas señales al sistema CANICs. La salida del sistema se guardó en la memoria de la DE0, para posteriormente obtenerla a nivel de software y poder determinar el desempeño. El entorno construido para efectuar las medidas se presenta en la figura 8.2, donde se observa la PCB CANICs junto con la DE0 y el generador utilizado imponiendo señales a la entrada del circuito. La salida del sistema, una vez finalizada la captura de datos, se envía a la PC a través de comunicación USB para ser procesada posteriormente por el bloque *Metricas*, a través de scripts en Python.

Como se mencionó en el capítulo 7, la PCB tiene varios modos de configuración, y fuentes de alimentación redundantes, por lo que se especificará el modo de uso que se utilizó para realizar las medidas. Primero, se utilizó la FPGA como fuente de alimentación principal (nodo +5V), al igual que como tensión de alimentación de 5 V para los componentes de la PCB (nodo +5V_pwr). Para el resto de tensiones de alimentación se usaron los reguladores de la PCB, dado que no se utilizaron fuentes externas, que era la otra alternativa de uso. Se utilizó el atenuador de entrada, por los motivos explicados en el capítulo 7. Respecto a los filtros, se utilizó el filtro de suavizado del DAC, y para la entrada de la *Etapa 2* se eligió el filtro Butterworth.

Respecto al reloj del sistema, se utilizó un generador de señales exclusivamente para este fin, el cual se sincronizó con el generador de las señales de entrada al sistema. Para hacer esto, se utilizaron conectores auxiliares que poseen los generadores, obteniendo que las señales (SA, ECAP y CLK) se generan a partir del mismo reloj base, asegurando su sincronismo. Este es un aspecto clave, dado que la frecuencia de reloj y la frecuencia de la señal de entrada deben ser múltiplos entre sí. De lo contrario, el sistema presenta una importante degradación de performance, debido a las variaciones de sincronismo de la señal entrante respecto a la frecuencia de

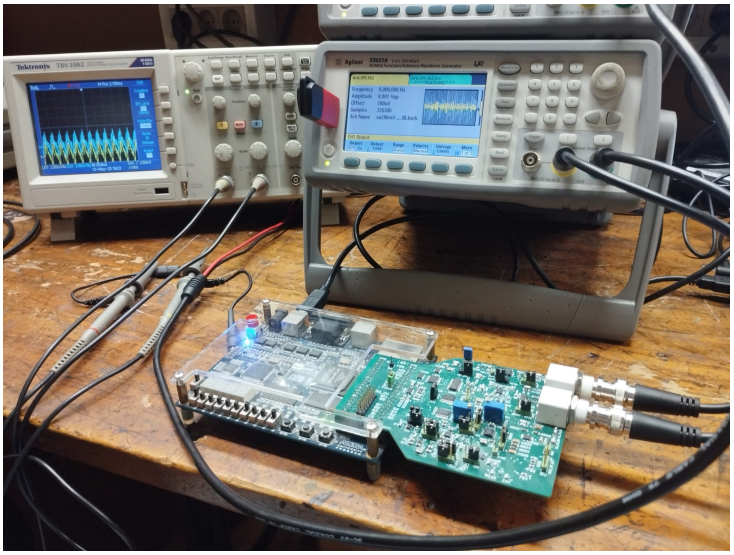

Figura 8.2: Montaje del setup de validación del sistema. Se muestra la PCB diseñada conectada a la placa DE0, con el generador de señales acoplado a la entrada. El osciloscopio, visible en segundo plano, permite monitorear las señales durante las pruebas.

procesamiento del sistema. Esta problemática se explicó en detalle en la sección 3.5. Por lo tanto, la FPGA se configuró para recibir señal de reloj externa a través de un pin específicamente diseñado de la PCB.

8.3. Utilización de la FPGA

Para programar la FPGA, se utiliza el software *Quartus* de Intel-Altera. Dentro de sus herramientas se encuentra el *Signal-Tap Logic Analyzer*, permitiendo observar y analizar el comportamiento de señales internas de circuitos de la FPGA en tiempo real, guardándolas en la memoria SDRAM de la DE0. Esta funcionalidad evita la necesidad de utilizar pines adicionales, dado que utiliza la comunicación USB para transferir los datos de la memoria al software de la PC. A la hora de utilizarlo, se configuró de forma de capturar las señales de salida de la *Etapa 2*, **ecap_available**, **ecap_sample_done** y el bus de datos **ecap**, siendo a su vez las señales de salida del sistema CANICs, como se mostró en el capítulo 4.

Debido a las limitaciones de memoria de la DE0 y dado que la señal ECAP solo está disponible luego de la fase de convergencia del sistema CANICs, la captura de datos se configuró para comenzar únicamente tras detectar un flanco de subida en la señal **ecap_sample_done**. Esto permite almacenar solo los datos relevantes, omitiendo la fase de generación de los templates y convergencia, optimizando así el uso del espacio disponible de memoria de la DE0.

La figura 8.3 muestra un ejemplo de captura de datos, donde se observan las tres señales registradas: **ecap_sample_done**, **ecap_done** (equivalente a **ecap_available**) y **ecap**. Se puede apreciar que la adquisición de la señal de

Figura 8.3: Captura de las señales **ecap**, **ecap_done** y **ecap_sample_done**. La adquisición se realiza en cada flanco de subida de **ecap_sample_done**, con un período de $19\,\mathrm{T_{CLK}}$.

interés **ecap** ocurre en cada flanco de subida de **ecap_sample_done**. A modo de referencia temporal, el período de esta última señal es de $19\,\mathrm{T_{CLK}}$.

Signal-Tap Logic Analyzer permite exportar los datos capturados a archivos en formato CSV. Además, permite ejecutar capturas y diferentes funcionalidades mediante scripts TCL, el cual fue el medio elegido para utilizar este programa. El script desarrollado inicia la captura de datos, y cuando se completa, guarda los datos en un archivo CSV para procesarlos posteriormente en un script de Python, en donde se obtienen los resultados presentados en las siguientes secciones.

8.4. Ganancia del circuito

Para poder calcular la amplitud pico a pico de la ECAP impuesta a la entrada, es necesario conocer la ganancia que posee el sistema CANICs.

Esta ganancia se puede determinar de dos maneras: experimentalmente, al imponer una señal SENSADO con amplitud de SA y ECAP conocidas en la entrada, y al medir la amplitud de la ECAP a la salida, estimar la ganancia del sistema ($\frac{ECAP_{OUT}}{ECAP_{IN}}$). O, por otro lado, calcular la ganancia en base a las distintas etapas de amplificación que posee el sistema, calculando la ganancia de cada sección individualmente y luego concatenarlas adecuadamente.

Se optó por definir la ganancia del circuito de forma experimental imponiendo una señal SENSADO, donde se conoce la amplitud de la ECAP impuesta a la entrada, y posteriormente se mide su amplitud a la salida, para determinar la ganancia del sistema. Esta ganancia se nombró $G_{Circuito}$ y tiene un valor de $G_{Circuito} = 34{,}1\,\frac{\mathrm{V}}{\mathrm{V}}$.

La ganancia total del sistema es entonces G_{Total} definida en la ecuación 8.1, donde $G_{SA\_TO\_RAIL}$ es la ganancia que se le impone a la señal al cargársela al generador, con el fin de maximizar la excursión utilizada. Este valor de ganancia es conocido.

$$G_{Total} = G_{Circuito} \times G_{SA\_TO\_RAIL} \tag{8.1}$$

Por otro lado, si se quiere determinar la ganancia al calcular las distintas etapas de amplificación, se debe considerar que la segunda etapa impone una amplificación variable configurable con un potenciómetro, como se explicó en la sección 4.2, cuya ganancia se fijó de forma experimental en $155\,\frac{\mathrm{V}}{\mathrm{V}}$. Además, se cuenta con la ganancia del atenuador diferencial de $0{,}23\,\frac{\mathrm{V}}{\mathrm{V}}$. La $G_{Teorica}$ es entonces $G_{Teorica} = 35{,}7\,\frac{\mathrm{V}}{\mathrm{V}}$. La diferencia observada entre $G_{Teorica}$ y $G_{Circuito}$ se debe a ciertas atenuaciones intrínsecas del sistema CANICs. Dichas atenuaciones se deben en menor medida al efecto de los filtros sobre la señal de interés y en mayor medida al efecto de la resta

del template de la *Etapa 2* sobre su señal entrante. Esta última atenuación depende de la generación particular del template y, por tanto, varía entre convergencias. Por último, al igual que en el primer método de cálculo de la ganancia, se debe considerar que al cargar las señales en el generador, las mismas se amplificaron $G_{SA\_TO\_RAIL}$.

8.5. Definición de las métricas

A continuación, se definen dos métricas cuantitativas para validar la funcionalidad del sistema CANICs. Para la primera métrica, se evalúa la correlación entre la señal ECAP impuesta a la entrada y la señal ECAP recuperada por el sistema. Esto permite definir si la señal recuperada es efectivamente una ECAP. Luego, la segunda métrica define una manera de calcular la amplitud pico a pico de la ECAP impuesta a la entrada, a partir de la ECAP obtenida a la salida.

Por lo tanto, estas métricas permiten determinar la calidad de la señal recuperada y cuantificar su amplitud pico a pico.

8.5.1. Correlación entre ECAP extraída y ECAP impuesta a la entrada

Como se comentó, un parámetro relevante que se utilizará para la validación del sistema diseñado es el máximo de la función de correlación[1] entre la señal ECAP impuesta a la entrada, detallada en la sección 2.2, del capítulo 2, y la señal ECAP obtenida a la salida.

En la ecuación 8.2, se presenta la expresión de la función de correlación cruzada normalizada $R[k]$ entre las señales ECAP (señal impuesta a la entrada) y ECAP_ob (señal obtenida a la salida). Esta operación cuantifica la similitud entre ambas señales de forma independiente de su amplitud, con un resultado acotado entre -1 y 1. El valor máximo de 1 se alcanza únicamente cuando las señales coinciden exactamente, salvo por un posible desplazamiento temporal. Resulta particularmente útil para identificar la posición relativa en la que una señal presenta la mayor correspondencia con otra.

$$R[k] = \frac{\sum_n \mathrm{ECAP}[n] \cdot \mathrm{ECAP\_ob}[n-k]}{\sqrt{\sum_n \mathrm{ECAP}[n]^2} \cdot \sqrt{\sum_n \mathrm{ECAP\_ob}[n]^2}} \tag{8.2}$$

Para computar la correlación descrita en la ecuación 8.2 se utilizó un script de Python, parte del bloque *Metricas*. El script recibe la cantidad de muestras correspondientes a una ventana del sistema, y el vector de la ECAP impuesto a la entrada del sistema, para calcular la correlación entre ellos. A partir del valor máximo de correlación, se obtiene la ubicación del comienzo de la ECAP recuperada. A partir de ese momento, se toma la cantidad de puntos correspondiente al

[1] La correlación entre dos señales es la medida estadística que cuantifica el grado de similitud o dependencia lineal entre ellas, indicando cómo varía una en función de la otra.

largo del vector ECAP de entrada para obtener las muestras que corresponden a la ECAP recuperada. De esta forma se obtiene la correlación de la ECAP presente en esa ventana temporal y la traza de la ECAP recuperada (muestras que únicamente pertenecen a la señal ECAP). Posteriormente, se realizan cálculos para obtener sus valores de tensión pico a pico.

Basándose en el trabajo realizado en el doctorado de S. Culaclii [10], se definió que una señal capturada es una ECAP, si el valor máximo obtenido de la función correlación entre la señal capturada y la señal ECAP de referencia es mayor a 0.83. Se eligió que la ECAP de referencia sea la ECAP impuesta a la entrada del sistema, como se mencionó anteriormente.

8.5.2. Amplitud pico a pico de señal ECAP a la entrada

Una vez definida la métrica para determinar la calidad de la recuperación de la señal, se definió una métrica para determinar la amplitud pico a pico de la ECAP impuesta a la entrada. La misma se especifica en la ecuación 8.3, donde para estimar la amplitud se realiza un promedio con N cantidad de amplitudes pico a pico de ECAP obtenidas a la salida. Aquí se definió $V_{PP}^{ECAP_{OUT}}[i]$ como la amplitud pico a pico de una traza de ECAP obtenida, en la cual la correlación es mayor a 0,83 según la métrica definida anteriormente. Por último, conociendo la ganancia del sistema G_{Total} se obtiene la amplitud pico a pico de la ECAP a la entrada $\hat{V}_{PP}^{ECAP}$.

$$\hat{V}_{PP}^{ECAP} = \frac{\frac{1}{N}\sum_{i=0}^{N-1}(V_{PP}^{ECAP_{OUT}}[i])}{G_{Total}} \tag{8.3}$$

Para obtener esta amplitud, se utiliza el bloque *Metricas* de la figura 8.1. Usando un script en Python, se capturan varias ventanas de la salida del sistema, luego de que el sistema converge. A partir de estas ventanas se obtienen las trazas de ECAP filtradas por correlación, seleccionando aquellas que sean mayores a 0.83. Con las trazas de ECAP obtenidas se hace un promedio de las N amplitudes pico a pico. A partir de este promedio y conociendo la ganancia del sistema, es posible obtener la métrica de la amplitud pico a pico de la ECAP a la entrada.

Para determinar el valor N de trazas de ECAP para promediar, se usó el caso de estudio de la figura 8.4, donde se observa una gráfica con el valor de amplitudes pico a pico promedio obtenido por el sistema CANICs, y el valor esperado en función de los promedios, con distinta cantidad de trazas utilizadas. En particular, se tomaron tres casos distintos en función de la relación SA/ECAP. Las relaciones representadas son $450\,\frac{\mathrm{V}}{\mathrm{V}}$, $1000\,\frac{\mathrm{V}}{\mathrm{V}}$ y $2200\,\frac{\mathrm{V}}{\mathrm{V}}$, en azul, rojo y verde respectivamente. Los puntos muestran los valores de amplitud pico a pico obtenidos al promediarlos sobre N trazas de señales ECAP extraídas y normalizadas[2]. Mientras que la línea negra indica el valor esperado a partir de la ECAP impuesta a la entrada. Se ha definido una tolerancia del 20 % para evaluar la precisión de las estimaciones y decidir cuántas trazas promediar. A modo ilustrativo, al promediar 4 trazas, se observa que algunas mediciones caen dentro del margen de tolerancia, pero varias

[2]La normalización se realizó dividiendo la amplitud de la ECAP estimada a la entrada con la amplitud de la ECAP que se impuso a la entrada

se desvían más allá de él, lo que indica que dicha cantidad de promedios resulta insuficiente para obtener una estimación confiable.

Aquí se evidencia que al aumentar la cantidad de trazas promediadas, la dispersión de los valores obtenidos decrece hasta cierto punto, resultando ineficiente seguir aumentando la cantidad de trazas tomadas para promediar por encima de 6. Basándose en estas medidas, CANICs promedia 6 amplitudes pico a pico a partir de las 6 trazas filtradas obtenidas.

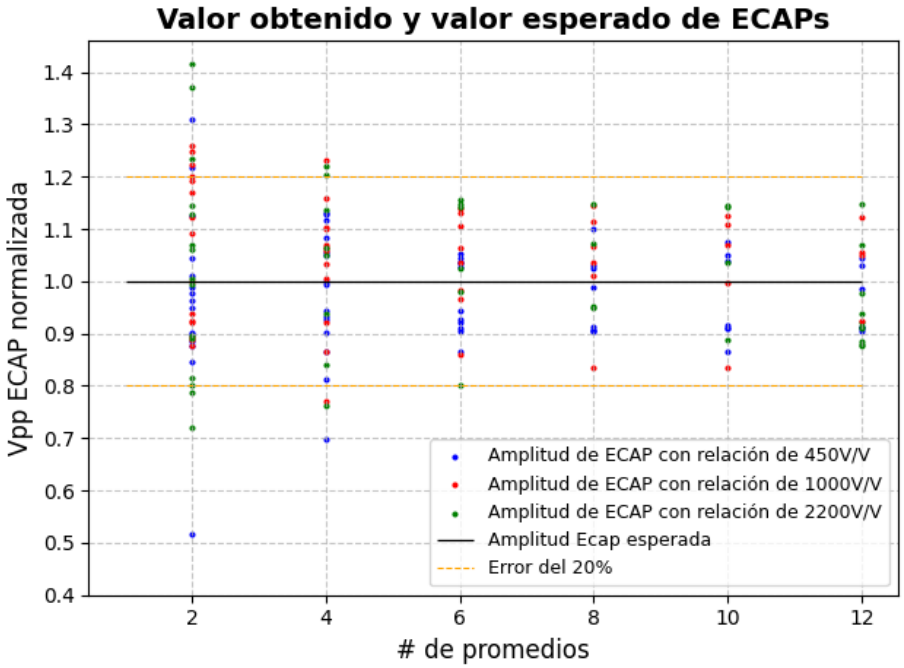

Figura 8.4: Gráfica con la amplitud pico a pico promedio obtenida de la ECAP normalizada para distintos casos de relación SA/ECAP. En función del valor N utilizado para realizar el promedio. Se definió el margen de 20 % para cuantificar la cantidad de promedios a realizar. Se optó por utilizar 6 trazas de ECAPs obtenidas por el sistema para el promedio.

Por lo tanto, la métrica de la amplitud pico a pico de señal ECAP a la entrada, $\hat{V}_{PP}^{ECAP}$, se define como el promedio de 6 amplitudes pico a pico de ECAP obtenidas que cumplen el criterio de correlación, dividido por la ganancia del sistema, tal como se muestra en la ecuación 8.4.

$$\hat{V}_{PP}^{ECAP} = \frac{\frac{1}{6}\sum_{i=0}^{5}(V_{PP}^{ECAP_{OUT}}[i])}{G_{Total}} \tag{8.4}$$

8.6. Resultados y desempeño obtenido

A continuación se trabajará en la evaluación de los resultados referidos a diferentes puntos del sistema.

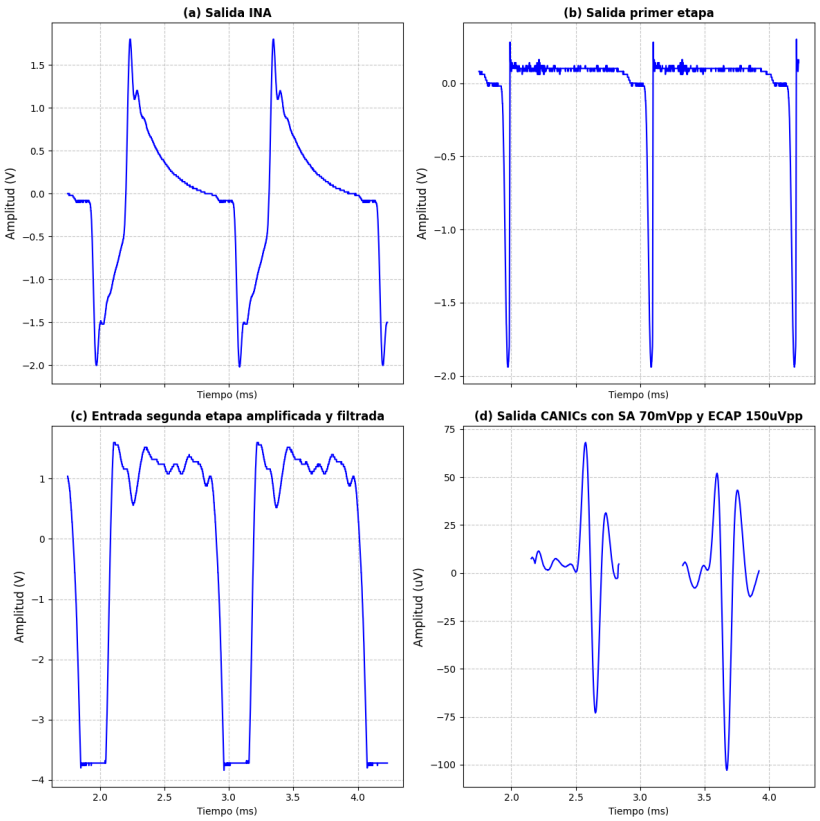

Figura 8.5: Señales observadas en distintos puntos del sistema. En (a) salida del amplificador de instrumentación, utilizada por la *Etapa 1* para generar el template de cancelación. En (b) resultado de la cancelación del artefacto SA, donde se observa el residuo junto con la ECAP. En (c) señal amplificada y filtrada por la *Etapa 2*. En (d) muestras obtenidas a la salida del sistema CANICs, donde se visualiza la ECAP extraída en dos ventanas temporales.

En la figura 8.5 se muestran medidas relevadas en diferentes puntos del sistema. Se estimuló el sistema con una señal SA de 70 mV y una ECAP de 150 μV de amplitud, lo que corresponde a una relación señal-estímulo de 53 dB.

En la subfigura 8.5.a se muestra la salida del amplificador de instrumentación. Esta señal es la que busca cancelar la *Etapa 1* generando su correspondiente template. Notar que la señal está amplificada y es del orden de los V, con el fin de maximizarla en el rango del ADC. Esto se logró amplificando de manera acorde la entrada previamente a cargarla en el generador de señales, resultando en que la amplitud de la SA alcanzara amplitudes cercanas al máximo del output swing del circuito.

En la subfigura 8.5.b se muestra el resultado de la cancelación del artefacto (SA) realizado por la *Etapa 1*. Los picos negativos corresponden a componentes de la señal que quedan fuera de la ventana de procesamiento. Por otro lado, la porción de señal visible por encima de 0 V, de menor magnitud, representa el residuo junto con la ECAP tras la cancelación. Se destaca la baja amplitud de esta señal en comparación con la señal original que se busca cancelar, es decir, la SA, mostrando

la efectividad de la *Etapa 1*.

En la subfigura 8.5.c se observa una situación análoga a la mostrada en la subfigura 8.5.b, con la diferencia de que, en este caso, la señal ha sido amplificada por el amplificador de ganancia variable utilizado para maximizar la excursión del ADC (como se vio en la sección 4.2) y posteriormente filtrada por el filtro pasabajos Butterworth, de la *Etapa 2*. Al igual que antes, se aprecia el enventanado aplicado a los modelos de cancelación, lo que provoca que, fuera del intervalo temporal definido, la señal amplificada llegue a saturar en la entrada de la segunda etapa. Este comportamiento es esperado, dado que fuera de la ventana definida no se aplica cancelación activa.

Por último, en la subfigura 8.5.d se ven las muestras obtenidas a la salida del sistema CANICs, en donde se puede ver la señal ECAP extraída en dos ventanas temporales.

Es importante recalcar que las primeras tres imágenes corresponden a señales medidas por un osciloscopio y la última es una señal obtenida a partir de la FPGA, por lo tanto, es una señal del dominio digital, representada con una traza continua únicamente con fines ilustrativos.

Las fluctuaciones que presentan las muestras de la salida que se observa en la subfigura 8.5.d, se pueden caracterizar observando la salida del sistema sin imponer la señal ECAP a la entrada. En la figura 8.6 se observan tres capturas distintas del sistema en estas condiciones junto con las muestras obtenidas para el caso en el que la relación SA/ECAP es 53 dB. Aquí es notable la diferencia observada de la amplitud de estas fluctuaciones con respecto a la ECAP extraída. A partir de la figura se puede determinar cualitativamente que para el caso mencionado se está significativamente por encima de las fluctuaciones provocadas por el ruido del sistema.

8.6.1. Correlación medida entre la señal ECAP extraída y la señal ECAP impuesta a la entrada

En esta sección se presentarán los resultados obtenidos de calcular la correlación de las muestras extraídas a la salida del sistema CANICs junto con la señal ECAP impuesta a la entrada. Como fue definido anteriormente, se considerará que la señal capturada es una ECAP si la correlación máxima obtenida se encuentra por encima de 0,83.

En la figura 8.7 se muestran dos gráficas, donde en la superior se puede observar la captura del sistema (señal punteada en rojo) con la señal ECAP impuesta a la entrada (señal continua azul), la superposición de ambas señales se realiza de tal forma que estén en el intervalo temporal de correlación máxima entre ambas señales. En la gráfica inferior se muestra la función de correlación entre ambas señales, de ella se toma el máximo para determinar el comienzo de la ocurrencia de la ECAP.

El máximo de correlación que se alcanzó entre las señales fue de **0.928**. Se estimuló el sistema con las señales en el rango de la especificación detallada en el capítulo 2. Este caso es el de menor relación entre la SA y la ECAP, siendo la amplitud pico a pico de $SA = 70\,\text{mV}$ y la $ECAP = 150\,\mu\text{V}$, donde su relación es

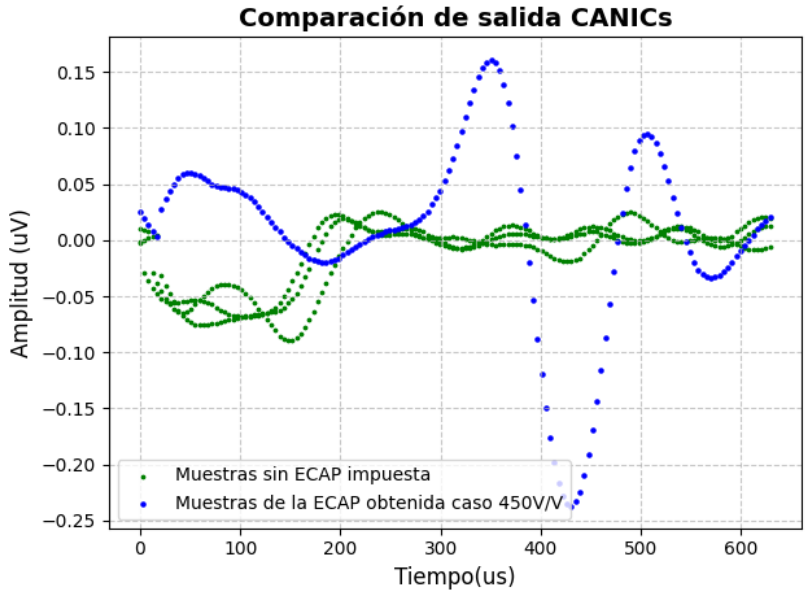

Figura 8.6: Salida del sistema sin imponer ECAP a la entrada, comparada con la salida del sistema en la que se impone la ECAP. Para este caso, la relación de señales entre la ECAP y la SA es de $450\,\frac{\mathrm{V}}{\mathrm{V}} = 53\,\mathrm{dB}$.

de $\mathrm{SA/ECAP} = 450\,\frac{\mathrm{V}}{\mathrm{V}} = 53\,\mathrm{dB}$.

A continuación se presenta la tabla 8.1 con valores de correlación para distintos factores de amplitudes pico a pico de SA y ECAP. Sus resultados evidencian que el sistema mantiene una correlación por encima de los mínimos especificados en la métrica (0.83) para un alto rango de relaciones SA/ECAP, llegando a obtener una cancelación con una relación entre señales de 70 dB como valor máximo, para el cual el sistema es capaz de obtener una ECAP. Las dos primeras columnas muestran los pares de amplitud pico a pico SA y ECAP respectivamente. Las diferentes relaciones SA/ECAP se obtuvieron variando la relación entre las señales, debido a que a la hora de generarlas, el script calcula la ganancia para que la señal resultante obtenga la máxima excursión admisible por el sistema a la hora de ingresarla al generador. Por lo tanto, la SA tiene la misma amplitud siempre, y lo que cambia es la amplitud de la ECAP generada.

Se destaca entre los resultados obtenidos que para valores en el mínimo de relación SA/ECAP especificado, la correlación alcanzó valores por encima de 0.9. Al aumentar la relación SA/ECAP por encima de los 70 dB, el desempeño disminuye drásticamente por debajo de los mínimos establecidos como aceptables (0.83). Esto se debe a que las variaciones provocadas por el ruido en estos casos son comparables con la amplitud pico a pico de la ECAP, deformando la señal e imposibilitando su obtención.

A continuación se muestran dos figuras para casos de relación mayor que $450\,\frac{\mathrm{V}}{\mathrm{V}} = 53\,\mathrm{dB}$.

Por un lado, en la figura 8.8 se puede ver el caso en el que la relación SA/ECAP

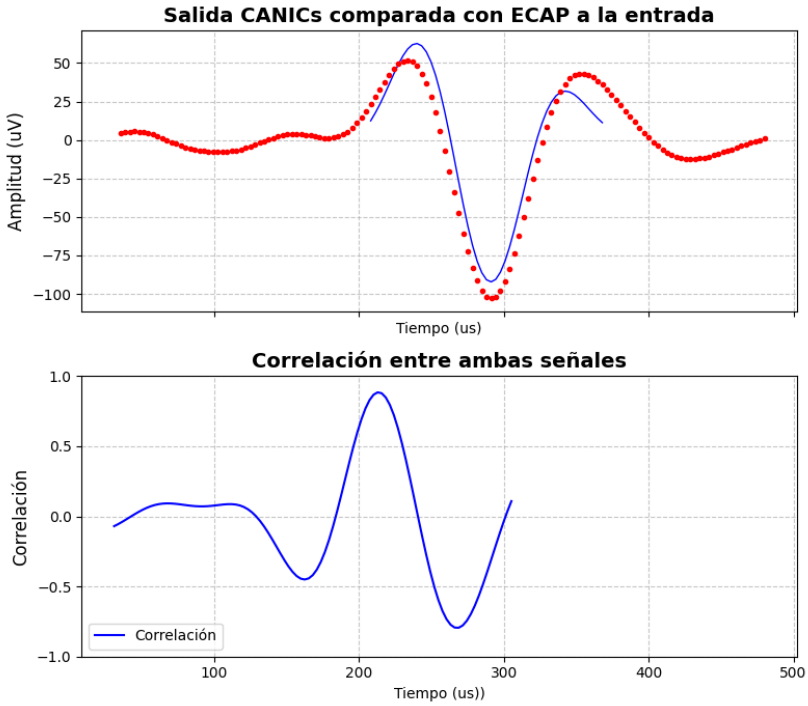

Figura 8.7: Muestras extraídas del sistema en el caso $SA = 70\,\mathrm{mV}$ y $ECAP = 150\,\mu\mathrm{V}$, donde su relación es de SA/ECAP $= 450\,\frac{\mathrm{V}}{\mathrm{V}}$. En rojo, la señal extraída por el sistema comparada con la ECAP impuesta en la entrada en azul. Abajo se muestra una gráfica de la correlación entre ambas señales.

SA (mV)	ECAP (μV)	Relación SA/ECAP (V/V)	dB	Correlación obtenida
70	150	450	53	0.928
70	75	900	59	0.865
70	40	1800	65	0.853
70	30	2400	68	0.862
70	20	3200	70	0.833
70	15	4800	73	0.723
Correlación obtenida sin ECAP a la entrada				
0.412				

Tabla 8.1: Correlaciones obtenidas entre diferentes pares de SA y ECAP. El color verde hace referencia a los valores que superaron el valor mínimo aceptable de correlación, que es 0.83.

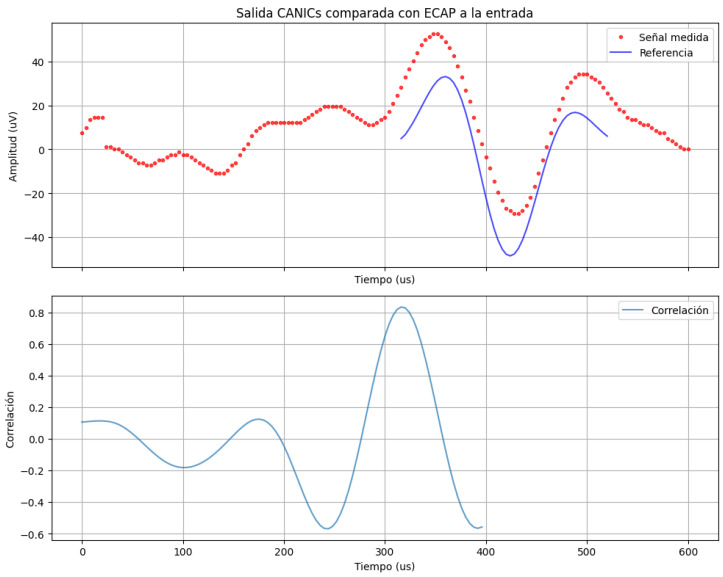

Figura 8.8: Señal extraída por el sistema CANICs comparada con la ECAP impuesta en la entrada para el caso de una relación entre las señales de $3200\,\frac{\mathrm{V}}{\mathrm{V}}$. Abajo se muestra una gráfica de la correlación entre ambas señales.

es de $3200\,\frac{\mathrm{V}}{\mathrm{V}}$ con una SA de amplitud pico a pico de 70 mV y una ECAP de $20\,\mu$V. Para tal caso, el sistema recuperó una ECAP con una correlación de **0.833** (ver tabla 8.1). Se observa en la imagen la gran similitud en la forma de onda de la señal medida, roja punteada, respecto a la ECAP de referencia, siendo la señal azul continua. La gráfica inferior de la figura indica la función de correlación, donde en el máximo valor, se toma el inicio de la ocurrencia de la ECAP.

Por otro lado, en la figura 8.9 se observa la salida del sistema para una entrada con una relación de $4800\,\frac{\mathrm{V}}{\mathrm{V}}$. Esta relación se obtuvo al imponer una SA de amplitud pico a pico de 70 mV y una ECAP de $15\,\mu$V. En este caso se observa que el sistema no es capaz de recuperar la señal ECAP de entrada, reportando una correlación de **0.723**, presentando el máximo de correlación entre la señal extraída con la ECAP de entrada en un intervalo temporal donde no se presenta la ECAP. Si observamos la imagen, el mínimo de la ocurrencia de la ECAP se estima que se encuentra cercano a los $400\,\mu$s y el sistema detecta que el máximo de la correlación se da aproximadamente a los $100\,\mu$s.

8.6.2. Amplitud pico a pico extraída de la entrada

A continuación se muestran los resultados obtenidos de la amplitud pico a pico de la ECAP que ingresó a la entrada del sistema utilizando la métrica definida en la sección 8.5.2. Se consideran únicamente aquellas trazas de ECAP cuya correlación supera el umbral de 0.83. Se utilizaron 6 trazas para promediar, y se estima la amplitud pico a pico a la entrada utilizando la ganancia del sistema medida experimentalmente. En la figura 8.10 se pueden observar las amplitudes obtenidas

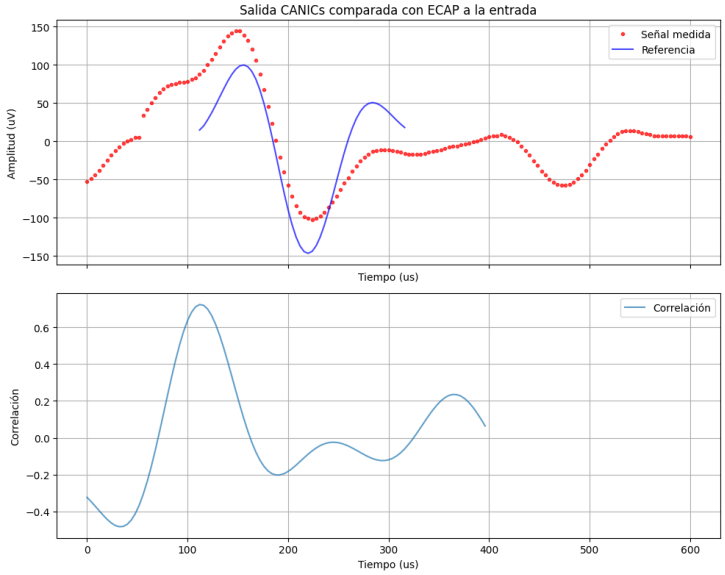

Figura 8.9: Señal extraída por el sistema comparada con la ECAP impuesta en la entrada para el caso de una relación entre las señales de $4800\,\frac{\mathrm{V}}{\mathrm{V}}$. Abajo se muestra una gráfica de la correlación entre ambas señales.

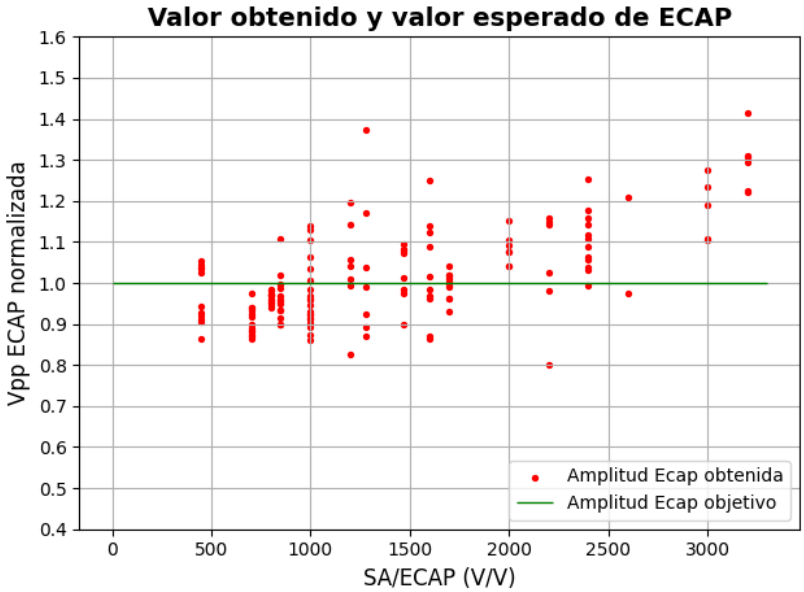

Figura 8.10: Amplitud obtenida por el bloque *Métricas*, comparada con la ECAP a la entrada normalizada en función de la relación SA/ECAP. En rojo se presentan 148 datos de la amplitud pico a pico de la ECAP obtenida normalizada. En verde se ilustra la linea con el valor objetivo de la amplitud pico a pico.

para distintas relaciones entre SA y ECAP normalizadas. Los puntos rojos corresponden a la amplitud pico a pico promedio de 6 trazas de la ECAP a la entrada del sistema, mientras que la línea verde indica el valor objetivo. Este último representa la amplitud pico a pico de la ECAP que fue aplicada a la entrada del sistema.

De los 148 puntos ilustrados, el 95 % presenta un error inferior al 20 % respecto del valor objetivo, lo que indica que la métrica empleada permite una estimación precisa de la amplitud pico a pico en un amplio rango de relaciones SA/ECAP.

Los datos presentan un menor error respecto al valor objetivo para las relaciones más bajas SA/ECAP. Mientras que la dispersión de los datos aumenta ligeramente en la medida que la relación aumenta. En particular, para valores de SA/ECAP cercanos al máximo alcanzado por el sistema (SA/ECAP = $3200\frac{V}{V}$, como se mostró en la tabla 8.1) se observa la mayor dispersión en los resultados, con errores en la estimación de la amplitud que superan el 40 %.

Este comportamiento observado en los datos concuerda con los resultados de correlación presentados en la tabla 8.1. A medida que disminuye la relación SA/ECAP, se observa un incremento de la correlación, lo que indica que la forma de la señal ECAP obtenida se asemeja más a la señal impuesta. En consecuencia, la estimación de la amplitud pico a pico resulta más precisa. Por el contrario, a mayor relación SA/ECAP, la correlación disminuye, reflejando una mayor distorsión de la ECAP y una menor fidelidad en la medición de su amplitud.

Este error, observado para valores de relación SA/ECAP superiores a 3000 V/V, se debe a condiciones de baja relación señal a ruido (SNR). En estos casos, una alta relación SA/ECAP implica que la amplitud de la señal ECAP a la entrada es considerablemente pequeña respecto a la SA. Si dicha amplitud es comparable al ruido del sistema, las fluctuaciones introducidas por este adquieren relevancia, dificultando la adquisición confiable de la amplitud pico a pico de la ECAP y, en consecuencia, impidiendo su recuperación precisa.

En general, la concordancia entre la amplitud extraída y el valor objetivo es alta (menor al 20 %), lo que evidencia que la métrica utilizada para calcular la amplitud pico a pico de la ECAP a la entrada del sistema resulta robusta y confiable dentro del rango de operación considerado, específicamente en los casos en que se detecta una ECAP a la salida, es decir, cuando la correlación supera el umbral de 0.83. A partir de la figura 8.10 puede concluirse que el sistema CANICs permite recuperar señales ECAP cuya amplitud pico a pico presenta, en promedio (sobre 6 repeticiones), un error menor al 20 % respecto de la amplitud objetivo, siempre que la relación SA/ECAP sea inferior a 3000 V/V.

8.6.3. Rechazo al modo común del circuito

Es de esperarse en la entrada el aporte de señales provenientes de interferencias de la actividad muscular. Este tipo de contribuciones se presentan en modo común para ambos nodos de entrada, por lo tanto, el sistema debe ser capaz de presentar un rechazo al modo común en la banda [0,400 Hz] de al menos 2 mV.

La etapa inicial del amplificador de instrumentación es una etapa de alto rechazo al modo común y será la encargada de suprimir el modo común proveniente

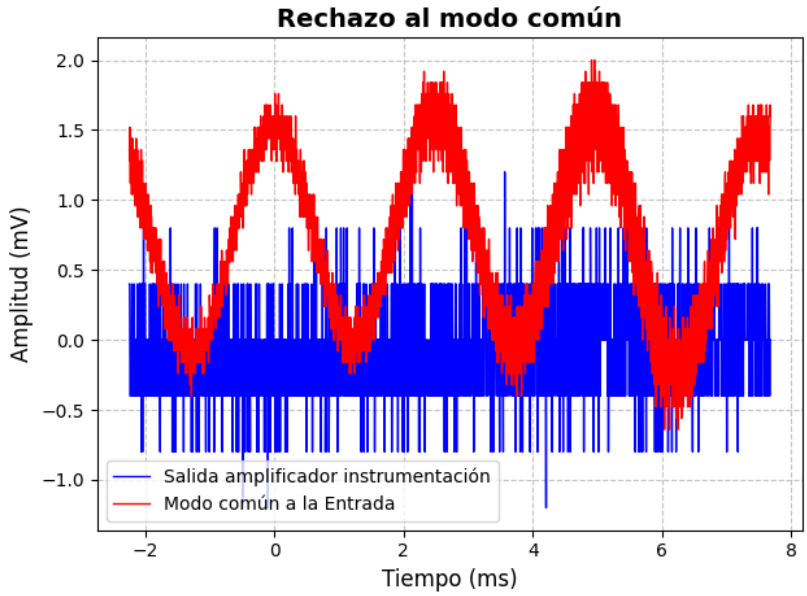

Figura 8.11: Respuesta del amplificador de instrumentación frente a componentes en modo común de amplitud $2\,\mathrm{mV_{pp}}$ y frecuencia $400\,\mathrm{Hz}$.

de los electrodos. En la figura 8.11 se observa la medición del rechazo al modo común del sistema. Esto se hizo imponiendo una señal sinusoidal de amplitud $2\,\mathrm{mV_{pp}}$ y frecuencia $400\,\mathrm{Hz}$ a la entrada del amplificador de instrumentación, para luego medir su salida. La salida, en azul, presenta valores entre $\pm 400\,\mu\mathrm{V}$, donde no se aprecia una forma de onda sinusoidal debido al ruido presente en el circuito, por lo que se considera que el rechazo al modo común es satisfactorio.

Por otra parte, a la entrada también se presentan señales en continua diferenciales con un valor de $\pm 100\,\mathrm{mV}$. De igual modo, estas señales no son deseadas y deben ser eliminadas. Para esto, el amplificador de instrumentación se diseñó con una respuesta del tipo pasa-altos capaz de filtrar las componentes en DC diferenciales. El desempeño de este circuito se puede observar claramente en la imagen 8.12 donde se observa una de las entradas del amplificador de instrumentación en rojo y en azul su salida con una continua de $0\,\mathrm{V}$.

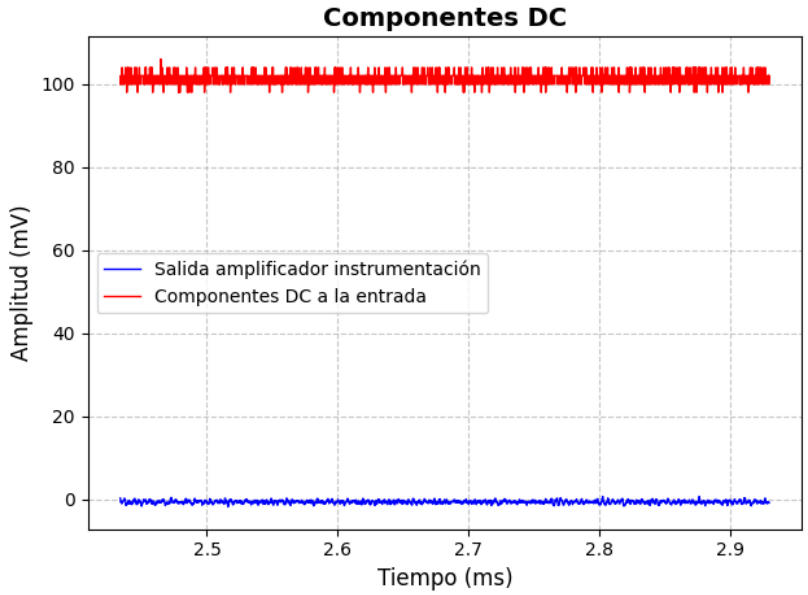

Figura 8.12: Respuesta del amplificador de instrumentación frente a señales diferenciales en continua de $\pm 100\,\mathrm{mV}$.

Capítulo 9

Conclusiones

El presente proyecto logró diseñar, implementar y validar un sistema de cancelación de artefactos en tiempo real para neuroestimuladores implantables en lazo cerrado, denominado CANICs. El diseño del sistema se centró en crear el procesamiento digital a medida, diseñado para su futura implementación en circuitos integrados, para formar parte de un System on Chip (SoC).

Utilizando una arquitectura estructurada por dos etapas secuenciales de cancelación, se logró obtener la señal ECAP a la salida, tanto en pruebas en simulaciones (mediante el uso de la herramienta Cadence) como a través de pruebas en HW.

Para realizar las simulaciones, se generaron modelos comportamentales, en el lenguaje Verilog-A, de los componentes analógicos utilizados por el sistema, siendo estos el ADC, DAC y amplificadores. Se verificaron individualmente, comparando su comportamiento con el establecido en sus respectivas hojas de datos, cumpliendo las características mínimas requeridas para efectuar una correcta simulación del sistema CANICs completo.

Se realizó una verificación del funcionamiento de los bloques digitales implementados individualmente, corroborando que todos cumplen con sus especificaciones de diseño y producen las salidas esperadas.

Por último, se realizó la validación del sistema CANICs. Para ello, se generaron simulaciones analógico-digital en donde se ingresaron vectores de señales SA y ECAP y, luego de que el sistema convergió, se obtuvieron las ECAPs en la salida.

Se validó el desempeño a nivel de HW con una PCB específica para este sistema, diseñada por este grupo de trabajo. La misma funciona en conjunto con la placa de desarrollo DE0, que contiene la FPGA utilizada. Se alcanzó un diseño para el conjunto PCB-DE0 orientado a la evaluación de un prototipo, brindando diferentes modos de alimentación, tipos de filtrado y puntos de medición en nodos de interés, además de permitir aislar etapas entre sí para pruebas individuales. Esto resultó fundamental para la validación de cada etapa del procesamiento a nivel de HW.

Desde el punto de vista del desempeño en HW, los resultados obtenidos fueron satisfactorios. El sistema CANICs demostró ser capaz de recuperar señales ECAP con alta correlación frente a un amplio rango de relaciones SA/ECAP. Se observó que, para relaciones comprendidas entre 53 dB y 70 dB, las señales ECAP extraídas presentaron una correlación superior a 0.833 respecto a la señal de referencia. El

valor máximo de correlación registrado fue de 0.928, correspondiente a una relación de 53 dB. Además, en términos de precisión en la estimación de la amplitud pico a pico a la entrada del sistema, se logró un error relativo inferior al 20 % siempre que la relación SA/ECAP se mantuviera por debajo de 69 dB. Los resultados obtenidos para caracterizar la amplitud de la ECAP a la entrada están condicionados, debido a que fueron obtenidos asumiendo que la señal ECAP posee una morfología única y conocida. La métrica de correlación para obtener la amplitud pico a pico no es implementable en aplicaciones reales, debido a las variaciones en la morfología de la ECAP.

En comparación con trabajos previos, CANICs alcanzó una relación de cancelación de 70 dB, frente a los 56 dB de CANE y los 100 dB de S. Culaclii. No obstante, a diferencia de estos enfoques basados en microprocesadores, CANICs implementa una arquitectura digital dedicada, totalmente integrada en HW, destacándose la inclusión de la segunda etapa de cancelación en el mismo, que en estos trabajos anteriores se realizaba por software. Esta implementación en HW reduce la dependencia de procesamiento externo. Por lo tanto, CANICs representa una solución adecuada para aplicaciones implantables, donde es necesario integrar todo el procesamiento en un único sistema, con el objetivo de reducir el tamaño del dispositivo, minimizar el consumo energético y eliminar la dependencia de unidades de procesamiento externas.

Si bien el diseño analógico específico para ASICs y la validación *in vivo* quedaron fuera del alcance de este trabajo, el desarrollo alcanzado constituye un avance significativo en la implementación de sistemas de estimulación en lazo cerrado, orientados a mejorar la calidad de vida de pacientes con dolor crónico.

En conjunto, el proyecto CANICs representa una prueba de concepto capaz de integrarse a futuros desarrollos y servir como base para investigaciones posteriores en el ámbito de la microelectrónica para dispositivos médicos implantables, en donde se dejó un entorno sólido de simulación del sistema, al igual que el entorno de HW implementado, en el cual se pueden probar diferentes conceptos, y ser un punto de partida para desarrollos futuros.

9.1. Trabajo futuro y posibles mejoras

Como se explicó en el alcance de este proyecto, el mismo contemplaba el diseño y desarrollo de la sección digital del sistema. Aunque se definió la arquitectura de todo el sistema, diversos detalles de algunos circuitos analógicos quedaron indefinidos debido a estos motivos. El principal circuito a profundizar sería la etapa de amplificación y acondicionamiento de la señal de entrada al sistema. La misma debe implementarse con un amplificador de ganancia automática, de forma que maximice la excursión de la señal.

Dentro del sistema CANICs, la etapa de amplificación de entrada de la *Etapa 2* que se implementó con un amplificador de ganancia variable, no tiene adaptación dinámica de su valor, dado que se implementó simplemente con un potenciómetro. Una gran mejora en la performance general del sistema sería generar que el am-

plificador posea ganancia variable pero con un lazo de realimentación que permita ajustar dinámicamente la ganancia, de forma de maximizar en todo momento la excursión de la señal.

Por más que el diseño digital se concluyó satisfactoriamente y se probó su funcionamiento, no se le realizó una verificación formal, como por ejemplo con metodologías UVM, que son estándar en la industria de circuitos digitales integrables. Además, no se logró hacer la síntesis ni el place and route del diseño, por lo que no se tuvieron estimativos del consumo del bloque digital, ni del área que requiere. Estos puntos serían importantes de retomar a la hora de utilizar este diseño en un sistema futuro.

Por otro lado, el diseño tiene una serie de parámetros que permiten darle una amplia configuración, desde tiempos de espera, niveles de cota de error hasta cantidad de iteraciones del template de la *Etapa 2*. El inconveniente actual es que estos son parámetros de Verilog, por lo que una vez determinados quedan fijos en el diseño. Una mejora interesante sería implementar un controlador junto con una memoria que permita elegir entre un abanico de opciones preestablecidas para los parámetros de interés, de forma que se puedan cambiar dinámicamente con el fin de tener diferentes configuraciones del sistema.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice A

Sistema analógico

A.1. Filtro Butterworth antialias

A continuación se detallarán los pasos seguidos al momento de diseñar el filtro Butterworth utilizado a la salida de la segunda etapa, previo a ser muestreado por el ADC. Es por esto que este es un filtro denominado Anti-alias.

Cabe destacar que la señal de interés en este filtro es únicamente la ECAP siendo que es la que se desea observar, por lo que el ancho de banda de interés es $BW_{ECAP} = 5\,\text{KHz}$.

Por otro lado, se tiene que el ADC tiene una resolución de 12 bits y está funcionando a una frecuencia de muestreo de 118 KHz

A.1.1. Atenuación en banda de rechazo (Stopband)

A continuación se detalla la mínima atenuación que debe tener el filtro para estar por debajo del ruido de cuantización en la banda de rechazo.

- El ADC es ideal: sin errores no lineales ni jitter, y solo introduce ruido de cuantización uniforme.
- La señal de entrada es una onda senoidal de amplitud máxima, perfectamente adaptada al rango dinámico del ADC.
- El ruido de cuantización se modela como una variable aleatoria uniforme en el intervalo $[-\Delta/2,\Delta/2]$, donde Δ es el tamaño del paso de cuantización.

Con las hipótesis planteadas:

$$P_S = \frac{A^2}{2}$$

EL paso de cuantización es

$$\Delta = \frac{2A}{2^N} = \frac{A}{2^{N-1}}$$

La potencia de una distribución uniforme $[-\Delta/2,\Delta/2]$ es :

$$P = \frac{\Delta^2}{12} \Longrightarrow P_N = \frac{A^2}{3{,}2^{2N}}$$

Entonces la relación Señal/Ruido es:

$$SNR = \frac{P_S}{P_N} = \frac{A^2/2}{A^2/3{,}2^{2N}} = \frac{3{,}2^{2N}}{2}$$

$$\Longrightarrow SNR_{dB} = 10.log(\frac{3}{2}{,}2^{2N}) = 20N.log(2) + 10.log(\frac{3}{2})$$

Tomando en consideración que tenemos 12 bits de resolución (N = 12)

$$\Longrightarrow SNR_{dB} = 75\,\text{dB}$$

Esto lo que significa es que es necesario que el filtro posea una atenuación mayor a 75 dB en la banda de rechazo para que la potencia de la señal sea en cualquier caso menor al ruido de cuantización, no afectando en la resolución efectiva del sistema debido al aliasing.

A.1.2. Respuesta en frecuencia

El estudio de la respuesta en frecuencia se realizó en el entorno de simulación Cadence, en la imagen A.1 se observa su filtro de corte y la atenuación por década de 80 dB.

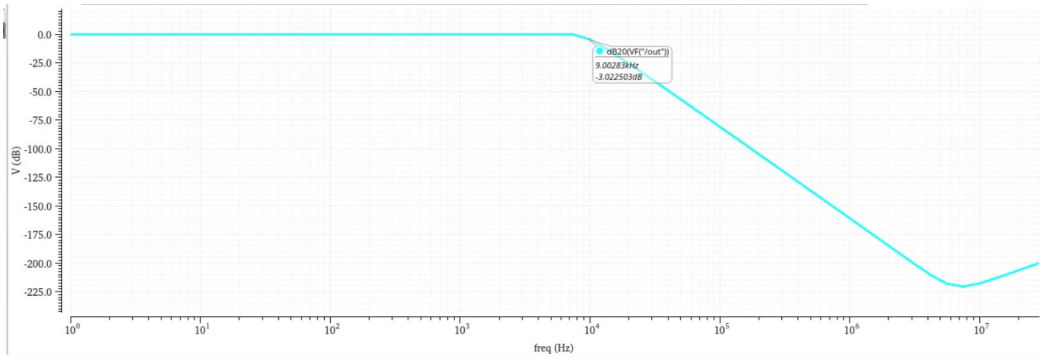

Figura A.1: Respuesta del filtro de cuarto orden con respuesta Butterworth diseñado.

A.2. Circuito Atenuador

El generador [34] cuenta con una resolución de 16 bits y utilizaremos el rango de 20 V, su LSB es, por tanto, 300 μV. Dadas las especificaciones de las señales de entrada (ver 2) no es posible siquiera generar el caso de la ECAP con mayor amplitud.

Se tiene entonces que la señal diferencial excursiona con 10V de amplitud [-10V,10V]. La ganancia máxima aplicada a las señales de entrada es G = 10V/240mV = 41 V/V. Por lo que la ECAP generada tiene una amplitud pico a pico en el mejor caso de 6,25mV, aproximadamente 10 veces el LSB. Por otro lado, para el peor caso de ECAP (Vpp = 2uV) sigue siendo insuficiente esta solución, Vpp generada es de 83uV. Para probar estos casos se debe buscar un generador de señales con mayor resolución.

Por otro lado, este amplificador introduce ruido blanco al circuito. Para minimizar esta contribución, se constituyó en esta etapa un filtrado de tipo pasabajos con frecuencia de corte $F_C = 100\,\mathrm{KHz}$.

Se tiene entonces que su ruido es.

$$v_n = \sqrt{B(e_n^2 + 4K_BTR_1)} \tag{A.1}$$

- $B = Fc * \frac{\pi}{2} = 156\,\mathrm{KHz}$
- $e_n = 4\frac{nV}{\sqrt{Hz}}$
- $R_1 = 43\,\mathrm{K\Omega}$

$$\implies v_n = \sqrt{B(e_n^2 + 4K_BTR_1)} = 10\,\mu\mathrm{V} \tag{A.2}$$

Esta página ha sido intencionalmente dejada en blanco.

Apéndice B

Sistema digital

B.1. Flip-Flops utilizados para las memorias

Teniendo en cuenta que la memoria de la *Etapa 1*, mostrada en la figura 3.23, posee una profundidad de $DEPTH = 192$ y un ancho de palabra de 12 bits, y que está implementada como un banco de registros, es posible calcular la cantidad de FFs requeridos. En este caso, la cantidad total es $FFs\_mem1 = 12 \times 192 = 2304$.

El cálculo es el mismo para la memoria de la *Etapa 2*. En este caso, la cantidad total es $FFs\_mem2 = 16 \times 192 = 3072$.

Por lo tanto, considerando ambas memorias, se requiere un total de 5376 FFs. Utilizando la definición de que una unidad de memoria corresponde a un registro compuesto por 12 FFs, se concluye que ambas memorias ocupan $\frac{5376}{12} = 448$ unidades de memoria.

B.2. Control de los Buffers

Como se explicó en capítulos anteriores, se utilizan dos buffers de la familia 74AHC9541A de Nexperia [20]. Su función es permitir, cuando sea necesario, la imposición de la palabra de configuración hacia el ADC, así como la transmisión del dato muestreado hacia el controlador. Esta intermediación es fundamental debido a que el puerto de datos del ADC opera en modo triestado, pudiendo funcionar como entrada, salida o permanecer en alta impedancia.

Para garantizar un funcionamiento correcto y evitar conflictos en el bus de datos (como posibles cortocircuitos), es necesario diseñar un sistema digital que actúe como controlador de los buffers. Este controlador debe ajustarse al comportamiento temporal tanto del ADC como de los buffers, de modo que el flujo de datos hacia la FPGA ocurra en los momentos adecuados.

En la figura B.1 se aprecia el comportamiento temporal del controlador diseñado. Dependiendo de sus entradas **wr_n**, **rd_n**, el comportamiento de las señales de control a su salida para los buffers **oe_n1** y **oe_n2**. La señal **int_n** del ADC es meramente de referencia, y **in_adc** representa el estado en el cual está el bus

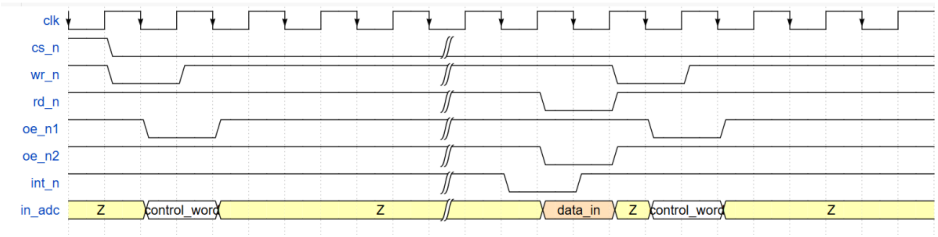

Figura B.1: Diagrama de tiempos del controlador diseñado para los buffers. Se aprecian las señales de habilitación por nivel bajo de los buffers **oe_n1** y **oe_n2**.

de datos del ADC.

Puede aparecer un problema en la transición entre estos dos estados (entrada y salida). Si ambos buffers están habilitados simultáneamente (**oe_n1** y **oe_n2** habilitados al mismo tiempo), aunque sea por un breve instante, puede ocurrir un cortocircuito porque tanto los buffers como el ADC intentan escribir en el mismo bus.

Durante la transición entre la deshabilitación del buffer de lectura (**oe_n2**) y la habilitación del buffer de escritura (**oe_n1**), puede ocurrir un conflicto si ambos buffers están activos simultáneamente durante un breve intervalo. Esto se debe a los tiempos no ideales de activación y desactivación de las salidas (t_en, t_dis) especificados por los fabricantes. Si el buffer de escritura de la palabra de control se habilita antes de que el ADC libere el bus (es decir, antes de que su salida pase a alta impedancia), puede generarse un cortocircuito al intentar imponer señales en direcciones opuestas sobre el mismo bus.

Teniendo en cuenta los posibles conflictos derivados del comportamiento temporal de las señales, según lo especificado por los fabricantes, la solución adoptada consiste en desfasar medio período de reloj la habilitación del buffer de escritura de la palabra de control (**oe_n1**) respecto al buffer de lectura (**oe_n2**), asegurando así que el ADC haya liberado el bus antes de que se imponga la palabra de control, tal como se observa en la figura B.1 en la transición de **in_adc** entre $data\_in$, Z y $control\_word$.

B.3. Funcionamiento del bloque *mem_handler*

En la figura B.2 se observa una simulación del bloque $mem\_handler$ en un arranque típico. Las señales **cs_n** y **th_ena** no son de este bloque, sino que **cs_n** proviene del controlador del DAC, y genera un pulso a cero cuando escribe un nuevo dato. **th_ena** es una señal proveniente del ADC, y produce un pulso a cero cuando el ADC está muestreando datos. Estas señales se agregaron a modo ilustrativo para identificar cuándo el DAC está generando un nuevo dato (pulso bajo de **cs_n**) y cuándo el ADC está capturando datos (pulso bajo de **th_ena**). Un arranque típico es, con ambas direcciones en cero, enviar al DAC el primer valor

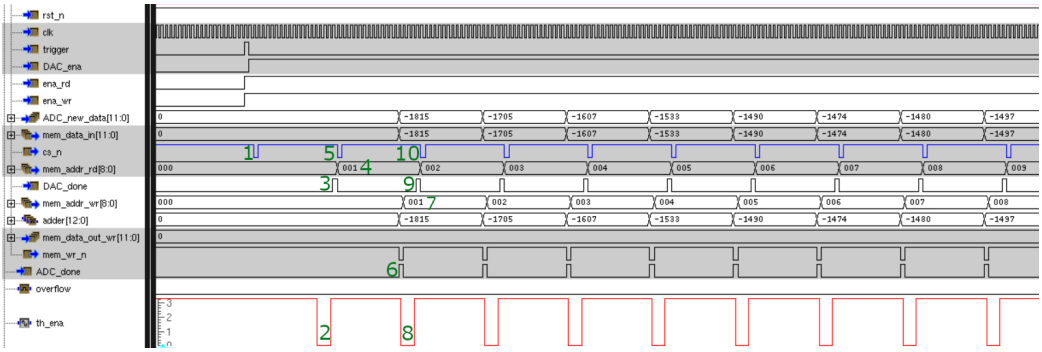

Figura B.2: Simulación del bloque *mem_handler*. Las señales **cs_n** y **th_ena** no son de este bloque, sino que **cs_n** es proveniente del DAC, y genera un pulso a cero cuando escribe un nuevo dato. Análogamente, **th_ena** es una señal proveniente del ADC, y produce un pulso a cero de duración $3\,\mathrm{T_{CLK}}$ cuando el ADC está muestreando datos.

de memoria, utilizando un pulso por nivel bajo de la señal **cs_n** (1)[1]. Unos períodos después, se observa un pulso por nivel bajo de la señal **th_ena**, proveniente del ADC (2). Esta señal indica que el ADC tomó la muestra del primer valor impuesto por el DAC, por lo que ahora el DAC puede generar otro valor, lo cual ocurre luego del pulso en la señal **DAC_done** (3). Con este pulso, aumenta en uno el puntero de memoria de lectura (4), de forma que ahora se envía al DAC el siguiente valor, utilizando nuevamente la señal **cs_n** (5). Luego de enviar el segundo dato al DAC, surge una nueva señal **ADC_done** (6), la cual indica que el ADC tiene un nuevo dato, el cual se corresponde con el primer valor generado por el DAC (generado por el pulso de (1)). Por lo tanto, el controlador procede a aplicar el algoritmo del template (explicado en la sección 3.13.1) y guardar este dato en la dirección cero de memoria. Luego, aumenta en uno el valor de la dirección de escritura (7) y finalmente muestrea el nuevo dato, correspondiente al pulso (5), con otro pulso bajo de **th_ena** (8). Luego se espera a la llegada de otro pulso de **DAC_done** (9) y el ciclo continúa de forma análoga, comenzando nuevamente con otro pulso de **cs_n** (10), hasta que las señales de enable desciendan a cero.

Notar que el valor de la señal **mem_data_out_wr** es cero durante todo el primer ciclo de la señal entrante, de esta forma se capturan los datos entrantes intactos y se genera la primera iteración del template. En los siguientes ciclos de señal SENSADO, la señal **mem_data_out_wr** tendrá como valor el template guardado en memoria, el cual varía hasta lograr la convergencia del template (se verá en el capítulo 5), manteniéndose constante hasta que sea necesario nuevamente habilitar el lazo de control para actualizarlo y disminuir el error hasta una cota tolerable.

[1]Los números son las referencias numéricas a puntos destacables, presentes en la figura B.2

Esta página ha sido intencionalmente dejada en blanco.

Referencias

[1] E. Parliament and Council, "Directive 90/385/eec on active implantable medical devices," 1990.

[2] Clínica Universidad de Navarra, "Neuroestimulación," 2025.

[3] A. Gómez Borja, "Tratamiento analgésico y neuroestimulador en el tratamiento del dolor crónico," *NPunto*, vol. IV, Agosto 2021.

[4] Memorial Sloan Kettering Cancer Center, "Información sobre el estimulador de la médula espinal (scs)," 2024.

[5] E. Times, "A tradeoff between microcontroller, dsp, fpga, and asic technologies," *EE Times*, 2022.

[6] M. Ker and M. Cheng, "System on chip (soc) for neuromodulation," in *Encyclopedia of Biomedical Engineering* (Y. Liu, ed.), Springer, 2012.

[7] R. M. Parker and colleagues, "Experimental recordings of spinal cord ecaps in humans," *Journal of Neuroscience Methods*, vol. 210, pp. 149–155, 2012.

[8] S. Medical, "Evoke, sistema de estimulación de médula espina – tratamiento en lazo cerrado para el dolor crónico," 2025. Folleto técnico, MK.049.

[9] R. Vallejo, K. Chakravarthy, A. Will, K. Trutnau, and D. Dinsmoor, "A new direction for closed-loop spinal cord stimulation: Combining contemporary therapy paradigms with evoked compound action potential sensing," *Journal of Pain Research*, vol. 14, pp. 3909–3918, 2021.

[10] S. Culaclii, *Design of A System for Cancelling Stimulus Artifact in Multi-Channel Neural Interfaces*. PhD thesis, University of California, Los Angeles, Los Angeles, CA, 2019.

[11] J. Evia, R. Garcia Ordeig, and J. Schmitd, "Cancelación de artefactos en neuroestimuladores," tesis de grado, Universidad de la República (Uruguay), Facultad de Ingeniería, Uruguay, 2024.

[12] S. Medical, "Evoke: Closed-loop spinal cord stimulation with ecap control for chronic pain," *PubMed*, 2023.

[13] FDA, "Evoke spinal cord stimulation (scs) system," 2019.

[14] C. J. D. Luca, "The use of surface electromyography in biomechanics," *Journal of Applied Biomechanics*, vol. 13, no. 2, pp. 135–163, 1997.

[15] S. Culaclii, B. Kim, Y.-K. Lo, and W. Liu, "A hybrid hardware and software approach for cancelling stimulus artifacts during same-electrode neural stimulation and recording," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 6190–6193, 2016.

[16] Analog Devices, *AD8221: Precision Instrumentation Amplifier*, 2009. Rev. C, Datasheet.

[17] Texas Instruments, *OPA197: 36-V, Precision, Rail-to-Rail Input/Output, Low Offset Voltage Operational Amplifier*, 2018. SBOS737C – January 2016 – Revised March 2018, Datasheet.

[18] Texas Instruments, *OPA182: 36-V, 5-MHz, Low-Noise, Zero-Drift, MUX-Friendly, Precision Op Amp*, 2022. SBOS936E – November 2019 – Revised August 2022, Datasheet.

[19] *Analog Devices. MAX1266: 12-Bit, Multichannel, Serial Output ADC, 2004. Rev. 3, Datasheet.*, 2004.

[20] Nexperia, *74AHC9541APW: Octal buffer/line driver; 3-state*, 2023. Datasheet.

[21] Analog Devices, *AD5424/AD5433/AD5445: CMOS Multiplying DACs*, 2009. Rev. E, Datasheet.

[22] H. Zumbahlen, *Linear Circuit Design Handbook*. Newnes/Elsevier, 2008.

[23] Altera, *DE0 User Manual*, 2010. SBOS936E – November 2009 – Revised August 2010.

[24] Texas Instruments, *REG71055: Buck-Boost Charge Pump with up to 60-mA Output Current*, June 2005. Datasheet, Rev. D.

[25] onsemi, *NCP160: 250 mA, Ultra-Low Quiescent Current, Low Dropout Linear Regulator*, Sept. 2022. Datasheet, Rev. 7.

[26] Texas Instruments, *LP5907: 250-mA Ultra-Low-Noise, High-PSRR LDO Voltage Regulator*, June 2015. Datasheet, Rev. M.

[27] Analog Devices, Inc., *MAX6072: Precision, Low-Noise, Low-Dropout, Series Voltage Reference*, Feb. 2014. Datasheet, Rev. 6.

[28] Analog Devices, Inc., *LT1983: Micropower Low Noise Negative Regulator in ThinSOT*, Dec. 2003. Datasheet, Rev. B.

[29] Analog Devices, Inc., *LT1983: 100mA Regulated Charge-Pump Inverters in ThinSOT*, Dec. 2003. Datasheet, Rev. B.

[30] Analog Devices, Inc., *ADP7185: -2.5 V to -5.5 V, 500 mA, Low Noise, CMOS Linear Regulator*, Jan. 2016. Datasheet, Rev. B.

[31] Texas Instruments, "Ths4561: Low-noise, high-precision, fully differential amplifier." `https://www.ti.com/lit/ds/symlink/ths4561.pdf`, 2020. Datasheet.

[32] Texas Instruments, *LSF0108: 8-Bit Bidirectional Multi-Voltage Level Translator for Open-Drain and Push-Pull Applications*, May 2021. Datasheet, Rev. S.

[33] Texas Instruments, *OPA2863-Q1: Precision, Low-Power, Zero-Drift, Rail-to-Rail Output Operational Amplifier*, 2023. Datasheet.

[34] Keysight Technologies, *33522A: 30 MHz Function / Arbitrary Waveform Generator*, 2013. Datasheet.

Esta página ha sido intencionalmente dejada en blanco.

Índice de tablas

8.1. Correlaciones obtenidas entre diferentes pares de SA y ECAP. El color verde hace referencia a los valores que superaron el valor mínimo aceptable de correlación, que es 0.83. 129

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

1.1. Sistema de neuroestimulación medular implantado. Se observa el neuroestimulador implantable, ubicado en la región glútea, los cables de derivación (leads) que conducen las señales eléctricas, y los electrodos posicionados adyacentes a la médula espinal, donde se aplica la estimulación. Imagen adaptada de [4]. 2

1.2. Morfología típica de la ECAP registrada en humanos durante estimulación de la médula espinal. Se observa una forma trifásica compuesta por una onda positiva inicial (P1), un pico negativo (N1) y una segunda onda positiva (P2). La amplitud del ECAP varía en función de la intensidad de corriente de estimulación, lo que permite su uso como señal de retroalimentación en sistemas de estimulación en lazo cerrado. Imagen tomada de [7]. 3

1.3. Diferentes amplitudes de la ECAP definen una ventana terapéutica, asegurando el alivio del dolor y el confort del paciente. Imagen tomada de [8] 4

1.4. ECAP resultante (ondas N1 y P2 con P1 omitida por superposición con el artefacto, solo con fines ilustrativos) junto con el artefacto de estimulación, para el caso de lazo abierto y lazo cerrado. Cuando el paciente tose, los electrodos de estimulación medular se acercan a la médula espinal. Esto es detectado por los sensores espinales como un aumento en la amplitud de la ECAP. En la configuración de lazo abierto (A), los parámetros de estimulación son fijos; generando que la ECAP aumente en amplitud (aumento en N1 y P2) a medida que disminuye la distancia entre los electrodos y la médula. En la configuración de lazo cerrado (B), el sistema detecta el aumento de la ECAP y reduce dinámicamente la amplitud de estimulación para mantener constante la amplitud de la ECAP (amplitud de N1 y P2 constantes). Imagen tomada de [9]. 5

1.5. Segmento de médula espinal con estimulación en corriente (I_{EST}) aplicada a las columnas dorsales en un extremo del electrodo. En el extremo opuesto, se mide la ECAP (ondas N1 y P2 con P1 omitida por superposición con el artefacto, solo con fines ilustrativos) resultante, junto con el artefacto de estimulación asociado (SA). Imagen tomada de [9]. 6

1.6. Ilustración del sistema CANICs en el contexto de sistema de neuroestimulación. Se observa que a la entrada del neuroestimulador ingresa la señal SA + ECAP, para luego ser procesadas por el sistema CANICs. Este extrae la señal ECAP, que luego sirve como realimentación para que el generador de estimulación ajuste la corriente de estimulación I_{EST} para la terapia. 9

2.1. Modelo de la señal SA utilizado en el sistema. 13

2.2. Modelo de la señal ECAP utilizado en el sistema 13

2.3. Cuatro posibles ocurrencias de la señal ECAP. El eje de tiempo se referencia respecto al comienzo de la fase negativa de la SA. 14

2.4. Arquitectura del sistema CANICs con dos etapas de procesamiento y un control principal del sistema. Además, se tiene un entorno de trabajo para la generación y análisis de señales. 15

2.5. Diagrama de las etapas del sistema CANICs. 17

2.6. Resultado de realizar el promedio de dieciséis ECAPs, debido a la ventana de ocurrencia variable. En gris se observa las dieciséis posibles ocurrencias de las ECAPs, mientras que en azul se observa el resultado de su promedio, *remanente_ecap*, destacándose la atenuación del promedio. 19

3.1. Diagrama de bloques simplificado de la etapa 1. 24

3.2. Amplificador de instrumentación de entrada. 26

3.3. Circuito comparador en cascada con un rectificador de media onda. Su entrada POS_IN se corresponde a la señal SENSADO. 28

3.4. Ventana de histéresis del amplificador. Las flechas rojas y azules indican el comportamiento en régimen del circuito, mientras que las marrones son un posible caso de arranque que puede tener el circuito en caso de que, al encenderse, la salida AMP_OUT sea $-3{,}3\,\mathrm{V}$ y la señal de entrada POS_IN sea menor al valor VTHNEG. 28

3.5. Simulación del circuito detector de señales de entrada. 29

3.6. Bloque digital del circuito detector de señal. 29

3.7. Ejemplo de sincronización periódica entre el sistema digital y la señal analógica de entrada SENSADO. Al ser la frecuencia de reloj ($\frac{1}{T_{CLK}}$) un múltiplo de la frecuencia de estimulación ($\frac{1}{T}$), el tiempo entre la activación del Schmitt Trigger (línea punteada) y la generación del pulso trigger (flecha verde) se mantiene constante, garantizando una señal de trigger periódica. El valor de $4T_{CLK}$ mostrado es ilustrativo y no representa un valor especifico del sistema. 31

3.8. Ejemplo de sincronización NO periódica entre el sistema digital y la señal analógica de entrada SENSADO. Al no ser la frecuencia de reloj ($\frac{1}{T_{CLK}}$) un múltiplo de la frecuencia de estimulación ($\frac{1}{T}$), el tiempo entre la activación del Schmitt Trigger (línea punteada) y la generación del pulso trigger (flecha verde) no se mantiene constante, esto genera una señal de trigger que no es periódica. Los valores de $3T_{CLK}$ y $4T_{CLK}$ son ilustrativos y no representan valores específicos del sistema. 32

3.9. Circuito implementado para el amplificador diferencial. 32

3.10. Circuito implementado del filtro anti aliasing. 33

3.11. Diagrama de tiempos del ADC MAX1266 en *internal acquisition mode*. Se observa el comportamiento esperado de las señales de control $\overline{CS}$, $\overline{WR}$, $\overline{RD}$ con la señal de interrupción $\overline{INT}$, junto con el bus triestado `D11-D0`. Fuente: Hoja de datos del MAX1266 [19]. . . 35

3.12. Esquema de conexión del controlador del ADC con el ADC, con sus respectivas señales. En punteado dos buffer triestado. La señal **vin** corresponde a la entrada analógica que se desea muestrear. La señal de control **cs_n** habilita el funcionamiento del ADC. Las señales **wr_n**, **int_n** y **rd_n** permiten controlar el proceso de muestreo y lectura de datos. La salida digital del ADC muestreada se representa mediante **adc_data**. Finalmente, la señal **ena** habilita el controlador, mientras que **done** indica que un nuevo dato digital está disponible. 37

3.13. Diagrama de tiempos del comportamiento del controlador del ADC diseñado. Al habilitarse mediante la señal **ena**, se activa por un ciclo de reloj la señal **wr_n** para iniciar el muestreo en el ADC. Cuando el controlador detecta que la señal **int_n** está activa, genera un pulso en **rd_n** y almacena el dato digital muestreado. Este proceso se repite para adquirir datos de forma periódica. El controlador indica que un nuevo dato está disponible en **adc_data** mediante un pulso en alto en la señal **done**. La cadencia a la que se obtiene un nuevo dato muestreado es de $19\,\mathrm{T_{CLK}}$. 37

3.14. Simulación de tiempos del comportamiento del lazo de control principal, donde se observan los retardos en las señales $\widetilde{sa}$ y **adc_in**, impuestos por los filtros utilizados. 39

3.15. Arquitectura R-2R utilizada por el DAC. Fuente: Hoja de datos del AD5445 [21]. 40

3.16. Configuración bipolar para el DAC. R1 es para ajuste de ganancia y R2 y C1 se usan en caso de usar amplificadores de alta velocidad, el cual no es el caso de aplicación implementado. Fuente: Hoja de datos del AD5445 [21]. 41

3.17. Diagrama de tiempos de señales de control del DAC AD5445. $\overline{CS}$ habilita el funcionamiento del DAC y $R/\overline{W}$ fija el funcionamiento en lectura o escritura. Fuente: Hoja de datos del AD5445 [21] . . . 42

3.18. Esquema de conexión del controlador del DAC, con sus respectivas señales. **ena** es la habilitación del controlador para su funcionamiento. **done** es un pulso de sincronización con el sistema digital, **cs_n** es la señal de control que habilita el funcionamiento del DAC. . . . 42
3.19. Filtro pasabajos implementado para la salida del DAC. 43
3.20. Simulación de arranque típico del controlador del DAC. 44
3.21. Bloque de procesamiento de datos digitales de la primer etapa. . . 45
3.22. Estructura interna del bloque *DSP*. 46
3.23. Arquitectura de la memoria implementada, compuesta por un banco de registros de FFs, con una profundidad de $DEPTH = 192$ y un ancho de palabra de $N = 12$ bits. 47
3.24. Algoritmo generador del template de la etapa 1. 48
3.25. Simulación del funcionamiento típico del bloque *converters_handler*. En este caso se activa el DAC con la llegada de un pulso de **trigger**, y se desactiva cuando la señal **ena_rd** baja. 48

4.1. Diagrama simplificado de *Etapa 2*. 50
4.2. Residuo de la operación realizada por la *Etapa 1* junto con la ECAP, luego del acondicionamiento analógico, referenciadas a 2,5 V. Se observa la ventana de interés a ser procesada. La figura es ilustrativa y no representa con exactitud la forma real de las señales. 50
4.3. Resultado de realizar el promedio de 16 ECAPs, debido a la ventana de ocurrencia variable. En gris se observa las 16 posibles ocurrencias de las ECAPs, mientras que en azul se observa el resultado de su promedio, *remanente_ecap*, destacándose la atenuación del promedio. 52
4.4. Se observa la ECAP extraída en naranja comparada con la ECAP original para un instante de tiempo arbitrario. La señal naranja es el resultado de aplicar la ecuación 4.5 al $residuo + ECAP$ de la figura 4.2. Se destaca el parentesco en morfología respecto a la original. La figura es ilustrativa y no representa con exactitud la forma real de las señales. 53
4.5. Porcentaje de la amplitud pico a pico de la ECAP versus la amplitud pico a pico del remanente de la ECAP ($100 \times \frac{remanente\_ecap_{pp}}{ECAP_{pp}}$ %) en función de las iteraciones ($ITER$). Se optó por $ITER$ =dieciséis. . 53
4.6. Amplificador no inversor con ganancia variable de la entrada de la *Etapa 2*. 55
4.7. Circuito implementado del filtro anti aliasing con respuesta Butterworth. 57
4.8. Circuito implementado del filtro anti aliasing con arquitectura RC. 57
4.9. Diagrama de estados del sistema digital de la *Etapa 2*. Se distinguen tres fases principales: inicialización del template vacío, cálculo del template promedio del residuo, y cancelación del residuo entrante con el template. El *Control Principal* tiene la potestad de llevar al sistema digital a la fase 1. 59

4.10. Diagrama de bloques del MSD. Las señales de entrada **s2_trg**, **s2_clear** y **s2_enable** son señales de control provenientes del *Control principal del sistema*. Las señales **ADC_data** y **ADC_done** provienen del *Control Principal*. La señal de salida **ecap** corresponde a la ECAP extraída del sistema, mientras que las señales de control **ecap_available** y **ecap_sample_done** indican cuándo la ECAP está disponible. Las señales de control de los submodulos se omitieron para simplicidad. 60

4.11. Comportamiento temporal transitorio del bloque MSD. Se observa que después de $T \times ITER$ segundos, se obtiene el *residuo* en la señal interna **template**. Además en este instante se observa que a la salida **ecap** se tiene la $ECAP\_extraida$ junto con sus señales de control. Para simplificar, se ha omitido la señal de reloj y se asume que el bloque está habilitado por el *Control Principal*. 61

4.12. Comportamiento temporal en régimen estacionario del bloque MSD. Se observa la señal **ecap(i)[n]** representando la muestra n-ésima de la i-ésima señal *ecap*, para $i > ITER$. La señal **ecap_available** indica la validez temporal de la ventana de la señal, activándose únicamente durante su duración. La señal **ecap_sample_done** genera pulsos de duración T_{CLK} cada 19 T_{CLK} para señalar la disponibilidad de nuevas muestras, sincronizando el procesamiento posterior. Para simplificar, se omite la señal de reloj y se asume que el bloque está habilitado por el *Control Principal*. 61

4.13. Arquitectura de la memoria implementada en la *Etapa 2*, compuesta por un banco de registros realizado FFs, con una profundidad de $DEPTH = 192$ y un ancho de palabra de $N = 16$ bits. 62

4.14. Interfaz del bloque *template*. 62

4.15. Interfaz del bloque template_handler. 63

4.16. Interfaz del bloque stop_average. 63

4.17. Interfaz del bloque subtractor 64

4.18. Respuesta en frecuencia del filtro pasa-bajos Blackman FIR, de orden 50 con frecuencia de corte de 7 kHz. 65

4.19. Interfaz del bloque digital_filter. 66

4.20. Comportamiento temporal del bloque *digital_filter*. Se ilustra el retardo de 25 muestras que introduce el *filter_50* a la señal **ecap[n]**. A su vez, se aprecia el comportamiento temporal de las señales de control **ecap_available** y **ecap_sample_done**). Se asume que el módulo está habilitado. 66

5.1. Diagrama simplificado del Control Principal. Se detallan los bloques principales que contiene y el intercambio simplificado de señales entre ellos. Los colores de los bloques indican sobre que sección del sistema actúa cada uno. 68

5.2. Sistema de manejo de datos del ADC. El mismo incluye un registro para guardar el nuevo dato entrante, y dos MUX para direccionar los datos a la etapa correspondiente. 69
5.3. Comparación de la señal de entrada SENSADO, en rosa, con el template generado dentro de la ventana temporal, en naranja. 70
5.4. Interfaz del bloque *window_handler*. Cuenta con la entrada **trigger** y las salidas **window** y **trigger_s1**. 71
5.5. Simulación del bloque *window_handler*, mostrando el comportamiento de sus señales. 72
5.6. Diagrama de la máquina de estados de control de etapas, con sus señales de control. Los círculos indican los estados, mientras que el rombo es una condición lógica, en donde dependiendo de la condición **stop_s1_ena** && **window_active**, la señal **s1_ena_wr** vale uno o cero. 72
5.7. Interfaz del bloque *stop_s1*. Cuenta con las señales de control **trigger_s1** y **enable** para sincronizar el funcionamiento del bloque y la señal **ADC_done** para recibir los datos del ADC, a través del bus **ADC_data**. La salida del bloque es la señal **enable_template**, la cual indica por nivel alto cuando el error del ciclo está por encima del umbral preestablecido. 74
5.8. Simulación del bloque *stop_s1*. Se muestra el funcionamiento en un ciclo arbitrario de la señal SENSADO. Se omitieron valores intermedios para lograr observar tanto el ciclo de arranque como la fase de apagado. 75
5.9. Interfaz del bloque *overflow_monitor*. El mismo cuenta con dos señales de control de entrada y un bus de datos. A su salida tiene la señal de control **overflow**. 76
5.10. Simulación del bloque *overflow_monitor*, en donde se observa un caso de overflow por el margen superior. 76
5.11. Simulación que muestra el arranque del template de la *Etapa 1* luego de su convergencia. Para que la *Etapa 2* comience a funcionar, es necesario esperar el transitorio de varios retardos. Primero que el template de la *Etapa 1* alcance el régimen (1), luego que el amplificador variable de entrada de la *Etapa 2* haga su transitorio de arranque (2), esperar el transitorio del filtro Butterworth (3), y finalmente que alcance el régimen, indicado con el *Marker 7*. 77
5.12. Interfaz del bloque *data_counter*. El mismo cuenta con tres señales de control de entrada. A su salida tiene la señal de control **s2_settled**, la cual indica que el contador alcanzó el valor objetivo. 78
5.13. Simulación del bloque *data_counter*, en donde se observa su funcionamiento típico. 78
5.14. Esquema del comportamiento de las señales de control de la segunda etapa. 79

6.1. Esquema con los bloques principales que componen el testbench utilizado. 82
6.2. Circuito de entrada y fuentes de alimentación del testbench utilizado. 83
6.3. Circuito de cambio de ganancia de la señal de entrada. 83
6.4. Circuito de la etapa de amplificación diferencial del testbench. . . . 84
6.5. Sección correspondiente al ADC y módulos relacionados. 85
6.6. Amplificador y filtro de salida de acondicionamiento para la *Etapa 2*. 85
6.7. Circuito del bloque digital principal del testbench. 86
6.8. Circuito del DAC y sus módulos dependientes. 86
6.9. Funcionamiento de la *Etapa 1* en la simulación top level. Se simuló con una relación de 53 dB entre la SA y la ECAP. La salida del DAC **loop_filt** se ilustra en celeste y la entrada **vin** en rojo. 87
6.10. Comienzo de generación de datos del sistema. Se simuló con una relación de 53 dB entre la SA y la ECAP. La salida del DAC **loop_filt** se ilustra en rojo y la entrada **vin** en celeste. 87
6.11. Caso en que la señal muestreada por la *Etapa 2* excursiona fuera del rango aceptable, en donde el sistema retoma el funcionamiento de la *Etapa 1*. Se simuló con una relación de 53 dB entre la SA y la ECAP. 88
6.12. Caso de retorno a la generación de ECAPs luego de que el sistema atravesara un overflow en la *Etapa 2*. Se simuló con una relación de 53 dB entre la SA y la ECAP. 88
6.13. Diagrama de tiempos del ADC MAX1266 en *internal acquisition mode*. Se observa el comportamiento esperado de las señales de control $\overline{CS}$, $\overline{WR}$, $\overline{RD}$ con la señal de interrupción $\overline{INT}$, junto con el bus triestado `D11-D0`. Fuente: Hoja de datos del MAX1266 [19]. . . 90
6.14. Interfaz del modelo del ADC MAX1266. 90
6.15. Módulos internos del modelo del ADC MAX1266. Está compuesto por los bloques *ADC T&H*, *selector* y *Main control* 90
6.16. Bloque de tracking and hold de datos, junto con el conversor a palabras digitales. 91
6.17. Simulación del bloque *T&H*. Cuando ena vale uno el bloque retiene el valor de entrada. Cuando ena vale cero el sigue a su entrada. . . 91
6.18. Tiempos asociados a los estados (*state*) de seguimiento (*acquisition*) y retención (*conversion*) de la señal analógica durante el muestreo: $4T_{CLK}$ para el *tracking* y $12T_{CLK}$ para la conversión. Fuente: hoja de datos del MAX1266 [19]. 92
6.19. Diagrama de tiempos del comportamiento del bloque *Main control*. 92
6.20. Interfaz del modelo del DAC AD5445. 93
6.21. Interfaz de un modelo de amplificador operacional implementado. . 94

7.1. Diagrama de bloques de los componentes de la PCB junto con la FPGA. 96
7.2. FPGA DE0 utilizada, con la identificación de sus componentes principales. Extraída de [23]. 100
7.3. Circuito implementado para las fuentes de entrada. **J5** es un jumper de tres pines. 100

7.4. Circuito implementado para el conversor de tensión. El elemento **J9** es un jumper de 3 pines, que actúa como llave de encendido del conversor. 101
7.5. Circuito implementado para el LDO de 5,14 V. 102
7.6. Circuito implementado para el LDO de 3,3 V. 102
7.7. Circuito implementado para las referencias de 2,5 V y 5 V. **J8** y **J10** son jumpers de 3 pines que habilitan el funcionamiento de cada referencia individualmente. 103
7.8. Circuito implementado del inversor conmutado junto con su filtro de salida. **J7** es un jumper de 3 pines que conecta la entrada del conversor con la alimentación o la mantiene apagada, fijando el nodo a tierra. 104
7.9. Circuito implementado del regulador lineal de −3,3 V. Este es alimentado a partir del charge pump de −5 V. 104
7.10. Jumpers de selección del origen de los diferentes niveles de alimentación. El pin central es la salida. Dada la composición de tres pines, se puede además utilizar el pin central como nodo de entrada para una fuente externa. 105
7.11. Circuito que implementa el atenuador full diferencial a la entrada. 106
7.12. Amplificador de instrumentación de entrada. 107
7.13. Circuito implementado del Schmitt-Trigger y el rectificador. 107
7.14. Circuito que implementa el amplificador diferencial. 108
7.15. Circuito implementado del Amplificador de ganancia variable. . . . 109
7.16. Imagen donde se detalla el valor de V_{COM}. El manual especifica que debe ser $V_{COM} \geq \frac{V_{REF}}{2}$, donde $V_{REF} = 5\,\text{V}$. Extraída de [19]. . . . 109
7.17. Conexión de el *Buffer 1* y *Buffer 2* representando el flujo de datos entre el ADC y la FPGA. 111
7.18. Comportamiento temporal impuesto por los buffers entre el ADC y la FPGA. 111
7.19. Circuito esquemático implementado en la PCB para la conexión de los level shifters entre el ADC y la FPGA. 112
7.20. Circuito esquemático implementado en la PCB que incluye el DAC y su circuito de acondicionamiento de la señal de salida DAC_OUT, que transforma la salida del DAC en corriente (IOUT1, IOUT2), en la tensión de salida DAC_OUT. 113
7.21. Vista superior de la PCB diseñada por CANICs con sus componentes. Se resalta en rojo la división de componentes por su tipo de funcionamiento. La inferior corresponde a la sección de conversores de tensión conmutados y fuentes de alimentación, la central a señales analógicas y la superior a señales digitales. 115
7.22. Plano de tierra de la PCB, mostrando las diferentes zonas creadas con el objetivo de evitar la interferencia entre ellas. 116
7.23. Plano de alimentación de la PCB. Se separó el plano de 5 V siguiendo la división por familia de componentes, a modo de evitar la interferencia y reducir el ruido. 116

7.24. Montaje entre la PCB diseñada y la placa DE0 utilizada. 117

8.1. Diagrama del entorno de validación del sistema. Se distingue el módulo WaveGen, encargado de generar las señales biológicas; el sistema CANICs, representado por la interconexión entre la PCB diseñada y la placa DE0; y el módulo Métricas, responsable de calcular medidas objetivas sobre el desempeño del sistema. 120

8.2. Montaje del setup de validación del sistema. Se muestra la PCB diseñada conectada a la placa DE0, con el generador de señales acoplado a la entrada. El osciloscopio, visible en segundo plano, permite monitorear las señales durante las pruebas. 121

8.3. Captura de las señales **ecap**, **ecap_done** y **ecap_sample_done**. La adquisición se realiza en cada flanco de subida de **ecap_sample_done**, con un período de $19\,T_{CLK}$. 122

8.4. Gráfica con la amplitud pico a pico promedio obtenida de la ECAP normalizada para distintos casos de relación SA/ECAP. En función del valor N utilizado para realizar el promedio. Se definió el margen de 20 % para cuantificar la cantidad de promedios a realizar. Se optó por utilizar 6 trazas de ECAPs obtenidas por el sistema para el promedio. 125

8.5. Señales observadas en distintos puntos del sistema. En (a) salida del amplificador de instrumentación, utilizada por la *Etapa 1* para generar el template de cancelación. En (b) resultado de la cancelación del artefacto SA, donde se observa el residuo junto con la ECAP. En (c) señal amplificada y filtrada por la *Etapa 2*. En (d) muestras obtenidas a la salida del sistema CANICs, donde se visualiza la ECAP extraída en dos ventanas temporales. 126

8.6. Salida del sistema sin imponer ECAP a la entrada, comparada con la salida del sistema en la que se impone la ECAP. Para este caso, la relación de señales entre la ECAP y la SA es de $450\,\frac{V}{V} = 53\,dB$. . 128

8.7. Muestras extraídas del sistema en el caso $SA = 70\,mV$ y $ECAP = 150\,\mu V$, donde su relación es de SA/ECAP = $450\,\frac{V}{V}$. En rojo, la señal extraída por el sistema comparada con la ECAP impuesta en la entrada en azul. Abajo se muestra una gráfica de la correlación entre ambas señales. 129

8.8. Señal extraída por el sistema CANICs comparada con la ECAP impuesta en la entrada para el caso de una relación entre las señales de $3200\,\frac{V}{V}$. Abajo se muestra una gráfica de la correlación entre ambas señales. 130

8.9. Señal extraída por el sistema comparada con la ECAP impuesta en la entrada para el caso de una relación entre las señales de $4800\,\frac{V}{V}$. Abajo se muestra una gráfica de la correlación entre ambas señales. 131

8.10. Amplitud obtenida por el bloque *Métricas*, comparada con la ECAP a la entrada normalizada en función de la relación SA/ECAP. En rojo se presentan 148 datos de la amplitud pico a pico de la ECAP obtenida normalizada. En verde se ilustra la linea con el valor objetivo de la amplitud pico a pico. 131
8.11. Respuesta del amplificador de instrumentación frente a componentes en modo común de amplitud $2\,\mathrm{mV_{pp}}$ y frecuencia $400\,\mathrm{Hz}$. 133
8.12. Respuesta del amplificador de instrumentación frente a señales diferenciales en continua de $\pm 100\,\mathrm{mV}$. 134

A.1. Respuesta del filtro de cuarto orden con respuesta Butterworth diseñado. 140

B.1. Diagrama de tiempos del controlador diseñado para los buffers. Se aprecian las señales de habilitación por nivel bajo de los buffers **oe_n1** y **oe_n2**. 144
B.2. Simulación del bloque *mem_handler*. Las señales **cs_n** y **th_ena** no son de este bloque, sino que **cs_n** es proveniente del DAC, y genera un pulso a cero cuando escribe un nuevo dato. Análogamente, **th_ena** es una señal proveniente del ADC, y produce un pulso a cero de duración $3\,\mathrm{T_{CLK}}$ cuando el ADC está muestreando datos. . 145

Esta es la última página.
Compilado el viernes 18 julio, 2025.
http://iie.fing.edu.uy/